Large Discrete Policy: Advancing Explicit Behavior Modeling with Stochastic Iterative Scoring

Zhenxin Li[1,2*], **Nadine Chang**[3], **Xinglong Sun**[3], **Jingde Chen**[3], **Wenhao Yao**[1,2], **Zi Wang**[3]
Maying Shen[3], **Yu-Gang Jiang**[1,2†], **Zuxuan Wu**[1,2†], **Shiyi Lan**[3], **Jose M. Alvarez**[3]

[1]Institute of Trustworthy Embodied AI, Fudan University
[2]Shanghai Key Laboratory of Multimodal Embodied AI [3]NVIDIA

Abstract

Behavior policies are often formulated as continuous generative models, whose iterative denoising processes are expressive but difficult to interpret and prone to producing implausible actions. We propose the Large Discrete Policy (LDiP), a fully discrete behavior modeling framework that selects actions from a large vocabulary of physically plausible candidates. Rather than perturbing actions, LDiP improves expressivity through stochastic iterative scoring: it progressively re-scores and prunes candidates with score-space stochasticity, enabling fine-grained ranking and exploration among plausible actions while preserving an explicit decision process. Across end-to-end planning, closed-loop driving, robotic manipulation, and vision-language-action settings, LDiP consistently outperforms strong discrete and continuous baselines in autonomous driving, and exceeds or matches continuous generative policies in robotic manipulation. These results show that discrete policies, when equipped with effective scoring mechanisms, offer an expressive, plausible, and interpretable alternative for behavior modeling. Project website: https://zhenxinli.net/LargeDiscretePolicy/.

1 Introduction

Behavior learning is a central problem in physical AI systems, with applications ranging from autonomous driving [18] to robotic manipulation [56, 9]. The goal is to learn a policy that models behavior conditioned on visual observations, which is commonly achieved by leveraging offline demonstrations from expert agents [49].

Since the action space in the physical world is typically continuous, behavior policies are commonly formulated as continuous models, including direct regression [18, 56] and generative models [17, 9, 37, 4]. Recently, iterative denoising models (e.g. Diffusion Models [17, 9], Flow-Matching Models [37, 4]) demonstrate strong capacity for modeling multimodal behavior distributions [9], making them a popular choice in both autonomous driving [35] and robotics [9, 4]. Nevertheless, their generation process is largely implicit: actions emerge through repeated refinement of noisy samples, making it difficult to interpret how a behavior is formed, as illustrated in Fig. 1 (a). Moreover, errors in the denoising process may lead to physically implausible actions [53].

Discrete policies provide an alternative. They are widely adopted in language modeling [50], where the action space is a finite vocabulary of symbols. Similar ideas have also been explored for behavior modeling, including trajectory prediction in autonomous driving [42, 43, 8, 31, 34] and robotic manipulation [43]. Instead of generating continuous actions, these methods explicitly select from a

*Work done during an internship at NVIDIA. †Corresponding authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

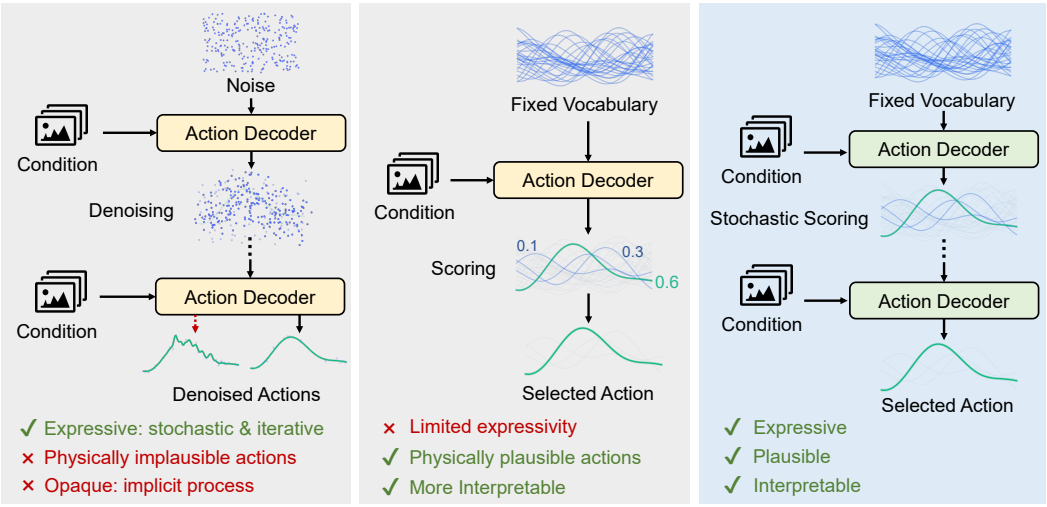

Figure 1: **Paradigms for Behavior Modeling Policies.** (a) Iterative denoising policies generate actions through stochastic iterative refinement, offering strong expressivity but remaining an implicit and less interpretable generation process. (b) One-shot discrete policies explicitly select from a fixed action vocabulary, but are limited by single-pass scoring. (c) Our approach combines the strengths of both paradigms by performing stochastic iterative scoring over a discrete action vocabulary, enabling expressive, plausible, and interpretable behavior generation.

discrete action vocabulary, as shown in Fig. 1 (b). To reduce the discretization gap, prior work either predicts continuous offsets from selected candidates [43], or constructs dense action vocabularies [42] with physically plausible actions. The latter approach, which we refer to as fully discrete policies, assigns scores to each action candidate. By evaluating all candidates explicitly, it provides better interpretability than the implicit denoising process.

Despite these advantages, fully discrete policies still suffer from limited expressivity. A common concern is that a fixed action vocabulary may be too rigid to cover all behavior modes. To address this, existing discrete policies augment candidates with action-level perturbations [55]. While effective, such perturbations blur the fully discrete formulation and may produce physically implausible actions [53, 57, 55]. On the other hand, recent work suggests that a sufficiently dense action vocabulary can already approximate continuous behavior spaces well [47]. This motivates a different hypothesis: the bottleneck may not be the lack of feasible action candidates, but the difficulty of accurately scoring and ranking many near-optimal candidates. Prior methods improve ranking with additional refinement modules [54], but they leave the key question unanswered: can we improve the expressivity of discrete policies with an advanced, general scoring mechanism, while keeping the action vocabulary fixed and physically plausible throughout the process?

We propose the **Large Discrete Policy** (LDiP), which establishes fully discrete action scoring as a general and expressive framework for behavior modeling. LDiP builds upon an advanced scoring mechanism, which preserves the explicitness and plausibility of discrete policies while substantially improving the behavior modeling ability.

To realize this goal, LDiP first constructs a large action vocabulary by clustering offline demonstrations [42, 31]. To facilitate fine-grained comparisons among near-optimal candidates, LDiP performs *iterative scoring*: action candidates are progressively pruned according to their scores until a single executable action remains, as shown in Fig. 1 (c). This iterative process makes the decision process explicit: it reveals which candidates are considered, how they are ranked, and how the final action is selected. To further address uncertainty and approximation errors in the scoring model [44], we introduce *stochastic scoring*, a novel mechanism that perturbs the scores during training rather than injecting noise into actions, which encourages exploration among plausible candidates while preserving the fully discrete formulation.

We benchmark LDiP on various behavior modeling tasks, including end-to-end planning for autonomous driving [13, 7], simulated closed-loop driving [58], and robotic manipulation [40]. LDiP consistently outperforms both discrete and continuous baselines in autonomous driving, while achieving performance stronger than or on par with the standard Diffusion Policy [9] in high-DoF manipulation tasks. Moreover, we extend LDiP by integrating it with Vision-Language Models (VLMs), yielding a Vision-Language-Action variant, **LDiP-VLA**. LDiP-VLA outperforms prior approaches, including diffusion-based [30] and flow-based [60] VLAs. Our main contributions are summarized as follows:

- We introduce the **Large Discrete Policy** (LDiP), a fully discrete behavior modeling framework. LDiP employs an iterative scoring process over a large action vocabulary, together with a novel stochastic scoring mechanism. These designs preserve the fully discrete formulation while enhancing policy expressivity, enabling both physically plausible and interpretable behavior modeling.
- We benchmark LDiP across end-to-end planning, closed-loop driving, and robotic manipulation, where it consistently outperforms strong discrete and continuous baselines, and matches or exceeds diffusion-based policies in high-DoF tasks. When extended with Vision-Language models, LDiP-VLA outperforms recent diffusion- and flow-based VLAs.

