2019 International Conference on Robotics and Automation (ICRA)
Palais des congres de Montreal, Montreal, Canada, May 20-24, 2019

Online Vehicle Trajectory Prediction using Policy Anticipation Network and Optimization-based Context Reasoning

Wenchao Ding and Shaojie Shen

***Abstract*— In this paper, we present an online two-level vehicle trajectory prediction framework for urban autonomous driving where there are complex contextual factors, such as lane geometries, road constructions, traffic regulations and moving agents. Our method combines high-level policy anticipation with low-level context reasoning. We leverage a long short-term memory (LSTM) network to anticipate the vehicle's driving policy (e.g., forward, yield, turn left, turn right, etc.) using its sequential history observations. The policy is then used to guide a low-level optimization-based context reasoning process. We show that it is essential to incorporate the prior policy anticipation due to the multimodal nature of the future trajectory. Moreover, contrary to existing regression-based trajectory prediction methods, our optimization-based reasoning process can cope with complex contextual factors. The final output of the two-level reasoning process is a continuous trajectory that automatically adapts to different traffic configurations and accurately predicts future vehicle motions. The performance of the proposed framework is analyzed and validated in an emerging autonomous driving simulation platform (CARLA).**

I. Introduction

In recent years, there has been growing interest in building fully autonomous vehicles. Our requirement of such vehicles is to have accurate anticipation over other traffic participants so that their planned motions are neither too aggressive nor too conservative. To achieve this goal, autonomous vehicles are expected to reason about the behavior and intentions of surrounding vehicles and subsequently predicts future trajectories of these vehicles.

Given an urban driving environment where there are complex latent factors such as lane geometries, traffic regulations, road constructions and dynamical agents, the complexity of the prediction problem is high. Under such a scenario, there are two challenges to be addressed. First, given the complex environment, it is essential to consider the multimodal nature of the future trajectory [1]. For example, at the intersection as depicted in Fig. 1, there are two distinct choices, moving forward and turning left, which result in totally different future trajectories. Second, the prediction method must be highly flexible and able to easily adapt to the complex contextual factors.

Many handcrafted prediction models, such as [2]–[5], may lack flexibility and require refactoring when a new contextual factor is introduced. Meanwhile, other methods, especially the popular RNN-based models [6, 7], treat the trajectory prediction as a pure regression problem in spite of the multimodal nature of the future trajectory. We are therefore motivated to develop a flexible trajectory prediction framework which can easily adapt to various complex urban environments while incorporating high-level intentions to enhance the prediction accuracy.

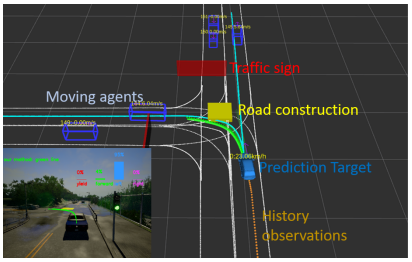


Fig. 1: Illustration of the two-level reasoning methodology at an intersection. The two reference lines corresponding to the two possible policies (turn left or go forward) are shown in *cyan*, and the predicted trajectory is shown in *green*. In this example, the high-level policy is first anticipated (namely, turn left) and the relevant contextual information (lane geometry, construction, other agents) is then used in the optimization-based trajectory prediction. More examples can be found in the video `https://www.youtube.com/watch?v=r-gSUyFoK8Q`.

In this paper, we propose an *online* two-level vehicle trajectory prediction framework. We develop a policy anticipation network using a long short-term memory (LSTM) network to anticipate high-level policies of vehicles (such as moving forward, yielding, turning and lane changing) based on sequential past observations. Given the high-level policy, we propose an optimization-based context reasoning process in which the complex contextual information is naturally encoded in a multi-layer cost map structure. A policy interpreter is set up to bridge the high-level and low-level reasoning by transforming the policy to a trajectory initial guess of the non-linear optimization. The policy anticipation network is used to capture the intention and guide the trajectory prediction process. Our optimization-based context reasoning process can easily adapt to different traffic configurations by transforming different factors into a unified notation of cost.

The motivation for modeling trajectory prediction as an optimization problem is that human drivers internally balance their maneuvers in terms of the "cost". For example, driving through red lights or breaking speed limits would risk receiving penalties, and human drivers have an inborn ability to balance various kinds of costs during driving. The optimization-based reasoning process can be easily extended by adding another cost term to the unified cost map structure.

This work was supported by the Hong Kong PhD Fellowship Scheme (HKPFS). All authors are with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong, China. `wdingae@ust.hk, eeshaojie@ust.hk`

The idea of modeling drivers as optimizing agents is not new [8]–[11], especially in the field of imitating human driving behaviors using inverse reinforcement learning (IRL). However, from the prediction perspective, the multimodal nature of the future trajectory [1, 12] is not well modeled by the optimization process. For example, the non-linear optimization process may converge to either of the two possible intentions in Fig. 1. To this end, we propose the policy anticipation network, which guides the optimization process to the anticipated high-level intention. Note that our optimization-based context reasoning can also incorporate the IRL technique for weight tuning, which is left as important future work.

