PlasticMem: Adding Temporal Reasoning to Diffusions for Consistent Long Video Generation

Yufei Huang[2] **Chao Liang**[3] **Zerong Zheng**[4] **Tianshu Hu**[4] **Jianwen Jiang**[5]
Yuan Zhang[6] **Mingyuan Gao**[3] **Chang Yu**[1,*] **Stan Z. Li**[7]

[1]School of Intelligence Science and Technology, Nanjing University
[2]Zhejiang University [3]Seedance Team, ByteDance [4]Seed, ByteDance
[5]Meta Super Intelligence, Meta [6]Intelligent Creation Team, ByteDance
[7]AI Lab, Research Center for Industries of the Future, Westlake University

Abstract

Scaling video generation from short to long durations has attracted growing attention as a way to democratize video creation. The central challenge lies in *long-term consistency*, which requires *stable rollout* over extended horizons and accurate *visual memory* of previously generated elements. However, the scarcity of high-quality long video data remains a fundamental bottleneck, forcing prior methods to rely on suboptimal synthetic data that either limits dynamic range and visual memory or suffers from a large simulation-to-reality gap. To address this, we explore a new generation paradigm, Temporal Reasoning-based Rendering (TRR), which better exploits existing short video data for consistent long video generation. Under TRR, we propose *PlasticMem*, which plans contents from scripts, then reasons about and independently renders each action to form a consistent long video. Benchmark results show that PlasticMem, trained on short videos, achieves stable minute-scale long-horizon generation and visual memory.

1 Introduction

Video generation has drawn increasing attention as a way to democratize high-quality video creation [Zhang et al., 2025a, HaCohen et al., 2025], allowing creators to focus on expression rather than production overhead. Despite recent progress in text- or image-conditioned generation (TI2V) [Wan et al., 2025, Gao et al., 2025], current methods are largely limited to short video clips, which remain insufficient for visual storytelling [Chen et al., 2025a, Yu et al., 2025b].

Scaling from short to long videos is constrained by **long-term consistency**: keeping stable under long-term simulation with large visual changes (i.e., **Stable Rollout**) and preserving recurring elements across distant temporal segments (i.e., **Visual Memory**). Crucially, these challenges are intertwined, as repeated recall of past content during long-horizon generation induces error accumulation, leading to gradual drift, which substantially complicates consistent long video generation.

Ensuring long-term consistency remains difficult largely due to the lack of high-quality long video data with long durations and recurring elements (e.g., subject recall or scene revisit). To mitigate this data scarcity, existing methods primarily rely on synthetic data including generated and rendered videos [Cui et al., 2025, Yu et al., 2025a]. Methods based on generated video data have made progress in mitigating drift [Huang et al., 2025a, Yang et al., 2025a, Liu et al., 2025]. However, these methods require segmenting long videos to perform segment-wise distribution matching distillation with pretrained models [Yin et al., 2025, 2024b], making it **difficult to directly model long-range dependency** (as in Fig. 1). On the other hand, methods based on rendered data [Yu et al., 2025a,

*Corresponding author.

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



Figure 1: Streaming long video generation often forgets generated elements, degrading long-term consistency. We attribute this to a lack of understanding of the underlying world behind generated videos. *PlasticMem* addresses this via explicit world reasoning learned from short videos, achieving consistent long video generation without relying on scarce long-video data.

Xiao et al., 2025, Li et al., 2025] leverage highly controllable game engines or interactive renderers to generate large-scale training data with recurring elements, which enables models to learn to accurately recall past content during generation. However, these approaches are **difficult to generalize to more general video generation scenarios** due to the large gap between simulation and reality. Real-world videos exhibit a much larger action space and lack the fine-grained dynamic control signals provided by simulations (as shown in Fig. 2).

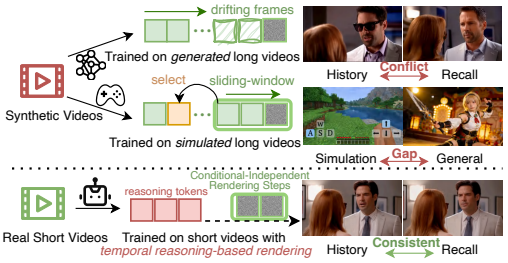


Figure 2: Illustration of our motivation. Top: Prior synthetic-data-based approaches either fail to preserve visual memory of past elements or suffer from a simulation-to-generalization gap. Bottom: Our framework decouples consistent long video generation into reasoning about what to generate and conditionally independent short-video rendering, both trainable on short videos.

Instead of relying on suboptimal synthetic data, we explore a new generation paradigm that better exploits existing short-video data for consistent long video generation. Inspired by reasoning paradigms in texts [Hurst et al., 2024], where complex tasks are solved via planning and decomposition, we propose ***Temporal Reasoning-based Rendering*** (TRR). TRR decouples video generation as visual content reasoning and *independent* rendering steps, and thus learns each from abundant short videos, enabling stable rollout and visual memory with improved data efficiency of existing short video data. Accordingly, we propose ***PlasticMem***, a TRR implementation that first generates latent reasoning frames to plan what to generate from the script (***text-to-reasoning***), and then progressively renders *conditionally independent* short videos in a sliding window conditioned on the script and reasoning frames (***reasoning-for-rendering***). During rendering, PlasticMem infers contents of each window in a chain-of-frames manner, based on its relative position in the full video encoded by temporal RoPE, and conditions on relevant reasoning tokens and local context frames. Unlike subject or keyframe-based approaches with provided references, reasoning frames here are *self-generated* from text and may not appear in the final video, serving only as probes of the underlying world.

To improve data efficiency, we adopt an in-context unified training scheme, analogous to Large Language Models [Hurst et al., 2024], that unifies standard video generation with the proposed temporal reasoning and rendering tasks, enabling effective knowledge sharing across tasks. To support in-context learning and chain-of-frame reasoning, we introduce a Temporal RoPE strategy that assigns different token types to distinct temporal regions. For evaluation, we construct a consistent long-video benchmark covering diverse video lengths, multi-shot storyboards, and large-scale motions.

Experiments show that PlasticMem, trained on short videos with an average length of about 5 seconds under the TRR framework, achieves stable long-horizon generation and visual memory at the minute scale. Our contributions are summarized as follows:

- We study consistent long video generation in general TI2V settings using only short video data, and introduce benchmarks to evaluate long-term consistency.
- We propose the TRR framework that decouples long video generation into reasoning about what to generate and independent, progressive rendering of dynamics, enabling generalization from short to long videos.
- We introduce *PlasticMem*, a data-efficient TRR method that achieves stable rollout and visual memory with in-context unified training on short video data only.

2 Related Works

Autoregressive video diffusion models. A central thread in long video generation bridges autoregressive (AR) and diffusion models to combine temporal scalability with high-quality rendering [Kim et al., 2024, Chen et al., 2024]. Existing AR–diffusion hybrids mainly follow two paradigms. The first assigns different noise levels to different frames, inducing autoregressive generation by conditioning noisier future frames on cleaner past frames, as in FIFO-Diffusion [Kim et al., 2024] and Diffusion Forcing [Chen et al., 2024]. This paradigm enables efficient streaming generation and has been scaled by methods such as CausVid [Yin et al., 2025], MAGI-1 [Teng et al., 2025], SkyReels-V2 [Chen et al., 2025a], and StreamDiT [Kodaira et al., 2025]. The second paradigm integrates AR and diffusion more explicitly via next-frame diffusion with anti-drifting mechanisms for stable rollouts, exemplified by self-forcing [Huang et al., 2025a] and its extensions [Cui et al., 2025, Yang et al., 2025a, Liu et al., 2025]. These methods rely on segment-wise distribution-matching distillation [Yin et al., 2024b,a] and thus do not explicitly model long-term dependency for visual memory. In contrast, we propose a simpler paradigm that jointly supports memory and stable rollouts, shifting long-term consistency from conditioning on degraded past frames to reasoning over stable visual anchors.

Visual Memory in Video Generation. Long-form video storytelling often requires scene revisits and subject recall, making visual memory critical for long-term consistency [Cai et al., 2025, Ji et al., 2025]. Existing approaches mainly fall into retrieval- and compression-based paradigms. Retrieval-based methods maintain consistency by looking up past content during generation, such as FoV-based history retrieval in WorldMem [Xiao et al., 2025] and Context-as-Memory [Yu et al., 2025a], sparse context selection in Mixture-of-Contexts [Cai et al., 2025], and prompt-indexed retrieval in MemFlow [Ji et al., 2025]. Compression-based methods instead summarize unbounded histories into compact representations, including explicit 3D scene compression [Ren et al., 2025, Li et al., 2025], test-time parameter encoding [Dalal et al., 2025, Zhang et al., 2025b], token budget allocation [Zhang et al., 2025a, Gu et al., 2025], and fixed-size hidden states [Yu et al., 2025b, Savov et al., 2025]. However, both paradigms rely on high-quality long video data and thus perform better in simulation videos than on real videos, where samples with recurring elements are sparse. In contrast, we rethink memory by reversing the order of generation and memory: instead of retrieving or compressing histories, we first generate compact text-aligned reasoning frames specifying what to generate, then condition rendering on them for consistent long video generation. Unlike keyframe-anchored long-take generators that condition rendering on keyframes that themselves appear in the final video, our reasoning frames are *self-generated* from text and serve only as world probes that are discarded at output time, sidestepping reliance on dense keyframe annotations.

3 Preliminaries

3.1 Video Generation

Task Formulation. Given a user-provided prompts $\boldsymbol{c} = \{c_{txt}, c_{img}\}$ containing a story script c_{txt}, optionally the first frame image condition c_{img}, our goal is to generate a consistent long video $\boldsymbol{x} = \{x_i\}_{i=1}^{F}$ of a specified frame number F that intelligently allocates appropriate duration to each action in the script and completes the story.

