Reactive Trajectory Planning in Structured Dynamic Urban Scenarios with Static and Dynamic Obstacles

Pradeep Korivi

From the Faculty IV - Electrical Engineering and Computer Science

Technische Universität Berlin

Department of Telecommunication Systems -Communication and Operating Systems

Master of Science

Guides : Prof. Dr.-Ing. Reinhardt Karnapke
Prof. Dr. habil. Odej Kao

Abstract

The goal of any motion planner is to achieve a driving trajectory that is collision free, smooth and responsive (reactive), at the same time being far-sighted to maintain consistency, deliberative and allow smooth transitions. Prior approaches solved this problem through different methods having various complexity levels. This thesis proposes an on road motion planner with combination of path-velocity combination along with a low computational overhead collision avoidance system. The reactive nature tunability of the proposed thesis is suitable for urban driving scenarios.

Abstrakt

To be translated

Eidesstattliche Erklärung
Ich bestätige, dass diese Masterarbeit meine eigene Arbeit ist und ich alle verwendeten Quellen und Materialien dokumentiert habe. Diese Arbeit wurde zuvor keinem anderen Prüfungsausschuss vorgelegt und ist nicht veröffentlicht worden.

Statutory Declaration
I confirm that this Master's thesis is my own work and I have documented all sources and material used. This thesis was not previously presented to another examination board and has not been published.

....
Pradeep Korivi
Berlin, xxxx yyyyyyyyyy 2018

Contents

CHAPTER 1

Introduction

1.1 Motivation

1.2 Problem Statement

1.3 Thesis Statement

1.4 Thesis Contributions

1.5 Thesis Structure

Chapter 2

Related Work

2.1 Planning Approaches

2.1.1 On Road Motion Planning

Look if the sampling based, RRT based, lattice based etc etc should be combined together?

2.1.2 Off Road Motion Planning

2.2 Collision Checking

2.3 Evaluation Criteria

Chapter 3

Vehicle Setup

3.1 Vehicle Base

3.2 Sensors

3.3 Computational Power and Software Architecture

3.4 Vehicle Control

3.5 Localisation

3.6 Introduction

Mention that ego vehicle, model car, robot are used interchangeably in the document

The goal of the path planner is to navigate the robot from the start configuration to destination by blending in the traffic. Path planning module is dependent on various modules to receive the data regarding the perceived environment and invoke a set of modules to move the robot safely. It has to drive the robot ahead considering the traffic rules, obstacles, kinematic and dynamic constraints of the robot and not compromising on the safety and comfort of the passengers inside. The main aim of this chapter is to derive path planning techniques to drive the robot safely to destination.

This section is organised to provide an overview of different modules needed for path planning and methods used in each module to achieve the goal. Path planning starts from the initial understanding of where the ego vehicle is and where to go, subsection 3.7 describes regarding localization of ego vehicle to provide this data, subsection 3.9 provides the information on how a global path to the destination is calculated. An autonomous vehicle should have the understanding of surroundings in terms of where other vehicles are, where pedestrians are, traffic signal information etc., Prediction module described in 3.8 details further on how the ego vehicle perceives environment. The next subsection 3.10 describes further in-depth details about the short term planning algorithm or the trajectory planner. The final module in the discussion is control unit, described in subsection 3.11. It is responsible for translating the path in space-time into steering and acceleration values to drive the robot.

Figure 3.1 represents the general architecture of the planning module. It details on the flow of information, dependencies, relative execution frequency. All the modules are implemented as independent nodes in Robot Operating System(ROS) and communicate with each other using ROS messages.

Add reference as udacity course material for path planning overview picture, add different color to highlight the main modules I am working on

3.7 Localization

Localization module is responsible for providing the current state of the vehicle in terms of position, orientation, speed(linear and angular) and acceleration. The localization module implemented on the modelcar has two sub components Vehicle Odometry and Global Position estimation using Visual GPS. Odometry is calculated with dead reckoning [3] with speed information from motor and yaw information from Inertial Measurement Unit(IMU). The localization module combines odometry with the information received from a visual GPS node(tracks markers on roof) to correctly estimate the state of

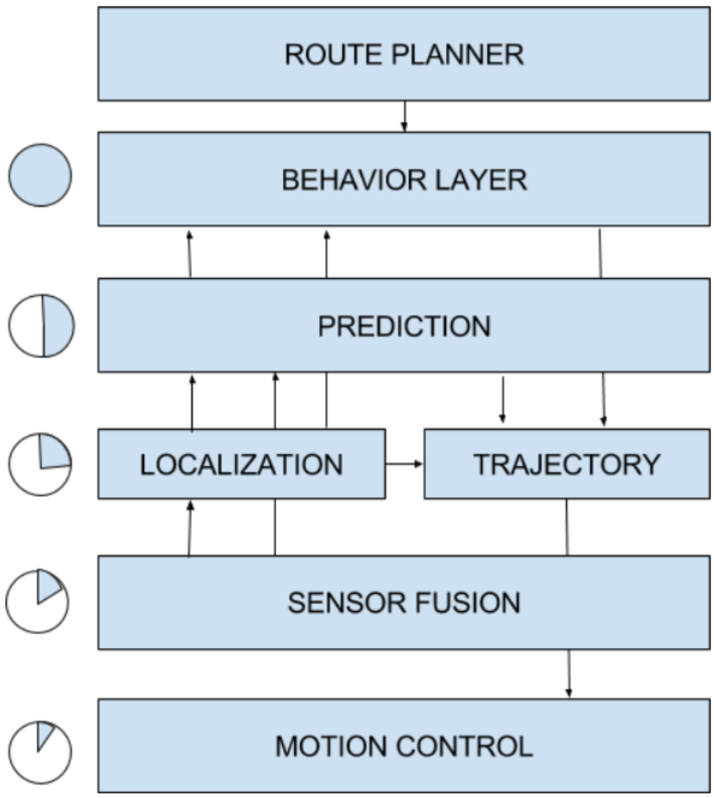

Figure 3.1: Path Planning Module

the ego vehicle.

3.8 Prediction

Prediction and Sensor fusion modules receive the data from various sensors such as Cameras, LIDAR etc and fuse them together to create an environment model, classify objects into different categories and predict the state of the obstacles in the surroundings. Due to time constraints this thesis simulates a prediction module to provide motion planner with obstacle information in different traffic scenarios.

3.9 Route Planner

Route Planner is responsible for finding a global route between the vehicle current state and the goal state based on the static characteristics of the

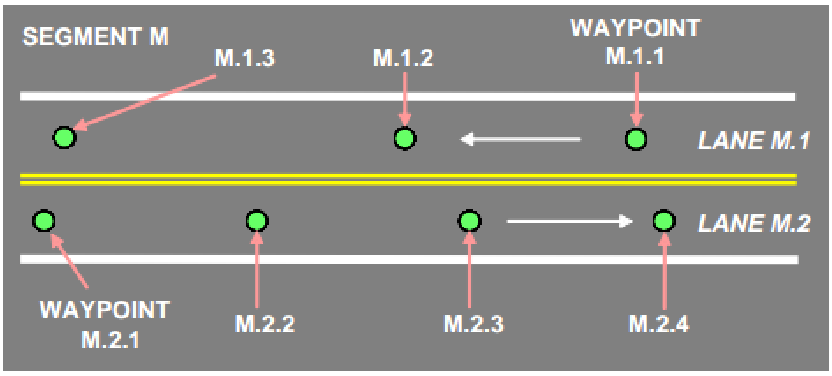

Figure 3.2: Segment representation in RNDF - Segement M has two lanes M1, M2 and each lane has way points 1-N

environment/map such as lane information, speed limits etc. Route planner obtains this information generally from the maps or other formats to represent the road network. In this thesis a simple model called " Road Navigation Definition File(RNDF)" [5] [4] is used to represent the route network. The next subsections details further about RNDF and how global reference route is calculated.