We summarize the contributions of this paper as follows:

- An online two-level trajectory prediction framework which incorporates the multimodal nature of future trajectories.
- A highly flexible optimization-based context reasoning process which incorporates a multi-layer cost map structure to encode various contextual factors.
- Integration of the vehicle trajectory prediction framework and presentation of the results on accuracy, efficiency, and flexibility in various traffic configurations.

The related literature is reviewed in Sect. II. A system overview is given in Sect. III. The main methodology is presented in Sect. IV and Sect. V. The implementation details and experimental results are provided in Sect. VI and Sect. VII. Conclusions and future work are given in Sect. VIII.

II. Related Works

The problem of vehicle trajectory prediction has been actively studied in the literature. As concluded in [13], there are three levels of prediction models, namely, physics-based, maneuver-based and interaction-aware motion models. Physics-based motion models use dynamic and kinematic vehicle models to propagate future states [14, 15]. However, the prediction results only hold for the very short-term (less than one second). Maneuver-based motion models are more advanced in the sense that the model may forecast relatively complex maneuvers, such as lane change and turns at intersections, by revealing the maneuver pattern. Many of the works on this level present a probabilistic framework to account for the uncertainty and variation of the motion patterns, such as Gaussian processes (GPs) [3, 16], Monte Carlo sampling [17], Gaussian mixture models (GMMs) [5] and hidden Markov models [18]. However, they typically assume vehicles are independent entities and fail to model interactions within the context and with other agents.

Interaction-aware models, on the other hand, take the driving context and vehicle interactions into account, and most of them, such as [4, 19] and [2], are based on dynamic Bayesian networks (DBNs). Though these methods are context-aware, they require refactoring the models when considering a new contextual factor. Our method belongs to the interaction-aware level. Compared to the DBN-based prediction methods, our method is more flexible and can be easily adapted to different traffic configurations.

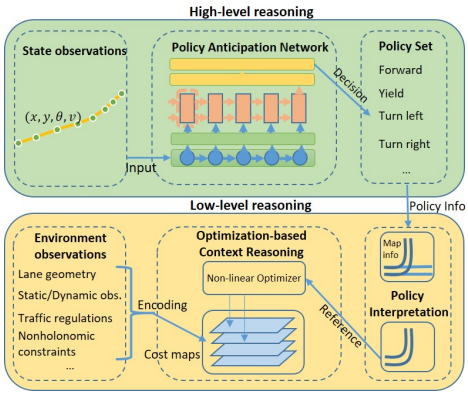


Fig. 2: Illustration of the two-level reasoning framework.

It is notable that recurrent neural networks (RNNs) and their variants, such as LSTM networks, have recently been applied to predict or track moving targets, as in [6, 20] and [21]. Our policy anticipation network shares a similar structure with [20]. But the fundamental difference is that the network in [20] is only used to analyze the maneuver pattern at an intersection and cannot actively predict the future trajectories. Many learning-based end-to-end trajectory prediction models [6, 7, 12] lack the ability to encode the contextual information. In [1], Lee *et al.* suggest combining IRL with an environment feature map to learn the interaction with contextual factors. However, this requires a large amount of training data to generalize due to the high complexity of the model. Also, it is hard to learn the interaction in some rare driving situations, such as red light offences.

III. System Overview

The overview of our vehicle trajectory prediction framework is shown in Fig. 2. During the high-level reasoning, the sequential state observations are fed to the policy anticipation network, which provides the future policy that a vehicle is likely to execute. Together with the map information, the policy can be properly interpreted in the driving context and a reference prediction is generated and fed to the optimization-based context reasoning process. The optimization process renders various environment observations and encodes them into the multi-layer cost map structure. A non-linear optimization process is then conducted to generate the predicted vehicle trajectory.

IV. Policy Anticipation and Interpretation

A. Problem formulation

We assume that the vehicle is equipped with a detector that provides the pose estimation $\mathbf{p}_i^k = (x_i, y_i, \theta_i, v_i)$ of a neighboring vehicle with ID k at different time-instants, where x_i and y_i denote global coordinates at frame i, θ_i denotes the vehicle orientation in the 2-D plane, and v_i denotes the body velocity. We accumulate observations from different time-instants inside a sliding window with a total window size of T_{obs}. And the network predicts vehicle k's

future policy in a look-ahead window from T_{obs} to T_{pred}. The annotated labels include *forward*, *yield*, *turn left*, *turn right*, *lane change left*, and *lane change right*, and the labels can be easily extended when considering complex lane geometries.

