SIMVTA: SIMULATION-GROUNDED VIDEO–TRAJECTORY ALIGNMENT FOR DRIVING WORLD ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Driving world action models (WAMs) learn from paired trajectories and future observations. Post-training with reinforcement learning allows the policy to explore alternative trajectories and improve planning using driving rewards. However, driving logs contain future videos only for the recorded trajectories, leaving these alternatives without corresponding visual supervision. Thus, we introduce SIMVTA, a simulation-grounded post-training framework that combines trajectory optimization with video supervision. First, we sample candidate trajectories for each observation and use real-world simulator (e.g., World Engine) to render future videos for high-reward candidates in real-world scenes reconstructed with 3D Gaussian Splatting (3DGS). This provides trajectory-matched supervision independent of the model's own predictions. Under a limited rendering budget, we prioritize video supervision for higher-error futures among these rendered candidates. Finally, we optimize trajectory planning with Group Relative Policy Optimization (GRPO) and use a flow-matching loss on the matched videos to update shared representations. At inference time, the model predicts trajectories directly, without simulator calls. On NAVSIM v1, applying SIMVTA to a WAM initialization improves PDMS from 86.2 to 91.3, a gain of 5.1 points. It achieves 88.2 EPDMS on NAVSIM v2 and an average HD-Score of 37.3 in closed-loop evaluation on HUGSIM. Ablations show a clear planning benefit from allowing video supervision to update shared representations and suggest an additional benefit from prioritizing higher-error futures among high-reward trajectories.

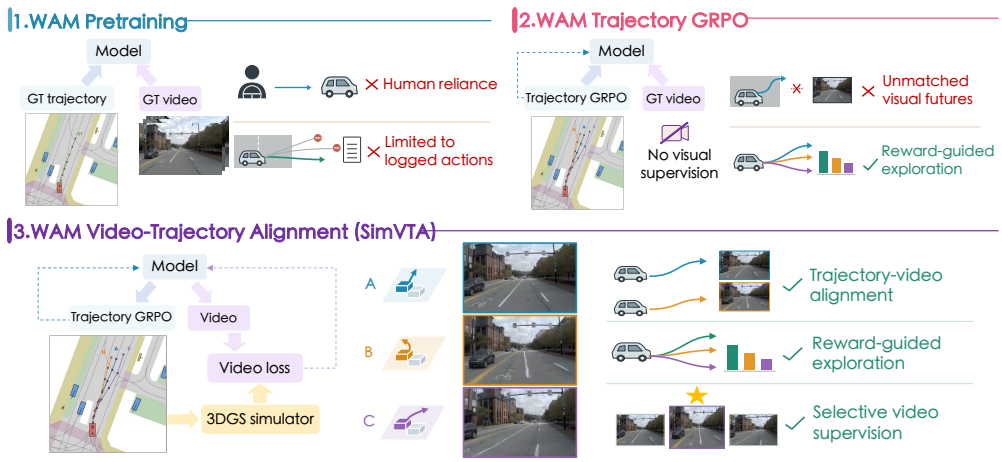


Figure 1: **From logged futures to candidate-matched supervision.** (1) WAM pretraining uses the ground-truth (GT) trajectory and its logged future video. (2) Trajectory GRPO explores candidate trajectories using driving rewards, without matched video supervision. (3) World Engine renders reconstructed scenes along selected candidate paths, providing matched supervision independent of the model's predictions.

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

Driving WAMs combine future-scene prediction with trajectory planning for the ego vehicle. Predicting future frames provides dense visual supervision that complements sparse trajectory targets and helps models learn representations for planning (Li et al., 2025b). Recent driving models retain video prediction during training but predict trajectories directly at inference time, avoiding the computational cost of future-video generation (Zhao et al., 2026).

Reinforcement learning can refine a planner trained on demonstrations to explore alternative trajectories and learn from driving rewards (Zhou et al., 2025). However, driving logs contain future videos only for recorded trajectories. For example, when a candidate turns while the recorded trajectory continues straight, the logged video does not show the visual consequences. Reusing such videos creates incorrect trajectory–video pairs, while omitting video training leaves these candidates without visual supervision. We therefore seek matched future-video targets for current-policy trajectories to improve planning (Figure 1).

To address this gap, we introduce SIMVTA, a simulation-grounded framework for video–trajectory alignment. A WAM's own video predictions provide no external reference for checking whether they follow a candidate trajectory. We use the 3DGS-based simulator in World Engine (Li et al., 2026b) to render reconstructed real-world scenes from camera poses along executed candidate paths (Figure 1). These renderings provide targets based on scene geometry and candidate motion, independently of the WAM's predictions. This establishes *video–trajectory alignment* by providing a corresponding simulated future video for selected trajectory.

We jointly train video prediction with flow matching and trajectory planning with GRPO, allowing the video loss to update shared representations. The former teaches the model to predict the visual consequences of candidate trajectories; the latter compares driving rewards among candidates and favors higher-reward trajectories.

Rendering videos for every sampled trajectory is costly. We first select high-reward candidates for rendering, then prioritize video supervision for higher-error futures among the rendered candidates. This separates trajectory exploration from video-supervision target selection. Our ablations examine shared representation updates, target selection, and trajectory–video correspondence. At inference, the planner predicts trajectories directly, without simulator calls or future-video generation.

Our main contributions are summarized as follows:

- We introduce SIMVTA, a post-training framework that combines trajectory GRPO with simulator-based video supervision. It connects policy sampling, trajectory execution, rendering, and joint updates to learn the visual consequences of actions beyond those recorded in driving logs.
- We introduce a video-target selection strategy based on trajectory reward and video prediction error. It prioritizes higher-error futures of high-reward trajectories for video supervision without restricting trajectory exploration.
- Applying SIMVTA to a WAM initialization improves NAVSIM v1 PDMS by 5.1 points, from 86.2 to 91.3, and achieves 88.2 EPDMS on NAVSIM v2 and an average HD-Score of 37.3 on HUGSIM.

2 RELATED WORK

Driving World Action Models. Driving world models such as DriveDreamer (Wang et al., 2024) and Vista (Gao et al., 2024) generate controllable visual predictions of scene evolution. DriveVLA-W0 (Li et al., 2025b) enriches planning representations through future-image prediction. Epona (Zhang et al., 2025b) combines video and trajectory prediction through shared temporal representations. DriveVA (Liu et al., 2026b) and DriveWAM (Shi et al., 2026) adapt video-generative models to joint video–action prediction, while UNIVERSE (Liu et al., 2026a) studies shared generative parameters for video and trajectory prediction. Latent-WAM (Wang et al., 2026b) and WA-JEPA (Wang et al., 2026c) use latent world representations for future prediction and planning, while DriveWorld-VLA (Jia et al., 2026) unifies world modeling and action prediction in a shared latent space. Fast-WAM (Yuan et al., 2026) studies video co-training with direct action inference in robotics. SimWAM (Zhao et al., 2026) retains video co-training while bypassing future-video generation for driving.

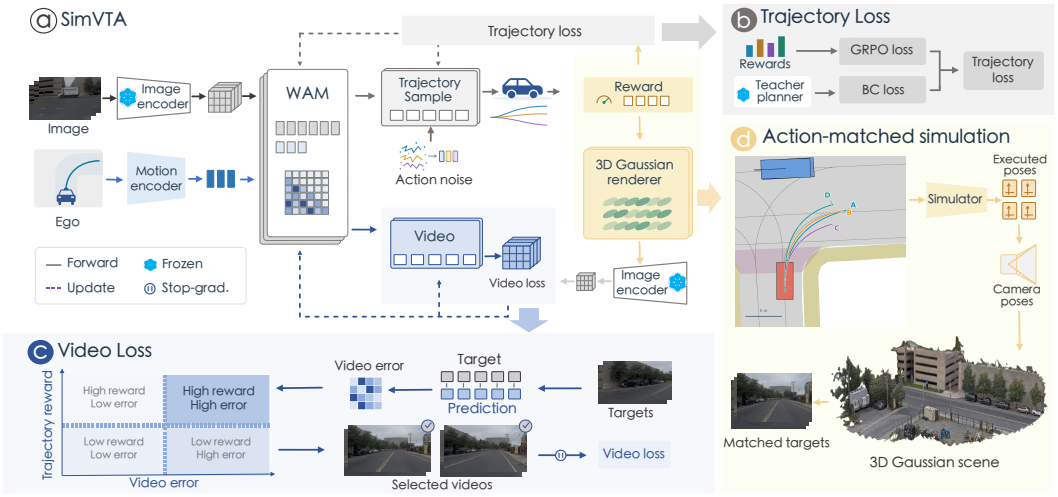


Figure 2: **Simulation-grounded video–trajectory alignment.** (a) Training overview. (b) Trajectory learning combines GRPO with a behavior-cloning (BC) regularizer. (c) Video supervision selects high-reward, higher-error futures. (d) World Engine renders matched futures along executed paths.

