SWIFTI2V: EFFICIENT HIGH-RESOLUTION IMAGE-TO-VIDEO GENERATION VIA CONDITIONAL SEGMENT-WISE GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

High-resolution image-to-video (I2V) generation aims to synthesize realistic temporal dynamics while preserving fine-grained appearance details of the input image. At 2K resolution, it becomes extremely challenging, and existing solutions suffer from various weaknesses: 1) end-to-end models are often prohibitively expensive in memory and latency; 2) cascading low-resolution generation with a generic video super-resolution tends to hallucinate details and drift from input-specific local structures, since the super-resolution stage is not explicitly conditioned on the input image. To this end, we propose **SwiftI2V**, an efficient framework tailored for high-resolution I2V. Following the widely used two-stage design, it addresses the efficiency–fidelity dilemma by first generating a low-resolution motion reference to reduce token costs and ease the modeling burden, then performing a strongly image-conditioned 2K synthesis guided by the motion to recover input-faithful details with controlled overhead. Specifically, to make generation more scalable, SwiftI2V introduces *Conditional Segment-wise Generation (CSG)* to synthesize videos segment-by-segment with a bounded per-step token budget, and adopts *bidirectional contextual interaction* within each segment to improve cross-segment coherence and input fidelity. On VBench-I2V at 2K resolution, SwiftI2V achieves performance comparable to end-to-end baselines while reducing total GPU-time by 202×. Particularly, it enables practical 2K I2V generation on a single datacenter GPU (*e.g.*, H800) or consumer GPU (*e.g.*, RTX 4090).

1 INTRODUCTION

Diffusion Transformer (DiT) (Peebles & Xie, 2023) architectures have improved video quality and temporal coherence (Wan et al., 2025; Hunyuan, 2025; Yang et al., 2024; HaCohen et al., 2026). High-resolution synthesis (*e.g.*, 2K and above) is increasingly important, but most studies target text-to-video (T2V) generation aligned with textual semantics (Zhang et al., 2025; Ren et al., 2025; Qiu et al., 2025a;b). Users with a high-resolution image seek to **generate a plausible dynamic video while faithfully preserving the image's spatial structure and fine-grained textures**, *i.e.*, image-to-video (I2V). Despite advances in high-resolution T2V (Zhang et al., 2025; Ren et al., 2025; Qiu et al., 2025a;b), efficient, strongly image-conditioned 2K I2V remains challenging.

High-resolution I2V faces two challenges. First, *computational scaling*: visual tokens grow rapidly with resolution, increasing attention computation and memory. Second, *fidelity under strong image conditioning*: plausible motion must preserve input-specific high-frequency details (*e.g.*, textures, identity cues) with minimal frame-to-frame drift, especially at higher resolutions.

Currently, research on high-resolution I2V remains relatively limited, and two practical paradigms are commonly considered: 1) **End-to-end**: End-to-end high-resolution generation with a single model (Qiu et al., 2025b) is conceptually simple and can sometimes yield high-fidelity outputs, but must *process all tokens while jointly learning global motion and fine details.* Such a coupled learning objective often necessitates a larger backbone and more sampling steps. Meanwhile, processing all tokens drives GPU memory usage and computation to prohibitive levels, making training and inference difficult to scale. 2) **LR+VSR**: One can first generate a low-resolution (LR) video to reduce spatiotemporal modeling cost, and then upscale it using a relatively small video super-resolution

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

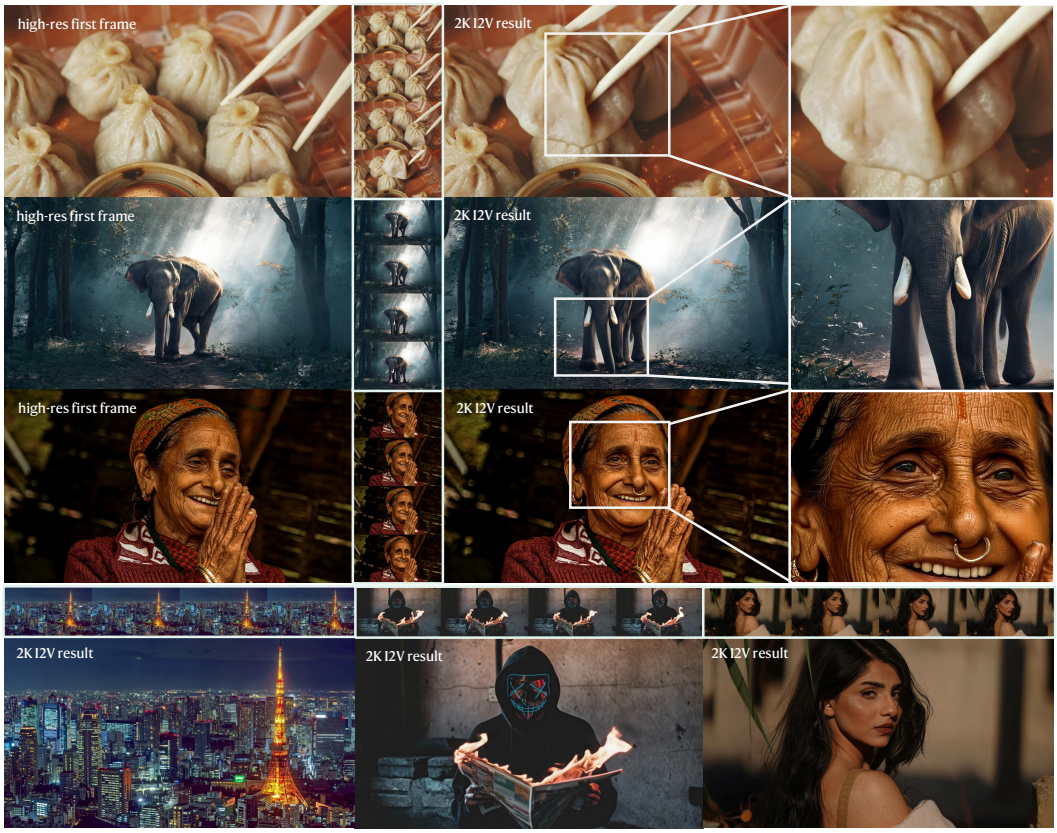

Figure 1: Given a high-quality first frame, SwiftI2V enables faithful 2K image-to-video generation with fine-grained details in 2 minutes.

(VSR) model (Li et al., 2025; Shiu et al., 2025; Zhuang et al., 2026; Wang et al., 2025a;b). This improves efficiency, but the VSR stage is often *not explicitly guided by the input image*, making it prone to hallucinated details and input-structure drift.

Despite substantial progress, existing high-resolution I2V pipelines still struggle to balance *efficiency* and *fidelity*. To address both challenges simultaneously, we propose **SwiftI2V**, an efficient framework tailored for 2K-resolution I2V, as shown in Tab. 1. SwiftI2V balances efficiency and fidelity by starting with low-resolution motion generation to reduce token costs, and then proceeding to a 2K refinement stage that simultaneously controls computational overhead and introduces strong image conditioning for detail synthesis.

Pipeline	Runtime	Memory	I2V fidelity
End-to-end	☹	☹	☺
LR + VSR	☺	😐	☹
SwiftI2V	☺	☺	☺

Table 1: **Qualitative comparison** of common 2K I2V pipelines under strong image condition.

Our key observation is that globally coherent motion can be reliably inferred at much lower spatial resolution, whereas preserving input-specific high-frequency structures is primarily a high-resolution refinement problem that hinges on *strong conditioning on the given image.* This observation naturally fits a motion–detail decoupled two-stage design that is widely adopted in recent high-resolution video generation (Zhang et al., 2025; Ren et al., 2025), where a low-resolution stage handles motion and a high-resolution stage handles appearance. SwiftI2V follows this common framework, and focuses our design on *how each stage is realized for 2K I2V*: the low-resolution stage focuses on global motion and coarse appearance, while the high-resolution stage is cast as a *conditional high-resolution video generator* that natively synthesizes 2K frames under joint image and motion conditioning, rather than a generic video super-resolution model. To close the train–test gap at the stage interface, we employ a simple stage-transition strategy that produces Stage I-like degraded LR videos for training Stage II, enabling it to handle low-resolution generation artifacts that generic VSR cannot address.

For scalability, we further introduce **Conditional Segment-wise Generation (CSG)**, which partitions the temporal sequence into bounded segments for controllable memory and streaming generation. Within each segment, an image-anchored **bidirectional contextual interaction** lets neighboring and current segments interact, mitigating discontinuities and error accumulation while improving fidelity.

Our **contributions** are summarized as follows:

(i) We propose **SwiftI2V**, an efficient high-resolution I2V framework that tackles the efficiency–fidelity dilemma. On VBench-I2V at 2K, SwiftI2V matches strong end-to-end high-resolution baseline (Qiu et al., 2025b) on key I2V metrics while reducing total GPU-time by 202×, and supports practical 2K I2V on a single consumer GPU (*e.g.*, RTX 4090).

(ii) We propose **Conditional Segment-wise Generation (CSG)** with bidirectional contextual interaction for streaming with bounded per-step 2K tokens, mitigating autoregressive error accumulation.

(iii) We introduce a simple **stage-transition training** strategy that injects Stage I-like artifacts into Stage II inputs, reducing the cascade's train–test gap.

2 Related Work

Video Diffusion Models (VDMs). Following early video diffusion models (Ho et al., 2022), latent diffusion models (LDMs) (Rombach et al., 2022) improve scalability through compressed latent spaces (Blattmann et al., 2023). Transformer (Peebles & Xie, 2023) backbones now dominate, improving visual fidelity, aesthetic quality, and spatiotemporal coherence (Yang et al., 2024; Wan et al., 2025; Hunyuan, 2025). Most VDMs target text-to-video (T2V) (Yang et al., 2024; Wan et al., 2025; Hunyuan, 2025), whereas I2V additionally requires strict spatial and semantic consistency with an input image. Approaches either add image conditioning to T2V models (Wan et al., 2025; Hunyuan, 2025) or are designed specifically for I2V (Shi et al., 2024; Zhang et al., 2023).

High-Resolution Video Generation. High-resolution T2V uses resolution-specific training (Guo et al., 2024; Zhang et al., 2025; Ren et al., 2025; Wang et al., 2025b) or tuning-free strategies (Qiu et al., 2025b); recent work also explores high-resolution I2V under strong image conditions (Qiu et al., 2025b). These approaches increase computational and memory costs, limiting scalability. An alternative generates low-resolution videos and applies video super-resolution (Zhou et al., 2024; Li et al., 2025; Shiu et al., 2025; Zhuang et al., 2026; Xie et al., 2025; Wang et al., 2025a), but may recover plausible rather than input-faithful fine details.

Efficient Video Generation. Multi-step denoising and quadratic attention costs motivate distillation to reduce sampling steps (Wang et al., 2023; Yin et al., 2025) and causal modeling to accelerate attention (Yin et al., 2025; ai et al., 2025; Qiu et al., 2025a). These techniques extend to high-resolution T2V (Qiu et al., 2025a), but remain less explored for high-resolution I2V. Concurrent LTX-2 (HaCohen et al., 2026) is an efficient joint audio–visual foundation model supporting 2K I2V, but is not tailored to strong image anchoring, leaving a fidelity–motion gap.

3 Method

Overview. SwiftI2V achieves 2K I2V within a tractable budget via two stages (Fig. 2): Stage I generates a low-resolution motion reference, and Stage II synthesizes input-faithful high-resolution details through a lightweight conditioning interface. Stage II further uses CSG with bidirectional contextual interaction to control the per-step token budget while preserving fidelity.

3.1 Two-stage High-Resolution I2V Framework

Given a high-resolution input image $\mathbf{x} \in \mathbb{R}^{H\times W\times 3}$, our goal is to synthesize a T-frame 2K video $\hat{\mathbf{V}} \in \mathbb{R}^{T\times H\times W\times 3}$ that exhibits realistic temporal dynamics while faithfully preserving input-specific spatial structure and fine-grained textures. Below we describe how each stage is instantiated and how motion reference is transferred across stages.