The standard practice for video generation models [Wan et al., 2025, HaCohen et al., 2025] is to compress video frames $\boldsymbol{x}$ into latent frames $\boldsymbol{z} = \{z^j\}_{j=1}^{N}$ with Variational AutoEncoder (VAE) and model generation using diffusion models [Ho et al., 2020]. Standard training and sampling follows

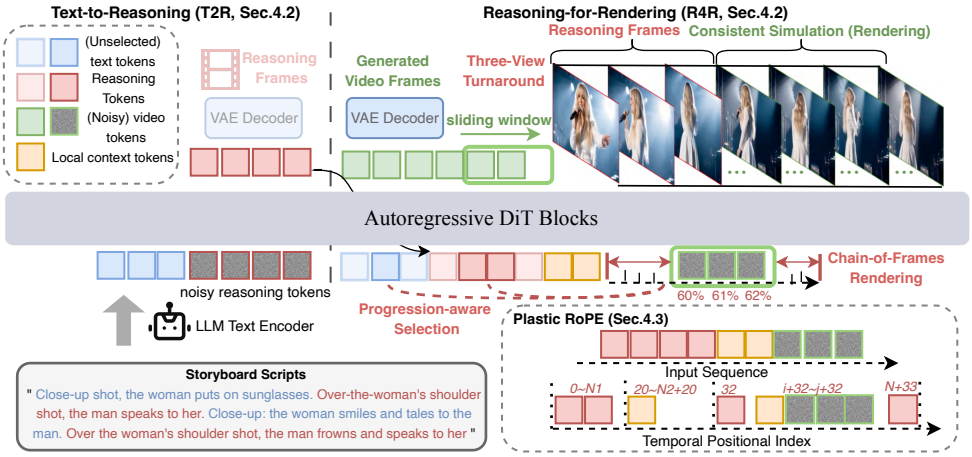


Figure 3: Temporal Reasoning-based Rendering framework of *PlasticMem*. Reasoning frames are generated from a storyboard script to probe the underlying world and support conditionally independent rendering steps, enabling consistent long video generation with visual memory. During streaming rendering, reasoning frames are prepended before noisy video frames within the current window, serving only as visual references rather than key frames in the final video. The model then performs chain-of-frame reasoning to infer progression and determine the content to render, assisted by PlasticRoPE (dashed lines indicate token-type–aware index partitioning).

the Flow Matching framework [Lipman et al., 2023]. During training and inference, each noisy latent frame is flattened into a sequence of latent tokens and processed with a Diffusion Transformer (DiT) denoising network [Peebles and Xie, 2022]. The noisy latent frames are defined as $z_t^j = (1-t)\epsilon + tz^j$, where ϵ is a standard Gaussian noise and t is the flow-matching timestep.

3.2 Diffusion Forcing

To enable streaming video generation, we adopt Diffusion Forcing (DF) [Chen et al., 2024, Kodaira et al., 2025], which combines the autoregressive and diffusion models. Unlike conventional video diffusion models that apply the same noise level to all frames in a video and denoise them synchronously, DF assigns different noise levels $\boldsymbol{t} = [t_1, ..., t_N]$ to different frames and denoises earlier frames faster. This noise-level difference naturally induces autoregressive modeling in video diffusion, where noisier future frames condition on less noisy past frames. Formally, the DiT is trained with the following per-frame flow matching objective:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{z},\boldsymbol{t},\boldsymbol{\epsilon}}\left[\|\boldsymbol{v}_\theta(\boldsymbol{z_t}, \boldsymbol{t}) - \boldsymbol{v_t}\|\right], \boldsymbol{v_t} = \frac{d\boldsymbol{z}_t}{dt} = \boldsymbol{z}_1 - \boldsymbol{z}_0 \tag{1}$$

where $\boldsymbol{z_t} = \{z_{t_j}^j\}_{j=1}^N$, $\boldsymbol{\epsilon}$ denotes the noise added to all frames and $\boldsymbol{v_t}$ is the conditional velocity to a clean video. At inference, the noisy video is updated by predicted velocity:

$$\boldsymbol{z_{t+\Delta t}} = \{z_{t_j+\Delta t^j}^j\}_{j=1}^N, \quad z_{t_j+\Delta t^j}^j = z_{t_j}^j + v_\theta^j \Delta t^j \tag{2}$$

where $\boldsymbol{\Delta t} = [\Delta t^j]_{j=1}^N$ is a per-chunk step-size sequence.

4 Method

4.1 Temporal Reasoning-based Rendering

Motivations. Existing paradigms for long video generation are largely centered on modeling history for complete long video contexts, which can be formalized as:

$$p(\boldsymbol{z}^{1:N}|\boldsymbol{c}) = \prod_{i=1}^{N} p(z^i|\boldsymbol{z}^{<i}, \boldsymbol{c}) \approx \prod_{i=1}^{N} \hat{p}(z^i|\hat{\boldsymbol{z}}^{<i}, \boldsymbol{c}). \tag{3}$$

where $p(\boldsymbol{z}^{1:N}|\boldsymbol{c})$ denotes the joint conditional distribution of the long video sequence, which could be factorized as above via the chain rule. $\boldsymbol{z}^{<i}$ represents the history preceding the current video segment,

while $\hat{\boldsymbol{z}}^{<i}$ denotes the *approximated history* including self-generated frames used for anti-drifting, as well as compressed or selectively retained histories with limited size to ensure computational tractability for visual memory. $\hat{p}$ denotes an estimated distribution that may be biased. For example, the DMD-based self-forcing methods distill long video sequences by decomposing them into short clips, which neglects long-range dependencies and thus introduces bias.

These paradigms, which generate the future by conditioning on an unbounded past, **inherently rely on large-scale long-video data** with similarly unbounded histories to achieve training–inference consistency and strong performance. However, **such long-video data are scarce** and often lie in the long tail of the data distribution.

A New Perspective. Beyond seeking additional sources of long-video data (e.g., synthetic videos), we explore a fundamentally different generative modeling paradigm that instead fully leverages the existing abundant short-video data. Specifically, we decompose and model long videos under a ***conditional independence*** assumption:

$$p(\boldsymbol{z}^{1:N}|\boldsymbol{c}) \approx p(\boldsymbol{z}^{1:N}|\boldsymbol{z}^{\boldsymbol{r}})\, p(\boldsymbol{z}^{\boldsymbol{r}}|\boldsymbol{c}) \approx p(\boldsymbol{z}^{\boldsymbol{r}}|\boldsymbol{c}) \prod_{i=1}^{N} p(z^i|\boldsymbol{z}^{\boldsymbol{r}})$$

where $\boldsymbol{z}^{\boldsymbol{r}}$ is latent frames as latent variables that abstract the global dependencies underlying the rendered video, such as visual references which describe the character and environment that supports consistent simulation. In this way, our paradigm avoids explicit modeling of long and redundant histories by reasoning about the underlying world implied by the script with compact latent visual frames, which in turn enables the independent rendering of progressive short segments conditioned on the reasoning frames and the script.

This *reasoning-based rendering* paradigm mirrors human creation: creators first establish a coherent world through concept art (e.g., character references, backgrounds, and storyboards), and then produce each clip independently, without repeatedly revisiting or memorizing how previous segments were rendered [Tang et al., 2025]. Notably, removing explicit long-history dependence allows training solely on short videos, **improving data efficiency of existing data to enable consistent long-video generation**. To improve temporal continuity, we could add K most recent latent frames as condition in practice. Formally, the complete modeling paradigm of TRR framework is given by:

$$p(\boldsymbol{z}^{1:N}|\boldsymbol{c}) \approx p(\boldsymbol{z}^{\boldsymbol{r}}|\boldsymbol{c}) \prod_{i=1}^{N} p(z^i|\boldsymbol{z}^{i-1:i-K}, \boldsymbol{z}^{\boldsymbol{r}}) \tag{4}$$

The equation above can be viewed as a (K-order) *conditional Markov simplification*: long-range identity, layout, and scene dependencies are absorbed into the reasoning tokens $\boldsymbol{z}^r$, so directly applying a Markov assumption no longer breaks long-term consistency, while fine-grained pixel-level continuity (e.g., fast motion under occlusion) is preserved by the K local context frames $\boldsymbol{z}^{i-1:i-K}$ in the rendering window. Our formulation in Eq. 4 involves two probability terms, corresponding to the two steps of the **TRR framework: Text-to-Reasoning and Reasoning-for-Rendering**, which are effectively parameterized within our TRR-based model below.

4.2 Plastic Memory

In this section, we present a concrete TRR model, *PlasticMem* (as shown in Fig. 3), and detail efficient parameterization methods for both $p(z^i|\boldsymbol{z}^{i-1:i-K}, \boldsymbol{z}^{\boldsymbol{r}})$ and $p(\boldsymbol{z}^{\boldsymbol{r}}|\boldsymbol{c})$.

Text-to-Reasoning (T2R, $p(\boldsymbol{z}^r \mid \boldsymbol{c})$). Video foundation models (VFMs) already map scripts to visual content, implying an implicit ability to reason about *what* should be generated from text. However, in existing models, such reasoning is tightly coupled with short-horizon rendering and thus limited in temporal scope. We explicitly decouple reasoning from rendering by shifting the generation target from rendered videos to intermediate, rendering-agnostic *reasoning frames* $\boldsymbol{z}^r$, which compactly describe the underlying world via sparse snapshots of stages and actors. This design encourages the model to focus on temporal reasoning over the underlying world, while remaining agnostic to rendering duration. For simplicity, we construct supervision from existing videos by randomly sampling non-consecutive latent frames from each shot or action segment and concatenating them into a reasoning frame sequence. Since per-shot/action annotations already group temporally coherent frames, this simple sampling yields reasonable and compact storyboards. More advanced selection



Figure 4: Qualitative Comparison for Visual Memory. Using the same streaming long video generation model, StreamDiT, we implement and compare two classical long-context modeling strategies for visual memory, context compression and context selection, along with our method. The results show that our temporal reasoning–based approach is more data-efficient under the same short-video training data and can generate consistent long videos even under large-scale motions.

strategies can be incorporated under the same formulation. All reasoning frames share a unified noise level and are trained using the standard flow-matching objective (as shown in Eq. 1).

Streaming Video Generation. After decoupling reasoning from video rendering, rendering textual dynamics becomes Markovian, as formalized in Eq. 4. We therefore adopt sliding-window Diffusion Forcing [Kodaira et al., 2025] for streaming long video generation. Given a denoising window of size B, Diffusion Forcing assigns structured noise levels $\boldsymbol{t} = [t_i, \ldots, t_{i+B}]$ to latent frames $\boldsymbol{z}^{i:i+B}$. At each step, the earliest frames are fully denoised and popped from the window, while newly initialized noisy frames are appended, enabling continuous rollout to arbitrary length. Training and inference follow Eq. 1 and Eq. 2; further details are in App. C and Alg. 2.

Reasoning-for-Rendering (R4R, $p(z^i|\boldsymbol{z}^{i-1:i-K}, \boldsymbol{z^r})$). We prepend N_1 reasoning frames at the start of the denoising window, using a chain-of-frames temporal reasoning approach to guide the model in generating the correct content for the current window based on the storyboard script. Under the Diffusion Forcing framework, we assign the reasoning tokens to the final timestep $t = 1$, allowing them to serve as contexts during denoising without being further denoised. To improve temporal continuity across sliding-window steps, we additionally concatenate the last N_2 fully denoised frames to the beginning of the current denoising window as local contexts, and likewise assign them to the final timestep. Formally the input sequence is:

$$\boldsymbol{z_t}^{R4R} = \boldsymbol{z_1^r} \parallel \boldsymbol{z_1}^{i-N_2:i-1} \parallel \boldsymbol{z_t}^{i:i+B} \tag{5}$$

where $\parallel$ represents direct frame concatenation.

