Motion Planning Framework for Autonomous Vehicles: A Time Scaled Collision Cone Interleaved Model Predictive Control Approach

by

Raghu Ram Theerthala Theerthala, Sai Bhargav Kumar A V S, Mithun Babu Nallana, Phaniteja S, Madhava Krishna

Report No: IIIT/TR/2019/-1

Centre for Robotics
International Institute of Information Technology
Hyderabad - 500 032, INDIA
June 2019

Motion Planning Framework for Autonomous Vehicles: A Time Scaled Collision Cone Interleaved Model Predictive Control Approach

Raghu Ram Theerthala[1], A. V. S. Sai Bhargav Kumar[1], Mithun Babu[1], Phaniteja S[1] and K Madhava Krishna[1]

***Abstract*— Planning frameworks for autonomous vehicles must be robust and computationally efficient for real time realization. At the same time, they should accommodate the unpredictable behavior of the other participants and produce safe trajectories. In this paper, we present a computationally efficient hierarchical planning framework for autonomous vehicles that can generate safe trajectories in complex driving scenarios, which are commonly encountered in urban traffic settings. The first level of the proposed framework constructs a *Model Predictive Control*(MPC) routine using an efficient *difference of convex programming* approach, that generates smooth and collision-free trajectories. The constraints on curvature and road boundaries are seamlessly integrated into this optimization routine. The second layer is mainly responsible to handle the unpredictable behaviors that are typically exhibited by the other participants of traffic. It is built along the lines of *time scaled collision cone*(TSCC) which optimize for the velocities along the trajectory to handle such disturbances. We additionally show that our framework maintains optimal balance between temporal and path deviations while executing safe trajectories. To demonstrate the efficacy of the presented framework we validated it in extensive simulations in different driving scenarios like over taking, lane merging and jaywalking among many dynamic and static obstacles.**

I. INTRODUCTION

A. Motivation

The research in the field of Autonomous driving is making significant progress, and a lot of importance is being placed on developing reliable motion planning frameworks. For achieving this, the planners need to have high update rates so that they can handle emergency scenarios that arise due to the unpredictable behavior of the other traffic participants. There have been quite a few approaches like [1], [2] for complex maneuver generation for autonomous vehicles. Both these works propose an MPC framework with a linear model, making the optimization problem simpler. But these frameworks don't guarantee collision-free trajectories. In approaches such as [3] motion planning is done in two steps. They initially plan the geometric path and later plan the velocities along the path. However such reactive planners are prone to get stuck in local minima and are not readily adaptable in complex urban scenarios. The similar problem tackled in [4] using nonlinear optimization techniques are time-consuming and majorly target efficiency in Vehicle dynamic behavior over computational complexity. In this work, we propose a non-myopic hierarchical planner which overcomes these shortcomings and is more reliable. This framework is compared with some existing techniques to show it's significance.

[1]The authors are with Robotics Research Center, IIIT Hyderabad, India. venkata.sai@research.iiit.ac.in, mkrishna@iiit.ac.in.

B. Overview of the proposed Approach and Contributions

The presented approach is on lines with the work presented in [3], which proposes a path velocity decomposition in a two-layer optimization problem. Our work extends beyond, in the way the receding horizon is implemented and making it more general when compared. In contrast to [3] where the notion of horizon plan is restricted to one step we propose a finite time horizon planning. We develop a hierarchical planning framework which solves for an initial trajectory in an optimization scheme in the first layer. This trajectory is then passed to the velocity optimization scheme which operates in a receding horizon strategy to solve for the collision-free velocities along the trajectory. The velocity optimization scheme forms the second layer of the framework. The framework switches back again to the first layer only if it cannot obtain the collision-free velocities along the trajectory or at the end of the horizon. By getting the notion of horizon plan, we reduce the abrupt changes in the motion profile that are encountered in [3] at the same time leveraging on its features. The trajectory optimization scheme proposed in the paper is formulated as an MPC routine which seamlessly integrates curvature bound and road boundaries as constraints. The second layer of the framework solves the *time scaled collision cone* constraint repeatedly by formulating it as a convex problem to obtain the collision-free velocities. Due to the closed form nature of the second layer and as the first layer operates at large intervals the computation time of the presented framework is very low.

