Smoothed Optimal Planning for Autonomous Vehicles

Charan Karthikeyan Parthasarathy Vasanthi
School of Engineering
University of Maryland
College Park, Maryland 20740
Email: cparthas@terpmail.umd.edu

Nagireddi Jagadesh Nischal
School of Engineering
University of Maryland
College Park, Maryland 20740
Email: njnischal@gmail.com

Arpit Maclay
School of Engineering
University of Maryland
College Park, Maryland 20740
Email: arpitmac74@gmail.com

***Abstract*—The current survey shows that nearly 1.25 million accidents that take place are due to car crashes[National Highway Traffic Safety Administration, 2013]. So, we decided come up with some possible solution to this problem. Planning an Autonomous Vehicle involves various interactive components in the external and internal environment, such as obstacles, lanes and comfort. To achieve this we used different methods and algorithms such as splines and flexible nodes. This project uses an extended A* algorithm to plan the movement of the car and the software for simulation used is called Carla. The planner and its modules are explained in the following sections of the paper.**

Keywords- Planning, flexible nodes, autonomous vehicles, control based smoothing, graphical search

I. Introduction

Trajectory planning plays a pivotal role in Autonomous Vehicles. It plays the role of an actor in a planning environment and communicates between the top layer planner and the external environment[1]. The Trajectory planner takes in the output of the route-planner and generates motion-level planning, which is responsible for steering the vehicle. The planner is a collaboration of sample based planning and optimized based planning. The hybrid technology of the planner is to eliminate the limitation of the two planners when implemented separately. The planner uses edge-augmented graph that would allow the methods used for sampling to numerically estimate certain trajectories of optimization methods[2]. The project focuses on the working of the motion planning.
The planning has two distinct object types: static obstacles and dynamic obstacles. In the case of the static obstacle, we can start to plan the path and their speed profile and prune the search graph based on the cost estimates for each generated path with and without obstacles. In order to handle the dynamic obstacles or moving obstacles the planner uses previous on road methods of planning [1],[2] primarily uses the temporal dimension for construction of spatio-temporal trajectory search-space.

II. Motion Planning Algorithms

The explanation of the different methods of planner and how they are combined to achieve the required hybrid planner is given below.

A. *Edge Augmented Search Based Planning*

Graphs are commonly used in sampling-based planning methods. The graphs are constructed using the state of lattice approaches with motion planning primitives. Some of the assumptions that we need to make this work are stated below.

- We capitalize on the fact that the car only drives forward in the corridor-like environment, for the on-road planning task.
- We sample the whole road or needed area of search points to minimize the error which you get from the path optimization methods.
- A geometric formulation is accounted for road geometry and modelling simple nudging maneuvers which are done by imitating the smooth and continuous path obtained through optimization methods.

The outcome from this efficient graph search algorithms that numerically approximates continuous optimization-based path, smoothing and nudging with resolution- achieved.

1) Continuous Path Nudging and Smoothing: The movement of the vehicle in the environment i.e the lane can be modelled as a corridor like structure with densely sampled way-points sequence representing the center of the environment, and the vertical boundaries of the structure i.e the edge nodes. The path planning is done by moving along the points in a horizontal fashion, these are called nodes. The nodes are then subjected to the boundary criteria of the environment of the road. The path planning algorithm is able to change or edit the paths according to the object and plan around them if possible. For path nudging and smoothing, we typically want a nudge on the way-points, which leads to improvement in the smoothness of the path. From the above figure of the nudging graph we can see two aspects that play a major role in smoothing the final path and plan for it. The two aspects are

- Proximity towards center line of the lane or environment and vehicle.
- Metric for the overall smoothness of the plan to be executed

The way-points on the center-point of the lanes at different points in the graph are represented by the variable $x^*{}_i$,to find the new position represented by x_i and the cost function

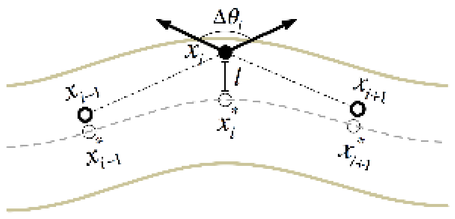


Fig. 1. Nudged Path

between two way-points is given by the equation below. The

$$\underset{\{x_i\}}{\operatorname{argmin}} \sum_i \omega_{cl} \cdot \|x_i - x_i^*\|^2 + \omega_{sm} \cdot \frac{(x_i - x_{i-1})(x_i - x_{i+1})}{\|x_i - x_{i-1}\|\|x_i - x_{i+1}\|}$$