Unified Training of Temporal Reasoning. Thanks to Diffusion Forcing framework's compatibility with different context types and noise schedules (including standard flow matching), we jointly train one VFM with T2R, R4R, and TI2V tasks, enabling knowledge and capability (e.g., text-following) transfer across tasks, leading to mutual gains and practical simplicity. Formally we have:

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{z},\boldsymbol{t},\boldsymbol{\epsilon}} \left[\| \boldsymbol{v}_\theta(\boldsymbol{z_t}^{task}, \boldsymbol{t}) - \boldsymbol{v_t}^{task} \| \right] \tag{6}$$

where $\boldsymbol{z_t}^{task}$ could be sampled from:

$$\boldsymbol{z_t}^{task} \sim \text{Uniform}(\boldsymbol{z_t}^{R4R}, \boldsymbol{z_t^r}, \boldsymbol{z_t}^{1:N}) \tag{7}$$

where $\boldsymbol{z_t}^{1:N}$ is inputs for TI2V, which applies the diffusion forcing noise schedule as in Sec. 3.2 to the whole short video sequence and $\boldsymbol{v_t^{task}}$ is the ground-truth velocity for tasks.



Figure 5: Qualitative Comparison with SOTA methods. We conducted a system-level comparison across different types of models. The results show that PlasticMem can maintain accurate visual memory for interactive objects and supports stable rollouts of up to 60 seconds.

4.3 PlasticRoPE

To distinguish different token types, support unified in-context multi-task training, and enable effective temporal reasoning, we introduce **PlasticRoPE**, a Temporal indexing strategy built on 3D Rotary Positional Embedding [Gao et al., 2025, Su et al., 2021]. The core idea is to partition the temporal axis into disjoint regions and assign different token types to fixed, non-overlapping temporal indices.

Temporal index partitioning. We redefine the temporal positional index τ in 3D RoPE via a type-aware mapping $\tau = f_{\text{type}}(i)$, where i denotes the original frame index and $f_{\text{type}}(\cdot)$ maps different token types to predefined temporal regions. Specifically, we partition the temporal axis into:

$$\tau_{\text{reason}} \in [0, 16], \tau_{\text{context}} \in [20, 25], \tau_{\text{video}} \in [33,\ 32 + N].$$

Reasoning tokens are assigned temporal indices in $\mathcal{R}_{\text{reason}}$, local context frames are placed in $\mathcal{R}_{\text{context}}$, and noisy video frames are mapped to $\mathcal{R}_{\text{video}}$ via a constant offset:

$$\tau_{\text{video}}(i) = i + 32, \quad i \in [1, N]. \tag{8}$$

This design separates reasoning, context, and rendering along the temporal axis (Fig. 3).

Task-specific indexing schemes. We unify T2R, TI2V, and R4R under a single task-specific temporal indexing scheme, summarized in Table 1. Each task maps its active token types to disjoint regions along the temporal axis, while leaving unused regions empty. This design enables unified multi-task training with a shared 3D RoPE formulation, while preserving clear semantic separation between reasoning, context, and video tokens.

R4R involves all three token types and generates progressive video segments conditioned on storyboard scripts and reasoning tokens. To explicitly encode generation progress, we

Task	τ_{reason}	τ_{context}	τ_{video}
T2R	$\{0, ..., 16\}$	–	–
TI2V	–	–	$\{32 + i\}_{i=1}^{N}$
R4R	$\{0{:}K\} \cup \{32, 33{+}N\}$	$\{20{:}25\}$	$\{32 + j\}_{j=i}^{i+B}$

Table 1: Task-specific temporal indexing schemes in PlasticRoPE. "–" indicates unused regions. For T2R, the first and last reasoning frames are assigned fixed indices at the boundaries of $\mathcal{R}_{\text{reason}}$, while intermediate reasoning frames are assigned random discontinuous indexes. This encourages the model to decouple non-contiguous reasoning from continuous rendering.

define the temporal indices of reasoning tokens as

$$\tau_{\text{reason}} = \begin{cases} 32, & \text{first reasoning frame,} \\ 33 + N, & \text{last reasoning frame,} \\ k, & k \in \mathcal{R}_{\text{reason}} \text{ for remaining frames.} \end{cases}$$

This aligns the reasoning tokens with the start and end of the full video region, while preserving a compact reasoning subspace. The current denoising window over video frames $[i,\ i + B]$ is mapped to the video region by:

$$\tau_{\text{video}}([i,\ i + B]) = [\, i + 32,\ i + B + 32\,].$$

Together, the reasoning tokens and the denoising window provide the model with an explicit cue of the relative position of the current segment within the global video timeline, enabling temporally-aligned and consistent generation conditioned on scripts and storyboards. Notably, this formulation allows the model to implicitly learn action durations from text, avoiding artifacts caused by manually predefined temporal boundaries such as incomplete or repeated actions.

Train-inference Close-Loop via Self-Generated Control. To improve temporal continuity, the most recent generated frame is placed at $\tau_{\text{last}} = i + 31$, adjacent to the current denoising window in τ_{video}, while other local context frames are sequentially assigned to τ_{context}. To avoid over-reliance on temporal index proximity and to make rendering robust against (and even self-correct) imperfect reasoning frames, we apply noise and deformation augmentations to all local context frames and a random subset of reasoning frames during training. Furthermore, to achieve a consistent train–inference closed loop, we train the model on self-rollout reasoning frames and local context frames. To keep the self-rollout aligned with the ground-truth training video and support standard flow-matching training, rather than sampling reasoning frames and local context frames from scratch, we perturb their ground-truth frames with a moderate level of noise as the starting point and resample them to obtain self-rollout frames. This on-policy self-rollout training loop enables the model to maximally adapt to the distribution of its own self-generated control conditions, allowing its reference-conditioned generation to outperform conditioning on third-party controls from unknown distributions. As an early prototype of an end-to-end agentic video generation system, it thus surpasses multi-stage generation pipelines pieced together step by step. Finally, PlasticRoPE generalizes beyond training-time length via RoPE extrapolation, since rendering relies on relative indices within each window.

5 Experiments

5.1 Experimental Setups

Datasets. We construct the consistent long video benchmark based on the switch-prompt dataset from LongLive [Yang et al., 2025a]. These prompts are further modified and expanded using LLMs to include larger-scale motions and camera movements, requiring visual recall of previously generated elements for more effective evaluation of visual memory. We train our model using short video data collected from the internet, with an average duration of approximately 5 seconds. Details are provided in the App. C.1.

Implementation Details. To align with the video modeling length supported by most VFMs, we set the maximum number of reasoning latent frames to 8, local context frames to 3, and noisy latent video frames to 30. In total, the input for single denoising step corresponds to approximately 7 seconds short video, which falls within the modeling capacity of most VFMs [Wan et al., 2025, Kodaira et al., 2025]. PlasticMem is trained for 10,000 steps on 24 A100 GPUs with a learning rate of 3e-5. We adopt our reimplementation of StreamDiT [Kodaira et al., 2025] as the pretrained VFM for streaming video generation and fine-tune the base DiT model across multiple GPUs using Fully Sharded Data Parallel (FSDP). More details are provided in the App. D.

Baselines. We compare PlasticMem with representative long video generation methods, including stable rollout approaches such as Self-Forcing [Huang et al., 2025a], Rolling-Forcing [Liu et al., 2025], and LongLive [Yang et al., 2025a]. We also include FramePack [Zhang et al., 2025a], which achieves long-term consistency via context compression, and MemFlow [Ji et al., 2025], which extends Self-Forcing with text-based context selection. In addition, we compare against our base model, StreamDiT [Kodaira et al., 2025], which adopts sliding-window diffusion forcing. For controlled comparisons, we implement context selection (based on prompts) and context compression (following MALT [Yu et al., 2025b]) baselines under the same base model and training settings. Details are provided in the App. E.

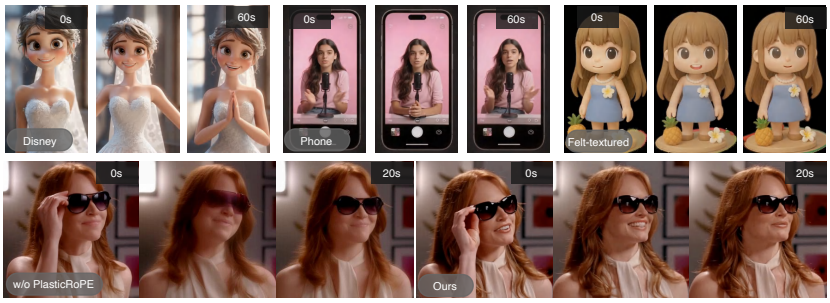


Figure 6: Ablation Study and More Results. The results demonstrate that PlasticRoPE stabilizes generation and supports Unified In-context Training and CoF reasoning, allowing the model to reason based on frame types and context rather than relying solely on frame index proximity. PlasticMem also preserves the diverse generation capabilities of the base model.

Method	Text Alignment↑	Consistency↑		Temporal↑			Frames↑	
		Background	Subject	Smoothness	Dynamic Degree	Flickering	Image Quality	Aesthetic Quality
Self-Forcing	21.58	94.05	94.77	99.18	35.89	98.58	73.24	58.79
Rolling-Forcing	23.75	94.72	95.88	98.85	**56.41**	97.91	73.32	57.57
LongLive	24.89	96.19	97.07	99.17	30.65	98.53	72.8	60.38
MemFlow	24.36	95.37	96.26	99.15	33.33	98.57	72.64	59.16
FramePack	21.79	96.29	96.85	**99.60**	17.95	**99.33**	73.16	61.11
StreamDiT	23.72	94.35	93.38	99.12	51.76	98.33	70.97	59.85
PlasticMem	**25.54**	**96.57**	**97.37**	99.24	39.75	98.68	**74.41**	**61.13**

Table 2: Quantitative Comparison. We evaluate consistency, visual and temporal quality using metrics from VBench Huang et al. [2024], and measure instruction following with CLIP similarity Radford et al. [2021]. The results show that PlasticMem achieves better long-video consistency and frame quality with competitive temporal quality, demonstrating the effectiveness of the TRR framework.

5.2 Performance Comparison of Visual Memory

Visual memory is critical for consistent long video generation, requiring models to accurately recall previously generated content. We evaluate memory by prompting large-scale back-and-forth motions involving both camera and subject movements, which stress prompt following and motion generation of the base video foundation model. To enable a direct comparison, we implement PlasticMem and two long-context baselines on the same StreamDiT backbone, trained with identical data and settings. Specifically, for the context compression method, we follow MALT, which recursively compresses history into a fixed-size memory vector via cross-attention. For the context selection method, we follow MemFlow, which selects relevant past frames based on text-frame correlation for in-context conditioning. As shown in Fig. 4, PlasticMem consistently recalls past content and maintains stable long-term rendering, whereas the baselines fail to fully reproduce the correct visual elements. More direct comparisons of long-term consistency are provided in App. A.

