2021 IEEE Intelligent Transportation Systems Conference (ITSC)
Indianapolis, USA. September 19-21, 2021

Learn collision-free self-driving skills at urban intersections with model-based reinforcement learning

Yang Guan#1, Yangang Ren#1, Haitong Ma1, Shengbo Eben Li*1, Qi Sun1, Yifan Dai2, Bo Cheng1

2021 IEEE International Intelligent Transportation Systems Conference (ITSC) | 978-1-7281-9142-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/ITSC48978.2021.9564880

Abstract— **Intersection is one of the most complex and accident-prone urban traffic scenarios for autonomous driving wherein making safe and computationally efficient decisions with high-density traffic flow is usually non-trivial. Current rule-based methods decompose the decision-making task into several serial sub-modules, resulting in long computation time at complex scenarios for on-board computing devices. In this paper, we formulate the decision-making and control problem under intersections as a process of optimal path selection and tracking, where the former selects a path with the best safety measure from a set generated only considering static information, while the latter then considers dynamic obstacles and solve a tracking problem with safety constraints using the chosen path. To avoid the heavy computation introduced by that, we develop a reinforcement learning algorithm called generalized exterior point (GEP) to find a neural network (NN) solution offline. It first constructs a multi-task problem involving all the candidate paths and transforms it into an unconstrained problem with a penalty on safety violations. Afterward, the approximate feasible optimal control policy is obtained by alternatively performing gradient descent and enlarging the penalty. As an exterior point type method, GEP permits control policy to violate inequality constraints during the iterations. To verify the effectiveness of our method, we carried out experiments both in simulation and in a real road test. Results demonstrate that the learned policy can realize collision-free driving under different traffic conditions while reducing the computation time by a large margin.**

Index Terms— **Intersection, decision-making, reinforcement learning, penalty function**

I. INTRODUCTION

Autonomous driving has great potential to benefit traffic safety and efficiency as well as change the pattern of future transportation. Among multifarious urban scenarios in autonomous driving, intersection has been regarded as the most challenging one due to the highly dynamic, stochastic and sophisticated nature of the traffic environment [1]. At intersections, the complex conflicting relationship between vehicles results in complicated vehicle operations to avoid collision and maintain high computing efficiency. Statistics have found that intersection accidents account for 30–60% of severe traffic injuries in Europe. Therefore, developing safe and computationally efficient decision-making strategies at intersections becomes vital for popularization of autonomous driving.

Currently, most of decision-making algorithms at intersections can be categorized into two major paradigms: rule-based method and learning-based method [2]. The former usually decomposes decision-making module into a series of sub-modules such as prediction, behavior selection and trajectory planning, whereas the latter learns a policy mapping from output of perception to control signal directly using advanced learning techniques [3], [4]. Under the rule-based scheme, prediction module aims to predict the trajectory of traffic participants to construct the feasible region. Recently, recurrent neural network makes an increasingly important role in traffic prediction because of its high representation and approximation capability for long time series [5]. Behavior selection module decides on a local driving task that drives the car towards the destination and abides by rules of the road. Traditionally, finite state machine is widely adopted to transfer between priorly defined driving behaviors in terms of current vehicle situation [6]. Trajectory planning module selects a continuous path through the environment to accomplish a local navigational task [7]–[10]. Commonly, this process is time-consuming to search one feasible and safe trajectory, which would be exacerbated by inadequate computing capability of industrial computers. The state-of-the-art result of average planning time is 31ms using a directed acyclic graph search and improved A* algorithm [11]. However, the time tends to increase largely with the growing number of surrounding vehicles. To date, rule-based framework has been widely attempted to handle some typical urban scenarios like signalized junctions and pedestrian crossings. However, much human design and high computation complexity restrict its scalability to the complex scenarios with large-scale vehicles.

Learning-based method aims to train one driving policy usually carried by a deep neural network which maps from perception information to control commands, thus this is also referred as "end-to-end" scheme [12]. Inspired by recent success of reinforcement learning (RL) on games and robotics [13], [14], some researchers have attempted to utilize RL paradigms to conquer decision-making at complicated intersections. The advantage lies in its self-learning ability and high online computation efficiency due to the fast propagation of neural networks (NN). As a pioneering work for decision-making at intersections, Intel laboratory introduced the asynchronous advantage actor-critic (A3C) algorithm [15] to develop driving policies based on urban

This work is supported by the National Key Research and Development Plan of China under Grant 2017YFB0102603. It is supported in part by International Science & Technology Cooperation Program of China under 2019YFE0100200, NSF China with 51575293, and U20A20334. It is also partially supported by Tsinghua University-Toyota Joint Research Center for AI Technology of Automated Vehicle.

#The first two authors contributed equally to this study. 1School of Vehicle and Mobility, Tsinghua University, Beijing, 100084, China. 2Suzhou Automotive Research Institute, Tsinghua University, Suzhou, 215200, China. All correspondence should be sent to S. E. Li with email: lisb04@gmail.com.

road simulator in the presence of other vehicles and pedestrians [16]. They concluded that RL-enabled methods was not enough successful at avoiding collisions with cars and static objects compared with decomposed and supervised methods. Guan *et al.* (2020) developed a decision-making framework for cooperation of eight connected automated vehicles at unsignalized intersection [17]. They adopted Proximal Policy Optimization (PPO) algorithm and obtained the final converged policy after days of training, which is quite slow to obtain a fair performance. Li *et al.* (2020) established an end-to-end framework using convolutional neural networks to map relationship between traffic images and vehicle operations. Relative velocity between vehicles was calculated by traffic images collected at two consecutive time steps and the famous deep Q-network algorithm [18] was utilized to obtain the optimal driving policy regarding safety and efficiency [19]. However, only two vehicles from different directions are considered. A shortcoming of aforementioned works is that the intersection traffic scenarios they studied are rather simple with only a few surrounding vehicles, which suffers a great discrepancy with the real intersection condition. Besides, although current algorithms have been developed and work well on sorts of simulators like CARLA [16] and SUMO [20], the driving policy is in fact trained without safety guarantee, making it impractical to be applied on real world intersections.