3.9.1 RNDF

This chapter details about the RNDF file [5] that defines the road network(set of roads/ areas connected together) over which the vehicle can traverse. This form of representation of road is developed by DARPA for its Autonomous Vehicles Urban Grand Challenge. RNDF representation first divides the traversable areas into two parts, segments and free zones. Free zones represent areas such as parking lots etc connected by road segments. Each segment has multiple lanes, each lane has way points along the driving direction. More significant information about way points such as whether it is a stop sign etc can be added. Each segment/zone is connected to one another using exits. Exits represent the connections between one segment way points at start/end to another. Figures 3.2 3.3 3.4 [5] represent various portions of the route representation and Figure 3.5 details regarding the route network of map used in Lab.

foot note - source of images

Add Appendix for how to create the RNDF file for model car

3.9.2 Path Representation and calculation

The data in the RNDF is represented in the form of a tree with connections across segments, lanes, way-points. The global path from source to destination

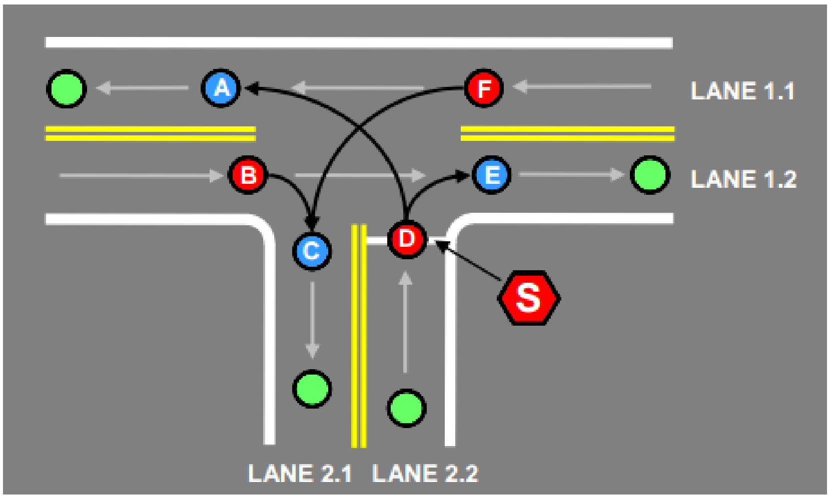

Figure 3.3: Exit representation in RNDF - Connections between two segments in a T-Junction

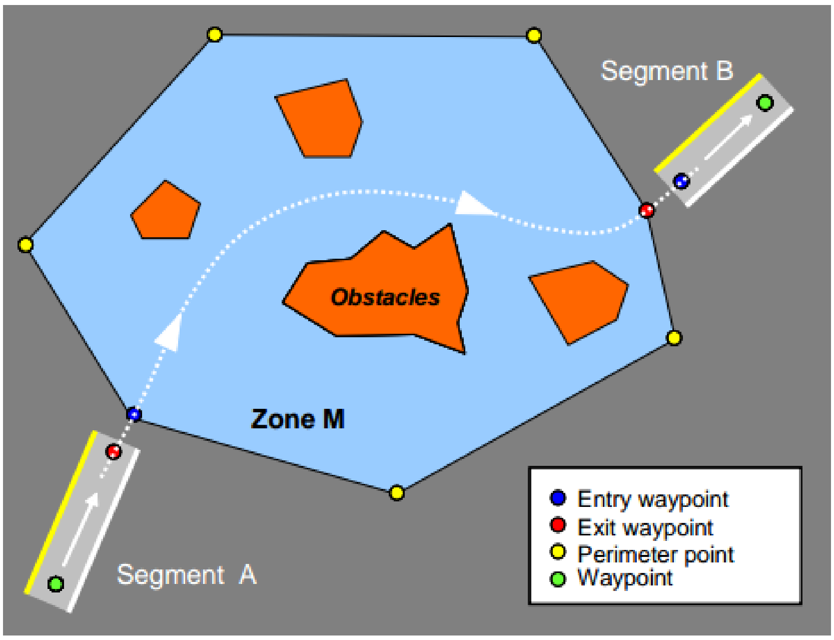

Figure 3.4: Zone Representation in RNDF - Connection between Segments and Zone

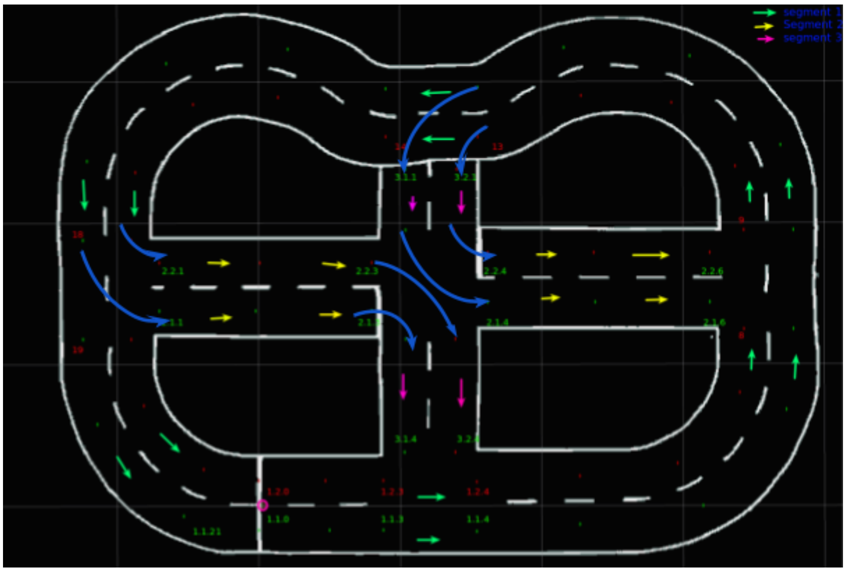

Figure 3.5: Road network for the map used to test model Car

is the shortest path between the closest way-point to ego vehicles current position the and closest way-point to the destination in the graph. Then the shortest path found is sub divided into sub-paths based on which segment the way points lie. A sub-path represents a set of way points in one segment, once the ego vehicle is at the end of one sub-path it receives a notification from the below trajectory planner that a goal has been reached, then the route planner transmits the next sub path to the trajectory planner, this process is repeated till destination is reached. Figure 3.6 details further about the division of shortest path across different segments. This method also reduces the memory needed in modelling the road in trajectory planning stage.

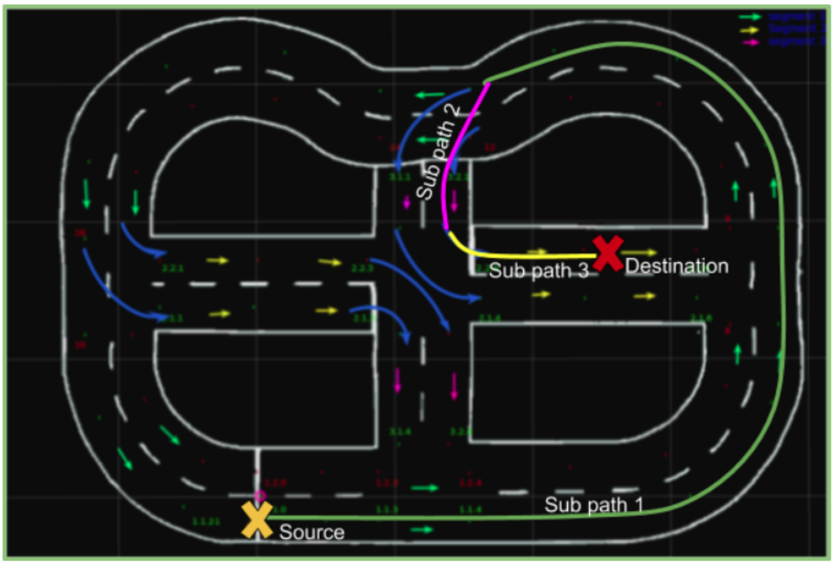

Figure 3.6: Division of shortest path in road network into Sub paths across different segments

3.10 Motion Planner

In this section we will discuss the planning algorithm to create short term trajectories create in accordance with the global path to reach destination. The subsection 3.10.1 provides the overview of timing Horizon and timing constraints in dynamic environments for different modules in planning. The next subsection 3.10.2 details regarding path modelling and how this will improve the efficiency of planning along with constraints it introduces. It also discusses on approximation method used to convert coordinates from Cartesian frame to Frenet frame. The core of this chapter is creation of trajectories which is discussed in detail in subsection 3.10.3. The next sections 3.10.4 and 3.10.5 detail further on how the trajectories created are evaluated for collision with static and dynamic obstacles. The next subsection 3.10.6 further details on how a final trajectory is selected from the set of evaluated trajectories.

3.10.1 Temporal Horizon

Time is an important aspect for motion planning in dynamic environments and there are several timing variables associated with planning. This section is mainly adopted from the doctoral thesis [13]. These timing parameters define how far into the future different sub modules of planning will be valid.

check if this name is needed and add footnote to link "Autonomous vehicle navigation in dynamic urban

The initial timing variable in motion planning is timing horizon T_m. It is measure of how far into the future the trajectory of the vehicle is planned. Second is the prediction horizon T_p, it is measure of how far into the future the motion of dynamic obstacles around can be predicted. The fundamental requirement of planning to be valid is that $T_m \leq T_p$ such that planning is done only so far into the future as the environment is predictable.

Thirdly T_d indicates the computation time of the motion plan. Assuming planning is done in cycles, the plan created in previous cycle is executed in current cycle, thus $T_d \leq T_m$. If this condition fails then the planner will run out of path for the next cycle. In general $T_d \ll T_m$. T_s is the perception update cycle time, i.e., perception module updates the trajectories of the surrounding dynamic obstacles every T_s seconds. In general world the predicted trajectories for duration T_p will not hold true as the behaviour of these vehicles is not controlled by the ego vehicle. Thus the constraint $T_s \leq T_p$ should be valid. This creates an uncertainty in modelling of the environment, thus the execution duration of current plan T_e beyond T_s is not sensible. This is due to fact that obstacle trajectories may have changed in T_s and executing the old trajectory may lead to collisions invalidating the trajectory created for T_m.

check if it is needed to write on time discretization

The next timing constraint in consideration is T_e, control execution time of the current plan. T_e should not exceed the perception update time T_s. This restriction also imposes additional constraint on T_d (motion plan computation time), $T_d \leq T_e$.

