BRIDGEDRIVE: DIFFUSION BRIDGE POLICY FOR CLOSED-LOOP TRAJECTORY PLANNING IN AUTONOMOUS DRIVING

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion-based planners have shown great promise for autonomous driving due to their ability to capture multi-modal driving behaviors. However, guiding these models effectively in reactive, closed-loop environments remains a significant challenge. Simple conditioning often fails to provide sufficient guidance in complex and dynamic driving scenarios. Recent work attempts to use typical expert driving behaviors (*i.e.*, anchors) to guide diffusion models but relies on a truncated schedule, which introduces theoretical inconsistencies and can compromise performance. To address this, we introduce *BridgeDrive*, a novel anchor-guided diffusion bridge policy for closed-loop trajectory planning. Our approach provides a principled diffusion framework that effectively translates anchors into fine-grained trajectory plans, appropriately responding to varying traffic conditions. Our planner is compatible with efficient ODE solvers, a critical factor for real-time autonomous driving deployment. We achieve state-of-the-art performance on the Bench2Drive benchmark, improving the success rate by 5% over prior arts.

1 INTRODUCTION

Closed-loop planning with reactive agents is a critical challenge in autonomous driving, which requires effective interaction with complex and dynamic traffic environments (Jia et al., 2024). Diffusion models has become a powerful paradigm for this task due to their ability to model complex, multi-modal distributions and incorporate flexible guidance (Liao et al., 2025; Zheng et al., 2025b; Yang et al., 2024; Xing et al., 2025). A key challenge, however, is to determine which sources of guidance information are most salient and how to integrate them effectively into these models to produce plans that are not only plausible but also safe and reactive in real-world driving conditions.

A promising source for guidance is to leverage typical human expert driving behaviors, often represented as coarse *anchor* trajectories, as they provide a strong prior for safe and sensible maneuvers, constraining the vast solution space. Recently, DiffusionDrive (Liao et al., 2025) implements this strategy by training a denoiser on a truncated diffusion schedule, starting from a noisy version of the anchor rather than pure Gaussian noise. While achieving state-of-the-art empirical performance, this approach introduces a theoretical inconsistency: its denoising process does not match the forward diffusion process that it is trained on, which diverges from the core principle of diffusion models and can lead to unpredictable behavior and compromised performance.

To address this, we introduce *BridgeDrive*, a principled diffusion framework that integrates anchor-based guidance for autonomous driving planning using a theoretically sound diffusion bridge formulation. Instead of heuristically truncating the diffusion process, we formally define the planning task as learning a diffusion process that *bridges* the gap from a given coarse anchor trajectory to a refined, context-aware final trajectory plan. This formulation ensures that the forward and denoising processes are perfectly symmetric, allowing our model to learn a direct and robust transformation from anchors to final trajectories. By adhering to the principles of diffusion, our method fully leverages the expressive power of anchors for guidance while maintaining diffusion models' ability to represent diverse human-like driving behaviors. Furthermore, our approach is compatible with efficient ODE-based samplers, enabling real-time performance crucial for on-road deployment. Empirically, we achieve 72.27% success rate on the Bench2Drive closed-loop evaluation benchmark, outperforming previous state-of-the-art method by a significant 5% margin.

2 PRELIMINARIES

2.1 AUTONOMOUS DRIVING PLANNING AND EVALUATION

The planning task in autonomous driving can be formulated as predicting future trajectories of the ego-vehicle based on raw sensor inputs. Conventionally, there are two trajectory representations (Renz et al., 2025): (1) **Temporal speed waypoints** $x := x^{\text{temp}} \in \mathbb{R}^{N_{\text{point}} \times 2}$, represent equal temporal-spaced (e.g., every 0.25 seconds) future coordinates of ego-vehicle, which inherently contain speed control information. (2) **Geometric path waypoints** $x := (x^{\text{geo}}, v) \in \mathbb{R}^{N_{\text{point}} \times 2} \times \mathbb{R}$, represent equal geometric-spaced (e.g., every 1 meter) future coordinate of ego-vehicle; for geometric path waypoints-based planning, the model needs to predict the speed v of ego-vehicle. In this paper, we choose to use geometric path waypoints as our model output, which differs from DiffusionDrive Liao et al. (2025) where temporal speed waypoints are used. This design choice is based on prior works (Chitta et al., 2023; Zimmerlin et al., 2024) and our ablation study in Section 4.

Evaluation of autonomous driving can be broadly categorized into open-loop and closed-loop settings. The closed-loop setting is more challenging and can better reflect a policy's real-world planning capability, since the ego vehicle's decisions affect its own future states and those of the surrounding agents, creating a feedback loop that can amplify small prediction errors over time. To minimize the sim-to-real gap, closed-loop evaluation requires high-fidelity simulators to capture the interactions between the ego vehicle and its surrounding environment, which are typically both computationally expensive and time-consuming. CARLA (Dosovitskiy et al., 2017) has emerged as the most widely used platform, with a series of benchmarks building on top of it, such as CARLA Leaderboard, Longest6 (Chitta et al., 2023), and Bench2Drive (Jia et al., 2024). Interestingly, existing methods that achieve near-perfect results on open-loop datasets, such as NavSim (Dauner et al., 2024) or nuScenes (Caesar et al., 2019), still struggle to achieve comparable performance under closed-loop evaluation (Li et al., 2024b; Liao et al., 2025; Fu et al., 2025; Renz et al., 2025). This discrepancy emphasizes the inherent difficulty of closed-loop planning and highlights the need for more robust methods to handle the complexities of dynamic, interactive traffic environments.

2.2 DIFFUSION MODELS

Diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021a;b; Karras et al., 2022) generate data $x_0 \sim p_d(x_0)$ from pure Gaussian noise $x_T \sim p(x_T) := \mathcal{N}(x_T|0, \sigma_{\text{max}}^2 I)$ by reverting a forward diffusion process. Mathematically, the forward diffusion process, which gradually corrupts data into noise, can be defined by a linear SDE (Song et al., 2021b):

$$\mathrm{d}x_t = f(t)x_t\mathrm{d}t + g(t)\mathrm{d}w_t, \quad x_0 \sim p_d, \tag{1}$$

where $t \in [0, T]$ denotes the diffusion timestep, $f : [0, T] \to \mathbb{R}$ is the linear drift coefficient, $g : [0, T] \to \mathbb{R}_+$ is the diffusion coefficient function, and $w_t \in \mathbb{R}^d$ is a standard Brownian motion. It turns out that this linear SDE owns a Gaussian transition kernel $q(x_t|x_0) = \mathcal{N}(x_t|\alpha_t x_0, \sigma_t^2 I)$, where $\alpha_t = \exp\left(\int_0^t f(s)\mathrm{d}s\right)$ and $\sigma_t^2 = \alpha_t^2 \int_0^t \frac{g(s)^2}{\alpha_s^2}\mathrm{d}s$ are the noise schedules (Kingma et al., 2021). The forward SDE defines a series of marginals densities $\{q(x_t)\}_{t\in[0,T]}$ along the diffusion path, where $q(x_t) = \int q(x_t|x_0)p_d(x_0)\mathrm{d}x_0$. Since $q(x_T) \approx p(x_T)$ for sufficiently large T, we can generate data $x_0 \sim p_d(x_0)$ by transforming a noise sample $x_T \sim p(x_T)$ through a probability flow ODE (PF-ODE) (Song et al., 2021b):

$$\frac{\mathrm{d}x_t}{\mathrm{d}t} = f(t)x_t - \frac{g(t)^2}{2}\nabla_{x_t} \log q(x_t), \tag{2}$$

which shares identical marginal densities $\{q(x_t)\}_{t\in[0,T]}$ as the forward SDE. The score function $\nabla_{x_t} \log q(x_t)$ in Eq. (2) can be approximated by $\nabla_{x_t} \log q(x_t) \approx (\alpha_t x_\theta(x_t, t) - x_t)/\sigma_t^2$ (Vincent, 2011), where the denoiser $x_\theta(x_t, t)$ is parameterized by a neural network and learned by minimizing the mean squared denoising error (Karras et al., 2022):

$$\min_\theta \ \mathbb{E}_{p(t)p_d(x_0)q(x_t|x_0)} \left[w(t)\|x_\theta(x_t, t) - x_0\|^2\right]. \tag{3}$$

For conditional generation, the denoiser $x_\theta(x_t, t, z)$ takes in an extra conditional variable z, which corresponds to the conditional score function $\nabla_{x_t} \log q(x_t|z) \approx (\alpha_t x_\theta(x_t, t, z) - x_t)/\sigma_t^2$. Furthermore, Ho & Salimans (2021) propose to linearly interpolate between $\nabla_{x_t} \log q(x_t|z)$ and $\nabla_{x_t} \log q(x_t)$ with a hyperparameter to adjust the guidance strength of the conditional information.

