TRAJECTORY-AWARE TIMESTEP SCHEDULING FOR VIDEO DIFFUSION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Video diffusion models require expensive model evaluations, making timestep allocation critical under limited inference budgets. Existing schedules allocate steps in fixed proportions and apply the same grid to all samples. We observe that trajectory acceleration magnitudes are initially high and decay rapidly, with substantial variation across samples concentrated in early denoising. Motivated by these findings, we propose a scheduling framework combining Double-Shift mapping with acceleration-driven online scheduling. Double-Shift reserves an absolute early-stage step budget and applies a separate nonlinear mapping. The online scheduler adapts early-stage timesteps to each sample using acceleration estimated from cached velocity predictions and a step-size rule derived from Euler local truncation error analysis. Experiments across models, benchmarks, and sampling budgets demonstrate improved generation quality, robustness to scheduling parameters, and negligible scheduling overhead. Both components are effective independently and yield further gains when combined, while remaining compatible with model variants and inference-time enhancement techniques.

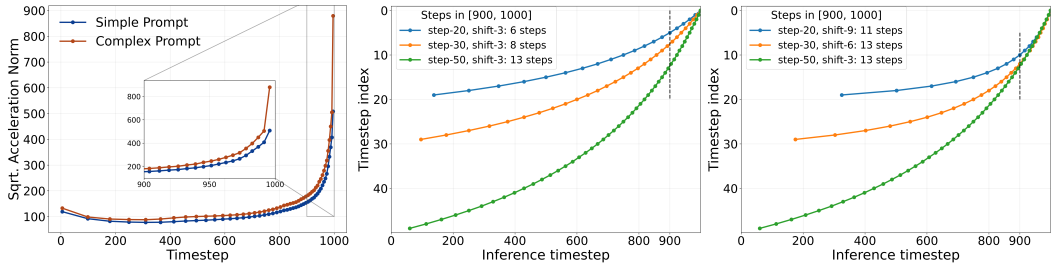
1 INTRODUCTION

Video Diffusion Models (VDMs) (Ho et al., 2022; Blattmann et al., 2023) have achieved progress in video generation. Their inference process can be viewed as evolving from a noise distribution toward the data distribution along a continuous-time denoising trajectory (Ho et al., 2020; Song et al., 2021b; Lipman et al., 2023), which is discretized through a finite number of model evaluations and numerical integration steps (Song et al., 2021a; Karras et al., 2022; Lu et al., 2022). How to allocate a limited number of solver steps is critical to both generation quality and sampling efficiency. Shift scheduling (Esser et al., 2024) is a common strategy in VDMs, using a fixed shift parameter to nonlinearly remap timesteps. Like Shift scheduling, other existing methods in VDMs (Yang et al., 2025; Polyak et al., 2024; Lee et al., 2025) allocate steps across the entire denoising interval according to a fixed global proportion and apply the same schedule to all samples. However, these methods do not adapt to the geometric characteristics of VDM denoising trajectories.

We conduct a geometric analysis of the generative trajectories of VDMs and obtain two key observations. (1) The acceleration magnitude of the denoising trajectory drops sharply at the initial stage and subsequently remains at a low level, as shown in Fig. 1a. This indicates changes in the magnitude or direction of the trajectory's velocity during the early stage, calling for more solver steps to control discretization error. However, existing schedulers allocate steps according to a fixed global proportion. As the sampling budget decreases, fewer steps are assigned to the early stage, making the high-acceleration region increasingly difficult to capture accurately, as illustrated in Fig. 1b. (2) The acceleration magnitude evolves differently across samples. The more complex the prompt, the higher the sample's initial acceleration magnitude tends to be and the more sharply it decreases. Once the trajectory enters the stable phase, acceleration magnitudes across samples converge to similarly low levels. Thus, scheduling needs vary primarily across samples during early denoising, whereas existing methods apply a uniform schedule to all samples.

Motivated by the first observation, we propose a Double-Shift mapping strategy. Unlike later stages, early-stage errors can propagate and accumulate over subsequent steps (Ning et al., 2023; Li et al., 2024). Moreover, a VDM's early stage largely determines the video's global structure and motion patterns (Pardo et al., 2025), so errors in this stage can substantially affect generation quality and

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.



(a) Trajectory acceleration. (b) Fixed hyperparameter. (c) Tuned hyperparameter.

Figure 1: (a) Denoising trajectory acceleration magnitude of Wan2.2-5B under simple (VBench Object Class) and complex (VideoPhy2.0) prompts, with 79 prompts per group. (b) Shift timestep distributions under different sampling budgets with shift=3.0; fewer steps reduce early-stage allocation. (c) Optimal tuned shifts and their timestep distributions, which consistently allocate about 13 steps to the early stage.

physical plausibility. We therefore decouple the early stage, where acceleration magnitude is high and decays rapidly, assign it an independent absolute step budget, and apply a separate nonlinear mapping within this interval. We find that optimal Shift schedules obtained through costly tuning for different step counts implicitly allocate sufficient steps to this stage, as shown in Fig. 1c, supporting our design. Increasing the early-stage step budget consistently improves generation quality, while a separate nonlinear mapping yields additional gains, as shown in Fig. 2a.

Motivated by the second observation, we propose an acceleration-driven online scheduler. From the local truncation error of numerical integration, we derive a relationship between the solver step size and the trajectory acceleration norm: for a fixed error tolerance, larger acceleration calls for a smaller step size. Since the acceleration bound is unavailable at inference time, we estimate acceleration using finite differences of cached velocity predictions. As shown in Fig. 2c, across samples, the Root-Mean-Square (RMS) of estimated early-stage acceleration is significantly positively correlated with the RMS of local error, providing evidence for using finite-difference acceleration as a proxy for local error. Using the acceleration-guided step-size rule, we adaptively schedule timesteps online. We extend the method to higher-order solvers, demonstrating its generality.

The two methods work in concert to enable stage-wise step allocation and sample-wise adaptation. Guided by the geometry of VDM trajectories, Double-Shift provides the online scheduler with the early-stage boundary, step budget, and warm-up points. The scheduler then adapts early-stage timestep locations to each sample's dynamics: under a fixed budget, it redistributes Double-Shift's reserved steps; with a variable budget, it can insert additional steps according to sample complexity.

We evaluate our approach across multiple benchmarks and foundation models of diverse architectures and parameter scales, demonstrating its effectiveness and generality. Comparisons with widely used video diffusion timestep schedules show superior generation performance. To assess robustness to the sampling budget, we keep all scheduling hyperparameters fixed across different inference step counts. We further apply our scheduler to multiple fine-tuned models and combine it with various inference-time enhancement techniques, demonstrating broad compatibility and practical utility. Ablations reveal the complementary effects of the two components, sensitivity analyses show robustness to hyperparameter choices, and runtime evaluations confirm negligible scheduling overhead.

- We reveal rapid early-stage acceleration decay and substantial variation across samples in video denoising trajectories, motivating stage-wise budget allocation and sample-adaptive timestep scheduling.
- We propose a scheduling framework combining Double-Shift mapping with acceleration-driven online scheduling to reserve an absolute early-stage step budget and adapt timestep locations to individual samples.
- Extensive experiments demonstrate improvements over widely used schedules across models, benchmarks, and sampling budgets, with negligible scheduling overhead, robustness to scheduling parameters, and compatibility with inference-time enhancement techniques.

