

A DATA PREPARATION

Data Sources and Filtering. Our dataset comprises two components: (i) licensed commercial videos purchased from authorized providers; and (ii) web videos manually collected from open platforms, primarily [Mixkit](#), [Pexels](#), and [Pixabay](#). For all candidate videos, we first generate textual captions with Qwen-72B and compute an aesthetic score using the LAION aesthetic predictor. For licensed data, we rank by the aesthetic score and retain only the top 1%. For web-collected data, we conduct human quality control to remove low-quality clips, content-caption mismatches, and potential copyright risks, and to verify caption consistency. We then merge the two subsets, yielding approximately $\sim 50k$ high-quality samples to train our long-video generation model.

Structured Annotation Pipeline. To obtain rich and structured annotations, we drive Qwen-72B with carefully designed instruction prompts to analyze video frames and output a JSON object with a fixed schema. The JSON includes: a short scene summary (`short_caption`); a dense contextual description (`dense_caption`, covering main subject, background, visual style, camera movement, shot type, lighting, and atmosphere); detailed subject descriptions (for persons: facial expressions, emotional state, and ethnicity); background information; standardized style/shot/movement labels; aesthetic tags; and role statistics (e.g., number of humanoid characters, coverage extent, depiction style, and motion dynamics). Concretely, `short_caption` is generated with the instruction “*Brief scene summary in 1 sentence*”, while `dense_caption` uses “*Detailed context including main subject, background, visual style, camera movement tech, shot type, lighting, and atmosphere*”. All outputs are in English, follow predefined field orders and constraints, and employ standardized vocabulary for key attributes.

Quality Control and Training Setup. After annotation, we re-evaluate aesthetic quality with the LAION predictor to ensure consistency. During training, for each of the $\sim 50k$ videos we sample the conditioning text with probability 0.8 from `dense_caption` and with probability 0.2 from `short_caption`. This strategy preserves the high information density of dense captions while maintaining robustness and diversity from concise summaries. As shown in Figure 9, we present several representative examples from our curated dataset.



Figure 9: **Examples of training samples.** The dataset combines licensed and web-collected videos, curated via aesthetic scoring and manual screening.

B USER STUDY DETAILS

To complement the quantitative metrics, we conduct a user study on long-video generation. In each trial, participants evaluate five videos generated from the same prompt by ranking them (1 = best, 5 = worst) along three dimensions: *text–visual alignment*, *content consistency*, and *long-sequence color stability*. In addition, participants select a single overall favorite video.

User Study Details

Rank 5 videos on 3 criteria

Hide Instructions

Instruction

Click to collapse

You will be shown **five** AI-generated videos created from the same text prompt. Please **rank all 5 videos** for each criterion below (1 = best, 5 = worst), and then select your single **overall favorite** video.

- Text Alignment: faithfulness to the prompt semantics.
- Content Consistency: temporal coherence and absence of discontinuities.
- Color Shift: stability of color/illumination without drift or flicker.

Each criterion must be a permutation of 1–5 (no ties).


Text Prompt

A close-up shot of a majestic white dragon with pearlescent, silver-edged scales, icy blue eyes, and elegant ivory horns. The dragon's face is detailed with subtle wrinkles and sharp, defined features, capturing a regal and serene expression. Its breath forms a gentle mist, adding to the ethereal quality. The scales are meticulously textured, reflecting light in a way that highlights their depth and shine. ...

Videos


Option A

☒ Favorite




Option B

☐ Favorite




Option C

☐ Favorite




Option D

☐ Favorite



Option E

☐ Favorite



Rankings (1 = best, 5 = worst)

Video	Text Alignment	Content Consistency	Color Shift
Option A	1 <input type="button" value="v"/>	1 <input type="button" value="v"/>	1 <input type="button" value="v"/>
Option B	2 <input type="button" value="v"/>	2 <input type="button" value="v"/>	2 <input type="button" value="v"/>
Option C	3 <input type="button" value="v"/>	3 <input type="button" value="v"/>	3 <input type="button" value="v"/>
Option D	4 <input type="button" value="v"/>	4 <input type="button" value="v"/>	4 <input type="button" value="v"/>
Option E	5 <input type="button" value="v"/>	5 <input type="button" value="v"/>	5 <input type="button" value="v"/>

Text Alignment: valid permutation

Content Consistency: valid permutation

Color Shift: valid permutation

Optional Notes

(Optional) Briefly justify your rankings (e.g., color stability, motion, or alignment).