2 Related Work

2.1 Iterative Denoising Policies for Behavior Modeling

Iterative denoising policies have become a prominent paradigm for continuous behavior modeling. These methods formulate behavior modeling as an action generation problem. They are typically instantiated with diffusion models [17, 45, 11] or flow-matching models [37]. Diffusion Policy [9] first demonstrated the effectiveness of diffusion-based action generation for robotic manipulation, showing strong capability in modeling multimodal action distributions from visual observations. Similar ideas have since been adopted in autonomous driving, including the DiffusionDrive series [35, 63] for end-to-end planning with visual inputs, as well as planning-centric approaches [48, 57] that leverage privileged information (e.g. HD maps and 3D bounding boxes). More recently, the π series [4, 19] integrates a flow-based policy with pretrained Vision-Language Models, demonstrating promising open-world generalization. GR00T-N1 [3] similarly adopts diffusion-based action generation within a large-scale robot foundation model. While these approaches achieve strong empirical performance, their generation process remains largely implicit: actions are produced through repeated refinement of noisy samples, making it difficult to inspect why certain behaviors are considered and how the final behavior is generated.

2.2 Discrete Policies for Behavior Modeling

Discrete policies provide an alternative to continuous action generation by representing behaviors through a finite set of action candidates. Early work such as Behavior Transformers [43] uses K-Means to construct action cluster centers and predicts continuous offsets from the selected cluster to the target behavior. While this introduces a discrete structure, the final action remains continuous and the policy is therefore not fully discrete. In autonomous driving, a line of trajectory scoring methods [42, 8, 31, 32, 34, 33, 47] adopts a more explicit discrete formulation: they construct a vocabulary of trajectory candidates, often represented as waypoint sequences or action chunks, score each candidate under the current scene context, and select the highest-scoring trajectory for execution. More recent hierarchical trajectory scorers [54, 55] improve expressivity by first pruning a large candidate set and then refining a smaller subset. However, these methods often rely on additional refinement modules or inject noise directly into action candidates [55], which may corrupt physically plausible trajectories and blur the fully discrete formulation.

In the Vision-Language-Action literature, another family of methods discretizes actions into language tokens and predicts them through autoregressive generation, including manipulation models such as RT-1 [5], RT-2 [62], and OpenVLA [23], as well as autonomous driving models [59, 39, 12]. These approaches benefit from the token-generation interface of pretrained language models, but they do not explicitly model actions in the manner of trajectory scorers. Our work builds on the explicitness of fully discrete trajectory scoring, and extends it into a general behavior modeling framework with a large action vocabulary and stochastic iterative scoring. With these components, LDiP improves expressivity while preserving interpretability and physical plausibility.

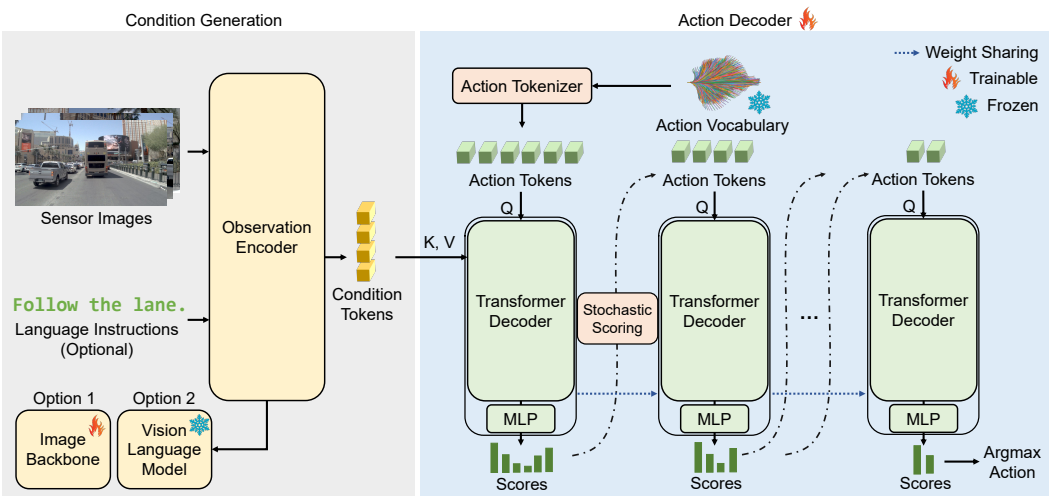

Figure 2: **The Model Architecture of LDiP.** LDiP first tokenizes a fixed action vocabulary into action tokens, which attend to visual observations and optional language instructions in a Transformer decoder. The model then scores all candidates with an MLP, stochastically prunes the vocabulary, and iteratively re-scores the remaining candidates until a single action is selected.

3 Method

3.1 Model Architecture

In this section, we describe the model architecture and learning procedure of LDiP, as illustrated in Fig. 2. We begin by introducing how LDiP constructs the fixed action vocabulary and performs scoring in a single forward pass.

Vocabulary Generation and Scoring. Following LSS [42], we formulate behavior generation as discrete scoring over an action vocabulary. Given an offline demonstration dataset, we first segment expert behaviors into fixed-horizon action chunks $\mathbf{a}_{1:H} \in \mathbb{R}^{H\times d}$, where H is the prediction horizon and d is the action dimension. We then apply K-Means clustering to construct a large discrete vocabulary $\mathcal{V}$ with K action candidates following [43, 42, 31]:

$$\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^{K} = \text{K-Means}\left(\{\mathbf{a}_{1:H}^{(n)}\}_{n=1}^{N}\right), \tag{1}$$

where each cluster center $\mathbf{v}_i$ corresponds to a physically plausible action chunk observed in the demonstration distribution. Given visual observations, and optionally language instructions in the VLA setting, an observation encoder produces condition tokens $\mathbf{C}$. Each action chunk $\mathbf{v}_i$ is mapped into an action token by an action tokenizer $\phi(\cdot)$, and serves as the query to a Transformer decoder [50]. The decoder attends to the condition tokens and predicts scores for candidate action chunks:

$$\mathbf{h}_i^{(t)} = \text{Dec}_\theta\left(\phi(\mathbf{v}_i^{(t)}), \mathbf{C}\right), \qquad s_i^{(t)} = g_\theta\left(\mathbf{h}_i^{(t)}\right), \tag{2}$$

where $\mathcal{V}^{(t)} = \{\mathbf{v}_i^{(t)}\}_{i=1}^{K^{(t)}}$ is the candidate set at the t-th scoring stage, and g_θ is an MLP scoring head.

Iterative Scoring. Instead of scoring the vocabulary only once, LDiP performs coarse-to-fine iterative scoring. The first stage evaluates the full vocabulary $\mathcal{V}^{(1)} = \mathcal{V}$. Subsequent stages keep only the top-ranked candidates from the previous stage,

$$\mathcal{I}^{(t+1)} = \text{TopK}\left(\mathbf{s}^{(t)}, K^{(t+1)}\right), \qquad \mathcal{V}^{(t+1)} = \{\mathbf{v}_i^{(t)} \mid i \in \mathcal{I}^{(t+1)}\}, \tag{3}$$

where $\mathcal{I}$ denotes the action indices. LDiP then re-scores this smaller candidate set with the shared action decoder (Dec_θ and g_θ). After repeating this process iteratively for T stages, the policy outputs the highest-scoring action chunk:

$$\hat{\mathbf{a}}_{1:H} = \mathbf{v}^{(T)}_{\arg\max_i s_i^{(T)}}. \tag{4}$$

This design preserves a fully discrete decision process while allowing the model to progressively prune the vocabulary to the most promising regions of the action space.