In summary, timing constraints described above identify the relation between different modules such as motion planning, motion prediction and motion execution. It is also important to predict farther into future than T_s or T_e for completeness of motion planner with respect to goal objective, uncertainty also increases with time. In general a farsighted uncertain motion plan potentially directing the vehicle towards goal is better, but this plan needs to be re-evaluated and re-executed in short intervals for correctness.

time to see minimum distance ahead with current speed, minimum time to bring car to halt at good speed etc etc.

add why 5s is chosen, why other execution parameters are chosen

3.10.2 Path Modelling

Planned global path is in Cartesian coordinate system, one of the problems with the Cartesian coordinate system is that the due to variation in curvature local planning becomes complex. To address this issue planning in curviliniear system or Frenet Frame as been adopted by researchers. [19] [20] [17] [11] [2] [9] are some of the research works in which Frenet frame or lane adoptive (SL) coordinate system is adopted. In this method a centre line passing across the centre of the lane/road or pre-planned path in un-structured environment is used as a reference path for longitudinal coordinate(S) and the perpendicular distance with respect to this line is used as lateral coordinate(l) as represented in Figure 3.7 [17]. Thus once converted S,L coordinate system acts like a straight road.

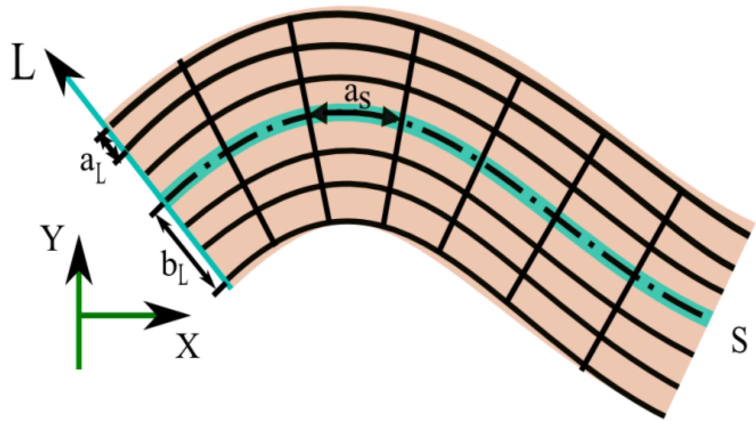

Figure 3.7: SL coordinate system laid over XY coordinate system

add diss shui reference in foot note

Conversion from Frenet frame to Cartesian and vice versa is a widely researched topic and many techniques exist to offer a different level of complexity and accuracy. In this thesis an approximation method is used to convert between these two coordinate systems similar to [2] as it is computationally cheap and provides required level of accuracy. To convert an x,y coordinate to SL coordinate, (x,y) is projected onto the current path represented by different way points as in figure 3.8, cumulative distance till this point gives the S coordinate and the perpendicular distance between the projected point and the current point provides the L coordinate. A similar process is used to convert S,L coordinate to x,y coordinate. S is used to find a point on a segment represented by way points, a point at a perpendicular distance L gives the x,y coordinate. We assume that the path between two way points is linear which reduces the computational complexity in approximation. This approximation how ever approaches zero error when the spacing between two way points approaches zero. Adding dense way points in the curves significantly reduces the approximation error. There are different methods discussed in [16] [8] which provide better accuracy in calculating the paths.

In summary, curvilinear coordinate system makes planning easier but needs extra computation in conversion from one format to other. It also introduces errors and inefficiencies in planning if the complete planning is done in SL coordinate system. The research mentioned above tackes these problems differently. Further details and approximations, errors of using curilinear coordinate system will be further discussed in 3.10.3 section.

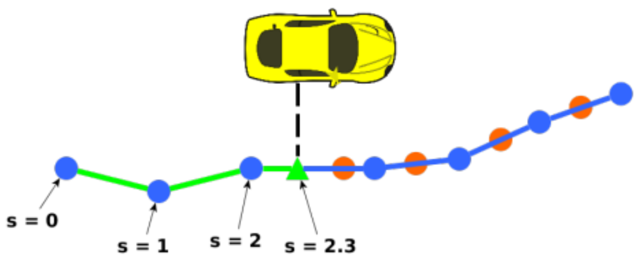

Figure 3.8: Showing projection of the current position of the car shown by green triangle to s coordinate system. Each circle represents a node, and the lines between them are links.

add VOLVO reference reference in foot note and also remove the orange dots

write more about this process from this paper

Provide the papers using frenet frame - stanford, volvo/honda , dual carriage way, Diss Shui Thesis, and others

Details on what is frenet frame how to convert from Cartesian to frenet frame - basic assumptions, lines across way points to make the frame.

3.10.3 Trajectory Creation

The core of the this thesis is the trajectory planner that drives the robot from source to destination. Understanding from the behaviour of the human drivers in structured environments(road networks) it is necessary for the trajectory planner to create trajectories that align with the road network. There are different ways to create these road aligned trajectories as discussed in the survey [7].

The approach of this thesis to create trajectories is inspired from [1] which proposes a trajectory planning technique combining path and velocity planning. This approach works on principle that from current state that a vehicle can accelerate to desired speed, decelerate to desired speed or zero speed and stay at constant velocity for planning duration. This leads to different velocities attained, different lengths traversed in the planning horizon T_m as discussed in section ??. Timing horizon is chosen to be 5s to allow long enough planning to attain goal and short enough to have valid boundary conditions for dynamic obstacles.

The trajectory planning starts with initial approach of sampling different accelerations the vehicle can follow. Figure 3.9 indicates how a vehicle which approached acceleration A3 at time t_0 has options to follow different acceleration profiles from A1 to A8. Each of this profile leads to a different final velocities as described in figure 3.10 and different lengths traversed as in figure .

create a distances travelled

How the accelerations profiles are attained defines the jerk which is an important factor in driving comfort. A direct change in acceleration from one profile to another induces a very high jerk similar to bumper cars. Trapezoidal acceleration profile as shown in 3.9 will have constant acceleration and further jerk free trajectories can be achieved by connecting various profiles using higher order curves as in [17] which uses second order equations in velocity planning. The selection of smoothness should also depend on the tracking capabilities and computational capabilities of the vehicle.

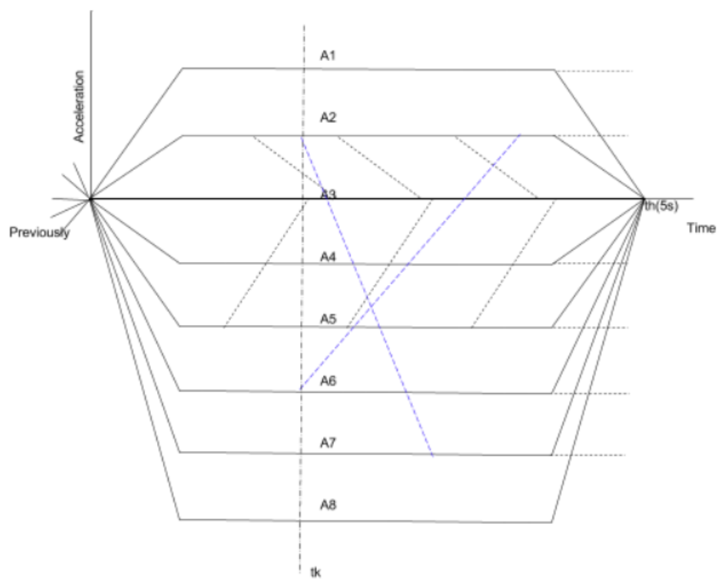

Figure 3.9: Different acceleration profiles a car can follow from current state.

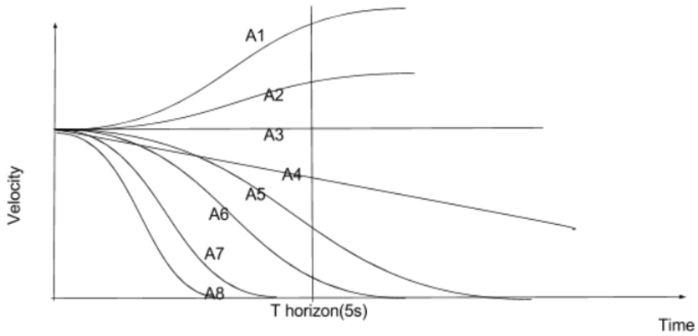

Figure 3.10: Different acceleration profiles a car can follow from current state.