Stage I: Low-Resolution Motion Reference Generation. Stage I models globally coherent motion at low resolution by downsampling the input image as $\mathbf{x}^{\mathrm{LR}} = \mathrm{Down}(\mathbf{x})$ and using a large-capacity

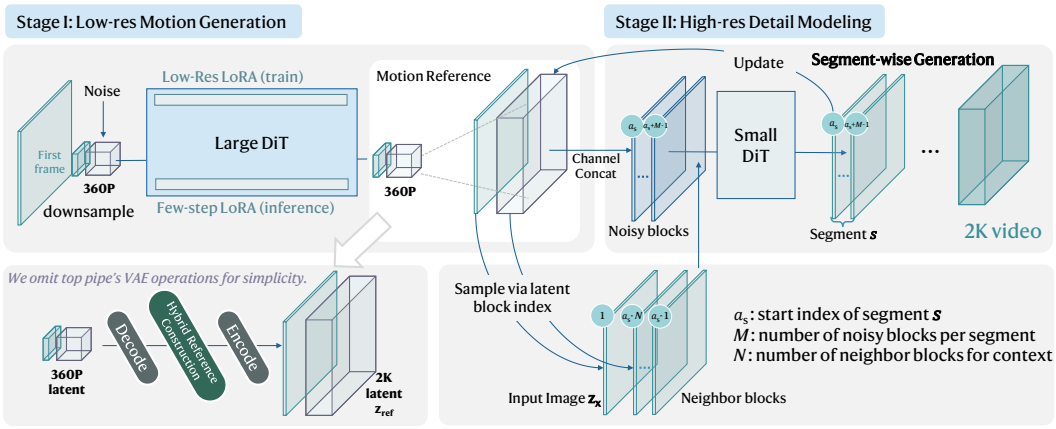

Figure 2: **Overview of SwiftI2V.** Stage I generates a low-resolution motion reference, which is fused with the input high-resolution image and concatenated with Stage II DiT noise. Stage II then generates the high-resolution video segment-by-segment.

DiT backbone $\mathcal{G}_1$ to generate a low-resolution video $\hat{\mathbf{V}}^{\mathrm{LR}}$ as a motion and structure reference:

$$\hat{\mathbf{V}}^{\mathrm{LR}} = \mathcal{G}_1\big(\mathbf{x}^{\mathrm{LR}}\big) \in \mathbb{R}^{T\times H_{\mathrm{LR}}\times W_{\mathrm{LR}}\times 3}. \tag{1}$$

Operating at low resolutions greatly reduces the token count, allowing us to afford a large-capacity backbone that robustly learns motion priors while keeping the compute budget manageable. On top of this backbone, we train a *Low-Res LoRA* (Hu et al., 2022) for resolution adaptation, and further couple it with an off-the-shelf *Few-Step LoRA* (LightX2V, 2025) at inference to reduce the number of denoising steps, yielding a fast yet motion-faithful reference generator.

Pixel-Space Transition: Hybrid Reference Construction. To transfer Stage I motion priors to high-resolution synthesis, we upsample its output to the target resolution:

$$\mathbf{V}^{\mathrm{up}} = \mathrm{Up}\Big(\hat{\mathbf{V}}^{\mathrm{LR}}\Big) \in \mathbb{R}^{T\times H\times W\times 3}. \tag{2}$$

Let $\mathbf{V}^{\mathrm{up}}_{\tau} \in \mathbb{R}^{H\times W\times 3}$ denote the τ-th frame, $\tau \in \{1,\dots,T\}$. We then construct a hybrid reference video $\tilde{\mathbf{V}}$ by replacing the first frame with the input:

$$\tilde{\mathbf{V}}_{\tau} = \begin{cases} \mathbf{x}, & \tau = 1, \\ \mathbf{V}^{\mathrm{up}}_{\tau}, & \tau = 2,\dots,T. \end{cases} \tag{3}$$

This first-frame replacement injects the input image as a boundary condition to reduce drift and first-frame mismatch compared with traditional VSR pipelines, while frames $\mathbf{V}^{\mathrm{up}}_{2:T}$ preserve Stage I motion and structure as a stable reference for Stage II.

Stage II: High-Resolution Video Synthesis. Stage II focuses on synthesizing input-faithful high-frequency details conditioned on the Stage I motion reference and the input appearance constraint. Since it does not need to re-model motion from scratch, a smaller DiT backbone is sufficient, allowing its limited capacity to be devoted to high-frequency detail synthesis rather than motion modeling. To further reduce the number of tokens at high resolution, Stage II adopts a 3D VAE with higher downsampling factors $(16, 16, 4)$ (Wan et al., 2025); Appd. C.8 validates its 2K reconstruction fidelity. Let $\mathcal{E}^{\mathrm{HR}}$ and $\mathcal{D}^{\mathrm{HR}}$ denote the VAE encoder and decoder. We encode the hybrid reference video $\tilde{\mathbf{V}}$ and the input image $\mathbf{x}$ as

$$\mathbf{z}_{\mathrm{ref}} = \mathcal{E}^{\mathrm{HR}}(\tilde{\mathbf{V}}), \qquad \mathbf{z}_x = \mathcal{E}^{\mathrm{HR}}(\mathbf{x}), \tag{4}$$

where $\mathbf{z}_{\mathrm{ref}} \in \mathbb{R}^{t\times h\times w\times c}$ and $\mathbf{z}_x \in \mathbb{R}^{h\times w\times c}$. Here (t, h, w) are the latent spatiotemporal dimensions.

During denoising step k, let $\mathbf{z}_k \in \mathbb{R}^{t\times h\times w\times c}$ be the noisy latent, and write it along the temporal axis as t blocks $\mathbf{z}_k = (\mathbf{z}_{k,1},\dots,\mathbf{z}_{k,t})$, $\mathbf{z}_{k,i} \in \mathbb{R}^{h\times w\times c}$. Since the 3D VAE encodes the first frame separately during encoding, we further anchor the high-resolution appearance information by replacing the first block of the noisy latent $\mathbf{z}_k$ with $\mathbf{z}_x$:

$$\bar{\mathbf{z}}_{k,i} = \begin{cases} \mathbf{z}_x, & i = 1, \\ \mathbf{z}_{k,i}, & \text{otherwise}. \end{cases} \tag{5}$$

and concatenate $\bar{\mathbf{z}}_k$ with $\mathbf{z}_{\text{ref}}$ along the channel dimension to construct the Stage II DiT input:

$$\mathbf{u}_k = \text{Concat}_c(\bar{\mathbf{z}}_k,\, \mathbf{z}_{\text{ref}}) \in \mathbb{R}^{t\times h\times w\times 2c}. \tag{6}$$

Here, $\mathbf{z}_x$ acts as an explicit appearance anchor, while $\mathbf{z}_{\text{ref}}$ provides motion cues and structural appearance information. We then denoise $\mathbf{z}_k$ to obtain $\mathbf{z}_0$ in combination with our Conditional Segment-wise strategy. Finally, the high-resolution video is decoded as: $\hat{\mathbf{V}} = \mathcal{D}^{\text{HR}}(\mathbf{z}_0)$.

3.2 Conditional Segment-wise Generation (CSG)

Even with a highly compressed VAE, 2K Stage II still has many visual tokens. Since the input image and Stage I reference already provide global structure and dynamics, Stage II mainly needs to recover high-frequency details with smooth temporal transitions. We therefore introduce CSG, which denoises high-resolution latents in short temporal segments with bounded per-step token budgets. The term *conditional* emphasizes native 2K synthesis under the input-image anchor and Stage I motion reference, rather than low-resolution upsampling.

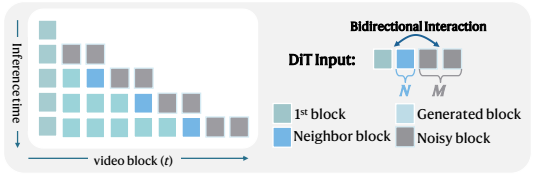

Figure 3: **Conditional Segment-wise Generation of SwiftI2V.** SwiftI2V adopts a CSG strategy in Stage II. To ensure fidelity and mitigate error accumulation, SwiftI2V allows bidirectional interaction between conditioning blocks and noisy blocks.

Temporal Block and Segment-level Windows. Following Eq. (6), the DiT input at diffusion step k is $\mathbf{u}_k = \text{Concat}_c(\bar{\mathbf{z}}_k, \mathbf{z}_{\text{ref}})$, where $\bar{\mathbf{z}}_k$ is the noised high-resolution latent sequence and $\mathbf{z}_{\text{ref}}$ is the hybrid reference latent sequence. Split $\mathbf{u}_k$ along time into t blocks:

$$\mathbf{u}_k = (\mathbf{u}_{k,1}, \mathbf{u}_{k,2}, \ldots, \mathbf{u}_{k,t}), \qquad \mathbf{u}_{k,i} \in \mathbb{R}^{h\times w\times 2c}. \tag{7}$$

By the substitutions in Eq. (3) and Eq. (5), the first temporal block $\mathbf{u}_{k,1}$ is anchored to the HR input image, which we denote as the *anchor block*. Blocks $\{\mathbf{u}_{k,2}, \ldots, \mathbf{u}_{k,t}\}$ are to be generated. CSG aims to inject high-fidelity details consistent with the HR input on top of the motion cues from $\mathbf{z}_{\text{ref}}$, while keeping the per-step token budget bounded.

We partition the target indices $\{2, \ldots, t\}$ into S consecutive, non-overlapping segments, each containing M noisy blocks. Define

$$S \triangleq \left\lceil \frac{t-1}{M} \right\rceil, \qquad a_s \triangleq 2 + (s-1)M,$$

and the noisy-block index set of segment s as

$$\mathcal{I}_s \triangleq \{a_s, a_s+1, \ldots, min(a_s+M-1, t)\}, \qquad s \in \{1, \ldots, S\}. \tag{8}$$

To promote cross-segment continuity, we additionally include a short context consisting of the last N blocks immediately preceding the current segment. Specifically, we define the neighbor index set

$$\mathcal{N}_s \triangleq \begin{cases} \emptyset, & s = 1, \\ \{\max(2, a_s - N),\ \ldots,\ a_s - 1\}, & s > 1. \end{cases} \tag{9}$$

We then construct the segment-wise temporal window $\mathcal{W}_s \triangleq \{1\} \cup \mathcal{N}_s \cup \mathcal{I}_s$, and feed the gathered subsequence into DiT at each diffusion step, as shown in Fig. 3:

$$\mathbf{u}_k^{(s)} \triangleq \left(\mathbf{u}_{k,i}\right)_{i\in\mathcal{W}_s} \in \mathbb{R}^{|\mathcal{W}_s|\times h\times w\times 2c}. \tag{10}$$

During inference, segments are processed sequentially; once segment $\mathcal{I}_s$ finishes diffusion, we can decode its frames and cache the last N blocks as $\mathcal{N}_{s+1}$, enabling segment-wise streaming output.

Bidirectional Contextual Interaction. A key design choice is how conditioning blocks (the *anchor block* and the *neighbor blocks*) are used within the window. Some streaming methods (Qiu et al., 2025a; Yin et al., 2025; ai et al., 2025; Deng et al., 2024) use an auto-regressive (AR) formulation, where previous blocks serve as fixed read-only context for the current blocks. While this controls the token budget, the rigid dependence on imperfect history can cause boundary artifacts and error accumulation in high-fidelity I2V.

To better preserve input-image fidelity and to mitigate segment-wise degradation, CSG introduces a bidirectional contextual interaction strategy within each window $\mathcal{W}_s$: we apply standard attention over $\mathbf{u}_k^{(s)}$, so that the anchor block, the neighbor blocks, and the current noisy blocks all attend to each other bidirectionally within the window. *As a result, conditioning blocks are not merely static providers of features, but actively participate in the attention computation together with the current noisy blocks.* This lets the context be dynamically reorganized to match the denoising needs of the current segment, facilitating the fusion of anchored HR appearance and reference motion cues and mitigating cascading error accumulation across segments.

