TTT2REEL: END-TO-END TEST-TIME TRAINING FOR LONG VIDEO GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Generating a long, coherent video requires preserving visual information about previously-generated frames in the window beyond a model's bounded context window. We present TTT2Reel, an end-to-end test-time training framework for chunk-wise autoregressive long video generation that stores evolving visual history in lightweight LoRA parameters. Before generating each new chunk of a video, the model updates these parameters by optimizing a flow-matching loss on the preceding chunk, while keeping the pretrained backbone frozen. We train the model through bilevel optimization, where an inner optimization loop updates the parametric memory through test-time training, and the outer loop optimizes its initial parameters to minimize subsequent prediction losses across the video. To withstand compounding errors in self-generated context at inference, we estimate model-dependent residuals online and replay them into both the memorization and prediction steps during training. Experiments on minute-long video generation show that TTT2Reel achieves the best average rank across six VBench-Long dimensions among the evaluated methods, balancing dynamic content with visual quality and consistency. Video samples are available on our demo page: `https://anon-submission-01.github.io/ttt2reel-demo/`.

1 INTRODUCTION

Video diffusion transformers now produce strikingly realistic short clips (Peebles & Xie, 2023; Yang et al., 2025c; Wan et al., 2025). However, extending video generation to minutes-long videos requires preserving visual information far beyond the bounded context windows of these models. For example, a three-minute video clip at 720p resolution and 24 FPS already exceeds a million tokens, making full-sequence attention computationally prohibitive. As a result, most video diffusion models generate only a few seconds at a time (Dalal et al., 2025). Longer videos are generated autoregressively in chunks, each comprising a short sequence of consecutive frames that the model jointly denoises using a compact representation of prior frame chunks as conditioning.

Long-video generation requires addressing two fundamental challenges (Yuan et al., 2026; Wang et al., 2026). First, as generation proceeds, the video history grows without bound, but the context window capacity that can serve as a conditioning representation typically remains fixed in size (known as the *capacity problem*). As a result, earlier relevant content in the video is progressively lost as more video chunks are generated. Methods that store history in input space (re-injected frames or their compressed latent representations; Zhang & Agrawala, 2025; Song et al., 2025; Chen et al., 2025) or feature space (temporal KV caches, cached appearance features; Henschel et al., 2025; Huang et al., 2025a; Yang et al., 2025a) attempt to address this capacity problem, but only retain lossy compressions of the earlier video history.

The second problem is *exposure bias*. During training, models learn to predict video chunks from clean, ground-truth context. At inference time, however, the model conditions on its own outputs when generating the next chunk. Over the course of generating a long video, small generation errors from the model propagate across chunks and compound into visible deterioration, such as drifting subject appearance, background changes, and discontinuous motion. Such errors in generated chunks can therefore enter the memory update and then influence subsequent predictions, creating a mismatch that compounds over time, decreasing the quality of long-horizon generation.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

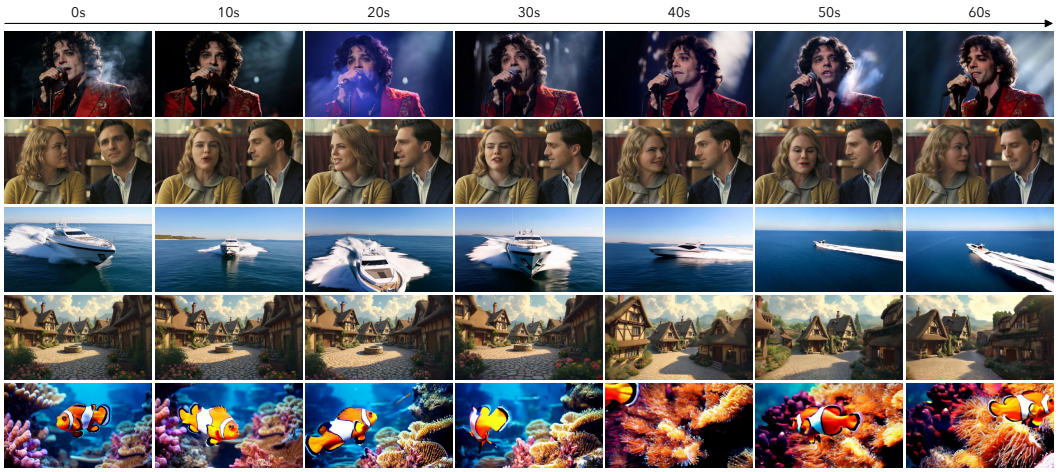


Figure 1: **Qualitative results of TTT2Reel.** Each row shows a one-minute image-to-video rollout, with frames sampled at 10-second intervals. TTT2Reel preserves recognizable subjects and scene structure with minimal drifting and high consistency across different scenarios. More one-minute video rollouts are available on our demo page.

In this work, we present **TTT2Reel**, an end-to-end test-time training video-generation framework that jointly addresses both challenges. TTT2Reel encodes each completed chunk into lightweight LoRA parameters (Hu et al., 2021) through gradient updates with respect to a flow-matching objective (Lipman et al., 2022) before generating the next chunk. The adapted parameters carry visual information across chunks, complementing the bounded input context while keeping the backbone frozen. We train the model through bilevel optimization, where the inner loop updates the memory (i.e., the LoRA parameters), and the outer loop meta-optimizes the initialization of the LoRA parameters to minimize subsequent prediction losses across the video. Motivated by Li et al. (2025), we incorporate an error replay mechanism that estimates model-dependent residuals and replays them in both stages of training. The memorization step adapts to imperfect past chunks, while the prediction step learns to generate from the resulting perturbed context. By exposing both memory updates and the generator to errors resembling those self-generated by the model during autoregressive generation, error replay makes long-horizon generation more robust to its own mistakes.

Our results show TTT2Reel outperforms other leading video generation methods (Yuan et al., 2026; Zhang & Agrawala, 2025; Chen et al., 2026a) on long video generation, ranking first on average across VBench-Long metrics (Huang et al., 2024), and achieving the highest VisionReward (Xu et al., 2024) pass rate under Gemini-3.7-Flash evaluation. Qualitatively, TTT2Reel videos preserve long-term consistency and motion coherence (Figure 1), sustaining generation quality over long horizons, even when extrapolating to unseen, longer video horizons at inference. Finally, TTT2Reel's strong performance does not produce additional inference overhead compared to the most efficient baselines. Overall, TTT2Reel reveals a practical path for long video generation that remains coherent, dynamic, efficient, and visually compelling throughout.

2 METHOD

TTT2Reel performs autoregressive long-video generation by alternating between encoding video history into a parametric memory and predicting the next chunk of the video conditioned on this memory. Before generating each new video chunk, the model encodes video tokens from the preceding chunk into an evolving memory by updating a LoRA adapter using test-time training with gradients computed from a flow-matching objective (Figure 2, left). To learn effective test-time training, we use bilevel optimization with an outer loop that meta-learns the LoRA initialization by minimizing the sum of future prediction losses across the entire video (Figure 2, right). Figure 3 compares how TTT2Reel and other long-video generation methods retain and access context history. Only TTT2Reel implicitly encodes information from the entire history in a parametric memory with a fixed size. During training, error replay introduces model-dependent residuals in both the

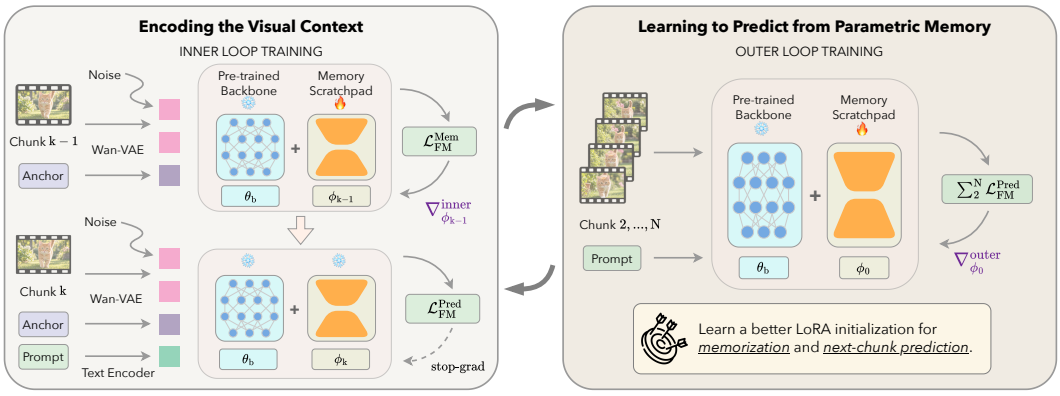


Figure 2: Overview of TTT2Reel. **Left (inner loop).** To generate chunk k, the model first memorizes chunk $\mathrm{k}-1$ by taking a few gradient steps on the flow-matching loss $\mathcal{L}_{\mathrm{FM}}^{\mathrm{Mem}}$. The adapted memory is then frozen in turn, and the model uses it to predict chunk k from noise, conditioned on the narrative and the persistent anchor frame. The resulting $\mathcal{L}_{\mathrm{FM}}^{\mathrm{Pred}}$ is used exclusively for outer-loop training. **Right (outer loop).** The prediction losses are summed across the entire rollout, and backpropagated to update only the initial LoRA parameters ϕ_0, meta-learning an initialization that supports effective memorization and next-chunk prediction throughout the video generation process.

inner-loop memorization and outer-loop prediction to account for imperfect self-generated context at inference time (Figure 4).

