VELOFORESIGHT: STRUCTURED ADMISSION FOR TRAINING-FREE FLOW ACCELERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion transformers generate high-quality videos, but repeated network evaluations make inference expensive. Predicting velocities from sparse exact evaluations can reduce this cost. However, forecasts based on sparse evaluations can become unreliable over longer prediction horizons, and their errors affect subsequent sampling states. In this paper, we introduce VELOFORESIGHT, a training-free sampler built around a structured predictor of velocity changes. The predictor estimates trend and curvature from past exact evaluations, controls their contributions separately, and progressively attenuates curvature over longer forecast horizons. Relative-norm bounds limit the size of the predicted changes before they enter the native scheduler. Across image and video generation, VELOFORESIGHT achieves up to $5.18\times$ speedup on Wan2.1-14B and $4.76\times$ on SD3.5 Large relative to 50-step sampling.

1 INTRODUCTION

Diffusion and flow-based models have become central to high-quality image and video generation. Flow matching and rectified flow generate images and videos by progressively transforming noise along a learned time-dependent velocity field (Lipman et al., 2023; Liu et al., 2023). Many recent generators represent this field using large diffusion transformer (DiT) (Peebles & Xie, 2023). Although these architectures produce high-quality outputs, repeatedly evaluating the network over many sampling steps makes inference computationally expensive. Training-free caching reduces this cost by reusing or predicting intermediate features or model outputs at selected steps, without retraining the underlying model.

Existing caching methods exploit similarities across sampling steps to replace expensive network evaluations with inexpensive approximations. TaylorSeer and Spectrum forecast intermediate features from their history (Liu et al., 2025b; Han et al., 2026), while FastFlow and VDE estimate the velocity output directly (Bajpai et al., 2026b; Tan et al., 2026). However, longer gaps between exact evaluations can make these estimates less reliable, and errors in the resulting velocity updates affect subsequent sampling states. Our controlled study shows that extrapolating velocity changes from sparse observations can produce excessively large updates. To address this problem, we introduce VELOFORESIGHT, a training-free sampler with a bounded predictor of velocity changes. At a skipped step, our predictor uses velocities from previous model evaluations to estimate a change relative to the latest exact velocity. The proposed change consists of trend and curvature components. We apply structured reliability control to these components, accounting for prediction order, forecast horizon, and relative magnitude. This constrains the extrapolated change before the resulting velocity is passed to the sampler. We refer to these two stages as *proposal construction* and *update admission*.

VELOFORESIGHT combines this predictor with two auxiliary feedback paths. At exact evaluations, residual calibration measures the difference between predicted and actual velocities and uses this information to correct subsequent forecasts. Clean-state consistency uses estimates of the noise-free latent from earlier exact evaluations, together with the current accelerated latent, to construct a state correction which is later scaled by a bounded gain. We allocate a fixed number of exact model evaluations according to how the scheduler's noise level changes during sampling. Figure 1 summarizes the method.

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.

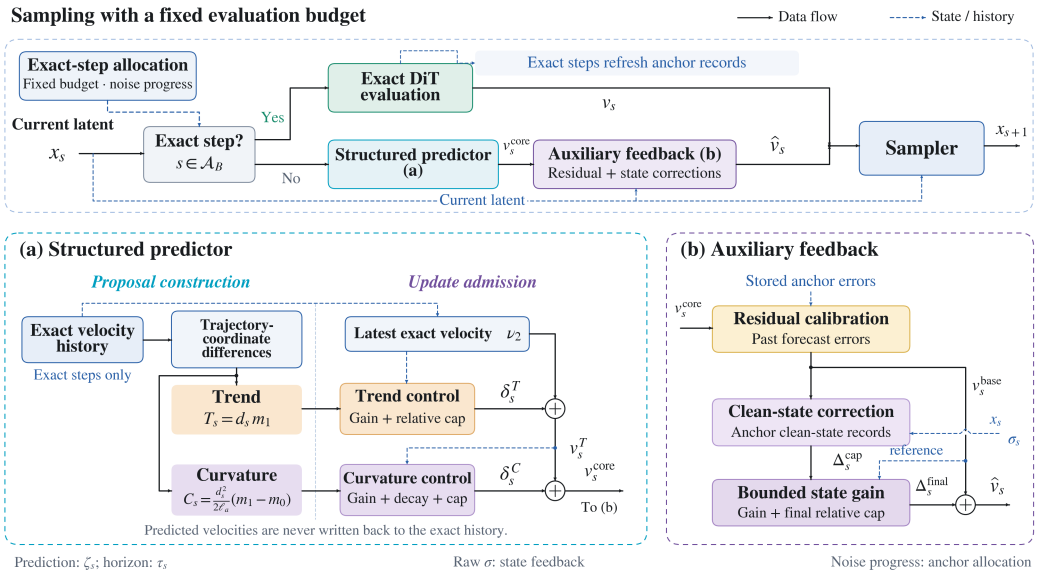


Figure 1: **Overview of VELOFORESIGHT.** Exact steps refresh anchor records, while skipped steps use predicted velocities. (a) Trend and curvature proposals are controlled through separate gains, horizon-dependent curvature attenuation, and sequential relative-norm bounds. (b) Residual calibration and bounded state feedback produce the final velocity supplied to the native sampler. Prediction uses normalized cumulative noise progress on Wan2.1-14B and sampling-step indices on HunyuanVideo. The curvature normalization is $\ell_a = (g_0 + g_1)/2$ on Wan and $\ell_a = g_1$ on Hunyuan, where g_0 and g_1 are consecutive anchor intervals in the prediction coordinate. Both backbones measure the attenuation horizon in elapsed sampling steps.

We evaluate VELOFORESIGHT on Wan2.1-14B (Wan Team et al., 2025) and HunyuanVideo (Kong et al., 2024) at 13 exact sampling steps (logical NFE13), assessing both fidelity to full-computation references and VBench (Huang et al., 2023) Quality. A three-seed study further examines the robustness of the fidelity results to different initial noise samples. Controlled experiments on Wan show that the proposed structured reliability control improves fidelity over the tested simpler controls.

The contributions of this paper include:

- We introduce a multi-anchor velocity predictor whose complete scheduler-facing change proposal is controlled before the sampler update. Its structured admission rule uses distinct trend and curvature gains, horizon-dependent curvature attenuation, and relative-norm bounds.

- We develop VELOFORESIGHT, a training-free exact-budget sampler that combines this structured predictor with anchor-revealed residual calibration, realized-state feedback, and prompt-independent noise-progress anchor allocation, while leaving the denoiser and native scheduler unchanged.

- We evaluate VELOFORESIGHT across four image and video backbones on VBench and DrawBench200, with local comparisons against Spectrum, TaylorSeer, and adapted VDE-core. At NFE9, video-generation speedups reach $5.18\times$, while image-generation speedups reach $4.76\times$ at NFE10, relative to 50-step sampling. On both video backbones, VELOFORESIGHT at NFE13 improves PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) over Spectrum at NFE14 while further reducing runtime. Multi-seed evaluations and controlled ablations provide additional evidence for the NFE13 fidelity gains and the effectiveness of structured control.

2 Related Work

Diffusion sampling acceleration. Distillation methods train a student model to reproduce a teacher sampler with fewer sampling steps (Salimans & Ho, 2022). Fast numerical solvers, such as DPM-Solver, reduce the number of network evaluations through higher-order integration without additional training (Lu et al., 2022). Learned caching methods such as DisCa train lightweight predictors to approximate intermediate features and reduce computation within each sampling step (Zou et al., 2026). Our work focuses on training-free acceleration by predicting velocities at skipped steps, while retaining the pretrained denoiser and native scheduler.

Feature reuse and forecasting. Feature caching reuses attention outputs or hidden features from earlier sampling steps to avoid repeated computation (Zhao et al., 2024; Zou et al., 2025; Liu et al., 2025a). Forecasting methods extend this idea by estimating how these features evolve between evaluations. TaylorSeer predicts features using Taylor expansions with derivatives estimated from past observations (Liu et al., 2025b), while Spectrum fits Chebyshev polynomials to feature histories for multi-step forecasting (Han et al., 2026). VeloForesight predicts changes in the model's velocity output relative to its latest exact evaluation.

Direct velocity prediction. Velocity-based acceleration estimates sampling updates from previous model evaluations. MeanCache constructs interval-average velocities using cached Jacobian–vector products (Gao et al., 2026), while FastFlow extrapolates velocities through finite differences and adapts the number of skipped steps with a bandit policy (Bajpai et al., 2026b). VDE takes a geometric approach: it decomposes velocity into components parallel and orthogonal to the current latent, extrapolates their coefficients, and reuses the orthogonal direction (Tan et al., 2026). For autoregressive video generation, SCOPE combines noise-level Taylor extrapolation with a cache–predict–recompute policy and selective frame computation (Cui et al., 2026). Its released frame-wise predictor uses threshold-based curvature screening and elementwise clipping to constrain extrapolation. In comparison, VeloForesight constructs velocity-change proposals in the chosen trajectory coordinate and attenuates curvature according to the elapsed sampling-step horizon.

Prediction reliability and correction. Recent methods assess forecast reliability to control how predictions are used during sampling. Confidence-Gated Taylor measures forecast error in an early transformer block to decide whether to accept a feature forecast or complete the network evaluation (Guan et al., 2025). FlowCast extrapolates future states using the current velocity and verifies the proposed trajectory through parallel full-model evaluations (Bajpai et al., 2026a). GP-Refiner combines feature correction with uncertainty estimation: it corrects cached features through online Gaussian-process regression and triggers full computation when posterior uncertainty is high (Shen et al., 2026). RACER uses disagreement between predictors to continuously shrink the forecast toward the latest exact feature, while relocating evaluations within a fixed budget (Li et al., 2026). These methods regulate forecast use through model-based verification, uncertainty estimation, or error correction. VeloForesight focuses on the structure of the velocity change itself, applying separate weights and norm bounds to its trend and curvature components.

3 Method

3.1 Sampling formulation

Let x_s denote the latent at sampling step s, and let $v_s = F_\theta(x_s, t_s, c)$ denote the velocity obtained from the pretrained model with timestep t_s, conditioning c, and the configured guidance. We write the native sampling update as

$$(x_{s+1}, \xi_{s+1}) = \Psi_s(x_s, \xi_s, w_s), \qquad s = 0, \dots, S-1, \tag{1}$$

where ξ_s contains any internal solver history and w_s is the velocity supplied to the solver. This notation includes both single-step and multistep solvers.