B. Network structure

Our policy anticipation network is based on an RNN encoder structure [22]. We refer interested readers to [22] and [1] for the detailed structure. Note that the output layer is modified to a softmax layer to provide the likelihood for all the policy labels. The probability distribution is used in the interpretation of the policy in Sec. IV-C. We adopt negative log-likelihood (NLL) loss for this classification problem.

C. Policy interpretation

The policy interpretation module combines the policy anticipation results with a local map, so that the optimization-based context reasoning can start with a reasonable initial guess. As shown in Fig. 3, with different initial guesses

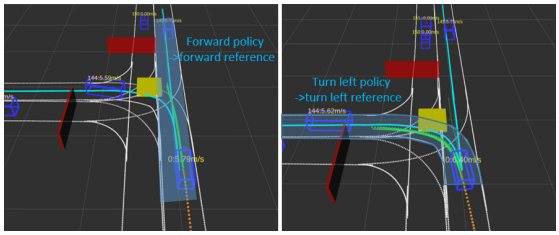


Fig. 3: Illustration of the policy interpretation. The local reference lines extracted from the map are marked in *cyan*. The figure on the left illustrates that when the vehicle (ID 0) has not shown any intention to turn, it is anticipated to be executing a "forward" policy, so the forward reference line is extracted. After the vehicle shows a left-turn pattern, the left-turn reference line is extracted. The local context region is marked in the transparent cyan area.

(turning left or forward in this case), the optimization will be devoted to finding a solution in a totally different local solution space. Specifically, we use the likelihood provided by the policy anticipation network as follows: 1) we prune the infeasible anticipations (turning right in this example); 2) we take the policy of the maximum likelihood, and 3) we generate an initial trajectory prediction by extracting reference points corresponding to the selected policy. The initial guess is fed to the optimization-based context reasoning for further processing. In the future, instead of using deterministic reasoning based on one selected policy, we plan to use a probabilistic interpretation process.

V. Optimization-based Context Reasoning

A. Cost map structure

In this section, we present the cost map structure, which encodes the whole driving context. We specify different kinds of costs by separating them into different layers with distinct physical meanings, for the sake of illustration. A toy example of the multi-layer cost map is given in Fig. 4. We adopt a four-layer cost map design in which we encode the cost induced by the lane geometry and static obstacles into the *static layer*, the cost induced by the moving objectives (MO) into the *MO layer*, the cost induced by traffic regulations into the *context layer*, and the cost induced by the vehicles' nonholonomic constraints into the *nonholonomic layer*.

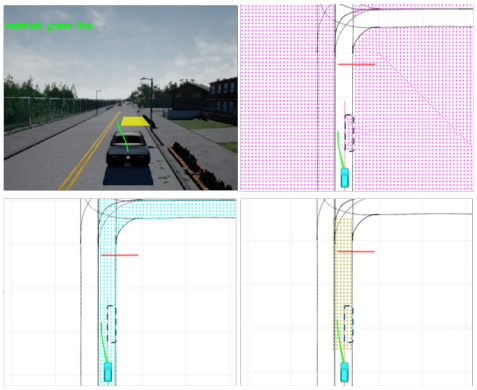


Fig. 4: Illustration of the multi-layer cost map structure. The top-left image is captured from CARLA, and a road construction site is marked in *yellow*. The top-right figure shows the static layer with repulsive forces (cost) pointing to the free space. The bottom-left image illustrates the costs induced by the desired velocity, and the bottom-right image shows the cost induced by the red light. Different forces may be conflicting (as around the dashed box).

B. Cost functions

We adopt a discrete notation of the vehicle trajectory [23] where a continuous trajectory $\mathbf{x}(t) = \big(x(t), y(t)\big)^{\mathrm{T}}$ is represented by a series of rear axle center points $\mathbf{x}_i = (x_i, y_i)^{\mathrm{T}}$ in a global coordinate system. Namely, the predicted trajectory is approximated by N points $\mathbf{x}_i = \mathbf{x}(t_i)$, which are sampled at equidistant times $t_i = t_0 + ih, 0 \leq i < N$ of sampling step width h. The dynamics of the trajectory $\mathbf{x}(t)$ can be expressed as a function of its time derivatives, which are the finite differences of the sampling points. The orientation and curvature of the trajectory can be expressed by its time derivatives [23]. Following these notations, we introduce the cost functions $f(\mathbf{x})$.

