Consistency Policy with Categorical Critic for Autonomous Driving

Xing Fang
Institute of Automation, CASIA
Beijing, China
School of Artificial Intelligence, UCAS
Beijing, China
fangxing2022@ia.ac.cn

Qichao Zhang
Institute of Automation, CASIA
Beijing, China
School of Artificial Intelligence, UCAS
Beijing, China
zhangqichao2014@ia.ac.cn

Haoran Li
Institute of Automation, CASIA
Beijing, China
School of Artificial Intelligence, UCAS
Beijing, China
lihaoran2015@ia.ac.cn

Dongbin Zhao
Institute of Automation, CASIA
Beijing, China
School of Artificial Intelligence, UCAS
Beijing, China
dongbin.zhao@ia.ac.cn

ABSTRACT

In the domain of autonomous driving, employing reinforcement learning (RL) for decision-making must effectively capture the range of feasible actions and accurately predict their consequences. The classical actor-critic framework in RL achieves this through an actor that selects actions and a critic that evaluates their values. However, traditional Gaussian-distributed actors are limited to learning unimodal distributions, which limits their ability to fully represent the diversity of executable actions that can be learned from past interactions. Moreover, the mean squared error (MSE) loss often employed by the critic is prone to significant estimation biases due to the non-stationary nature of RL training, leading to inaccurate assessments of future outcomes. In this paper, we introduce Consistency Policy with Categorical Critic (CPCC), a novel approach that leverages recent advancements in diffusion models, particularly consistency models, to serve as the actor, enabling the representation of multimodal action distributions. Additionally, we utilize classification loss (cross-entropy loss) for training the categorical critic, which mitigates overfitting to noisy targets and yields more precise approximations of Q-values. Experimental results obtained from the simulated driving environment MetaDrive substantiate the effectiveness of our proposed method. Code is available at https://github.com/weiaiF/cpcc.

KEYWORDS

Reinforcement Learning, Autonomous Driving, Diffusion Model, Consistency Model

ACM Reference Format:
Xing Fang, Qichao Zhang, Haoran Li, and Dongbin Zhao. 2025. Consistency Policy with Categorical Critic for Autonomous Driving. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 9 pages.

Corresponding author: Qichao Zhang.

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

1 INTRODUCTION

Autonomous driving (AD) technology is gradually changing human transportation, providing individuals with a more intelligent travel experience. Decision-making is a vital component in ensuring the reliability of an autonomous driving system [55]. Two primary machine learning approaches utilizing machine learning are employed to enhance decision-making in autonomous driving: one involves imitation learning [19, 24] to fit expert driving trajectories, while the other utilizes reinforcement learning (RL) [46, 48, 50] to learn safe behavioral decisions through trial-and-error interactions with the environment. In the imitation learning paradigm, actions are selected based on maximizing probability, which lacks consideration of the consequences associated with those actions. The reinforcement learning paradigm, which learns the Q-function through interactions with the environment, has the potential to address issues such as causal confusion that are prevalent in imitation learning. This makes RL a more promising approach for the decision-making task.

In the reinforcement learning framework, it is crucial to clearly identify the set of feasible actions available at any given state and to understand the consequences of executing those actions. However, current RL-based driving policies do not account for the multimodal nature of driving behavior [25], instead relying solely on a simplistic Gaussian distribution to represent the policy distribution, which fails to accurately capture the feasible set of candidate actions. Recently, consistency models [43], as an innovative enhancement of diffusion models [17], have gained considerable attention due to their powerful distribution representation capabilities and efficient inference speed in the image generation tasks [31, 42]. Consequently, it is a natural progression to explore their application in the domain of driving decision-making, which requires real-time responsiveness and accurate multimodal driving action representation. Inspired by the Consistency Policy with Q-Learning (CPQL) [4], we utilize consistency models to serve as the actor in

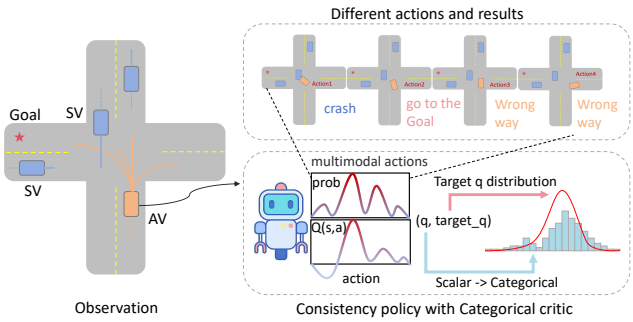


Figure 1: Illustration of our proposed planning approach. The ego vehicle utilizes the consistency policy to obtain multi-modal action candidates for the current state. The categorical critic employs the histogram loss Gauss (HL-Gauss) to transform scalars into categorical distributions.

the RL driving policy, which can derive the executed action from the noise in a single step, enabling timely decision-making and capturing the multimodal action candidates for the current state. Figure 1 illustrates the different action choices and corresponding outcomes for the ego vehicle at a crossroad. A reasonable autonomous agent should be able to clearly identify its feasible action options and make safe decisions consistently. Meanwhile, when training a RL-based driving policy, the accuracy and robustness of the critic play a central role in mitigating risky driving behaviors, such as collisions, and deviations from the road. Farebrother et al. [13] find that utilizing categorical distributions rather than scalars to represent the output of the critic effectively reduces overfitting to noisy targets, resulting in an improved decision-making performance. Motivated by [13], we employ Histogram Losses [23] to leverage the ordinal structure of the regression task by distributing probability mass across neighboring bins as categorical distributions. Additionally, we utilize cross-entropy loss to minimize the distance between these categorical distributions, thereby constructing a robust and accurate categorical critic. In summary, the contributions of our work are summarized as follows:

- To enhance the representational capability of the actor, we utilize the consistency model with rapid sampling speed to serve as the actor, capturing the multimodality present in driving decision-making tasks.
- To address the common issue of non-stationarity in critic learning, we transform the output of the critic from a scalar to a categorical distribution and use the cross-entropy loss as the corresponding loss function. This approach enables a more accurate estimation of action outcomes.
- Experimental results on the MetaDrive simulator demonstrate the effectiveness of the proposed method and achieve comprehensive performance.

