VAST

TECHNICAL REPORT

EdenLive: Real-Time Generative Rendering from Any Visual Reference

Zehuan Huang[1,*], Tian-Xing Xu[1,*], Chunli Peng[1,*], Cheng Zhang[1,*], Zeyi Zheng[1,†], Yifei Song[1,†], Zekai Gu[1,†], Zhipeng Yu[1], Peng Wang[1,‡], Yan-Pei Cao[1,✉]

[1]VAST

[*]Equal contribution [†]Intern at VAST [‡]Project lead [✉]Corresponding author

CORRESPONDENCE Yan-Pei Cao caoyanpei@gmail.com

PROJECT https://vast-eden.ai/?p=edenlive

CODE https://github.com/VAST-AI-Research/EdenLive

WEIGHTS https://huggingface.co/VAST-AI/EdenLive

DATE October 2026

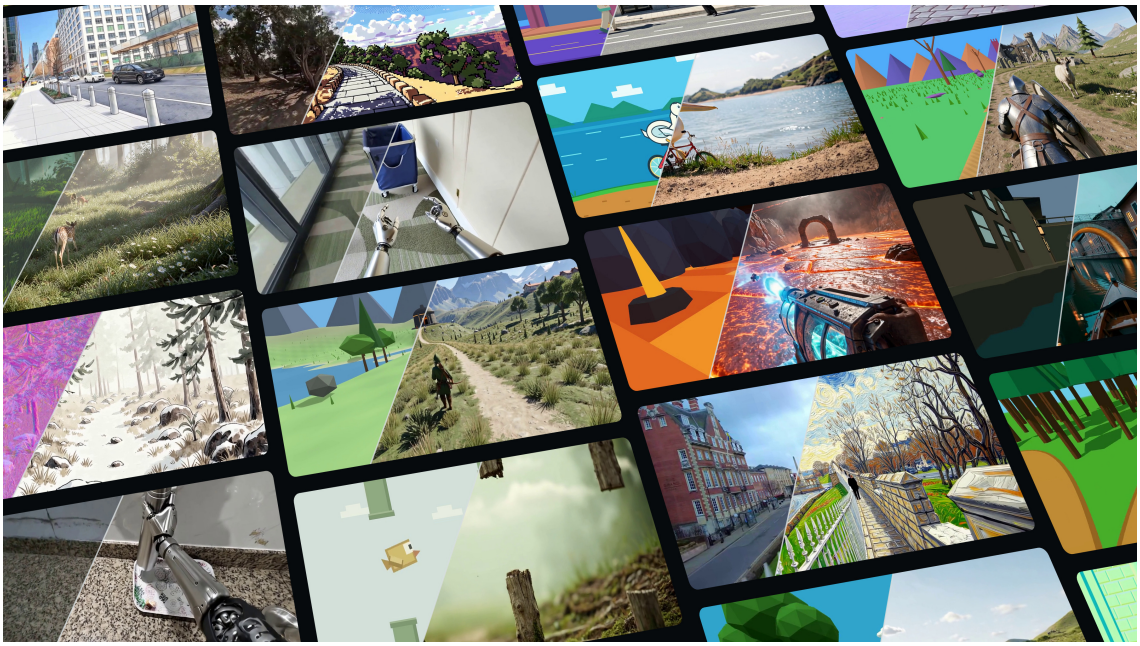


Figure 1 Real-time generative rendering from diverse visual references. Each tile pairs a reference (left) with the generated frame at the same timestamp (right). Examples include game and 2D proxies, normal maps, scene previews, video restyling, and embodied scenes. All outputs use the EdenLive 5B model.

ABSTRACT

We present EdenLive, a real-time generative video renderer for interactive applications in the Project Eden series[1]. EdenLive continuously renders enhanced and stylized video from reference videos of different geometric granularities that range from proxy videos, namely simplified renderings produced by a coding agent or a game engine, to directly recorded videos, with a target first frame and a text prompt as input to guide the appearance. This design conveys the world state to the model through the reference video and thereby decouples its interaction-induced evolution from visual rendering: the upstream code or engine maintains the game logic and world state explicitly and precisely, whereas EdenLive renders reference video into detailed visual output. This separation enables the world state to be inspected, modified, and tested directly, and interaction can extend to any quantity exposed by the upstream program. It also relieves the neural model from learning both state transitions and appearance, allowing the limited model capacity available for real-time deployment to concentrate on visual synthesis. To handle references for diverse rendering scenarios, we employ a vision encoder that maps them into a shared feature space despite their differences in geometric granularity and appearance, and condition the video generation model on these features, which mitigates leakage of the reference appearance into the generated video. To enable streaming generation, we distill the generator into a few-step chunk-wise autoregressive model and deploy it in a serving system that pipelines program simulation, reference encoding, and generation with a one-chunk lookahead, so that the system presents the generated video in real time and responds to user interaction. Experiments on short- and long-horizon generation show that EdenLive achieves good reference control and visual quality, while generating 720P videos in real time at over 24 FPS on a single GPU.

1 https://www.tripo3d.ai/research/project-eden

1 Introduction

World models characterize how an environment evolves in response to actions and events [32, 34], a capability that underpins simulation for autonomous driving [10, 37], embodied intelligence [9, 100], and interactive games [18, 86]. Two properties are central in these settings: interactivity, since a player or an agent chooses its next action on the basis of its current observation, and low latency, since each observation must follow the corresponding action quickly enough to sustain closed-loop interaction. Video-based world models have recently advanced rapidly in both respects by leveraging the generative priors of large pretrained video models [71, 85, 103]: starting from latent-action and action-conditioned models [4, 13, 86], recent methods synthesize environments that users explore through keyboard, mouse, and camera inputs [36, 39, 48, 54, 79, 84, 92], alter through text-triggered events [23, 29, 74], and revisit with the help of meory mechanisms [75, 95, 99], while achieving increasingly photorealistic visual quality.

These models, however, learn the evolution of the world state jointly with its visual appearance in a single generative network, which predicts each new observation from the generated history and the current input. The state of an interactive world is complex and heterogeneous, spanning entity positions and attributes, numerical quantities such as health and score, objects outside the current view, and the rules that determine how each action takes effect. Scaling model parameters and training data can improve how well such state is captured implicitly, but the same network must then devote its capacity to both dynamics and appearance, which is in tension with the computational budget permitted by real-time deployment. Moreover, the state is held only implicitly in the latent context and the generated frames, so it cannot be directly inspected, edited, or verified [42, 70], and control remains largely confined to actions, camera trajectories, and text prompts rather than extending to individual entities and their interaction rules.

To address this limitation, in Project Eden [88] we proposed to decouple the evolution of the world state from its visual rendering. An evolving structured state persists over time, is updated by actions, and can be queried from different viewpoints. A state-to-observation interface converts this state into camera-conditioned constraints that expose its local semantics and geometry. And a generative renderer maps these state-derived constraints to high-fidelity visual output. In this work, we instantiate this principle with EdenLive. Upstream code or an engine maintains and advances the world state by executing the game logic and interaction mechanisms. The engine renders each updated state into a temporally aligned video stream. Then a reference-conditioned, real-time generative renderer, which is trained starting from pretrained video diffusion models [85], follows the scene layout, camera motion, and dynamics conveyed by this video stream while synthesizing detailed visual output. Compared with modeling state evolution and appearance jointly in pixels, this separation keeps the game logic, interaction mechanisms, and world state explicit and precise: they can be inspected, modified, and tested directly, and interaction can extend to any quantity exposed by the upstream program. It also relieves the neural model from learning both state transitions and appearance, allowing the limited model capacity available for real-time deployment to concentrate on visual synthesis.

We refer to the conditioning video supplied to the renderer as a *reference video*, which is either a *proxy video*, namely a simplified rendering produced by a coding agent, a game engine, or a workbench that captures scene layout, camera motion and coarse dynamics, or a recorded video that is used directly as the reference. This formulation leads to a question: *can a single renderer accept reference videos from diverse upstream sources, ranging from proxies of 2D and 3D games to recorded footage, and render them in real time?* Answering this question requires the renderer to

recover layout and motion from references of widely varying geometry and appearance without copying unwanted reference appearance, learn despite the limited availability of paired reference–target videos, and generate causally and continuously within a real-time latency budget.

To address these challenges, We present **EdenLive**, a real-time generative video renderer that conditions on a reference video, a target first frame, and a text prompt. To handle heterogeneous references, we condition the generator on dense features rather than on reference pixels, extracted by a vision encoder that we fine-tune on reference–target image pairs to map reference videos that differ in appearance and geometric granularity but share layout and motion into a shared feature space. Conditioning on this feature space eases the learning of the generator and mitigates the degradation of its output toward the appearance of the reference video. Furthermore, the same feature space allows unpaired videos to serve as self-paired training data, in which the generator is conditioned on features extracted from an appearance-perturbed copy of the target video itself. To enable streaming generation, we distill the generator into a few-step model that performs chunkwise autoregressive inference over a bounded attention cache, so that each chunk depends only on the current reference frames and the retained history. Finally, our serving system overlaps upstream simulation and neural rendering, and streams the decoded frames to the user.

Our contributions are threefold:

- A vision encoder fine-tuning approach that aligns the representations of reference–target image pairs, enabling a generator trained on both self-paired and paired videos to be controlled by reference videos at inference.
- A streaming generative renderer that combines few-step chunkwise generation, a bounded historical context, and pipelined serving to synthesize video continuously from evolving reference videos.
- A complete real-time interactive system built on EdenLive, in which an upstream engine executes the game rules and maintains the world state while EdenLive renders the resulting proxy videos into the displayed stream; the system supports real-time previews of agent-built games with exact game state and appearance control through the target first frame, and extends to the offline re-rendering and stylization of recorded videos.

We evaluate both the learned representation and the complete renderer through image and video encoder ablations, short- and long-horizon generation comparisons, and runtime measurements. Compared with the pretrained DINOv3 encoder, the fine-tuned encoder improves the transfer from proxy conditions to detailed outputs in image generation and reduces camera rotation error in video generation, and the generation comparisons show competitive reference adherence and appearance preservation. The optimized 5B renderer exceeds 24 FPS at 1280×704 on a single NVIDIA H100, measured from model-side condition processing to decoded frames in CPU memory.

2 Related Work

2.1 Controllable Video Generation

Large pretrained video diffusion models [52, 71, 85, 103] provide strong priors over appearance and motion, but precise structural control often requires additional conditioning. Beyond first-frame appearance editing [53, 72], a broad line of work introduces dense visual cues to guide scene structure, object motion, and camera motion, such as depth, edges, and segmentation [2, 3, 19, 49, 90], or through camera poses, trajectories, optical-flow-warped noise, and rendered 3D caches [14,

35, 77, 91]. These conditioning signals differ in both their representation and the level of structural detail they provide.

These conditioning interfaces also vary in how much scene structure they require, ranging from dense rendering buffers to coarse proxies and sparse geometric primitives. Buffer-based methods [43, 58, 62, 78] receive dense rendering buffers to provide rich scene attributes for photorealistic image and video synthesis. Some works [15, 50] utilize coarse meshes and rendered proxies to reduce the required geometric detail while retaining control over layout and motion. ProxyUp [117] uses proxy videos with foreground masks to preserve selected dynamics while regenerating appearance and surroundings. Geometric conditions such as depth, 3D point tracks, and oriented bounding boxes also guide video generation [11, 31, 98]. These approaches rely on geometric representations or region-specific guidance that must be constructed or extracted from the source.

Inspired by the feature-based conditioning of Coarse-to-Real [30], EdenLive learns a shared representation for references with varying appearance and geometric granularity. Whereas Coarse-to-Real uses a frozen DINOv3 encoder [81], we fine-tune the encoder on a large, independently curated set of reference–target image pairs to preserve shared structure while suppressing appearance differences. This alignment reduces reference appearance leakage and enables a generator trained on self-paired ordinary videos to generalize to proxy inputs.

2.2 Interactive World Models

Interactive world models formulate environment simulation from generation history and user interaction. Genie [13] learns latent actions from unlabeled videos, while action-conditioned diffusion models demonstrate playable game simulation [4, 86]. Subsequent video world models extend action- and camera-conditioned generation across diverse environments [18, 36, 39, 54, 79, 84, 92, 111], with advances in real-time interaction and long-horizon generation [48, 112, 118]. Interaction also extends to text-driven events and scene editing [23, 29, 74] and object interactions [96], while Cosmos 3 [1] unifies multimodal understanding, visual generation, and action prediction. When world evolution is modeled through observation prediction, consistency with underlying rules is learned rather than enforced by an executable state-transition mechanism, and control depends on the inputs the model has learned to interpret.

A recent trend in interactive world modeling is to decouple state evolution from visual rendering, using video models as renderers. The Project Eden research preview [88] outlines a shared-state architecture in which camera-conditioned semantic and geometric constraints connect the evolving world state to generative rendering. Related efforts make world state explicit: StatePlay [66] jointly predicts game-state variables and visual content, while Marionette [70] and MASS [16] advance an explicit world state through learned dynamics and render it into observations. Other systems delegate state evolution to executable logic. Code World Model [20], Programmable World Model [42], Oneira [102], and CoDeR [24] use coding agents to maintain and evolve explicit world states, which are translated into proxy videos or spatial conditions. AgentGarten [21] and Magpie [108] advance states through simulators and game engines, while GameDirector [65] introduces an agentic layer that tracks game state and enforces gameplay rules. Several renderers use specialized geometric interfaces, including oriented boxes, depth and surface normals, or articulated poses [21, 42, 70]. EdenLive learns a common reference-video interface across varying appearance and geometric granularity, allowing diverse upstream programs to condition the same renderer on inputs ranging from agent-built proxies to recorded videos.

