SV-WAM: AN EFFICIENT SURROUND-VIEW WORLD-ACTION MODEL FOR END-TO-END AUTONOMOUS DRIVING

Anonymous authors
Paper under double-blind review

ABSTRACT

World models (WMs) have demonstrated strong potential for end-to-end autonomous driving by learning predictive representations of future scene dynamics. Recent unified world–action models further demonstrate strong planning performance and promising zero-shot transfer capability, highlighting the potential of pretrained video-generation priors for driving. However, generating future videos during inference introduces substantial computational overhead, and many recent driving WMs therefore favor single-front-view inference for efficiency, which limits its spatial coverage in safety-critical scenarios. To address this limitation, we propose SV-WAM, a surround-view world-action model (WAM) that preserves full six-camera observations while maintaining efficient inference. SV-WAM leverages future-video prediction as dense training supervision for action learning within a shared generative model, rather than as an inference-time output. At the core of this design is an action-centered causal mask that prevents action tokens from attending to future-video tokens during joint action–video denoising. Consequently, the video branch can be discarded at deployment, enabling efficient action-only planning. Furthermore, we introduce a differentiable drivable-area compliance regularizer that penalizes vehicle-footprint corners approaching or crossing drivable boundaries, improving planning safety and boundary awareness. Extensive experiments on the closed-loop NAVSIMv2 benchmark and the open-loop nuScenes benchmark demonstrate that SV-WAM achieves strong planning performance with low inference latency and competitive zero-shot transfer capability. **Code:** https://anonymous.4open.science/r/SV-WAM-F74E/

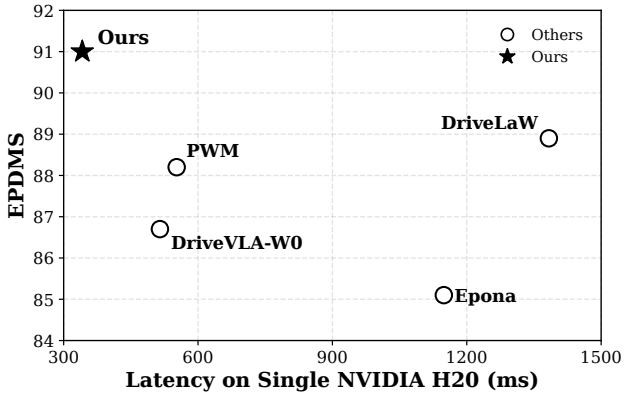

Figure 1: Efficiency–performance overview on a single NVIDIA H20 GPU (preprocessed-input-to-trajectory latency). SV-WAM reaches the best NAVSIMv2 EPDMS at the lowest latency despite using six-camera input.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

World models have recently emerged as a prominent approach for end-to-end autonomous driving (Zhang et al., 2025; Zhao et al., 2025; Xu et al., 2026; Liu et al., 2026; Li et al., 2026b; Zheng et al., 2024a; Hu et al., 2023a; Wang et al., 2024; Gao et al., 2024). Compared with BEV-based and VLA-based approaches that often rely on annotated datasets, world models can exploit large-scale driving videos as predictive supervision, reducing reliance on costly manual labeling. (Li et al., 2022; Hu et al., 2023b; Tian et al., 2024; Zhou et al., 2025; Xu et al., 2026; Zheng et al., 2025). By forecasting the evolution of the driving scene, they provide planners with future-aware representations that extend beyond sparse action imitation (Zhang et al., 2025; Zhao et al., 2025; Xu et al., 2026; Liu et al., 2026; Li et al., 2026b). This predictive supervision is particularly valuable for closed-loop planning, in which safe actions depend not only on the current observation but also on the constraints imposed by surrounding agents and road structure.

Building on this idea, recent driving world models have explored different ways of coupling world knowledge with action planning (Zhang et al., 2025; Zhao et al., 2025; Xia et al., 2026; Liu et al., 2026). Notably, DriveVA (Liu et al., 2026) further demonstrates the potential of large-scale pretrained video-generation models for driving, showing strong zero-shot transfer by leveraging motion and physical priors acquired during video pretraining. However, tighter unification of video and action generation also introduces a practical tension because visual generation remains part of the online inference process, increasing computational overhead. Meanwhile, several recent high-performing world-model-based planners operate on a single front-facing camera at inference (Zhang et al., 2025; Zhao et al., 2025; Xia et al., 2026; Liu et al., 2026), and extending such visual-generation pipelines to full six-view surround observations would further amplify this cost. This motivates us to ask whether future video can be decoupled from the inference path while still improving action learning through training-time predictive supervision.

To answer this question, we propose **SV-WAM**, an efficient surround-view world-action model that uses future video as training-time supervision rather than an inference-time output. As illustrated in Figure 2, SV-WAM retains future-video prediction during training while removing the video branch from the deployed action path. During training, SV-WAM jointly denoises future actions and six-view future-video latents within a shared diffusion-transformer backbone, allowing future visual prediction to shape the shared representations used for action generation. To prevent this training signal from becoming an inference-time dependency, we introduce an **action-centered causal mask** that prevents action tokens from attending to future-video tokens. Future-video prediction therefore contributes to action learning through the shared model parameters, while remaining outside the forward information path of the action branch. This design allows the entire video branch to be removed at deployment, enabling efficient two-step action-only inference while retaining full six-view spatial coverage. We further introduce a differentiable **drivable-area compliance regularizer**, which backpropagates road-boundary constraints through a differentiable LQR-based vehicle rollout to penalize vehicle footprints near or beyond drivable boundaries without additional inference cost.

Experiments show that SV-WAM achieves strong planning performance with low latency. On NAVSIMv2, SV-WAM reaches **91.0 EPDMS** and runs in **342 ms** on a single NVIDIA H20 GPU from preprocessed inputs to trajectory output, outperforming recent WM-based planners. On nuScenes, SV-WAM transfers **zero-shot** with **0.89 m** average L2 error and **0.16%** average collision rate.

Our contributions can be summarized as follows:

1. We propose SV-WAM, an efficient surround-view world-action model that decouples future-video supervision from action inference through an action-centered causal mask, enabling joint action–video training while retaining action-only inference with full six-view conditioning.

2. We introduce a differentiable drivable-area regularizer that propagates road-boundary constraints through an LQR-based vehicle rollout to the predicted actions, improving road compliance and safety without increasing inference cost.

3. Our method achieves strong planning performance on NAVSIMv2 and demonstrates competitive zero-shot transfer on nuScenes with low latency.

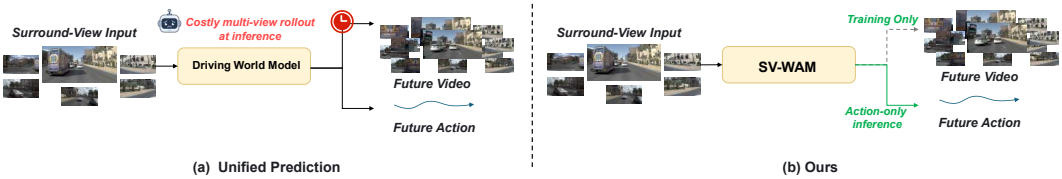

Figure 2: Unified world–action prediction retains future-video generation in the inference loop, which becomes costly when extended to multi-view surround observations. In contrast, SV-WAM uses future-video prediction only during training and performs action-only inference at deployment.

2 Related Work

End-to-End Autonomous Driving. End-to-end autonomous driving predicts future ego trajectories directly from raw sensor inputs (Hu et al., 2022). BEV-based methods such as UniAD (Hu et al., 2023b) and VAD (Jiang et al., 2023) integrate perception, prediction, and planning into a unified framework, while DiffusionDrive (Liao et al., 2025) improves multimodal trajectory generation with a truncated diffusion policy. Recent VLA-based planners further enhance driving with semantic reasoning; for example, ReCogDrive (Li et al., 2026c) connects scene-level understanding with continuous trajectory generation, and OpenDriveVLA (Zhou et al., 2026) grounds action prediction on 2D/3D visual representations and language commands (Tian et al., 2024; Zhou et al., 2025; Li et al., 2026a). Despite their strong performance, many existing structured planners rely on multiple task-specific modules to construct explicit intermediate representations, such as object detections or occupancy maps, before trajectory generation. These modules increase architectural complexity. Moreover, they lack explicit modeling of future scene evolution.

Driving World Models for Planning. Driving world models have recently emerged as a promising paradigm for end-to-end planning (Chen et al., 2025). Epona (Zhang et al., 2025) learns a shared historical world representation that supports modular visual prediction and trajectory planning, while PWM (Zhao et al., 2025) transfers predictive world knowledge to planning through collaborative state–action modeling. DriveLaW (Xia et al., 2026) further tightens this coupling by injecting latent representations from its video generator into the planner. Recent latent world models seek more compact world representations. World4Drive (Zheng et al., 2025) leverages spatial–semantic priors from vision foundation models for latent future prediction and trajectory selection, while WorldRFT (Yang et al., 2026) incorporates vision–geometry priors and reinforcement fine-tuning for planning-oriented latent representation learning. Latent-WAM (Wang et al., 2026) compresses multi-view observations into compact scene tokens and autoregressively predicts future world states in latent space, improving efficiency without explicitly visualizing the predicted future and therefore providing less direct interpretability of world evolution. In a complementary direction, DriveVA (Liu et al., 2026) jointly generates future video latents and actions with a pretrained video-generation backbone and demonstrates zero-shot transfer to unseen domains without target-domain fine-tuning, highlighting the transferable potential of large-scale video priors. However, its unified video–action generation retains future-video generation in the online inference process and therefore introduces additional visual-generation computation. More broadly, several recent WM-based planners operate on a single front-facing camera at inference (Zhang et al., 2025; Zhao et al., 2025; Xia et al., 2026; Liu et al., 2026), and extending such generative pipelines to full surround-view observations would further amplify the computational burden. Our work instead retains explicit future-video prediction as training-time supervision while enabling six-view action-only inference at deployment.

3 Methodology

3.1 Overview

Figure 3 presents the overall framework of SV-WAM, which consists of four components. First, input tokenization converts historical surround-view observations, ego states, and action sequences into transformer-compatible tokens. Second, the surround-view world–action model jointly denoises

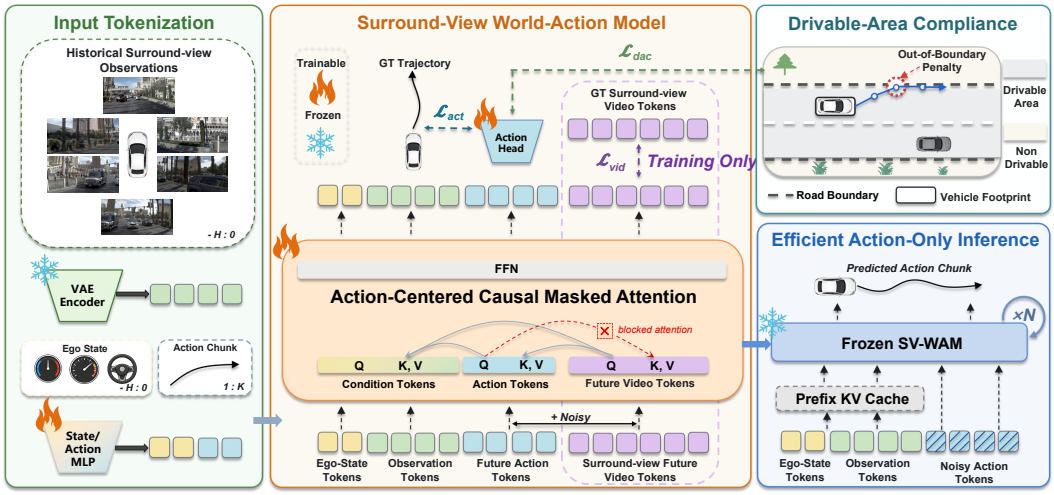

Figure 3: Overview of SV-WAM. During training, SV-WAM jointly denoises future actions and future-video latents conditioned on six-view historical observations. The action-centered causal mask blocks action-token attention to future-video tokens, while the drivable-area compliance regularizer promotes road-compliant planning. At inference, the future-video branch is removed, enabling efficient action-only planning with full six-view context.

future actions and future-video latents using an action-centered causal mask. Third, the drivable-area compliance regularizer penalizes vehicle footprints that approach or cross non-drivable boundaries during training. Finally, at inference, SV-WAM removes the future-video branch and reuses the cached condition prefix across denoising steps, enabling efficient action-only planning.

3.2 Input Tokenization

This subsection describes how SV-WAM converts surround-view images, ego states, actions, and future-video latents into transformer inputs.