5.3 Consistent Long Video Generation

To evaluate consistent long video generation, we perform a system-level comparison against multiple state-of-the-art methods using both qualitative and quantitative experiments. Qualitatively, we assess whether models can retain newly introduced interactive elements. For MemFlow and LongLive, we follow their multi-prompt generation settings for better action control, while for PlasticMem and FramePack, we use concatenated single-storyboard scripts. FramePack is initialized from the same first frame as PlasticMem to prevent introducing future elements early and ensure fair comparison. The results (Fig. 5) show that PlasticMem accurately remembers newly introduced elements, whereas other methods exhibit varying degrees of distortion, with MemFlow and LongLive showing noticeable background drifting. Table 2 provides quantitative results on the designed benchmark.

5.4 Ablation Study

To illustrate the effect of TRR and PlasticRoPE in PlasticMem, we compare it with the base StreamDiT model and a version without PlasticRoPE using the standard 3D RoPE, where frames are indexed sequentially from 0. The results show that removing TRR (Fig. 1) significantly reduces the model's

memory ability, while removing PlasticRoPE (Fig. 6) makes the model overly reliant on nearby local context frames, leading to drift, although it can sometimes be corrected in subsequent generation, indicating the conditional independence of short video segments. We also provide additional videos in diverse styles (Fig. 6 and Fig. A.1) to demonstrate the generality of our method. To further examine PlasticMem beyond visual-memory focused prompts, we evaluate it on a standard multi-prompt benchmark with more diverse dynamics [Yang et al., 2025a]. More details and results of this evaluation, together with additional quantitative ablations, robustness analysis for imperfect reasoning frames, and computational-cost comparisons, are reported in Appendix A and C.1.

6 Conclusion

We propose Temporal Reasoning-based Rendering framework for consistent long video generation. It decomposes long video generation into reasoning about the underlying visual world and a sequence of independent short-video rendering steps, enabling training with only short-video data. We further introduce our framework implementation, *PlasticMem*, which decouples temporal reasoning from rendering in video foundation models and explicitly strengthens their temporal reasoning. PlasticMem supports efficient unified in-context training of temporal reasoning and dynamic rendering with PlasticRoPE. Remaining limitations on reasoning-frame quality and under-explored design space of underlying world description formats are detailed in App. F.

Acknowledgments and Disclosure of Funding

This work was supported in part by the New Generation Artificial Intelligence-National Science and Technology Major Project (No. 2025ZD0123501), Open Research Fund of the State Key Laboratory of Multimodal Artificial Intelligence Systems (No.MAIS2025064), and InnoHK Program.

References

Shengqu Cai, Ceyuan Yang, Lvmin Zhang, Yuwei Guo, Junfei Xiao, Ziyan Yang, Yinghao Xu, Zhenheng Yang, Alan Yuille, Leonidas Guibas, Maneesh Agrawala, Lu Jiang, and Gordon Wetzstein. Mixture of contexts for long video generation. In *arXiv*, 2025.

Boyuan Chen, Diego Marti Monso, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *ArXiv*, abs/2407.01392, 2024. URL `https://api.semanticscholar.org/CorpusID:270869622`.

Guibin Chen, Dixuan Lin, Jiangping Yang, Chunze Lin, Junchen Zhu, Mingyuan Fan, Hao Zhang, Sheng Chen, Zheng Chen, Chengcheng Ma, Weiming Xiong, Wei Wang, Nuo Pang, Kang Kang, Zhiheng Xu, Yuzhe Jin, Yupeng Liang, Yubing Song, Peng Zhao, Boyuan Xu, Di Qiu, Debang Li, Zhengcong Fei, Yang Li, and Yahui Zhou. Skyreels-v2: Infinite-length film generative model, 2025a. URL `https://arxiv.org/abs/2504.13074`.

Harold Haodong Chen, Disen Lan, Wen-Jie Shu, Qingyang Liu, Zihan Wang, Sirui Chen, Wenkai Cheng, Kanghao Chen, Hongfei Zhang, Zixin Zhang, et al. Tivibench: Benchmarking think-in-video reasoning for video generative models. *arXiv preprint arXiv:2511.13704*, 2025b.

Justin Cui, Jie Wu, Ming Li, Tao Yang, Xiaojie Li, Rui Wang, Andrew Bai, Yuanhao Ban, and Cho-Jui Hsieh. Self-forcing++: Towards minute-scale high-quality video generation. *arXiv preprint arXiv:2510.02283*, 2025.

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*, pages 17702–17711, 2025.

Yu Gao, Haoyuan Guo, Tuyen Hoang, Weilin Huang, Lu Jiang, Fangyuan Kong, Huixia Li, Jiashi Li, Liang Li, Xiaojie Li, et al. Seedance 1.0: Exploring the boundaries of video generation models. *arXiv preprint arXiv:2506.09113*, 2025.

Yuchao Gu, weijia Mao, and Mike Zheng Shou. Long-context autoregressive video modeling with next-frame prediction. *arXiv preprint arXiv:2503.19325*, 2025.

Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan 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*, 2025.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *arXiv preprint arXiv:2506.08009*, 2025a.

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*, 2024.

Ziqi Huang, Ning Yu, Gordon Chen, Haonan Qiu, Paul Debevec, and Ziwei Liu. Vchain: Chain-of-visual-thought for reasoning in video generation. *arXiv preprint arXiv:2510.05094*, 2025b.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Sihui Ji, Xi Chen, Shuai Yang, Xin Tao, Pengfei Wan, and Hengshuang Zhao. Memflow: Flowing adaptive memory for consistent and efficient long video narratives, 2025. URL `https://arxiv.org/abs/2512.14699`.

Jihwan Kim, Junoh Kang, Jinyoung Choi, and Bohyung Han. Fifo-diffusion: Generating infinite videos from text without training. In *NeurIPS*, 2024.

Akio Kodaira, Tingbo Hou, Ji Hou, Masayoshi Tomizuka, and Yue Zhao. Streamdit: Real-time streaming text-to-video generation, 2025. URL `https://arxiv.org/abs/2507.03745`.

Runjia Li, Philip Torr, Andrea Vedaldi, and Tomas Jakab. Vmem: Consistent interactive video scene generation with surfel-indexed view memory. *arXiv preprint arXiv:2506.18903*, 2025.

Long Lian, Baifeng Shi, Adam Yala, Trevor Darrell, and Boyi Li. Llm-grounded video diffusion models. *arXiv preprint arXiv:2309.17444*, 2023.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=PqvMRDCJT9t`.

Kunhao Liu, Wenbo Hu, Jiale Xu, Ying Shan, and Shijian Lu. Rolling forcing: Autoregressive long video diffusion in real time. *arXiv preprint arXiv:2509.25161*, 2025.

William Peebles and Saining Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

Xuanchi Ren, Tianchang Shen, Jiahui Huang, Huan Ling, Yifan Lu, Merlin Nimier-David, Thomas Müller, Alexander Keller, Sanja Fidler, and Jun Gao. Gen3c: 3d-informed world-consistent video generation with precise camera control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Nedko Savov, Naser Kazemi, Deheng Zhang, Danda Pani Paudel, Xi Wang, and Luc Van Gool. StateSpaceDiffuser: Bringing Long Context to Diffusion World Models. In *Advances in Neural Information Processing Systems*, 2025.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2021.

Yunlong Tang, Junjia Guo, Pinxin Liu, Zhiyuan Wang, Hang Hua, Jia-Xing Zhong, Yunzhong Xiao, Chao Huang, Luchuan Song, Susan Liang, Yizhi Song, Liu He, Jing Bi, Mingqian Feng, Xinyang Li, Zeliang Zhang, and Chenliang Xu. Generative ai for cel-animation: A survey. *arXiv preprint arXiv:2501.06250*, 2025. URL https://arxiv.org/abs/2501.06250.

Hansi Teng, Hongyu Jia, Lei Sun, Lingzhi Li, Maolin Li, Mingqiu Tang, Shuai Han, Tianning Zhang, W. Q. Zhang, Weifeng Luo, Xiaoyang Kang, Yuchen Sun, Yue Cao, Yunpeng Huang, Yutong Lin, Yuxin Fang, Zewei Tao, Zheng Zhang, Zhongshu Wang, Zixun Liu, Dai Shi, Guoli Su, Hanwen Sun, Hong Pan, Jie Wang, Jiexin Sheng, Min Cui, Min Hu, Ming Yan, Shucheng Yin, Siran Zhang, Tingting Liu, Xianping Yin, Xiaoyu Yang, Xin Song, Xuan Hu, Yankai Zhang, and Yuqiao Li. Magi-1: Autoregressive video generation at scale, 2025. URL https://arxiv.org/abs/2505.13211.

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.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Thaddäus Wiedemer, Yuxuan Li, Paul Vicol, Shixiang Shane Gu, Nick Matarese, Kevin Swersky, Been Kim, Priyank Jaini, and Robert Geirhos. Video models are zero-shot learners and reasoners. *arXiv preprint arXiv:2509.20328*, 2025.

Zeqi Xiao, Yushi Lan, Yifan Zhou, Wenqi Ouyang, Shuai Yang, Yanhong Zeng, and Xingang Pan. Worldmem: Long-term consistent world simulation with memory, 2025. URL https://arxiv.org/abs/2504.12369.

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

Xiangpeng Yang, Ji Xie, Yiyuan Yang, Yan Huang, Min Xu, and Qiang Wu. Unified video editing with temporal reasoner. *arXiv preprint arXiv:2512.07469*, 2025b.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T Freeman. Improved distribution matching distillation for fast image synthesis. In *NeurIPS*, 2024a.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *CVPR*, 2024b.

Tianwei Yin, Qiang Zhang, Richard Zhang, William T Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast autoregressive video diffusion models. In *CVPR*, 2025.

Jiwen Yu, Jianhong Bai, Yiran Qin, Quande Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Xihui Liu. Context as memory: Scene-consistent interactive long video generation with memory retrieval. *arXiv preprint arXiv:2506.03141*, 2025a.

Sihyun Yu, Meera Hahn, Dan Kondratyuk, Jinwoo Shin, Agrim Gupta, José Lezama, Irfan Essa, David Ross, and Jonathan Huang. Malt diffusion: Memory-augmented latent transformers for any-length video generation. *arXiv preprint arXiv:2502.12632*, 2025b.