2 RELATED WORK

2.1 Multi-modal Behaviors of Human Driving

Human driving behavior is characterized by uncertainty and multiple modes [27]. To model human-like driving policy, two types of action spaces are often used, including the continuous spatiotemporal planning space (an action means a trajectory in the next few seconds) and continuous reactive control space (an action means current steering&braking values). The continuous spatiotemporal planning space is often used for imitation learning methods. To reduce the difficulty of exploration, the simplified continuous reactive control space is mainly used for RL methods, such as AdapMen [30] in MetaDrive and DMVE [49] in Carla. Recently, waymax [15] has tried to explore the discretization of continuous reactive control in the driving domain to obtain discrete action spaces [11, 47]. However, those RL methods tend to produce a unimodal policy distribution or output a deterministic action directly, which weakens the expressiveness of complicated policy and decays the ability of exploration [37, 53]. We also adapt the continuous reactive control space but utilize the consistency model to learn complicated multimodal distributions, which is consistent with the multimodal characteristics of human driving behaviors.

2.2 Diffusion Model for Autonomous Driving

The diffusion model [17] serves as a robust generative deep learning framework that employs a denoising process for data generation. By effectively capturing the multimodal characteristics inherent in the data, it demonstrates significant potential for producing diverse and high-quality outputs, which has achieved significant success across image, audio, and video generation [3, 17, 34, 40]. Recently, they have been introduced to the field of autonomous driving and have made significant progress in applications such as trajectory prediction and traffic simulation. Specifically, [25, 36] utilize the diffusion model to predict future trajectories of environment vehicles and pedestrians, which exhibit a highly multimodal distribution of diverse future intentions. Furthermore, the diffusion models [5, 22, 56] have been extended to generate realistic and controllable traffic simulation, achieving significant advancements in simulating complex driving environments with greater fidelity. More recently, researchers have started to explore the application of diffusion models in the domain of driving decision-making [21, 52, 54]. They leverage diffusion models to learn from large datasets and generate multiple candidate trajectories, then employ either an existing evaluation model [7] or a self-generated evaluation model to produce the final executable trajectory. However, a key challenge with diffusion models lies in their reliance on a large number of diffusion steps, which hinders their practicality in real-time applications. This limitation is particularly problematic in autonomous driving tasks, where rapid decision-making and immediate control are critical for ensuring safety and responsiveness.

In our study, we leverage consistency models [43] to enhance the efficiency of diffusion models in driving decision-making. We position the consistency model as an actor within the reinforcement learning framework for autonomous agents, aiming to augment flexibility and diversity in ego decision-making strategies.

2.3 Consistency Model

To improve the sampling efficiency of existing diffusion models (DMs) [33, 41], consistency models (CMs) [43] are initially introduced in the field of image generation. The key principle of consistency models is to train the model to map any point in time back

to the origin of the Probability Flow Ordinary Differential Equation (PF-ODE) trajectory. This approach enables efficient image generation, offering a balance between minimal inference steps and the quality of the generated image. Recently, latent consistency models (LCMs) [31] have achieved remarkable improvements in text-conditioned image generation speed. Additionally, consistency models have been extended to various domains, including video [51], 3D human motion [6], and audio [1], unlocking new possibilities for real-time applications in these fields. While initial success has been achieved in incorporating robot tasks with the consistency model, as explored in [4, 8, 28, 35], its application to complex driving tasks involving interactions with surrounding vehicles still remains largely unexplored. To fill this research gap, we make the first attempt to achieve a real-time driving consistency policy.

2.4 Classification Losses in RL

In the fields of computer vision and tabular regression, existing work [26, 38, 39] has demonstrated that replacing regression with classification can effectively enhance performance. Recently, [23] proposes the HL-Gauss cross-entropy loss for regression and shows its efficacy on small-scale supervised regression tasks, outside of RL. Most notably, [13] illustrates for the first time that a classification objective trained with cross-entropy, particularly HL-Gauss, can effectively scale value-based reinforcement learning across various domains and network architectures.

3 PRELIMINARIES

3.1 Reinforcement Learning

In reinforcement learning, the process that an agent interacts with the environment is typically described as a Markov Decision Process (MDP) $< S, A, P, R, \gamma >$, where S represents the set of states, A represents the set of actions, $P(\cdot|s,a) : S \times A \times S \rightarrow [0,1]$ represents the dynamic transition model, $R(s,a)$ represents the reward function, and $\gamma \in [0,1]$ is the discount factor. The goal of RL is to learn a policy $\pi : S \rightarrow A$ that maximizes the expectation of the sum of discounted rewards, known as the return $G_t = \sum_{i=t}^{\infty} \gamma^{i-t} r_i$. Each policy π has a corresponding state-action value function (also known as Q function), which denotes the expected return $Q(s,a)$ when following the policy π after taking an action a in state s.

$$Q(s,a) = \mathbb{E}[G_t | s_t = s, a_t = a] \tag{1}$$

For traditional Q-learning, a behavior policy such as ϵ-greedy policy is used to interact with the environment. Based on off-policy evaluation, the optimal Q function is usually approached with a neural network approximator Q_ϕ based on the following equation:

$$Q_\phi(s,a) \leftarrow \mathbb{E}\left(r + \gamma \max_{a' \in A} Q_\phi(s', a')\right) \tag{2}$$

We also call the right of Eq. 2 the Bellman operator $\hat{\mathcal{T}} Q(s,a;\phi)$. Next, we can obtain the optimal policy:

$$\pi^*(s) = \arg\max_{a \in A} Q_\phi(s,a) \tag{3}$$

For large or continuous state and action spaces, the experience replay and target network techniques [32] proposed in DQN are used to improve the stability of training.

$$Q_\phi(s,a) \leftarrow \mathbb{E}\left(r + \gamma \max_{a' \in A} Q_{\phi'}(s', a')\right) \tag{4}$$

where the tuple (s,a,r,s') is sampled from an experience replay buffer, and ϕ' is the parameter of the target network, which is updated to the current Q network parameter ϕ after a fixed number of time steps.

3.2 RL for Autonomous Driving

When using reinforcement learning to address the driving decision-making task, we commonly encounter state representations that can be categorized into two types: image representations [10] and vector representations [12, 20, 29, 45]. Image representations refer to the perception results obtained by the ego vehicle through cameras and radar, capturing visual and sensor data. Vector representations, on the other hand, consist of processed environmental information, such as vehicle data and road topology, providing a structured and abstracted view of the environment.

The action space in these scenarios is typically represented as a continuous reactive control space in two dimensions: lateral control (steering) and longitudinal control (acceleration or braking). This dual-dimensional action space enables the vehicle to execute precise and coordinated movements.