Surround-View Video Encoding. Let $N_{\text{cam}} = 6$ denote the number of cameras. At each timestamp i, the camera images are arranged in a fixed order and concatenated along the width dimension:

$$X_i = \text{Concat}_w(I_i^1, I_i^2, \ldots, I_i^{N_{\text{cam}}}), \tag{1}$$

where I_i^v denotes the image from the v-th camera and X_i is the resulting surround-view frame.

We encode the full training video sequence $X_{-H+1:F}$ using the frozen 3D causal VAE encoder of Wan2.2-TI2V-5B (Wan Team, 2025). The resulting latents are split along the temporal dimension:

$$[Z^{\text{ref}}, Z^{\text{fut}}] = E_{\text{vae}}(X_{-H+1:F}). \tag{2}$$

Here, H and F denote the numbers of historical and future frames, respectively; Z^{ref} represents the clean historical prefix used for visual conditioning, while Z^{fut} contains the future-video latents used only during training. Both are tokenized using a shared video tokenizer:

$$[\mathcal{V}^{\text{ref}}, \mathcal{V}^{\text{fut}}] = \text{Tok}_v([Z^{\text{ref}}, Z^{\text{fut}}]). \tag{3}$$

In implementation, Tok_v partitions the latents into spatiotemporal patches and embeds them into a flattened transformer-token sequence.

State and Action Encoding. Following the NAVSIM input convention (Dauner et al., 2024), we denote the ego-state history as $S = \{s_i\}_{i=-H+1}^{0}$, where $s_i \in \mathbb{R}^8$. An MLP maps the state history into state tokens:

$$\mathcal{S} = \phi_s(S). \tag{4}$$

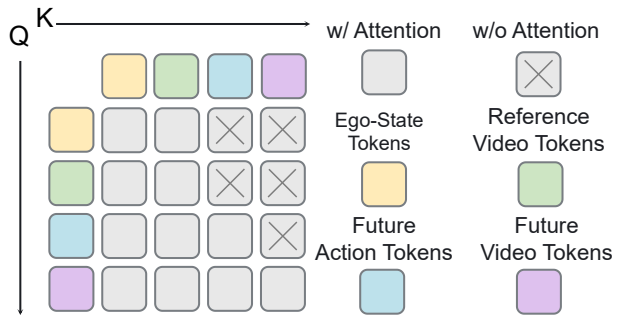

Figure 4: Action-centered causal mask. Action tokens attend to ego-state and reference-video tokens but are blocked from attending to future-video tokens, whereas future-video tokens can attend to action tokens. This asymmetric dependency supports future-video co-training without making action prediction dependent on future-video tokens at inference.

The future action sequence is written as $A = \{a_k = (\Delta x_k, \Delta y_k, \Delta \psi_k)\}_{k=1}^{K}$, where each action represents a stepwise ego-relative motion increment. The sequence is projected into action tokens using an MLP tokenizer:

$$\mathcal{A} = \phi_a(A). \tag{5}$$

3.3 Surround-View World-Action Model

This subsection introduces the action-centered causal mask mechanism for efficient inference and the joint flow-matching training used in SV-WAM.

Action-Centered Causal Mask. Recent video-action driving models jointly generate future visual forecasts and action sequences in a shared generative process (Liu et al., 2026). Inspired by embodied world-action modeling (Ye et al., 2026a), we design an action-centered causal mask for efficient planning. The mask blocks the direct attention dependency of action tokens on future-video tokens, which are expensive to denoise and are not instantiated in the action-only inference path.

The mask is applied over the packed token sequence during self-attention. During flow training, we sample a shared base time $u \sim \mathcal{U}(0, 1)$ and apply a shifted flow schedule to obtain t for both action and future-video denoising. Given Gaussian noises ϵ_a and ϵ_z, we construct

$$A_t = (1-t)A_0 + t\epsilon_a, \qquad Z_t^{\text{fut}} = (1-t)Z_0^{\text{fut}} + t\epsilon_z, \tag{6}$$

where A_0 and Z_0^{fut} denote the clean future action sequence and clean future video latents. Using the tokenizers defined in the previous subsection, A_t and Z_t^{fut} are converted into noisy action tokens $\mathcal{A}_t$ and noisy future-video tokens $\mathcal{V}_t^{\text{fut}}$. The resulting self-attention sequence is

$$\mathcal{Y}_t = [\mathcal{S}, \mathcal{V}^{\text{ref}}, \mathcal{A}_t, \mathcal{V}_t^{\text{fut}}]. \tag{7}$$

As shown in Figure 4, state tokens and reference video tokens form the clean condition prefix and attend only to themselves. Action tokens attend to the condition prefix and action tokens, while future video tokens attend to the condition prefix, action tokens, and future video tokens.

Let $\Omega = \{Z^{\text{ref}}, S\}$ denote the condition consisting of reference video latents and ego-state history. The mask induces an action-centered factorization:

$$p_\theta(A_0, Z_0^{\text{fut}} \mid \Omega) = p_\theta(A_0 \mid \Omega) p_\theta(Z_0^{\text{fut}} \mid \Omega, A_0). \tag{8}$$

Thus, future-video denoising provides future-dynamics supervision through the shared model parameters, while action tokens do not attend to future-video tokens in the forward pass. This property enables action-only inference.

SV-WAM Training. We train SV-WAM with flow matching (Lipman et al., 2023; Liu et al., 2023) on both future actions and future video latents. Given the noisy variables defined above, the target

velocity fields for the two branches are obtained by differentiating the interpolations A_t and Z_t^{fut} with respect to t:

$$v_a^\star = \epsilon_a - A_0, \qquad v_z^\star = \epsilon_z - Z_0^{\text{fut}}. \tag{9}$$

The masked transformer takes the self-attention sequence $\mathcal{Y}_t$ and the flow time t as input:

$$(\hat{v}_a, \hat{v}_z) = f_\theta(\mathcal{Y}_t, t), \tag{10}$$

where f_θ denotes the masked velocity predictor with learnable parameters θ, and $\hat{v}_a$ and $\hat{v}_z$ are the predicted action- and video-velocity fields. The action velocity is decoded from the action token positions using an MLP action head, while the video velocity is projected from the future video token positions and unpatchified back to the VAE latent space.

The flow matching loss is defined as

$$\begin{aligned}\mathcal{L}_{\text{fm}} = \lambda_{\text{act}} \left\| \hat{v}_a - v_a^\star \right\|_2^2 \\ + \lambda_{\text{vid}} \left\| \hat{v}_z - v_z^\star \right\|_2^2 .\end{aligned} \tag{11}$$

The action branch directly optimizes trajectory generation, while the future video branch encourages the model to capture future scene dynamics. Because the causal mask prevents action-token queries from attending to future-video keys and values, future-video supervision introduces no inference-time dependency into the action path.

3.4 Drivable Area Compliance Regularization

Flow matching learns future ego actions from trajectory supervision, but it does not explicitly constrain the simulated vehicle footprint with respect to drivable boundaries. Motivated by the drivable area compliance metric in NAVSIM (Dauner et al., 2024), we propose a drivable area compliance regularizer that softly penalizes predicted trajectories whose footprint approaches or leaves the drivable region. Given the predicted action velocity $\hat{v}_a$, we first recover the clean action prediction from the noisy action input:

$$\hat{A}_0 = A_t - t\hat{v}_a. \tag{12}$$

Given a binary drivable area map b, we construct a signed distance field Φ in the ego-centric coordinate system:

$$\Phi(u) = D_{\text{non}}(u) - D_{\text{drive}}(u), \tag{13}$$

where $D_{\text{non}}(u)$ and $D_{\text{drive}}(u)$ denote the Euclidean distances from location u to the nearest non-drivable and drivable cells, respectively. Under this convention, $\Phi(u) > 0$ indicates that u lies inside the drivable area, while $\Phi(u) < 0$ indicates an off-road location.

After denormalization, we follow the controller structure of the official NAVSIM evaluation pipeline (Dauner et al., 2024) and roll out $\hat{A}_0$ using a differentiable PyTorch reimplementation of its LQR-based tracker and kinematic vehicle dynamics, obtaining simulated ego poses $Q = \{q_i\}_{i=1}^{N_q}$, where $q_i = (x_i, y_i, \psi_i)$. This differentiable rollout allows the compliance loss to backpropagate through the simulated ego trajectory to the predicted actions; implementation details are provided in Appendix A. For each simulated pose, we compute the four corner points $c_{i,j}$ of the vehicle footprint and sample their signed distances from the signed distance field:

$$d_{i,j} = \Phi(c_{i,j}), \qquad j = 1, \ldots, 4. \tag{14}$$

Since the corner points are continuous coordinates, $\Phi(c_{i,j})$ is evaluated by bilinear interpolation on the signed distance field grid.

We apply a smooth margin penalty to each sampled distance:

$$\ell(d_{i,j}) = \beta \log\left(1 + \exp\left(\frac{m - d_{i,j}}{\beta}\right)\right), \tag{15}$$

where m is the safety margin and β controls the smoothness. The final drivable area compliance loss is computed by log-mean-exp aggregation over all footprint penalties:

$$\mathcal{L}_{\text{dac}} = \rho \log\left(\frac{1}{4N_q} \sum_{i=1}^{N_q} \sum_{j=1}^{4} \exp\left(\frac{\ell(d_{i,j})}{\rho}\right)\right), \tag{16}$$

where ρ is the aggregation temperature. This aggregation acts as a smooth maximum and assigns larger weights to high-risk footprint violations than simple averaging. The overall training objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{fm}} + \lambda_{\mathrm{dac}}\mathcal{L}_{\mathrm{dac}}. \tag{17}$$

3.5 Efficient Action-Only Inference

At inference time, we keep the same reference video and state conditioning path but omit noisy future video tokens from the transformer sequence. The inference sequence becomes

$$\mathcal{Y}_t^{\mathrm{act}} = [\mathcal{S}, \mathcal{V}^{\mathrm{ref}}, \mathcal{A}_t]. \tag{18}$$

The same trained model is evaluated under the corresponding action-only mask:

$$\hat{v}_a = f_\theta(\mathcal{Y}_t^{\mathrm{act}}, t), \tag{19}$$

where f_θ denotes the trained action-velocity predictor. Starting from Gaussian action noise, the action scheduler iteratively updates the action sequence using the predicted velocity field. Since noisy future video tokens are not instantiated and the video output branch is skipped, SV-WAM avoids future video denoising and decoding during deployment while retaining the representation benefit of training-time video co-denoising. To further improve efficiency, we cache and reuse the conditioning prefix across denoising steps.

4 Experiments

4.1 Experimental Setup

Benchmarks and Metrics. We evaluate closed-loop planning on NAVSIMv2 (Dauner et al., 2024; Cao et al., 2025), which reports the Extended Predictive Driver Model Score (EPDMS). EPDMS aggregates no at-fault collision (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), where the compliance terms act as multiplicative safety penalties. To test cross-dataset generalization, we further evaluate on nuScenes (Caesar et al., 2020) without target-domain fine-tuning. For nuScenes, we report L2 trajectory error and collision rate at 1s, 2s, 3s, and their averages. NAVSIMv1 and NAVSIMv2 navhard results are reported in the Appendix.

Implementation Details. We use four historical frames from six surround-view cameras at 2 Hz, encode them with the frozen Wan2.2-TI2V-5B causal VAE, and use Wan2.2-5B as the DiT backbone (Peebles & Xie, 2023). The model predicts 12 future action tokens and is trained with 12-frame six-view future-video latent supervision. All main results use two-step action-only inference; the future-video branch is removed at deployment. Full training, optimization, and preprocessing details are provided in Appendix A.

4.2 Main Results

NAVSIMv2. As demonstrated in Table 1, SV-WAM achieves the best performance under the closed-loop NAVSIMv2 metric suite, outperforming recent end-to-end driving planners on the navtest split. We follow the evaluation setting of EponaV2 (Xu et al., 2026). The results for DriveVLA-W0 and DriveLaW are reproduced using their official code and checkpoints, while the remaining baseline results are taken from the corresponding papers or official leaderboard. SV-WAM achieves the best EPDMS of 91.0. These gains should not be attributed to surround-view input alone. The six-view observations provide complementary side and rear context, while future-video co-training and the drivable-area regularizer further improve future-aware representation learning and road-boundary compliance, respectively, as analyzed in Table 3.

nuScenes Zero-Shot Transfer. As shown in Table 2, we further evaluate zero-shot transfer to nuScenes. The upper block lists representative methods fine-tuned on nuScenes, whereas the lower block compares world-model planners without nuScenes fine-tuning. We directly evaluate the officially released checkpoints of DriveVA, DriveVLA-W0, and PWM without further fine-tuning on

Table 1: Detailed closed-loop planning comparison on the NAVSIMv2 navtest split with human penalty enabled. Metrics are reported as percentages. C denotes camera input and L denotes LiDAR input; for example, C×1 is front-camera-only input, C×3 denotes three front-facing camera input, and C×6 denotes six-view surround-camera input. Bold and underlined values denote the best and second-best results, respectively. † denotes WM-based methods.

