DriveWAM: Video Generative Priors Enable Scalable World-Action Modeling for Autonomous Driving

Chen Shi[1*] **Jinrui Xu**[1*] **Shaoshuai Shi**[2] **Kehua Sheng**[2] **Bo Zhang**[2] **Li Jiang**[1,3†]

[1]The Chinese University of Hong Kong, Shenzhen
[2]Voyager Research, Didi Chuxing [3]Shenzhen Loop Area Institute

Project Page: `https://chenshi3.github.io/drivewam.github.io/`

Abstract

Pretrained foundation models have become an important basis for end-to-end autonomous driving. In contrast to vision-language models pretrained primarily on static image-text pairs, video generative models capture temporal dynamics and motion priors that are naturally suited for driving. We present DriveWAM, a driving world-action model that adapts a pretrained video diffusion transformer into an autoregressive video-action policy. DriveWAM organizes video and action streams into a unified temporal token sequence and trains them under a joint flow-matching objective, preserving the pretrained video-generation architecture while adapting its large-scale video priors to action generation. To incorporate high-level scene understanding, we introduce scene-evolving driving guidance, where a frozen VLM produces chunk-specific semantic intent to guide video-action generation. To keep long-horizon rollout bounded, we further introduce selective KV memory, which maintains bounded modality-aware video and action memory pools through relevance-redundancy cache selection at inference time. Experiments on NAVSIM and the PhysicalAI-Autonomous-Vehicles benchmark show that DriveWAM achieves strong planning performance, and a data-scaling study from 4k to 100k driving clips further confirms the scaling potential of world-action modeling for end-to-end autonomous driving.

1 Introduction

Recent end-to-end autonomous driving systems increasingly leverage pretrained foundation models as policy backbones. A major line of work builds on vision-language-action (VLA) models [1–7], transferring the semantic knowledge and instruction-following ability of large-scale VLMs [8–13] to action generation. Such VLA-based policies are well suited to high-level scene understanding and semantic reasoning, but driving decisions also require temporally dense visual cues such as spatial layout, motion continuity, and how the scene may evolve in the near future. Since VLM backbones are pretrained primarily on image-text data rather than video dynamics, VLM-centric policies must acquire these temporal priors largely from downstream driving data.

Video generative models offer a complementary foundation. They are pretrained on large-scale videos to model object persistence, motion patterns, and scene evolution, making them naturally suited for dynamic decision problems. Recent VLA-based driving methods [3, 14, 15] have begun to incorporate future image or video generation to improve spatio-temporal awareness, but visual generation is often used as an auxiliary signal or a modular component on top of a VLM-centric policy. In parallel, world-action (WA) models in robotics [16–21] show that pretrained video foundation models can be adapted more directly for action prediction and planning.

∗:Equal Contribution. Work done during an internship at Voyager Research, Didi Chuxing. †: Corresponding author.

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

Adapting this paradigm to autonomous driving, however, remains non-trivial. First, a video foundation model is pretrained for visual generation rather than ego-action control, so turning it into an autoregressive video-action policy requires preserving its future-generation prior while coupling it to continuous action prediction. Second, video foundation models capture near-future dynamics but lack high-level semantic planning, whereas the appropriate driving decision depends on route intent, right-of-way, and decision-relevant traffic participants. Third, deploying such autoregressive policies over long horizons requires persistent historical context, but full KV caching grows with horizon length and sliding-window caching may discard old yet critical evidence. Existing driving-oriented world-action methods [22–24] often rely on separate planners, discrete video tokenizers, or customized generation architectures, leaving open how to directly adapt a modern video foundation model into a semantically guided and scalable end-to-end driving policy.

In this paper, we present DriveWAM, a driving world-action model that adapts a pretrained video foundation model into an end-to-end autonomous driving policy. DriveWAM uses a flow-matching video diffusion transformer as the policy core and formulates driving as autoregressive video-action generation. Given observed video-action history and ego state, the model first generates future video latents and then decodes ego actions conditioned on the generated future latent, realizing inverse-dynamics action generation. Both video and action streams share the same transformer and are trained under a joint flow-matching objective [25], preserving the pretrained spatio-temporal generative prior while learning to convert imagined future world evolution into executable ego motion.

To supply the missing high-level driving semantics, DriveWAM introduces scene-evolving driving guidance. A frozen VLM uses only causally available context, including the latest observation, recent ego motion, and route command, and produces chunk-specific guidance for the next prediction horizon. This guidance is injected through temporally localized cross-attention, ensuring that each future video-action chunk receives its own semantic intent while preserving the causal structure of full-clip autoregressive training. Thus, the VLM acts as a semantic guide, while the video foundation model remains responsible for dense temporal prediction.

For long-horizon rollout, DriveWAM further introduces selective KV memory. Instead of storing all historical tokens or evicting tokens by age, DriveWAM maintains separate bounded memory pools for video and action KVs. Each pool is updated by a relevance-redundancy selection rule inspired by efficient video-generation caching [26]: prediction-relevant tokens are retained, while redundant patterns are filtered out. This training-free memory provides a compact video-action history for autoregressive inference without changing the training objective.

We evaluate DriveWAM on NAVSIM [27] and the large-scale PhysicalAI-Autonomous-Vehicles benchmark [28]. DriveWAM achieves strong planning performance with an autoregressive world-action architecture. Beyond benchmark comparison, we conduct a data-scaling study over 4k, 20k, and 100k driving clips, where DriveWAM improves consistently as training data increases. These results suggest that semantically guided world-action modeling provides a scalable foundation for end-to-end autonomous driving. Our contributions are summarized as follows:

- We propose DriveWAM, a driving world-action model that adapts a pretrained video diffusion transformer into an autoregressive video-action policy under a joint flow-matching objective.
- We introduce scene-evolving driving guidance to supply high-level driving semantics, where a frozen VLM provides causally available chunk-specific intent that guides video-action generation through temporally localized cross-attention.
- We propose selective KV memory for bounded long-horizon rollout, maintaining modality-aware video and action memory pools through relevance-redundancy cache selection at inference time.
- Experiments on NAVSIM and PhysicalAI-Autonomous-Vehicles, together with a scaling study from 4k to 100k clips, demonstrate the effectiveness and scalability of DriveWAM.

2 Related Work

2.1 Vision-Language-Action Models in Autonomous Driving

Recent autonomous driving methods increasingly leverage the general knowledge and semantic reasoning capabilities of large vision-language models. Early efforts use LLMs or VLMs mainly as high-level reasoning modules [29–36], producing scene descriptions, maneuver suggestions, command tokens, or coarse trajectories that are further consumed by downstream planners. More recent

VLA methods [6, 37, 4] move toward end-to-end action prediction by coupling VLM backbones with trajectory decoders or planning heads. DriveMoE [6] introduces an MoE-based policy head on top of a VLM to route different driving situations to specialized action experts. AutoVLA [37] discretizes continuous trajectories into action primitives and casts driving policy learning as autoregressive token prediction. ReCogDrive [4] combines VLM-based reasoning with a diffusion trajectory planner and further aligns the policy through imitation learning and reinforcement learning.

Building on this line, a parallel set of works incorporates visual world modeling into the VLA pipeline. FSDrive [15] introduces future visual prediction as a visual reasoning process, while DriveVLA-W0 [3] and DriveDreamer-Policy [14] augment VLM-based policies with generative world-model components [38–41]. LCDrive [42] performs latent chain-of-thought reasoning by interleaving action proposals with world-model tokens. Although these designs improve the spatio-temporal awareness of VLA-based driving, their policy core remains VLM-centric, with visual generation serving as an auxiliary branch rather than the policy backbone. In contrast, DriveWAM inherits a pretrained video generative model as the policy core to jointly model future world evolution and ego actions, while leveraging VLM reasoning as complementary scene-evolving guidance for high-level semantic intent.

2.2 World-Action Models

The world-action paradigm reuses pretrained video generative models as the foundation for policy learning. Recent works in robotic manipulation [16–19, 21] have shown that large-scale video pretraining can transfer favorably to action generation, motivating its adoption in autonomous driving. WorldDrive [22] transfers representations learned by a trajectory-aware driving world model to a downstream planner, bridging scene generation and planning but keeping planning as a separate module. VaViM/VaVAM [23] formulates autonomous driving as autoregressive video modeling with discrete VQ-VAE tokens [43] through a GPT-style transformer [44], and extends the model with an action expert for trajectory prediction. Epona [24] couples a spatiotemporal transformer with twin diffusion transformers for separate next-frame generation and ego-trajectory prediction. While these designs establish important baselines, they do not directly adopt a modern video foundation model as a unified video-action policy backbone, and thus cannot fully inherit the latest pretrained video priors. DriveWAM instead builds directly on a pretrained video diffusion transformer and adapts both video and action streams under a unified flow-matching objective.