2.3 Acceleration, Distillation, and Streaming

Few-step diffusion methods reduce generation cost by distilling iterative sampling into fewer model evaluations. Latent Consistency Models [69] learn consistency mappings for few-step latent-space synthesis, while Distribution Matching Distillation [104, 105] trains a one-step generator to match a diffusion teacher's output distribution. Streaming video generation additionally requires causal prediction from generated histories. CausVid [106] distills a bidirectional teacher into a few-step causal generator, while Self Forcing [41] addresses the training-inference history mismatch through autoregressive training on the model's own predictions. Subsequent work extends generation to longer sequences [22, 101] and improves autoregressive distillation through student initialization, rollout perturbation, and gradient propagation through historical context [113, 115, 116, 119, 120]. Sustained streaming also requires bounded context and efficient execution. Attention sinks [94], combined with rolling windows, retain initial tokens and recent history while bounding the context size. Streaming pipelines use batched denoising and scheduling for continuous frame delivery [25, 51], while streaming video-to-video methods preserve temporal consistency through historical feature reuse or dedicated training objectives [57, 114]. EdenLive combines few-step chunkwise autoregressive generation with a bounded cache containing an attention sink and a rolling window. Its serving system overlaps upstream simulation and reference encoding with generation, conditioning each chunk only on reference frames that have already arrived.

3 Problem Formulation and System Overview

3.1 Problem Formulation

We decouple the interaction-induced evolution of the world state from its visual presentation: an upstream program maintains the world state explicitly, whereas EdenLive acts as a reference-conditioned generative renderer that synthesizes only its appearance. Given an N-frame reference video $r_{1:N}$, a target first frame y, and a text condition p, EdenLive generates a refined or restyled video

$$\hat{x}_{1:N} = \mathcal{R}(r_{1:N}, y, p), \tag{1}$$

in which the three inputs play complementary roles: the reference video conveys the evolving world state, namely the layout, camera motion, object trajectories, and event timing; the first frame, aligned with the opening reference view, anchors subject identity and appearance; and p is either a single prompt held fixed over the sequence or a time-varying prompt schedule, which may include time-specific event descriptions in interactive applications.

Achieving this in real time poses two challenges: a generator conditioned on reference videos that range from coarse proxies to recorded videos tends to degrade toward their appearance, and training a model for real-time, minute-scale generation is difficult. We address them with a vision encoder that maps reference videos into a shared feature space (Sec. 4.2) and a distilled, chunkwise autoregressive generator (Secs. 4.3 and 4.4), respectively.

3.2 Real-Time Interactive System

The real-time generation capability of EdenLive allows it to be integrated into an interactive system that responds to user input as it arrives. Fig. 2 illustrates the system we build upon EdenLive, which operates as a closed loop over successive chunks indexed by k. While viewing the most recently displayed chunk $\hat{x}^{(k-1)}$, the user issues keyboard and mouse actions a_t at interaction steps $t \in \mathcal{I}_k$, where $\mathcal{I}_k$ denotes the steps covered by chunk k. For concreteness, we take as a running example an

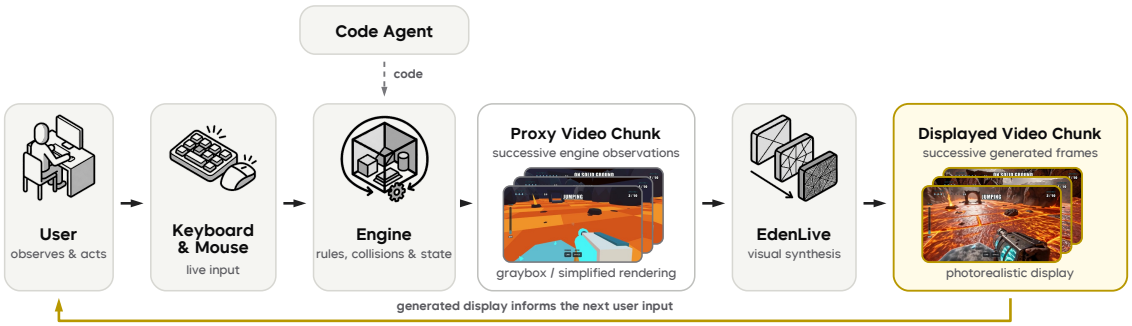


Figure 2 Real-time interactive system built on EdenLive. User input drives an upstream engine that maintains the rules and state and emits proxy video chunks, from which EdenLive synthesizes the displayed video; the displayed video in turn informs the user's next input.

upstream game engine whose code is authored by a coding agent (Sec. 6.1); the engine executes the rules, collisions, and state updates through a transition function $\mathcal{T}$, and an observation function $\mathcal{O}$ renders each updated state as a graybox proxy frame:

$$s_t = \mathcal{T}(s_{t-1}, a_t), \qquad r_t = \mathcal{O}(s_t), \qquad t \in \mathcal{I}_k. \tag{2}$$

The proxy frames form the reference chunk $r^{(k)} = \{r_t\}_{t \in \mathcal{I}_k}$, and the engine may additionally emit a caption $e^{(k)}$ describing salient events within the chunk, which we append to the static prompt p to obtain the chunk-level text condition $p^{(k)} = p \oplus e^{(k)}$. EdenLive then synthesizes the next displayed chunk

$$\hat{x}^{(k)} = \mathcal{R}_{\text{stream}}\big(r^{(k)}, y, p^{(k)}, \mathcal{H}_k\big), \tag{3}$$

where $\mathcal{H}_k$ denotes the retained generation history, and $\hat{x}^{(k)}$ in turn informs the user's next actions, closing the loop. With the state s_t conveyed to EdenLive only through $r^{(k)}$ and $p^{(k)}$, the renderer can be paired with diverse upstream programs without application-specific interfaces. Under this division of responsibility, collisions, scores, health bars, and other discrete outcomes are computed by $\mathcal{T}$ rather than inferred from generated pixels, and exact text and numerical indicators can be rendered directly from s_t and composited over the generated video. In contrast to recent interactive video world models [29, 79, 84, 109, 110], which model dynamics and appearance jointly and therefore cannot guarantee that such outcomes remain consistent with the underlying rules, our design keeps rule execution in the engine and dedicates the model to visual synthesis.

3.3 Offline Re-rendering System

For offline applications, EdenLive re-renders or restyles existing footage, taking as the reference video $r_{1:N}$ a complete recorded video or animation whose frames are all available before generation. For a fixed-length clip, it generates

$$\hat{x}_{1:N} = \mathcal{R}_{\text{bi}}\big(r_{1:N}, y, p\big), \tag{4}$$

where $\mathcal{R}_{\text{bi}}$ denotes bidirectional generation that jointly synthesizes all N output frames conditioned on the entire clip $r_{1:N}$, y is the target first frame that anchors the output appearance, p is a text condition authored in advance as either a single prompt or a time-varying prompt schedule, and $\hat{x}_{1:N}$ is the re-rendered video. Longer videos are instead processed in successive chunks by the streaming renderer $\mathcal{R}_{\text{stream}}$ of Eq. (3), with each reference chunk $r^{(k)}$ taken directly from the recorded sequence.

This mode makes EdenLive applicable to visual content creation, such as film, animation, and game production, as well as to data synthesis for embodied intelligence.

4 Method

4.1 Data

Training EdenLive requires reference–target video pairs with reliable correspondence. We construct such pairs through fine-to-coarse transformations of real-world and game-rendered videos and through coarse-to-fine synthesis from agent-built 3D proxy videos [6], and we complement them with unpaired open-source videos to improve generalization. Neither source guarantees perfect supervision, and we therefore curate each separately: paired data curation (Sec. 4.1.1) screens for spatial and temporal alignment, geometric granularity, and temporal consistency, whereas unpaired data curation (Sec. 4.1.2) screens for visual quality, motion intensity, and semantic suitability.

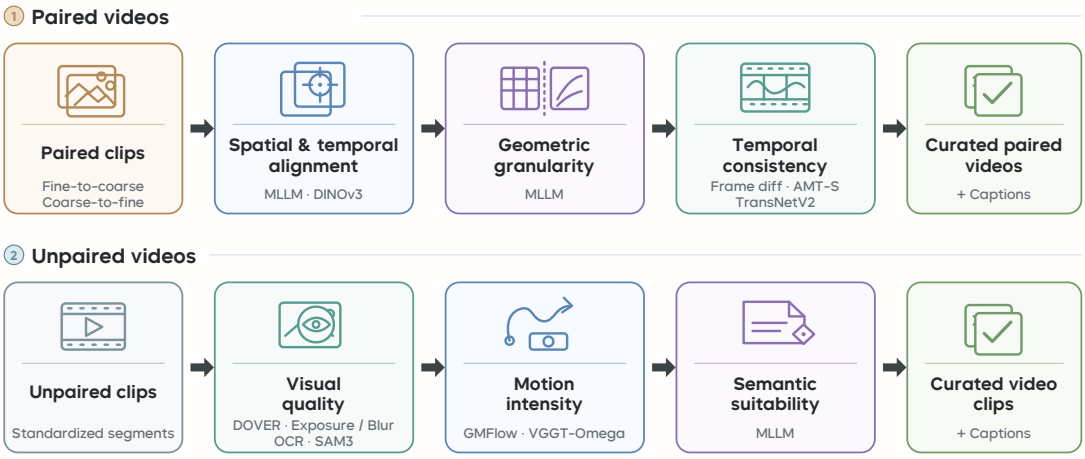


Figure 3 Data curation pipeline. Paired reference–target videos and unpaired open-source videos are screened by separate pipelines, and the curated data from both sources are accompanied by captions.

4.1.1 Paired Data Curation

Spatial and temporal alignment. Reference and target frames at the same index must depict corresponding scene states. We stack frames sampled at identical indices from both videos and use a multimodal large language model (MLLM) [7] to assess their scene layout, viewpoint, and object placement, as well as camera motion and object trajectories across the sequence. We additionally average the cosine similarity between DINOv3 [81] patch features at corresponding grid positions to obtain a per-frame alignment score.

Geometric granularity. During pair construction, the generated video may degrade toward its input: a fine-to-coarse reference may retain the detailed geometry of its source video, and a coarse-to-fine target may inherit the simplified geometry of its proxy. We prompt the MLLM to score whether sampled frames of the generated video exhibit the intended level of geometric detail, and discard pairs whose scores fall below a threshold.

Temporal consistency. We measure the intensity variation of each reference video as the mean absolute difference between consecutive sampled frames, its motion smoothness as the interpolation error of AMT-S [56], and its shot transitions with TransNetV2 [82], and exclude references with excessive flicker, irregular motion, or abrupt discontinuities.

4.1.2 Unpaired Data Curation

Visual quality. We split each video into segments that do not cross editing boundaries using shot or transition detection, score each segment with DOVER [93], mean brightness, and Laplacian-based sharpness, and detect visible text with PaddleOCR [73] and watermarks, logos, and borders with SAM3 [17]. Based on these measurements, we remove clips that are degraded, poorly exposed, blurred, or visually obstructed.

Motion intensity. We measure image motion as the average magnitude of GMFlow [97] optical flow between sampled neighboring frames, and camera motion as the mean angular speed during rotation and the peak angular speed derived from VGGT-Omega [89] camera poses, and exclude clips with sustained fast rotation or abrupt viewpoint changes.

Semantic suitability. From sampled frames of each clip, the MLLM generates a descriptive caption and multi-label category annotations under a fixed taxonomy; we exclude clips with any label in a predefined exclusion set and retain the captions with the selected clips.

4.2 Vision Encoder

4.2.1 Shared Feature Space

EdenLive accepts reference videos ranging from graybox proxies to recorded footage. These references convey the desired layout and motion, but their appearance may differ from the intended output. We therefore fine-tune a vision encoder on a large, independently curated set of reference–target image pairs so that references and targets that share the same layout are mapped to a shared feature space that retains their common structure while suppressing their appearance differences, which reduces reference appearance leakage in the generated output. This shared space allows a generator trained on self-paired ordinary videos (conditioned on their own features) to generalize to proxy inputs at inference.

Inspired by Coarse-to-Real [30], we initialize the vision encoder E_ϕ from pretrained DINOv3 ViT-L/16 [81]. Reference and target images share the same backbone and learned linear projection. For an input image I, we project each backbone patch token and apply L2 normalization, yielding unit-normalized features $\mathbf{f}_q(I)$ at spatial position q and the feature grid $E_\phi(I) = [\mathbf{f}_q(I)]_q$ of spatial resolution $\bar{H} \times \bar{W}$. The parameter set ϕ collects the backbone and projection parameters. Both are updated during fine-tuning and frozen during downstream generator training. For video conditioning, we apply this image encoder frame by frame and leave temporal modeling to the downstream video generator. We next describe how we fine-tune the encoder on reference–target image pairs to align corresponding content in this shared feature space.

4.2.2 Representation Learning

To exploit both the dense correspondence within each reference–target pair and the contrast across samples, we fine-tune the encoder with a combination of supervised and contrastive learning, complemented by domain-adversarial learning. Let $\{(I_i^{\text{ref}}, I_i^{\text{tgt}})\}_{i=1}^{B}$ denote a batch of B reference–target image pairs, where B is the global batch size across training workers, and let $\mathbf{f}_{i,q}^{\text{ref}}$ and $\mathbf{f}_{i,q}^{\text{tgt}}$ denote the normalized features of I_i^{ref} and I_i^{tgt} at patch q.

Supervised learning. Given that the reference and target images in each pair share the same layout, we directly supervise the encoder by aligning their features at corresponding grid positions:

$$\mathcal{L}_{\text{align}} = \frac{1}{B\bar{H}\bar{W}} \sum_{i=1}^{B} \sum_{q} \left(1 - \left\langle \mathbf{f}_{i,q}^{\text{ref}}, \mathbf{f}_{i,q}^{\text{tgt}} \right\rangle\right), \tag{5}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product, which equals the cosine similarity for the unit-normalized features.

Patch-level contrastive learning. To preserve discrimination across samples, we apply a symmetric, multi-positive InfoNCE objective [87] between domains at matching spatial positions. The positive set $\mathcal{P}(i)$ includes pair i and pairs sharing its group identifier, such as multiple references of the same target image; all other pairs in the batch are negatives. For query domain U and candidate domain V:

$$\ell_{i,q}^{U \to V} = -\log \frac{\sum_{j \in \mathcal{P}(i)} \exp\left(\left\langle \mathbf{f}_{i,q}^{U}, \mathbf{f}_{j,q}^{V} \right\rangle / \tau\right)}{\sum_{j=1}^{B} \exp\left(\left\langle \mathbf{f}_{i,q}^{U}, \mathbf{f}_{j,q}^{V} \right\rangle / \tau\right)}, \tag{6}$$

where $\tau = 0.07$ is the temperature. We obtain $\mathcal{L}_{\text{ctr}}$ by averaging $\ell_{i,q}^{U \to V}$ over all samples, spatial positions, and both reference-to-target and target-to-reference directions.