The reward function generally includes both sparse and dense rewards. Sparse rewards are often associated with specific events, such as collisions, going out of roads, or reaching the destination. These rewards provide significant feedback for critical events. Dense rewards, in contrast, are more frequent and may include factors like speed, distance to the center of the lane, and smoothness of the driving trajectory. These continuous rewards help guide the learning process and encourage the agent to adopt more desirable behaviors.

3.3 Diffusion Model

In this section, we describe how to use the diffusion model to build the expected policy and define this policy as diffusion policy. Before formally introducing this process, we first declare that there are two different types of timesteps in the following sections. Rather than using subscripts denoting the trajectory timesteps, we use superscripts $k \in [0,K]$, $K > 0$ is a fixed constant to denote the diffusion timesteps.

The diffusion model starts by diffusing $p_{data}(a)$ (original clean data) with a stochastic differential equation (SDE):

$$da^k = \mu(a^k, k)dk + \sigma(k)dw^k \tag{5}$$

where $\mu(\cdot,\cdot)$ and $\sigma(\cdot)$ are the drift and diffusion coefficients, respectively, and $\{w^k\}_{k\in[0,K]}$ denotes the standard Brownian motion (also known as the Wiener process) capturing stochastic, Gaussian white noise excitations. Starting from a^K (noise data obtained after K iterations of adding noise to the original clean data), the diffusion model aims to recover the original data a^0 by solving a reverse process from K to 0 with the PF-ODE [44]:

$$da^k = [\mu(a^k, k) - \frac{1}{2}\sigma(k)^2 \nabla \log p_k(a^k)]dk \tag{6}$$

where $\nabla \log p_k(a^k)$ is the score function of $p_k(a)$. Thus, the diffusion model trains a neural network parameterized by θ to estimate

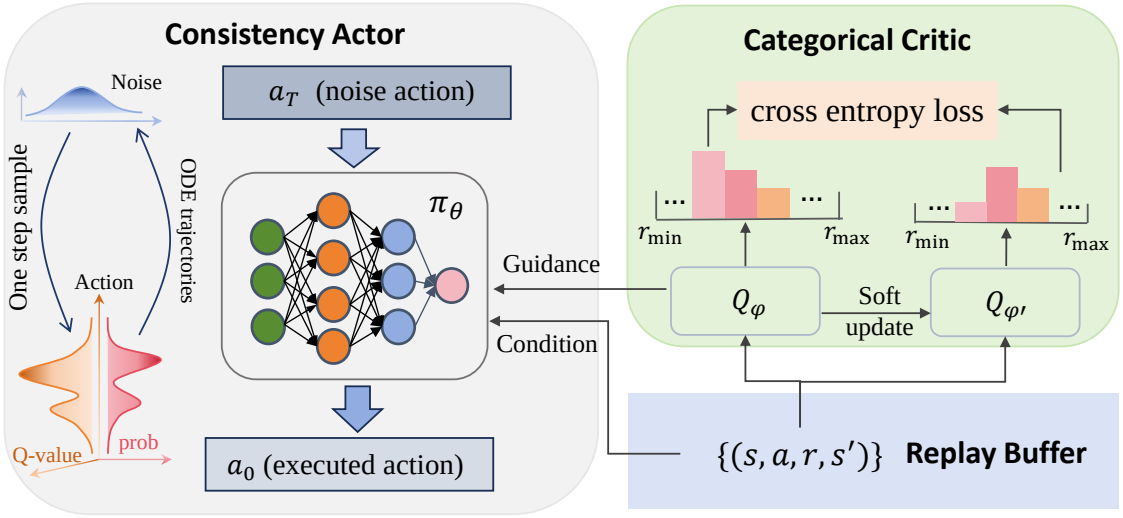


Figure 2: The framework of the CPCC algorithm.

the score function: $s_\theta(a^k, k) \approx \nabla \log p_k(a^k)$. By setting $\mu(a^k, k) = 0$ and $\sigma(k) = \sqrt{2}$, we can obtain an empirical estimate of the PF ODE:

$$\frac{da^k}{dk} = -k s_\theta(a^k, k) \tag{7}$$

We call Eq. 7 the empirical PF ODE. The reverse process along the solution trajectory $\{\hat{a}_k\}_{k\in[0,K]}$ of this ODE is the data generation process from initial random samples $\hat{a}_k \sim \mathcal{N}(\mathbf{0}, K^2\mathbf{I})$. The resulting $\hat{a}_0$ can then be viewed as an approximate sample from the data distribution $p_{data}(a)$. DMs are constrained by their slow sampling speed. Despite the availability of more advanced ODE solvers [9], they still require 10 sampling steps to achieve competitive results.

4 CONSISTENCY POLICY WITH CATEGORICAL CRITIC

4.1 Framework Overview

In this study, we conceptualize the decision-making process of autonomous vehicles through the lens of a Markov Decision Process. As shown in Figure 2, our framework follows an Actor-Critic architecture while extending the capabilities of both the actor and critic. The policy network π_θ, coupled with the Q-network Q_ϕ and target Q-network $Q_{\phi'}$ are set in the framework. We propose the Consistency Policy with Categorical Critic (CPCC) to achieve real-time driving consistency policy for the first time.

4.2 Consistency-based policy

In this section, we will introduce how to use the consistency-based policy for reinforcement learning. In the driving scenario, the ego vehicle needs to recognize the state information of surrounding vehicles and current road conditions to make appropriate decisions. Here we take the state information as the input of the consistency policy and then define it as:

$$\pi_\theta(a|s) \triangleq f_\theta(a^k, k, s) = c_{skip}(k)a^k + c_{out}(k)F_\theta(a^k, k|s) \tag{8}$$

where k represents the timestep corresponding to the ODE, $a^k \sim \mathcal{N}(\mathbf{0}, k\mathbf{I})$ and $F_\theta(a^k, k|s)$ is the neural network that we need to train, which outputs the action of the same dimension as a^k, conditioned on s, the ego state information. $c_{skip}(k)$ and $c_{out}(k)$ are differentiable functions such that $c_{skip}(\epsilon) = 1$, and $c_{out}(\epsilon) = 0$. Thus, the consistency policy $F_\theta(a^k, k|s)$ is differentiable at $t = \epsilon$ if $F_\theta(a^k, k|s), c_{skip}(k), c_{out}(k)$ are all differentiable; they will play a critical role in the later training process of the consistency model. To avoid numerical instability, we typically stop the solver at $t = \epsilon$, where ϵ is a small constant close to 0 for handling the numerical problem at the boundary. Finally, we use $\hat{a}_\epsilon \sim \pi_\theta(a|s)$ as the final executed action.

