# Efficient and Generalized end-to-end Autonomous Driving System with Latent Deep Reinforcement Learning and Demonstrations

Zuojin Tang[1,2], Xiaoyu Chen[3], Yongqiang Li[4], and Jianyu Chen[1,3] (✉)

[1]Shanghai Qizhi Institute
[2]College of Computer Science and Technology, Zhejiang University
[3]Institute for Interdisciplinary Information Sciences, Tsinghua University
[4]Mogo Auto Intelligence and Telematics Information Technology Co. , Ltd
Corresponding to: jianyuchen@tsinghua.edu.cn

**Abstract.** An intelligent driving system should dynamically formulate appropriate driving strategies based on the current environment and vehicle status while ensuring system security and reliability. However, methods based on reinforcement learning and imitation learning often suffer from high sample complexity, poor generalization, and low safety. To address these challenges, this paper introduces an efficient and generalized end-to-end autonomous driving system (EGADS) for complex and varied scenarios. The RL agent in our EGADS combines variational inference with normalizing flows, which are independent of distribution assumptions. This combination allows the agent to capture historical information relevant to driving in latent space effectively, thereby significantly reducing sample complexity. Additionally, we enhance safety by formulating robust safety constraints and improve generalization and performance by integrating RL with expert demonstrations. Experimental results demonstrate that, compared to existing methods, EGADS significantly reduces sample complexity, greatly improves safety performance, and exhibits strong generalization capabilities in complex urban scenarios. Particularly, we contributed an expert dataset collected through human expert steering wheel control, specifically using the G29 steering wheel. Our code is available: *https://github.com/Mark-zjtang/EGADS?tab=readme-ov-file.*

## 1 Introduction

An intelligent autonomous driving systems must be able to handle complex road geometry and topology, complex multi-agent interactions with dense surrounding dynamic objects, and accurately follow the planning and obstacle avoidance. Current, autonomous driving systems in industry are mainly using a highly modularized hand-engineered approach, for example, perception, localization, behavior prediction, decision making and motion control, etc. [40] and [41]. Particularly, the autonomous driving decision making systems are focusing on the

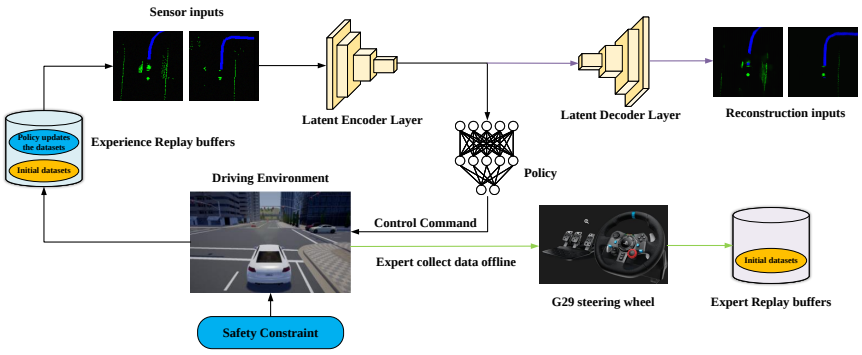


**Fig. 1.** Overview of the efficient and generalized end-to-end autonomous driving system with latent deep reinforcement learning and demonstrations.

non-learning model-based methods, which often requires to manually design a driving policy [14] and [31]. However, the manually designed policy could have two several weaknesses: 1) Accuracy. The driving policy of human heuristics and pre-training model can be suboptimal, which will lead to either conservative or aggressive driving policies. 2) Generality. For different scenarios and complicated tasks, we might need to be redesigned the model policy manually for each new scenario.

To solve those problems, existing works such as imitation learning (IL) is most popular approach, which can learn a driving policy by collecting the expert driving data. However, those methods can suffer from the following shortcomings for imitation learning: (1) High training cost and sample complexity. (2) Conservation. Due to the collect driving data from the human expert, which can only learn driving skills that are demonstrated in the datasets. (3) Limitation of driving performance. What's more, the driving policy based on reinforcement learning (RL) is also popular method in recent years, which can automatically learn and explore without any human expert data in various kinds of different driving cases, and it is possible to have a better performance than imitation learning. However, the existing methods also have some weakness: (1) Existing methods in latent space are based on specific distribution assumptions, whereas distributions in the real world tend to be more flexible, resulting in a failure to learn more precisely about belief values. (2) High costs of learning and exploration. (3) The safety and generalization of intelligent vehicles need further improvement.

Combining the advantages of RL and IL, the demonstration of enhanced RL is not only expected to accelerate the initial learning process, but also gain the potential of experts beyond performance. In this paper, we introduce an efficient and generalized end-to-end autonomous driving system (EGADS) for complex and varied scenarios. The RL agent in our EGADS combines variational inference with normalizing flows independent of distribution assumptions, allowing it

to sufficiently and flexibly capture historical information useful for driving in latent space, thereby significantly reducing sample complexity. In addition, unlike traditional methods that constrain policy actions directly, we integrate safety constraints into the reward function, which allows the agent to consider safety during training, thereby improving its robustness and generalization. To further increase the upper limit of the overall system, we further enhance the RL search process with a dataset of human experts. In particular, we contributed a dataset of human experts to driving by driving the G29 steering wheel. The experimental results show that compared with the existing methods, our EGADS greatly improves the safety performance, shows strong generalization ability in multiple test maps, and significantly reduces the sample complexity. In summary, our contributions are:

- We present an EGADS framework designed for complex and varied scenarios.
- The RL agent in EGADS uses variational inference with normalizing flows (NFRL), independent of distribution assumptions, to capture historical driving information in latent space, significantly reducing sample complexity.
- we incorporate Safety Constraints (SC) directly into the reward function to enable the agent to account for safety considerations during training.
- By fine-tuning with a small amount of human expert dataset via using the G29 steering wheel, NFRL agents can learn more general driving principles, significantly improving generalization and sample efficiency.

## 2 Related Work

Imitation learning, which utilizes an efficient supervised learning approach, has gained widespread application in autonomous driving research due to its simplicity and effectiveness. For instance, imitation learning has been employed in end-to-end autonomous driving systems that directly generate control signals from raw sensor inputs [27, 8, 1, 5].

Deep reinforcement learning (DRL) has demonstrated its strength in addressing complex decision-making and planning problems, leading to a series of breakthroughs in recent years. Researchers have been trying to apply deep RL techniques to the domain of autonomous driving. Lillicarp et.al [24] introduced a continuous control DRL algorithm that trains a deep neural network policy for autonomous driving on a simulated racing track. Wolf et.al [43] used Deep Q-Network to learn to steer an autonomous vehicle to keep in the track in simulation. Chen et.al [4] developed a hierarchical DRL framework to handle driving scenarios with intricate decision-making processes, such as navigating traffic lights. Kendall et.al [22] marked the first application of DRL in real-world autonomous driving, where they trained a deep lane-keeping policy using only a single front-view camera image as input. Chen et.al [3] proposed an interpretable DRL method for end-to-end autonomous driving. Nehme et.al [30] proposed safe navigation. Murdoch et.al [29] propose a partial end-to-end algorithm that decouples the planning and control tasks. Zhou et.al [46] proposes a method to identify and protect unreliable decisions of a DRL driving policy. Zhang et.al [44]

a framework of constrained multi-agent reinforcement learning with a parallel safety shield for CAVs in challenging driving scenarios. Liu et.al [26] propose the Scene-Rep Transformer to enhance RL decision-making capabilities.