1) *Lane geometry*: Ideally, the point $\mathbf{x}$ that exceeds the solid-lane boundary should receive a repulsive force (cost) $f_g(\mathbf{x})$ pointing into the travelable lanes. For broken-lane boundaries, we pose the cost of the same structure, but the magnitude is much smaller to allow for lane changing. We present a bi-directional signed distance field (bi-SDF) to describe the corresponding cost characteristics:

$$f_g(\mathbf{x}) = \begin{cases} \alpha_g(-d_b(\mathbf{x}) + \tau_b)^2 & \text{if } d_b(\mathbf{x}) \leq 0 \\ \alpha_g(d_b(\mathbf{x}) - \tau_b)^2 & \text{if } 0 < d_b(\mathbf{x}) \leq \tau_b \\ 0 & \text{if } \tau < d_b(\mathbf{x}), \end{cases} \quad (1)$$

where $d_b(\mathbf{x})$ measures the distance to the nearest solid-lane boundary, τ_b is the distance threshold, and α_g is the cost magnitude. Note that $d_b(\mathbf{x}) > 0$ means the in-boundary area, while $d_b(\mathbf{x}) < 0$ represents that the point exceeds the boundary and needs to be pushed back to the travelable lanes. Different from the traditional SDF, which does not define the gradient when $d_b(\mathbf{x}) < 0$, we slightly extend the definition

so that the point outside of the boundary will receive a force pointing inside the lane. The benefit of extending $d_b(\mathbf{x}) < 0$ is that the optimization process is less likely to get stuck in the infeasible out-of-boundary area.

2) *Static obstacles/ driveable area*: The cost $f_s(\mathbf{x})$ induced by static obstacles shares a similar form to $f_g(\mathbf{x})$, and these two costs are categorized into the static layer. The distance measure to static obstacles $d_s(\mathbf{x})$ is also extended to allow a negative distance.

3) *Moving obstacles*: To take interaction with other agents into account, we introduce a cost $f_d^j(\mathbf{x}_i)$ for t_i if the position $\mathbf{x}_i$ of the predicted vehicle is within a distance threshold τ_d of the prediction $\mathbf{x}_{\text{pred},i}^j$ of another agent $j \in \mathcal{J}$, where $\mathcal{J}$ denotes the set of all the interacting agents. The practical method of acquiring $\mathbf{x}_{\text{pred},i}^j$ is introduced in Sec. VI-C. The MO cost at time t_i is given by

$$f_d^j(\mathbf{x}_i) = \alpha_d (d_o(\mathbf{x}_i, j) - \tau_o)^2 \mathbb{1}_{d_o(\mathbf{x}_i,j)<\tau_o}, \tag{2}$$

where $f_d^j(\mathbf{x}_i)$ is specified by the quadratic error between the distance $d_o(\mathbf{x}_i, j)$ to the moving agent j and τ_0 if the distance threshold τ_0 is reached.

4) *Red lights*: We argue that red lights should not be enforced as hard constraints since in a real-world driving scenario there exist red light offences. To capture the real intention of other drivers under traffic control, we introduce a red light repulsive force $r(\mathbf{x})$. The repulsive force is supposed to produce larger resistance for vehicles travelling at higher velocity. It is notable that if a vehicle refuses to brake and tries to go through a red light, as shown in Fig. 5, the cost $f_r(\mathbf{x})$ will not dominate the optimization process and the abnormal behavior is captured. The overall cost $f_r(\mathbf{x})$ can be expressed by the dot product of the velocity $\dot{\mathbf{x}}$ and the repulsive force $r(\mathbf{x})$ as follows:

$$\begin{aligned} f_r(\mathbf{x}) &= \left\| \dot{\mathbf{x}} \cdot r(\mathbf{x}) \right\|^2 \\ &= \begin{cases} \alpha_r (d_r(\mathbf{x}) - \tau_r)^2 \left\| \dot{\mathbf{x}} \cdot \hat{\mathbf{r}}(\mathbf{x}) \right\|^2 & \text{if } 0 < d_r(\mathbf{x}) \le \tau_r \\ 0 & \text{if } \tau_r < d_r(\mathbf{x}), \end{cases} \end{aligned} \tag{3}$$

where α_r is the cost magnitude, $\hat{\mathbf{r}}(\mathbf{x})$ denotes the unit direction of the force $r(\mathbf{x})$, $d_r(\mathbf{x})$ denotes the distance to the red light, and τ_r is the distance threshold below which the force $r(\mathbf{x})$ will take effect.

5) *Speed limits*: Like red lights, speed limits should not be encoded in hard constraints when taking speed limit offences into account. We introduce the cost $f_v(\mathbf{x})$, which is induced by the speed limit and should also allow the vehicle to stop in the case of a traffic jam. As a result, we model $f_v(\mathbf{x})$ as the quadratic error between the predicted velocity $\dot{\mathbf{x}}$ and a desired velocity $\mathbf{v}_{\text{des}}$. The magnitude of the desired velocity $\|\mathbf{v}_{\text{des}}\|$ is determined by the minimum between two factors, namely, the speed limit $v_{\max}$ and the velocity trend v_{trend}. Specifically, v_{trend} is obtained by conducting velocity fitting for the historical velocity observations in T_{obs}, which captures the acceleration and deceleration trend of the predicted vehicle and is close to zero in the case of a traffic jam. The direction of the desired velocity $\hat{\mathbf{v}}_{\text{des}}$ conforms to the lane geometry. Mathematically, we have $\|\mathbf{v}_{\text{des}}\| = \min(v_{\max}, v_{\text{trend}})$ and $f_v(\mathbf{x}) = \|\dot{\mathbf{x}} - \mathbf{v}_{\text{des}}\|^2$.