Crucially, bidirectional interaction is only used for feature interaction and does not alter previously finalized latents. Although DiT produces predictions for all blocks inside $\mathcal{W}_s$, we only apply the update to the current segment $\mathcal{I}_s$ and never write back updates to the conditioning latents:

$$\bar{\mathbf{z}}_{k-1,i} = \begin{cases} \mathrm{Update}\Big(\bar{\mathbf{z}}_{k,i};\ \mathrm{DiT}_\theta(\mathbf{u}_k^{(s)})_i\Big), & i \in \mathcal{I}_s, \\ \bar{\mathbf{z}}_{k,i}, & \text{otherwise.} \end{cases} \tag{11}$$

Generated historical segments remain immutable as input, while still providing stronger, more adaptive interactive features inside the attention in the DiT forward pass.

Overall, the design of CSG aligns with the refinement objective in Stage II—recovering input-faithful details while maintaining smooth temporal transitions. Meanwhile, it bounds per-step high-resolution token budget, improving the model's scalability, and enables segment-wise decoding for low-latency, streaming outputs (Appd. C.5 provides an exploratory deployment study.)

To mitigate train–test mismatch, we train the Stage II model using CSG. We use teacher-forcing (Williams & Zipser, 1989) for the conditioning blocks, which are taken from the ground-truth training video, and compute the diffusion loss only on the noisy blocks indexed by $\mathcal{I}_s$. This trains the model to exploit the anchored HR appearance and short-range context, on top of the reference guidance $\mathbf{z}_{\mathrm{ref}}$, to generate high-fidelity segments.

3.3 Stage-Transition Training

Training the two stages separately keeps each stage lightweight and tractable at 2K resolution, but it introduces an interface gap that is common to separately trained cascades: Stage II is trained with "clean" LR inputs (downsampled HR videos), whereas at inference it consumes Stage I outputs that may contain generation artifacts caused by VAE distortion or low-resolution flickering, leading to error amplification. To close this gap without re-introducing motion modeling into Stage II, we synthesize Stage II inputs by lightly corrupting downsampled clips and denoising them with Stage I.

For a training clip V^{HR} (with first frame as input image x), we construct

$$V^{LR}_{\mathrm{noisy}} \triangleq (1-\sigma)\mathrm{Down}(V^{HR}) + \sigma\epsilon, \qquad \epsilon \sim \mathcal{N}(0, I),$$

and $x^{LR} \triangleq \mathrm{Down}(x)$. Then we denoise it using the Stage I model,

$$\tilde{V}^{LR} \triangleq \mathrm{Denoise}_{G_1}\left(V^{LR}_{\mathrm{noisy}};\ x^{LR}\right). \tag{12}$$

The synthesized $\tilde{V}^{LR}$ preserves ground-truth motion while inheriting Stage I–style artifacts, better matching Stage II's inference inputs and maintaining reliable motion–appearance supervision. We pair $\tilde{V}^{LR}$ with the original V^{HR} and train Stage II on $(\tilde{V}^{LR}, V^{HR})$. This reduces the stage-to-stage gap and stabilizes inference with the separately trained stages (Appd. C.7).

4 Experiments

4.1 Experimental Setup

Unless stated otherwise, all experiments are conducted on NVIDIA H800 GPUs. Our default generation setting produces 81-frame videos at 2K resolution (2560×1408), where **Stage I** generates a low-resolution video at 360P (640×352), and **Stage II** synthesizes the final 2K result.

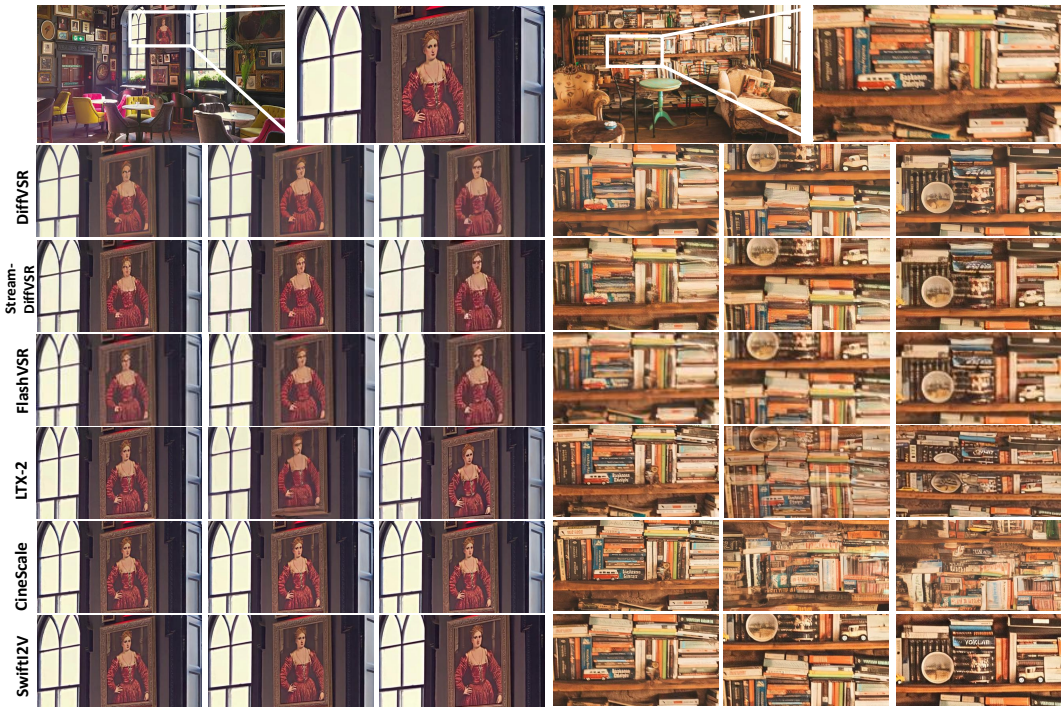

Figure 4: Qualitative comparison between SwiftI2V and representative baselines on 2K I2V generation. Best viewed **zoomed in**.

Implementation Details. For Stage I, we adopt Wan2.1-I2V-480P (Wan et al., 2025) as the backbone. We train a LoRA to perform I2V generation at 360P. At inference time, we additionally load an existing few-step LoRA (LightX2V, 2025) to accelerate sampling, enabling 4-step generation in Stage I. For Stage II, we adopt Wan2.2-TI2V-5B (Wan et al., 2025) as the backbone and fully fine-tune its DiT for high-resolution video synthesis. We observe in the experiment that 4-step inference already yields stable and competitive results for this refinement stage, and we therefore use it as the default. For Conditional Segment-wise Generation (CSG), we use $M{=}3$ and $N{=}1$ for inference; see Appd. C.6 for the experiment that motivates this choice. Since 2K training data is relatively limited, we employ a curriculum strategy: we first train with 1080P videos from OpenViD-HD (Nan et al., 2024), then continue for 10K optimization steps in the final 2K stage on a pool of 2K videos from UltraVideo (Xue et al., 2025), mixed with our synthesized samples.

Evaluation Metrics. We use VBench-I2V (Huang et al., 2025) as our primary evaluation suite. It measures I2V-specific fidelity (*e.g.*, *i2v subject* and *i2v background*) as well as general video quality metrics. I2V generation is conditioned on both the input image and text, where the text prompts are taken from the official VBench-I2V prompt set. We also report runtime and GPU memory efficiency in Section 4.3 to validate different pipelines' practicality.

Human Study. To complement the automatic metrics, we conduct a small-scale blinded user study. Multiple raters score six anonymized methods on a 1–3 scale (higher is better) in four dimensions: I2V Fidelity, HR Detail Preservation, Temporal Stability, and Overall Quality (reported as Overall Score). Tab. 2 reports means of 150 valid ratings per method and dimension.

4.2 Comparison with High-Resolution I2V Methods

Baselines. We compare SwiftI2V with CineScale (Qiu et al., 2025b), a single-model end-to-end 2K generator; LTX-2 (HaCohen et al., 2026), an efficient audio–visual foundation model supporting 2K I2V; and three VSR pipelines: DiffVSR (Li et al., 2025), Stream-DiffVSR (Shiu et al., 2025), and FlashVSR (Zhuang et al., 2026). All VSR baselines upscale the same SwiftI2V Stage I outputs to 2K, controlling the low-resolution input.

Table 2: **Automatic and human evaluation at 2K.** Higher is better (↑); human ratings use a 1–3 scale. **Bold**/underline: best/second best; "–": unavailable.

Model	VBench-I2V			Human Study			
	I2V Subj.	I2V Backg.	Total	Fidelity	Detail	Stability	Overall
Upscale	0.9881	0.9946	6.4173	–	–	–	–
DiffVSR	0.9878	0.9907	6.4228	2.73	2.12	2.11	2.06
Stream-DiffVSR	0.9879	0.9933	6.4240	2.75	1.69	2.32	1.83
FlashVSR	0.9845	0.9901	**6.4382**	2.83	2.41	2.80	2.53
LTX–2	0.9914	0.9932	6.3579	2.17	1.96	1.69	1.21
CineScale[†]	**0.9924**	0.9973	6.3638	2.39	2.41	2.37	1.93
SwiftI2V (ours)	0.9910	**0.9975**	6.4244	**2.90**	**2.59**	**2.89**	**2.69**

[†] CineScale's VBench-I2V scores are evaluated on a random subset due to high computational cost. Appd. C.2 reports the matched-subset comparison.

Automatic Evaluation. Tab. 2 summarizes the 2K VBench-I2V results. SwiftI2V achieves a competitive Total Score (6.4244) and the highest I2V Background score (0.9975), while maintaining competitive I2V Subject fidelity (0.9910). Compared with SwiftI2V, VSR pipelines show weaker I2V faithfulness, especially background preservation, indicating that post-hoc super-resolution is less reliable at recovering input-specific structures under strong image conditioning. CineScale and LTX-2 also obtain high I2V scores, which may partly reflect their limited motion: smaller deviations from the conditioning image make subject and background appearance easier to preserve. This interpretation is supported by their lower Dynamic Degree in Fig. 7 (Appd. C.1): 0.1667 and 0.0488, respectively, compared with 0.3008 for SwiftI2V. Their lower Total Scores also reflect this fidelity–dynamics trade-off, whereas SwiftI2V maintains strong input fidelity alongside more substantial motion. As shown in Fig. 4, SwiftI2V preserves higher fidelity and finer details than the visualized VSR baselines, while the strong generative capacity of Stage I enables plausible and coherent dynamics.

Human Evaluation. In Tab. 2, SwiftI2V has the highest descriptive mean among the six methods for I2V Fidelity, HR Detail, Temporal Stability, and Overall Score. Compared with other methods, SwiftI2V retains the most accurate details and meets the requirements of high-resolution I2V. FlashVSR is the closest competitor, but its slightly higher VBench-I2V Total does not translate into higher mean human ratings. These results complement the automatic metrics and support SwiftI2V's balance of input fidelity, high-resolution detail, and temporal stability.

4.3 Efficiency and Memory Analysis

Settings. We further evaluated inference efficiency and memory consumption under our experiment settings. Tab. 3 reports the *total wall-clock time* to obtain the final 81-frame 2K video. For VSR pipelines (DiffVSR, Stream-DiffVSR, and FlashVSR), the reported time includes both the base video generation stage (our Stage I) and the subsequent super-resolution stage.

Results. As shown in Tab. 3, FlashVSR is the fastest method at 96s, followed by SwiftI2V at 111s on a single GPU. While SwiftI2V takes 15s longer, it achieves higher I2V Subject and Background scores and higher mean human ratings across all four dimensions. Compared with end-to-end high-resolution I2V methods, SwiftI2V offers a clear computational efficiency advantage. Specifically, CineScale requires 4 H800 GPUs—the minimum runnable configuration in our tests—and 5600s, so SwiftI2V still reduces aggregate GPU-time (#GPUs $\times$ time) by 202$\times$ relative to CineScale (111 vs. 22400 GPU·s). It is also 1.37$\times$ faster than LTX-2 (111s vs. 152s).