By combining the advantages of RL and IL is also a relatively popular method in recent years. The techniques outlined in [38], [42], [47] and [45] have proven to be efficient in merging demonstrations and RL for improving learning speed. Liu et.al [25] propose a novel framework combining RL and expert demonstration to learn a motion control strategy for urban scenarios. Huang et.al [19] introduces a predictive behavior planning framework that learns to predict and evaluate from human driving data. Huang et.al [20] propose an enhanced human in-the-loop reinforcement learning method, while they rely on human expert performance and can only accomplish simple scenario tasks. DPAG [32] combines RL and imitation learning to solve complex dexterous manipulation problems. Our approach utilizes the potential for reinforcement learning and normalization flows to learn useful information from historical trajectory information, further learning expert demonstrations through DPAG methods.

## 3 Methodology

The proposed framework of our EGADS is illustrated in Figure 1. Firstly, human experts collect demonstrations offline using the G29 steering wheel. These expert demonstrations are then utilized as the RL fine-tuning experience replay buffers for training the entire model. Subsequently, a pre-training process is conducted to establish a model with human expert experience that does not update environmental data during training. The resulting model, enriched with human expert experience, is then used to fine-tune the policy for RL agent. Additionally, we have designed safety constraints for the intelligent vehicle, enhancing its safety performance.

### 3.1 Preliminaries

We model the control problem as a Partially Observable Markov Decision Process (POMDP), which is defined using the 7-tuple: $(S, A, T, R, \Omega, O, \gamma)$, where $S$ is a set of states, $A$ is a set of actions, $T$ is a set of conditional transition probabilities between states, $R$ is the reward function, $\Omega$ is a set of observations, $O$ is a set of conditional observation probabilities, and $\gamma$ is the discount factor. The goal of the RL agent is to maximize expected cumulative reward $E[\sum_{t=0}^{\infty} \gamma_t r_t]$. After having taken action $a_{t-1}$ and observing $o_t$, an agent needs to update its belief state, which is defined as the probability distribution of the environment state conditioned on all historical information: $b(s_t) = p(s_t \mid \tau_t, o_t)$, where $\tau_t = \{o_1, a_1, \ldots, o_{t-1}, a_{t-1}\}$.

### 3.2 Latent dynamic model for autonomous driving

We propose the use of latent variables to solve problems in end-to-end autonomous driving. This potential space is used to encode complex urban driv-

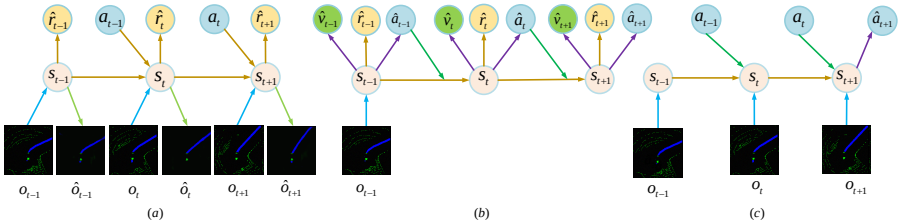

**Fig. 2.** (a) RL agent learns potential dynamics from past experience datasets. (b) RL agent predicts driving action in an imaginary space. (c) RL agent interacts with driving environment. Where $o$ is observation, $a$ is action, $s$ is latent state, $\hat{r}_t$ is reward, $\hat{o}_t$ is reconstructed and $\hat{v}_t$ is value.

ing environments, including visual inputs, spatial features, and road conditions. Historical high-dimensional raw observation data is compressed into this low-dimensional latent space and learned through a sequential latent environment model that learns in conjunction with the maximum entropy RL process. We introduce RL agent model consists of components can be constructed the probabilistic graphical model of POMDP as follow:

$$\begin{aligned} \text{State transition model:} \quad & p_\theta(s_t|s_{t-1}, a_{t-1}) \\ \text{Reward model:} \quad & p_\theta(r_t|s_t) \\ \text{Observation model:} \quad & p_\theta(o_t|s_t) \end{aligned} \tag{1}$$

where $p$ is prior probability, $q$ is posterior probability, $o$ is observation, $a$ is action, is latent state and $\theta$ is the parameter of the model.

### 3.3 RL agent in the latent space

Visual control [39], [34], [2] can be defined as a POMDP. The traditional components of agents that learn through imagination include dynamics learning, behavior learning, and environment interaction [16], [17]. The RL agent in the latent space in our EGADS mainly includes the following:

(1) RL agent learns potential dynamics from past experience datasets of autonomous vehicle. As shown in Figure 2(a), using $p$ to represent prior probability, $q$ to represent posterior probability, agent learns to encode observation and action into compact latent state, and $\hat{o}_t$ is reconstructed with $q(\hat{o}_t|s_t)$ while $s_t$ is determined via $p(s_t|s_{t-1}, a_{t-1}, o_t)$.

(2) RL agent predicts driving action in an imaginary space. As shown in Figure 2(b), RL agent is in a close latent state space where it can predict value $\hat{v}_t$, reward $\hat{r}_t$ and action $\hat{a}_t$ based on current input $o_{t-1}$ with $q(\hat{v}_t, \hat{r}_t, \hat{a}_t|s_t)$, $p(s_t|s_{t-1}, \hat{a}_{t-1})$, $q(\hat{a}_{t-1}|s_{t-1})$.

(3) RL agent interacts with driving environment. As shown in Figure 2(c), RL agent predicts next action values $\hat{a}_{t+1}$ by encoding historical trajectory information via $q(\hat{a}_{t+1}|s_{t+1})$, $p(s_{t+1}|s_t, a_t, o_{t+1})$.

### 3.4 Normalizing Flow for inferred belief

Existing latent RL models in autonomous driving either suffer from the curse of dimensionality or make some assumptions and only learn approximate distributions. This approximation imposes strong limitations and is problematic, whereas distributions in the real world tend to be more flexible. In the continuous and dynamic space, existing methods based on normalizing flows (NF) [10] , [18], [33] can learn more flexible and generalized beliefs. These methods provide a solid foundation for RL agents to accurately predict future driving actions. Inspired by [7], we added a belief inference model: $q_\theta(s_t|\tau_t, o_t)$, where $\theta$ is the parameter of the model. The belief model can be substituted for the probability density with NF in the KL-divergence term of equation 2.