The key contributions of this work include :

- A formulation of a framework that can perform complex maneuvers while handling the unpredictable behavior of other participants.
- A hierarchical optimization routine that has high repeatability and fidelity despite several constraints.

The rest of the paper is organized as follows: Section II presents an overview of the work related to this field. Section III summarizes the symbols and notations used in this work. Section IV presents the trajectory optimization scheme. Section V reviews the time scaling and introduces the velocity optimization scheme. The switching method used in this formulation is discussed in Section VI. Section VII provides the validation results of the framework in simulations.

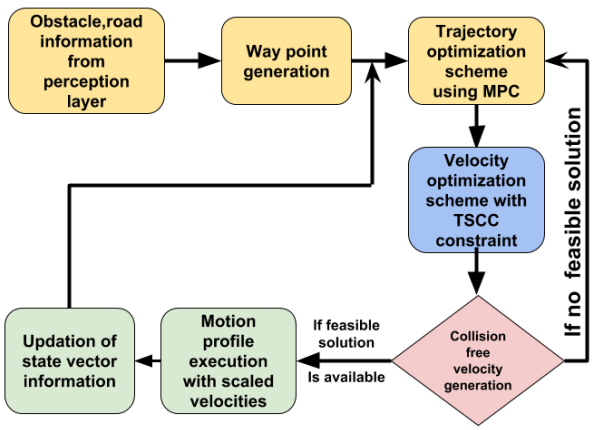


Fig. 1: Overview of the proposed framework

II. Related Work

In this section, we discuss the already existing approaches that are designed for autonomous driving to handle complex scenarios.

The approaches in the field of motion planning span from sampling search based planners to optimization based planners. In [5] provide an excellent review of the current state of the art approaches in motion planning and control techniques for self-driving urban vehicles. Many works like [6], [7] and [8] integrates both sampling-based techniques with optimization to generate refined trajectories. A trajectory planning technique which includes intention predictions integrated is presented in [9]. But the drawback this approach has is that for switching between accelerations they assume high jerk values. In [10] an optimal control based trajectory generation method in a semi-reactive model is proposed. But it needs to generate a large set of candidate trajectories to handle complex scenarios which require a huge computational power. A control scheme based on Lyapunov control scheme is presented in [4] to generate maneuver strategies for the autonomous vehicles. An approach that includes both kino-dynamic constraint and collision avoidance constraint in the optimization framework but is more reliant on general non-linear optimization techniques making its computational complexity high. As discussed earlier, we rely on the hierarchical planning approach to limit our dependency on general non-linear optimization technique. Methods like [11], [12] also provide frameworks that learn to plan paths through experience. In [13] presents a continual planning approach that switches between classic planner that computes the overall strategy and the decision theoretic planners which solve the abstract problems locally.

III. Notation

In this paper, $\mathbf{x}(t_i)$ denotes state of vehicle and $v(t_i)$ and $\dot{\theta}(t_i)$ denotes linear velocity and angular velocity input to vehicle at time instant t_i. We adopted the simple non-holonomic model presented in [14] to model our vehicle. We denote this model by $f_{car}(\mathbf{x}(t_i), v(t_i), \dot{\theta}(t_i))$. We linearize the model of the vehicle around v_{guess} and $\dot{\theta}_{guess}$ using taylor exapansion as follows. We update the linearized model after every iteration of optimization routine in .

$$\begin{aligned}\hat{f}_{car}(\mathbf{x}(t_i), v(t_i), \dot{\theta}(t_i)) = f^*_{car}(\mathbf{x}(t_i), v_{guess}, \dot{\theta}_{guess}) \\ + \nabla_v f_{car}(v(t_i) - v_{guess}) \\ + \nabla_{\dot{\theta}} f_{car}(\dot{\theta}(t_i) - \dot{\theta}_{guess})\end{aligned} \tag{1}$$

IV. Trajectory Optimization Scheme

For the proposed framework, we assume that the high-level goal is being specified and use methods such as [15], [16] to generate a set of guiding way-points lying on the road. We also assume that our framework works hand-in-hand with the perception layer that provides obstacle locations, velocities, and points along the road boundary to this as input. Using this information, trajectory optimization formulated in this layer seamlessly integrates collision avoidance and road boundary constraints along with commonly used actuation constraints into an efficient convex optimization problem. We formulate our problem for a finite time horizon t_N where N denotes the total number of discrete time-steps(Δt). We find the set of all way-points that lie within our horizon denoted by $\{W^k_x\}$ and $\{W^k_y\}$ for $k = 1 \dots p$ and their corresponding time-steps $\{T^k\}$.