Moreover, existing driving-oriented world-action methods mostly rely on simple navigation commands as high-level guidance, leaving rich scene-level semantic reasoning largely unexplored. DriveWAM addresses this by injecting chunk-specific VLM guidance through temporally localized cross-attention. Efficient memory is another requirement for autoregressive video-action policies during long-horizon rollout, but prior models either use a limited observation window [16, 18] or maintain a standard KV cache [17, 19] whose cost grows with the sequence length. Recent works on efficient autoregressive video generation explore sliding-window attention [45–47], sparse attention [48, 49], and cache compression [26, 50]. DriveWAM adapts the relevance-redundancy criterion of FlowCache [26] to maintain bounded video and action memory pools for long-horizon driving.

3 Method

We propose DriveWAM, a semantically guided world-action model that adapts a pretrained video foundation model into a unified backbone for future world evolution and ego-action generation in autonomous driving, complemented by guidance from a frozen VLM for scene-evolving driving semantics. Specifically, as shown in Figure 1, we first formulate driving as autoregressive video-action generation, where a pretrained video diffusion transformer predicts future video latents and ego actions under a joint flow-matching objective (Sec. 3.1). We then introduce scene-evolving guidance, using a frozen VLM to provide causally available chunk-level intent that steers the video-action generation process (Sec. 3.2). Finally, we present selective KV memory, which retains prediction-relevant and non-redundant video-action history for bounded long-horizon rollout (Sec. 3.3).

3.1 Autoregressive Video-Action Generation

A driving clip contains synchronized streams of camera images, ego actions, and ego states. We divide the clip into K consecutive chunks and then consider the driving task as the next-chunk generation. At decision step k, the model has observed the clip up to chunk k and predicts the future video-action

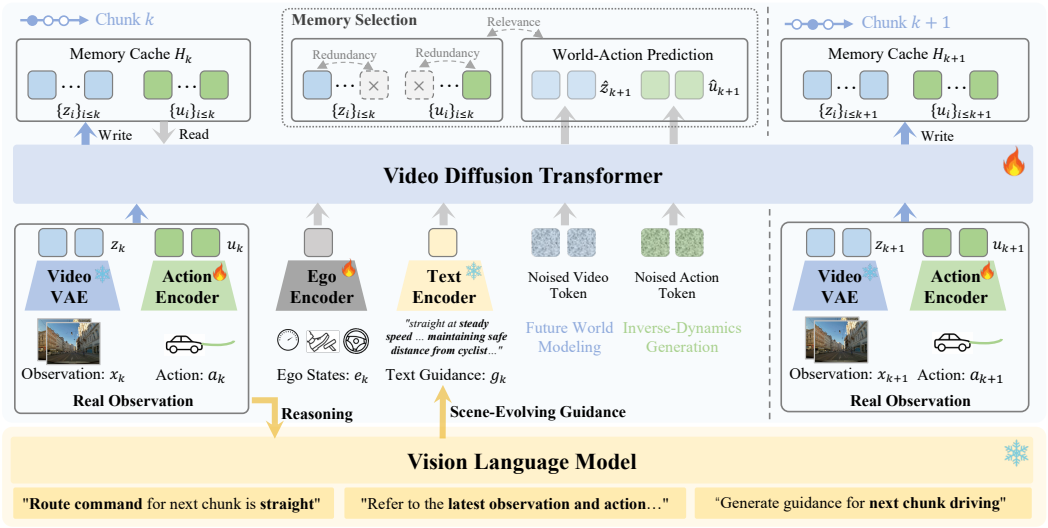


Figure 1: Overview of DriveWAM, which adapts a pretrained video generation backbone into a unified video-action policy. Building on this backbone, DriveWAM uses a frozen VLM to provide chunk-specific scene-evolving guidance for high-level scene reasoning and introduces selective KV memory to preserve compact prediction-relevant history for long-horizon rollout.

chunk (x_{k+1}, a_{k+1}), where x_{k+1} is the next video segment and a_{k+1} is the corresponding ego action. The causally available conditions include the historical context H_k (video and action tokens of all observed chunks up to k), the ego state e_k at the end frame of chunk k (*e.g.*, velocity, acceleration, and curvature), and a textual guidance g_k for the predicted chunk.

Tokenization. To jointly model video-action generation, we organize video and action chunks into a unified temporal token sequence while preserving their temporal order. Each observed video chunk is encoded by the pretrained VAE [40], and each ego-action chunk, represented as normalized ego-frame translation and yaw increments, is embedded by an MLP action encoder E_a, as follows:

$$z_k = \mathrm{VAE}(x_k), \qquad u_k = E_a(a_k), \qquad H_k = \{(z_i, u_i)\}_{i \le k}. \tag{1}$$

Here, $z_k \in \mathbb{R}^{N_x \times d_z}$ and $u_k \in \mathbb{R}^{N_a \times d}$ denote encoded video and action tokens, respectively. N_x and N_a are the numbers of tokens per chunk, d_z is the VAE latent channel dimension, and d is the transformer hidden dimension. In practice, the VAE latents z_k are also mapped to dimension d by the latent input embedding layer inherited from the pretrained video diffusion transformer, yielding a unified representation for video-action generation.

World-action flow. DriveWAM adopts the autoregressive video-action generation scheme, which factors the driving task into future world modeling and inverse-dynamics action generation. Specifically, DriveWAM utilizes a pretrained flow-matching video diffusion transformer T_ω [40] for predicting the next video chunk and action chunk. During training, we sample a flow timestep $\tau \in [0, 1]$ along the rectified-flow path [25, 51], where $\tau = 1$ is the Gaussian-noise endpoint and $\tau = 0$ represents clean data. For the next video chunk, the clean latent z_{k+1} is noised along the standard rectified-flow path, producing a query $z_{k+1,\tau}$ and target velocity $v^z_{k+1,\tau}$. The video branch predicts this velocity under the current driving context:

$$\hat{v}^z_{k+1,\tau} = T_\omega(z_{k+1,\tau}; H_k, e_k, g_k, \tau). \tag{2}$$

Here, e_k is embedded by a lightweight MLP and injected through a separate ego-state cross-attention branch. Notably, this conditioning repurposes the video model as a policy prior, with the backbone retaining its native future-visual-prediction objective while the predicted future is shaped by driving history, ego state, and semantic intent.

Actions are generated by an inverse-dynamics flow on the same diffusion transformer. We perturb the next action chunk directly in the normalized action space and embed it with the MLP action encoder E_a to obtain $u_{k+1,\tau}$. Conditioned on the future world latent and the current driving context, the

shared transformer predicts the action velocity as:

$$\hat{v}^a_{k+1,\tau} = D_a(T_\omega(u_{k+1,\tau}; \tilde{z}_{k+1}, H_k, e_k, g_k, \tau)), \tag{3}$$

where D_a is an MLP action decoder. The conditioning latent $\tilde{z}_{k+1}$ is the clean future video latent z_{k+1} during teacher-forced training and the generated latent $\hat{z}_{k+1}$ during inference. This design grounds action generation in the predicted world evolution, so the action decoder behaves as an inverse-dynamics readout of the predicted future rather than an independent trajectory head. We use noisy-history augmentation [19] to reduce this train-test mismatch.

Training objective. We train the video and action branches with a joint flow-matching objective:

$$\mathcal{L} = \mathbb{E}_{k,\tau}\left[\left\|\hat{v}^z_{k+1,\tau} - v^z_{k+1,\tau}\right\|_2^2 + \beta_a \left\|\hat{v}^a_{k+1,\tau} - v^a_{k+1,\tau}\right\|_2^2\right], \tag{4}$$

where β_a controls the balance between future world modeling and action generation. The video term preserves the pretrained spatio-temporal generative prior during policy adaptation, while the action term teaches the shared backbone to decode this prior into executable ego motion.

Full-clip training and autoregressive rollout. During training, we process all chunks of a clip in a single forward pass for efficiency. The video-action tokens are arranged in temporal order and denoised in parallel under a causal teacher-forcing mask (Figure 2), which realizes the conditional dependencies in Eqs. 2 and 3 while preserving the causal pattern used during inference [46, 19, 45]. At inference, DriveWAM rolls out one chunk at a time. Given history H_k, the model first samples the future video latent $\hat{z}_{k+1}$ and then samples the action chunk $\hat{a}_{k+1}$ conditioned on this generated future. When the next real observation becomes available, it is encoded and appended to the history to form H_{k+1}, keeping long-horizon rollout grounded in observed driving context.

3.2 Scene-Evolving Driving Guidance

The video foundation model provides dense dynamic priors for near-future scene evolution, but it lacks semantic planning ability. In driving, the appropriate future is determined not only by short-term dynamics but also by route intent, traffic participants, and other decision-level semantics. For example, at an intersection, multiple future evolutions may be visually plausible from the current observation, while the desired one depends on the high-level driving intent. However, existing world-action methods typically use a single clip-level text condition, applying the same semantic guidance to every chunk. DriveWAM instead introduces a frozen VLM as a scene-evolving semantic guide. At each decision step k, the VLM produces fresh guidance g_k from the latest causally available context, so each future video-action chunk is conditioned on its own up-to-date semantic intent while the video model remains the policy backbone for dense temporal prediction.