$$\begin{aligned} q_K(s_t|\tau_t, o_t) &= \log q_0(s_t|\tau_t, o_t) - \sum_{k=1}^{K} |det \frac{\partial f_{\psi_k}}{\partial s_{t,k-1}}| \\ p_K(s_t|\tau_t) &= \log p_0(s_t|\tau_t) - \sum_{k=1}^{K} |det \frac{\partial f_{\omega_k}}{\partial s_{t,k-1}}| \end{aligned} \tag{2}$$

where $q_0 = q_\theta$, $q_K = q_{\theta,\psi}$, $p_0 = p_\theta$, $p_K = p_{\theta,\omega}$, $\psi$ and $\omega$ are the parameters of a series of mapping transformations of the posterior and prior distributions. Where $\tau_t = \{o_1, a_1, \cdots, o_{t-1}, a_{t-1}\}$. The input images $o_{1:t}$ and actions $a_{1:t-1}$ are encoded with $q_\theta(s_t|\tau_t, o_t)$. Then the final inferred belief is obtained by propagating $q_\theta(s_t|\tau_t, o_t)$ through a set of NF mappings denoted $f_{\psi_K} \dots f_{\psi_1}$ to get a posterior distribution $q_{\theta,\psi}(s_t|\tau_t, o_t)$. The final prior is obtained by propagating $p_\theta(s_t|\tau_t)$ through a set of NF mappings denoted $f_{\omega_K} \dots f_{\omega_1}$ to get a prior distribution $p_{\theta,\omega}(s_t|\tau_t)$. Where $p_K(s_t|\tau_t) = p_K(s_t|s_{t-1}, a_{t-1})$, given the sampled $s_{t-1}$ from $q_K(s_{1:t}|\tau_t, o_t)$. Finally, our NF inference RL model (NFRL) is optimized by variational inference method, in which the evidence lower bound (ELBO) [21], [9] is maximized. The loss function is defined as:

$$\begin{aligned} \mathcal{M}_{\text{model}}(\theta, \psi, \omega) = \sum_{t=1}^{T} \big( & \mathbb{E}_{q(s_t|o_{\leq t}, a_{<t})}[\log p_\theta(o_t \mid s_t) + \\ & \log p_\theta(r_t \mid s_t)] - \mathbb{E}_{q_K(s_{1:T}|o_{1:T}, a_{1:T-1})}[D_{\text{KL}}\big(q_K(s_t \mid \tau_t, o_t) \| \\ & p_K(s_t \mid \tau_t, o_t))] \big) \end{aligned} \tag{3}$$

### 3.5 Policy optimization

The action model implements the policy and is designed to predict the actions that are likely to be effective in responding to the simulated environment. The value model estimates the expected reward generated by the behavior model at each state $s_\tau$.

$$a_\tau \sim q_\phi(a_\tau|s_\tau), \quad v_\eta(s_\tau) = E_{q_\phi}[\sum_{t=t}^{t+H} \gamma^{\tau-t} r_\tau] \tag{4}$$

where $\phi$, $\eta$ are the parameters of the approximated policy and value. The obejective of the action model is to use high value estimates to predict action that result in state trajectories

$$\mathcal{M}_{\text{actor}}(\phi) = E_{q_\phi}(\sum_{\tau=t}^{t+H} V_\tau^\lambda) \tag{5}$$

To update the action and value models, we calculate the value estimate $v_\eta(s_\tau)$ for all states $s_\tau$ along the imagined trajectory. $V_\tau^\lambda$ can be defined as follow:

$$V_\tau^\lambda = (1-\tau)v_\eta(s_{\tau+1}) + \lambda V_{\tau+1}^\lambda, \quad \tau < t+H \tag{6}$$

Then we can train the critic to regress the TD($\lambda$) [35] target return via a mean squared error loss:

$$\mathcal{M}_{\text{critic}}(\eta) = \mathbb{E}\Big[\sum_{\tau=t}^{t+H} \frac{1}{2}\Big(v_\eta\left(s_\tau\right) - V_\tau^\lambda\Big)^2\Big] \tag{7}$$

where $\eta$ denote the parameters of the critic network and $H$ is the prediction horizon. Then the loss function is as follows:

$$\min_{\psi,\eta,\phi,\theta,\omega,\eta} \alpha_0 \mathcal{M}_{\text{critic}}(\eta) - \alpha_1 \mathcal{M}_{\text{actor}}(\phi) - \alpha_2 \mathcal{M}_{\text{model}}(\theta,\psi,\omega) \tag{8}$$

we jointly optimize the parameters of model loss $\psi, \theta, \omega$, critic loss $\eta$ and actor loss $\phi$, where $\alpha_0, \alpha_1, \alpha_2$ are coefficients for different components.

### 3.6 Safety constraint

In the Gym-Carla benchmark, the reward function proposed by Chen et.al [6] is denoted as $f_1$. To ensure the intelligent vehicle operates safely and smoothly in complex environments, we incorporated additional safety and robustness constraints into $f_1$, denoted as $f_2 = f_1 + 200r_{ft} + 50r_{lt} + 2r_{sc}$. $r_{ft}$ is the front time to collision. $r_{lt}$ is lateral time to collision. $r_{sc}$ is the smoothness constraint.

(1) Front time to collision. When around vehicles are within the distance of ego vehicle (our agent vehicle) head in our setting, then we can calculate the front time to collision between ego vehicle and around vehicles. Firstly, the speed and steering vector $(s_\tau, a_\tau) \in \mathbb{S}$ of the ego vehicle are defined, where $s_\tau$ represents the angle vector of vehicle steering and $a_\tau$ represents the acceleration vector of the vehicle in local coordinate system. Secondly, two waypoints closest to the current ego vehicle are selected from the given navigation routing as direction vectors $w_p$ for the entire route progression, where $\rightarrow$ indicates a vector in world coordinates. The position vectors for both ego vehicle and around vehicles are represented by $(x_t^*, y_t^*)$, respectively. Finally, $\delta_e$ and $\delta_a$ representing angles between position vectors for ego vehicle and around vehicles with respect to $w_p$ are calculated respectively.

$$\begin{aligned} w_p &= \big[(\frac{w_{t+1}^x - w_t^x}{2}) - (w_t^x), (\frac{w_{t+1}^y - w_t^y}{2}) - (w_t^y)\big] \\ \delta_e &= \frac{[v_t^{x*}, v_t^{y*}] \cdot w_p}{\|v_t^{x*}, v_t^{y*}\|_2 \quad \|w_p\|_2}, \delta_a = \frac{[v_t^x, v_t^y] \cdot w_p}{\|v_t^x, v_t^y\|_2 \quad \|w_p\|_2} \end{aligned} \tag{9}$$

where, $l$ is the length of the set of waypoints $\mathbb{W}$ stored. The variable $t \in \tau$, and $w_t^x \in \mathbb{W}_1$ represents the $x$ coordinate of the first navigation point closest to the intelligent vehicle on its current route at time $t$. Similarly, $w_{t+1}^x \in \mathbb{W}_2$. Furthermore, it is possible to calculate the $F_{ttc}$ as follows:

$$F_{ttc} = \frac{\|x_t - x_t^*, y_t - y_t^*\|_2}{|\|v_t^{x*}, v_t^{y*}\|_2 sin(\delta_e) - \|v_t^x, v_t^y\|_2 sin(\delta_a)|} \tag{10}$$

(2) Lateral time to collision. When around vehicles are not within the distance of ego vehicle head in our setting, we consider significantly the $L_{ttc}$. The calculation method for $L_{ttc}$ and $F_{ttc}$ is the same. However, the collision constraint effect of $L_{ttc}$ on intelligent vehicle is limited, mainly due to the slow reaction time of intelligent vehicle to $L_{ttc}$, lack of robustness and generalization ability. Therefore, we have implemented a method of assigning values to different intervals for $L_{ttc}$ as follows:

$$\begin{cases} \min(z_\tau, c_\tau + 1.0), & \nu_g \leq (c_\tau - 1.5) \text{ and } \mu_a \leq (c_\tau - 0.5). \\ \min(z_\tau, c_\tau - 1.8), & \nu_g \leq (c_\tau - 3.0) \text{ and } \mu_a \leq (c_\tau - 2.0). \\ \min(z_\tau, c_\tau - 3.0), & \nu_g \leq (c_\tau - 3.5) \text{ and } \mu_a \leq (c_\tau - 3.0). \end{cases} \tag{11}$$

where $c_\tau$ is the empirical const of $L_{ttc}$ in our setting at (5,7), $z_\tau$ is the ttc based on their combined speed. $\nu_g$ is the ttc obtained by calculating the longitudinal velocity. $\mu_a$ is the ttc obtained by calculating the lateral velocity.