2 Related Work

VDMs (Ho et al., 2022; Blattmann et al., 2023) extend diffusion models (Ho et al., 2020; Lipman et al., 2023) to spatiotemporal video latents with an additional temporal dimension, jointly modeling spatial structure and temporal dynamics. Representative models include Sora (Brooks et al., 2024), Stable Video Diffusion (Blattmann et al., 2023), VideoCrafter (Chen et al., 2023), CogVideo (Hong et al., 2023), HunyuanVideo (Kong et al., 2024), and Wan (Wan et al., 2025). Similar to image diffusion models, VDMs typically adopt predefined timestep scheduling strategies, such as Uniform (Yang et al., 2025), EDM (Karras et al., 2022), Shift (Esser et al., 2024), Beta (Lee et al., 2025), and Linear-Quadratic (Polyak et al., 2024) schedules, among which Shift is one of the most widely used strategies, such as HunyuanVideo, Wan, and Skyreels-v2 (Chen et al., 2025). Specifically, this strategy first constructs a uniform timestep grid in the normalized time domain and then redistributes the timesteps using a global Shift mapping. Existing timestep schedules for VDMs have two main limitations. First, fixed global mappings allocate fewer early-stage NFEs as the sampling budget decreases, making low-step sampling sensitive to the Shift parameter. Second, they apply the same schedule to all samples, failing to adapt to the distinct geometric curvature characteristics exhibited by different samples in the early stage.

Additionally, studies on image diffusion models have explored non-uniform and adaptive timestep scheduling, including AYS (Sabour et al., 2024), OYS Zhang et al. (2026a), ATS Chen et al. (2024), and TPDM (Ye et al., 2025). These methods typically rely on computationally expensive offline optimization or additional training, and have not yet been adopted by VDMs. For example, TPDM trains a time prediction module via Reinforcement Learning (RL) to dynamically adjust the timestep schedule. However, RL-based methods are computationally expensive, while reinforcement learning and reward modeling for VDMs remain underdeveloped, making direct transfer to VDMs challenging. In contrast, our method leverages the geometric information of video denoising trajectories to achieve stage-level and sample-level adaptive timestep allocation without training an additional scheduler or relying on a video reward model.

3 Preliminaries

3.1 Video Diffusion Models

VDMs typically adopt a Diffusion Transformer (DiT) (Peebles & Xie, 2023) backbone and generate conditional videos from random noise in a spatiotemporal latent space. Their training paradigms mainly include conventional diffusion modeling(Ho et al., 2020) and Flow Matching (Lipman et al., 2023). Although these paradigms differ in training objectives, their inference processes can both be formulated as the numerical integration of a continuous-time dynamical system.

Let an RGB video be denoted by $\boldsymbol{x} \in \mathbb{R}^{F\times H\times W\times 3}$. A video VAE (Kingma & Welling, 2013) encodes it into a spatiotemporal latent representation: $\boldsymbol{z} \in \mathbb{R}^{f\times h\times w\times c}$, where f, h, and w are the temporally and spatially compressed dimensions, and c is the latent channel dimension. The generation process can be described by the ordinary differential equation (ODE):

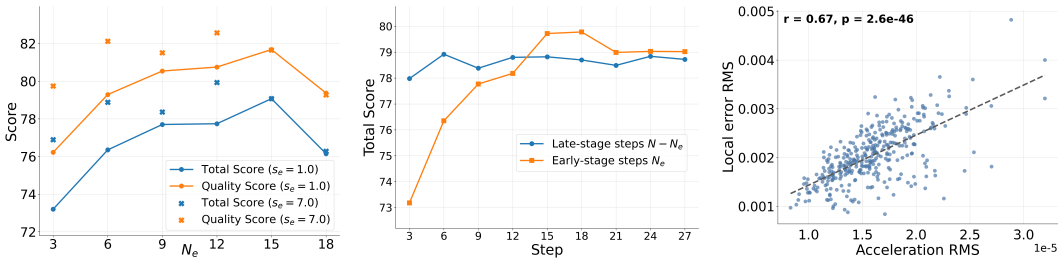
$$\frac{\mathrm{d}\boldsymbol{z}_t}{\mathrm{d}t} = \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t, \boldsymbol{c}), \tag{1}$$

where $\boldsymbol{z}_t$ denotes the latent state at time t, $\boldsymbol{c}$ denotes the conditioning information, and $\boldsymbol{v}_{\boldsymbol{\theta}}$ is the learned vector field. In conventional diffusion models, the vector field is derived from the predicted noise or score, whereas Flow Matching learns the vector field directly. When $\boldsymbol{c} = \varnothing$, the model reduces to an unconditional generative model.

Given discrete timesteps $\{t_i\}_{i=0}^{N}$ ordered along the sampling trajectory, the first-order Euler solver Song et al. (2021a) discretizes the ODE in Eq. (1) as

$$\boldsymbol{z}_{t_{i+1}} = \boldsymbol{z}_{t_i} + (t_{i+1} - t_i)\boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{z}_{t_i}, t_i, \boldsymbol{c}). \tag{2}$$

Here, N denotes the number of sampling steps, and each vector-field evaluation corresponds to one network function evaluation (NFE).



(a) Early-step scaling. (b) Double-Step scaling. (c) Acceleration-error correlation.

Figure 2: (a) With the total number of sampling steps fixed at $N = 20$, increasing the number of early steps substantially improves the generation quality. Applying a nonlinear mapping within the early truncation interval further improves performance. (b) When the number of early steps is fixed at $N_e = 14$, increasing the number of later steps yields only limited performance gains. In contrast, when the number of later steps is fixed at $N - N_e = 14$, increasing the number of early steps leads to substantially stronger scaling in terms of the Total Score. (c) Correlation between the estimated acceleration and local trajectory error. The estimated acceleration RMS exhibits a strong and statistically significant positive correlation with the local trajectory error RMS.

4 Method

We develop a timestep scheduling framework that combines stage-level geometric priors with sample-dependent trajectory dynamics. We first characterize the geometry of video denoising trajectories, then introduce a two-stage time mapping to allocate the sampling budget, followed by an online controller that adapts early-stage timesteps to individual samples.

4.1 Geometric Observations of Video Denoising

To understand how to allocate a limited sampling budget, we analyze the acceleration of video diffusion denoising trajectories. Let $\boldsymbol{v}_i = \boldsymbol{f}_\theta(\boldsymbol{x}_i, t_i; c)$ denote the model-predicted velocity at timestep t_i, where $1 \geq t_i > t_{i+1} \geq 0$. After flattening each predicted velocity into a vector, we estimate trajectory acceleration from consecutive velocity predictions and compute its magnitude as

$$\widehat{\boldsymbol{a}}_i = \frac{\boldsymbol{v}_i - \boldsymbol{v}_{i-1}}{t_i - t_{i-1}}, \qquad A_i = \|\widehat{\boldsymbol{a}}_i\|_2, \qquad i \geq 1. \tag{3}$$

Acceleration reflects changes in the magnitude or direction of the denoising velocity. We compute acceleration magnitudes independently for each sample before aggregating the resulting curves within each prompt group.

We analyze denoising trajectories generated by Wan2.2-5B (Wan et al., 2025) using 79 prompts from the Object Class subset of VBench (Huang et al., 2024) and 79 prompts from VideoPhy2.0. Object Class prompts mainly describe individual object categories and provide relatively simple semantic conditions. In contrast, VideoPhy2.0 prompts emphasize actions, transitions, and adherence to physical laws, providing more complex generation conditions. We compare the two groups under identical sampling configurations and obtain two key findings.

Key Finding 1. Acceleration magnitudes drop sharply during early denoising and remain substantially lower afterward, as shown in Fig. 1a. Large acceleration magnitudes motivate finer timestep discretization through the Euler error analysis below. Early errors can accumulate, while early denoising determines global structure and motion, affecting visual quality and physical plausibility. However, existing schedulers allocate steps in fixed proportions, reducing the early-stage budget as the total budget decreases (Fig. 1b) and potentially increasing discretization errors.

Key Finding 2. Early-stage acceleration patterns depend on the conditioning prompts. The more complex VideoPhy2.0 group exhibits larger initial acceleration magnitudes and more pronounced early decreases than Object Class (Fig. 1a), with smaller differences later. This suggests that sample-dependent scheduling needs are concentrated in early denoising, whereas existing schedulers apply the same timestep rule to all samples.