Causal guidance generation. At each decision step k, the frozen Qwen3-VL-8B [9] receives only causally available information: the latest observation x_k, a recent ego trajectory a_k, and the route command c_k for the upcoming horizon. It produces a concise guidance text as follows:

$$g_k = \Phi_{\mathrm{VLM}}(x_k, a_k, c_k), \tag{5}$$

which summarizes the current road context and provides ego behavior guidance for the upcoming horizon, such as proceeding, yielding, stopping, or merging. Since no observation from the target chunk is used, g_k provides semantic intent for predicting (x_{k+1}, a_{k+1}) without leaking future information. During training, guidance texts are precomputed and cached; during inference, the VLM is queried once per decision step and reused across all denoising steps, keeping the semantic condition aligned with the current prediction horizon.

Temporally localized guidance injection. Scene-evolving guidance introduces a separate text condition g_k at each decision step. Without an additional constraint, tokens of chunk $k+1$ could attend to guidance from other chunks, including future guidance from later decision steps, breaking causal consistency. We therefore apply an additional block-diagonal text mask, which allows video-action tokens of target chunk $k+1$ to attend only to the guidance tokens of g_k. This keeps semantic conditioning temporally localized and prevents cross-chunk leakage. The resulting attention pattern is illustrated in Figure 2.

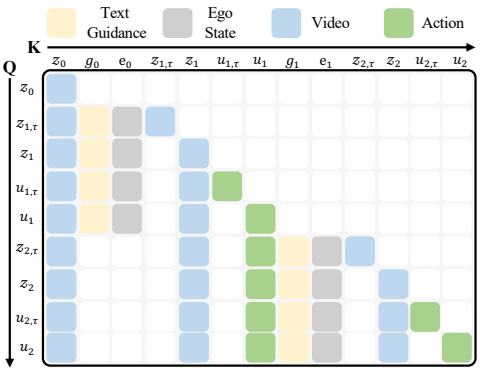


Figure 2: Attention mask used during DriveWAM training. Colored entries indicate allowed attention; blank entries are masked.

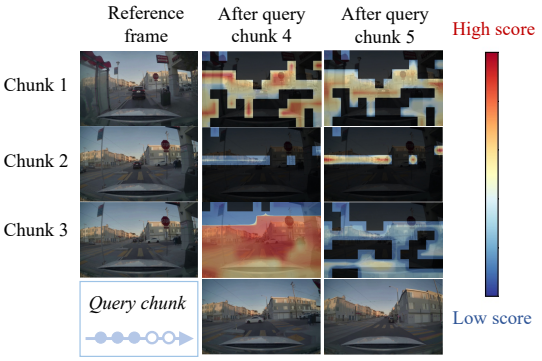


Figure 3: Video-token retention under selective KV memory. Columns 2 and 3 visualize the tokens retained after query chunks 4 and 5.

3.3 Selective KV Memory for Long-Horizon Rollout

Autoregressive world-action rollout conditions on the historical context H_k defined in Sec. 3.1, where H_k denotes the causal video-action history up to step k. During inference, this abstract history is implemented as layer-wise KV caches that store the keys and values produced by previous video and action chunks, so the model can attend to past context without recomputing all historical tokens. A full-window cache preserves complete history but grows linearly with rollout length, while a sliding-window cache bounds the cost by evicting the oldest tokens under FIFO rules [52, 53]. However, age-based eviction is suboptimal for driving tasks: older tokens may remain decision-relevant, such as motion trend of a nearby vehicle or a briefly occluded pedestrian, whereas newer tokens may correspond to repeated static background. To keep long-horizon inference bounded without discarding useful context, DriveWAM adopts an inference-time, training-free selective KV memory inspired by FlowCache [26], retaining a compact, prediction-relevant approximation of H_k during rollout.

Modality-aware memory pools. Video and action histories have different token densities and functional roles. Video tokens are numerous and encode scene context, while action tokens are compact and encode ego-motion history. A single global cache would therefore be dominated by visual tokens and may under-preserve motion context. DriveWAM decomposes H_k into two bounded modality pools H_k^v and H_k^a, with $|H_k^v| \le B^v$ and $|H_k^a| \le B^a$, where B^v and B^a are the video and action memory budgets. This modality-aware design keeps both scene evidence and ego-motion history available during long-horizon rollout.

Relevance-redundancy retention. When a memory pool exceeds its budget, DriveWAM ranks cached tokens by both current relevance and memory complementarity. For modality $m \in \{v, a\}$, let Q_k^m denote the current query tokens of modality m, and let $\mathbf{k}_j^m$ be the cached key of token j in H_k^m. We measure relevance ρ_j^m by the average attention mass assigned to token j from current queries, and redundancy η_j^m by its average similarity to other cached keys:

$$\rho_j^m = \frac{1}{|Q_k^m|} \sum_{\mathbf{q} \in Q_k^m} \left[\mathrm{softmax}_{\ell \in H_k^m} \left(\frac{\mathbf{q}^\top \mathbf{k}_\ell^m}{\sqrt{d}} \right) \right]_j, \qquad \eta_j^m = \mathrm{mean}_{\ell \neq j} \cos(\mathbf{k}_j^m, \mathbf{k}_\ell^m), \tag{6}$$

where d is the transformer hidden dimension. The final retention score is:

$$s_j^m = \lambda \rho_j^m - (1 - \lambda) \eta_j^m, \tag{7}$$

where $\lambda \in [0, 1]$ balances relevance and redundancy, and tokens with low scores are evicted. As shown in Figure 3, this criterion has a natural driving-oriented interpretation: repeated road surfaces, sky, buildings, and other static background regions tend to be filtered out, while prediction-relevant cues such as moving vehicles and lane geometry are more likely to be retained.

Inference procedure. Selective KV memory is applied only at inference time and does not change the training objective or model parameters. During rollout, each transformer layer attends to the current chunk together with the bounded video and action memory pools. After chunk $k + 1$ is

Table 1: Comparison on NAVSIM v1. *: results with imitation learning. †: trained with multiple trajectory anchors from [54]. MV: multi-view cameras; SV: single-view camera; L: LiDAR.

Method	Ref	Sensors	NC ↑	DAC ↑	TTC ↑	C. ↑	EP ↑	PDMS ↑
Human	–	–	100	100	100	99.9	87.5	94.8
UniAD [55]	CVPR'23	MV	97.8	91.9	92.9	**100.0**	78.8	83.4
TransFuser [56]	TPAMI'23	MV & L	97.7	92.8	92.8	**100.0**	79.2	84.0
PARA-Drive [57]	CVPR'24	MV	97.9	92.4	93.0	99.8	79.3	84.0
LAW [58]	ICLR'25	SV	96.4	95.4	88.7	99.9	81.7	84.6
DiffusionDrive [59]	CVPR'25	MV & L	98.2	96.2	94.7	**100.0**	82.2	88.1
WoTE [60]	ICCV'25	MV & L	98.5	96.8	94.4	99.9	81.9	88.3
VLA-based Methods								
ReCogDrive* [4]	ICLR'26	MV	98.1	94.7	94.2	100.0	80.9	86.5
DriveVLA-W0 [3]	ICLR'26	SV	**98.7**	96.2	95.5	**100.0**	82.2	88.4
AutoVLA [37]	NeurIPS'25	MV	98.4	95.6	**98.0**	99.9	81.9	89.1
DriveDreamer-Policy [14]	arXiv'26	MV	98.4	97.1	95.1	**100.0**	83.5	89.2
DriveVLA-W0† [3]	ICLR'26	SV	98.7	**99.1**	95.3	99.3	83.3	90.2
WA-based Methods								
Epona [24]	ICCV'25	SV	97.9	95.1	93.8	99.9	80.4	86.2
WorldDrive [22]	arXiv'26	SV	98.4	95.8	95.2	99.8	83.3	89.0
DriveWAM	–	SV	98.3	**98.1**	95.2	**100.0**	**84.3**	**90.1**

Table 2: Comparison on our curated 1,000-clip test subset of the PhysicalAI-Autonomous-Vehicles benchmark. "# Params" denotes the number of model parameters; "PhysicalAI data" indicates whether the model was trained on data from this benchmark. *: original released checkpoint, which only supports up to 3s prediction.

Method	# Params	PhysicalAI data	ADE@3s ↓	FDE@3s ↓	ADE@4s ↓	FDE@4s ↓
VaVAM* [23]	1.3B	✗	2.31	4.32	–	–
VaVAM [23]	1.3B	✓	1.38	3.29	2.15	6.67
Alpamayo-1.5 [28]	10B	✓	0.80	2.31	1.44	4.18
DriveVLA-W0 [3]	8.5B	✓	0.80	2.06	1.22	3.24
DriveWAM	5B + 8B	✓	**0.47**	**1.35**	**0.83**	**2.47**

processed, the existing memory is ranked by the retention score, and the lowest-scored historical tokens are evicted to make room for the newly generated KVs ΔH^m_{k+1}. The modality pool is then updated as:

$$H^m_{k+1} \leftarrow \mathrm{Top}_{B^m - |\Delta H^m_{k+1}|}(H^m_k) \cup \Delta H^m_{k+1}, \qquad m \in \{v, a\}. \tag{8}$$

Here B^m denotes the memory budget for modality m. This training-free update keeps long-horizon inference bounded while retaining a compact approximation to full-history attention.