(3) Smooth steering is defined as $|s_t^\delta - s_t^{*\delta}| \in e_c$. $s_t^\delta$ is the actual steering angle. $s_t^{*\delta}$ is the predicted steering angle based on policy $\pi$. The range of $e_c$ can be established based on empirical data.

### 3.7 Augmenting RL policy with demonstrations

Though, NFRL can significantly reduce complexity, and reward design based on safety constraints can enhance safety. Demonstrations can mitigate the need for painstaking reward shaping, guide exploration, further reduce sample complexity, and help generate robust, natural behaviors. We propose the demonstration augmented RL agent method which incorporates demonstrations into NFRL agent in two ways:

(1) Pretraining with behavior cloning. We use behavior cloning to provide a policy $\pi^*$ via expert demonstrations and then to train a model $\mathcal{M}_{\text{expert}}$ with some expert ability.

$$\mathcal{M}_{\text{expert}} = \underset{\xi}{\text{maximize}} \sum_{(s',a') \in \pi^*(\mathcal{D}_e)} \ln \pi_\xi^*(a'_\tau | s'_\tau) \tag{12}$$

where $\mathcal{D}_e$ is a human expert dataset obtained from driving G29 steering.

(2) RL fine-tuning with augmented loss: we employ $\mathcal{M}_{\text{expert}}$ to initialize a model trained by deep RL policies, which reduces the sampling complexity of the deep RL policy. The training loss of the actor model as follows:

$$\mathcal{M}_{\hat{\text{actor}}}(\phi, \xi) = \mathcal{M}_{\text{actor}}(\phi) + k \ln \pi_\xi^*(a'_\tau | s'_\tau), (a'_\tau, s'_\tau) \in \mathcal{D}_e \tag{13}$$

where $k$ represents the balance between the behavior cloning policy and NFRL policy, and is set as a constant based on empirical data. We only changed the actor model of NFRL, and the optimization of the other parts is exactly the same.

# 4 Experiment

## 4.1 Experiment setup

Models were trained on an NVIDIA RTX 3090 GPU using Python 3.8. Our experiments were conducted on a benchmark called Gym-carla, a third-party environment for OpenAI gym that is used with the CARLA simulator[11]. In our experiments, as shown in Figure 3, the NFRL series models and baseline methods were trained in Town03 (random) and evaluated across four scenarios: Town03, Town04, Town05, and Town06. These scenarios are abbreviated as Town03-Town06, encompassing both random and roundabout modes. Town03 simulates a realistic urban environment with diverse features such as tunnels, intersections, roundabouts, curves, and turnaround bends. Here, "random" refers to randomly selected intersections and driving scenarios, while "roundabout" focuses specifically on roundabout intersections. Town04, a small town embedded in the mountains with a special infinite highway. Town05, squared-grid town with cross junctions and a bridge. Town06, long many lane highways with many highway entrances and exits.


**Fig. 3.** The road networks of the CARLA include routes for Town03, Town04, Town05, and Town06

## 4.2 Comparison settings

In order to evaluate the performance of our autonomous driving system more effectively, we have conducted various comparisons with existing methods such as DDPG [24], SAC [15], TD3 [12], DQN [28], Latent_SAC [3], Dreamer [16], CQL [23]. We decomposed EGADS into three components: NFRL, Safety Constraint (SC) and augmenting RL policy with demonstrations (Demo). We then conducted evaluations using four comparison settings, NFRL, NFRL+SC, BC+Demo, NFRL+SC+Demo. BC+Demo indicates the use of behavioral cloning to imitate the expert dataset, while NFRL+SC+Demo involves using expert datasets to augment the NFRL policy combined with SC.

### 4.3 Hyperparameter settings

$\mathcal{M}_{model}$, the KL regularizer is clipped below 3.0 free nats for imagination range $H = 15$ using the same trajectories for updating action and value models separately with $\lambda = 0.99$ and $\lambda = 0.95$, while $k = 1.5$. The size of all our trainig and evaluating images is $128 \times 128 \times 3$. A random seed $S = 5$ is used to collect datasets for the *ego* vehicle before updating the model every $C = 100$ steps during training process.

All baseline methods share a model size of 32 but differ in other hyperparameters: conventional RL algorithms (DDPG, SAC, TD3, DQN) and Latent_SAC use a larger batch size of 256 with 5 evaluation episodes and action repeat of 2, while NFRL-based methods (NFRL, NFRL+SC, BC+Demo, NFRL+SC+Demo) and Dreamer adopt a smaller batch size of 32 with 10 evaluation episodes and action repeat of 1.

Regarding learning rates: The standard RL methods (DDPG, SAC, TD3, DQN) and Latent_SAC share identical learning rate configurations, with model learning rate at $1 \times 10^{-4}$, and both actor and value learning rates at $3 \times 10^{-4}$. The model-based approaches (Dreamer, NFRL series methods and BC+Demo) demonstrate different patterns, using a higher model learning rate of $1 \times 10^{-3}$, while maintaining lower actor and value learning rates at $8 \times 10^{-5}$. Notably, all NFRL variants (NFRL, NFRL+SC, NFRL+SC+Demo) and BC+Demo maintain identical learning rate configurations with the Dreamer method.

