TriTD: Tri-Partite Trajectory–Distribution Distillation for Real-Time Autoregressive Video Generation

Jia Li[1,2,*], **Xurui Peng**[2,*], **Haowei Zhu**[3], **Tianyu Zhao**[2], **Jiexi Wang**[2], **Youwei Zheng**[2], **Yuxi Ren**[2], **Xinglong Wu**[2], **Xing Wang**[2,†], **Hayden Kwok-Hay So**[1,†]
[1]The University of Hong Kong [2]ByteDance Seed [3]Tsinghua University
[*]Equal contribution [†]Corresponding authors

Abstract

Autoregressive (AR) video diffusion has emerged as a leading paradigm for efficient video generation, yet its practical deployment remains bottlenecked by the trade-off between generation quality and inference efficiency. We attribute this bottleneck to three error sources in AR rollouts: the base model error, the train–inference distribution mismatch, and the upstream error propagation. Building on this decomposition, we derive closed-form upper bounds on the per-step and cumulative errors, providing a principled basis for jointly optimizing generation quality and inference efficiency. Guided by this analysis, we propose **TriTD**, a **Tri**-Partite **T**rajectory–**D**istribution co-optimized distillation framework with three complementary modules. First, for ODE initialization, we adopt a Next-Node Trajectory Distillation (NTD) objective, which shortens the supervision horizon and admits a lower regression-error floor. Second, we introduce Chunk-Graded Noising (CGN), a monotonically non-decreasing per-chunk noising schedule that helps narrow the expected train–inference noise mismatch and removes redundant KV-cache updates, yielding substantial speedups at no quality cost. Third, we propose Orthogonal-Parallel DMD (OP-DMD), a distribution-matching loss that retains the original DMD supervision while introducing an additional regression constraint in the orthogonal subspace to suppress off-axis drift, improving both training stability and final generation quality. Extensive experiments show that TriTD consistently surpasses existing baselines in visual quality while reaching a real-time throughput of 26.8 FPS on a single H100 GPU, delivering a balanced quality–efficiency solution for interactive streaming video generation.

1 Introduction

Video generation has made remarkable progress in recent years [1–5], enabling a wide range of applications such as text-to-video synthesis [6] and image-to-video animation [7–9]. Research efforts have increasingly focused on improving the visual quality, inference efficiency, and longer-horizon video generation [10, 11]. To meet these demands, autoregressive (AR) video diffusion [12–15] has emerged as a promising paradigm for real-time interactive video generation, owing to its causal, chunk-wise decoding. While recent works such as Self-Forcing [16] have made notable progress, the trade-off between generation quality and inference efficiency remains unresolved.

First, we analyze AR video generation through the lens of error propagation. We decompose the per-chunk inference error into three sources: the base model error, the train–inference distribution mismatch, and the upstream error propagation, and derive a closed-form upper bound on the cumulative error along the generated sequence. This decomposition reveals two underexplored optimization directions: lowering the base model error from its source, and reducing the context noise distribution mismatch between training and inference.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Motivated by these insights, we propose **TriTD**, a **Tri**-Partite **T**rajectory–**D**istribution co-optimized distillation framework for AR video generation. It comprises three complementary modules. (1) *Next-Node Trajectory Distillation (NTD).* For ODE initialization, we replace conventional endpoint consistency distillation with next-node supervision under an L1 regression loss, improving the initial trajectory fit. (2) *Chunk-Graded Noising (CGN).* To reduce the train–inference distribution mismatch, we introduce a noising scheme that enforces a monotonically non-decreasing training-time noise level across chunks, applied consistently across both training stages. This schedule brings training context noise closer to inference conditions and removes the redundant per-chunk KV Cache update at inference, reducing per-chunk cost without quality degradation. (3) *Orthogonal-Parallel Distribution Matching Distillation (OP-DMD).* To stabilize distribution-level training, we augment the standard DMD objective with a regression constraint on the residual component orthogonal to the DMD update direction. This auxiliary supervision suppresses off-axis drift, reduces training variance, and leads to more stable optimization and higher final generation quality.

Our main contributions are summarized as follows:

- We establish a closed-form upper-bound decomposition for the per-chunk and cumulative errors of AR video generation, offering a new analytical perspective on the quality–efficiency trade-off, and provide theoretical motivation for both NTD and CGN within our distillation framework.
- We propose **TriTD**, a tri-partite trajectory–distribution co-optimized distillation framework integrating Next-Node Trajectory Distillation (NTD), Chunk-Graded Noising (CGN), and Orthogonal-Parallel DMD (OP-DMD), whose complementary modules jointly improve generation quality and inference efficiency.
- Extensive experiments demonstrate that TriTD consistently outperforms state-of-the-art baselines while achieving a real-time throughput of 26.8 FPS on a single H100 GPU, offering a practical solution for real-time interactive video generation.

2 Related work

Diffusion models and rectified flow. Diffusion models [17–19] have become the mainstream framework for visual generation. Given a clean sample x_0 and Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$, the forward process is $x_t = \alpha_t\, x_0 + \sigma_t\, \epsilon$ with noise schedule α_t, σ_t, and the model $v_\theta(x_t, t)$ is trained by minimizing $\mathbb{E}_{t,\, x_0,\, \epsilon}\Big[\, \|v_\theta(x_t,\, t) - v^\star\|_2^2 \,\Big]$. Rectified Flow [20–22] further simplifies the forward process to a linear interpolation $x_t = (1-t)\, x_0 + t\, \epsilon$ with the corresponding denoising ODE $\frac{\mathrm{d}x_t}{\mathrm{d}t} = v_\theta(x_t,\, t)$, and has become the backbone of many modern generative models [23–25].

Efficient diffusion inference. Beyond reducing sampling steps, a broad family of training-free methods accelerates pretrained diffusion models by reusing temporally redundant features or attention outputs, selectively updating tokens, and pruning redundant conditional computation [26–31]. These methods reduce the total computation along the denoising trajectory without learning a few-step generator and are therefore complementary to diffusion distillation.

Diffusion model distillation. Iterative denoising remains a core efficiency bottleneck of diffusion models, motivating two complementary distillation paradigms. (1) **Trajectory distillation** aligns the student to the teacher denoising ODE. Progressive Distillation [32] progressively halves the number of sampling steps via cascaded teacher-student rounds. Consistency Models [33] impose the constraint directly on the ODE trajectory, enforcing that model outputs at any two points on the same trajectory remain consistent. Targeting the linear ODE structure of Rectified Flow, InstaFlow [34] and SlimFlow [35] further shorten the effective trajectory length via reflow and re-distillation, unlocking the potential of few-step generation. Trajectory distillation provides fine-grained, per-node supervision on the denoising path and exhibits a *mode-covering* (zero-avoiding) behavior. (2) **Distribution Matching Distillation** [36] minimizes the reverse-KL divergence between the student distribution p_ϕ and the teacher distribution p_{tea} for efficient distillation and has been widely used to accelerate both diffusion and flow-matching models. DMD2 [37] and subsequent works further introduce GAN-style discriminators to sharpen high-frequency details. As a reverse-KL objective, DMD is *mode-seeking* (zero-forcing) and provides weak per-node supervision. Moreover,

in autoregressive video generation, the training-time context distribution from DMD self-rollout typically diverges from the inference-time context, inducing a non-trivial train–inference gap.

Distilled autoregressive video generation. Recently, causal AR diffusion has emerged as a popular research direction in video generation, performing chunk-by-chunk sequential generation with block-causal attention [12–14, 38, 39, 11]. Among distillation-based approaches, CausVid [40] combines few-step distillation with block-causal AR decoding, Self-Forcing [16] simulates the AR self-rollout process during training to alleviate exposure bias, and Causal Forcing [15] further strengthens generation dynamics with an AR teacher. Long-horizon inference introduces additional challenges from accumulated rollout errors and an increasingly expensive context cache. Recent work explores training-free horizon extension, context compression, and memory-efficient rollout mechanisms [10, 41, 11]. These inference-time techniques are broadly applicable to and complementary with existing chunk-wise AR video generators, including CausVid, Self-Forcing, and Causal-Forcing. However, existing distilled AR methods do not explicitly model the inter-chunk context-noise distribution and typically require an additional forward pass per chunk to refresh the KV cache.

3 Method

3.1 Preliminary

We adopt the chunk-wise autoregressive paradigm [40, 16], partitioning a video into C equal-length chunks $\{X^{(1)}, X^{(2)}, \ldots, X^{(C)}\}$ and factorizing the joint distribution as

$$p\Big(X^{(1:C)}\Big) = \prod_{c=1}^{C} p\Big(X^{(c)} \,\big|\, X^{(<c)}\Big)\,. \tag{1}$$

Each conditional $p\big(X^{(c)} \mid X^{(<c)}\big)$ is modeled by a few-step student Φ_θ. For chunk c, start from pure noise $X^{(c)}_{t_1} \sim \mathcal{N}(0, I)$ and denoise at K descending timesteps $t_1 = 1000 > \cdots > t_K > t_{K+1} = 0$. Each update ($k = 1, \ldots, K$) is conditioned on the historical context $X^{(<c)}$:

$$X^{(c)}_{t_{k+1}} = \Phi_\theta\Big(X^{(c)}_{t_k},\, t_k;\, X^{(<c)}\Big)\,. \tag{2}$$

Historical context as condition. To avoid recomputing historical chunks, the student adopts block-causal attention and injects the historical context with KV Cache $\mathcal{K}^{(<c)}$. Let $t^c_{\text{exit}} \in \{t_1, \ldots, t_K\}$ denote the actual exit timestep of the denoising chain for the c-th chunk, and let $\hat{X}^{(c)}_0$ denote the one-step prediction obtained by mapping $X^{(c)}_{t^c_{\text{exit}}}$ directly to $t = 0$ via Φ_θ. Denoting the KV-cache update operator as Ψ, at baseline inference an additional forward pass is performed after the denoising chain terminates with $\hat{X}^{(c)}_0$ at $t = 0$ to refresh the KV cache:

$$\mathcal{K}^{(\le c)} = \Psi\Big(\hat{X}^{(c)}_0,\, t = 0;\, \mathcal{K}^{(<c)}\Big)\,. \tag{3}$$

When $t^c_{\text{exit}} \neq t_K$, the denoising chain is incomplete and $\hat{X}^{(c)}_0$ deviates from the ideal $X^{(c)}_0$ by a residual error correlated with t^c_{exit}, which in turn affects the noise level and distribution of the KV Cache that conditions subsequent chunks. The original Self-Forcing [16] samples t^c_{exit} uniformly at random during training for efficiency, so the full denoising chain is completed only with probability $1/K$, whereas at inference the full chain is always executed, i.e., $t^c_{\text{exit}} = t_K$. This systematic mismatch in the distribution of t^c_{exit} between training and inference constitutes one of the sources of the train-inference distribution mismatch.