2.1 Problem Setup and Preliminaries

We consider long video generation in both text-to-video (T2V) and image-to-video (I2V) settings. Given a textual prompt p and, optionally, an initial image I, the model generates a video consisting of N consecutive chunks $(x_1, \ldots, x_N)$, each containing K frames. In T2V, the first chunk is generated from the prompt alone, while in I2V, it is additionally conditioned on I. Subsequent chunks are generated autoregressively through an iterative procedure that encodes the visual memory and predicts the next chunk of the video. The pretrained text encoder maps p to a conditioning embedding c, while the video encoder maps each chunk to a latent tensor $z_k = \mathcal{E}(x_k) \in \mathbb{R}^{d_z \times t_z \times h_z \times w_z}$. Here, d_z denotes the number of latent channels, and (t_z, h_z, w_z) denote the latent temporal and spatial dimensions. The diffusion transformer parametrizes a time-dependent velocity field $v_t(\cdot; \Theta)$ in the video latent space. We decompose the model's parameters as $\Theta = (\theta_{\mathrm{b}}, \phi)$, where θ_{b} denotes the frozen pretrained backbone parameters and ϕ identifies the trainable LoRA parameters that serve as the model's evolving parametric memory (Hu et al., 2021).

Each chunk after the first is additionally conditioned on a persistent anchor that provides a fixed appearance reference throughout the rollout. During training, the anchor is the first frame of the ground-truth video. At inference, it is the first frame of the generated video in the T2V setting and the supplied image I in the I2V setting. Its latent a enters through the backbone's native image-conditioning path and remains available throughout generation. Related anchor-based conditioning is used in StreamingT2V (Henschel et al., 2025), while in TTT2Reel, the anchor complements the visual history stored in the LoRA parameters.

The Wan backbone is pretrained using the conditional flow-matching objective (Lipman et al., 2022). Given a clean latent z, noise $\epsilon \sim \mathcal{N}(0, I)$, and $t \sim \mathcal{U}[0, 1]$, we define the interpolant $z_t = tz + (1-t)\epsilon$ and its target velocity $u_t(z_t \mid z) = z - \epsilon$. The flow-matching loss for a chunk is

$$\mathcal{L}(z; \phi, c, a) = \mathbb{E}_{t,\epsilon}\left[\left\| v_t(z_t; \Theta, c, a) - u_t(z_t \mid z) \right\|_2^2\right], \tag{1}$$

where $\Theta = (\theta_{\mathrm{b}}, \phi)$. For the first chunk in T2V generation, we omit a from the conditioning since no anchor is available. Updating ϕ therefore provides a parameter-efficient mechanism for writing information into the network without modifying the backbone.

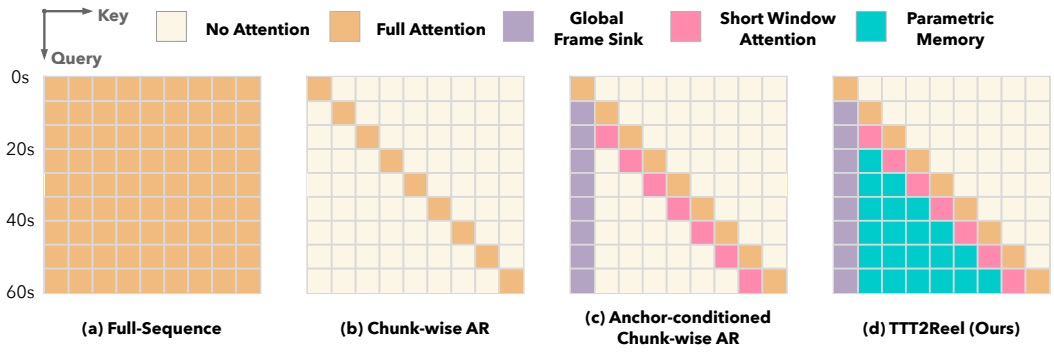


Figure 3: **Schematic comparison of temporal context and memory coverage in long-video generation.** All chunks in (a) are generated jointly with bidirectional attention over a context that grows with video length, making it computationally prohibitive for long videos. (b) generates each chunk sequentially, with attention confined to the current chunk and the final frame of the preceding chunk (omitted from the figure). (c) retains video history as frames, or as cached features such as a KV cache, optionally supplemented by persistent anchor context. In TTT2Reel (d), the persistent anchor, short local window, and LoRA-encoded video history (blue) cover the causal lower triangle.

2.2 End-to-End Training as Bilevel Optimization

Inner-loop memorization. The model memorizes each completed chunk before predicting the next one (Figure 2, left). Let ϕ_0 denote the learned initialization and set $\phi_1 = \phi_0$, since no memory encoding precedes the first chunk. For clarity, we express a single memorization step as a gradient-descent update:

$$U(\phi; z, c, a) \triangleq \phi - \eta \nabla_\phi \mathcal{L}(z; \phi, c, a), \qquad \phi_k = U^M(\phi_{k-1}; z_{k-1}, c, a), \tag{2}$$

where η denotes the inner-loop learning rate and U^M denotes M successive updates on the preceding chunk. After memorization, the adapted model $(\theta_\mathrm{b}, \phi_k)$ predicts chunk k conditioned on the prompt and anchor, with prediction loss $\ell_k = \mathcal{L}(z_k; \phi_k, c, a)$. The memory is updated sequentially across chunks, and its adapted parameters are held fixed during each prediction step. The same memorization procedure is used during training and inference, with error replay during training accounting for imperfectly generated history, as described below.

Outer-loop training. The outer loop learns an initialization that supports visual memory encoding and next-chunk prediction throughout the rollout (Figure 2, right). We optimize the initial LoRA parameters $\phi = \phi_0$ while keeping the pretrained backbone parameters θ_b fixed. For a training video, the objective is

$$\min_{\phi_0} \mathcal{J}(\phi_0) = \sum_{k=2}^{N} \mathcal{L}(z_k; \phi_k, c, a), \qquad \phi_k = U^M(\phi_{k-1}; z_{k-1}, c, a). \tag{3}$$

Each adapted state ϕ_k depends on ϕ_0 through the preceding inner-loop updates. This yields a bilevel meta-learning problem in which the inner loop adapts the memory to the current video and the outer loop learns an initialization that minimizes prediction error across the complete rollout. Differentiating this objective exactly through the inner-loop updates requires second-order derivatives and the retention of intermediate states throughout the rollout. To reduce these computational and memory costs, we use first-order model-agnostic meta-learning (FOMAML; Finn et al., 2017), which approximates the adaptation Jacobian $\partial\phi_k/\partial\phi_0$ by the identity matrix. The prediction gradients evaluated at the adapted states ϕ_k are thus accumulated to update the initialization ϕ_0. Appendix B.1 provides the derivations of the full MAML and its first-order meta-gradient approximation.

Error replay in bilevel training. The formulation above uses clean ground-truth chunks during training, whereas inference requires the model to memorize and predict from its own imperfect outputs. To expose both stages to this mismatch during training, we design an error-replay strategy motivated by Li et al. (2025). Given an interpolant z_t, the model predicts $\hat{v}_t = v_t(z_t; \Theta, c, a)$. Using $\hat{v}_t$ as a constant velocity approximation along each integration interval, forward integration gives the clean-latent estimate $\hat{z} = z_t + \int_t^1 v_s(z_s; \Theta, c, a)\,\mathrm{d}s \approx z_t + (1-t)\hat{v}_t$, while backward integration

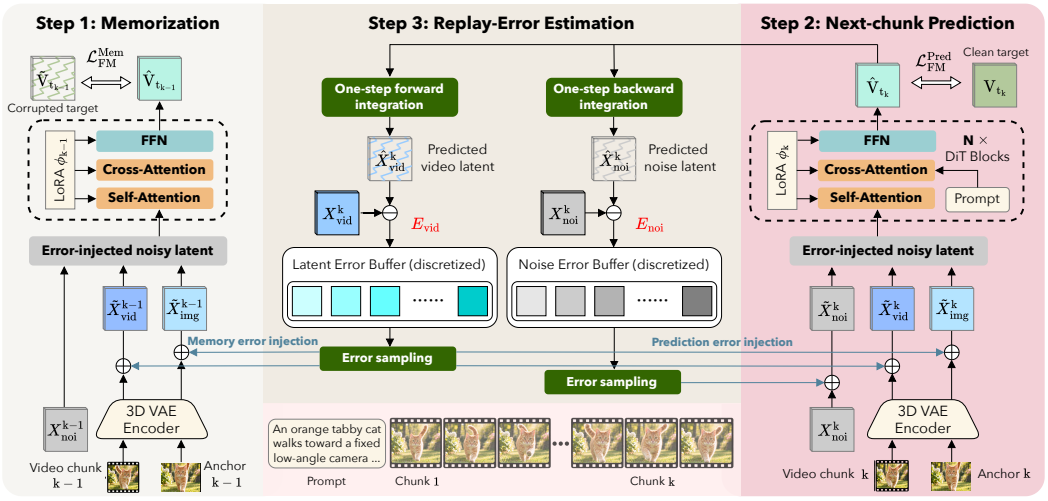


Figure 4: Error-replay training in TTT2Reel. **Left (memorizing chunk** k−1**).** Sampled model-dependent residuals perturb the video and anchor latents used for memorization, and the memorization loss uses a perturbed target. These perturbations simulate learning from imperfect generated context. **Right (predicting chunk** k**).** Sampled residuals perturb the video latent, anchor latent, and noise used for next-chunk prediction, while the prediction loss retains the clean target. **Middle (replay residual estimation).** Single Euler steps toward the clean-data and noise endpoints produce estimates $\hat{X}^{\mathrm{k}}_{\mathrm{vid}}$ and $\hat{X}^{\mathrm{k}}_{\mathrm{noi}}$. Their differences from the corresponding clean video latent and sampled noise yield model-dependent residuals E_{vid} and E_{noi}, which are stored online for subsequent replay.

gives the noise estimate $\hat{\epsilon} = z_t - \int_0^t v_s(z_s; \Theta, c, a)\,\mathrm{d}s \approx z_t - t\hat{v}_t$. The differences $\hat{z} - z$ and $\hat{\epsilon} - \epsilon$ provide model-dependent residual estimates, which are stored for subsequent replay (Figure 4, middle). During inner-loop memorization, sampled video-latent residuals perturb the chunk used to update the LoRA memory (Figure 4, left). During outer-loop prediction, residuals perturb the video latent, anchor latent, and noise (Figure 4, right), exposing the outer objective to both memory states adapted from perturbed history and prediction inputs affected by model-generated residuals.