4 Experiments

4.1 Datasets

NAVSIM [27] is a standard end-to-end planning benchmark derived from OpenScene [61, 62], with 103k trainval samples and 12k test samples. Following the standard NAVSIM protocol, we report No at-fault Collisions (NC), Drivable Area Compliance (DAC), Time-To-Collision (TTC), Comfort (C.), Ego Progress (EP), and the overall Predictive Driver Model Score (PDMS).

PhysicalAI-Autonomous-Vehicles is a large-scale real-world driving benchmark released with Alpamayo-R1 [28]. It contains approximately 1,700 hours of driving logs, organized into 306,152 clips of 20 seconds each, with 153,625 clips for training, 90,928 for validation, and 61,599 for testing. We use the front-view camera stream and ego-motion labels. To focus on non-trivial driving scenarios, we use a VLM to tag each clip with a scene description and filter out simple scenes. Finally, we select 100k clips from the training split, and construct a curated 1,000-clip test subset from the test split. Details of the filtering procedure are provided in Appendix A. We report Average Displacement Error (ADE) and Final Displacement Error (FDE) over 3-second and 4-second future trajectories.

Table 3: Closed-loop comparison on 50 randomly selected reconstructed PhysicalAI scenes in AlpaSim. Metric names follow the Alpamayo-1.5 report.

Method	Close encounter all ↓	Close encounter at-fault ↓	Off-road ↓	AlpaSim score all ↑	AlpaSim score at-fault ↑
VaVAM [23]	46.0%	16.0%	28.0%	0.22	0.30
Alpamayo-1.5 [28]	28.0%	10.0%	20.0%	0.39	0.51
DriveWAM	**16.0%**	**8.0%**	**14.0%**	**0.47**	**0.55**

Table 4: Ablation of scene-evolving (SE) driving guidance under different training data scales on the PhysicalAI-Autonomous-Vehicles benchmark. ✗: fixed global prompt as text conditioning.

# Clips	# Iters	SE Guidance	ADE@4s ↓	FDE@4s ↓
4k	50k	✗	1.21	3.65
4k	50k	✓	**1.01**	**2.95**
20k	50k	✗	0.95	2.94
20k	50k	✓	**0.94**	**2.65**
100k	50k	✗	0.92	2.75
100k	50k	✓	**0.83**	**2.47**

4.2 Implementation Details

We build DriveWAM based on the code framework of [19]. DriveWAM uses Wan2.2-TI2V-5B [40] as the video backbone, initialized from the base checkpoint released by [19]. Unless otherwise specified, we fine-tune the full video diffusion transformer together with the newly introduced action and ego-state modules. The action encoder E_a and action decoder D_a are implemented as MLPs with hidden dimension 3072, and the ego-state features are encoded by a separate MLP. The scene-evolving guidance is generated by a frozen Qwen3-VL-8B [9], which is queried once per chunk. Details of the VLM prompt template are provided in Appendix B.

All models are trained at 256×448 resolution on 48 NVIDIA H20 GPUs. We use AdamW [63] with $\beta = (0.9, 0.95)$, weight decay 0.1, learning rate 1×10^{-5}, and per-device batch size 1. The action loss weight is set to $\beta_a = 1.0$. DriveWAM uses a 4-second chunk for video-action generation. On NAVSIM, we train for 100k iterations and decay the learning rate by a factor of 0.5 at 50k, 70k, and 90k iterations. Each sample uses the current frame as the condition and predicts a 4-second future horizon at 1 Hz. Since NAVSIM provides a single future planning horizon per sample, this setting reduces to one chunk-level prediction. On the PhysicalAI-Autonomous-Vehicles benchmark, we train for 50k iterations. Each training sample is a 12-second segment randomly cropped from a 20-second clip. The video stream is downsampled to 1 Hz, while ego actions remain at 10 Hz.

For inference, following [19], we use an Euler ODE solver with 3 steps for video tokens and 10 steps for action tokens. The video solver integrates the flow trajectory from $\tau = 1$ to $\tau = 0.6$, while the action solver integrates from $\tau = 1$ to $\tau = 0$. For selective KV memory, we follow FlowCache [26] and set $\lambda = 0.07$. The video and action cache capacities are set to 448 and 160 tokens, respectively.

4.3 Main Results

NAVSIM. We compare DriveWAM against state-of-the-art end-to-end planners on NAVSIM v1, including classical end-to-end pipelines [55–60], VLA-based policies [4, 3, 37, 14], and WA-based methods [24, 22]. As shown in Table 1, DriveWAM achieves a PDMS of 90.1 using only a single front-view camera, outperforming all competing methods under comparable training settings. We attribute this to the underlying video generative backbone, which provides effective spatio-temporal priors for modeling scene geometry, motion dynamics, and fine-grained action prediction. We provide additional NAVSIM v2 results in Appendix E.

PhysicalAI-Autonomous-Vehicles. We evaluate DriveWAM on the PhysicalAI-Autonomous-Vehicles benchmark, comparing against the WA-based VaVAM [23] and two VLA-based methods, Alpamayo-1.5 [28] and DriveVLA-W0 [3]. For consistency, all methods use a single front-view camera and output a single trajectory at inference. For VaVAM, we evaluate both its released checkpoint,

Table 5: Ablation of video backbone initialization and joint video supervision. All models are trained on 100k clips for 50k iterations.

Pretrained init.	Video sup.	ADE@4s ↓	FDE@4s ↓
✗	✗	1.40	4.21
✓	✗	1.23	3.79
✗	✓	1.10	3.26
✓	✓	**0.83**	**2.47**

Table 6: Ablation of KV memory strategies. Planning is evaluated on 20s clips; memory and GFLOPs are profiled on a 300s rollout.

	Planning		Efficiency	
KV memory	ADE@4s ↓	FDE@4s ↓	Mem. (GB) ↓	GFLOPs ↓
Full	**0.83**	**2.47**	3.07	17.37
FIFO	1.40	3.47	**0.25**	**1.05**
Selective	0.89	2.52	**0.25**	1.44



Figure 4: Qualitative results on NAVSIM (left) and PhysicalAI-Autonomous-Vehicles benchmark (right). The predicted ego trajectories are consistent with the jointly generated future scenes.

trained on approximately 1,700 hours of OpenDV [64] driving data, and a variant fine-tuned on the same 100k PhysicalAI training clips used by DriveWAM. DriveVLA-W0 is also trained on these clips. Alpamayo-1.5 is evaluated using its released checkpoint, trained on roughly 80,000 hours of data, including the PhysicalAI-Autonomous-Vehicles training set. As reported in Table 2, DriveWAM achieves ADE/FDE of 0.47/1.35 at 3 seconds and 0.83/2.47 at 4 seconds, substantially outperforming all baselines at both prediction horizons. We also present results on the full official test split and under a best-of-six evaluation protocol in Appendix E.

AlpaSim. We evaluate DriveWAM in closed loop using AlpaSim [65], comparing against Alpamayo-1.5 and VaVAM. All methods use a single front-view camera. As shown in Table 3, DriveWAM achieves the lowest close encounter rates and the highest AlpaSim scores.

Qualitative results. Figure 4 visualizes future scenes and ego trajectories jointly generated by DriveWAM. Additional qualitative examples are provided in Appendix D.

4.4 Ablation Study

We conduct ablation studies on the PhysicalAI-Autonomous-Vehicles benchmark to investigate the individual components of DriveWAM. Unless otherwise noted, all ablation models are trained on 100k clips for 50k iterations under the same optimization settings as in the main results.

Scene-evolving Driving Guidance. Table 4 studies the contribution of injecting chunk-specific VLM guidance. Replacing the global prompt with scene-evolving guidance consistently improves trajectory prediction at every training data scale, reducing ADE@4s from 1.21 to 1.01 with 4k clips and from 0.92 to 0.83 with 100k clips, while also yielding consistent reductions in FDE@4s. These results indicate that high-level scene reasoning provides a complementary semantic conditioning to the low-level WA backbone. We also observe that the benefit does not vanish as training data grows. Appendix B provides qualitative examples of guidance evolving with scene context and route intent.

Data Scaling. We investigate the data scalability of DriveWAM by varying the training set size from 4k to 20k and 100k clips under a fixed 50k-iteration training procedure. As shown in Table 4 and Figure 5, both ADE@4s and FDE@4s improve consistently with more data, regardless of whether scene-evolving guidance is applied. This consistent scaling trend reflects the effectiveness of the video-action modeling as a scalable policy foundation, and suggests that DriveWAM has not yet saturated at the current data scale.