Simulation for Driving. CARLA (Dosovitskiy et al., 2017) and MetaDrive (Li et al., 2022) support interactive driving simulation, while ScenarioNet (Li et al., 2023) organizes real-world traffic scenarios. 3DGS (Kerbl et al., 2023) renders reconstructed scenes from specified camera poses, and HUGSIM (Zhou et al., 2024) integrates 3DGS with closed-loop driving simulation. SimScale (Tian et al., 2026) trains planners with simulated and real driving data. World Engine (Li et al., 2026b) supports policy post-training with 3DGS observations rendered along simulated ego motion. CausalDrive (Yan et al., 2026) uses an interactive learned renderer to support policy post-training.

Post-Training for Driving. CarPlanner (Zhang et al., 2025a) uses reinforcement learning for autoregressive trajectory planning. ReCogDrive (Li et al., 2026c) and Plan-R1 (Tang et al., 2026) adapt GRPO (Shao et al., 2024) to diffusion and autoregressive planners, respectively. Flow-GRPO (Liu et al., 2025) provides the stochastic differential equation (SDE) sampling formulation used in our trajectory optimization. AdaWM (Wang et al., 2025) selects whether to adapt a policy or its latent dynamics model based on their mismatch. ReSim (Yang et al., 2025) uses simulated non-expert trajectories to improve a learned video simulator's action controllability. SIMVTA combines trajectory GRPO with video supervision from a fixed renderer. The rendered videos provide matched targets for current-policy trajectories, and the video loss updates the shared representations.

3 METHOD

SIMVTA's post-training framework is illustrated in Figure 2 and comprises trajectory sampling, video rendering, and video supervision. We sample candidate trajectories from the current policy and execute them to obtain paths and driving rewards. World Engine (Li et al., 2026b) renders videos along selected paths during training. We select video targets based on trajectory reward and video prediction error, then update the model with trajectory GRPO and video supervision. The next sampling round uses the updated policy.

3.1 TRAJECTORY SAMPLING

Given observation history o, we compute a shared representation $F = h_\theta(o)$ from visual and ego-motion history for trajectory and video prediction. The trajectory policy π_ψ uses SDE sampling to generate G candidate trajectories:

$$F = h_\theta(o), \qquad \tau_i \sim \pi_\psi(\cdot \mid F), \qquad i = 1, \ldots, G. \tag{1}$$

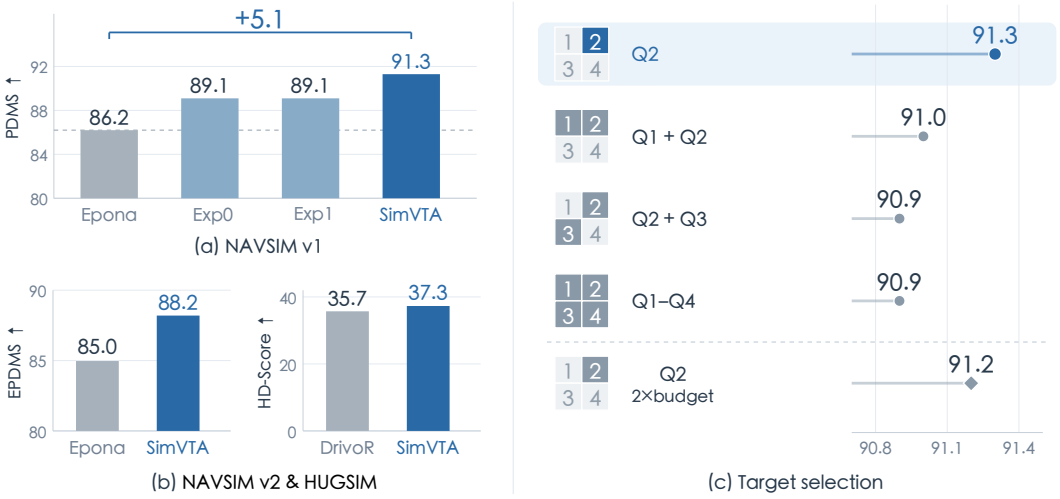


Figure 3: **Planning results and video-supervision comparisons.** (a) NAVSIM v1. (b) NAVSIM v2 and HUGSIM. (c) Video-supervision configurations and rendering budget on NAVSIM v1.

NAVSIM's PDM executor $\mathcal{E}$ converts each candidate into an executed ego path $\tilde{\tau}_i = \mathcal{E}(o, \tau_i)$. We compute its reward R_i from the PDM component scores, using the progress-weighted aggregation in Eq. 6. All G candidates can contribute to trajectory learning.

3.2 Video Rendering

We first check candidate execution validity, motion-conditioning support, and safety, then rank eligible candidates by R_i. World Engine renders the highest-reward candidates within the rendering budget K. These checks determine eligibility for video supervision and do not remove candidates from trajectory learning. For each selected candidate, the simulator $\mathcal{S}$ renders a target video along its executed path, and the video model W_ϕ predicts the future along the same path:

$$V_i^{\text{sim}} = \mathcal{S}(o, \tilde{\tau}_i), \qquad V_i^{\text{pred}} = W_\phi(h_\theta(o), \tilde{\tau}_i). \tag{2}$$

Here W_ϕ denotes the video prediction process. We render the reconstructed scene from camera poses along the executed path, as shown in Figure 2(d). The same path conditions video prediction, so both the rendered target and the prediction correspond to the same ego motion.

Rendering during training allows visual supervision to follow changes in the policy. A fixed offline cache contains videos only for trajectories sampled before optimization, whereas policy updates continuously change the trajectory distribution explored during post-training. The stored videos remain valid for their original paths, but do not provide matching targets for new paths with different motions or viewpoints. In each sampling round, SimVTA renders videos for selected candidates from the current policy, maintaining trajectory–video correspondence as training proceeds.

3.3 Video Supervision

Video Loss. Let $z_{i,r}^{\text{sim}}$ be the image latent of future frame r in V_i^{sim}, distinct from the shared representation F. Flow matching (Lipman et al., 2023; Liu et al., 2023) interpolates between Gaussian noise and the target latent:

$$x_{i,r,t} = t z_{i,r}^{\text{sim}} + (1-t)\epsilon, \qquad u_{i,r}^{\star} = z_{i,r}^{\text{sim}} - \epsilon, \quad t \sim \mathcal{U}(0,1),\ \epsilon \sim \mathcal{N}(0, I). \tag{3}$$

The velocity network D_ϕ conditions on history features $c_{\theta,i,r}$ and an ego-motion embedding $p_{\theta,i,r}$. During training, preceding rendered frames supply the history context; the clean current target is excluded. This teacher-forcing construction is given in Appendix A.2. The video loss is

$$\mathcal{L}_{\text{video}}(i) = \mathbb{E}_{t,\epsilon,r}\left[\frac{1}{d_z}\left\| D_\phi(x_{i,r,t}, t; c_{\theta,i,r}, p_{\theta,i,r}) - \text{sg}(u_{i,r}^{\star})\right\|_2^2\right], \tag{4}$$

Table 1: **Planning results on NAVSIM v1 NavTest.** UniAD results are from Liao et al. (2025); DriveVA also uses CARLA training data. Scores for the VLM-based baselines are taken from Zhao et al. (2026). Epona is adapted to NAVSIM following the official recipe and re-evaluated using the same protocol as SIMVTA.

Method	NC ↑	DAC ↑	TTC ↑	Comfort ↑	EP ↑	PDMS ↑
Traditional E2E planners						
UniAD (Hu et al., 2023)	97.8	91.9	92.9	100.0	78.8	83.4
DiffusionDrive (Liao et al., 2025)	98.2	96.2	94.7	100.0	82.2	88.1
WoTE (Li et al., 2025c)	98.5	96.8	94.9	99.9	81.9	88.3
MeanFuser + BeyondDrive (Wang et al., 2026a)	98.5	98.4	95.1	100.0	84.2	90.3
ResAD (Zheng et al., 2026)	98.9	97.8	94.9	100.0	87.0	90.6
VLM-based planners						
AutoVLA (Zhou et al., 2025)	98.4	95.6	98.0	99.9	81.9	89.1
Uni-World VLA (Liu et al., 2026c)	98.7	96.7	96.1	100.0	83.2	89.4
ExploreVLA (Sheng et al., 2026)	98.8	98.4	96.5	99.9	83.5	90.4
ReCogDrive (Li et al., 2026c)	97.9	97.3	94.9	100.0	87.3	90.8
SGDrive (Li et al., 2026a)	98.6	97.8	96.2	100.0	85.8	91.1
World-model-based planners						
LAW (Li et al., 2025a)	96.4	95.4	88.7	99.9	81.7	84.6
EPONA (Zhang et al., 2025b)	97.8	95.0	93.9	99.9	80.3	86.2
PWM (Zhao et al., 2025)	98.6	95.9	95.4	100.0	81.8	88.1
DreamerAD (Yang et al., 2026)	98.0	97.2	94.3	100.0	83.1	88.7
DriveWAM (Shi et al., 2026)	98.3	98.1	95.2	100.0	84.3	90.1
DriveVA (Liu et al., 2026b)	99.2	97.5	98.7	100.0	83.5	90.9
SIMVTA	98.0	97.2	97.0	100.0	86.7	**91.3**

where d_z is the number of elements in a frame latent and sg blocks gradients. The expectation averages over frames, flow times, and noise samples. The loss trains video prediction and the shared representations used for planning.