Domain-adversarial learning. To suppress domain-specific cues, we train a transformer discriminator D_ψ on the complete projected feature grid $\mathbf{F}_i^U = [\mathbf{f}_{i,q}^U]_q$, $U \in \{\text{ref}, \text{tgt}\}$, with target and reference grids labeled 1 and 0, respectively:

$$\mathcal{L}_{\text{dom}} = -\frac{1}{2B} \sum_{i=1}^{B} \left[\log\left(1 - D_\psi(\mathbf{F}_i^{\text{ref}})\right) + \log D_\psi(\mathbf{F}_i^{\text{tgt}})\right]. \tag{7}$$

The discriminator minimizes $\mathcal{L}_{\text{dom}}$, while gradient reversal [28] sends a reversed and scaled gradient to the encoder, updating both in a single backward pass.

Overall objective. Combining the three terms, we minimize

$$\mathcal{L}_{\text{enc}} = \mathcal{L}_{\text{align}} + \lambda_{\text{ctr}}\mathcal{L}_{\text{ctr}} - \lambda_{\text{dom}}\mathcal{L}_{\text{dom}}, \qquad \mathcal{L}_{\text{disc}} = \mathcal{L}_{\text{dom}}, \tag{8}$$

with respect to ϕ and ψ, respectively, where $\lambda_{\text{ctr}} = 0.5$ weights the contrastive term and the reversal coefficient $\lambda_{\text{dom}} = 0.1 \min(n/2000, 1)$ increases linearly over the first 2000 optimization steps n and remains constant thereafter.

4.3 Reference-Conditioned Streaming Video Generation

We initialize the video generator from two pretrained latent video diffusion transformers, Wan2.2-TI2V-5B and Wan2.1-I2V-14B [85], and progressively adapt them for reference conditioning, chunk-causal autoregressive generation, and few-step sampling. To realize the rendering interface in Eq. (1), dense features extracted from the reference video by the frozen vision encoder E_ϕ (Sec. 4.2) serve as the condition through which the world state is conveyed to the generator, alongside the target first frame and the text prompt.

4.3.1 Reference-Conditioned Video Generation

Reference injection. The pretrained generators operate in the latent space of a video VAE, which compresses the input video by a temporal factor ρ_t. To align the reference with these latent time steps, we encode one reference frame out of every ρ_t frames. Each encoded frame is a feature grid of C_r channels at resolution $\bar{H} \times \bar{W}$ (Sec. 4.2); we resize it to the spatial size of a latent frame and concatenate the resulting features with the native DiT input along the channel dimension, widening the patch-embedding projection by C_r zero-initialized input channels. With our encoder mapping references of different appearance into a shared feature space (Sec. 4.2), this concatenation injects the reference while mitigating the degradation of the output toward the reference appearance. Compared with token concatenation or additional cross-attention layers, it also leaves the token length and the attention cost of the base model unchanged.

Self-paired training. The shared feature space further allows us to construct self-paired examples from unpaired videos, in which the reference and the target are the same clip and only the reference branch receives appearance perturbations before encoding, so that the generator is discouraged from reproducing the target by copying the reference appearance. Self-paired examples thus turn abundant unpaired videos into self-supervised training data, and we combine them with paired reference–target examples, which share the underlying motion but differ in rendering and anchor the intended cross-domain behavior. We train the generator with flow matching [67], minimizing the squared error between its predicted velocity and the flow-matching target, on all latent frames except the conditioned first frame.

4.3.2 Autoregressive Video Generation

To satisfy the chunk-causal streaming requirement in Sec. 3.2, we convert the bidirectional model of Sec. 4.3.1 into a causal, few-step autoregressive generator, a transition that has been explored in prior work [22, 41, 106, 113, 119, 120]. Specifically, teacher-forcing supervised fine-tuning (SFT) first adapts the model to chunk-causal attention, and few-step distillation on the model's own rollouts then concludes training.

Teacher-forcing SFT. To expose the model to the causal structure used at inference while retaining the efficiency of parallel training, we divide a latent sequence into chunks $\mathcal{C}_k$ of L latent frames and, within a single forward pass, process the two-copy sequence $[\, z^{\text{clean}}_{1:T} \mid z^{\text{noisy}}_{1:T} \,]$, in which the clean copy serves as the history and the noisy copy as the prediction target. The attention mask allows a noisy query in $\mathcal{C}_k$ to attend to the clean history preceding $\mathcal{C}_k$ and to the noisy tokens within $\mathcal{C}_k$, but never to future chunks, whereas clean queries attend only to clean history. To match deployment, we further bound the context to M latent frames, so that the mask reproduces the key/value visibility of the streaming cache within a single teacher-forcing forward pass.

Few-step distillation. The model is trained on ground-truth history during teacher-forcing SFT, whereas at inference it conditions on its own previously generated chunks. We close this gap with few-step distillation on the model's own rollouts, initialized from teacher-forcing SFT and following the on-policy design of Self Forcing [41] and the rollout–replay formulation of Self-Gradient Forcing [120]. We generate chunks autoregressively with a few-step denoising schedule and a bounded key/value cache of M latent frames, whose first S latent frames form a permanent prefix and whose remainder holds the most recent history, and commit each predicted clean chunk to the cache before proceeding to the next one. The rollout states are then detached and replayed in a differentiable teacher-forcing pass whose attention visibility and rotary positions match those of the inference

cache (Sec. 4.4.1), which allows gradients to reach the generator without backpropagating through the entire denoising rollout.

For the generator update, a frozen real score model, namely the bidirectional model of Sec. 4.3.1, and a learned fake score model evaluate noisy versions of the replayed samples, following the score-difference construction of distribution matching distillation [105]. The normalized difference between their predictions of the clean latent, $g \propto \hat{z}_0^{\text{fake}} - \hat{z}_0^{\text{real}}$, supplies the update direction for the generated latent z_{gen}, and the generator minimizes the stop-gradient objective

$$\mathcal{L}_{\text{SGF}} = \frac{1}{2} \left\| z_{\text{gen}} - \text{stopgrad}(z_{\text{gen}} - g) \right\|_2^2, \tag{9}$$

whose gradient with respect to z_{gen} equals g, while the fake score model is trained on generated samples with the same flow-matching loss. Both updates exclude the conditioned first frame.

4.4 Real-Time Inference

To support real-time interaction with programs such as the agent-built games, we build a serving system that comprises four stages—player interaction, program simulation, generative rendering, and video streaming—and requires no additional training. Sec. 4.4.1 describes the inference-time adaptations of the generator and the serving optimizations we adopt, and Sec. 4.4.2 presents the module pipeline that couples the four stages, as illustrated in Fig. 4.

4.4.1 Autoregressive Streaming Generation

At inference, EdenLive generates the latent stream in chunks of L latent frames and, as in training, attends to a bounded cache of M latent frames, of which the first S form a permanent attention sink [94] and the rest a rolling window. Each chunk is sampled in four denoising steps without classifier-free guidance, so that only the conditional branch and a single key/value cache are required. We make three inference-time adaptations to this procedure.

RoPE repositioning. Under a 3D rotary embedding [83], attention logits depend on the temporal offset between query and key; since sink keys keep their original indices while queries advance with the stream, this offset soon exceeds the range seen in training. We instead index frames by cache slot: keys are cached without rotation, and queries and keys are rotated at attention time according to their current slots. Once the cache is full, the temporal indices of the cached frames, from oldest to newest, are fixed to

$$\big[\underbrace{0, \dots, S-1}_{\text{sink}},\ \underbrace{S, \dots, M-L-1}_{\text{window history}},\ \underbrace{M-L, \dots, M-1}_{\text{current chunk}}\big], \tag{10}$$

so that query–key offsets remain within the training range throughout the session.

Lightweight decoding. To reduce per-chunk decoding latency, we replace the native VAE decoder with the lightweight LightTAE decoder of LightX2V [61], which shares the compression ratios of the corresponding backbone and can therefore decode the same latents.

Event-driven text conditioning. In our game instantiation, the program emits a short sentence for each salient gameplay event, such as a landed hit, and the text condition of a chunk appends the recent event sentences to the static scene prompt. The updated embeddings replace only the cross-attention key/value cache, leaving the self-attention history intact.

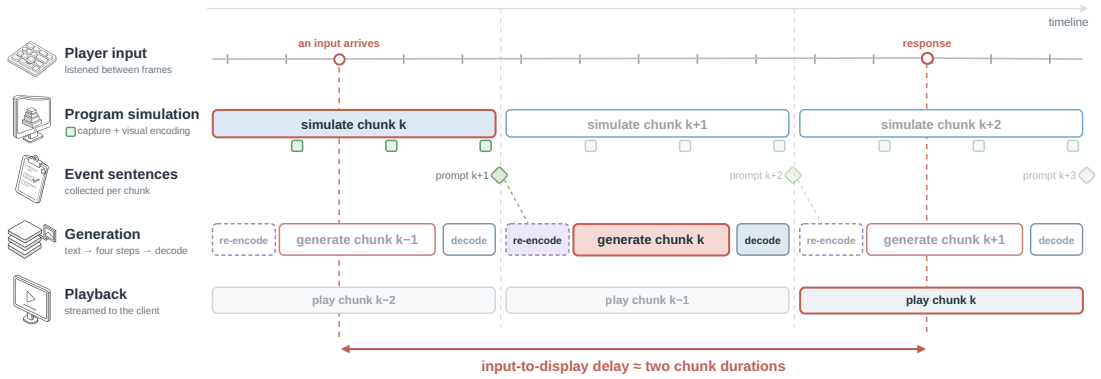


Figure 4 Module pipeline over three consecutive chunks. Time flows left to right, and each column spans one chunk duration in steady state. Simulation and proxy rasterization of chunk k overlap the generation of chunk $k-1$ (one-chunk lookahead), and each captured proxy frame (green squares) is visually encoded as it is rendered. The generation row is serial: text re-encoding (only when the composed prompt has changed), four denoising steps, and decoding. The red marker denotes a player input that arrives during the generation of chunk $k-1$.

Serving optimizations. Following OmniDreams [10], we build the inference system upon FlashDreams [26], and compile the transformer and the decoder with `torch.compile` [5]. At the start of a session, the generator warms up until the key/value cache reaches its full M frames; since the key/value tensors are pre-allocated at full size, the attended length and hence all tensor shapes remain fixed from then on, and we capture the transformer step once as a CUDA graph and replay it for every subsequent chunk. In the multi-GPU setting, we additionally shard the token sequence of the diffusion transformer with Ulysses-style sequence parallelism [45].

4.4.2 Module Pipeline

Real-time interaction requires the program, the encoders, and the generator to sustain a common frame rate, and a player action to appear in the displayed video as early as the causal generator permits. Rather than executing the stages sequentially, which adds the time of every stage to each chunk, we pipeline them with a one-chunk lookahead (Fig. 4) that hides simulation and encoding behind generation and thus raises throughput.

Player interaction and program simulation. The program owns the game logic and the world state and advances at a fixed simulation step that is independent of the generator. For visual encoding, we capture one proxy frame per latent frame, namely the last frame of each group of four, which matches the $4\times$ temporal compression of the video VAE.

Generative rendering and streaming. Before the generator starts chunk $k-1$, a dedicated worker begins simulating and rendering chunk k concurrently with the denoising of chunk $k-1$, and the visual encoding of chunk k is chained to that rendering on a separate GPU stream, so that its features are ready when chunk k is denoised; only the opening chunk exposes rendering and encoding on the critical path. To keep simulation and generation on a common clock, we stretch the rendering time of chunk k to $\tau_k^{\text{render}} = \max(\tau_{k-1}^{\text{gen}}, n_k/f_{\text{pre}})$, where τ_{k-1}^{gen} is the wall time of the preceding generation, n_k is the number of frames in the chunk, and f_{pre} is the upper bound on the presentation rate, so that the program state never advances ahead of the generated video. The text condition of chunk

k, in contrast, is complete only once its simulation has finished. Since text encoding and LightTAE decoding are inexpensive relative to denoising, we keep both on the critical path: the generator produces chunk k from the L encoded reference frames, the text cache, and the self-attention history, and the decoded frames are streamed to the client over WebRTC, which also carries keyboard and mouse events back.

Response delay and chunk size. Under this schedule, an input that arrives during the generation of chunk $k-1$ enters the simulation of chunk k and becomes visible only once chunk k has been generated, decoded, and played back; in steady state, the input-to-display delay is therefore about two chunk durations plus transmission and scales with the chunk length L. A shorter chunk reduces this delay, but a rollout of fixed duration then involves more autoregressive steps and is more prone to drift, and each step denoises fewer tokens, which lowers GPU parallelism and the achievable frame rate. We use $L = 3$ latent frames to trade off the response delay against these costs.

5 Experiments

We evaluate EdenLive through short- and long-horizon video generation, runtime measurements, and encoder fine-tuning ablations.

5.1 Evaluation Setup

Evaluation dataset. Our evaluation set spans diverse scenes, subject motions, camera movements, and interactions, including video restyling scenarios. The short- and long-horizon evaluations target five- and thirty-second videos, respectively. Each sample provides a reference video, a target first frame, and text conditions. The reference conveys scene layout, camera motion, and coarse dynamics, while the target first frame specifies the desired subject appearance and visual style.

Baselines and generation settings. The short-horizon baselines are C2R [30], CWM [20], depth-conditioned Wan2.1-VACE-14B (VACE) [49, 85], LTX-2.5 22B with the Layout-to-Render IC-LoRA (LTX 2.5) [33, 40, 60], Seedance 2.0 [80], and MiniMax H3 [71]. We compare them with the bidirectional 5B and 14B configurations of EdenLive. The long-horizon comparison evaluates autoregressive methods, including AgentGarten [21], StreamCWM [59], and the 5B and 14B configurations of EdenLive. Each method uses its supported conditioning interface.

For VACE, we provide depth estimated by DA3 [63] from the reference video, the target first frame as a reference image, and the text prompt. To adapt StreamCWM to our reference inputs, we use OneFormer [46] semantic predictions and ViPE–DA3 [38, 63] depth estimates, while keeping the generator weights unchanged.