3.2 Cumulative error bound for autoregressive video diffusion

In this section, we establish a closed-form upper bound on the cumulative error of AR video generation under an idealized error model, providing theoretical motivation for our design. For analytical convenience, let $\sigma_{\text{tr}}(c)$ and $\sigma_{\text{inf}}(c)$ denote the training-time and inference-time context noise levels of the c-th chunk, respectively, modeled as monotonically related to the exit timesteps $t^c_{\text{exit,tr}}$ and $t^c_{\text{exit,inf}}$ defined in Section 3.1, where a larger exit timestep induces a larger residual prediction error and a larger σ.

We observe that the single-step inference error e_c at the c-th chunk admits a three-way decomposition: (i) the *base model error* ε, a uniform bound on the model's residual error under matched, error-free context in this analysis; (ii) the *train–inference distribution mismatch* arising from the gap $|\sigma_{\text{inf}}(c) - \sigma_{\text{tr}}(c)|$ between the inference-time and training-time context noise levels; and (iii) the *upstream error propagation* term inherited from the cumulative error of the previous $c-1$ chunks via the KV Cache. The following lemma formalizes this decomposition.

Lemma 1 (Single-step error upper bound)**.** *Assume the matched-context residual is bounded by ε, noise-level perturbations contribute at most $\kappa|\sigma_{\text{inf}}(c) - \sigma_{\text{tr}}(c)|$, and the upstream context-error contribution is bounded by Le_{c-1}, with $e_0 = 0$. Then*

$$e_c \le \varepsilon + \kappa \cdot |\sigma_{\text{inf}}(c) - \sigma_{\text{tr}}(c)| + L \cdot e_{c-1}, \qquad L \ge 0,\ \kappa \ge 0, \tag{4}$$

where κ scales the distribution-mismatch contribution and L is the Lipschitz constant of Φ_θ.

Proof sketch. Apply the triangle inequality to the matched-context residual, noise-level perturbation, and upstream context perturbation. The stated residual and Lipschitz-type bounds give Eq. 4. □

Recursively unrolling Lemma 1 along the chunk axis and summing yields the cumulative bound.

Corollary 1 (Cumulative error bound)**.** *The total error $E_C = \sum_{c=1}^{C} e_c$ satisfies*

$$E_C \ \le \ \sum_{c=1}^{C} \underbrace{\left(\frac{1 - L^{C-c+1}}{1 - L}\right)}_{w_c} \Big(\varepsilon \ + \ \kappa \cdot \big|\sigma_{\text{inf}}(c) - \sigma_{\text{tr}}(c)\big|\Big), \tag{5}$$

where the position-dependent weight $w_c = (1 - L^{C-c+1})/(1 - L)$ captures error amplification along the autoregressive chain: errors introduced at smaller c propagate longer and therefore receive larger weight. The complete derivation, together with the corresponding closed forms of w_c for $L = 1$ and $L > 1$ and a discussion of their effect on the bound, is given in Appendix A.1.

The upper bound in Equation (5) decomposes into two terms:

$$\underbrace{T \ = \ \sum_{c=1}^{C} w_c \cdot \varepsilon,}_{\text{Model-error term}} \qquad \underbrace{N \ = \ \sum_{c=1}^{C} w_c \cdot \kappa \cdot \big|\sigma_{\text{inf}}(c) - \sigma_{\text{tr}}(c)\big|,}_{\text{Noise-mismatch term}} \tag{6}$$

which jointly characterize the optimization surface of autoregressive video distillation.

3.3 TriTD: a tri-partite trajectory-distribution framework

Guided by this analysis, we propose **TriTD**, a tri-partite trajectory-distribution co-optimization framework for autoregressive video distillation. As illustrated in Figure 1, TriTD (i) reshapes the training objective of ODE-initialization stage with Next-Node Trajectory Distillation (NTD) to reduce the base model error ε; (ii) introduces a chunk-graded noising training strategy that shrinks the train-inference noise distribution mismatch; and (iii) adopts the Orthogonal-Parallel DMD (OP-DMD) loss to mitigate unsupervised off-axis drift and training oscillation. The regression-error analysis motivates NTD's next-node supervision, while the expected mismatch reduction motivates Chunk-Graded Noising. OP-DMD complements these components with distribution-level optimization control.

3.3.1 NTD: tightening the base-model error ε

Existing ODE-initialization methods typically adopt an *endpoint-consistency* objective: at any intermediate timestep t_i, student Φ_θ is required to regress the teacher's clean prediction $x_0^\star$ at $t = 0$:

$$\mathcal{L}_{\text{ode-init}} \ = \ \mathbb{E}_{c,\,i}\Big[\ \big\|\ \Phi_\theta(x_{t_i}) - x_0^\star\ \big\|_2^2\Big]. \tag{7}$$

We propose to replace the endpoint target with the *next node* prediction $x^\star_{t_{i+1}}$ on the teacher trajectory and adopt L_1 instead of L_2. The *Next-Node Trajectory Distillation* (NTD) loss is formulated as:

$$\mathcal{L}_{\text{NTD}} = \mathbb{E}_{c,\,i}\Big[\ \big\|\ \Phi_\theta(x_{t_i}) - x^\star_{t_{i+1}}\ \big\|_1\Big]. \tag{8}$$

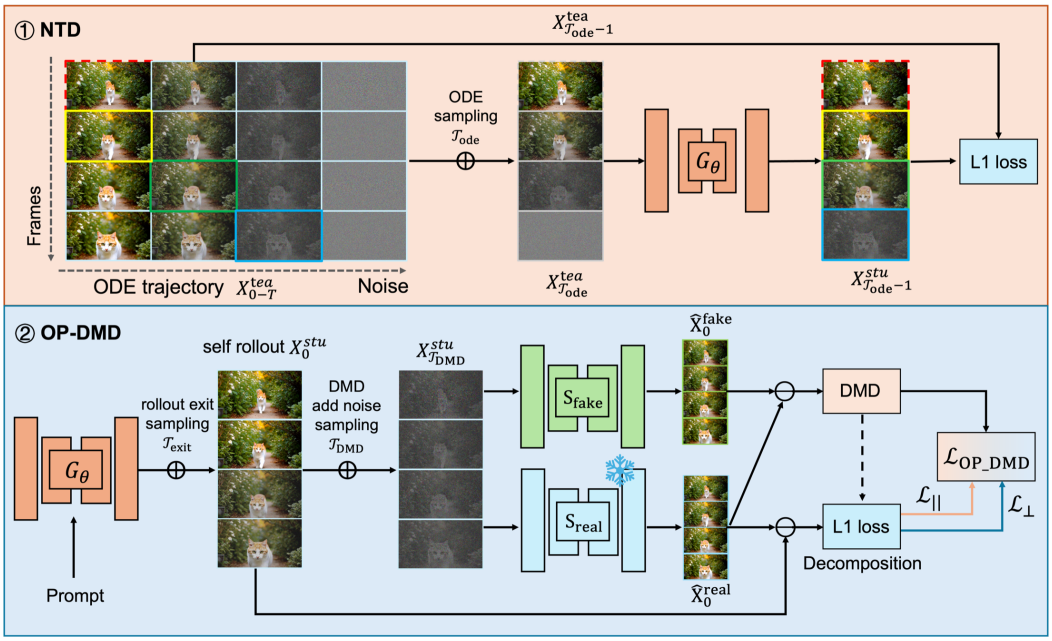

Figure 1: Overview of the proposed TriTD Framework, including (1) NTD-based ODE initialization stage and (2) OP-DMD-based distribution distillation stage. Chunk-Graded noising training is consistently applied throughout both stages. Frames enclosed by solid boxes of the same color form paired student-teacher loss, while frames in red dashed boxes are excluded from loss computation.

Whereas Equation (7) regresses the endpoint with MSE, NTD uses the next saved teacher ODE node and an outlier-robust L_1 loss. To motivate this choice, we analyze the regression-error floors associated with the two objectives.

Lemma 2 (Comparison of regression-error floors)**.** *Assume a locally affine teacher target relation and coordinate-wise Gaussian endpoint conditionals, as specified in Appendix A.2. At a fixed node i, let $\varepsilon^{\text{lower}}_{\text{init},i}$ be the average coordinate-wise optimal conditional RMSE for endpoint prediction, and $\varepsilon^{\text{lower}}_{\text{NTD},i}$ the corresponding optimal MAE for next-node prediction. Then*

$$\varepsilon^{\text{lower}}_{\text{NTD},i} = \lambda_i \sqrt{\tfrac{2}{\pi}}\, \varepsilon^{\text{lower}}_{\text{init},i} < \varepsilon^{\text{lower}}_{\text{init},i}, \tag{9}$$

where $\lambda_i = 1 - t_{i+1}/t_i \in [0,1]$ and the strict inequality requires $\varepsilon^{\text{lower}}_{\text{init},i} > 0$. Averaging over nodes also preserves the strict comparison whenever the average endpoint floor is positive.

Proof sketch. For a scalar Gaussian conditional with standard deviation s, the optimal conditional RMSE is s and the optimal MAE is $s\sqrt{2/\pi}$. The affine target relation scales the conditional standard deviation by λ_i, yielding Equation (9). This connects next-node target contraction to the coordinate-wise regression-error floors. See Appendix A.2 for the full derivation and interpretation. □

3.3.2 Chunk-Graded Noising: compressing the noise mismatch N

Existing distillation frameworks like Self-Forcing sample the per-chunk noise level independently during training: in the trajectory-distillation stage the starting node timestep t_c is obtained at random, and in the distribution-matching stage the exit timestep t^c_{exit} is also sampled at random from $\{t_1, \ldots, t_K\}$. The induced training-time noise level distribution $\sigma_{\text{tr}}(c)$ can therefore deviate from the inference-time level $\sigma_{\text{inf}}(c)$ as context errors accumulate across chunks. This deviation $|\sigma_{\text{inf}}(c) - \sigma_{\text{tr}}(c)|$ contributes to the mismatch term N in Equation (6).

We compress this mismatch by imposing two constraints on per-chunk noise sampling. (1) **Clean Start.** The first chunk ($c = 1$) is generated without contamination from previous chunks, so its inference-time noise level is identically zero. We mirror this at training by enforcing $\sigma_{\text{tr}}(1) = 0$ in both stages, removing mismatch on the first chunk. Since the geometric weights w_c decrease monotonically with c, eliminating mismatch at early chunks substantially reduces the cumulative

Algorithm 1 One generator update with NTD + Chunk-Graded Noising.