$$\underset{\dot{\theta}(t_i), v(t_i)}{\arg\min}\ J = J_{goal} + w_s * J_{smooth} \tag{2a}$$
$$\mathbf{x}(t_{i+1}) = \hat{f}_{car}(\mathbf{x}(t_i), \dot{\theta}(t_i), v(t_i)) \tag{2b}$$
$$v_{min} \leq v(t_i) \leq v_{max} \tag{2c}$$
$$\dot{\theta}_{min} \leq \dot{\theta}(t_i) \leq \dot{\theta}_{max} \tag{2d}$$
$$v_{guess} - \Delta_v \leq v(t_i) \leq v_{guess} + \Delta_v \tag{2e}$$
$$\dot{\theta}_{guess} - \Delta_{\dot{\theta}} \leq \dot{\theta}(t_i) \leq \dot{\theta}_{guess} + \Delta_{\dot{\theta}} \tag{2f}$$
$$a_{min} \leq \tfrac{v(t_i) - v(t_{i-1})}{\Delta t} \leq a_{max} \tag{2g}$$
$$\ddot{\theta}_{min} \leq \tfrac{\dot{\theta}(t_i) - \dot{\theta}(t_{i-1})}{\Delta t} \leq \ddot{\theta}_{max} \tag{2h}$$
$$C_{obst}(\mathbf{x}(t_i), \mathbf{x}_i(t_i), R_i) \leq 0 \tag{2i}$$
$$C_{road}(\mathbf{x}(t_i)) \tag{2j}$$

Where,

$$J_{goal} = \Sigma^p_{k=1}(x(T^k) - W^k_x)^2 + (y(T^k) - y^k_x)^2 \tag{3a}$$
$$J_{smooth} = \Sigma^N_{i=1}\ddot{\theta}(t_i)^2 + \Sigma^N_{i=1}\tfrac{V(t_{i-1}) - 2V(t_i) + V(t_{i+1})}{\Delta t^2} \tag{3b}$$
$$C_{obst} = -(x(t_i) - x_i(t_i))^2 - (y(t_i) - y_i(t_i))^2 + R_i^2 \leq 0 \tag{3c}$$
$$C_{road} = \begin{cases} -(x(t_i)\ a_{left} + y(t_i)\ b_{left} + c_{left}) \leq 0 \\ x(t_i)\ a_{right} + y(t_i)\ b_{right} + c_{right} \leq 0 \end{cases} \tag{3d}$$

The cost function(2a) consists of two parts. The first term drives the vehicle towards intermediate way-points while the second term helps to maintain smoothness in control profiles. w_s is used to maintain the balance between way-point reaching and smoothness of control profiles.

The equation (2b) incorporates linearized motion model into the optimization using (1). The inequality constraints (2c) and (2d) enforce bounds on linear velocity and angular velocity respectively which correspond to physical limits of

vehicle. The inequalities (2e)-(2f) ensure that the approximated vehicle model is valid around the v_{guess} and $\dot{\theta}_{guess}$. These constraints are often termed as trust region constraints in literature and methods such as [17] are used to control Δ_v and $\Delta_{\dot{\theta}}$.

The inequalities (2g)-(2h) incorporate linear acceleration and angular acceleration constraints respectively. It should be noted that at $i = 1$, (2g) helps to maintain continuity between the input linear velocity ($v(t_0)$) and current velocity profile. Similarly, the inequality (2h) helps to maintain continuity between the input angular velocity to the current angular velocity profile. These constraints ensure that switching between two layers happens smoothly.

One of the key advantages of this framework comes from its ability to model collision avoidance between the vehicle and multiple dynamic vehicles using equation(2i). This equation is a Euclidean distance constraint between ego-vehicle and obstacle vehicle (k). We approximate this constraint around v_{guess} and $\dot{\theta}$. Such approximation remains globally valid due to its difference in convex form [18]. The inequality constraints presented in (2j) enforce road boundaries presented in [IV-A]. This constraint function is motivated by the MPC framework presented in [19].