Target Selection. We first select candidates for rendering by trajectory reward, then select rendered videos for supervision by prediction error (Figure 2(c)). For each valid rendered candidate, we compute the teacher-forced latent prediction error E_i^{TF} in Eq. 9, which combines frame reconstruction and temporal-change errors. Within each scene, we retain candidates whose error is at or above the mean over valid rendered candidates. This subset, denoted Q2, contains high-reward, high-error candidates. Their matched videos serve as targets for the flow-matching loss in Eq. 4, focusing supervision on visual consequences that the model predicts less accurately.

3.4 TRAINING AND INFERENCE

We use SDE trajectory sampling following Flow-GRPO (Liu et al., 2025) and group-relative advantages following GRPO (Shao et al., 2024). The trajectory loss $\mathcal{L}_{\mathrm{traj}}$ weights clipped transition scores by these advantages and applies a discount across denoising steps. Video prediction error does not enter the trajectory advantages. Our overall training objective combines the trajectory loss, the video loss on selected targets $\mathcal{L}_{\mathrm{video}}^{\mathrm{sel}}$, and the BC regularizer, as defined in Eq. 5:

$$\mathcal{L} = \mathcal{L}_{\mathrm{traj}} + \lambda_v \mathcal{L}_{\mathrm{video}}^{\mathrm{sel}} + \lambda_{\mathrm{BC}} \mathcal{L}_{\mathrm{anchor}}, \tag{5}$$

where $\mathcal{L}_{\mathrm{anchor}}$ is the BC regularizer. The trajectory loss favors candidates with higher relative driving rewards, while the BC regularizer trains the model to follow denoising chains generated by a teacher planner (Figure 2(b)). We use the frozen initial trajectory network (TrajDiT) as the teacher. The teacher and student use the current shared representation, with gradients stopped on the teacher side. The weights λ_v and λ_{BC} control video supervision and BC regularization. Training repeats sampling, execution, rendering, and model updates. At inference, the model predicts trajectories directly from observation history, without simulator calls, video generation, or video-based reranking.

Table 2: **Planning results on NAVSIM v2 NavTest.** Epona is re-evaluated using the same protocol as SIMVTA. SGDrive's scores are taken from Zhao et al. (2026).

Method	NC ↑	DAC ↑	DDC ↑	TLC ↑	EP ↑	TTC ↑	LK ↑	HC ↑	EC ↑	EPDMS ↑
Traditional E2E planners										
ResAD (Zheng et al., 2026)	97.8	97.2	99.5	99.8	88.2	96.9	97.0	98.4	88.2	85.5
DriveSuprim (Yao et al., 2026)	98.4	98.6	99.6	99.8	90.5	97.8	97.0	98.3	78.6	87.1
DiffRefiner (Yin et al., 2026)	98.6	98.5	99.7	99.9	87.6	98.1	98.1	98.3	87.9	87.4
VLM-based planners										
SGDrive (Li et al., 2026a)	98.6	94.3	99.5	99.9	86.0	97.9	96.1	98.3	85.9	86.2
DriveWorld-VLA (Jia et al., 2026)	98.6	99.1	99.6	99.8	87.4	97.9	97.0	97.8	78.6	86.8
ELF-VLA-8B (Luo et al., 2026)	98.9	98.1	99.4	99.8	88.5	98.4	96.9	98.3	87.2	87.1
World-model-based planners										
EPONA (Zhang et al., 2025b)	97.2	95.6	99.3	99.7	88.4	96.4	96.9	97.9	67.6	85.0
SafeDrive (Kim et al., 2026)	99.5	99.0	99.7	99.9	88.6	98.9	97.5	98.2	81.9	87.5
DreamerAD (Yang et al., 2026)	98.0	97.2	99.5	99.8	87.8	97.4	97.5	98.3	72.4	87.7
SIMVTA	98.5	98.5	99.8	99.0	90.8	98.5	97.5	99.0	60.0	**88.2**

Table 3: Planning results on HUGSIM. E/M/H/X denote easy, medium, hard, and extreme scenarios. Baseline results are taken from Driving on Registers, Table 2. Higher values are better for all metrics.

	RC ↑					HD-Score ↑				
Method	E	M	H	X	Avg.	E	M	H	X	Avg.
VAD	51.3	31.1	25.3	26.5	31.4	36.3	9.5	8.0	11.5	13.4
LTF	67.8	35.1	26.2	40.5	38.9	58.9	18.0	9.8	25.9	23.7
UniAD	78.4	60.5	33.6	17.8	45.9	64.9	45.8	20.6	6.6	32.7
DrivoR	80.9	50.5	33.8	47.1	49.8	73.3	34.6	18.8	32.5	35.7
SIMVTA	78.8	56.1	38.5	39.6	**50.9**	69.7	42.3	22.9	25.7	**37.3**

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Implementation Details. We initialize SIMVTA from an Epona checkpoint adapted to NAVSIM as described in Appendix A.1, and perform post-training on NavTrain samples with available World Engine scene reconstructions. We use the 3DGS-based simulator in World Engine (Li et al., 2026b) to render matched future videos. In SIMVTA (Exp2), the video flow-matching loss uses Q2 targets and updates both the video network (VideoDiT) and the shared encoder. All ablations retain trajectory GRPO and the BC regularizer, with trajectory GRPO updating the shared encoder throughout. The mixed high/low-reward variants, Exp4 and Exp6, additionally apply the safety-specific advantage treatment described in Appendix A.4.

Benchmarks and Metrics. We evaluate SIMVTA on the NavTest splits of NAVSIM v1 and v2 (Dauner et al., 2024; Cao et al., 2025), using PDMS and EPDMS, respectively, to assess safety, progress, and comfort. On HUGSIM (Zhou et al., 2024), we report route completion (RC) and HD-Score, which combines RC with safety and comfort. Training uses the reward in Eq. 6; Benchmark scores follow their respective evaluation protocols. We report results averaged over three independent runs with different random seeds. Metric details are provided in Appendix A.6.

4.2 MAIN RESULTS

NAVSIM. Starting from the NAVSIM-adapted Epona initialization at 86.2 PDMS, SIMVTA reaches 91.3 on NAVSIM v1, a gain of 5.1 points (Figure 3a; Table 1). Ego progress rises from 80.3 to 86.7, time-to-collision compliance from 93.9 to 97.0, and drivable-area compliance from 95.0 to 97.2. On NAVSIM v2, SIMVTA improves EPDMS from 85.0 to 88.2, a gain of 3.2 points, and achieves 90.8 ego progress and 98.5 for both drivable-area and time-to-collision compliance (Table 2).

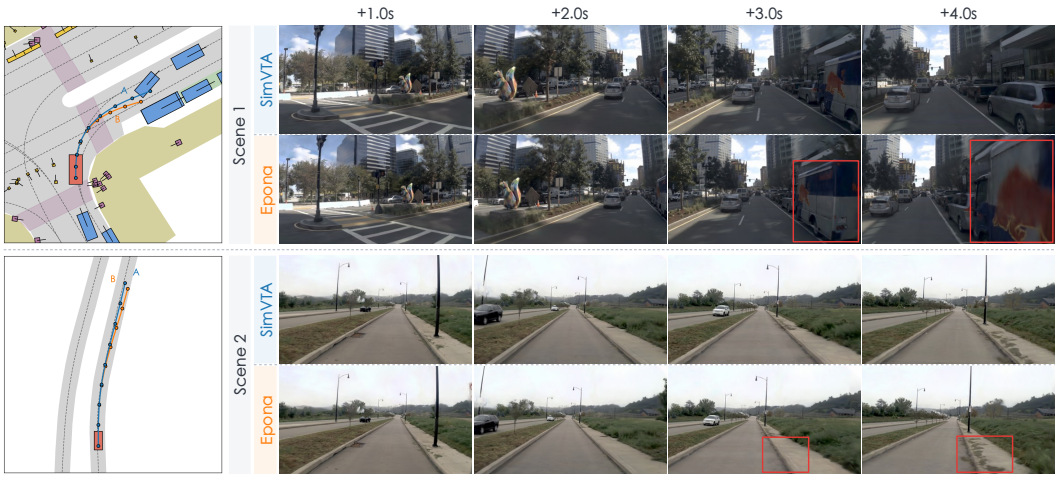


Figure 4: **Qualitative planning comparison.** Left: planned trajectories of SIMVTA (A, blue) and Epona (B, orange). Right: World Engine renderings along the corresponding paths at 1–4 s. Red boxes highlight Epona's collision and road-departure regions.

SIMVTA also improves planning in scenes where the Epona initialization fails (Figure 4). In these examples, Epona's trajectories lead to a collision or leave the drivable area, whereas SIMVTA avoids the collision and stays within the drivable area.