Require: Student Φ_θ, teacher trajectories full $\{z^{\text{tea}}_{t_k,c}\}_{k=0,\,c=1}^{T,\,C}$, student nodes $\{\tau_i\}_{i=1}^{N+1}$ ($\tau_{N+1}=0$), chunk count C, learning rate η.

Sample $s_c \sim \mathcal{U}\{1,\dots,N\}$ for $c=1,\dots,C$
$(s_1,\dots,s_C) \leftarrow \text{sort}_\downarrow(s_1,\dots,s_C)$
Init start-point:
    $x_{\tau_{s_c},c} \leftarrow z^{\text{tea}}_{\tau_{s_c},c}$ for $c=1,\dots,C$
$\{x_{\tau_{s_c+1},c}\}_{c=1}^{C} \leftarrow \Phi_\theta\big(\{x_{\tau_{s_c},c}\}_{c=1}^{C}, \{\tau_{s_c}\}_{c=1}^{C}\big)$
$\mathcal{L}_{\text{NTD}} \leftarrow \sum_{c=1}^{C} \big\| x_{\tau_{s_c+1},c} - z^{\text{tea}}_{\tau_{s_c+1},c} \big\|_1$
$\theta \leftarrow \theta - \eta\, \nabla_\theta \mathcal{L}_{\text{NTD}}$

Algorithm 2 OP-DMD Loss

Require: Student Φ_θ, real score s_{real}, fake score s_{fake}, range $[t_{\min}, t_{\max}]$, weights $\lambda_\perp, \lambda_\parallel$.

$\hat{x}_0^{\text{stu}} \leftarrow \text{GENERATOR}(\Phi_\theta)$
Sample $t \sim \mathcal{U}[t_{\min}, t_{\max}]$, $\epsilon \sim \mathcal{N}(0, I)$
$x_t \leftarrow (1-t)\,\hat{x}_0^{\text{stu}} + t\,\epsilon$
$\hat{x}_0^{\text{real}} \leftarrow s_{\text{real}}(x_t, t)$; $\hat{x}_0^{\text{fake}} \leftarrow s_{\text{fake}}(x_t, t)$
$r \leftarrow \hat{x}_0^{\text{real}} - \hat{x}_0^{\text{fake}}$; $\hat{\mu}_{\text{DMD}} \leftarrow \text{sg}(r/\|r\|)$
$\delta \leftarrow \hat{x}_0^{\text{stu}} - \hat{x}_0^{\text{real}}$; $\alpha \leftarrow \langle \hat{\mu}_{\text{DMD}}, \delta \rangle$
$\delta_\parallel \leftarrow \alpha\, \hat{\mu}_{\text{DMD}}$; $\delta_\perp \leftarrow \delta - \delta_\parallel$
$\mathcal{L} \leftarrow \mathcal{L}_{\text{DMD}} + \lambda_\perp \mathcal{R}(\delta_\perp) + \lambda_\parallel \mathcal{R}(\alpha)$
$\theta \leftarrow \theta - \eta\, \nabla_\theta \mathcal{L}$
Update s_{fake}

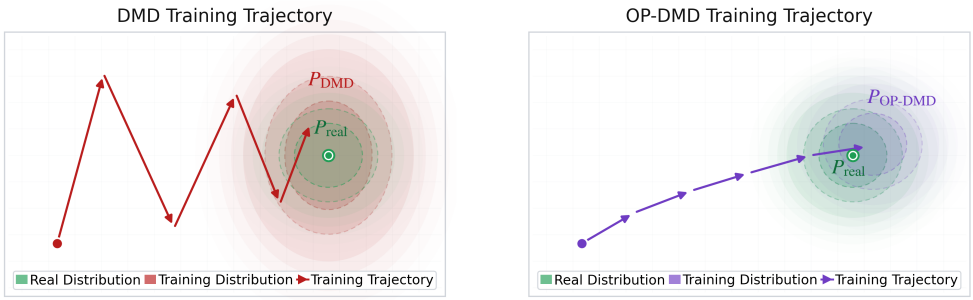

Figure 2: Training-trajectory comparison in a 2D latent toy. **Left:** Standard DMD with training oscillation and unconstrained subspace shift between P_{DMD} and P_{real}. **Right:** OP-DMD with orthogonal penalty constraints, yields a stable training trajectory toward P_{real}.

mismatch error N. (2) **Sorted Noise.** For subsequent chunks ($c>1$), we sample per-chunk noise level under a non-decreasing constraint: (i) during NTD ODE stage, the sampled startpoint timesteps satisfy $t_c \geq t_{c-1}$; (ii) during distribution matching stage, chunk-wise exit timestep satisfies $t^c_{\text{exit}} \geq t^{c-1}_{\text{exit}}$. Both constraints induce a monotonically non-decreasing $\sigma_{\text{tr}}(c)$, mirroring the natural error accumulation along the chunk axis at inference.

Lemma 3 (Reduction of expected noise mismatch)**.** *Let N_{ours} and N_{SF} denote the cumulative mismatch of CGN and the baseline, respectively. Under the effective-noise model and conditions in Appendix A.3,*

$$\mathbb{E}[N_{\text{ours}}] < \mathbb{E}[N_{\text{SF}}]. \tag{10}$$

The derivation is given in Appendix A.3. We integrate the Chunk-Graded Noising schedule with NTD to formulate the ODE initialization regression training. The comprehensive training algorithm is presented in Algorithm 1. By training the model to be robust to noisy context, we shrink the train-inference distribution shift. As a result, at inference time we can *completely skip* the post-chunk KV-cache re-computation step, which dramatically reduces the computation cost and boosts actual generation speed.

3.3.3 Orthogonal-Parallel DMD (OP-DMD): narrowing the realized-to-theoretical gap

The standard DMD loss updates the student generator with the gradient signal $r = \hat{x}_0^{\text{real}} - \hat{x}_0^{\text{fake}}$, which provides KL supervision along the DMD update direction $\hat{\mu}_{\text{DMD}}$ in the latent space. However, the orthogonal subspace receives no explicit supervision, which induces potential shift in the training distribution and oscillation in the optimization trajectory. We refer to this effect as *off-axis drift*.

Based on this observation, we propose *Orthogonal-Parallel DMD* (OP-DMD), which decomposes the student-teacher residual along $\hat{\mu}_{\text{DMD}}$ and applies separate supervision on the parallel and orthogonal components. As shown in Algorithm 2, $\hat{\mu}_{\text{DMD}} = r/\|r\|$ and $\delta = \hat{x}_0^{\text{student}} - \hat{x}_0^{\text{real}}$, then OP-DMD decomposes δ along $\hat{\mu}_{\text{DMD}}$:

$$\delta = \delta_\parallel + \delta_\perp, \quad \delta_\parallel = \langle \hat{\mu}_{\text{DMD}}, \delta \rangle\, \hat{\mu}_{\text{DMD}}, \quad \delta_\perp = \delta - \delta_\parallel, \quad \langle \hat{\mu}_{\text{DMD}}, \delta_\perp \rangle = 0. \tag{11}$$

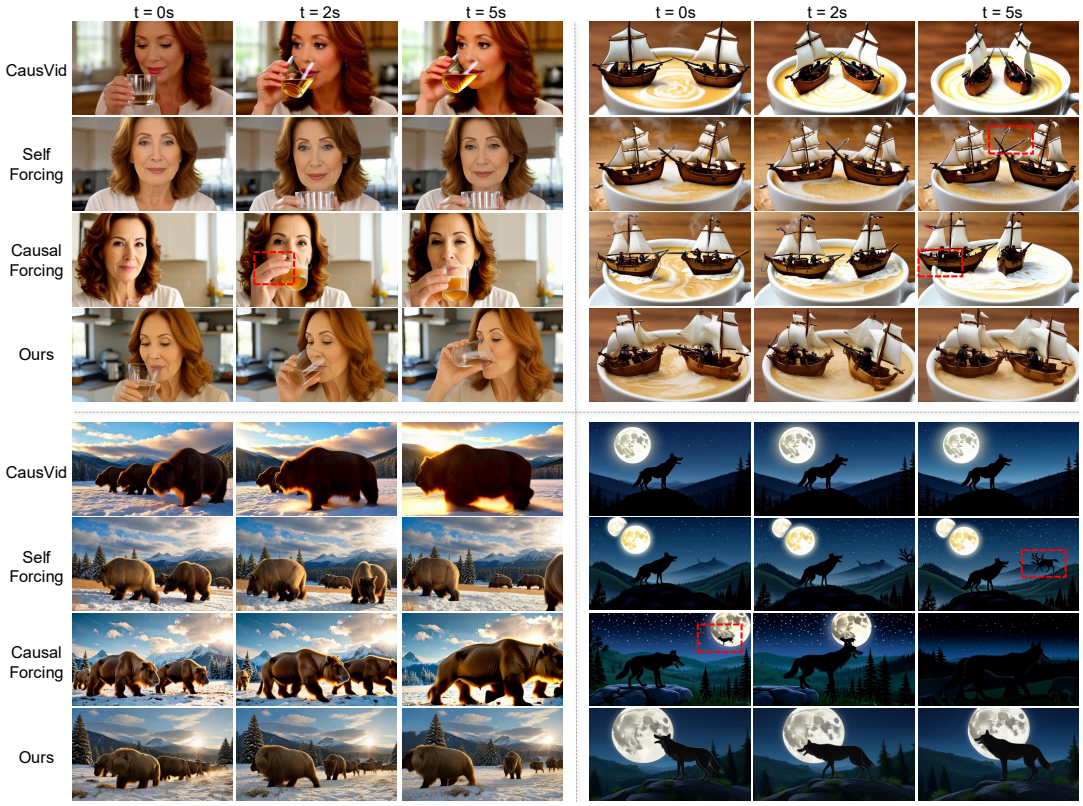

Figure 3: Qualitative comparison against three distillation-based AR baselines (CausVid, Self-Forcing, Causal-Forcing). Each row shows key frames sampled from 5-second generation.