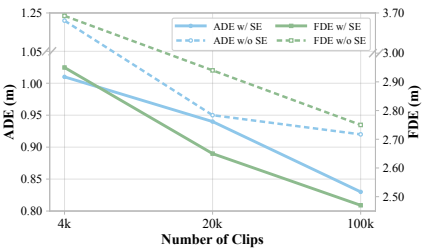


Figure 5: Data scaling on PhysicalAI-Autonomous-Vehicles.

Video Foundation Model Adaptation. We ablate DriveWAM's capability by removing the pretrained video-backbone initialization and the joint video flow-matching supervision. As reported in Table 5,

training entirely from scratch removes the large-scale spatio-temporal priors inherited from video pretraining, and degrades ADE@4s/FDE@4s to $1.10/3.26$. Initializing from the pretrained backbone but removing video supervision also performs poorly, yielding $1.23/3.79$, suggesting that action-only adaptation fails to preserve the generative video priors needed for WA policy learning. The full configuration combines pretrained initialization with joint video-action flow-matching supervision and achieves the best performance.

Video Quality and Planning. To examine the relationship between video quality and planning performance, we report PSNR, LPIPS, FID, and FVD for generated videos in Appendix E.3. The results show that pretrained initialization and joint video supervision improve both video quality and planning accuracy. Across data scales, however, better video metrics do not always correspond to more accurate planning. This mismatch may arise because generic video metrics emphasize overall visual appearance, including sky and building textures, without fully capturing the road geometry and traffic dynamics most relevant to planning.

Selective KV Memory. Table 6 compares three inference-time memory strategies for autoregressive rollout. Full KV caching retains the entire video-action history, while FIFO and our selective KV memory operate under the fixed-size cache budget. As shown in the 1^{st} and 3^{rd} rows, selective KV memory largely closes the accuracy gap to full caching, achieving $0.89/2.52$ ADE@4s/FDE@4s, while FIFO degrades substantially to $1.40/3.47$. To examine the long-horizon overhead of the memory module, we further profile each strategy on a 300-second rollout, reporting KV memory summed over all DiT layers and attention GFLOPs of one causal self-attention layer. As presented in Table 6, full caching requires 3.07 GB memory and 17.37 GFLOPs per step, whereas selective KV memory reduces them to 0.25 GB and 1.44 GFLOPs, yielding over $12\times$ reductions.

5 Conclusion

We present DriveWAM, a unified world-action policy that adapts a pretrained video foundation model directly into an end-to-end driving policy. DriveWAM introduces scene-evolving driving guidance that injects chunk-specific semantic intent through temporally localized cross-attention, and selective KV memory that maintains modality-aware video and action memory pools via relevance-redundancy selection at inference time. Experiments on NAVSIM and the PhysicalAI-Autonomous-Vehicles benchmark show that DriveWAM achieves strong planning performance, and a data-scaling study from 4k to 100k clips further confirms its scalability.

Limitations and broader impacts. DriveWAM currently operates from a single front-view camera, leaving surround-view extension as a natural next step. Its total size of around 13B parameters across the video backbone and the VLM also poses deployment challenges, motivating future work on compression and distillation. We hope DriveWAM inspires further efforts on adapting video foundation models for autonomous driving and other embodied AI tasks.

Acknowledgments

This work is supported by the Guangdong Basic and Applied Basic Research Foundation (Grant No. 2025A1515011434).

References

[1] 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.

[2] 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.

[3] Yingyan Li, Shuyao Shang, Weisong Liu, Bing Zhan, Haochen Wang, Yuqi Wang, Yuntao Chen, Xiaoman Wang, An Yasong, Chufeng Tang, Lu Hou, Lue Fan, and Zhaoxiang Zhang. DriveVLA-w0: World models amplify data scaling law in autonomous driving. In *ICLR*, 2026.

[4] 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. In *ICLR*, 2026.

[5] Katrin Renz, Long Chen, Elahe Arani, and Oleg Sinavski. Simlingo: Vision-only closed-loop autonomous driving with language-action alignment. In *CVPR*, 2025.

[6] Zhenjie Yang, Yilin Chai, Xiaosong Jia, Qifeng Li, Yuqian Shao, Xuekai Zhu, Haisheng Su, and Junchi Yan. Drivemoe: Mixture-of-experts for vision-language-action model in end-to-end autonomous driving. In *CVPR*, 2026.

[7] Xiangbo Gao, Yuheng Wu, Rujia Wang, Chenxi Liu, Yang Zhou, and Zhengzhong Tu. Langcoop: Collaborative driving with language. In *CVPR*, 2025.

[8] 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.

[9] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[10] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023.

[11] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

[12] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. In *NeurIPS*, 2022.

[13] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[14] Yang Zhou, Xiaofeng Wang, Hao Shao, Letian Wang, Guosheng Zhao, Jiangnan Shao, Jiagang Zhu, Tingdong Yu, Zheng Zhu, Guan Huang, et al. Drivedreamer-policy: A geometry-grounded world-action model for unified generation and planning. *arXiv preprint arXiv:2604.01765*, 2026.

[15] Shuang Zeng, Xinyuan Chang, Mengwei Xie, Xinran Liu, Yifan Bai, Zheng Pan, Mu Xu, and Xing Wei. Futuresightdrive: Thinking visually with spatio-temporal cot for autonomous driving. In *NeurIPS*, 2025.

[16] Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, and Jinwei Gu. Cosmos policy: Fine-tuning video models for visuomotor control and planning. In *ICLR*, 2026.

[17] Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

[18] Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026.

[19] Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, et al. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026.

[20] Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jie Li, Jindi Lv, Jingyu Liu, et al. Gigaworld-policy: An efficient action-centered world–action model. *arXiv preprint arXiv:2603.17240*, 2026.

[21] Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. In *ICML*, 2025.

[22] Xingtai Gui, Meijie Zhang, Tianyi Yan, Wencheng Han, Jiahao Gong, Feiyang Tan, Cheng-zhong Xu, and Jianbing Shen. Bridging scene generation and planning: Driving with world model via unifying vision and motion representation. *arXiv preprint arXiv:2603.14948*, 2026.

[23] Florent Bartoccioni, Elias Ramzi, Victor Besnier, Shashanka Venkataramanan, Tuan-Hung Vu, Yihong Xu, Loick Chambon, Spyros Gidaris, Serkan Odabas, David Hurych, Renaud Marlet, Alexandre Boulch, Mickael Chen, Eloi Zablocki, Andrei Bursuc, Eduardo Valle, and Matthieu Cord. Vavim and vavam: Autonomous driving through video generative modeling. *arXiv preprint arXiv:2502.15672*, 2025.

[24] Kaiwen Zhang, Zhenyu Tang, Xiaotao Hu, Xingang Pan, Xiaoyang Guo, Yuan Liu, Jingwei Huang, Li Yuan, Qian Zhang, Xiao-Xiao Long, et al. Epona: Autoregressive diffusion world model for autonomous driving. In *ICCV*, 2025.

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

[26] Yuexiao Ma, Xuzhe Zheng, Jing Xu, Xiwei Xu, Feng Ling, Xiawu Zheng, Huafeng Kuang, Huixia Li, Xing Wang, Xuefeng Xiao, et al. Flow caching for autoregressive video generation. In *ICLR*, 2026.

[27] Daniel Dauner, Marcel Hallgarten, Tianyu Li, Xinshuo Weng, Zhiyu Huang, Zetong Yang, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, et al. Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking. In *NeurIPS*, 2024.

[28] Yan Wang, Wenjie Luo, Junjie Bai, Yulong Cao, Tong Che, Ke Chen, Yuxiao Chen, Jenna Diamond, Yifan Ding, Wenhao Ding, et al. Alpamayo-r1: Bridging reasoning and action prediction for generalizable autonomous driving in the long tail. *arXiv preprint arXiv:2511.00088*, 2025.

[29] Zhenhua Xu, Yujia Zhang, Enze Xie, Zhen Zhao, Yong Guo, Kwan-Yee K Wong, Zhenguo Li, and Hengshuang Zhao. Drivegpt4: Interpretable end-to-end autonomous driving via large language model. *IEEE Robotics and Automation Letters*, 2024.

[30] Xiaoyu Tian, Junru Gu, Bailin Li, Yicheng Liu, Zhiyong Zhao, Yang Wang, Kun Zhan, Peng Jia, Xianpeng Lang, and Hang Zhao. Drivevlm: The convergence of autonomous driving and large vision-language models. In *CoRL*, 2024.

[31] Bo Jiang, Shaoyu Chen, Bencheng Liao, Xingyu Zhang, Wei Yin, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Senna: Bridging large vision-language models and end-to-end autonomous driving. *arXiv preprint arXiv:2410.22313*, 2024.

[32] 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 *ECCV*, 2024.

[33] Wenhai Wang, Jiangwei Xie, ChuanYang Hu, Haoming Zou, Jianan Fan, Wenwen Tong, Yang Wen, Silei Wu, Hanming Deng, Zhiqi Li, et al. Drivemlm: Aligning multi-modal large language models with behavioral planning states for autonomous driving. *arXiv preprint arXiv:2312.09245*, 2023.