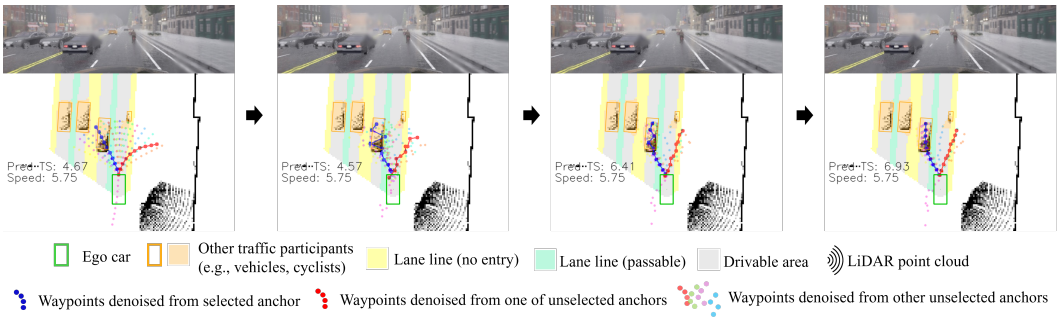

Figure 1: Visualization of the denoising process of BridgeDrive ($t = T \to 0$ from left to right), with the leftmost figure being anchor x_T and the rightmost being the planned trajectory x_0. In each figure, the blue solid line depicts the denoised trajectory of the selected anchor at a specific timestep t, the red solid line depicts an example of the denoised trajectory of an un-selected anchor, and the rest scattered dots of other colors depict the denoised trajectories of other anchors at the timestep t.

2.3 DiffusionDrive with Truncated Diffusion

DiffusionDrive (Liao et al., 2025) is a diffusion planner based on temporal speed waypoints, which leverages a truncated diffusion schedule and a fixed set of K-means clustered anchor trajectories $\mathcal{Y} = \{y^i\}_{i=1}^{N_{\text{anchor}}}$ that represent typical human driving behaviors. The truncated forward diffusion process adds a small amount of noise to each anchor until $t = T_{\text{trunc}} \ll T$ to obtain a set of noisy anchors $\{y^i_{T_{\text{trunc}}}\}_{i=1}^{N_{\text{anchor}}}$. The truncated denoising process starts from noisy anchors at $t = T_{\text{trunc}}$. Given conditional information z (e.g., sensor inputs and target point), a neural network $x_\theta(\{y_t^i\}_{i=1}^{N_{\text{anchor}}}, t, z)$ is trained to predict the best anchor and output a denoised trajectory from the noisy version of the best anchor. The denoised trajectory is then used to compute the score function for denoising.

However, as discussed in the previous section, the learned denoising process of diffusion models must revert the forward diffusion process. Although DiffusionDrive demonstrates strong empirical performance, it utilizes a truncated diffusion schedule where the forward diffusion process adds noise to anchor trajectories and the denoising process attempts to recover the ground-truth trajectories. This design choice creates an asymmetry between its forward and denoising processes, framing the model's task as regressing from noisy anchors to ground-truth trajectories, rather than as a reversal of the forward diffusion process.

3 BridgeDrive: Diffusion Bridge Policy for Trajectory Planning

To ensure the symmetry between the forward and backward processes of anchor-based diffusion planners, we propose a novel diffusion bridge policy, *BridgeDrive*, which provides a principled diffusion framework that leverages the powerful inductive biases of anchor-based guidance, while ensuring that the symmetry between the forward and denoising processes is maintained.

3.1 Anchor Construction for Geometric Path Waypoints

Anchors $\mathcal{Y} = \{y^i\}_{i=1}^{N_{\text{anchor}}}$ are pre-defined, high-priority trajectories that represent typical human expert driving behaviors. They form a discrete set of atomic building blocks that planners use to construct solutions, which can dramatically reduce planning complexity, enforce safety constraints, improve robustness to dynamic environments, and align planning with task objectives (Chai et al., 2020; Chen et al., 2024; Li et al., 2024b). Our anchor definition is slightly different from Liao et al. (2025) since our model outputs geometric path waypoints as discussed in Section 2.1. Specifically, each anchor is formulated as[1] $y := (x_y^{\text{geo}}, v_y) \in \mathbb{R}^{N_{\text{point}} \times 2} \times \mathbb{R}$, where $x_y^{\text{geo}} \in \mathbb{R}^{N_{\text{point}} \times 2}$ represents a series of coordinates of future path, N_{point} is the geometric prediction horizon, and v_y denotes the anchor speed. Each anchor trajectory x_y^{geo} is defined as a K-means clustering center of the training

[1]The subscript in x_y, v_y indicates that the trajectory and its speed *belongs to an anchor*, which differs from an ordinary trajectory x and its speed v. The superscript index i in y^i is omitted for notation simplicity.

set, and the anchor speed v_y is defined as the average speed of the trajectories in the training set that belong to this anchor. All values are normalized to the ego-vehicle coordinate system.

Algorithm 1 BridgeDrive Training (ours)

Initialize θ # denoiser parameter
repeat
  $x, y, z \sim p_d(x, y, z)$
    # x: GT traj, y: anchor, z: guidance
  $t \sim p(t)$ # $t \in [0, T]$
  $\epsilon \sim \mathcal{N}(0, I)$ # random noise
  $x_t = a_t y + b_t x + c_t \epsilon$ # noisy trajectory
  Update θ with the gradient

$$w(t)\nabla_\theta \|x_\theta(x_t, t, y, z) - x\|^2$$

until convergence
return θ

Algorithm 2 DiffusionDrive Training

Initialize θ # denoiser parameter
repeat
  $x, y, z \sim p_d(x, y, z)$
    # x: GT traj, y: anchor, z: guidance
  $t \sim p_{\text{trunc}}(t)$ # $t \in [0, T_{\text{trunc}}]$
  $\epsilon \sim \mathcal{N}(0, I)$ # random noise
  $y_t = \alpha_t y + \sigma_t \epsilon$ # noisy anchor
  Update θ with the gradient

$$w(t)\nabla_\theta \|x_\theta(y_t, t, z) - x\|^2$$

until convergence
return θ

3.2 A Generative Paradigm for Anchor-Guided Diffusion Policy

To incorporate anchors into diffusion models in a principled way, we propose to factorize the joint distribution of the ground-truth trajectory x, anchor y, and guidance information z as

$$p_d(x, y, z) = p_d(x|y, z)p_d(y|z)p_d(z). \tag{4}$$

This factorization defines a two-step generative process. First, for a driving scene $z \sim p_d(z)$, we sample an anchor $y \sim p_d(y|z)$ given the scene information in z (e.g., BEV, agent/map queries, and a target point). Then, the planned trajectory $x \sim p_d(x|y, z)$ is generated according to the guidance of the chosen anchor x_T and scene information z.