Short-horizon comparisons use a static prompt. For long-horizon generation, EdenLive combines a static prompt with dynamic prompts associated with temporal intervals, whereas AgentGarten and StreamCWM use a static prompt.

Evaluation metrics. We evaluate reference control, first-frame consistency, text alignment, temporal consistency, and visual quality using the metrics below.

Reference control. Reference-DINO Patch averages cosine similarities between corresponding spatial tokens, using the same frozen, fine-tuned DINOv3-L/16 encoder [81] and projection for all methods. We also independently estimate reference and generated camera trajectories with DA3 [63], expressing each relative to its initial pose. Camera Rotation Error is the mean angular difference

in degrees; Camera Translation Error is the mean positional distance after least-squares scale alignment, in the reference reconstruction scale. Both feature and trajectory comparisons align clips at normalized temporal positions.

First-frame consistency. I2V Subject and I2V Background measure preservation of the target first frame using VBench++ [44].

Text alignment. VQAScore [64] measures video–text semantic alignment using the Qwen2.5-VL-3B [8] video backend.

Temporal consistency. For long videos, we use VBL Subject [44] with semantic scene splitting disabled, and Helios Aesthetic Drift [107], which compares aesthetic quality over the first and last 15% of frames.

Visual quality. DOVER-Technical [93] measures technical quality, applying the official standardization and sigmoid mapping to 0–100 per video before averaging. We use VBL Aesthetic for long videos [44].

VLM-based pairwise evaluation. We additionally use Gemini-3.8-Flash to assess a 45-case subset of the long-horizon evaluation set. The judge receives the target first frame, text conditions, and reference and candidate frame sequences sampled synchronously at 2 fps. It independently evaluates reference control, first-frame consistency, physical plausibility, visual quality, and reference-appearance leakage, following the core instructions in Fig. 5. We evaluate every model pair in both presentation orders, average the order-swapped preferences, and fit a Bradley–Terry model separately for each dimension to obtain Elo ratings. Higher ratings indicate stronger preference; for Leakage Avoidance, they indicate less reference-appearance leakage.

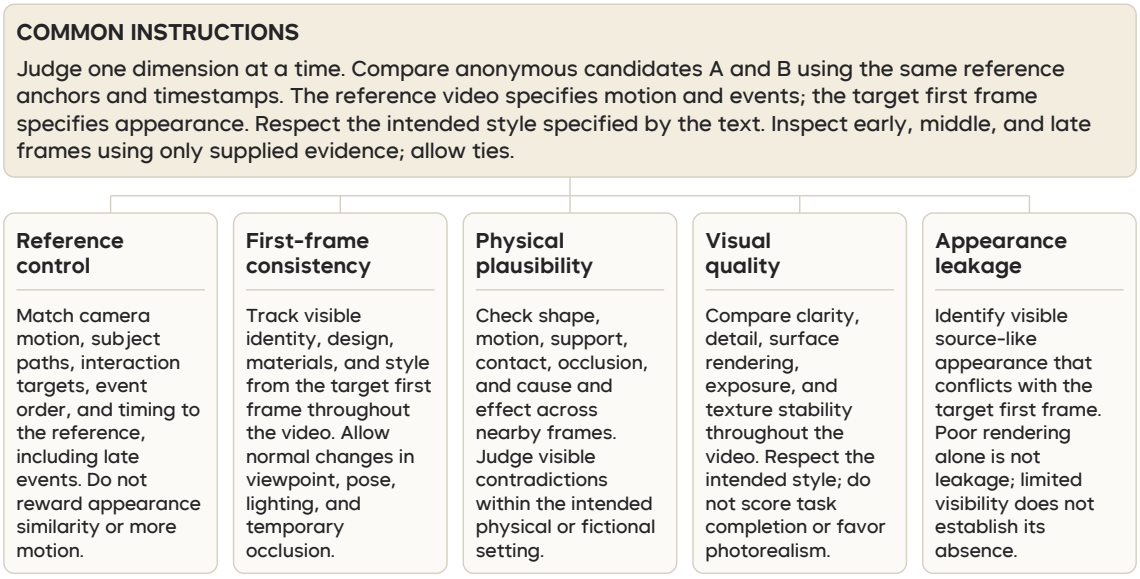


Figure 5 Core evaluation instructions. Condensed shared and dimension-specific instructions used by Gemini-3.8-Flash for pairwise assessment. Each comparison evaluates one dimension independently.

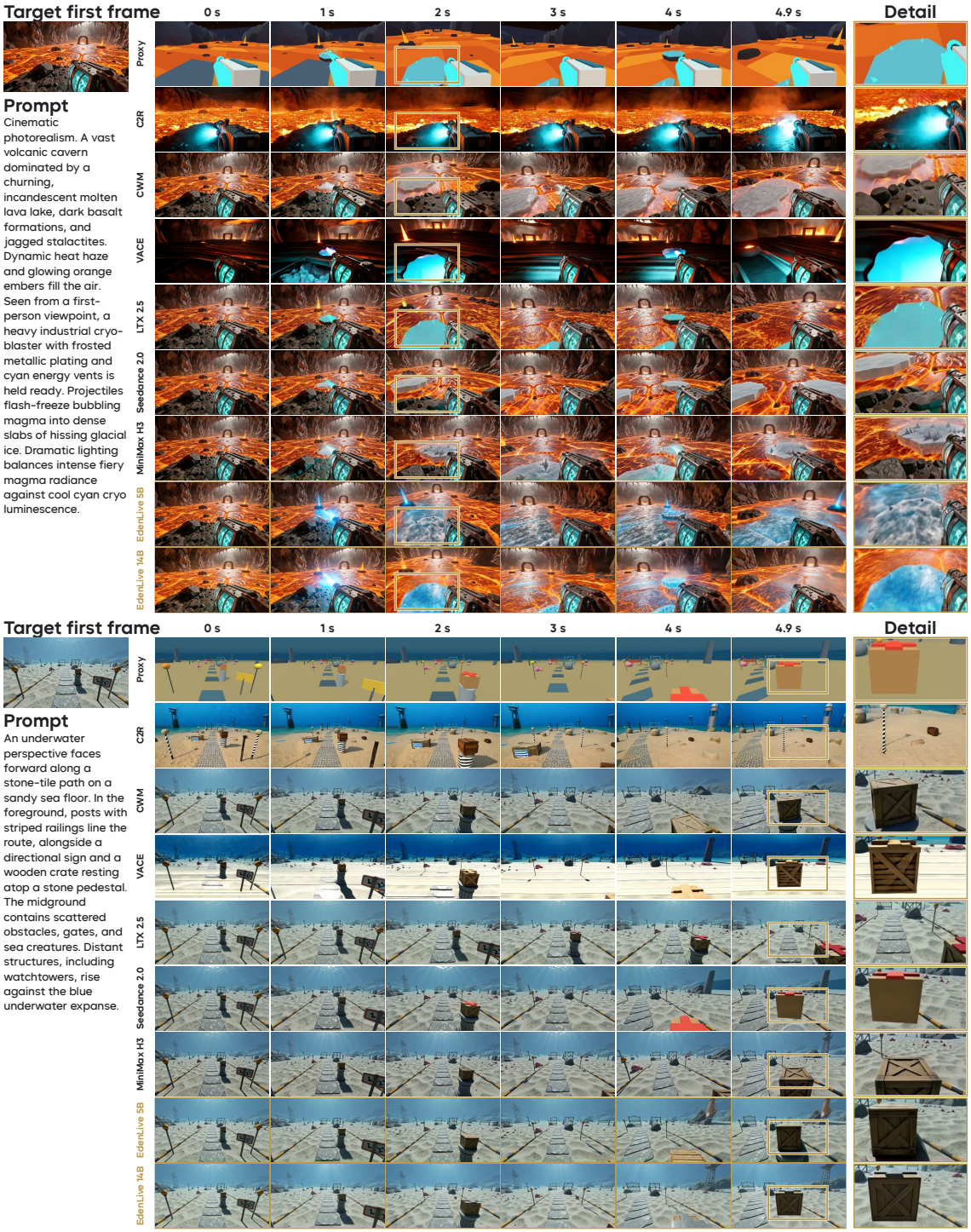


Figure 6 Short-horizon qualitative comparison: ice platform (top) and underwater path (bottom). Each panel shows the target first frame, prompt, and temporally aligned proxy and model outputs. EdenLive 5B and 14B occupy the bottom two rows.

5.2 Short-Horizon Video Generation

Fig. 6 illustrates how EdenLive combines reference structure with target appearance. In the ice-platform scene, the model preserves the lava cavern and weapon appearance shown in the target first frame, while rendering the simplified ice platforms in the proxy with ice and snow textures. In the underwater-path scene, both configurations retain the main appearance of the stone-tile path, wooden crates, and underwater environment, while depicting spatial relationships corresponding to changes in the reference viewpoint.

5.3 Long-Horizon Video Generation

Quantitative results. Tab. 1 compares long-horizon generation across five evaluation dimensions, complemented by VLM-based pairwise assessment. EdenLive 14B achieves the lowest camera rotation error and the highest I2V Subject, I2V Background, VQAScore, and VBL Subject scores, together with the lowest aesthetic drift, among the compared methods. It also obtains the second-lowest camera translation error and the second-highest Reference-DINO Patch score. These results support strong reference control, target-appearance preservation, text alignment, and temporal consistency over long rollouts. EdenLive 5B achieves the highest DOVER-Technical score and the second-highest VBL Aesthetic score, demonstrating strong visual quality. In the pairwise assessment, EdenLive 14B and 5B obtain the highest and second-highest Elo ratings, respectively, in all five judged dimensions.

Table 1 Long-horizon quantitative comparison. AgentGarten, StreamCWM, and the autoregressive 5B and 14B configurations of EdenLive are evaluated across five dimensions, complemented by VLM-based pairwise assessment.

Dimension	Metric	AgentGarten	StreamCWM	EdenLive 5B	EdenLive 14B
Reference control	Camera Rotation Error (°) ↓	<u>12.23988</u>	19.25996	12.71493	**11.60634**
	Camera Translation Error ↓	**0.15169**	0.23013	0.20645	<u>0.18507</u>
	Reference-DINO Patch ↑	**0.93466**	0.86020	0.90893	<u>0.93038</u>
First-frame consistency	I2V Subject ↑	0.93756	0.91277	<u>0.93950</u>	**0.94649**
	I2V Background ↑	0.94899	0.93066	<u>0.95575</u>	**0.95691**
Text alignment	VQAScore ↑	0.78072	0.78125	<u>0.79790</u>	**0.80361**
Temporal consistency	VBL Subject ↑	0.95899	0.96008	<u>0.96199</u>	**0.96404**
	Aesthetic Drift ↓	0.04859	0.05477	<u>0.04311</u>	**0.04186**
Visual quality	DOVER-Technical ↑	<u>9.80891</u>	9.05039	**9.93847**	9.42643
	VBL Aesthetic ↑	0.55307	**0.58448**	<u>0.58085</u>	0.55882
VLM-based Pairwise Evaluation					
VLM judge (Elo ↑)	Reference Control	1495.30	1226.85	<u>1573.50</u>	**1704.35**
	First-Frame Consistency	1442.02	1299.00	<u>1560.64</u>	**1698.34**
	Physical Plausibility	1471.42	1268.76	<u>1545.62</u>	**1714.20**
	Visual Quality	1481.20	1226.68	<u>1579.45</u>	**1712.67**
	Leakage Avoidance	1464.04	1500.99	<u>1509.71</u>	**1525.26**

↑/↓ indicate higher/lower is better. Bold and underlined values indicate the best and second-best results, respectively.

Fig. 7 compares long-horizon generation results. In the megalithic-destruction scene, EdenLive depicts flames and building debris around the subject. In the armored-warrior scene, both EdenLive configurations generate a cyan energy shield, and the 5B model additionally depicts thruster flames. These examples illustrate EdenLive's ability to render local interaction details while preserving the appearance of the subject and scene over long rollouts.



Figure 7 Long-horizon qualitative comparison: megalithic destruction (top) and armored warrior (bottom). Each panel shows the target first frame and static prompt on the left, with dynamic prompts above temporally aligned proxy and model outputs. EdenLive 5B and 14B occupy the bottom two rows.

5.4 Runtime Performance

Measurement protocol. We evaluate the optimized inference system described in Sec. 4.4 on NVIDIA H100 GPUs, and compare it with AgentGarten [21], StreamRender-H3 [55, 68], and StreamCWM [59]. All methods use the same approximately thirty-second offline proxy inputs and random seed. After warmup, we time each request from model-side condition processing until all generated RGB frames are available in CPU memory. This includes condition encoding, generation, decoding, and host transfer; proxy rendering, offline preprocessing, video-file encoding, and network transmission are outside the timed interval. Inference throughput is FPS $= N/T$, where N is the total number of newly generated frames across timed requests and T is their total elapsed time in seconds.

Table 2 Inference throughput on NVIDIA H100 GPUs after warmup.

Method	Resolution	H100 GPUs	Latent frames per chunk	Inference throughput (FPS)
EdenLive 5B	832 × 480	1	3	73.52
EdenLive 5B	1280 × 704	1	3	29.80
EdenLive 14B	832 × 480	8	3	26.76
AgentGarten	832 × 480	1	4	36.14
StreamRender-H3	1344 × 768	8	9 → 3/2	2.56
StreamCWM	1344 × 768	7 + 1	5	1.69

9 → 3/2: nine latent frames initially, then alternating chunks of three and two. 7 + 1: seven sampling GPUs and one conditioning/codec GPU. Baseline geometry and semantic maps are precomputed; EdenLive's reference encoding is timed. EdenLive and AgentGarten use fast streaming decoding; StreamCWM uses full-video decoding and CPU offload, with transfers included in timing.

Inference throughput. Tab. 2 summarizes the measured configurations. On a single H100, EdenLive 5B achieves 73.52 FPS at 832×480 and 29.80 FPS at 1280×704. EdenLive 14B achieves 26.76 FPS at 832 × 480 on eight H100s. All three configurations exceed 24 FPS in model-side inference throughput. With further optimization, the 5B model additionally supports real-time inference on a single consumer GPU, the NVIDIA RTX 5090.