In this paper, we propose an integrated decision and control (IDC) framework for automated vehicles at complex intersections, which combines prior knowledge and RL technique to make safe and computationally efficient decisions. Besides, as an important support for IDC, we develop a model-based RL algorithm called generalized exterior point (GEP) method to obtain NN solutions of large-scale constrained optimal control problems (OCP). The main contributions of the paper can be summarised as follows:

(1) Present the IDC framework for automated vehicles at intersections. The IDC framework decomposes the driving task into optimal path selection and tracking, where the former chooses a path with the best safety measure from a path set generated in advance, while the latter then solves a tracking problem with safety constraints to realize collision-free driving. To mitigate the online computation burden, the path set is formed only considering static information, and GEP is developed to solve a control policy of the tracking problem offline. The framework enhances the computing efficiency by removing intensive optimizations using RL technique and exhibits complex driving behaviors by selecting and tracking among different paths. Moreover, it owns great adaptability and is promising to be deployed on different kinds of scenarios with a variety of traffic participants.

(2) Propose a model-based RL algorithm called GEP to approximately solve a NN control policy for the constrained tracking problems to support the real-time online application. The GEP is in fact an extension of the exterior point method in optimization domain to the field of NN, for the propose of obtaining NN solutions of large-scale constrained OCPs. It first constructs a multi-task problem involving all the candidate paths and transforms it into an unconstrained problem with a penalty on safety violations. Afterwards, the approximate feasible optimal control policy is obtained by alternatively performing gradient descent and enlarging the penalty. To the best of our knowledge, GEP is the first approximate solver for constrained OCPs that are parameterized by NNs.

II. Integrated decision and control framework

Here we demonstrate the integrated decision and control framework at intersections. As shown in Fig.1, the framework consists of four modules: path generating, training, path selecting and optimal tracking, of which the first two mainly involve offline training and another two are related to the online application. Note that the training module design is the core segment under this framework, which is prone to substitute all the expensive online optimizations with an offline trained policy network based on RL technique.

The path generating module will produce multiple candidate paths only considering static information such as road structure, speed limit and traffic lights. Note that the intention is to generate one path set rather than find the optimal path, by which the planning time will be reduced largely. Then the training module will further considers these paths and the dynamic information such as surrounding vehicles, pedestrians and bicycles. Formally, an constrained OCP is constructed concerning each path, where the objective function is to minimize the tracking error within a finite horizon and the constraints characterize safety requirements. Instead of solving these OCPs in sequence, we develop a model-based RL algorithm to solve a control policy parameterized by NN which is capable of tracking different shape of paths while maintaining high flexibility to avoid collision with surrounding vehicles. Subsequently, in the online application, the ego vehicle choose a relatively better path which is prone to reduce potential collision and passing time using simple rules. For this given path, the trained policy will quickly calculate the driving actions by the feed-forward propagation of NN, leading the ego vehicle to the next location.

Compared to traditional methods, the framework owns several advantages to be applied in intersections with large-scale traffic flow. Above all, it has high online computing efficiency benefiting from two improvements. One is that path planning module can embed key parameters of multiple paths priorly into the electronic map and read directly to generate candidate paths. The other is that tracking module utilizes the trained policy network to compute control commands, which also enjoys high efficiency due to the fast propagation of neural networks. Second, our algorithm is flexible enough to deal with the dynamic and complexity at intersections. Generally, the more complex the intersection, the more candidate paths there are. Fortunately, the design of our algorithm guarantees the trained policy can handle various paths at complex intersections. Besides, the formulation of constrained optimal problem is applicable for different kinds

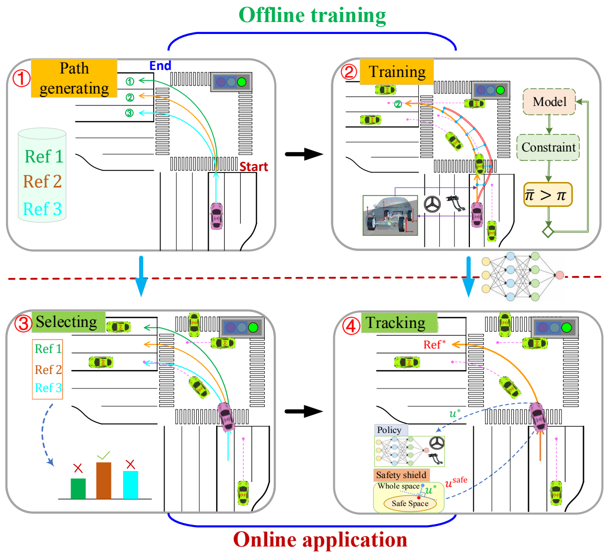


Fig. 1: Integrated decision and control framework under intersections

of dynamic traffic participants including red lights, bicycles and pedestrians.

III. Methodology

In this section, we mainly focus on the details of the training module, the central task, including the formulation of the path tracking problem and the proposed algorithm GEP. Besides, the details of the online application will also be involved.