Redraw these profiles neatly with proper labels - acc and velocity

The above acceleration profiles solve the problem of longitudinal planning but a vehicle on road should also plan a lateral shift to avoid obstacles, this is solved by sampling various lateral points and achieving the shift in lateral distance over the planned horizon. There are two ways to map lateral shift, either as a function of distance travelled or time and as [18] suggests that at lower speeds it is beneficial to plan lateral shift as a function of distance travelled and at high speeds as a function of time. This thesis is aimed at urban driving scenarios where final speeds are assumed to be low, thus it lateral shift l_f is planned as a function of distance. To attain a smooth shift from current lateral coordinate to sampled coordinate, polynomial splines act a great functions. Lateral shift planning in this thesis is adopted from [11], which uses cubic splines and models lateral shift as a parameter of longitudinal distance as shown in equation 3.1.

$$l(s) = c_0 + c_1 s + c_2 s^2 + c_3 s^3 \tag{3.1}$$

The first and second derivative of the equation 3.1 are equations for lateral velocity 3.2 and acceleration 3.3.

$$\frac{dl}{ds} = c_1 + 2c_2 s + 3c_3 s^2 \tag{3.2}$$

$$\frac{d^2 l}{d^2 s} = 2c_2 + 6c_3 s. \tag{3.3}$$

Form the boundary conditions, we have

$$l(s_0) = l_0, l(s_f) = l_f \tag{3.4}$$

The angle between the road frame and the vehicle is defines as $\theta(s)$, it can be derived from the first derivative of the lateral shift with respect to s.

$$\theta(s) = arctan(\frac{dl}{ds}) \tag{3.5}$$

To ensure the generated path follows current curvature and orientation of car and the final orientation is parallel to the road segment. Following conditions should be met.

$$\theta(s_0) = \theta_0, \theta(s_f) = 0 \tag{3.6}$$

The figure 3.11 indicates how the initial orientation will affect the shape of the trajectory.

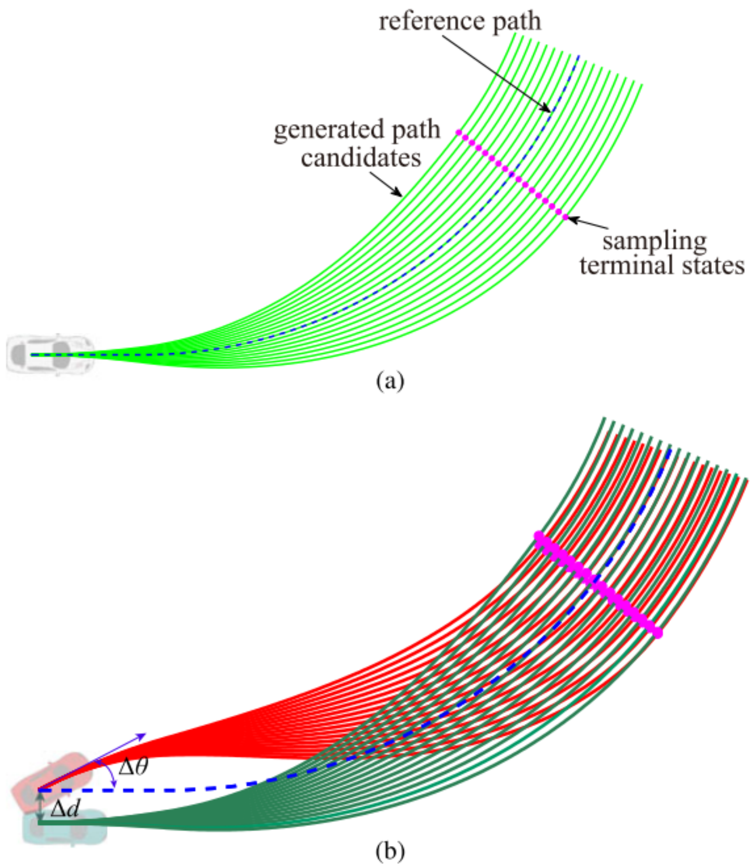

Figure 3.11: Path candidates generation results. (a) $l_0 = 0$ and $\theta_0 = 0$ (b) $l_0 = \Delta_d$ and $\theta_0 = \Delta\theta$.

create own figure

The constants c_0, c_1, c_2, c_3 in equation **??** can be obtained by solving the equations 3.2 to 3.6.

This creates multiple trajectories in (s,d) with multiple accelerations reach various velocities, distances and lateral shifts.

add a picture of how this will look in the end - various trajectories in planning

Further details on how these trajectories are converted to x,y coordinates used by the trajectory follower and the constraints in implementation for model car will be discussed in 4

3.10.4 Checking for Static Obstacles

The main objective of the motion planner is to derive a path avoiding the obstacles. The trajectories must be evaluated for collision or driving close to the static obstacles. There are various techniques for collision detection as discussed in the background study. This thesis employs a simple two step process to check for collision. A road parallel model in Frenet frame is used in checking for collision ignoring the orientation of the car as the this can be mitigated over the length of the path as described in [14].

The coordinates of the obstacle are transformed into Frenet frame initially, this thesis assumes that the obstacles are represented by rectangle surrounding the complete obstacle. This is a conservative approach but inline with general things found on road (vehicles, construction material, signs, people, pets, trees etc). The static objects are dilated as per best approximation to find the bounding box.

In initial step is to check if the path being checked has an intersection in s coordinate for the distance planned and the length of the static obstacle. As shown in figure 3.12 trajectories T0,T1 have intersection in S for obstacle O1 and no intersection for obstacle O2. The next step is to find the intersection region I1 and I2(extra buffer including the length of the car) where the length of the obstacle collides with the length of the trajectory. The next step is to compute if there is a collision in d(lateral dimension) for these trajectories. This is computed by checking if at any point between I1 and I2, the distance between the lateral coordinate of the car and the obstacle is less than safety. It is unsafe if

$$|d_{\text{ego}} - d_{\text{obst}}| < car\_width/2 + obstacle\_width/2 + safety\_margin \quad (3.7)$$

represent the parameters in terms of d_e, c_w, o_w etc

It is clearly representative from figure 3.12 that the trajectory T1 has collision and trajectory T0 has no collision. Different costs can be added based on how close the car and the obstacle are. The fact that the lateral shift proceeds in only one direction as per the planner it is sufficient to check for the collision at the start, end and middle of the intersected path.

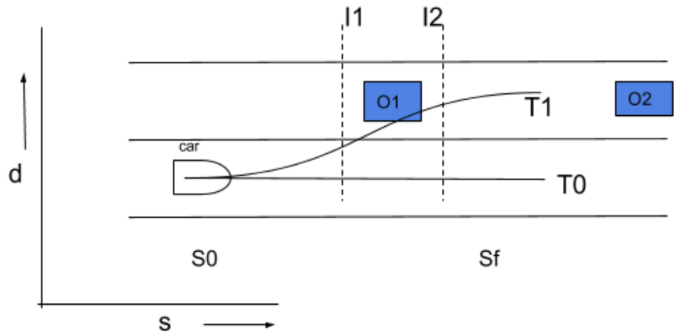

Figure 3.12: Collision check for static obstacles

3.10.5 Checking for Dynamic Obstacles

In dynamic environments collision check with dynamic obstacles is a key for path planning. Dynamic obstacles are modelled as squares moving across/along the road. It is assumed that the dynamic obstacles tend to continue in the same lane of their current detection for the rest of the planning duration. This is a general assumption in many planners like [1], it can be justified by the fact that the trajectories are re-evaluated at high frequencies and any changes in obstacles lateral distance will be evaluated in next cycle thus keeping the vehicle safe from collision.

The collision check for dynamic obstacles is a 3 step process where the first two steps are equivalent to the static obstacle collision check. In step one the intersection in S coordinate for the obstacle and the ego vehicle is found, here the length of the obstacle is dilated over the distance travelled by obstacle as represented by dotted line ahead of obstacle in figure 3.13. Then the collision for the ego vehicle and obstacle in the intersection region I1 to I2 is tested in similar way to static obstacle collision check.

If the collision in s dimension exists then it is checked if there is collision in d dimensions using boundary conditions similar to static obstacles. If the collision in d dimension exists then the s dimension where there is collision in lateral dimension is found, represented with J1-J2 in figure 3.13 (generally this will be shorter than I1 - I2). For the range J1-J2 it is checked if they collide in time also i.e., if they reach the same same location in same time there is a collision, a buffer of few seconds is added to be safe. As per instructions for safe driving it is required for the car to maintain a minimum time gap 2s with the vehicle ahead. There are more formal methods [15] on safety distances for self driving cars. This thesis implements simple 2s rule to safety. Sub Figure d of 3.14 indicates the collision in time scenario in figure 3.13.

