StreamWAM: Real-Time World-Action Modeling with Closed-Loop Streaming Diffusion

Anonymous authors
Paper under double-blind review

Abstract

World-action models (WAMs) have recently emerged as a promising paradigm for autonomous driving, unifying future video generation and trajectory planning in a single generative framework. However, conventional diffusion-based WAMs require multiple denoising steps to generate each action chunk, leading to high inference latency and limiting the frequency of closed-loop feedback. To address this, we propose **StreamWAM**, a world-action model capable of real-time driving action planning by closed-loop streaming diffusion. By overlapping the denoising processes of consecutive video-action chunks, the model is able to roll out an action chunk from each inference step, even though the complete diffusion process takes multiple steps. Furthermore, we propose a high-frequency closed-loop planning method to consider the most recent observation for the intermediate action. We employ a latent alignment module to continuously correct the denoised latent states given the new observations. Experiments on the NAVSIM benchmark show that StreamWAM achieves state-of-the-art planning performance with a PDMS of 91.6 and an EPDMS of 94.3. We demonstrate that high-frequency feedback improves planning performance while the model maintains a practical 10 Hz planning frequency. We believe StreamWAM represents a concrete step toward deploying world-action models in real autonomous driving systems.

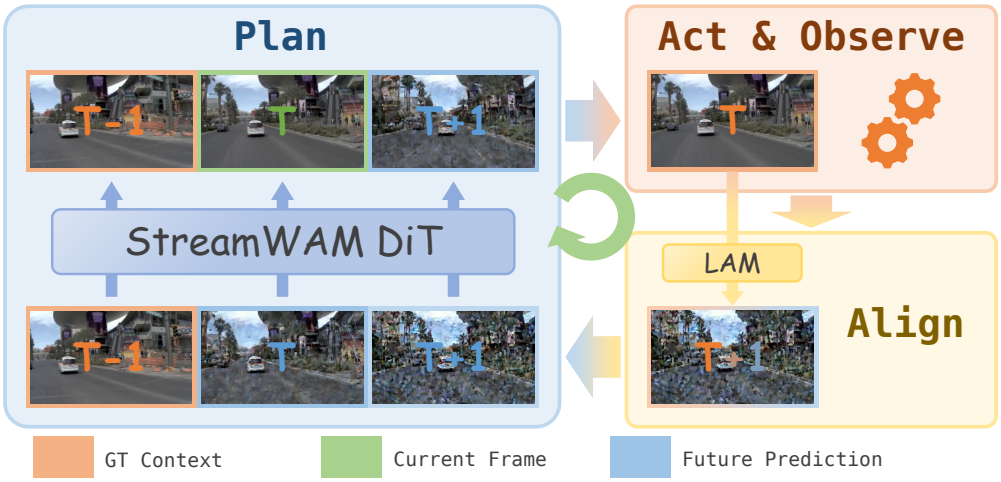

Figure 1: Overview of StreamWAM. The model follows a closed-loop driving process of planning with world-action model, acting and observing, then aligning with latent alignment module.

1 Introduction

Autonomous driving has advanced rapidly with end-to-end methods that learn planning from sensor observations (Hu et al., 2023; Chitta et al., 2022; Liao et al., 2025). Recent vision-language-action (VLA) models build on pretrained vision-language models (VLMs), using their semantic understanding and reasoning capabilities to connect visual observations, navigation instructions, and driving actions (Zhou et al., 2026b; Li et al., 2025c;b). World-action models (WAMs) further incorporate

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.

future scene evolution into planning by jointly predicting future observations and actions, or by inferring actions from predicted future states (Xia et al., 2026; Liu et al., 2026; Shi et al., 2026).

However, for many existing WAMs, strong benchmark performance has yet to translate into responsive closed-loop driving. Diffusion-based WAMs typically generate each video-action chunk through several sequential forward passes of a large video generator (Xia et al., 2026; Liu et al., 2026; Shi et al., 2026). Because an action becomes available only after this denoising process, the latency of a planning update grows with both model cost and the number of sampling steps. Predicting a longer action chunk can spread this cost across more actions, but it does not shorten the wait for a new plan. Subsequent actions may also continue to follow a future imagined from observations collected several steps earlier. As the vehicle moves, new observations can invalidate that imagined future, yet many chunk-wise pipelines do not incorporate them until the next planning cycle. Thus, even methods with reasonable action throughput can have feedback loops spanning multiple seconds, limiting their ability to respond promptly to a changing scene. High-frequency closed-loop planning calls for both fast action generation and frequent grounding in newly observed states.

To bridge this gap, we propose StreamWAM, a high-frequency world-action model with closed-loop streaming diffusion for real-time autonomous driving, as shown in Figure 1. StreamWAM adapts a video diffusion transformer to jointly model driving observations and actions. Unlike conventional autoregressive diffusion pipelines that denoise each video-action chunk independently, StreamWAM overlaps the denoising processes of consecutive chunks using a progressive noise schedule (Chen et al., 2024; Ruhe et al., 2024; Zhang et al., 2024; Høeg et al., 2025). After initialization, this streaming diffusion paradigm produces one action chunk per forward pass, even though each chunk undergoes multiple denoising steps. However, directly applying it to closed-loop driving creates a conditioning mismatch: when a new observation arrives, the most recent completed prediction and partially denoised future chunks were generated without it. Replacing that prediction with the observation changes the conditioning context midway through denoising, while the future latents still reflect the earlier prediction. To address this mismatch, we propose a high-frequency closed-loop planning strategy that updates the ongoing streaming rollout as new observations arrive. We employ a lightweight latent alignment module to correct partially denoised future video latents with the latest observation, keeping subsequent predictions grounded in the observed scene. At a planning frequency of 10 Hz, StreamWAM achieves a PDMS of 91.6 on NAVSIM v1 and an EPDMS of 94.3 on NAVSIM v2, showing the benefit of streaming diffusion and frequent feedback.

In summary, our contributions are as follows:

- We propose StreamWAM, a streaming world-action model for real-time autonomous driving, addressing the latency deficit of prior methods and achieving real-time planning.
- We introduce a latent alignment module that updates the predictions of the model with ground truth observations, enabling grounding in long horizon planning.
- We evaluate StreamWAM on planning benchmarks and visual generation metrics, demonstrating its superior performance. We also validate the designs through ablations.

2 Related Work

Vision-Language-Action Models for Autonomous Driving. Vision-Language-Action (VLA) models extend end-to-end driving from a vision-to-action mapping into policies that couple visual observations, language-conditioned intent, and vehicle actions (Jiang et al., 2025; Zhou et al., 2026b). Recent methods increasingly supplement action supervision with future imagery, depth, or compact dynamics representations (Li et al., 2025b; Sheng et al., 2026; Shang et al., 2026). In this line of work, world modeling mainly strengthens the driving representation, while future-image generation need not be part of online planning. StreamWAM instead uses predicted future observations together with actions during inference and updates this joint generation as new observations arrive.