A. Problem formulation

Provided one possible path $\tau \in \Pi$ where Π is the candidate path set, the constrained OCP can be constructed as (1), aiming to optimize tracking performance while satisfying the safety constraints.

$$
\begin{aligned}
\min_{u_t} \quad & J = \sum_{i=0}^{T-1} (x^{\text{ref}}_{i|t} - x_{i|t})^\top Q (x^{\text{ref}}_{i|t} - x_{i|t}) + u_{i|t}^\top R u_{i|t} \\
\text{s.t.} \quad & x_{i+1|t} = F_{\text{ego}}(x_{i|t}, u_{i|t}), \\
& x^j_{i+1|t} = F_{\text{pred}}(x^j_{i|t}), \\
& (x_{i|t} - x^j_{i|t})^\top M (x_{i|t} - x^j_{i|t}) \geq D^{\text{safe}}_{\text{veh}}, \\
& (x_{i|t} - x^{\text{road}}_{i|t})^\top M (x_{i|t} - x^{\text{road}}_{i|t}) \geq D^{\text{safe}}_{\text{road}}, \\
& x_{i|t} \leq L_{\text{stop}}, \text{if light} = \text{red} \\
& x_{0|t} = x_t, x^j_{0|t} = x^j_t, u_{0|t} = u_t \\
& i = 0 : T-1, j \in I
\end{aligned}
\tag{1}
$$

where T is the prediction horizon and i is the virtual time step starting from the current time step t, $x_{i|t}$ and $x^{\text{ref}}_{i|t}$ respectively represent the ego vehicle state and its corresponding reference state at predictive time step i, whose deviation is usually weighted by a positive-definite matrices Q. $u_{i|t}$ is control action to be solved and R is the weighting matrix. $x^j_{i|t}$ is the state of the j-th vehicle in the interested vehicle set I. F_{ego} represents the dynamic model of ego vehicle and F_{pred} is the kinematics model of surrounding vehicles. In addition, $D^{\text{safe}}_{\text{veh}}$ and $D^{\text{safe}}_{\text{road}}$ define the safe distance considering other vehicles and the road edge, and M is the distance weighted matrix. L_{stop} indicates the position of stop line and defines the constraints regarding red lights. Note that in (1) the virtual states can be generated by the dynamics and the kinematics model except that $x_{0|t}$ and $x^j_{i|t}$ are assigned with the current real state x_t and x^j_t.

As for the dynamics of ego vehicle, here we adopt the classic 2-DOF bicycle model widely adopted in vehicle control, which owns a 6-dimensional state vector and a 2-dimensional action

$$
x_{i|t} = \begin{bmatrix} p_{\text{x}} & p_{\text{y}} & v_{\text{lon}} & v_{\text{lat}} & \phi & \omega \end{bmatrix}^T_{i|t}, \quad u_{i|t} = \begin{bmatrix} \delta & a \end{bmatrix}^T_{i|t},
$$

where $p_{\text{x}}, p_{\text{y}}$ are the position coordinates of the ego vehicle center of gravity (CG), $v_{\text{lon}}, v_{\text{lat}}$ are the longitudinal and lateral velocities, ϕ is the heading angle, ω is the yaw rate, δ and a are the front wheel angle and the acceleration commands, respectively. And the state space equation is

$$
F_{\text{ego}} = \begin{bmatrix}
p_{\text{x}} + \Delta t (v_{\text{lon}} \cos\phi - v_{\text{lat}} \sin\phi) \\
p_{\text{y}} + \Delta t (v_{\text{lon}} \sin\phi + v_{\text{lat}} \cos\phi) \\
v_{\text{lon}} + \Delta t (a + v_{\text{lat}} \omega) \\
\frac{m v_{\text{lon}} v_{\text{lat}} + \Delta t [(L_f k_f - L_r k_r)\omega - k_f \delta v_{\text{lon}} - m v_{\text{lon}}^2 \omega]}{m v_{\text{lon}} - \Delta t (k_f + k_r)} \\
\phi + \Delta t \omega \\
\frac{-I_z \omega v_{\text{lon}} - \Delta t [(L_f k_f - L_r k_r) v_{\text{y}} - L_f k_f \delta v_{\text{lon}}]}{\Delta t (L_f^2 k_f + L_r^2 k_r) - I_z v_{\text{lon}}}
\end{bmatrix}.
\tag{2}
$$

Specially, throughout this paper, the key parameters of ego vehicle are well-designed in accordance with the physical vehicle we use in real car test, as shown in TABLE I. Considering the surrounding vehicle only involves safety

TABLE I: Parameters for F_{ego}

Parameter	Meaning	Value
k_f	Front wheel cornering stiffness	-155495 [N/rad]
k_r	Rear wheel cornering stiffness	-155495 [N/rad]
L_f	Distance from CG to front axle	1.19 [m]
L_r	Distance from CG to rear axle	1.46 [m]
m	Mass	1520 [kg]
I_z	Polar moment of inertia at CG	2642 [kg·m^2]
Δt	Discrete time	0.1 [s]

requirements, we simply construct a kinematics model to prediction its motion

$$
F_{\text{pred}} = \begin{bmatrix}
p^j_{\text{x}} + \Delta t (v^j_{\text{lon}} \cos\phi^j - v^j_{\text{lat}} \sin\phi^j) \\
p^j_{\text{y}} + \Delta t (v^j_{\text{lon}} \sin\phi^j + v^j_{\text{lat}} \cos\phi^j) \\
v^j_{\text{lon}} \\
0 \\
\phi^j + \Delta t \omega^j_{\text{pred}} \\
0
\end{bmatrix}
\tag{3}
$$

in which ω^j_{pred} depends on the route and position of the j-th vehicle. Besides, the states of surrounding vehicles, reference path and road are represented with a deduced state vector respectively:

$$x_{i|t}^{\dagger} = \begin{bmatrix} p_{\mathrm{x}}^{\dagger} & p_{\mathrm{y}}^{\dagger} & v_{\mathrm{lon}}^{\dagger} & 0 & \phi^{\dagger} & 0 \end{bmatrix}_{i|t}^{T}.$$

where $\dagger \in \{j, \mathrm{ref}, \mathrm{road}\}$.

B. Generalized exterior point method (GEP)

The OCP (1) can be solved by traditional online optimization methods like model predictive control (MPC), but this would lead to computational intractability for the on-board device especially in the presence of large-scale constraints. Therefore, we propose a model-based RL algorithm, which is utilized to find a neural network solution for the constrained OCP in offline with the model guidance, facilitating the online computing efficiency and overall performance. However, the problem (1) has limited generalization because it is defined on a particular path and seeks to find a single control under a single state.