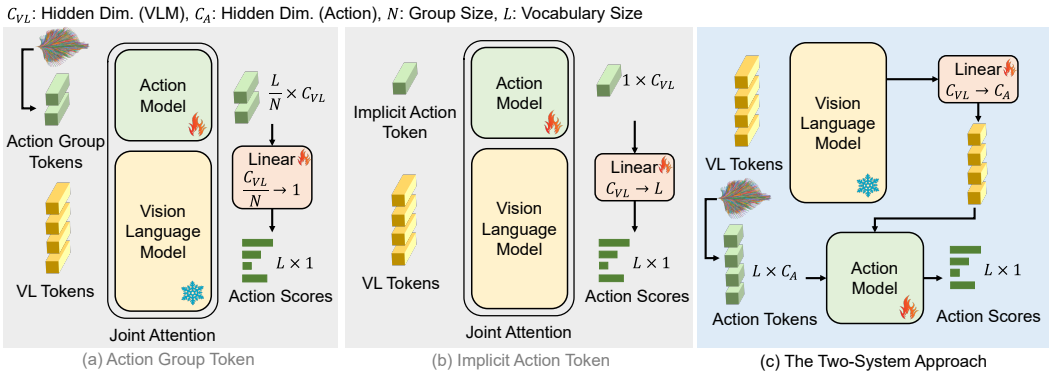

Figure 3: **Vision-Language-Action (VLA) Architecture Variants.**

Model Learning. For imitation learning, we follow the soft target formulation used in Hydra-MDP [31]. Given a ground-truth action chunk $\mathbf{a}^*_{1:H}$, each candidate receives a distance-based target score, which is converted into a soft-label distribution:

$$q_i^{(t)} = \frac{\exp\left(-\|\mathbf{v}_i^{(t)} - \mathbf{a}^*_{1:H}\|_2^2/\sigma\right)}{\sum_j \exp\left(-\|\mathbf{v}_j^{(t)} - \mathbf{a}^*_{1:H}\|_2^2/\sigma\right)}, \tag{5}$$

where σ is a hyperparameter that controls the sharpness of the distribution. The imitation objective at each stage is then formulated as the cross-entropy between the predicted logit scores $s^{(t)}$ and the soft label $q^{(t)}$:

$$\mathcal{L}_{\text{imi}}^{(t)} = -\sum_i q_i^{(t)} \log \frac{\exp(s_i^{(t)})}{\sum_j \exp(s_j^{(t)})}. \tag{6}$$

For robotic manipulation, we use only this imitation objective, summed over all scoring stages. For autonomous driving, following trajectory scorers [31, 27, 33, 47], we additionally attach multiple scoring heads to predict open-loop driving objectives $\mathcal{M}$ [13, 7], including collision avoidance, drivable-area compliance, and related driving criteria. The driving loss is

$$\mathcal{L}_{\text{drive}} = \sum_{t=1}^{T} \left(\lambda_{\text{imi}} \mathcal{L}_{\text{imi}}^{(t)} + \sum_{m \in \mathcal{M}} \lambda_m \mathcal{L}_m^{(t)} \right), \tag{7}$$

where $\mathcal{L}_m^{(t)}$ is implemented as binary cross-entropy between the predicted score and the ground-truth label. In this setting, the top-k selection in the iterative scoring process is based on a weighted average of multiple predicted scores, following Hydra-MDP [31].

Stochastic Scoring. During training, we introduce stochasticity in the pruning process rather than perturbing the action chunks themselves. Specifically, before top-k selection at intermediate stages, we perturb the original scores with noise sampled from a Gumbel distribution [20]:

$$\tilde{s}_i^{(t)} = s_i^{(t)} + \tau \epsilon_i, \qquad \epsilon_i \sim \text{Gumbel}(0, 1). \tag{8}$$

The next candidate set is selected by $\text{TopK}(\tilde{s}^{(t)}, K^{(t+1)})$ during training. For inference, we empirically find that using deterministic scores from the model yields the best performance in driving, while using noise-perturbed scores for selection improves online exploration in manipulation tasks, leading to improved manipulation performance. This score-space perturbation encourages exploration among plausible action candidates without corrupting the discrete vocabulary or producing physically implausible action candidates.

3.2 Vision-Language-Action (VLA) Architecture

Fig. 3 shows three variants for extending LDiP to the VLA setting:

- **Action Group Token.** This variant follows the π-style joint-attention design [4]: we tokenize the discrete action vocabulary into action group tokens, concatenate them with vision-language tokens

Table 1: **Performance on the Navtest v2 Benchmark, End-to-end Planning Models.**

Backbone	Method	Behavior Modeling	NC ↑	DAC ↑	DDC ↑	TLC ↑	EP ↑	TTC ↑	LK ↑	HC ↑	EC ↑	EPDMS ↑
-	Human Agent	-	100	100	99.8	100	87.4	100	100	98.1	90.1	90.3
-	Ego Status MLP	Regression	93.1	77.9	92.7	99.6	86.0	91.5	89.4	98.3	85.4	64.0
ResNet34	Transfuser [10]	Regression	96.9	89.9	97.8	99.7	87.1	95.4	92.7	**98.3**	87.2	76.7
	HydraMDP++ [27]	Discrete	97.2	97.5	99.4	99.6	83.1	96.5	94.4	98.2	70.9	81.4
	DriveSuprim [54]	Discrete	97.5	96.5	99.4	99.6	88.4	96.6	95.5	**98.3**	77.0	83.1
	ZTRS [33]	Discrete	97.9	**98.0**	**99.7**	**99.8**	80.1	96.9	94.9	98.1	79.0	83.2
	DiffusionDrive [35]	Diffusion	**98.4**	95.5	99.5	**99.8**	87.5	**97.5**	**96.9**	98.4	**87.7**	84.2
	LDiP (Ours)	Discrete	97.7	97.5	99.4	99.7	**88.5**	97.0	96.6	**98.3**	79.4	**84.7**
ViT-L	HydraMDP++ [27]	Discrete	98.5	98.5	99.5	99.7	87.4	97.9	95.8	98.2	75.7	85.6
	DriveSuprim [54]	Discrete	98.4	98.6	99.6	99.8	**90.5**	97.8	97.0	**98.3**	78.6	87.1
	GTRS-Dense [34]	Discrete	**99.0**	97.8	99.3	**99.9**	86.3	**98.3**	95.1	**98.3**	71.9	84.7
	ZTRS [33]	Discrete	98.2	**99.1**	**99.7**	99.8	86.9	97.5	96.6	98.2	78.2	86.2
	LDiP (Ours)	Discrete	98.7	**99.1**	**99.7**	99.8	90.3	98.0	**97.9**	98.2	**80.7**	**88.1**
V2-99	Diffusion Policy [9]	Diffusion	96.8	92.8	98.7	99.7	86.6	95.9	94.9	98.2	**83.7**	79.1
	HydraMDP++ [27]	Discrete	**98.4**	98.0	99.4	99.8	87.5	97.7	95.3	**98.3**	77.4	85.1
	DriveSuprim [54]	Discrete	97.8	97.9	99.5	**99.9**	90.6	97.1	96.6	**98.3**	77.9	86.0
	GTRS-Dense [34]	Discrete	97.6	98.5	99.5	**99.9**	89.5	97.2	96.8	97.2	57.2	84.0
	ZTRS [33]	Discrete	97.8	**99.4**	**99.8**	99.8	84.1	97.0	96.2	98.2	77.2	85.3
	LDiP (Ours)	Discrete	98.3	98.9	99.7	99.8	**91.0**	**97.9**	**97.5**	98.1	79.1	**87.8**