Geometrically, $\delta_{\parallel}$ measures how much $\hat{x}_0^{\text{student}}$ deviates from the teacher prediction along the KL-improving direction, while $\delta_{\perp}$ captures drift in the subspace unconstrained by DMD. The OP-DMD objective augments the standard DMD loss with pointwise penalties on both components:

$$\mathcal{L}_{\text{OP-DMD}} = \mathcal{L}_{\text{DMD}} + \lambda_{\perp}\,\mathcal{R}(\delta_{\perp}) + \lambda_{\parallel}\,\mathcal{R}(\langle \hat{\mu}_{\text{DMD}}, \delta \rangle), \quad \lambda_{\parallel} \ll \lambda_{\perp}, \tag{12}$$

where $\mathcal{R}(\cdot)$ denotes an ℓ_1 regression loss. As shown in Figure 2 (right), the orthogonal term shrinks $\delta_{\perp}$ in the DMD-unsupervised subspace and constrains the orthogonal directions that would otherwise drift, suppressing off-axis drift. The parallel term lightly penalizes the on-axis projection coefficient (with $\lambda_{\parallel} \ll \lambda_{\perp}$) and stabilizes training as $\|r\| \to 0$. A full OP-DMD step is described in Algorithm 2.

4 Experiments

4.1 Experimental setup

Implementation. We adopt Wan2.1-1.3B [5] as our base model, which generates 5-second videos by default at 832×480 resolution. Following CausVid [40], we first fine-tune the model with block-causal attention masking on 6K teacher ODE pairs, using prompts sampled from MixKit. Our student model runs 4 denoising steps per chunk, and each chunk consists of 3 frames. The OP-DMD loss coefficients $\lambda_{\parallel}$ and $\lambda_{\perp}$ are set to 0.0001 and 0.01, respectively. Additional implementation details including hyper-parameter settings are provided in Section A.4.

Evaluation & baselines. We compare our method against representative open-source video generation models. (1) *Bidirectional video diffusion models* such as Wan2.1-1.3B [5] and LTX-Video-1.9B [24]. (2) *Autoregressive video diffusion models* like SkyReels-V2-DF-540P [38], MAGI-1 (base) [39]. In particular, we compare with three distillation-based autoregressive baselines: CausVid [40], Self-Forcing [16], and the recent work Causal-Forcing [15]. We evaluate all methods with a 5-second generation horizon on VBench [42] benchmark to assess the generation quality. Following CausVid [40], we use the first 128 prompts from MovieGen to generate videos. We report

Table 1: Quantitative comparison on VBench using the first 128 MovieGen prompts under a 5-second generation horizon. Throughput (FPS) is measured on a single NVIDIA H100 GPU.

Method	FPS w/ compile	FPS w/o compile	Subject Cons.	Background Cons.	Dynamic Degree	Image Quality	Motion Smooth.	Aesthetic Quality	Temporal Flicker.	**Quality Score**
Bidirectional Video Diffusion Models										
Wan2.1	1.0	0.8	94.16	96.21	57.03	66.64	98.19	64.71	96.88	82.17
LTX-Video	10.8	9.0	92.69	95.04	41.41	63.07	98.92	55.22	97.49	79.10
Autoregressive Video Diffusion Models										
SkyReels-V2	0.6	0.5	95.87	96.86	32.81	62.51	98.72	63.45	97.91	80.63
MAGI-1	0.3	0.2	96.29	96.50	35.94	61.71	99.22	64.76	98.39	81.42
Distilled Autoregressive Video Models										
CausVid	20.3	16.9	94.65	94.51	45.31	68.00	98.27	62.10	97.06	80.93
Self-Forcing	20.3	16.9	93.53	94.62	63.28	69.04	98.27	63.36	96.70	82.33
Causal-Forcing	20.3	16.9	93.99	94.93	78.91	70.73	97.43	64.03	95.02	82.91
Ours	**26.8**	**22.2**	94.25	94.66	82.03	69.25	98.26	63.46	95.82	**83.59**

speed test results both with and without `torch.compile` optimizations. FPS measures diffusion-backbone throughput, excluding VAE decoding.

4.2 Experiment results

Qualitative comparison. Figure 3 shows key frames from 5-second videos. CausVid exhibits notable chunk-level *error accumulation*, manifesting as severe color oversaturation and degraded image quality. Self-Forcing mitigates exposure bias through self-rollout and accumulates less error than CausVid, yet still suffers from artifacts. Causal-Forcing, while achieving stronger dynamic degree, is more prone to over-saturated colors and exhibits unrealistic spatial arrangements and structural artifacts. In contrast, our method preserves stable subject identity and image quality throughout the sequence. More visualization results and comparison are provided in Section C.

Quantitative comparison. For chunk-wise autoregressive baselines, though they adopt 4-step generation, they actually require 5-NFE to finish single chunk generation, as an extra forward pass is required after each chunk's 4 denoising steps to construct a clean KV-cache (higher quality but higher latency). In contrast, our method uses only 4 NFEs per chunk, eliminating the extra forward pass. Table 1 shows that our method achieves the best quality score of **83.59** and the highest throughput, reaching **26.8/22.2** FPS (with/without compilation). Notably, with only 4 NFE, our method still consistently outperforms all distilled autoregressive baselines with 5-NFE. For example, surpassing Self-Forcing and Causal-Forcing by $+1.26$ and $+0.68$ in Quality Score, respectively, delivering impressive quality-efficiency trade-off among existing approaches.

Human preference. To complement the automatic metrics, we conducted a blinded pairwise evaluation on 100 randomly sampled VBench prompts with five evaluators. As shown in Table 2, TriTD receives the majority of preference votes against all three distilled AR baselines despite requiring fewer effective NFEs. The complete evaluation protocol and absolute-rating results are provided in Section A.5.

Table 2: Pairwise human preference on 100 VBench prompts (500 votes per comparison).

Comparison	TriTD (4 NFE)	Baseline (5 NFE)
vs. CausVid	**438 (87.6%)**	62 (12.4%)
vs. Self-Forcing	**323 (64.6%)**	177 (35.4%)
vs. Causal-Forcing	**354 (70.8%)**	146 (29.2%)

4.3 Ablation and analysis

Ablation study. We further ablate the three components on top of the Self-Forcing-like baseline and report the Quality Score on MovieGen prompts in Table 3. Replacing the endpoint regression at the ODE-init stage with our NTD L1 trajectory distillation (M1) lifts the baseline from 81.82 to 82.07, showing the benefit of next-node supervision for ODE initialization. Applying chunk-graded noising schedule with M1 strategy (M2) further pushes the score to 82.10 and, more importantly, reduces the train-inference KV-cache noise mismatch, allowing the student to skip the per-chunk KV-cache refresh at inference and thereby enabling the generation speed-up reported in Table 1. Finally, replacing the standard DMD objective in M2 strategy with OP-DMD (M3) yields the largest jump to **83.59**. By imposing a pointwise constraint on the DMD-orthogonal subspace, OP-DMD suppresses off-axis drift and complements the trajectory and context-distribution objectives motivated by our analysis.

Table 3: Ablation of the three proposed components.

#	Variant	NTD	CGN	OP-DMD	Quality ↑
M0	Baseline	✗	✗	✗	81.82
M1	+ NTD init	✓	✗	✗	82.07
M2	+ CGN	✓	✓	✗	82.10
M3	+ OP-DMD	✓	✓	✓	**83.59**

Efficiency comparison & breakdown. As shown in Figure 4, the VAE decoding time remains nearly constant at ≈ 1.74 s across all settings, indicating that the gain is almost entirely due to the diffusion backbone. In the SF baseline, each chunk requires four denoising forward passes and one additional forward pass to refresh the KV-cache, resulting in 4.79 s diffusion cost and 6.53 s total latency per video. Applying `torch.compile` to SF alone reduces the diffusion cost to 3.98 s ($1.20\times$ core speedup) and the total latency to 5.73 s ($1.14\times$ end-to-end speedup).

Because the chunk-graded noising schedule trains the student to remain robust to noisy KV Cache context, our method can directly reuse the noisy KV Cache produced by the last denoising step at inference and skip the refresh pass entirely. This reduces the diffusion cost to 3.65 s ($1.31\times$ core speedup) and the end-to-end latency to 5.39 s ($1.21\times$ end-to-end speedup) without compilation. Combined with `torch.compile`, the diffusion cost further drops to 3.02 s ($1.58\times$ core speedup), yielding 4.77 s total latency per video ($1.37\times$ end-to-end speedup) and the 26.8 FPS reported in Table 1. Notably, our uncompiled variant already outperforms the compiled SF baseline (5.39 s vs. 5.73 s), suggesting that the gain mainly comes from the algorithmic redesign of the KV-cache pipeline rather than compilation-level engineering.

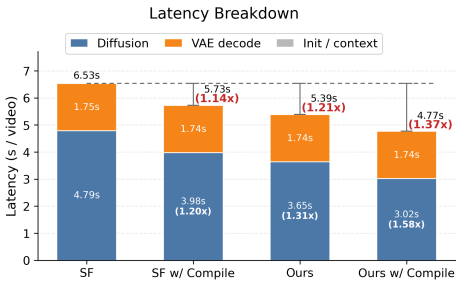

Figure 4: **Latency breakdown for four settings.**

Long-horizon generalization. Although trained on 5-second clips, TriTD outperforms baselines when the inference horizon is extended to 60 seconds.

As shown in Table 4, the two baselines exhibit qualitatively distinct failure modes: Self-Forcing falls into near-static rollouts, behaving as low dynamic degree and high motion smoothness in return. Causal-Forcing exhibits rapid visual breakdown, of which both image quality and motion smoothness significantly degrade. In contrast, TriTD retains the highest quality score at 60 seconds, supporting its stability over longer generation horizons. Full-dimension VBench metrics are provided in Appendix B.

Table 4: **Long-horizon generation over 60 seconds using MovieGen prompts.**

Method	Dyn. Deg.	Motion Sm.	Quality
Self-Forcing	33.26	98.11	80.35
Causal-Forcing	46.30	95.97	77.01
Ours	**85.23**	**97.66**	**82.64**

5 Conclusion and limitations

We presented **TriTD**, a tri-partite trajectory-distribution co-optimized distillation framework for real-time autoregressive video diffusion. Starting from a decomposition of the per-chunk inference error into base model error, train-inference noise mismatch, and upstream error propagation, we derive a closed-form upper bound on cumulative error that guides three complementary modules: NTD uses next-node trajectory supervision to improve ODE initialization; CGN aligns training context noise with inference and enables KV-cache reuse; and OP-DMD constrains the DMD-orthogonal subspace to stabilize optimization. As a result, our 4-NFE generator attains the highest VBench quality score among existing AR models and reaches 26.8 FPS on a single H100, a $1.32\times$ speedup over Self-Forcing, pushing AR video generation into the real-time interactive regime.