Method	Input	NC ↑	DAC ↑	DDC ↑	TLC ↑	EP ↑	TTC ↑	LK ↑	HC ↑	EC ↑	EPDMS ↑
ReCogDrive (2026c)	C×1	98.3	95.2	99.5	99.8	87.1	97.5	96.6	98.3	86.5	83.6
Epona † (2025)	C×1	97.1	95.7	99.3	99.7	**88.6**	96.3	97.0	98.0	67.8	85.1
DriveVLA-W0 † (2026b)	C×1	98.3	95.0	99.4	**99.9**	86.6	97.8	97.7	**98.4**	82.4	86.7
PWM † (2025)	C×1	98.8	95.9	99.4	**99.9**	86.4	98.4	97.6	98.3	85.3	88.2
DriveLaW † (2026)	C×1	98.8	96.9	**99.6**	99.8	87.4	98.3	97.7	**98.4**	80.2	88.9
EponaV2 † (2026)	C×1	98.5	97.4	99.5	**99.9**	87.9	98.1	97.7	98.2	77.4	88.9
Ours †	C×1	98.4	98.4	99.1	99.8	85.5	97.8	95.9	**98.4**	86.5	89.3
AutoDrive-P3 (2026b)	C×3	**99.1**	97.4	99.2	99.8	88.0	**98.7**	96.3	98.3	85.5	89.9
WoTE † (2025)	C×3&L	98.5	96.8	98.8	99.8	86.1	97.9	95.5	98.3	82.9	87.7
DiffusionDrive (2025)	C×3&L	98.2	96.2	99.5	99.8	87.4	97.3	96.9	**98.4**	87.7	88.2
Ours †	C×3	98.7	**99.1**	99.4	99.8	86.1	98.1	96.5	**98.4**	86.6	90.5
Ours †	C×6	98.6	98.8	**99.6**	**99.9**	86.8	98.3	**97.8**	**98.4**	**88.4**	**91.0**

Table 2: End-to-end motion planning performance on nuScenes. "nuS. FT" marks whether the method is fine-tuned on nuScenes. DriveVA, DriveVLA-W0, and PWM are evaluated zero-shot on nuScenes using their officially released checkpoints, without additional fine-tuning. Bold and underlined values denote the best and second-best results, respectively. † denotes WM-based methods.

Method	nuS. FT	Ref	L2 (m) ↓				Collision Rate (%) ↓			
			1s	2s	3s	Avg.	1s	2s	3s	Avg.
UniAD (2023b)	✓	CVPR'23	0.48	0.96	1.65	1.03	0.05	0.17	0.71	0.31
OccWorld † (2024a)	✓	ECCV'24	0.52	1.27	2.41	1.40	0.12	0.40	2.08	0.87
VAD-Tiny (2023)	✓	ICCV'23	0.60	1.23	2.06	1.30	0.31	0.53	1.33	0.72
VAD-Base (2023)	✓	ICCV'23	0.54	1.15	1.98	1.22	0.04	0.39	1.17	0.53
GenAD (2024b)	✓	ECCV'24	0.36	0.83	1.55	0.91	0.06	0.23	1.00	0.43
Doe-1 (2024c)	✓	arXiv'24	0.50	1.18	2.11	1.26	0.04	0.37	1.19	0.53
Epona † (2025)	✓	ICCV'25	0.61	1.17	1.98	1.25	0.01	0.22	0.85	0.36
DriveLaW † (2026)	✓	CVPR'26	0.44	1.10	1.91	1.15	0.15	0.10	0.48	0.24
DriveVLA-W0 † (2026b)	×	ICLR'26	0.43	1.26	2.60	1.43	0.22	0.66	1.42	0.77
PWM † (2025)	×	NeurIPS'25	2.06	3.91	6.00	3.99	0.12	0.15	0.86	0.36
DriveVA † (2026)	×	ECCV'26	0.33	**0.76**	**1.43**	**0.84**	**0.00**	**0.07**	**0.12**	**0.06**
Ours †	×	–	**0.31**	0.80	1.57	0.89	0.07	0.13	0.27	0.16

NAVSIM or nuScenes. Under this setting, SV-WAM achieves competitive zero-shot performance, with 0.89 m average L2 error and 0.16% average collision rate, outperforming DriveVLA-W0 and PWM and remaining competitive with DriveVA while also surpassing several in-domain fine-tuned planners. This result provides encouraging evidence of cross-dataset transfer without target-domain supervision. Additional zero-shot qualitative results on in-house driving data are provided in the Appendix.

4.3 Ablation Studies

Training Components. Table 3 ablates the three ingredients of the final model. All variants share the same C×6 architecture and evaluation setup, differing only in the enabled training components. Future-video co-training improves EPDMS from 83.1 to 87.7, showing that grounding actions in future visual dynamics helps even though the video branch is removed at deployment. Adding the DAC regularization raises EPDMS to 90.1 and DAC from 95.6 to 98.6, indicating fewer drivable-area violations. Final fine-tuning yields 91.0 EPDMS.

Table 3: Component ablation on NAVSIMv2 navtest. *Video Sup.* denotes future-video supervision co-training, *DAC-reg* denotes the drivable-area regularizer, and *FT* denotes final fine-tuning.

Video Sup.	DAC-reg	FT	DAC ↑	EPDMS ↑
×	×	×	93.4	83.1
✓	×	×	95.6	87.7
✓	✓	×	98.6	90.1
✓	✓	✓	**98.8**	**91.0**

Table 4: Joint ablation of the action-centered causal mask and inference steps on NAVSIMv2 navtest with C×6 input. Metrics are reported as percentages. All total latencies are marked ∼ to denote approximate wall-clock values. Two-step latencies are measured on a single NVIDIA H20 GPU, while five- and ten-step latencies are estimated using approximately 46 ms per additional step without the mask and 76 ms per additional step with the mask.

Mask	Steps	NC ↑	DAC ↑	DDC ↑	TLC ↑	EP ↑	TTC ↑	LK ↑	HC ↑	EC ↑	EPDMS ↑	Total (ms) ↓
×	2	97.4	96.6	99.0	99.7	**87.8**	96.8	97.0	98.3	86.0	87.3	∼848.0
×	5	97.6	96.8	99.2	99.7	87.7	96.9	97.5	98.3	87.3	87.9	∼986.0
×	10	97.7	96.9	99.3	99.7	87.6	97.0	97.6	98.3	87.3	88.2	∼1216.0
✓	2	**98.6**	**98.8**	99.6	**99.9**	86.8	98.3	97.8	**98.4**	**88.4**	**91.0**	∼**341.6**
✓	5	**98.6**	98.6	99.6	**99.9**	86.8	**98.4**	**97.9**	**98.4**	**88.4**	90.9	∼569.6
✓	10	**98.6**	98.5	**99.7**	99.8	86.9	98.2	**97.9**	**98.4**	**88.4**	90.8	∼949.6

Action-Centered Causal Mask and Inference Steps. Table 4 jointly evaluates the causal mask and the number of denoising steps under the same C×6 architecture and training setting. Without the mask, future-video denoising remains in the inference path, and increasing the number of steps from 2 to 10 only modestly improves EPDMS from 87.3 to 88.2 while substantially increasing latency. With the mask, the future-video branch is removed at deployment: two action-only denoising steps achieve the best EPDMS of 91.0 at 341.6 ms, whereas additional steps provide no performance gain and only increase computation. These results support our central design choice that future-video prediction is useful as training-time supervision but need not remain in the deployed action path.

Surround-View Coverage. Under the same architecture and training schedule, Table 1 shows that increasing camera coverage consistently improves EPDMS, with C×6 also improving DDC, LK, and EC. This supports the value of side and rear context for planning.

4.4 Efficiency Analysis

Figure 1 summarizes the efficiency–performance trade-off against recent world-model-based planners. The mask/step latency trend is summarized in Table 4, while Table 8 in Appendix E provides the detailed two-step online trajectory-output latency breakdown on a single H20 GPU. The timing starts from preprocessed inputs to final trajectory. For the compared methods, we use official code, released checkpoints, and default inference configurations. All methods use batch size 1 and bf16 where supported; 30 samples remain after 5 warm-up samples. Despite using six cameras, SV-WAM achieves the lowest latency at 341.6 ± 2.1 ms. Removing the action-centered causal mask requires the future-video branch to remain active at inference, increasing the trajectory-output latency to 848.0 ± 1.8 ms, with optional video decoding excluded in both cases. The full measurement protocol and optional video-rendering costs are provided in the Appendix.

5 Conclusion

We present SV-WAM, an efficient surround-view world-action model for end-to-end autonomous driving. The key insight is that six-camera context should be retained for reliable planning, whereas future video prediction can be used as training-time supervision and removed from inference. SV-

WAM follows this insight with an action-centered causal mask that blocks the direct attention path from future-video tokens to action tokens while retaining future-video supervision through shared model parameters. The video branch can therefore be removed at deployment, yielding efficient action-only planning with full surround-view coverage. We further introduce a differentiable drivable area compliance regularizer to improve road compliance. Experiments on NAVSIMv2 and nuScenes validate the effectiveness and efficiency of the proposed design.

AI Use Statement

Generative AI tools were used for language polishing and drafting assistance, literature discovery, and discussion of research presentation, methodological framing, and experimental analysis. The authors independently reviewed and verified all AI-assisted material, cited literature, technical claims, implementation choices, experiments, numerical results, and final text, and take responsibility for the complete content of this work.

Reproducibility Statement

Detailed model architecture, training settings, data processing, evaluation protocols, and latency measurement details are provided in Section 4.1 and the Appendix. Additional ablations and qualitative results are also included to support the reported findings. The anonymized code is available at https://anonymous.4open.science/r/SV-WAM-F74E/.

References

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11621–11631, 2020.

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 *Conference on Robot Learning*, 2025.

Yuntao Chen, Yuqi Wang, and Zhaoxiang Zhang. DrivingGPT: Unifying driving world modeling and planning with multi-modal autoregressive transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

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

Lan Feng, Yang Gao, Eloi Zablocki, Quanyi Li, Wuyang Li, Sichao Liu, Matthieu Cord, and Alexandre Alahi. RAP: 3d rasterization augmented end-to-end planning. In *The Fourteenth International Conference on Learning Representations*, 2026.

Shenyuan Gao, Jiazhi Yang, Li Chen, Kashyap Chitta, Yihang Qiu, Andreas Geiger, Jun Zhang, and Hongyang Li. Vista: A generalizable driving world model with high fidelity and versatile controllability. In *Advances in Neural Information Processing Systems*, 2024.

Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023a.

Shengchao Hu, Li Chen, Penghao Wu, Hongyang Li, Junchi Yan, and Dacheng Tao. ST-P3: End-to-end vision-based autonomous driving via spatial-temporal feature learning. In *European Conference on Computer Vision*, 2022.

Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, et al. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023b.

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*, 2023.

Jason Kong, Mark Pfeiffer, Georg Schildbach, and Francesco Borrelli. Kinematic and dynamic vehicle models for autonomous driving control design. In *IEEE Intelligent Vehicles Symposium (IV)*, 2015.

Pengxiang Li, Yinan Zheng, Yue Wang, Huimin Wang, Hang Zhao, Jingjing Liu, Xianyuan Zhan, Kun Zhan, and Xianpeng Lang. Discrete diffusion for reflective vision-language-action models in autonomous driving. In *The Fourteenth International Conference on Learning Representations*, 2026a.

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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 27137–27146, 2025.

Yingyan Li, Shuyao Shang, Weisong Liu, Bing Zhan, Haochen Wang, Yuqi Wang, Yuntao Chen, Xiaoman Wang, Yasong An, Chufeng Tang, Lu Hou, Lue Fan, and Zhaoxiang Zhang. DriveVLA-W0: World models amplify data scaling law in autonomous driving. In *The Fourteenth International Conference on Learning Representations*, 2026b.

Yongkang Li, Kaixin Xiong, Xiangyu Guo, Fang Li, Sixu Yan, Gangwei Xu, Lijun Zhou, Long Chen, Haiyang Sun, Bing Wang, Kun Ma, Guang Chen, Hangjun Ye, Wenyu Liu, and Xinggang Wang. ReCogDrive: A reinforced cognitive framework for end-to-end autonomous driving. In *The Fourteenth International Conference on Learning Representations*, 2026c.

Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *European Conference on Computer Vision*, 2022.