Table 2: **Performance on the Navtest v1 Benchmark, Vision-Language-Action Models.**

Method	VLM	Behavior Modeling	NC↑	DAC↑	TTC↑	C↑	EP↑	PDMS↑
Human	–	–	100	100	100	99.9	87.5	94.8
AutoVLA [59]	Qwen2.5-VL-3B [2]	Token Generation	98.4	95.6	**98.0**	99.9	81.9	89.1
ReCogDrive [30]	InternVL3-8B [61]	Diffusion	98.2	97.8	95.2	99.8	83.5	89.6
DriveVLA-W0 [29]	Emu3-8B [51]	Discrete	98.7	**99.1**	95.3	99.3	83.3	90.2
AdaThinkDrive [39]	InternVL3-8B [61]	Token Generation	98.4	97.8	95.2	**100**	84.4	90.3
SpanVLA [60]	Qwen2.5-VL-3B [2]	Flow Matching	**99.1**	97.1	95.2	**100**	86.3	90.3
DriveFine [12]	Lavida-8B [28]	Token Generation	98.8	98.6	96.2	**100**	86.9	91.8
LDiP-VLA (Ours)	Qwen3-VL-2B [1]	Discrete	99.0	98.1	98.6	98.2	**88.8**	**92.1**

from the VLM, and allow joint attention between action and vision-language representations. The output action-token features are then decoded into vocabulary scores through lightweight linear scoring heads, preserving the iterative scoring and pruning procedure in Sec. 3.

- **Implicit Action Token.** The second variant removes direct access to the action vocabulary from the VLM. Instead, the model receives a small number of learned implicit action tokens, which attend to vision-language tokens and are linearly decoded into scores over the current candidate set. This design maintains a small number of tokens and tests whether the VLA can learn action scoring without explicitly observing the vocabulary.
- **The Two-System Approach.** The final variant follows the design of GR00T-N1 [3]: a frozen VLM first encodes visual observations and language instructions into vision-language tokens, which are then projected to the action-model dimension and used as conditioning tokens for a separate action decoder. The action model performs the same action selection process as described above.

4 Experiments

4.1 Implementation Details

We train our autonomous driving models on the NAVSIM Navtrain split [13], which contains 103K scenes. For standard end-to-end planning models, we train with 24 NVIDIA A100 GPUs for 20 epochs using Adam [24] with a learning rate of 2×10^{-4} and a per-GPU batch size of 12. For VLAs, we train with 8 NVIDIA A100 GPUs for 8 epochs using Adam with a learning rate of 7.5×10^{-5} and a per-GPU batch size of 8. The action vocabulary contains 16,384 trajectories, each spanning 4 seconds at 10 Hz. Unless specified, the vocabulary is pruned from 16,384 to 512 and then to 16 candidates. On HUGSIM [58], we perform zero-shot closed-loop inference following [33]. For the controller variant, we truncate the predicted trajectory to 3 seconds and fix the heading conversion [25].

For robotic manipulation, we largely follow the training and evaluation protocol of Diffusion Policy [9], which is also our major baseline. We use action chunks with a horizon length of 16, 2 observation steps, and execute 8 steps per prediction. Models are trained with AdamW [38] using a learning rate of 1×10^{-4} and task-dependent batch sizes following the Diffusion Policy setup. The vocabulary size is 8,192 for PushT and Can, 16,384 for ToolHang due to its higher dexterity

Table 3: **Zero-shot Performance on the HUGSIM Benchmark.** UniAD* and VAD* are trained on nuScenes [6], which partially overlap with the evaluation scenarios. Other methods are evaluated with zero-shot transfer. † The controller is tuned for closed-loop driving.

Method	Easy		Medium		Hard		Extreme		Overall	
	RC ↑	HD-Score ↑	RC ↑	HD-Score ↑	RC ↑	HD-Score ↑	RC ↑	HD-Score ↑	RC ↑	HD-Score ↑
VAD* [21]	38.7	24.3	27.0	9.9	25.5	10.4	23.0	8.2	27.9	12.3
UniAD* [18]	58.6	48.7	41.2	29.5	**40.4**	**27.3**	26.0	14.3	40.6	28.9
LTF [10]	68.4	52.8	40.7	24.6	36.9	19.8	25.5	8.1	41.4	24.8
Diffusion Policy [9]	72.4	60.8	30.5	15.2	27.4	13.4	28.0	14.9	36.9	23.1
ZTRS [33]	74.8	66.9	46.1	37.9	31.6	21.1	20.0	9.8	42.0	32.9
GTRS-Dense [34]	62.0	57.8	40.5	35.2	23.9	16.2	13.7	5.8	34.5	28.2
LDiP (Ours)	80.5	71.1	38.7	22.8	30.7	16.1	**31.7**	**16.0**	43.0	28.6
LDiP (Ours) †	**85.5**	**82.0**	**50.3**	**40.7**	29.0	20.6	19.4	10.6	**44.7**	**36.7**

Table 4: **Performance on Visual-based Robotic Manipulation**. Following Diffusion Policy [9], we report averaged success rates over the last 10 checkpoints. We compare against the Transformer-based Diffusion Policy, as both methods adopt a Transformer Decoder for behavior modeling.

Task	DoF	IBC [15]	LSTM-GMM [40]	Diffusion Policy [9]	LDiP (Ours)
		Continuous			Discrete
Lift	6+1	0.73	0.96	**1.00**	0.99
Can	6+1	0.01	0.88	**0.98**	0.94
ToolHang	6+1	0.00	0.49	0.47	**0.52**
PushT	2	0.64	0.54	0.66	**0.76**

requirement, and 6,666 for Lift, corresponding to the number of available training trajectories. The remaining pruning stages are consistent with autonomous driving.

4.2 Benchmarks and Metrics

Autonomous driving. We evaluate autonomous driving on NAVSIM [13, 7] and HUGSIM [58]. NAVSIM contains 103K training scenes in Navtrain and 12k evaluation scenes in Navtest. For standard discrete end-to-end models, we report Navtest v2 with the Extended Predictive Driver Model Score (EPDMS) [13, 7, 27], which aggregates No-at-fault Collisions (NC), Drivable Area Compliance (DAC), Driving Direction Compliance (DDC), Traffic Light Compliance (TLC), Ego Progress (EP), Time-to-Collision (TTC), Lane Keeping (LK), History Comfort (HC), and Extended Comfort (EC). For VLA models, we follow recent VLA works and report Navtest v1 with PDMS. On the other hand, HUGSIM evaluates zero-shot closed-loop performance in photo-realistic scenarios built with 3D Gaussian Splatting [22]. It contains 345 scenarios from KITTI-360 [36], Waymo [46], nuScenes [6], and PandaSet [52]. Easy scenarios involve regular driving, Medium scenarios introduce inserted vehicles, and Hard/Extreme scenarios include aggressive vehicles. We report Route Completion (RC) and HD-Score, which jointly measure progress and closed-loop safety.

Robotic manipulation. We evaluate visual-based robotic manipulation following Diffusion Policy [9]. The benchmark includes three Robomimic tasks [40], Lift, Can, and ToolHang, where the policy controls a 6-DoF end-effector and gripper from visual observations. We also evaluate on PushT, a 2-DoF task that requires pushing a T-shaped block to a target pose.

4.3 Main Results

NAVSIM, End-to-end Planning. Tab. 1 shows that LDiP achieves the best EPDMS across all evaluated backbones, reaching 84.7 with ResNet34 [16], 88.1 with ViT-L [14], and 87.8 with V2-99 [26]. The gains are consistent across CNN- and Transformer-based architectures, indicating that LDiP is not tied to a particular visual encoder. Instead, its advantage comes from the action decoder: by scoring a large vocabulary iteratively and stochastically, LDiP outperforms both discrete trajectory scorers [27, 34, 33] and diffusion-based policies [35, 9] by a significant margin.