Latency and memory. Tab. 3 reports EdenLive's runtime diagnostics. Generation throughput excludes condition processing. First-frame/chunk latency measures the time from request start until the first frame/complete chunk is available in CPU memory; inter-chunk gap P95 measures the 95th percentile of intervals between successive chunk completions. The real-time factor (RTF) is request time divided by playback duration. Memory figures report peak allocated and reserved memory per GPU.

Table 3 Warm latency, throughput, and memory usage of EdenLive. All configurations use three latent frames per chunk, four sampling steps, and LightTAE decoding.

Metric	**5B** (1 H100) 832 × 480	**5B** (1 H100) 1280 × 704	**14B** (8 H100s) 832 × 480
Generation throughput (FPS)	80.68	31.04	28.81
Inference throughput (FPS)	73.52	29.80	26.76
First-frame latency (ms)	826.36	1161.79	1714.00
First-chunk latency (ms)	826.38	1161.81	1714.02
Inter-chunk gap P95 (ms)	148.24	391.54	429.90
RTF	0.2172	0.5359	0.5966
Peak allocated (GiB)	28.53	36.78	50.83
Peak reserved (GiB)	29.34	38.94	60.67

5.5 Effect of Encoder Fine-Tuning

Encoder transfer to image generation. We train two matched FLUX.2 Klein Base 9B [12] generators on the same dataset of unpaired images, using features from either the original DINOv3 encoder (Pretrained) or the fine-tuned encoder (Fine-tuned); both encoders remain frozen. We evaluate the models on coarse–fine image pairs collected from separate sources. Coarse inputs test cross-domain transfer, while fine inputs provide a reconstruction control. Tab. 4 reports DreamSim [27] distance to the paired fine image and CLIP-T cosine similarity using CLIP ViT-L/14 [76].

Table 4 Image encoder ablation on held-out pairs. Coarse conditioning tests cross-domain transfer; fine conditioning measures reconstruction. Both generators keep their conditioning encoders frozen.

	Coarse input		Fine input	
Frozen encoder	DreamSim ↓	CLIP-T ↑	DreamSim ↓	CLIP-T ↑
Pretrained	0.5134	0.2214	**0.1049**	0.2976
Fine-tuned	**0.3675**	**0.3061**	0.3344	**0.3091**

With coarse conditioning, encoder fine-tuning reduces DreamSim distance from 0.5134 to 0.3675 and increases CLIP-T from 0.2214 to 0.3061. Fig. 8 illustrates this transfer across object close-ups, stylized landscapes, and street scenes. With fine conditioning, DreamSim distance increases from 0.1049 to 0.3344, consistent with the intended suppression of source-appearance information in the conditioning features.

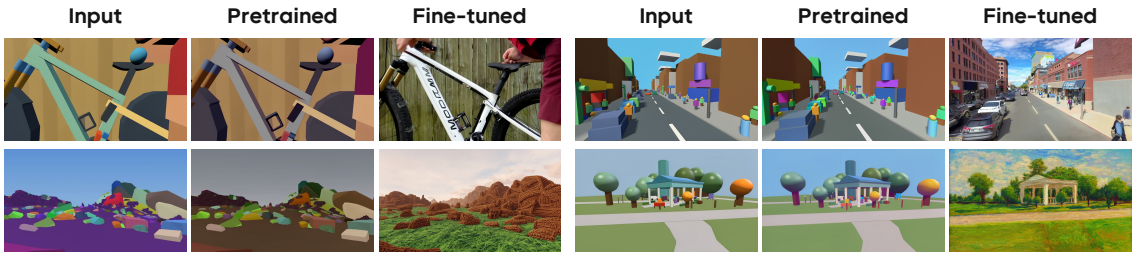


Figure 8 Image generation from coarse inputs: input, Pretrained output, and Fine-tuned output (left to right), using the same prompt and seed.

Effect on video generation. We compare two 5B configurations conditioned on Pretrained and Fine-tuned DINOv3 features, using the same sampling settings. Fig. 9 illustrates structural fidelity under coarse reference conditioning. In the collapsing-temple example, Pretrained omits the stone obstacle appearing in the proxy, whereas Fine-tuned retains an obstacle in front of the character. In the melee example, Pretrained produces intertwined limb and equipment structures where the characters meet; Fine-tuned maintains more distinct character silhouettes in the same region.

We also compare Pretrained and Fine-tuned in fine-to-fine style transfer, using a photorealistic source video for motion conditioning and a stylized target first frame with a text prompt to specify appearance. Fig. 10 illustrates source-appearance leakage and style drift in Pretrained: the oil-painted wall develops dark, mixed textures, while the pixel-art walkway loses its paving pattern and approaches the smooth surface of the source video. Fine-tuned better preserves the target painting and pixel-art appearance in these examples.

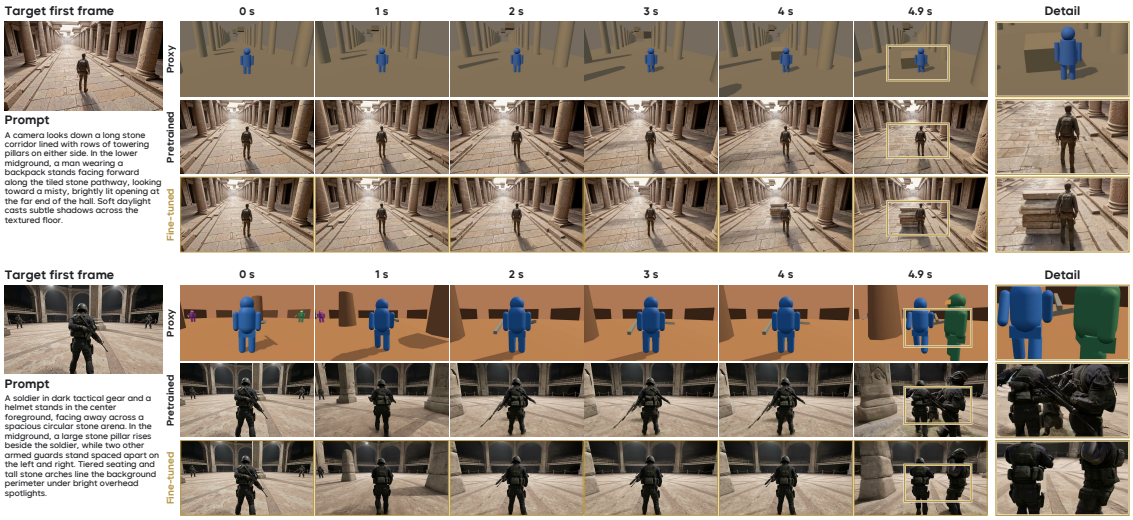


Figure 9 Structural fidelity with different conditioning encoders on the collapsing-temple (upper) and third-person melee (lower) examples. Each example shows the target first frame and prompt on the left, followed by sampled frames from the proxy, Pretrained, and Fine-tuned (top to bottom). The rightmost column enlarges the highlighted region of the final sampled frame.



Figure 10 Fine-to-fine style transfer with different conditioning encoders.

6 Applications

We demonstrate EdenLive in interactive gaming, appearance exploration for scenes and animations, video re-rendering and stylization, and embodied data augmentation.

6.1 Real-Time Previews of Agent-Built Games

EdenLive delegates the evolution of the world state to an upstream program and only synthesizes the visuals, and playable graybox games provide a natural testbed for this design: their programs maintain explicit game state, while their coarse geometry allows coding agents to build them rapidly without detailed assets or rendering effects and still yields a reference stream suitable for conditioning. We first introduce Graybox Agent, which converts natural-language gameplay descriptions into playable 3D graybox games, and then demonstrate real-time interactive rendering and appearance control through the target first frame.

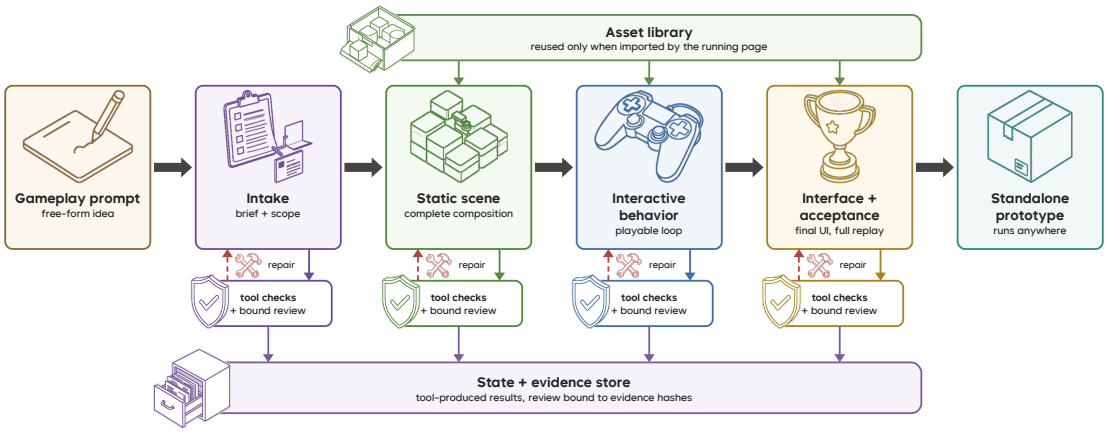


Figure 11 Overview of Graybox Agent. Four stages progressively turn a gameplay description into a standalone, playable graybox game. Each stage must pass tool checks and a bound review before the next begins, with failed checks triggering repair within that stage; all results are recorded in a shared evidence store.

Graybox Agent. *Graybox Agent* guides a coding agent to turn a natural-language gameplay description into a standalone, playable 3D graybox game (Fig. 11). It creates the upstream program that responds to player input and supplies visual references to EdenLive. Using stage-specific instructions and reusable graybox assets, the agent progressively builds the same game through four stages:

- **Intake** defines what the game should do: its mechanics, key parameters, win/loss conditions, and scene theme.
- **Static scene** builds a viewable graybox scene containing the environment, characters, and interactive objects.
- **Interactive behavior** combines controls, collisions, progression, and win/loss logic into a playable game.
- **Interface and acceptance** adds the player-facing interface and verifies complete playthroughs, win/loss outcomes, restarting, and standalone execution.

Three design choices make this pipeline reliable. First, to ensure that completion is demonstrated rather than asserted, each stage must pass checks appropriate to its output before the next begins: interactive behavior is checked by running the game with real input and observing the resulting state, and the final stage checks the complete game as a whole. Second, to avoid unbounded repair loops and silent simplification of the request, failed checks trigger repairs within a fixed budget, and missing assets are replaced with simple primitives while the requested gameplay is preserved. Third, to generalize beyond familiar genres, we compose behavior from reusable capabilities such as locomotion, aiming, and timing rather than from genre labels, so that different combinations of mechanics share the same components.

Integration with EdenLive. The resulting games have coarse visuals but complete game logic (Fig. 12): during play, the game program handles movement, collisions, and win/loss decisions. To connect a game to EdenLive, we supply its rendered graybox frames and short descriptions of gameplay events as the visual and text conditions of each chunk, from which, together with the target first frame, EdenLive generates the displayed video. Fig. 13 shows the workflow from game creation and first-frame generation to live preview.

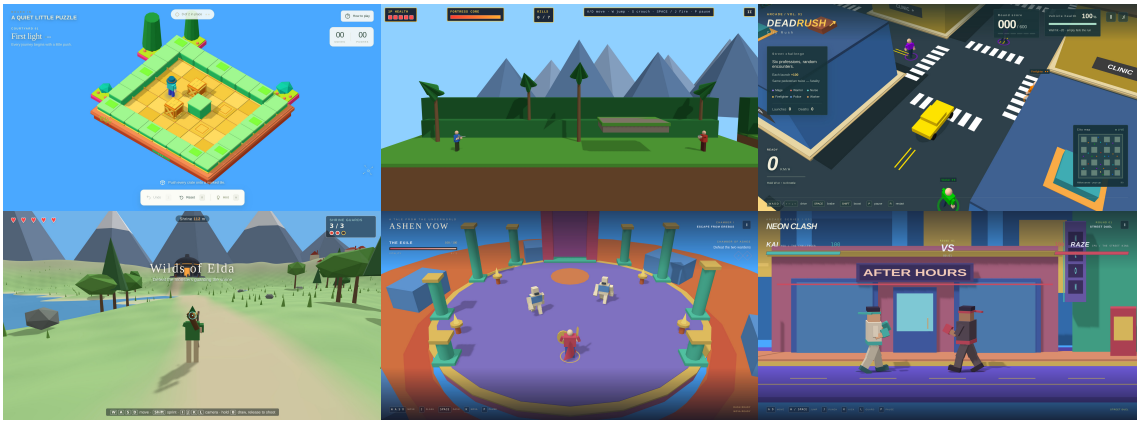


Figure 12 Games built by Graybox Agent. Playable graybox games generated from natural-language descriptions, spanning puzzle, shooting, driving, adventure, arena, and fighting mechanics.