Appendix A supports both findings across Wan2.1-1.3B, CogVideoX-2B, and HunyuanVideo1.5-8B; VBench Object Class, VBench Overall Consistency, and VideoPhy2.0 prompts; and shift values of 3.0, 5.0, and 9.0. This consistency strengthens the empirical basis for stage-wise budget allocation and sample-adaptive timestep scheduling.

4.2 DOUBLE-SHIFT

Key Finding 1 reveals clear stage-wise geometric characteristics in the denoising trajectories of VDMs, suggesting that a single global scheduling curve cannot fully match the varying solver demands across different stages. Furthermore, large-scale searches over scheduling parameters at different sampling budgets show that the optimal configurations tend to implicitly allocate sufficient computation steps to the early stage, as shown in Fig. 1c. This result is consistent with our geometric analysis and further indicates that the early stage requires a larger computational budget.

We therefore decouple the rapidly decaying early high-curvature region, assign it an independent absolute step budget, and apply a separate nonlinear mapping. We adopt the widely used Flow-Shift mapping, whose general form over an arbitrary interval is defined as

$$G_s(u; a, b) = a + (b - a)g_s(u), \qquad g_s(u) = \frac{su}{1 + (s-1)u}, \qquad u \in [0, 1]. \tag{4}$$

When $[a, b] = [0, 1]$, Eq. (4) reduces to the standard global Flow-Shift (Esser et al., 2024) mapping widely adopted by VDM such as Wan and HunyuanVideo.

For a given total number of sampling steps N and a boundary timestep t_e, instead of applying a single mapping over the entire interval, we divide $[0, 1]$ into an early stage $[t_e, 1]$ and a later stage $[0, t_e]$. We assign the early stage an independent absolute step budget N_e and a dedicated shift parameter $s_e \geq 0$. The remaining $N - N_e$ steps are allocated to the later stage, using the baseline shift parameter $s > 0$. In other words, the two intervals are mapped independently and then concatenated to form the final timestep sequence:

$$(t_0, \ldots, t_N) = \text{concat}\left(\left[G_{s_e}\left(1 - \frac{i}{N_\text{e}}; t_e, 1\right) \right]_{i=0}^{N_\text{e}}, \left[G_s\left(1 - \frac{j}{N - N_\text{e}}; 0, t_e\right) \right]_{j=1}^{N-N_\text{e}} \right). \tag{5}$$

Eq. (5) controls step allocation between stages and nonlinear mapping within each stage. Under a fixed budget, increasing N_e improves generation quality, with further gains from a separate nonlinear mapping that accommodates rapid early acceleration decay (Fig. 2a). Without a fixed total budget, adding early steps yields substantially greater gains than adding late steps (Fig. 2b). Experimental details are provided in Appendix B.

Additionally, we fix $t_e = 0.9$ as a coarse boundary for the early high-curvature regime. Rather than precisely locating the transition point, our goal is to isolate the rapidly varying early dynamics. The nonlinear mapping parameter s_e further captures the decay pattern within this interval, reducing sensitivity to the exact choice of t_e.

4.3 ACCELERATION-DRIVEN ONLINE SCHEDULING

Double-Shift allocates steps by stage but uses a shared early-stage grid. Thus, we use cached acceleration estimates to adapt step sizes across samples: Euler error analysis motivates the rule, correlation with local error supports the proxy, and Double-Shift provides the budget and boundary constraints.

Euler-based step-size relation. Let the denoising trajectory satisfy $\frac{\mathrm{d}\boldsymbol{x}(t)}{\mathrm{d}t} = \boldsymbol{v}(t)$, and define $\boldsymbol{a}(t) = \frac{\mathrm{d}\boldsymbol{v}(t)}{\mathrm{d}t}$. A Taylor expansion along the decreasing-time direction gives

$$\boldsymbol{x}(t_i - h_i) = \boldsymbol{x}(t_i) - h_i \boldsymbol{v}(t_i) + \frac{h_i^2}{2}\boldsymbol{a}(t_i) + O(h_i^3). \tag{6}$$

Since the Euler solver retains only the first two terms, its local truncation error satisfies

$$\|e_i\|_2 \leq \frac{h_i^2}{2} M_i, \qquad M_i = \sup_{\tau \in [t_i - h_i, t_i]} \|\boldsymbol{a}(\tau)\|_2. \tag{7}$$

Thus, for a prescribed local error tolerance $\eta > 0$, a sufficient condition for $\|e_i\|_2 \leq \eta$ is

$$h_i \leq \sqrt{\frac{2\eta}{M_i}}, \tag{8}$$

where $\eta > 0$ is a local error tolerance. For fixed η, Eq. (8) theoretically motivates smaller steps at larger accelerations, and .

Acceleration estimation and online update. The sufficient bound in Eq. (7) depends on the maximum acceleration over the current interval, M_i, which is unavailable at inference time. Its leading-order error term is governed by the instantaneous acceleration $\|\boldsymbol{a}(t_i)\|_2$. Since this quantity is also unavailable, we estimate it from the two most recent velocity predictions; their finite difference, computed using Eq. 3 with index $i-1$, approximates the average acceleration over the preceding interval $[t_{i-2}, t_{i-1}]$ for $i \geq 2$. We denote this cached acceleration estimate by $\boldsymbol{a}_{i-1}$.

Because this estimate describes the preceding interval, it may lag behind the current trajectory dynamics. We account for the observed early-stage acceleration decay with a factor $\rho \in (0, 1]$:

$$\widehat{A}_i = \rho \|\boldsymbol{a}_{i-1}\|_2, \tag{9}$$

Substituting this estimate for the acceleration bound in Eq. (8) gives the candidate step size

$$\widetilde{h}_i = \sqrt{\frac{2\eta}{\max(\widehat{A}_i, \epsilon)}}. \tag{10}$$

Here, $\epsilon > 0$ is a small constant to prevent division by zero. η controls the candidate step size: larger η permits larger steps. This estimator reuses cached predictions without additional model prediction.

To assess whether estimated acceleration reflects sample-wise early-stage integration difficulty, we compute the acceleration RMS and local-error RMS over $t \in [0.9, 1]$ for each sample. Fig. 2c shows that their strong, statistically significant positive correlation supports acceleration as a sample-wise proxy for early-stage integration difficulty, with more details in Appendix C. The controller uses stepwise acceleration estimates to assign smaller steps where acceleration is larger via Eq. (10).

Synergy with Double-Shift. We clip each early-stage step to prevent crossing t_e. At t_e, sampling resumes on the preset late-stage grid, ensuring a smooth transition:

$$h_i = \min\{\widetilde{h}_i, t_i - t_e\}, \qquad t_{i+1} = t_i - h_i, \qquad t_i > t_e. \tag{11}$$

Notably, the first two evaluations follow the preset schedule to initialize the velocity history. Their timestep locations affect the initial acceleration estimate, so Double-Shift's nonlinear early-stage mapping also provides the controller's warm-up points.

Under a fixed NFE budget, the early stage uses exactly N_e steps, and the controller only redistributes their locations. To prevent large candidate steps from reaching t_e prematurely, we impose

$$h_i \leq \frac{t_i - t_e}{N_e - N_i}, \tag{12}$$

where N_i is the number of early-stage steps taken, including the warm-up steps. Conversely, if the steps are too small to reach t_e within N_e steps, we set the N_e-th step to end exactly at t_e. Since the trajectory typically becomes smoother near t_e, this budget-closing step tends to incur lower local error. Double-Shift defines the early-stage boundary and budget, provides the warm-up points, and preserves the late-stage grid, while the controller redistributes early-stage steps according to sample-specific dynamics. Appendix D details the algorithm and its workflow.