We propose to parameterize the conditional planning distribution $p_d(x|y, z)$ with a conditional diffusion bridge model $p_\theta(x_t|x_T, z)$, which constructs a diffusion bridge (Zhou et al., 2024; Zheng et al., 2025a) between the ground-truth trajectory $x_0 \coloneqq x$ and anchor $x_T \coloneqq y$ (Doob & Doob, 1984):

$$\mathrm{d}x_t = f(t)x_t\mathrm{d}t + g(t)^2\nabla_{x_t} \log q(x_T|x_t) + g(t)\mathrm{d}w_t, \quad x_0 \sim p_d,\ x_T = y, \tag{5}$$

where $t \in [0, T]$ denotes the diffusion timestep, the definitions of $f(t), g(t)$ follow those in Eq. (1), and $\nabla_{x_t} \log q(x_T|x_t) = \nabla_{x_t} \log q(x_t|x_0, x_T) - \nabla_{x_t} \log q(x_t|x_0)$. It turns out that Eq. (5) also owns an analytical Gaussian transition kernel for any given trajectory x_0 and anchor x_T:

$$q(x_t|x_0, x_T) = \mathcal{N}(x_t|a_t x_T + b_t x_0, c_t^2 I), \tag{6}$$

$$a_t = \alpha_t \gamma_t^2/\alpha_T, \quad b_t = \alpha_t(1 - \gamma_t^2), \quad c_t^2 = \sigma_t^2(1 - \gamma_t^2), \tag{7}$$

where $\alpha_t = \exp\left(\int_0^t f(s)\mathrm{d}s\right)$, $\sigma_t^2 = \alpha_t^2 \int_0^t \frac{g(s)^2}{\alpha_s^2}\mathrm{d}s$, and $\gamma_t = \frac{\alpha_T \sigma_t}{\alpha_t \sigma_T}$ (Zheng et al., 2025a), which defines a diffusion bridge $x_t = a_t x_T + b_t x_0 + c_t \epsilon_t$ that interpolates between x_0 and x_T with added Gaussian noise $c_t\epsilon_t$. Zhou et al. (2024) show that there exists a PF-ODE that shares identical marginal densities $\{q(x_t)\}_{t\in[0,T]}$ as the forward diffusion bridge SDE in Eq. (5):

$$\frac{\mathrm{d}x_t}{\mathrm{d}t} = f(t)x_t - g(t)^2\left(\frac{\nabla_{x_t} \log q(x_t|x_T, z)}{2} - \nabla_{x_t} \log q(x_T|x_t)\right), \tag{8}$$

which allows us to translate an anchor x_T to a planned trajectory x_0 given the driving scene z. To simulate this PF-ODE, we need to approximate the score function $\nabla_{x_t} \log q(x_t|x_T, z)$ for the conditional diffusion bridge model. In the next section, we will introduce our training and planning algorithms for this diffusion bridge policy.

3.3 Training and Planning Algorithms

In our diffusion bridge planner, each diffusion bridge is constructed between a ground-truth trajectory $x_0 \coloneqq x$ and the nearest anchor $x_T \coloneqq y \in \mathcal{Y}$ to it. During training, we fit a neural network

Algorithm 3 BridgeDrive Planning

$z \sim p_d(z)$ # sample a driving scene
$x_T = h_\phi(z, \mathcal{Y})$ # choose which anchor to use
for $i = N, \cdots, 1$ **do** # discretize timesteps into $T \coloneqq t_N < \cdots < t_1 < t_0 \coloneqq 0$
  $\hat{x}_{0|t_i} = x_\theta(x_{t_i}, t_i, x_T, z)$ # compute the denoised mean trajectory
  $\hat{s}_{t_i} = (a_{t_i} x_T + b_{t_i} \hat{x}_{0|t_i} - x_{t_i})/c_{t_i}^2$ # compute the score function
  $d_{t_i} = f(t_i) x_{t_i} - g(t_i)^2 \left(\hat{s}_{t_i}/2 - \nabla_{x_{t_i}} \log q(x_T|x_{t_i})\right)$ # compute the derivative dx_t/dt
  $x_{t_{i-1}} = \text{ODESolverStep}(x_{t_i}, d_{t_i}, t_i, t_{i-1})$ # simulate the diffusion bridge PF-ODE
end for
return x_0

denoiser $x_\theta(x_t, t, x_T, z)$ to predict the denoising mean $\hat{x}_{0|t} \approx \mathbb{E}[x_0|x_t, x_T, z]$ given noisy trajectory $x_t \sim q(x_t|x_0, x_T)$ at timestep t, the nearest anchor $x_T = y$ to x_0, and the conditional information z for the driving scene. This denoiser is trained by minimizing the mean squared denoising error:

$$\min_\theta \ \mathbb{E}_{p(t)p_d(x_0,x_T,z)q(x_t|x_0,x_T)} \left[w(t) \| x_\theta(x_t, t, x_T, z) - x_0 \|^2 \right]. \tag{9}$$

Our training algorithm is summarized in Algorithm 1. Notice that our forward and reverse diffusion paths both result in the end point $x_0 = x$ since $a_0 = c_0 = 0$ and $b_0 = 1$, ensuring that the denoiser is trained to reverse the forward diffusion process. On the other hand, in DiffusionDrive (Liao et al., 2025) (Algorithm 2), for all α_t and σ_t, the noisy anchor y_t deviates from x, failing to adhere to the symmetry requirement of diffusion models. Also, the training procedure of BridgeDrive is simulation-free, which allows us to efficiently train the denoiser without simulating the forward SDE in Eq. (5) or the PF-ODE in Eq. (8). In addition, since the ground-truth trajectory x_0 is not available for computing the nearest anchor y at inference time, we also train a classifier $h_\phi(z, \mathcal{Y})$ to predict the nearest anchor y to x_0 given z with the cross entropy loss.

Similar to standard diffusion models, the trained denoiser $x_\theta(x_t, t, x_T, z)$ can be used to approximate the conditional score function for our conditional diffusion bridge model (Zheng et al., 2025a):

$$\nabla_{x_t} \log q(x_t|x_T, z) \approx \frac{a_t x_T + b_t x_\theta(x_t, t, x_T, z) - x_t}{c_t^2}. \tag{10}$$

Our planning algorithm is summarized in Algorithm 3 and depicted in Fig. 2. Specifically, for a given driving scene z, we first use the classifier $h_\phi(z, \mathcal{Y})$ to choose an anchor $y \in \mathcal{Y}$, which is the starting point $x_T = y$ of the denoising process in our diffusion bridge planner. Then, we iteratively compute the denoised mean trajectory $\hat{x}_{0|t}$ using our denoiser $x_\theta(x_t, t, x_T, z)$, calculate the score function $\hat{s}_t$ using Eq. (10), and simulate the PF-ODE in Eq. (8) with the score $\hat{s}_t$ using a numerical ODE solver. Although image diffusion models use higher-order ODE solvers (Karras et al., 2022; Lu et al., 2022) to accelerate sampling, we find that first-order methods, such as the DDIM sampler (Song et al., 2021a), are sufficient for the planning task with minimal number of function evaluations. Fig. 1 visualizes the denoising process of BridgeDrive for an example driving scenario.

3.4 Model Architecture

Our model consists of three major components: perception module, denoiser, and classifier. Implementation and training details are provided in Appendix D.

Perception Module. The perception module extract useful features from lidar, front camera image, and target point for the downstream diffusion planner. We use a pre-trained perception backbone from TransFuser++ (Jaeger et al., 2023) to obtain BEV segmentation, bounding boxes of traffic participants, general traffic information (e.g., stop signs and traffic lights), and fused features from the inputs. The output of the perception module is denoted as z and will be used as the conditional guidance information in the denoiser module $x_\theta(x_t, t, x_T, z)$.