Lvmin Zhang, Shengqu Cai, Muyang Li, Gordon Wetzstein, and Maneesh Agrawala. Frame context packing and drift prevention in next-frame-prediction video diffusion models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025a.

Tianyuan Zhang, Sai Bi, Yicong Hong, Kai Zhang, Fujun Luan, Songlin Yang, Kalyan Sunkavalli, William T Freeman, and Hao Tan. Test-time training done right. *arXiv preprint arXiv:2505.23884*, 2025b.

Appendix Contents

Appendix A More Experimental Results **14**

A.1 Hyperparameter Ablations 14

A.2 Component Ablations 14

A.3 Direct Human Evaluation of Visual Memory 15

A.4 Evaluation Beyond Visual-Memory Focused Prompts 15

A.5 Computational Cost 15

A.6 More General Videos 16

Appendix B More Backgrounds and Related Works **16**

Appendix C Consistent Streaming Video Generation **18**

C.1 Training Details 18

C.2 Inference Details 21

C.3 Initialization and Termination for Diffusion Forcing 21

Appendix D Reimplementation Details of StreamDiT **22**

Appendix E Experiment Setups **22**

Appendix F Limitations and Future Work **24**

Appendix G Broader Impact **24**

A More Experimental Results

Metric	Reasoning Frame Num			Context Frame Num			Sliding Window Size		
	6	8	10	0	1	3	24	30	36
Subject Consistency	96.38	96.97	96.89	96.45	96.88	96.97	96.67	96.97	96.85
Temporal Flickering	98.49	98.68	98.92	96.78	98.12	98.68	98.33	98.68	98.92
Image Quality	74.39	74.41	74.12	74.41	74.43	74.41	74.37	74.41	73.96
Aesth. Quality	61.34	61.13	61.22	61.07	61.01	61.13	61.26	61.13	60.98

Table A.1: Ablation Study. We compare performance under different hyperparameter settings. The default setting uses 8 reasoning frames, 3 context frames, and a sliding window size of 30. When varying one hyperparameter, all others are kept fixed to the default setting.

A.1 Hyperparameter Ablations

Table A.1 analyzes the impact of key hyperparameters in PlasticMem, including the number of reasoning frames, the number of local context frames, and the sliding window size. Overall, the results show that PlasticMem is robust to moderate variations of these hyperparameters, while exhibiting consistent trends aligned with their intended roles.

Reasoning Frame Num. Varying the number of reasoning frames primarily affects temporal stability and consistency. Increasing the number of reasoning frames from 6 to 8 improves subject consistency and reduces temporal flickering, indicating that richer reasoning tokens provide more stable visual anchors for long-horizon generation. Further increasing to 10 yields diminishing returns, with marginal gains in flickering but no improvement in image or aesthetic quality. This suggests that a moderate number of reasoning frames is sufficient to capture the underlying world structure, while excessive reasoning tokens introduce redundancy without additional benefit.

Context Frame Num. Local context frames mainly contribute to short-range temporal continuity. As the number of context frames increases from 0 to 3, both subject consistency and temporal flickering improve consistently, confirming that recent high-quality frames help stabilize progressive rendering. In contrast, image and aesthetic quality remain largely unchanged, indicating that local context primarily affects temporal coherence rather than per-frame visual fidelity. These results support our design choice of using a small number of local context frames to complement reasoning tokens without relying on long raw histories.

Sliding Window Size. The sliding window size controls the temporal span of diffusion forcing during streaming generation. A moderate window size (30) achieves the best overall balance across metrics. Smaller windows limit the temporal receptive field and slightly degrade consistency, while larger windows introduce longer-range dependencies that marginally reduce image and aesthetic quality, likely due to increased denoising difficulty and accumulated uncertainty. This observation is consistent with our Markovian rendering design, where reasoning tokens, rather than an excessively large denoising window, are responsible for maintaining long-term structure.

In summary, the ablation results validate the intended roles of each component: reasoning frames provide stable long-term anchors, context frames enhance local temporal smoothness, and a moderate sliding window size enables efficient and stable rollout. The default configuration offers a balanced trade-off across consistency, temporal stability, and visual quality.

A.2 Component Ablations

Beyond hyperparameter ablations, we further isolate the contributions of PlasticRoPE and unified training under the same backbone. PlasticRoPE assigns different token types to separated temporal regions, allowing the model to distinguish reasoning, context, and video tokens under one shared RoPE formulation. Unified training jointly optimizes T2V, TI2V, T2R, and R4R tasks under the same frame-level flow-matching objective, enabling knowledge transfer across standard video generation, temporal reasoning, and rendering. The ablation in Table A.2 shows that both components improve consistency and visual quality, with PlasticRoPE contributing most strongly to the separation of heterogeneous token types.

Method	Consistency Score	Quality Score	Success Rate (%)
PlasticMem	**96.97**	**67.77**	**81**
w/o PlasticRoPE	93.12	61.38	74
w/o Unified Training	95.56	64.85	78

Table A.2: Ablation of PlasticRoPE and unified training under the same backbone. The Success Rate (%) is reported under the human evaluation protocol described in Appendix A.3.

Method	LongLive	MemFlow	PlasticMem
Success Rate (%)	47	65	**81**

Table A.3: Human evaluation of long-term visual memory. Evaluators judge whether each generated video remains consistent throughout the full sequence, following the protocol in Appendix A.3.

A.3 Direct Human Evaluation of Visual Memory

Video memory is an under-explored problem, and designing a metric that directly measures whether a disappeared object, subject, or scene is faithfully recalled after a long temporal gap remains challenging. Existing long-video memory methods therefore still rely heavily on qualitative evaluation. To provide a more direct quantitative signal, we additionally recruit human evaluators to judge whether each generated video maintains consistency throughout the full rollout. A video is counted as successful if no clear identity, subject, or scene inconsistency is observed during long-horizon generation.

Human evaluation protocol. We design a controlled human study so that the Success Rate (%) reported across our experiments is computed under a unified protocol. We randomly select 25 prompts from our long-video benchmark. For each prompt, every model generates four 60-second videos using independent random seeds, yielding 100 videos per model and 300 videos in total across the three compared models. All videos are anonymized (with model identities hidden) and randomly distributed to 10 human evaluators, with each video reviewed independently. Each evaluator marks every assigned video as either *consistent* (no noticeable identity, subject, or scene inconsistency throughout the full rollout) or *inconsistent*. The Success Rate of a model is then computed as the fraction of its 100 videos labeled as consistent. The same protocol is reused for every Success Rate column in the appendix tables.

As shown in Table A.3, PlasticMem achieves a higher consistency success rate than representative long-video baselines under this unified protocol.

A.4 Evaluation Beyond Visual-Memory Focused Prompts

Many visual-memory prompts intentionally contain subjects recalls, scene revisits, or back-and-forth movements because these controlled cases reliably expose long-range recall failures. This setting isolates visual memory from other confounding factors such as text understanding, motion generation, and scene planning. However, to examine behavior beyond visual memory focused prompts, we also evaluate PlasticMem in a standard multi-prompt long-video generation setting with more diverse dynamics. For PlasticMem, we first compress multiple prompts into a short reasoning prompt that covers the main subjects and scenes. During rendering, each shot keeps its corresponding detailed prompt, while all shots share the same reasoning frames and use the last frames of the previous shot as local context for continuity. The Success Rate column in Table A.4 follows the same human evaluation protocol as in Appendix A.3. The results show that PlasticMem maintains strong consistency while also achieving competitive dynamic degree and visual quality.

A.5 Computational Cost

Reasoning frames are generated only once and in small numbers, so they add limited overhead relative to the base video model. During rendering, PlasticMem avoids attending to an ever-growing raw history. Under the conditional Markov assumption in Eq. 4, each rendering step conditions on

Method	Consistency Score	Dynamic Degree	Quality Score	Success Rate (%)
LongLive	91.54	62.93	59.89	71
MemFlow	95.27	50.18	58.13	85
PlasticMem	**97.36**	**68.85**	**63.26**	**93**

Table A.4: Evaluation in a standard multi-prompt long-video generation setting with more diverse dynamics. The Success Rate (%) measures whether human evaluators observe consistency failures, following the protocol in Appendix A.3.

Method	GPU Memory	Throughput (FPS)
LongLive	∼50 GB	2.54
MemFlow	∼60 GB	2.17
PlasticMem	∼60 GB	2.33

Table A.5: Computational cost comparison under the same 8-step inference setting.

compact reasoning frames and a small local context, keeping the per-step input size nearly constant as video length increases. Therefore, the total inference cost scales approximately linearly with generated video length rather than quadratically with the accumulated history. Table A.5 reports representative GPU usage and throughput under 8 diffusion steps.

A.6 More General Videos

We provide more video cases in Figure A.1, covering animals, fantasy creatures, robots, outdoor scenes, and imaginary scenarios to demonstrate the generality of our method. More video cases can be found in the supplementary material.

B More Backgrounds and Related Works

Democratizing Visual Storytelling with Video Generation. Visual storytelling has played a central role in human expression [Tang et al., 2025], spanning professional productions such as films and animation as well as a wide range of user-generated content, and conveying emotions and



Figure A.1: More video cases across diverse categories.

ideas across time and space. However, this magic of light and motion comes at a substantial cost. High-quality video creation typically demands industrial-scale resources, resulting in a high barrier to entry. Consequently, video generation has attracted growing interest as a means to democratize high-quality video creation, enabling creators to focus primarily on expression rather than production overhead [Wan et al., 2025, Chen et al., 2025a]. Recent progress in text- or image-conditioned video generation has enabled high-quality short clips from textual prompts or a single reference frame. However, generation is typically limited to under ten seconds, which is insufficient for visual storytelling [Chen et al., 2024, 2025a, Yin et al., 2025, Teng et al., 2025].

Works for memory in video diffusion models. Video storytelling beyond a few seconds often requires scene revisit or subject recall, making visual memory essential for long-term consistency [Cai et al., 2025, Ji et al., 2025]. Existing approaches mainly rely on retrieval or compression of history. Retrieval-based methods leverage history lookup to maintain consistency during scene revisits. For example, WorldMem [Xiao et al., 2025] and Context-as-Memory [Yu et al., 2025a] use FoV-based history retrieval to assist autoregressive generation, while Mixture-of-Contexts [Cai et al., 2025] introduces learnable sparse attention for selecting relevant past contexts, and MemFlow [Ji et al., 2025] uses text prompts as indices to retrieve required historical frames. Compression-based approaches instead summarize unbounded history into compact representations. GEN3C [Ren et al., 2025] and Vmem [Li et al., 2025] compress past frames into explicit 3D scene representations, while TTTVideo [Dalal et al., 2025] and LaCT [Zhang et al., 2025b] encode history into test-time–trained network parameters. FramePack [Zhang et al., 2025a] and FAR [Gu et al., 2025] compress history by allocating different token budgets to near and distant frames, whereas MALT [Yu et al., 2025b] and StateSpaceDiffuser [Savov et al., 2025] further abstract prior contexts into fixed-size hidden states. However, both paradigms rely on high-quality long video data and therefore work better in simulation than on real videos, where scene revisit and subject recall are rare. To address this, we reverse the order of generation and memory: instead of retrieving or compressing long histories, we first generate compact, text-aligned reasoning frames and condition autoregressive rendering on them, enabling consistent long video generation from short-video data.