### 4.4 Measure Driving Performance Metrics

In the Gym-Carla benchmark, an episode terminates under any of the following conditions: the number of collisions exceeds one, the maximum number of time steps is reached, the destination is reached, the cumulative lateral deviation from the lane exceeds 10 meters, or the vehicle remains stationary for 50 seconds. EGADS is an end-to-end autonomous driving system. We implemented our trained model on an autonomous vehicle for urban navigation, assessing performance through five standard metrics: Off-road Rate (OR), Episode Completion Rate (ER), Average Safe Driving Distance (ASD), Average Reward (AR) using the reward function from Chen et.al [6] that accounts for driving dynamics (yaw, collisions, speeding, and lateral velocity), and Driving Score (DS): a composite metric calculated as DS = ER $\times$ AR in accordance with CARLA Leaderboard standards. During model selection, we focused on checkpoints that simultaneously optimized DS and AR, while implementing the remaining metrics (ER, OR, AR, ASD) based on the methodology from Gao et.al [13] and Tang et.al [37] [36]. As below:

$$OR = \frac{N_{\text{off\_road\_events}}}{N_{\text{total\_episodes}}}, ER = \frac{N_{\text{completed\_steps}}}{N_{\text{total\_steps}}}, AR = \frac{\sum_{i=1}^{N_{\text{episodes}}} \text{rewards}_i}{N_{\text{total\_episodes}}} \quad (14)$$

$$ASD = \frac{\sum_{i=1}^{N_{\text{episodes}}} \text{distance}_i}{N_{\text{total\_episodes}}}, DS = ER \times AR \quad (15)$$

Where $N_{\text{off\_road\_events}}$ is the number of times the vehicle went off-road, and $N_{\text{total\_steps}}$ is the total number of episodes. Where $N_{\text{total\_episodes}}$ is the total number of episodes in the test. Where $\text{distance}_i$ is the distance driven during the $i$-th safe driving episode. Where $N_{\text{completed\_steps}}$ is the number of successfully completed steps, and $N_{\text{total\_steps}}$ is the total number of steps in the episode. Where $AR$ is the average reward $f$ collected during the episode.

### 4.5 Collect expert datasets

CARLA can be operated and controlled through using the python API. Figure 4 shows that we establish a connection between the Logitech G29 steering wheel and the CARLA, and then human expert can collect the datasets of teaching via the G29 steering wheel. Specifically, we linearly map accelerator pedals, brake pedals, and turning angles into *accel[0,3](min,max)*, *brake[-8,0](min,max)*, *steer[-1,1](left,right)*. The tensors are written into user-built Python scripts and combined with CARLA built-in Python API so that users can provide input from their steering wheels to autonomous driving cars in CARLA simulator for $\mathcal{D}_{expert}$ collection. Particularly, we contributed a dataset collected through human expert steering wheel control.

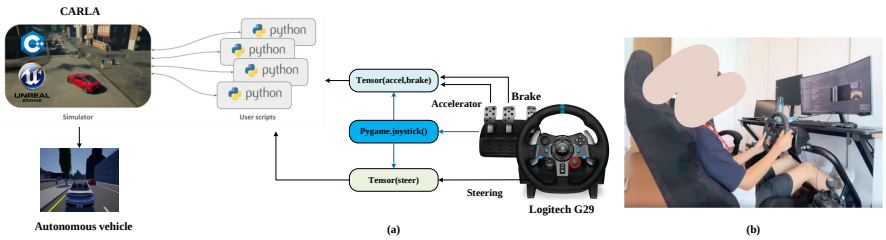


**Fig. 4.** (a) CARLA connects with the G29 steering wheel (b) Human expert collects the datasets via the G29 steering wheel

### 4.6 Results on trajectory prediction

In order to accurately evaluate our model prediction of driving actions for intelligent vehicle, this problem can be viewed as a special POMDP problem with the reward value maintained at 0. As shown in Figure 5, the comparison with ground-truth data demonstrates that our NFRL model achieves higher accuracy and greater diversity than Dreamer, with no mode collapse and significantly reduced blurring effects.

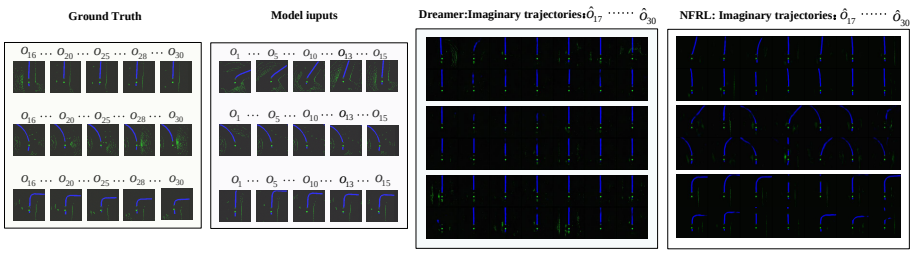


**Fig. 5.** Randomly sample sensor inputs Lidar_noground $o_1, o_2, \cdots, o_{15}$, and then our model can imagine driving behaviors $\hat{o}_{16}, \hat{o}_{17}, \cdots, \hat{o}_{30}$.

**Table 1.** In training, all methods were compared under different RL baselines in Town03 (random), with episodes of 500 steps. $+\infty$ indicates failure to reach the baseline within the maximum tested runtime of 250 GPU hours.

| Method | ASD=50m | | ASD=100m | |
|---|---|---|---|---|
| | episodes↓ | times↓ | episodes↓ | times ↓ |
| DDPG | $+\infty$ | $+\infty$ | $+\infty$ | $+\infty$ |
| SAC | $+\infty$ | $+\infty$ | $+\infty$ | $+\infty$ |
| TD3 | $\geq$161 | $\geq$192h | $+\infty$ | $+\infty$ |
| DQN | $\geq$163 | $\geq$53h | $+\infty$ | $+\infty$ |
| Latent_SAC | $\geq$167 | $\geq$43h | $\geq$352 | $\geq$105h |
| Dreamer | $+\infty$ | $+\infty$ | $+\infty$ | $+\infty$ |
| NFRL(our) | $\geq$141 | $\geq$**21h** | $\geq$121 | $\geq$**65h** |

### 4.7 How to reduce sampling complexity ?

To evaluate the sampling complexity of different methods, we used the average ASD as the test threshold and set three distinct checkpoints at 50m, 100m, and 200m. We measured the GPU hours required for each method to reach the corresponding ASD threshold, with a maximum testing duration capped at 250 GPU hours, as shown in Tables 1 and 2. Notably, in Table 1, although different methods require varying numbers of episodes to reach the ASD threshold, the actual time consumed differs significantly. This is because each episode has a fixed length of 500 steps. Some methods remain stationary for most of the episode, yet the episode does not terminate early, leading to prolonged total runtime. In contrast, other methods may collide or deviate from the lane, triggering early termination of the episode.

As shown in Table 1, our proposed NFRL method significantly improves training time efficiency, achieving at least a 2-fold acceleration in reaching the 50-meter and 100-meter baselines compared to existing reinforcement learning methods. However, due to frequent collision issues observed in experiments, the method fails to surpass the 150-meter baseline. To address this limitation, we innovatively design a reward function incorporating Safety Constraints (SC). Experimental results, as presented in Table 2, show that the enhanced NFRL+SC method not only successfully achieves the 200-meter baseline but also improves

**Table 2.** In training, all methods were compared under different NFRL baselines in Town03 (random), with episodes of 500 steps. $+\infty$ indicates failure to reach the baseline within the maximum tested runtime of 250 GPU hours.

| Method | ASD=50m | | ASD=100m | | ASD=200m | |
|---|---|---|---|---|---|---|
| | episodes↓ | times↓ | episodes↓ | times ↓ | episodes↓ | times ↓ |
| NFRL | ≥141 | **≥21h** | ≥121 | **≥65h** | $+\infty$ | $+\infty$ |
| NFRL+SC | ≥71 | ≥12h | ≥301 | ≥40h | ≥1100 | ≥146h |
| NFRL+SC+Demo | ≥21 | **≥1.3h** | ≥58 | **≥3h** | ≥321 | **≥48h** |