World-Action Models for Autonomous Driving. World-action models (WAMs) couple future-state prediction with executable actions, either by generating both jointly or by inferring actions from predicted future states (Xia et al., 2026; Liu et al., 2026; Shi et al., 2026). Recent work has increasingly drawn on video generative models to capture driving dynamics, while other approaches use future-video modeling during training but predict actions directly at inference (Zhao et al., 2026b).

Despite strong planning results, iterative video denoising remains costly, and many WAMs use low-frequency feedback or multi-second generation windows (Zhao et al., 2026a; Xia et al., 2026; Liu et al., 2026; Shi et al., 2026; Zhang et al., 2025c). StreamWAM instead overlaps the denoising of successive video-action chunks and incorporates each new observation into the partially denoised future latents for high-frequency closed-loop planning.

Real-Time Video Generative Models. The inference latency of a diffusion model depends on the latency of each forward pass and the number of denoising steps. Prior work reduces the cost of each pass through efficient attention, caching, quantization, or token compression (Chen et al., 2025; Zhang et al., 2025a; Chu et al., 2025; Liu et al., 2025; Zhang et al., 2025b; HaCohen et al., 2024). Other work reduces sequential sampling through distillation (Yin et al., 2024b;a) or overlaps denoising across adjacent chunks with staggered noise levels (Ruhe et al., 2024; Zhang et al., 2024; Høeg et al., 2025). Our work extends this streaming paradigm to closed-loop streaming diffusion for joint video-action planning, using latent alignment to incorporate new observations during driving.

3 Method

StreamWAM is a world-action model that plans by jointly generating future driving videos and actions. Section 3.1 introduces the unified video-action model and its block-causal attention. Section 3.2 describes streaming diffusion, which overlaps the denoising of consecutive chunks so that a new video-action chunk is produced at every forward pass. Section 3.3 presents latent alignment, which grounds the partially denoised future in each new observation and closes the planning loop.

3.1 Unified Video-Action Model

Problem Formulation. Let v_t denote the camera frame and a_t the ego action at raw-frame time t, where executing a_t leads to the next observation v_{t+1}. Given the history $h_t = (v_{\leq t}, a_{<t})$ of observed frames and executed actions, StreamWAM models the joint distribution of future actions and observations:

$$p_\theta\big(a_{t:t+L-1},\, v_{t+1:t+L} \mid h_t\big), \tag{1}$$

where L is the prediction horizon in raw frames. The vehicle executes the first few predicted actions, receives new observations, and repeats the prediction. Throughout this section, t indexes raw frames, n indexes video-action chunks, and s indexes forward passes of the denoiser.

As shown in Figure 2, StreamWAM maps videos and actions into a shared latent sequence and models this sequence with a single diffusion transformer (DiT).

Video and Action Latents. Given raw frames $[v_0, v_1, \ldots, v_T]$, a 3D causal VAE encodes them into temporally downsampled latents $[x_0^V, x_1^V, \ldots, x_N^V]$ with $N = T/4$. The first frame is encoded on its own, and every subsequent group of 4 frames is compressed into one latent frame. Because the VAE depends only on past frames, new observations can be encoded and decoded in a streaming manner. For actions, we compute the relative displacement and rotation between adjacent waypoints of the trajectory $[P_0, P_1, \ldots, P_T]$ and apply per-dimension normalization. To match the $4\times$ temporal downsampling of the video, we group every 4 normalized actions into one action latent, giving $[x_1^A, x_2^A, \ldots, x_N^A]$. The action latent x_n^A contains the 4 actions that lead to the frames in x_n^V, so we define the n-th video-action chunk as $B_n = [x_n^A, x_n^V]$. At 10 Hz, one chunk covers 0.4 s of driving. We recover the trajectory by de-normalizing the action latents and integrating the relative motion.

Unified DiT. The DiT backbone is initialized from Wan2.2-TI2V-5B (Wan et al., 2025) and processes the interleaved sequence $[x_0^V, x_1^A, x_1^V, \ldots, x_N^A, x_N^V]$. Video and action tokens share all transformer weights, and new linear heads project the action latents into and out of the token space. We also tried separate modality weights in the style of Mixture-of-Transformers (MoT) (Liang et al., 2025) but observed no performance gain. To place both modalities in one positional space, we extend the 3D RoPE of the video grid to action tokens, in a similar way to Qwen-Image (Wu et al., 2025). A video latent x_n^V with a spatial grid of $H_v \times W_v$ receives temporal position n and spatial positions $\{0, \ldots, H_v - 1\} \times \{0, \ldots, W_v - 1\}$. The k-th action element of x_n^A, with $k \in \{0, 1, 2, 3\}$, receives the same temporal position n and the spatial position $(H_v + k,\, W_v + k)$. This places the action tokens outside the video grid and gives each a distinct position along both axes.

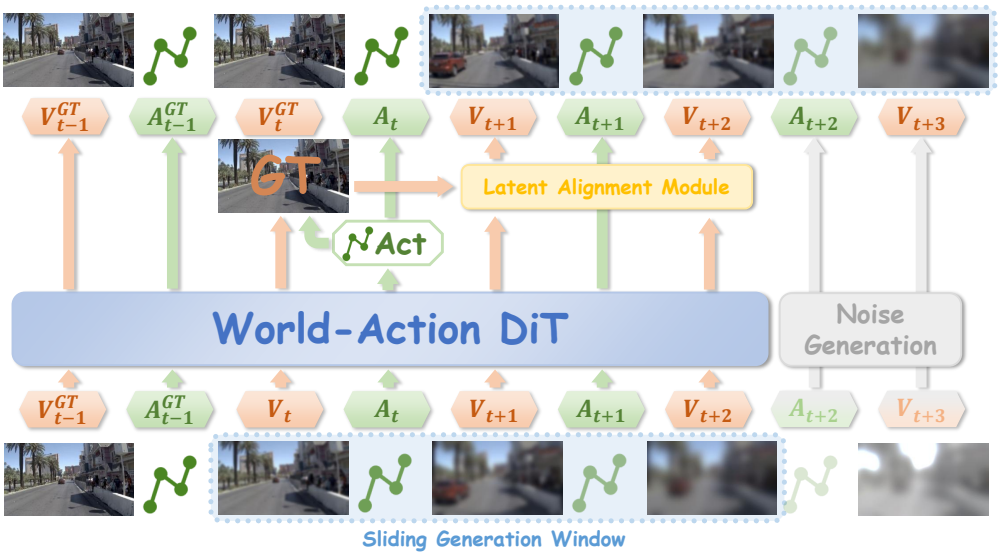

Figure 2: Architecture of StreamWAM. A unified DiT jointly models interleaved video and action latents with block-causal attention, while streaming diffusion continuously emits future chunks.