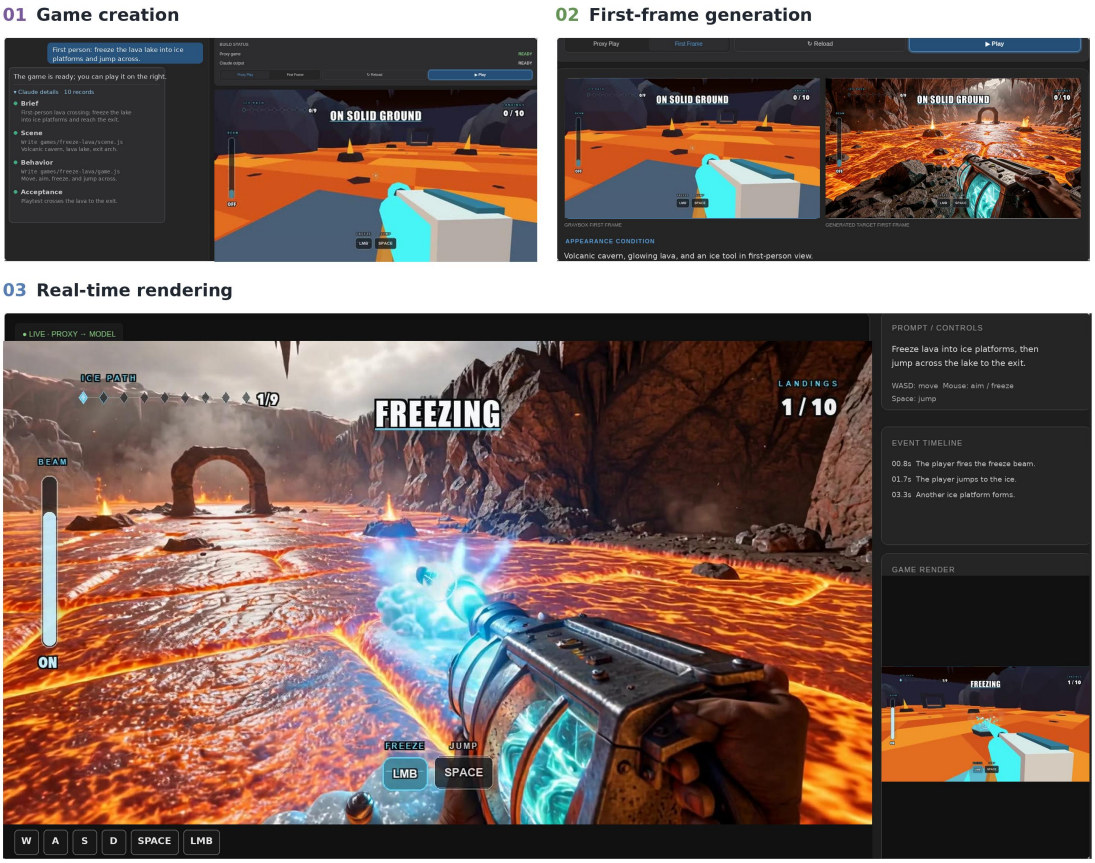


Figure 13 Interactive game workbench. From a gameplay description to a playable graybox, target-first-frame preparation, and real-time interactive rendering.

Interactive previews with explicit game state. Fig. 14 shows three interactive games built in this way, namely a street fight, a Sokoban-style puzzle, and a city-destruction game. Since the game state is maintained by the program rather than inferred from pixels, interaction outcomes strictly

follow the game rules: in the street fight, a kick lowers the opponent's health from 100 to 83 only once the program registers the hit; in the puzzle, each push moves the crate exactly one tile and advances the move and push counters; and in the city-destruction game, a building collapses under the fire breath, which advances the destruction counter from 0 to 1 out of 10. Interface elements such as health bars, move counters, and destruction counters are rendered directly from the game state and overlaid on the generated video, and therefore remain exact and legible.

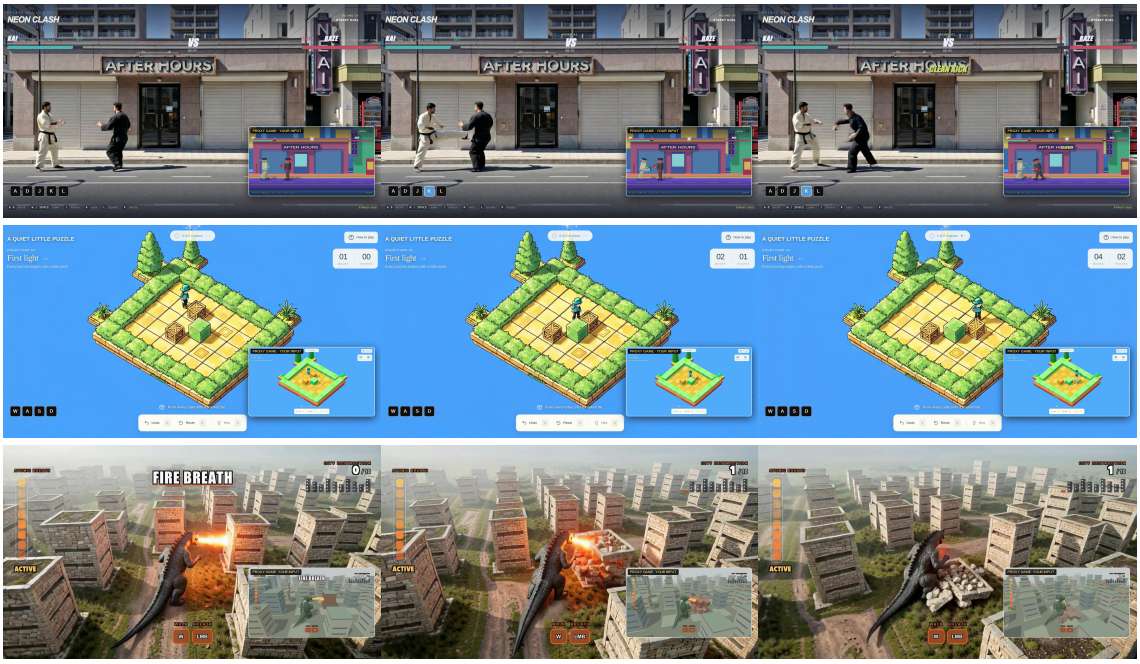


Figure 14 **Real-time previews of agent-built games.** Each row shows three moments, from left to right, of one game during interactive play. The full frame is generated by EdenLive, the inset shows the graybox frame rendered by the game program, and the highlighted keys indicate the current player input. Top to bottom: a street fight, a Sokoban-style puzzle, and a city-destruction game.

Appearance control via the target first frame. A single game can be rendered in different visual styles without changing its code (Fig. 15). The target first frame defines the desired appearance while preserving the spatial layout of the initial reference frame.

In summary, these applications render games in real time and vary their appearance without modifying the game program. The following subsections extend the same interface to applications beyond games.

6.2 Appearance Exploration for Scenes and Animations

EdenLive supports exploration of different visual appearances for the same scene or animation. References can come from agent-built blockouts, artist-authored coarse animations, or normal-map videos rendered from detailed geometry (Fig. 16). These inputs provide scene structure and motion, while the target first frame and text prompt specify the desired appearance. Keeping the reference fixed allows users to explore materials, lighting, and color palettes without modifying the underlying scene or animation.



Figure 15 Games under different target appearances. Each row shows the initial reference frame (left) and generated frames at matching times under two appearance conditions (middle and right), using the same reference video. Game inputs in the second and third rows come from SUNBREAK by Imirushik and Fishslop by vasu-devs, respectively.

6.3 Video Re-rendering and Stylization

EdenLive can re-render recorded videos or live camera streams without requiring a scene graph or corresponding 3D assets (Fig. 16). The target first frame and text prompt specify a new appearance, such as cartoon or pixel art, while the reference video supplies the underlying actions and camera motion.

6.4 Embodied Data Augmentation

EdenLive supports visual data augmentation for embodied tasks by varying the appearance of an interaction while retaining its motion and scene layout (Fig. 16). For human egocentric videos, the target first frame and text prompt can specify a robot gripper in place of the human hand [47], a different manipulated object, or changes to object color and material. These transformations provide visual variants of an existing demonstration without requiring the interaction to be recorded again.

7 Limitations and Future Work

EdenLive leaves the mapping between reference objects and their intended identities implicit; explicit object associations could improve control over which entity each proxy represents. Its bounded temporal cache also lacks a dedicated memory for fine appearance details, motivating retrieval of object- or scene-specific visual evidence after extended occlusion or scene revisiting. Finally, the encoder provides a single learned representation without explicit control over which levels of reference detail to preserve. Learning multi-level coarse–fine features could allow users to adjust control strength and granularity, balancing reference fidelity with freedom for visual refinement.

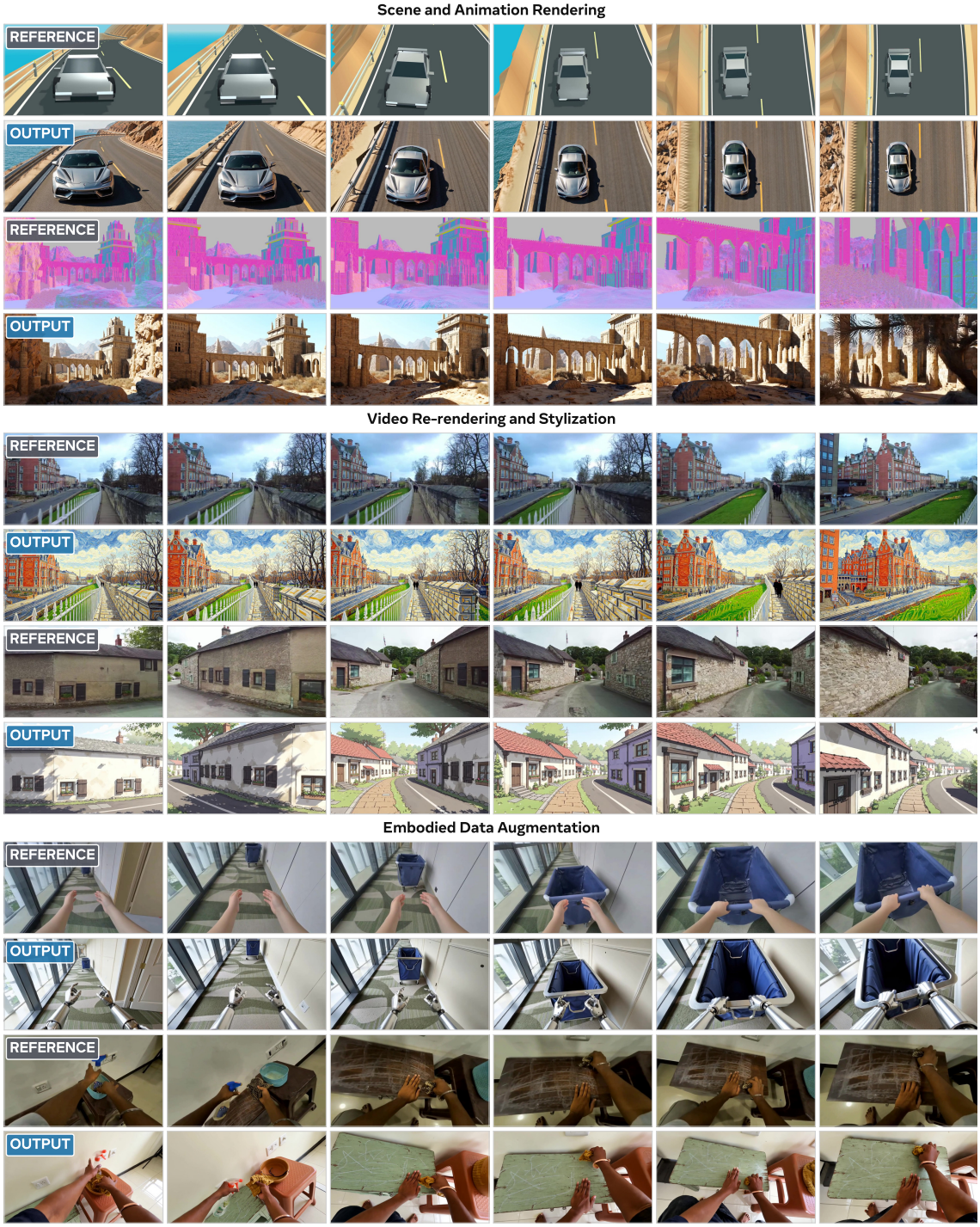


Figure 16 Application gallery. Each group pairs a reference stream with its generated output for scene and animation rendering, video re-rendering and stylization, or embodied data augmentation.

8 Conclusion

We have presented EdenLive, a real-time generative renderer that treats the reference video as the interface between an interactive program and neural visual synthesis: the upstream program executes the rules and maintains the world state, whereas EdenLive renders the resulting reference video, guided by a target first frame and a text prompt, into detailed frames. A fine-tuned vision encoder that maps heterogeneous references into a shared feature space, together with a distilled chunkwise autoregressive generator served in a pipelined system, makes this division practical under real-time constraints. Through their visual references, diverse upstream sources, ranging from graybox game engines to camera recordings, can drive the same renderer, so that a single interface supports agent-built game previews, appearance exploration, video re-rendering, and embodied data augmentation without application-specific designs. We hope that this work encourages viewing the visual reference as a general interface to generative rendering, through which existing engines, tools, and recordings can be connected to neural renderers, and that decoupling state from rendering offers a practical path toward interactive worlds whose logic remains exact and inspectable in code while their appearance is synthesized by generative models.

9 Authors

9.1 Core Contributors

Zehuan Huang, Tian-Xing Xu, Chunli Peng, Cheng Zhang, Zeyi Zheng, Yifei Song, Zekai Gu, Zhipeng Yu, Peng Wang, Yan-Pei Cao.

9.2 Contributors

Yige Chen, Xinyang Li, Ze Yuan, Xiaoyu Zhao.

Acknowledgments

We thank Yuan-Chen Guo, Yumeng Li, Hanxiao Wang, Dongyu Chen, Yang-Tian Sun, and Ding Liang for their valuable discussions and suggestions. We thank Shiqiao Gu, Yang Yong, and Ruihao Gong from LightX2V for their support in optimizing inference on the RTX 5090.

References

1 Niket Agarwal, Arslan Ali, Jon Allen, Martin Antolini, Adeline Aubame, Alisson Azzolini, Junjie Bai, Maciej Bala, Yogesh Balaji, Josh Bapst, et al. Cosmos 3: Omnimodal world models for physical AI. *arXiv preprint arXiv:2606.02800*, 2026.

2 Hassan Abu Alhaija, Jose Alvarez, Maciej Bala, Tiffany Cai, Tianshi Cao, Liz Cha, Joshua Chen, Mike Chen, Francesco Ferroni, Sanja Fidler, et al. Cosmos-Transfer1: Conditional world generation with adaptive multimodal control. *arXiv preprint arXiv:2503.14492*, 2025.

3 Arslan Ali, Junjie Bai, Maciej Bala, Yogesh Balaji, Aaron Blakeman, Tiffany Cai, Jiaxin Cao, Tianshi Cao, Elizabeth Cha, Yu-Wei Chao, et al. World simulation with video foundation models for physical AI. *arXiv preprint arXiv:2511.00062*, 2025.

4 Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and François Fleuret. Diffusion for World Modeling: Visual Details Matter in Atari. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2405.12399.

5 Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, et al. PyTorch 2: Faster machine learning through dynamic Python bytecode transformation and graph compilation. In *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, pages 929–947, 2024.