**Table 3.** Performance Comparison Across multiple Towns (Trained in Town03, Evaluated in Town04-Town06, hereinafter referred to as T04-T06)

| Method | DS ↑ | | | AR ($f_1$) ↑ | | | EC (%) ↑ | | | OR (%) ↓ | | | ASD (m) ↑ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | T04 | T05 | T06 | T04 | T05 | T06 | T04 | T05 | T06 | T04 | T05 | T06 | T04 | T05 | T06 |
| DDPG | -0.10 | -0.01 | -0.08 | -10.01 | -10.1 | -10.02 | 0.00 | 0.01 | 0.00 | - | - | - | 0.00 | 0.00 | 0.00 |
| DQN | 17.50 | 60.67 | 69.09 | 76.37 | 174.89 | 206.66 | 11.38 | 15.84 | 16.34 | 11.83 | 11.83 | 15.26 | 20.29 | 31.01 | 36.83 |
| TD3 | -15.89 | -2.24 | -25.62 | -131.36 | -84.30 | -195.60 | 9.18 | 8.16 | 5.82 | 33.32 | 16.94 | 16.02 | 6.17 | 10.05 | 4.50 |
| SAC | -20.56 | -14.89 | -15.02 | -14.08 | -18.92 | -16.67 | 4.95 | **69.07** | **85.60** | 0.00 | 0.00 | 0.00 | 6.71 | 6.07 | 8.03 |
| L_SAC | 102.61 | 110.66 | 21.52 | 170.79 | 8.70 | 145.77 | 15.09 | 12.78 | 12.21 | 1.05 | 4.96 | 4.64 | 15.97 | 21.24 | 42.15 |
| Dreamer | -0.01 | -0.03 | -0.03 | -15.10 | -15.10 | -15.20 | 0.00 | 0.12 | 0.20 | - | - | - | 0.01 | 0.01 | 0.00 |
| NFRL (base) | **326.78** | **390.54** | **431.44** | **1509.90** | **785.92** | **947.26** | **15.81** | 22.61 | 29.59 | 30.88 | 12.05 | 16.50 | **220.18** | **123.24** | **143.61** |

training efficiency by at least 1.5 times compared to the original NFRL method. To further optimize performance, we introduce expert datasets for fine-tuning. Experimental data indicate that the NFRL+SC+Demo method achieves a remarkable 3-fold improvement in training efficiency over the NFRL+SC method when reaching the 200-meter baseline.

The performance improvements are primarily driven by three key mechanisms: (1) The NFRL framework employs Normalizing Flow technology to reconstruct training data distributions, aligning them more closely with real-world driving scenarios. This technique enables both accurate future trajectory prediction and comprehensive coverage of possible trajectories across diverse driving situations. Such high-quality data representation allows the model to rapidly learn correct behavioral patterns. (2) The Safety Constraint (SC) module dynamically limits the policy exploration scope to safe regions, thereby minimizing costly divergent behaviors. (3) Demonstration data accelerates reward function discovery by injecting domain-specific prior knowledge. This co-design enables EGADS to achieve efficient convergence in complex autonomous driving scenarios, establishing it as a paradigm for sample-efficient reinforcement learning.

### 4.8 Ablation study

In the ablation study of the EGADS system, we evaluated the contributions of each module in cross-domain scenarios (evaluated in Town04, Town05 and Town06, trained in Town03) to validate the generalization performance of the NFRL, SC and Demo modules, as shown in Tables 5. The driving score (DS) served as the primary comprehensive metric, with other indicators providing supplementary reference. The addition of the SC module significantly improves

**Table 4.** Evaluation results for different methods in CARLA Town03 (random) and Town03 (roundabout): we denote RND as random and RBT as roundabout. For a fair comparison, all reward functions are in the form of $f_1$. Particularly, − indicates that valid data could not be obtained because the episode completion rate for this method is close to 0.

| Method | DS ↑ | | AR ($f_1$) ↑ | | EC(%) ↑ | | OR(%) ↓ | | ASD(m) ↑ | |
|---|---|---|---|---|---|---|---|---|---|---|
| | RND | RBT | RND | RBT | RND | RBT | RND | RBT | RND | RBT |
| DDPG | −0.11 | −0.08 | −10.01 | −10.02 | 0.01 | 0.00 | – | – | 0.00 | 0.00 |
| DQN | 30.64 | 36.33 | 86.50 | 121.24 | 17.02 | 16.42 | 8.52 | 11.83 | 21.68 | 26.27 |
| TD3 | 2.40 | −6.60 | −18.15 | −129.52 | 6.91 | 4.07 | 51.53 | 49.32 | 7.51 | 3.12 |
| SAC | −7.57 | −20.56 | −19.90 | −24.74 | **63.76** | **67.95** | **0.00** | **0.65** | 6.27 | 6.71 |
| L_SAC | 125.95 | 31.59 | 161.24 | 84.13 | 11.98 | 10.50 | 14.72 | 1.14 | 31.31 | 13.87 |
| Dreamer | −0.03 | −0.02 | −15.12 | −15.12 | 0.01 | 0.02 | – | – | 0.01 | 0.01 |
| NFRL (base) | **170.03** | **48.73** | **424.60** | **249.17** | 24.04 | 10.88 | 20.93 | 18.11 | **72.16** | **46.27** |

**Table 5.** During the evaluation, an ablation study of EGADS's three modules across scenarios was conducted (Trained in Town03, Evaluated in Town04-Town06, hereinafter referred to as T04-T06)

| Method | DS ↑ | | | AR ($f_1$) ↑ | | | EC (%) ↑ | | | OR (%) ↓ | | | ASD (m) ↑ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | T04 | T05 | T06 | T04 | T05 | T06 | T04 | T05 | T06 | T04 | T05 | T06 | T04 | T05 | T06 |
| NFRL | 326.78 | 390.54 | 431.44 | **1509.90** | 785.92 | 947.26 | 15.81 | 22.61 | 29.59 | 30.88 | 12.05 | 16.50 | **220.18** | **123.24** | 143.61 |
| NFRL+SC | 649.46 | 213.17 | 1234.89 | 418.22 | 381.39 | 1571.85 | 48.70 | 38.80 | 63.42 | **0.00** | **0.00** | **0.00** | 91.34 | 50.42 | 195.36 |
| NFRL+SC+Demo | **1174.16** | **723.90** | **2155.40** | 1329.89 | **894.58** | **2294.30** | **57.41** | **46.85** | **82.47** | 3.25 | 11.51 | 6.05 | 159.42 | 116.96 | **265.92** |

the cross-scenario performance of NFRL (e.g. NFRL vs. NFRL + SC), demonstrating the effectiveness of our SC module design. Further incorporating the Demo learning module on top of NFRL+SC, the experimental results show that NFRL+SC+Demo achieves the highest scores in Town04 (1174.16), Town05 (723.90), and Town06 (2155.40), with substantial improvements over both the baseline NFRL and NFRL+SC configurations. This proves that the Demo module enhances cross-domain generalization through expert knowledge.