For higher-order solvers, we retain each solver's native state update and use acceleration as an empirical error proxy. At matched step lengths, solvers show a strong positive correlation between acceleration magnitude and local truncation error. Supporting evidence is provided in Appendix E.

In summary, error analysis motivates acceleration-based scheduling, correlation analysis supports the proxy, and Double-Shift provides the structure for online control. The ablation study in Tab. 5 verifies the practical gains from combining the two components.



Figure 3: Qualitative comparisons between the baseline and our method.

5 Experiments

Base Models. We evaluate our method on Wan2.1-1.3B, AnyFlow-FAR-Wan2.1-1.3B (Gu et al., 2026), CogVideoX-2B, CogVideoX-5B, and AnyFlow-Wan2.1-14B under varied step budgets to assess its effectiveness and generalizability across model families and scales. We test the method with fixed parameters across multiple timestep settings to evaluate its robustness.

Baselines. We compare our method with mainstream scheduling strategies for current VDM, including uniform (Hong et al., 2023), Shift (Esser et al., 2024), Beta (Lee et al., 2025), and linear-quadratic (Linear-Quad.) (Polyak et al., 2024), to evaluate its effectiveness relative to existing approaches. To the best of our knowledge, timestep scheduling for VDMs remains largely unexplored, with even fewer studies on dynamic scheduling. Existing adaptive or learnable schedulers for image diffusion typically rely on offline optimization (Chen et al., 2024; Sabour et al., 2024) or additional training (Ye et al., 2025), and have not been adopted by current mainstream VDMs. In contrast, our method introduces negligible computational overhead. We therefore mainly compare against the scheduling strategies actually used in mainstream VDMs.

Benchmarks. We use VBench (Huang et al., 2024) to evaluate overall video quality, VideoPhy2.0 (Bansal et al., 2026) for physical plausibility, and PAI-Bench (Zhou et al., 2026) for physical realism in robot simulation videos. For VBench, we report Quality, Semantic, and Total scores. For VideoPhy2.0, we report semantic adherence (SA) and the joint score (Joint), where Joint denotes the score for jointly satisfying semantic and physical consistency. For PAI-Bench, following WanPP (Jang et al., 2026), we report Grasp, VQA, and Quality scores. We sample 198 prompts from each of VBench and VideoPhy2.0 and generate two videos per prompt using fixed random seeds, with 155 prompt–image pairs selected for PAI-Bench.

Compatibility. We integrate our methods into CFG-Zero (Fan et al., 2025), AnyFlow, VideoREPA (Zhang et al., 2026b), and Cosmos3 (Agarwal et al., 2026), covering CFG enhancement, physical alignment, on-policy distillation (OPD), and world models. The results highlight the importance of timestep scheduling and demonstrate the generalizability and practicality of our method.

5.1 Qualitative Results

Our method substantially improves the visual quality and physical plausibility of generated videos, as shown in Fig. 3. For the beach-walking and dog-running prompts, it produces more visually appealing scenes. For the bird prompt, it generates a clearly recognizable bird that follows the prompt. For the bouncing-ball prompt, it preserves the ball's appearance across frames while depicting the bounce, whereas the baseline changes the ball's appearance.

Table 1: Results on VBench and VideoPhy2.0.

Method	VBench						VideoPhy2.0			
	CogVideoX-1.5-2B			Wan2.1-1.3B			CogVideoX-1.5-2B		Wan2.1-1.3B	
	Total	Quality	Semantic	Total	Quality	Semantic	SA	Joint	SA	Joint
Uniform	71.76	74.68	60.08	77.57	79.96	68.02	10.10	9.09	15.66	11.62
Shift	77.40	79.30	**69.79**	77.57	79.96	68.02	27.78	21.21	**29.29**	22.22
Beta	77.20	79.76	66.95	78.41	81.48	66.13	28.79	**25.25**	27.27	19.19
Line-Quad	77.26	79.21	69.48	79.23	81.67	**69.50**	26.77	20.71	26.77	21.72
Ours	**77.53**	**79.78**	68.55	**80.05**	**82.69**	69.45	**29.29**	23.74	**29.29**	**24.24**

Table 2: Results on VBench with different step budgets.

Method	Type	20 Steps		30 Steps		50 Steps	
		Total	Semantic	Total	Semantic	Total	Semantic
Shift	Fixed	76.65	65.98	77.66	65.23	79.30	67.75
Shift*	Tuned	79.06	67.20	79.13	67.47	79.30	67.75
Ours	Fixed	**79.53**	**66.75**	**79.65**	**67.30**	**79.61**	**67.01**

Table 3: Results on 5B and 14B models.

Method	Step	SA	Joint
CogVi-5B	20	8.08	7.07
+Ours	20	**26.77**	**21.21**
Anyflow-14B	4	26.26	20.71
+Ours	4	**27.02**	**22.47**

5.2 Main Results

To evaluate the effectiveness, generality, and superiority of our method, we conduct experiments on different VDMs and benchmarks, and compare it with existing scheduling strategies. Specifically, effectiveness is reflected in the performance improvements across different models, generality is assessed by the consistent performance across different benchmarks, and superiority is validated through comparisons with existing methods, as shown in Tab. 1. We emphasize that VBench Total and VideoPhy2.0 Joint are the most representative overall metrics for the two benchmarks, respectively. Our method achieves the highest VBench Total score on both CogVideoX-1.5-2B and Wan2.1-1.3B, and obtains the highest VideoPhy2.0 Joint score on Wan2.1-1.3B, demonstrating good model adaptability and overall performance.

5.3 Different Steps

To evaluate robustness across sampling budgets, we compare fixed and per-budget tuned settings. Shift-Fixed uses $s = 3.0$, the officially recommended shift value for Wan2.1-1.3B, for all budgets, whereas Shift-Tuned searches for the best shift at each budget.Our parameters are fixed at $N_c = 10$, $s_e = 3.0$, and $\eta = 1.6$, with the same configuration used across 20, 30, and 50 sampling steps. It achieves Total scores of 79.53, 79.65, and 79.61 at 20, 30, and 50 steps, respectively, exceeding both Shift settings at every budget, as shown in Tab. 2. With only 0.12 points of variation in Total score, these results demonstrate robustness to changes in the sampling budget.

5.4 Larger, Enhanced, and Fine-tuned Models

We evaluate generalizability and practicality on larger models, enhanced or fine-tuned models, and a world model. AnyFlow uses flow-map-based OPD, VideoREPA incorporates physical knowledge into VDMs through relational distillation, and Cosmos3 is a multimodal world model. We test AnyFlow-Wan2.1-14B and CogVideoX-5B for scalability, and use AnyFlow-FAR-Wan2.1-1.3B, CogVideoX-2B, and Cosmos3-Edge-4B for the AnyFlow, VideoREPA, and Cosmos3 experiments, respectively. Grasp and VQA are evaluated using Qwen2.5-VL-72B-Instruct (Bai et al., 2025).

As shown in Tabs. 3 and 4, our scheduler improves performance across larger models and specialized model variants. On VideoPhy2.0, Uniform scheduling performs poorly for CogVideoX-based models at 20 steps; our scheduler raises Joint scores from 7.07 to 21.21 for CogVideoX-5B and from 7.58 to 19.19 for VideoREPA, with substantial SA gains. It also improves SA and Joint on AnyFlow-14B at just four steps and all reported VBench metrics for AnyFlow. For Cosmos3 on PAI-Bench, Grasp

Table 4: Results across physics-aware, and world models.

Benchmark	VBench			VideoPhy2.0		PAI-Bench		
Model	AnyFlow			VideoREPA		Cosmos3		
Method	Total	Quality	Seman	SA	Joint	Grasp	VQA	Quality
Base	82.27	85.43	69.66	8.08	7.58	74.51	78.31	**75.61**
Ours	**83.50**	**86.02**	**73.41**	**20.20**	**19.19**	**81.61**	**79.77**	75.51

Table 5: Ablation study.