4.3 Training loss for consistency-based actor

Consistency models can be trained in either a distilled approach or an isolation approach. The former requires a pre-trained diffusion model, while the latter involves training from scratch without any prior models. Here, we adopt the independent training approach.

The core of training the consistency model is to obtain the consistency function $f : (x_t, t) \to x_\epsilon$ with the property of self-consistency: outputs are consistent for arbitrary pairs of (x_t, t) that belong to the same PF ODE trajectory, *i.e.*, $f(x_t, t) = f(x_{t'}, t')$ for all $t, t' \in [\epsilon, T]$.

Consider discretizing the time horizon $[\epsilon, K]$ into $N-1$ sub-intervals, with boundaries $k_1 = \epsilon < ... < k_N = K$. In practice, we follow Karra et al. [2] to determine the boundaries with the formula $k_i = [\epsilon^{\frac{1}{\rho}} + \frac{i-1}{N-1}(K^{\frac{1}{\rho}} - \epsilon^{\frac{1}{\rho}})]^\rho$, where $\rho = 7$. To learn the consistency policy, we minimize the objective with stochastic gradient descent on the parameter θ, while updating θ^- with with an exponential moving average (EMA). Common consistency model loss is the below equation:

$$L_{CT}(\theta) = \mathbb{E}[d(f_\theta(a + k_{m+1}z, k_{m+1}|s), f_{\theta^-}(\hat{a}^{k_m}_{\tau*}, k_m|s)] \tag{9}$$

where d represents the loss function and $\hat{a}^{k_m}_{\tau^*}$ is calculated with the Euler solver and the optimal score function $s_{\tau^*}(a^k, k)$. Based on our observations, we have identified that the aforementioned loss function may produce exceedingly small backward gradients, leading to instability during the training process and ultimately resulting in a degradation of policy performance. To achieve a more stable training outcome, we follow the approach outlined in [4] by utilizing reconstruction loss as an improvement:

$$L_{RC}(\theta, \theta^-) = \mathbb{E}_{(s,a)\sim\mathcal{D}, z\sim\mathcal{N}(\mathbf{0},\mathbf{I})}[d(f_\theta(a + k_{m+1}z, k_{m+1}|s), a)] \quad (10)$$

Furthermore, as we are operating within an online reinforcement learning research paradigm, the corresponding Q-value guidance becomes a necessary component. We anticipate that the consistency actor will generate actions with higher Q-values that are more reasonable. Therefore, we use the following equation as the final optimization objective:

$$L(\theta, \theta^-) = \alpha L_{RC}(\theta, \theta^-) - \frac{\eta}{\mathbb{E}_{(s,a)\sim\mathcal{D}[Q(s,a)]}} \mathbb{E}_{s\sim\mathcal{D}, \hat{a}^\epsilon\sim\pi_\theta}[Q(s,a)] \quad (11)$$

where α and η are hyperparameters that control whether to imitate actions from the dataset or to select actions with higher Q-values. The Q-function plays a crucial role in guiding the training of the consistency actor, which will be discussed in the next section.

4.4 Categorical critic

We now introduce the categorical critic and utilize the cross-entropy loss to train it. We adopt the framework proposed by [13] for replacing MSE with cross-entropy loss. The approach involves parameterizing the Q-function to the set of categorical distributions supported on $[r_{min}, r_{max}]$ segmented into m bins with widths $(r_{max} - r_{min})/m$, where r_{max} and r_{min} denote the maximum and minimum rewards encountered during the interaction process, respectively. The scalar value of the Q-function is then computed as:

$$Q(s, a, \phi) = \mathbb{E}[Z(s, a, \phi)], Z(s, a, \phi) = \sum_{i=1}^{m} \hat{p}_i(s, a, \phi)\delta_i \quad (12)$$

where δ_i denotes the corresponding bin value, and $\hat{p}_i$ represents the probability of the i-th bin. Specifically, $\hat{p}_i$ is derived from $l_i(s, a; \phi)$, the output of the categorical critic through the softmax function:

$$\hat{p}_i(s, a, \phi) = \frac{\exp(l_i(s, a; \phi))}{\sum_{j=1}^{m} \exp(l_j(s, a; \phi))} \quad (13)$$

We then need to consider how to convert the reward scalars generated during the interaction process into categorical distributions. HL-Gauss method [23] is particularly effective for mapping continuous values into bins within this framework. In detail, we define the random variable $Y|S, A$ with probability density $f_{Y|S,A}$ and cumulative distribution function $F_{Y|S,A}$ whose expectation is $\hat{\mathcal{T}}Q(S, A; \phi')$. Note that we need to transform the value of the target Q-function from a scalar to a categorical form in order to generate the cross-entropy supervision signal. We project the distribution $Y|S, A$ onto the histogram with bins of width $\zeta = (r_{max} - r_{min})/m$ centered at z_i by integrating over the interval $[z_i - \zeta/2, z_i + \zeta/2]$ to obtain the probabilities:

$$\begin{aligned} p_i(S, A; \phi') &= \int_{z_i-\zeta/2}^{z_i+\zeta/2} f_{Y|S,A}(y|S, A)dy \\ &= F_{Y|S,A}(z_i + \zeta/2|S, A) - F_{Y|S,A}(z_i - \zeta/2|S, A). \end{aligned} \quad (14)$$

Then, we follow [23] and choose the Gaussian distribution $Y|S, A \sim \mathcal{N}(\mu = \hat{\mathcal{T}}Q(S, A; \phi'), \sigma^2)$. Thus this formulation allows us to express the TD error using cross-entropy and utilize it for updating the value function parameters:

$$TD_{CE}(\phi) = \mathbb{E}_{\mathcal{D}} \sum_{i=1}^{m} p_i(s, a, \phi') \log \hat{p}_i(s, a, \phi) \quad (15)$$

5 EXPERIMENTS

5.1 Experiment Settings

In the paper, we use the MetaDrive simulator [29] as the online RL training and evaluation environment since it has realistic vehicle dynamics and offers a diverse range of road maps, which are randomly composed from basic elements like straight roads, curves, intersections, roundabouts, ramps, and forks. Each road map features a starting point and a destination, linked by a route consisting of a sequence of navigation points. In our experiments, the ego vehicle needs to interact with the environment to acquire environmental information, enabling it to avoid collisions and boundaries while successfully navigating from the starting point to the destination. Next, we will provide a detailed description of the states, actions, and reward information required for training the ego vehicle in the reinforcement learning framework.