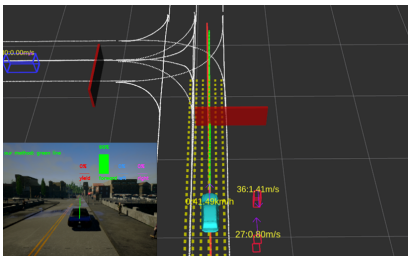


Fig. 5: Illustration of red light offence. When the kinetic energy of the vehicle can overcome the artificial repulsive force induced by the red light, the red light offence is captured and a warning is provided by our prediction system.

6) *Nonholonomic constraints*: The predicted trajectory should obey the limits of the vehicle motion model. Due to the steering geometry of the vehicle, the curvature should be bounded by the maximum curvature allowed. However, when taking abnormal operations, such as skidding, into account, the hard curvature constraint should also be modeled by the feasibility cost $f_\kappa(\mathbf{x})$ as follows:

$$f_\kappa(\mathbf{x}) = \alpha_\kappa (\kappa(\mathbf{x}) - \kappa_{\max})^2 \mathbb{1}_{\kappa(\mathbf{x})>\kappa_{\max}}, \tag{4}$$

where the cost takes effect when the curvature exceeds the limit $\kappa_{\max}$ and α_κ is the cost magnitude. Similarly, due to the friction limit of tires and throttle limit of vehicles, the maximum acceleration of vehicles cannot exceed a limit $a_{\max}$. We model the acceleration feasibility cost $f_a(\mathbf{x})$ as follows:

$$f_a(\mathbf{x}) = \alpha_a (\ddot{\mathbf{x}} - a_{\max})^2 \mathbb{1}_{\ddot{\mathbf{x}}>a_{\max}}, \tag{5}$$

where the maximum acceleration is denoted by $a_{\max}$ and cost magnitude is denoted by α_a.

The motivation for using quadratic functions with barriers for the cost functions is that 1) they tolerate a mild deviation from the best driving practices, and 2) they penalize abnormal behaviors while still allowing their existence.

C. Non-linear optmization

Based on the cost functions, we now introduce the optimization formulation. At a top level, the predicted trajectory is generated by minimizing $J(\mathbf{x}(t))$, which is the integral of the overall loss $L(\mathbf{x}(t))$ over time T, i.e., $J(\mathbf{x}(t)) = \int_{t_0}^{t_0+T} L(\mathbf{x}(t))$, which can be approximated by finite summation in the discrete case as follows:

$$\begin{aligned} J(\mathbf{x}(t)) = \sum_{i=0}^{N-1} \Big(& w_g f_g(\mathbf{x}_i) + w_s f_s(\mathbf{x}_i) + w_d \sum_{j \in \mathcal{J}} f_d^j(\mathbf{x}_i) \\ & + w_r f_r(\mathbf{x}_i) + w_v f_v(\mathbf{x}_i) + w_\kappa f_\kappa(\mathbf{x}_i) + w_a f_a(\mathbf{x}_i) \Big) h. \end{aligned} \tag{6}$$

The weights of different costs represent the tradeoff among different contextual factors. We tune the weights so that predicted trajectories match a human prior for different traffic configurations. As mentioned in Sec. I, the optimization process can incorporate IRL for automatic weight tuning, which is important future work.

VI. Implementation Details

A. Simulation environment

We adopt an open-source urban autonomous driving simulator named CARLA [24]. In this section, we present our environment setup. For a scene containing n vehicles, the first $n-2$ vehicles (agent vehicles) are controlled by the autopilot module provided by CARLA, the $n-1$-th vehicle (player vehicle) is controlled by a human player and the n-th vehicle is an observer vehicle which is supposed to closely follow the player vehicle, sense the environment, and predict the trajectory of the player vehicle. We focus on predicting the trajectories for the player vehicle since it reflects real human intentions. Another reason is that the agent vehicles do not have complex maneuver patterns due to the fixed handwritten logic of the autopilot module. Hence, when presenting the experimental results (Sec. VII) we will focus on illustrating the prediction results for the player vehicle, to give a clean and informative visualization.

B. Data collection and network training

We collect the training data for the policy anticipation network from CARLA by driving the player vehicle ourselves using a Logitech G29 racing wheel. During the driving, we follow the traffic rules most of the time and conduct different maneuver patterns, but we also commit intentional traffic rule offences, as in Fig. 5, to examine how our prediction module will respond. Moreover, we add virtual road construction sites, as in Fig. 4, and respond to them during driving using the feedback from our visualization system. The collected data is $21,260$ frames in total. T_{obs} and T_{pred} are both set to 40 frames (4s). The policy label can be determined by examining the statistics on the steering angle and acceleration in the T_{pred} in an unsupervised way. One problem with the data collected from CARLA is that the current version[1] only includes two-lane roads with traffic moving in opposite directions, which means that lane change behavior cannot be effectively incorporated. In the future, we will collect data from more complex environments to enrich the dataset.