Denoiser. The architecture of the denoiser $x_\theta(x_t, t, x_T, z)$ is illustrated in the light blue box in Fig. 2. For a noisy trajectory x_t at timestep t and its corresponding anchor $x_T = y$, we first interact them with BEV via deformable spatial cross-attention modules. Subsequently, cross-attention with fused features from lidar, front camera, and target point is applied. The resulting feature vectors

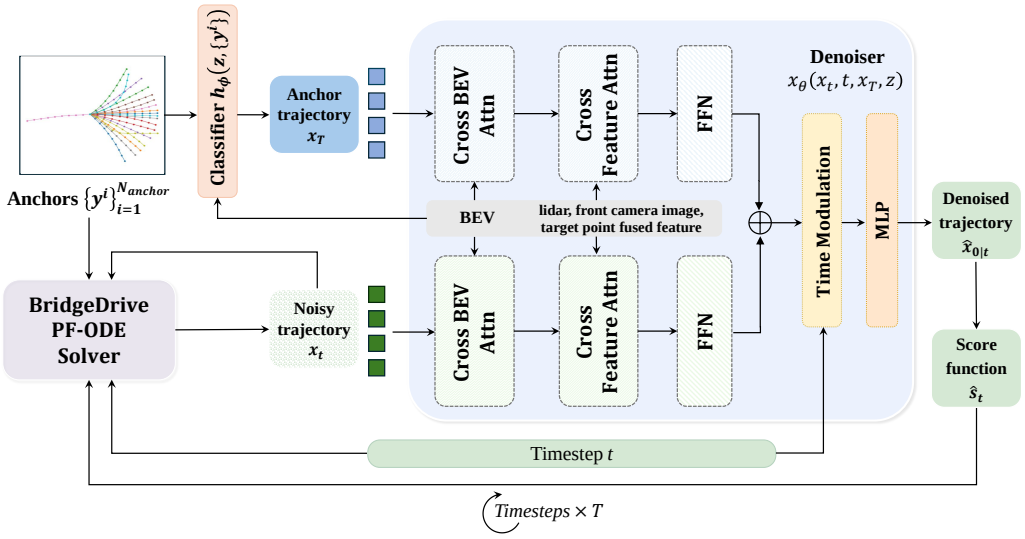

Figure 2: Diagram for the planning procedure of BridgeDrive in Algorithm 3. The model architecture of the neural network denoiser $x_\theta(x_t, t, x_T, z)$ is detailed in the light blue box.

are further processed by feed-forward networks (FFNs), and their concatenation is modulated by the timestep t. Finally, an MLP network is employed to predict the denoised mean trajectory $\hat{x}_{0|t}$.

Anchor Classifier. The classifier $h_\phi(z, \mathcal{Y})$ employs cross BEV attention module and cross feature attention module between z and all anchors $\mathcal{Y}$, followed by an FFN which outputs the probability that each anchor $y^i \in \mathcal{Y}$ should be used for trajectory generation. We select the anchor y with the highest probability as the input x_T to the denoiser $x_\theta(x_t, t, x_T = y, z)$. Note that the classifier only needs to be run once prior to the iterative denoising process.

4 Experiments

4.1 Benchmark, Dataset and Baselines

Benchmark. In this paper, we focus on closed-loop evaluation since it simulates dynamic traffic conditions which can better reflect a policy's real-world planning capability. Bench2Drive (Jia et al., 2024) is a widely used closed-loop evaluation benchmark (Jaeger et al., 2023; Jia et al., 2025; Fu et al., 2025; Renz et al., 2025), which contains 220 routes for evaluation under the CARLA Leaderboard 2.0 protocol for end-to-end autonomous driving. Each route is around 150 meters in length and contains a specific driving scenario, which allows for a detailed assessment of autonomous driving systems' proficiency in different driving skills.

Dataset. While Bench2Drive provides an official training set, empirical studies (Zimmerlin et al., 2024; Renz et al., 2025; Fu et al., 2025) showed that official dataset collected by (Li et al., 2024a) leads to suboptimal performance. Therefore, data augmentation and cleansing scheme are applied to enhance the performance. For example, ORION (Fu et al., 2025) generated Visual Question Answering (VQA) to enhance their Vision-Language-Action (VLA) models' capability, such as scene description, behavior description, meta-driving decision and reasoning, and recall of essential historical information. Chitta et al. (2023) and Zimmerlin et al. (2024) use PDM-lite (Beißwenger, 2024; Sima et al., 2025), an open source rule-based expert to collect ground-truth trajectories for imitation learning. SimLingo (Renz et al., 2025) generates additional driving data and applies intricate filtration on training routes of Chitta et al. (2023) and the official CARLA LB 2.0 routes. We use the datasets proposed by (Zimmerlin et al., 2024). The dataset actively filters for critical change frames and refines expert behavior, thereby focusing on high-quality decision-making moments while re-

Table 1: Comparison of BridgeDrive to previous baselines on Bench2Drive. Our method shows SOTA performance on both Driving Score (DS) and Success Rate (SR). Notably, by using a principled diffusion bridge model, our method achieves significant improvements over previous diffusion baselines (including those with prior knowledge from VLA), demonstrating the effectiveness of the diffusion module in the autonomous driving task when following our paradigm as discussed in Section 3.2. A potential avenue to further improve our method is to integrate prior knowledge from VLA, which is left as future work.

Method	Expert	VLA	Diffusion	DS	SR(%)
DiffusionDrive^{temp} (Liao et al., 2025)	PDM-Lite	✘	✔	77.68	52.72
TCP-traj* (Wu et al., 2022)	Think2Drive	✘	✘	59.90	30.00
UniAD-Base (Hu et al., 2023)	Think2Drive	✘	✘	45.81	16.36
VAD (Jiang et al., 2023)	Think2Drive	✘	✘	42.35	15.00
DriveTransformer (Jia et al., 2025)	Think2Drive	✘	✘	63.46	35.01
ORION (Fu et al., 2025)	Think2Drive	✔	✘	77.74	54.62
ORION diffusion (Fu et al., 2025)	Think2Drive	✔	✔	71.97	46.54
SimLingo (Renz et al., 2025)	PDM-Lite	✔	✘	85.07	67.27
TransFuser++ (Zimmerlin et al., 2024)	PDM-Lite	✘	✘	84.21	67.27
BridgeDrive (ours)	PDM-Lite	✘	✔	86.87(+1.8)	72.27(+5.0)

ducing its size, which improves training efficiency and strengthens the model's learning in crucial driving scenarios. For more details please refer to Appendix D.1.

Baselines. We compare against the following baselines. *TCP-traj* (Wu et al., 2022) is a monocular camera-based method that jointly learns planning and direct control with a situation-based fusion. *UniAD* (Hu et al., 2023) is a unified end-to-end framework that integrates full-stack driving tasks through query-based interfaces. *VAD* (Jiang et al., 2023) is an end-to-end vectorized paradigm that models driving scenes with vectorized representations. *DriveTransformer* (Jia et al., 2025) employs task parallelism, sparse representation, and streaming to enable efficient cross-task knowledge transfer and temporal fusion. *ORION* (Fu et al., 2025) integrates a QT-Former for history aggregation, a reasoning large language model (LLM), and a VAE for planning. *Simlingo* (Renz et al., 2025) leverages VLA and achieves current SOTA performance on Bench2Drive. *TransFuser++* (Chitta et al., 2023) (Zimmerlin et al., 2024) (Jaeger et al., 2023) ranks second in the 2024 CARLA challenge and first on the Bench2Drive test routes. In addition, we adapt *DiffusionDrive* to Bench2Drive benchmark (denoted as DiffusionDrive^{temp}). Adaptation details are provided in Appendix D.2.