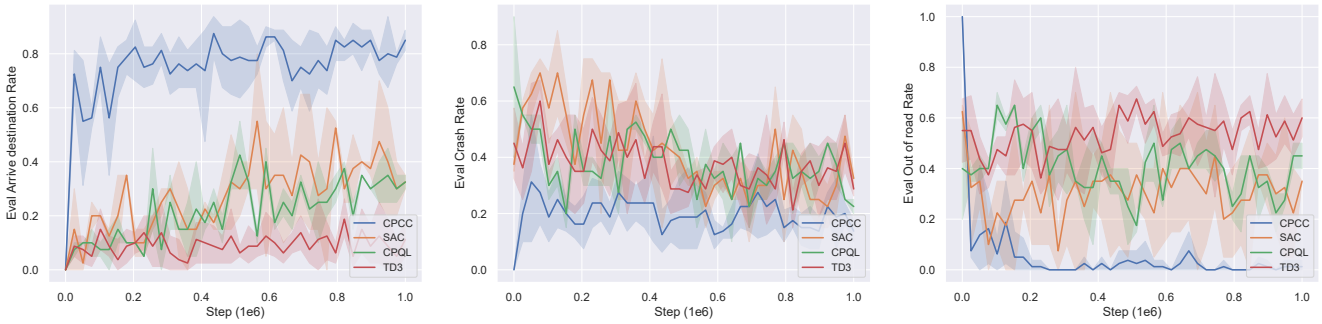
For environmental observation, we employ a 49-dimensional vector representation that incorporates the positional information of surrounding vehicles, the ego vehicle's location and motion data, as well as information related to the navigation points:

- A 30-dimensional vector representing the distances measured by a 2D-lidar to surrounding objects, with a 50-meter detection range centered on the ego vehicle.
- A 9-dimensional vector describing the ego vehicle's state, including steering, heading, speed, and its distance to the left and right boundaries.
- A 10-dimensional vector indicating the distances from the ego vehicle to evenly spaced checkpoints along the road, set 50 meters apart

The action output of the ego in the Metadrive is a normalized action $a = [a_1, a_2] \in [-1, 1]^2$. Then the above action is converted into the steering u_s, acceleration u_a and brake signal u_b. In detail, the conversion formula is as follows: $u_s = S_{max}a_1$, $u_a = F_{max}\max(0, a_2)$ and $u_b = -B_{max}\min(0, a_2)$, where S_{max}, F_{max} and B_{max} are the hyperparameters of the Metadrive.

The reward function, as shown by Eq. 16, includes the dense rewards based on driving distance and speed, as well as the sparse rewards associated with departing from the lane, colliding with surrounding vehicles or objects, and reaching the destination:

$$R = c_1 R_{dis} + c_2 * R_{speed} + R_{sparse} \quad (16)$$



(a) Eval Successful Rate (b) Eval Crash Rate (c) Eval Out of road Rate

Figure 3: Evaluation safety metric curves during the training. We evaluate CPCC and several baselines on the MetaDrive simulator every 25k training iterations. The shaded area represents half a standard deviation. The bold black line measures the average return of episodes.

where R_{dis} represents the longitudinal distance that the vehicle moves between two consecutive time steps, encouraging the ego vehicle to move forward. $R_{speed} = v_t/v_{max}$ where v_t and v_{max} denote the current velocity and the maximum velocity (80 km/h), respectively, encouraging the ego vehicle to drive fast. c_1 and c_2 are the coefficients of dense rewards R_{dis}, R_{speed}, and they are set to 1.0, 0.1, respectively, following the default setting of the Metadrive. R_{sparse} is the sparse reward, which is nonzero only when the task terminates. The ego vehicle receives a +10 reward when reaching the destination and a -5 reward when straying from the road or when crashing with an object or vehicle.

The training process is carried out on 20 road maps, each randomly generated from basic components. These maps also include randomly generated traffic vehicles, exhibiting diverse types and typical driving behaviors like lane-changing and following. Additionally, traffic accidents are randomly introduced at various locations on these maps. For evaluation, we use an independent set of 20 newly generated road maps that are unseen by the methods being assessed.

Our hardware platform is Tesla V100 paired with Intel(R) Xeon(R) CPU E5-2698 , while the software platform is Ubuntu 20.04, with PyTorch as the deep learning framework. The hyperparameters of our method are presented in the Table 1.

Table 1: Hyperparameters of CPCC

KEY	VALUE
Number of hidden layers of actor-net and critic-net.	2
Hidden layer size of actor-net and critic-net.	256
Hidden Layers Activation of actor-net and critic-net	ReLU
Discount factor γ	0.99
Consistency model ϵ	0.002
Consistency model T	80
Discount factor ρ	7
Learning rate of actor-net and critic-net	3e-4

5.2 Metrics and Baselines

Here, we follow MetaDrive[29] and [18] commonly used autonomous driving safety evaluation metrics, including the arrive destination rate, crash rate, and out of road rate. The arrival rate (success rate) represents the proportion of episodes that successfully reach the destination, while the crash rate and off-road rate indicate the ratios of total crash events and instances of vehicles leaving the road, respectively, relative to the total number of evaluation episodes.

Additionally, we employ the evaluation episode reward, defined as the average reward per episode, calculated by dividing the total reward across all episodes by the number of evaluation episodes. This metric is useful for reflecting the learning performance of RL algorithms.

Next, we present the baseline algorithms used in our study. We employed two established off-policy reinforcement learning (RL) algorithms: TD3 [14] and SAC [16]. Additionally, we included the

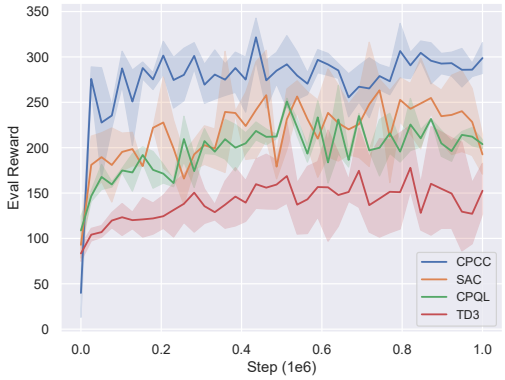


Figure 4: Evaluation reward curves during the training. We evaluate CPCC and several baselines on the MetaDrive simulator every 25k training iterations. The shaded area represents half a standard deviation. The bold black line measures the average return of episodes.