HUGSIM. SIMVTA achieves the highest average RC (50.9) and HD-Score (37.3) among the compared methods, exceeding the reported DrivoR results by 1.1 and 1.6 points, respectively (Table 3). It also leads on both metrics in hard scenarios, reaching 38.5 RC and 22.9 HD-Score.

Video–Trajectory Alignment. Beyond the planning gains, we compare SIMVTA with Epona to examine whether post-training improves the consistency of predicted videos with the scene along a trajectory (Figure 6). Epona generates an additional vehicle on the right (highlighted by red boxes) that is absent from both the bird's-eye view (BEV) and the corresponding World Engine renderings. SIMVTA avoids this inconsistency in the illustrated scene. This example provides qualitative evidence, alongside the PDMS and EPDMS gains, that SimVTA better predicts the visual consequences of the shown trajectory, supporting matched video supervision.

Video Quality. We evaluate both World Engine renderings and autoregressive model predictions against real videos on NAVSIM v1 (Table 4). Lower Fréchet Inception Distance (FID) and Fréchet Video Distance (FVD) indicate better video quality. Along logged GT trajectories, World Engine achieves an FID of 7.6 and an FVD of 21.8, indicating good rendering fidelity at logged viewpoints. Compared with trajectory GRPO alone (Exp0), SIMVTA reduces FID from 20.4 to 13.1 and FVD from 105.4 to 46.3, showing that video supervision also improves video prediction quality.

Table 4: **Video quality.**

Source	FID ↓	FVD ↓
3DGS renderings		
World Engine	7.6	21.8
WAM predictions		
Trajectory GRPO only (Exp0)	20.4	105.4
SimVTA (Exp2)	13.1	46.3

4.3 ABLATION STUDIES

Updating the Shared Representations. Video supervision improves planning when its gradients reach the shared encoder. Restricting video gradients to VideoDiT (Exp1) yields 89.1 PDMS, matching trajectory GRPO alone (Exp0; Figure 3a). Allowing video gradients to also update the shared encoder (Exp2) raises PDMS to 91.3, a gain of 2.2 points (Table 5).



Figure 5: **Matched futures for alternative actions.** Left: shared observation history and executed paths for going straight (A), changing lanes to the left (B), and changing lanes to the right (C). Right: corresponding future frames rendered by the simulator. Colors match the paths and image rows.

Selecting Informative Futures. We compare video supervision across four candidate groups: Q1 (high reward, low error), Q2 (high reward, high error), Q3 (low reward, low error), and Q4 (low reward, high error). High- and low-reward candidates come from the top and bottom of the trajectory reward ranking, respectively. Within each scene, errors below the mean over valid rendered candidates are low; those at or above it are high.

Q2 supervision yields the highest PDMS among the four settings. Exp2 uses Q2 supervision and achieves 91.3 PDMS. Exp5 uses the same high-reward rendering pool but expands video supervision to Q1+Q2, yielding 91.0 PDMS. Correcting larger prediction errors on high-reward trajectories may provide more useful updates to the shared representations. Exp4 and Exp6 use mixed high/low-reward pools with Q2+Q3 and Q1–Q4 supervision, respectively, both achieving 90.9 PDMS (Table 6). These two variants also apply the safety-specific advantage treatment described in Appendix A.4.

Table 5: **Ablation on shared encoder updates.**

Setting	Video loss updates	NC ↑	PDMS ↑
Exp0	None	97.1	89.1
Exp1	VideoDiT	97.6	89.1
Exp2	VideoDiT + shared encoder	98.0	91.3

Table 6: **Ablation on video target selection.**

Setting	Targets	NC ↑	PDMS ↑
Exp4	Q2 + Q3	97.4	90.9
Exp6	Q1 + Q2 + Q3 + Q4	97.7	90.9
Exp5	Q1 + Q2	98.0	91.0
Exp2	Q2	98.0	91.3

Table 7: **Video-training variants.**

Setting	Video supervision	NC ↑	PDMS ↑
Exp8	All valid rendered + video GRPO	97.9	91.0
Exp2	Q2 + flow matching	98.0	91.3

Table 8: **Online versus offline rendering.**

Setting	Rendering strategy	PDMS ↑
Exp0	No video supervision	89.1
Exp9	Fixed offline cache	90.4
Exp2	Online rendering	91.3

Comparing Video-Training Variants. Under the same rendering budget of four high-reward candidates, Exp8 applies video GRPO to all valid rendered candidates and obtains 91.0 PDMS. In contrast, Exp2 first selects the Q2 subset according to E_i^{TF} and then optimizes the selected targets with the flow-matching loss, reaching 91.3 PDMS (Table 7). Both variants allow video gradients to update the shared encoder. This comparison suggests that, under the four-candidate setting used here, selective flow-matching supervision is more effective than group-relative video optimization over the full rendered set.

Comparing Online and Offline Rendering. We compare online rendering with a fixed offline cache (Table 8). Exp9 renders trajectories sampled from the initial policy and reuses the matched trajectory–video pairs throughout post-training, using the same training objectives as Exp2. Exp2

instead refreshes video targets for current-policy trajectories. Exp9 achieves 90.4 PDMS, compared with 89.1 for trajectory-only Exp0 and 91.3 for Exp2. These results support refreshing video supervision as the policy changes.

Preserving Trajectory–Video Correspondence. Shuffling video targets within the same scene breaks alignment and reduces PDMS from 91.3 (Exp2) to 83.1 (Exp3) (Table 9). Figure 5 shows that each target video is rendered along its corresponding path.

Increasing the Rendering Budget. Doubling the rendering budget provides no PDMS gain: Exp7 achieves 91.2 versus 91.3 for Exp2 with the same target selection (Figure 3c; Table 10).

Table 9: **Trajectory–video correspondence.**

Setting	Video targets	NC ↑	PDMS ↑
Exp3	Shuffled	94.1	83.1
Exp2	Matched	98.0	91.3

Table 10: **Rendering budget.**

Setting	Targets	Budget	NC ↑	PDMS ↑
Exp7	Q2	2×	97.8	91.2
Exp2	Q2	1×	98.0	91.3



Figure 6: **Video–trajectory alignment before and after post-training.** Left: observation history and trajectory in BEV. Right: World Engine renderings and predictions from SIMVTA and Epona.

5 CONCLUSION

We presented SIMVTA, a simulation-grounded post-training framework that combines trajectory GRPO with video supervision. Rendering futures along current-policy trajectories provides visual targets beyond logged action–future pairs, allowing video prediction to train shared representations alongside trajectory optimization. The full framework improves NAVSIM v1 PDMS from 86.2 to 91.3, a gain of 5.1 points, and achieves strong planning results on NAVSIM v2 and HUGSIM. Ablations show a clear benefit from allowing video supervision to update shared representations and suggest an additional benefit from prioritizing higher-error futures among high-reward trajectories.

Limitations and directions for future work are discussed in Appendix A.7.

AI USE STATEMENT

Generative AI tools were used to assist with language editing and polishing of the manuscript text. All research ideas, experimental design, implementation, and analysis were carried out by the authors, who have reviewed all AI-assisted output and take full responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Appendices A.1–A.3 specify the training configuration, rewards, candidate selection, and losses. Appendix A.4 describes the ablation settings, and Appendices A.5–A.6 describe the video evaluation and benchmark protocols.

REFERENCES

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.

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

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *CoRL*, 2017.

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 *NeurIPS*, 2024.

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.

Feiyang Jia, Lin Liu, Ziying Song, Caiyan Jia, Hangjun Ye, Xiaoshuai Hao, and Long Chen. DriveWorld-VLA: Unified latent-space world modeling with vision-language-action for autonomous driving. In *ICML*, 2026.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *TOG*, 2023.

Jungho Kim, Jiyong Oh, Seunghoon Yu, Hongjae Shin, Donghyuk Kwak, and Jun Won Choi. SafeDrive: Fine-grained safety reasoning for end-to-end driving in a sparse world. In *CVPR*, 2026.

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

Jingyu Li, Junjie Wu, Dongnan Hu, Xiangkai Huang, Bin Sun, Zhihui Hao, Xianpeng Lang, Xiatian Zhu, and Li Zhang. SGDrive: Scene-to-goal hierarchical world cognition for autonomous driving. In *CVPR*, 2026a.

Quanyi Li, Zhenghao Peng, Lan Feng, Qihang Zhang, Zhenghai Xue, and Bolei Zhou. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. *TPAMI*, 2022.

Quanyi Li, Zhenghao Peng, Lan Feng, Zhizheng Liu, Chenda Duan, Wenjie Mo, and Bolei Zhou. ScenarioNet: Open-source platform for large-scale traffic scenario simulation and modeling. In *NeurIPS*, 2023.

Tianyu Li, Li Chen, Caojun Wang, Haochen Liu, Kashyap Chitta, Zhenjie Yang, Yuhang Lu, Naisheng Ye, Yihang Qiu, Haochen Tian, Yufei Wang, et al. World engine: Towards the era of post-training for autonomous driving. *arXiv preprint arXiv:2606.19836*, 2026b.

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*, 2025a.