2.3 Inference

At inference, starting from the learned LoRA initialization ϕ_0, TTT2Reel generates the first video chunk using ϕ_0 and fixes the latent of its first frame as the anchor a. The anchor remains fixed throughout the rollout. For each subsequent chunk k, the model encodes the previous output as $z_{k-1} = \mathcal{E}(x_{k-1})$ and performs M memorization updates according to equation 2, obtaining ϕ_k. The adapted model $(\theta_{\mathrm{b}}, \phi_k)$ then generates x_k conditioned on c and a, with ϕ_k held fixed throughout sampling. Whereas training evaluates the next-chunk prediction loss at a sampled timestep, inference generates each chunk by numerically integrating the velocity field v_t over $S = 50$ sampling steps. Figure 3 compares the history context accessible to different rollout strategies.

3 Experiments

3.1 Experimental Setup

Models and Data. We instantiate TTT2Reel using Wan2.2-TI2V-5B and Wan2.1-I2V-14B-480P (Wan et al., 2025) as backbone models, using their pretrained UMT5-XXL text encoders and respective VAEs to encode prompts and video chunks. We train on UltraVideo-Long (Xue et al., 2025), which contains 17K videos paired with captions (Appendix C). For both backbones, we keep the pretrained weights frozen and optimize the initial LoRA parameters using FOMAML as described in Section 2.2. Implementation details are provided in Appendix D.

Baselines. We compare TTT2Reel with two training-free, chunk-wise rollout strategies using the corresponding pretrained backbone. The first conditions each chunk on the preceding chunk's final

Table 1: **Evaluation results on VBench-Long for 60-second text-to-video generation.** Scores are reported in percentage points. Avg. Rank is the mean over the six metrics of each method's rank (1 = best) among all listed methods (lower is better). The best score per column is in bold, and the second-best is underlined. TTT2Reel achieves the highest rank among the strong baselines.

Method	# Params	Avg. Rank ↓	Dynamic Degree ↑	Aesthetic Quality ↑	Imaging Quality ↑	Subject Consistency ↑	Background Consistency ↑	Motion Smoothness ↑
Causal-Forcing (Zhu et al., 2026)	1.3B	7.50	<u>72.32</u>	51.24	62.30	93.52	94.12	95.74
Self-Forcing (Huang et al., 2025a)	1.3B	6.67	51.72	56.05	62.22	95.84	95.27	98.20
Rolling-Forcing (Liu et al., 2026)	1.3B	5.00	36.00	**63.50**	**72.42**	94.09	94.77	98.65
LongLive (Yang et al., 2025a)	1.3B	4.67	44.56	58.17	67.56	<u>97.13</u>	95.89	98.61
MAGI-1 (Teng et al., 2025)	4.5B	7.67	56.00	52.10	54.54	79.46	87.76	**99.26**
Wan2.2-TI2V (Wan et al., 2025)	5B	5.00	60.89	55.69	60.08	96.94	96.42	98.65
Wan2.2-TI2V† (Wan et al., 2025)	5B	5.50	66.40	56.00	65.45	96.39	96.15	97.76
LongLive-2.0 (Chen et al., 2026a)	5B	<u>3.67</u>	60.62	53.68	65.51	**97.48**	**97.00**	<u>98.86</u>
SkyReels-V2 (Chen et al., 2025)	14B	6.33	44.00	57.64	66.67	84.99	89.95	98.67
TTT2Reel (Ours)	5B	**2.83**	**73.57**	<u>58.90</u>	<u>67.64</u>	96.88	<u>96.77</u>	98.64

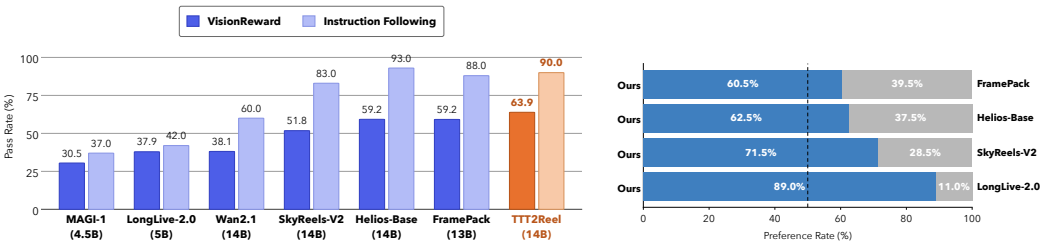


Figure 5: **Preference evaluation results on VisionReward. Left:** VisionReward and instruction-following pass rates for each model. **Right:** Head-to-head preference rates of TTT2Reel against four baselines. All results use Gemini-3.7-Flash as the judge and process videos at 4.0 FPS.

frame. The second, termed anchor-conditioned rollout (Wan2.2-TI2V† in Table 1), conditions each chunk on a persistent anchor frame and the final three latent frames of the preceding chunk. We also compare with SkyReels-V2 (Chen et al., 2025), which conditions generation on historical frames, and KV-cache-based methods including Causal-Forcing (Zhu et al., 2026), Self-Forcing (Huang et al., 2025a), Rolling-Forcing (Liu et al., 2026), MAGI-1 (Teng et al., 2025), LongLive (Yang et al., 2025a), and LongLive-2.0 (Chen et al., 2026a). For the I2V preference evaluation, we additionally include FramePack (Zhang & Agrawala, 2025) and Helios-Base (Yuan et al., 2026), both of which retain history as compressed input-space context. Unlike these baselines, TTT2Reel updates its LoRA parameters during inference to encode the video history.

Evaluation. We evaluate 60-second video generation in both T2V and I2V settings. For T2V, we evaluate the 5B variant using the official VBench-Long prompts (Huang et al., 2024; 2025b) and report six dimensions: Subject Consistency, Background Consistency, Motion Smoothness, Dynamic Degree, Aesthetic Quality, and Imaging Quality. We also report each method's mean rank across these dimensions among the compared methods (Table 1). For I2V, we evaluate the 14B variant on prompts and paired initial images from the SVI benchmark (Li et al., 2025),[1] using identical text and image inputs across methods. We assess these videos using VisionReward's hierarchical video-assessment questions (Xu et al., 2024), with Gemini-3.7-Flash as the judge. Videos are resampled at 4.0 FPS while preserving their original resolutions before being passed to the judge model. We report VisionReward and instruction-following scores, together with head-to-head preference rates against four baselines (Figure 5). Further evaluation details are provided in Appendix E.

3.2 Results

TTT2Reel outperforms strong baselines on long-video generation benchmark. We evaluate 60-second generation on VBench-Long using its official prompts (Table 1). The baselines reveal a recurring trade-off: methods that score highly on visual quality tend to lose temporal consistency, while methods that preserve consistency often produce less dynamic or less visually appealing videos. Our TTT2Reel offers a stronger balance. We achieve the best average rank across all six metrics and

[1] https://huggingface.co/datasets/epfl-vita/svi-benchmark

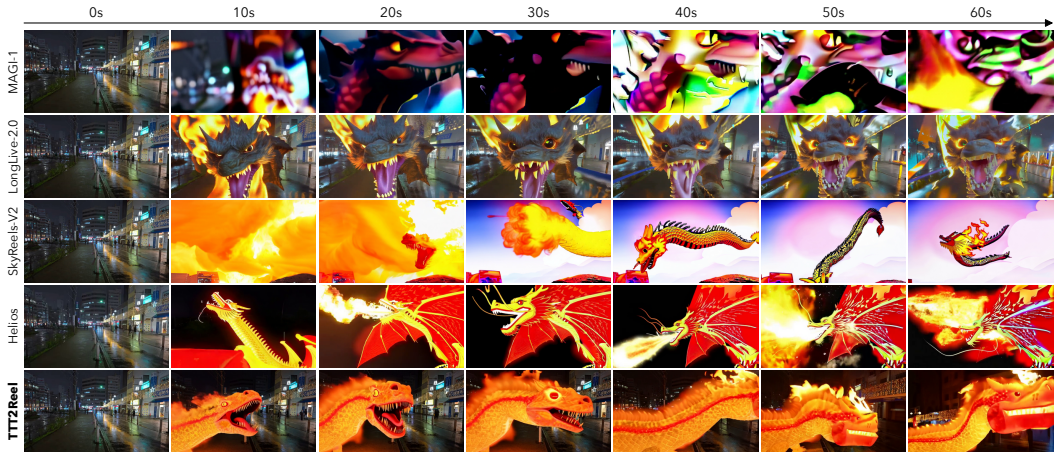


Figure 6: **Qualitative comparison with baseline long-video generation methods.** Frames are sampled at 10-second intervals from one-minute rollouts generated from the same initial frame and dragon prompt. Compared with the baselines, TTT2Reel achieves a better balance between long-term visual consistency and dynamic content evolution. Additional video comparisons with state-of-the-art methods are available in Appendix F and on our demo page.

the highest Dynamic Degree, while placing second on Background Consistency, Aesthetic Quality, and Imaging Quality. Compared with the anchor-conditioned rollout (Wan2.2-TI2V†) on the same 5B backbone, TTT2Reel differs only in the parametric memory and end-to-end bilevel training that teaches the model to store and use visual history. The resulting improvements across all six metrics demonstrate the effectiveness of this learned memory mechanism for long-video generation. Overall, the results show that TTT2Reel can sustain coherent, dynamic, and visually compelling motion over long horizons.