C. Non-linear optimization procedure

The non-linear optimization formulation (6) is implemented in Ceres [25] since the objectives can be rewritten into non-linear least squares. If more complex objectives are involved, non-linear solvers such as NLOPT [26] can be used. The maximum number of iterations is set to 20. Recall that the prediction for a certain vehicle can depend on the prediction of other vehicles due to the moving obstacle cost term $f_d^j(\mathbf{x})$. In practice, we use the prediction results from the last prediction round to calculate $f_d^j(\mathbf{x})$.

VII. Results

A. Prediction accuracy

We adopt the root mean square error (RMSE) between the predicted coordinates and the true coordinates as the error metric. We are concerned with how the RMSE error statistics change with respect to the look-ahead time, especially when the look-ahead time is large. To this end, we plot the mean and variance of the RMSE loss with respect to the look-ahead time, as shown in Fig. 6. We compare our method with the following two methods:

- *Naive fitting method*. The future trajectory is generated using least mean square polynomial regression with an acceleration regulator. This method can capture the trend but cannot incorporate the driving context.
- *RNN encoder-decoder trajectory regression*. This method uses an RNN to encode the past maneuver history and directly outputs the future trajectory through the RNN decoder. This structure is popular, and is adopted in [1] and [12].

Since the source code of [1] is not officially available and [12] is mainly tested in a highway dataset, we adopt the RNN encoder-decoder part in [1] according to the available implementation details [27]. We conduct the experiments in the form of case studies to show that our proposed framework can easily adapt to various traffic configurations, as elaborated in Sec. VII-B.

B. Testing in different traffic configurations

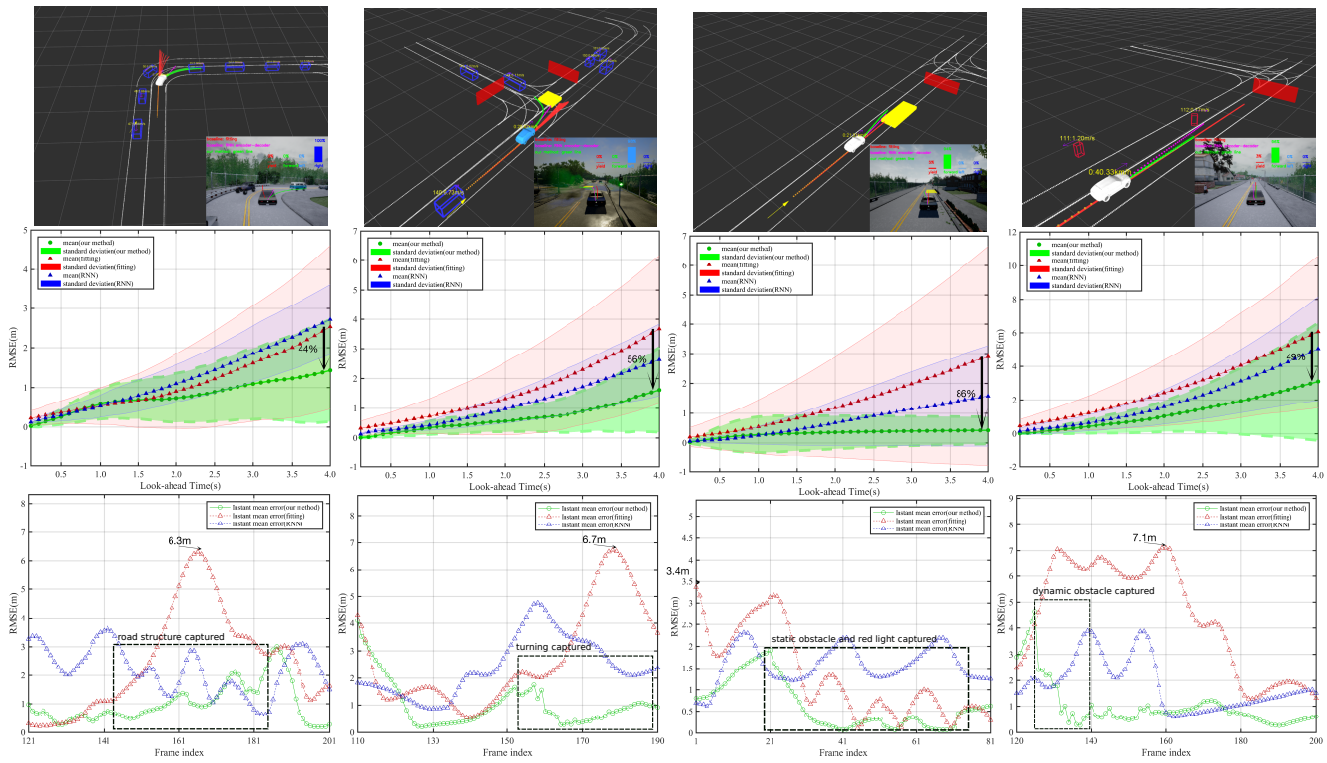
To verify that our proposed method can automatically adapt to different traffic configurations and take various latent factors into account, we design five test cases: driving along a curved road, heading towards a pedestrian who is crossing the road, passing through an intersection with road construction, heading towards a red light with road construction, and committing a red light offence. To give a clean visualization, we focus on the prediction for the vehicle being driven by us, namely, the vehicle with ID 0.