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

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Mengmeng Liu, Diankun Zhang, Jiuming Liu, Jianfeng Cui, Hongwei Xie, Guang Chen, Hangjun Ye, Michael Ying Yang, Francesco Nex, and Hao Cheng. DriveVA: Video action models are zero-shot drivers. In *European Conference on Computer Vision*, pages 315–335, 2026.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023.

Long Nguyen, Micha Fauth, Bernhard Jaeger, Daniel Dauner, Maximilian Igl, Andreas Geiger, and Kashyap Chitta. LEAD: Minimizing learner-expert asymmetry in end-to-end driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision*, 2023.

Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7077–7087, 2021.

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

Wan Team. Wan2.2. https://github.com/Wan-Video/Wan2.2, 2025. Official Wan2.2 model release; Wan2.2-TI2V-5B model and VAE.

Linbo Wang, Yupeng Zheng, Qiang Chen, Shiwei Li, Yichen Zhang, Zebin Xing, Qichao Zhang, Xiang Li, Deheng Qian, Pengxuan Yang, Yihang Dong, Ce Hao, Xiaoqing Ye, Junyu Han, Yifeng Pan, and Dongbin Zhao. Latent-wam: Latent world action modeling for end-to-end autonomous driving. *arXiv preprint arXiv:2603.24581*, 2026.

Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. Drivedreamer: Towards real-world-driven world models for autonomous driving. In *European Conference on Computer Vision*, 2024.

Tianze Xia, Yongkang Li, Lijun Zhou, Jingfeng Yao, Kaixin Xiong, Haiyang Sun, Bing Wang, Kun Ma, Guang Chen, Hangjun Ye, Wenyu Liu, and Xinggang Wang. DriveLaW: Unifying planning and video generation in a latent driving world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 39701–39712, June 2026.

Jiawei Xu, Zhizhou Zhong, Zhijian Shu, Mingkai Jia, Mingxiao Li, Jia-Wang Bian, Qian Zhang, Kaicheng Zhang, Jin Xie, Jian Yang, and Wei Yin. EponaV2: Driving world model with comprehensive future reasoning. *arXiv preprint arXiv:2605.14696*, 2026.

Pengxuan Yang, Ben Lu, Zhongpu Xia, Chao Han, Yinfeng Gao, Teng Zhang, Kun Zhan, Xianpeng Lang, Yupeng Zheng, and Qichao Zhang. WorldRFT: Latent world model planning with reinforcement fine-tuning for autonomous driving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 11649–11657, 2026. doi: 10.1609/aaai.v40i14.38149.

Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jie Li, Jindi Lv, Jingyu Liu, Min Cao, Peng Li, Qiuping Deng, Wenjun Mei, Xiaofeng Wang, Xinze Chen, Xinyu Zhou, Yang Wang, Yifan Chang, Yifan Li, Yukun Zhou, Yun Ye, Zhichao Liu, and Zheng Zhu. GigaWorld-Policy: An efficient action-centered world–action model. *arXiv preprint arXiv:2603.17240*, 2026a.

Yuqi Ye, Zijian Zhang, Junhong Lin, Shangkun Sun, Changhao Peng, and Wei Gao. AutoDrive-P3: Unified chain of perception-prediction-planning thought via reinforcement fine-tuning. In *The Fourteenth International Conference on Learning Representations*, 2026b.

Kaiwen Zhang, Zhenyu Tang, Xiaotao Hu, Xingang Pan, Xiaoyang Guo, Yuan Liu, Jingwei Huang, Li Yuan, Qian Zhang, Xiao-Xiao Long, Xun Cao, and Wei Yin. Epona: Autoregressive diffusion world model for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

Zhida Zhao, Talas Fu, Yifan Wang, Lijun Wang, and Huchuan Lu. From forecasting to planning: Policy world model for collaborative state-action prediction. In *Advances in Neural Information Processing Systems*, 2025.

Wenzhao Zheng, Weiliang Chen, Yuanhui Huang, Borui Zhang, Yueqi Duan, and Jiwen Lu. OccWorld: Learning a 3d occupancy world model for autonomous driving. In *European Conference on Computer Vision*, pages 55–72, 2024a. doi: 10.1007/978-3-031-72624-8_4.

Wenzhao Zheng, Ruiqi Song, Xianda Guo, Chenming Zhang, and Long Chen. GenAD: Generative end-to-end autonomous driving. In *European Conference on Computer Vision*, 2024b. doi: 10.1007/978-3-031-73650-6_6.

Wenzhao Zheng, Zetian Xia, Yuanhui Huang, Sicheng Zuo, Jie Zhou, and Jiwen Lu. Doe-1: Closed-loop autonomous driving with large world model. *arXiv preprint arXiv:2412.09627*, 2024c.

Yupeng Zheng, Pengxuan Yang, Zebin Xing, Qichao Zhang, Yuhang Zheng, Yinfeng Gao, Pengfei Li, Teng Zhang, Zhongpu Xia, Peng Jia, Xianpeng Lang, and Dongbin Zhao. World4Drive: End-to-end autonomous driving via intention-aware physical latent world model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 28632–28642, 2025.

Xingcheng Zhou, Xuyuan Han, Feng Yang, Yunpu Ma, Volker Tresp, and Alois Knoll. OpenDriveVLA: Towards end-to-end autonomous driving with large vision language action model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 13782–13790, 2026. doi: 10.1609/aaai.v40i16.38386.

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 *Advances in Neural Information Processing Systems*, 2025.

A Implementation Details

Architecture. SV-WAM builds on the Wan2.2-5B diffusion-transformer (DiT) video backbone and the 3D causal VAE encoder from Wan2.2-TI2V-5B. The four-frame six-view history is encoded by the VAE after the six views at each timestamp are concatenated along the width dimension into a single mosaic, and the resulting latents are converted into video tokens by a 3D convolutional patch embedding followed by flattening. Ego states and actions are projected into the transformer hidden space by lightweight MLP tokenizers, and a linear head maps the action token outputs back to the flow-velocity space. The transformer keeps the Wan2.2-5B DiT configuration unchanged—30 blocks and ≈ 5B parameters, with the original hidden width and head count—and the only additions are the small action and ego-state MLP tokenizers, whose parameter count is negligible.

Inputs and Outputs. Each sample uses $H = 4$ history frames from the six cameras (front-left, front, front-right, rear-right, rear, rear-left), each resized to 448×224. The ego state at each history step is

$$s_t = [c_t^1, c_t^2, c_t^3, c_t^4, v_t^x, v_t^y, a_t^x, a_t^y] \in \mathbb{R}^8, \tag{20}$$

where c_t is the four-dimensional one-hot driving command. The command entries remain in their native $\{0, 1\}$ space. We min–max normalize the continuous components to $[-1, 1]$ using statistics computed on the official NAVSIM training split. In the order (v^x, v^y, a^x, a^y), the extrema are

$$\begin{aligned} s_{\min} &= (-2.0, -0.5, -5.1, -5.4), \\ s_{\max} &= (20.4, 0.5, 3.6, 5.6). \end{aligned} \tag{21}$$

Each continuous component is transformed as $\tilde{s} = 2(s - s_{\min})/(s_{\max} - s_{\min}) - 1$ and clipped to $[-1, 1]$.

Unlike the ego state, the action targets are not normalized with dataset statistics. Inspired by the coordinate-wise bounded normalization of Epona (Zhang et al., 2025), we use fixed ranges adapted to our stepwise $0.5\,\mathrm{s}$ ego-relative action representation. Each action is $a_k = (\Delta x_k, \Delta y_k, \Delta\psi_k)$, with nominal bounds

$$a_{\min} = (0, -0.5, -8^\circ), \qquad a_{\max} = (8, 0.5, 8^\circ), \tag{22}$$

where the translational components are in meters. We apply $\tilde{a} = 2(a - a_{\min})/(a_{\max} - a_{\min}) - 1$, without clipping values outside the nominal range. The model predicts $K = 12$ future action tokens, which are denormalized and accumulated in $\mathrm{SE}(2)$; the first eight tokens form the standard $4\,\mathrm{s}$ NAVSIM trajectory. During training, the model additionally denoises 12 future six-view video latents.

Training. We train on the official NAVSIM trainval split for 10k iterations with AdamW (batch size 80, learning rate 10^{-4}, weight decay 10^{-2}) on 16 H800 GPUs, then fine-tune for 1k iterations with an effective batch size of 640 (gradient accumulation of 8 over a per-step batch of 80) and a learning rate decayed from 10^{-5} to 10^{-6}. Each stage takes roughly one day ($\approx$24h) on the 16 H800 GPUs. We use rectified-flow training with Euler sampling; the full ≈ 5B model is trainable. The flow matching loss uses $\lambda_{\mathrm{act}} : \lambda_{\mathrm{vid}} = 1 : 1$. The implementation samples a uniform base time u and applies the shifted flow schedule

$$t = \frac{\alpha u}{1 + (\alpha - 1)u}, \qquad u \sim \mathcal{U}(0, 1), \tag{23}$$

where the flow-shift coefficient is set to $\alpha = 5.0$ for both action and future-video flow matching. The drivable area regularizer uses $\lambda_{\mathrm{dac}} = 0.01$, safety margin $m = 0.2$, softplus temperature $\beta = 0.2$, and log-mean-exp aggregation temperature $\rho = 0.1$. The signed distance field is constructed on a local grid $x \in [-20, 80]\,\mathrm{m}$, $y \in [-40, 40]\,\mathrm{m}$ at $0.5\,\mathrm{m}$ resolution.

Drivable Area Regularization Details. The predicted $2\,\mathrm{Hz}$ action sequence specifies a geometric reference rather than the trajectory attained by a tracking controller. We therefore roll it out with a linear quadratic regulator (LQR) and a kinematic bicycle model (Kong et al., 2015), following NAVSIM's official tracker (Dauner et al., 2024). LQR is a finite-horizon feedback controller that balances reference-tracking accuracy against control effort. In its standard form, it selects a control sequence according to

$$u^\star_{0:H-1} = \arg\min_{u_{0:H-1}} \sum_{h=0}^{H-1} \left(e_h^\top Q e_h + u_h^\top R u_h \right), \tag{24}$$

where e_h denotes the tracking-error state and u_h denotes the control. In our rollout, the predicted poses at $0.5\,\mathrm{s}$ intervals are first interpolated to the $0.1\,\mathrm{s}$ simulation grid. The longitudinal controller tracks the reference velocity through acceleration commands, while the lateral LQR uses lateral displacement, heading, and steering-angle errors to determine the steering-rate command. First-order actuator dynamics are applied before propagating the 11-dimensional vehicle state with the kinematic bicycle model. We simulate $N_q = 60$ poses with an LQR horizon of 10 steps.

NAVSIM's official tracker, simulator, and drivable-area routines operate on NumPy arrays and therefore cannot preserve gradients from the compliance objective to the action generator. We retain the state definition, controller structure, vehicle geometry, and drivable-map semantics of the official implementation, but reimplement the rollout and footprint queries as batched PyTorch tensor operations. Specifically, we obtain the binary drivable region using the official PDMDrivableMap, convert it into an ego-centric signed-distance tensor, and evaluate the four vehicle-corner points at every simulated pose by differentiable bilinear interpolation. Out-of-bound samples are assigned negative distances. The softplus margin penalty and log-mean-exp aggregation described in the main paper are then evaluated entirely in PyTorch, allowing $\mathcal{L}_{\mathrm{dac}}$ to backpropagate through the LQR rollout to the predicted actions.

Inference. At deployment, the future-video branch is removed. The observation prefix is encoded once and its key/value states are cached, and the 12 action tokens are denoised in two flow steps. When future-video generation is needed for analysis, noisy future-video latent tokens can be appended to the input and denoised by the same model, after which the predicted latents are decoded into future video. Because the action-centered causal mask prevents action tokens from attending to future-video tokens, this optional video branch does not affect the action path. We empirically verify that, under the same number of inference steps, the masked model produces identical trajectories up to numerical precision with and without future-video tokens. This observation is consistent with the causal masking design and supports action-only deployment. The main-paper latency numbers are measured on a single NVIDIA H20 with batch size 1 and bf16 precision; additional details are provided in the inference-efficiency section.

B Dataset and Metric Details

B.1 NAVSIM

Overview. The NAVSIM benchmark family is built on OpenScene, a $2\,\mathrm{Hz}$ subset of the nuPlan driving logs with approximately 120 hours of real-world driving data. OpenScene provides the standard trainval, test, and mini splits. The trainval split contains $14\,\mathrm{GB}$ of log files and more than $2000\,\mathrm{GB}$ of sensor data, the test split contains $1\,\mathrm{GB}$ of log files and $217\,\mathrm{GB}$ of sensor data, and the mini split contains $1\,\mathrm{GB}$ of log files and $151\,\mathrm{GB}$ of sensor data. NAVSIM further defines filtered planning splits over these data to emphasize non-trivial driving scenes. Specifically, navtest is derived from test and is used for standardized NAVSIMv1 evaluation. In our experiments, we train SV-WAM on the NAVSIM trainval data and report the main results on the NAVSIMv2 (Dauner et al., 2024; Cao et al., 2025) navtest split.