Method	Total	Quality	Seman
Wan2.1	77.57	79.96	68.02
+DS	78.82	81.47	68.22
+AD	79.18	82.38	66.41
+Ours	**80.05**	**82.69**	**69.45**

and VQA increase from 74.51 and 78.31 to 81.61 and 79.77, while Quality slightly decreases from 75.61 to 75.51. These results demonstrate practical benefits across model scales and specialized generation settings, highlighting compatibility with diverse model architectures and enhancements.

5.5 Ablation Study

Synergistic Effect. We compare Double-Shift (DS) and acceleration-driven online scheduling (AD), individually and in combination, against the Wan2.1 baseline on VBench (Tab 5). DS improves the Total, Quality, and Semantic scores from 77.57, 79.96, and 68.02 to 78.82, 81.47, and 68.22, respectively. AD raises the Total and Quality scores to 79.18 and 82.38, while its Semantic score decreases to 66.41. These two components address complementary aspects of timestep allocation: DS allocates a fixed number of NFEs to the early high-curvature stage, whereas AD adaptively refines timesteps according to sample-specific dynamics. Combining them yields the highest scores across all three metrics—80.05 Total, 82.69 Quality, and 69.45 Semantic—demonstrating their synergistic effect.

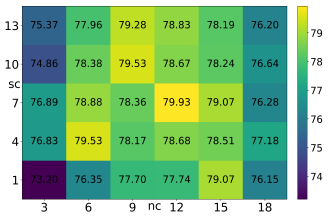


Figure 4: Robustness.

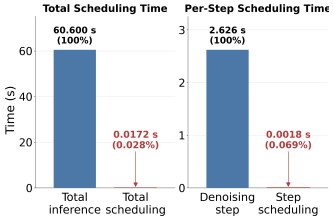


Figure 5: Time ablation.

Parameter Robustness. We further analyze the robustness of our method to key parameter choices. For Double-Shift, we perform a Cartesian-product sweep over the early-step parameter n_c and the mapping parameter s_c on VBench. As shown in Fig. 4, Total scores remain between 77.96 and 79.93 for $n_c \in \{6, 9, 12, 15\}$ and $s_c \in \{4, 7, 10, 13\}$, indicating a broad effective parameter region. See Appendix F for the sensitivity analysis of η in Eq. (10).

Computational Overhead. Our method introduces negligible computational overhead, as shown in Fig. 5. On 100 VBench prompts, 20-step CogVideoX-2B inference on a single NVIDIA H20 GPU takes $60.600\,\mathrm{s}$, while scheduling requires only $0.0172\,\mathrm{s}$ (0.028% overhead). At each step, dynamic scheduling takes $0.0018\,\mathrm{s}$ compared with $2.626\,\mathrm{s}$ for denoising, accounting for only 0.069% of the step time. Thus, our scheduler adds virtually no practical cost.

6 Conclusion

In this work, we study timestep allocation for video diffusion models under limited inference budgets. We identify rapid early-stage acceleration decay and substantial variation across samples, motivating a framework that combines Double-Shift mapping with acceleration-driven online scheduling. Double-Shift reserves an absolute early-stage step budget, while the online scheduler adapts timestep locations using acceleration estimated from cached velocity predictions. The step-size rule follows Euler error analysis, with acceleration serving as an empirical error proxy for higher-order solvers. Our framework requires no additional scheduler training. Extensive experiments demonstrate complementary gains from both components, improved generation quality across models and sampling budgets, robustness to scheduling parameters, and negligible scheduling overhead.

AI USE STATEMENT

We used generative AI tools for translation, manuscript drafting and editing, LaTeX formatting, and assistance with interpreting and presenting experimental results. We also used Qwen2.5-VL-72B-Instruct to evaluate Grasp and VQA, as described in the experiments. The authors reviewed all AI-assisted content and take full responsibility for the final manuscript and its findings.

REPRODUCIBILITY STATEMENT

We provide explicit scheduling equations and detailed pseudocode, with the sampling workflow described in Appendix D. We clearly explain the roles and settings of the scheduling parameters, including fixed constants and tunable hyperparameters. Together with the experimental configurations and evaluation protocols described in the paper, these details provide a clear basis for reimplementing our method and reproducing the reported results.

REFERENCES

Niket Agarwal, Arslan Ali, Jon Allen, Martin Antolini, Adeline Aubame, Alisson Azzolini, Junjie Bai, Maciej Bala, Yogesh Balaji, Josh Bapst, et al. Cosmos 3: Omnimodal world models for physical ai. *arXiv preprint arXiv:2606.02800*, 2026.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Hritik Bansal, Clark Peng, Yonatan Bitton, Roman Goldenberg, Aditya Grover, and Kai-Wei Chang. Videophy-2: A challenging action-centric physical commonsense evaluation in video generation. In *The Fourteenth International Conference on Learning Representations*, 2026.

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. Video generation models as world simulators. 2024. *URL https://openai. com/research/video-generation-models-as-world-simulators*, 3(1):3, 2024.

Guibin Chen, Dixuan Lin, Jiangping Yang, Chunze Lin, Junchen Zhu, Mingyuan Fan, Hao Zhang, Sheng Chen, Zheng Chen, Chengcheng Ma, et al. Skyreels-v2: Infinite-length film generative model. *arXiv preprint arXiv:2504.13074*, 2025.

Haoxin Chen, Menghan Xia, Yingqing He, Yong Zhang, Xiaodong Cun, Shaoshu Yang, Jinbo Xing, Yaofang Liu, Qifeng Chen, Xintao Wang, et al. Videocrafter1: Open diffusion models for high-quality video generation. *arXiv preprint arXiv:2310.19512*, 2023.

Yuzhu Chen, Fengxiang He, Shi Fu, Xinmei Tian, and Dacheng Tao. Adaptive time-stepping schedules for diffusion models. In *UAI*, pp. 685–697, 2024.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

Weichen Fan, Amber Yijia Zheng, Raymond A Yeh, and Ziwei Liu. Cfg-zero*: Improved classifier-free guidance for flow matching models. *arXiv preprint arXiv:2503.18886*, 2025.

Yuchao Gu, Guian Fang, Yuxin Jiang, Weijia Mao, Song Han, Han Cai, and Mike Zheng Shou. Anyflow: Any-step video diffusion model with on-policy flow map distillation. In *European Conference on Computer Vision*, pp. 163–181. Springer, 2026.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *the Advances in Neural Information Processing Systems*, 2020.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in neural information processing systems*, 35:8633–8646, 2022.

Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=rB6TpjAuSRy.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Sangwon Jang, Taekyung Ki, Jaehyeong Jo, Saining Xie, Jaehong Yoon, and Sung Ju Hwang. Self-refining video sampling. In *Forty-third International Conference on Machine Learning*, 2026.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

Haeil Lee, Hansang Lee, Seoyeon Gye, and Junmo Kim. Beta sampling is all you need: Efficient image generation strategy for diffusion models using stepwise spectral analysis. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2025.

Mingxiao Li, Tingyu Qu, Ruicong Yao, Wei Sun, and Marie-Francine Moens. Alleviating exposure bias in diffusion models through sampling with shifted time steps. In *ICLR*, 2024.

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

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. *Advances in neural information processing systems*, 35:5775–5787, 2022.

Mang Ning, Enver Sangineto, Angelo Porrello, Simone Calderara, and Rita Cucchiara. Input perturbation reduces exposure bias in diffusion models. In *ICML*, 2023.

Alejandro Pardo, Fabio Pizzati, Tong Zhang, Alexander Pondaven, Philip Torr, Juan Camilo Perez, and Bernard Ghanem. Matchdiffusion: Training-free generation of match-cuts. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025.