Reasoning with Video Foundation Models. The notion of reasoning has been widely studied in large language models, where techniques such as Chain-of-Thought prompting [Wei et al., 2022] encourage step-by-step inference to solve complex tasks. Recently, similar reasoning behaviors have been observed to emerge implicitly in video foundation models (VFMs) in a Chain-of-Frame style [Wiedemer et al., 2025]. Since VFMs are trained to understand textual descriptions and render temporally coherent visual content, they internally reason about what to generate at each step based on the input text and prior frames. This is supported by recent work showing that VFMs can be directly applied to visual reasoning tasks [Chen et al., 2025b, Yang et al., 2025b], such as maze solving and temporal logic inference. In these tasks, visual rendering itself is often trivial, as most pixels remain unchanged, while the core challenge lies in inferring intermediate solution steps from textual instructions to guide how the scene should evolve over time. The success of VFMs in such settings suggests that, beyond modeling visual fidelity and temporal continuity, large video generation models also exhibit latent temporal reasoning capabilities.

Long Video Generation. The challenges of long video generation can be broadly categorized into three aspects: inference and training efficiency, stable long-term rollout, and high-quality content generation with complex motions and long-range consistency. To improve efficiency, a large body of video generation work explores architectural and inference-level optimizations, including linear or sparse attention [Cai et al., 2025, Savov et al., 2025], spatial-temporal decomposition [Gao et al., 2025] and Video Extension [Teng et al., 2025, Chen et al., 2025a]. In parallel, addressing instability in long-horizon rollout has motivated methods that explicitly mitigate exposure bias and error accumulation, most notably Self-Forcing [Huang et al., 2025a, Cui et al., 2025, Yang et al., 2025a, Liu et al., 2025], along with related efforts combining autoregressive and diffusion model long-sequence stabilization [Chen et al., 2024]. However, generating higher-quality long videos—particularly those with complex motion dynamics and strong long-term visual consistency—remains comparatively under-explored due to its intrinsic difficulty. Recently, inspired by reasoning paradigms in large language models, several works attempt to tackle complex video generation by leveraging reasoning primarily in the textual domain. Representative approaches extend the user prompt into structured or staged descriptions, such as scripts or storyboards, enabling complex videos to be generated by extra controls [Huang et al., 2025b, Lian et al., 2023]. In this work, we explore an orthogonal paradigm

and introduce reasoning for consistent long video generation. Instead of relying on reasoning in MLLMs, we directly leverage the reasoning capability of video foundation models themselves to perform visual-native reasoning and generative control for long-term consistency.

C Consistent Streaming Video Generation

Text-to-Reasoning. Since video foundation models (VFMs) already understand scripts and render corresponding visual content, we hypothesize that they implicitly possess a form of reasoning ability: the model must reason about what to generate based on the text to render videos. This hypothesis is supported by recent work that applies VFMs to visual reasoning tasks [Wiedemer et al., 2025]. However, in current video generation models, the reasoning capability remains tightly coupled with short-horizon video rendering and is therefore limited to short temporal ranges. Thus, we decouple reasoning and rendering to explicitly elicit the VFM's temporal reasoning ability by shifting the generation target from final rendered short videos to intermediate, rendering-agnostic reasoning frames, which compactly describe the underlying world with snapshots of the stages and actors. This enables the model to focus on reasoning the underlying world from texts, while supporting dynamic rendering at arbitrary durations. This is akin to preparing the stages and actors (the underlying world) before filming scenes of arbitrary length (video rendering). For proof of concept and simplicity, we directly construct reasoning frames from existing videos: we randomly sample non-consecutive latent frames from each shot or action segment in one video, and concatenate them to form a visual storyboard for the entire video. We then train text-to-reasoning (visual storyboard generation) using the standard flow-matching loss (Eq. 1) with all reasoning tokens share the same noise level.

Streaming Video Rendering. For a video diffusion model with denoising window size B, we allow a non-decreasing sequence of noise levels $\boldsymbol{t} = [t_i, ..., t_{i+B}]$ for latent frames $\boldsymbol{z}^{i:i+B}$ within the denoising window. Specifically, we partition the latent frames in the buffer into n chunks, each containing c latent frames, such that $B = n \cdot c$. Latent frames within the same chunk share an identical noise level and are generated synchronously under the same schedule, which improves local temporal continuity. The noise levels across chunks form a monotonically increasing arithmetic sequence, spanning from the lowest to the highest noise level. The total number of denoising steps is thus $n \cdot d$, where d denotes the number of denoising steps required for a single sliding-window step and corresponds to the common difference between noise levels of adjacent chunks. This design ensures that, during long video generation with a sliding window, executing d denoising steps in each window step brings the first chunk to the final denoising stage. The window can then advance by popping the first chunk from the buffer and pushing a new full noise chunk to the end. In this manner, we enable streaming generation of long video sequences. The training and inference follows the standard practice of Diffusion Forcing as in Eq. 1 and Eq. 2.

C.1 Training Details

PlasticRoPE and Augmentation for unified training We modified the temporal positional index within the 3D Rotary Position Embedding (3D RoPE), where different types of tokens are assigned with distinct regions along the temporal axis.

Specifically, reasoning frames are assigned temporal positional indices starting from 0 and occupy the range 0–16. Similarly, local context frames are assigned indices 20–25. Noisy video frames are offset by 32, shifting their original indices from $1:N$ to $33:N+32$, as shown in Fig. 3. For T2R, the first and last reasoning frames are assigned with indices 0 and 16. Intermediate reasoning frames are placed with random index gaps, encouraging the model to decouple continuous rendering from non-contiguous temporal reasoning. For TI2V, the temporal index of each video frame is set to its original frame index plus an offset of 32. The two tasks above involve only a single frame (token) type, which is placed in its corresponding region, while the other regions remain empty. In contrast, R4R involves all three frame types and generates a segment of the original video. To indicate the current generation progress and align generation with the storyboard scripts, we assign the first and last reasoning frames to the start and end indices of the video region (32 and N+33), while assigning the remaining reasoning frames consecutive indices starting from 0. The current denoising window, corresponding to video frames $i:i+B$, is likewise shifted by 32 and mapped to the noisy video region.

Algorithm 1 Unified Training of PlasticMem with Temporal-Aware PlasticRoPE

Require: Training batch $\mathcal{B}$, VAE encoder $\mathcal{E}$, DiT denoiser $\mathcal{D}$

for each training iteration **do**
  for each sample $(\boldsymbol{x}, \boldsymbol{c}) \in \mathcal{B}$ **do**
    Sample task task $\in \{\text{T2V}, \text{IT2V}, \text{T2R}, \text{IT2R}, \text{R4R}\}$
    Encode video latents $\mathbf{z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_N\} \leftarrow \mathcal{E}(x)$
    if task $\in \{\text{T2V}, \text{IT2V}\}$ **then**
      Sample chunk size $s \in \{4, 5, 6, 7, 8\}$
      Partition $\mathbf{z}$ into chunks $\mathcal{C}_k$ with chunk index k
      Add structured noise ϵ_k with $\epsilon_{k+1} \geq \epsilon_k$
    else if task $\in \{\text{T2R}, \text{IT2R}\}$ **then**
      Sample one frame per action or shot segment to form $\mathbf{r}$
      Add equal-level noise to all reasoning frames
    else
      Sample $N_1 \in \{4, 5, 6, 7, 8\},\ N_2 \in \{1, 2, 3\}$
      Select N_1 reasoning frames $\boldsymbol{z^r}$ as in temporal reasoning
      Sample contiguous video segment $\boldsymbol{z}^{i:i+B}$ with $|\mathbf{v}| \in [N/2, N - N_1]$
      Select local context frames $\mathbf{z}^{i-N_2:i-1}$ as the first N_2 frames of $\mathbf{v}$
      Add structured noise only to $\boldsymbol{z}^{i:i+B}$
      Concatenate input $\mathbf{z}_{\text{in}} = [\boldsymbol{z^r}_1; \boldsymbol{z}_1^{i-N_2:i-1}; \boldsymbol{z_t}^{i:i+B}]$
    end if
    Apply PlasticRoPE to obtain temporal index $\boldsymbol{\tau}$ as in Table 1.
      $\theta_i = \alpha_k \cdot \tau_i$
      $(\mathbf{q}_i, \mathbf{k}_i) \leftarrow \text{RoPE}(\mathbf{q}_i, \mathbf{k}_i; \theta_i)$
    $\hat{\mathbf{z}} \leftarrow \mathcal{D}(\mathbf{z}_{\text{in}}, c)$
    Compute frame-level flow-matching loss $\mathcal{L}_{\text{DF}}$ in Eq. 1
  end for
end for

Notably, although the input to the model consists of absolute temporal indices, the model primarily reasons over relative positional information, such as index differences within the same denoising window or ratios between distances to the start and end indices. As a result, even when trained on short videos, the model can generalize to longer videos to a certain extent. More advanced techniques for RoPE extrapolation could be further explored to improve this behavior, which we leave as future work. This indexing scheme allows the model to infer the relative position of the current segment within the full video and, together with the reasoning tokens and text, generate temporally consistent content. Moreover, it enables the model to implicitly learn action durations from text, avoiding artifacts caused by manually predefined temporal boundaries, such as incomplete or repeated actions.

In addition, to improve temporal continuity, we place the last generated frames in the noisy video region with indices contiguous to the current noisy video segment, while assigning the remaining local context frames sequentially to the range 20–25. To prevent the model from relying on temporal index proximity for attention and to encourage it to select the most relevant high-quality reasoning tokens at inference time, we apply noise and deformation augmentations to all local context frames and to a random subset of reasoning frames. The rationale of these augmentations is twofold: they reduce the risk of direct copying from the reasoning frames or local contexts, and they encourage the model to focus on semantic subject and scene information rather than raw pixels, which makes rendering more robust to imperfect reasoning frames.