So, firstly and significantly, our algorithm convert the original problem (1) to the following one:

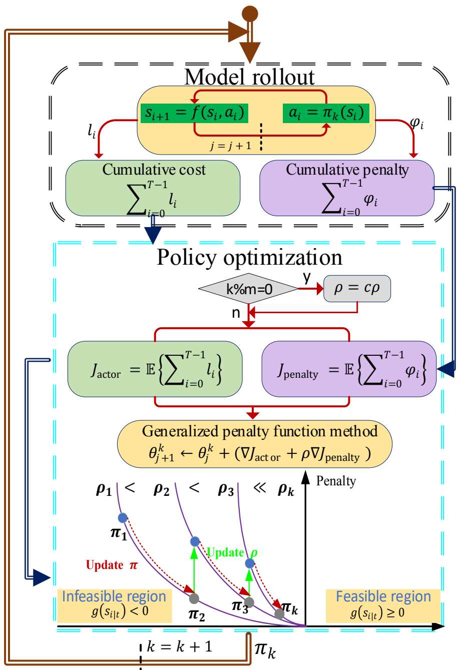
$$\begin{aligned}
\min_{\theta} \quad & \mathbb{E}_{\tau, x_t, x_t^j}\Bigg\{\sum_{i=0}^{T-1}(x_{i|t}^{\mathrm{ref}} - x_{i|t})^{\top} Q (x_{i|t}^{\mathrm{ref}} - x_{i|t}) + \\
& \qquad \pi_{\theta}^{\top}(s_{i|t}) R \pi_{\theta}(s_{i|t})\Bigg\} \\
\text{s.t.} \quad & x_{i+1|t} = F_{\mathrm{ego}}(x_{i|t}, \pi_{\theta}(s_{i|t})), \\
& x_{i+1|t}^{j} = F_{\mathrm{pred}}(x_{i|t}^{j}), \\
& g(s_{i|t}) = \begin{cases} (x_{i|t} - x_{i|t}^{j})^{\top} M (x_{i|t} - x_{i|t}^{j}) \geq D_{\mathrm{veh}}^{\mathrm{safe}}, \\ (x_{i|t} - x_{i|t}^{\mathrm{road}})^{\top} M (x_{i|t} - x_{i|t}^{\mathrm{road}}) \geq D_{\mathrm{road}}^{\mathrm{safe}}, \\ -x_{i|t} \geq -L_{\mathrm{stop}}, \text{if light} = \text{red}, \end{cases} \\
& s_{i|t} = [x_{i|t}^{\mathrm{ref}}, \quad x_{i|t}, \quad x_{i|t}^{j}]^{T}, \\
& i = 0 : T-1, j \in I
\end{aligned} \tag{4}$$

The first distinction between (1) and (4) is the variable to be optimized is no longer a single control quantity but the parameters of a policy π_θ defined as a continuous function mapping from state space to action space, i.e. $u_t = \pi_\theta(s_t)$, where s is a notation of the input of the policy and is designed to contain all necessary information to determine driving actions, including that of the reference path, the ego vehicle and surrounding vehicles, as shown in (4). With the policy we can attain the optimal control actions for every state. Second, the objective function is no longer about a single state s, that is, a particular pack of ego state, vehicle states and reference path, but about a joint distribution of these components. Thus it brings another great advantage that one policy is enough to handle tracking tasks for different paths, avoiding training a policy for each path. That is why we also incorporate the path information as a part of state s. During training, the reference path will be sampled uniformly from the path set to generate a series of states such that our policy can experience different paths information.

For the large-scale constrained problem (4), we subsequently propose GEP, which is adapted from the one introduced in optimization field, wherein the constrained problem is converted into an unconstrained one. For clarity, we use $l(s_{i|t}, \pi_\theta(s_{i|t}))$ to denote one-step cost in (4) and use $g(s_{i|t}) \geq 0$ as the unified denotation of constraints in (4). Then the unconstrained problem can be derived as

$$\begin{aligned}
\min_{\theta} J_p &= J_{\mathrm{actor}} + \rho J_{\mathrm{penalty}} \\
&= \mathbb{E}_{s_{0|t}}\left\{\sum_{i=0}^{T-1} l(s_{i|t}, \pi_\theta(s_{i|t}))\right\} + \rho \mathbb{E}_{s_{0|t}}\left\{\sum_{i=0}^{T-1}\varphi_i(\theta)\right\}
\end{aligned} \tag{5}$$

where $\varphi_i(\theta) = [\max\{0, -g(s_{i|t})\}]^2$ is the penalty function and ρ is the penalty factor. The illustration of GEP for constrained optimal control is shown in Fig. 2. The optimization process mainly involves alternative update of policy parameters and the penalty factor. The former can be updated by gradient descent methods while the latter is designed as a given monotonic increasing sequence. Intuitively, this factor aims to penalize the violation of constraint, which will become prohibitive with a fair large penalty factor and push the policy to the feasible region progressively. This algorithm is shown in the offline training part of Algorithm 1.