When evaluated by frontier LLMs, TTT2Reel outperforms baselines head-to-head. We evaluate 60-second image-to-video generation using VisionReward's assessment questions, with Gemini-3.7-Flash as the judge (Figure 5). The 14B variant of TTT2Reel achieves the highest VisionReward pass rate while maintaining a 90.0% instruction-following pass rate, second only to Helios-Base. Its advantage also holds in head-to-head comparisons as the judge prefers TTT2Reel over each of the four evaluated baselines, including Helios-Base and FramePack, which both retain history frames as memory and tie for the highest VisionReward score among the baselines. The judge also prefers TTT2Reel over LongLive-2.0 in 89.0% of cases, which retains visual history in a KV cache. Together, the results indicate that TTT2Reel generates long videos that are both compelling to frontier evaluators and faithful to the input instructions.

Qualitatively, TTT2Reel produces dynamic, high-quality, consistent videos across long time horizons. Figure 6 compares one-minute rollouts from the same nighttime street image and dragon prompt. The sampled frames reveal several failure modes among the baselines: MAGI-1 develops severe color and structural distortions, SkyReels-V2 and Helios-Base lose the original street setting, and LongLive-2.0 retains the setting but repeatedly shows a near-frontal dragon. In contrast, TTT2Reel keeps the wet street and illuminated storefronts recognizable as the video develops. The dragon turns its head, shifts between close and wider views, and extends its body across the scene before returning toward the camera. Its fiery appearance remains recognizable through these changes, giving the sequence both a consistent setting and a sense of progression. Additional one-minute examples in Figure 1 show varied expressions, viewpoints, and object positions while retaining recognizable subjects and scenes. Further comparisons appear in Appendix F.

3.3 Analysis

TTT2Reel generalizes to longer videos at test time. Figure 7 shows that TTT2Reel retains its advantage as generation extends beyond the longest training videos (one minute), remaining ahead of Helios-Base through four minutes despite a similar rate of degradation. In contrast, at the same model scale, FramePack declines sharply. Finally, TTT2Reel's strong results at 15 and 30 seconds

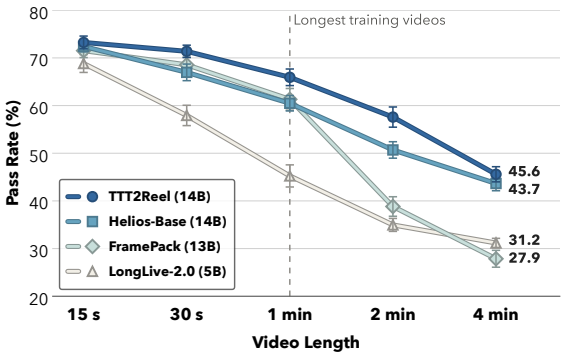


Figure 7: **Long-horizon generalization.** TTT2Reel achieves the highest pass rate on VisionReward across video lengths from 15 seconds to 4 minutes.

Table 2: **Computational cost of retaining visual history.** Measured normalized compute and memory-state size. TTT2Reel has the second-lowest normalized compute and uses less memory than KV caches, but more than retained frames or latents.

Method	Model size	Normalized compute	Memory state
LongLive-2.0	5B	1.97×	7.25 GB
Causal-Forcing	8B	1.64×	10.28 GB
FramePack	13B	1.22×	7.3 MB
Helios-Base	14B	1.16×	4.4 MB
TTT2Reel	14B	1.17×	1.14 GB

Table 3: **Ablation study of TTT2Reel** on Wan2.2-TI2V-5B. We evaluate test-time updates, end-to-end training, and error replay across six VBench-Long dimensions. Checkmarks indicate enabled components. The best and second-best results are shown in **bold** and underlined, respectively.

Test-time Training	End-to-end Training	Error Replay	Dynamic Degree ↑	Aesthetic Quality ↑	Imaging Quality ↑	Subject Consistency ↑	Background Consistency ↑	Motion Smoothness ↑
✓	✓	✓	**73.57**	**58.90**	**67.64**	96.88	96.77	**98.64**
–	✓	✓	72.10	57.90	66.12	**96.90**	**96.92**	98.52
✓	–	✓	70.40	54.50	62.29	95.40	95.54	97.65
–	–	✓	69.80	58.00	66.54	96.30	96.12	98.22
✓	✓	–	72.70	56.80	64.57	96.69	96.66	98.46

suggest that the longer-horizon capability does not come at the expense of short-clip quality. Overall, TTT2Reel improves performance across both short and extended horizons, achieving the strongest long-video generation among the baselines.

Storing context in weights is among the cheapest in compute. Table 2 compares the runtime and memory overhead of retaining visual history across chunks. Normalized compute is measured against each method's vanilla chunk-wise autoregressive generation setting under the same backbone and sampling configuration, with 1.0 indicating no additional overhead. Memory state measures the vRAM of the representation storing video history (GB). Although TTT2Reel's weights update at test time with an additional backward pass, its compute overhead is close to that of Helios-Base and lower than FramePack and KV-cache methods, LongLive-2.0 and Causal-Forcing. Therefore, the test-time training step with a lightweight LoRA is inexpensive relative to the underlying video sampling, avoiding the additional overhead required to attend to cached history and refresh its stored features. The tradeoff is memory. TTT2Reel requires less memory than the measured KV caches, but more than the frame or latent histories used by Helios-Base and FramePack. Importantly, because TTT2Reel incorporates generated history into fixed-size LoRA modules, retaining more history does not require a continually growing stored context or additional attention over that context. The memory and per-chunk update overhead remain independent of the accumulated history length.

Ablation studies. Table 3 shows that the complete TTT2Reel achieves the best motion and visual quality scores while maintaining competitive consistency relative to ablating different components of the method. Removing test-time training marginally increases both consistency scores but reduces motion smoothness, dynamics, and visual quality. Thus, updating the LoRA memory at test time supports sustained visual dynamics. With test-time training enabled, removing end-to-end meta-training degrades all six metrics. Moreover, without end-to-end training, direct test-time training lowers five of the six scores relative to freezing the LoRA parameters, with only Dynamic Degree increasing slightly. Taken together, these results show that learning a meta-initialization that makes the test-time-trained memory useful for subsequent prediction is critical, since fitting the preceding chunk alone does not ensure better next-chunk generation. Removing error replay lowers all six scores, indicating that exposing both memorization and prediction to model-dependent residuals during training improves robustness.

4 Related Work

Long Video Generation Video diffusion models generate high-quality short clips, but quadratic attention costs and bounded temporal context make full-sequence generation over long horizons computationally challenging (Xi et al., 2025; Yang et al., 2025b; Zhou et al., 2026). Existing long-video systems therefore commonly adopt autoregressive rollout, generating frames or chunks conditioned on earlier outputs (Henschel et al., 2025; Song et al., 2025; Chen et al., 2025; Teng et al., 2025). They retain visual history either in input space through re-injected frames or compressed latent representations, or in feature space such as temporal KV caches. Anchor frames, attention sinks, and context-packing schemes preserve access to earlier content while keeping computation bounded (Henschel et al., 2025; Zhang & Agrawala, 2025; Liu et al., 2026; Yang et al., 2025a), but information discarded from these representations remains inaccessible. A complementary challenge is exposure bias, which arises when models trained on clean context encounter their own imperfect outputs at inference. Self-Forcing and Causal-Forcing address this mismatch through rollout-aware distillation (Huang et al., 2025a; Zhu et al., 2026), while Stable Video Infinity replays estimated model-dependent errors during training (Li et al., 2025). These strategies improve robustness to generated context but do not explicitly preserve information beyond the retained input or feature memory. TTT2Reel instead maintains an evolving parametric memory in the LoRA parameters and integrates error replay into bilevel training.

Test-Time Training. Test-time training (TTT) adapts models to unlabeled test inputs through a self-supervised objective, with early work focusing on robustness under distribution shift (Sun et al., 2020; Zhao et al., 2023). Recent work uses TTT for sequence modeling, treating the parameters of a small neural network as a recurrent state that is updated online to encode past tokens (Sun et al., 2024). Meta-learning extends this approach by optimizing how information is encoded and utilized from the evolving state (Chen et al., 2023). PERK (Chen et al., 2026b) encodes long contexts into LoRA adapters through the inner-loop TTT and meta-learns to reason over and apply the updated memory in an outer loop, while TTT-E2E meta-learns a set of initial model parameters for continual next-token updates (Tandon et al., 2025). Related formulations have also been explored for visual sequence modeling (Han et al., 2026). For video generation, Dalal et al. (2025) incorporate TTT layers, whose online-updated MLP weights capture dependencies across video chunks, into a pre-trained video diffusion transformer. LaCT (Zhang et al., 2026) combines sliding-window attention with fast-weight memory updated from large chunks of video tokens, improving update efficiency and enabling larger memory states for autoregressive video generation. Most closely related to our memory representation, SlowFast-VGen updates temporal LoRA modules online to retain episodic history for action-conditioned video prediction, but does not meta-optimize their initialization using future-chunk prediction losses (Hong et al., 2025). TTT2Reel extends this direction to general long-video generation by meta-learning the LoRA initialization through sequential visual memory encoding and next-chunk prediction without dedicated TTT layers.