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

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. *arXiv preprint arXiv:2504.01941*, 2025c.

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*, 2026c.

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.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for generative modeling. In *ICLR*, 2023.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-GRPO: Training Flow Matching models via online RL. In *NeurIPS*, 2025.

Mengmeng Liu, Diankun Zhang, Jiuming Liu, Jianfeng Cui, Hongwei Xie, Guang Chen, Hangjun Ye, Francesco Nex, Hao Cheng, and Michael Ying Yang. UNIVERSE: Unified video action models for autonomous driving with flexible mask-modulated modality generation. *arXiv preprint arXiv:2607.05133*, 2026a.

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. *arXiv preprint arXiv:2604.04198*, 2026b.

Qiqi Liu, Huan Xu, Jingyu Li, Bin Sun, Zhihui Hao, Dangen She, Xiatian Zhu, and Li Zhang. Uni-World VLA: Interleaved world modeling and planning for autonomous driving. *arXiv preprint arXiv:2603.27287*, 2026c.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with Rectified Flow. In *ICLR*, 2023.

Yuechen Luo, Fang Li, Qimao Chen, Shaoqing Xu, Jiaxin Liu, Ziying Song, Zhi-xin Yang, and Fuxi Wen. Unleashing VLA potentials in autonomous driving via explicit learning from failures. In *CVPR*, 2026.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Zihao Sheng, Xin Ye, Jingru Luo, Sikai Chen, and Liu Ren. ExploreVLA: Dense world modeling and exploration for end-to-end autonomous driving. In *ECCV*, 2026.

Chen Shi, Jinrui Xu, Shaoshuai Shi, Kehua Sheng, Bo Zhang, and Li Jiang. DriveWAM: Video generative priors enable scalable world-action modeling for autonomous driving. *arXiv preprint arXiv:2605.28544*, 2026.

Xiaolong Tang, Meina Kan, Shiguang Shan, and Xilin Chen. Plan-R1: Safe and feasible trajectory planning as language modeling. In *ICLR*, 2026.

Haochen Tian, Tianyu Li, Haochen Liu, Jiazhi Yang, Yihang Qiu, Guang Li, Junli Wang, Yinfeng Gao, Zhang Zhang, Liang Wang, Hangjun Ye, Long Chen, and Hongyang Li. Simscale: Learning to drive via real-world simulation at scale. In *CVPR*, 2026.

Hang Wang, Xin Ye, Feng Tao, Chenbin Pan, Abhirup Mallik, Burhaneddin Yaman, Liu Ren, and Junshan Zhang. Adawm: Adaptive world model based planning for autonomous driving. In *ICLR*, 2025.

Junli Wang, Zhihua Hua, Xueyi Liu, Zebin Xing, Haochen Tian, Kun Ma, Hangjun Ye, Guang Chen, Long Chen, and Qichao Zhang. Beyond imitation: Learning safe end-to-end autonomous driving from hard negatives. *ECCV*, 2026a.

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*, 2026b.

Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. Drivedreamer: Towards real-world-drive world models for autonomous driving. In *ECCV*, 2024.

Xinlin Wang, Yujiao Xiang, Yuheng Zhou, Jingqi Wang, Minqing Huang, Jiajie Huang, Dongxu Wei, Tingguang Zhou, Xiyang Wang, Gong Chen, Zhi Xu, Feiyang Tan, Hangning Zhou, and Mu Yang. WA-JEPA: Rethinking the video JEPA paradigm for world-action modeling in autonomous driving. *arXiv preprint arXiv:2608.20974*, 2026c.

Tianyi Yan, Huan Zheng, Dubing Chen, Meizhi Qu, Yingying Shen, Lijun Zhou, Mingfei Tu, Bing Wang, Guang Chen, Hangjun Ye, Haiyang Sun, Cheng zhong Xu, and Jianbing Shen. CausalDrive: Real-time causal world models for autonomous driving. In *ECCV*, 2026.

Jiazhi Yang, Kashyap Chitta, Shenyuan Gao, Long Chen, Yuqian Shao, Xiaosong Jia, Hongyang Li, Andreas Geiger, Xiangyu Yue, and Li Chen. Resim: Reliable world simulation for autonomous driving. In *NeurIPS*, 2025.

Pengxuan Yang, Yupeng Zheng, Deheng Qian, Zebin Xing, Qichao Zhang, Linbo Wang, Yichen Zhang, Shaoyu Guo, Zhongpu Xia, Qiang Chen, Junyu Han, Lingyun Xu, Yifeng Pan, and Dongbin Zhao. DreamerAD: Efficient reinforcement learning via latent world model for autonomous driving. In *ECCV*, 2026.

Wenhao Yao, Zhenxin Li, Shiyi Lan, Zi Wang, Xinglong Sun, Jose M Alvarez, and Zuxuan Wu. Drivesuprim: Towards precise trajectory selection for end-to-end planning. In *AAAI*, 2026.

Liuhan Yin, Runkun Ju, Guodong Guo, and Erkang Cheng. DiffRefiner: Coarse to fine trajectory planning via diffusion refinement with semantic interaction for end to end autonomous driving. In *AAAI*, 2026.

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.

Dongkun Zhang, Jiaming Liang, Ke Guo, Sha Lu, Qi Wang, Rong Xiong, Zhenwei Miao, and Yue Wang. CarPlanner: Consistent auto-regressive trajectory planning for large-scale reinforcement learning in autonomous driving. In *CVPR*, 2025a.

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*, 2025b.

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

Zongchuang Zhao, Xin Zhou, Tianyang Xu, Zhengyang Sun, Kaixuan Zhou, Yu Wu, Honglin Li, Dingkang Liang, and Xiang Bai. SimWAM: A simple world action model for end-to-end autonomous driving. *arXiv preprint arXiv:2608.07468*, 2026.

Zhiyu Zheng, Shaoyu Chen, Haoran Yin, Xinbang Zhang, Jialv Zou, Xinggang Wang, Qian Zhang, and Lefei Zhang. ResAD: Normalized residual trajectory modeling for end-to-end autonomous driving. In *CVPR*, 2026.

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

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.

A Appendix

A.1 Training Procedure

Before applying SimVTA, we fine-tune the nuPlan-pretrained Epona checkpoint on NAVSIM following the official adaptation recipe to adapt it from 10 Hz to 2 Hz. The adapted checkpoint initializes both Exp0 and Exp2. We evaluate it using the same NAVSIM v1 protocol as the post-trained models and obtain 86.2 PDMS, matching the overall score reported for Epona. Unless stated otherwise, the implementation details below describe Exp2.

Training Loop.

1. Encode observation histories and sample G trajectories, retaining their normalized denoising states.
2. Execute each candidate with PDM, compute its training reward, and normalize rewards within each observation group.
3. Check execution validity, pose support, and safety. Rank eligible candidates by reward and select the rendering pool.
4. Render the selected paths and encode their future frames, retaining the candidate-to-video correspondence.
5. Compute detached teacher-forced errors and classify the valid rendered candidates into quadrants.
6. Select the video-training subset and future frame for this update. Generate the detached BC teacher chain from the current shared representation.
7. Combine the trajectory loss, selected video loss, and BC regularizer. Update the trainable parameters and repeat with the updated policy.

Architecture and Temporal Settings. Epona's spatiotemporal Transformer is the shared encoder, paired with TrajDiT and VideoDiT. Four observed frames condition eight trajectory waypoints spanning four seconds. PDM evaluates the path at 10 Hz; video pose conditions are sampled at 2 Hz. Exp2 evaluates teacher-forced errors on the first two future frames and trains one of these frames per update. This one-second video-training horizon is distinct from the four-second planning and video-evaluation horizons. The shared encoder and TrajDiT receive trajectory and BC gradients; VideoDiT and the shared encoder receive the selected video gradients.

Trajectory Reward. Each candidate is scored together with the cached PDM reference trajectory. The training reward is

$$R_i = \mathrm{NC}_i \, \mathrm{DAC}_i \frac{10\,\mathrm{EP}_i + 5\,\mathrm{TTC}_i + 2\,\mathrm{Comfort}_i}{17}. \tag{6}$$

Following ReCogDrive (Li et al., 2026c), we use weights of 10, 5, and 2 for EP, TTC, and Comfort, respectively, with the higher EP weight encouraging efficient driving. The component scores lie in $[0, 1]$. Progress follows the PDM scorer's normalization over the candidate and reference trajectories. Driving-direction compliance is used for candidate screening but has zero weight in this reward. Exp2 excludes video prediction error from trajectory rewards.

Candidate Validity and Safety. An eligible video candidate must execute successfully and have finite pose changes within VideoDiT's support at every 2 Hz step: $0 \le \Delta x \le 16$ m, $|\Delta y| \le 1.5$ m, and $|\Delta\mathrm{yaw}| \le 32^\circ$. Its maximum reverse speed must not exceed 10^{-3} m/s. For Exp2, the NC, DAC, TTC, and DDC scores must each be at least 0.999. We rank eligible candidates by R_i, break ties by sampling index, and select the first K. If fewer than K qualify, only those candidates are rendered. Failed renders are excluded from video supervision. A scene skips video supervision if all candidates fail the safety checks or its maximum reward is at most 10^{-6}. These filters do not remove candidates from the Exp2 trajectory loss.