Model rollout
$s_{i+1} = f(s_i, a_i)$
$a_i = \pi_k(s_i)$
$j = j + 1$
l_i
φ_i
Cumulative cost $\sum_{i=0}^{T-1} l_i$
Cumulative penalty $\sum_{i=0}^{T-1} \varphi_i$
Policy optimization
k%m=0
y
n
$\rho = c\rho$
$J_{\mathrm{actor}} = \mathbb{E}\{\sum_{i=0}^{T-1} l_i\}$
$J_{\mathrm{penalty}} = \mathbb{E}\{\sum_{i=0}^{T-1} \varphi_i\}$
Generalized penalty function method
$\theta_{j+1}^k \leftarrow \theta_j^k + (\nabla J_{\mathrm{actor}} + \rho \nabla J_{\mathrm{penalty}})$
$\rho_1 < \rho_2 < \rho_3 \ll \rho_k$
Penalty
π_1
π_2
π_3
π_k
Update π
Update ρ
Infeasible region $g(s_{i|t}) < 0$
Feasible region $g(s_{i|t}) \geq 0$
$k = k + 1$
π_k

Fig. 2: Diagram of the generalized exterior point method

C. Online application

Now that we have obtained the optimal control policy after offline training, the next step is to deploy it online. In each time step we first choose the optimal path from the candidate set Π in terms of a rule-based safety indicator shown as below,

$$V(\tau) = \sum_{j \in I} d(\tau, x^j)$$

where $d(\tau, x^j)$ means distance of the j-th surrounding vehicle to the path τ. Then the optimal path is the one with maximal V, i.e.,

$$\tau^* = \arg\max_{\tau_i}\{V(\tau_i)|\tau_i \in \Pi\}. \tag{6}$$

After that, we establish the state s_t using the optimal path τ^* and the trained policy π_θ will tell corresponding driving actions u_t^*. Moreover, we further consider the application safety issue caused by approximation technique. Actually, the trained policy has no guarantee on the safety in a given point of the state space for the reason that we rely on a sample-based method to approximately solve this problem and the whole state space cannot be swept during training. To tackle this, a safety shield is further added before employing the u_t^* to vehicle, which projected the unsafe action to the nearest safe one u_t^{safe}:

$$u_t^{\text{safe}} = \begin{cases} u_t^*, & \text{if} \quad u_t^* \in \mathcal{U}_{\text{safe}}(s_t) \\ \arg\min\limits_{u \in \mathcal{U}_{\text{safe}}(s_t)} ||u - u_t^*||^2, & \text{else} \end{cases} \tag{7}$$

where $\mathcal{U}_{\text{safe}}(s_t) = \{u_t|g(s_{1|t}) \geq 0\}$ guarantees the next state $s_{1|t}$ is safe. In practical, $s_{1|t}$ is replaced by model prediction to judge whether an action is safe. Finally, the safety action will be acted on the ego. The integrated decision and control framework is shown in Algorithm 1.

Algorithm 1: IDC framework

```
Input: Path set $\Pi$, policy network $\pi_\theta$ with random
       parameters $\theta$, buffer $\mathcal{B} \leftarrow \emptyset$, learning rate $\beta_\theta$,
       penalty factor $\rho = 1$, penalty amplifier $c$
Offline training
    for each iteration $k$ do
        Randomly select a path $\tau \in \Pi$, initialize ego
          state $x_t$ and vehicle states $x_t^j, j \in I$;
        for each environment step do
            $s_t \leftarrow \{\tau, x_t, x_t^j, j \in I\}$;
            $\mathcal{B} \cup \{s_t\},\ a_t = \pi_\theta(s_t)$;
            Apply $a_t$ to observe $x_{t+1}$ and $x_{t+1}^j, j \in I$
        Sample a batch of states from $\mathcal{B}$ and for each
          state, predict $J_p$ in (5);
        if $k \mod m$ then
            $\rho \leftarrow c\rho$;
        $\theta \leftarrow \theta + \beta_\theta \nabla_\theta J_p$
    Obtain the optimal policy $\pi_\theta^*$
Online application
    for each environment step do
        for each $\tau_i \in \Pi$ do
            $V(\tau_i) = \sum_{j \in I} d(\tau_i, x^j)$;
        $\tau^* = \arg\max_{\tau_i}\{V(\tau_i)|\tau_i \in \Pi\}$;
        $s_t \leftarrow \{\tau^*, x_t, x_t^j, j \in I\},\ u_t^* = \pi_\theta^*(s_t)$;
        Calculate $u_t^{\text{safe}}$ by (7);
        Apply $u_t^{\text{safe}}$ to observe $x_{t+1}$ and $x_{t+1}^j, j \in I$
```

IV. Simulation verification

Here we focus on a typical 4-direction intersection, shown as Fig. 3, which is signalized and each direction has three lanes. The controlled ego vehicle aims to travel through this intersection safely and efficiently with a dense traffic flow over 800 vehicles per hour on each lane of the entrance. There are three tasks in total including left-turn, right-turn and straight-go. The simulated videos are available online (https://youtu.be/MEm2ZCi2iao).

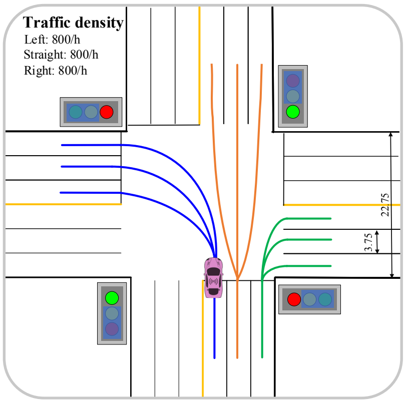


Fig. 3: Intersection and candidate paths generation

A. Algorithm Details and Training Results

The state s is constructed as a 41-d vector composed of a 6-d ego state described in section III, a 4-d vector for each of 8 vehicles who have potential conflicts with ego vehicle and a 3-d vector for the reference path including the tracking error of position, speed and heading. The weighting matrices are $Q = diag(0.04, 0.04, 0.01, 0.01, 0.1, 0.02)$, $R = diag(0.1, 0.005)$ and the predictive horizon T is set to be 25 with a frequency of 10Hz. For the training parameters, the policy network is a multi-layer perceptron with two hidden layer, each with 256 units. The update interval m of penalty factor is 10000 and the amplifier is 1.1. Besides, the batch size is chosen as 1024 and a decayed learning rate $3e-4 \rightarrow 1e-5$ is employed. Training curves are shown in Fig. 4, in which both policy loss and penalty loss decrease consistently for all the tasks, indicating an improving tracking and safety performance. Specially, penalty loss decreases nearly to 0, which shows the proposed algorithm GEP eventually find the policy to satisfy safety constraints approximately.