A. Constraint on Road boundaries

Our approach uses a simple implementation of the road boundary constraints which is not restricted to straight lines. The points along road boundaries obtained from the perception layer are fitted into sets of cubic splines. These cubic splines are arc length parameterized following the definition in [20]. The road boundaries are defined hereafter as left and right boundaries with respect to the position of the ego vehicle. The nearest spline to the current position of the ego vehicle is determined by projecting vehicle position on to the left and right boundaries. By using the equation of nearest spline, $f_{spline}(\theta_i)$ with θ_i, being the arc length parameter of the i^{th} spline, the tangent at the projected point is calculated. The constraint is then formulated to facilitate the current position of the ego vehicle to lie in between the tangents of the closest spline as shown in Fig.(2). The $[a_{left}, b_{left}, c_{left}]$ and $[a_{right}, b_{right}, c_{right}]$ shown in the constraint(3d) refer to left and right tangent coefficients at respective boundaries. Here a clearance distance is added to the tangent equations to ensure that the edges of ego vehicle completely lie inside the boundaries.

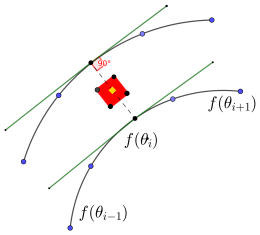
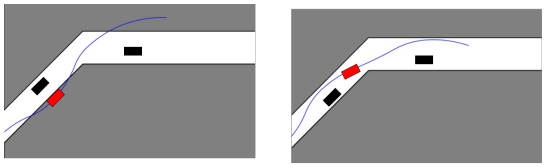


Fig. 2: The figure shows an approximation of road boundaries as tangents(green) at the projections of the current position of ego vehicle(red).



(a) (b)

Fig. 3: (a): Trajectories generated without road constraints on a complex path with static obstacles. (b): Trajectories generated with road constraints on a complex path with static obstacles

V. Velocity Optimization scheme

The first layer optimization provides us with a smooth collision-free trajectory for a finite time horizon. But this cannot guarantee collision avoidance since the optimization scheme does not consider the local changes of the other participants for trajectory generation. For this, we incorporate an efficient reactive setting for velocity modification that handles these unpredictable behaviors. In this section, we discuss the *time scaled collision cone*(TSCC) that is applied to the trajectory to adjust to the local changes and ensure collision avoidance. We first introduce the concepts of time scaling, and TSCC followed by the construction of velocity optimization scheme.

A. Time Scaling

Time scaling transformation doesn't alter the geometric path of a trajectory($\mathbf{X}(t)$), but modifies the time scale from t to τ. This change in scale leads to change in motion profile. That means the same point of the trajectory is reached at a different time instant. For a scaling function, $\dot{s}(t)$ the following equations characterize the change in motion profile of the trajectory.

$$\begin{aligned} \dot{\mathbf{X}}(\tau) &= \dot{\mathbf{X}}(t)\dot{s}(t) \\ \ddot{\mathbf{X}}(\tau) &= \ddot{\mathbf{X}}(t)(\dot{s}(t))^2 + \dot{\mathbf{X}}(t)\ddot{s}(t) \end{aligned} \tag{4}$$

The scaling function $\dot{s}(t)$ can be any monotonic functions. But here we approximated it to a linear function for an arbitrary time interval $[t_i, t_{i+1}]$ for having less computationally complexity. When it is time scaled, the time interval transforms to $[\tau_i, \tau_{i+1}]$. This transformation is given by the relation described in [21].

$$\tau_{i+1} - \tau_i = \frac{2(t_{i+1} - t_i)}{\dot{s}(t_i) + \dot{s}(t_{i+1})} = \frac{2\Delta t}{\dot{s}(t_i) + \dot{s}(t_{i+1})} \tag{5}$$

Using (5) the scaling function gradient $\ddot{s}(t)$ can be derived and is given as

$$\ddot{s}(t) \approx \frac{\dot{s}(t_{i+1})^2 - \dot{s}(t_i)^2}{2\Delta t} \tag{6}$$

$\ddot{s}(t)$ is constant in the time interval $[t_i, t_{i+1}]$ as the scaling function ($\dot{s}$) is approximated to be linear function.