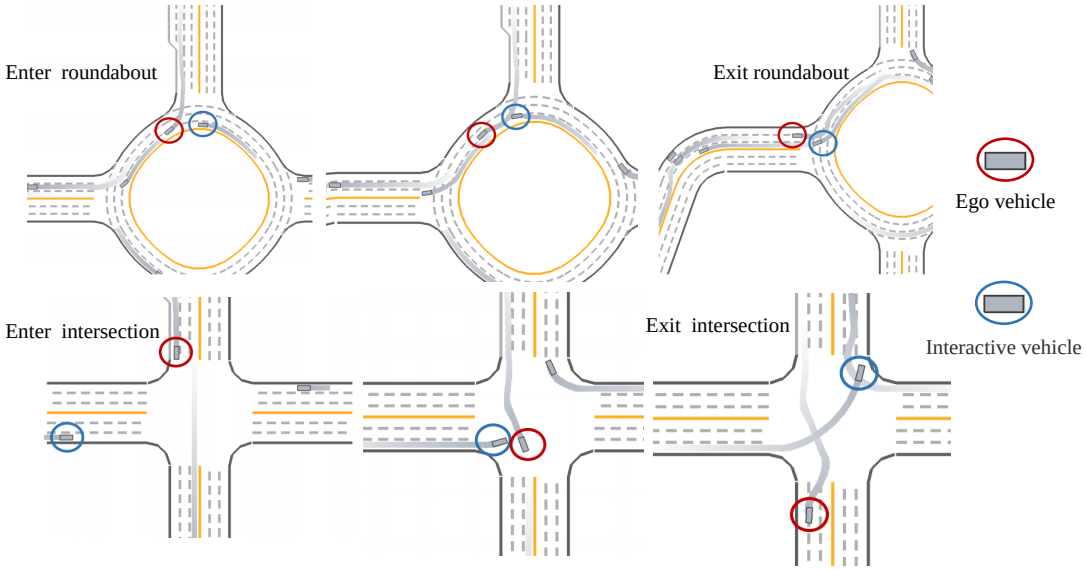


Figure 5: The Visualization results of CPCC in the roundabout scenario and intersection scenario.

CPQL algorithm [4], which innovatively incorporates consistency models within the robotics domain, as part of our baseline comparisons.

5.3 Analysis of Driving Performance

We first consider the driving safety evaluation metrics, which are presented in Table 2. Specifically, CPCC demonstrates superior driving decision-making performance, achieving an impressive success rate of 0.8125. As illustrated in Figure 3, it is clear that the CPCC algorithm quickly acquires the capability to avoid driving off the road during the early stages of training. However, it is also noted that the CPCC algorithm subsequently encounters collisions with other vehicles in the environment. Through visualization analysis, we discover that most collisions occur in scenarios characterized by strong interactions. This finding indicates that there is still room for improvement in the CPCC algorithm.

In contrast, the classical TD3 algorithm exhibits poor performance, achieving a success rate of only 0.083. The visualization results indicate that TD3 frequently fails during the initial phases of scenarios involving rapid movements. Furthermore, both the SAC and CPQL algorithms demonstrate similar performance, with success rates of 0.325 and 0.3, respectively. However, these rates remain relatively low, highlighting the effectiveness of using the categorical critic.

We then consider the results corresponding to the evaluation reward metric, which are presented in Figure 4. We find that the CPCC algorithm also achieves superior performance on the evaluation reward metric, with a score around 300, significantly surpassing the performance of the other methods.

5.4 Ablation study

We performe ablation studies on various modules, with the results summarized in Table 3. Our findings indicate that employing a standard critic (scalar critic) leads to a modest performance improvement. In contrast, using a standard actor (naive MLP) yields more substantial performance gains, albeit with increased variance. By integrating both approaches, we achieve optimal performance.

Table 2: The results of safety evaluation metrics on the Metadrive evaluation scenarios. The bold values are the highest among each row.

Method	Success rate ↑	Crash rate ↓	Out of raod rate ↓
TD3 [14]	0.083 ± 0.062	0.2833 ± 0.024	0.6333 ± 0.047
SAC [16]	0.325 ± 0.024	0.325 ± 0.026	0.35 ± 0.007
CPQL [4]	0.3 ± 0.040	0.325 ± 0.155	0.375 ± 0.125
CPCC(ours)	**0.8125** ± 0.074	**0.175** ± 0.083	**0.0125** ± 0.022

Table 3: Ablation study results of safety evaluation metrics across different modules on the MetaDrive scenarios. Bold values indicate the highest scores among each row.

Method	Success rate ↑
CPCC w normal actor and critic	0.083 ± 0.062
CPCC w normal critic	0.325 ± 0.024
CPCC w normal actor	0.533 ± 0.386
CPCC	**0.8125** ± 0.074

6 CONCLUSION

In this paper, we propose an effective driving decision-making method CPCC, which captures the multimodal action distribution

during the reinforcement learning interaction process and effectively utilizes a categorical critic to learn the consequences of different actions, thereby facilitating rational driving decision-making. We demonstrate the benefits of CPCC within the Metadrive Simulator, where our results indicate that CPCC outperforms classical off-policy reinforcement learning algorithms as well as recent consistency model planning approaches. For the future work, we aim to leverage the latest advancements in consistency models to further enhance the representational capability of the actor. Additionally, we plan to conduct corresponding experiments in real-world scenarios characterized by higher vehicle density and more complex road topologies.

ACKNOWLEDGMENTS

This work is supported by the National Key Research and Development Program of China under Grants 2022YFA1004000, the Beijing Natural Science Foundation under No. 4242052, the National Natural Science Foundation of China under Grants 62173325, and the CAS for Grand Challenges under Grants 104GJHZ2022013GC.

REFERENCES

[1] Yatong Bai, Trung Dang, Dung Tran, Kazuhito Koishida, and Somayeh Sojoudi. 2024. ConsistencyTTA: Accelerating Diffusion-Based Text-to-Audio Generation with Consistency Distillation. In *Interspeech 2024*. 3285–3289.