Fig. 2. Way-point cost function

cost will also includes the smoothness constant of the new generated paths and is in direct correlation with the jerkiness of the car when moving in cases such as avoiding an obstacle, tackling a tight curve and changing of lanes. The smoothness captures the angular change $\Delta\theta_i$ at each way-point x_i because, in case of path nudging, we choose an extended node-based search pattern. This gives us a physically interactive way to tune the parameters for changing and experimenting with the values of the planning output. When the way-points $x^*{}_i$ are nudged into the lateral direction parallel to the way-point node, the point is determined based on the three artificial forces to be taken into account. The following are the type of forces:

- Attraction of the node to the actual path or the nominal path.
- The repulsive force of the obstacle to the nominal path of the planner.
- The contraction or the extent of the extended nodes in the world map.

.An example of this will be when the nudged lateral node is on the edge of the lane, it is not recommended to go that far out to the edge of the road in the real world point as the car moves to center itself to the p[provided way-point. To eliminate this possibility a repulsive force is applied to balance the remaining forces laterally moving towards the nominal path. The equations of the forces are given below. where $\delta\rho(x)$ is representative of the potential function in relation with the present/current state of x,δ is the test function that

$$f_a(x_i) = \omega_a \cdot (x_i - x_i^*)$$
$$f_c(x_i) = \omega_c \cdot \left(\frac{x_{i-1} - x_i}{\|x_{i-1} - x_i\|} + \frac{x_{i+1} - x_i}{\|x_{i+1} - x_i\|}\right)$$
$$f_r(x_i) = \omega_r \cdot \frac{\delta[\rho(x_i) < \rho_t]}{[\rho_t - \rho(x_i)]^2} \cdot u[\frac{\partial\rho}{\partial x}(x_i)]$$

Fig. 3. Repulsive Force Equation

returns 'true' if the if condition is satisfied,u is the function that gives the unit direction of the vector path. The iterative search converges if the value satisfies the below equation.

$$\underset{\{x_i\}}{\operatorname{argmin}} \sum_i \|f_a(x_i) + f_c(x_i) + f_r(x_i)\|$$

Fig. 4. Satisfying Condition

The path smoothing and the nudging uses a very similar kind of a method, they will be using the same way-point representation with only one DOF along the lateral direction to that of the way-point $x^*{}_i$. The cost function not only dependant on these parameters but also on the neighboring states such as the number of lanes. This introduces a new problem in terms of convexity and decreases the robustness of the resultant graph. To remove this problem we use the method of Graph based search method. The upcoming section of the paper gives a more detailed explanation of this method.

B. Edge Augmented Graph

The graph based search method uses a discrete graph in the world environment, where the way-points are spread out over the full length of the corridor of the world environment. The basic idea is to connect all the nodes and between the way-points to make it continuous. The problem with it is that it lies in finding and connecting the correct nodes to get the smoothed path with the influence of nudging. The equation of the smoothed path does not comply with this and makes it difficult to calculate the cost of the path.[1]

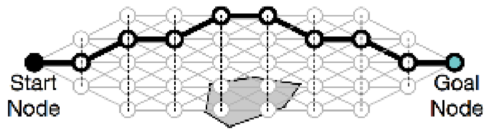


Fig. 5. Edge Augmented Graph

The edge augmented graph is constructed in two steps.

- Create a traditional spatial graph by laterally sampling the way-points along the nominal path and evaluate the spatial edges for collision detection when constructing the elastic nodes. This removes the unfeasible edges while searching.
- Compose the Augmented Nodes and Augmented edges by augmenting the spatial nodes with a directional graph such as IN and OUT edge for connection to the neighbour spatial nodes, the elastic-nodes are then joined with the help of elastic edges.