Block-Causal Attention. We replace the bidirectional attention of the original DiT with a block-causal mask defined at the chunk level. A token in chunk n can attend to all tokens of chunks $m < n$. Within chunk n, video tokens attend to both x_n^A and x_n^V, while action tokens attend only to x_n^A. The first rule enforces temporal causality, which is required for streaming generation. The second rule prevents actions from reading video latents of the same chunk, which are equally noisy and carry little reliable information. Video latents can still use the actions of the same chunk, so the model learns how actions shape observations. Section 4.3 confirms that both rules improve planning.

3.2 Streaming Diffusion for Low-Latency Planning

Flow Matching. For a clean latent x and Gaussian noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, we define the noisy latent at noise level $\tau \in [0, 1]$ as

$$x^{(\tau)} = (1 - \tau)\, x + \tau\, \epsilon, \tag{2}$$

so that $\tau = 0$ is clean data and $\tau = 1$ is pure noise. The DiT predicts the velocity $u = x - \epsilon$ that points from noise to data. Given a prediction $\hat{u}$, one sampling step from level τ to a lower level τ' is

$$x^{(\tau')} = x^{(\tau)} + (\tau - \tau')\, \hat{u}. \tag{3}$$

Overlapped Denoising. Conventional autoregressive diffusion runs all denoising steps of one chunk before starting the next. If each chunk needs S steps, a new plan becomes available only after S forward passes. Enlarging the chunk improves throughput, but it does not shorten this waiting time. StreamWAM instead keeps a sliding window of K chunks at staggered noise levels and denoises them (Ruhe et al., 2024; Zhang et al., 2024; Høeg et al., 2025), as shown in Figure 3.

Let $B_n^{(s)} = [x_n^{A,(s)}, x_n^{V,(s)}]$ be the noisy latents of chunk n before forward pass s, and let the window contain chunks $i, \ldots, i + K - 1$. We use a linear staggered schedule in which the k-th chunk of the window has noise level $\tau_{i+k}^{(s)} = (k + 1)/K$ for $k = 0, \ldots, K - 1$. Let $\mathcal{C}^{(s)}$ denote the clean context, which contains all completed chunks and the observed video latents. One forward pass predicts the velocities of the whole window:

$$\hat{\mathbf{u}}_{i:i+K-1}^{(s)} = D_\theta\big(B_{i:i+K-1}^{(s)}, \mathcal{C}^{(s)}, \boldsymbol{\tau}^{(s)}\big), \tag{4}$$

where $\boldsymbol{\tau}^{(s)} = (\tau_i^{(s)}, \ldots, \tau_{i+K-1}^{(s)})$. Each chunk then takes one sampling step that lowers its noise level by $1/K$. After this step the earliest chunk B_i reaches $\tau = 0$, so StreamWAM outputs its actions, appends it to $\mathcal{C}$, and shifts the window. A new Gaussian noise chunk enters at $\tau = 1$.

Latency and Throughput. Every chunk still receives K denoising steps, yet after a warm-up of $K - 1$ passes at the start of a drive, each forward pass completes one chunk. The planning latency

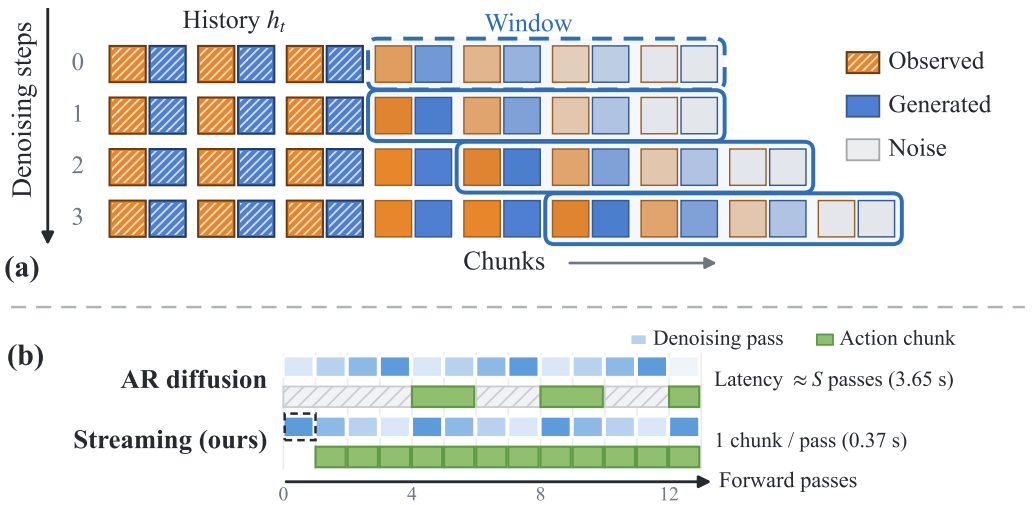

Figure 3: Streaming diffusion paradigm. (a) Future latents receive staggered noise levels and are denoised in a sliding window, so StreamWAM emits one chunk per forward pass after initialization. (b) Compared to AR diffusion models, StreamWAM is able to emit chunks at a higher frequency.

therefore drops from K forward passes to one, without reducing the number of denoising steps per chunk. We use $K = 10$, so the window spans 40 frames, which matches the 4 s planning horizon of NAVSIM. Each completed chunk provides 4 actions at 10 Hz, and the model receives a new observation once per chunk, that is, every 0.4 s. A single forward pass takes 0.368 s (Section 4.3), which is shorter than the 0.4 s covered by one chunk and allows real-time operation.

Training. To support mixed noise levels at inference, we train with per-chunk noise levels following Diffusion Forcing (Chen et al., 2024). The history chunks are kept clean, and each future chunk n receives an independently sampled noise level $\tau_n \sim \mathcal{U}(0, 1)$ and noise ϵ_n. We optimize the flow-matching loss on both modalities:

$$\mathcal{L}_n^A = \mathbb{E}\big[\big\|\hat{u}_n^A - (x_n^A - \epsilon_n^A)\big\|_2^2\big], \quad \mathcal{L}_n^V = \mathbb{E}\big[\big\|\hat{u}_n^V - (x_n^V - \epsilon_n^V)\big\|_2^2\big], \quad \mathcal{L} = \sum_n \big(\mathcal{L}_n^A + \mathcal{L}_n^V\big), \tag{5}$$

where $\hat{u}_n^A$ and $\hat{u}_n^V$ are the predicted velocities, and the expectation is over training samples, noise, and noise levels. Since the staggered schedule at inference is one particular assignment of per-chunk noise levels, the trained model can be applied to it directly. The block-causal mask ensures that each prediction uses only the context that is also available at inference.