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

Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, et al. Movie gen: A cast of media foundation models. *arXiv preprint arXiv:2410.13720*, 2024.

Amirmojtaba Sabour, Sanja Fidler, and Karsten Kreis. Align your steps: Optimizing sampling schedules in diffusion models. In *Forty-first International Conference on Machine Learning*, 2024.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021b.

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.

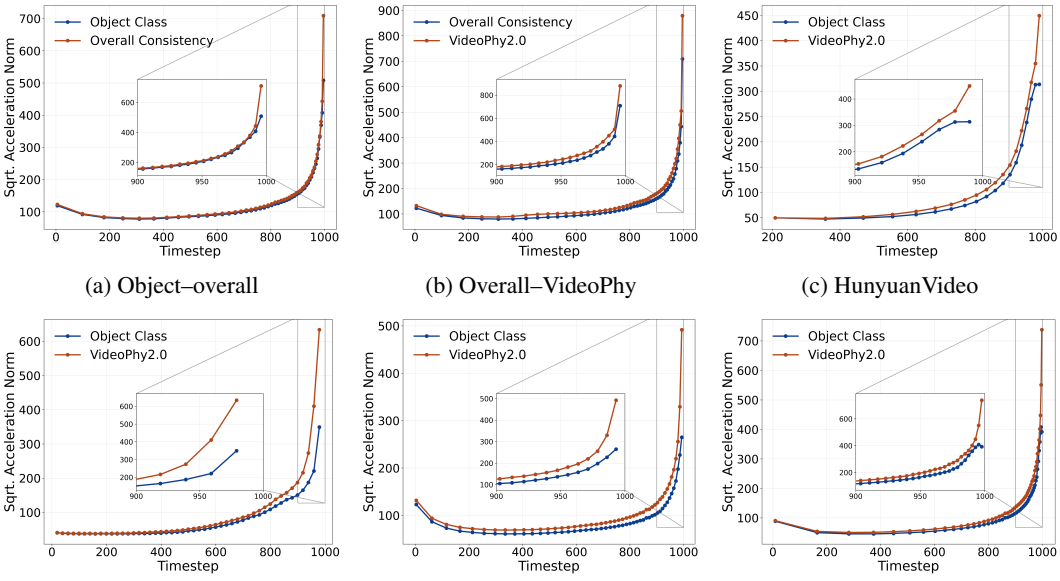
Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations*, 2025.

Zilyu Ye, Zhiyang Chen, Tiancheng Li, Zemin Huang, Weijian Luo, and Guo-Jun Qi. Schedule on the fly: Diffusion time prediction for faster and better image generation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Travis Zhang, Christian Belardi, Justin Lovelace, Jin Peng Zhou, Saebyeol Shin, Carla P Gomes, and Kilian Q Weinberger. Optimize your sampling: Tuned diffusion sampling with bayesian optimization. *arXiv preprint arXiv:2608.18040*, 2026a.

Xiangdong Zhang, Jiaqi Liao, Shaofeng Zhang, Fanqing Meng, Xiangpeng Wan, Junchi Yan, and Yu Cheng. Videorepa: Learning physics for video generation through relational alignment with foundation models. In *Advances in Neural Information Processing Systems*, 2026b.

Fengzhe Zhou, Jiannan Huang, Jialuo Li, Deva Ramanan, and Humphrey Shi. Pai-bench: A comprehensive benchmark for physical ai. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.



(a) Object–overall (b) Overall–VideoPhy (c) HunyuanVideo

(d) CogVideoX (e) Wan2.1, shift 3 (f) Wan2.1, shift 9

Figure 6: Trajectory patterns across prompt pairs, models, and shift settings. (a) Wan2.2-5B, shift=5.0, comparing VBench Object Class and Overall Consistency prompts. (b) Wan2.2-5B, shift=5.0, comparing VBench Overall Consistency and VideoPhy2.0 prompts. (c) HunyuanVideo-1.5B, shift=5.0, comparing VBench Object Class and VideoPhy2.0 prompts. (d) CogVideoX-2B, shift=1.0, comparing VBench Object Class and VideoPhy2.0 prompts. (e) Wan2.1-1.3B, shift=3.0, comparing VBench Object Class and VideoPhy2.0 prompts. (f) Wan2.1-1.3B, shift=9.0, comparing VBench Object Class and VideoPhy2.0 prompts.

A Robustness of Trajectory Patterns

We examine the robustness of the two findings in Sec. 4.1: (1) acceleration magnitudes decrease sharply during early denoising and remain substantially lower afterward; and (2) acceleration patterns differ across prompt groups, with more complex conditions generally producing larger early-stage acceleration magnitudes. The main-text comparison uses 79 VBench Object Class prompts and 79 VideoPhy2.0 prompts. Here, we extend the analysis across prompt subsets, model backbones, and shift settings, as shown in Fig. 6. Acceleration is estimated using Eq. 3; the visualizations apply a square-root transformation to the acceleration norm. Denoising proceeds from larger to smaller timesteps, so the rightmost portions of the curves correspond to the early stage.

Across prompt subsets. Using Wan2.2-5B with shift 5.0, we compare VBench Object Class with Overall Consistency in Fig. 6(a), and Overall Consistency with VideoPhy2.0 in Fig. 6(b). Object Class prompts mainly describe individual object categories, whereas Overall Consistency prompts involve combinations of objects, actions, attributes, and relationships. VideoPhy2.0 further emphasizes actions, transitions, and physical constraints. Both comparisons exhibit a pronounced acceleration decrease during early denoising. Overall Consistency produces larger initial acceleration magnitudes than Object Class, while VideoPhy2.0 produces larger initial magnitudes than Overall Consistency. The group differences diminish at later timesteps. Together with the main-text comparison, these results support both findings across the three prompt subsets.

Across model backbones. We compare Object Class and VideoPhy2.0 prompts using HunyuanVideo1.5-8B with shift 5.0 in Fig. 6(c) and CogVideoX-2B with shift 1.0 in Fig. 6(d). Both models show the same qualitative pattern as Wan2.1-1.3B: acceleration is largest during early denoising and decreases rapidly as sampling progresses. VideoPhy2.0 also exhibits larger early-stage acceleration magnitudes than Object Class, with smaller differences later in denoising. Although the absolute magnitudes and decay profiles differ between models, the two findings remain consistent across the evaluated backbones.

Across shift settings. We further evaluate Wan2.1-1.3B using the same Object Class–VideoPhy2.0 prompt pair with shift values of 3.0 and 9.0, as shown in Fig. 6(e) and Fig. 6(f), respectively. Despite the different timestep allocations, both settings preserve the sharp early acceleration decrease and the larger initial acceleration magnitudes of VideoPhy2.0. The experiments with shift 5.0 described above exhibit the same two patterns. Thus, the observations remain consistent in the evaluated configurations using shifts of 3.0, 5.0, and 9.0.

Summary. Across the evaluated prompt subsets, models, and shift settings, the curves consistently support both findings: rapid early acceleration decay and prompt-dependent differences concentrated in early denoising. Modest late-stage increases in some curves do not change the pronounced contrast between the early peak and the subsequent lower-acceleration stages. This consistency provides strong empirical evidence for the robustness of our observations and supports allocating sufficient computation to the early stage while adapting timestep locations to individual samples.

B Scaling Effects of Early- and Late-Stage Steps

Experimental setup. We conduct the scanning experiments in Fig. 2a and Fig. 2b using Wan2.1-1.3B and a 198-prompt subset of VBench, with the remaining generation settings fixed. VBench provides 16 evaluation metrics. We report the Total and Quality scores in Fig. 2a and the Total score in Fig. 2b. The experiments study the effects of the early-stage step count N_e and the early-stage shift parameter s_e. When $s_e = 1$, the early interval uses a linear mapping; when $s_e = 7$, it uses a nonlinear mapping.