For summary, we unify T2R, TI2V, and R4R under a single task-specific temporal indexing scheme, summarized in Table 1. Each task maps its active token types to disjoint regions along the temporal axis, while leaving unused regions empty. This design enables unified multi-task training with a shared 3D RoPE formulation, while preserving clear semantic separation between reasoning, context, and video tokens. For R4R, we explicitly align temporal reasoning with generation progress. Let the full video length be N and the current denoising window be $[i,\ i + B]$. Video frames are indexed as

$$\tau_{\text{video}}(i) = i + 32, \quad i \in [1,\ N], \tag{1}$$

The active denoising window is therefore mapped to

$$[i + 32,\ i + B + 32]. \tag{2}$$

We assign the first and last reasoning frames to indices 32 and $33 + N$, respectively, anchoring reasoning tokens to the global video timeline, while intermediate reasoning frames are placed in τ_{reason} to encode high-level temporal structure. To improve temporal continuity, recently generated frames are placed adjacent to the current denoising window in τ_{video}, while local context frames are sequentially assigned to τ_{context}. To avoid over-reliance on temporal index proximity, we apply noise and deformation augmentations to all local context frames and to a random subset of reasoning frames during training.

Data Preparation We use approximately 700k in-house video clips with an average duration of about 5 seconds (roughly 1,000 hours in total). To obtain fine-grained textual supervision, we manually annotate a subset of the data at the level of individual actions and shot transitions. These high-quality annotations are used to train Qwen2.5-VL as a video understanding and annotation model. The trained model is then applied to the full dataset to generate detailed video-paired textual descriptions, explicitly covering multiple actions and shot changes within each clip. This semi-automatic pipeline enables scalable annotation while maintaining consistency with the manually curated annotations.

Unified Training. Although video generation, rendering and reasoning tasks differ in their input contexts and supervision signals, they are all formulated under the same frame-level flow-matching objective. As a result, these tasks can be unified under a single training objective and jointly optimized within the same batch, with task-specific behavior emerging solely from differences in the provided contexts. In each training iteration, for every sample in the batch, we randomly select one task from five candidates: text-to-video, image–text-to-video, text-to-reasoning, text–image-to-reasoning, and reasoning-for-rendering. Depending on the selected task, we apply task-specific processing to the VAE-encoded video latent frames. We summarize the training algorithm in Alg. 1.

For video generation tasks, we directly apply structured noise to the entire latent video sequence, where the noise level monotonically increases at the chunk level. The chunk size is randomly sampled from $[4, 5, 6, 7, 8]$, and all frames within the same chunk are perturbed with the same noise magnitude.

For temporal reasoning tasks, we randomly sample one frame from each action or shot segment and concatenate these frames to form a set of reasoning frames. We then apply flow-matching-based denoising training by adding noise with the same magnitude to all reasoning frames, following the standard flow matching formulation. For training temporal reasoning, we use video prompts that are identical to the video captions provided in the dataset, as video captions typically cover all major content that appears in the video. This design choice allows us to directly reuse existing annotations from standard TI2V training data without requiring additional data collection.

In principle, temporal reasoning does not have to rely on the same textual annotations as video rendering. More specialized reasoning prompts could be designed to better probe and describe the underlying world implied by the video captions, such as explicit multi-view character descriptions for character consistency or multi-view environment descriptions for scene consistency. However, such approaches would require substantially denser annotations of video subjects and reasoning prompts, which, while potentially unlocking additional training potential from existing data, are orthogonal to the focus of this work and left for future exploration.

For the Reasoning-for-Rendering task, we reuse the frames selected in the temporal reasoning stage as the reasoning frames. In addition, we randomly sample a contiguous video segment of length $N \in [L/2,\, L - N_1]$ from the video latent frames, where N denotes the total number of video latent frames and N_1 is the number of reasoning frames. From this segment, we further select its first N_2 frames as local context frames, where $N_1 \in [4, 5, 6, 7, 8]$ and $N_2 \in [1, 2, 3]$. During training, reasoning frames that overlap with the continuous video segments used for rendering supervision are excluded, in order to prevent the model from learning trivial copying behaviors from the reasoning frames. The reasoning frames, local context frames, and the selected video segment are then concatenated as the model input. We apply the same structured noise as in the video generation task to the video segment, while fixing the noise levels of both the reasoning frames and the local context frames to zero. The concatenated sequence is finally fed into the DiT model for denoising training.

Algorithm 2 PlasticMem AR Inference with Temporal Reasoning-based Rendering

Require: Noises $\boldsymbol{z}_0 = \{\epsilon^i\}_{i=0}^{N-1}$, conditions, the DiT model. text and image conditions $\boldsymbol{c}^+, \boldsymbol{c}^-$
Ensure: Video V

Build noise levels $\boldsymbol{t}_{\text{beg}}, \boldsymbol{t}_{\text{mid}}, \boldsymbol{t}_{\text{end}} \subset \{t_0, \ldots, t_M\}^B,\ t_0 = 0, t_M = 1$
Text embeddings $c_{txt}^+ = \text{TextEnc}(c_{txt}^+),\ c_{txt}^- = \text{TextEnc}(c_{\text{neg-txt}}^-)$
Define CFG:
  $f_{\text{cfg}}(\boldsymbol{z}_{in}) = f^-(\boldsymbol{z}_{in}) + \gamma\,(f^+(\boldsymbol{z}_{in}) - f^-(\boldsymbol{z}_{in}))$
  $f^+(\boldsymbol{z}_{in}) = \text{DiT}([\boldsymbol{z}_{in}; \boldsymbol{c}^+]), \quad f^-(\boldsymbol{z}_{in}) = \text{DiT}([\boldsymbol{z}_{in}; \boldsymbol{c}^-])$
Set noisy reasoning frames $\boldsymbol{r} = \{r_i\}_{i=1}^{N_1}$
Full-diffusion reasoning on reasoning frames:
  $\boldsymbol{z}^{\boldsymbol{r}} = \text{Sampler}(\boldsymbol{z}^{\boldsymbol{r}}, \boldsymbol{t}_{\text{FM}}, f_{\text{cfg}})$
Initialize global buffer $\boldsymbol{z} \leftarrow \boldsymbol{z}_0^{0:N-1}$.
Initialize noisy video latents $\boldsymbol{z_t}^{0:B-1}$ within the first denoising window as in section C.3.
for each AR window w **do**
  Sample context indices $\mathcal{C}_w$ from recent range, set current video indices $\mathcal{V}_w = \boldsymbol{z}^{i:i+B-1}$
  Pack tokens $\boldsymbol{z}_{in} = \boldsymbol{z}_1^{\boldsymbol{r}} \cup \boldsymbol{z}_1^{i-N_2:i-1} \cup \boldsymbol{z}_t^{i:i+B-1}$
  Assign PlasticRoPE temporal ids:
  $\tau_{in} = \tau_{\text{reason}}(\boldsymbol{z}_1^{\boldsymbol{r}}) \cup \tau_{\text{ctx}}(\boldsymbol{z}_1^{i-N_2:i-1}) \cup \tau_{\text{vid}}(\boldsymbol{z}^{i:i+B-1})$
  Choose schedule $\boldsymbol{t}_w \in \{\boldsymbol{t}_{\text{beg}}, \boldsymbol{t}_{\text{mid}}, \boldsymbol{t}_{\text{end}}\}$
  Guided diffusion on packed tokens:
    $\boldsymbol{z}_w \leftarrow \text{Sampler}(\boldsymbol{z}_{in}, \boldsymbol{t}_w, f_{\text{cfg}})$
  update the denoised video frames array $\boldsymbol{z}$ with newly denoised frames from the last window step.
end for
Decode final latents: $V = \text{VAEdecode}(\boldsymbol{z})$

C.2 Inference Details

We perform auto-regressive inference with PlasticMem model under Temporal Reasoning-based Rendering framework. A summary can be found in Algorithm 2. Given initial noise latents, conditioning latents, and text embeddings, we first build a multi-stage timestep schedule for AR generation, including initialization, mid-stage, and ending schedules, and prepare text embeddings for each shot (positive/negative) under classifier-free guidance. We then run a *full-diffusion reasoning* stage on a set of N_2 noisy reasoning frames, producing reasoning frames that serve as long-term consistency anchors. Next, we execute sliding-window denoising over the video: at each window, we (i) sample additional context frames from a recent temporal range, (ii) pack anchor, context, and current video frames into a joint token set, and (iii) update their Temporal RoPE-based frame indices to distinguish anchors, context, and video frames (PlasticRoPE). We then perform guided diffusion on the packed tokens with the window-specific timestep schedule; the resulting denoised latents overwrite the corresponding frames in the global latent buffer and are also injected into the conditioning map for future steps. This process repeats until all frames are generated. Finally, we decode the completed latent sequence with the VAE to obtain the output video.

C.3 Initialization and termination for Diffusion Forcing

Warm start. We apply a diffusion-forcing warm start for the first sliding window. Specifically, we first perform a standard full-clip diffusion sampling pass on the initial packed clip, obtaining the final denoised latents as well as intermediate latents along the sampling trajectory. Rather than initializing all frames from the fully denoised result, we re-initialize each temporal chunk by overwriting its frames with the corresponding intermediate latent state, such that different chunks start from noise levels consistent with the subsequent auto-regressive schedule. After re-initialization, we reset the timestep plan to the AR-begin schedule $\boldsymbol{t}_{\text{mid}}$ and continue with standard sliding-window denoising. This procedure provides a coherent global initialization while preserving the intended auto-regressive noise progression.

Termination. When the sliding window reaches the final frame during inference, we terminate the rollout and apply asynchronous denoising. All remaining noisy frames within the window are

denoised using the same number of denoising steps but with different step sizes, allowing video chunks at different noise levels to complete denoising simultaneously. This avoids shrinking the window size or continuing to denoise already clean frames, enabling efficient and stable termination.

D Reimplementation Details of StreamDiT

StreamDiT. We re-implement StreamDiT [Kodaira et al., 2025] as our primary baseline, following the original paper in terms of diffusion forcing schedule, and training objective. Our StreamDiT adopts a DiT-style spatiotemporal Transformer operating on VAE-encoded latent video frames $\boldsymbol{z} = \{z_1, \ldots, z_T\}$, with text conditioning injected via Multi-Modal Attention [Gao et al., 2025] (MM-Attn) to text embeddings. To enable streaming long video generation, it employs sliding-window Diffusion Forcing, where a denoising window of size B is applied to latent frames $\boldsymbol{z}^{i:i+B}$ with a structured noise schedule $\boldsymbol{t} = [t_i, \ldots, t_{i+B}]$, assigning lower noise levels to earlier frames and higher noise levels to later ones. This asymmetric noise allocation induces an autoregressive dependency within the diffusion process, allowing future frames to condition on partially denoised past frames. The model is trained using the per-frame flow-matching objective [Chen et al., 2024], where each frame in the window is independently supervised at its assigned noise level:

$$\mathcal{L}_{\text{DF}} = \mathbb{E}_{\boldsymbol{z},\boldsymbol{t}} \sum_{k=i}^{i+B} \| v_\theta(z_k, t_k, \boldsymbol{c}) - v^\star(z_k, t_k) \|_2^2 ,$$

with v_θ denoting the predicted velocity field and $\boldsymbol{c}$ the text condition. During training, video clips are randomly sampled and packed into sliding windows, and Gaussian noise is injected independently into each frame according to its timestep. At inference time, StreamDiT performs progressive sliding-window denoising: once the earliest frame in the window reaches the final noise level, it is popped from the buffer and emitted, while a newly initialized noisy frame is appended, enabling rollout to arbitrary length with constant computation per step. As StreamDiT relies solely on autoregressive diffusion dynamics and local attention within the window, long-term consistency is modeled implicitly, motivating our reasoning-based memory extension.