Limitations. TriTD targets chunk-wise video diffusion distillation and is designed to establish a stable and efficient training paradigm rather than to address minute-scale long-video synthesis, for which dedicated mechanisms (e.g., Deep Forcing [43]) are likely complementary to our framework and represent a promising direction for future work. In addition, our training data consists of 6K prompts from MixKit, substantially smaller than the 16K setting used in Self-Forcing and related baselines. Scaling up training data is therefore a natural direction for further improving TriTD. Following common practice in autoregressive video distillation [16, 15], all experiments are conducted on Wan2.1. Verifying TriTD on additional backbones and exploring its transferability to other generative tasks (e.g., image and audio synthesis) are also left for future investigation.

Acknowledgments and Disclosure of Funding

This work was carried out using resources provided by the authors' affiliated institutions. No external funding was received. The authors declare no competing interests.

References

[1] OpenAI. Video generation models as world simulators. `https://openai.com/index/video-generation-models-as-world-simulators/`, 2024. Accessed: 2024-05-07.

[2] Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, David Yan, Dhruv Choudhary, Dingkang Wang, Geet Sethi, Guan Pang, Haoyu Ma, Ishan Misra, Ji Hou, Jialiang Wang, Kiran Jagadeesh, Kunpeng Li, Luxin Zhang, Mannat Singh, Mary Williamson, Matt Le, Matthew Yu, Mitesh Kumar Singh, Peizhao Zhang, Peter Vajda, Quentin Duval, Rohit Girdhar, Roshan Sumbaly, Sai Saketh Rambhatla, Sam Tsai, Samaneh Azadi, Samyak Datta, Sanyuan Chen, Sean Bell, Sharadh Ramaswamy, Shelly Sheynin, Siddharth Bhattacharya, Simran Motwani, Tao Xu, Tianhe Li, Tingbo Hou, Wei-Ning Hsu, Xi Yin, Xiaoliang Dai, Yaniv Taigman, Yaqiao Luo, Yen-Cheng Liu, Yi-Chiao Wu, Yue Zhao, Yuval Kirstain, Zecheng He, Zijian He, Albert Pumarola, Ali Thabet, Artsiom Sanakoyeu, Arun Mallya, Baishan Guo, Boris Araya, Breena Kerr, Carleigh Wood, Ce Liu, Cen Peng, Dimitry Vengertsev, Edgar Schonfeld, Elliot Blanchard, Felix Juefei-Xu, Fraylie Nord, Jeff Liang, John Hoffman, Jonas Kohler, Kaolin Fire, Karthik Sivakumar, Lawrence Chen, Licheng Yu, Luya Gao, Markos Georgopoulos, Rashel Moritz, Sara K. Sampson, Shikai Li, Simone Parmeggiani, Steve Fine, Tara Fowler, Vladan Petrovic, and Yuming Du. Movie gen: A cast of media foundation models, 2025. URL `https://arxiv.org/abs/2410.13720`.

[3] Kling Team, Jialu Chen, Yuanzheng Ci, Xiangyu Du, Zipeng Feng, Kun Gai, Sainan Guo, Feng Han, Jingbin He, Kang He, Xiao Hu, Xiaohua Hu, Boyuan Jiang, Fangyuan Kong, Hang Li, Jie Li, Qingyu Li, Shen Li, Xiaohan Li, Yan Li, Jiajun Liang, Borui Liao, Yiqiao Liao, Weihong Lin, Quande Liu, Xiaokun Liu, Yilun Liu, Yuliang Liu, Shun Lu, Hangyu Mao, Yunyao Mao, Haodong Ouyang, Wenyu Qin, Wanqi Shi, Xiaoyu Shi, Lianghao Su, Haozhi Sun, Peiqin Sun, Pengfei Wan, Chao Wang, Chenyu Wang, Meng Wang, Qiulin Wang, Runqi Wang, Xintao Wang, Xuebo Wang, Zekun Wang, Min Wei, Tiancheng Wen, Guohao Wu, Xiaoshi Wu, Zhenhua Wu, Da Xie, Yingtong Xiong, Yulong Xu, Sile Yang, Zikang Yang, Weicai Ye, Ziyang Yuan, Shenglong Zhang, Shuaiyu Zhang, Yuanxing Zhang, Yufan Zhang, Wenzheng Zhao, Ruiliang Zhou, Yan Zhou, Guosheng Zhu, and Yongjie Zhu. Kling-omni technical report, 2025. URL `https://arxiv.org/abs/2512.16776`.

[4] Google DeepMind. Veo 3: Generative video models. Technical report, Google DeepMind, 2025. URL `https://storage.googleapis.com/deepmind-media/veo/Veo-3-Tech-Report.pdf`.

[5] Wan Team. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[6] Guoqing Ma, Haoyang Huang, Kun Yan, Liangyu Chen, Nan Duan, Shengming Yin, Changyi Wan, Ranchen Ming, Xiaoniu Song, Xing Chen, Yu Zhou, Deshan Sun, Deyu Zhou, Jian Zhou, Kaijun Tan, Kang An, Mei Chen, Wei Ji, Qiling Wu, Wen Sun, Xin Han, Yanan Wei, Zheng Ge, Aojie Li, Bin Wang, Bizhu Huang, Bo Wang, Brian Li, Changxing Miao, Chen Xu, Chenfei Wu, Chenguang Yu, Dapeng Shi, Dingyuan Hu, Enle Liu, Gang Yu, Ge Yang, Guanzhe Huang, Gulin Yan, Haiyang Feng, Hao Nie, Haonan Jia, Hanpeng Hu, Hanqi Chen, Haolong Yan, Heng Wang, Hongcheng Guo, Huilin Xiong, Huixin Xiong, Jiahao Gong, Jianchang Wu, Jiaoren Wu, Jie Wu, Jie Yang, Jiashuai Liu, Jiashuo Li, Jingyang Zhang, Junjing Guo, Junzhe Lin, Kaixiang Li, Lei Liu, Lei Xia, Liang Zhao, Liguo Tan, Liwen Huang, Liying Shi, Ming Li, Mingliang Li, Muhua Cheng, Na Wang, Qiaohui Chen, Qinglin He, Qiuyan Liang, Quan Sun, Ran Sun, Rui Wang, Shaoliang Pang, Shiliang Yang, Sitong Liu, Siqi Liu, Shuli Gao, Tiancheng Cao, Tianyu Wang, Weipeng Ming, Wenqing He, Xu Zhao, Xuelin Zhang, Xianfang Zeng, Xiaojia Liu, Xuan Yang, Yaqi Dai, Yanbo Yu, Yang Li, Yineng Deng, Yingming Wang, Yilei Wang, Yuanwei Lu, Yu Chen, Yu Luo, Yuchu Luo, Yuhe Yin, Yuheng Feng, Yuxiang Yang, Zecheng Tang, Zekai Zhang, Zidong Yang, Binxing Jiao, Jiansheng Chen, Jing Li, Shuchang Zhou, Xiangyu Zhang, Xinhao Zhang, Yibo Zhu, Heung-Yeung Shum, and Daxin Jiang. Step-video-t2v technical report: The practice, challenges, and future of video foundation model, 2025. URL https://arxiv.org/abs/2502.10248.

[7] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. Stable video diffusion: Scaling latent video diffusion models to large datasets, 2023. URL https://arxiv.org/abs/2311.15127.

[8] Li Hu, Xin Gao, Peng Zhang, Ke Sun, Bang Zhang, and Liefeng Bo. Animate anyone: Consistent and controllable image-to-video synthesis for character animation, 2024. URL https://arxiv.org/abs/2311.17117.

[9] Li Hu, Guangyuan Wang, Zhen Shen, Xin Gao, Dechao Meng, Lian Zhuo, Peng Zhang, Bang Zhang, and Liefeng Bo. Animate anyone 2: High-fidelity character image animation with environment affordance, 2025. URL https://arxiv.org/abs/2502.06145.

[10] Jia Li, Xiaomeng Fu, Xurui Peng, Weifeng Chen, Youwei Zheng, Tianyu Zhao, Jiexi Wang, Fangmin Chen, Xing Wang, and Hayden Kwok-Hay So. Train short, inference long: Training-free horizon extension for autoregressive video generation, 2026. URL https://arxiv.org/abs/2602.14027.

[11] Shenghai Yuan, Yuanyang Yin, Zongjian Li, Xinwei Huang, Xiao Yang, and Li Yuan. Helios: Real real-time long video generation model, 2026. URL https://arxiv.org/abs/2603.04379.

[12] Mingzhen Sun, Weining Wang, Gen Li, Jiawei Liu, Jiahui Sun, Wanquan Feng, Shanshan Lao, SiYu Zhou, Qian He, and Jing Liu. Ar-diffusion: Asynchronous video generation with auto-regressive diffusion, 2025. URL https://arxiv.org/abs/2503.07418.

[13] Boyuan Chen, Diego Marti Monso, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion, 2024. URL https://arxiv.org/abs/2407.01392.

[14] Kunhao Liu, Wenbo Hu, Jiale Xu, Ying Shan, and Shijian Lu. Rolling forcing: Autoregressive long video diffusion in real time, 2025. URL https://arxiv.org/abs/2509.25161.

[15] Hongzhou Zhu, Min Zhao, Guande He, Hang Su, Chongxuan Li, and Jun Zhu. Causal forcing: Autoregressive diffusion distillation done right for high-quality real-time interactive video generation. *arXiv preprint arXiv:2602.02214*, 2026.

[16] Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. Spotlight. arXiv:2506.08009.

[17] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[18] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021.

[19] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, June 2022.

[20] 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 (ICLR)*, 2023.

[21] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling, 2023. URL https://arxiv.org/abs/2210.02747.

[22] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis, 2024. URL https://arxiv.org/abs/2403.03206.

[23] Chenfei Wu, Jiahao Li, Jingren Zhou, Junyang Lin, Kaiyuan Gao, Kun Yan, Sheng ming Yin, Shuai Bai, Xiao Xu, Yilei Chen, Yuxiang Chen, Zecheng Tang, Zekai Zhang, Zhengyi Wang, An Yang, Bowen Yu, Chen Cheng, Dayiheng Liu, Deqing Li, Hang Zhang, Hao Meng, Hu Wei, Jingyuan Ni, Kai Chen, Kuan Cao, Liang Peng, Lin Qu, Minggang Wu, Peng Wang, Shuting Yu, Tingkun Wen, Wensen Feng, Xiaoxiao Xu, Yi Wang, Yichang Zhang, Yongqiang Zhu, Yujia Wu, Yuxuan Cai, and Zenan Liu. Qwen-image technical report, 2025. URL https://arxiv.org/abs/2508.02324.