Table 11: Exp2 video-supervision settings and default optimization parameters.

Parameter	Value
Sampled trajectories / rendering budget	$G = 32$ / $K = 4$
Batch size per GPU / default random seed	4 / 1234
Training hardware	16 NVIDIA H20 GPUs
Initial learning rate / AdamW betas	10^{-5} / $(0.9, 0.95)$
Weight decay / LR decay factor	0.01 / 0.5
LR decay steps	300, 1500
Trajectory denoising steps / noise scale	10 / 0.3
Diffusion-coefficient floor / denoising discount	0.01 / 0.6
Video loss / BC weights	$\lambda_v = 0.1$ / $\lambda_{\mathrm{BC}} = 0.1$
Video gradient scale into shared parameters	$\eta = 0.2$
Video-error horizon / temporal discount	$H = 2$ / $\gamma_{\mathrm{v}} = 0.85$
Video-training candidates per scene per update	All valid Q2
Trained future frames per candidate per update	1
Mixed precision / distributed optimizer	bfloat16 / ZeRO-2, CPU offload

Optimization Configuration. Table 11 lists the video-supervision settings for Exp2 and the optimization defaults. Training uses AdamW, implemented by DeepSpeedCPUAdam in AdamW mode when optimizer offloading is enabled. Bias and normalization parameters are excluded from weight decay. The learning rate is halved at steps 300 and 1500. The shared encoder uses the same learning rate as the heads. When no video target participates across all devices, VideoDiT parameters remain unchanged for that update.

A.2 Conditional Video Prediction

The next-frame prediction head VideoDiT is named VisDiT in Epona. The process W_ϕ applies the velocity network D_ϕ and decodes the resulting image latents. Flow time t in Eq. 3 runs from noise at zero to the target at one and is distinct from frame index r. The factor $1/d_z$ in Eq. 4 averages the squared error over the elements of one frame latent.

Internal Motion Conditioning. For each selected candidate, World Engine renders the supervision target along its executed path. The WAM uses the same executed pose sequence for its internal video conditioning:

$$V_i^{\mathrm{sim}} = \mathcal{S}(o, \tilde{\tau}_i), \qquad V_i^{\mathrm{pred}} = W_\phi(h_\theta(o), \tilde{\tau}_i).$$

This pairing is used during post-training; the motion conditioning is internal to the WAM.

To predict frame r, we construct conditions from a rolling history window containing observed or preceding rendered frames and pose changes:

$$c_{\theta,i,r} = h_\theta(o, z^{\mathrm{sim}}_{i,<r}, \mathrm{sg}(\Delta\tilde{\tau}_{i,<r})), \qquad p_{\theta,i,r} = e_\theta(\mathrm{sg}(\Delta\tilde{\tau}_{i,r})). \tag{7}$$

Here $< r$ denotes preceding frames, $\Delta\tilde{\tau}_{i,r}$ is the executed pose change, and e_θ embeds that motion. The expression for h_θ denotes the rolling context rather than an ever-growing sequence. The clean current target is excluded from this context. Rendered latents and pose inputs are fixed; the shared encoder and motion embedding remain trainable.

Teacher-forced Prediction Error. We compute the prediction error at $t_\star = 0.5$ using one Gaussian tensor ϵ_o shared by all candidates and future frames of an observation:

$$\begin{aligned} x^\star_{i,r} &= t_\star z^{\mathrm{sim}}_{i,r} + (1 - t_\star)\epsilon_o, \\ \hat{z}_{i,r} &= x^\star_{i,r} + (1 - t_\star) D_\phi(x^\star_{i,r}, t_\star; c_{\theta,i,r}, p_{\theta,i,r}), \\ e_{i,r} &= \hat{z}_{i,r} - z^{\mathrm{sim}}_{i,r}. \end{aligned} \tag{8}$$

With $\alpha_r = \gamma_{\mathrm{v}}^{r-1} / \sum_{j=1}^{H} \gamma_{\mathrm{v}}^{j-1}$, we define

$$E_i^{\mathrm{TF}} = 0.7 \sum_{r=1}^{H} \alpha_r \frac{\|e_{i,r}\|_2^2}{d_z} + 0.3 \sum_{r=2}^{H} \frac{\alpha_r}{\sum_{j=2}^{H} \alpha_j} \frac{\|\sqrt{\omega_{i,r}} \odot (e_{i,r} - e_{i,r-1})\|_2^2}{d_z}. \tag{9}$$

For token p and channel c, the motion weight is

$$m_{i,r,p} = \mathrm{mean}_c \, |z^{\mathrm{sim}}_{i,r,p,c} - z^{\mathrm{sim}}_{i,r-1,p,c}|, \qquad \omega_{i,r,p} = \mathrm{clip}\left(\frac{m_{i,r,p}}{\max(\mathrm{mean}_{p'} \, m_{i,r,p'}, 10^{-6})}, 0.25, 4 \right). \tag{10}$$

The weight is broadcast over channels. We use $H = 2$ and $\gamma_{\mathrm{v}} = 0.85$. The complete error computation is detached. Unlike the video-training loss, this score uses a fixed flow time, shared evaluation noise, and a temporal-change term.

A.3 Video Supervision and Policy Optimization

Quadrant Assignment and Training Subset. Reward labels follow the ranked rendering pool: Exp2 uses the high-reward pool, while mixed-pool variants retain separate high- and low-reward labels. No reward-mean threshold is applied. The error threshold is the mean of E_i^{TF} over valid rendered candidates within each scene. An error below the mean is low; an error equal to or above it is high.

For each scene, Exp2 trains on all valid Q2 candidates at every update. At training step s, the trained future frame for each candidate is $1 + (s \bmod H)$. Preceding target frames still provide context when only a later frame is trained. Flow time and training noise are sampled independently for each selected latent. Candidate selection is detached.

Video-gradient Masks. Let m_i^{head} include candidate i in video training, and let $m_i^{\mathrm{shared}} \le m_i^{\mathrm{head}}$ additionally allow gradients into the shared parameters. We use $g_\eta(x) = \mathrm{sg}(x) + \eta(x - \mathrm{sg}(x))$, which preserves the forward value and scales its backward gradient by η. For the current update, we estimate Eq. 4 at the cyclic frame $r = 1 + (s \bmod H)$ using newly sampled flow time and noise. Write $\mathcal{L}^{\mathrm{joint}}_{\mathrm{video}}(i)$ for this estimate with $g_\eta(c_{\theta,i,r})$ and $g_\eta(p_{\theta,i,r})$ as conditions, and $\mathcal{L}^{\mathrm{head}}_{\mathrm{video}}(i)$ for the estimate with both conditions detached. Then

$$\mathcal{L}^{\mathrm{sel}}_{\mathrm{video}} = \frac{1}{Z} \sum_i \left[m_i^{\mathrm{shared}} \mathcal{L}^{\mathrm{joint}}_{\mathrm{video}}(i) + (m_i^{\mathrm{head}} - m_i^{\mathrm{shared}}) \mathcal{L}^{\mathrm{head}}_{\mathrm{video}}(i) \right], \tag{11}$$

where $Z = \max(1, \sum_i m_i^{\mathrm{head}})$ and the sum covers participating candidates across all devices. In Exp2, both masks select Q2 candidates and $\eta = 0.2$. This scale affects only video gradients entering the shared encoder and motion embedding; trajectory, BC, and VideoDiT gradients retain their own loss weights. The video loss is zero when no target participates.

Trajectory GRPO. We use group-relative advantages with a clipped transition-score objective. Let $a_i^{(k)}$ be the normalized trajectory state at step k, with N transitions and d_τ scalar coordinates. For a stored transition (a, a'), the sampler specifies a mean $\mu_{\theta,\psi,k}(a, o)$ and standard deviation σ_k. The implemented score is

$$\ell_{\theta,\psi,k}(a, a'; o) = \frac{1}{d_\tau} \sum_{d=1}^{d_\tau} \mathrm{clip}\left(-\frac{(a'_d - \mu_{\theta,\psi,k,d}(a, o))^2}{2\sigma_k^2}, -5, 2 \right). \tag{12}$$

This score omits the Gaussian normalizing term and is clipped before averaging over coordinates. For each observation, we compute

$$\begin{aligned}
\bar{R} &= \frac{1}{G}\sum_{j=1}^{G} R_j, \\
s_R^2 &= \frac{1}{G-1}\sum_{j=1}^{G}(R_j-\bar{R})^2, \\
A_i &= \frac{R_i-\bar{R}}{s_R+\varepsilon_{\text{adv}}}, \\
\mathcal{L}_{\text{traj}} &= -\mathbb{E}_o\left[\frac{1}{GN}\sum_{i=1}^{G}\sum_{k=0}^{N-1}\gamma_{\text{d}}^{N-1-k}\,\text{sg}(A_i)\ell_{\theta,\psi,k}(a_i^{(k)}, a_i^{(k+1)}; o)\right].
\end{aligned} \tag{13}$$

We use $\varepsilon_{\text{adv}} = 10^{-8}$ and $\gamma_{\text{d}} = 0.6$. Sampled states and advantages are detached; gradients through the transition mean update the shared encoder and TrajDiT. The trajectory objective uses no current-to-sampling-policy probability ratio.