☒ I have read the instructions and will evaluate fairly.

Reset

Submit

Figure 10: User study instruction screenshots.

This protocol provides fine-grained human judgments on both quality and temporal robustness that are not fully captured by automated metrics. Detailed instructions are shown in Figure 10.

C TRAINING SETTINGS

C.1 HYPERPARAMETER SETTINGS

Most experiments are conducted on 32 NVIDIA GPUs (80 GB each), using a per-GPU batch size of 1 without gradient accumulation. The detailed hyperparameters are summarized in Table 4. Training the Teacher Forcing 14B model for 8,000 steps required about three days, while the DMD 1.3B model reached 8,000 steps within roughly one day.

Table 4: Specification of training hyperparameters

Hyperparameters	Teacher Forcing	Self Forcing
Generate network	Wan2.1-T2V-14B	Wan2.1-T2V-1.3B
Real score network	N/A	Wan2.1-T2V-14B
Fake score network	N/A	Wan2.1-T2V-14B
Batch size	32	32
Optimizer (G_θ)	AdamW, $\beta_1 = 0$, $\beta_2 = 0.999$, $\epsilon = 1 \times 10^{-8}$, weight_decay = 0.01	Adam, $\beta_1 = 0$, $\beta_2 = 0.999$, $\epsilon = 1 \times 10^{-8}$, weight_decay = 0.01
Optimizer (f_ψ)	N/A	Adam, $\beta_1 = 0$, $\beta_2 = 0.999$, $\epsilon = 1 \times 10^{-8}$, weight_decay = 0.01
Learning rate (G_θ)	1×10^{-5}	2×10^{-6}
Learning rate (f_ψ)	N/A	4×10^{-7}
Gen./Cri. update ratio	N/A	5
EMA decay	N/A	0.99

C.2 PLANNING SETTINGS

Settings. To clarify the generation process, we detail the model’s computation using the 21 latent tokens of the full Wan model as shown in Figure 11. Tokens are indexed from 0: indices 0–1 correspond to the initial frame, indices 2–3 to early planning frames, indices 10–12 to midpoint planning frames, and indices 19–20 to terminal planning frames.

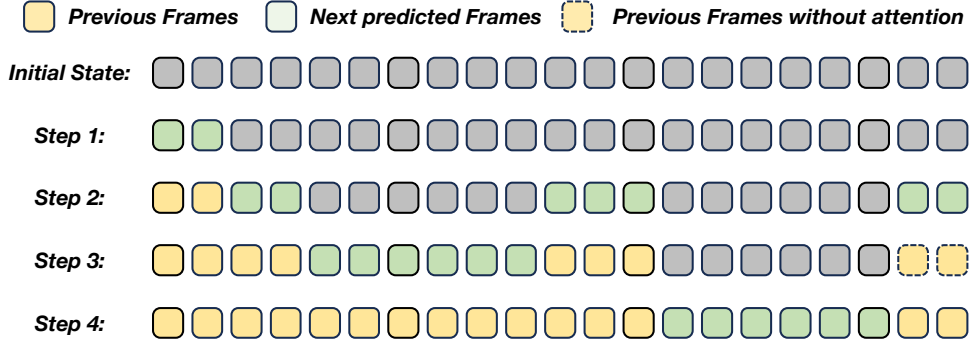


Figure 11: **Overview of our planning-based inference on an 81-frame sequence with 21 latent tokens (full Wan model).** Pre-Planning latent tokens at the beginning, midpoint, and terminal positions serve as stable anchors in the denoising schedule, guiding the synthesis of all intermediate frames and ensuring long-range temporal coherence.