B. Performance of the GEP

To verify the control precision and computing efficiency of our model-based solver on constrained optimal problem, we conduct comparison with MPC. Here we adopt the Ipopt solver [21], an open and popular package to solve nonlinear optimization problem, to obtain the exact solution of the constrained optimal control problem, which is then used as the benchmark for our solver. Fig. 5 demonstrates the comparison of our algorithm on control effect and computation time. Results show the output actions, steer wheel and acceleration, have similar trends, which indicates our proposed algorithm can approximate the optimal solution with small error. Besides, there exists obvious difference in computation time that our method can output the actions

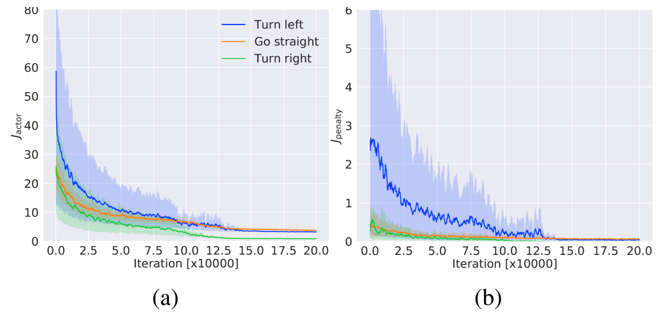


Fig. 4: Training process of model-based RL. (a) Tracking performance. (b) Safety performance. The solid lines correspond to mean value and the shaded regions correspond to 95% confidence interval over 5 runs.

within 10ms while MPC will take 1000ms to perform that, demonstrating the superiority of the proposed offline solver.

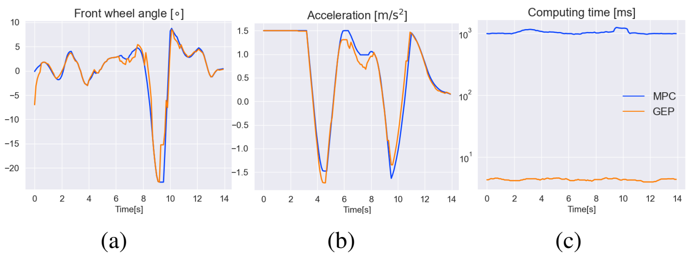


Fig. 5: Comparison with MPC. (a) Front wheel angle (b) Acceleration (c) Computing time

C. Driving performance comparison

We compare our method with a rule-based method which adopts A* algorithm to generate a feasible trajectory and a PID controller to track it [11], as well as a model-free RL method which uses a punish and reward system to learn a policy for maximizing the long-term reward (If the ego vehicle passes the intersection safely, a large positive reward 100 is given, otherwise -100 is given wherever a collision happens) [22]. We choose six indicators including computing time, comfort, travel efficiency, collisions, failure rate and driving compliance to evaluate the three algorithms. Comfort is reflected by the mean root square of lateral and longitudinal acceleration, i.e., $I_{\text{comfort}} = 1.4\sqrt{(a_x^2) + (a_y^2)}$. Travel efficiency is evaluated by the average time used to pass the intersection. Failure rate means the accumulated times of that decision signal is generated for more than 1 seconds and driving compliance shows times of breaking red light. The results of 100 times simulation are shown in Table II. Benefiting from the framework design, the computational efficiency of our method remains as fast as the model-free approach, and the driving performance such as safety, compliance and failure rate, is better than the other two approaches.

TABLE II: Comparison of driving performance

	IDC (Ours)	Rule-based	Model-free RL
Computing time [ms]			
Upper-quantile	5.81	73.99	4.91
Standard deviation	0.60	36.59	0.65
Comfort index	1.84	1.41	3.21
Time to pass [s]	7.86(±3.52)	24.4(±16.48)	6.73(±3.32)
Collisions	0	0	31
Failure Rate	0	13	0
Decision Compliance	0	0	17

V. Real road test

To validate the effectiveness in real world, our algorithm is deployed on a real automated vehicle at a two-way two-lane intersection. The system composition is shown as Fig. 6. The test vehicle is a Chang-An CS55 SUV, in which an industrial computer is equipped and served as the vehicle controller, while the state of ego vehicle can be obtained from RTK and CAN bus. Specially, we adopt the digital twin system to generate different driving modes of surrounding vehicle, which is a simulator developed by 51WORLD Co., Ltd. and is able to synchronize with the real-world system in real time. The experiment intersection is located at ($36^{\circ}18'20''$N, $120^{\circ}40'25''$E) in Suzhou, China. The pipeline of the real road test is as follows: first, we construct the training environment in accordance with the real road condition and train a policy network by GEP. Then it is applied online. In each time step, the ego vehicle state will be gathered from RTK and CAN, and mapped to the digital twin system, where we collect the state of surrounding vehicles. Then they both are sent to the on-board computer, in which the trained policies is embedded, to calculate the optimal and safe steering angle and acceleration. Finally, these actions will be delivered through the CAN bus to act on the automated vehicle.