5 Conclusion

We presented **TTT2Reel**, an end-to-end test-time training framework for long video generation that encodes accumulating visual history in lightweight LoRA parameters. Before generating each new chunk, the model updates this parametric memory using the preceding chunk while keeping the pretrained backbone frozen. Bilevel training with FOMAML learns an initialization of the LoRA parameters that makes these memory updates useful for subsequent prediction, while error replay exposes both memorization and prediction to model-dependent residuals during training. On minute-long generation, TTT2Reel achieves the best average rank across six VBench-Long dimensions among the evaluated methods, balancing dynamic content, visual quality, and consistency. Preference evaluation with Gemini-3.7-Flash further demonstrates strong performance alongside high prompt adherence. Ablation studies underscore the importance of learning the memory initialization end-to-end, showing that test-time updates alone do not reliably improve generation. Together, these results suggest that learned parametric memory is a promising approach to extending video generation beyond a model's bounded context window. Future work will investigate memory retention over longer horizons and more efficient test-time updates.

AI USE STATEMENT

In this work, we used generative AI tools to help with the implementation of our method and to design or provide feedback on research methodology or experiments. We have not used generative AI tools to generate synthetic datasets, develop theoretical models or conceptual frameworks, assist in the writing of proofs, or interpret results as these are not applicable to this work. Additionally, we used generative AI tools to summarize and analyze existing literature, look for relevant work, and edit the paper to improve readability. We have checked LLM-generated research ideas for potential plagiarism through a manual literature survey. LLM-generated code was verified and tested for correctness by the first author of the paper. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

The human figures depicted in this paper are AI-generated and are presented solely for research demonstration and evaluation. These visual examples are intended to illustrate the capabilities and limitations of the proposed method and should not be interpreted as authentic recordings of real individuals or events. We acknowledge the contributions of the original model and dataset providers and the importance of respecting the licenses, usage restrictions, and rights associated with these resources. Public availability alone does not establish unrestricted permission for subsequent use or redistribution. We recognize that video generation technologies may reproduce biases present in training data or be misused to create misleading, impersonating, or otherwise harmful content. Responsible deployment should therefore account for privacy, consent, intellectual property, and transparent disclosure of AI-generated media. The research demonstrations presented in this work do not constitute an endorsement of deceptive or non-consensual uses. Any downstream application should assess the suitability of the models and data for its intended context and implement safeguards proportionate to the risks involved.

REFERENCES

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

Yukang Chen, Luozhou Wang, Wei Huang, Shuai Yang, Bohan Zhang, Yicheng Xiao, Ruihang Chu, Weian Mao, Qixin Hu, Shaoteng Liu, et al. Longlive-2.0: An nvfp4 parallel infrastructure for long video generation. *arXiv preprint arXiv:2605.18739*, 2026a.

Zeming Chen, Gail Weiss, Eric Mitchell, Asli Celikyilmaz, and Antoine Bosselut. Reckoning: Reasoning through dynamic knowledge encoding. *Advances in neural information processing systems*, 2023.

Zeming Chen, Angelika Romanou, Gail Weiss, and Antoine Bosselut. Perk: Long-context reasoning as parameter-efficient test-time learning. In *International Conference on Learning Representations*, volume 2026, pp. 10350–10377, 2026b.

Karan Dalal, Daniel Koceja, Jiarui Xu, Yue Zhao, Shihao Han, Ka Chun Cheung, Jan Kautz, Yejin Choi, Yu Sun, and Xiaolong Wang. One-minute video generation with test-time training. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 17702–17711, 2025.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017.

Dongchen Han, Yining Li, Tianyu Li, Zixuan Cao, Ziming Wang, Jun Song, Yu Cheng, Bo Zheng, and Gao Huang. ViT^3: Unlocking test-time training in vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 51–61, 2026.

Roberto Henschel, Levon Khachatryan, Hayk Poghosyan, Daniil Hayrapetyan, Vahram Tadevosyan, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. Streamingt2v: Consistent, dynamic, and extendable long video generation from text. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025.

Yining Hong, Beide Liu, Maxine Wu, Yuanhao Zhai, Kai-Wei Chang, Linjie Li, Kevin Lin, Chung-Ching Lin, Jianfeng Wang, Zhengyuan Yang, et al. Slowfast-vgen: Slow-fast learning for action-driven long video generation. In *International Conference on Learning Representations*, volume 2025, pp. 76609–76636, 2025.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *Advances in neural information processing systems*, 2025a.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21807–21818, 2024.

Ziqi Huang, Fan Zhang, Xiaojie Xu, Yinan He, Jiashuo Yu, Ziyue Dong, Qianli Ma, Nattapol Chanpaisit, Chenyang Si, Yuming Jiang, et al. Vbench++: Comprehensive and versatile benchmark suite for video generative models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025b.

Wuyang Li, Wentao Pan, Po-Chien Luan, Yang Gao, and Alexandre Alahi. Stable video infinity: Infinite-length video generation with error recycling. *arXiv preprint arXiv:2510.09212*, 2025.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Kunhao Liu, Wenbo Hu, Jiale Xu, Ying Shan, and Shijian Lu. Rolling forcing: Autoregressive long video diffusion in real time. In *International Conference on Learning Representations*, 2026.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 10674–10685. ieee, 2022.

Kiwhan Song, Boyuan Chen, Max Simchowitz, Yilun Du, Russ Tedrake, and Vincent Sitzmann. History-guided video diffusion. In *International conference on machine learning*, 2025.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pp. 9229–9248. PMLR, 2020.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, et al. Learning to (learn at test time): Rnns with expressive hidden states. *arXiv preprint arXiv:2407.04620*, 2024.

Arnuv Tandon, Karan Dalal, Xinhao Li, Daniel Koceja, Marcel Rød, Sam Buchanan, Xiaolong Wang, Jure Leskovec, Sanmi Koyejo, Tatsunori Hashimoto, et al. End-to-end test-time training for long context. *arXiv preprint arXiv:2512.23675*, 2025.

Hansi Teng, Hongyu Jia, Lei Sun, Lingzhi Li, Maolin Li, Mingqiu Tang, Shuai Han, Tianning Zhang, WQ Zhang, Weifeng Luo, et al. Magi-1: Autoregressive video generation at scale. *arXiv preprint arXiv:2505.13211*, 2025.

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

Zile Wang, Zexiang Liu, Jiaxing Li, Kaichen Huang, Baixin Xu, Fei Kang, Mengyin An, Peiyu Wang, Biao Jiang, Yichen Wei, et al. Matrix-game 3.0: Real-time and streaming interactive world model with long-horizon memory. *arXiv preprint arXiv:2604.08995*, 2026.

Haocheng Xi, Shuo Yang, Yilong Zhao, Chenfeng Xu, Muyang Li, Xiuyu Li, Yujun Lin, Han Cai, Jintao Zhang, Dacheng Li, et al. Sparse videogen: Accelerating video diffusion transformers with spatial-temporal sparsity. *arXiv preprint arXiv:2502.01776*, 2025.

Jiazheng Xu, Yu Huang, Jiale Cheng, Yuanming Yang, Jiajun Xu, Yuan Wang, Wenbo Duan, Shen Yang, Qunlin Jin, Shurun Li, et al. Visionreward: Fine-grained multi-dimensional human preference learning for image and video generation. *arXiv preprint arXiv:2412.21059*, 2024.

Zhucun Xue, Jiangning Zhang, Teng Hu, Haoyang He, Yinan Chen, Yuxuan Cai, Yabiao Wang, Chengjie Wang, Yong Liu, Xiangtai Li, et al. Ultravideo: High-quality uhd video dataset with comprehensive captions. *arXiv preprint arXiv:2506.13691*, 2025.

Shuai Yang, Wei Huang, Ruihang Chu, Yicheng Xiao, Yuyang Zhao, Xianbang Wang, Muyang Li, Enze Xie, Yingcong Chen, Yao Lu, Song Han, and Yukang Chen. Longlive: Real-time interactive long video generation. *arXiv preprint arXiv:2509.22622*, 2025a.

Shuo Yang, Haocheng Xi, Yilong Zhao, Muyang Li, Jintao Zhang, Han Cai, Yujun Lin, Xiuyu Li, Chenfeng Xu, Kelly Peng, et al. Sparse videogen2: Accelerate video generation with sparse attention via semantic-aware permutation. *arXiv preprint arXiv:2505.18875*, 2025b.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations*, volume 2025, pp. 83048–83077, 2025c.

Shenghai Yuan, Yuanyang Yin, Zongjian Li, Xinwei Huang, Xiao Yang, and Li Yuan. Helios: Real real-time long video generation model. *arXiv preprint arXiv:2603.04379*, 2026.

Lvmin Zhang and Maneesh Agrawala. Packing input frame context in next-frame prediction models for video generation. *arXiv preprint arXiv:2504.12626*, 2025.

Tianyuan Zhang, Sai Bi, Yicong Hong, Kai Zhang, Fujun Luan, Songlin Yang, Kalyan Sunkavalli, William Freeman, and Hao Tan. Test-time training done right. In *International Conference on Learning Representations*, volume 2026, pp. 157604–157638, 2026.

Hao Zhao, Yuejiang Liu, Alexandre Alahi, and Tao Lin. On pitfalls of test-time adaptation. *arXiv preprint arXiv:2306.03536*, 2023.