B. TSCC

The solution of velocity space that the ego vehicle can achieve at the time t_{i+1} can be calculated by

$$\dot{\mathbf{X}}(\tau_{i+1}) = \dot{s}(t_{i+1})\dot{\mathbf{X}}(t_{i+1}) \tag{7}$$

Now we reformulate the collision cone [22] as a time scaled variant using (7) which results in

$$\frac{((r_j)^T v_j)^2}{||v_j||^2} - ||r_j||^2 + R_j^2 < 0 \tag{8}$$

where

$$r_j = \begin{bmatrix} x(t_{i+1}) - x_j(t_{i+1}) \\ y(t_{i+1}) - y_j(t_{i+1}) \end{bmatrix}$$
$$V_j = \begin{bmatrix} \dot{s}(t_{i+1})\dot{x}(t_{i+1}) - \dot{x}_j(t_{i+1}) \\ \dot{s}(t_{i+1})\dot{y}(t_{i+1}) - \dot{y}_j(t_{i+1}) \end{bmatrix} \tag{9}$$

As all the terms in equation (9) are known excluding the scaling function ($\dot{s}(t)$) we can simplify equation (8) to single variable in inequality of the form

$$a_j \dot{s}(t_{i+1})^2 + b_j \dot{s}(t_{i+1}) + c_j < 0 \tag{10}$$

where a_j, b_j, c_j are functions of $x(t_{i+1})$, $y(t_{i+1})$, $\dot{x}(t_{i+1})$, $\dot{y}(t_{i+1})$.

This equation is TSCC constraint and solving it will result in a set of velocities that avoid the obstacle for a time horizon.

C. Velocity Optimization

As discussed in [V-B], the TSCC constraint(10) will consider the instantaneous velocities of the obstacles and produce a set of velocities that result in guaranteed collision avoidance. We utilize this and construct a velocity optimization layer which is built on the similar approach presented in [23] and [24].

We formulate an optimization problem to compute for collision-free velocities along the path ($\mathbf{X}(t)$) by computing the scaling function ($\dot{s}(t)$).

$$\arg\min_{\dot{s}(t)} J_2 = (\dot{s}(t_{i+1}) - \dot{s}_{pref})^2 \tag{11a}$$
$$v_{min} \leq \dot{s}(t_{i+1})\sqrt{\dot{x}(t_{i+1})^2 + \dot{y}(t_{i+1})^2)} \leq v_{max} \tag{11b}$$
$$\frac{a_{min}^{lon}}{\sqrt{2}} \leq \dot{s}(t_i)^2\ddot{x}(t_i) + \ddot{s}(t_i)\dot{x}(t_i) \leq \frac{a_{max}^{lon}}{\sqrt{2}} \tag{11c}$$
$$\frac{a_{min}^{lat}}{\sqrt{2}} \leq \dot{s}(t_i)^2\ddot{y}(t_i) + \ddot{s}(t_i)\dot{y}(t_i) \leq \frac{a_{max}^{lat}}{\sqrt{2}} \tag{11d}$$
$$a_j \dot{s}_i(t_{i+1})^2 + b_j \dot{s}(t_{i+1}) + c_j < 0 \forall j = 1, 2, 3, ..., n \tag{11e}$$
$$s(t_1) = 1 \ \forall j = 1, 2, 3, ..., n \tag{11f}$$

Where,

$$\dot{s}_{pref} = \frac{v_{pref}}{\sqrt{\dot{x}(t_{i+1})^2 + \dot{y}(t_{i+1})^2}} \tag{12}$$

Equation(11a) is the cost function which computes the scaling function with minimum deviation from the preferred scale. Inequality(11b) maintains the velocity bound. The inequalities (11c) and (11d) maintain the acceleration bounds. Inequality(11e) is the collision avoidance constraint. The convexity of this constraint depends on the a_j sign. When $a_j < 0$ we linearize it and get a close approximation to the normal constraint. We use the similar method presented in [23] for the constraint convexification. This optimization problem is solved as a Sequential Convex Programming(SCP) problem.

VI. Switching Policy

In this section, we present a switching method used to alter between the layers of the framework. The ego vehicle hierarchically implements the motion profile for a planning horizon and immediately switches to the second layer to handle unpredictable changes of the other participants. The second layer solves for the collision-free velocities of the ego-vehicles repeatedly by considering the present positions and velocities of the other participants. We switch to the first layer only at the end of the horizon, or when the second layer cannot solve for collision-free velocities, that is when the path perturbation is needed to avoid a collision. The compatibility during the switch is maintained by enforcing a continuity constraint as discussed in Section IV. This way the computational complexity is reduced significantly as we avoid solving the computationally expensive MPC iteratively. The responsibility to handle the unpredictable behavior is offloaded to the second layer, which is computationally inexpensive as it has a closed form nature as discussed in Section V.