NAVSIM, Vision-Language-Action Models. Tab. 2 further shows that the action modeling strategy of LDiP transfers effectively to vision-language representations. Despite using a compact Qwen3-VL-2B [1] backbone, LDiP-VLA achieves the best overall PDMS of 92.1, outperforming existing VLA baselines. This result highlights an important property of LDiP: its scoring mechanism is robust across substantially different architectures, from perception-only planning models to Vision-Language-Action models. The improvement suggests that LDiP provides a transferable action modeling interface, rather than an architecture-specific enhancement.

Table 5: **Ablation Study on Key Components.**

Model Variants	NC ↑	DAC ↑	DDC ↑	TLC ↑	EP ↑	TTC ↑	LK ↑	HC ↑	EC ↑	EPDMS ↑
Hydra-MDP-$\mathcal{V}_{16384}$ [31]	**98.5**	98.7	98.9	**99.9**	88.5	**98.2**	97.0	**98.3**	**80.5**	86.2
+ Iterative Scoring	97.9	98.3	99.5	99.8	90.6	97.3	**98.1**	98.1	77.1	86.7
+ Improved Tokenizer & Cross-Attention Decoder	98.4	98.7	**99.7**	99.8	89.6	97.9	97.4	**98.3**	78.6	87.1
+ Stochastic Scoring	98.3	**98.9**	**99.7**	99.8	**91.0**	97.9	97.5	98.1	79.1	**87.8**
Hydra-MDP-$\mathcal{V}_{16384}$ [31]	**98.5**	98.7	98.9	**99.9**	88.5	**98.2**	97.0	**98.3**	**80.5**	86.2
+ Noise Injection on Top-8 Actions [55]	98.6	98.7	**99.7**	**99.9**	88.7	98.2	97.0	**98.3**	80.2	87.2

Table 6: **Ablation Study on VLAs.**

Model Variants	NC↑	DAC↑	TTC↑	C↑	EP↑	PDMS↑
(a) Action Group Token	97.9	96.5	97.3	98.2	84.9	87.9
+ Stochastic Iterative Scoring	97.6	95.9	96.9	97.7	**89.2**	88.8
(b) Implicit Action Token	97.6	97.3	96.9	97.0	88.3	89.4
+ Stochastic Iterative Scoring	98.7	96.7	**99.3**	**98.3**	88.8	90.6
(c) The Two-System Approach	**99.0**	98.0	98.7	98.2	85.8	90.8
+ Stochastic Iterative Scoring	**99.0**	**98.1**	98.6	98.2	88.8	**92.1**

Table 7: **Ablation Study on Manipulation.**

Model Variants at Epoch 50	ToolHang	PushT
One-shot Discrete Policy	0.54	0.67
+ Iterative Scoring	0.56	0.69
+ Stochastic Scoring	**0.60**	**0.78**

HUGSIM, Closed-loop Driving. Tab. 3 evaluates zero-shot closed-loop driving on HUGSIM. LDiP achieves strong closed-loop performance without fine-tuning on HUGSIM, and the controller variant obtains the best overall RC of 44.7 and HD-Score of 36.7. Compared with prior methods, the gains are especially clear on Easy and Medium scenarios, suggesting that LDiP transfers well from open-loop training to closed-loop execution when the domain gap is moderate. However, we observe a performance drop in Extreme scenarios after controller tuning, suggesting that these cases remain highly challenging despite the advancements in action modeling.

Visual-based Robotic Manipulation. Our results demonstrate that LDiP can transfer across task domains from autonomous driving to robotic manipulation. As shown in Tab. 4, LDiP achieves success rates of 0.99 on Lift and 0.94 on Can, remaining competitive with the Transformer-based Diffusion Policy [9]. More importantly, it outperforms Diffusion Policy on ToolHang and PushT, reaching 0.52 and 0.76 success rates, respectively. These results show that stochastic discrete scoring is not limited to trajectory planning, but also applies to high-dimensional manipulation.

4.4 Ablation Studies

Key Components. Tab. 5 ablates the main components of LDiP. Starting from Hydra-MDP-$\mathcal{V}_{16384}$, iterative scoring improves EPDMS from 86.2 to 86.7, showing the benefit of re-scoring a pruned candidate set. We further introduce an architectural improvement by replacing the MLP action tokenizer used in prior trajectory scorers [31, 34] with a 1D convolutional tokenizer, which facilitates action-wise temporal modeling. We also adopt a cross-attention-only Transformer decoder, where action tokens no longer perform self-attention over the full vocabulary. This improves EPDMS to 87.1 while remaining efficient for large vocabularies. Adding stochastic scoring achieves the best EPDMS of 87.8. Compared with injecting noise into top-ranked actions, which reaches 87.2, stochastic scoring performs better while keeping the action vocabulary physically plausible.

VLA Architectures. The ablation in Tab. 6 shows that stochastic iterative scoring consistently improves all three VLA designs. The Two-System approach performs best, suggesting that separating vision-language understanding from low-level action scoring [41] is more effective than directly folding action scoring into the VLM. The Action Group Token variant faces the challenge of too many action tokens during joint attention, as the original vocabulary can contain up to 16K candidates and must be downsampled into fewer groups, (e.g. 2,048 group tokens). In contrast, the Implicit Action Token variant does not access the vocabulary directly, which reduces token cost but loses fine-grained geometric information for candidate selection.

Robotic Manipulation. Tab. 7 shows that the same design choices transfer to manipulation. Iterative pruning improves both ToolHang and PushT over the one-shot discrete policy, while stochastic scoring gives the best success rates, improving ToolHang from 0.54 to 0.60 and PushT from 0.67 to 0.78. This confirms that stochastic score perturbation is useful beyond autonomous driving.

Width and Depth. Fig. 5 studies the number of scoring stages and vocabulary sizes. Performance improves from one to three stages, reaching 87.8 EPDMS, but slightly drops with four stages, indicating that additional pruning stages bring diminishing returns. For vocabulary width, the best setting uses 512 intermediate candidates and 16 final candidates. Smaller candidate sets lose useful actions too early, while oversized sets make final scoring more ambiguous.

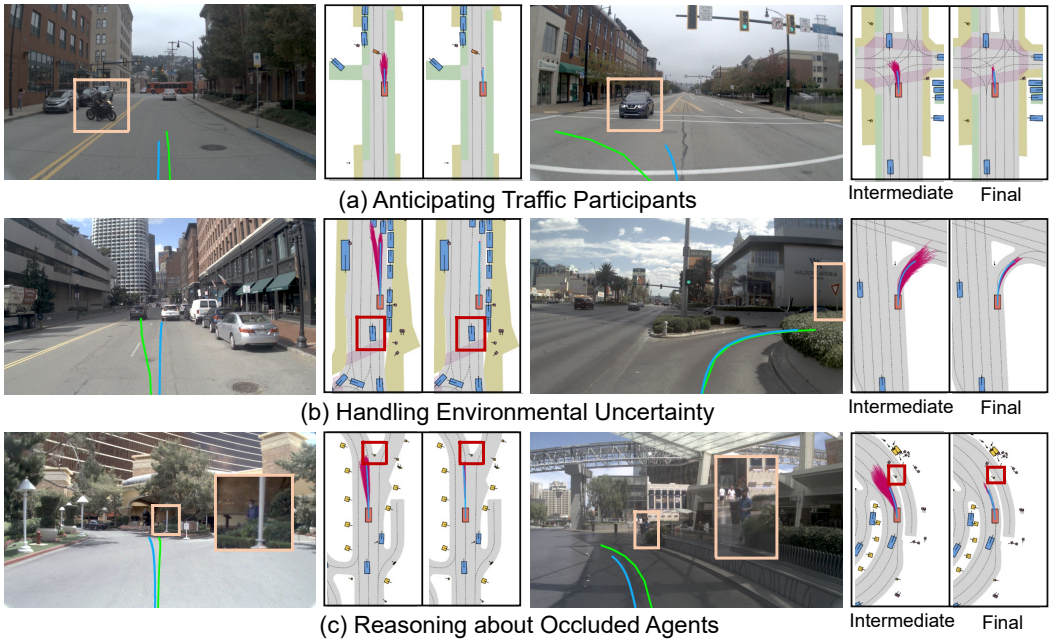

Figure 4: **Visualizations on Navtest.** The figure shows the selected trajectory, the human trajectory, and the remaining vocabulary at the intermediate and final stages.