Analysis. While the proposed planning setup places anchors over long horizons, a central challenge remains: enabling the autoregressive (AR) decoder to effectively exploit these anchors when synthesizing intermediate frames. Relying on a single planning token at the *early*, *midpoint*, and *terminal* boundaries is intrinsically fragile in the presence of the AR decoder’s pronounced *recency bias*—the tendency to overweight the most recent observations while underutilizing distant context. This bias causes the model, at each sub-segment junction, to condition predominantly on the tail of the preceding sub-segment, thereby inheriting and amplifying residual errors and inducing cross-boundary

propagation. Consequently, a single planning token per boundary is insufficient to arrest drift arising from accumulated errors. Formally, this bias in MMPL is expressed in Eq. 6:

$$p(x^{s_k+1:e_k-1} | x^{1:s_k}, x^{e_k}) \approx p(x^{s_k+1:e_k-1} | x^{s_k-K:s_k}, x^{e_k}). \quad (6)$$

Here, s_k and e_k denote the starting and ending reference indices of sub-segment k , corresponding to the pre-planned *planning frames*. The hyperparameter K specifies the size of the recent-context window on which the AR decoder conditions—namely, the K frames immediately preceding s_k . Because $\{s_k - K : s_k\}$ overlaps with the tail of the previous sub-segment, residual errors inevitably leak into the current one, leading to error propagation across boundaries.

To counteract this bias, we replace the single predecessor at each boundary with a *local multi-frame set*. Concretely,

$$\mathcal{P}_{s_1} = \{2, 3\}, \quad \mathcal{P}_{e_1} = \{10, 11, 12\}, \quad \mathcal{P}_{s_2} \approx \{10, 11, 12\}, \quad \mathcal{P}_{e_2} = \{19, 20\},$$

where \mathcal{P}_{s_k} denotes the local index set of expanded pre-planning frames around boundary s_k . Using these expanded anchors, the conditional distribution for sub-segment k is refined to

$$p(x^{s_k+1:e_k-1} | x^{1:s_k}, x^{e_k}) \approx p(x^{s_k+1:e_k-1} | x^{(s_k-2-K):(s_k-3)}, x^{s_k-2:s_k}, x^{e_k}). \quad (7)$$

Conditioning on a compact bundle of early-step, low-drift frames—rather than a single predecessor—dilutes residual errors inherited from the previous sub-segment. At the same time, the model’s recency bias naturally prioritizes the most recent elements within this bundle, thereby stabilizing long-horizon synthesis and suppressing cross-boundary error propagation without discarding information from the planned anchors.

D ERROR ACCUMULATION ANALYSIS IN AR MODELS

Autoregressive (AR) Models. Autoregressive (AR) models generate a sequence $x = (x^1, \dots, x^T)$ by factorizing its joint probability distribution according to the chain rule of probability:

$$p_\theta(x) = \prod_{t=1}^T p_\theta(x^t | x^{<t}), \quad (8)$$

where $x^{<t} = (x^1, \dots, x^{t-1})$ denotes all previously generated elements. In practice, AR models are commonly trained with the *teacher forcing* strategy, which replaces the model’s own past predictions with the ground-truth history during training. This reduces the training objective to a standard negative log-likelihood (NLL) minimization:

$$\mathcal{L}(\theta) = - \sum_{t=1}^T \log p_\theta(x^t | x^{<t} \text{gt}), \quad (9)$$

where $x^{<t} \text{gt}$ denotes the ground-truth prefix of the sequence. Such training ensures stable and efficient optimization, but it also introduces a train-test discrepancy—commonly referred to as *exposure bias* (Ning et al., 2024)—because the model will rely on its own predictions rather than ground-truth history during inference, potentially leading to error accumulation over long sequences.

To analyze the underlying sources and impacts of error accumulation, we follow (Arora et al., 2022) and formulate AR generation as a sequential decision process under the imitation learning (IL) framework. Here, the state is defined as $s^t = x^{<t}$, the action as $a^t = x^t$, the policy as $\pi_\theta(a^t | s^t) = p_\theta(x^t | x^{<t})$, and the oracle policy as $\pi^*(a^t | s^t) = p_{\text{data}}(x^t | x^{<t})$. Maximum-likelihood training corresponds to behavior cloning, which minimizes training loss on the oracle-induced state distribution but suffers from compounding errors once the policy is executed on its own rollouts.