[24] Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dor Moshe, Eran Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, Poriya Panet, Sapir Weissbuch, Victor Kulikov, Yaki Bitterman, Zeev Melumian, and Ofir Bibi. Ltx-video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024.

[25] Yoav HaCohen, Benny Brazowski, Nisan Chiprut, Yaki Bitterman, Andrew Kvochko, Avishai Berkowitz, Daniel Shalem, Daphna Lifschitz, Dudu Moshe, Eitan Porat, Eitan Richardson, Guy Shiran, Itay Chachy, Jonathan Chetboun, Michael Finkelson, Michael Kupchick, Nir Zabari, Nitzan Guetta, Noa Kotler, Ofir Bibi, Ori Gordon, Poriya Panet, Roi Benita, Shahar Armon, Victor Kulikov, Yaron Inger, Yonatan Shiftan, Zeev Melumian, and Zeev Farbman. Ltx-2: Efficient joint audio-visual foundation model, 2026. URL https://arxiv.org/abs/2601.03233.

[26] Xinyin Ma, Gongfan Fang, and Xinchao Wang. Deepcache: Accelerating diffusion models for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15762–15772, 2024. URL https://arxiv.org/abs/2312.00858.

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

[28] 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 (ICLR)*, 2025. URL https://arxiv.org/abs/2410.05317.

[29] Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep embedding tells: It's time to cache for video diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. URL https://arxiv.org/abs/2411.19108.

[30] Xiaomeng Fu and Jia Li. TCFG: Truncated classifier-free guidance for efficient and scalable text-to-image acceleration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 18552–18562, October 2025. doi: 10.1109/ICCV51701.2025.01724.

[31] Haowei Zhu, Tingxuan Huang, Xing Wang, Tianyu Zhao, Jiexi Wang, Weifeng Chen, Xurui Peng, Fangmin Chen, Junhai Yong, and Bin Wang. TAP: A token-adaptive predictor framework for training-free diffusion acceleration, 2026. URL https://arxiv.org/abs/2603.03792.

[32] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations (ICLR)*, 2022.

[33] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *International Conference on Machine Learning (ICML)*, 2023.

[34] Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, and Qiang Liu. Instaflow: One-step is enough for high-quality diffusion-based text-to-image generation. In *International Conference on Learning Representations (ICLR)*, 2024.

[35] Yuanzhi Zhu, Xingchao Liu, and Qiang Liu. Slimflow: Training smaller one-step diffusion models with rectified flow. In *European Conference on Computer Vision (ECCV)*, 2024.

[36] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T. Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[37] Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Frédo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[38] SkyReels Team. Skyreels-v2: Infinite-length film generative model. *arXiv preprint arXiv:2504.13074*, 2025.

[39] Sand AI, Hansi Teng, et al. Magi-1: Autoregressive video generation at scale. *arXiv preprint arXiv:2505.13211*, 2025.

[40] Tianwei Yin, Qiang Zhang, Richard Zhang, William T. Freeman, Frédo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast autoregressive video diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[41] Xiaomeng Fu, Jia Li, Yiming Hu, Yong Wang, Hayden Kwok-Hay So, Jiao Dai, Xiangxiang Chu, and Jizhong Han. Towards memory-efficient autoregressive video generation via instance-specific parametric absorption, 2026. URL `https://arxiv.org/abs/2607.00712`.

[42] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21807–21818, 2024.

[43] Jung Yi, Wooseok Jang, Paul Hyunbin Cho, Jisu Nam, Heeji Yoon, and Seungryong Kim. Deep forcing: Training-free long video generation with deep sink and participative compression. *arXiv preprint arXiv:2512.05081*, 2025.

A Technical appendices and supplementary material

This appendix contains: (1) the detailed proofs of the lemmas and corollaries stated in Section 3.2–3.3.2 of the main text, (2) Detailed experimental settings, (3) supplementary experimental results and (4) additional visualization results and comparison with representative baselines.

A.1 Proof of Corollary 1: Total Accumulated Error Upper Bound

Step 1: Quantifying each error term.

- **Base error.** Assume the residual under matched, error-free context is uniformly bounded by ε. Here ε denotes the residual upper bound; Appendix A.2 separately analyzes optimal regression-error floors.
- **Distribution-shift error.** At inference time the model receives a KV-cache $\mathcal{K}^{\text{inf}}_{<c}$ whose noise level is $\sigma_{\text{inf}}(c)$, whereas at training time it has been trained on caches of noise level $\sigma_{\text{tr}}(c)$. The resulting mismatch is modeled as a term proportional to the absolute noise-level difference, namely $\kappa \cdot |\sigma_{\text{inf}}(c) - \sigma_{\text{tr}}(c)|$, where $\kappa > 0$ reflects the sensitivity of the model to the noise-level change.
- **Propagation error.** The accumulated error e_{c-1} from the previous $c-1$ chunks affects the generation of chunk c via the KV-cache. Under a *Lipschitz-continuity assumption*, this effect is bounded: the denoising operator f processes an input that contains an upstream error, and so this error is scaled by L to yield $L \cdot e_{c-1}$.

Step 2: Combining the three terms. Summing the three sources, we obtain the single-step upper bound

$$e_c \leq \varepsilon + \kappa \cdot |\sigma_{\text{inf}}(c) - \sigma_{\text{tr}}(c)| + L \cdot e_{c-1}. \tag{13}$$

This follows from the residual and context-perturbation bounds in Lemma 1.

Step 3: Recursive expansion and summation. *Recursive expansion of* e_c. Assuming $e_0 = 0$ and that the upstream scaling is uniform with constant L, and writing $\varepsilon_c := \varepsilon + \kappa \cdot |\sigma_{\text{inf}}(c) - \sigma_{\text{tr}}(c)|$, we have

$$e_C \le \varepsilon_C + L \cdot e_{C-1} \tag{14}$$

$$\le \varepsilon_C + L \cdot \varepsilon_{C-1} + L^2 \cdot e_{C-2} \tag{15}$$

$$\vdots$$

$$e_c \le \sum_{i=1}^{c} L^{c-i} \cdot \varepsilon_i. \tag{16}$$

Summing over all chunks.

$$E_C = \sum_{c=1}^{C} e_c \le \sum_{c=1}^{C} \sum_{i=1}^{c} L^{c-i} \varepsilon_i. \tag{17}$$

Exchanging the order of summation. A fixed term ε_i appears in every chunk $c \ge i$, each time scaled by some power of L, so

$$E_C \le \sum_{i=1}^{C} \left(\sum_{c=i}^{C} L^{c-i} \right) \varepsilon_i. \tag{18}$$

Geometric series.

$$\sum_{c=i}^{C} L^{c-i} \;=\; L^0 + L^1 + \cdots + L^{C-i} \;=\; \frac{1 - L^{C-i+1}}{1 - L}. \tag{19}$$

Final form. Substituting the geometric-series identity and renaming i as c, we arrive at

$$E_C \le \sum_{c=1}^{C} \left(\frac{1 - L^{C-c+1}}{1 - L} \right) \left(\varepsilon + \kappa \cdot |\sigma_{\text{inf}}(c) - \sigma_{\text{tr}}(c)| \right). \qquad \square$$

Remark on the Lipschitz constant L. The derivation above implicitly assumes $L \neq 1$ when applying the geometric-series identity. For completeness, the closed form of w_c covers all $L \ge 0$:

$$w_c \;=\; \begin{cases} \dfrac{1 - L^{C-c+1}}{1 - L}, & L \in [0, 1) \cup (1, \infty), \\ C - c + 1, & L = 1 \text{ (by L'Hôpital).} \end{cases} \tag{20}$$

For any *finite* chunk count C, w_c is therefore a finite constant regardless of the value of L, and Corollary 1 holds without modification. The Lipschitz constant L only governs the asymptotic regime $C \to \infty$: w_c converges to $1/(1-L)$ when $L < 1$, grows linearly when $L = 1$, and grows exponentially when $L > 1$. For fixed L and κ, reducing either modeled contribution reduces the bound for any finite C. This identifies base-model fitting and context-noise alignment as complementary optimization directions. The NTD analysis below examines the regression-error floors associated with the first direction.

A.2 Proof of Lemma 2: NTD Regression-Error Floors

Training objectives. The endpoint baseline is trained with MSE, as in Equation (7), while NTD uses an L_1 loss against the next saved teacher ODE node, as in Equation (8). Teacher trajectories are collected before training, and their saved next nodes directly supervise NTD. The following analysis adopts idealized local-affine and conditional-Gaussian assumptions to motivate this design.

Target model and assumptions. Fix a chunk c and node i. Let X contain the information available to the causal student, including the current noisy chunk x_{t_i} and its visible context. Let $Y = x_0^\star \in \mathbb{R}^D$ and $Z = x_{t_{i+1}}^\star \in \mathbb{R}^D$ be the endpoint and next-node targets. We assume

$$Z = \lambda_i Y + b_i(X), \qquad \lambda_i = 1 - \frac{t_{i+1}}{t_i} \in [0, 1], \qquad b_i(X) = (1 - \lambda_i) x_{t_i}, \tag{21}$$

where $t_i > 0$ and $0 \le t_{i+1} \le t_i$. This is a locally straight trajectory idealization. For each coordinate q, assume a Gaussian conditional

$$Y_q \mid X, c, i \sim \mathcal{N}\big(a_{c,i,q}(X), s^2_{c,i,q}(X)\big), \qquad \mathbb{E}[s^2_{c,i,q}(X)] < \infty. \tag{22}$$

The coordinate-wise analysis allows correlated coordinates and characterizes target uncertainty given the causal student's information.

Coordinate-wise regression-error floors. For a fixed X, c, i, q, define the optimal endpoint conditional RMSE and next-node conditional MAE over scalar predictions u and v:

$$r_{\mathrm{init},c,i,q}(X) := \inf_{u\in\mathbb{R}} \sqrt{\mathbb{E}[(u - Y_q)^2 \mid X, c, i]} = s_{c,i,q}(X), \tag{23}$$

$$r_{\mathrm{NTD},c,i,q}(X) := \inf_{v\in\mathbb{R}} \mathbb{E}[|v - Z_q| \mid X, c, i]. \tag{24}$$

The conditional mean minimizes the endpoint MSE, and its square root defines the RMSE measure above; the training objective itself remains MSE. We average these scalar floors over inputs, chunks, and coordinates:

$$\varepsilon^{\mathrm{lower}}_{\mathrm{init},i} := \mathbb{E}_{c,X}\left[\frac{1}{D}\sum_{q=1}^{D} r_{\mathrm{init},c,i,q}(X)\right], \tag{25}$$

$$\varepsilon^{\mathrm{lower}}_{\mathrm{NTD},i} := \mathbb{E}_{c,X}\left[\frac{1}{D}\sum_{q=1}^{D} r_{\mathrm{NTD},c,i,q}(X)\right]. \tag{26}$$

Thus the endpoint floor averages coordinate-wise optimal conditional RMSEs, while the next-node floor averages coordinate-wise optimal MAEs.