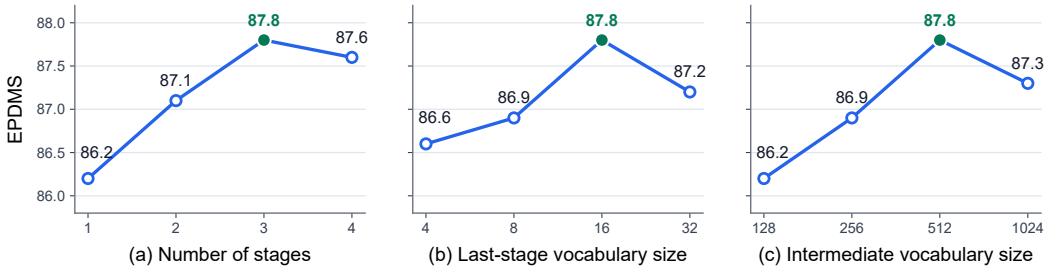

Figure 5: **Ablation Study on the Width and Depth of LDiP.**

4.5 Qualitative Results

Fig. 4 visualizes the iterative scoring process of LDiP on Navtest. At early stages, the remaining vocabulary covers diverse plausible futures, while later stages progressively concentrate on a smaller set of scene-consistent trajectories. (a): LDiP anticipates nearby traffic participants and prunes candidates toward a socially compliant trajectory. (b): the model preserves aggressive candidates in early stages but shifts towards a safer selection in response to the environmental uncertainty. (c): LDiP reasons about potentially occluded, distant agents and selects a conservative trajectory.

5 Conclusion

We present the Large Discrete Policy (LDiP), a fully discrete behavior modeling framework that improves the expressivity of discrete policies without compromising their explicit and physically plausible formulation. By constructing a large fixed action vocabulary and introducing iterative scoring with stochastic score perturbation, LDiP progressively compares and prunes plausible action candidates while keeping the decision process interpretable. Across end-to-end planning, closed-loop driving, robotic manipulation, and vision-language-action settings, LDiP consistently demonstrates strong performance against both discrete trajectory scorers and iterative denoising policies. These results suggest that LDiP provides a practical and interpretable alternative for behavior modeling.

Limitations. While LDiP shows promising performance in behavior modeling, it still falls behind Diffusion Policy on certain manipulation tasks. Further evaluation in real-world settings is also needed. The scope of this work is to study real-world planning for autonomous driving and simulation-based closed-loop performance in both autonomous driving and robotic manipulation.

6 Acknowledgements

This work is supported by the National Natural Science Foundation of China (Grant No. 62427819) and the Science and Technology Commission of Shanghai Municipality (No. 24511103100).

References

[1] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[2] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[3] Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

[4] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[5] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[6] 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. In *CVPR*, pages 11621–11631, 2020.

[7] Wei Cao, Marcel Hallgarten, Tianyu Li, Daniel Dauner, Xunjiang Gu, Caojun Wang, Yakov Miron, Marco Aiello, Hongyang Li, Igor Gilitschenski, et al. Pseudo-simulation for autonomous driving. *CoRL*, 2025.

[8] 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.

[9] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

[10] 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*, 2022.

[11] Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. Diffusion models in vision: A survey. *IEEE TPAMI*, 45(9):10850–10869, 2023.

[12] Chenxu Dang, Sining Ang, Yongkang Li, Haochen Tian, Jie Wang, Guang Li, Hangjun Ye, Jie Ma, Long Chen, and Yan Wang. Drivefine: Refining-augmented masked diffusion vla for precise and robust driving. *arXiv preprint arXiv:2602.14577*, 2026.

[13] 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 *NeurIPS*, 2024.

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[15] Pete Florence, Corey Lynch, Andy Zeng, Oscar Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. *CoRL*, November 2021.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *NeurIPS*, volume 33, pages 6840–6851, 2020.

[18] Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. Planning-oriented autonomous driving. In *CVPR*, pages 17853–17862, 2023.

[19] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_0.5$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

[20] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[21] 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. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8350, 2023.

[22] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

[23] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[25] Ellington Kirby, Alexandre Boulch, Yihong Xu, Yuan Yin, Gilles Puy, Éloi Zablocki, Andrei Bursuc, Spyros Gidaris, Renaud Marlet, Florent Bartoccioni, Anh-Quan Cao, Nermin Samet, Tuan-Hung Vu, and Matthieu Cord. Driving on registers. In *CVPR*, 2026.

[26] Youngwan Lee, Joong-won Hwang, Sangrok Lee, Yuseok Bae, and Jongyoul Park. An energy and gpu-computation efficient backbone network for real-time object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 0–0, 2019.

[27] Kailin Li, Zhenxin Li, Shiyi Lan, Yuan Xie, Zhizhong Zhang, Jiayi Liu, Zuxuan Wu, Zhiding Yu, and Jose M Alvarez. Hydra-mdp++: Advancing end-to-end driving via expert-guided hydra-distillation. *arXiv preprint arXiv:2503.12820*, 2025.

[28] Shufan Li, Konstantinos Kallidromitis, Hritik Bansal, Akash Gokul, Yusuke Kato, Kazuki Kozuka, Jason Kuen, Zhe Lin, Kai-Wei Chang, and Aditya Grover. Lavida: A large diffusion language model for multimodal understanding. *arXiv preprint arXiv:2505.16839*, 2025.

[29] Yingyan Li, Shuyao Shang, Weisong Liu, Bing Zhan, Haochen Wang, Yuqi Wang, Yuntao Chen, Xiaoman Wang, Yasong An, Chufeng Tang, et al. Drivevla-w0: World models amplify data scaling law in autonomous driving. *arXiv preprint arXiv:2510.12796*, 2025.

[30] Yongkang Li, Kaixin Xiong, Xiangyu Guo, Fang Li, Sixu Yan, Gangwei Xu, Lijun Zhou, Long Chen, Haiyang Sun, Bing Wang, et al. Recogdrive: A reinforced cognitive framework for end-to-end autonomous driving. *arXiv preprint arXiv:2506.08052*, 2025.

[31] Zhenxin Li, Kailin Li, Shihao Wang, Shiyi Lan, Zhiding Yu, Yishen Ji, Zhiqi Li, Ziyue Zhu, Jan Kautz, Zuxuan Wu, et al. Hydra-mdp: End-to-end multimodal planning with multi-target hydra-distillation. *arXiv preprint arXiv:2406.06978*, 2024.

[32] Zhenxin Li, Shihao Wang, Shiyi Lan, Zhiding Yu, Zuxuan Wu, and Jose M Alvarez. Hydra-next: Robust closed-loop driving with open-loop training. *arXiv preprint arXiv:2503.12030*, 2025.

[33] Zhenxin Li, Wenhao Yao, Zi Wang, Xinglong Sun, Jingde Chen, Nadine Chang, Maying Shen, Jingyu Song, Zuxuan Wu, Shiyi Lan, et al. Ztrs: Zero-imitation end-to-end autonomous driving with trajectory scoring. *arXiv preprint arXiv:2510.24108*, 2025.

[34] Zhenxin Li, Wenhao Yao, Zi Wang, Xinglong Sun, Joshua Chen, Nadine Chang, Maying Shen, Zuxuan Wu, Shiyi Lan, and Jose M Alvarez. Generalized trajectory scoring for end-to-end multimodal planning. *arXiv preprint arXiv:2506.06664*, 2025.

[35] 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 *CVPR*, pages 12037–12047, 2025.