In the imitation learning literature (Ross et al., 2011), rolling out a policy trained via behavior cloning often leads to error accumulation. This happens because the policy is executed on its own predictions rather than the oracle states seen during training. To analyze this effect, researchers use inference-time regret, which measures the performance gap between the behavior cloning policy π_{BC} and the oracle policy o during rollout:

$$\mathcal{R}(\pi_{BC}) = L^I(\pi_{BC}) - L^I(o). \quad (10)$$

Here, $L^I(\pi)$ denotes the expected cumulative loss (or cost) when executing policy π over the entire rollout horizon. Let ϵ denote the average expected error of executing the behavior cloning policy π_{BC} over T steps, which itself is upper-bounded. The regret of behavior cloning is bounded by

$$T\epsilon \leq \mathcal{R}(\pi_{BC}) \leq T^2\epsilon, \quad (11)$$

Building on this analysis, and following (Arora et al., 2022), we further extend it to the AR video generation setting with model p_θ and decoding strategy \mathcal{F} , which yields

$$T\epsilon \leq \mathcal{R}(p_\theta, \mathcal{F}) \leq T^2\epsilon, \quad (12)$$

which demonstrates that even small per-step errors can accumulate linearly in expectation and quadratically in the worst case, thereby explaining the progressive drift and long-horizon degradation observed in autoregressive generation under exposure bias.

E IMPORTANCE OF VAE

We compare the extrapolation procedure from the public Causvid (Yin et al., 2025) codebase against our proposed *drift-resilient re-encoding and decoding strategy* as shown in Figure 12. When extrapolation goes beyond the training context length and requires segment stitching, the baseline suffers from severe color drift and visual artifacts, whereas our method effectively mitigates these degradations.