NAVSIMv2 extends the original NAVSIM benchmark with pseudo-simulation (Cao et al., 2025). Instead of evaluating only the logged observation at the initial state, NAVSIMv2 uses pre-generated synthetic observations to approximate shifted future states that a planner may encounter after deviating from the expert trajectory. This design allows the benchmark to evaluate error recovery and causal-confusion sensitivity while avoiding expensive sequential sensor rendering. Besides the main navtest evaluation, we also report results on the harder navhard two-stage split, which contains real Stage 1 observations and synthetic Stage 2 observations and applies a two-stage evaluation protocol with human penalty. Finally, we include NAVSIMv1 navtest results under the original PDMS metric to connect our results with earlier NAVSIM-based work.

Data Processing. Each raw camera view in NAVSIM has a resolution of 1920×1080 pixels. We first resize each selected camera view to 455×256 pixels and then center-crop it to 448×224 pixels. At each timestamp, the six processed camera views are arranged in the fixed order described above and concatenated along the width dimension into a single surround-view mosaic. We use four history frames sampled at $2\,\mathrm{Hz}$, corresponding to a temporal interval of $0.5\,\mathrm{s}$.

NAVSIMv2 Metric (EPDMS). The Extended Predictive Driver Model Score (EPDMS) aggregates multiplicative safety penalties and weighted quality terms,

$$\mathrm{EPDMS} = \underbrace{\mathrm{NC} \cdot \mathrm{DAC} \cdot \mathrm{DDC} \cdot \mathrm{TLC}}_{\text{multiplicative safety}} \cdot \frac{5(\mathrm{EP}{+}\mathrm{TTC}) + 2(\mathrm{LK}{+}\mathrm{HC}{+}\mathrm{EC})}{16}. \tag{25}$$

Here the four safety terms act as multiplicative gates: NC (no at-fault collision) penalizes collisions for which the ego is responsible, DAC (drivable-area compliance) checks that the rolled-out vehicle footprint stays inside the drivable area, DDC (driving-direction compliance) penalizes wrong-way or illegal-direction motion, and TLC (traffic-light compliance) penalizes running red lights. The remaining terms are quality scores combined as a weighted average: EP (ego progress) measures route progress relative to a privileged reference, TTC (time-to-collision) rewards keeping a safe temporal margin to other agents, LK (lane keeping) measures staying within the lane, and HC (history comfort) and EC (extended comfort) bound acceleration and jerk for ride comfort. Because NC, DAC, DDC, and TLC enter multiplicatively, a single violation (for example, leaving the drivable area) zeroes the entire scenario score; this is exactly the failure mode our drivable-area regularizer targets. The navtest split contains 12,146 scenarios and the navhard split 450 scenarios (official sizes).

NAVSIMv2 navhard. The navhard split (Table 5) is a harder NAVSIMv2 setting that targets interactive, safety-critical situations. It uses a two-stage evaluation: the planner is first scored on the recorded scene and then re-scored under more challenging conditions, and an additional human-behavior penalty is applied. Because the second stage stresses reactive behavior, navhard scores are markedly lower than navtest for all methods, so the split mainly measures robustness in difficult scenes.

B.2 nuScenes

Overview. nuScenes (Caesar et al., 2020) contains 1000 driving scenes of roughly 20 s each, collected in Boston and Singapore with a synchronized six-camera surround rig and keyframe annotations at 2 Hz. It spans diverse urban traffic, lighting, and weather conditions, which makes it a strong testbed for cross-dataset generalization. We use it purely for evaluation and never train or fine-tune on it.

Zero-Shot Protocol. For zero-shot evaluation, the compared world-model planners, including DriveVA, DriveVLA-W0, PWM, and SV-WAM, are trained only on NAVSIM and evaluated directly on nuScenes without target-domain fine-tuning. We evaluate on the nuScenes validation split using the standard open-loop planning protocol. For SV-WAM, we use the four most recent keyframes at 2 Hz from all six cameras. Each native RGB image has a resolution of 1600×900 pixels. We bilinearly resize each image to 455×256 pixels and then center-crop it to 448×224 pixels, matching the deterministic preprocessing used during NAVSIM training. The six processed views are concatenated along the width dimension in the same fixed order. The provided planning command is mapped to our four-way command encoding, while ego velocity and acceleration are normalized using the NAVSIM training statistics. We neither compute target-domain normalization statistics nor perform any parameter adaptation on nuScenes.

Metrics. Following the standard nuScenes open-loop planning evaluation used in UniAD (Hu et al., 2023b) and subsequent planning works (Jiang et al., 2023), we report L2 displacement error and collision rate at 1 s, 2 s, and 3 s. The L2 displacement error measures the Euclidean distance between the predicted ego position and the ground-truth ego position at each future horizon, in meters. The collision rate measures the percentage of predicted trajectories that collide with annotated scene agents within each evaluation horizon. We also report the average L2 error and average collision rate over the three horizons.

C Additional Quantitative Results

We provide the harder NAVSIMv2 navhard comparison omitted from the main paper for space (Table 5).

Table 5: Detailed comparison on the NAVSIMv2 navhard split with human penalty enabled. Each method is evaluated in two stages; EPDMS is the final split score.

Method	Stage	NC ↑	DAC ↑	DDC ↑	TLC ↑	EP ↑	TTC ↑	LK ↑	HC ↑	EC ↑	EPDMS ↑
LTF (2021)	1	96.2	79.6	99.1	99.6	84.1	95.1	94.2	97.6	79.1	25.1
	2	77.8	70.2	84.3	98.1	85.1	75.7	45.4	95.7	76.0	
LTFv6 (2026)	1	96.6	86.7	99.2	99.6	84.5	95.1	94.4	97.8	76.4	31.9
	2	79.9	75.6	86.3	97.9	89.6	76.1	50.1	95.2	66.7	
RAP (2026)	1	97.1	94.4	98.8	99.8	83.9	96.9	94.7	96.4	66.2	39.6
	2	83.2	83.9	87.4	98.0	86.9	80.4	52.3	95.2	52.4	
DriveVLA-W0 (2026b)	1	96.8	83.3	99.0	99.6	**84.6**	95.3	96.4	97.6	78.2	24.4
	2	76.8	64.3	79.9	98.3	**89.2**	75.0	46.8	95.8	53.1	
DriveLaW (2026)	1	**97.3**	89.1	99.2	99.6	84.3	97.1	96.2	**97.8**	67.6	30.6
	2	82.5	67.6	83.5	98.1	84.8	78.5	45.8	**96.4**	57.3	
EponaV2 (2026)	1	**97.3**	90.7	**99.4**	**100.0**	83.3	**97.3**	**97.3**	97.6	60.9	**36.1**
	2	**83.6**	**78.0**	**88.0**	**98.9**	86.0	**80.3**	**50.1**	96.1	52.0	
Ours	1	96.6	**94.9**	98.8	99.8	82.5	96.9	97.1	**97.8**	**84.4**	**36.1**
	2	82.9	75.8	84.3	98.0	82.1	79.1	48.2	**96.4**	**80.1**	

Table 6: Action-token ablation on NAVSIMv2 navtest (two denoising steps, C×6 input). Metrics are reported as percentages.

Predicted Action Tokens	NC ↑	DAC ↑	DDC ↑	TLC ↑	EP ↑	TTC ↑	LK ↑	HC ↑	EC ↑	EPDMS ↑
8	98.4	98.5	99.7	99.9	86.3	98.4	97.5	98.4	88.3	90.4
12	98.6	98.8	99.6	99.9	86.8	98.3	97.8	98.4	88.4	**91.0**
16	98.5	98.2	99.7	99.8	86.9	98.1	97.9	98.3	88.1	90.5

NAVSIMv2 navhard. On the harder navhard split (Table 5), which combines a two-stage evaluation with a human-behavior penalty, SV-WAM reaches 36.1 EPDMS, tying EponaV2 and ranking behind RAP (39.6) among the compared methods. It also achieves 94.9 first-stage DAC and strong extended-comfort scores (EC, 84.4 and 80.1 across the two stages), indicating robust behavior under the more challenging reactive conditions.

D Additional Ablations

Unless varied, all ablations use two denoising steps, twelve action tokens, and six-view (C×6) input on NAVSIMv2 navtest.

Number of Action Tokens. Table 6 evaluates the sensitivity of SV-WAM to the number of predicted action tokens. Using eight tokens shortens the prediction horizon and slightly reduces EPDMS, while increasing the number to sixteen introduces additional long-horizon uncertainty. Performance remains stable across the tested settings, with twelve action tokens yielding the highest EPDMS in this sensitivity analysis. We use twelve action tokens in all main experiments.

Training-Side Hyperparameters. Table 7 reports sensitivity analyses for the drivable-area regularization weight and the effective fine-tuning batch size. EPDMS remains stable for small regularization weights: both $\lambda_{\mathrm{dac}} = 0.01$ and 0.05 achieve 91.0 EPDMS, whereas increasing the weight to 0.1 reduces the score to 90.5. We use $\lambda_{\mathrm{dac}} = 0.01$ in all main experiments. Performance also varies only marginally across effective fine-tuning batch sizes of 320, 640, and 1280, with 640 yielding the highest score. We further observe that planning performance is largely insensitive to the future-video loss weight within the tested range, and use $\lambda_{\mathrm{act}} : \lambda_{\mathrm{vid}} = 1 : 1$ throughout.

Table 7: Additional training hyper-parameter ablations on NAVSIMv2 navtest. (a) Drivable-area regularizer weight λ_{dac}; (b) effective fine-tuning batch size. Defaults are $\lambda_{\mathrm{dac}} = 0.01$ and batch size 640. EPDMS is reported as a percentage.

λ_{dac}	EPDMS ↑
0.01	**91.0**
0.05	**91.0**
0.1	90.5

(a) Drivable-area weight

FT batch	EPDMS ↑
320	90.4
640	**91.0**
1280	90.9

(b) Fine-tuning batch size

Table 8: Online trajectory-output latency comparison among WM-based planning methods on a single NVIDIA H20 GPU. Vis. Enc., Core Gen., and Other denote visual tokenization, iterative backbone generation, and remaining pipeline overhead, respectively. Component-wise values are means in milliseconds, while total latencies are reported as mean ± standard deviation. Optional video rendering and future-video VAE decoding are excluded.

Method	Vis. Enc.	Core Gen.	Other	Total (ms) ↓ mean±std
Epona	725.4	415.2	8.2	1148.8±4.5
DriveLaW	459.7	663.7	259.9	1383.3±19.7
PWM	210.2	336.0	6.1	552.3±5.4
DriveVLA-W0	76.6	369.0	69.3	514.9±3.8
Ours (w/o mask)	438.4	392.5	17.1	848.0±1.8
Ours	174.7	151.4	15.5	**341.6±2.1**

E Inference-Efficiency Details

H20 Latency Breakdown. Table 8 reports the per-component H20 latency used for the main-paper comparison.

The main paper compares the online latency of SV-WAM with world-model-based baselines. Latency is measured from preprocessed model inputs to the final trajectory, covering visual encoding/tokenization, the core generator, and Other, which includes online state and prompt preparation, tensor transfers, latent/token packing, scheduler updates, and action decoding or post-processing. For reference, on a single H20 GPU, SV-WAM's deployed action-only path is dominated by six-view VAE encoding (∼175 ms) and two-step denoising (∼151 ms); enabling the future-video branch raises the trajectory-only latency to 848.0 ± 1.8 ms, while optional future-video VAE decoding, required only when future-video outputs are requested, adds a further 1394.3 ± 3.3 ms. We additionally provide a deployment-oriented breakdown on a single H800 GPU below.

Deployment at 2 Hz on H800. On a single H800 GPU, SV-WAM runs the full preprocessed-input-to-trajectory pipeline at roughly 2 Hz in the deployed action-only setting, i.e. within the 0.5 s budget of a 2 Hz planner. Table 9 breaks down the H800 cost of the action-only path against the joint action–video path.

F Additional Qualitative Results

F.1 Comparison in a Challenging Turning Scenario

Figure 5 compares SV-WAM with DriveLaW and DriveVLA-W0 in a challenging turning scenario. Benefiting from richer spatial context and the drivable-area compliance regularizer, SV-WAM predicts a trajectory that more closely follows the ground-truth path and remains within the drivable corridor.