VII. Results and Discussion

A. Simulation Framework

To validate the presented framework performance, we evaluate it in different driving scenarios. The kinematic motion model of the vehicle is considered for urban scenarios moving with low speeds and assuming the implication of no-slip condition. The simulations are done for various obstacle configurations in each scenario. The planning horizon in the first layer of the framework is varied from 50 steps to 150 steps. In the snapshots of the simulations, the ego-vehicle is shown in red, and the other participants are shown in black. A simulation video of the scenarios can be found in *https://youtu.be/6vLWrOY-pWE*

B. Benchmark Scenarios

We demonstrate the efficacy of the presented framework in Overtaking, Pedestrian jaywalking and Lane merging scenarios.

1) Overtaking: Fig.(4a)-(4b) shows the ego-vehicle overtaking the slow-moving participants ahead of it. The preferred velocity,V^{pref} of the ego-vehicle is set to 10m/s, and the speed of the other participants are varied from 5-8m/s. The planning horizon for this case is set to 50 steps. The ego-vehicle velocity increases initially to overtake the immediate obstacle in front of it. The speed decreases momentarily to come back to the lane where it faces other slow-moving obstacles both in current and immediate next lane. To avoid them the ego-vehicle accelerates to 13m/s and completes the overtaking maneuver and then slows down to V^{pref}. The

Fig.(4d)-(4e) validates the smoothness of the velocity profiles of the ego-vehicle.

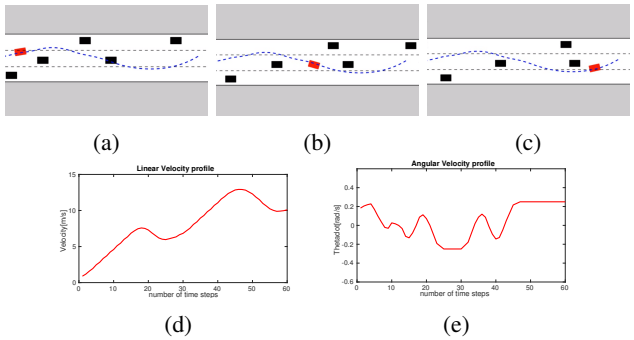


Fig. 4: (4a)-(4c) shows the overtaking maneuver using our framework. (4d)-(4e) show the linear and angular velocity profiles of the ego-vehicle respectively

2) Pedestrian Jaywalking Case: In this case, the initial trajectory is planned to keep the other vehicle only into consideration. The reactive nature of the planner is tested by introducing unexpected pedestrians jaywalking across the road. Fig.(5a)-(5d) shows the simulation of jaywalking scenario. Here the ego-vehicle preferred velocity(V^{pref}) is set to 10m/s. Initially, the ego-vehicle accelerates to reach the preferred velocity, when it faces unexpected pedestrians, it scales down its velocity to avoid the pedestrians. After pedestrians cross the road, the ego-vehicle accelerates and finally settles at V^{pref}. The same is validated in the figures (5e)-(5f).

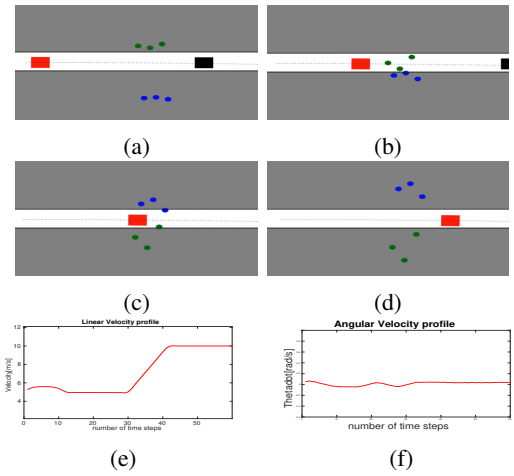


Fig. 5: (5a)-(5d) show the pedestrian jaywalking case. (5e)-(5f) show the linear and angular velocities of the ego-vehicle respectively