4.2 Main Results

Evaluation results on Bench2Drive benchmark are demonstrated in Table 1. For all models that exceeded previous SOTA result, we performed experiments with three random seeds to ensure reproducibility. The performance of our work significantly exceeds that of all previous work. In particular, our BridgeDrive outperform the SimLingo (Renz et al., 2025), the latest SOTA, by +1.8 and +5% in driving score and success rate, respectively. Moreover, BridgeDrive exhibits outstanding multi-ability capability, especially in Merging (+4.75), Overtaking (+1.12), and Traffic Sign (+6.50), resulting in overall improvement by +2.9 than SOTA, as shown in Table 4 in the Appendix. However, BridgeDrive demonstrates suboptimal performance in the Comfortness and Give Way metrics, suggesting a tendency toward frequent or poorly timed braking. This outcome may imply that our model prioritizes safety considerations, potentially at the expense of passenger comfort. Furthermore, the inference speed of BridgeDrive is suitable for real-time deployment, as detailed in Table 5.

4.3 Ablation Study and Qualitative Analysis

The primer focus of this paper is on the design and study of diffusion models for trajectory planning; therefore, we prioritize the most vital aspects that could influence the performance, namely 1) what

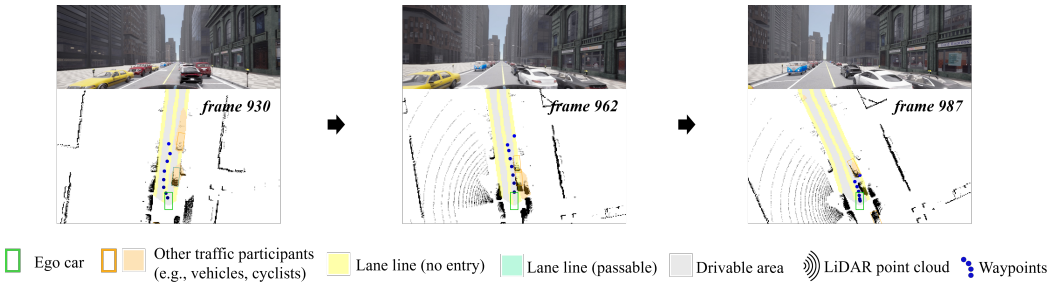

Figure 3: A consecutive four frames of a sample Bench2Drive scene. Overtaking maneuver performed by **BridgeDrive**$^{\text{temp}}$. The ego car exhibited deficiencies in overtaking maneuver coordination and speed control, which directly led to a collision with the white vehicle. For video demonstration, please refer to supplementary materials.

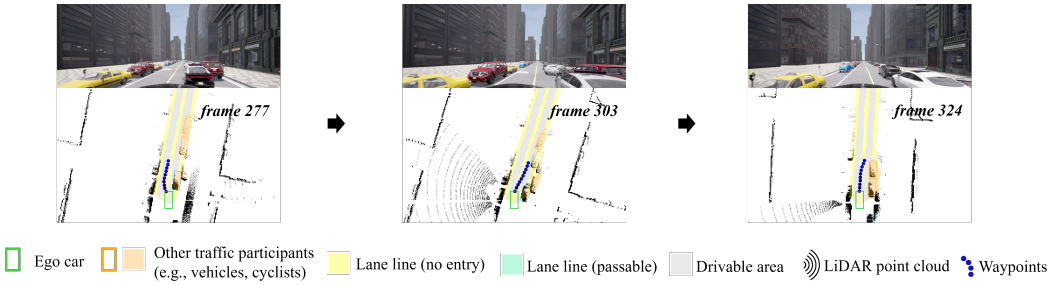

Figure 4: On the same scene as in Fig. 3, overtaking maneuver performed by **BridgeDrive**$^{\text{geo}}$. The ego vehicle adapts its planning to overtake a sequence of parked cars. For video demonstration, please refer to supplementary materials.

kind of trajectory representation is more compatible with diffusion model; 2) how our diffusion bridge policy with anchor guidance differs from other diffusion planners;

Effectiveness of the representation of geometric path waypoints. To account for the influence of different representations of the trajectory, namely the temporal speed waypoints vs. geometric path waypoints, we implement these two configurations for each version of diffusion models, denoted as *temp* and *geo*, respectively. It should be noted that, for DiffusionDrive$^{\text{geo}}$, all modules remain identical to those of our method except for the diffusion part to ensure a fair comparison. The results are compared in Table 2. It can be seen that the representation of geometric path waypoints outperforms their temporal counterpart, with an improvement of +5.46%, +9.09%, +12.37% in the success rate for DiffusionDrive, Full Diffusion, BridgeDrive, respectively. We argue the main reasons for this are as follows. 1) Temporal waypoints encode speed control information in the spacing between subsequent waypoints. Such an encoding is ambiguous and difficult to generalize. For example, for overtaking maneuvers with different speeds, geometric waypoints only require a model to learn the similar geometric pattern of driving path plus a varying speed scalar. In comparison, the generalization of temporal waypoints require a model to stretch spacing between waypoints to account for different speeds. 2) Geometric waypoints are more compliant with route topology and is therefore less likely to violate route lane constrain; similar arguments are also provided in (Jaeger et al., 2023).

The advantage of the denoising diffusion bridge model. As illustrated in Table 2, benefiting from the multi-modality of diffusion models, both Full Diffusion$^{\text{geo}}$ and BridgeDrive$^{\text{geo}}$ outperform DiffusionDrive$^{\text{geo}}$ by a large margin. In addition, compared with full diffusion, BridgeDrive further leverages anchor information to guide its diffusion process. This is of particular importance when facing ambiguous situations. An example is visualized in Fig. 5 and Fig. 6. In this case, the target point for lane change is given in a short distance ahead of the ego vehicle. Due to inherent inertial of the ego car, it is unlikely for full diffusion to change lane. Therefore, the ego car kept traveling in a straight path and missed the target point; subsequently, the ego car was unable to make a sharp turn to the left lane and hit the road barrier. In comparison, BridgeDrive, under the strong guidance of the anchor, was able to strictly follow the target point and entered the correct lane at the road fork.

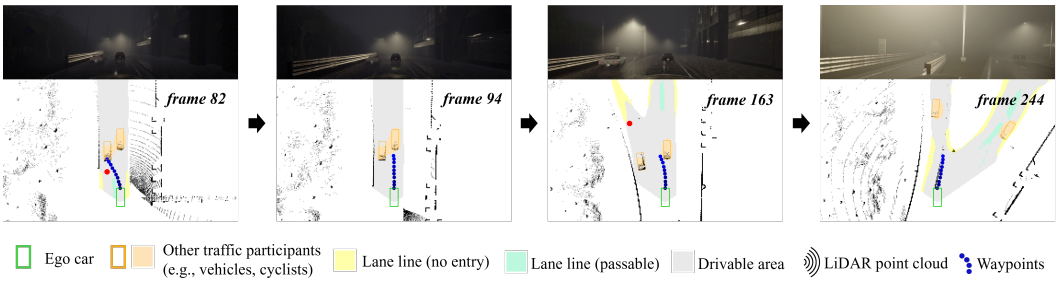

Figure 5: The **Full Diffusion model** in a consecutive four frames of a sample Bench2Drive scene. It **failed** to adhere to the target time window for lane-changing maneuvers, which consequently led to a collision with the road barrier. For video demonstration, please refer to supplementary materials.

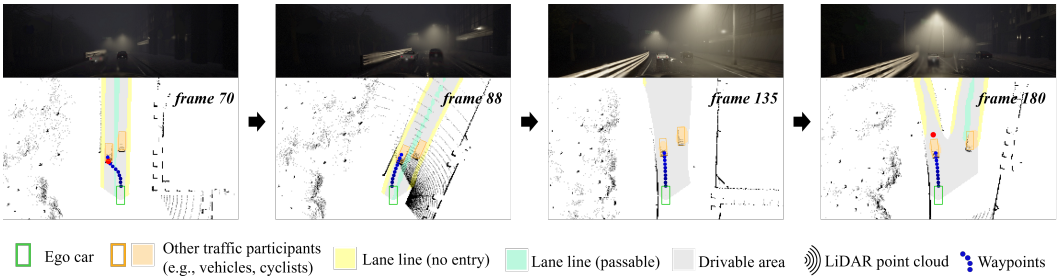

Figure 6: **BridgeDrive** in the same scene as in Fig. 5. The BridgeDrive model **achieved** timely lane changing due to anchor guidance and successfully navigated through the road fork. For video demonstration, please refer to supplementary materials.