6 Anthropic. Claude code. https://www.anthropic.com/claude-code, 2026. Version 2.1.285. Accessed October 16, 2026.

7 Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-VL Technical Report. *arXiv preprint arXiv:2511.21631*, 2025.

8 Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-VL Technical Report. *arXiv preprint arXiv:2502.13923*, 2025. URL https://arxiv.org/abs/2502.13923.

9 Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15791–15801, 2025.

10 Aarti Basant, Amlan Kar, Despoina Paschalidou, Fangyin Wei, Francesco Ferroni, Guillermo Garcia Cobo, Haithem Turki, Huan Ling, Jaewoo Seo, James Lucas, Jay Zhangjie Wu, Jialiang Wang, Jonathan Lorraine, Jun Gao, Kai He, Katarina Tothova, Kevin Xie, Michał Tyszkiewicz, Qi Wu, Riccardo de Lutio, Ruilong Li, Sanja Fidler, Seung Wook Kim, Tianchang Shen, Tianshi Cao, Tobias Pfaff, William Lew, Xindi Wu, Xuanchi Ren, Yifan Lu, Yuxuan Zhang, Zan Gojcic, and Zian Wang. NVIDIA OmniDreams: Real-time generative world model for closed-loop autonomous vehicle simulation, 2026. URL https://arxiv.org/abs/2606.03159.

11 Shariq Farooq Bhat, Niloy J. Mitra, and Kalyan Sunkavalli. LooseControlVideo: Directorial Video Control using Spatial Blocking. In *European Conference on Computer Vision*, 2026. URL https://arxiv.org/abs/2606.19495.

12 Black Forest Labs. FLUX.2 [klein] 9B Base. Official Hugging Face model card, 2026. URL https://huggingface.co/black-forest-labs/FLUX.2-klein-base-9B.

13 Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal Behbahani, Stephanie Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative Interactive Environments. In *International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2402.15391.

14 Ryan Burgert, Yuancheng Xu, Wenqi Xian, Oliver Pilarski, Pascal Clausen, Mingming He, Li Ma, Yitong Deng, Lingxiao Li, Mohsen Mousavi, Michael Ryoo, Paul Debevec, and Ning Yu. Go-with-the-Flow: Motion-Controllable Video Diffusion Models Using Real-Time Warped Noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL https://arxiv.org/abs/2501.08331.

15 Shengqu Cai, Duygu Ceylan, Matheus Gadelha, Chun-Hao Paul Huang, Tuanfeng Yang Wang, and Gordon Wetzstein. Generative Rendering: Controllable 4D-Guided Video Generation with 2D Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. URL https://arxiv.org/abs/2312.01409.

16 Ziqi Cai, Siqi Yang, Yimu Wang, Zixian Gao, Yunheng Liu, Shuchen Weng, Erwin Wu, Kaipeng Zhang, and Boxin Shi. Mass: Multiplayer world models with authoritative shared state. *arXiv preprint arXiv:2608.06257*, 2026.

17 Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris Coll-Vinent, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. SAM 3: Segment anything with concepts. In *International Conference on Learning Representations*, volume 2026, pages 138846–138923, 2026.

18 Haoxuan Che, Xuanhua He, Quande Liu, Cheng Jin, and Hao Chen. GameGen-X: Interactive open-world game video generation. In *International Conference on Learning Representations*, 2025.

19 Weifeng Chen, Yatai Ji, Jie Wu, Hefeng Wu, Pan Xie, Jiashi Li, Xin Xia, Xuefeng Xiao, and Liang Lin. Control-A-Video: Controllable Text-to-Video Diffusion Models with Motion Prior and Reward Feedback Learning. *arXiv preprint arXiv:2305.13840*, 2023. URL https://arxiv.org/abs/2305.13840.

20 Yiwen Chen, Guosheng Lin, and Chi Zhang. Code world model: Coding agent as world brain. *arXiv preprint arXiv:2608.25927*, 2026. URL https://arxiv.org/abs/2608.25927.

21 Jiawei Chi, Shangchen Miao, Zhiyuan Shi, Kailu Wu, Hanyang Wang, Weiliang Chen, Qiyu Dai, Chengyang Li, Jinshan Ren, Jun Gao, Mingsheng Long, Yueqi Duan, Jiangran Lyu, Jialong Wu, and Fangfu Liu. AgentGarten: Code Worlds for Evolving Agents. *arXiv preprint arXiv:2610.12374*, 2026. URL https://arxiv.org/abs/2610.12374.

22 Justin Cui, Jie Wu, Ming Li, Tao Yang, Xiaojie Li, Rui Wang, Andrew Bai, Yuanhao Ban, and Cho-Jui Hsieh. Self-Forcing++: Towards minute-scale high-quality video generation. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2510.02283.

23 DreamX Team, Yancheng Bai, Rui Chen, Xiangxiang Chu, Rujing Dang, Hao Dou, Bingjie Gao, Qiwen Gu, Siyu Hong, Jiachen Lei, Geng Li, Jifan Li, Ruimin Lin, Qingfeng Shi, Bingze Song, Lei Sun, Jing Tang, Ruitian Tian, Jun Wang, Jiahong Wu, Pengfei Zhang, Shen Zhang, and Jiashu Zhu. DreamX-World 1.0: A General-Purpose Interactive World Model. *arXiv preprint arXiv:2606.16993*, 2026. URL https://arxiv.org/abs/2606.16993.

24 Zixun Fang, Yawen Shao, Kai Zhu, Jie Xiao, Shihan Chen, Yu Liu, Xueyang Fu, Yang Cao, Wei Zhai, and Zheng-Jun Zha. Code plans, diffusion renders: Open-ended generative world modeling. *arXiv preprint arXiv:2609.26458*, 2026. URL https://arxiv.org/abs/2609.26458.

25 Tianrui Feng, Zhi Li, Shuo Yang, Haocheng Xi, Muyang Li, Xiuyu Li, Lvmin Zhang, Keting Yang, Kelly Peng, Song Han, Maneesh Agrawala, Kurt Keutzer, Akio Kodaira, and Chenfeng Xu. StreamDiffusionV2: A Streaming System for Dynamic and Interactive Video Generation. In *Proceedings of Machine Learning and Systems*, 2026. URL https://arxiv.org/abs/2511.07399.

26 FlashDreams Contributors. Flashdreams: High-performance inference and serving for interactive autoregressive video and world models. https://github.com/NVIDIA/flashdreams, 2026.

27 Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. DreamSim: Learning new dimensions of human visual similarity using synthetic data. In *Advances in Neural Information Processing Systems*, volume 36, pages 50742–50768, 2023. URL https://arxiv.org/abs/2306.09344.

28 Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17:1–35, 2016. URL https://arxiv.org/abs/1505.07818.

29 Zelin Gao, Qiuyu Wang, Jiapeng Zhu, Jingye Chen, Zichen Liu, Qingyan Bai, Jiahao Wang, Yufeng Yuan, Hanlin Wang, Yichong Lu, Ka Leong Cheng, Haojie Zhang, Jian Gao, Tianrui Feng, Yuzheng Liu, Yao Yao, Yinghao Xu, Xing Zhu, Yujun Shen, and Hao Ouyang. Infinite Worlds with Versatile Interactions. *arXiv preprint arXiv:2607.07534*, 2026. URL https://arxiv.org/abs/2607.07534.

30 Gonzalo Gomez-Nogales, Yicong Hong, Chongjian Ge, Peiye Zhuang, Marc Comino-Trinidad, Dan Casas, and Yi Zhou. Coarse-to-real: Generative rendering for populated dynamic scenes. *arXiv preprint arXiv:2601.22301*, 2026. URL https://arxiv.org/abs/2601.22301.

31 Zekai Gu, Rui Yan, Jiahao Lu, Peng Li, Zhiyang Dou, Chenyang Si, Zhen Dong, Qifeng Liu, Cheng Lin, Ziwei Liu, Wenping Wang, and Yuan Liu. Diffusion as Shader: 3D-aware Video Diffusion for Versatile Video Generation Control. In *ACM SIGGRAPH Conference Papers*, 2025. URL https://arxiv.org/abs/2501.03847.

32 David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

33 Yoav HaCohen, Benny Brazowski, Nisan Chiprut, Yaki Bitterman, Andrew Kvochko, Avishai Berkowitz, Daniel Shalem, Daphna Lifschitz, Dudu Moshe, Eitan Porat, et al. LTX-2: Efficient joint audio-visual foundation model. *arXiv preprint arXiv:2601.03233*, 2026.

34 Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

35 Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. CameraCtrl: Enabling Camera Control for Video Diffusion Models. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2404.02101.

36 Xianglong He, Chunli Peng, Zexiang Liu, Boyang Wang, Yifan Zhang, Qi Cui, Fei Kang, Biao Jiang, Mengyin An, Yangyang Ren, et al. Matrix-game 2.0: An open-source real-time and streaming interactive world model.

arXiv preprint arXiv:2508.13009, 2025.

37 Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. GAIA-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023.

38 Jiahui Huang, Qunjie Zhou, Hesam Rabeti, Aleksandr Korovko, Huan Ling, Xuanchi Ren, Tianchang Shen, Jun Gao, Dmitry Slepichev, Chen-Hsuan Lin, Jiawei Ren, Kevin Xie, Joydeep Biswas, Laura Leal-Taixé, and Sanja Fidler. ViPE: Video Pose Engine for 3D Geometric Perception. *arXiv preprint arXiv:2508.10934*, 2025. URL https://arxiv.org/abs/2508.10934.

39 Junchao Huang, Guian Fang, Shengju Qian, Xianghao Kong, Zhuoran Zhao, Wei Huang, Yihua Du, Zixin Zhang, Justin Cui, Yuchao Gu, Yukang Chen, Xinting Hu, Tianyu He, Shaoshuai Shi, Zhuotao Tian, Xin Wang, Mike Zheng Shou, and Li Jiang. SolarWM: Open Data and Scalable Training for Long-Horizon Video World Models. *arXiv preprint arXiv:2609.02886*, 2026. URL https://arxiv.org/abs/2609.02886.

40 Lianghua Huang, Wei Wang, Zhi-Fan Wu, Yupeng Shi, Huanzhang Dou, Chen Liang, Yutong Feng, Yu Liu, and Jingren Zhou. In-context LoRA for diffusion transformers. *arXiv preprint arXiv:2410.23775*, 2024.

41 Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *Advances in Neural Information Processing Systems*, 38:167283–167308, 2026.

42 Zheng-Hui Huang, Guixu Lin, Jiacheng Lin, Yi-Chuan Huang, Ruihan Yu, Muyao Niu, Siqi Yang, Yu-Lun Liu, Yung-Yu Chuang, Kaipeng Zhang, and Zhixiang Wang. Programmable world model. *arXiv preprint arXiv:2609.10540*, 2026. URL https://arxiv.org/abs/2609.10540.

43 Zheng-Hui Huang, Zhixiang Wang, Jiaming Tan, Ruihan Yu, Yidan Zhang, Bo Zheng, Yu-Lun Liu, Yung-Yu Chuang, and Kaipeng Zhang. Generative world renderer. *arXiv preprint arXiv:2604.02329*, 2026.

44 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. URL https://arxiv.org/abs/2411.13503.

45 Sam Ade Jacobs, Masahiro Tanaka, Chengming Zhang, Minjia Zhang, Shuaiwen Leon Song, Samyam Rajbhandari, and Yuxiong He. DeepSpeed Ulysses: System optimizations for enabling training of extreme long sequence transformer models. *arXiv preprint arXiv:2309.14509*, 2023. URL https://arxiv.org/abs/2309.14509.

46 Jitesh Jain, Jiachen Li, MangTik Chiu, Ali Hassani, Nikita Orlov, and Humphrey Shi. OneFormer: One Transformer to Rule Universal Image Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. URL https://github.com/SHI-Labs/OneFormer.

47 Jinho Jeong, Se June Joo, Jaehyun Kang, Dongyun Kim, Yena Kim, Hanjung Kim, and Seon Joo Kim. HuRo: Robotizing human videos for scalable VLA pretraining. *arXiv preprint arXiv:2609.10706*, 2026. URL https://arxiv.org/abs/2609.10706. Accepted at CoRL 2026.

48 Fan Jiang, Zhaoxu Sun, Mengchao Wang, Ziyu Zhu, Chiyu Wang, Yunpeng Zhang, Wenlin Liu, Yun Wang, Xue Zheng, Rui Sun, et al. ABot-World-0: Infinite interactive world rollout on a single desktop GPU. *arXiv preprint arXiv:2607.19191*, 2026.

49 Zeyinzi Jiang, Zhen Han, Chaojie Mao, Jingfeng Zhang, Yulin Pan, and Yu Liu. VACE: All-in-One Video Creation and Editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025. URL https://arxiv.org/abs/2503.07598.

50 Geonung Kim, Janghyeok Han, and Sunghyun Cho. VideoFrom3D: 3D Scene Video Generation via Complementary Image and Video Diffusion Models. In *SIGGRAPH Asia Conference Papers*, 2025. URL https://arxiv.org/abs/2509.17985.

51 Akio Kodaira, Chenfeng Xu, Toshiki Hazama, Takanori Yoshimoto, Kohei Ohno, Shogo Mitsuhori, Soichi Sugano, Hanying Cho, Zhijian Liu, Masayoshi Tomizuka, and Kurt Keutzer. StreamDiffusion: A Pipeline-level Solution for Real-time Interactive Generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025. URL https://arxiv.org/abs/2312.12491.

52 Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. HunyuanVideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

53 Max Ku, Cong Wei, Weiming Ren, Harry Yang, and Wenhu Chen. AnyV2V: A Tuning-Free Framework For Any Video-to-Video Editing Tasks. *Transactions on Machine Learning Research*, 2024. URL https://arxiv.org/abs/2403.14468.

54 Jiaqi Li, Junshu Tang, Zhiyong Xu, Longhuang Wu, Yuan Zhou, Shuai Shao, Tianbao Yu, Zhiguo Cao, and Qinglin Lu. Hunyuan-GameCraft: High-dynamic Interactive Game Video Generation with Hybrid History Condition. *arXiv preprint arXiv:2506.17201*, 2025. URL https://arxiv.org/abs/2506.17201.

55 Yitong Li, Junsong Chen, Haopeng Li, Haozhe Liu, Jincheng Yu, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Sol Video Inference Engine: Agent-Native Full-Stack Acceleration Framework for Efficient Video Generation. *arXiv preprint arXiv:2606.23743*, 2026. URL https://arxiv.org/abs/2606.23743.