An anchor set $\mathcal{A}_B$ specifies B exact evaluation steps. At $s \in \mathcal{A}_B$, we use $w_s = v_s$. Otherwise, we use a predicted velocity $w_s = \widehat{v}_s$. Exact velocities are evaluated at the current accelerated latent, rather than taken from a separate reference trajectory. Only these exact observations enter the predictor history. The native solver still updates its own state at every sampling step.

The budget B counts exact evaluation steps with the configured guidance. It need not equal the number of physical transformer calls. Appendix A specifies this distinction for each backbone.

3.2 VELOCITY-CHANGE PREDICTION

At a skipped sampling step, we predict the change relative to the latest exact velocity using recent velocity observations. Let $a_0 < a_1 < a_2 < s$ be the three most recent exact evaluation steps, with velocities $\nu_i = v_{a_i}$.

Let ζ_s denote an increasing coordinate along the sampling trajectory, such as the sampling-step index or normalized noise progress. We distinguish the extrapolation distance in this coordinate from the number of elapsed sampling steps:

$$d_s = \max(\zeta_s - \zeta_{a_2}, 0), \qquad \tau_s = s - a_2. \tag{2}$$

The predictor uses d_s to extrapolate velocity changes, whereas the reliability control uses τ_s to attenuate long-horizon corrections.

We estimate the recent trend from consecutive exact observations. Define the coordinate intervals and secant slopes as

$$g_1 = \max(\zeta_{a_2} - \zeta_{a_1}, \varepsilon_{\mathrm{fd}}), \qquad g_0 = \max(\zeta_{a_1} - \zeta_{a_0}, \varepsilon_{\mathrm{fd}}), \tag{3}$$

$$m_1 = \frac{\nu_2 - \nu_1}{g_1}, \qquad m_0 = \frac{\nu_1 - \nu_0}{g_0}, \tag{4}$$

where $\varepsilon_{\mathrm{fd}}$ protects small coordinate intervals.

The latest slope gives the trend proposal, while the change between successive slopes supplies a curvature correction:

$$T_s = d_s m_1, \qquad C_s = \frac{d_s^2}{2\ell_a}(m_1 - m_0). \tag{5}$$

Here ℓ_a is the interval used to normalize the slope difference. The evaluated implementations use either the mean of the two observation intervals or the latest interval:

$$\ell_a = \begin{cases} (g_0 + g_1)/2, & \text{mean-interval normalization,} \\ g_1, & \text{latest-interval normalization.} \end{cases} \tag{6}$$

The prediction coordinate and normalization convention are fixed for each implementation. Their definitions and the settings used in each experiment are specified in Appendix B.

The resulting velocity proposal is

$$\widetilde{v}_s = \nu_2 + T_s + C_s. \tag{7}$$

The trend and curvature terms extrapolate recent changes between exact velocities in the chosen trajectory coordinate. For a fixed anchor history, their magnitudes scale linearly and quadratically with the extrapolation distance, respectively. Structured admission regulates these increments before they are used in the native solver update.

With one exact observation, the predictor holds its velocity. With two observations, it constructs only the trend proposal. Curvature is included once three observations are available. Predicted velocities are never inserted into the exact-anchor history.

3.3 STRUCTURED RELIABILITY CONTROL

To account for their different growth rates with extrapolation distance, we control the trend and curvature proposals using order-specific gains and relative-norm bounds. The curvature gain additionally decreases with the elapsed sampling-step horizon τ_s.

For a positive relative radius r, radial limiting has the form

$$\mathcal{C}_r(\delta; b) = \begin{cases} 0, & \|\delta\|_2 = 0, \\ \delta \min\left(1, \dfrac{r\|b\|_2}{\|\delta\|_2}\right), & \|\delta\|_2 > 0. \end{cases} \tag{8}$$

Here δ is the proposed velocity increment and b is the reference velocity. The operator rescales δ only when its norm exceeds $r\|b\|_2$. For the core predictor, the norm is computed over the entire velocity tensor. Numerical safeguards omitted from this expression are detailed in Appendix D, together with the norm reductions used in the feedback modules.

The two increments are controlled sequentially:

$$\delta_s^T = \mathcal{C}_{r_T}(\alpha_T T_s; \nu_2), \qquad v_s^T = \nu_2 + \delta_s^T, \tag{9}$$

$$\delta_s^C = \mathcal{C}_{r_C}\left(\frac{\alpha_C}{1+\tau_s/H_C} C_s; v_s^T\right), \qquad v_s^{\text{core}} = v_s^T + \delta_s^C. \tag{10}$$

The gains α_T and α_C determine the contributions of the two orders. The horizon scale H_C controls the attenuation of the curvature gain, while r_T and r_C bound the increments relative to the velocities they modify. In particular, the curvature cap uses v_s^T, which already includes the bounded trend.

Equations 7 and 10 define proposal construction and update admission, respectively. Hyperparameter values are provided in Appendix C.

3.4 Practical extensions

Residual calibration. An exact evaluation reveals the error of a forecast constructed from earlier observations. At an exact step a, we reconstruct $v_a^{\text{core},-}$ from the old anchor history, before inserting the new velocity. This prediction includes the structured control but excludes residual and state feedback. Its residual is $e_a = v_a - v_a^{\text{core},-}$.

We maintain an exponentially weighted average of capped residuals, $\bar{e}_a$, together with an alignment weight ρ_a. A new residual is incorporated into the average when its cosine similarity with the previous average is at least a prescribed threshold. The alignment weight summarizes this directional agreement and scales the residual correction at subsequent skipped steps. At a skipped step s, let a denote the most recent exact evaluation step. The residual-corrected velocity is

$$v_s^{\text{base}} = v_s^{\text{core}} + \mathcal{C}_{r_R}\left(\frac{\alpha_R \rho_a}{1+(s-a)/H_R} \bar{e}_a; v_s^{\text{core}}\right). \tag{11}$$

Only exact evaluations update these statistics. At the first valid update, we initialize $\bar{e}_a = \widetilde{e}_a$ using the capped residual and set $\rho_a = \rho_{\text{init}}$, where ρ_{init} is a hyperparameter. Appendix D specifies the update rules, and Table 8 lists the parameter values used in our experiments.

State feedback. To incorporate the current accelerated latent, we record a clean-latent estimate at eligible exact evaluations:

$$\chi_a = x_a - \sigma_a v_a. \tag{12}$$

Under the convention $x_\sigma = (1-\sigma)x_{\text{clean}} + \sigma\epsilon$ and $v = \epsilon - x_{\text{clean}}$, this is an estimate of the clean latent obtained from the learned velocity.

We extrapolate the two latest records linearly in the raw scheduler noise level to obtain $\widehat{\chi}_s$, holding the available record when only one exists. The resulting state-derived velocity is

$$v_s^{\text{state}} = \frac{x_s - \widehat{\chi}_s}{\max(\sigma_s, \varepsilon_{\sigma,b})}, \tag{13}$$

where b denotes the backbone. Feedback is bypassed when no record is available or $\sigma_s \le \varepsilon_{\sigma,b}$.

Division by a small noise level can amplify errors in $\widehat{\chi}_s$, so we bound the correction before adding it to the residual-corrected velocity:

$$\Delta_s^{\text{cap}} = \mathcal{C}_{r_X}^{(b)}\left(\lambda_s(v_s^{\text{state}} - v_s^{\text{base}}); v_s^{\text{base}}\right), \tag{14}$$

where $\lambda_s = \alpha_X \,\text{clip}(1-\sigma_s, 0, 1)$ and $\mathcal{C}_{r_X}^{(b)}$ uses the backbone-specific norm reduction.

Corrected gain adaptation (CGA) adjusts this correction using its normalized relative magnitude q_s and the progress ϕ_s since the latest exact anchor:

$$\gamma_s = \text{clip}(1 + g_+(1-q_s) - g_-\phi_s q_s,\ \gamma_{\min}, \gamma_{\max}). \tag{15}$$

Table 1: **Image-generation results on DrawBench200.** Local results on 200 paired prompts per configuration. Fidelity is measured against each backbone's Full50 reference and speedup is relative to matched local Full50 sampling. CLIP is reported as $100\times$ cosine similarity. Within each block and backbone, the best and second-best values for each metric are shown in red and blue, respectively. Ties are determined at the displayed precision.

	FLUX.1-dev							SD3.5 Large						
	Acceleration		Fidelity			Semantic quality		Acceleration		Fidelity			Semantic quality	
Method	Latency (s)↓	Speedup↑	PSNR↑	SSIM↑	LPIPS↓	ImageReward↑	CLIP↑	Latency (s)↓	Speedup↑	PSNR↑	SSIM↑	LPIPS↓	ImageReward↑	CLIP↑
TaylorSeer ($N = 4, O = 1$)	5.87	2.67×	22.055	0.8304	0.1655	**1.02**	**27.59**	6.47	2.13×	13.005	0.5916	0.4173	0.83	28.05
VDE-core + VF anchors (NFE14)	**4.54**	**3.46×**	23.877	0.8166	0.1475	0.97	27.36	**4.00**	3.46×	17.374	0.6830	**0.2435**	0.96	**28.80**
Spectrum (NFE14)	4.70	3.38×	25.666	0.8732	**0.1150**	0.99	27.40	4.18	3.38×	**17.890**	0.7491	0.2508	0.97	28.58
VELOFORESIGHT (NFE14)	4.62	3.44×	**25.854**	**0.8740**	0.1192	0.97	27.41	4.05	**3.49×**	17.827	**0.7543**	0.2464	**0.99**	28.39
TaylorSeer ($N = 6, O = 1$)	4.69	3.34×	19.728	0.7614	0.2419	**1.00**	**27.69**	5.59	2.47×	11.146	0.4895	0.5453	0.32	27.26
VDE-core + VF anchors (NFE10)	**3.30**	**4.76×**	21.174	0.7297	0.2486	0.95	27.57	**2.91**	4.75×	14.852	0.5776	**0.3628**	**0.86**	**28.49**
Spectrum (NFE10)	3.40	4.67×	22.735	0.8028	**0.1946**	0.98	27.63	3.03	4.67×	14.708	0.6166	0.3905	0.74	27.82
VELOFORESIGHT (NFE10)	3.35	4.74×	**23.000**	**0.8078**	0.1989	0.98	27.56	2.97	**4.76×**	**14.993**	**0.6472**	0.3690	0.85	27.85

TaylorSeer uses $W = 5$ and 16/12 full block refreshes for $N = 4/6$, with 50 outer transformer calls and its SD3.5 implementation is an interface adaptation. VDE-core is adapted to use frozen VeloForesight image anchors at matched budgets. Spectrum latency excludes each loaded process's first image; TaylorSeer and VDE latency averages all 200 images. Their speedups use 10 matched prompts over three rounds, with Full50 means of 15.68 s (FLUX) and 13.79 s (SD3.5).