For example, if there are M incoming nodes and N outgoing nodes. The total number of nodes is given by $M*N$ augmented nodes through augmented edges. Each of these augmented nodes has a corresponding spatial node to map to. This allows

helps us to calculate the smooth path using the cost function. The sizes of the spatial graph are predefined and hence the size of the edge-augmented graph is also of predefined size. Since the graph is predefined the search process, it returns a deterministic run-time.
This type of modeled graph is known as Single-Source-Shortest-Path(SSSP) problem. The edges of the graph predominately always point ahead i.e to the front.

C. Constraint Satisfying Reference Generation

From the algorithm we can generate a sequence of augmented nodes, this is used to generate a trace of linear spatial path i.e the edges on the graph. But his method has two limitation factors[3].

- The edges create a non-smooth i.e the generated path is a piece wise nonlinear.
- The generated nodes and edges does not have a speed profile attached with the nodes. They also take into account the moving objects and their dynamic constraints such as lateral and longitudinal acceleration when adding a speed profile.

To negate or minimize this limitation the planner is added with a Linearquadratic regulator-trajectory controller to smooth out the trajectory to rectify the first limitation. To rectify the second limitation we adopt an efficient algorithm to take care of it. We assume that the reference path is denoted by ρ and the speed profile asν .The reference path consists of a set of spatial states such as the position, heading, curvature and arc-length of the world environment or the lane of the road. For simplicity purposes, the curvature and the arc-length are denoted by K and S respectively. The dynamic constraints are imposed and added to the speed profile and are denoted as a sequence of scalar speed values. We assume that for every iteration, the lateral acceleration constraint $a^{lat}{}_{m}ax$ is imposed, and is set to be the highest speed at each point along the path. The speed profile uses a finite-difference form to generate the speed profile as shown in the Fig.6 The values of the speed

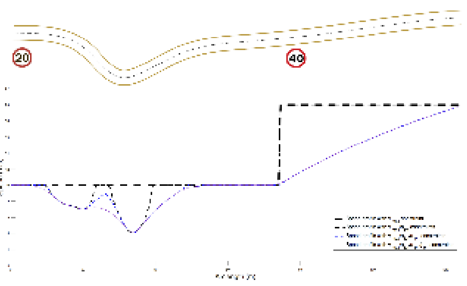


Fig. 6. Speed Profile with dynamic constraints

profile based on the lateral acceleration as given below The acceleration constraint inside the root can be seen to reach singularity on small-curvature or on the straight segments in Fig 6. We can also see that even though the acceleration constraint is limited but the speed profile cannot be used as it does not take into consideration the longitudinal acceleration

$$\text{continuous: } v \leq min\{\sqrt{\frac{a^{lat}_{max}}{\kappa}}, v_{max}\}$$

$$\text{finite difference: } v_i \leq min\{\sqrt{\frac{a^{lat}_{max}}{\kappa_i}}, v_{max}\}$$

$$\text{continuous: } -d^{lon}_{max} \leq \dot{v} \leq a^{lon}_{max}$$

$$\text{finite difference: } -d^{lon}_{max} \leq \frac{v_i^2 - v_{i-1}^2}{2|s_i - s_{i-1}|} \leq a^{lon}_{max}$$

which is denoted by $a^{lon}{}_{m}ax$ and the deceleration constraint is given by $d^{lat}{}_{m}ax$. The equation to implement the longitudinal are given by From Fig 6 we can see that the jerk in the graph which is caused by the longitudinal acceleration and deceleration to prevent this we limit the maximum amount of jerk that the car can have by the constraint $j^{lon}{}_{m}ax$. We also assume that the speed profile can be interpolated in the form of a quadratic polynomial function of the arc length. This is given by the equation. Here s denotes the arc-length and

$$v = \alpha \cdot s^2 + \beta \cdot s + \gamma$$

$$a = 2\alpha \cdot sv + \beta \cdot v$$

$$j = 2\alpha \cdot (v^2 + sa) + \beta \cdot a$$

ν denotes the speed profile. Calculating the First-order and Second-order Derivatives of the equations w.r.t time, we get the analytic expression for acceleration a and jerk j. Assuming that we know the values of the previous, current and next speed profile along with their arc lengths we can get the values of α,β and γ. The coefficients can be solved from this equation and