Xuanyi Zhou, Qiuyang Mang, Shuo Yang, Haocheng Xi, Jintao Zhang, Huanzhi Mao, Joseph E Gonzalez, Kurt Keutzer, Ion Stoica, and Alvin Cheung. Svg-ear: Parameter-free linear compensation for sparse video generation via error-aware routing. *arXiv preprint arXiv:2603.08982*, 2026.

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.

A Algorithm of TTT2Reel

Algorithm 1 summarizes the training and inference procedures of TTT2Reel.

B More details about TTT2Reel

B.1 Meta-Gradient Estimation

Optimizing the outer objective in equation 3 requires computing its gradient with respect to the initial LoRA parameters Θ. We first describe exact differentiation through the sequential memorization updates, following model-agnostic meta-learning (MAML) (Finn et al., 2017). We then introduce the first-order approximation used to train all TTT2Reel.

Algorithm 1 Training and inference procedures of TTT2Reel.

Require: Training distribution $\mathcal{D}$ over pairs $(x_{1:N}, p)$ of chunked videos and prompts, frozen backbone θ_{b}, inner steps M, inner learning rate η, number of chunks N, and sampling steps S

procedure TTT2REEL-TRAIN($\mathcal{D}, \phi_0$)
  Initialize residual buffers $\mathcal{B} = \{\mathcal{B}_{\mathrm{vid}}, \mathcal{B}_{\mathrm{anc}}, \mathcal{B}_{\mathrm{noi}}\}$
  while not converged **do**
    Sample $(x_{1:N}, p) \sim \mathcal{D}$
    $c \leftarrow \mathrm{UMT5{-}XXL}(p)$, $z_{1:N} \leftarrow \mathcal{E}(x_{1:N})$, $a \leftarrow \mathcal{E}(x_1^{(1)})$
    $\phi_1 \leftarrow \phi_0$
    for $k = 2, \ldots, N$ **do**
      $(\phi_k, \mathcal{B}) \leftarrow$ REPLAYMEMORIZE$(\phi_{k-1}, z_{k-1}, c, a, \mathcal{B}, M, \eta)$
      $(\ell_k, \mathcal{B}) \leftarrow$ REPLAYPREDLOSS$(z_k, \phi_k, c, a, \mathcal{B})$ ▷ One-step estimates in Section 2.2
    end for
    $\mathcal{J} \leftarrow \sum_{k=2}^{N} \ell_k$
    $g_{\phi_0} \leftarrow \sum_{k=2}^{N} \nabla_{\phi_k} \ell_k$ ▷ FOMAML, equation 5
    $\phi_0 \leftarrow$ OUTERUPDATE(ϕ_0, g_{ϕ_0})
  end while
  return ϕ_0
end procedure

procedure TTT2REEL-INFERENCE(p, ϕ_0, N, S)
  $c \leftarrow \mathrm{T5}(p)$ and $\phi_1 \leftarrow \phi_0$
  $x_1 \leftarrow$ GENERATE$(\theta_{\mathrm{b}}, \phi_1, c, \varnothing, S)$
  $a \leftarrow \mathcal{E}(x_1^{(1)})$
  for $k = 2, \ldots, N$ **do**
    $z_{k-1} \leftarrow \mathcal{E}(x_{k-1})$
    $\phi_k \leftarrow U^M(\phi_{k-1}; z_{k-1}, c, a)$ ▷ Apply equation 2 M times
    $x_k \leftarrow$ GENERATE$(\theta_{\mathrm{b}}, \phi_k, c, a, S)$
  end for
  $x_{\mathrm{long}} \leftarrow x_1 \,\|\, \cdots \,\|\, x_N$ ▷ Concatenate generated chunks
  return x_{long}
end procedure

Exact meta-gradients (MAML). Exact MAML differentiates the outer objective $\mathcal{J}$ through the entire sequence of inner-loop updates. For the objective in equation 3, the chain rule gives

$$\nabla_{\phi_0} \mathcal{J} = \sum_{k=2}^{N} \left(\frac{\partial \phi_k}{\partial \phi_0} \right)^{\top} \nabla_{\phi_k} \ell_k. \tag{4}$$

The Jacobian $\partial \phi_k / \partial \phi_0$ accounts for all memorization steps preceding the prediction of chunk k. For the gradient-descent update illustrated in equation 2, each step contributes a Jacobian of the form $I - \eta \nabla_{\phi}^2 \mathcal{L}$. Differentiating through these steps therefore involves Hessian-vector products and retaining or recomputing intermediate states across the rollout. This cost is prohibitive for video diffusion backbones, so we include exact MAML only as a theoretical reference.

FOMAML. We approximate $\partial \phi_k / \partial \phi_0$ in equation 4 with the identity matrix, following FOMAML (Finn et al., 2017). This gives,

$$\nabla_{\phi_0} \mathcal{J} \approx \sum_{k=2}^{N} \nabla_{\phi_k} \ell_k. \tag{5}$$

Each prediction gradient is evaluated at the corresponding adapted LoRA parameters ϕ_k, and the accumulated gradients are used to update the initialization ϕ_0. This approximation avoids differentiating through the memorization updates and therefore requires neither second-order derivatives nor retaining the inner-loop computation graph. All models in this paper are trained using FOMAML. We leave the evaluation of exact meta-gradients at this scale to future work.

B.2 Preliminaries on Latent Diffusion Models

Latent diffusion models perform generative modeling in the compact latent space of a pretrained autoencoder rather than directly in pixel space (Rombach et al., 2022). Given a video chunk x_k, the Wan-VAE encoder $\mathcal{E}$ produces a latent representation $z_k = \mathcal{E}(x_k)$, while its decoder $\mathcal{G}$ maps the latent back to pixel space. This representation reduces the computational cost of modeling high-resolution videos while retaining the information required for reconstruction.

The Wan backbone used by TTT2Reel learns the latent dynamics through conditional flow matching (Lipman et al., 2022; Wan et al., 2025). As defined in equation 1, a clean latent z and Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$ are connected by the linear path $z_t = tz + (1-t)\epsilon$, where $t \sim \mathcal{U}[0,1]$. The model with weights θ learns the conditional velocity field $v_t(z_t; \theta, c)$ by matching the target velocity $u_t(z_t \mid z) = z - \epsilon$, where c denotes the text conditioning.

Wan parameterizes the velocity field with a diffusion transformer (Peebles & Xie, 2023; Wan et al., 2025). The noisy video latent is patchified into a sequence of spatiotemporal tokens and processed by transformer blocks. Timestep information is incorporated through adaptive modulation, while text conditioning is introduced through cross-attention. The resulting tokens are projected back to the latent grid to predict the velocity field.

At inference, sampling starts from $z_0 \sim \mathcal{N}(0, I)$ and numerically integrates

$$\frac{\mathrm{d}z_t}{\mathrm{d}t} = v_t(z_t; \theta, c)$$

from $t = 0$ to $t = 1$ using S sampling steps. The VAE decoder $\mathcal{G}$ then maps the terminal latent z_1 to the generated video chunk. In TTT2Reel, this latent-space objective underlies both inner-loop memorization and outer-loop next-chunk prediction.

B.3 Preliminaries on Low-Rank Adaptation (LoRA)

Low-rank adaptation (LoRA) provides a parameter-efficient way to adapt a pretrained model while keeping its original parameters frozen (Hu et al., 2021). For a linear projection with frozen weight $W_0 \in \mathbb{R}^{d_{\mathrm{out}} \times d_{\mathrm{in}}}$, LoRA parameterizes its update as

$$W = W_0 + \Delta W, \qquad \Delta W = BA, \tag{6}$$

where $A \in \mathbb{R}^{r \times d_{\mathrm{in}}}$ and $B \in \mathbb{R}^{d_{\mathrm{out}} \times r}$ are trainable factors, and $r \ll \min(d_{\mathrm{in}}, d_{\mathrm{out}})$. The update satisfies $\mathrm{rank}(\Delta W) \le r$ and introduces $r(d_{\mathrm{in}} + d_{\mathrm{out}})$ trainable parameters, compared with $d_{\mathrm{in}} d_{\mathrm{out}}$ parameters for a full weight update. Given a hidden activation h, the adapted projection computes $Wh = W_0 h + BAh$. Only the low-rank factors are updated, while W_0 remains fixed.

In TTT2Reel, we collectively denote the LoRA factors across all adapted projections by ϕ, yielding the parameter decomposition $\Theta = (\theta_{\mathrm{b}}, \phi)$, where θ_{b} is the frozen Wan backbone. Rather than serving as a static task-specific adapter, ϕ evolves across video chunks as a running parameter-space state. Starting from the meta-learned initial LoRA parameters ϕ_0, the inner update in equation 2 writes information from the completed chunk z_{k-1} into ϕ_k. The adapted model $(\theta_{\mathrm{b}}, \phi_k)$ then uses this state when generating chunk k. The size of this parametric state is determined by the adapted projections and their ranks and does not grow with the number of generated chunks.

B.4 Preliminaries on Model-Agnostic Meta-Learning (MAML)

Model-agnostic meta-learning (MAML) learns a parameter initialization that can be rapidly adapted through a small number of gradient steps (Finn et al., 2017). Let τ denote a task, and let $\mathcal{L}_\tau^{\mathrm{in}}$ and $\mathcal{L}_\tau^{\mathrm{out}}$ denote its inner-loop adaptation loss and outer-loop evaluation loss. Starting from the meta-parameters $\phi_\tau^{(0)} = \Theta$, MAML performs M inner updates

$$\phi_\tau^{(m+1)} = \phi_\tau^{(m)} - \eta \nabla_\phi \mathcal{L}_\tau^{\mathrm{in}}\big(\phi_\tau^{(m)}\big), \qquad m = 0, \dots, M-1, \tag{7}$$

and optimizes the post-adaptation objective

$$\min_{\Theta} \; \mathbb{E}_\tau \left[\mathcal{L}_\tau^{\mathrm{out}}\big(\phi_\tau^{(M)}\big) \right]. \tag{8}$$

Table 4: Statistics of the public UltraVideo-Long release (Xue et al., 2025).