As shown in Table 6, we conducted comprehensive comparisons with various mainstream baselines (online RL methods such as L_SAC and Dreamer; offline or imitation learning approaches including BC+Demo and CQL+Demo) across two challenging scenarios (Town03 RND and RBT). The multi-dimensional evaluation metrics clearly demonstrate that: 1) The NFRL framework itself surpasses existing online RL methods; 2) The SC module universally and significantly enhances both safety and overall performance across all methods, including baselines; 3) The NFRL framework effectively utilizes demonstration data, achieving far superior results compared to imitation learning and offline RL baselines; 4) The final NFRL+SC+Demo solution comprehensively outperforms all methods, including enhanced baselines, across nearly all positive metrics (DS, AR, EC, ASD) while maintaining excellent safety performance. These results fully validate the absolute superiority of our proposed method, the effectiveness of each module, and the powerful synergistic effects of their combination.

**Table 6.** Evaluation results for different methods in CARLA Town03 (random) and Town03 (roundabout): we denote RND as random and RBT as roundabout. For a fair comparison, all reward functions are in the form of $f_1$. Particularly, − indicates that valid data could not be obtained because the episode completion rate for this method is close to 0.

| Method | DS ↑ | | AR ($f_1$) ↑ | | EC(%) ↑ | | OR(%) ↓ | | ASD(m) ↑ | |
|---|---|---|---|---|---|---|---|---|---|---|
| | RND | RBT | RND | RBT | RND | RBT | RND | RBT | RND | RBT |
| L_SAC | 125.95 | 31.59 | 161.24 | 84.13 | 11.98 | 10.50 | 14.72 | 1.14 | 31.31 | 13.87 |
| Dreamer | −0.03 | −0.02 | −15.12 | −15.12 | 0.01 | 0.02 | − | − | 0.01 | 0.01 |
| NFRL | 170.03 | 48.73 | 424.60 | 249.17 | 24.04 | 10.88 | 20.93 | 18.11 | 72.16 | 46.27 |
| L_SAC+SC | 156.23 | 64.76 | 284.02 | 148.50 | 13.98 | 15.91 | 10.64 | 12.88 | 50.52 | 18.67 |
| Dreamer+SC | 98.12 | 50.02 | 124.74 | 74.98 | 10.42 | 11.20 | 18.08 | 16.35 | 42.85 | 16.90 |
| NFRL+SC | 192.84 | 101.29 | 341.56 | 181.28 | 38.46 | 34.66 | **5.87** | **4.04** | 80.21 | 50.24 |
| BC+Demo | −6.30 | −1.63 | −62.43 | −27.92 | 9.22 | 10.31 | 15.49 | 15.57 | 14.78 | 15.34 |
| CQL+Demo | 8.52 | 4.35 | 42.10 | 49.06 | 10.58 | 8.21 | 13.45 | 19.08 | 19.25 | 16.01 |
| NFRL+Demo | 203.26 | 143.03 | 478.04 | 26.15 | 25.71 | 20.66 | 10.50 | 12.82 | 81.32 | 64.80 |
| NFRL+SC+Demo | **485.92** | **380.17** | **720.27** | **653.21** | **44.25** | **36.63** | 7.69 | 5.48 | **100.13** | **84.92** |

### 4.9 How to improve generalization capabilities ?

The EGADS system enhances cross-scenario generalization through the co-design of the NFRL framework, SC module, and Demo module. NFRL decouples state representation from policy learning, establishing a transferable foundation for driving policies. As shown in Table 3, in the cross-town evaluation (Town04-Town06), NFRL achieves a significantly higher DS value compared to traditional reinforcement learning methods, demonstrating robust generalization capabilities.

As evidenced in Tables 5 and 6, the SC module effectively mitigates high-risk behaviors through trajectory smoothing, improving overall DS values compared to standalone NFRL and enhancing system robustness. Meanwhile, the Demo module accelerates policy convergence and optimizes exploration via imitation learning. As shown in Tables 5 and 6, the NFRL+SC+Demo configuration demonstrates significant improvements across multiple metrics including DS and AR , confirming that demonstration data effectively reduces inefficient sampling.

The synergy between the SC module and Demo data can be summarized as follows: the SC module establishes safety boundaries to prevent the policy from entering hazardous or suboptimal states, while Demo data alleviates the conservatism of the SC module. EGADS integrates imitation learning (BC loss) and reinforcement learning (NFRL loss), dynamically balancing their weights to enable the agent to leverage expert knowledge while exploring autonomously within safe limits. This balanced mechanism enhances the policy's generalization capability and environmental adaptability, enabling efficient task execution across diverse scenarios and rapid adaptation to new challenges.

## 5 Conclusion

In summary, our EGADS framework effectively enhances sample efficiency, safety, and generalization in autonomous driving systems. The inclusion of safety constraints significantly enhances vehicle safety. NFRL, our proposed method, accurately predicts future driving actions, reducing sample complexity. By fine-tuning with a small amount of expert data, NFRL agents learn more general driving principles, which greatly improve generalization and sample complexity reduction, offering valuable insights for autonomous driving system design.

## References

1. Bansal, M., Krizhevsky, A., Ogale, A.: Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. arXiv preprint arXiv:1812.03079 (2018)
2. Bengtsson, T., Bickel, P., Li, B.: Curse-of-dimensionality revisited: Collapse of the particle filter in very large scale systems. In: Probability and statistics: Essays in honor of David A. Freedman, vol. 2, pp. 316–335. Institute of Mathematical Statistics (2008)
3. Chen, J., Li, S.E., Tomizuka, M.: Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning. IEEE Transactions on Intelligent Transportation Systems **23**(6), 5068–5078 (2021)
4. Chen, J., Wang, Z., Tomizuka, M.: Deep hierarchical reinforcement learning for autonomous driving with distinct behaviors. In: 2018 IEEE Intelligent Vehicles Symposium. pp. 1239–1244. IEEE (2018)
5. Chen, J., Yuan, B., Tomizuka, M.: Deep imitation learning for autonomous driving in generic urban scenarios with enhanced safety. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 2884–2890. IEEE (2019)
6. Chen, J., Yuan, B., Tomizuka, M.: Model-free deep reinforcement learning for urban autonomous driving. In: 2019 IEEE Intelligent Transportation Systems Conference. pp. 2765–2771. IEEE (2019)
7. Chen, X., Mu, Y.M., Luo, P., Li, S., Chen, J.: Flow-based recurrent belief state learning for pomdps. In: International Conference on Machine Learning. pp. 3444–3468. PMLR (2022)
8. Codevilla, F., Müller, M., López, A., Koltun, V., Dosovitskiy, A.: End-to-end driving via conditional imitation learning. In: 2018 IEEE International Conference on Robotics and Automation. pp. 4693–4700. IEEE (2018)
9. De Cao, N., Aziz, W., Titov, I.: Block neural autoregressive flow. In: Uncertainty in artificial intelligence. pp. 1263–1273. PMLR (2020)
10. Dinh, L., Sohl-Dickstein, J., Bengio, S.: Density estimation using real nvp. arXiv preprint arXiv:1605.08803 (2016)
11. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: Carla: An open urban driving simulator. In: Conference on Robot Learning. pp. 1–16. PMLR (2017)
12. Fujimoto, S., Hoof, H., Meger, D.: Addressing function approximation error in actor-critic methods. In: International conference on machine learning. pp. 1587–1596. PMLR (2018)
13. Gao, Z., Mu, Y., Chen, C., Duan, J., Luo, P., Lu, Y., Li, S.E.: Enhance sample efficiency and robustness of end-to-end urban autonomous driving via semantic masked world model. IEEE Transactions on Intelligent Transportation Systems (2024)