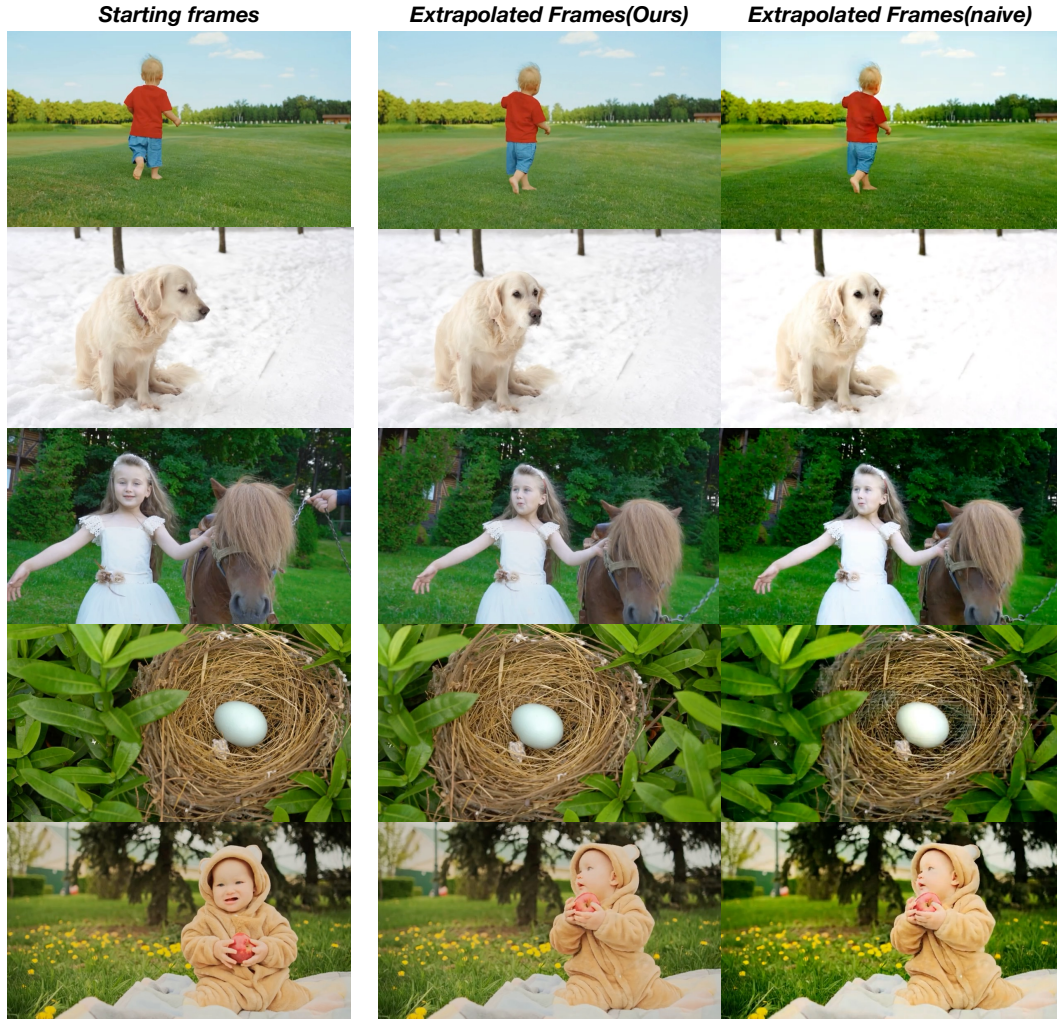


Figure 12: Qualitative comparisons on video extrapolation.

F NOISE INITIALIZATION STRATEGY FOR SMOOTHING GENERATION

In this work, we propose a specialized noise initialization strategy to address potential temporal discontinuities and instability at the transition boundaries between planning frames and content frames as shown in Figure 13. This approach ensures smooth visual transitions by strategically incorporating noise information from adjacent planning frames during the content frame generation process. Let P_{n-1} and P_n represent the planning frames at temporal positions $n-1$ and n , respectively, and let C_{n+1} denote the target content frame at position $n+1$. To establish the theoretical foundation, we first recall the standard diffusion forward process formulation. Given a clean frame \mathbf{x}_0 at diffusion timestep t , the noisy observation \mathbf{x}_t is generated through the Gaussian perturbation:

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1 - \bar{\alpha}_t)\mathbf{I}), \quad (13)$$

where $\bar{\alpha}_t$ denotes the cumulative product of the noise schedule coefficients. This process can be equivalently expressed as:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \cdot \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \cdot \epsilon, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}). \quad (14)$$

Building upon this formulation, our methodology initializes the noise vector $\epsilon_{C_{n+1}}$ for the content frame C_{n+1} through a weighted interpolation of the noise vectors associated with the preceding planning frames. Specifically, the initialization follows:

$$\epsilon_{C_{n+1}} = \alpha \cdot \epsilon_{P_n} + (1 - \alpha) \cdot \epsilon_{P_{n-1}}, \quad (15)$$

where $\epsilon_{C_{n+1}}$ represents the noise vector utilized in the reverse diffusion process for generating content frame C_{n+1} , ϵ_{P_n} and $\epsilon_{P_{n-1}}$ correspond to the noise vectors derived from planning frames P_n and P_{n-1} . This noise initialization strategy ensures a continuous evolution of stochastic patterns across frame boundaries, effectively mitigating visual artifacts and temporal inconsistencies. By controlling the interpolation weight α , our method provides precise adjustment over the temporal smoothness characteristics, enabling stable and coherent video generation while maintaining high visual quality throughout the sequence.