E Experiment Setups

Benchmark Construction To evaluate a model's ability for stable rollout and visual memory in long video generation, it is necessary to introduce large-scale camera motions and subject movements, which require the model to accurately recall and reproduce previously generated visual content. Such scenarios are rare in standard text-to-video benchmarks, motivating us to construct a dedicated evaluation benchmark with long-horizon dynamics and repeated scene revisits.

Starting from existing text-to-video datasets, we leverage a multimodal large language model (MLLM) to rewrite and filter text prompts into long-form story scripts that explicitly involve multi-stage actions, large motions, and revisits to previously described subjects or scenes. Each resulting test sample consists of a structured story script, including a description of the initial frame followed by multiple action descriptions that unfold over time. This design allows us to assess both prompt-following accuracy and the ability to maintain visual consistency across distant temporal segments. Following this procedure, we curate a total of 100 test story scripts for evaluation. The instruction used for the MLLM-based rewriting and filtering process is provided below.

MLLM Prompt for Story Script Generation.

> **Role and Task.** You are a professional prompt designer for video generation. Given a textual description of the *first frame* of a video, your task is to generate a coherent and natural *dynamic action description* that can be used as a prompt for video generation models.
>
> The goal is to evaluate long-term consistency by designing reasonable action progressions and by introducing elements that appear, disappear, and reappear over time, while strictly preserving consistency with the first-frame description.
>
> **Input.** You will be given a description of the first video frame, which typically includes: (1) the subject's appearance (e.g., gender, age, hairstyle, clothing, accessories); (2) the subject's pose and expression; (3) camera framing and composition

(e.g., close-up or medium shot); (4) scene and environment (e.g., indoor/outdoor, lighting, background). All generated content must strictly adhere to this description. Do not modify the subject's identity, appearance, or the scene setup.

Output Format.

- Output in **Chinese**.
- Produce a **single paragraph** describing a continuous sequence of actions.
- Use the token `\n` to indicate shot changes or stage transitions, but do not insert actual line breaks.
- Include **3–4 consecutive action segments**, corresponding to a short video of approximately 20 seconds.
- Do not explicitly label shots (e.g., "Shot 1").
- Keep the description concise. Do not restate the subject's appearance from the first frame.

Camera and Composition Constraints.

- Avoid complex camera motions (e.g., rapid panning or tracking).
- Motion should mainly arise from the subject, scene elements, or shot transitions.
- If a shot change is needed, directly describe the next shot and separate it with `\n`.

Action Design Principles.

- Actions must be physically and spatially plausible given the first-frame setup.
- Newly introduced elements must lie within the visible region implied by the framing.
- Avoid abrupt or implausible changes (e.g., sudden outfit changes, inconsistent props, or scene-incompatible events).

Long-Term Consistency Design. Introduce consistency challenges using one or more of the following strategies:

- *Multi-view depiction*: reveal different views of the same subject via slow turns or shot changes.
- *Multiple subjects*: alternate focus across subjects using shot transitions.
- *Visual recall*: introduce new elements that later disappear and reappear, requiring the model to recall previously generated content.

Now, given the first-frame description, generate a detailed and logically consistent action prompt to evaluate long-term visual consistency.

Baselines For system-level comparisons with SOTA methods, we directly use their official open-source implementations with the recommended inference settings. We also implemented context compression (following MALT) and context selection (following MemFlow) based on the same base model and training configs with our PlasticMem as follows.

We add context compression following MALT [Yu et al., 2025b], on top of the StreamDiT backbone described above, while following the original MALT design for context compression and memory modeling. MALT addresses long-term consistency by recursively compressing historical contexts into a fixed-size memory representation, avoiding the need to attend to unbounded past frames. Concretely, at each generation step, a memory vector $\boldsymbol{m}$ is updated by aggregating the current video segment hidden states and the previous memory through cross-attention, where the memory serves as queries and the newly generated latent frames serve as keys and values. This compression operation projects variable-length historical contexts into a constant-size latent memory, which is then injected into subsequent denoising steps via the same cross-attention mechanism. During training, memory updates are performed autoregressively with augmented video segments, simulating the memory vector being updated after window advancements in inference. The denoising network itself remains unchanged from StreamDiT, and no additional loss terms are introduced beyond the standard

per-frame flow-matching objective (Eq. 2); the memory is learned implicitly through backpropagation across generation steps. At inference time, the compressed memory is iteratively carried forward and reused for conditioning future segments, enabling long video generation with constant memory cost.

We add context selection following MemFlow [Ji et al., 2025], on top of the StreamDiT backbone, migrating its core idea of *prompts as memory indices* while discarding the original Self-Forcing and Distribution Matching Distillation (DMD) framework. MemFlow models visual memory by using textual prompts to select relevant historical frames for conditioning future generation. In our implementation, we directly train on multi-prompt videos, where each video is segmented into multiple consecutive segments, each associated with its own text prompt. When generating a target video segment, we compute text–frame relevance between the current segment prompt and latent frames from previous history, and select a subset of frames with the highest similarity scores. These retrieved frames are then injected as in-context visual conditions to guide denoising of the current segment. The denoising network, noise schedule, and training objective follow the standard StreamDiT setup with sliding-window Diffusion Forcing and the per-frame flow-matching loss (Eq. 1). During inference, the prompt-based retrieval mechanism is applied autoregressively: for each new segment, relevant historical frames are selected from the accumulated video history and provided as contextual conditioning. This design enables text-guided context selection for long video generation with bounded context size, while relying on retrieval quality to maintain long-term consistency across segments.

F Limitations and Future Work

The current implementation uses a simple proof-of-concept construction for reasoning-frame supervision: one frame is sampled from each annotated action or shot segment. Since the annotation process already separates videos into semantically coherent segments, random sampling within each segment is usually sufficient to produce a compact visual storyboard. More advanced selection strategies, such as feature clustering, saliency-based sampling, or specialized reasoning prompts, could further improve the quality of reasoning anchors and are promising directions for future work.

Training only on short videos also imposes boundary conditions. Our method can learn reasoning and rendering from abundant short clips, but unconstrained long-form narrative generation is still limited by the maximum text length that the base model can understand, which restricts the richness and complexity of the described content, as well as by its motion understanding and ability to handle complex scene transitions. For videos with many scenes or dense events, one practical strategy is to decompose the full script and generate reasoning frames scene by scene, so that each reasoning prompt remains short and explicit while rendering still benefits from shared long-term anchors. Finally, PlasticRoPE currently relies on temporal index extrapolation when generating videos longer than those seen during training; more advanced RoPE extrapolation or length-generalization techniques may further improve extremely long generation.

G Broader Impact

This work aims to make long-form visual storytelling more accessible by reducing the dependence on scarce long-video training data. If used responsibly, methods such as PlasticMem could lower production barriers for education, art, entertainment, and communication, enabling individual creators and small teams to produce coherent long videos with fewer resources. The reasoning-based decomposition may also support more controllable generation workflows, where users can inspect or edit intermediate visual plans before rendering.

At the same time, improving long-video generation can increase the risk of misuse, including synthetic-video misinformation, impersonation, copyright infringement, or the creation of misleading narratives at longer temporal scales. Generated videos may also inherit biases, stereotypes, or unsafe content from the underlying base video model and training data. These risks are not unique to PlasticMem, but longer and more consistent videos may make harmful synthetic media more persuasive. Potential mitigations include watermarking or provenance tracking, clear disclosure of generated content, restrictions on impersonation and harmful uses, careful dataset and prompt filtering, and staged release policies with safety documentation if models or data are made public.

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 introduction state the main contributions: TRR, PlasticMem, PlasticRoPE, unified training, and evaluation for consistent long-video generation. The scope and remaining limitations are further discussed in the conclusion and Appendix F.

   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 paper discusses limitations in the conclusion and in Appendix F, including dependence on reasoning-frame quality, short-video training boundary conditions, base-model limitations, and temporal-index extrapolation.

   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: [N/A]

Justification: The paper does not present formal theoretical results, theorems, or proofs. The equations are used to formulate the modeling assumptions and objectives of the proposed framework.

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: The paper provides the model formulation, training objectives, algorithmic workflows, data preparation, benchmark construction, baseline descriptions, and implementation details in Sections 3–5 and Appendices C–E.

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: [Yes]

   Justification: The paper provides the model formulation, training objectives, algorithmic workflows, data preparation, benchmark construction, baseline descriptions, and implementation details in Sections 3–5 and Appendices C–E. The benchmark source data is 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 paper specifies the main training and evaluation settings, including frame budgets, training steps, learning rate, GPU type/count, benchmark construction, baseline implementations, and ablation settings in Section 5 and Appendices C.1–E.

   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: The paper reports quantitative comparisons and ablations according to the same protocol as previous works. The experiments are computationally expensive long-video generation evaluations, so the reported tables focus on aggregate metrics.

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: The paper reports the main training compute as 24 A100 GPUs for 10,000 training steps and provides representative inference GPU memory and throughput in Appendix A.5.

   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: The work is intended for research on generative modeling and long-video consistency and does not involve deceptive deployment, surveillance, or decision-making about individuals. The authors have considered responsible use under 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: [Yes]

    Justification: The paper discusses the potential positive societal impacts of the work, including lowering barriers for visual storytelling, and the potential negative societal impacts of the work, including the risk of synthetic-video misinformation and impersonation.

    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: The paper does not release a high-risk pretrained generative model or scraped dataset. If models or data are released in the future, appropriate access policies, usage restrictions, and safety documentation will be provided.

    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 paper cites the main existing methods, models, and benchmarks used for comparison, including StreamDiT, LongLive, MemFlow, FramePack, and VBench.

    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 paper does not introduce new assets.

    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: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    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: The paper does not involve crowdsourcing nor research with human subjects.

    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: [Yes]

    Justification: The paper describes LLM/MLLM usage for constructing long-form evaluation prompts and for scalable video annotation, including the prompt used for story-script generation in Appendix E. LLMs are not the central model being proposed, but they are used as part of the data and benchmark preparation pipeline.

    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.