1) *Curved road*: This case is used to verify the capability of reasoning about lane geometries. As illustrated in Fig. 6a, both baseline methods can capture the motion trend. However, because they are unaware of the lane geometries, they take a long time to conform to the shape of the road. On the other hand, our proposed method produces a reasonable prediction immediately. Quantitatively, our method achieves 44% accuracy improvement for the ending frame in T_{pred}. From the instantaneous error statistics, i.e., the average error for the whole predicted trajectory, we observe that the maximum instantaneous error is reduced from 6.3 m to 3 m. This testing case verifies the effectiveness of optimization-based context reasoning.

2) *Intersection with road construction*: This case is used to illustrate the importance of high-level reasoning, which the two baseline methods lack. As shown in Fig. 6b, neither baseline methods can effectively capture the turning left intention and both converge slowly. The benefit of incorporating high-level behavior anticipation is validated by an accuracy gain of 56% for the ending frame and a lower instantaneous error during the intersection entrance. The results verify that it is essential to incorporate the high-level intention.

3) *Red light with road construction*: This case is taken as one example of the non-linear optimization (Fig. 4). The

[1]CARLA release 0.7.0 is used for all the experiments.



(a) Curved road (b) Intersection (c) Traffic control (d) Moving obstacles

Fig. 6: Illustration of the comparsion in different driving scenarios. The top row of figures visualizes the environment together with the image from the virtual sensor. The vehicles are represented by *blue* bounding boxes with their IDs and velocities on the top. The past observations are marked in *orange*, and the prediction results are marked in *green* (ours), *red* (fitting) and *purple* (RNN) for our proposed method and the two baselines, respectively. The middle row of figures shows the error statistics during the entire test cases (the semantics are elaborated in Sec. VII-A). The bottom row of figures illustrates the instantaneous average prediction error over the prediction horizon of each frame in the region of interest (about 4 s around the event).

statistics are provided in Fig.6c, which confirms the necessity of modeling contextual factors.

4) *Heading towards a pedestrian*: This case is used to illustrate the ability to reason about other moving agents. As shown in Fig. 6d, the predicted vehicle is moving at high speed, but a pedestrian is crossing the road ahead of the vehicle. Sudden braking of the vehicle should be the reaction. As shown in Fig. 6d, the two baselines are still giving out forward trajectories, while our method expects hard braking by modeling the interactions between agents. The instantaneous error shows that our method predicts the braking intention beforehand.

5) *Red light offence*: This case is used to show how the proposed method responds to abnormal driving behavior, and is elaborated in Fig. 5.

C. Run-time efficiency

In this section, we test the run-time efficiency. We collect 3886 rounds of predictions and record the time consumption of the three parts of the system, namely, network inference, cost map rendering, and non-linear optimization. The experiment is conducted on a desktop computer equipped with an Intel I7-8700K CPU and an NVIDIA GTX 1080-Ti graphics card for network training and inference.

TABLE I: Run-Time Analysis

# of Predictions	Time(ms)	Network Inference	Cost Map Rendering	Non-linear Optimization	**Total**
3886	Avg	**2.9**	**16.2**	**3.4**	**22.6**
	Std	2.2	3.1	6.0	7.6
	Max	7.2	27.0	40.0	68.5

As we can see from Tab. I, the network inference (on GPU) consumes 2.9 ms on average since the network structure is not complex. It takes an average computing time of 16.2 ms to render a 4-layer 40×40 2-D cost map (CPU implementation). The non-linear optimization is efficient, with an average time consumption of 3.4 ms. In total, our prediction system typically consumes 22.6 ms to complete one round of prediction, and a large part of that time is consumed in the cost map rendering.

VIII. Conclusion and Future Work

In this paper, we propose an online two-level vehicle trajectory prediction framework which utilizes a policy anticipation network for high-level policy reasoning and a non-linear optimization process for low-level context reasoning. We highlight the flexibility of the proposed framework, and provide various test cases, including normal operations and abnormal driving behavior, in urban environments. In the future, we will explore using IRL [9] to acquire the weights from data. Modeling interaction in prediction is another direction we are actively exploring [28].