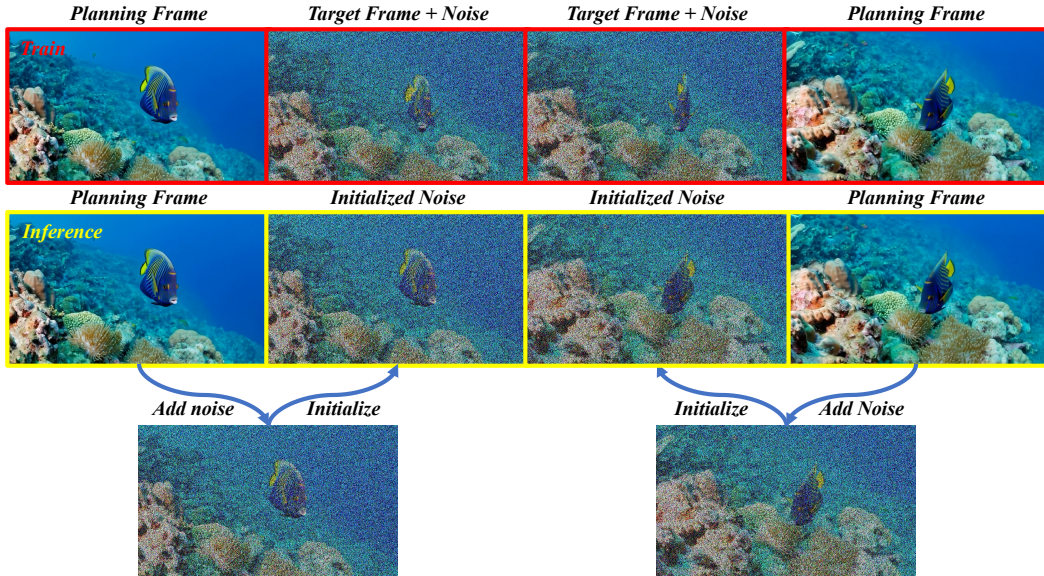


Figure 13: Framework for stable and smoothing frame generation via coherent noise initialization.

G DISCLOSURE OF LARGE LANGUAGE MODEL (LLM) USAGE

In this paper, we used Large Language Models (LLMs) to assist in various aspects of the writing process. Specifically, LLMs were employed to help polish the writing, improve clarity, and enhance the overall presentation of the text. The models were utilized to provide suggestions for improving the grammar, coherence, and flow of certain sections of the manuscript. This assistance was integral to the refinement of the paper’s language, but all scientific content, methodology, and conclusions were independently developed by the authors. The use of LLMs is limited to language-related tasks and does not extend to the intellectual contributions to the research findings or data analysis.

H SCALABILITY

H.1 IMAGE-TO-VIDEO EXTENSION

Our framework is not restricted to the text-to-video (T2V) task; it can be seamlessly extended to image-to-video (I2V) generation without introducing any architectural modifications or additional image encoders. This flexibility derives from the unified autoregressive design, which only requires lightweight adjustments to the number and ordering of autoregressive steps. As a result, the framework adapts naturally to different input modalities while maintaining temporal consistency and generation quality as shown in Figure 14.



Figure 14: **Qualitative results of extending our unified autoregressive framework from text-to-video (T2V) to image-to-video (I2V) generation.** Without any architectural modifications or additional image encoders, the framework adapts seamlessly by only adjusting the number and ordering of autoregressive steps, while preserving temporal consistency and visual quality.

H.2 ADAPTATION TO SELF-FORCING AND DMD

Our approach can be seamlessly integrated with self-forcing strategies without any architectural modifications. Specifically, it only requires adjusting the attention visibility range and the prediction order during both training and inference. This lightweight adaptation enables direct compatibility with existing self-forcing pipelines, while retaining the benefits of our planning-based design. Combined with parallelized decoding, the resulting system achieves substantial inference speedups, sustaining over 32 FPS in long-horizon video generation as shown in Figure 15.