Tab. 3 further reports the *peak GPU memory* measured over the entire generation pipeline. Without any further memory optimizations like model off-loading or quantization, SwiftI2V has the lowest peak memory usage during inference, which is 33.5GB and occurs in Stage I, while Stage II peaks at only 20.8GB. This indicates that CSG effectively controls the memory consumption of the 2K generation stage. Moreover, by incorporating existing memory-saving inference techniques (ModelScope, 2024) for Stage I, SwiftI2V can be deployed on a single consumer-grade GPU (*e.g.*, RTX 4090); detailed configurations and results are provided in Appd. C.4. This substantially lowers the hardware barrier for practical 2K I2V generation.

Table 3: **Efficiency and peak memory** for generating an 81-frame 2K video.

Model	#GPUs	Peak mem.	Time
DiffVSR	1	58.1GB	3386s
Stream-DiffVSR	1	49.2GB	141s
FlashVSR	1	49.7GB	**96s**
LTX-2	1	45.7GB	152s
CineScale	4	42.7GB	5600s
SwiftI2V	**1**	**33.5GB**	111s

Table 4: **Ablation results**. All methods share Stage I; we report Stage II peak memory and DiT time.

Method	Stream infer.	Peak mem.↓	DiT time↓	I2V sub.↑	I2V back.↑	Total score↑
w/o CSG	✗	22.1GB	30s	0.991	0.998	6.414
w/o bi-inter.	✓	23.2GB	16s	0.989	0.996	6.392
w/o stage-trans	✓	20.8GB	25s	0.989	0.996	6.398
SwiftI2V	✓	**20.8GB**	**25s**	**0.991**	**0.998**	**6.424**

4.4 Ablation Studies

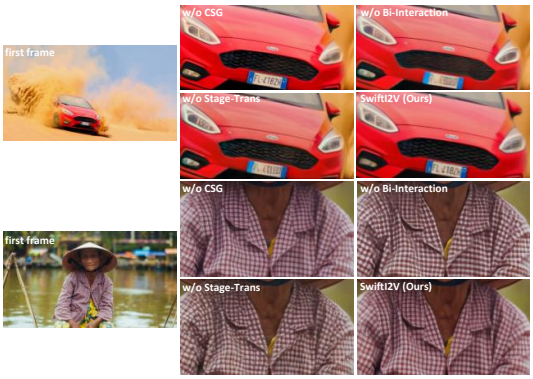

Figure 5: **Ablation studies.** We show qualitative comparisons between different methods.

We ablate SwiftI2V's core components. Unless otherwise specified, all variants are re-trained under their corresponding training protocols with the same Stage-II backbone and optimization budget. Since Stage I is fixed across ablations, Tab. 4 reports Stage II peak memory and DiT runtime, with qualitative results in Fig. 5.

Conditional Segment-wise Generation (CSG). CSG is designed to keep Stage II computation and memory within a fixed-size window, enabling streaming inference and improving computational scalability for longer videos. In contrast, `w/o CSG` removes segmentation and uses full-window attention (*i.e.*, $N{=}0$, $M{=}t{-}1$), which therefore does *not* support streaming output. Tab. 4 shows that CSG reduces runtime and peak memory while preserving I2V Subject/Background fidelity and slightly improving total quality. Beyond efficiency and quality, CSG also preserves temporal smoothness across segment boundaries (Appd. C.9) and improves computational scalability to longer videos (Appd. C.3).

Bidirectional Interaction. We ablate the bidirectional interaction using `w/o bi-interaction`, where we enforce a causal mask such that the information flow becomes unidirectional, consistent with standard auto-regressive (AR) formulations (Qiu et al., 2025a; Yin et al., 2025). We also enable *key–value (KV) cache* for this AR variant as a standard acceleration strategy. As shown in Tab. 4, `w/o bi-interaction` runs faster but uses more peak memory (due to caching), and it yields a clear drop in both I2V fidelity and overall quality. We attribute this degradation to weaker cross-segment alignment under the causal AR masking, which tends to amplify segmentation-induced discontinuities and error accumulation (Fig. 5; see Appd. C.10 for a detailed analysis). In contrast, bidirectional interaction enables more adaptive contextual alignment and improves temporal continuity and fidelity.

Stage Transition Training. We further ablate the stage transition training (*cf.*, Section 3.3) with `w/o stage-trans`. Removing this strategy consistently degrades quantitative results. Fig. 5 shows that Stage I artifacts persist after refinement, confirming that transition training improves robustness to stage-interface mismatches.

In summary, our ablations validate SwiftI2V's synergistic design: **CSG** enables bounded-cost streaming inference, **bidirectional interaction** maintains input-conditioned fidelity, and **stage transition training** improves two-stage robustness and the quality of generation.

5 Conclusion

We presented **SwiftI2V** to address the efficiency–fidelity dilemma in 2K I2V generation. By decoupling motion modeling from detail synthesis via Conditional Segment-wise Generation (CSG), it avoids the prohibitive costs of end-to-end models. SwiftI2V achieves competitive quality while reducing GPU-time by over 200× compared to a full-sequence baseline, and lowers the hardware barrier to consumer-grade GPUs. This efficient segment-based paradigm offers a promising direction toward scalable, long-duration, and interactive generative video.

AI USE STATEMENT

Generative AI tools were used to assist with language editing and improving the readability of the manuscript, to help draft preliminary versions of figures and visualizations, and to provide limited assistance with code development and debugging. All AI-assisted figures and code were reviewed, modified, and verified by the authors. Generative AI tools were not used to formulate the main research ideas, design the methodology or experiments, interpret experimental results, or draw scientific conclusions. The authors take full responsibility for the final content of this work.

REFERENCES

Sand. ai, 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.

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models, 2023. URL https://arxiv.org/abs/2304.08818.

Haoge Deng, Ting Pan, Haiwen Diao, Zhengxiong Luo, Yufeng Cui, Huchuan Lu, Shiguang Shan, Yonggang Qi, and Xinlong Wang. Autoregressive video generation without vector quantization. *arXiv preprint arXiv:2412.14169*, 2024.

Lanqing Guo, Yingqing He, Haoxin Chen, Menghan Xia, Xiaodong Cun, Yufei Wang, Siyu Huang, Yong Zhang, Xintao Wang, Qifeng Chen, et al. Make a cheap scaling: A self-cascade diffusion model for higher-resolution adaptation. In *European conference on computer vision*, pp. 39–55. Springer, 2024.

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

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

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

Ziqi Huang, Fan Zhang, Xiaojie Xu, Yinan He, Jiashuo Yu, Ziyue Dong, Qianli Ma, Nattapol Chanpaisit, Chenyang Si, Yuming Jiang, Yaohui Wang, Xinyuan Chen, Ying-Cong Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench++: Comprehensive and versatile benchmark suite for video generative models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025. doi: 10.1109/TPAMI.2025.3633890.

Tencent Hunyuan. Hunyuanvideo 1.5 technical report, 2025. URL https://arxiv.org/abs/2511.18870.

Xiaohui Li, Yihao Liu, Shuo Cao, Ziyan Chen, Shaobin Zhuang, Xiangyu Chen, Yinan He, Yi Wang, and Yu Qiao. Diffvsr: Revealing an effective recipe for taming robust video super-resolution against complex degradations, 2025. URL https://arxiv.org/abs/2501.10110.

LightX2V. Lightx2v: Light video generation inference framework. https://github.com/ModelTC/lightx2v, 2025.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

ModelScope. Diffsynth-studio: An open-source diffusion model engine, 2024. URL https://github.com/modelscope/DiffSynth-Studio. GitHub repository.

Kepan Nan, Rui Xie, Penghao Zhou, Tiehan Fan, Zhenheng Yang, Zhijie Chen, Xiang Li, Jian Yang, and Ying Tai. Openvid-1m: A large-scale high-quality dataset for text-to-video generation. *arXiv preprint arXiv:2407.02371*, 2024.

William Peebles and Saining Xie. Scalable diffusion models with transformers, 2023. URL https://arxiv.org/abs/2212.09748.

Haonan Qiu, Shikun Liu, Zijian Zhou, Zhaochong An, Weiming Ren, Zhiheng Liu, Jonas Schult, Sen He, Shoufa Chen, Yuren Cong, et al. Histream: Efficient high-resolution video generation via redundancy-eliminated streaming. *arXiv preprint arXiv:2512.21338*, 2025a.

Haonan Qiu, Ning Yu, Ziqi Huang, Paul Debevec, and Ziwei Liu. Cinescale: Free lunch in high-resolution cinematic visual generation, 2025b. URL https://arxiv.org/abs/2508.15774.

Jingjing Ren, Wenbo Li, Zhongdao Wang, Haoze Sun, Bangzhen Liu, Haoyu Chen, Jiaqi Xu, Aoxue Li, Shifeng Zhang, Bin Shao, et al. Turbo2k: Towards ultra-efficient and high-quality 2k video synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 18155–18165, 2025.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Xiaoyu Shi, Zhaoyang Huang, Fu-Yun Wang, Weikang Bian, Dasong Li, Yi Zhang, Manyuan Zhang, Ka Chun Cheung, Simon See, Hongwei Qin, et al. Motion-i2v: Consistent and controllable image-to-video generation with explicit motion modeling. In *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–11, 2024.

Hau-Shiang Shiu, Chin-Yang Lin, Zhixiang Wang, Chi-Wei Hsiao, Po-Fan Yu, Yu-Chih Chen, and Yu-Lun Liu. Stream-diffvsr: Low-latency streamable video super-resolution via auto-regressive diffusion. *arXiv preprint arXiv:2512.23709*, 2025.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Jianyi Wang, Shanchuan Lin, Zhijie Lin, Yuxi Ren, Meng Wei, Zongsheng Yue, Shangchen Zhou, Hao Chen, Yang Zhao, Ceyuan Yang, Xuefeng Xiao, Chen Change Loy, and Lu Jiang. Seedvr2: One-step video restoration via diffusion adversarial post-training, 2025a. URL https://arxiv.org/abs/2506.05301.

Xiang Wang, Shiwei Zhang, Han Zhang, Yu Liu, Yingya Zhang, Changxin Gao, and Nong Sang. Videolcm: Video latent consistency model. *arXiv preprint arXiv:2312.09109*, 2023.

Zhongdao Wang, Guodongfang Zhao, Jingjing Ren, Bailan Feng, Shifeng Zhang, and Wenbo Li. Turbovsr: Fantastic video upscalers and where to find them, 2025b. URL https://arxiv.org/abs/2506.23618.

Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

Rui Xie, Yinhong Liu, Penghao Zhou, Chen Zhao, Jun Zhou, Kai Zhang, Zhenyu Zhang, Jian Yang, Zhenheng Yang, and Ying Tai. Star: Spatial-temporal augmentation with text-to-video models for real-world video super-resolution, 2025. URL https://arxiv.org/abs/2501.02976.

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

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024.

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 *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 22963–22974, 2025.

Shilong Zhang, Wenbo Li, Shoufa Chen, Chongjian Ge, Peize Sun, Yida Zhang, Yi Jiang, Zehuan Yuan, Binyue Peng, and Ping Luo. Flashvideo: Flowing fidelity to detail for efficient high-resolution video generation, 2025. URL https://arxiv.org/abs/2502.05179.

Shiwei Zhang, Jiayu Wang, Yingya Zhang, Kang Zhao, Hangjie Yuan, Zhiwu Qin, Xiang Wang, Deli Zhao, and Jingren Zhou. I2vgen-xl: High-quality image-to-video synthesis via cascaded diffusion models. *arXiv preprint arXiv:2311.04145*, 2023.

Shangchen Zhou, Peiqing Yang, Jianyi Wang, Yihang Luo, and Chen Change Loy. Upscale-a-video: Temporal-consistent diffusion model for real-world video super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2535–2545, 2024.

Junhao Zhuang, Shi Guo, Xin Cai, Xiaohui Li, Yihao Liu, Chun Yuan, and Tianfan Xue. Flashvsr: Towards real-time diffusion-based streaming video super resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 43482–43493, 2026.