[34] Jiageng Mao, Junjie Ye, Yuxi Qian, Marco Pavone, and Yue Wang. A language agent for autonomous driving. *arXiv preprint arXiv:2311.10813*, 2023.

[35] Shihao Wang, Zhiding Yu, Xiaohui Jiang, Shiyi Lan, Min Shi, Nadine Chang, Jan Kautz, Ying Li, and Jose M. Alvarez. OmniDrive: A holistic vision-language dataset for autonomous driving with counterfactual reasoning. In *CVPR*, 2025.

[36] Jyh-Jing Hwang, Runsheng Xu, Hubert Lin, Wei-Chih Hung, Jingwei Ji, Kristy Choi, Di Huang, Tong He, Paul Covington, Benjamin Sapp, et al. Emma: End-to-end multimodal model for autonomous driving. *arXiv preprint arXiv:2410.23262*, 2024.

[37] 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. In *NeurIPS*, 2025.

[38] Chuanxia Zheng, Long Tung Vuong, Jianfei Cai, and Dinh Phung. MoVQ: Modulating quantized vectors for high-fidelity image generation. In *NeurIPS*, 2022.

[39] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023.

[40] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[41] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Yuxuan Zhang, Weihan Wang, Yean Cheng, Bin Xu, Xiaotao Gu, Yuxiao Dong, and Jie Tang. Cogvideox: Text-to-video diffusion models with an expert transformer. In *ICLR*, 2025.

[42] Shuhan Tan, Kashyap Chitta, Yuxiao Chen, Ran Tian, Yurong You, Yan Wang, Wenjie Luo, Yulong Cao, Philipp Krahenbuhl, Marco Pavone, et al. Latent chain-of-thought world modeling for end-to-end driving. In *CVPR*, 2026.

[43] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. In *NeurIPS*, 2017.

[44] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

[45] Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *NeurIPS*, 2025.

[46] Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *NeurIPS*, 2024.

[47] Tianwei Yin, Qiang Zhang, Richard Zhang, William T Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast autoregressive video diffusion models. In *CVPR*, 2025.

[48] Haocheng Xi, Shuo Yang, Yilong Zhao, Chenfeng Xu, Muyang Li, Xiuyu Li, Yujun Lin, Han Cai, Jintao Zhang, Dacheng Li, Jianfei Chen, Ion Stoica, Kurt Keutzer, and Song Han. Sparse video-gen: Accelerating video diffusion transformers with spatial-temporal sparsity. In *ICML*, 2025.

[49] Shuo Yang, Haocheng Xi, Yilong Zhao, Muyang Li, Jintao Zhang, Han Cai, Yujun Lin, Xiuyu Li, Chenfeng Xu, Kelly Peng, Jianfei Chen, Song Han, Kurt Keutzer, and Ion Stoica. Sparse videogen2: Accelerate video generation with sparse attention via semantic-aware permutation. In *NeurIPS*, 2025.

[50] Kumara Kahatapitiya, Haozhe Liu, Sen He, Ding Liu, Menglin Jia, Chenyang Zhang, Michael S Ryoo, and Tian Xie. Adaptive caching for faster video generation with diffusion transformers. In *ICCV*, 2025.

[51] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023.

[52] Jihwan Kim, Junoh Kang, Jinyoung Choi, and Bohyung Han. Fifo-diffusion: Generating infinite videos from text without training. *NeurIPS*, 2024.

[53] Chaoda Zheng, Sean Li, Jinhao Deng, Zhennan Wang, Shijia Chen, Liqiang Xiao, Ziheng Chi, Hongbin Lin, Kangjie Chen, Boyang Wang, et al. X-world: Controllable ego-centric multi-camera world models for scalable end-to-end driving. *arXiv preprint arXiv:2603.19979*, 2026.

[54] 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.

[55] 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*, 2023.

[56] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *TPAMI*, 2023.

[57] Xinshuo Weng, Boris Ivanovic, Yan Wang, Yue Wang, and Marco Pavone. Para-drive: Parallelized architecture for real-time autonomous driving. In *CVPR*, 2024.

[58] Yingyan Li, Lue Fan, Jiawei He, Yuqi Wang, Yuntao Chen, Zhaoxiang Zhang, and Tieniu Tan. Enhancing end-to-end autonomous driving with latent world model. In *ICLR*, 2025.

[59] 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*, 2025.

[60] Yingyan Li, Yuqi Wang, Yang Liu, Jiawei He, Lue Fan, and Zhaoxiang Zhang. End-to-end driving with online trajectory evaluation via bev world model. In *ICCV*, 2025.

[61] OpenScene Contributors. Openscene: The largest up-to-date 3d occupancy prediction benchmark in autonomous driving. `https://github.com/OpenDriveLab/OpenScene`, 2023.

[62] Napat Karnchanachari, Dimitris Geromichalos, Kok Seang Tan, Nanxiang Li, Christopher Eriksen, Shakiba Yaghoubi, Noushin Mehdipour, Gianmarco Bernasconi, Whye Kit Fong, Yiluan Guo, et al. Towards learning-based planning: The nuplan benchmark for real-world autonomous driving. In *ICRA*, 2024.

[63] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[64] Jiazhi Yang, Shenyuan Gao, Yihang Qiu, Li Chen, Tianyu Li, Bo Dai, Kashyap Chitta, Penghao Wu, Jia Zeng, Ping Luo, et al. Generalized predictive model for autonomous driving. In *CVPR*, 2024.

[65] NVIDIA, Yulong Cao, Riccardo de Lutio, Sanja Fidler, Guillermo Garcia Cobo, Zan Gojcic, Maximilian Igl, Boris Ivanovic, Peter Karkus, Janick Martinez Esturo, Marco Pavone, Aaron Smith, Ellie Tanimura, Michal Tyszkiewicz, Michael Watson, Qi Wu, and Le Zhang. Alpasim: A modular, lightweight, and data-driven research simulator for autonomous driving, October 2025. URL `https://github.com/NVlabs/alpasim`.

[66] Wei Cao, Marcel Hallgarten, Tianyu Li, Daniel Dauner, Xunjiang Gu, Caojun Wang, Yakov Miron, Marco Aiello, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, Andreas Geiger, and Kashyap Chitta. Pseudo-simulation for autonomous driving. In *CoRL*, 2025.

[67] 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.

[68] 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. In *ECCV*. Springer, 2026.

[69] Chun-Peng Chang, Chen-Yu Wang, Julian Schmidt, Holger Caesar, and Alain Pagani. Seeing clearly, forgetting deeply: Revisiting fine-tuned video generators for driving simulation. *arXiv preprint arXiv:2508.16512*, 2025.

A Dataset Curation

The PhysicalAI-Autonomous-Vehicles benchmark contains roughly 1,700 hours of driving organized into 306,152 20-second clips. To focus evaluation and training on non-trivial driving scenarios, we tag every clip with a frozen Qwen3-VL-8B [9] and use the resulting tags to construct a 100k-clip training subset, and a curated 1,000-clip test subset with balanced coverage of rare and ordinary scenarios.

Scene tagging. For each clip, we uniformly sample 20 frames from the front-view stream and pass them to Qwen3-VL-8B with four structured prompts. Each prompt focuses on one facet of driving complexity:

- **Scene attributes**: weather (clear/rainy/snowy/foggy), lighting (day/dusk/night/tunnel transition/strong backlight), road type (urban/highway/ramp/intersection/etc.), traffic density, and ego behavior.
- **Vulnerable road-user events**: whether the scene contains pedestrian crossing, jaywalking, occluded pedestrian popout, children or elderly participants, cyclist conflict, or crowd.
- **Vehicle interaction events**: whether the scene contains cut-in, cut-out, sudden braking ahead, wrong-way vehicle, large-vehicle occlusion, emergency vehicle, door opening, or stopped/broken vehicle.
- **Intersection and long-tail events**: whether the scene contains unprotected left turn, roundabout, irregular intersection, traffic-police gesture, road debris, accident scene, construction, animal on road, water puddle, or railway crossing, together with the traffic-light state.

The four prompts are run sequentially on the same sampled frames and merged into a single per-clip record. We additionally compute a scalar interest score by summing rule-based weights over detected event tags. In practice, rare or safety-critical events receive larger weights, e.g., accident scenes (5.0), occluded pedestrian popouts (4.0), animals on the road (3.5), and traffic-police gestures (3.0), while frequent or lower-impact attributes receive smaller weights (0.5–1.5). Figure 6 shows representative tagged clips with their detected attributes and interest scores.

Training subset. The training subset is curated from the tagged training split through a two-stage procedure. We first retain all clips with interest score no smaller than 2.0, preserving rare-event and interaction-rich cases. We then uniformly sample 50% of the remaining lower-score clips, so that ordinary driving scenarios are still represented without dominating the training distribution. For the data-scaling study, we sample 20k and 4k subsets from this 100k subset.