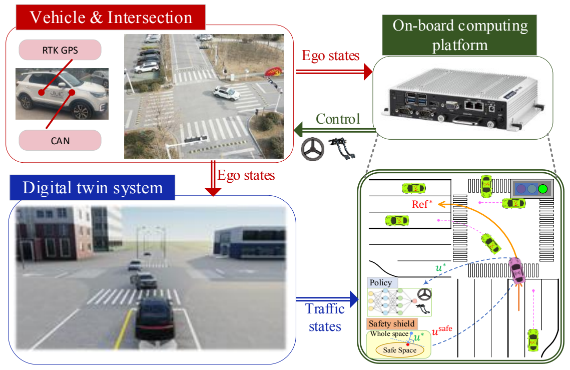
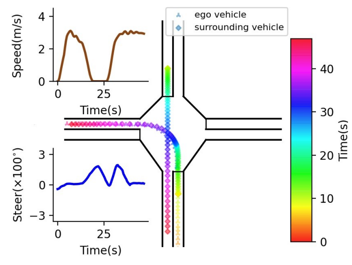
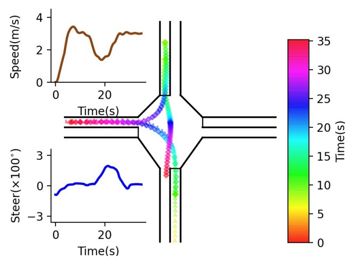


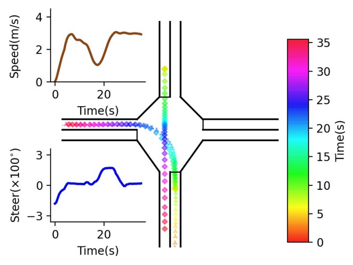
Fig. 6: Real road test system

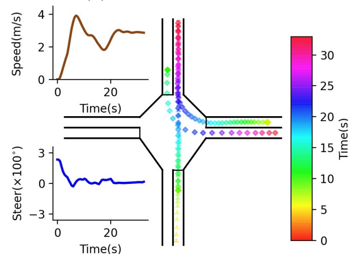
A. Functionality verification

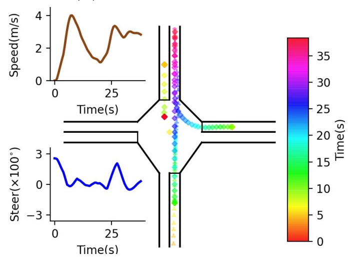
Especially, we design 9 typical cases of surrounding vehicles, 3 cases for each task. Fig. 7 shows the control effect of trained policies for ego vehicle under different behavior of surrounding vehicles. The steer and velocity show our policy has learned reasonable collision avoidance behaviors such as acceleration (2nd right-turn case), deceleration (2nd left-turn case), pulling up (1st left-turn case) and turning

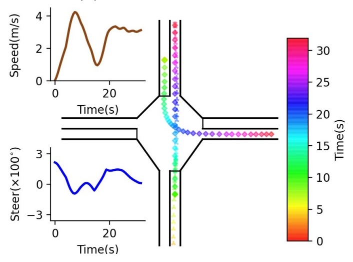

(a) lst left-turn case

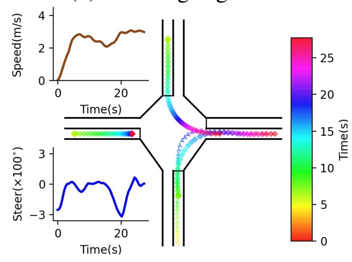

(b) 2nd left-turn case

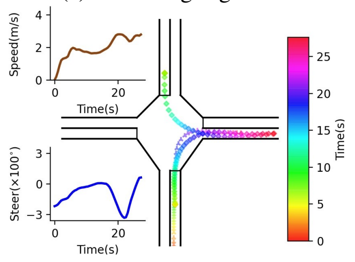

(c) 3rd left-turn case


(d) lst straight-go case


(e) 2nd straight-go case


(f) 3rd straight-go case


(g) lst right-turn case


(h) 2nd right-turn case

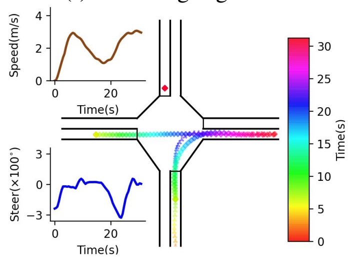

(i) 3rd right-turn case

Fig. 7: Visualization of control under different tasks and scenarios

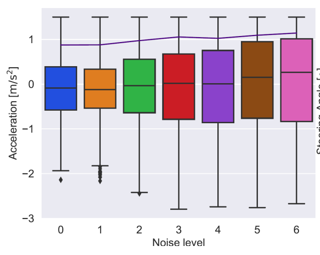
(2nd straight-go case). All the videos are available online (https://youtu.be/XQOewggafjc).

B. Robustness to noise

This experiment aims to test the robustness of the trained policies under different levels of noises added manually. To that end, we specially design 7 levels of noise, where each item is Gaussian white noise with different variance, as shown in Table III. Noises are implemented on different dimension of states in the first left-turn case and we observe their effect on output actions, shown as in Fig. 8. We can conclude that our method is rather robust to low level noises (0-3) because the data distributions in those noises have no significant change. Although the variation of actions are inevitably enlarged if we add more noise, the bounds of featured values always remain in a reasonable range, proving the robustness of our algorithm.

TABLE III: Noise level and the corresponding standard deviation

Noise level	0	1	2	3	4	5	6
δ_p [m]	0	0.017	0.033	0.051	0.068	0.085	0.102
δ_ϕ [°]	0	0.017	0.033	0.051	0.068	0.085	0.102
p_x^j, p_y^j [m]	0	0.05	0.10	0.15	0.20	0.25	0.30
$v_{\rm lon}^j$ [m/s]	0	0.05	0.10	0.15	0.20	0.25	0.30
ϕ^j [°]	0	1.4	2.8	4.2	5.6	7.0	8.4