3) Lane Merging Case: Fig.(6a)-(6d) shows the ego-vehicle merging from a free lane to the lane with dynamic participants. The ego-vehicle merges on to the second lane as there is an obstacle in very close proximity at the time of merging in the innermost lane. The preferred velocityV^{pref} of the ego-vehicle is set to 10m/s, and the planning horizon is set to 100 steps. The other participants on the main lane are moving at 5m/s-8m/s. The ego-vehicle initially accelerates to on the free lane to achieve the preferred velocity. It then slows at the intersection to avoid collision during merging. As the obstacles ahead and behind the ego-vehicle have different speeds, it tries to adjust between them leading to continuous change in velocity. This behavior is shown in Fig.(6e)

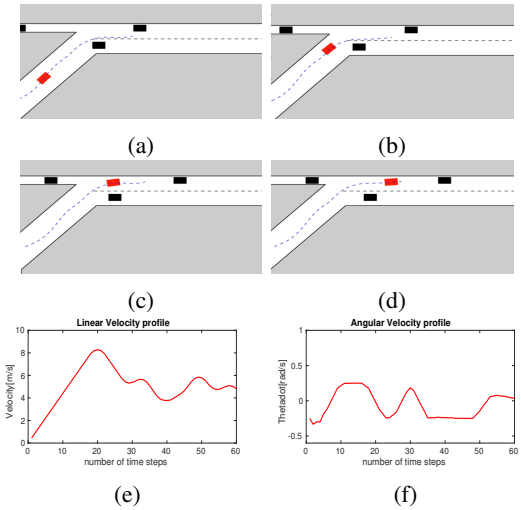


Fig. 6: (6a)-(6d) show the lane merging scenario. (6e)-(6f) show the linear and angular velocities of the ego-vehicle

C. Computational Time

Along with the above-presented examples we have evaluated our framework in several scenarios multiple times. All the simulations are performed on Intel i5 processor with 2.50 GHz clock speed and 8GB RAM. It takes around *12ms* to solve each iteration of the framework. This results in achieving update rates up to *83Hz*.

D. Statistical Comparison

To showcase the efficacy of our framework, we simulated it thoroughly in the presented scenarios multiple times with different parameter settings and compared with a standalone MPC.

1) Run time Comparison: The bar chart in Fig.(7) shows the run time comparison of standalone MPC and our framework with the variation in the planning horizon(N). This is justified as the re-planning occurs at larger intervals in our framework compared to stand alone MPC.

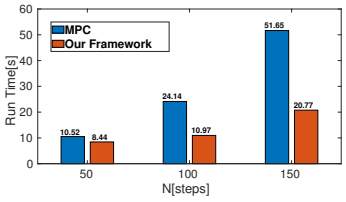


Fig. 7: Run time comparison of MPC and Our framework with variation in number of time steps N

2) Mean Path Deviation Comparison: The bar chart in Fig.(8) shows the mean path deviation of the ego-vehicle from the initial path under lane merging and overtaking scenarios. This is chosen as the metric as continuous perturbations in the heading angle of the ego vehicle might add to the discomfort of the driver. This can be mitigated with the hierarchical planner proposed. Here we quantitatively compare the mean path deviation from the initial path given by the first layer of the planner. It can be seen that the deviation from the initial path is significantly less when compared to MPC. This is achieved as our framework re-scales the velocities on the same trajectory to avoid the obstacles.

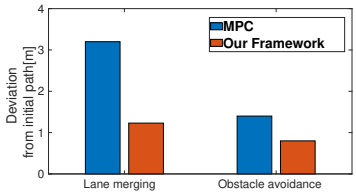


Fig. 8: Comparison of Mean Path deviation from the initial path given by first layer under different driving scenarios

VIII. Conclusion and Future work

In this work, we propose a framework that generates a global plan with complex constraints for a realistic planning horizon.Then we offload the local planning behavior onto Time-scaled collision cone layer. The framework presents a hierarchical planning scheme that switches between the two layers to ensure safe navigation in complex scenarios. The compatibility of switching between layers is taken care by adding continuity constraints in the primary layer. We have validated the framework in different scenarios including pedestrian jaywalking, merging and overtaking. We have quantitatively compared our framework with the standalone MPC and have shown a clear advantage of the proposed framework when compared to it. We have also shown that the framework achieves update rates up to 83 Hz.