Fixed-node comparison. By Equations (21) and (22), $Z_q \mid X, c, i$ has standard deviation $\lambda_i s_{c,i,q}(X)$. Its median and mean coincide, and the absolute deviation of a scalar Gaussian about its mean is its standard deviation times $\sqrt{2/\pi}$. Therefore

$$r_{\mathrm{NTD},c,i,q}(X) = \lambda_i s_{c,i,q}(X)\sqrt{\frac{2}{\pi}}. \tag{27}$$

Averaging at the fixed node gives

$$\varepsilon^{\mathrm{lower}}_{\mathrm{NTD},i} = \lambda_i \sqrt{\frac{2}{\pi}}\, \varepsilon^{\mathrm{lower}}_{\mathrm{init},i}. \tag{28}$$

Since $\lambda_i \le 1$ and $\sqrt{2/\pi} < 1$, this proves Equation (9) whenever $\varepsilon^{\mathrm{lower}}_{\mathrm{init},i} > 0$. If that floor is zero, both sides are zero.

Averaging over nodes. Using the same node-sampling distribution for both measures, define $\varepsilon^{\mathrm{lower}}_{\mathrm{init}} := \mathbb{E}_i[\varepsilon^{\mathrm{lower}}_{\mathrm{init},i}]$ and $\varepsilon^{\mathrm{lower}}_{\mathrm{NTD}} := \mathbb{E}_i[\varepsilon^{\mathrm{lower}}_{\mathrm{NTD},i}]$. Keeping the node-dependent λ_i inside the expectation,

$$\begin{aligned}\varepsilon^{\mathrm{lower}}_{\mathrm{NTD}} &= \sqrt{\frac{2}{\pi}}\, \mathbb{E}_i[\lambda_i \varepsilon^{\mathrm{lower}}_{\mathrm{init},i}] \\ &\le \sqrt{\frac{2}{\pi}}\, \varepsilon^{\mathrm{lower}}_{\mathrm{init}} < \varepsilon^{\mathrm{lower}}_{\mathrm{init}},\end{aligned} \tag{29}$$

where the last inequality requires a positive average endpoint floor. □

Interpretation. The factor λ_i captures the contraction of conditional target variability, while $\sqrt{2/\pi}$ relates optimal MAE to conditional RMSE under the Gaussian model. Together, they explain the regression-error comparison in Lemma 2 and motivate next-node trajectory supervision.

A.3 Proof of Lemma 3: Expected Noise-Mismatch Reduction

By Equation (6), write

$$N = \kappa \sum_{c=1}^{C} w_c \Delta_c, \qquad \Delta_c = |\sigma_{\text{inf}}(c) - \sigma_{\text{tr}}(c)|, \qquad w_c = \sum_{j=0}^{C-c} L^j > 0. \tag{30}$$

We compare expectations over the training-time sampling, holding $L \geq 0$, $\kappa > 0$, and the weights fixed across methods.

Idealized additive effective-noise model. The context of chunk c contains only the preceding $m = c - 1$ chunks. The scalar σ is an additive proxy for effective context noise. Assume:

1. A rollout samples C exit timesteps $T_1, \ldots, T_C$ from the same marginal distribution. Let $U_j = \delta(T_j)$, where δ is non-decreasing, $U_j \geq d > 0$ almost surely, and $\mathbb{E}[U_j] = \mu < \infty$. The samples may be dependent. For the uniform four-step setting, μ is the mean of the four increment values.
2. In the SF baseline considered here, training retains the sampled noisy cache without a clean refresh, whereas inference refreshes the cache. The model assigns $\sigma_{\text{tr}}^{\text{SF}}(c) = \sum_{j=1}^{m} U_j$ and $\sigma_{\text{inf}}^{\text{SF}}(c) = 0$. Zero denotes the refreshed-cache noise reference.
3. CGN orders the sampled increments as $U_{(1)} \leq \cdots \leq U_{(C)}$ and retains the final-denoising cache at inference. We model $\sigma_{\text{tr}}^{\text{Ours}}(c) = \sum_{j=1}^{m} U_{(j)}$ and $\sigma_{\text{inf}}^{\text{Ours}}(c) = md$, where d is the fixed retained increment and does not exceed any sampled increment.

Both methods therefore have $\Delta_1 = 0$, consistent with Clean Start.

Baseline mismatch. Nonnegativity and linearity of expectation give

$$\mathbb{E}[\Delta_c^{\text{SF}}] = \mathbb{E}\left[\sum_{j=1}^{m} U_j\right] = m\mu. \tag{31}$$

Sorted-prefix bound. For every realization and $1 \leq m \leq C$, the mean of the m smallest increments is no larger than the mean of all C:

$$\sum_{j=1}^{m} U_{(j)} \leq \frac{m}{C} \sum_{j=1}^{C} U_j. \tag{32}$$

Consequently,

$$\mathbb{E}\left[\sum_{j=1}^{m} U_{(j)}\right] \leq m\mu. \tag{33}$$

Thus sorting bounds the expected cumulative contribution of the early chunks.

Our mismatch and cumulative comparison. Since $U_{(j)} \geq d$, for $m = c - 1$,

$$\begin{aligned} \mathbb{E}[\Delta_c^{\text{Ours}}] &= \mathbb{E}\left[\left|\sum_{j=1}^{m} U_{(j)} - md\right|\right] \\ &= \mathbb{E}\left[\sum_{j=1}^{m} U_{(j)}\right] - md \leq m(\mu - d). \end{aligned} \tag{34}$$

The same formula holds for $m = 0$ with empty sums. Thus

$$\mathbb{E}[\Delta_c^{\text{SF}}] - \mathbb{E}[\Delta_c^{\text{Ours}}] \geq (c-1)d. \tag{35}$$

For $C \geq 2$, the common positive weights imply

$$\mathbb{E}[N_{\text{SF}}] - \mathbb{E}[N_{\text{Ours}}] \geq \kappa d \sum_{c=2}^{C} w_c (c-1) > 0, \tag{36}$$

which proves Equation (10). □

Boundary cases. For $C = 1$, both mismatches are zero. If the retained increment is allowed to be $d = 0$, the same argument yields a non-strict inequality.

A.4 Detailed Experimental Settings

Generation and evaluation protocol. Unless otherwise specified, all main-paper comparisons use a 5-second generation horizon at 832×480 resolution. Following CausVid [40], we use the first 128 prompts from MovieGen and evaluate the generated videos with VBench. The reported quality score aggregates the visual-quality dimensions used in the main comparison, including subject consistency, background consistency, dynamic degree, image quality, motion smoothness, aesthetic quality, and temporal flickering. The long-horizon experiment extends the same autoregressive generation setting to 60 seconds; its detailed protocol and results are reported in Section B.

Baselines. We compare TriTD with two bidirectional video diffusion models, Wan2.1-1.3B [5] and LTX-Video-1.9B [24]; two autoregressive video diffusion models, SkyReels-V2-DF-540P [38] and MAGI-1 (base) [39]; and three distilled autoregressive baselines, CausVid [40], Self-Forcing [16], and Causal-Forcing [15]. This grouping separates the quality–speed comparison with bidirectional generators from the direct comparison with chunk-wise autoregressive methods.

Inference and efficiency measurement. Our student performs four denoising steps per chunk, with three frames in each chunk. Throughput is measured on a single NVIDIA H100 GPU and is reported both with and without `torch.compile`. The latency breakdown uses end-to-end wall-clock time and separately reports the diffusion-backbone and VAE-decoding costs; all speedups use the uncompiled Self-Forcing configuration as the common reference unless explicitly stated otherwise.

A.5 Human Evaluation Protocol and Additional Results

Protocol. We randomly sample 100 prompts from VBench, covering categories such as consistency, scene, action, and color, and invite five evaluators to participate. For every prompt, each evaluator views three pairs of 5-second videos generated from the same prompt, comparing TriTD separately against CausVid, Self-Forcing, and Causal-Forcing. Method names are hidden, and the left–right positions are independently randomized. Each TriTD–baseline comparison therefore contains $100 \times 5 = 500$ pairwise votes. Evaluators select the overall better video by jointly considering visual quality and prompt alignment; the resulting pairwise preferences are reported in Table 2. Participation was voluntary, and no compensation was provided.

Absolute ratings. Evaluators also rate each generated video from 0 to 10 along four dimensions: aesthetics, color harmony, motion/action rationality, and structural detail. The overall score is the equally weighted mean of these four dimensions. As shown in Table 5, TriTD obtains the highest overall score of 8.93, compared with 7.40 for CausVid, 8.73 for Self-Forcing, and 8.15 for Causal-Forcing. It achieves the highest or tied-highest score on every evaluated dimension.

Table 5: Absolute human ratings on a 0–10 scale. The overall score is the equally weighted mean of the four dimensions. Best scores are in **bold**.

Method	Aesthetics	Color	Motion	Detail	Overall
CausVid	7.3	7.6	6.8	7.9	7.40
Self-Forcing	8.6	**9.2**	8.4	8.7	8.73
Causal-Forcing	7.9	8.1	8.1	8.5	8.15
TriTD	**8.8**	**9.2**	**8.5**	**9.2**	**8.93**

A.6 Implementation Details

We build on the Wan2.1-T2V 1.3B and 14B models as our base backbone and real score teacher model. Both are originally bidirectional video diffusion models. Our training pipeline consists of three stages.

Stage 1: Teacher ODE trajectory collection. To construct the supervision signal for NTD trajectory distillation, we adopt the same ODE trajectory preparation protocol as the official CausVid repository. Concretely, we use $6{,}000$ prompts from the MixKit dataset and, for each prompt, run the bidirectional teacher model for 48 denoising steps with a time-shift of 5.0 to obtain a full denoising trajectory. For every prompt, we cache the complete 49-step trajectory (including x_0), including the predicted clean endpoint, so that any intermediate teacher node can serve as a supervision target. For the 4-step student, we select $\{1000.0,\ 923.913,\ 750.0,\ 500.0\}$ out of the 48 teacher time steps as the student inference timesteps.

Stage 2: ODE regression initialization. We train the student with the proposed NTD L1 trajectory regression loss under the chunk-graded noising schedule (Clean-Start + Sorted-Noise). This stage runs for $2{,}700$ iterations with a total batch size of 64.