(a) Acceleration

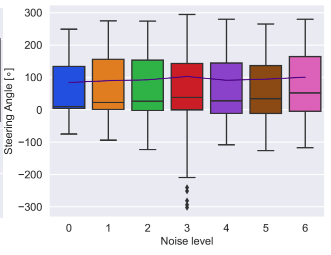
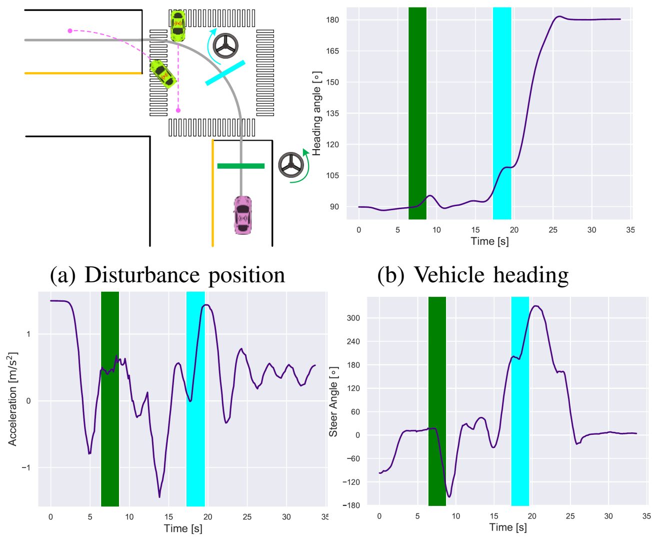

(b) Steer

Fig. 8: Actions variation in different noise level

C. Robustness to human interference

The experiment is conducted to verify the ability of our algorithm to cope with human disturbance. We also use the first left-turn case, during which we perform two times of human intervention on the steer wheel, as shown in Fig. 9(a) and the ego vehicle heading is plotted in Fig. 9(b). The first one is conducted on 6s when ego vehicle is still out of intersection and the steer wheel is turned to left manually to make it turn in advance. The second happens at 17s, when the ego is turning to the left to pass the crossroad. We turn the steering wheel right to interrupt the process. We also draw vehicle heading and actions in Fig. 9, where two colored regions are the process where human disturbance is acted on. After the first interfere, our policy turn the steering wheel right immediately to correct the excessive ego heading. About the second intervention, the driving system is able to turn the steering wheel left immediately to continue to complete the turn left operation. To sum up, the trained policy is capable of handling human disturbance on the steering wheel by quick responses after taking over.



(a) Disturbance position (b) Vehicle heading

(c) Acceleration (d) Steer

Fig. 9: Robustness to human interference

VI. Conclusions

In this paper, we focus on the complex intersection scenario and propose the integrated decision and control framework to obtain safe and computationally efficient driving performance. Specially, a model-based reinforcement learning algorithm is developed to approximately find an optimal control policy offline and its online application can guarantee the real-time requirements for on-board computer. Furthermore, we apply our algorithm on a real world intersection to verify the effectiveness. Results show that the trained policy can output the control commands within 10ms and is persistence to the disruption from environmental perception and human interference. This study provides a promising approach to utilize reinforcement learning to conquer the complex driving tasks, wherein the safety can be assured strictly and high computation efficiency is obtained. About the future work, we intend to design a learning method to choose the optimal path to replace the current rule-based version. Besides, we will consider other traffic participants like pedestrians and bicycles with more complex behavior.

References

[1] R. Tay, "A random parameters probit model of urban and rural intersection crashes," *Accident Analysis & Prevention*, vol. 84, pp. 38–40, 2015.

[2] J. Duan, S. E. Li, Y. Guan, Q. Sun, and B. Cheng, "Hierarchical reinforcement learning for self-driving decision-making without reliance on labeled driving data," *IET Intelligent Transport Systems*, 2019.

[3] 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.

[4] J. Chen, B. Yuan, and M. Tomizuka, "Model-free deep reinforcement learning for urban autonomous driving," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 2765–2771.

[5] L. Hou, L. Xin, S. E. Li, B. Cheng, and W. Wang, "Interactive trajectory prediction of surrounding road users for autonomous driving using structural-lstm network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 11, pp. 4615–4625, 2020.

[6] R. Kala and K. Warwick, "Motion planning of autonomous vehicles in a non-autonomous vehicle environment without speed lanes," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 5-6, pp. 1588–1601, 2013.

[7] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[8] E. Schmerling, L. Janson, and M. Pavone, "Optimal sampling-based motion planning under differential constraints: The driftless case," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 2368–2375.

[9] S. Karaman and E. Frazzoli, "Sampling-based optimal motion planning for non-holonomic dynamical systems," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 5041–5047.

[10] Y. Li, Z. Littlefield, and K. E. Bekris, "Sparse methods for efficient asymptotically optimal kinodynamic planning," in *Algorithmic foundations of robotics XI*. Springer, 2015, pp. 263–282.

[11] L. Xin, Y. Kong, S. E. Li, J. Chen, Y. Guan, M. Tomizuka, and B. Cheng, "Enable faster and smoother spatio-temporal trajectory planning for autonomous vehicles in constrained dynamic environment," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 235, no. 4, pp. 1101–1112, 2021.

[12] S. Aradi, "Survey of deep reinforcement learning for motion planning of autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[13] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, "Mastering the game of go without human knowledge," *nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[14] S. E. Li, "Reinforcement learning and control," 2020, tsinghua University: Lecture Notes. http://www.idlab-tsinghua.com/thulab/labweb/publications.html.

[15] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1928–1937.

[16] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[17] Y. Guan, Y. Ren, S. E. Li, Q. Sun, L. Luo, and K. Li, "Centralized cooperation for connected and automated vehicles at intersections by proximal policy optimization," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 11, pp. 12 597–12 608, 2020.

[18] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[19] G. Li, S. Li, S. E. Li, Y. Qin, D. Cao, X. Qu, and B. Cheng, "Deep reinforcement learning enabled decision-making for autonomous driving at intersections," *Automotive Innovation*, vol. 3, no. 4, pp. 374–385, 2020.

[20] P. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, and Johannes, "Microscopic traffic simulation using sumo," in *International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018.

[21] J. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi-A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[22] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proceedings of the 35th International Conference on Machine Learning*. PMLR, 2018, pp. 1861–1870.