Fixed-budget early-step scaling. In Fig. 2a, we fix the total number of sampling steps at $N = 20$ and vary N_e. Increasing N_e reduces the number of late-stage steps, $N - N_e$, while increasing both the absolute early-stage budget and its share of the total. The Total and Quality scores rise substantially with N_e, demonstrating the importance of allocating more steps to the early stage. The mapping parameter also matters: for a given N_e, the nonlinear mapping with $s_e = 7$ generally outperforms the linear mapping with $s_e = 1$. Thus, both the number of early-stage steps and their distribution within the early interval affect performance.

Scaling without a fixed total budget. Fig. 2b compares the returns from adding steps to the late and early stages. In the late-stage sweep, we fix $N_e = 14$ and increase $N - N_e$. In the early-stage sweep, we fix $N - N_e = 14$ and increase N_e. The total number of steps grows in both sweeps, and each pair of points at the same horizontal value has the same total budget. Increasing the number of late-stage steps brings very limited gains in Total Score. By contrast, adding early-stage steps produces clear, substantial gains over the evaluated range, showing a pronounced scaling effect.

Summary. The results show that performance depends on both the size of the early-stage budget and how its steps are distributed. More importantly, the gains from increasing the total number of steps are concentrated in the early stage: increasing N_e produces strong scaling gains, whereas adding late-stage steps brings very limited additional gains.

C Acceleration–Error Correlation Analysis

Diagnostic setup. We obtain Fig. 2c from an Euler sampling run of Wan2.1-T2V-1.3B using 20 steps, a Shift value of 3.0, and a CFG scale of 5.0. We use 832×480 resolution, 81 frames, 15 FPS, four samples per prompt, and seed 42. The diagnostic is restricted to the early denoising stage, defined by $t_i > 900$, where $t_i = 1000\sigma_i$. At each step, the CFG velocity is $\boldsymbol{v}_i = \boldsymbol{v}_i^{\text{uncond}} + w(\boldsymbol{v}_i^{\text{cond}} - \boldsymbol{v}_i^{\text{uncond}})$, and the corresponding ODE vector field is $\boldsymbol{f}_i = \boldsymbol{v}_i/1000$. We estimate acceleration from consecutive vector-field predictions as $\boldsymbol{a}_i = (\boldsymbol{f}_i - \boldsymbol{f}_{i-1})/(t_i - t_{i-1})$ and compute its RMS over the latent coordinates for each sample. The first step, for which a previous vector-field prediction is unavailable, is excluded from the correlation analysis.

Local-error estimate and observed correlation. For each valid step, we estimate local error by comparing one Euler step with a reference solution computed using two Euler half-steps. Specifically, with $h_i = t_{i+1} - t_i < 0$, the coarse solution is $\boldsymbol{x}_{i+1}^{\text{coarse}} = \boldsymbol{x}_i + h_i \boldsymbol{f}_i$. The fine solution takes two steps of size $h_i/2$, recomputing the vector field at the intermediate state. We define the absolute local-error RMS as $E_i = \text{RMS}(\boldsymbol{x}_{i+1}^{\text{coarse}} - \boldsymbol{x}_{i+1}^{\text{fine}})$. Each point in Fig. 2c represents one valid

Algorithm 1 Double-Shift with acceleration-driven online scheduling

Require: Initial latent $\boldsymbol{z}_0$, model $\boldsymbol{v}_{\boldsymbol{\theta}}$, solver $\mathcal{S}$; $N, N_e, t_e, s, s_e, \eta, \rho, \epsilon$
Ensure: Final latent $\boldsymbol{z}_N$

Construct the preset grid $\{\bar{t}_i\}_{i=0}^{N}$ using Eq. 5
$t_0 \leftarrow \bar{t}_0$
for $i = 0, 1$ **do** ▷ Preset warm-up steps
  $\boldsymbol{v}_i \leftarrow \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, t_i, \boldsymbol{c})$
  $t_{i+1} \leftarrow \bar{t}_{i+1}$
  $\boldsymbol{z}_{i+1} \leftarrow \mathcal{S}(\boldsymbol{z}_i, \boldsymbol{v}_i, t_i, t_{i+1})$
end for
for $i = 2, \ldots, N_e - 1$ **do** ▷ Adaptive early stage
  $\boldsymbol{v}_i \leftarrow \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, t_i, \boldsymbol{c})$
  $d \leftarrow t_i - t_e$
  if $i = N_e - 1$ **then**
    $h_i \leftarrow d$ ▷ End exactly at t_e
  else
    $\boldsymbol{a}_{i-1} \leftarrow (\boldsymbol{v}_{i-1} - \boldsymbol{v}_{i-2})/(t_{i-1} - t_{i-2})$
    $\widehat{A}_i \leftarrow \rho \|\boldsymbol{a}_{i-1}\|_2$
    $\widetilde{h}_i \leftarrow \sqrt{2\eta / \max(\widehat{A}_i, \epsilon)}$
    $h_i \leftarrow \min\{\widetilde{h}_i, d, d/(N_e - i)\}$
  end if
  $t_{i+1} \leftarrow t_i - h_i$
  $\boldsymbol{z}_{i+1} \leftarrow \mathcal{S}(\boldsymbol{z}_i, \boldsymbol{v}_i, t_i, t_{i+1})$
end for
for $i = N_e, \ldots, N - 1$ **do** ▷ Preset late stage
  $\boldsymbol{v}_i \leftarrow \boldsymbol{v}_{\boldsymbol{\theta}}(\boldsymbol{z}_i, \bar{t}_i, \boldsymbol{c})$
  $\boldsymbol{z}_{i+1} \leftarrow \mathcal{S}(\boldsymbol{z}_i, \boldsymbol{v}_i, \bar{t}_i, \bar{t}_{i+1})$
end for
return $\boldsymbol{z}_N$

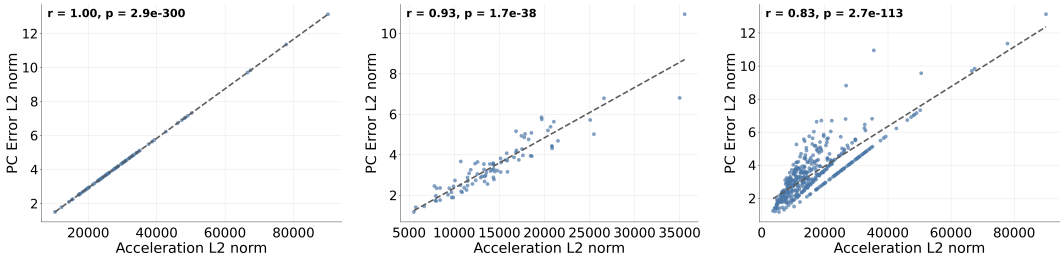
sample–step pair, plotted by its acceleration RMS and absolute local-error RMS. The scatter shows a positive association, with step-level Pearson correlation $r = 0.67$ and $p = 2.6 \times 10^{-46}$.

Implication for timestep scheduling. The step-level trend is consistent with larger trajectory acceleration accompanying larger Euler local error, supporting acceleration as a practical proxy for early-stage integration difficulty. This provides empirical motivation for the acceleration-based step-size rule in the main text. The fine solution is a more resolved numerical reference, not the exact ODE solution; moreover, acceleration is estimated from the preceding interval whereas local error is measured on the current interval, and local error also depends on step length. We therefore interpret the correlation as evidence that acceleration is informative about local error, rather than as an exact error bound.