$$\begin{bmatrix} s_{i-1}^2 & s_{i-1} & 1 \\ s_i^2 & s_i & 1 \\ s_{i+1}^2 & s_{i+1} & 1 \end{bmatrix} \cdot \begin{bmatrix} \alpha_i \\ \beta_i \\ \gamma_i \end{bmatrix} = \begin{bmatrix} v_{i-1} \\ v_i \\ v_{i+1} \end{bmatrix}$$

the output values are plugged into the polynomial equations to get the value of jerk j. This algorithm operates in an iterative way and modifies the speed profile to match and adjust the goal point to match the limitation placed on the jerk. The algorithms are also acceptable since the speed profile takes into account all the constraints placeable on the vehicle.

D. Trajectory tracking Control

This section deals with the checking of the trajectory for dynamic feasibility and smoothness. This is tasked with finding the sequence of control or states related to the trajectory and convert the coarse trajectory into a control plan for execution.

To prevent the planner to add unwanted errors and limitations we use a closed-loop system and use spline interpolation to smooth out the coarse trajectory. The trajectory is also based on the fact of the dynamic properties of the vehicle and needs to be taken into account when profiling the path.

1) Controller Based Smoothing: The controller-based smoothing plan uses the Linearquadratic regulator based plan refined by taking into account the kinematics of the vehicle and the type of control states from the APV states. Depending on the model of the vehicle, the final trajectory is checked to be kinematically or dynamically feasible. Even though the majority of the smoothing process is taken care of controller-based sampling methods. the preliminary smoothing is done, as mentioned in the previous sections of the papers. The path smoothing is a conversion of the path to the trajectory by appending a constant slow speed to the spatial trace. For the final model to be kinematically feasible, the final behavior of the car should exhibit the kinematic vehicle model behavior and the Linearquadratic regulator takes care of this. While the trajectory from the coarse plan may have a guarantee of being a collision free. The smoothened path nullifies this guarantee, however, this does not pose a serious threat to the system since it is used to generate the smoothed seeding trajectory. The smoothed trajectory might have a large maximum tracking error based on the quality of the trajectory, which is due to the maximum point-wise Euclidean distance between the smoothed and the nominal trajectories of the planner. if we assume that 1 meter to be the arbitrary threshold of the error. We can say that the maximum threshold between the correspondingly timed points of the smoothed and the nominal trajectories has to be less than or equal to 1 meter. If the values are less than 1 we can say the nominal trajectories to be mild and if the value is 1 it is termed as an aggressive trajectory. From this image above we can see the trajectories that can capture the behavior that we need. The Black dot that we find in the figure is the piece-wise-linear nominal trajectory that shows the behavior. The vehicle nudges away from the objects by moving towards the left. The purple-colored faded points show the movements of the pedestrians and the smoothed trajectory of the Linearquadratic regulator controller is denoted by the red colored non-solid circles and the object is denoted by the black colored square-shaped box. We can observe that dots are concentrated in the parts containing objects, denoting a slow speed profile and after that, we can also see the points are less densely or more sparsely populated, this shows that the speed profile resumes to the previous road set values. From this, we can also conclude that the error is smaller in magnitude than the value of the threshold during low-speed profiles and increase to near the maximum attainable value of the threshold.

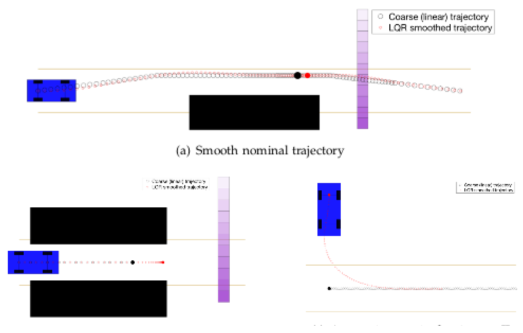


Fig. 7. Linearquadratic regulator Smoothing with aggressive and mild nominal trajectories

The second image of the Figure with two objects on both sides depicts the aggressive behaviors of the car as it does not take into account the objects as the nudging also considers the kinematics of the vehicle and then judges this the object does not comply as a danger to the vehicle. To prevent this we set a cap on the max speed profile to decrease the acceleration and velocity in these parts of the planning.