APPENDIX

A Limitations and Broader Impacts 13
B More Implementation Details 14
B.1 Training Details 14
B.2 Inference Details 15
C Additional Experimental Results 15
— Benchmark Results —
C.1 Full VBench-I2V Results 15
C.2 Subset Evaluation Results 15
— Extended Settings and Applications —
C.3 Scalability Analysis 17
C.4 Experiments on Consumer GPUs 17
C.5 Streaming Generation with CSG 18
— In-depth Analysis —
C.6 Ablation on Segmentation Hyperparameters M and N 19
C.7 Analysis of Stage Transition 19
C.8 VAE Reconstruction Fidelity 20
C.9 Temporal Smoothness across Segment Boundaries 22
C.10 Analysis of Cross-Segment Error Accumulation 22
C.11 Analysis of Dynamic Degree 23
D More 2K I2V Visual Results of SwiftI2V 24

A LIMITATIONS AND BROADER IMPACTS

Limitations. SwiftI2V substantially improves the efficiency and deployability of 2K I2V generation, and its Conditional Segment-wise Generation further supports streaming generation. However, it still does not achieve strict real-time synthesis. Reaching real-time 2K video generation remains an important direction that may require further advances in model compression, caching, scheduling, and hardware-aware inference.

Another practical consideration is system integration. SwiftI2V deliberately decouples low-resolution motion generation and high-resolution detail synthesis, allowing each stage to use a backbone and resolution setting suited to its role. This modular design improves scalability and fidelity, but deploying two specialized stages can be more involved than serving a single monolithic model, especially when optimizing memory sharing, batching, and model loading in production systems. We view this as an engineering trade-off for efficient high-resolution I2V, and future work can simplify deployment through unified distillation, shared components, or integrated inference runtimes.

Dependence on the Motion Reference. The benchmark results in Appd. C.1 indicate that Stage I can provide an effective low-resolution motion reference, but aggregate performance does not exclude individual motion failures. Fig. 6 illustrates two such cases: a motorcycle exhibits reversed motion, and a tram remains nearly static despite the intended travel. Both issues are already present in the Stage-I outputs and persist in the final high-resolution videos. Since Stage II synthesizes appearance details conditioned on the supplied motion reference, it does not reliably correct incorrect or missing global motion; these examples illustrate a limitation of the complete two-stage pipeline rather than motion errors introduced by high-resolution refinement.



Figure 6: **Dependence on the Stage-I motion reference.** (a) Reversed motorcycle motion. (b) Insufficient tram motion. Each case compares the Stage-I reference (360P) with the final Stage-II output (2K) at corresponding frames, ordered from left to right. The motion failures are present in the low-resolution references and remain in the final outputs.

Broader Impacts. By reducing the compute and memory cost of high-resolution I2V synthesis, SwiftI2V may broaden access to video creation and research tools, enable faster prototyping and visualization workflows, and reduce the energy cost per generated sample compared with less efficient high-resolution pipelines. These benefits are most relevant for responsible applications such as content creation, education, design, and research.

At the same time, more efficient I2V generation may also increase risks associated with synthetic media, including deceptive or non-consensual content, impersonation, misinformation, or biased generations. Because the method is conditioned on an input image, responsible deployment should include consent-aware use policies for identifiable people, clear disclosure of synthetic content, provenance or watermarking mechanisms when applicable, and downstream abuse monitoring.

B More Implementation Details

B.1 Training Details

All experiments use the DiffSynth-Studio (ModelScope, 2024) framework. Bicubic interpolation handles spatial preprocessing and low-resolution video construction. For both stages, we optimize *only* the DiT; other components in the pipeline (*e.g.*, VAE and text encoder) are kept frozen.

Stage I: Concretely, 360P-LoRA (Hu et al., 2022) is inserted into the DiT attention projections and MLP layers with rank $r = 128$. We optimize with AdamW (Loshchilov & Hutter, 2017) using learning rate $\mathrm{lr} = 1 \times 10^{-4}$ and a constant learning-rate schedule.

Stage II: We train on fixed 2560×1408 videos with 81 frames. To incorporate the low-resolution reference via channel-wise concatenation, we expand the DiT patch-embedding input channels from 48 to 96. The expanded patch embedding is initialized by copying pretrained weights for the first 48 channels and zero-initializing the additional 48 channels, which stabilizes training at the stage-transition interface. We fully fine-tune the DiT with AdamW using learning rate $\mathrm{lr} = 5 \times 10^{-6}$ and a constant learning-rate schedule. For robustness to different segmentations, we randomize the segment hyperparameters by sampling (M, N) uniformly from the four combinations $M \in \{2, 3\}$ and $N \in \{1, 2\}$ at each iteration. Following Section 3.3, we train Stage II using pre-generated stage-transition samples, and mix them with downsampled counterparts at a ratio of $7 : 3$ (generated:downsampled).

B.2 Inference Details

Stage I: During inference, we load two LoRA adapters into the DiT of Wan2.1-I2V-14B-480P (Wan et al., 2025) with $\alpha = 1$: a few-step distillation LoRA (LightX2V, 2025) and our 360P-LoRA. We sample at 360P (640×352) resolution with 4 denoising steps and without CFG.

Stage II: For Stage II inference, we apply CSG with fixed $(M, N) = (3, 1)$. After VAE encoding, the latent sequence contains 21 temporal blocks; the first block corresponds to the input-image anchor and is kept fixed, while the remaining 20 blocks are denoised. We partition these 20 blocks into 7 sequential segments (6 segments of length 3 and a shorter final segment). For each segment, we run 4 denoising steps without CFG. To further accelerate inference and reduce peak memory, Stage II employs tiled VAE encoding/decoding.

C Additional Experimental Results

C.1 Full VBench-I2V results

Fig. 7 reports the complete VBench-I2V evaluation results of SwiftI2V and other baselines (DiffVSR (Li et al., 2025), Stream-DiffVSR (Shiu et al., 2025), LTX-2 (HaCohen et al., 2026), CineScale (Qiu et al., 2025b)) across both the total score and individual dimensions. The figure complements the three automatic metrics in Tab. 2 with motion and per-frame quality measurements for the visualized methods. SwiftI2V performs consistently well on I2V-related criteria and attains competitive performance on aesthetic-related metrics. SwiftI2V also achieves a high dynamic degree.

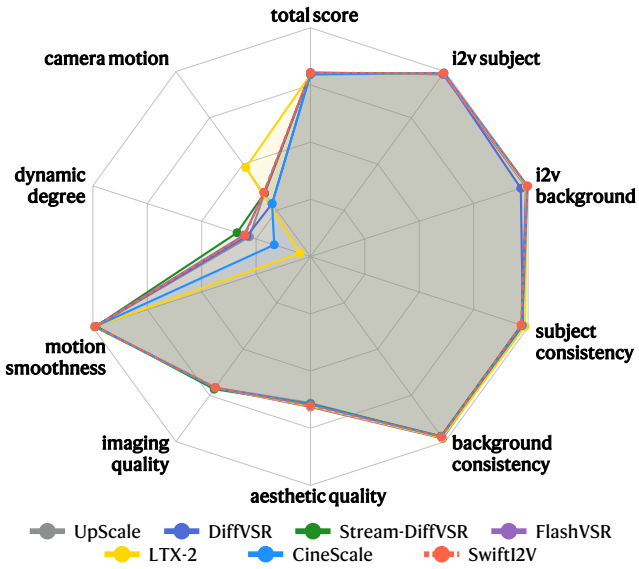

Figure 7: **VBench-I2V score visualization.** We show all dimension scores here.

C.2 Subset Evaluation Results

In the main comparison in Section 4.2, we include CineScale (Qiu et al., 2025b) as a representative end-to-end 2K I2V baseline. However, under the 2K resolution and 81-frame setting, CineScale incurs substantially higher inference cost than other methods: even with 8 H800 GPUs in parallel, generating a single sample takes approximately 50 minutes. Due to limited computational resources, we cannot afford to evaluate CineScale on the full official VBench-I2V test set (1118 samples). Therefore, we construct a randomly sampled subset for a feasible yet informative comparison.

Specifically, we randomly select a subset of 63 samples from the full test set. *While ensuring randomness, we also keep the ratio of samples associated with each evaluation dimension close to that of the original test set*, so as to reduce potential sampling bias. Except for using the subset instead of the full set, all evaluation protocols remain identical to those in the main paper.

Tab. 5 reports the VBench-I2V results of SwiftI2V and CineScale on this subset. SwiftI2V consistently maintains a high Dynamic Degree while outperforming CineScale on the overall score and all I2V-related metrics on this subset. These results further support the conclusions in the main paper and indicate that, compared with CineScale, SwiftI2V can generate 2K I2V videos more efficiently while achieving both strong motion dynamics and high input fidelity.

Table 5: Comparison of the results of CineScale and SwiftI2V on the VBench-I2V subset.

Model	Total Score↑	I2V Subject↑	I2V Background↑	Dynamic Degree↑
CineScale	6.3638	0.9924	0.9973	0.1667
SwiftI2V	**6.4284**	**0.9927**	**0.9983**	**0.2917**

To further validate the reliability of the performance improvements on this 63-sample subset, we conduct a sample-wise paired statistical significance test between SwiftI2V and CineScale. We use the Wilcoxon signed-rank test for continuous metrics (I2V Subject and I2V Background) and McNemar's exact test for the binary metric (Dynamic Degree). The results are summarized in Tab. 6.

Table 6: Statistical significance test between SwiftI2V and CineScale on the VBench-I2V subset.

Metric	SwiftI2V	CineScale	p-value
I2V Subject ↑	0.993 ± 0.005	0.992 ± 0.005	0.944
I2V Background ↑	$\mathbf{0.998 \pm 0.001}$	0.997 ± 0.002	0.042^{*}
Dynamic Degree ↑	$\mathbf{0.292 \pm 0.455}$	0.167 ± 0.373	0.375 ($g = 0.30$)

The results indicate that SwiftI2V achieves comparable image fidelity to CineScale (I2V Subject, $p = 0.944$) while demonstrating a statistically significant advantage in background consistency (I2V Background, $p < 0.05$). Regarding the Dynamic Degree, SwiftI2V increases the proportion of dynamic videos from 16.7% to 29.2%. Although McNemar's test does not reach statistical significance ($p = 0.375$) due to the limited number of discordant pairs, the effect size reaches Cohen's $g = 0.30$ (a large effect), indicating a practically meaningful improvement. This trend is also evident in our qualitative comparisons, where CineScale occasionally fails to generate any meaningful motion. Overall, these subset results support that SwiftI2V achieves a $202\times$ reduction in aggregate GPU-time (Tab. 3) without sacrificing image fidelity, while showing advantages in background consistency and motion dynamics.

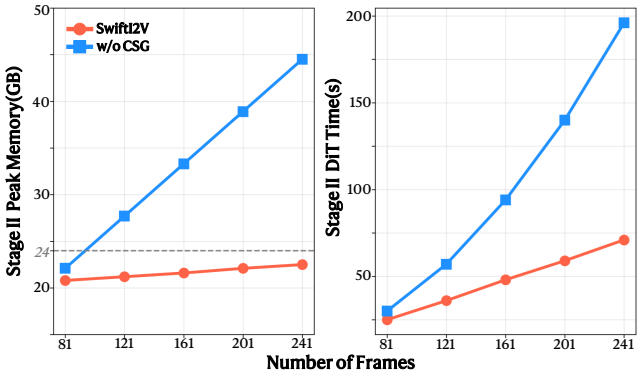

Figure 8: **Scaling behavior at 2K resolution.** Left: Stage II peak GPU memory vs. number of frames. Right: Stage II DiT generating time vs. number of frames. SwiftI2V keeps peak memory below 24 GB even at 241 frames and exhibits near-linear time growth, while removing CSG leads to rapidly increasing memory and runtime.