Table 2: Ablation study for the effects of temporal and geometric path waypoints for DiffusionDrive, full diffusion, and BridgeDrive. All methods use identical modules except for the diffusion part. Our BridgeDrive^{geo} achieves SOTA DS and SR, prioritizing safety over Comfortness.

Configuration	Principled	Anchor	DS	SR(%)	Efficiency	Comfortness
DiffusionDrive^{temp}	✘	✔	77.68	52.72	248.18	24.56
DiffusionDrive^{geo}	✘	✔	80.79	58.18	245.34	15.49
Full Diffusion^{temp}	✔	✘	79.75	58.18	246.31	24.42
Full Diffusion^{geo}	✔	✘	83.85	67.27	238.90	21.40
BridgeDrive^{temp}	✔	✔	81.97	59.90	243.88	22.61
BridgeDrive^{geo}	✔	✔	86.86 $\pm$ 0.88	72.27 $\pm$ 1.37	238.74 $\pm$ 2.50	17.70 $\pm$ 0.80

5 Conclusions and Future Work

We presented BridgeDrive–an autonomous driving solution based on diffusion bridge policy. Our method provides a principled bridge diffusion framework incorporating anchor guidance and outperformed prior work by 5% in success rate. Extensive experiments validated that BridgeDrive yielded significant performance improvements in closed-loop planning tasks.

Limitations and future work. (1) While some existing methods only relies on camera input (e.g., Renz et al. (2025)), BridgeDrive also requires lidar input. Future work is expected to investigate BridgeDrive's capability in the absence of lidar. (2) Although the inference speed of BridgeDrive is suitable for real-time deployment, further acceleration can be achieved by distilling our model into a one-step model without sacrificing the generation quality (Xie et al., 2024). (3) Despite BridgeDrive's extraordinary capacity to learn complex planning tasks, it still struggles to handle the out-of-distribution scenario, as illustrated in Appendix C.3. This limitation could be overcome by incorporating prior knowledge from VLA and post-training with reinforcement learning.

Reproducibility Statement

For all models that exceeded previous SOTA result, we performed experiments with three random seeds to ensure reproducibility. Code will be made available upon publication of this work.

References

Jens Beißwenger. PDM-lite: A rule-based planner for Carla leaderboard 2.0. Technical report, University of Tübingen, 2024.

Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*, 2019.

Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. MultiPath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 86–99. PMLR, 2020.

Shaoyu Chen, Bo Jiang, Hao Gao, Bencheng Liao, Qing Xu, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Vadv2: End-to-end vectorized autonomous driving via probabilistic planning. *arXiv preprint arXiv:2402.13243*, 2024.

Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. TransFuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

Daniel Dauner, Marcel Hallgarten, Tianyu Li, Xinshuo Weng, Zhiyu Huang, Zetong Yang, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, Andreas Geiger, and Kashyap Chitta. NAVSIM: Data-driven non-reactive autonomous vehicle simulation and benchmarking. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Joseph L Doob and JI Doob. *Classical potential theory and its probabilistic counterpart*, volume 262. Springer, 1984.

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In Sergey Levine, Vincent Vanhoucke, and Ken Goldberg (eds.), *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pp. 1–16. PMLR, 2017.

Haoyu Fu, Diankun Zhang, Zongchuang Zhao, Jianfeng Cui, Dingkang Liang, Chong Zhang, Dingyuan Zhang, Hongwei Xie, Bing Wang, and Xiang Bai. ORION: A holistic end-to-end autonomous driving framework by vision-language instructed action generation. *arXiv:2503.19755*, 2025.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, Lewei Lu, Xiaosong Jia, Qiang Liu, Jifeng Dai, Yu Qiao, and Hongyang Li. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17853–17862, 2023.

Bernhard Jaeger, Kashyap Chitta, and Andreas Geiger. Hidden biases of end-to-end driving models. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2023.

Xiaosong Jia, Zhenjie Yang, Qifeng Li, Zhiyuan Zhang, and Junchi Yan. Bench2Drive: Towards multi-ability benchmarking of closed-loop end-to-end autonomous driving. In *NeurIPS 2024 Datasets and Benchmarks Track*, 2024.

Xiaosong Jia, Junqi You, Zhiyuan Zhang, and Junchi Yan. DriveTransformer: Unified transformer for scalable end-to-end autonomous driving. In *International Conference on Learning Representations*, 2025.

Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. VAD: Vectorized scene representation for efficient autonomous driving. *ICCV*, 2023.

Xuefeng Jiang, Yuan Ma, Pengxiang Li, Leimeng Xu, Xin Wen, Kun Zhan, Zhongpu Xia, Peng Jia, XianPeng Lang, and Sheng Sun. TransDiffuser: End-to-end trajectory generation with decorrelated multi-modal representation for autonomous driving. *arXiv preprint arXiv:2505.09315*, 2025.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

Qifeng Li, Xiaosong Jia, Shaobo Wang, and Junchi Yan. Think2Drive: Efficient reinforcement learning by thinking in latent world model for quasi-realistic autonomous driving (in CARLA-v2). In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024a.

Zhenxin Li, Kailin Li, Shihao Wang, Shiyi Lan, Zhiding Yu, Yishen Ji, Zhiqi Li, Ziyue Zhu, Jan Kautz, Zuxuan Wu, Yu-Gang Jiang, and Jose M. Alvarez. Hydra-MDP: End-to-end multimodal planning with multi-target hydra-distillation. *arXiv preprint arXiv:2406.06978*, 2024b.

Bencheng Liao, Shaoyu Chen, Haoran Yin, Bo Jiang, Cheng Wang, Sixu Yan, Xinbang Zhang, Xiangyu Li, Ying Zhang, Qian Zhang, et al. DiffusionDrive: Truncated diffusion model for end-to-end autonomous driving. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 12037–12047, 2025.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017a.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017b.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in neural information processing systems*, 35:5775–5787, 2022.

Katrin Renz, Long Chen, Elahe Arani, and Oleg Sinavski. Simlingo: Vision-only closed-loop autonomous driving with language-action alignment. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Abbas Sadat, Sergio Casas, Mengye Ren, Xinyu Wu, Pranaab Dhawan, and Raquel Urtasun. Perceive, predict, and plan: Safe motion planning through interpretable semantic representations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Jens Beißwenger, Ping Luo, Andreas Geiger, and Hongyang Li. DriveLM: Driving with graph visual question answering. In *European Conference on Computer Vision (ECCV)*, pp. 256–274. Springer, 2025.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b.

Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

Penghao Wu, Xiaosong Jia, Li Chen, Junchi Yan, Hongyang Li, and Yu Qiao. Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline. In *Advances in Neural Information Processing Systems*, 2022.

Sirui Xie, Zhisheng Xiao, Diederik Kingma, Tingbo Hou, Ying Nian Wu, Kevin P Murphy, Tim Salimans, Ben Poole, and Ruiqi Gao. EM distillation for one-step diffusion models. *Advances in Neural Information Processing Systems*, 37:45073–45104, 2024.

Zebin Xing, Xingyu Zhang, Yang Hu, Bo Jiang, Tong He, Qian Zhang, Xiaoxiao Long, and Wei Yin. GoalFlow: Goal-driven flow matching for multimodal trajectories generation in end-to-end autonomous driving. *arXiv preprint arXiv:2503.05689*, 2025.

Zhenhua Xu, Yan Bai, Yujia Zhang, Zhuoling Li, Fei Xia, Kwan-Yee K. Wong, Jianqiang Wang, and Hengshuang Zhao. DriveGPT4-V2: Harnessing large language model capabilities for enhanced closed-loop autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Brian Yang, Huangyuan Su, Nikolaos Gkanatsios, Tsung-Wei Ke, Ayush Jain, Jeff Schneider, and Katerina Fragkiadaki. Diffusion-ES: Gradient-free planning with diffusion for autonomous driving and zero-shot instruction following. *arXiv preprint arXiv:2402.06559*, 2024.