Time gap between the ego vehicle and the obstacle at the S intersection borders is checked as shown in figure 3.14. If the gap is greater than 2 seconds and doesn't change sign then there is no collision, figure 3.14 a) shows a similar situation where the obstacles get close but does not collide. Extra costs are added if the ego vehicle gets too close to obstacle. A collision occurs when the sign of the time gap between the ego vehicle and the obstacle changes as shown in sub figure b and c of 3.14. Figure e of 3.14 shows the collision when the obstacle is moving in opposite direction.

If a dynamic obstacle is found moving laterally across the road then the trajectory is considered to be in collision if there is a collision in s and no collision in d,t are checked. This could be further improved anyways.

write why simulating over time and checking for collision is not a good idea - how computational complexity increases

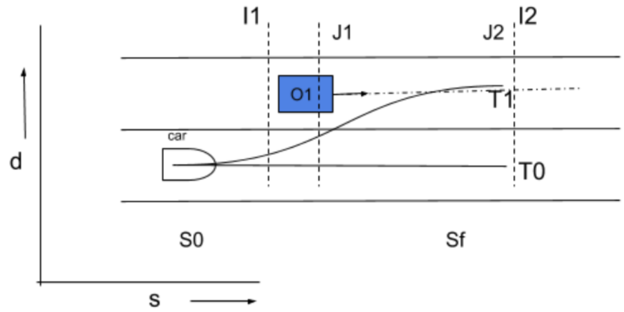

Figure 3.13: Collision check for Dynamic obstacles

Remove these pictures and add further pictures with evaluation for obstacles in same lane and opposite lane.

3.10.6 Cost Functions and Trajectory Selection

add reference to Daniel Thesis for forbidden regions etc

Initial cost for trajectory

$$cost = |V_a - V_t| + |a_t| + |(d_t - d_e) * k1| + |(d_p - d_e) * k2| \tag{3.8}$$

V_a - velocity achieved by trajectory. V_t - Target Velocity. a_t - Target Acceleration. d_t - Target lateral distance. d_e - Trajectory lateral distance. d_p - Previous target lateral for trajectory. k_1, k_2 - Factors to adjust weights, currently used at 0.8 and 0.2

Initially all the sampled trajectories are assigned the costs based on the cost function 3.8 and sorted, then the trajectory with the lowest cost is evaluated

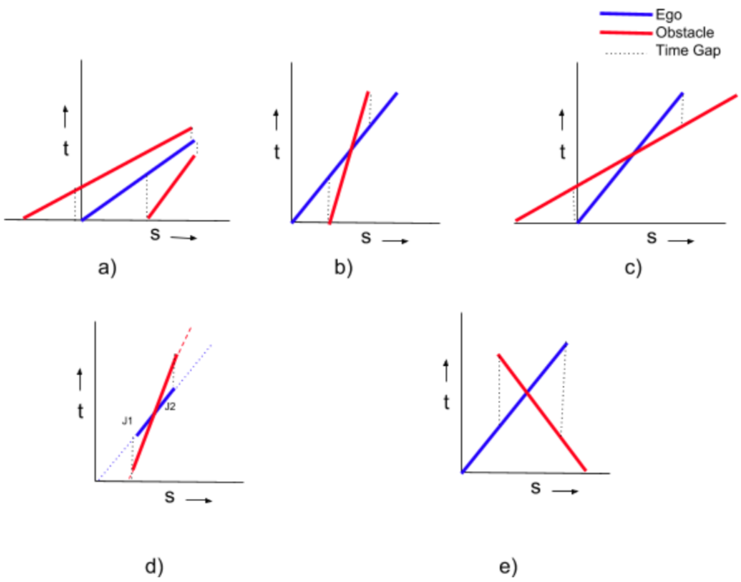

Figure 3.14: Collision check in space Time a) indicates obstacles getting close to the ego vehicle but not colliding. b) Ego vehicle hits the slow moving obstacle ahead. c) Ego vehicle is hit by fast moving obstacle behind - Situation during unchecked lane change by ego vehicle

first. The list is sorted and evaluated again till the top of the list has the lowest cost and is evaluated.

Different costs for comfort of the passengers and to improve the life of the vehicle can be added, [19] details a list of static and dynamic costs that contribute to a comfortable journey. In this thesis only a simple cost function is used as the target platform is a model car with limitations in fineness of control and measurement drive the vehicle.

3.10.7 Velocity Planning

Velocity planning is an important aspect of the planer, in general behavioural layer defines the target velocity based on speed regulation, other traffic participants, required behaviour, road condition etc. In this thesis a simple approach of velocity limiting is used based on road curvature to limit lateral accelerations. Max velocity $\mathrm{V}_{\mathrm{limit}}$ is calculated based on the equation 3.9.

$$V_{\text{limit}} = \sqrt{Acc_{\text{MaxLat}}/|k(s)|} \tag{3.9}$$

3.11 Trajectory Follower

3.12 Implementation

speak about splines, how the state machine works, write a flowchart etc
how the conversion happens and where it happens etc etc

Chapter 4

Implementation

Check if this chapter is needed,

Here I can write mostly about how the state machine is implemented to choose profiles

How the collision check is implemented

How costs are added

How the messages flow,

Which messages are received, which messages to be

I would prefer to finish this in Planning Chapter

CHAPTER 5

Evaluation

In previous chapters detailed working of the planning algorithm has been discussed, this chapter discusses the evaluation criteria, results of evaluation in detailed. The various concepts discussed previously will be examined here through a series of experiments reflecting real life driving scenarios In this chapter first a systematic evaluation of the planner is performed by exposing planner to various scenarios equivalent to on-road driving conditions. The next subsection performs a criteria based evaluation for the planner similar to any algorithm in the form of feasibility, optimality, completeness and run-time. The final sub-section compares this approach with other proposed algorithms in various criteria.

5.1 Experiments

In this section different experiments performed on the model car and the simulator are described. Most of the experiments involving dynamic obstacles are performed in simulator due to time and effort involved in creating scenarios in real world. Most of the test cases involve finding a collision free path with obstruction in current lane, slow moving traffic, merging into ongoing traffic, lane changes etc. The following subsections detail further on each experiment.

5.1.1 Lane blocked

In this driving scenario 5.1, the driving lane is blocked by a static obstacle and a slow moving obstacle in the next lane, robots drives slowly till it finds enough room in the next lane, once obstacle is avoided the robot continues to shift to intended lane and drives with increasing speed.

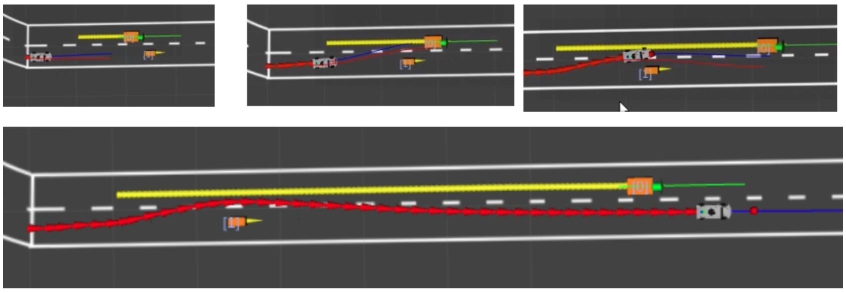

Figure 5.1: Driving Lane Blocked

5.1.2 Slow Moving Traffic

In scenario 1 presented in Figure 5.2 the ego vehicle starts changing into left lane once a slow moving obstacle is encountered, once the obstacle is passed the vehicle starts driving into right lane again. In scenario 2 presented in Figure 5.3 same behaviour is observed but as we can see as the vehicle moves forward it encounters a slow moving obstacle again and starts a lane change. This behaviour is caused because of cost functions drive the vehicle into intended lane without knowledge of global information. A behavioural layer with longer scenario analysis horizon will result in better path selection.

Figure 5.2: Slow Moving Traffic Situation 1

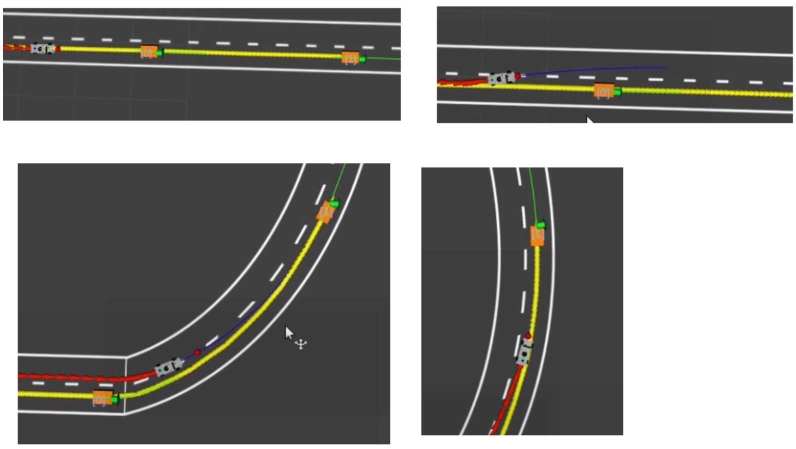

Figure 5.3: Slow Moving Traffic Situation 2

5.1.3 Merging into traffic

In scenario presented in Figure 5.4, lane changing is requested to merge into the traffic in left lane. Here ego vehicle speed is $1ms^{(} - 1)$ and obstacle speed is $0.6m^{(} - 1)$. Initially lane change does not occur as cost functions tuned to maintain speed over maintaining required lane. As the vehicle enters the curve, target driving speed is reduced and the vehicle merges into the traffic in left lane. Depending on which portion of the lane the ego vehicle is in i.e, near intersections or exits target lane should have higher priority over maintaining speed and during rest of the regions target speed should be of higher priority to reach destination quickly. Cost functions implemented in this thesis provide flexibility in tuning the behaviour of the ego vehicle.