C.3 Scalability Analysis

This subsection evaluates different methods' scalability with respect to the target video length, with emphasis on how Conditional Segment-wise Generation (CSG) affects GPU memory and runtime. We fix the output resolution to 2560×1408 and vary the generated length $T \in \{81, 121, 161, 201, 241\}$ frames. All measurements are conducted on a single NVIDIA H800 GPU. We report the peak GPU memory footprint of Stage II and the DiT time of Stage II for clear comparison. We compare **SwiftI2V** against the variant that removes CSG (`w/o CSG`). The results are summarized in Fig. 8.

As shown in Fig. 8, CSG yields a highly controllable memory footprint: even at $T = 241$ frames, the Stage II peak memory remains below 24 GB. Moreover, the Stage II sampling time increases approximately linearly with T. This trend is consistent with CSG's design: under a fixed spatial resolution, CSG enforces a constant token budget per segment, so the overall computation is dominated by the number of segments S, leading to near-linear scaling in S (and thus in T).

In contrast, removing CSG requires full-temporal denoising over the entire sequence, for which the token count—and consequently the attention footprint and compute—grows with T. As a result, both the peak memory and runtime increase substantially faster as the target length increases, reflecting the unfavorable scaling of full-temporal denoising at 2K resolution.

Overall, these results highlight CSG as a key enabler for computationally scalable Stage-II refinement of longer sequences: it provides a stable memory bound and near-linear time scaling with the number of segments S, making Stage II inference feasible and predictable for substantially longer outputs.

C.4 Experiments on Consumer GPUs

To validate the practicality of SwiftI2V beyond datacenter GPUs, we further deploy our full pipeline on a single consumer-grade GPU, **NVIDIA RTX 4090 (24GB)**. All experiments are conducted under the DiffSynth-Studio (ModelScope, 2024) framework with the same default 2K I2V setting as in the main paper (*i.e.*, **2560**$\times$**1408** resolution and **81** frames).

For Stage I, we enable `cpu_offload` in DiffSynth-Studio to satisfy the memory constraint on a 24GB GPU, which allows the large motion backbone to run with acceptable peak VRAM usage. Thanks to our Conditional Segment-wise Generation (CSG), Stage II is *highly memory-controllable* with a bounded token budget. As a result, Stage II can be deployed **directly on RTX 4090**, while still maintaining stable 2K generation quality.

With the above settings, SwiftI2V completes a full high-quality 2K I2V sample in about **380s** on a single RTX 4090. (*Note: due to hardware/software constraints in our environment, we cannot enable* `flash_attn` *as an acceleration mechanism on RTX 4090. We expect the runtime can be further reduced with* `flash_attn` *or other optimized attention kernels.*) We also provide representative 2K I2V examples generated on RTX 4090 in Fig. 9.

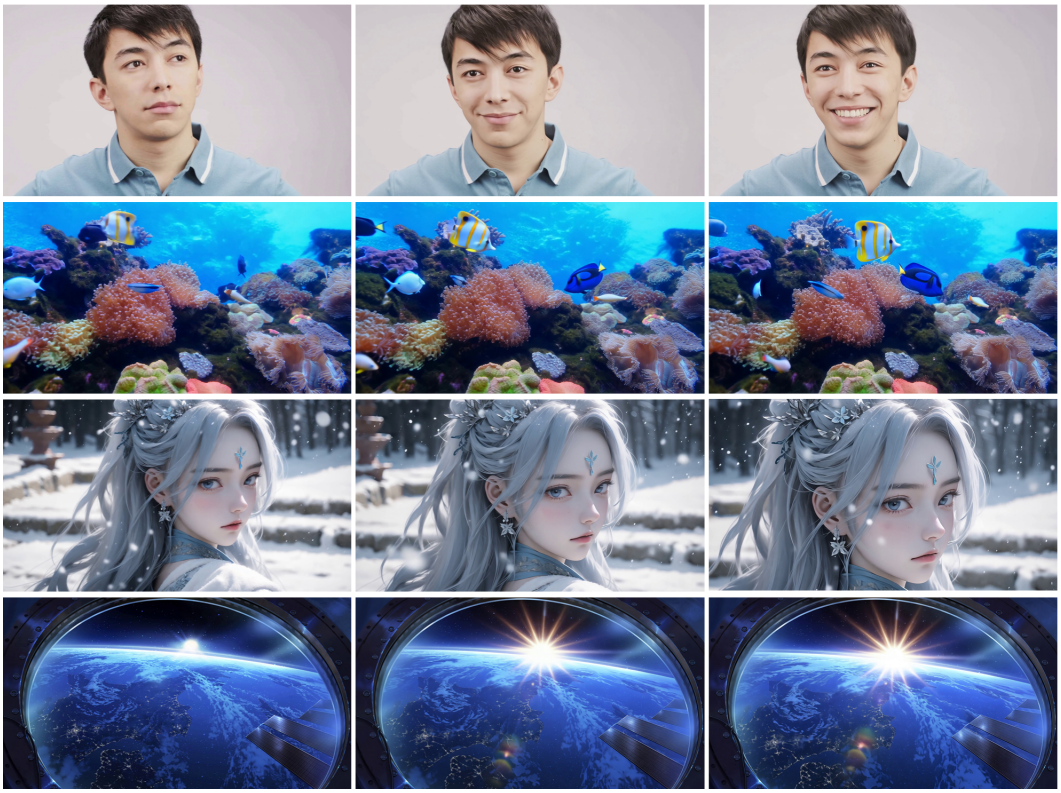

Figure 9: Representative 2K I2V samples generated by SwiftI2V **on a single NVIDIA RTX 4090 (24GB)**.

Overall, SwiftI2V substantially lowers the hardware barrier of practical high-resolution I2V generation and enables accessible 2K deployment in commodity settings.

C.5 Streaming Generation with CSG

This subsection supplements Section 3.2 with a deployment-oriented study of *streaming output* in Stage II. In this study, we assume the Stage I motion reference is available and focus on the Stage II *generation-to-output* path (DiT denoising of 2K latents $\rightarrow$ VAE decoding $\rightarrow$ video saving). We report *time-to-first-viewable-output* and the *incremental delivery rate*. These measurements provide an actionable baseline for building future low-latency 2K I2V systems on top of SwiftI2V.

In the standard sequential execution, Stage II completes DiT denoising before running VAE decoding and saving. Under our default setting, DiT denoising takes $25\,\mathrm{s}$ and VAE decoding takes $22\,\mathrm{s}$. Including video materialization overheads, the Stage-II wall-clock time is $\sim 50\,\mathrm{s}$, and viewable frames are produced only near the end.

To enable streaming output, we implement a simple pipeline that overlaps DiT denoising and VAE decoding at CSG segment boundaries. Since concurrent DiT+VAE execution on one GPU is impractical due to compute contention, we deploy DiT and VAE on two NVIDIA H800 GPUs. Once DiT finishes denoising a CSG segment, we transfer the corresponding latent segment to the decoder GPU and put it in a queue for decoding. Decoding and output are performed in blocks of 4 frames, each written out as soon as it becomes available.

As shown in Tab. 7, DiT produces the first segment latent after $7.4\,\mathrm{s}$, and **the first decoded 4-frame block becomes available after $8.5\,\mathrm{s}$** (Stage-II time-to-first-viewable-output). The full video finishes decoding and output at $\sim 30\,\mathrm{s}$, close to the ideal pipelining behavior where total time approaches "DiT latency + full VAE decoding." **During streaming, the system outputs one 4-frame block every $1.2\,\mathrm{s}$ on average, *i.e.*, an incremental delivery rate of ≈ 3.33 fps.**

Table 7: **Stage II streaming output enabled by CSG (Stage-II-only timing).** "First latent": time until DiT finishes the first CSG segment. "First output": time-to-first-viewable-output (first decoded 4-frame block). "Output FPS": average incremental delivery rate during streaming.

Setting	First latent	First output	Full output	Output FPS
Sequential	–	$\sim 50\,\mathrm{s}$	$\sim 50\,\mathrm{s}$	–
Streaming	$7.4\,\mathrm{s}$	$8.5\,\mathrm{s}$	$\sim 30\,\mathrm{s}$	3.33

Overall, these measurements substantiate the claim in Section 3.2 that CSG not only bounds the per-step 2K token budget for scalable refinement, but also enables low-latency, incremental delivery by making segment-wise decoding possible. While the demonstrated throughput is below real-time playback and currently requires a two-GPU deployment, the results provide a concrete baseline and suggest a promising direction for future low-latency 2K I2V systems (*e.g.*, via faster decoding, better overlap, or dedicated accelerators).

C.6 Ablation on Segmentation Hyperparameters M and N

The segmentation hyperparameters M (number of noisy blocks denoised per step) and N (number of conditioning blocks that anchor the segment) directly determine the per-step token budget and the amount of cross-segment context available to CSG. We sweep $M \in \{2, 3\}$ and $N \in \{1, 2\}$ on the VBench-I2V subset with all other settings fixed (Tab. 8).

Table 8: Ablation on the segmentation hyperparameters M and N of CSG. "Time" denotes the per-sample inference time of Stage II at 2560×1408.

M	N	Time	I2V Subj.	I2V Bg.	Subj. Cons.	Motion Smooth.	Dyn. Deg.	Total
3	**1**	**25s**	**0.9910**	**0.9975**	**0.9675**	**0.9885**	**0.3009**	**6.4243**
2	1	26s	0.9908	0.9975	0.9660	0.9871	0.2967	6.4107
3	2	33s	0.9909	0.9976	0.9665	0.9876	0.2967	6.4158
2	2	37s	0.9908	0.9975	0.9633	0.9854	0.2886	6.3916

We make the following observations. **(i)** Increasing the number of noisy blocks per step from $M{=}2$ to $M{=}3$ is consistently beneficial at a fixed N: with $N{=}1$, the total score improves from 6.4107 to 6.4243 while per-sample time even decreases slightly (26s vs. 25s) because fewer segmentation steps are needed to cover the full video. A similar trend holds for $N{=}2$. This indicates that within the token budget allowed by a 2K denoising step, a moderately larger M amortizes the segmentation overhead more efficiently and yields slightly better cross-segment coherence. **(ii)** Enlarging the conditioning window from $N{=}1$ to $N{=}2$ does not translate into overall quality gains; although it slightly improves I2V Background, it degrades subject consistency (0.9675→0.9665 with $M{=}3$), motion smoothness (0.9885→0.9876), Dynamic Degree (0.3009→0.2967), and total score while incurring higher latency. We attribute this to the image-anchored bidirectional interaction in CSG already providing sufficient global context through $\mathbf{z}_{\mathrm{ref}}$ and the first conditioning block; doubling the conditioning blocks enlarges the attention span but mainly dilutes the anchoring effect, slightly suppressing motion magnitude without adding useful information. **(iii)** $N{=}2$ also incurs a non-trivial efficiency cost (e.g., 25s→33s at $M{=}3$ and 26s→37s at $M{=}2$) as the per-step token budget grows.

Overall, $(M, N){=}(3, 1)$ provides the best overall quality–efficiency trade-off, and we therefore adopt it as the default configuration throughout our experiments.

C.7 Analysis of Stage Transition

This section further analyzes the stage transition strategy in Section 3.3. Since Stage I and Stage II are trained separately, Stage II inevitably faces a train–test mismatch: during inference, its input comes from Stage I and may contain typical artifacts (*e.g.*, VAE distortions or low-resolution

Table 9: Stage transition diagnostics under different noise strengths σ and denoising steps S.

σ	S	SNR ↑	PSNR ↑	SSIM ↑
0.01	1	7.412	35.165	0.896
0.1	4	5.948	33.701	0.833
0.1	1	6.152	33.905	0.841
0.3	4	5.069	32.828	0.780
0.5	4	4.593	32.346	0.748
0.7	4	4.013	31.766	0.702



Figure 10: Qualitative comparison of stage-transition synthesis under different settings.

temporal flicker), whereas during training it is fed with clean low-resolution videos obtained by direct downsampling. Such a mismatch causes Stage II to amplify artifacts in the high-resolution outputs. Our objective is to inject Stage I-like artifacts into the low-resolution training inputs ***without*** *changing the motion correspondence* between V^{LR} and V^{HR}, so as to preserve the motion–detail decoupled design.