[2] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, Tero Karras, and Ming-Yu Liu. 2022. eDiff-I: Text-to-Image Diffusion Models with Ensemble of Expert Denoisers. *arXiv preprint arXiv:2211.01324* (2022).

[3] Omer Bar-Tal, Hila Chefer, Omer Tov, Charles Herrmann, Roni Paiss, Shiran Zada, Ariel Ephrat, Junhwa Hur, Guanghui Liu, Amit Raj, et al. 2024. Lumiere: A space-time diffusion model for video generation. In *SIGGRAPH Asia 2024 Conference Papers*. 1–11.

[4] Yuhui Chen, Haoran Li, and Dongbin Zhao. 2024. Boosting Continuous Control with Consistency Policy. In *Proceedings of the 2024 International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. International Foundation for Autonomous Agents and Multiagent Systems, 335–344.

[5] Kashyap Chitta, Daniel Dauner, and Andreas Geiger. 2024. Sledge: Synthesizing driving environments with generative models and rule-based traffic. In *European Conference on Computer Vision (ECCV)*. Springer, 57–74.

[6] Wenxun Dai, Ling-Hao Chen, Jingbo Wang, Jinpeng Liu, Bo Dai, and Yansong Tang. 2024. Motionlcm: Real-time controllable motion generation via latent consistency model. In *European Conference on Computer Vision (ECCV)*. Springer, 390–408.

[7] Daniel Dauner, Marcel Hallgarten, Andreas Geiger, and Kashyap Chitta. 2023. Parting with misconceptions about learning-based vehicle motion planning. In *Conference on Robot Learning (CoRL)*. PMLR, 1268–1281.

[8] Zihan Ding and Chi Jin. 2024. Consistency Models as a Rich and Efficient Policy Class for Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*.

[9] Tim Dockhorn, Arash Vahdat, and Karsten Kreis. 2022. GENIE: Higher-Order Denoising Diffusion Solvers. In *Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 35. 30150–30166.

[10] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. CARLA: An open urban driving simulator. In *Conference on Robot Learning (CoRL)*. PMLR, 1–16.

[11] Gabriel Dulac-Arnold, Richard Evans, Hado van Hasselt, Peter Sunehag, Timothy Lillicrap, Jonathan Hunt, Timothy Mann, Theophane Weber, Thomas Degris, and Ben Coppin. 2015. Deep reinforcement learning in large discrete action spaces. *arXiv preprint arXiv:1512.07679* (2015).

[12] Xing Fang, Qichao Zhang, Yinfeng Gao, and Dongbin Zhao. 2022. Offline Reinforcement Learning for Autonomous Driving with Real World Driving Data. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 3417–3422.

[13] Jesse Farebrother, Jordi Orbay, Quan Vuong, Adrien Ali Taiga, Yevgen Chebotar, Ted Xiao, Alex Irpan, Sergey Levine, Pablo Samuel Castro, Aleksandra Faust, Aviral Kumar, and Rishabh Agarwal. 2024. Stop Regressing: Training Value Functions via Classification for Scalable Deep RL. In *International Conference on Machine Learning (ICML)*.

[14] Scott Fujimoto, Herke Hoof, and David Meger. 2018. Addressing function approximation error in actor-critic methods. In *International conference on Machine Learning (ICML)*. PMLR, 1587–1596.

[15] Cole Gulino, Justin Fu, Wenjie Luo, George Tucker, Eli Bronstein, Yiren Lu, Jean Harb, Xinlei Pan, Yan Wang, Xiangyu Chen, John D. Co-Reyes, Rishabh Agarwal, Rebecca Roelofs, Yao Lu, Nico Montali, Paul Mougin, Zoey Yang, Brandyn White, Aleksandra Faust, Rowan McAllister, Dragomir Anguelov, and Benjamin Sapp. 2023. Waymax: An Accelerated, Data-Driven Simulator for Large-Scale Autonomous Driving Research. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*.

[16] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on Machine Learning (ICML)*. PMLR, 1861–1870.

[17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 33. 6840–6851.

[18] Xuemin Hu, Pan Chen, Yijun Wen, Bo Tang, and Long Chen. 2024. Long and Short-Term Constraints Driven Safe Reinforcement Learning for Autonomous Driving. *arXiv preprint arXiv:2403.18209* (2024).

[19] Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. 2023. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 17853–17862.

[20] Zhiyu Huang, Chen Tang, Chen Lv, Masayoshi Tomizuka, and Wei Zhan. 2024. Learning Online Belief Prediction for Efficient POMDP Planning in Autonomous Driving. *arXiv preprint arXiv:2401.15315* (2024).

[21] Zhiyu Huang, Xinshuo Weng, Maximilian Igl, Yuxiao Chen, Yulong Cao, Boris Ivanovic, Marco Pavone, and Chen Lv. 2024. Gen-Drive: Enhancing Diffusion Generative Driving Policies with Reward Modeling and Reinforcement Learning Fine-tuning. *arXiv preprint arXiv:2410.05582* (2024).

[22] Zhiyu Huang, Zixu Zhang, Ameya Vaidya, Yuxiao Chen, Chen Lv, and Jaime Fernández Fisac. 2024. Versatile Scene-Consistent Traffic Scenario Generation as Optimization with Diffusion. *arXiv preprint arXiv:2404.02524* (2024).

[23] Ehsan Imani and Martha White. 2018. Improving regression performance with distributional losses. In *International conference on Machine Learning (ICML)*. PMLR, 2157–2166.

[24] Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. 2023. Vad: Vectorized scene representation for efficient autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 8340–8350.

[25] Chiyu Jiang, Andre Cornman, Cheolho Park, Benjamin Sapp, Yin Zhou, Dragomir Anguelov, et al. 2023. Motiondiffuser: Controllable multi-agent motion prediction using diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 9644–9653.

[26] Adam Khakhar and Jacob Buckman. 2022. Neural Regression For Scale-Varying Targets. *arXiv preprint arXiv:2211.07447* (2022).

[27] Ding Li, Qichao Zhang, Zhongpu Xia, Yupeng Zheng, Kuan Zhang, Menglong Yi, Wenda Jin, and Dongbin Zhao. 2024. Planning-Inspired Hierarchical Trajectory Prediction via Lateral-Longitudinal Decomposition for Autonomous Driving. *IEEE Transactions on Intelligent Vehicles (TIV)* 9, 1 (2024), 692–703.