SDE Transitions. Let $s_0 = 1$ and $s_N = 0$, with a decreasing grid between them. For transitions $k = 0, \ldots, N-1$, define $\Delta s_k = s_{k+1} - s_k < 0$. The trajectory velocity network v_ψ uses ascending flow time $1 - s_k$. Define

$$\begin{aligned}
\bar{s}_k &= \begin{cases} s_1, & k = 0, \\ s_k, & k > 0, \end{cases} \\
\beta_k &= \max\left(\beta_{\min}, \nu\sqrt{\frac{s_k}{1-\bar{s}_k}}\right), \\
\mu_{\theta,\psi,k}(a,o) &= a\left(1+\frac{\beta_k^2}{2s_k}\Delta s_k\right) \\
&\quad - v_\psi(a, 1-s_k; h_\theta(o))\left(1+\frac{\beta_k^2(1-s_k)}{2s_k}\right)\Delta s_k, \\
\sigma_k &= \beta_k\sqrt{-\Delta s_k}.
\end{aligned} \tag{14}$$

Starting from Gaussian noise, we sample $a^{(k+1)} = \mu_{\theta,\psi,k} + \sigma_k\,\text{clip}(\xi_k, -5, 5)$ with independent $\xi_k \sim \mathcal{N}(0, I)$. We use $\nu = 0.3$, $\beta_{\min} = 0.01$, and $N = 10$. The shifted time grid is $s_k = e^m u_k/(e^m u_k + 1 - u_k)$, where $u_k = 1 - k/N$ and $m = 0.5 + (1.15 - 0.5)(8 - 256)/(4096 - 256)$. Before denormalization and reward evaluation, final longitudinal, lateral, and yaw action coordinates are clipped to $[-1.7, 1.7]$, $[-2.6, 2.6]$, and $[-10, 10]$, respectively. These bounds are in normalized coordinates. The inverse mapping is $x = 25(a_x + 1)$ m, $y = 10a_y$ m, and $\text{yaw} = 12a_{\text{yaw}}$ degrees, with waypoints expressed relative to the current ego frame. Stored policy transitions retain the unclipped final state.

BC Regularizer. Let ψ_0 denote the frozen initial TrajDiT parameters. For each observation, one Gaussian initial state generates an ODE chain $b^{(0)}, \ldots, b^{(N)}$ using ψ_0 and $\text{sg}(h_\theta(o))$. The teacher follows $b^{(k+1)} = b^{(k)} - \Delta s_k v_{\psi_0}(b^{(k)}, 1 - s_k; \text{sg}(h_\theta(o)))$. Its final state uses the action-clipping bounds above. We optimize

$$\mathcal{L}_{\text{anchor}} = -\mathbb{E}_{o,b^{(0)}}\left[\frac{1}{N}\sum_{k=0}^{N-1}\ell_{\theta,\psi,k}(\text{sg}(b^{(k)}), \text{sg}(b^{(k+1)}); o)\right]. \tag{15}$$

The teacher parameters remain fixed, while teacher chains are regenerated from the current shared representation. Student-side gradients update the shared encoder and TrajDiT. The BC objective averages over steps without the trajectory loss's denoising discount.

A.4 Experimental Variants and Budget Accounting

All ablations use Exp2 as the reference setting. Paper experiment numbers differ from the implementation entry names: Exp5, Exp6, and Exp8 correspond to `exp8_high4_all`,

Table 12: Video-supervision configurations. K is the maximum number of rendered candidates per observation in each sampling round. For Exp9, this budget applies to offline cache construction. FM denotes flow matching.

Setting	K	Rendered pool	Video-training set	Video objective	PDMS
Exp0	0	None	None	None	89.1
Exp1	4	High reward	Q2	FM	89.1
Exp2	4	High reward	Q2	FM	91.3
Exp3	4	High reward	Q2, shuffled	FM	83.1
Exp4	4	2 high + 2 low	Q2 + Q3	FM	90.9
Exp5	4	High reward	Q1 + Q2	FM	91.0
Exp6	4	2 high + 2 low	Q1–Q4	FM	90.9
Exp7	8	High reward	Q2	FM	91.2
Exp8	4	High reward	Valid rendered candidates	Video GRPO	91.0
Exp9	4	Initial-policy cache	Q2 from cache	FM	90.4

Table 13: NAVSIM v1 ablation results. Configurations are listed in Table 12.

Setting	NC	DAC	TTC	Comfort	EP	PDMS
Exp0	97.1	96.7	96.0	99.8	83.6	89.1
Exp1	97.6	96.7	96.4	99.7	83.3	89.1
Exp2	98.0	97.2	97.0	100.0	86.7	91.3
Exp3	94.1	91.3	91.8	99.6	81.2	83.1
Exp4	97.4	97.3	96.4	99.9	86.4	90.9
Exp5	98.0	97.1	97.0	99.9	86.1	91.0
Exp6	97.7	97.2	96.6	99.9	86.5	90.9
Exp7	97.8	97.3	96.9	99.9	86.6	91.2
Exp8	97.9	97.2	96.9	99.9	86.2	91.0
Exp9	97.1	97.2	95.9	99.9	85.9	90.4

`exp11_high2_low2_all`, and `exp10_high4_video_grpo`, respectively. Table 12 summarizes the supervision configurations, and Table 13 reports their NAVSIM v1 results. Both tables use paper experiment numbers and include Exp0–Exp9.

Gradient Paths and Candidate Pools. Exp0 uses trajectory GRPO and BC without video supervision. In Exp1, video gradients update only VideoDiT; trajectory and BC gradients still update the shared encoder. Exp2 also allows video gradients to update shared parameters. Both use Q2 targets. Exp3 shuffles rendered targets across trajectory conditions within a scene after selection. Scenes without a valid target permutation contribute no shuffled video supervision.

Exp4 selects the two highest- and two lowest-reward candidates from the execution- and pose-valid pool. If this pool contains $n < 4$ candidates, the high and low subsets contain $\lceil n/2 \rceil$ and $\lfloor n/2 \rfloor$ candidates, respectively. The subsets are disjoint, with reward ties broken by sampling index. Exp4 applies the four safety checks before training on Q2 and Q3. Exp5 renders four safe high-reward candidates and trains on all valid Q1 and Q2 candidates. Exp6 uses the same high/low pool as Exp4 but trains on all valid rendered candidates, including individually unsafe candidates. In Exp4 and Exp6, an unsafe candidate's positive trajectory advantage is additionally truncated to zero: $A_i \leftarrow \min(A_i, 0)$. Exp2 does not apply this projection. Exp7 increases the rendering budget from four to eight while retaining Exp2's Q2 selection and video-training rule. Both settings train on all valid Q2 candidates. Table 12 distinguishes the rendered pool from the subset used for video training.

Fixed Offline Cache (Exp9). Exp9 follows the same experimental settings as Exp2, except that trajectories are sampled offline from the initial model and their rendered videos are stored in a fixed cache. The cached trajectory–video pairs are reused throughout post-training. All other settings, including candidate selection, rendering budget, training objectives, and optimization hyperparameters, are identical to Exp2. Exp9 achieves 90.4 PDMS, compared with 89.1 for Exp0 and 91.3 for Exp2 (Table 8).

Table 14: NAVSIM v1 PDMS across random seeds. The final column reports the mean and sample standard deviation over three runs.

Setting	Random seed			Mean ± SD
	1234	20260901	20260910	
Exp0	89.1	89.2	89.0	89.1 ± 0.1
Exp2	91.3	91.4	91.3	91.3 ± 0.1
Exp5	91.0	90.9	91.0	91.0 ± 0.1
Exp7	91.1	91.2	91.2	91.2 ± 0.1

Video GRPO Control. Exp8 evaluates video GRPO under the same four-candidate rendering budget used by Exp2. Exp8 applies video GRPO to all valid rendered high-reward candidates, whereas Exp2 first selects the Q2 subset and optimizes the selected targets with flow matching. For each candidate, it generates future latents with five stochastic Euler steps and noise scale 0.3, using preceding target frames as context. The reward is the negative framewise latent MSE, aggregated with normalized temporal weights proportional to 0.85^{r-1}. It does not include the motion-weighted temporal-change term in Eq. 9. Different trajectory conditions within the same scene form the video comparison group. The video policy objective uses group-normalized advantages and a clipped ratio computed from dimension-normalized Gaussian log densities, with clipping parameter 0.2. One future frame is replayed for optimization per update, cycling through the video horizon. Groups with fewer than two valid candidates or reward standard deviation below 10^{-6} contribute no video policy update. Video GRPO replaces video flow matching and updates VideoDiT and shared parameters; its reward does not enter trajectory advantages. Exp8 achieves 91.0 PDMS, compared with 91.3 for Exp2: video GRPO over all valid rendered candidates does not outperform flow matching on selected targets at the four-video rendering budget. This comparison evaluates the two training strategies, including their respective candidate subsets.