[36] Yiyi Liao, Jun Xie, and Andreas Geiger. Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE TPAMI*, 45(3):3292–3310, 2022.

[37] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *ICLR*, 2023.

[38] I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[39] Yuechen Luo, Fang Li, Shaoqing Xu, Zhiyi Lai, Lei Yang, Qimao Chen, Ziang Luo, Zixun Xie, Shengyin Jiang, Jiaxin Liu, et al. Adathinkdrive: Adaptive thinking via reinforcement learning for autonomous driving. *arXiv preprint arXiv:2509.13769*, 2025.

[40] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *CoRL*, volume 164, pages 1678–1690, 2022.

[41] NVIDIA, Johan Bjorck, Nikita Cherniadev Fernando Castañeda, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. GR00T N1: An open foundation model for generalist humanoid robots. In *ArXiv Preprint*, March 2025.

[42] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *ECCV*, pages 194–210. Springer, 2020.

[43] Nur Muhammad Shafiullah, Zichen Cui, Ariuntuya Arty Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone. *NeurIPS*, 35:22955–22968, 2022.

[44] Chonghao Sima, Kashyap Chitta, Zhiding Yu, Shiyi Lan, Ping Luo, Andreas Geiger, Hongyang Li, and Jose M Alvarez. Centaur: Robust end-to-end autonomous driving with test-time training. *arXiv preprint arXiv:2503.11650*, 2025.

[45] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021.

[46] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, pages 2446–2454, 2020.

[47] Wenchao Sun, Xuewu Lin, Keyu Chen, Zixiang Pei, Xiang Li, Yining Shi, and Sifa Zheng. Sparsedrivev2: Scoring is all you need for end-to-end autonomous driving. *arXiv preprint arXiv:2603.29163*, 2026.

[48] Tianyi Tan, Yinan Zheng, Ruiming Liang, Zexu Wang, Kexin Zheng, Jinliang Zheng, Jianxiong Li, Xianyuan Zhan, and Jingjing Liu. Flow matching-based autonomous driving planning with advanced interactive behavior modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[49] Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. *arXiv preprint arXiv:1805.01954*, 2018.

[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 30, 2017.

[51] Xinlong Wang, Xiaosong Zhang, Zhengxiong Luo, Quan Sun, Yufeng Cui, Jinsheng Wang, Fan Zhang, Yueze Wang, Zhen Li, Qiying Yu, et al. Emu3: Next-token prediction is all you need. *arXiv preprint arXiv:2409.18869*, 2024.

[52] Pengchuan Xiao, Zhenlei Shao, Steven Hao, Zishuo Zhang, Xiaolin Chai, Judy Jiao, Zesong Li, Jian Wu, Kai Sun, Kun Jiang, et al. Pandaset: Advanced sensor suite dataset for autonomous driving. In *ITSC*, pages 3095–3101. IEEE, 2021.

[53] Ningyuan Yang, Jiaxuan Gao, Feng Gao, Yi Wu, and Chao Yu. Fine-tuning diffusion policies with backpropagation through diffusion timesteps. *arXiv preprint arXiv:2505.10482*, 2025.

[54] Wenhao Yao, Zhenxin Li, Shiyi Lan, Zi Wang, Xinglong Sun, Jose M Alvarez, and Zuxuan Wu. Drivesuprim: Towards precise trajectory selection for end-to-end planning. *arXiv preprint arXiv:2506.06659*, 2025.

[55] Wenhao Yao, Xinglong Sun, Zhenxin Li, Shiyi Lan, Zi Wang, Jose M Alvarez, and Zuxuan Wu. Had: Combining hierarchical diffusion with metric-decoupled rl for end-to-end driving. *arXiv preprint arXiv:2604.03581*, 2026.

[56] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *RSS*, 2023.

[57] 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*, 2025.

[58] Hongyu Zhou, Longzhong Lin, Jiabao Wang, Yichong Lu, Dongfeng Bai, Bingbing Liu, Yue Wang, Andreas Geiger, and Yiyi Liao. Hugsim: A real-time, photo-realistic and closed-loop simulator for autonomous driving. *arXiv preprint arXiv:2412.01718*, 2024.

[59] Zewei Zhou, Tianhui Cai, Seth Z. Zhao, Yun Zhang, Zhiyu Huang, Bolei Zhou, and Jiaqi Ma. Autovla: A vision-language-action model for end-to-end autonomous driving with adaptive reasoning and reinforcement fine-tuning. *NeurIPS*, 2025.

[60] Zewei Zhou, Ruining Yang, Yiluan Guo, Sherry X Chen, Tao Feng, Kateryna Pistunova, Yishan Shen, Lili Su, Jiaqi Ma, et al. Spanvla: Efficient action bridging and learning from negative-recovery samples for vision-language-action model. *arXiv preprint arXiv:2604.19710*, 2026.

[61] Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025.

[62] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.

[63] Jialv Zou, Shaoyu Chen, Bencheng Liao, Zhiyu Zheng, Yuehao Song, Lefei Zhang, Qian Zhang, Wenyu Liu, and Xinggang Wang. Diffusiondrivev2: Reinforcement learning-constrained truncated diffusion modeling in end-to-end autonomous driving. *arXiv preprint arXiv:2512.07745*, 2025.

A Technical appendices and supplementary material

Here we provide the technical appendices and supplementary material to complement the main paper. These materials include additional qualitative results and extended analyses of our approach.

A.1 Additional Qualitative Results (Autonomous Driving)

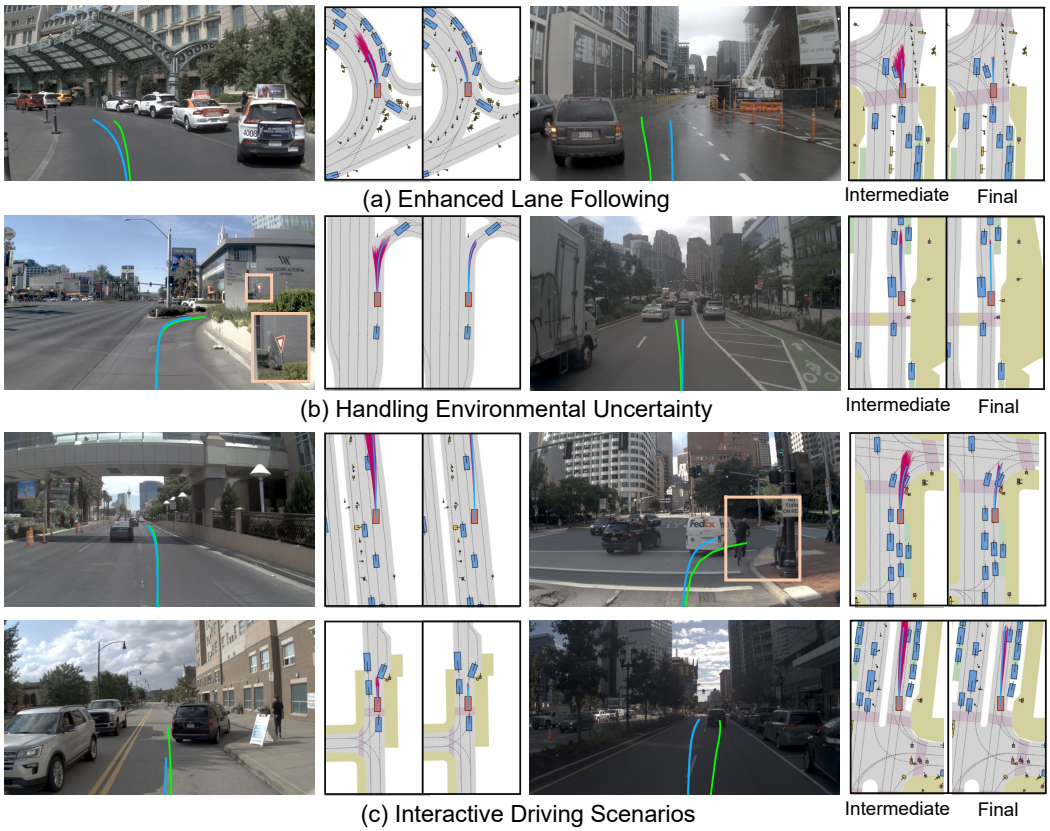

Figure 6: **Additional Visualizations on Navtest.** The figure shows the selected trajectory, the human trajectory, and the remaining vocabulary at the intermediate and final stages.