3.3 Closed-Loop Planning with Latent Alignment

Conditioning Mismatch. After the vehicle executes the actions of chunk i, it receives the corresponding camera frames, which are encoded into the observed latent x_i^V. Closed-loop planning replaces the predicted latent $\hat{x}_i^V$ in the context $\mathcal{C}$ with x_i^V. In autoregressive diffusion, the next chunk only starts denoising after this replacement, so the observation is used from the first step. In streaming diffusion, however, chunks $i+1, \ldots, i+K-1$ have already been partially denoised while conditioned on $\hat{x}_i^V$. Changing their context midway through denoising makes the partially denoised latents inconsistent with the new context, which degrades the remaining denoising. The discrepancy grows as the observation deviates from the prediction and accumulates over a long drive.

Latent Alignment Module. Let $\hat{x}^V_{i+1:i+K-1}$ denote the partially denoised future video latents obtained under the predicted chunk $\hat{x}_i^V$. Let $\tilde{x}^V_{i+1:i+K-1}$ denote the latents at the same noise levels that would have been obtained under the observed chunk x_i^V. The latent alignment module R_ψ is a lightweight 3D causal residual CNN that predicts the difference between the two:

$$\Delta x^V_{i+1:i+K-1} = \tilde{x}^V_{i+1:i+K-1} - \hat{x}^V_{i+1:i+K-1}, \tag{6}$$

$$x^{V,\text{aligned}}_{i+1:i+K-1} = \hat{x}^V_{i+1:i+K-1} + R_\psi\big(x_i^V, \hat{x}^V_{i+1:i+K-1}\big). \tag{7}$$

Predicting a residual keeps the module close to identity when the observation agrees with the prediction. It also lets the module focus on how the observation should change the future. The module

corrects only video latents. The future action latents are left unchanged, because they attend only to preceding chunks and therefore read the observed latent in all of their remaining denoising steps.

Training the Alignment Module. The module is trained after the DiT and does not require retraining it. We run a frozen StreamWAM on the training set and record x_i^V and $\hat{x}^V_{i+1:i+K-1}$ at each chunk boundary. To obtain the target $\tilde{x}^V_{i+1:i+K-1}$, we repeat the denoising of the same window with the same noise, but with x_i^V in the context from the beginning. We then minimize

$$\mathcal{L}_{\text{align}} = \mathbb{E}\Big[\big\| R_\psi\big(x_i^V, \hat{x}^V_{i+1:i+K-1}\big) - \Delta x^V_{i+1:i+K-1} \big\|_2^2\Big]. \tag{8}$$

The costly re-denoising is only needed for training targets, and inference runs only R_ψ.

Closed-Loop Inference. At each chunk boundary, StreamWAM executes the completed actions, replaces the predicted video latent with the observation, aligns the partially denoised future, and shifts the window before the next forward pass. Algorithm 1 in Section A gives the complete procedure. Overlapped denoising and latent alignment form closed-loop streaming diffusion, which updates the plan with each observation at the cost of one forward pass.

4 Experiments

4.1 Experimental Setup

Benchmarks. We evaluate on NAVSIM v1 (Dauner et al., 2024) and NAVSIM v2 (Cao et al., 2025), which select challenging scenarios from the nuPlan dataset (Caesar et al., 2021). NAVSIM v1 scores each planned 4 s trajectory with a non-reactive simulator and reports PDMS. NAVSIM v2 adds reactive traffic and extended metrics, which are aggregated into EPDMS (Section B.1).

Data. We use the NAVSIM train and test splits and restore the frame rate from 2 Hz to 10 Hz with the original nuPlan sensor data, which provides the dense frames required by streaming generation.

Implementation. StreamWAM is initialized from Wan2.2-TI2V-5B (Wan et al., 2025), takes 13 history frames at 672×384, and predicts 40 future frames (4 s). We train the DiT on 85k samples with the VAE frozen, and then train the latent alignment module on latents collected from the frozen model. At inference, we use a linear schedule with 10 sampling steps and a window of $K = 10$ chunks. StreamWAM* denotes closed-loop driving, in which logged observations are fed back after every chunk (Section B.2). Full hyperparameters are listed in Table 7.

4.2 Main Results

Results on NAVSIM v1. Table 1 compares StreamWAM with perception-based, VLA-based, and WAM-based methods on NAVSIM v1. Under the closed-loop driving setting, StreamWAM achieves a PDMS of 91.6, surpassing the strongest WAM-based baseline DriveVA (90.9) and all VLA-based methods except DynVLA (91.7), and narrowing the gap to the human driver (94.8) to 3.2 points. The improvement is consistent across the individual sub-metrics. On the safety side, StreamWAM obtains the highest No at-fault Collision score of 99.7, showing that the world model rarely causes at-fault collisions when replayed against real traffic. On the efficiency side, it attains an Ego Progress of 86.0, close to the best methods (DynVLA, 86.8) and to the human score of 87.5, indicating that the safety gain does not stem from overly conservative driving. It further matches the best Comfort score of 100.0. Its TTC of 97.6 is slightly below the best value of 98.7 reported by DriveVA, and its DAC of 97.3 trails the 99.1 of DriveVLA-W0, yet the overall PDMS is the highest among all WAM-based methods and the second best overall, suggesting that StreamWAM achieves a better safety–progress trade-off rather than over-optimizing any single sub-metric.

Results on NAVSIM v2. Table 2 reports results on NAVSIM v2 with the extended EPDMS. StreamWAM reaches an EPDMS of 91.0 in the open-loop setting, already surpassing the best prior method EponaV2 (88.9) by 2.1 points, and further improves to 94.3 with closed-loop driving (+3.3), exceeding EponaV2 by 5.4 points. All existing methods cluster between the Ego Status baseline (64.0) and 88.9, whereas StreamWAM clearly breaks away from this cluster. StreamWAM* ranks first on six of nine sub-metrics, including No at-fault Collision (99.7 vs. 98.8), Driving Direction Compliance (99.7 vs. 99.6), Ego Progress (97.8), Time-To-Collision (99.1), Lane Keeping (98.2),

Table 1: **Comparison with state-of-the-art methods on NAVSIM v1.** *: Closed-loop driving (Section B.2). Metrics are defined in Section B.1.