Table 9: Inference-speed breakdown on a single H800 GPU (batch size 1, 12 action tokens, 2-step inference). Total is measured from preprocessed inputs to the outputs produced by each inference path. Action-only is the deployed path; joint action–video additionally denoises and decodes future video (the reported total also includes ∼12 ms of other online operations, such as tensor packing).

Inference path	Input Enc.	Core Gen.	VAE Dec.	Total	Speedup
Action-only (deployed)	∼53 ms	∼123 ms	–	∼176 ms	∼4.4×
Joint action–video	∼135 ms	∼201 ms	∼428 ms	∼776 ms	∼1.0×



Figure 5: Qualitative comparison in a challenging turning scenario from NAVSIMv2. Left: current six-camera observations. Right: the corresponding BEV visualization. SV-WAM more closely follows the ground truth and road geometry than DriveLaW and DriveVLA-W0.

F.2 Effect of Camera Coverage

Figure 6 complements the attention analysis and the controlled camera-coverage ablation reported in the main paper. Compared with C×1 and C×3, full C×6 coverage provides rear and lateral context that captures nearby vehicles and road boundaries not fully visible under reduced camera coverage. Consequently, the C×6 prediction remains closer to the ground-truth trajectory and lane center, whereas the reduced-view variants exhibit larger lateral deviations.

Surround-View Attention Analysis. To further examine how SV-WAM uses the additional camera views, we visualize the attention from the future action tokens to the six historical camera views during a left lane-change maneuver, averaged over the 30 transformer blocks. As shown in Figure 7, instead of relying on the front view alone, the action tokens place their strongest weight on the front-left and rear views, which contain the neighboring vehicle that constrains the maneuver. This provides direct evidence that the action generator reads side and rear context when making lateral decisions, consistent with the surround-view coverage ablation.

F.3 Zero-shot In-house Transfer

Figure 8 compares zero-shot planning on an in-house left-turn waiting-lane scenario. Epona and PWM use their official code and released checkpoints (PWM uses the NAVSIM-finetuned checkpoint), and none of the methods is fine-tuned on the target domain. Front-only Epona misses the vehicle approaching from the rear-left and collides. PWM avoids colliding with the rear-left vehicle, but its zero-shot trajectory is poor and deviates substantially from the ground truth. In contrast, SV-WAM uses the surround-view observation and remains closest to the ground truth while clearing the rear-left vehicle.

F.4 Scenario-wise Planning Examples

Figures 9 and 10 present representative high-scoring trajectories on the NAVSIMv2 navtest split, across straight driving, turning, and intersection scenarios. The same action-only planner remains



Figure 6: Qualitative comparison of camera coverage on NAVSIMv2. Left: synchronized six-camera observations with ground-truth and predicted trajectories projected onto the front view. Right: the corresponding BEV trajectories. Full C×6 coverage provides rear and lateral context unavailable to the reduced-view settings, keeping the predicted trajectory closer to the ground truth and lane center.

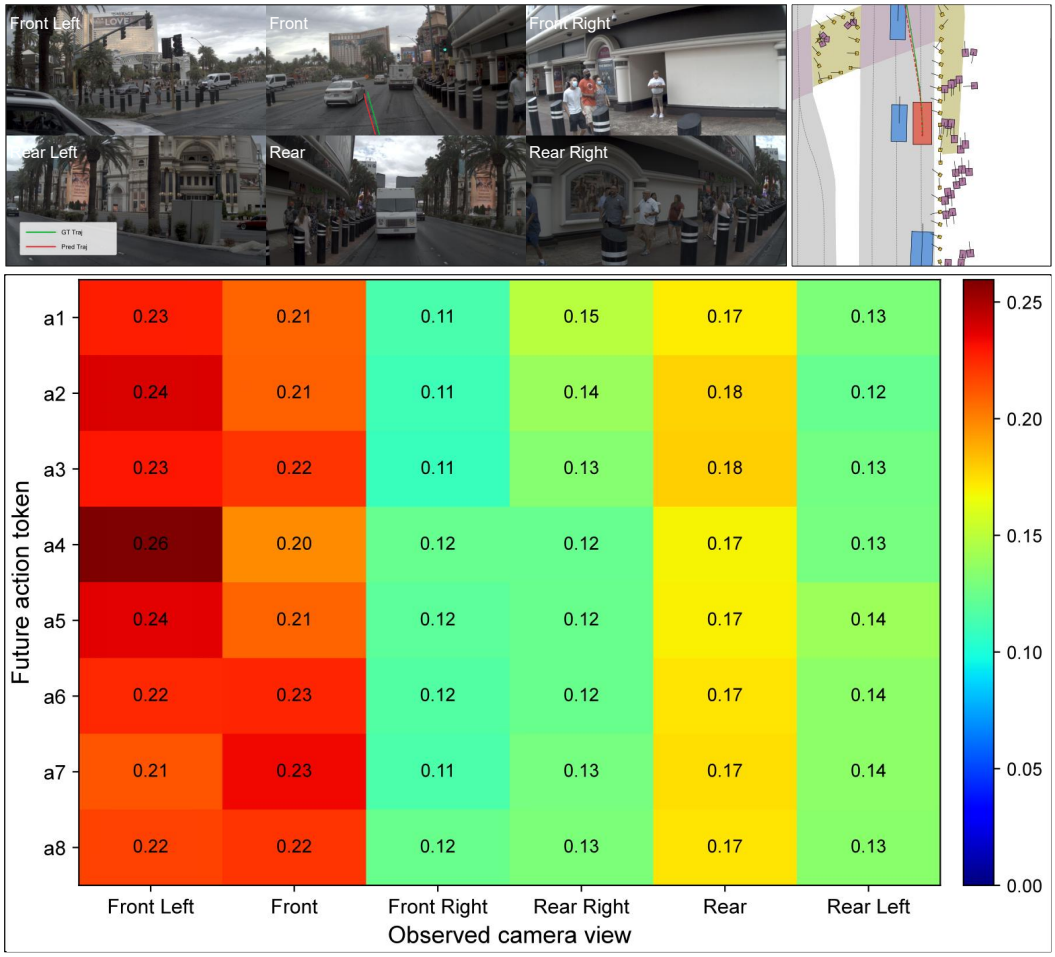

Figure 7: Action-to-view attention in a left lane-change, averaged over the 30 transformer blocks. The future action tokens attend most strongly to the front-left and rear views, where the neighboring vehicle that constrains the maneuver appears, evidencing that the planner exploits surround-view input.

stable across these common closed-loop maneuvers despite their different interaction and geometry patterns.

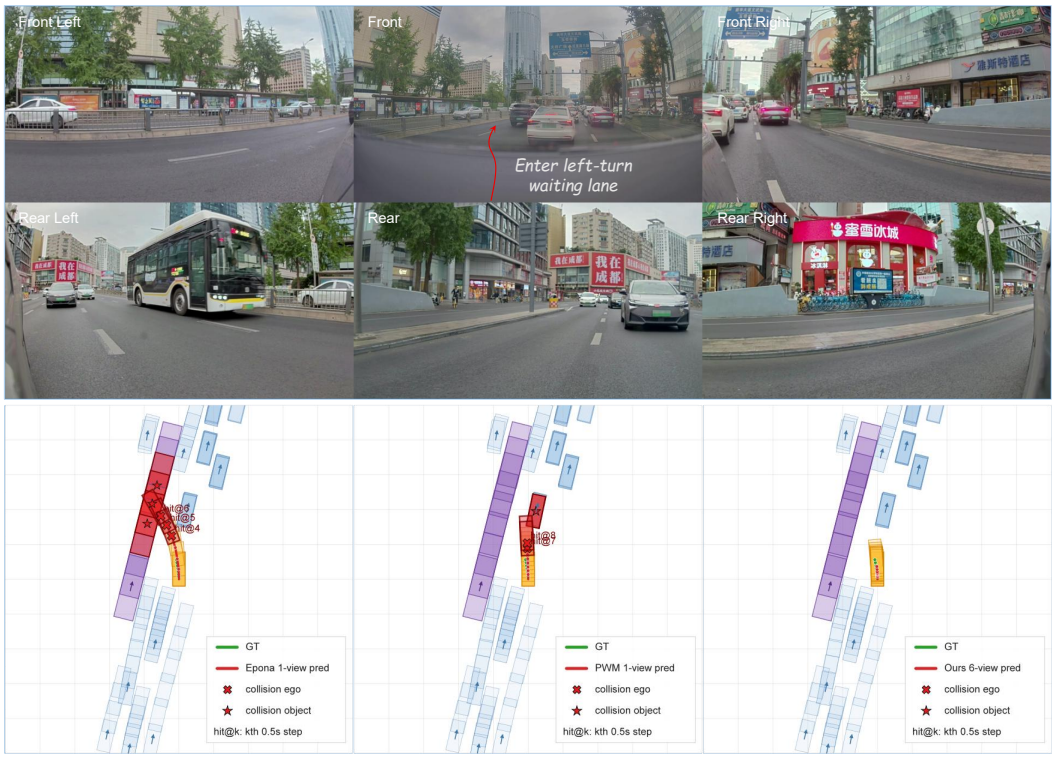

Figure 8: Zero-shot qualitative comparison on in-house data when entering a left-turn waiting lane. Epona (front view only) collides with the rear-left vehicle; PWM avoids the rear-left vehicle but its zero-shot trajectory drifts substantially from the ground truth; SV-WAM remains closest to the ground-truth trajectory.

F.5 Comparison with Other Methods

Figures 11–15 compare DriveLaW, DriveVLA-W0, and SV-WAM across 15 representative NAVSIMv2 navtest scenarios. Each row is one scene, and the three columns correspond to DriveLaW, DriveVLA-W0, and SV-WAM. DriveLaW and DriveVLA-W0 use front-view input, whereas SV-WAM uses six-camera surround-view input. The examples include dense intersections, turning maneuvers, and interactions with nearby traffic.

F.6 Failure Cases

Figure 16 presents representative failure cases to characterize the remaining limitations of SV-WAM. In the first case, the ambiguous two-branch left-turn geometry causes all three methods to select an incorrect path. In the second case, heavy rain obscures the forward view and traffic signal, leading all three methods to fail. In the third case, SV-WAM accelerates too conservatively after the signal turns green and is rear-ended, while DriveLaW and DriveVLA-W0 complete the scenario successfully. These examples suggest that stronger interaction reasoning and more robust uncertainty handling remain important directions for future work.

G Limitations and Future Work

Although SV-WAM achieves strong planning performance with efficient action-only inference, its approximately 5B-parameter backbone remains relatively large for deployment on resource-constrained on-board platforms. Looking forward, we plan to scale training with larger and more diverse real-world driving data to further improve robustness and generalization, while exploring knowledge distillation, pruning, and quantization to derive lightweight variants suitable for edge

deployment. We will also extend the current benchmark-based evaluation to closed-loop testing on real vehicles, providing a more comprehensive assessment of reliability and practical deployability under diverse traffic and environmental conditions.