Fig. 6 provides supplemental examples showing that our policy generates stable actions across diverse urban scenes. In (a) and (b), the model follows curved and straight lanes smoothly while remaining aligned with the intended drivable corridor, even in visually complex settings with parked vehicles, dense traffic, and distant traffic signs. Compared with the human trajectory, the predictions of the policy are more conservative, maintaining a larger safety buffer against traffic participants. The paired camera and Bird's-Eye-Views show that the predicted paths are not only visually plausible in image space, but also consistent with lane geometry and surrounding agents.

Fig. 6 (c) further highlights interactive driving scenarios, where the ego vehicle must plan around nearby vehicles, pedestrians, cyclists, and intersection layouts. Across these examples, the actions remain conservative and context-aware, avoiding abrupt deviations while adapting to local traffic structure. Overall, LDiP demonstrates end-to-end planning robustness by maintaining smooth lane-following behavior and reasonable interaction awareness in complex urban environments.

A.2 Additional Qualitative Results (Robotic Manipulation)

Fig. 7 visualizes representative frames for the ToolHang task. The sequence compares the pruned action vocabulary, shown as the top-8 candidate action chunks at each step, with the final selected

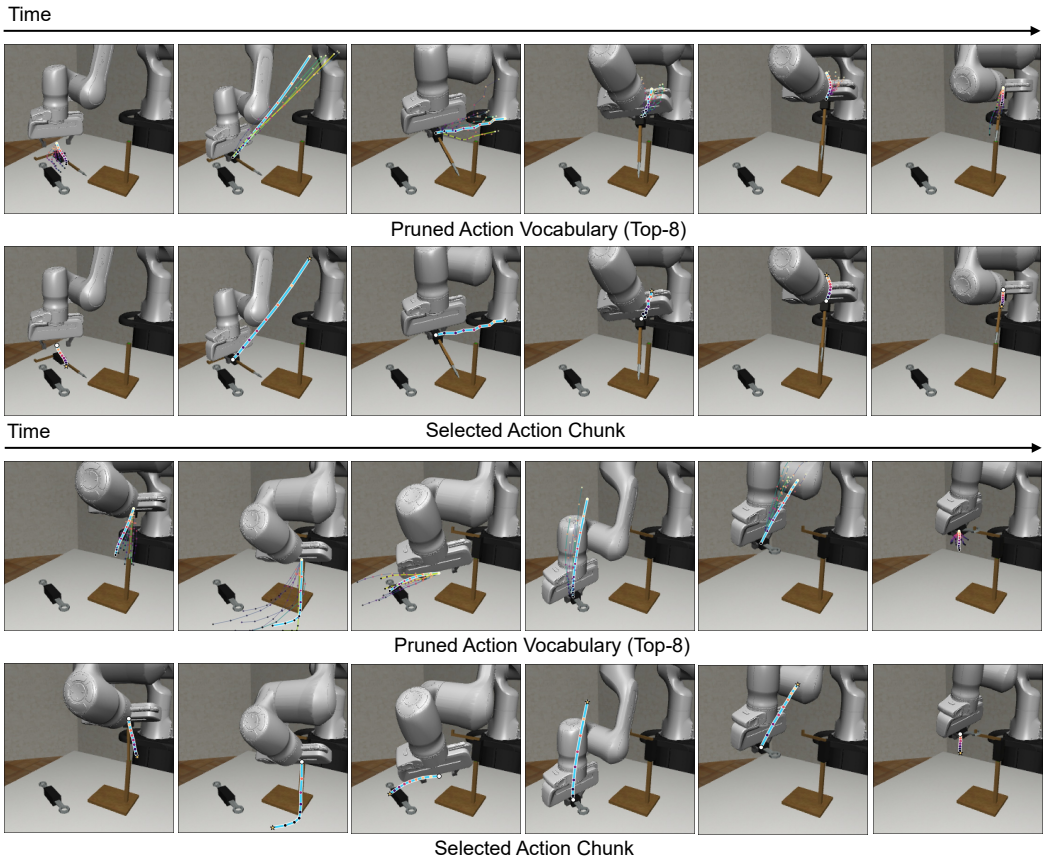

Figure 7: **Additional Visualizations on Robotic Manipulation.** The figure shows a full episode of the ToolHang task. We show the key frames with top-8 action chunks from the last scoring stage and the selected action chunk.

action chunk executed by the policy. Across time, the candidate chunks capture diverse feasible motions for assembling the frame and lifting the tool toward the hook.

A.3 Efficiency

Table 8: **Comparisons on Efficiency.** We compare the model complexity and runtime efficiency of different policies. The FPS metric is measured on one single NVIDIA A100 GPU with the PyTorch FP32 backend. * The Diffusion Policy utilizes extra networks to encode the Bird's-Eye-View condition tokens following LTF [10], while the DDPM Scheduler [17] is used for iterative denoising.

Method	Backbone	Params.	Vocab. Size	Iteration Steps	FPS	EPDMS
Diffusion Policy* [9]	V2-99	115.0M	-	100	4.1	79.1
Hydra-MDP-$\mathcal{V}_{16384}$ [31]	V2-99	83.0M	16384	1	13.5	86.2
GTRS-Dense [34]	V2-99	83.0M	8192	1	**18.5**	84.0
ZTRS [33]	V2-99	83.4M	8192	1	18.4	85.3
LDiP	V2-99	82.8M	16384	3	12.1	**87.8**

Tab. 8 compares the model complexity and runtime efficiency of different policy architectures. Compared with Diffusion Policy, LDiP avoids expensive iterative denoising over 100 DDPM steps, resulting in substantially higher inference speed while also improving EPDMS. For the implementation of the Diffusion Policy, we represent each trajectory as normalized displacements in the longitudinal and lateral directions. Among discrete policies, LDiP maintains a comparable model size with the V2-99 backbone [26], while using a larger action vocabulary and three lightweight selection

steps. Although this slightly reduces FPS compared with prior one-step methods, LDiP still runs at 12.1 FPS on a single NVIDIA A100 GPU with the PyTorch FP32 backend, while achieving the best EPDMS score of 87.8. This indicates that the proposed multi-step action selection provides a favorable trade-off between planning accuracy and runtime efficiency.

A.4 Broader Impact

This work may contribute to more interpretable and physically plausible policies for autonomous driving and robotic manipulation. However, deploying such systems in the real world directly can introduce safety risks if the policy fails in unseen conditions or inherits biases from demonstration data. Practical use should therefore require rigorous closed-loop testing, safety monitoring, and human oversight.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please refer to the conclusion.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper describes the model architecture, training settings, datasets, and evaluation protocols needed to reproduce the main results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: The data is public. The code will be released in the future.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Please refer to the implementation details section for these details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The baselines in our main experiments do not report error bars. Following this convention, we present single-run results for autonomous driving. For robotic manipulation, we report the average success rate over the last 10 checkpoints, consistent with the baseline protocol.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: The paper reports the GPU type and training setup used for the main experiments, but does not disclose the full compute invested in the project, including preliminary or failed experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: The research in the paper conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Please refer to the appendix.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not pose such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We use publicly available assets such as NAVSIM, HUGSIM, Robomimic, Diffusion Policy, and pretrained VLM backbones, and cite their original papers or repositories. These assets are used only for research training and evaluation under their stated licenses and terms. We do not redistribute raw datasets or third-party model weights; released artifacts will include appropriate license and attribution information.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.