Method	NC ↑	DAC ↑	TTC ↑	Comf. ↑	EP ↑	PDMS ↑
Human	100	100	100	99.9	87.5	94.8
VLA-based Methods						
AdaThinkDrive (Luo et al., 2025)	98.5	94.4	94.9	100	79.9	86.2
AutoVLA (Zhou et al., 2026b)	98.4	95.6	98.0	99.9	85.9	89.1
DriveDreamer-Policy (Zhou et al., 2026a)	98.4	97.1	95.1	100	83.5	89.2
ReCogDrive (Li et al., 2025c)	98.2	97.8	95.2	99.8	83.5	89.6
DriveVLA-W0 (Li et al., 2025b)	98.7	**99.1**	95.3	99.3	83.3	90.2
ExploreVLA (Sheng et al., 2026)	98.8	98.4	96.5	99.9	83.5	90.4
DriveMA (Zheng et al., 2026)	98.7	97.8	95.5	99.0	86.7	91.2
DynVLA (Shang et al., 2026)	98.6	98.7	95.5	100	**86.8**	**91.7**
WAM-based Methods						
LAW (Li et al., 2025a)	96.4	95.4	88.7	99.0	81.7	84.6
Epona (Zhang et al., 2025c)	97.9	95.1	93.8	99.9	80.4	86.2
PWM (Zhao et al., 2026a)	98.6	95.9	95.4	100	81.8	88.1
DriveLaW (Xia et al., 2026)	99.0	97.1	96.7	100	81.3	89.1
DriveWAM (Shi et al., 2026)	98.3	98.1	95.2	100	84.3	90.1
EponaV2 (Xu et al., 2026)	98.6	97.9	95.7	100	84.8	90.4
DriveVA (Liu et al., 2026)	99.2	97.5	**98.7**	100	83.5	90.9
StreamWAM	99.0	95.6	95.4	100	82.0	88.0
StreamWAM*	**99.7**	97.3	97.6	100	86.0	91.6

Table 2: **Comparison with state-of-the-art methods on NAVSIM v2.** *: Closed-loop driving (Section B.2). Metrics are defined in Section B.1.

Method	NC ↑	DAC ↑	DDC ↑	TL ↑	EP ↑	TTC ↑	LK ↑	HC ↑	EC ↑	EPDMS ↑
Ego Status MLP	93.1	77.9	92.7	99.6	86.0	91.5	89.4	98.3	85.4	64.0
Transfuser (Chitta et al., 2022)	96.9	89.9	97.8	99.7	87.1	95.4	92.7	98.3	87.2	76.7
Hydra-MDP++ (Li et al., 2024)	97.2	97.5	99.4	99.6	83.1	96.5	94.4	98.2	70.9	81.4
DriveSuprim (Yao et al., 2026)	97.5	96.5	99.4	99.6	88.4	96.6	95.5	98.3	77.0	83.1
ARTEMIS (Feng et al., 2025)	98.3	95.1	98.6	99.8	81.5	97.4	96.5	98.3	-	83.1
DiffusionDrive (Liao et al., 2025)	98.2	95.9	99.4	99.8	87.5	97.3	96.8	98.3	87.7	84.5
DriveVLA-W0 (Li et al., 2025b)	98.5	**99.1**	98.0	99.7	86.4	98.1	93.2	97.9	58.9	86.1
DriveLaW (Xia et al., 2026)	98.7	96.9	99.6	99.8	87.5	98.3	97.6	**98.4**	77.4	88.6
ExploreVLA (Sheng et al., 2026)	98.8	96.2	99.6	99.8	87.1	98.2	97.8	98.3	86.8	88.8
EponaV2 (Xu et al., 2026)	98.5	97.4	99.5	**99.9**	87.9	98.1	97.7	98.2	77.4	88.9
StreamWAM	99.0	95.6	99.6	99.7	96.9	98.3	97.4	98.3	84.8	91.0
StreamWAM*	**99.7**	97.3	**99.7**	99.8	**97.8**	**99.1**	**98.2**	98.3	**89.8**	**94.3**

and Extended Comfort (89.8), falling behind on Drivable Area Compliance (99.1, DriveVLA-W0), Traffic Light Compliance (99.9, EponaV2), and History Comfort (98.4, DriveLaW).

Effect of Closed-Loop Driving. Even without closed-loop driving, StreamWAM already delivers a competitive PDMS of 88.0, on par with the WAM-based PWM (88.1) and ahead of Epona (86.2). With our streaming diffusion paradigm, we are able to reach a closed-loop frequency of over 2.5Hz, satisfying the requirements for real-time driving and lifting the PDMS from 88.0 to 91.6 (+3.6). This confirms that the closed-loop paradigm is a major source of improvement: by realigning partially denoised future latents with fresh observations, StreamWAM prevents small prediction errors from compounding over the rollout horizon, which is precisely the failure mode of open-loop planning.

Analysis. Our greatest performance leap appears in Ego Progress, where StreamWAM* reaches 97.8, compared with at most 88.4 for prior methods (+9.4). We attribute this to the joint video-action world model: by explicitly predicting future observations, StreamWAM anticipates the evolution of the scene and surrounding agents, and therefore commits to more decisive and efficient maneuvers instead of remaining overly cautious. Meanwhile, the top scores on safety-related metrics (99.7 NC and 99.1 TTC) confirm that the progress gain is not obtained through aggressive driving. The main remaining weakness is DAC, where StreamWAM (97.3) still trails DriveVLA-W0 (99.1), while its Traffic Light Compliance (99.8 vs. 99.9) and History Comfort (98.3 vs. 98.4) fall only marginally

Table 3: **Ablation on causal modeling.**

Causality	PDMS
Bidirectional	87.4
Temporal Causal	89.0
Causal	91.6

Table 4: **Ablation on loop frequency.**

Frequency	Action Size	PDMS
2.5Hz	4	77.4
1.25Hz	8	74.2
0.5Hz	20	73.9
0.25Hz	40	72.7

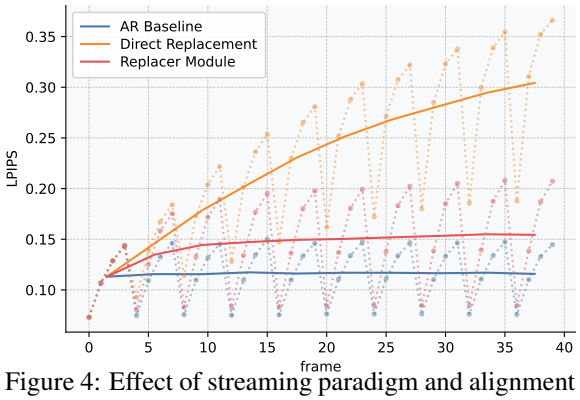

Figure 4: Effect of streaming paradigm and alignment method on planning and video generation.

Table 5: **Ablation on streaming diffusion.**

Diffusion Paradigm	PDMS	Latency / s
AR Diffusion	91.6	3.650
Single Step	78.3	0.365
Streaming Diffusion	91.6	0.368

Table 6: **Ablation on latent alignment.**

Alignment Method	PDMS	LPIPS
None	88.0	0.335
Direct Replacement	91.5	0.227
Latent Alignment Module	91.6	0.145

short of the best. Nevertheless, the substantial gains on all other metrics translate into the highest overall EPDMS. We attribute this lead to the unified video-action architecture that plans with future visual context, and to closed-loop streaming with latent alignment that keeps video and action rollouts grounded in real observations.