Statistic	Value
Number of videos	16,597
Duration criterion	≥ 10 seconds
Total duration	143 hours
Average duration	30.9 seconds
Long-side resolutions	1,920 and 960 pixels
Average detailed caption length	138.6 words per video

The separation between the inner and outer losses encourages Θ to support effective adaptation rather than merely minimize the loss before adaptation.

Exact MAML differentiates the outer loss through every inner update. Each gradient step contributes a Jacobian containing a Hessian term of the form $I - \eta \nabla^2_\phi \mathcal{L}^{\text{in}}_\tau$. First-order MAML (FOMAML) treats the inner-loop gradients as constants and approximates these update Jacobians by the identity (Finn et al., 2017; Nichol et al., 2018). It therefore uses

$$\nabla_\Theta \mathcal{L}^{\text{out}}_\tau\big(\phi^{(M)}_\tau\big) \approx \nabla_{\phi^{(M)}_\tau} \mathcal{L}^{\text{out}}_\tau\big(\phi^{(M)}_\tau\big), \tag{9}$$

which retains the outer gradient evaluated at the adapted parameters while avoiding second-order derivatives.

In TTT2Reel, each chunked video rollout plays the role of a meta-learning episode, and ϕ_0 denotes the initial LoRA parameters. At each transition, the completed chunk z_{k-1} provides the inner loss used to obtain ϕ_k through equation 2, while the following chunk z_k provides the post-adaptation prediction loss ℓ_k. These losses are accumulated over $k = 2, \ldots, C$ in equation 3. The corresponding exact and first-order meta-gradients are given in equation 4 and equation 5, respectively. All experiments use FOMAML because retaining the complete inner-loop computation graph is prohibitively expensive for video diffusion backbones.

C Dataset Statistics

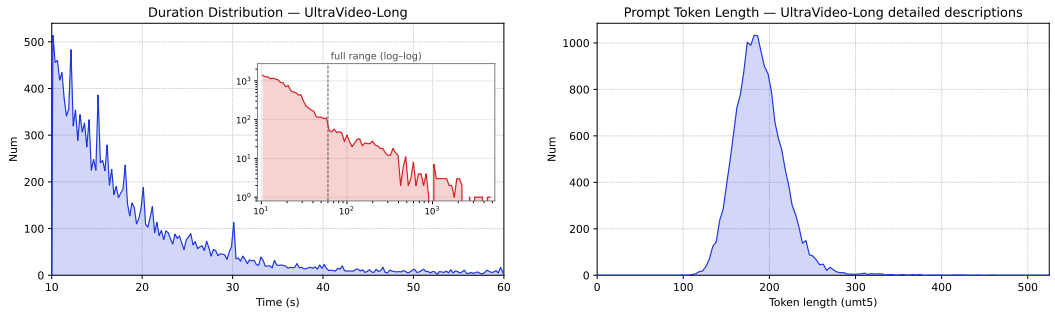
We use the public UltraVideo-Long release (Xue et al., 2025) downloaded from its official Hugging Face repository.[2] The dataset contains 16,597 videos of at least 10 seconds, with a total duration of 143 hours and an average duration of 30.9 seconds. The repository provides two spatially downsampled variants, `clips_long_1920` and `clips_long_960`, whose long sides are resized to 1,920 and 960 pixels, respectively. Table 4 summarizes the principal statistics.

The complete UltraVideo collection was constructed from approximately 5K source videos covering seven broad themes and 108 topics. The long-video subset follows the same four-stage curation pipeline as the short-video subset. Scene segmentation and duration filtering initially produced approximately 25K long clips. Statistical filtering reduced this set to 19K clips by removing videos with excessive text, black borders, abnormal exposure, or low saturation. Model-based purification then evaluated visual quality, temporal motion, text–video alignment, and other undesirable attributes, producing the final set of 16,597 videos.

Figure 8 shows the distributions of clip duration and prompt token length measured with the umT5 tokenizer in the UltraVideo-Long subset. Durations are heavy-tailed, with most clips sitting close to the 10-second lower bound and the count decays steadily out to a long tail of multi-minute videos. Prompts are uniformly long and detailed, concentrated around 180 tokens, which supplies the dense narrative conditioning that TTT2Reel relies on at each chunk transition.

Figure 9 shows four clip examples from UltraVideo-Long. Each example pairs the detailed caption with the first and last frame of the clip, illustrating both the visual diversity of the subset and the amount of narrative detail the captions carry across hundreds of frames.

[2] `https://huggingface.co/datasets/APRIL-AIGC/UltraVideo-Long`



(a) Clip duration.

(b) Prompt token length.

Figure 8: Statistics of the UltraVideo-Long subset. **(a)** Distribution of clip durations. The inset shows the full range on log-log axes, with the dashed line marking the 60-second cutoff of the main panel. **(b)** Distribution of prompt token lengths, measured with the umT5 tokenizer.

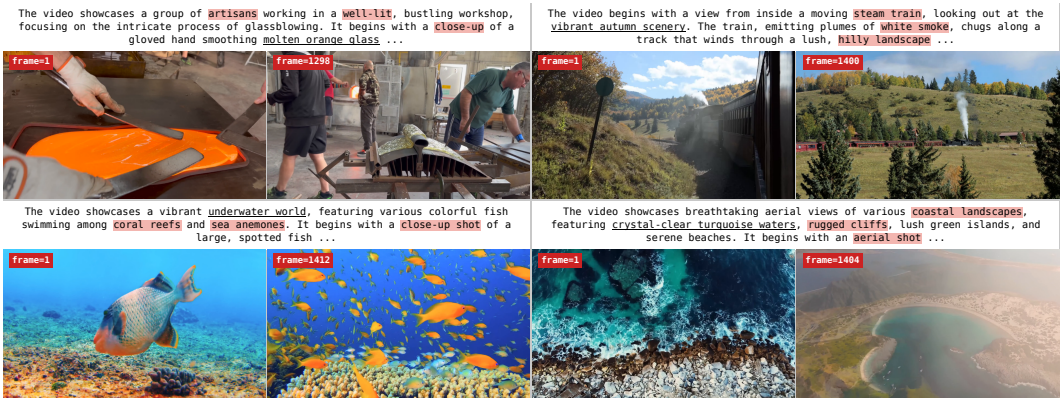


Figure 9: Video examples of UltraVideo-Long. For each clip, we show the beginning of its detailed caption together with the first and last frame. The highlighted spans mark the subjects and scene attributes that remain consistent over the full clip.

D Implementation Details

We apply rank-128 LoRA adapters with scaling parameter $\alpha = 128$ to the query, key, value, and output projections of attention and the two feed-forward projections in every transformer block, while keeping the pretrained backbone frozen. The LoRA matrices A and B are initialized with Kaiming-uniform weights and zeros, respectively, before meta-training. For the inner-loop visual memory encoding step in both training and inference, we use AdamW with a learning rate of 10^{-5} and two gradient steps per memorized chunk, with gradient clipping at 1.0. The memorization loss is conditioned on the text prompt, and the inner optimizer state is retained across chunks within each training clip. At inference, adaptation uses an FP32 master copy of the LoRA parameters, which are reset to the meta-learned initialization ϕ_0 for each new video.

We meta-train the LoRA initialization using FOMAML for three epochs with a global batch size of 16 on 16 H100 GPUs distributed across four nodes. Training uses BF16 precision and gradient checkpointing at each transformer block. The outer optimizer is AdamW, with learning rates of 10^{-5} for Wan2.2-TI2V-5B and 10^{-4} for Wan2.1-I2V-14B-480P. Both configurations use a 30-step warmup followed by cosine decay to 10% of the initial learning rate. Error replay uses 50 discretized timestep bins, with injection probabilities of 0.01 for noise residuals and 0.9 for both video-latent and anchor residuals. For the 5B and 14B models, respectively, the clean-input probabilities are 0.5 and 0.2, and each timestep bin stores at most 300 and 500 residuals per node.