Test subset. The test subset contains 1,000 clips and is constructed from the tagged test split to cover both long-tail and ordinary driving scenarios. We combine three sources:

- **Rare-event clips**: a tag is treated as rare if it appears in fewer than 1% of the test clips. For each rare tag, we select up to 30 top-scoring clips that contain it, covering events such as accident scenes, animals on the road, occluded pedestrian popouts, traffic-police gestures, and railway crossings.
- **High-interest clips**: clips above the 75th percentile of the interest-score distribution are grouped by weather, lighting, and road type. We assign an approximately equal quota to each group and select the highest-scoring clips within each group until the target size is reached.
- **Common-scene clips**: 200 clips uniformly sampled from below the high-interest threshold to serve as ordinary-driving controls.

The selected clips are merged to form the final 1,000-clip test set.

B VLM Guidance Details

This section details the pipeline that produces the chunk-specific guidance g_k used in Sec. 3.2. The pipeline operates in two stages. First, we classify the route of each upcoming 4-second chunk from ground-truth ego pose, producing a route command. Second, we prompt a frozen Qwen3-VL-8B with the route command, the front-camera frame at the end of the latest chunk, and a BEV visualization of

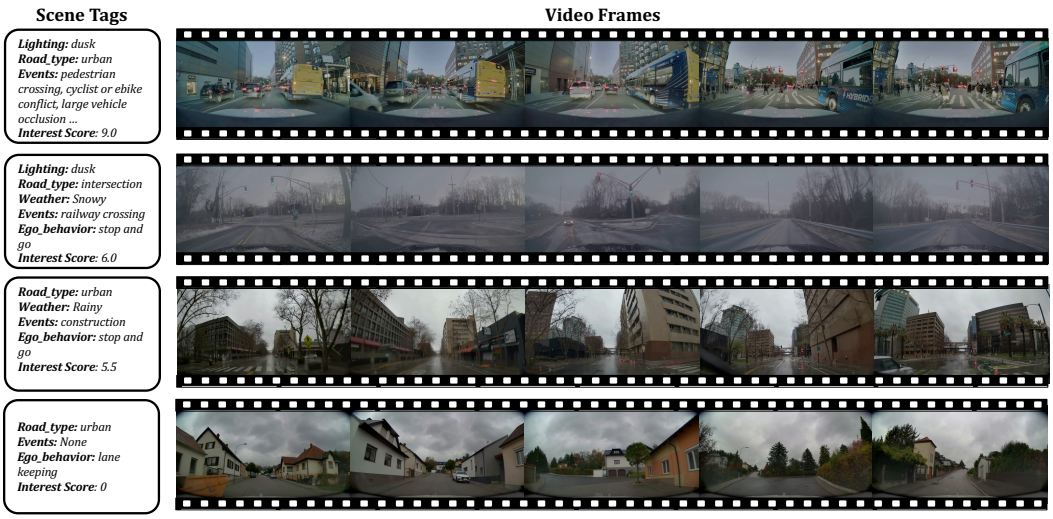


Figure 6: Representative scene tagging results for dataset curation. For each clip, the left panel shows Qwen3-VL-8B detected scene attributes, events, and the resulting interest score, while the right panel shows sampled front-view frames. High-score clips capture rare or interaction-rich scenarios, whereas low-score clips represent ordinary driving.

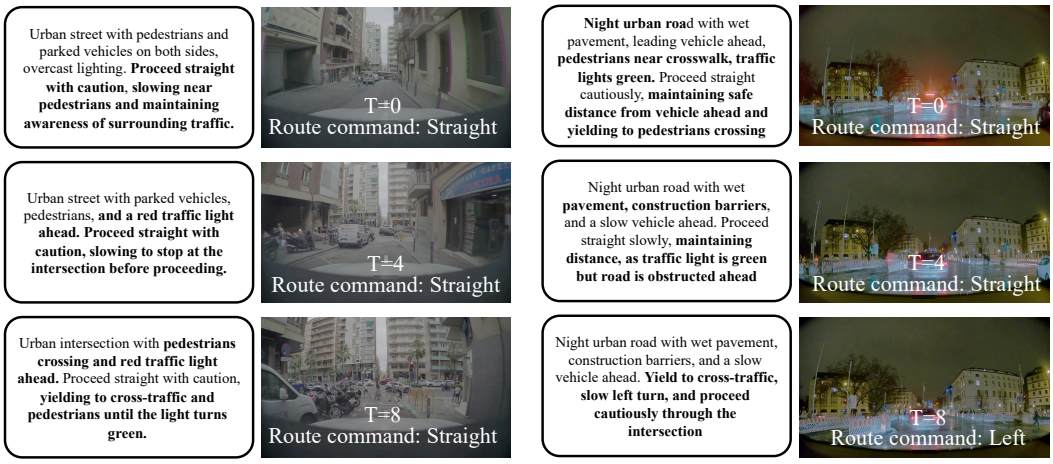


Figure 7: Examples of scene-evolving VLM guidance. The guidance adapts to changing scene context and route intent, such as pedestrians, traffic lights, construction barriers.

the ego trajectory from the previous 4-second chunk, asking it to produce a concise two-sentence guidance for the upcoming chunk. Figure 7 shows representative guidance examples, where the generated text evolves with the latest observation and route command.

Route command. Each chunk is assigned a high-level route command from {`straight`, `left`, `right`}. As explicit route annotations are unavailable, we construct this coarse command from the route/ego-yaw change for labeling purposes. Specifically, this command is derived from the yaw change of the ego vehicle over the chunk. Let R_0 and R_1 denote the ego rotations at the beginning and end of a chunk. We compute the relative yaw from $R_0^\top R_1$ and assign the command as `left` if the yaw change is larger than 15°, `right` if it is smaller than -15°, and `straight` otherwise. The command only specifies directional intent and does not contain future positions, velocities, distances, or trajectory coordinates.

Prompt template. The prompt template used for chunk-level guidance generation is shown below. The route command and visual inputs are filled at runtime.

Prompt template for VLM guidance

You are a navigation assistant for an autonomous-driving dataset. You generate short navigation guidance for an upcoming 4-second driving window. You do not see the future window; you only see the current road conditions, the recent ego trajectory when available, and a high-level route command.

Inputs.

- Route command for the upcoming window: `straight`, `left`, or `right`. Treat it as an authoritative navigation instruction.
- Latest causally available front-camera frame before the upcoming 4-second window, showing the current road conditions.
- BEV trajectory map of the previous 4-second window when available, showing recent ego motion and speed.

Output format. Output exactly two sentences, under 50 words in total. Do not use labels, bullets, markdown, or extra paragraphs. Use present tense only.

Sentence 1. Describe the current road context visible in the provided frame, including road type, traffic participants, traffic-light state if visible, weather, and lighting.

Sentence 2. Describe the ego navigation guidance for the upcoming window. Jointly reason from the road context, the previous BEV trajectory, and the route command. The direction must be consistent with the route command, and the caution level should reflect the traffic conditions. Capture required interactions such as yielding, waiting, or merging. Use qualitative language only; do not include numbers, units, distances, coordinates, or low-level trajectory values.

C Efficiency Analysis

We analyze the per-chunk inference cost of DriveWAM and compare it against Alpamayo-1.5 [28] on a single NVIDIA H20 GPU. As shown in Table 7, each inference pass consists of three stages: (1) VLM guidance generation, (2) video generation, and (3) action denoising.

VLM guidance. DriveWAM queries a frozen Qwen3-VL-8B once per 4-second chunk, taking 125 ms with the default vLLM compilation. Because the guidance is generated at the chunk boundary rather than per frame, the cost is amortized over the entire chunk. Alpamayo-1.5 processes a substantially larger number of visual tokens per query, which accounts for its higher VLM latency of 570 ms.

Video generation. DriveWAM generates a 4-second video clip using a 3-step Euler ODE solver over the video tokens, taking 372 ms. Alpamayo-1.5 does not perform explicit video generation.

Action denoising. By default, DriveWAM uses 10 denoising steps for action tokens, taking 765 ms. We find that reducing the steps from 10 to 5 incurs negligible change in trajectory metrics, while reducing action denoising time to 374 ms. The 5-step variant (DriveWAM*) brings the total per-chunk cost to approximately 871 ms, comparable to Alpamayo-1.5's 900 ms, while additionally producing a jointly generated future video.

Table 7: Per-chunk inference cost and trajectory prediction accuracy on a single H20 GPU. * indicates action denoising steps reduced from 10 to 5.

Method	VLM (ms)	Video Gen (ms)	Action (ms)	ADE@4s ↓	FDE@4s ↓
Alpamayo-1.5	570	—	330	1.44	4.18
DriveWAM (Ours)	125	372	765	**0.83**	2.47
DriveWAM* (Ours)	125	372	374	0.84	**2.45**

D Additional Qualitative Results

We present additional qualitative results to complement the main-paper visualization. Figure 8 shows representative examples from both NAVSIM and the PhysicalAI-Autonomous-Vehicles benchmark, spanning driving conditions and road layouts.