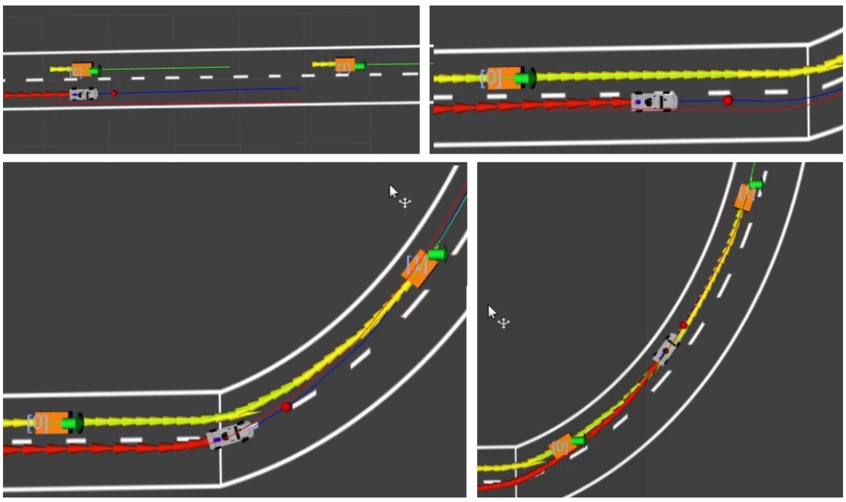

Figure 5.4: Merging into Traffic

5.1.4 Merging into next lane with opposite traffic

In scenario presented in Figure 5.5 driving lane is blocked by a series of obstacles and the left lane is occupied by a moving obstacle. Ego vehicle starts slow in the driving lane and waits till the obstacle is passed in the left lane and starts driving forward.

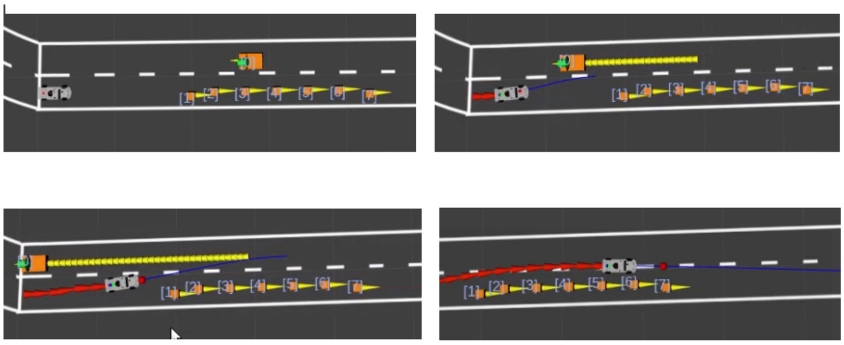

Figure 5.5: Lane blocked by series of static obstacles and vehicle in next lane driving opposite

This situation may lead to ego vehicle getting stuck in the middle of the road. If the driving speed is slow because of temporal horizon ego vehicle can see only short distance into the future, if a fast moving obstacle in left lane not visible in 7 seconds(5 planning and 2s of safety) of temporal horizon then the ego vehicle starts lane change and if there is time to abort it will abort and if not the ego vehicle will stop in middle of the lane lane due to no path ahead, if the obstacle proceeds without stopping for ego vehicle. This can be avoided by a behavioural layer with longer spatial scenario analysis. As the planner is not created for controlling the vehicle to back off, a different planner much be used generally behavioural layer switches to a off road planners in these scenarios.

5.1.5 Road Blocked or Pedestrian Ahead

In scenario presented in Figure 5.6, road is blocked by a series of static obstacles, the vehicle enters the empty left lane, slows down and finally stops when it cannot find route ahead.

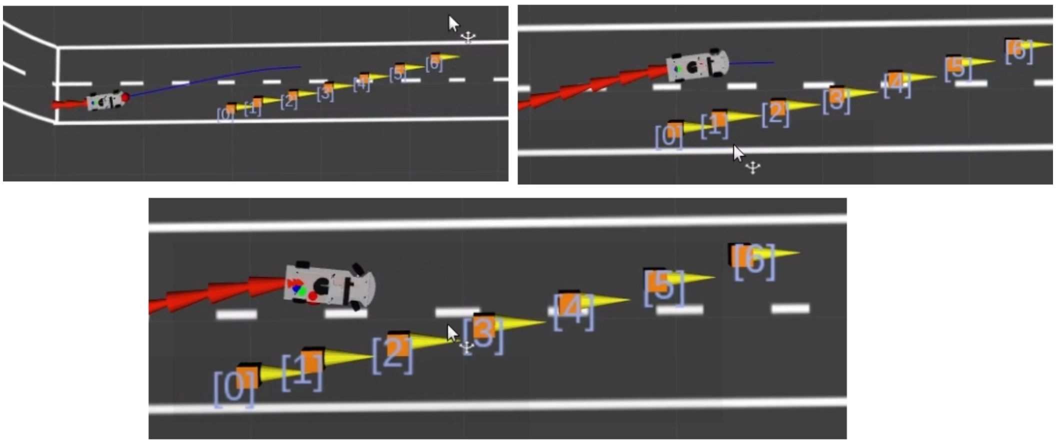

Figure 5.6: Road Blocked by series of obstacles

A pedestrian on road is considered similar to a road blocking, in this case as shown in Figure 5.7 the ego vehicle initially drives at full speed, then the vehicle slows down(shorter blue line representing a slowed vehicle speed) and the robot finally comes to halt few meters ahead of the pedestrian. This is a tunable parameter and currently at maximum value for safety.

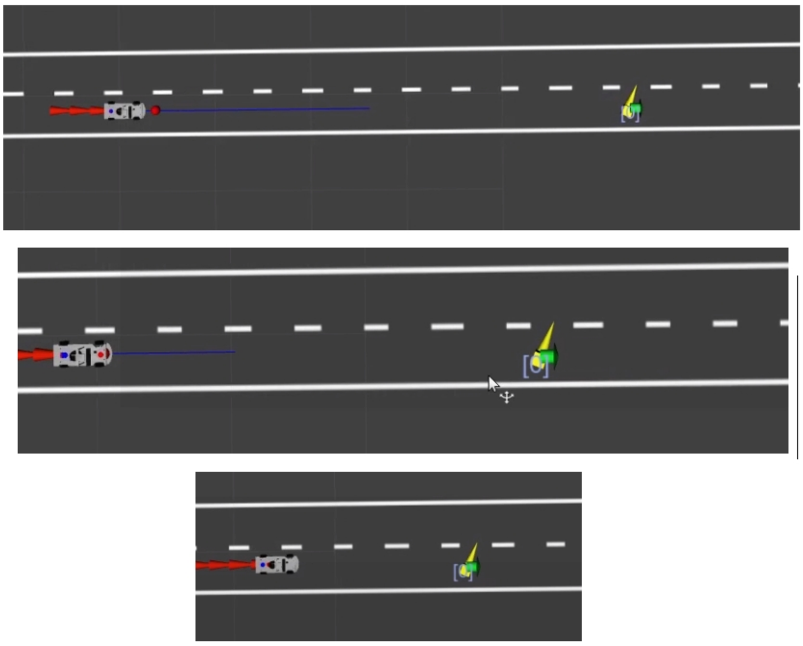

Figure 5.7: Pedestrian Ahead on Road

5.1.6 Dynamic Obstacles - other vehicles

The main objective of the planner is to adjust to the sudden changes in the environment caused by the dynamic obstacles in surroundings, here various sub scenarios are described where the ego vehicle has to react to sudden breaking from vehicles ahead.

In scenario presented in Figure 5.8, there are two situations. In situation 1 the car aborts a lane change when the slow moving dynamic obstacle in left lane is detected, then once the dynamic obstacle is passed the vehicle shifts to left lane to avoid the stopped dynamic obstacle in driving lane. This situation is similar when a vehicle ahead stops to drop off a passenger or waiting for parking spot. In situation 2, the car doesn't choose lane change initially and slows down till it finds enough room in left lane to drive ahead of the stopped dynamic obstacle.