[28] Haoran Li, Zhennan Jiang, Yuhui Chen, and Dongbin Zhao. 2024. Generalizing Consistency Policy to Visual RL with Prioritized Proximal Experience Regularization. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[29] Quanyi Li, Zhenghao Peng, Lan Feng, Qihang Zhang, Zhenghai Xue, and Bolei Zhou. 2022. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 45, 3 (2022), 3461–3475.

[30] Xu-Hui Liu, Feng Xu, Xinyu Zhang, Tianyuan Liu, Shengyi Jiang, Ruifeng Chen, Zongzhang Zhang, and Yang Yu. 2023. How To Guide Your Learner: Imitation Learning with Active Adaptive Expert Involvement. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. International Foundation for Autonomous Agents and Multiagent Systems, 1276–1284.

[31] Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. 2023. Latent Consistency Models: Synthesizing High-Resolution Images with Few-Step Inference. arXiv:2310.04378

[32] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.

[33] Alexander Quinn Nichol and Prafulla Dhariwal. 2021. Improved denoising diffusion probabilistic models. In *International conference on Machine Learning (ICML)*. PMLR, 8162–8171.

[34] William Peebles and Saining Xie. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 4195–4205.

[35] Aaditya Prasad, Kevin Lin, Jimmy Wu, Linqi Zhou, and Jeannette Bohg. 2024. Consistency Policy: Accelerated Visuomotor Policies via Consistency Distillation. In *Robotics: Science and Systems (RSS)*.

[36] Davis Rempe, Zhengyi Luo, Xue Bin Peng, Ye Yuan, Kris Kitani, Karsten Kreis, Sanja Fidler, and Or Litany. 2023. Trace and pace: Controllable pedestrian animation via guided trajectory diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 13756–13766.

[37] Allen Z. Ren, Justin Lidard, Lars Lien Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. 2025. Diffusion Policy Policy Optimization. In *International Conference on Learning Representations (ICLR)*.

[38] Gregory Rogez, Philippe Weinzaepfel, and Cordelia Schmid. 2019. Lcr-net++: Multi-person 2d and 3d pose detection in natural images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 42, 5 (2019), 1146–1161.

[39] Rasmus Rothe, Radu Timofte, and Luc Van Gool. 2018. Deep expectation of real and apparent age from a single image without facial landmarks. *International Journal of Computer Vision* 126, 2 (2018), 144–157.

[40] Ludan Ruan, Yiyang Ma, Huan Yang, Huiguo He, Bei Liu, Jianlong Fu, Nicholas Jing Yuan, Qin Jin, and Baining Guo. 2023. Mm-diffusion: Learning multi-modal diffusion models for joint audio and video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 10219–10228.

[41] Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502* (2020).

[42] Yang Song and Prafulla Dhariwal. 2024. Improved Techniques for Training Consistency Models. In *International Conference on Learning Representations (ICLR)*.

[43] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. 2023. Consistency Models. In *International Conference on Machine Learning (ICML)*. PMLR, 32211–32252.

[44] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2021. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations (ICLR)*.

[45] Jingbo Sun, Xing Fang, and Qichao Zhang. 2023. Reinforcement Learning Driving Strategy based on Auxiliary Task for Multi-Scenarios Autonomous Driving. In *2023 IEEE 12th Data Driven Control and Learning Systems Conference (DDCLS)*. IEEE, 1337–1342.

[46] Zhentao Tang, Yuanheng Zhu, Dongbin Zhao, and Simon M Lucas. 2020. Enhanced rolling horizon evolution algorithm with opponent model learning: Results for the fighting game AI competition. *IEEE Transactions on Games (ToG)* 15, 1 (2020), 5–15.

[47] Arash Tavakoli, Fabio Pardo, and Petar Kormushev. 2018. Action branching architectures for deep reinforcement learning. In *Proceedings of the aaai conference on artificial intelligence (AAAI)*, Vol. 32.

[48] Junjie Wang, Qichao Zhang, and Dongbin Zhao. 2022. Highway Lane Change Decision-Making via Attention-Based Deep Reinforcement Learning. *IEEE/CAA Journal of Automatica Sinica* 9, 3 (2022), 567–569.

[49] Junjie Wang, Qichao Zhang, and Dongbin Zhao. 2024. Dynamic-Horizon Model-Based Value Estimation With Latent Imagination. *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)* 35, 7 (2024), 8812–8825.

[50] Junjie Wang, Qichao Zhang, Dongbin Zhao, and Yaran Chen. 2019. Lane Change Decision-making through Deep Reinforcement Learning with Rule-based Constraints. In *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 1–6.

[51] Xiang Wang, Shiwei Zhang, Han Zhang, Yu Liu, Yingya Zhang, Changxin Gao, and Nong Sang. 2023. Videolcm: Video latent consistency model. *arXiv preprint arXiv:2312.09109* (2023).

[52] Brian Yang, Huangyuan Su, Nikolaos Gkanatsios, Tsung-Wei Ke, Ayush Jain, Jeff Schneider, and Katerina Fragkiadaki. 2024. Diffusion-ES: Gradient-free Planning with Diffusion for Autonomous and Instruction-guided Driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 15342–15353.

[53] Long Yang, Zhixiong Huang, Fenghao Lei, Yucun Zhong, Yiming Yang, Cong Fang, Shiting Wen, Binbin Zhou, and Zhouchen Lin. 2023. Policy representation via diffusion probability model for reinforcement learning. *arXiv preprint arXiv:2305.13122* (2023).

[54] Yinan Zheng, Ruiming Liang, Kexin Zheng, Jinliang Zheng, Liyuan Mao, Jianxiong Li, Weihao Gu, Rui Ai, Shengbo Eben Li, Xianyuan Zhan, and Jingjing Liu. 2025. Diffusion-Based Planning for Autonomous Driving with Flexible Guidance. In *International Conference on Learning Representations (ICLR)*.

[55] Yupeng Zheng, Zebin Xing, Qichao Zhang, Bu Jin, Pengfei Li, Yuhang Zheng, Zhongpu Xia, Kun Zhan, Xianpeng Lang, Yaran Chen, et al. 2024. PlanAgent: A Multi-modal Large Language Agent for Closed-loop Vehicle Motion Planning. *arXiv preprint arXiv:2406.01587* (2024).

[56] Ziyuan Zhong, Davis Rempe, Danfei Xu, Yuxiao Chen, Sushant Veer, Tong Che, Baishakhi Ray, and Marco Pavone. 2023. Guided conditional diffusion for controllable traffic simulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 3560–3566.