14. González, D., Pérez, J., Milanés, V., Nashashibi, F.: A review of motion planning techniques for automated vehicles. IEEE Transactions on Intelligent Transportation Systems **17**(4), 1135–1145 (2015)
15. Haarnoja, T., Zhou, A., Abbeel, P., Levine, S.: Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: International conference on machine learning. pp. 1861–1870. PMLR (2018)
16. Hafner, D., Lillicrap, T., Ba, J., Norouzi, M.: Dream to control: Learning behaviors by latent imagination. arXiv preprint arXiv:1912.01603 (2019)
17. Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., Davidson, J.: Learning latent dynamics for planning from pixels. In: International conference on machine learning. pp. 2555–2565. PMLR (2019)
18. Huang, C.W., Krueger, D., Lacoste, A., Courville, A.: Neural autoregressive flows. In: International Conference on Machine Learning. pp. 2078–2087. PMLR (2018)
19. Huang, Z., Liu, H., Wu, J., Lv, C.: Conditional predictive behavior planning with inverse reinforcement learning for human-like autonomous driving. IEEE Transactions on Intelligent Transportation Systems (2023)
20. Huang, Z., Sheng, Z., Ma, C., Chen, S.: Human as ai mentor: Enhanced human-in-the-loop reinforcement learning for safe and efficient autonomous driving. arXiv preprint arXiv:2401.03160 (2024)
21. Jordan, M.I., Ghahramani, Z., Jaakkola, T.S., Saul, L.K.: An introduction to variational methods for graphical models. Learning in graphical models pp. 105–161 (1998)
22. Kendall, A., Hawke, J., Janz, D., Mazur, P., Reda, D., Allen, J.M., Lam, V.D., Bewley, A., Shah, A.: Learning to drive in a day. In: 2019 International Conference on Robotics and Automation. pp. 8248–8254. IEEE (2019)
23. Kumar, A., Zhou, A., Tucker, G., Levine, S.: Conservative q-learning for offline reinforcement learning. Advances in Neural Information Processing Systems **33**, 1179–1191 (2020)
24. Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D.: Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971 (2015)
25. Liu, H., Huang, Z., Lv, C.: Improved deep reinforcement learning with expert demonstrations for urban autonomous driving. 2022 IEEE Intelligent Vehicles Symposium (IV) pp. 921–928 (2021), https://api.semanticscholar.org/CorpusID:231951804
26. Liu, H., Huang, Z., Mo, X., Lv, C.: Augmenting reinforcement learning with transformer-based scene representation learning for decision-making of autonomous driving. arXiv preprint arXiv:2208.12263 (2022)
27. Mero, L.L., Yi, D., Dianati, M., Mouzakitis, A.: A survey on imitation learning techniques for end-to-end autonomous vehicles. IEEE Transactions on Intelligent Transportation Systems **23**, 14128–14147 (2022), https://api.semanticscholar.org/CorpusID:246539766
28. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., et al.: Human-level control through deep reinforcement learning. nature **518**(7540), 529–533 (2015)
29. Murdoch, A., Schoeman, J.C., Jordaan, H.W.: Partial end-to-end reinforcement learning for robustness against modelling error in autonomous racing. arXiv preprint arXiv:2312.06406 (2023)
30. Nehme, G., Deo, T.Y.: Safe navigation: Training autonomous vehicles using deep reinforcement learning in carla. arXiv preprint arXiv:2311.10735 (2023)

31. Paden, B., Čáp, M., Yong, S.Z., Yershov, D., Frazzoli, E.: A survey of motion planning and control techniques for self-driving urban vehicles. IEEE Transactions on Intelligent Vehicles **1**(1), 33–55 (2016)
32. Rajeswaran, A., Kumar, V., Gupta, A., Vezzani, G., Schulman, J., Todorov, E., Levine, S.: Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. arXiv preprint arXiv:1709.10087 (2017)
33. Rezende, D., Mohamed, S.: Variational inference with normalizing flows. In: International conference on machine learning. pp. 1530–1538. PMLR (2015)
34. Silver, D., Veness, J.: Monte-carlo planning in large pomdps. Advances in neural information processing systems **23** (2010)
35. Sutton, R.S., Barto, A.G.: Reinforcement learning: An introduction. MIT press (2018)
36. Tang, Z., Chen, X., Li, Y., Chen, J.: Efficient and generalized end-to-end autonomous driving system with latent deep reinforcement learning and demonstrations. arXiv preprint arXiv:2401.11792 (2024)
37. Tang, Z., Hu, B., Zhao, C., Ma, D., Pan, G., Liu, B.: How to build a pre-trained multimodal model for simultaneously chatting and decision-making? arXiv preprint arXiv:2410.15885 (2024)
38. Theodorou, E., Buchli, J., Schaal, S.: Reinforcement learning of motor skills in high dimensions: A path integral approach. In: 2010 IEEE International Conference on Robotics and Automation. pp. 2397–2403. IEEE (2010)
39. Thrun, S.: Monte carlo pomdps. Advances in neural information processing systems **12** (1999)
40. Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., et al.: Stanley: The robot that won the darpa grand challenge. Journal of field Robotics **23**(9), 661–692 (2006)
41. Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M., Dolan, J., Duggins, D., Galatali, T., Geyer, C., et al.: Autonomous driving in urban environments: Boss and the urban challenge. Journal of field Robotics **25**(8), 425–466 (2008)
42. Van Hoof, H., Hermans, T., Neumann, G., Peters, J.: Learning robot in-hand manipulation with tactile features. In: 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids). pp. 121–127. IEEE (2015)
43. Wolf, P., Hubschneider, C., Weber, M., Bauer, A., Härtl, J., Dürr, F., Zöllner, J.M.: Learning how to drive in a real world simulation with deep q-networks. In: 2017 IEEE Intelligent Vehicles Symposium. pp. 244–250. IEEE (2017)
44. Zhang, Z., Han, S., Wang, J., Miao, F.: Spatial-temporal-aware safe multi-agent reinforcement learning of connected autonomous vehicles in challenging scenarios. In: 2023 IEEE International Conference on Robotics and Automation. pp. 5574–5580. IEEE (2023)
45. Zhao, C., Zhou, Z., Liu, B.: On context distribution shift in task representation learning for online meta rl. In: International Conference on Intelligent Computing. pp. 614–628. Springer (2023)
46. Zhou, W., Cao, Z., Deng, N., Jiang, K., Yang, D.: Identify, estimate and bound the uncertainty of reinforcement learning for autonomous driving. IEEE Transactions on Intelligent Transportation Systems (2023)
47. Zhou, Z., Hu, B., Zhao, C., Zhang, P., Liu, B.: Large language model as a policy teacher for training reinforcement learning agents. arXiv preprint arXiv:2311.13373 (2023)