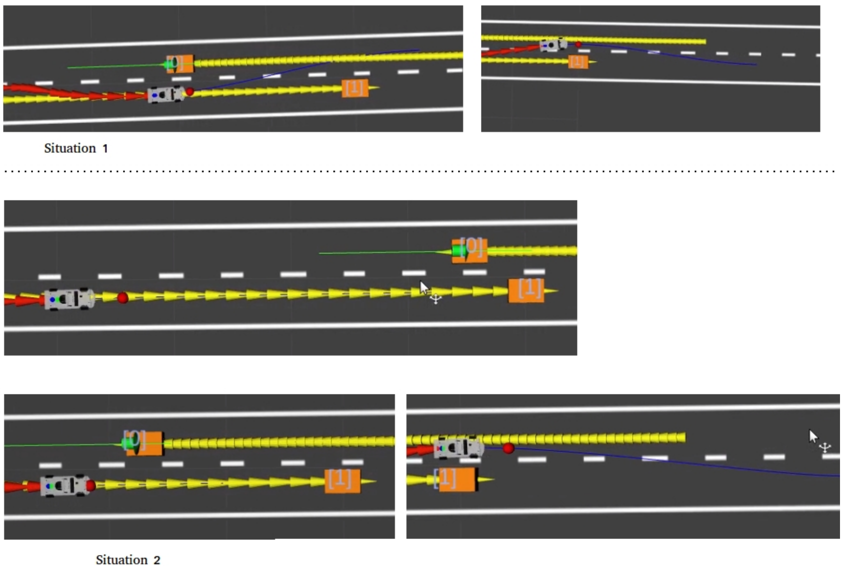

Figure 5.8: Dynamic Obstacle Ahead stops in middle of road

In scenario presented in Figure 5.9 there are two situations with different thresholds for driving, in situation 1 a safe 2s+ distance to vehicle ahead is chosen, here the ego vehicle stays far away from the vehicles ahead and when it stops it stops relatively farther from the vehicles ahead. In situation 2, the threshold has been adjusted to 0.5s leading to a aggressive behaviour of ego vehicle. The vehicle drives closes to the obstacles ahead and when the dynamic obstacles ahead stop suddenly, distance between the ego vehicle and the obstacles ahead is very narrow.

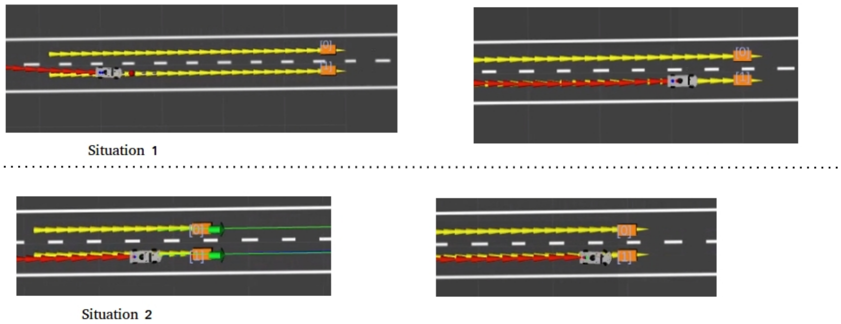

Figure 5.9: Two Dynamic Obstacles ahead stop suddenly

5.2 Criteria Based Evaluation

In this section, proposed planner is validated against the common criteria of evaluating any alogirithm, i.e. optimality, feasibility, completeness and run-time.

5.2.1 Optimality

In this thesis we discuss about the optimality of timing horizon, subsection 3.10.1 already defines regarding the various timing constraints chosen in this planner. A larger planning horizon will enable planner to create a longer and better path but due to the unpredictability of the environment the plan will not be valid after certain duration, a larger horizon will also increase the run-time of the planning algorithm. The planner proposed in this thesis is only a local planner and always needs inputs from a behavioural layer or a global planner to choose target lane, target velocity etc thus a short planning horizon is suitable for this proposed planner.

An example of how horizon will affect optimal planning for current planner is shown in figure 5.10. Here T0,T1 are the trajectories with horizon T and T2, T3 are trajectories with horizon T'. In this condition if a lane change has been requested then trajectory T1 is chosen but with increased horizon trajectory T3 will be chosen which is more efficient. These situations can be improved by lane selection algorithm in behavioural layer which looks for occupancy of different lanes and suggest the one best suitable lane. Similarly if an exit has to be taken on road, a long horizon would choose a plan with reduced speed compared to high speed path with short horizon. This can also be solved by having a velocity planner in the behavioral layer of planner.

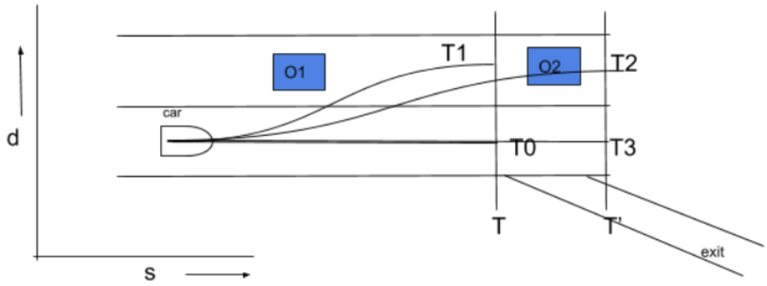

Figure 5.10: Horizon Optimality reference

move to future works

Another horizon generally in discussion for a planner is spatial horizon which discusses how long is the path generated, as per this planning criteria at low velocities the spatial horizon considered is very small thus the planner may not make right decisions because of conditions like missing the obstacles ahead etc. This shortcoming can be improved by creating a spatial path with longer horizon in behavioural layer at lower speeds and allowing the local planner to follow new spatial path than following the global reference path. This is an efficient method as the path planning is generally less expensive than trajectory planning. As shown in figure 5.11, following the original reference path will lead to trajectories that turn a lot causing discomfort due to obstacles on the side of road that enter the road, thus by using an optimised reference path robot can plan efficiently even using short horizons.

The resolution of the sampling in acceleration selection and lateral distance selection will also affect the optimality of planning, a chosen plan can only be optimal of the trajectories created by sampling, higher the number of samples, larger are the possibilities and a best selection is possible.

From the above discussion it can be stated that a planner that has a longer spatial horizon for path planning and short time horizons for trajectory planning will lead to an efficient planner.

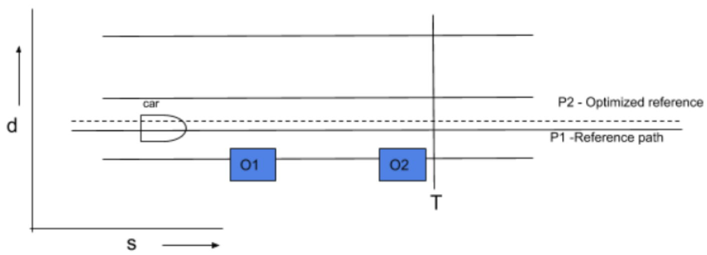

Figure 5.11: Optimized Reference Path

5.2.2 Feasibility

Ability of the vehicle to traverse the created trajectory determines feasibility, generally curvature of the path, smoothness and accelerations determine whether a trajectory is feasible or not. The proposed planner creates feasible trajectories at lower speeds because of the third order splines used, higher speeds require fifth or higher order splines to maintain continuity in path speed. Discussions in [14] [12] throw light on how to achieve higher degrees of smoothness, which approach is better in which driving conditions. As the intended application for this thesis is a modelcar, constant acceleration profiles are used due to limitation in ability of the car to track small changes in velocity and inaccuracies in measurement. these can be easily replaced by a smoother higher order polynomials with a better hardware platform. Consistency in paths evaluated with respect to previous plan is another factor in feasibility, the current implementation penalises the trajectories deviating from previous plan and also takes into account current orientation of the vehicle in choosing a path. Thus creating smoother transitions from one state to another by respecting current driving orientation.

5.2.3 Completeness

An algorithm is said to be complete if it can result in a solution every time. A motion planner can be called complete if it returns path if it exists in the space searched. Like many other sampling based approaches the planner proposed in this thesis only probabilistically complete. That is, probability of finding a solution approaches to one as the number of samples increases. If there are higher number of samples in the configuration space then higher are the chances of finding a solution. If a planner cannot find a solution within sampled region it forces the car to go into emergency manoeuvre.

In figure 5.12, there are only two sampled end states and there is no solution found by the vehicle, by increasing the number of lateral samples a solution can be easily found. In general condition of completeness can be improved in two ways, first is to sample as many points as possible and as closely as possible in the solution space. Second method is to keep on sampling till an end solution is found or timeout has been reached. The former method will reduce the computational performance while the later can be complex and expensive also. It is recommended to achieve completeness for safety purposes in autonomous driving.