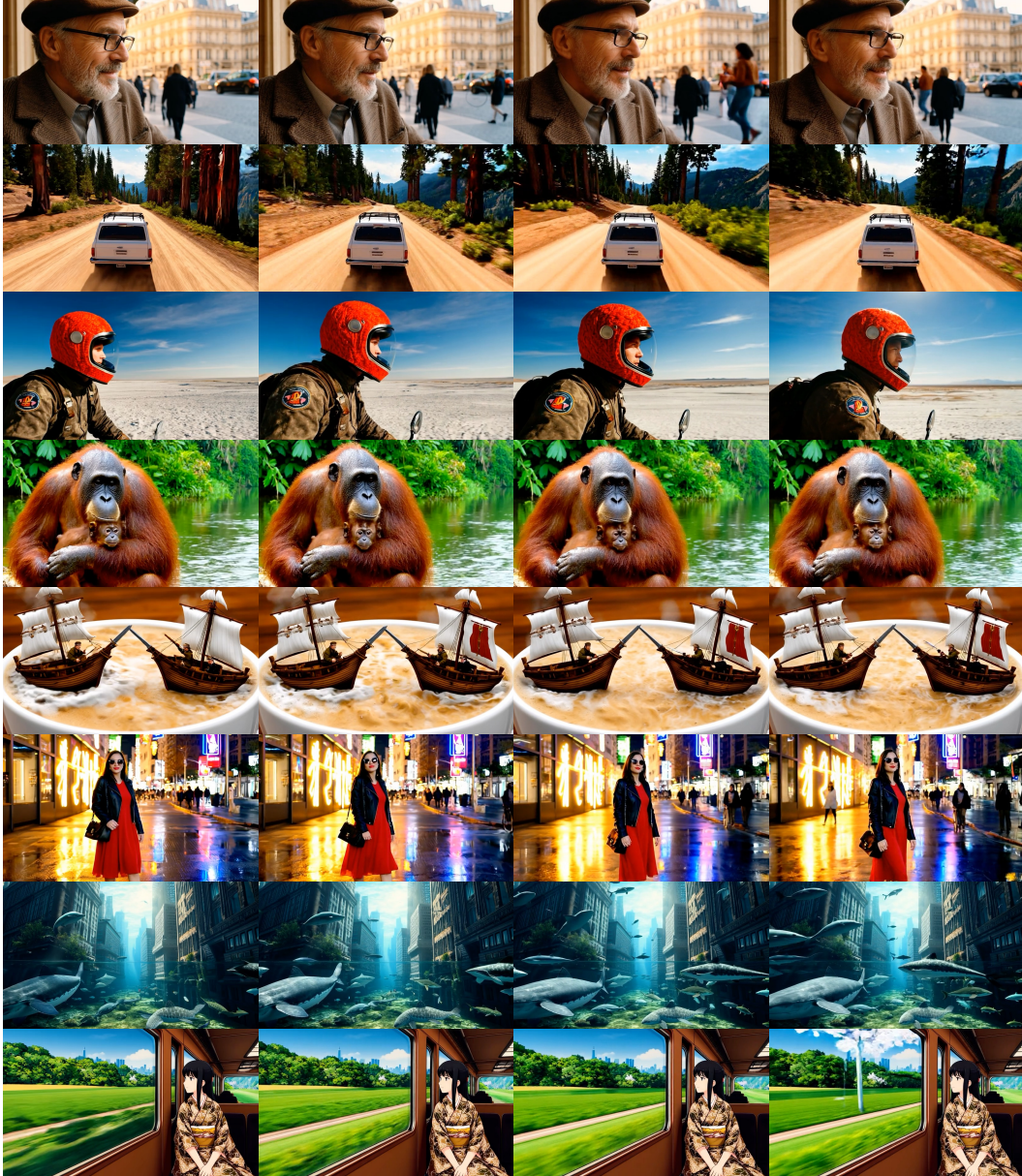


Figure 15: **Integration of our framework with Self Forcing (Huang et al., 2025a) and DMD (Yin et al., 2025) strategies.** The adaptation requires no architectural changes—only modifications to the attention visibility range and prediction order during training and inference. Combined with parallelized decoding, the method achieves substantial inference acceleration, sustaining over 32 FPS in long-horizon video generation.

I MORE QUALITATIVE RESULTS

To better demonstrate the robustness of our model, we present additional experimental results on 30s long video generation, as shown in Figure 16.



Figure 16: **Additional qualitative results of 30s long video generation.** Our model produces temporally coherent and visually consistent sequences across diverse scenarios, further demonstrating its robustness.