The final and most challenging of the smoothness of the trajectory is when the vehicle is placed at a 90-degree angle perpendicular to the nominal path and also placed at an offset. in this type of case, the tracking error is very high and does not comply with the limitation placed in them. This is the Linearquadratic regulator controller outshines the proportional integral derivative or PID and the other controllers with its stable and robust behaviors.

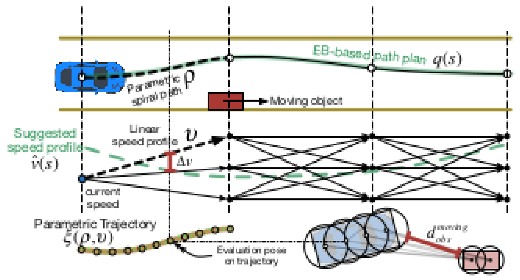

EB-based path plan $q(s)$
Parametric spiral path ρ
Moving object
Suggested speed profile $\hat{v}(s)$
Linear speed profile v
Δv
current speed
Parametric Trajectory $\xi(\rho, v)$
Evaluation pose on trajectory
d_{obs}^{moving}

Fig. 8. Resultant Graph and Nominal trajectory from LQR controller

E. Implemented Algorithm

The final trajectory and their resultant spatial nodes are pictured in the Fig 8. This shows the combination of all the previous explained algorithms.

III. Previous Work

The path generation schemes have been a foundation for motion planning.[4] and [5] proposed a real-time path planning algorithm to fit in to form smooth curves in order to change lanes but could not achieve due to multiple moving uncertain obstacles i.e vehicles and traffic.[6] We came up with optimized planning which provided higher computational efficiency. [7] used the previous results and tried to implement A star algorithm to optimize the path but still [7] did not have the capacity to meeting the real time constraints of the on road drive application even after [8] applied convex optimization to it. For on road motion planning in CMU's boss[9] one layer of fixed poses is sampled by laterally offsetting the center

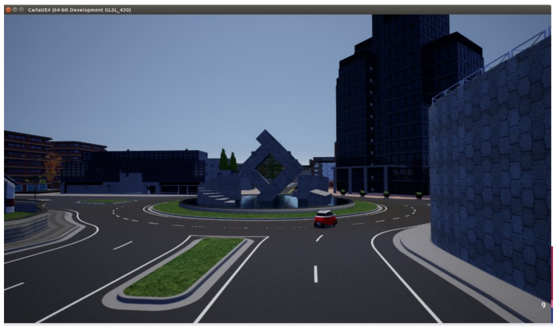


Fig. 9. Carla Simulator

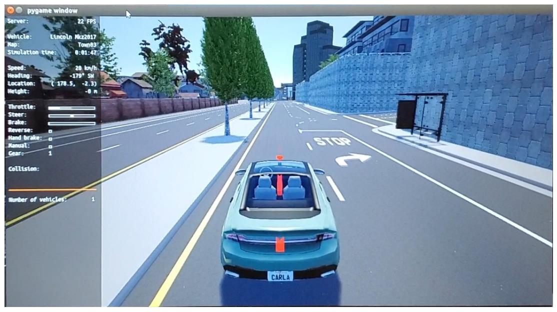
line pose at a short look ahead distance, yet keeping the road heading. Applying linear velocity profiles to each path generates a pool for trajectory evaluation. A graphical lattice is laid out covering the entire road for path sampling, and a temporary dimension is then appended to the graphical lattice to create a dense tempo-graphical search space. However, the computation of this algorithm is exhaustive and in a dynamic case like a moving car, the computations have to be quick. Since the algorithm explores unnecessary tempo-graphical space there has to be an addition of reasoning in the exploration process of the tempo-graphical space. The urban driving planning in [9] and for highways [10] motivate our work. Because the motions are very similar and unified which can be predicted with ease. It will be desirable for our algorithm to look ahead on the road geometry and be prepared for it beforehand and create a future velocity profile such that unnecessary computation can be avoided which in turn will reduce jerkiness for the vehicle. In cases of simple sampling patterns, as the speed increases the computation will start to lag behind hence the lattice approach used in [11] will be more effective. We will try to come to implement a 2 step planning algorithm. The Reference Trajectory Planner (I) a non-parametric human-like reference trajectory for road geometry and obstacles. The Tracking Trajectory Planner (II) promotes an optimal path and provides the system with a set of optimal nodes to be followed on which curve fitting is done using splines. We believe that we can implement a hybrid algorithm which will be a mixture of the two and will reflect high-level directives.