D Algorithmic Variants and Sampling Workflow

Algorithmic variants. Double-Shift and the online scheduler can each operate independently. Double-Shift constructs the two-stage schedule in Eq. 5 using N_e and s_e, but applies the same preset timesteps to every sample. The online scheduler can instead start from a baseline schedule: it uses its first two early-stage timesteps to initialize the velocity history, chooses subsequent early-stage steps using Eq. 10, and clips a step at t_e using Eq. 11 before joining the preset late-stage grid. In this standalone form, the early-stage step count is determined by the adaptive updates, and the two warm-up timesteps cannot be controlled independently. Combining the methods gives the online scheduler a prescribed early-stage budget, boundary, and warm-up points while retaining sample-dependent timestep placement.

Combined workflow. For a fixed total budget N, the combined method proceeds as follows:



(a) Timestep 982. (b) Timestep 944. (c) Timesteps 900–1000.

Figure 7: Association between finite-difference acceleration and the UniPC predictor–corrector discrepancy. Panels (a) and (b) each contain 88 cross-sample pairs at a fixed timestep. Panel (c) pools 440 sample–step pairs across five early-stage timesteps. Dashed lines show linear fits. The pooled p-value is nominal because observations from the same sample are repeated.

1. Construct the Double-Shift grid using Eq. 5. Reserve its first two early-stage timesteps for warm-up, and keep its late-stage grid fixed.
2. Once two velocity predictions are cached, estimate acceleration from their finite difference. Apply the decay factor in Eq. 9 and compute the candidate step size using Eq. 10.
3. Let N_i be the number of completed early-stage steps, including warm-up steps. While more than one early-stage step remains, clip the candidate step to the distance from t_e using Eq. 11 and impose the remaining-budget constraint in Eq. 12. Use the resulting step to advance the solver and update the cached velocity history.
4. Set the final early-stage step to end exactly at t_e. Continue along the fixed late-stage grid until $t = 0$.

Thus, the controller redistributes exactly N_e early-stage steps for each sample, while the $N - N_e$ late-stage steps retain their preset locations. The acceleration estimate reuses model predictions already obtained during sampling and requires no additional model evaluation.

Parameter choices. The combined method tunes only N_e, s_e, and η: they control the early-stage budget, the mapping within that stage, and the candidate adaptive step size, respectively. We fix $t_e = 0.9$ as a coarse boundary for the rapidly changing early regime; the separate mapping governed by s_e reduces the need to identify its transition point precisely. The late-stage parameter s follows the baseline configuration rather than being tuned for our method. We use $s = 5$ for the Wan2.2-5B and HunyuanVideo1.5 configurations evaluated here; for the uniform timestep grid of CogVideoX-2B, $s = 1$ denotes the identity mapping in Eq. 4. We fix $\rho = 0.8$ to reflect early acceleration decay and set $\epsilon = 10^{-6}$ for numerical stability. When $\rho\|\boldsymbol{a}_{i-1}\|_2 > \epsilon$, Eq. 10 depends on η and ρ through their ratio η/ρ. We therefore fix ρ and tune η, rather than treating both as independent scheduling parameters.

E Acceleration–Error Association in Multistep Sampling

Multistep solver background. Linear multistep methods integrate the ODE in Eq. 1 using states and vector-field evaluations from multiple time points. A general form is

$$\sum_{j=0}^{k} \alpha_{i,j} \boldsymbol{x}_{i-j} = \Delta\sigma_i \sum_{j=0}^{k} \beta_{i,j} \boldsymbol{v}_{i-j}, \qquad \Delta\sigma_i = \sigma_i - \sigma_{i-1}, \tag{13}$$

where the coefficients depend on the method and timestep grid. Our analysis uses the multistep predictor–corrector solver UniPC: its predictor produces $\boldsymbol{x}_i^P$ from cached history, and its corrector incorporates the current model prediction to obtain $\boldsymbol{x}_i^C$. Since higher-order integration errors generally involve higher-order trajectory information, the Euler bound in Eq. 7 does not directly establish an error bound for UniPC. We therefore examine acceleration empirically against UniPC's predictor–corrector discrepancy.

Table 6: VBench Total score under different values of η.

Total score	79.31	79.28	**80.10**	80.00	80.05	79.40	79.42
η	1.2	1.4	1.6	1.8	2.0	2.2	2.4

Experimental setup and diagnostic quantities. We use Wan2.1-T2V-1.3B with 20 UniPC denoising steps, the baseline flow shift of 3.0, and one sample for each of 88 VBench prompts. Following Eq. 3, we estimate acceleration from consecutive CFG-combined velocity predictions, using σ as the integration coordinate. For sample b, we define

$$A_i^{(b)} = \frac{\|\boldsymbol{v}_i^{(b)} - \boldsymbol{v}_{i-1}^{(b)}\|_2}{\max\{|\sigma_i - \sigma_{i-1}|, 10^{-12}\}}, \qquad \mathrm{PC}_i^{(b)} = \|\boldsymbol{x}_i^{C,(b)} - \boldsymbol{x}_i^{P,(b)}\|_2. \tag{14}$$

Both quantities use full L2 norms over all latent coordinates, without RMS normalization. The PC discrepancy measures the corrector's state adjustment and serves as a solver-internal error indicator; it is not a directly measured local truncation error against an independent reference solution. We pair the two quantities at the same sample and denoising step, excluding the first step because neither quantity is available.

Correlation results. Fig. 7 shows strong positive associations in all three settings. At timestep 982, the cross-sample Pearson correlation is nearly perfect ($r \approx 1.00$, $p = 2.9 \times 10^{-300}$), although this initial corrector step may involve structural coupling between the two indicators. At timestep 944, the association remains strong ($r = 0.927$, $p = 1.7 \times 10^{-38}$), with consistent Spearman correlation ($r_\mathrm{s} = 0.931$, $p = 1.6 \times 10^{-39}$). This fixed-timestep analysis compares samples with the same timestep grid and step length, removing variation across timesteps from the comparison. Pooling the five timesteps in $[900, 1000]$ gives $r = 0.830$ across 440 sample–step pairs. We interpret the pooled result as descriptive evidence because repeated observations and shared timestep trends can affect its significance.

Implication for online scheduling. The fixed-timestep results show that larger finite-difference acceleration is associated with larger UniPC state corrections across samples. This supports acceleration as an empirical indicator of early-stage integration difficulty and motivates using the online rule in Eq. 10 while retaining the solver's native state updates. The improvements in the Wan experiments in Tables 1 and 2, which use multistep sampling, provide complementary evidence of practical effectiveness. Together, these results support the applicability of our acceleration-based scheduler to the tested multistep setting, without asserting a higher-order local-error bound.

F Sensitivity to the Adaptive Step-Size Parameter

Experimental setup. The parameter η controls the candidate step size of our acceleration-driven online scheduler in Eq. 10: a larger η permits a larger candidate step, whereas a smaller value favors finer sampling in the early stage. To isolate its effect, we fix the Double-Shift parameters at $N_e = 10$ and $s_e = 1$, use $N = 20$ sampling steps and a baseline shift of $s = 3.0$, and follow the main-text setting of $\rho = 0.8$ for the acceleration decay factor. We vary only η over $\{1.2, 1.4, 1.6, 1.8, 2.0, 2.2, 2.4\}$ while keeping the other generation and evaluation settings unchanged. Under the fixed early-stage budget, changing η adjusts the placement of adaptive timesteps rather than the number of early-stage steps. We report the VBench Total score.

Results analysis. As shown in Table 6, the Total score remains between 79.28 and 80.10 throughout the tested interval $\eta \in [1.2, 2.4]$, with a maximum difference of only 0.82 points. The highest score occurs at $\eta = 1.6$, but performance does not depend on selecting this exact value: $\eta = 1.8$ and 2.0 achieve scores of 80.00 and 80.05, both within 0.10 points of the maximum. At the endpoints, $\eta = 1.2$ and 2.4 still achieve 79.31 and 79.42, respectively. These results show that the method maintains high overall performance across the evaluated range and does not require precise tuning of η.