Given V^{HR}, we downsample it to V^{LR} (360P), add noise with strength $\sigma \in [0, 1]$, and denoise it with Stage I for S steps to get $\tilde{V}^{LR}$. $\sigma = 0$ denotes no noise, while $\sigma = 1$ denotes pure noise.

We run a test on 36 videos and compute SNR/PSNR/SSIM between $\tilde{V}^{LR}$ and V^{LR}, shown in Tab. 9. When σ is too small (e.g., 0.01), the artifact injection is insufficient; when σ is too large (e.g., 0.70), the perturbation becomes excessive and the motion is corrupted (Fig. 10). We thus choose $\sigma = 0.10$ as a practical trade-off, and adopt $S = 1$ for efficient large-scale data synthesis.

C.8 VAE Reconstruction Fidelity

To quantitatively validate that the higher compression ratio of the VAE used in Stage II does not compromise the recovery of high-frequency details, we conduct a systematic VAE reconstruction fidelity experiment. Specifically, we extract 93 video clips (each with 81 frames) from 31 open-domain 4K source videos. We perform a full VAE encode-decode cycle on these clips at three different resolutions: 4K (3840×2160), 2K (2560×1440), and 720P (1280×720). We measure pixel-level metrics (PSNR, SSIM) and a perceptual metric (LPIPS). The results are summarized in Tab. 10.

Table 10: VAE reconstruction quality at different resolutions.

Resolution	PSNR (dB) ↑	SSIM ↑	LPIPS ↓
4K (3840 × 2160)	36.94	0.955	0.051
2K (2560 × 1440)	**35.25**	**0.941**	**0.049**
720P (1280 × 720)	33.20	0.914	0.048

As shown in Tab. 10, at the target 2K resolution of Stage II, the VAE achieves excellent reconstruction quality with a PSNR of 35.25 dB and an SSIM of 0.941, indicating high fidelity in preserving video content, including high-frequency details. Furthermore, the perceptual quality remains highly consistent across different resolutions, as evidenced by the similar LPIPS values (Fig. 11).

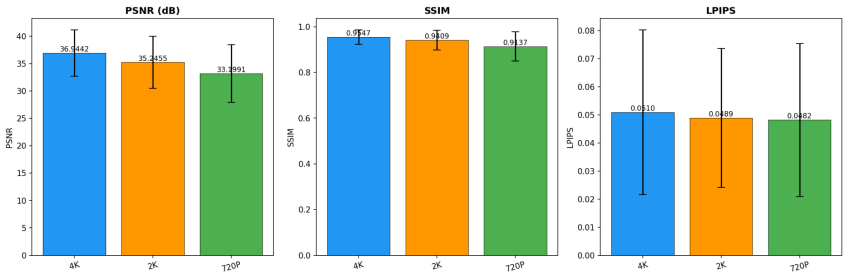

Figure 11: LPIPS comparison of VAE reconstruction across different resolutions.

To further confirm the preservation of high-frequency information, we compare the radial power spectral density (PSD) of the input videos and the VAE reconstructed videos across different resolutions (Fig. 12). The PSD curves of the input and reconstructed videos almost perfectly overlap, even in the high-frequency regions (normalized frequency > 0.6). This spectral analysis directly verifies that the VAE faithfully retains full-band information, from low to high frequencies, despite the high compression ratio.

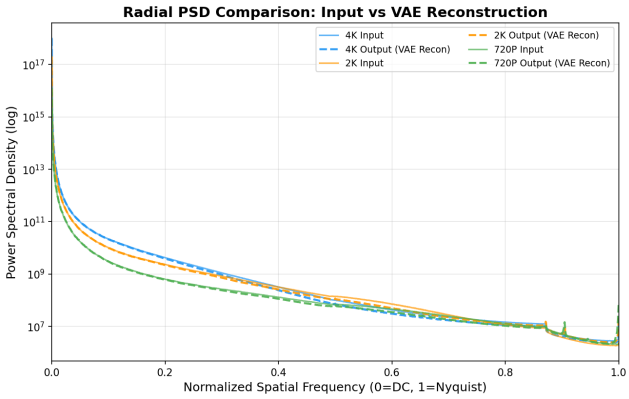

Figure 12: Radial Power Spectral Density (PSD) comparison between input and VAE reconstructed videos.

In conclusion, while Stage II employs a VAE with a higher compression ratio to reduce the token count, the spatial information required per latent token at the target 2K resolution is well within the VAE's representational capacity. This design achieves a favorable balance between quality and efficiency, significantly reducing the computational overhead of Stage II without sacrificing the fidelity of high-frequency detail recovery.

C.9 Temporal Smoothness across Segment Boundaries

To verify that our Conditional Segment-wise Generation (CSG) and bidirectional contextual interaction effectively eliminate segmentation-induced discontinuities, we conduct a quantitative analysis of temporal smoothness across segment boundaries. We measure frame-pair dissimilarity at the 6 CSG segment boundaries versus non-boundary positions across VBench-I2V subset test videos. As a discontinuity-free reference, we include the **Upscale** baseline (Stage I + bicubic upsampling), which involves no segmentation.

We evaluate three metrics: Optical Flow, $1-\text{SSIM}$, and Pixel Difference. The results are summarized in Tab. 11.

Table 11: Frame-pair dissimilarity at boundary vs. non-boundary positions.

Metric	Upscale Bnd	Upscale Non	SwiftI2V Bnd	SwiftI2V Non	Δ_{Up} / Δ_{Ours}
Optical Flow	0.727	0.706	0.705	0.688	$+2.9\%$ / $+2.5\%$
$1-\text{SSIM}$	0.115	0.109	0.133	0.131	$+6.1\%$ / $+1.9\%$
Pixel Diff	4.37	4.10	4.72	4.67	$+6.4\%$ / $+1.1\%$

Across all three metrics, the boundary-vs-non-boundary gap of SwiftI2V is consistently smaller than that of the segmentation-free reference: $+2.5\%$ vs. $+2.9\%$ for optical flow, $+1.9\%$ vs. $+6.1\%$ for $1-\text{SSIM}$, and $+1.1\%$ vs. $+6.4\%$ for pixel difference.

Furthermore, the per-frame optical flow curve (Fig. 13) confirms the absence of any spikes at boundary positions. These results validate that our CSG design, combined with bidirectional contextual interaction, effectively ensures temporal smoothness and prevents subtle temporal "popping" artifacts at segment boundaries.

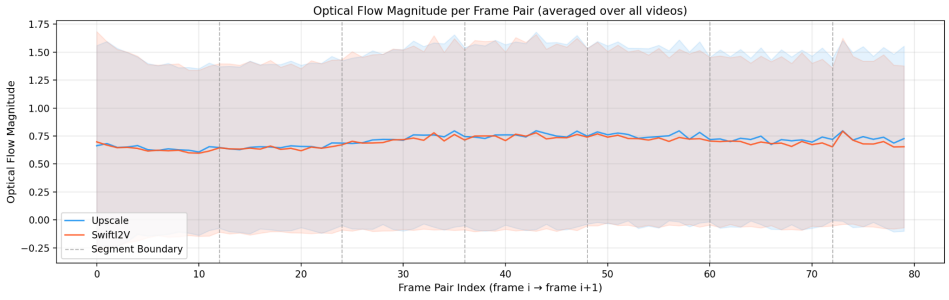

Figure 13: Per-frame optical flow curve across segment boundaries.

C.10 Analysis of Cross-Segment Error Accumulation

To comprehensively analyze how our Conditional Segment-wise Generation (CSG) mitigates error accumulation across segments, we conduct a systematic segment-wise quality degradation experiment. We evaluate the frame-level image quality metrics MUSIQ (↑) and NIQE (↓) independently for each segment on the VBench-I2V subset. We then perform linear regression on the segment scores to quantify the degradation trend. We compare three settings: **SwiftI2V** (with bidirectional interaction), **AR** (the `w/o bi-interaction` variant using a unidirectional causal mask), and **Upscale** (Stage I output directly upsampled, serving as a reference baseline without high-resolution detail synthesis).

The quantitative results are summarized in Tab. 12, and the corresponding degradation trends are visualized in Fig. 14 and Fig. 15.

Table 12: Segment-wise degradation analysis of MUSIQ and NIQE.

Method	NIQE Slope ↓	NIQE Degradation/step ↓	MUSIQ Slope ↑
SwiftI2V	$+0.050$	1.22%	$+0.243$
AR	$+0.093$	2.26%	-0.174
Upscale	-0.001	0.01%	$+0.100$

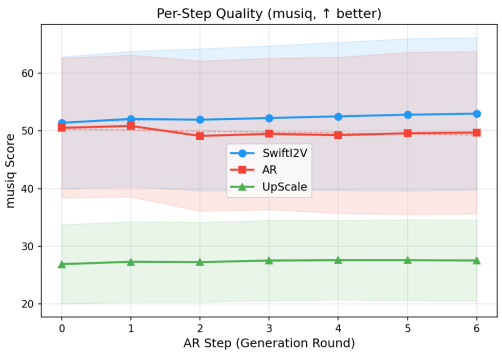

Figure 14: Segment-wise MUSIQ trend.

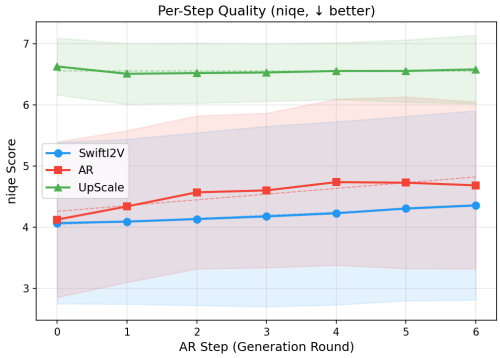

Figure 15: Segment-wise NIQE trend.

As shown in Tab. 12 and Fig. 15, the AR variant exhibits a clear error accumulation trend in NIQE, with a degradation slope of $+0.093$/step (a 2.26% degradation per step). In contrast, SwiftI2V significantly suppresses this accumulation, reducing the slope to $+0.050$/step (a 1.22% degradation per step), which is only 53% of the AR degradation rate. Furthermore, the absolute NIQE scores of SwiftI2V remain consistently better than those of AR across all segments.

For MUSIQ (Fig. 14), the AR variant shows a downward trend (slope -0.174), indicating a loss of detail quality over time. Conversely, SwiftI2V maintains a stable and even slightly improving trend (slope $+0.243$), demonstrating that the detail synthesis quality is well-preserved in later segments.

These results confirm that the bidirectional contextual interaction mechanism effectively mitigates the cascading error accumulation commonly observed in autoregressive generation. By allowing conditioning blocks to actively participate in the attention computation and dynamically adapt to the current segment's denoising needs, SwiftI2V reduces the propagation of historical errors, thereby helping maintain high fidelity throughout the progressive generation process.

C.11 Analysis of Dynamic Degree

In the full results visualized in Fig. 7 (Appd. C.1), SwiftI2V achieves a Dynamic Degree score of 0.3008. While this score is slightly lower than that of Stream-DiffVSR (0.3374), it is important to interpret these metrics in the context of visual quality and the two-stage generation paradigm.

First, compared to direct end-to-end generation models such as LTX-2 (0.0488) and CineScale (0.1667), SwiftI2V demonstrates a significantly higher Dynamic Degree. This indicates that our method is highly capable of generating substantial and meaningful motion, overcoming the common issue of static outputs in high-resolution I2V generation.

Second, Stream-DiffVSR's higher Dynamic Degree may partly reflect temporal flickering and structural instability rather than coherent semantic motion. Such artifacts are visible in our qualitative comparisons and can inflate the Dynamic Degree metric.

Finally, to address the concern that Stage II might overly constrain motion to preserve spatial details, we compare the Dynamic Degree of SwiftI2V with its Stage I output (denoted as Upscale in Fig. 7). The Upscale baseline achieves a Dynamic Degree of 0.2805, which increases to 0.3008 after Stage II

refinement. These results suggest that Stage II does not suppress dynamics as measured by this metric while refining spatial details.