Zhejun Zhang, Alexander Liniger, Dengxin Dai, Fisher Yu, and Luc Van Gool. End-to-end urban driving by imitating a reinforcement learning coach. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Kaiwen Zheng, Guande He, Jianfei Chen, Fan Bao, and Jun Zhu. Diffusion bridge implicit models. In *The Thirteenth International Conference on Learning Representations*, 2025a.

Yinan Zheng, Ruiming Liang, Kexin ZHENG, Jinliang Zheng, Liyuan Mao, Jianxiong Li, Weihao Gu, Rui Ai, Shengbo Eben Li, Xianyuan Zhan, and Jingjing Liu. Diffusion-based planning for autonomous driving with flexible guidance. In *The Thirteenth International Conference on Learning Representations*, 2025b.

Linqi Zhou, Aaron Lou, Samar Khanna, and Stefano Ermon. Denoising diffusion bridge models. In *The Twelfth International Conference on Learning Representations*, 2024.

Julian Zimmerlin, Jens Beißwenger, Bernhard Jaeger, Andreas Geiger, and Kashyap Chitta. Hidden biases of end-to-end driving datasets. *arXiv preprint arXiv:2412.09602*, 2024.

Table 3: Comprehensive comparison between BridgeDrive and Baselines. BridgeDrive prioritizes safety over Comfortness.

Method	Expert	Key technique	DS	SR(%)	Effi.	Comfort.
DiffusionDrive$^{\text{temp}}$	PDM-Lite	Diffusion	77.68	52.72	248.18	24.56
TCP-traj*	Think2Drive	CNN, MLP, GRU	59.90	30.00	76.54	18.08
UniAD-Base	Think2Drive	Transformer	45.81	16.36	129.21	43.58
VAD	Think2Drive	Transformer	42.35	15.00	157.94	46.01
DriveTransformer	Think2Drive	Transformer	63.46	35.01	100.64	20.78
ORION	Think2Drive	VLA+VAE	77.74	54.62	151.48	17.38
ORION diffusion	Think2Drive	VLA+Diffusion	71.97	46.54	NA	NA
SimLingo	PDM-Lite	VLA	85.07	67.27	259.23	33.67
TransFuser++	PDM-Lite	Transformer	84.21	67.27	NA	NA
BridgeDrive (ours)	PDM-Lite	Diffusion	86.87(+1.8)	72.27 (+5.0)	238.74	17.70

A Declaration

We used large language models (LLMs) to polish writing for part of this paper.

B Related Work

End-to-end autonomous driving. Traditional motion planning pipelines often decompose the task into separate stages—perception, prediction, and planning—which inevitably introduce latency and information degradation across modules (Sadat et al., 2020). To overcome these limitations, recent studies have shifted toward planning-centric, end-to-end autonomous driving frameworks. End-to-end autonomous driving aims to map raw sensory inputs directly to trajectory predictions or control commands, enabling holistic system optimization that mitigates error propagation across modules (Wu et al., 2022; Zhang et al., 2021). UniAD (Hu et al., 2023) shows the feasibility of end-to-end autonomous driving by unifying multiple perception tasks to benefit planning. Building on this, VAD (Jiang et al., 2023) introduces compact vectorized scene representations to boost efficiency. VADv2 (Chen et al., 2024) proposes a probabilistic planning framework that models the distribution over possible actions and samples one for vehicle control. SimLingo (Renz et al., 2025) and GPTDriverV2 (Xu et al., 2025) incorporate vision-language understanding and language-action alignment, aiming to enhance closed-loop driving performance.

Deterministic planners. Some end-to-end autonomous driving planners relies on models such as multilayer perceptrons (MLPs) or variational autoencoders (VAEs). Transfuser (Chitta et al., 2023) and its extension Transfuser++ (Jaeger et al., 2023) exemplify this line of work by fusing multi-modal sensor inputs—such as camera images and LiDAR point clouds—through transformer-based encoders and decoding them into trajectory outputs via compact MLP heads. These models highlight the importance of effective sensor fusion in improving closed-loop driving performance. ORION (Fu et al., 2025) adopts a VAE-based latent planning architecture, which enables the model to capture multi-modal trajectory distributions while maintaining computational efficiency. These methods demonstrate how MLP and VAE-style architectures can serve as efficient baselines for end-to-end planning, though they often face limitations in modeling the full multi-modality of human driving behaviors compared to generative paradigms such as diffusion or flow-based models.

Diffusion-based planners. Diffusion policies provide a generative paradigm which can model the multi-modal nature of human driving behaviors with enhanced guidance control. Diffusion-ES (Yang et al., 2024) exhibits zero-shot instruction-following ability in planning. Diffusion-Planner (Zheng et al., 2025b) uses joint prediction modeling to achieve safe and adaptive planning. GoalFlow (Xing et al., 2025) leverages flow matching to produce diverse goal-conditioned trajectories and further uses a trajectory scorer to efficiently select trajectory using the goal point as a reference. DiffusionDrive (Liao et al., 2025) points out the issue of mode collapse, wherein the generated

Table 4: Multi-Ability evaluation results. BridgeDrive outperforms all baselines in all categories except for Give Way.

Method	Ability (%)					
	Merg.	Overtak.	Emer. Brake	Give Way	Traf. Sign	Mean
DiffusionDrive$^{\textbf{temp}}$	50.63	26.67	68.33	50.00	76.32	54.38
TCP-traj*	8.89	24.29	51.67	40.00	46.28	34.22
UniAD-Base	14.10	17.78	21.67	10.00	14.21	15.55
VAD	8.11	24.44	18.64	20.00	19.15	18.07
DriveTransformer	17.57	35.00	48.36	40.00	52.10	38.60
ORION	25.00	71.11	78.33	30.00	69.15	54.72
SimLingo	54.01	57.04	88.33	53.33	82.45	67.03
TransFuser++	58.75	57.77	83.33	40.00	82.11	64.39
BridgeDrive (ours)	63.50 (+4.75)	58.89 (+1.12)	88.34 (+0.01)	50.00 (-3.33)	88.95 (+6.50)	69.93 (+2.9)

Table 5: Inference wall-clock time comparison. The full diffusion model is approximately half the size of BridgeDrive, as it does not have cross-attention modules for anchors. BridgeDrive achieves reasonable inference speed even without any additional optimization, indicating its suitability for real-time deployment.

Configuration	Inference speed per frame	#Diffusion timesteps
DiffusionDrive$^{\text{temp}}$	0.05 sec	2
DiffusionDrive$^{\text{geo}}$	0.05 sec	2
Full Diffusion$^{\text{temp}}$	0.05 sec	100
Full Diffusion$^{\text{geo}}$	0.05 sec	100
BridgeDrive$^{\text{temp}}$	0.10 sec	20
BridgeDrive$^{\text{geo}}$	0.10 sec	20

trajectories lack diversity, as different random noise inputs tend to converge to similar trajectories during the denoising process, proposing truncated diffusion policy that begins the denoising process from an anchored gaussian distribution instead of a standard Gaussian distribution to avoid mode collapse. TransDiffuser (Jiang et al., 2025) emphasizes another underlying bottleneck that leads to mode collapse in generated trajectories: The under-utilization of the encoded multi-modal conditional information. Therefore, it implements multi-modal representation decorrelation optimization mechanism during the denoising process, which aims to better exploit the multi-modal representation space to guide more diverse feasible planning trajectories from the continuous action space.