Wan2.2-TI2V-5B uses 133-frame chunks at a spatial resolution of 720×1280 and 24 FPS. It incorporates three clean latent frames from the preceding chunk by overwriting the leading chunk latents and retains the video's first latent as a persistent anchor. Wan2.1-I2V-14B-480P uses 81-

Parameter	Value	Description
Training		
Learning rate	1.0e-05	AdamW learning rate of the outer (meta) update
LR schedule	cosine	30-step warmup, cosine decay to $0.1\times$
Max epochs	3	Training epochs
Batch size	16	Videos per step (1 per GPU)
Training strategy	DDP	Data-parallel, 4 nodes $\times$ 4 H100
Precision	bf16	Weights, LoRA and optimizer states
Data workers	2	Data loading workers per GPU
Gradient checkpointing	Yes	Per-DiT-block activation recomputation
LoRA rank	128	Low-rank adaptation rank dimension
LoRA alpha	128	LoRA scaling parameter ($\alpha/r = 1$)
LoRA init	kaiming / zero	A Kaiming-uniform, B zero
LoRA position	q,k,v,o,ffn.0,ffn.2	Target modules, all 30 blocks
Frame height	720	Video frame height (pixels)
Frame width	1280	Video frame width (pixels)
Frame rate	24	Frames per second
Meta-algorithm	FOMAML	Memorize chunk $k-1$, predict chunk k
Inner optimizer	AdamW	Memorize-step optimizer
Inner learning rate	1.0e-05	Memorize-step learning rate
Inner steps	2	Gradient steps per memorized chunk
Sigma shift	5	Training noise-schedule shift
Text in memorize	Yes	Memorize loss conditioned on the caption
Anchoring route	fused	Clean latents overwrite leading chunk latents
Frames per chunk	133	Pixel frames per chunk
Local anchor latents	3	Clean latents from the previous chunk
First-frame sink	Yes	Clip's first latent pinned in every chunk
Error replay	Yes	Enable error-replay tuning
Warmup iterations	20	Iterations gathering errors before injection
Noise error p_{noi}	0.01	Noise error injection probability
Latent error p_{vid}	0.9	Latent error injection probability
Anchor error p_{img}	0.9	Anchor-latent error injection probability
Clean input p	0.5	Probability without any error
Timestep grids	50	Discretized timestep grids for error buffer
Maximum error Z	300	Maximum errors per grid
Inference		
Denoising steps	50	Flow-matching Euler steps per chunk
CFG scale	5.0	Text classifier-free guidance scale
Sigma shift	5.0	Sampler noise-schedule shift
Negative prompt	Wan default	Negative text for CFG
Resolution	720×1280	Output frame size
Frame rate	24	Output frames per second
Frames per chunk	133	Frames generated per chunk
TTT steps per chunk	1×2	Memorize passes $\times$ gradient steps
Test-time learning rate	1.0e-05	Inner learning rate
Test-time optimizer	AdamW	Reset to ϕ_0 per video

Table 5: Detailed hyperparameters of Wan2.2-TI2V-5B used in the training and test.

frame chunks at 480×832 and 16 FPS, with its pretrained image-to-video conditioning pathway and one motion reference frame from the preceding chunk. At inference, both models use 50 denoising steps per chunk, a classifier-free guidance scale of 5.0, a noise-schedule shift of 5.0, and the default Wan negative prompt. Tables 5 and 6 provide the complete hyperparameter configurations.

E Benchmark Details

E.1 VBench-Long

VBench-Long is the long-video extension of VBench (Huang et al., 2024), a comprehensive benchmark that decomposes video-generation quality into complementary evaluation dimensions. It is designed for extended-duration outputs and assesses both local perceptual quality and long-range

Parameter	Value	Description
Training		
Learning rate	1.0e-04	AdamW learning rate of the outer (meta) update
LR schedule	cosine	30-step warmup, cosine decay to $0.1\times$
Max epochs	3	Training epochs
Batch size	16	Videos per step (1 per GPU)
Training strategy	DDP	Data-parallel, 4 nodes $\times$ 4 H100
Precision	bf16	Weights, LoRA and optimizer states
Data workers	2	Data loading workers per GPU
Gradient checkpointing	Yes	Per-DiT-block activation recomputation
LoRA rank	128	Low-rank adaptation rank dimension
LoRA alpha	128	LoRA scaling parameter ($\alpha/r = 1$)
LoRA init	kaiming / zero	A Kaiming-uniform, B zero
LoRA position	q,k,v,o,ffn.0,ffn.2	Target modules, all 40 blocks
Frame height	480	Video frame height (pixels)
Frame width	832	Video frame width (pixels)
Frame rate	16	Frames per second
Meta-algorithm	FOMAML	Memorize chunk $k-1$, predict chunk k
Inner optimizer	AdamW	Memorize-step optimizer
Inner learning rate	1.0e-05	Memorize-step learning rate
Inner steps	2	Gradient steps per memorized chunk
Sigma shift	5	Training noise-schedule shift
Text in memorize	Yes	Memorize loss conditioned on the caption
Anchoring route	i2v	Pretrained I2V conditioning (y + CLIP)
Frames per chunk	81	Pixel frames per chunk
Motion frames	1	Number of motion reference frames
Reference padding	-1	First frame fills all non-motion slots of y
Error replay	Yes	Enable error-replay tuning
Warmup iterations	50	Iterations gathering errors before injection
Noise error p_{noi}	0.01	Noise error injection probability
Latent error p_{vid}	0.9	Latent error injection probability
Image error p_{img}	0.9	Image error injection probability
Clean input p	0.2	Probability without any error
Timestep grids	50	Discretized timestep grids for error buffer
Maximum error Z	500	Maximum errors per grid
Inference		
Denoising steps	50	Flow-matching Euler steps per chunk
CFG scale	5.0	Text classifier-free guidance scale
Sigma shift	5.0	Sampler noise-schedule shift
Negative prompt	Wan default	Negative text for CFG
Resolution	480 $\times$ 832	Output frame size
Frame rate	16	Output frames per second
Frames per chunk	81	Frames generated per chunk
TTT steps per chunk	1×2	Memorize passes $\times$ gradient steps
Test-time learning rate	1.0e-05	Inner learning rate
Test-time optimizer	AdamW	Reset to ϕ_0 per video

Table 6: Detailed hyperparameters of Wan2.1-I2V-14B-480P used in the training and test.

temporal behavior, including whether subjects and backgrounds remain stable as a video unfolds. Following its standard evaluation procedure, we report six reference-free dimensions: Subject Consistency, Background Consistency, Motion Smoothness, Dynamic Degree, Aesthetic Quality, and Imaging Quality. TTT2Reel generates 60 seconds of video with 50 denoising steps, a classifier-free guidance scale of 5.0, and the Wan negative prompt. All baselines use their own recommended video sampling setups.

E.2 VISIONREWARD

VisionReward is a fine-grained, multidimensional framework for learning and evaluating human preferences for generated images and videos (Xu et al., 2024). It decomposes overall visual preference into a hierarchy of interpretable assessment criteria, allowing evaluations to identify spe-

cific strengths and weaknesses rather than reducing quality to a single opaque score. We adopt its video-assessment questions for head-to-head comparisons, complementing VBench-Long's automated metrics with a preference-oriented evaluation of the generated videos.

The released VisionReward evaluator uses CogVLM2, whose 2,048-token context and 24-frame input limit provide at most two sampled frames per chunk on average for a 60-second video comprising 12 chunks. This temporal coverage is too sparse to reliably distinguish gradual degradation within a chunk from a discontinuity at a chunk boundary, even though these are central failure modes in long-video generation. We therefore retain VisionReward's original checklist that contains 29 questions, weights, and score reduction but replace its judge with Gemini-3.7-Flash, whose one million token context window supports denser temporal sampling. We process every video at 4.0 FPS, yielding approximately 240 sampled frames per 60-second video, or approximately 20 frames per chunk.

We issue each of the 29 questions as a separate request at temperature 0. To reduce the cost of separate requests, we encode each video once using context caching and reuse the cached representation across all 29 questions. The reference and modified evaluators load the same 29 questions and the same published weight vector $\mathbf{w} \in \mathbb{R}^{29}$. For question j, a "yes" answer is mapped to $a_j = +1$ and a "no" answer to $a_j = -1$. The score for a video is

$$S = \frac{1}{29} \sum_{j=1}^{29} w_j a_j.$$

The three prompt-conditioned questions account for 37.4% of the total absolute weight mass, making VisionReward substantially more sensitive to prompt adherence than the reference-free VBench-Long dimensions. Since the published weight vector is obtained by calibrating the CogVLM2 model, we cannot directly borrow the same weights for score calculation when using Gemini-3.7-Flash as the judge. So we report the pass rate instead that counts the number of questions with answer "yes".

We report the instruction-following pass rate as the percentage of evaluated videos for which the judge answers "yes" to the first question, "Does the video satisfy every requirement in `[[prompt]]`?"

The following checklist presents the 29 binary questions in the same order as the corresponding entries in the published weight vector.[3] For Questions 1–3, the placeholder `[[prompt]]` is replaced by the generation prompt. Our evaluator uses the original released query strings verbatim.

1. Does the video satisfy every requirement in `[[prompt]]`?
2. Does the video satisfy most requirements in `[[prompt]]`?
3. Does the video avoid completely failing to satisfy `[[prompt]]`?
4. Is the composition visually pleasing?
5. Is the composition free from conspicuous defects?
6. Is the camera movement free from conspicuous defects?
7. Do the colors avoid being markedly unattractive?
8. Is the lighting fully accurate?
9. Is the lighting free from conspicuous errors?
10. Is lighting present in the video?
11. Is the lighting exceptionally attractive?
12. Is the lighting attractive?
13. Is the lighting at least not unattractive?
14. Is the object's shape fully accurate at the beginning?
15. Is the object's initial shape free from conspicuous errors?
16. Is the object's initial shape coherent rather than chaotic?
17. Is the object's shape maintained perfectly throughout the video?
18. Does the object's shape remain coherent throughout the video?
19. Is the camera motion highly dynamic?
20. Is the amount of camera motion greater than minimal?
21. Is the object's movement highly smooth?

[3] `https://github.com/zai-org/VisionReward/blob/main/VisionReward_Video/VisionReward_video_qa_select.txt`



Figure 10: **Additional qualitative comparisons.** Frames sampled at 10-second intervals from one-minute rollouts.

22. Is the object's movement fully realistic?
23. Does the image quality remain highly stable?
24. Are the visual details highly refined?
25. Are the visual details reasonably free from roughness?
26. Are the visual details free from severe roughness?
27. Is all visible lettering correct?
28. Is any lettering visible?
29. Does the video depict content belonging to the physical world?

F More Experimental Results

F.1 More Qualitative Results

Figure 10 provides two additional qualitative comparisons. Both comparisons show frames sampled at fixed intervals from one-minute rollouts, with all methods initialized from the same first frame.