REFERENCES

[1] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker, "Desire: Distant future prediction in dynamic scenes with interacting agents," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 336–345.

[2] G. Agamennoni, J. I. Nieto, and E. M. Nebot, "Estimation of multivehicle dynamics by considering contextual information," *IEEE Trans. Robot.*, vol. 28, no. 4, pp. 855–870, 2012.

[3] C. Laugier, I. E. Paromtchik, M. Perrollaz, M. Yong, J.-D. Yoder, C. Tay, K. Mekhnacha, and A. Nègre, "Probabilistic analysis of dynamic scenes and collision risks assessment to improve driving safety," *IEEE Intell. Trans. Syst. Mag.*, vol. 3, no. 4, pp. 4–19, 2011.

[4] S. Lefèvre, C. Laugier, and J. Ibañez-Guzmán, "Evaluating risk at road intersections by detecting conflicting intentions," in *Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Syst.* IEEE, 2012, pp. 4841–4846.

[5] F. Havlak and M. Campbell, "Discrete and continuous, probabilistic anticipation for autonomous robots in urban environments," *IEEE Trans. Robot.*, vol. 30, no. 2, pp. 461–474, 2014.

[6] B. Kim, C. M. Kang, S. H. Lee, H. Chae, J. Kim, C. C. Chung, and J. W. Choi, "Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network," *arXiv preprint arXiv:1704.07049*, 2017.

[7] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 961–971.

[8] M. T. Wolf and J. W. Burdick, "Artificial potential functions for highway driving with collision avoidance," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. IEEE, 2008, pp. 3731–3736.

[9] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 1.

[10] M. Bahram, C. Hubmann, A. Lawitzky, M. Aeberhard, and D. Wollherr, "A combined model-and learning-based framework for interaction-aware maneuver prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 6, pp. 1538–1550, 2016.

[11] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for autonomous cars that leverage effects on human actions." in *Robotics: Science and Systems*, 2016.

[12] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," *arXiv preprint arXiv:1805.06771*, 2018.

[13] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *Robomech Journal*, vol. 1, no. 1, p. 1, 2014.

[14] S. Ammoun and F. Nashashibi, "Real time trajectory prediction for collision risk estimation between vehicles," in *Proc. of the IEEE Intl. Conf. on Intell. Comp. Comm. and Processing*. IEEE, 2009, pp. 417–422.

[15] M. Brannstrom, E. Coelingh, and J. Sjoberg, "Model-based threat assessment for avoiding arbitrary vehicle collisions," *IEEE Trans. on Intell. Trans. Syst.*, vol. 11, no. 3, pp. 658–669, 2010.

[16] Q. Tran and J. Firl, "Online maneuver recognition and multimodal trajectory prediction for intersection assistance using non-parametric regression," in *Proc. of the IEEE Intl. Veh. Sym.* IEEE, 2014, pp. 918–923.

[17] A. Eidehall and L. Petersson, "Statistical threat assessment for general road scenes using Monte Carlo sampling," *IEEE Trans. on Intell. Trans. Syst.*, vol. 9, no. 1, pp. 137–147, 2008.

[18] G. S. Aoude, V. R. Desaraju, L. H. Stephens, and J. P. How, "Driver behavior classification at intersections and validation on large naturalistic data set," *IEEE Trans. on Intell. Trans. Syst.*, vol. 13, no. 2, pp. 724–736, 2012.

[19] T. Gindele, S. Brechtel, and R. Dillmann, "Learning driver behavior models from traffic observations for decision making and planning," *IEEE Intell. Trans. Syst. Mag.*, vol. 7, no. 1, pp. 69–79, 2015.

[20] A. Khosroshahi, E. Ohn-Bar, and M. M. Trivedi, "Surround vehicles trajectory analysis with recurrent neural networks," in *Proc. of the IEEE Intl. Conf. on Intell. Trans. Syst.* IEEE, 2016, pp. 2267–2272.

[21] P. Ondruska and I. Posner, "Deep tracking: Seeing beyond seeing using recurrent neural networks," *arXiv preprint arXiv:1602.00991*, 2016.

[22] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[23] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for berthaa local, continuous method," in *Proc. of the IEEE Intl. Veh. Sym.* IEEE, 2014, pp. 450–457.

[24] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

[25] S. Agarwal, K. Mierle, and Others, "Ceres solver," http://ceres-solver.org.

[26] S. G. Johnson, *The NLopt nonlinear-optimization package*, 2011. [Online]. Available: http://ab-initio.mit.edu/nlopt

[27] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker, "Desire: Distant future prediction in dynamic scenes with interacting agents, supplementary material." http://www.robots.ox.ac.uk/~namhoon/doc/DESIRE-supp.pdf.

[28] W. Ding, J. Chen, and S. Shen, "Predicting vehicle behaviors over an extended horizon using behavior interaction network," in *Proc. of the IEEE Intl. Conf. on Robot. and Autom.* IEEE, 2019.