C Additional Results and Visualization

C.1 Comparison with existing works

A comprehensive evaluation on Bench2Drive metrics is provided in Tables 3 and 4. Our method shows SOTA performance on both Driving Score (DS) and Success Rate (SR). Moreover, BridgeDrive exhibits outstanding multi-ability capability, especially in Merging (+4.75), Overtaking (+1.12), and Traffic Sign (+6.50), resulting in overall improvement by +2.9 than SOTA. However, BridgeDrive demonstrates suboptimal performance in the Comfortness and Give Way metrics, suggesting a tendency toward frequent or poorly timed braking. This outcome may imply that our model prioritizes safety considerations, potentially at the expense of passenger comfort. This limitation should be addressed in the future work.

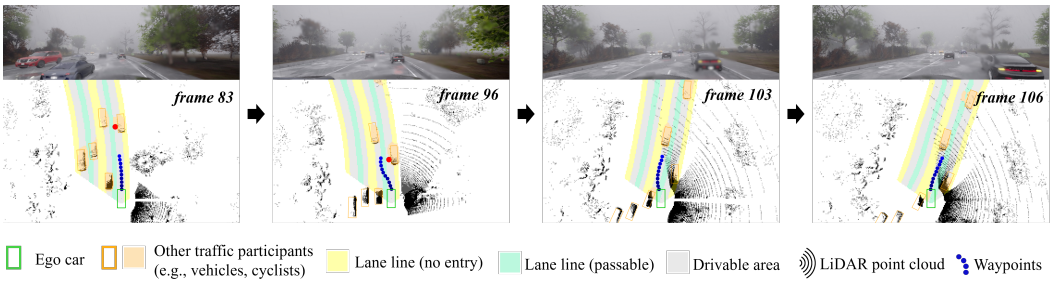

Figure 7: BridgeDrive cannot handle imperfect timing of lane-changing, which resulted from cumulative errors. Such an situation is out of distribution of training data.

C.2 Inference Speed

Inference wall-clock time comparison is detailed in Table 5. Note that the full diffusion model is slightly faster than BridgeDrive since it is approximately half the size of BridgeDrive. This is because full diffusion does not use anchors and thus omit all anchor-related cross-attention modules. BridgeDrive achieves reasonable inference speed even without any additional optimization, indicating its suitability for real-time deployment.

C.3 Case Study for an Out-of-distribution Scenario

Despite the extraordinary modeling capacity of BridgeDrive, it cannot generalize to out-of-distribution scenarios, which is very common in closed-loop evaluation. For instance, the overtaking maneuver shall be aborted if oncoming vehicles are present in the adjacent lane. However, there are almost no such data in the training set. The reason is that training data are collected by a rule-based expert with privileged information (i.e., the expert has direct access to the ground truth of other traffic participants' location and dynamics). This expert has long-term planning capability and will only perform an overtaking maneuver when there is sufficient longitudinal space in adjacent lanes. Such an ideal timing is not always feasible in closed-loop evaluation due to cumulative difference between predicted and ground-truth speed. An example of imperfect timing for lane changing is provided in Fig. 7. This limitation may be overcome by integrating scene understanding prior knowledge from VLA into BridgeDrive or posting-training with reinforcement learning, which is left for future work.

D Experiment Details

D.1 Datasets filtering and augmentation

Zimmerlin et al. (2024) proposed a data filtering method to reduce redundancy in training datasets. The method involves keeping frames where significant changes occur compared to the previous frame. Specifically, a frame is retained if either of the following conditions is met.

- The target speed changes by more than 0.1 m/s.
- The angle to any predicted geometric path waypoints changes by more than $0.5°$.

From the remaining frames, 14% are randomly selected and kept. This strategy results in a 50% reduction in the dataset size.

Additionally, the authors adjust the expert's driving style by modifying behaviors, such as the timing of braking when approaching pedestrians. This adjustment ensures the expert's actions are more interpretable and provide clearer learning signals for the model. Furthermore, the paper removes class weights for target speed values, particularly for over-represented classes like braking, to avoid biasing the model towards more frequent behaviors. This ensures the model learns from frames critical for driving tasks, rather than focusing on frequent but less important ones.

Table 6: Comparison among the major modules in the model architectures of DiffusionDrive, DiffusionDrive$^{\text{temp}}$, DiffusionDrive$^{\text{geo}}$, and BridgeDrive.

Module	DiffusionDrive	DiffusionDrive$^{\text{temp}}$	DiffusionDrive$^{\text{geo}}$	BridgeDrive
Perception	Transfuser	Transfuser++	Transfuser++	Transfuser++
Classifier	Transfuser	Transfuser++	Transfuser++	BridgeDrive
Denoiser	DiffusionDrive	DiffusionDrive	DiffusionDrive	BridgeDrive
Output	Temporal waypoints	Temporal waypoints	Geometric waypoints	Geometric waypoints

D.2 Adaptation of DiffuionDrive to Bench2Drive Benchmark

We denote adapted version of DiffusionDrive as DiffusionDrive$^{\text{temp}}$. We explain our adaption from 4 aspects.

Perception module. The original DiffusionDrive was built on the backbone of Transfuser Chitta et al. (2023), whereas our BridgeDrive is based on Transfuser++ (Jaeger et al., 2023). To ensure a competitive baseline for fair comparison, for the adaptation of DiffusionDrive, we use the perception module from Transfuser++, which is proven to achieve SOTA on Bench2Drive benchmark (Zimmerlin et al., 2024).

Denoiser. We keep the model architecture of the denoiser identical to its original design as it is unique to each model under comparison.

Classifier. The classifier in DiffusionDrive consists of cross-attention modules to process the perception features. We keep its architecture in line with the perception module.

Output. The output trajectory representation of DiffusionDrive$^{\text{temp}}$ is temporal waypoints, which is kept the same as DiffusionDrive. The analysis of the impact of output representation is provided in Section 4.3, where DiffusionDrive$^{\text{temp}}$'s geometric waypoints counterpart DiffusionDrive$^{\text{geo}}$ is implemented and evaluated.

An overview comparing the architectures of the major modules across DiffusionDrive, DiffusionDrive$^{\text{temp}}$, and BridgeDrive is provided in Table 6. The rest of the implementation and training details of DiffusionDrive$^{\text{temp}}$ are kept the same as BridgeDrive for a fair comparison.

D.3 Implementation and Training Details

As mentioned in Section 3.4, our model consists of three modules. For perception module, we keep the original design as described in Jaeger et al. (2023) and (Zimmerlin et al., 2024). Once the perception module is pre-trained, it is frozen during the training phase of the denoiser and classifier modules. The joint loss for the denoiser and classifier is defined as:

$$L_{\text{overall}} = w_{\text{diffusion}} L_{\text{diffusion}} + w_{\text{classification}} L_{\text{classification}}, \tag{11}$$

where $L_{\text{diffusion}}$ is as defined in Eq. (9) and $L_{\text{classification}}$ is the cross-entropy loss. By default, both $w_{\text{diffusion}}$ and $w_{\text{classification}}$ are set to 1. For these modules, we use the AdamW optimizer (Loshchilov & Hutter, 2017a) with a cosine annealing learning schedule (Loshchilov & Hutter, 2017b) for optimization. The learning rate is set as $\text{lr}_0 = 3 \times 10^{-4}$, $T_0 = 10$, $T_{mult} = 2$. In line with DiffusionDrive, we use $N_{\text{anchor}} = 20$ anchors in BridgeDrive. Our models are trained for 10 epochs on a single H20 GPU, which takes around 10 hours.

For diffusion schedule, we employ the variance preserving (VP) schedule from Karras et al. (2022). Specifically, we first define $s(t) = 1/\sqrt{e^{\beta_d t^2/2 + \beta_{min} t}}$ and $\sigma(t) = \sqrt{e^{\beta_d t^2/2 + \beta_{min} t} - 1}$. We then set the diffusion coefficients in the forward diffusion bridge SDE Eq. (5) to $f(t) = \dot{s}(t)/s(t)$ and $g(t) = \sqrt{2s(t)^2 \dot{\sigma}(t)\sigma(t)}$. We choose $\beta_d = 2.0$ and $\beta_{min} = 0.1$ following Zheng et al. (2025a).