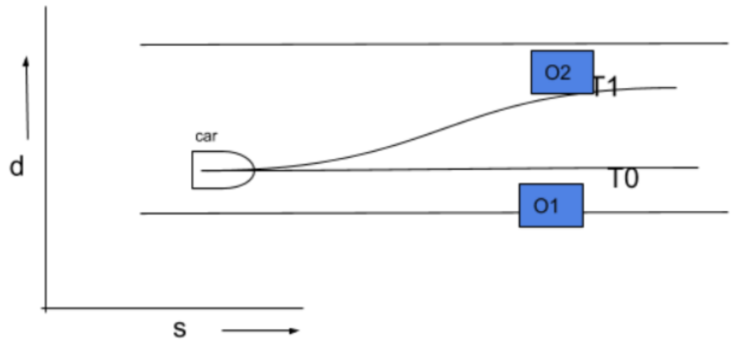

Figure 5.12: Probablistic Completeness

5.2.4 Runtime

Let n_a denote the number of acceleration/deceleration profiles, n_s denote the stopping deceleration profiles and n_l denote the number of lateral distance samples. Then the maximum number of samples created is $(n_a + n_s) * n_l$, generally stopping profiles are less as at higher deceleration lower number of lateral samples are chosen due to limitations in vehicle dynamics. This thesis employs a hybrid combination of two methods mentioned in 5.2.3 to achieve completeness. Initially all the sampled target states are assigned an initial cost as mentioned in subsection 3.10.6. These states are sorted based on costs and the lowest cost trajectory is evaluated first for the dynamic costs, then the states are sorted till the top of the queue is the lowest cost evaluated target state. Therefore in best case only one trajectory is evaluated and the best case complexity is $O(n)$ and the worst case complexity is $O((n_a + n_s) * n_l)$.

Trajectory evaluation with respect to dynamic obstacles is an expensive process in evaluation of trajectories, generally simulation based methods as discussed in [10] are generally expensive generally in terms of $O(n_o * n_n)$ where n_o is the number of obstacles and n_n is the number of simulation steps. This thesis employs a simple collision checking algorithm with constant time for evaluating one obstacle thus reducing the complexity to $O(n_o)$, number of obstacles. This process is not effective in intersections, currently a conservative approach to wait for other obstacles to pass is used, a simple approach as discussed in [6] which has a performance better than the simulation based algorithms can be employed in future.

Write about the execution time for different number of samples, maximum execution time, minimum time from eval-

5.2.5 Deliberative Approach

The planner proposed in this thesis maintains a mix of deliberative and reactive approach. Deliberative by evaluating a trajectory completely before committing to it. This approach is important to create trajectories adhering to traffic and comfort. In general all the planners evaluate all the sampled trajectories then choose the best based on different costs. The planner proposed in this thesis does not follow this convention and once it finds a best trajectory it stops evaluating the other trajectories as presented in subsection 3.10.6. This does not limit the real-time response of the trajectory as the sampling is chosen such that the worst case response time is within the hard real-time response required by the planner.

5.2.6 Low Computational Costs

Though computation power is available cheaply, it is important to create solutions which are cheaper and can be employed in large scale. In this case sampling based approaches generally fare well and run on a low computational hardware. In contrast lattice based approaches such as [14] [17] [18] computationally expensive and require a GPU to run. Low computational costs can enable the technology to be adopted to a larger market. Safety should not be compromised for sake of low computational costs and the planner proposed here employs large range of acceleration profiles to bring the vehicle to halt easily in case of an emergency.

5.3 Comparisons to other Planners

Advantages of combining path and velocity? - How it can reduce sampling state.

Acceleration profiles for emergency stopping

Simulation based approaches to the proposed approach in this planner for collision checking.

Why it is not important to validate other trajectories once the best trajectory is found.

Urban driving needs strong abilities to stop immediately and lattice planners cannot do this as the number of acceleration profiles increases the evaluated trajectories gets increased in thousands.

Eg: CMU - increasing one acceleration will add 13000+ more trajectories, discretizing time by one more step will add 200,000 thousand more trajectories

Comparison here doesn't make sense as there are no numbers to compare with other planners, theoretical comparisons can be presented in related work in forms of short comings in different planners. Here may be just add a table like in CMU or DISS shui to evaluate thus limit the spatial and temporal horizon the planner can evaluate. At high speeds a larger spatial horizon is needed but generally temporal horizon remains same at all the speeds thus we chose to plan in a temporal horizon to allow planning at all multiple speeds.

Divide the trajectory planning and behaviour layer, with this the complexity of solving the task can be reduced drastically. If the trajectory planner need not worry about the behaviours and focus solely on driving safely it will enhance the performance of the vehicle and achieve the costs at a low computational cost.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

6.2 Future Work

Replace by smoother polynomials over splines, especially in curves and when not following centre lane, they tend to be very bad.

// Diss shui thesis - read though page 80 and understand further on benefits and demerits of polynomials vs splines. Add some in evaluation and some in future work

Prediction of state from where the planner should start planning instead of current position. Due to inaccuracies in current planners measurement of speed and acceleration it is tough to estimate where the vehicle will be when the planner is under execution. Currently based on assumption that the vehicle will follow the current path for next few ms, it is made offset in control node. This can be improved to have better synchronizaton between planner and controller.

Bibliography

[1] Z. Boroujeni, D. Goehring, F. Ulbrich, D. Neumann, and R. Rojas. Flexible unit a-star trajectory planning for autonomous vehicles on structured road maps. In *2017 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, pages 7–12, June 2017. 14, 20

[2] R. G. Cofield and R. Gupta. Reactive trajectory planning and tracking for pedestrian-aware autonomous driving in urban environments. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 747–754, June 2016. 12, 13

[3] Gerald Cook. *Robot Navigation*, pages 324–. Wiley-IEEE Press, 2011. 6

[4] P. Czerwionka, M. Wang, and F. Wiesel. Optimized route network graph as map reference for autonomous cars operating on german autobahn. In *The 5th International Conference on Automation, Robotics and Applications*, pages 78–83, Dec 2011. 8

[5] DARPA. Route network definition file (rndf) and mission data file (mdf) formats. 8

[6] L. Garrote, C. Premebida, M. Silva, and U. Nunes. An rrt-based navigation approach for mobile robots and automated vehicles. In *2014 12th IEEE International Conference on Industrial Informatics (INDIN)*, pages 326–331, July 2014. 36

[7] Christos Katrakazas, Mohammed Quddus, Wen-Hua Chen, and Lipika Deka. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. *Transportation Research Part C: Emerging Technologies*, 60:416 – 442, 2015. 14

[8] Joseph Kearney, Hongling Wang, and Kendall Atkinson. Robust and efficient computation of the closest point on a spline curve. In *In in Proc. 5th International Conference on Curves and Surfaces*, pages 397–406, 2002. 13

[9] J. Kim, K. Jo, W. Lim, M. Lee, and M. Sunwoo. Curvilinear-coordinate-based object and situation assessment for highly automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 16(3):1559–1575, June 2015. 12

[10] Sascha Kolski. *Autonomous Driving in Dynamic Environments*. PhD thesis, ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE, 2008. 36

[11] X. Li, Z. Sun, D. Cao, Z. He, and Q. Zhu. Real-time trajectory planning for autonomous urban driving: Framework, algorithms, and verifications. *IEEE/ASME Transactions on Mechatronics*, 21(2):740–753, April 2016. 12, 16

[12] D. Madås, M. Nosratinia, M. Keshavarz, P. Sundström, R. Philippsen, A. Eidehall, and K. M. Dahlén. On path planning methods for automotive collision avoidance. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, pages 931–937, June 2013. 35

[13] Kristijan Maček. *Autonomous vehicle navigation in dynamic urban environments for increased traffic safety*. PhD thesis, ETH Zurich, 2010. 11

[14] Matthew McNaughton. *Parallel Algorithms for Real-time Motion Planning*. PhD thesis, Carnegie Mellon University, 2011. 35, 37

[15] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. On a formal model of safe and scalable self-driving cars. *CoRR*, abs/1708.06374, 2017. 20

[16] Hongling Wang, Joseph Kearney, and Kendall Atkinson. Arc-length parameterized spline curves for real-time simulation. In *In in Proc. 5th International Conference on Curves and Surfaces*, pages 387–396, 2002. 13

[17] Shuiying. Wang. *State Lattice-based Motion Planning for Autonomous On-Road Driving*. PhD thesis, Freie Universität Berlin, 2015. 12, 13, 15, 37

[18] M. Werling, J. Ziegler, S. Kammel, and S. Thrun. Optimal trajectory generation for dynamic street scenarios in a frenet frame. In *2010 IEEE International Conference on Robotics and Automation*, pages 987–993, May 2010. 16, 37

[19] Wenda Xu, Junqing Wei, J. M. Dolan, Huijing Zhao, and Hongbin Zha. A real-time motion planner with trajectory optimization for autonomous vehicles. In *2012 IEEE International Conference on Robotics and Automation*, pages 2061–2067, May 2012. 12, 22

[20] Julius Ziegler and Christoph Stiller. Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios. In *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IROS'09, pages 1879–1884, Piscataway, NJ, USA, 2009. IEEE Press. 12