Stage 3: OP-DMD distillation. We then switch to the OP-DMD objective and perform self-rollout training on the student. For each rollout, the exit-flag indices of all chunks are first sampled uniformly at random and then sorted in descending order, so that the KV-cache noise level across chunks is monotonically non-decreasing. This stage runs for 700 iterations with a total batch size of 64, taking approximately two days on the $8\times$ A100 node. We adopt a sliding attention window of 12 frames at this stage.

Hyperparameters. The two training stages use different optimizer settings, summarized in Table 6. Other hyperparameters are shared across stages: teacher CFG guidance scale 3.0, total batch size 64, 3 frames per chunk.

Table 6: Optimizer configuration of the two training stages. Stage 3 maintains separate optimizers for the generator and the critic.

Stage	Learning rate	β_1	β_2
Stage 2: ODE regression	5×10^{-6}	0.9	0.999
Stage 3: OP-DMD (generator)	2×10^{-6}	0.0	0.999
Stage 3: OP-DMD (critic)	4×10^{-7}	0.0	0.999

B Long-Horizon Generalization

Table 7 reports the full 7-dimension VBench metrics for the three chunk-wise distillation methods compared in Section 4.3, evaluated on the first 128 prompts from MovieGen. Consistent with the two failure modes described in the main text, Self-Forcing's Dynamic Degree collapses to 33.26 (from 63.28 at 5s), while its remaining static frames retain a high Motion Smoothness of 98.11, which is a signature of the near-static collapse mode. Causal-Forcing instead suffers from simultaneous degradation in Motion Smoothness (95.97) and Image Quality (64.26), with its Quality Score dropping to 77.01. TriTD retains a Dynamic Degree of 85.23 and achieves the highest Quality Score of 82.64, surpassing the two baselines by $+2.29$ and $+5.63$, respectively.

Table 7: **Comparison on 60-second generation**. Best score per column is **bold**.

Method	Subject Cons.	Background Cons.	Dynamic Degree	Image Quality	Motion Smooth.	Aesthetic Quality	Temporal Flicker.	**Quality Score**
Self-Forcing	**96.71**	**96.01**	33.26	**68.74**	**98.11**	**56.95**	98.10	80.35
Causal-Forcing	94.33	95.41	46.30	64.26	95.97	49.27	96.18	77.01
Ours	94.57	94.89	**85.23**	65.44	97.66	54.69	**98.10**	**82.64**

C Additional Qualitative Results

In this section, we provide additional qualitative comparisons between our proposed method and three representative distilled baselines: CausVid, Self Forcing, Causal Forcing. To comprehensively

evaluate the generation capabilities, we select a diverse set of text prompts covering realistic human portraits, animals with complex accessories, and stylized 3D cartoon characters.

As shown in the following visual comparisons (Figure 5 to Figure 8), existing methods often struggle with maintaining strict identity consistency and detail preservation across different frames or viewpoints. For instance, baseline methods tend to lose track of fine-grained attributes such as the shape of the camera harness (Figure 7) or the structure of the life vest (Figure 6). In contrast, our approach demonstrates superior robust subject consistency, temporal coherence, and visual quality, successfully preserving the intricate details of the subjects regardless of stylistic variations or complex background dynamics.

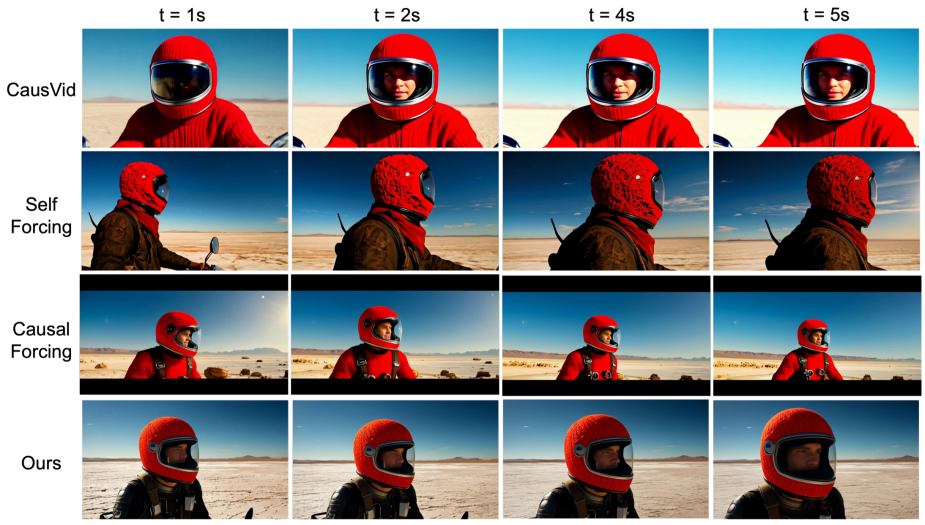

Figure 5: **Qualitative comparison on realistic human subjects.** The rows from top to bottom correspond to CausVid, Self Forcing, Causal Forcing and Ours. Our method maintains the structural integrity of the red helmet and the aesthetic quality significantly better than the baselines.

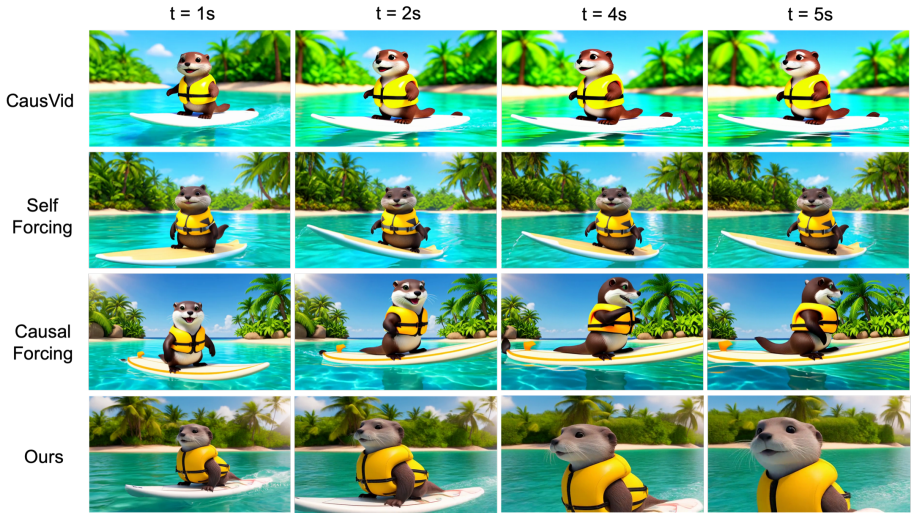

Figure 6: **Qualitative comparison on animals with accessories.** The baselines exhibit notable error accumulation, and identity drift. Our method ensures high-fidelity detail and temporal consistency.

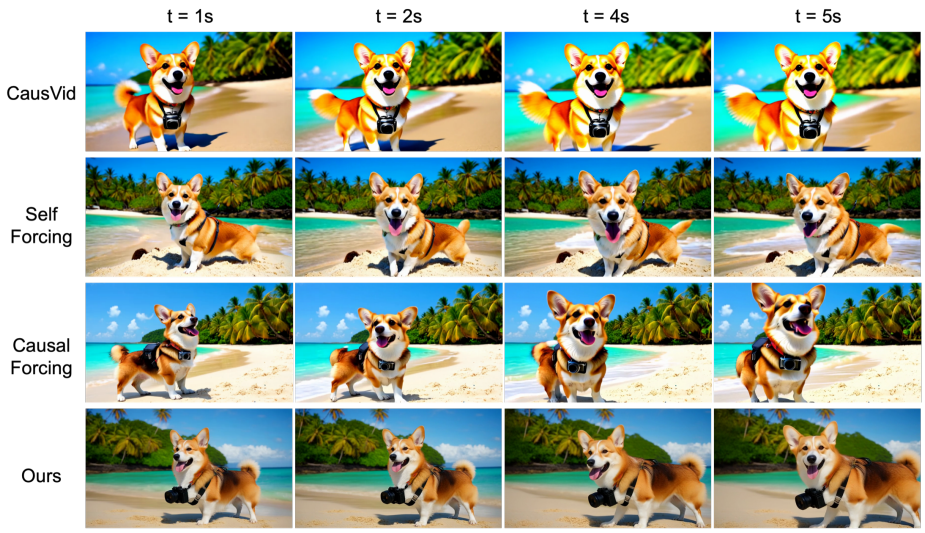

Figure 7: **Qualitative comparison on realistic animal portrait.** CausVid and Causal Forcing tend to suffer from the color oversaturation, While Self Forcing fails to generate the concept of *camera*. Our method shows better visual quality and prompt alignment.



Figure 8: **Qualitative comparison on stylized 3D cartoon generation.** Our approach robustly generalizes to non-realistic/stylized domains. Baselines like CausVid and Causal Forcing show severe oversaturation and identity drift. By contrast, our method maintains consistent appearance and stable background details compared to the degraded outputs of existing methods.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Section 1 summarize NTD, CGN, and OP-DMD; their analysis and evaluation appear in Sections 3–4 and Appendix A.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations paragraph in Section 5 discusses training-data scale, the single backbone, and transfer to other tasks.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Section 3 states the analytical results and their assumptions; Appendix A.1–A.3 gives the corresponding derivations.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 4 and Appendix A.4–A.6 report the backbone, prompt sets, evaluation protocols, training stages, and optimization settings.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: The base model and prompt sources are public, and implementation details are documented in Appendix A.4–A.6, but the TriTD code and trained weights are not yet publicly available.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: The evaluation setting is in Section 4.1 and Appendix A.4; training stages and optimizer parameters are in Appendix A.6.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: We report VBench point estimates and counts from 500 pairwise human votes per comparison, but do not report run-to-run error bars or confidence intervals.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix A.4 and A.6 specify the H100 inference device and the eight-A100 training node, including the Stage 3 training duration.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This research conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The paper focuses on technical quality and efficiency and does not include a dedicated discussion of positive and negative societal impacts.

    Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: This submission does not release a new model checkpoint or dataset.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The original creators of the models, datasets, and code used in this work are appropriately credited, and these assets are used in accordance with their licenses and terms of use.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The submission introduces a training method but does not release a new dataset, codebase, or model checkpoint.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: Appendix A.5 describes the evaluation instructions, blinded comparisons with randomized presentation, and voluntary participation without compensation.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Formal IRB or equivalent review was not applicable to this voluntary video-quality evaluation.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not a component of the proposed method; writing and formatting assistance does not require declaration under this question.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.