D MORE 2K I2V VISUAL RESULTS OF SWIFTI2V

We provide additional 2K image-to-video generation results produced by SwiftI2V in Figures 16, 17, and 18. These examples demonstrate that SwiftI2V can synthesize temporally coherent dynamics while faithfully preserving input-specific spatial structures and fine-grained textures under strong image conditioning.

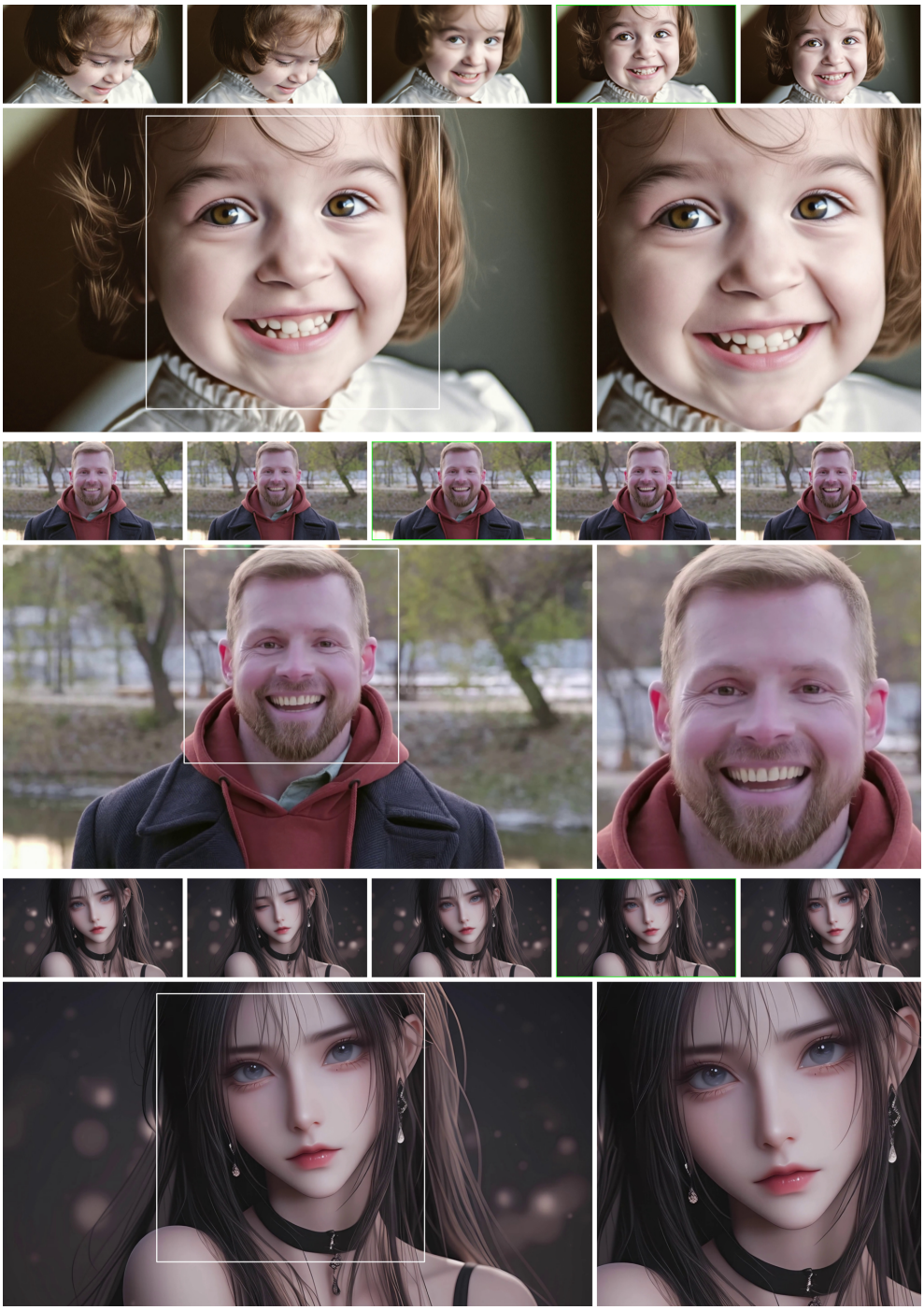

Figure 16: More 2K I2V visual results of SwiftI2V.

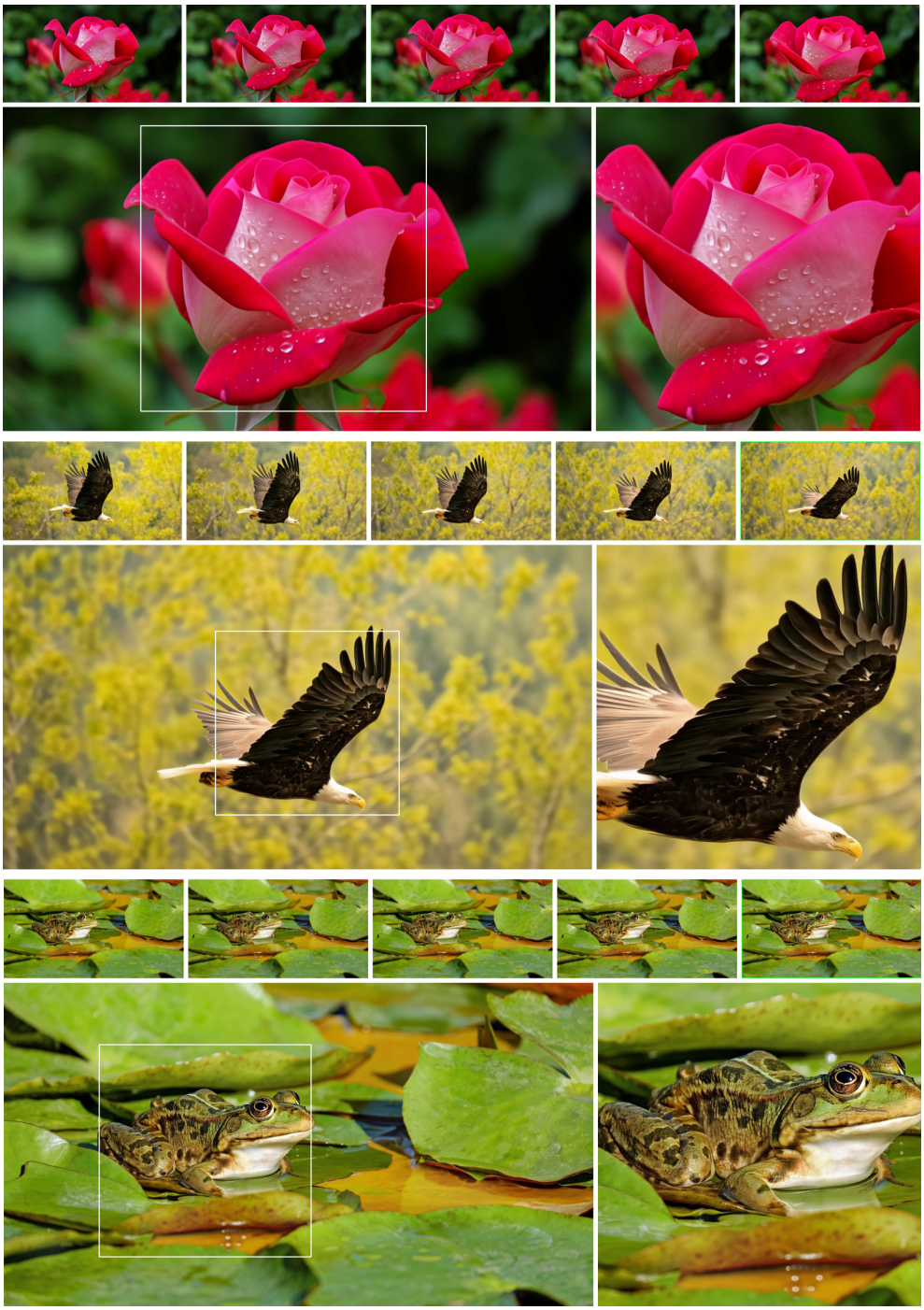

Figure 17: More 2K I2V visual results of SwiftI2V.

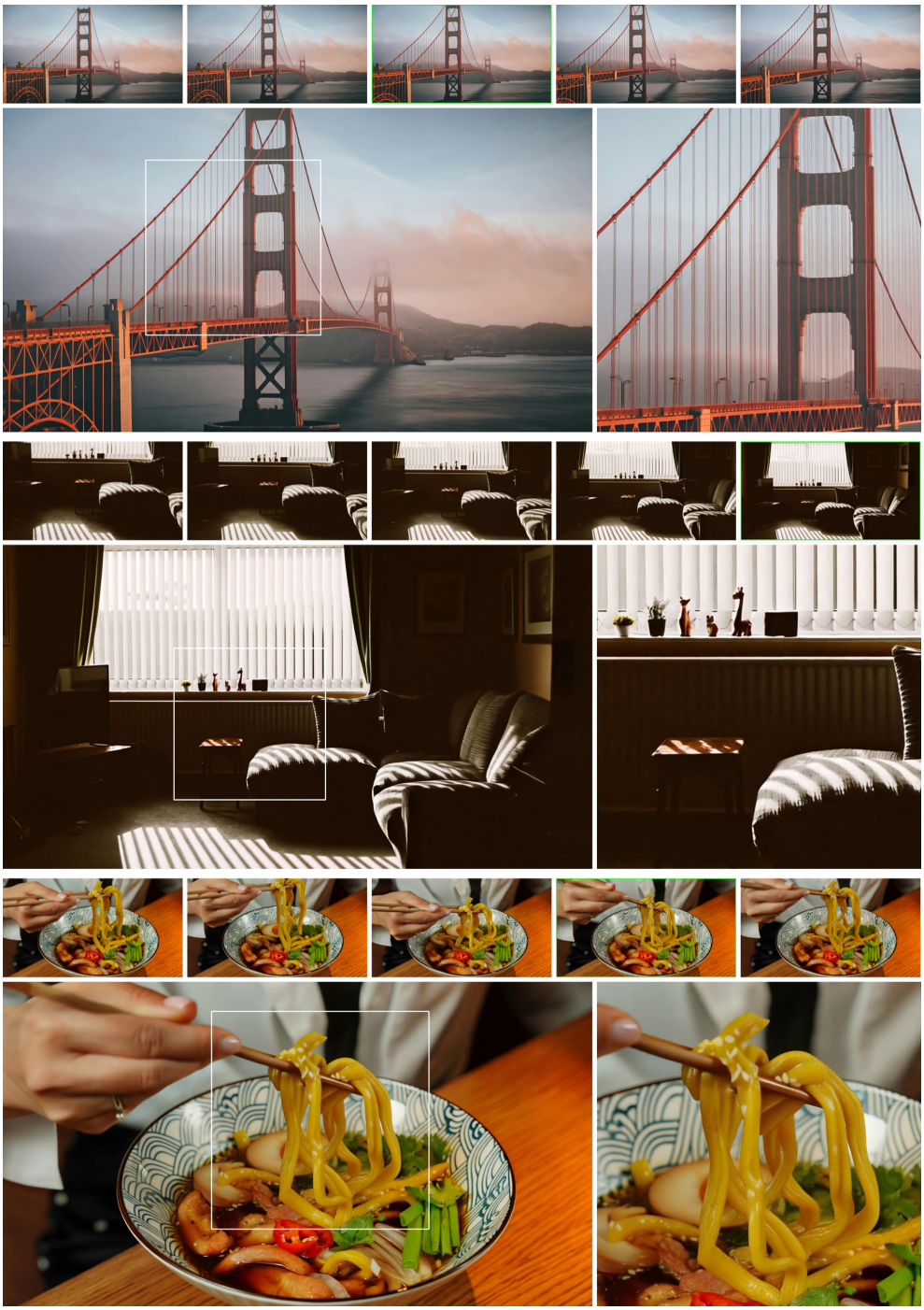

Figure 18: More 2K I2V visual results of SwiftI2V.