The final scheduler-facing velocity is

$$\widehat{v}_s = v_s^{\text{base}} + \gamma_s \Delta_s^{\text{cap}} \min\left(1, \frac{r_F}{\max(\eta'_s, \varepsilon_{\text{CGA}})}\right), \tag{16}$$

where η'_s measures the relative magnitude of the gain-scaled correction. CGA uses batch-mean norm statistics, so its gain and final scaling can depend on batch composition. Appendix D gives the exact magnitude statistics, phase definition, norm reductions, and numerical safeguards. Parameter values are listed in Tables 9 and 10.

Evaluation-step allocation. We use a noise-progress-based allocation strategy to select B exact evaluation steps before sampling. After reserving the initial consecutive anchors, we map evenly spaced midpoint targets in logarithmic cumulative noise progress to sampling steps. The resulting anchor set is prompt independent and remains unchanged during generation. Appendix E provides the allocation procedure and the schedules used in the experiments.

Computational cost. Let D denote the number of elements in a velocity tensor. The method stores a fixed number of previously computed exact velocities and clean-state latent estimates, together with a running average of prediction residuals. Prediction and feedback use elementwise arithmetic and norm reductions, requiring $O(D)$ operations per sampling step and $O(D)$ additional memory. At skipped steps, the predicted velocity is passed to the native sampler without evaluating the denoiser.The complete sampling procedure is summarized in Algorithm 1.

4 EXPERIMENTS

4.1 IMAGE GENERATION

We evaluate VELOFORESIGHT on image and video generation against local implementations of Spectrum, TaylorSeer, and VDE-core, and analyze its control and feedback components.

Experimental setup. We evaluate FLUX.1-dev (Black Forest Labs, 2024) and SD3.5 Large (Stability AI, 2024) on the 200 prompts of DrawBench (Saharia et al., 2022). Table 1 compares VELOFORESIGHT with local implementations of Spectrum, TaylorSeer, and VDE-core. VELOFORESIGHT, Spectrum, and VDE-core use 10 or 14 exact evaluation steps, with VDE-core sharing the VeloForesight image anchors. TaylorSeer uses $N = 4/6$, $O = 1$, and $W = 5$. PSNR, SSIM, and LPIPS measure fidelity to each backbone's Full50 outputs. ImageReward (Xu et al., 2023) and CLIP (Radford et al., 2021) provide complementary preference and text–image alignment scores. Speedups are measured relative to matched local Full50 sampling.

Table 2: **Video-generation results on VBench.** Local results at the indicated operating points. Fidelity is measured against each backbone's Full50 reference. Speedups are reported relative to local Full50 sampling. Quality is reported as a percentage. Within each block and backbone, the best and second-best values among accelerated methods for each metric are shown in red and blue, respectively.

	Wan2.1-14B					**HunyuanVideo**				
	Acceleration	Fidelity			VBench	Acceleration	Fidelity			VBench
Method	Speedup↑	PSNR↑	SSIM↑	LPIPS↓	Quality↑	Speedup↑	PSNR↑	SSIM↑	LPIPS↓	Quality↑
Full50	1.000×	–	–	–	83.141	1.000×	–	–	–	84.635
TaylorSeer ($N = 4, O = 1$)	2.349×	20.024	0.7021	0.2178	**83.007**	1.450×	23.786	0.8115	0.1643	**84.650**
VDE-core + VF anchors (NFE13)	3.674×	20.314	0.7187	0.1778	82.349	3.598×	24.373	0.8106	0.1487	80.762
Spectrum (NFE14)	3.376×	23.055	0.7551	0.1578	82.715	3.349×	26.867	0.8480	0.1238	84.048
VELOFORESIGHT (NFE13)	**3.685×**	**23.853**	**0.7820**	**0.1345**	82.867	**3.603×**	**27.563**	**0.8716**	**0.1116**	84.050
TaylorSeer ($N = 6, O = 1$)	2.970×	17.713	0.6322	0.2960	81.603	1.642×	21.136	0.7438	0.2310	**84.117**
VDE-core + VF anchors (NFE9)	5.171×	17.697	0.6036	0.2824	79.286	4.994×	22.157	0.7437	0.2320	78.689
Spectrum (NFE10)	4.670×	21.839	0.6940	0.2088	80.851	4.540×	**25.053**	0.7999	**0.1718**	82.914
VELOFORESIGHT (NFE9)	**5.178×**	**22.187**	**0.7279**	**0.1903**	**82.437**	**5.038×**	25.030	**0.8215**	0.1783	83.528

Results. Relative to Spectrum, VELOFORESIGHT improves PSNR and SSIM on FLUX.1-dev at both budgets, with slightly higher LPIPS. On SD3.5 Large, it improves all three fidelity metrics at 10 evaluations. At 14 evaluations, SSIM and LPIPS improve, while PSNR decreases slightly from 17.890 to 17.827 dB. At 10 evaluations, VELOFORESIGHT achieves $4.74\times$ and $4.76\times$ speedups on FLUX.1-dev and SD3.5 Large, respectively. Compared with TaylorSeer at the corresponding operating points, VELOFORESIGHT achieves higher speedups and improves all three fidelity metrics on both backbones. Under shared anchors, it also achieves higher PSNR and SSIM than VDE-core at both budgets. LPIPS favors VELOFORESIGHT on FLUX.1-dev and VDE-core on SD3.5 Large. ImageReward and CLIP exhibit method-dependent rankings, highlighting complementary trade-offs beyond reference fidelity.

4.2 VIDEO GENERATION

Experimental setup. We evaluate VELOFORESIGHT on Wan2.1-14B and HunyuanVideo using 946 prompt occurrences from VBench, with 50-step sampling as the reference. We consider two configurations with 9 and 13 exact model evaluations and conduct our timing measurements on NVIDIA RTX PRO 6000 GPUs. Alongside PSNR, SSIM, and LPIPS, we report the VBench Quality Score (%), computed using the official normalization and weights over its seven quality dimensions.

Results. Table 2 presents the video-generation results. With only nine exact evaluations, VELOFORESIGHT achieves $5.18\times$ speedup on Wan2.1-14B and $5.04\times$ on HunyuanVideo. Increasing the budget to 13 evaluations improves all three fidelity metrics on both backbones, while retaining speedups of $3.69\times$ and $3.60\times$, respectively. At this setting, PSNR reaches 23.85 dB on Wan2.1-14B and 27.56 dB on HunyuanVideo, compared with 22.19 dB and 25.03 dB at nine evaluations. These results demonstrate a consistent trade-off between sampling efficiency and reference fidelity across the two video models.

Sensitivity to the anchor schedule. We compare three anchor schedules for the VDE adaptation at NFE9 and NFE13 on 32 paired Wan prompts. The VeloForesight anchors yield the best PSNR, SSIM, and LPIPS for VDE among the tested schedules at both budgets. Under these shared anchors, VELOFORESIGHT retains higher reference fidelity at similar latency, suggesting that the observed gap is not attributable to an unfavorable choice among the tested VDE schedules. Full schedules and results are provided in Appendix H.

4.3 ABLATION STUDIES

Structured reliability control. We evaluate the control rule in Section 3.3 on 32 Wan prompts at nine exact sampling steps (logical NFE9). All variants use the normalized scheduler-noise progress

coordinate and mean-interval curvature normalization of the primary Wan experiment. Horizon attenuation remains a function of the elapsed sampling-step count.

The variants differ in how they control the trend and curvature increments before feedback. Raw proposal applies neither shrinkage nor norm bounds. Uniform shrinkage applies the same gain to both terms, while Horizon-only additionally decreases this shared gain with the prediction horizon. Order-only assigns separate gains to trend and curvature but applies neither horizon attenuation nor norm bounds. Structured control uses the complete rule in Eq. 10. The final variant additionally enables state feedback and gain modulation from Section 3.4. All variants use the same exact evaluation schedule and retain the same residual-calibration rule.

We measure prediction accuracy at shared donor latent states and with shared exact-anchor histories. At each skipped step, an additional model evaluation provides the exact guided velocity $v_s^\star = F_\theta(x_s, t_s, c)$. For each variant, $\widehat{v}_s$ denotes its final scheduler-facing velocity, including residual calibration and any enabled state feedback and gain modulation. We define

$$E_v(s) = \frac{\|\widehat{v}_s - v_s^\star\|_2}{\max(\|v_s^\star\|_2, 10^{-8})}. \tag{17}$$

The diagnostic covers all 41 skipped steps per prompt, giving 1,312 shared prompt–step targets. Neither the diagnostic evaluations nor the shadow predictions update the donor trajectory. We average errors over skipped steps within each prompt and then equally across the 32 prompts. Separately, each variant generates videos along its own trajectory, yielding 192 videos in total. PSNR, SSIM, and LPIPS are computed against the corresponding 50-step references. Full settings are provided in Appendix F.

Table 3 shows that uniform shrinkage substantially improves upon the raw proposal. Compared with uniform shrinkage, structured control reduces E_v from 0.2116 to 0.1181 and increases PSNR from 21.08 to 24.41 dB. It also improves all four reported metrics over the tested horizon-only and order-only controls. Adding state feedback and gain modulation further improves SSIM and LPIPS.

We additionally compare structured control with a baseline using one shared gain and a single norm cap, with parameters selected on a development set. On the 32-prompt Wan NFE9 evaluation, structured control improves PSNR by 0.691 dB and SSIM by 0.0184, and reduces LPIPS by 0.0083 relative to this baseline. The paired 95% confidence intervals (CIs) exclude zero for all three metrics (Appendix F, Table 13).

Table 3: **Control and feedback study on Wan at NFE9.** Results are averaged over 32 prompts. All variants use the primary Wan prediction coordinate and curvature normalization. E_v measures the final velocity error after all enabled feedback at shared donor states. PSNR, SSIM, and LPIPS measure closed-loop fidelity to the 50-step references. All rows retain residual calibration. The final row additionally enables state feedback and gain modulation. The best value in each column is shown in bold.