IV. Simulator Environment

To simulate our planner, we wanted a platform which can easily visualize the movement and having a 3D visual would really help. We shortlisted a few softwares and of these, we found Carla to be the best fit. It is an open source planning simulator for autonomous vehicles. It is built on the Unreal 4 Engine and the environment is easy to alter for different test cases of the planner. The maps can be generated using a software called RoadRunner. We have used the current urban environment called Town 3 in Carla.

V. Results

We were able to replicate the planner from the paper [3] and simulate it in our simulation environment. The planner currently, when initiated moves from one point to another without any static or dynamic obstacles.The planner as explained under methods was partially implemented to move without obstacles. The planner also generates certain errors due to the heavy computational requirement and also minor errors in the planner when changing lanes. We were able to get a smooth path for a straight lane. However, we encountered some errors during curvatures and lane changes due to the cost function.



```
Server:                 22 FPS

Vehicle:        Lincoln Mkz2017
Map:                     Town03
Simulation time:        0:01:47

Speed:                  20 km/h
Heading:               -179° SW
Location:        ( 178.5,  -2.3)
Height:                    -0 m

Throttle:
Steer:
Brake:
Reverse:
Hand brake:
Manual:
Gear:          1

Collision:

Number of vehicles:           1
```

Fig. 10. Planned path with Carla simulator

VI. Conclusion

The paper discusses a decoupled planning algorithm which prioritises efficient computation as well as tunability of the planner. Elastic band planning is implemented to take into consideration, static and dynamic objects for swerve movements. The constraint based trajectory planning methods is mainly to follow the generated path but modulates the velocity and acceleration vectors in such a way that it accounts for the constraint factors and the dynamic obstacles. The resultant path usually is a dynamically feasible output for controlling the vehicle. This takes care of all the drawbacks from the traditional approach such as the dimensionality curse, the optimality. Our approach allows us to search on a finer scale in the search space. This gives us a better stability and a tunable planner.

VII. References

1.)LaValle, S. M. (2014). Planning algorithms. New York (NY): Cambridge University Press.

2.)Kala, R. (2016). Graph Search-Based Hierarchical Planning. On-Road Intelligent Vehicles, 187-238. doi:10.1016/b978-0-12-803729-4.00008-8

3.)Gu, Tianyu. (2017). Improved Trajectory Planning for On-Road Self-Driving Vehicles Via Combined Graph Search, Optimization Topology Analysis. 10.13140/RG.2.2.14080.64003.

4.)J.Zielger and C.Stiller,"Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios",Intelligent Robots and Systems 2009,IROS 2009. IEEE/RSJ international Conference on,pp. 1879-1884,2009

5.)J.-W. Lee and B. Litkouhi, Control and Validation of Automated Lane Centering and Changing Maneuver, ASME Dynamic Systems and Control Conference, Oct 2009.

6.)J.-W. Lee and B. Litkouhi, A unified framework of the automated lane centering/changing control for motion smoothness adaptation, The 15th IEEE Intelligent Transportation Systems Conference, 2012.

7.)M. McNaughton, Parallel Algorithms for Real-time Motion Planning. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July 2011.

8.)D. D. et al., Practical search techniques in path planning for autonomous driving, in Proceedings of the First International Symposium on Search Techniques in Artificial Intelligence and Robotics (STAIR-08), (Chicago, USA), AAAI, June 2008.

9.)G. P. Bevan, H. Gollee, and J. OReilly, Trajectory generation for road vehicle obstacle avoidance using convex optimization, Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering, vol. 224, pp. 455473, Jan. 2010.

10.) C. U. et al., Autonomous driving in urban environments: Boss and the urban challenge, J. Field Robotics, vol. 25, no. 8, 2008.

11.) M. e. a. McNaughton, Motion Planning for Autonomous Driving with a Conformal Spatiotemporal Lattice, IEEE International Conference on Robotics and Automation, vol. 1, pp. 48894895, Sept. 2011.