4.3 Ablation Study

Causal Modeling. In our design, while we always predict (x_n^A, x_n^V) simultaneously, we enforce a causal relationship between the two modalities, preventing the action latents from attending to the video latents. We investigate the influence of causal modeling with two alternative settings: bidirectional attention, where all latents are visible to each other, and temporal causal attention, where we use bidirectional attention between x_n^A and x_n^V but maintain temporal causality. Both settings are trained under the same parameters as our original model. As shown in Table 3, the partially-denoised video latents can only provide information for more noisy latents. Exposing action latents to current or future video latents, which have high noise levels, will negatively impact planning capabilities. This reaffirms strict causality across time and modalities.

Closed-Loop Frequency. A key advantage of the low latency achieved by the streaming diffusion paradigm is that we can considerably increase the frequency of closed-loop driving. We evaluate the importance of a high closed-loop frequency with closed-loop driving experiments at various closed-loop frequencies without the latent alignment module. As shown in Table 4, the planning performance steadily increases with the closed-loop frequency. Therefore, models with lower inference latency are able to respond to new observations more timely, thus achieving superior performance compared to slower models. This further validates our efficient streaming diffusion paradigm.

Streaming Diffusion Paradigm. The traditional AR diffusion paradigm takes multiple inference steps to roll out one chunk of action. Although the throughput of this method can be increased by extending the chunk size, it still suffers from high latency (Table 5). While reducing the total inference steps mitigates this problem, too few inference steps can lead to performance degradation. By contrast, our streaming diffusion paradigm not only achieves low latency comparable to the single-diffusion-step method, but also retains the high performance of the AR diffusion paradigm. Coupled with the $4\times$ temporal compression, StreamWAM can reach an action frequency of over 10Hz. This best-of-both-worlds design enables us to accomplish low-latency, high-quality planning.

Latent Alignment Module. While efficient, streaming diffusion encounters an alignment problem in long-term inference (Section 3.3). StreamWAM solves this problem with a special latent alignment module. We ablate this design with two alternative settings. First, we ablate the closed-loop driving setting by not updating the model with new observations. Second, we directly replace the



Figure 5: Qualitative closed-loop driving results sampled every 2 s. Each frame overlays the planned trajectory on the observation and shows predicted views at $T + 0.4$ s, $T + 0.8$ s, and $T + 1.2$ s.

current predicted video latents with the corresponding GT latents. Beyond PDMS, we also evaluate video generation quality with LPIPS (Zhang et al., 2018).

In Table 6, the open-loop setting reaches a PDMS of 88.0 and underperforms the closed-loop settings, which match the planning quality of the AR baseline without sacrificing latency. While the latent alignment module offers only a marginal gain over direct replacement, its video generation quality, measured by LPIPS, is significantly superior. This distinction is notable as PDMS only considers planning performance within a fixed 4-second window. As shown in Figure 4, under the direct replacement setting, the video generation quality of StreamWAM, measured by LPIPS, rapidly degrades over time, indicating that although the planned trajectory remains safe within the short evaluation window, the internal world representation is already drifting. This accumulating representation drift will eventually lead to planning performance decay. By contrast, the latent alignment module constantly aligns future predicted latents with ground truth observations, keeping the internal world representation grounded and stablizing LPIPS to near AR diffusion levels. Consequently, StreamWAM is able to maintain effective action planning over longer horizons.

4.4 Visualizations

We offer a qualitative evaluation of the planning and video generation capabilities of StreamWAM in Figure 5. We visualize two random scenarios every 2 seconds. The top image of each frame depicts the predicted trajectory projected onto the current observed image. The bottom part shows the predicted future images at $T{+}0.4s, T{+}0.8s, T{+}1.2s$. Because of the streaming diffusion paradigm, the three images show increasing levels of noise. In both scenarios, StreamWAM accurately predicts the future driving observations and uses them to plan smooth, safe trajectories.

5 Conclusion

In this paper, we addressed two critical obstacles, low planning frequency and high inference latency, that prevent existing world-action models from being deployed in real autonomous vehicles. We proposed StreamWAM, a streaming world-action model that combines overlapped denoising and observation-driven latent alignment into closed-loop streaming diffusion. The former enables low-latency chunk-wise generation, while the latter aligns partially denoised future latents with new observations to mitigate conditioning drift during continuous driving. Extensive experiments on NAVSIM demonstrate that StreamWAM achieves state-of-the-art closed-loop planning performance (PDMS 91.6) while maintaining practical real-time throughput. Limitations and future work are discussed in Section D. We believe StreamWAM provides a solid foundation for bridging the gap between world-model research and practical autonomous driving systems.

AI USE STATEMENT

In this work, we have not used generative AI tools for any tasks with required disclosure, and all such tasks are not applicable to this work. We used generative AI tools only to polish the writing of parts of the manuscript. The authors reviewed all AI-assisted edits to ensure that they preserve the intended technical meaning and contain no unsupported claims. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, the main paper, Section A, and Section B provide detailed descriptions of our methodology, implementation, and evaluation protocol. We will release the complete source code and model weights.

REFERENCES

Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. *arXiv preprint arXiv:2106.11810*, 2021.

Wei Cao, Marcel Hallgarten, Tianyu Li, Daniel Dauner, Xunjiang Gu, Caojun Wang, Yakov Miron, Marco Aiello, Hongyang Li, Igor Gilitschenski, et al. Pseudo-simulation for autonomous driving. *arXiv preprint arXiv:2506.04218*, 2025.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024.

Junsong Chen, Yuyang Zhao, Jincheng Yu, Ruihang Chu, Junyu Chen, Shuai Yang, Xianbang Wang, Yicheng Pan, Daquan Zhou, Huan Ling, et al. Sana-video: Efficient video generation with block linear diffusion transformer. *arXiv preprint arXiv:2509.24695*, 2025.

Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE transactions on pattern analysis and machine intelligence*, 45(11):12878–12895, 2022.

Huanpeng Chu, Wei Wu, Guanyu Feng, and Yutao Zhang. Omnicache: A trajectory-oriented global perspective on training-free cache reuse for diffusion transformer models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16302–16312, 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 (NeurIPS)*, 2024.

Renju Feng, Ning Xi, Duanfeng Chu, Rukang Wang, Zejian Deng, Anzheng Wang, Liping Lu, Jinxiang Wang, and Yanjun Huang. Artemis: Autoregressive end-to-end trajectory planning with mixture of experts for autonomous driving. *arXiv preprint arXiv:2504.19580*, 2025.

Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, et al. Ltx-video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024.

Sigmund H Høeg, Yilun Du, and Olav Egeland. Fast policy synthesis with variable noise diffusion models. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4821–4828. IEEE, 2025.

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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 17853–17862, 2023.

Sicong Jiang, Zilin Huang, Kangan Qian, Ziang Luo, Tianze Zhu, Yang Zhong, Yihong Tang, Menglin Kong, Yunlong Wang, Siwen Jiao, et al. A survey on vision-language-action models for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4583–4595, 2025.

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 *International Conference on Learning Representations*, volume 2025, pp. 42942–42959, 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.