Our future work in this direction includes incorporating the uncertainties that arise from the sensors and also adding the probabilistic variant of TSCC, that is already developed in [24] to capture the intent of other participants more effectively.

Acknowledgements

The authors gratefully acknowledge the support provided by MathWorks, Inc for this work.

References

[1] J. Nilsson, P. Falcone, M. Ali, and J. Sjöberg, "Receding horizon maneuver generation for automated highway driving," *Control Engineering Practice*, vol. 41, pp. 124–133, 2015.

[2] J. Nilsson, M. Brännström, J. Fredriksson, and E. Coelingh, "Longitudinal and lateral control for automated yielding maneuvers," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 5, pp. 1404–1414, 2016.

[3] M. Babu, Y. Oza, A. K. Singh, K. M. Krishna, and S. Medasani, "Model predictive control for autonomous driving based on time scaled collision cone," in *2018 European Control Conference (ECC)*. IEEE, 2018, pp. 641–648.

[4] B. Sharma, J. Vanualailai, and R. Rai, "Lane changing and merging maneuvers of car-like robots," *World Academy of Science, Engineering and Technology (WASET)*, no. 72, pp. 1840–1849, 2012.

[5] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on intelligent vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[6] M. McNaughton, C. Urmson, J. M. Dolan, and J.-W. Lee, "Motion planning for autonomous driving with a conformal spatiotemporal lattice," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 4889–4895.

[7] W. Xu, J. Wei, J. M. Dolan, H. Zhao, and H. Zha, "A real-time motion planner with trajectory optimization for autonomous vehicles," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 2061–2067.

[8] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *The International Journal of Robotics Research*, vol. 29, no. 5, pp. 485–501, 2010.

[9] J. Wei, J. M. Dolan, and B. Litkouhi, "Autonomous vehicle social behavior for highway entrance ramp management," in *Intelligent Vehicles Symposium (IV), 2013 IEEE*. IEEE, 2013, pp. 201–207.

[10] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a frenet frame," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 987–993.

[11] Y.-C. Lin and D. Berenson, "Using previous experience for humanoid navigation planning," in *Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on*. IEEE, 2016, pp. 794–801.

[12] D. Berenson, P. Abbeel, and K. Goldberg, "A robot path planning framework that learns from experience," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3671–3678.

[13] M. Göbelbecker, C. Gretton, and R. Dearden, "A switching planner for combined task and observation planning." in *AAAI*, vol. 2, no. 2.1, 2011, pp. 2–3.

[14] M. Babu, R. R. Theerthala, A. K. Singh, B. Gopalakrishnan, and K. M. Kirshna, "Model predictive control for autonomous driving considering actuator dynamics," *arXiv preprint arXiv:1803.03478*, 2018.

[15] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *Ieee Pervas Comput*, vol. 7, no. 4, pp. 12–18, 2008.

[16] S. Sharma, "Autonomous waypoint generation with safety guarantees: On-line motion planning in unknown environments," *arXiv preprint arXiv:1709.00546*, 2017.

[17] Y. Yuan and W. Sun, "Optimization theory and methods," 1997.

[18] S. Boyd, "Sequential convex programming," *Lecture Notes, Stanford University*, 2008.

[19] D. Lam, C. Manzie, and M. Good, "Model predictive contouring control," in *Decision and Control (CDC), 2010 49th IEEE Conference on*. IEEE, 2010, pp. 6137–6142.

[20] H. Wang, J. Kearney, and K. Atkinson, "Arc-length parameterized spline curves for real-time simulation," in *Proc. 5th International Conference on Curves and Surfaces*, 2002, pp. 387–396.

[21] K. Hauser, "Fast interpolation and time-optimization with contact," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1231–1250, 2014.

[22] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *The International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, 1998.

[23] A. K. Singh and Q. C. Pham, "Reactive path coordination based on time-scaled collision cone," *Journal of Guidance, Control, and Dynamics*, pp. 1–8, 2018.

[24] A. S. B. Kumar, A. Modh, M. Babu, B. Gopalakrishnan, and K. M. Krishna, "A novel lane merging framework with probabilistic risk based lane selection using time scaled collision cone," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1406–1411.