Rendering and Optimization Budgets. K specifies the number of candidates budgeted for rendering; Table 12 separately lists those used for video training. Error-based selection follows rendering and scoring, so these costs apply to every rendered candidate. Total training cost also includes trajectory sampling, PDM evaluation, video optimization, and optimizer updates. Comparisons at equal K match rendering caps, while video optimization costs depend on the training subset and objective. Exp9 renders its cache once; Exp2 renders candidates at each sampling round.

Interpreting the Ablations. Exp0 omits video supervision, Exp1 restricts video gradients to VideoDiT, and Exp2 also updates the shared encoder through the video loss. The full framework achieves 91.3 PDMS, compared with 89.1 for both Exp0 and Exp1 (Table 13). This 2.2-point gain supports the contribution of video supervision through the shared encoder within the joint training framework. Exp3 tests trajectory–video correspondence by shuffling targets within a scene; skipping scenes without a valid permutation can also reduce supervision coverage. Exp9 compares online rendering with a fixed, correctly paired offline cache. The remaining variants compare the supervision configurations in Table 12. In particular, Exp2 and Exp5 provide a comparison of Q2 versus Q1+Q2 supervision from the same rendering pool; the number of video-training candidates differs. In contrast, Exp4 and Exp6 additionally modify the safety-specific treatment of trajectory advantages and are therefore interpreted as broader configuration comparisons.

Results across Random Seeds. Table 14 reports NAVSIM v1 PDMS for Exp0, Exp2, Exp5, and Exp7 across three random seeds. Exp2 outperforms the other three variants on every seed.

A.5 Video Supervision and Evaluation

Teacher-forced error measures agreement with a rendered target given preceding target frames. Renderer fidelity, autoregressive video quality, and action-conditioning accuracy measure different properties.

Renderer Fidelity. We compare front-view World Engine renderings along logged GT trajectories with real observations on NAVSIM v1 NavTest. Each clip contains eight future frames over four seconds at 2 Hz. FID is computed on frames and FVD on clips using the same preprocessing and feature extractors for the two domains. These measurements quantify rendering fidelity at logged viewpoints; novel-view coverage is discussed in Appendix A.7.

Model Video Quality. For each evaluation sample, Exp0 and Exp2 generate autoregressive video predictions along the same logged GT trajectory used for the World Engine rendering. Both models use the same observation history. We use the same dataset, clip length, preprocessing, and feature extractors as the renderer evaluation. FID and FVD compare the predictions with the corresponding real observations. The eight-frame evaluation is distinct from the two-frame teacher-forced error used for training-target selection.

Qualitative Examples. Figure 5 shows a shared observation history, a bird's-eye view of three alternative executed paths (A–C), and World Engine renderings along each path at $+1$, $+2$, $+3$, and $+4$ seconds. Figure 6 compares World Engine renderings with predictions from SIMVTA and the Epona initialization at the same displayed future times.

A.6 CROSS-BENCHMARK RESULTS

Evaluation Metrics and Scope. PDMS combines collision and drivable-area terms with progress, TTC, and comfort. EPDMS additionally evaluates driving direction, traffic lights, lane keeping, and further comfort measures. On HUGSIM, HD-Score combines safety and comfort over the rollout and scales the result by RC. Higher scores are better. We evaluate SIMVTA and the NAVSIM-adapted Epona initialization on the full NAVSIM v2 NavTest set using the NAVSIM v2.2 devkit with single-stage EPDMS evaluation, LQR tracking, and non-reactive traffic agents.

Result Sources and Comparison Scope. Figure 7 summarizes the NAVSIM and HUGSIM results. On NAVSIM v1, the full framework improves PDMS by 5.1 points over the initialization (86.2 to 91.3). The comparison with Exp0 (89.1 to 91.3) measures the additional gain when video supervision is added within the framework. NAVSIM v2 and HUGSIM extend the evaluation to additional planning criteria and closed-loop driving. On NAVSIM v2, the NAVSIM-adapted Epona initialization obtains 85.0 EPDMS under the same evaluation protocol as SIMVTA, which reaches 88.2. The 3.2-point aggregate gain coexists with lower traffic-light compliance (99.7 to 99.0) and extended comfort (67.6 to 60.0) (Table 2). We evaluate SIMVTA on the same 345 pre-challenge HUGSIM scenarios using the same corrected evaluator and controller as Kirby et al. (2026). Baseline results in Table 3 are taken from that work. SIMVTA achieves the highest reported averages among these methods and leads on hard scenarios, while DrivoR has higher extreme-scenario RC and HD-Score (47.1/32.5 versus 39.6/25.7). These closed-loop results assess the final planner; isolating the post-training gain on HUGSIM additionally requires the initialization and Exp0 under the same routes, controller, and rollout settings.

Recent WAM Results. Table 15 includes results from recent WAMs discussed in the related work, including methods whose reports also supply baseline values in the main tables. These published results complement the within-backbone ablations; the table retains each source's checkpoint and evaluation convention.

A.7 LIMITATIONS

Reconstruction Quality and Coverage. Occlusions and limited pose coverage can degrade renderings away from logged paths, while renderer metrics assess only logged viewpoints. Safety and pose-support checks cannot exclude all artifacts, so high prediction error may indicate either model difficulty or unreliable targets. Quality-aware filtering and evaluation at alternative poses remain necessary.

Distribution Shifts. Current-policy trajectories broaden action coverage, but observation histories remain log-based and rendered targets can differ from real images. Reconstruction coverage and candidate filtering further limit supervision. Teacher-forced selection does not capture autoregressive

Table 15: Recent WAM results reported by their respective papers. Training data, initialization, and evaluators differ across methods.

Method	NAVSIM v1 PDMS	NAVSIM v2 EPDMS
SimWAM (Zhao et al., 2026)	91.5	90.2
Latent-WAM (Wang et al., 2026b)	–	89.3
WA-JEPA (Wang et al., 2026c)	91.8	91.7
SIMVTA	91.3	88.2

Sources: SimWAM, arXiv:2608.07468v4, Tables 1–2; Latent-WAM, arXiv:2603.24581v1, Table 1; WA-JEPA, arXiv:2608.20974v2, Tables 1 and 3. SimWAM's v1 and v2 results are from its post-RL and pre-RL checkpoints, respectively. A dash indicates no result in the cited source.

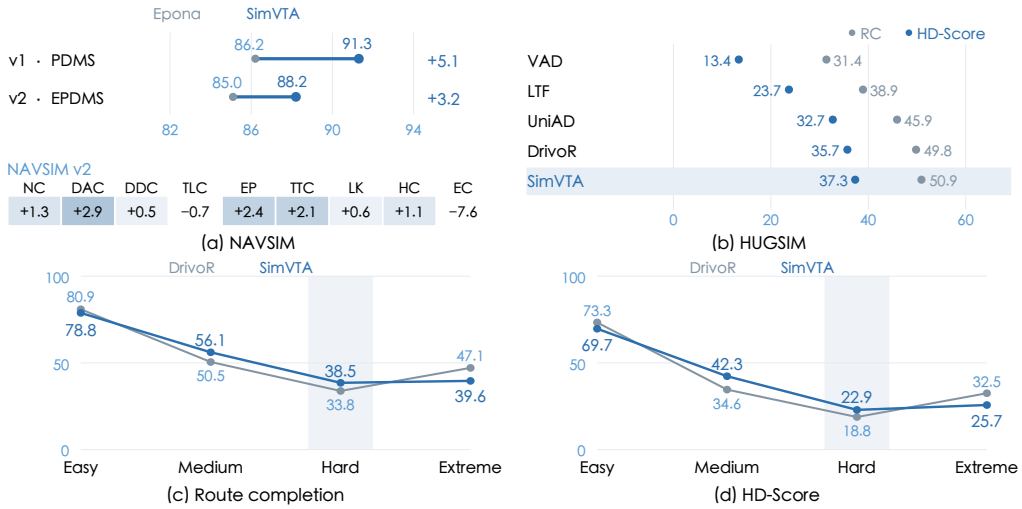


Figure 7: **Reported planning results across benchmarks.** (a) NAVSIM v1 PDMS and v2 EPDMS, with NAVSIM v2 component-score differences (SIMVTA minus the NAVSIM-adapted Epona initialization). (b) HUGSIM RC and HD-Score using benchmark aggregation. (c) RC and (d) HD-Score for DrivoR and SIMVTA by scenario difficulty. Values correspond to Tables 1, 2, and 3.

error accumulation, and replayed traffic may not react to ego actions. Robustness under shifted histories and reactive traffic remains untested.

Architecture and Training Cost. The framework is currently evaluated with Epona; transfer to other WAM architectures remains to be tested. Online rendering, error scoring, and video optimization increase training cost, while inference uses direct trajectory prediction. GPU time and effective supervision coverage require further assessment.