Variant	$E_v \downarrow$	PSNR↑	SSIM↑	LPIPS↓
Raw proposal	0.5560	12.23	0.4008	0.6059
Uniform shrinkage	0.2116	21.08	0.6888	0.2887
Horizon-only	0.1618	22.29	0.7206	0.2516
Order-only	0.2313	20.54	0.6732	0.3119
Structured control	**0.1181**	**24.41**	0.7725	0.1788
+ State feedback	0.1262	24.39	**0.7768**	**0.1603**

Controlled predictor swap. We compare VELOFORESIGHT and Hold-last on 200 paired Wan2.1-14B occurrences at logical NFE10. Both use the same anchor schedule, scheduler, prompts, seeds, and Full50 references. Residual calibration, state feedback, gain modulation, and relative-norm bounds use identical settings and only the velocity proposal changes. Hold-last is a zero-order predictor that reuses the latest exact guided velocity as its proposal at each skipped step.

Table 4 shows a fidelity trade-off. Hold-last improves PSNR by 0.559 dB (95% paired bootstrap CI: [0.473, 0.646]), whereas VELOFORESIGHT reduces LPIPS by 0.0156 (CI: [0.0116, 0.0197] for

the reduction). The SSIM difference is inconclusive. VELOFORESIGHT also has a slightly higher VBench Quality point estimate (80.860 versus 80.634).

The stage-wise diagnostic in the primary Wan coordinate shows that structured admission reduces the raw proposal's magnitude and same-state velocity error (Appendix G). Mean velocity-change alignment nevertheless remains negative. This diagnostic measures local prediction accuracy at fixed donor states, whereas the paired predictor swap above measures the proposal's effect on final video fidelity.

Table 4: **Controlled predictor swap on Wan2.1-14B.** Means over 200 paired occurrences at logical NFE10. Each occurrence uses 20 physical DiT forward calls because classifier-free guidance (Ho & Salimans, 2022) evaluates the conditional and unconditional branches separately. VBench Quality is the normalized seven-dimension aggregate.

Predictor	PSNR↑	SSIM↑	LPIPS↓	VBench Quality↑	Latency (s)↓
VELOFORESIGHT	22.400957	0.756832	0.174781	80.860	136.9886
Hold-last	22.959968	0.758340	0.190372	80.634	136.9158

Robustness to initial noise. We further evaluate 32 prompts with seeds 0, 1, and 2 on both video backbones. Results are averaged over seeds within each prompt, and 95% intervals are estimated using 10,000 paired prompt-bootstrap replicates. The prompt set is shared with the controlled study, while the additional seeds test sensitivity to initial noise.

Table 5 shows that at 13 exact evaluations, all six confidence intervals favor VELOFORESIGHT. At nine evaluations, SSIM remains favorable on both backbones, while the PSNR and LPIPS intervals include zero. These results provide stronger support for the 13-evaluation operating point after averaging over the three tested initial-noise seeds.

Table 5: **Paired gains across three initial-noise seeds.** Mean differences are VELOFORESIGHT minus locally reproduced Spectrum over 32 prompts, after averaging seeds 0, 1, and 2 within each prompt. NFE lists the VELOFORESIGHT/Spectrum budgets.

		Wan2.1-14B		**HunyuanVideo**	
NFE	Metric	Mean Δ	95% CI	Mean Δ	95% CI
9/10	PSNR↑	$+0.1720$	$[-0.1750, +0.5640]$	$+0.0080$	$[-0.3920, +0.4350]$
	SSIM↑	$\mathbf{+0.0333}$	$[+0.0206, +0.0495]$	$\mathbf{+0.0225}$	$[+0.0121, +0.0337]$
	LPIPS↓	-0.0085	$[-0.0182, +0.0000]$	$+0.0004$	$[-0.0075, +0.0085]$
13/14	PSNR↑	$\mathbf{+0.8980}$	$[+0.4700, +1.4050]$	$\mathbf{+0.5770}$	$[+0.2590, +0.8940]$
	SSIM↑	$\mathbf{+0.0278}$	$[+0.0137, +0.0497]$	$\mathbf{+0.0241}$	$[+0.0158, +0.0335]$
	LPIPS↓	$\mathbf{-0.0177}$	$[-0.0267, -0.0100]$	$\mathbf{-0.0132}$	$[-0.0204, -0.0065]$

CI: confidence interval from 10,000 paired prompt-bootstrap resamples. Bold means indicate unrounded intervals entirely on the favorable side of zero. PSNR differences are in dB.

5 CONCLUSION

VELOFORESIGHT separates sparse-anchor velocity proposal construction from structured reliability control at the sampler interface. It assigns trend and curvature distinct gains and bounds and attenuates curvature with horizon. Paired 946-occurrence and prompt-clustered multi-seed results support NFE13 on Wan2.1-14B and HunyuanVideo, while a controlled Wan intervention supports the complete rule over the tested fixed uniform, horizon-only, and order-only alternatives. The more aggressive Hunyuan NFE9 point and the auxiliary feedback path remain metric dependent. Within these boundaries, VELOFORESIGHT provides a practical way to control complete velocity changes under a fixed exact-forward budget without training.

6 AI USE DISCLOSURE

We used generative AI tools to assist with literature retrieval, research ideation, experiment design, code implementation and auditing, result analysis, and language editing. The authors reviewed and verified all AI-assisted code, analyses, claims, and manuscript text, and take full responsibility for the final submission.

REFERENCES

Divya Jyoti Bajpai, Shubham Agarwal, Apoorv Saxena, Kuldeep Sharad Kulkarni, Subrata Mitra, and Manjesh Kumar Hanawal. FlowCast: Trajectory forecasting for scalable zero-cost speculative flow matching. In *International Conference on Learning Representations*, 2026a. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/93e06b762457778f3e9832b9c90d5ad2-Abstract-Conference.html.

Divya Jyoti Bajpai, Dhruv Bhardwaj, Soumya Roy, Tejas Duseja, Harsh Agarwal, et al. FastFlow: Accelerating the generative flow matching models with bandit inference. In *International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=wWkyL8D9xd.

Black Forest Labs. FLUX.1-dev model card. https://huggingface.co/black-forest-labs/FLUX.1-dev, 2024.

Hanshuai Cui, Zhiqing Tang, Zhi Yao, Fanshuai Meng, Weijia Jia, and Wei Zhao. Not all frames deserve full computation: Accelerating autoregressive video generation via selective computation and predictive extrapolation. *arXiv preprint arXiv:2604.02979*, 2026. URL https://arxiv.org/abs/2604.02979.

Huanlin Gao, Ping Chen, Fuyuan Shi, Ruijia Wu, Yantao Li, et al. MeanCache: From instantaneous to average velocity for accelerating flow matching inference. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/62868cc2fc1eb5cdf321d05b4b88510c-Abstract-Conference.html.

Xiaoliu Guan et al. Forecasting when to forecast: Accelerating diffusion models with confidence-gated taylor. *arXiv preprint arXiv:2508.02240*, 2025. URL https://arxiv.org/abs/2508.02240.

Jiaqi Han, Juntong Shi, Puheng Li, Haotian Ye, Qiushan Guo, and Stefano Ermon. Adaptive spectral feature forecasting for diffusion sampling acceleration. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/papers/Han_Adaptive_Spectral_Feature_Forecasting_for_Diffusion_Sampling_Acceleration_CVPR_2026_paper.pdf.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. URL https://arxiv.org/abs/2207.12598.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, et al. VBench: Comprehensive benchmark suite for video generative models. *arXiv preprint arXiv:2311.17982*, 2023. URL https://arxiv.org/abs/2311.17982.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, et al. HunyuanVideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024. URL https://arxiv.org/abs/2412.03603.

Yanchao Li, Jiaqing Xie, Ben Gao, Wanhao Liu, Yanbo Wang, T. Y. Tsui, Jinfei Liu, Yuqiang Li, and Tianfan Fu. Disagree to accelerate: Closing the loop on diffusion feature forecasts. *arXiv preprint arXiv:2608.01740*, 2026. URL https://arxiv.org/abs/2608.01740.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.02747.

Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, et al. Timestep embedding tells: It's time to cache for video diffusion model. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025a. URL https://openaccess.thecvf.com/content/CVPR2025/html/Liu_Timestep_Embedding_Tells_Its_Time_to_Cache_for_Video_Diffusion_CVPR_2025_paper.html.

Jiacheng Liu, Chang Zou, Yuanhuiyi Lyu, Junjie Chen, and Linfeng Zhang. From reusing to forecasting: Accelerating diffusion models with TaylorSeers. In *IEEE/CVF International Conference on Computer Vision*, 2025b. URL https://openaccess.thecvf.com/content/ICCV2025/papers/Liu_From_Reusing_to_Forecasting_Accelerating_Diffusion_Models_with_TaylorSeers_ICCV_2025_paper.pdf.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2209.03003.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-Solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. In *Advances in Neural Information Processing Systems*, 2022.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision*, 2023. URL https://arxiv.org/abs/2212.09748.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Raphael Gontijo-Lopes, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. In *Advances in Neural Information Processing Systems*, volume 35, pp. 36479–36494, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/ec795aeadae0b7d230fa35cbaf04c041-Abstract-Conference.html.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

Zhirong Shen et al. Accelerating diffusion transformers with gaussian process rectified feature cache. *arXiv preprint arXiv:2609.05981*, 2026. URL https://arxiv.org/abs/2609.05981. Accepted by ECCV 2026.

Stability AI. Stable Diffusion 3.5 Large model card. https://huggingface.co/stabilityai/stable-diffusion-3.5-large, 2024.

Junwen Tan, Jinglin Liang, Hongyuan Chen, and Shuangping Huang. VDE: Training-free accelerating rectified flow model via velocity decomposition and estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/papers/Tan_VDE_Training-Free_Accelerating_Rectified_Flow_Model_via_Velocity_Decomposition_and_CVPR_2026_paper.pdf.

Wan Team, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. URL https://arxiv.org/abs/2503.20314.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13 (4):600–612, 2004. URL https://ece.uwaterloo.ca/~z70wang/publications/ssim.html.

Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. ImageReward: Learning and evaluating human preferences for text-to-image generation. In *Advances in Neural Information Processing Systems*, volume 36, pp. 15903–15935, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/33646ef0ed554145eab65f6250fab0c9-Abstract-Conference.html.

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. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_The_Unreasonable_Effectiveness_CVPR_2018_paper.html.

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. UniPC: A unified predictor-corrector framework for fast sampling of diffusion models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 49842–49869, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/9c2aa1e456ea543997f6927295196381-Abstract-Conference.html.

Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-time video generation with pyramid attention broadcast. *arXiv preprint arXiv:2408.12588*, 2024. URL https://arxiv.org/abs/2408.12588.

Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=yYZbZGo4ei.

Chang Zou, Changlin Li, Songtao Liu, Zhao Zhong, Kailin Huang, and Linfeng Zhang. DisCa: Accelerating video diffusion transformers with distillation-compatible learnable feature caching. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026. URL https://github.com/Tencent-Hunyuan/DisCa.

A Execution details

Exact evaluation counts. Throughout the paper, NFE counts exact sampling steps with the configured guidance. Wan evaluates the conditional and unconditional branches separately, so an NFE-B run uses $2B$ physical DiT calls. Hunyuan uses B transformer calls. Thus the primary NFE9/13 settings correspond to 18/26 physical calls on Wan and 9/13 calls on Hunyuan. Prediction and feedback do not introduce additional transformer calls. Observer calls used in diagnostics are additional experimental computations and are excluded only from the deployed sampler's NFE.

Executed schedules and output identity. The primary schedules are listed in Appendix E. Each completed FULL946 arm associates 946 occurrence indices with seed 42, the prompt manifest, executed anchors, evaluation counts, output hashes, and the source/configuration identities covered by its execution records. The older NFE10 nonuniform schedule is not attributed to the formal SPGS allocation.

Causal state. Skipped forecasts do not update the exact-velocity history, residual statistics, or exact-anchor clean-latent records. The native solver nevertheless updates its latent and internal solver history at every sampling step. State feedback receives the raw runtime sigma. Its post-final-anchor phase uses the scale H_{tail} defined in Table 9 and this phase calculation does not add an anchor or stop clean-latent extrapolation.

Image-generation configuration. We evaluate FLUX.1-dev and SD3.5 Large on DrawBench200, using seed 42 for every prompt, 1024×1024 resolution, `bfloat16` precision, and 50 FlowMatch sampling steps. FLUX.1-dev uses guidance scale 3.5 with dynamic shifting enabled; SD3.5 Large uses guidance scale 7.0 with dynamic shifting disabled. For both backbones, VeloForesight and the shared-anchor VDE adaptation use the zero-based exact-anchor indices $[0, 1, 2, 3, 4, 6, 11, 19, 30, 44]$ at NFE10 and $[0, 1, 2, 3, 4, 6, 8, 11, 15, 20, 25, 31, 38, 46]$ at NFE14. TaylorSeer uses $O = 1$ and $W = 5$, with $N = 4/6$ yielding 16/12 full block refreshes and 50 outer transformer calls per image.

Table 6: **Primary-generation configuration.** Checkpoint revisions and evaluator source identities are retained in the supplement. Timing is measured separately from multi-GPU generation.

Setting	Wan2.1-14B	HunyuanVideo
Checkpoint	Wan2.1-T2V-14B-Diffusers	HunyuanVideo community checkpoint
Native scheduler	UniPC (Zhao et al., 2023), flow prediction	FlowMatchEulerDiscreteScheduler
Precision	bfloat16	bfloat16
Resolution	832×480	960×544
Frames / fps	81 / 16	61 / 24
Guidance scale	5.0	6.0
Native steps / seed	50 / 42	50 / 42
Generation hardware	3× RTX PRO 6000	3× RTX PRO 6000

Image timing and aggregation. Evaluation metrics are averaged over the 200 DrawBench prompts. Speedup is the ratio of the mean Full50 generation time to the mean accelerated generation time under the corresponding hardware configuration and timing protocol. For FLUX.1-dev, speedups are computed using a mean Full50 generation time of 15.680 s/image for TaylorSeer and VDE-core, and 15.878 s/image for VELOFORESIGHT and Spectrum. For SD3.5 Large, the corresponding Full50 reference times are 13.788 s/image for TaylorSeer and VDE-core, and 14.131 s/image for VELOFORESIGHT and Spectrum.

B PREDICTION COORDINATES AND IMPLEMENTATION VARIANTS

Wan coordinate construction. The prediction helper reads the first S entries of `scheduler.sigmas`, where S is the number of sampling timesteps. It converts them to a one-dimensional float64 sequence for coordinate construction. A terminal sigma at index S is excluded. Coordinate entry s is used with sampling-loop index s and the numerical value of the timestep is not used as an array index.

For $S > 1$ and a finite, nondegenerate sigma sequence, we construct the prediction coordinate as

$$\widetilde{\zeta}_s = \mathrm{clip}\left(\frac{\sigma_s - \sigma_0}{\sigma_{S-1} - \sigma_0}, 0, 1\right), \qquad \zeta_s = \max_{0 \le j \le s} \widetilde{\zeta}_j. \tag{18}$$

The cumulative maximum makes the coordinate nondecreasing. The endpoints are explicitly set to $\zeta_0 = 0$ and $\zeta_{S-1} = 1$. A decreasing sigma sequence already gives increasing progress through the signed endpoint normalization.

An empty timestep sequence raises an error. A single timestep receives coordinate zero. If the absolute endpoint span is at most $\varepsilon_{\mathrm{coord}}$, the helper returns the uniform coordinate $s/(S-1)$. If sigma is unavailable, too short, or nonfinite, the helper attempts the same normalization using the timesteps. If that sequence is also unavailable, it uses uniform index progress.

Velocity history and small coordinate intervals. The Wan history stores triples (a, ζ_a, v_a) in execution order and retains the latest three exact observations. The stored velocity includes the configured guidance and is evaluated on the accelerated trajectory. Calibration is performed before the new observation is appended.

With one observation the predictor holds its velocity. With two, it uses the controlled trend. With three, the configured curvature predictor adds the curvature term. The Wan helper returns no prediction for an empty history, and its skipped-step caller raises an error in that case. The initial exact steps prevent this case in the reported schedules.

Each Wan coordinate interval is floored at $\varepsilon_{\mathrm{fd}}$ before computing the secant slopes. This flooring does not trigger an automatic lower-order fallback. In particular, the mean interval is formed from the already-floored consecutive intervals. If the target coordinate does not exceed the latest anchor coordinate, $d_s = 0$ and both proposal increments vanish.

The formal low-frequency ratio is one, for which the corresponding helper returns its input unchanged. Consequently, no additional spatial filtering is hidden between the proposals and their caps in the reported core predictor.

Table 7: **Predictor implementations used in different studies.** Prediction coordinates and curvature normalization are specified separately for each study.

Study	Prediction coordinate	Curvature normalization
FULL946 Wan NFE9/13	Normalized sigma progress	Mean interval
FULL946 Hunyuan NFE9/13	Sampling step	Latest interval, legacy
Experiment A, Wan	Normalized sigma progress	Mean interval
Wan controlled predictor swap (200 prompts, NFE10)	Normalized sigma progress	Mean interval (VF only)
Unified same-state diagnostic (Wan and Hunyuan)	Sampling step	Mean interval

Hunyuan curvature convention. For Hunyuan, we use sampling-step coordinates, $\zeta_s = s$, and latest-interval normalization, $\ell_a = g_1$. Let $h_1 = a_2 - a_1$ and $h_0 = a_1 - a_0$. The curvature proposal in Eq. 5 can equivalently be written as

$$C_s^{\text{legacy}} = \frac{1}{2}\left(\frac{\tau_s}{h_1}\right)^2 \left[(\nu_2 - \nu_1) - \frac{h_1}{h_0}(\nu_1 - \nu_0)\right]. \tag{19}$$

A mean-interval predictor in the same sampling-step coordinate would instead produce

$$C_s^{\text{mean}} = \frac{\tau_s^2}{h_0 + h_1}(m_1 - m_0). \tag{20}$$

The relationship is

$$C_s^{\text{legacy}} = \frac{h_0 + h_1}{2h_1} C_s^{\text{mean}}. \tag{21}$$

The expressions agree for equal consecutive anchor intervals, but generally differ for nonuniform intervals. Neither the mean-interval proposal nor the legacy proposal is claimed to be an exact quadratic interpolant through all three observations.

The Wan control ablation and controlled predictor swap use the same prediction coordinate and curvature normalization as the primary Wan experiments. The unified cross-backbone diagnostic uses sampling-step coordinates and mean-interval normalization for both backbones. Table 7 summarizes these study-specific conventions.

Scope of the implementation records. The primary formulas follow the recovered generation entrypoints and their bound prediction routines. The Wan path resolves to the scheduler-coordinate predictor, although its original source manifest did not include the coordinate helper and the subsequently recovered local and remote helper copies agree. The Hunyuan entrypoint and pipeline resolve to the step-based legacy branch, but the output metadata do not directly record a per-step branch trace. These distinctions concern historical execution evidence and they do not change the mathematical definitions above.

C Predictor and feedback parameters

Tables 8 and 9 give the recovered scalar settings for the primary Wan and Hunyuan video configurations. Structural constants in the equations, such as the factor $1/2$, are not independent tuning parameters.

Parameter roles and selection records. The trend and curvature gains set the relative contributions of the two proposal components. Their radii limit the corresponding increments, while H_C controls how quickly the curvature gain decreases with the elapsed sampling-step horizon. Residual and state parameters affect the auxiliary correction paths rather than the construction of T_s and C_s.

The pre-EMA radius $r_R/(\alpha_R\kappa_R)$ is derived from the listed residual parameters and is not an additional independent hyperparameter. Likewise, $1 - \beta_R$ is the EMA update weight, and r_X is shared

Table 8: **Shared primary-video settings for the structured predictor and residual calibration.** Horizon scales are measured in native sampling steps.

Symbol	Role	Value
Structured predictor		
α_T	Trend gain	0.30
α_C	Curvature gain	0.35
H_C	Curvature attenuation scale	12
r_T	Relative trend radius	0.12
r_C	Relative curvature radius	0.012
Residual calibration		
α_R	Residual correction gain	0.28
β_R	EMA retention coefficient	0.82
H_R	Residual attenuation scale	8
r_R	Relative residual correction radius	0.018
κ_R	Pre-EMA cap factor	0.75
θ_R	Minimum residual alignment	-0.05
ρ_{init}	Initial alignment state	0.5

Table 9: **Shared primary-video settings for state correction and gain modulation.** The two uses of r_X read the same configuration variable, `VELO_STATECONS_CLIP`.