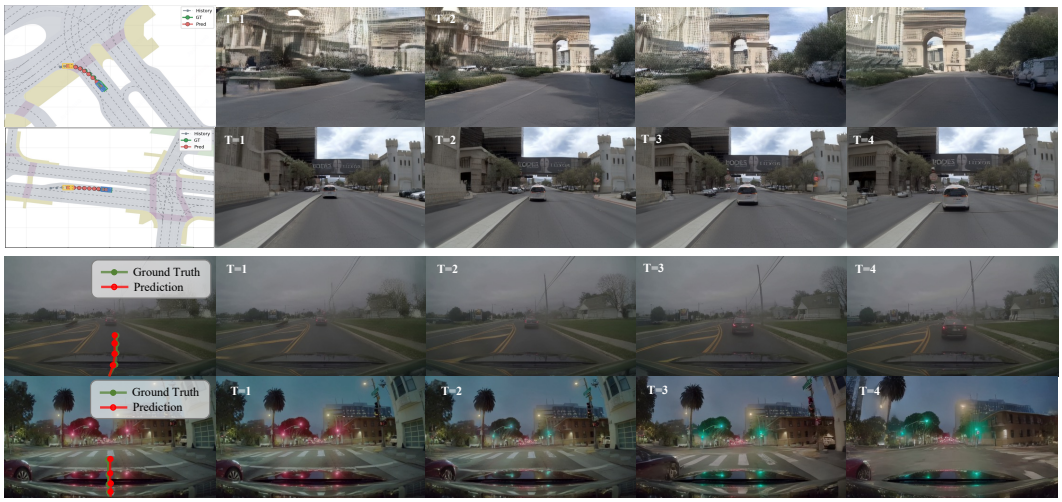


Figure 8: Qualitative results on NAVSIM (top two rows) and PhysicalAI-Autonomous-Vehicles (bottom two rows) benchmarks. Each row shows the predicted ego trajectory alongside the jointly generated future frames at T=1,2,3,4.

NAVSIM qualitative results. Each example shows a BEV map on the left, where the red trajectory is the DriveWAM prediction and the blue trajectory is the ground-truth. The yellow vehicle icon denotes the starting ego, the blue vehicle icon denotes the predicted ending ego, and the green vehicle icon denotes the ground-truth ending ego. In both cases, the predicted trajectory aligns closely with the ground truth despite the complexity of the surroundings, and the generated video maintains photometric and geometric consistency across the four future timesteps.

PhysicalAI-Autonomous-Vehicles qualitative results. Each example overlays the ground-truth and predicted ego trajectories on the current front-view frame. These results are consistent with the strong quantitative performance and further demonstrate that the joint video-action generation provides a coherent, physically plausible world model that supports accurate long-horizon planning across diverse real-world conditions.

E Additional Experiments

We evaluate DriveWAM on the NAVSIM v2 [66] navtest split and the two-stage navhard setting. As shown in Tables 8 and 9, DriveWAM achieves EPDMS scores of 87.2 and 35.6, respectively, outperforming the compared methods.

Table 8: Results on NAVSIM v2 navtest.

Method	EPDMS ↑
DiffusionDriveV2 [67]	85.5
SparseDriveV2 [68]	86.7
DriveWAM	**87.2**

Table 9: Comparison on NAVSIM v2 navhard under the two-stage evaluation protocol.

Method	Stage 1 ↑	Stage 2 ↑	EPDMS ↑
DiffusionDrive [59]	66.7	40.5	27.5
WorldDrive [22]	–	–	34.9
DriveWAM	**74.6**	**47.7**	**35.6**

E.1 Evaluation on the Full PhysicalAI Test Split

We evaluate DriveWAM and Alpamayo-1.5 on all 61,599 official test clips. Table 10 shows that DriveWAM's advantage persists on the full test split.

E.2 Six-Sample Trajectory Evaluation

We compare DriveWAM and Alpamayo-1.5 [28] using minADE_6 and minFDE_6 over six trajectory samples on our curated 1,000-clip PhysicalAI test subset. DriveWAM generates six trajectories

Table 10: Comparison on the full official PhysicalAI-Autonomous-Vehicles test split.

Method	ADE@3s ↓	FDE@3s ↓	ADE@4s ↓	FDE@4s ↓
Alpamayo-1.5	0.90	2.58	1.58	4.55
DriveWAM	**0.50**	**1.45**	**0.91**	**2.78**

from different noise initializations and still achieves lower minADE_6 and minFDE_6 at both horizons (Table 11).

Table 11: Six-sample trajectory evaluation on the curated 1,000-clip PhysicalAI test subset.

Method	minADE_6@3s ↓	minFDE_6@3s ↓	minADE_6@4s ↓	minFDE_6@4s ↓
Alpamayo-1.5	0.45	1.34	0.82	2.42
DriveWAM	**0.32**	**1.00**	**0.61**	**1.78**

E.3 Video Quality and Planning

We report PSNR, LPIPS, FID, and FVD alongside planning errors on the curated 1,000-clip PhysicalAI test subset. All models are trained for 50k iterations.

Pretraining and Video Supervision. Table 12 shows that pretrained initialization and joint video supervision benefit both video generation and planning. With video supervision, pretrained initialization improves FID/FVD from 111.37/821.22 to 66.21/367.99, while reducing ADE/FDE@4s from 1.10/3.26 to 0.83/2.47. Removing video supervision from the pretrained model increases FID/FVD to 328.17/12899.74 and ADE/FDE@4s to 1.23/3.79. These comparisons support retaining the future-video objective during policy adaptation to make effective use of pretrained video representations.

Table 12: Effect of pretrained video-backbone initialization and joint video supervision on planning and video quality. All models are trained on 100k clips.

Pretrained init.	Video sup.	ADE@4s ↓	FDE@4s ↓	PSNR ↑	LPIPS ↓	FID ↓	FVD ↓
✓	✗	1.23	3.79	13.94	0.78	328.17	12899.74
✗	✓	1.10	3.26	22.40	0.39	111.37	821.22
✓	✓	**0.83**	**2.47**	**23.28**	**0.33**	**66.21**	**367.99**

Data Scale and SE Guidance. Table 13 compares models across training data scales and guidance settings. Across data scales, however, better video metrics do not always correspond to lower planning errors. With SE guidance, the 20k model obtains lower FID/FVD (60.33/335.94) than the 100k model (66.21/367.99), despite higher ADE/FDE@4s (0.94/2.65 versus 0.83/2.47). This mismatch may arise because generic video metrics are sensitive to overall visual appearance, including sky and building textures, without fully capturing the road geometry and traffic dynamics most relevant to planning. This interpretation is consistent with prior findings that improved visual fidelity in driving-video generators does not necessarily imply more accurate modeling of dynamic elements [69].

Table 13: Effect of training data scale and scene-evolving (SE) guidance on planning and video quality. ✗: fixed global prompt. Best results within each data scale are bold.

# Clips	SE Guidance	ADE@4s ↓	FDE@4s ↓	PSNR ↑	LPIPS ↓	FID ↓	FVD ↓
20k	✗	0.95	2.94	22.91	0.35	73.81	392.71
20k	✓	**0.94**	**2.65**	**23.14**	**0.33**	**60.33**	**335.94**
100k	✗	0.92	2.75	23.15	0.34	71.86	393.49
100k	✓	**0.83**	**2.47**	**23.28**	**0.33**	**66.21**	**367.99**

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 abstract and introduction state the main contributions, including unified policy learning, chunk-specific semantic guidance, selective KV memory, and evaluation on two autonomous driving benchmarks. These claims are supported by the main results and ablation studies.

   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: The paper discusses limitations in the limitations section, including the scope of evaluation and practical deployment considerations.

   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 present theoretical results or formal proofs.

   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 main experiments section provides the training objectives, datasets, evaluation metrics, and training settings needed to understand and reproduce the experimental 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: [Yes]

   Justification: We use publicly available datasets and release our code and model checkpoints at `https://github.com/chenshi3/DriveWAM`. The repository provides instructions for environment setup, data preparation, training, and evaluation.

   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: The experiments section describes the datasets, splits, metrics, training hyperparameters, hardware setup, and inference settings, with further details in the appendix.

   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 paper reports results following standard benchmark evaluation protocols and does not include error bars or statistical significance tests.

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: [Yes]

   Justification: The paper reports the GPU type and number used for training in Section 4.

   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 follows 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: The paper discusses potential impacts in Section 5.

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 release new datasets or high-risk pretrained generative models.

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: The paper credits the datasets, pretrained models, and code frameworks used in the work through citations, and we follow their respective licenses and terms of use.

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: [Yes]

    Justification: The new assets, including code and model checkpoints, will be made publicly available upon completion of the approval process.

    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 experiments or 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 experiments or 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: [Yes]

Justification: We use a frozen Qwen3-VL-8B as part of the method. It generates chunk-specific scene-evolving driving guidance. LLMs were also used for writing and editing assistance.

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.