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

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

Weixin Liang, LILI YU, Liang Luo, Srini Iyer, Ning Dong, Chunting Zhou, Gargi Ghosh, Mike Lewis, Wen tau Yih, Luke Zettlemoyer, and Xi Victoria Lin. Mixture-of-transformers: A sparse and scalable architecture for multi-modal foundation models. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=Nu6N69i8SB.

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

Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep embedding tells: It's time to cache for video diffusion model. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 7353–7363, 2025.

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*, 2026.

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

David Ruhe, Jonathan Heek, Tim Salimans, and Emiel Hoogeboom. Rolling diffusion models. *arXiv preprint arXiv:2402.09470*, 2024.

Shuyao Shang, Bing Zhan, Yunfei Yan, Yuqi Wang, Yingyan Li, Yasong An, Xiaoman Wang, Jierui Liu, Lu Hou, Lue Fan, et al. Dynvla: Learning world dynamics for action reasoning in autonomous driving. *arXiv preprint arXiv:2603.11041*, 2026.

Zihao Sheng, Xin Ye, Jingru Luo, Sikai Chen, and Liu Ren. Explorevla: Dense world modeling and exploration for end-to-end autonomous driving. *arXiv preprint arXiv:2604.02714*, 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.

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

Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng-ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, et al. Qwen-image technical report. *arXiv preprint arXiv:2508.02324*, 2025.

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

Jiawei Xu, Zhizhou Zhong, Zhijian Shu, Mingkai Jia, Mingxiao Li, Jia-Wang Bian, Qian Zhang, Kaicheng Zhang, Jin Xie, Jian Yang, et al. Eponav2: Driving world model with comprehensive future reasoning. *arXiv preprint arXiv:2605.14696*, 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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 11910–11918, 2026.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in neural information processing systems*, 37:47455–47487, 2024a.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6613–6623, 2024b.

Jintao Zhang, Chendong Xiang, Haofeng Huang, Jia Wei, Haocheng Xi, Jun Zhu, and Jianfei Chen. Spargeattention: Accurate and training-free sparse attention accelerating any model inference. In *International Conference on Machine Learning*, pp. 76397–76413. PMLR, 2025a.

Jintao Zhang, Pengle Zhang, Jun Zhu, Jianfei Chen, et al. Sageattention: Accurate 8-bit attention for plug-and-play inference acceleration. In *International Conference on Learning Representations*, volume 2025, pp. 71566–71585, 2025b.

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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 27220–27230, 2025c.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Zihan Zhang, Richard Liu, Rana Hanocka, and Kfir Aberman. Tedi: Temporally-entangled diffusion for long-term motion synthesis. In *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–11, 2024.

Zhida Zhao, Talas Fu, Yifan Wang, Lijun Wang, and Huchuan Lu. From forecasting to planning: Policy world model for collaborative state-action prediction. *Advances in Neural Information Processing Systems*, 38:134585–134611, 2026a.

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

Weicheng Zheng, Yixin Huang, Qiao Sun, Derun Li, and Hang Zhao. Drivema: Driving vision-language-action models with verifiable meta-actions. *arXiv preprint arXiv:2605.31271*, 2026.

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

Zewei Zhou, Tianhui Cai, Seth 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. *Advances in Neural Information Processing Systems*, 38: 27920–27956, 2026b.

Appendix

A Implementation Details

Datasets. nuPlan (Caesar et al., 2021) is a large-scale planning benchmark that supports open-loop and closed-loop simulation and contains 1200 hours of diverse driving data with maps and labels. NAVSIM v1 (Dauner et al., 2024) is a pseudo-simulation benchmark built on nuPlan. It filters nuPlan for challenging scenarios, downsamples the data to 2 Hz, and scores the planned trajectory with a non-reactive simulator. NAVSIM v2 (Cao et al., 2025) improves the realism of the simulation with reactive traffic and extends the scoring with DDC, TL, LK, HC, and EC. The metrics of both versions are defined in Section B.1.

Training. The action projection heads are randomly initialized, and all other DiT weights are initialized from Wan2.2-TI2V-5B (Wan et al., 2025). Following NAVSIM, which provides 1.5 s of history and requires a 4 s plan, we use 13 history frames and a prediction window of 40 frames. The DiT is trained on 8 NVIDIA H20 GPUs with DeepSpeed ZeRO-2 for 20 epochs, with the VAE kept frozen. The latent alignment module is trained afterwards on a single H20 GPU with the data described below.

Hyperparameters. Table 7 lists the hyperparameters of StreamWAM and the latent alignment module. All experiments use the same configuration unless otherwise stated in Section 4.3.

Data Processing. NAVSIM provides annotations at 2 Hz. To train and evaluate at the 10 Hz frame rate of StreamWAM, we retrieve the corresponding camera frames and ego poses from the original nuPlan sensor logs for every NAVSIM scene. For each pair of adjacent waypoints, we compute the relative displacement and rotation in the ego frame of the earlier waypoint. Each action dimension is then normalized independently before being grouped into action latents of 4 consecutive actions. At inference, the predicted actions are de-normalized and cumulatively integrated to recover the trajectory in the ego frame of the current observation.

Closed-Loop Streaming Inference. Algorithm 1 summarizes the complete inference procedure described in Sections 3.2 and 3.3. The warm-up phase is executed once at the start of a drive. Afterwards, every iteration of the loop consists of a single DiT forward pass, one VAE encoding of the new observation, and one pass of the latent alignment module.

Training Data for Latent Alignment. The training pairs for R_ψ are collected offline with the frozen DiT on the training split. At every chunk boundary of a rollout, we store the observed latent x_i^V, the partially denoised future latents $\hat{x}^V_{i+1:i+K-1}$ obtained under the predicted chunk, and the target latents $\tilde{x}^V_{i+1:i+K-1}$ obtained under the observed chunk (Section 3.3). Since the DiT is frozen, this collection is performed only once, and the module can be trained on a single GPU. Since R_ψ is a lightweight CNN on video latents, its inference cost is small relative to a DiT forward pass.