Symbol	Role	Value
α_X	State correction coefficient	0.35
r_X	Pre-gain radius and magnitude scale	0.05
r_F	Final relative correction radius	0.065
g_+	Small-correction gain coefficient	0.45
g_-	Phase-dependent gain coefficient	0.35
$\gamma_{\min}$	Lower modulation bound	0.65
$\gamma_{\max}$	Upper modulation bound	1.40
H_{tail}	Post-final-anchor phase scale	4

between state pre-capping and CGA magnitude normalization. The formal final radius $r_F = 0.065$ comes from the bound launch configuration and it should not be replaced by a different source-code default.

The retained records establish the final settings but do not recover the original parameter-selection search space, objective, or prompt list. We therefore do not claim a verified five-prompt selection procedure or independence between an undocumented tuning set and the evaluation prompts. Existing local sensitivity results are separate from the original selection history. Experiment A compares the tested control configurations and it does not establish that every component is independently necessary or that the full rule outperforms an optimally tuned scalar shrinkage control.

D Feedback and numerical execution rules

Let $\text{clip}(z, l, u) = \min(u, \max(l, z))$. The radial notation in Eq. 8 expresses the common control operation above numerical safeguard thresholds. The norm reductions and the recovered exact numerical branches are distinguished below.

Wan core cap. For the recovered Wan core helper, define

$$R_W(\delta; b) = \frac{\|\delta\|_2}{\max(\|b\|_2, \varepsilon_{\text{base},W})}. \tag{22}$$

Its exact scalar branch is

$$\mathcal{C}_r^W(\delta; b) = \begin{cases} \delta, & r \le 0 \text{ or } R_W(\delta; b) \le r, \\ \delta \dfrac{r}{\max(R_W(\delta; b), \varepsilon_{\text{ratio},W})}, & \text{otherwise.} \end{cases} \tag{23}$$

Table 10: **Recovered numerical safeguards and schedule settings.** Each safeguard is associated with a specific operation. Equal values do not imply a shared configuration variable. The Wan core-helper guards are not asserted to be universal guards for all other cap implementations.

Symbol	Operation	Value
$\varepsilon_{\text{coord}}$	Wan endpoint-span test	10^{-12}
ε_{fd}	Wan coordinate-difference floor	10^{-8}
$\varepsilon_{\text{decay}}$	Wan curvature-scale floor	10^{-6}
$\varepsilon_{\text{base},W}$	Wan core reference-norm floor	10^{-6}
$\varepsilon_{\text{ratio},W}$	Wan core clipping-ratio floor	10^{-8}
$\varepsilon_{\Delta\sigma}$	Clean-record interval test, both backbones	10^{-8}
$\varepsilon_{\sigma,W}$	Wan state bypass and denominator floor	10^{-7}
$\varepsilon_{\sigma,H}$	Hunyuan state bypass and denominator floor	10^{-5}
ε_{CGA}	CGA norm and ratio safeguard, both backbones	10^{-8}
K_0	Initial consecutive exact steps	5
S	Primary native sampling steps	50
B	Primary exact evaluation budgets	$9, 13$

Norms are evaluated on float32 tensors over all dimensions, including the batch dimension. A nonpositive cap setting disables this helper's clipping and it does not force the increment to zero. The primary radii are positive.

For Wan, Eq. 10 uses this guarded operation. Its curvature attenuation evaluates the horizon scale as $\max(H_C, \varepsilon_{\text{decay}})$, which equals H_C under the reported setting. These recovered numerical branches are not used to infer unrecorded safeguards in other helpers.

Norm reductions across modules. The core predictor and residual calibration use whole-tensor L2 norms. The initial StateCons cap uses a whole-tensor norm on Wan and an independent norm for each video on Hunyuan. For a batch indexed by j, the latter reduces all non-batch dimensions of each video tensor. On video latents, these dimensions include channels, frames, height, and width.

Denote the initial state cap by $\mathcal{C}^b_{X,r}$. Its mathematical action is the radial operation in Eq. 8, applied globally for Wan and per video for Hunyuan:

$$[\mathcal{C}^H_{X,r}(\delta; b)]_j = \mathcal{C}_r(\delta_j; b_j). \tag{24}$$

CGA uses a different reduction, specified explicitly below. The reduction axes alone do not establish that the underlying helpers share the same numerical floors.

Residual observations and initialization. At an exact step a, let $v_a^{\text{core},-}$ be the core prediction reconstructed from the old exact-anchor history. It includes trend and curvature control but excludes the historical residual correction, StateCons, and CGA. The exact model output v_a is evaluated on the current accelerated latent.

The observed and pre-capped residuals are

$$e_a = v_a - v_a^{\text{core},-}, \qquad \widetilde{e}_a = \mathcal{C}_{r_R/(\alpha_R \kappa_R)}(e_a; v_a). \tag{25}$$

The first exact evaluation has no prediction history and does not update the residual. Under the reported consecutive startup schedule, the first update occurs at sampling step one. At this first valid update, initialize

$$\bar{e}_a = \widetilde{e}_a, \qquad \rho_a = \rho_{\text{init}}. \tag{26}$$

Residual updates. For subsequent exact evaluations, let

$$d_a = \cos_0(\widetilde{e}_a, \bar{e}_{a^-}), \tag{27}$$

where $\cos_0$ denotes cosine alignment with zero-vector alignment defined as zero. Once a residual average exists, a zero residual is not treated as perfectly aligned.

If $d_a \geq \theta_R$, update

$$\bar{e}_a = \beta_R \bar{e}_{a^-} + (1-\beta_R)\widetilde{e}_a, \tag{28}$$

$$\rho_a = \beta_R \rho_{a^-} + (1-\beta_R)\,\mathrm{clip}\left(\frac{d_a+1}{2}, 0, 1\right). \tag{29}$$

Otherwise,

$$\bar{e}_a = \bar{e}_{a^-}, \qquad \rho_a = \beta_R \rho_{a^-}. \tag{30}$$

The new exact velocity is appended to the anchor history only after this calibration. At skipped steps, Eq. 11 uses the latest residual state without updating it. Before a valid residual exists, the residual correction is zero. The alignment state is an auxiliary deterministic statistic, not a calibrated probability.

Clean-latent records. At an eligible exact evaluation, record (σ_a, χ_a) using Eq. 12. Wan skips both record creation and state correction when $\sigma \leq \varepsilon_{\sigma,W}$. Hunyuan's exact-record branch does not contain the same low-sigma guard and its skipped-step correction requires a nonempty record history and $\sigma > \varepsilon_{\sigma,H}$.

For two available records (σ_0, χ_0) and (σ_1, χ_1), ordered by evaluation time, define

$$\widehat{\chi}_s = \begin{cases} \chi_1 + \dfrac{\sigma_s - \sigma_1}{\sigma_1 - \sigma_0}(\chi_1 - \chi_0), & |\sigma_1 - \sigma_0| > \varepsilon_{\Delta\sigma}, \\ \chi_1, & |\sigma_1 - \sigma_0| \leq \varepsilon_{\Delta\sigma}. \end{cases} \tag{31}$$

With only one record, use that record. With no record, leave v_s^{base} unchanged.

The state velocity is given by Eq. 13. Its denominator is $\max(\sigma_s, \varepsilon_{\sigma,b})$, not $\max(|\sigma_s|, \varepsilon_{\sigma,b})$. The interval test in Eq. 31 retains the signed sigma difference when extrapolating; it does not replace the difference by a positive floor. After the final exact anchor, the same extrapolate-or-hold rule continues to use the latest records.

Initial state correction. When the correction branch is active, define

$$\lambda_s = \alpha_X\,\mathrm{clip}(1-\sigma_s, 0, 1), \tag{32}$$

$$\Delta_s^{\mathrm{prop}} = \lambda_s(v_s^{\mathrm{state}} - v_s^{\mathrm{base}}), \tag{33}$$

$$\Delta_s^{\mathrm{cap}} = \mathcal{C}^b_{X,r_X}(\Delta_s^{\mathrm{prop}}; v_s^{\mathrm{base}}). \tag{34}$$

The reference velocity is the residual-corrected velocity before StateCons. The per-video and whole-tensor reductions for this initial cap are specified above.

CGA magnitude statistic. Let N be the batch size. Both backbones compute

$$n_b = \frac{1}{N}\sum_{j=1}^{N} \max\left(\|v_{s,j}^{\mathrm{base}}\|_2, \varepsilon_{\mathrm{CGA}}\right), \qquad n_\Delta = \frac{1}{N}\sum_{j=1}^{N} \|\Delta_{s,j}^{\mathrm{cap}}\|_2. \tag{35}$$

Each norm reduces all non-batch dimensions. The reference-norm floor is applied separately to each video before averaging. The relative magnitude and its normalized value are

$$\eta_s = \frac{n_\Delta}{n_b}, \qquad q_s = \mathrm{clip}(\eta_s / r_X, 0, 1). \tag{36}$$

This is a ratio of mean norms, not a mean of per-video ratios. The normalization scale r_X and the StateCons pre-cap radius are the same configuration variable.

Phase and gain. Let a_s^- be the latest exact anchor and, when it exists, let a_s^+ be the next exact anchor. At a skipped step, define

$$\phi_s = \begin{cases} \dfrac{s - a_s^-}{a_s^+ - a_s^-}, & a_s^+ \text{ exists}, \\ \min\left(\dfrac{s - a_s^-}{H_{\mathrm{tail}}}, 1\right), & \text{otherwise}. \end{cases} \tag{37}$$

For the first branch, $a_s^- < s < a_s^+$ ensures a positive denominator and a phase between zero and one. The gain is

$$\gamma_s = \mathrm{clip}(1 + g_+(1 - q_s) - g_-\phi_s q_s, \gamma_{\min}, \gamma_{\max}) . \tag{38}$$

It acts only on the state-derived correction: $\Delta'_s = \gamma_s \Delta_s^{\mathrm{cap}}$.

Final state cap. After applying the gain, recompute

$$\eta'_s = \frac{\frac{1}{N}\sum_{j=1}^{N} \|\Delta'_{s,j}\|_2}{n_b}. \tag{39}$$

Then

$$\Delta_s^{\mathrm{final}} = \begin{cases} \Delta'_s, & \eta'_s \le r_F, \\ \Delta'_s \dfrac{r_F}{\max(\eta'_s, \varepsilon_{\mathrm{CGA}})}, & \eta'_s > r_F, \end{cases} \qquad \widehat{v}_s = v_s^{\mathrm{base}} + \Delta_s^{\mathrm{final}}. \tag{40}$$

The same scalar rescales the entire correction tensor. This operation is neither an independent cap for each video nor a whole-batch L2 cap. Its reference remains v_s^{base} throughout.