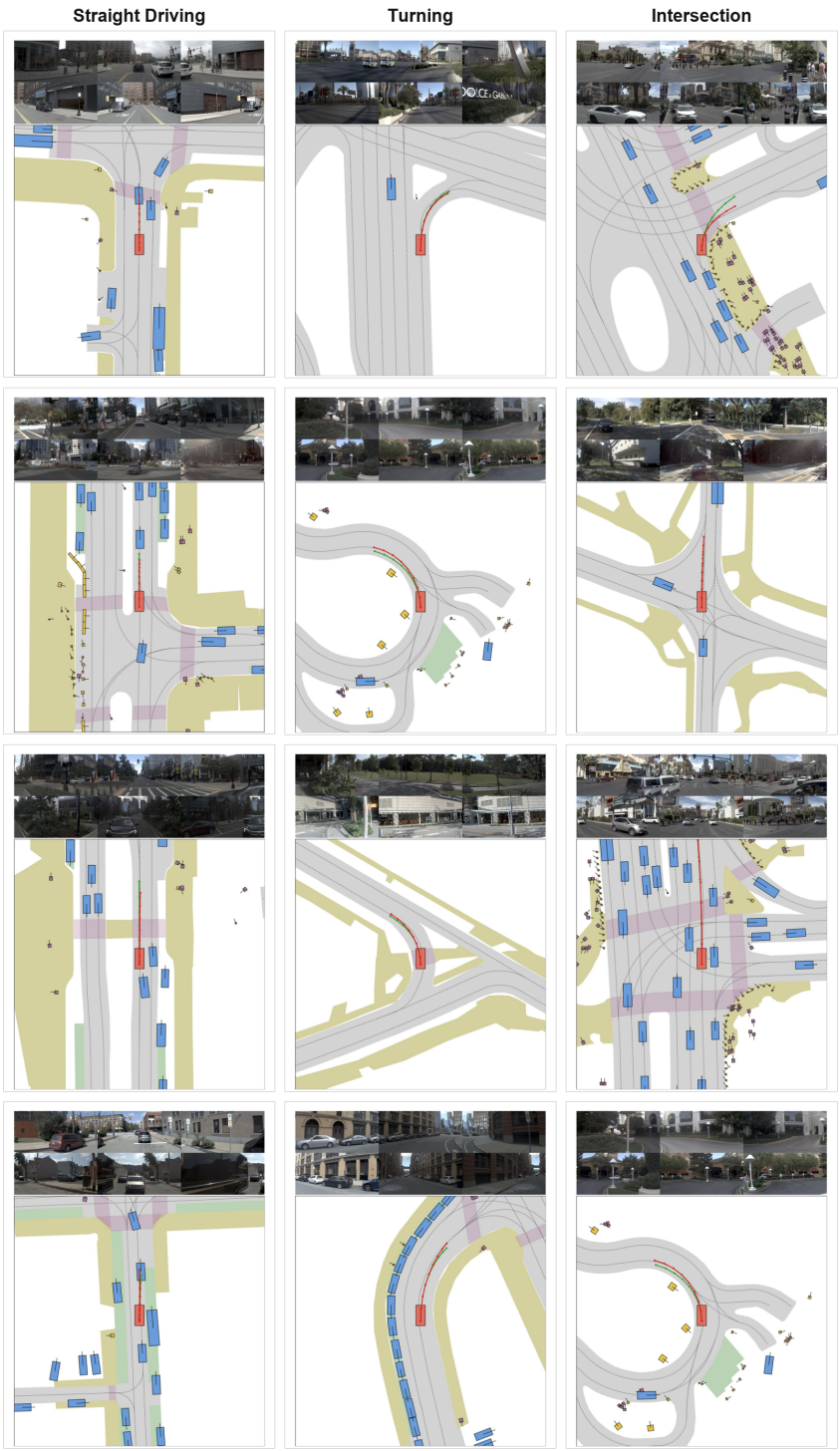

Figure 9: Representative high-scoring planning examples grouped by maneuver type. Columns are straight driving, turning, and intersection scenarios; each column shows four examples. Green curves denote the ground-truth trajectories, and red curves denote the predicted trajectories.



Figure 10: Additional high-scoring planning examples grouped by maneuver type, four per column.

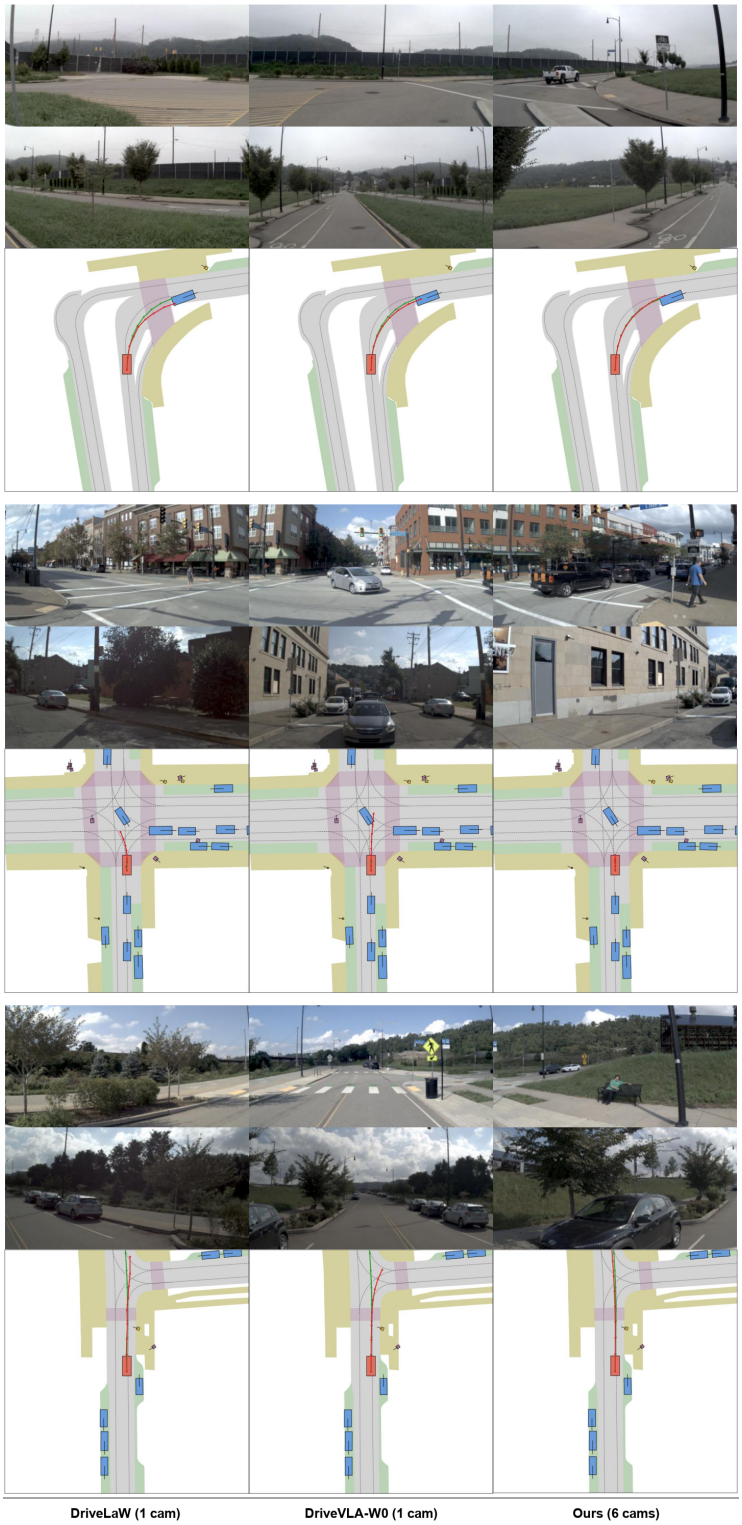

Figure 11: Qualitative comparison with front-view baselines. Each row is one scene; the three columns, labeled at the bottom of the figure, are DriveLaW and DriveVLA-W0 (front camera only, C×1) and SV-WAM (six-camera surround-view, C×6).

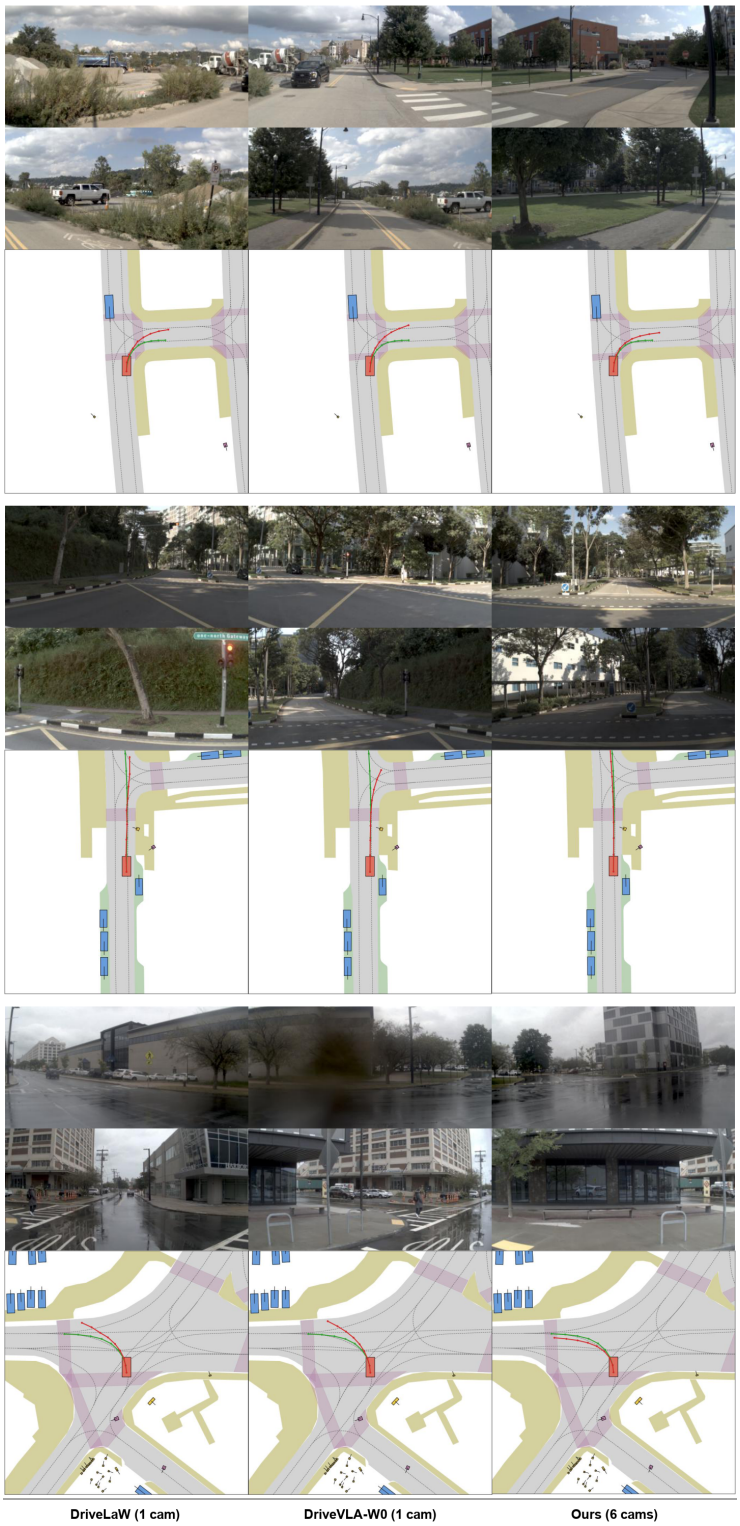

Figure 12: Additional qualitative comparison with front-view baselines; same three-column layout (DriveLaW, DriveVLA-W0, SV-WAM) and camera labeling as in Figure 11.

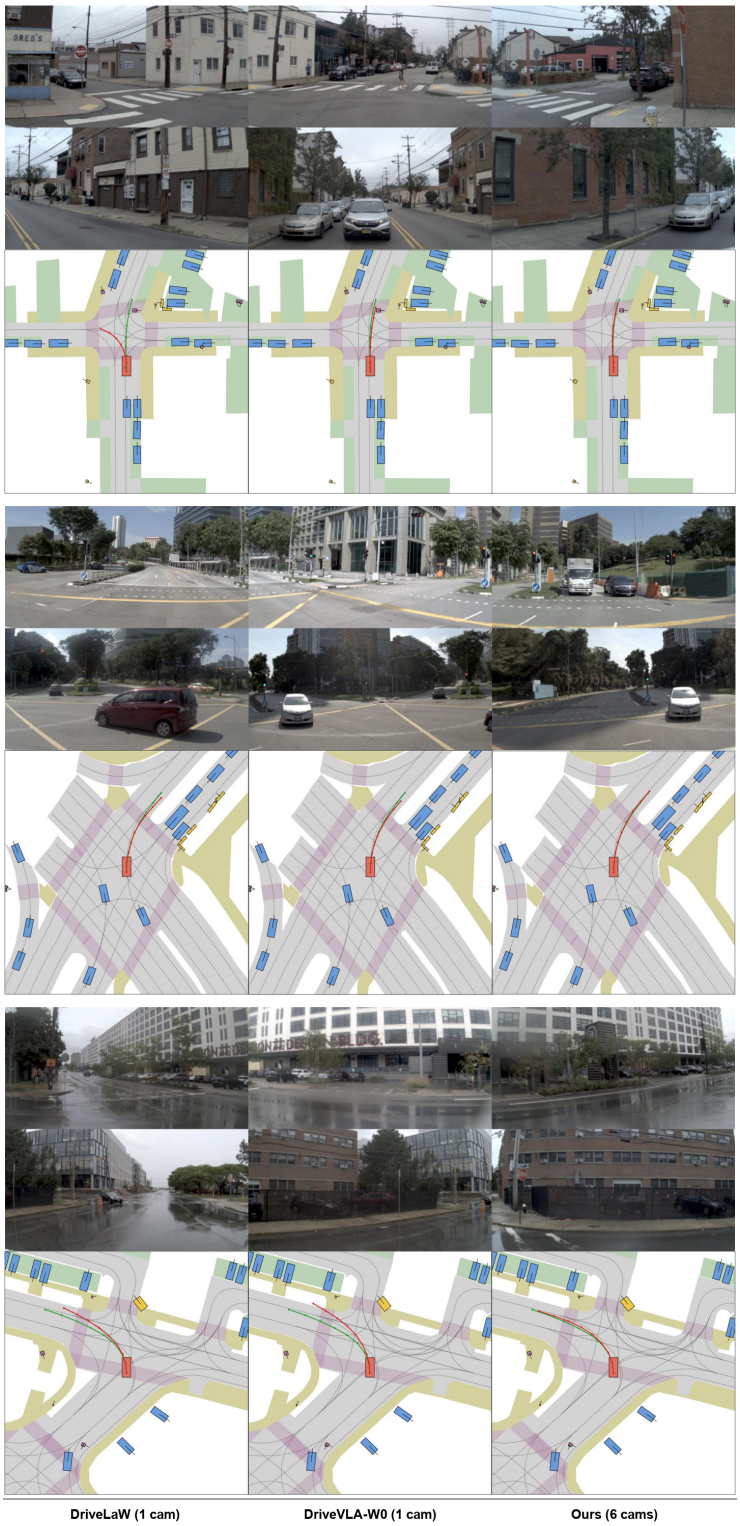

Figure 13: Additional qualitative comparison with front-view baselines; same three-column layout (DriveLaW, DriveVLA-W0, SV-WAM) and camera labeling as in Figure 11.