B Evaluation Protocol

B.1 Metrics

We follow the official evaluation of NAVSIM v1 (Dauner et al., 2024) and NAVSIM v2 (Cao et al., 2025). Each planned 4 s trajectory is simulated and scored with the following sub-metrics. No at-fault Collision (NC), Drivable Area Compliance (DAC), Driving Direction Compliance (DDC), and Traffic Light Compliance (TL) are multiplicative penalties that set or reduce the score when a rule is violated. Time-To-Collision (TTC), Ego Progress (EP), Comfort (Comf.), Lane Keeping (LK), History Comfort (HC), and Extended Comfort (EC) are weighted terms that measure safety margin, efficiency, and smoothness. On NAVSIM v1, the Predictive Driver Model Score (PDMS) combines them as

$$\text{PDMS} = \text{NC} \times \text{DAC} \times \frac{5 \cdot \text{TTC} + 2 \cdot \text{Comf.} + 5 \cdot \text{EP}}{12}. \tag{9}$$

On NAVSIM v2, the Extended Predictive Driver Model Score (EPDMS) adds DDC and TL to the penalties and LK, HC, and EC to the weighted terms:

$$\text{EPDMS} = \text{NC} \times \text{DAC} \times \text{DDC} \times \text{TL} \times \frac{5 \cdot \text{TTC} + 5 \cdot \text{EP} + 2 \cdot \text{LK} + 2 \cdot \text{HC} + 2 \cdot \text{EC}}{16}. \tag{10}$$

Table 7: **Hyperparameters of StreamWAM.**

Configuration	Value
Model	
Backbone	Wan2.2-TI2V-5B (Wan et al., 2025)
Video resolution	672×384 (resized from 1920×1080)
Frame rate	10 Hz
Temporal compression of the VAE	$4\times$
Actions per action latent	4
History length	13 frames
Prediction horizon L	40 frames (4 s)
Window size K	10 chunks
DiT Training	
Trainable modules	DiT and action projection heads (VAE frozen)
Training samples	85k
Epochs	20
Batch size	8
Optimizer	AdamW
Learning rate	1×10^{-5} (constant)
Weight decay	1×10^{-2}
Hardware	8 NVIDIA H20 GPUs, DeepSpeed ZeRO-2
Latent Alignment Module Training	
Architecture	3D causal residual CNN
Epochs	10
Batch size	4
Learning rate	1×10^{-4}
Hardware	1 NVIDIA H20 GPU
Inference	
Denoising schedule	Linear, 10 sampling steps

Algorithm 1 Closed-loop streaming inference of StreamWAM

Input. Observed history h_t, DiT D_θ, causal VAE encoder E, latent alignment module R_ψ, window size K.

1. $\mathcal{C} \leftarrow E(h_t)$, $B_{i:i+K-1} \sim \mathcal{N}(0, \mathbf{I})$, run $K-1$ warm-up passes ▷ fill the window
2. $\tau_{i+k} \leftarrow (k+1)/K$ for $k = 0, \ldots, K-1$ ▷ staggered noise levels
3. $\hat{\mathbf{u}}_{i:i+K-1} \leftarrow D_\theta(B_{i:i+K-1}, \mathcal{C}, \boldsymbol{\tau})$ ▷ one forward pass
4. $B_n \leftarrow B_n + \hat{u}_n/K$ for $n = i, \ldots, i+K-1$ ▷ one sampling step per chunk
5. Decode and execute the actions in x_i^A ▷ chunk i is complete
6. $x_i^V \leftarrow E(v_{\text{new}})$, $\mathcal{C} \leftarrow \mathcal{C} \cup \{x_i^A, x_i^V\}$ ▷ replace $\hat{x}_i^V$ with the observation
7. $\hat{x}^V_{i+1:i+K-1} \leftarrow \hat{x}^V_{i+1:i+K-1} + R_\psi(x_i^V, \hat{x}^V_{i+1:i+K-1})$ ▷ latent alignment
8. $i \leftarrow i+1$, append $B_{i+K-1} \sim \mathcal{N}(0, \mathbf{I})$, and go to step 2 ▷ shift the window

All scores are averaged over the test scenes and reported as percentages.

B.2 Open-Loop and Closed-Loop Settings

We report two settings for StreamWAM in Tables 1 and 2. In the default setting, denoted StreamWAM, the model receives only the observed history and generates the full 4 s horizon without any further observation. In the closed-loop setting, denoted StreamWAM*, the model advances chunk by chunk, and after each chunk it receives the logged camera frames of the next 0.4 s from the dataset. These frames replace the corresponding predicted latents and trigger latent alignment, as in Algorithm 1. The frequency ablation in Table 4 varies the number of actions executed between observation updates, and it is run without latent alignment to isolate the update frequency.

Because the observations are taken from driving logs, they follow the recorded trajectory rather than the trajectory planned by StreamWAM. This setting therefore measures how well the model exploits timely observations, but it does not model the feedback of its own actions on future observations. Evaluating StreamWAM with a reactive sensor simulator is an important direction for future work.

C Relation to Other Streaming Diffusion Methods

Several works assign different noise levels to different temporal positions of a sequence. Diffusion Forcing (Chen et al., 2024) trains a causal model with independent per-token noise levels, which enables flexible sampling schedules at inference. Rolling Diffusion (Ruhe et al., 2024) generates long videos with a sliding window in which the noise level increases with time. TEDi (Zhang et al., 2024) applies a similar temporally entangled schedule to motion synthesis, and SDP (Høeg et al., 2025) uses it to stream robot action trajectories with low latency. StreamWAM builds on this family of staggered schedules but differs in two respects. First, it jointly streams video and action latents in one interleaved sequence with a block-causal mask, so that each emitted action chunk is paired with an imagined future observation. Second, prior streaming methods assume that the conditioning context is fixed while a chunk is being denoised. In closed-loop driving, a new observation arrives while future chunks are still partially denoised, which changes their context midway through denoising. StreamWAM addresses this setting explicitly with latent alignment, which corrects the partially denoised future at every observation update.

D Limitations and Future Work

Limitations. StreamWAM currently relies on the assumption of perfect, instantaneous sensor observations for closed-loop updates. In practice, sensor latency and occlusions may violate this assumption. Additionally, due to the limited size of our Latent Alignment Module, StreamWAM is unable to completely match the AR baseline in Section 4.3 in terms of video generation quality.

Future Work. We plan to extend StreamWAM in three directions: (i) incorporating other efficiency-enhancing techniques including caching and distillation to further improve throughput, (ii) using benchmarks with longer horizons than NAVSIM to validate the long-term stability of the Latent Alignment Module, and (iii) integrating high-level semantic intents (e.g., navigation commands) as additional conditioning signals to improve the controllability of streaming generation.

E Extended Discussion

Latency. Streaming diffusion changes when a plan becomes available, but it does not change the total amount of computation per chunk, since every chunk is still denoised in K steps. The planning latency is therefore bounded by the latency of a single forward pass over the window. Techniques that reduce the cost of each forward pass, such as caching (Liu et al., 2025; Chu et al., 2025), efficient attention (Zhang et al., 2025b;a), or step distillation (Yin et al., 2024b;a), are orthogonal to our design and can be readily combined with StreamWAM.

Planning Frequency and Chunk Size. The chunk size is tied to the $4\times$ temporal compression of the VAE, so each forward pass completes 4 actions that cover 0.4 s. A VAE with weaker temporal compression allows smaller chunks and more frequent updates, at the cost of more video tokens.

Observation Assumptions. As discussed in Section D, StreamWAM assumes that each observation is available immediately after the corresponding chunk is executed. In a deployed system, sensor latency would delay the update by a fraction of a chunk. Since latent alignment only requires the observed latent and the current partially denoised latents, a delayed observation could be incorporated at the next chunk boundary without changing the model.