Input: initial latent x_0, native solver, anchor set $\mathcal{A}_B$, and empty exact-velocity, residual, and clean-latent records.
For $s = 0, \ldots, S-1$:

1. If $s \in \mathcal{A}_B$, evaluate the pretrained model with the configured guidance to obtain v_s. When old velocity history exists, reconstruct $v_s^{\mathrm{core},-}$, update the residual state, and then append the new exact observation. Update the clean-latent record when its recording condition is satisfied. Set $w_s = v_s$.
2. Otherwise, construct the backbone-specific trend and curvature proposals, apply Eq. 10, add residual correction, and apply the eligible state-feedback path. Set $w_s = \widehat{v}_s$. Leave all exact-anchor and residual records unchanged.
3. Apply the native update $(x_{s+1}, \xi_{s+1}) = \Psi_s(x_s, \xi_s, w_s)$.

Output: x_S. The number of exact evaluation steps is $|\mathcal{A}_B| = B$.

Algorithm 1: Exact-budget VELOFORESIGHT sampling

E EVALUATION-STEP ALLOCATION

SPGS constructs the anchor set before generation from the scheduler's noise sequence. The generator uses the first S runtime sigma values and excludes any terminal sigma after the final model-evaluation timestep.

Cumulative noise progress. Define

$$D_\sigma = \sum_{j=1}^{S-1} |\sigma_j - \sigma_{j-1}|, \qquad p_s = \frac{\sum_{j=1}^{s} |\sigma_j - \sigma_{j-1}|}{D_\sigma}, \qquad p_0 = 0. \tag{41}$$

The generator requires $D_\sigma > 0$ and explicitly sets $p_{S-1} = 1$. This cumulative-variation definition is distinct from the prediction helper in Eq. 18. The two can agree numerically on monotone sigma trajectories, but serve different purposes.

Midpoint targets. Reserve the initial anchors

$$\mathcal{A}_{\mathrm{start}} = \{0, \ldots, K_0 - 1\}. \tag{42}$$

The remaining candidates are $\mathcal{V} = \{K_0, \ldots, S-1\}$. Require $0 < p_{K_0} < 1$, and define

$$z_s = \ln \frac{p_s}{p_{K_0}}, \quad s \in \mathcal{V}, \qquad z_{\max} = \ln \frac{1}{p_{K_0}}. \tag{43}$$

For $K = B - K_0$ remaining anchors, the targets are

$$t_k = \frac{k + \frac{1}{2}}{K} z_{\max}, \qquad k = 0, \ldots, K-1. \tag{44}$$

These are interval midpoints in natural-log progress. There is no additional logarithmic offset or temperature.

Table 11: **Executed exact evaluation steps for the primary video experiments.** Indices are zero based.

Backbone	NFE	Exact evaluation steps
Wan	9	$[0, 1, 2, 3, 4, 7, 15, 27, 42]$
Wan	13	$[0, 1, 2, 3, 4, 6, 9, 12, 17, 23, 30, 38, 45]$
Hunyuan	9	$[0, 1, 2, 3, 4, 8, 18, 32, 45]$
Hunyuan	13	$[0, 1, 2, 3, 4, 6, 10, 15, 21, 28, 35, 42, 47]$

Discrete mapping and duplicate handling. For each target, the initial candidate is

$$a_k^{\mathrm{raw}} = \operatorname*{argmin}_{s \in \mathcal{V}} \big(|z_s - t_k|, s\big), \tag{45}$$

where the pair is ordered lexicographically, so equal distances favor the smaller step.

Targets are processed in increasing order. An unused candidate later than the previously selected anchor is accepted. Otherwise, the generator selects the nearest unused candidate that is later than the preceding selection, using the same tie-breaking rule. If no later unused candidate exists, it selects the nearest remaining unused candidate. Exhausting all unused candidates raises an error.

After mapping, the generator checks that the selected indices are strictly increasing and unique. It then prepends the startup anchors and checks that the final set contains exactly B distinct steps. Failure raises an error and the generator does not silently sort an invalid result or truncate excess anchors. Thus a successful allocation satisfies the budget and ordering requirements, without claiming successful allocation for every arbitrary budget.

Executed settings. The archived generator processes budgets $9, 10, 13, 14$ with $S = 50$ and $K_0 = 5$. The primary video experiments use the four sets in Table 11. Existing CPU verification reproduces these sets from the archived sigma sequences. None of these four sets requires duplicate repair. The last anchor follows from the last midpoint target and there is no separate forced late-refresh operation.

F Experiment A inference details

Experiment A uses 32 prompts with seed 42. All six variants use the normalized cumulative scheduler-noise progress coordinate and mean-interval curvature normalization of the primary Wan experiment. The zero-based exact-anchor schedule is

$$[0, 1, 2, 3, 4, 7, 15, 27, 42].$$

Each generated video uses nine exact sampling steps. Because Wan evaluates the conditional and unconditional classifier-free guidance branches separately, this corresponds to 18 physical DiT forward calls per video.

The same-state diagnostic contains 1,312 shared prompt–step targets and 7,872 variant–target pairs. At each skipped step, the variants share the donor latent, timestep, exact-anchor history, and diagnostic exact guided velocity. Each variant applies its configured control and feedback rules to produce the final scheduler-facing velocity used in Eq. 17. Diagnostic evaluations and shadow predictions do not modify the donor trajectory or its stored histories. Their additional model evaluations are excluded from the generation NFE and latency measurements.

Closed-loop evaluation contains 192 videos: one video per prompt for each of the six variants. Each variant maintains its own sampling trajectory and histories. Output fidelity is measured against the corresponding Full50 reference for the same prompt and seed. Same-state errors are first averaged over the 41 skipped steps within each prompt and then equally across prompts. Paired confidence intervals are computed by resampling the 32 prompts as clusters.

The values of $\alpha_T, \alpha_C, H_C, r_T, r_C$ are listed in Table 8. These comparisons evaluate complete control configurations. The transition from order-only to structured control introduces both horizon attenuation and norm bounds, so their individual contributions are not separately identified.

Table 12: **Intervention matrix for the Wan NFE9 ablation.** Expressions give velocity increments before the trend and curvature caps, which are applied sequentially when enabled. The horizon τ_s counts elapsed sampling steps. All variants retain residual calibration. State feedback includes gain modulation.

Variant	Pre-cap increment	Trend/curvature caps	State feedback
Raw	$T_s + C_s$	–	No
Uniform	$\alpha_T(T_s + C_s)$	–	No
Horizon-only	$\frac{\alpha_T}{1+\tau_s/H_C}(T_s + C_s)$	–	No
Order-only	$\alpha_T T_s + \alpha_C C_s$	–	No
Structured	$\alpha_T T_s + \frac{\alpha_C}{1+\tau_s/H_C} C_s$	$r_T,\ r_C$	No
Structured + feedback	$\alpha_T T_s + \frac{\alpha_C}{1+\tau_s/H_C} C_s$	$r_T,\ r_C$	Yes

Table 13: **Paired differences in the Wan NFE9 ablation.** Structured control (without state feedback) minus each comparison arm on 32 prompts, using normalized noise-progress coordinates.

Comparison arm	Metric	Mean Δ	95% CI
Raw proposal	PSNR↑	$+12.1810$	$[+11.0870, +13.2960]$
	SSIM↑	$+0.3717$	$[+0.3468, +0.3969]$
	LPIPS↓	-0.4271	$[-0.4587, -0.3950]$
Uniform shrinkage	PSNR↑	$+3.3300$	$[+2.6890, +3.9930]$
	SSIM↑	$+0.0837$	$[+0.0689, +0.0990]$
	LPIPS↓	-0.1099	$[-0.1294, -0.0912]$
Horizon-only	PSNR↑	$+2.1200$	$[+1.5950, +2.6930]$
	SSIM↑	$+0.0519$	$[+0.0412, +0.0634]$
	LPIPS↓	-0.0728	$[-0.0856, -0.0604]$
Order-only	PSNR↑	$+3.8740$	$[+3.1930, +4.5880]$
	SSIM↑	$+0.0992$	$[+0.0838, +0.1150]$
	LPIPS↓	-0.1331	$[-0.1546, -0.1127]$
Simple gain + cap	PSNR↑	$+0.6910$	$[+0.2530, +1.2080]$
	SSIM↑	$+0.0184$	$[+0.0126, +0.0244]$
	LPIPS↓	-0.0083	$[-0.0157, -0.0005]$

CI: confidence interval from paired prompt bootstrap. Positive PSNR/SSIM differences and negative LPIPS differences favor Structured control. PSNR differences are in dB.

Simple gain + cap. We additionally compare structured control with a shared gain and a single relative-norm cap on the combined proposal:

$$v_s^{\text{simple,core}} = \nu_2 + \mathcal{C}_r(\alpha(T_s + C_s); \nu_2)\,.$$

This variant uses neither separate trend and curvature gains nor horizon attenuation. It retains residual calibration, with state feedback and gain modulation disabled, as in the structured-control arm.

We evaluate all nine combinations of $\alpha \in \{0.1, 0.3, 1.0\}$ and $r \in \{0.03, 0.06, 0.15\}$ on eight development prompts. The development and evaluation sets have no overlapping prompt texts. We select $\alpha = 1.0$ and $r = 0.15$, which achieve the lowest prompt-averaged development LPIPS of 0.24258, and fix these values for the 32-prompt evaluation. Evaluation-set results are not used for parameter selection. The prediction coordinate, exact-anchor schedule, scheduler, and reference-pairing protocol match the six-arm study.

Table 14: **Selected same-state relative velocity errors.** Each cell contains eight prompts and 296 prompt–step observations. The structured row uses the common step-coordinate, mean-interval diagnostic predictor. Adapted baselines are evaluated at the same velocity interface and are not official end-to-end implementations.