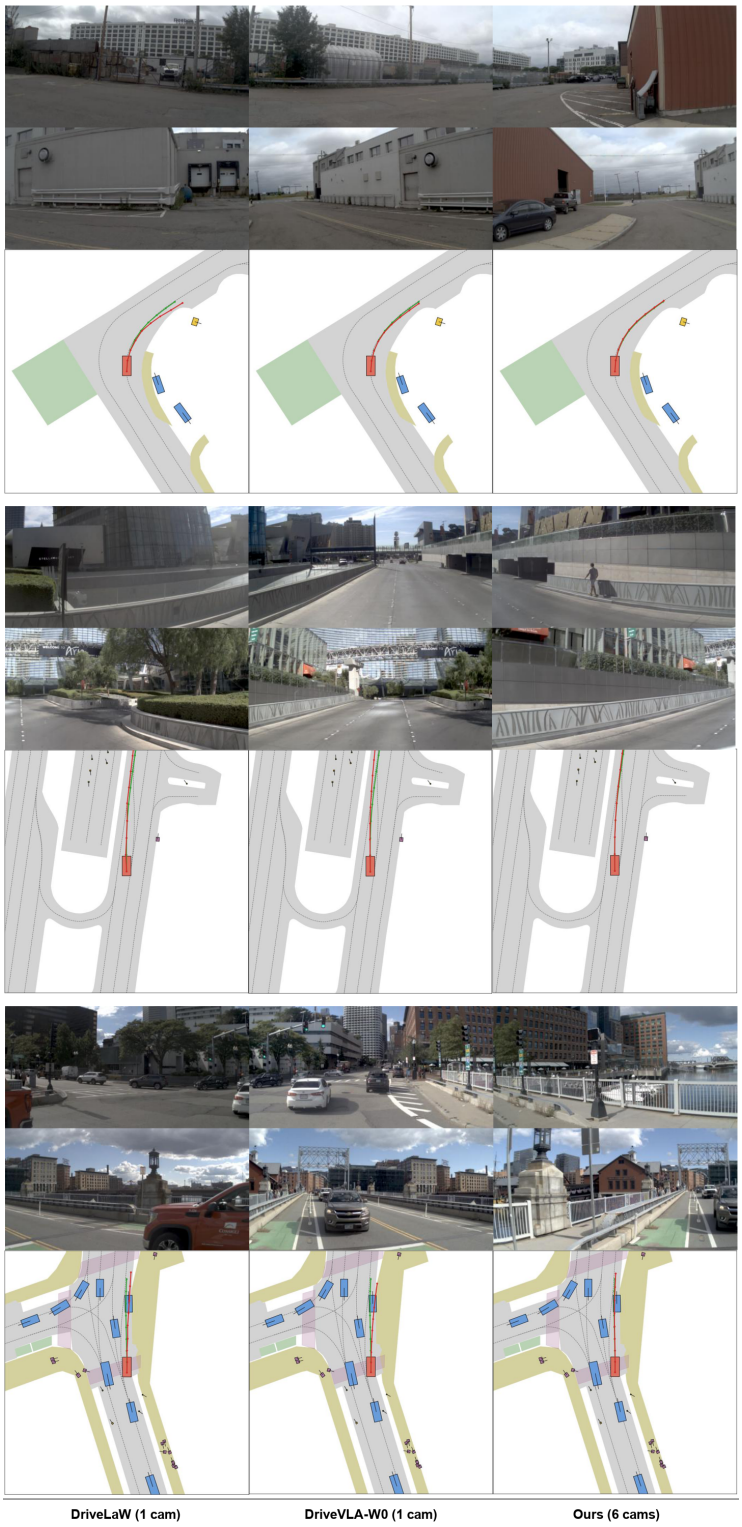

Figure 14: Additional qualitative comparison with front-view baselines; same three-column layout (DriveLaW, DriveVLA-W0, SV-WAM) and camera labeling as in Figure 11.

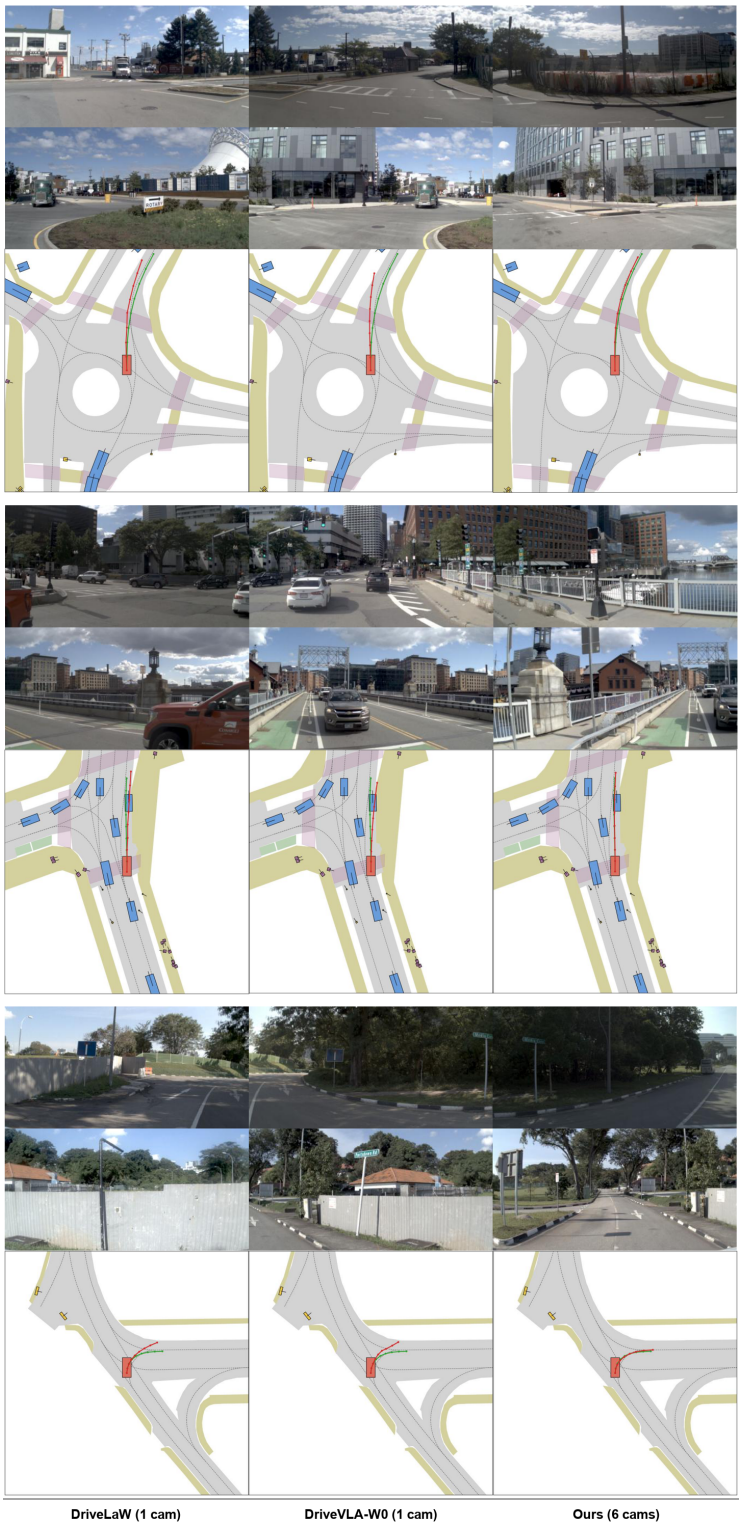

Figure 15: Additional qualitative comparison with front-view baselines; same three-column layout (DriveLaW, DriveVLA-W0, SV-WAM) and camera labeling as in Figure 11.

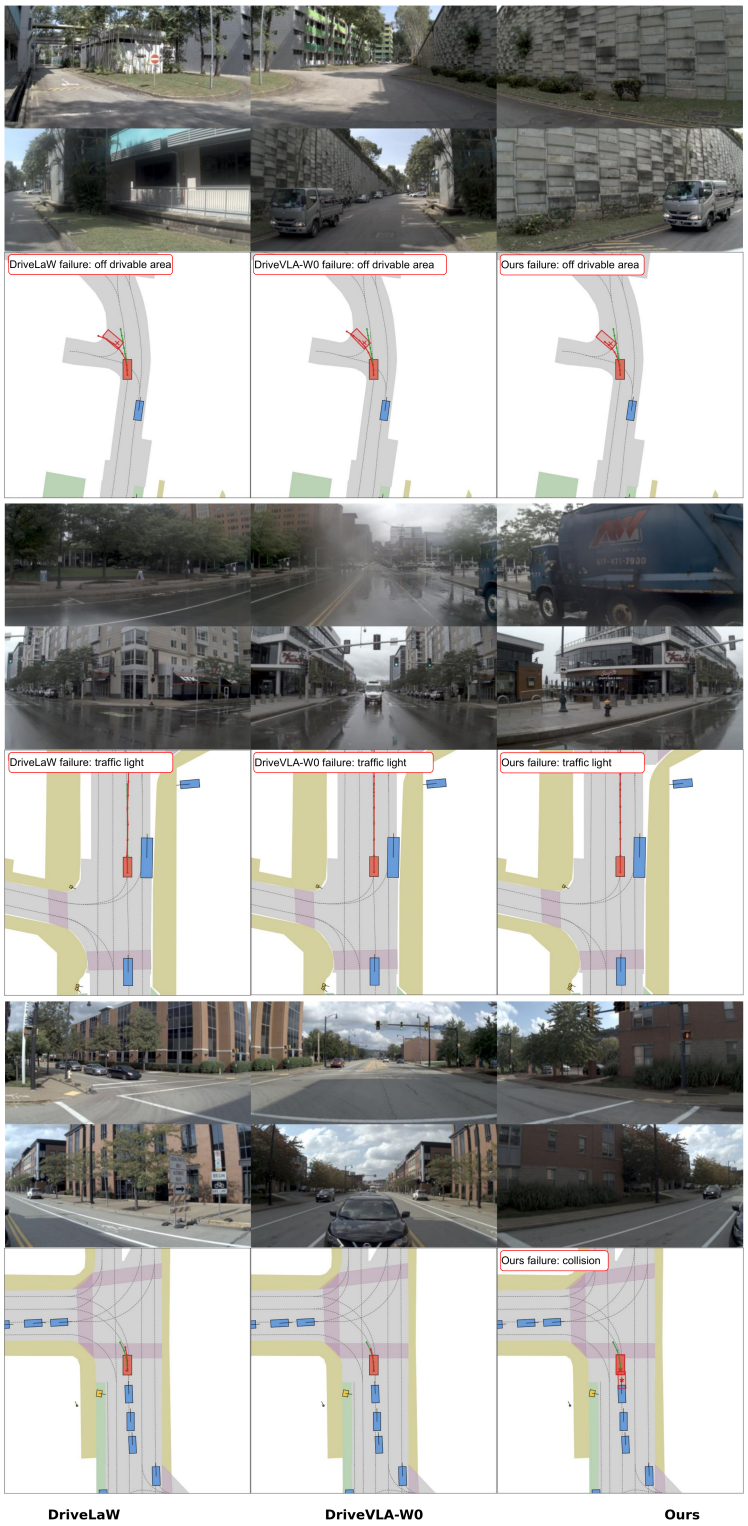

Figure 16: Representative failure cases under challenging conditions. Each row compares DriveLaW, DriveVLA-W0, and SV-WAM. The three cases correspond to ambiguous left-turn geometry, rain-obscured traffic-signal perception, and overly conservative acceleration after a green signal, respectively.