Predictor	Wan-Spec	Wan-VF	HY-Spec	HY-VF
FastFlow Eq. adaptation	0.18005	0.17802	0.04942	0.04587
VDE Eq. adaptation	0.18458	0.18603	0.08773	0.08923
Structured	0.13583	0.13392	0.03836	0.03589
Hold-last	0.12481	0.12194	0.04010	0.03779

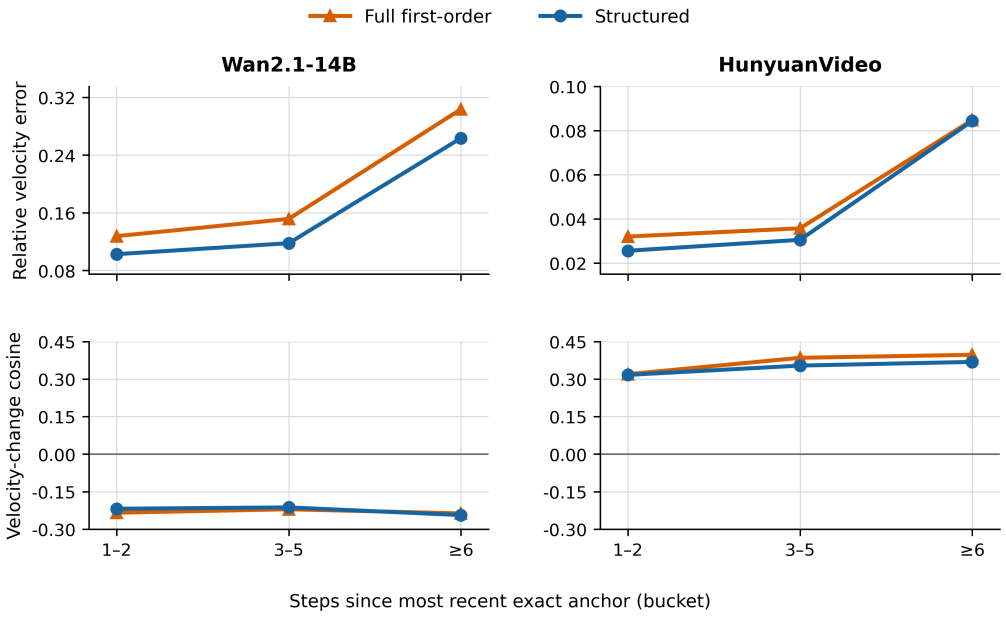


Figure 2: **Same-state horizon diagnostics.** The diagnostic uses the common step-coordinate predictor at NFE13. Columns show Wan2.1-14B and HunyuanVideo. The top row reports relative velocity error against the exact guided velocity at the same donor state. The bottom row reports cosine similarity between predicted and exact velocity changes from the latest exact anchor. Points are prompt-equal means over eight prompts per donor (Spectrum and VELOFORESIGHT), grouped by elapsed sampling steps since the latest exact anchor. Lines connect bucket means as visual guides.

G CONTROLLED PREDICTOR DIAGNOSTICS

Common-state protocol. This diagnostic is separate from the normalized-coordinate ablation in Appendix F. The unified same-state diagnostic uses a sampling-step coordinate and mean-interval curvature on both backbones. Its common NFE13 schedule is $[0, 1, 2, 3, 4, 6, 8, 11, 15, 20, 26, 33, 41]$. This is a diagnostic implementation, distinct from the primary-generation implementations in Table 7.

For a donor latent x_s at a skipped step, the observer evaluates the exact target $v_s^\star = F_\theta(x_s, t_s, c)$ without modifying the donor trajectory. We measure relative velocity error using Eq. 17, with $\widehat{v}_s$ given by the diagnostic predictor evaluated here. Each prompt is first averaged over its 37 skipped steps, and prompts are then weighted equally. Observer calls are absent from deployment and do not update the latent, solver history, residual state, or anchor history.

Shared exact-anchor history applies to the common-state diagnostic. In closed-loop generation, each method obtains its own exact observations along its own trajectory. Figure 2 reports relative velocity error and velocity-change alignment across prediction horizons.

No predictor dominates these local errors across all settings.

Table 15: **Stage-wise same-state diagnostic on eight Wan prompts in the primary normalized-coordinate NFE9 configuration.** R_Δ measures increment magnitude, not prediction accuracy. The zero-order cosine is undefined because its increment is zero.

Stage	$E_v \downarrow$	Cosine↑	R_Δ
Zero-order holding	0.10821	–	0.00000
Raw proposal	0.59273	-0.17119	0.54643
Structured admission	0.12585	-0.19020	0.04271
Residual corrected	0.12607	-0.18143	0.04351
Final state feedback	0.13213	-0.16960	0.05444

Controlled predictor-swap protocol. We use the first 200 prompts in the frozen Wan FULL946 manifest order, with seed 42 and one sample per prompt. Both predictors use normalized scheduler-noise progress and the zero-based exact-anchor indices $[0, 1, 2, 3, 4, 5, 11, 19, 30, 44]$. The structured proposal uses mean-interval curvature normalization, while Hold-last reuses the latest exact guided velocity without extrapolation. The 50-step UniPC scheduler, model, resolution, frame count, and control parameters follow the primary Wan configuration. Residual calibration, state feedback, gain modulation, and norm bounds are identical between the two arms and only the velocity proposal is replaced. Each run uses ten exact sampling steps and twenty physical DiT forward calls. Outputs are paired with the same prompt, seed, and Full50 reference.

Stage-wise diagnostic in the primary Wan coordinate. We additionally evaluate the normalized cumulative noise-progress configuration used by the primary Wan pipeline. The diagnostic uses the first eight prespecified prompts from the frozen 32-prompt manifest, all with seed 42. The exact anchors are $[0, 1, 2, 3, 4, 7, 15, 27, 42]$, giving logical NFE9 and 18 physical DiT forward calls per donor rollout.

All stages are evaluated along the same frozen VeloForesight donor trajectory with state feedback enabled. They share the donor latent and exact-anchor history at each of the 41 skipped steps, yielding 328 prompt–step targets. Additional exact evaluations provide the target guided velocities without updating the donor trajectory or its histories and these observer calls are excluded from the generation NFE.

For a stage output u_s, we report the relative velocity error defined in Eq. 17, the cosine between $u_s - \nu_2$ and $v_s^\star - \nu_2$, and the relative increment magnitude $R_\Delta = \|u_s - \nu_2\|_2 / \|\nu_2\|_2$, where ν_2 is the latest exact-anchor velocity. Each metric is averaged within each prompt and then equally across the eight prompts. Undefined cosines are excluded.

Table 15 shows that structured admission reduces the raw proposal's relative increment magnitude from 0.54643 to 0.04271 and its relative velocity error from 0.59273 to 0.12585. Mean directional alignment remains negative and does not improve. Zero-order holding achieves the lowest local error, and neither residual calibration nor state feedback further reduces this error in the diagnostic. These measurements show suppression of large extrapolation increments on the tested donor states. They do not determine the endpoint fidelity of independently evolving sampling trajectories.

H SENSITIVITY OF THE VDE ADAPTATION TO ANCHOR PLACEMENT

Setup. We evaluate anchor sensitivity on Wan2.1-14B using the first 32 prompt–seed pairs from the main evaluation manifest. At each of the NFE9 and NFE13 budgets, we compare the VDE adaptation under three anchor schedules and include VELOFORESIGHT under its main evaluation schedule. All configurations use the same prompts, seeds, initial-latent generation protocol, backbone, native scheduler, and Full50 references, with 50 sampling updates. Within the VDE comparison, only anchor placement changes. The decomposition, coefficient estimation, and history-update rules remain fixed.

Anchor schedules. The shared-anchor configuration uses the same exact evaluation steps as VELOFORESIGHT at the corresponding budget. The alternative schedules are defined below using zero-based sampling-step indices:

Table 16: **Anchor sensitivity on Wan2.1-14B.** Results use the same 32 paired samples at each budget. PSNR, SSIM, and LPIPS measure fidelity to the paired Full50 references and latency is the mean generation time per video. "VF anchors" denotes the corresponding main evaluation schedule of VELOFORESIGHT. Bold values indicate the best fidelity within each budget.

Predictor / schedule	PSNR↑	SSIM↑	LPIPS↓	Latency (s)↓
NFE9				
VELOFORESIGHT / VF anchors	**24.400**	**0.7774**	**0.1602**	122.167
VDE / VF anchors	19.477	0.6625	0.2423	123.963
VDE / A1	17.651	0.6157	0.3367	123.044
VDE / A2	18.050	0.6199	0.3579	123.078
NFE13				
VELOFORESIGHT / VF anchors	**26.157**	**0.8292**	**0.1120**	174.881
VDE / VF anchors	21.901	0.7663	0.1540	175.633
VDE / B1	20.811	0.7282	0.1995	174.771
VDE / B2	20.674	0.7245	0.1894	173.898

- **A1 (NFE9):** five startup evaluations followed by approximately uniform step spacing, $[0, 1, 2, 3, 4, 15, 27, 38, 49]$.
- **A2 (NFE9):** five startup evaluations followed by uniform spacing in cumulative absolute scheduler-noise progress, including the final sampling step, $[0, 1, 2, 3, 4, 25, 37, 44, 49]$.
- **B1 (NFE13):** five startup evaluations followed by approximately uniform step spacing, $[0, 1, 2, 3, 4, 10, 15, 21, 27, 32, 38, 43, 49]$.
- **B2 (NFE13):** nine startup evaluations followed by approximately periodic refreshes, $[0, 1, 2, 3, 4, 5, 6, 7, 8, 18, 28, 38, 49]$.

Each schedule is fixed across the 32 evaluated occurrences and is evaluated with the adapted VDE predictor.

Results. Table 16 shows that the VeloForesight anchors give VDE the highest PSNR and SSIM and the lowest LPIPS among the tested schedules at both budgets. Under these shared anchors, VELOFORESIGHT improves PSNR over VDE by 4.923 and 4.256 dB and reduces LPIPS by 0.0821 and 0.0420 at NFE9 and NFE13, respectively. The corresponding mean latency differences are 1.80 s at NFE9 and 0.75 s at NFE13. Thus, neither tested alternative schedule closes the fidelity gap to VELOFORESIGHT on this subset, supporting the shared-anchor comparison at these operating points.

I ADDITIONAL QUALITATIVE COMPARISONS

Figure 3 supplements the video-generation results in Table 2 with two paired cases from FULL946. Using a standardized PSNR/SSIM/LPIPS score, we select a Wan NFE13 occurrence closest to the median outcome and the worst-ranked Hunyuan NFE9 occurrence. Both cases are selected before frame inspection. For each case, we show the midpoint frame from the paired Full50, Spectrum, and VELOFORESIGHT videos.



Figure 3: Outcome-declared paired cases from FULL946. The first row is a median Wan NFE13 outcome. The second is the worst Hunyuan NFE9 outcome under a deterministic standardized three-metric rule. Selection precedes frame inspection, and no frame is synthesized, enhanced, or retouched.