56 Zhen Li, Zuo-Liang Zhu, Ling-Hao Han, Qibin Hou, Chun-Le Guo, and Ming-Ming Cheng. AMT: All-pairs multi-field transforms for efficient frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. URL https://arxiv.org/abs/2304.09790.

57 Feng Liang, Akio Kodaira, Chenfeng Xu, Masayoshi Tomizuka, Kurt Keutzer, and Diana Marculescu. Looking Backward: Streaming Video-to-Video Translation with Feature Banks. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2405.15757.

58 Ruofan Liang, Zan Gojcic, Huan Ling, Jacob Munkberg, Jon Hasselgren, Zhi-Hao Lin, Jun Gao, Alexander Keller, Nandita Vijaykumar, Sanja Fidler, and Zian Wang. DiffusionRenderer: Neural Inverse and Forward Rendering with Video Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL https://arxiv.org/abs/2501.18590.

59 Tingting Liao and Guangyi Liu. Streaming Code World Model. Official code repository, 2026. URL https://github.com/TingtingLiao/streaming-code-world-model.

60 Lightricks. LTX-2.5 22B IC-LoRA Layout to Render. Official Hugging Face model card, 2026. URL https://huggingface.co/Lightricks/LTX-2.5-22b-IC-LoRA-Layout-To-Render.

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

62 Guixu Lin, Zheng-Hui Huang, Siqi Yang, Ming-Hsuan Yang, Kaipeng Zhang, and Zhixiang Wang. Generative world renderer at the speed of play. *arXiv preprint arXiv:2607.18703*, 2026. URL https://arxiv.org/abs/2607.18703.

63 Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth Anything 3: Recovering the Visual Space from Any Views. *arXiv preprint arXiv:2511.10647*, 2025. URL https://arxiv.org/abs/2511.10647.

64 Zhiqiu Lin, Deepak Pathak, Baiqi Li, Jiayao Li, Xide Xia, Graham Neubig, Pengchuan Zhang, and Deva Ramanan. Evaluating text-to-visual generation with image-to-text generation. In *European Conference on Computer Vision*, pages 366–384. Springer, 2024.

65 Zijun Lin, Zhiyang Deng, Yuzhe Wu, Bihan Wen, and Yeying Jin. GameDirector: Decoupling gameplay logic from rendering for player-configurable game world models. *arXiv preprint arXiv:2609.25652*, 2026. URL https://arxiv.org/abs/2609.25652.

66 Zijun Lin, Zeqing Wang, Cheston Tan, Bihan Wen, and Yeying Jin. StatePlay: State-aware game world models for mechanics-consistent generation. *arXiv preprint arXiv:2607.26754*, 2026. URL https://arxiv.org/abs/2607.26754.

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

68 Yanzuo Lu, Tian Ye, Yitong Li, Shuchen Xue, Haozhe Liu, Song Han, and Enze Xie. StreamRender-H3: Interactive Coding Worlds with Real-time Streaming Diffusion Render. Official project page, 2026. URL https://nvlabs.github.io/Sana/Sol-Engine/StreamRender-H3/.

69 Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent Consistency Models: Synthesizing High-Resolution Images with Few-Step Inference. *arXiv preprint arXiv:2310.04378*, 2023. URL https://arxiv.org/abs/2310.04378.

70 Zian Meng, Zhen Li, Chuanhao Li, Qiang Li, and Kaipeng Zhang. Marionette: Predicting world states, rendering geometry, painting appearance. *arXiv preprint arXiv:2608.14530*, 2026.

71 MiniMax. MiniMax H3: An Open Model Breaking the Boundaries Between Tasks and Modalities. Official model release, 2026. URL https://www.minimax.io/blog/minimax-h3.

72 Wenqi Ouyang, Yi Dong, Lei Yang, Jianlou Si, and Xingang Pan. I2VEdit: First-Frame-Guided Video Editing via Image-to-Video Diffusion Models. In *SIGGRAPH Asia Conference Papers*, 2024. URL https://arxiv.org/abs/2405.16537.

73 PaddlePaddle Authors. PaddleOCR. https://github.com/PaddlePaddle/PaddleOCR, 2026. Accessed October 13, 2026.

74 Jack Parker-Holder and Shlomi Fruchter. Genie 3: A new frontier for world models. Google DeepMind research blog, 2025. URL https://deepmind.google/blog/genie-3-a-new-frontier-for-world-models/. Published August 5, 2025; accessed October 12, 2026.

75 Runjia Qian, Zile Wang, Jihai Zhang, Kai Zou, Wei Yu, Jiaxing Li, Zexiang Liu, Yaokun Li, Fei Kang, Kaichen Huang, Mengyin An, Haobo Zhang, Biao Jiang, Jiahua Wang, Haofeng Sun, Yang Liu, and Yangguang Li. Matrix-Game 3.5: Enhancing Real-Time Streaming Interactive World Models with Patch Memory. *arXiv preprint arXiv:2608.29910*, 2026. URL https://arxiv.org/abs/2608.29910.

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

77 Xuanchi Ren, Tianchang Shen, Jiahui Huang, Huan Ling, Yifan Lu, Merlin Nimier-David, Thomas Müller, Alexander Keller, Sanja Fidler, and Jun Gao. GEN3C: 3d-informed world-consistent video generation with precise camera control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6121–6132, 2025. URL https://arxiv.org/abs/2503.03751.

78 Stephan R. Richter, Hassan Abu AlHaija, and Vladlen Koltun. Enhancing photorealism enhancement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):1700–1715, 2022.

79 Robbyant Team, Zelin Gao, Qiuyu Wang, Yanhong Zeng, Jiapeng Zhu, Ka Leong Cheng, Yixuan Li, Hanlin Wang, Yinghao Xu, Shuailei Ma, Yihang Chen, Jie Liu, Yansong Cheng, Yao Yao, Jiayi Zhu, Yihao Meng, Kecheng Zheng, Qingyan Bai, Jingye Chen, Zehong Shen, Yue Yu, Xing Zhu, Yujun Shen, and Hao Ouyang. Advancing Open-source World Models. *arXiv preprint arXiv:2601.20540*, 2026. URL https://arxiv.org/abs/2601.20540.

80 Team Seedance, De Chen, Liyang Chen, Xin Chen, Ying Chen, Zhuo Chen, Zhuowei Chen, Feng Cheng, Tianheng Cheng, Yufeng Cheng, et al. Seedance 2.0: Advancing video generation for world complexity. *arXiv preprint arXiv:2604.14148*, 2026.

81 Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *Transactions on Machine Learning Research*, 2026. URL https://arxiv.org/abs/2508.10104.

82 Tomás Soucek and Jakub Lokoc. TransNet V2: An effective deep network architecture for fast shot transition detection. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 11218–11221, 2024.

83 Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

84 Wenqiang Sun, Haiyu Zhang, Haoyuan Wang, Junta Wu, Zehan Wang, Zhenwei Wang, Yunhong Wang, Jun Zhang, Tengfei Wang, and Chunchao Guo. WorldPlay: Towards long-term geometric consistency for real-time interactive world modeling. *arXiv preprint arXiv:2512.14614*, 2025.

85 Team Wan, Ang Wang, Baole Ai, Bin Wen, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. URL https://arxiv.org/abs/2503.20314.

86 Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. Diffusion models are real-time game engines. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2408.14837.

87 Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. URL https://arxiv.org/abs/1807.03748.

88 VAST AI Research. Project eden. Research blog, 2026. URL https://www.tripo3d.ai/research/project-eden. Published May 31, 2026; accessed October 12, 2026.

89 Jianyuan Wang, Minghao Chen, Shangzhan Zhang, Nikita Karaev, Johannes Schönberger, Patrick Labatut, Piotr Bojanowski, David Novotny, Andrea Vedaldi, and Christian Rupprecht. VGGT-Ω. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

90 Xiang Wang, Hangjie Yuan, Shiwei Zhang, Dayou Chen, Jiuniu Wang, Yingya Zhang, Yujun Shen, Deli Zhao, and Jingren Zhou. VideoComposer: Compositional Video Synthesis with Motion Controllability. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2306.02018.

91 Zhouxia Wang, Ziyang Yuan, Xintao Wang, Tianshui Chen, Menghan Xia, Ping Luo, and Ying Shan. MotionCtrl: A Unified and Flexible Motion Controller for Video Generation. In *ACM SIGGRAPH Conference Papers*, 2024. URL https://arxiv.org/abs/2312.03641.

92 Zile Wang, Zexiang Liu, Jiaxing Li, Kaichen Huang, Baixin Xu, et al. Matrix-Game 3.0: Real-time and streaming interactive world model with long-horizon memory. *arXiv preprint arXiv:2604.08995*, 2026. URL https://arxiv.org/abs/2604.08995.

93 Haoning Wu, Erli Zhang, Liang Liao, Chaofeng Chen, Jingwen Hou, Annan Wang, Wenxiu Sun, Qiong Yan, and Weisi Lin. Exploring video quality assessment on user generated contents from aesthetic and technical perspectives. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. URL https://arxiv.org/abs/2211.04894.

94 Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, 2024.

95 Zeqi Xiao, Yushi Lan, Yifan Zhou, Wenqi Ouyang, Shuai Yang, Yanhong Zeng, and Xingang Pan. WorldMem: Long-term consistent world simulation with memory. *Advances in Neural Information Processing Systems*, 38: 49632–49652, 2026.

96 Zhexiao Xiong, Yizhi Song, Hao Kang, Qing Yan, Liming Jiang, Jenson Yang, Zhoujie Fu, Stathi Fotiadis, Angtian Wang, Zichuan Liu, Bo Liu, Yiding Yang, Xin Lu, and Nathan Jacobs. ActWorld: From explorable to interactive world model via action-aware memory. *arXiv preprint arXiv:2606.17730*, 2026.

97 Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. GMFlow: Learning optical flow via global matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. URL https://arxiv.org/abs/2111.13680.

98 Hongli Xu, Weilong Yan, Anbang Wang, Chunyu Zou, Siyu Hong, and Jingwei Huang. Proxy2World: Learning to generate worlds from lightweight proxies without seeing them. *arXiv preprint arXiv:2609.35023*, 2026. URL https://arxiv.org/abs/2609.35023.

99 Tian-Xing Xu, Zi-Xuan Wang, Guangyuan Wang, Li Hu, Zhongyi Zhang, Peng Zhang, Bang Zhang, and Song-Hai Zhang. UCM: Unified modeling of camera control and memory with time-aware positional encoding warping for world models. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pages 1–12, 2026.

100 Sherry Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Leslie Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. In *International Conference on Learning Representations*, 2024.

101 Shuai Yang, Wei Huang, Ruihang Chu, Yicheng Xiao, Yuyang Zhao, Xianbang Wang, Muyang Li, Enze Xie, Yingcong Chen, Yao Lu, Song Han, and Yukang Chen. LongLive: Real-time interactive long video generation. In *International Conference on Learning Representations*, 2026.

102 Xindi Yang, Baolu Li, Liam Lee, Zhenfei Yin, Songxin Zhang, Zhuoyang Song, Xu Jia, Jianfei Cai, Tien-Tsin Wong, Bingyi Jing, et al. Oneira: From open-ended generation to open-world interaction in video world models. *arXiv preprint arXiv:2610.01614*, 2026.

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

104 Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and William T. Freeman. Improved Distribution Matching Distillation for Fast Image Synthesis. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2405.14867.

105 Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T. Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. URL https://arxiv.org/abs/2311.18828.

106 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 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. URL https://arxiv.org/abs/2412.07772.

107 Shenghai Yuan, Yuanyang Yin, Zongjian Li, Xinwei Huang, Xiao Yang, and Li Yuan. Helios: Real Real-Time Long Video Generation Model. *arXiv preprint arXiv:2603.04379*, 2026. URL https://arxiv.org/abs/2603.04379.

108 Xiaoyu Zhan, Xinyu Wang, Xiaohong Zhang, Huanjie Zhu, Tengjiao Sun, Pengcheng Fang, Jiaxing Yu, Yanwen Guo, and Dongjie Fu. Magpie: Real-time world renderer for interactive games. *arXiv preprint arXiv:2608.27168*, 2026. URL https://arxiv.org/abs/2608.27168.

109 Haiyu Zhang, Wenqiang Sun, Tengfei Wang, Junta Wu, Jun Zhang, Yunhong Wang, Yu Qiao, and Chunchao Guo. WorldPlay2: Extending Real-Time Interactive World Models in Control and Horizon. *arXiv preprint arXiv:2609.35560*, 2026. URL https://arxiv.org/abs/2609.35560.

110 Songchun Zhang, Yaowei Li, Junhao Zhuang, Weiyang Jin, Haoyu Wang, Xin Lu, Yilang Sun, Shiyi Zhang, Haoran Li, Xiaoxiao Ma, Yuming Li, Yijun Liu, Yaofeng Su, Yanwen Ma, Haoyu Wu, Zihan Su, Yue Ma, Lvmin Zhang, Haoyang Huang, Zeyue Xue, Anyi Rao, and Nan Duan. EchoWM: Open and Enterable Omnimodal World Models. *arXiv preprint arXiv:2608.23189*, 2026. URL https://arxiv.org/abs/2608.23189.

111 Yifan Zhang, Chunli Peng, Boyang Wang, Puyi Wang, Qingcheng Zhu, Fei Kang, Biao Jiang, Zedong Gao, Eric Li, Yang Liu, et al. Matrix-Game: Interactive world foundation model. *arXiv preprint arXiv:2506.18701*, 2025.

112 Min Zhao, Hongzhou Zhu, Bokai Yan, Zihan Zhou, Yimin Chen, Honglie Wang, Wenqiang Sun, Zhengwei Fang, Zizheng Xun, Zihao Li, Kaiwen Zheng, Guande He, Xiao Yang, Chongxuan Li, Fan Bao, and Jun Zhu. minWM: A Full-Stack Open-Source Framework for Real-Time Interactive Video World Models. *arXiv preprint arXiv:2605.30263*, 2026. URL https://arxiv.org/abs/2605.30263.

113 Min Zhao, Hongzhou Zhu, Kaiwen Zheng, Zihan Zhou, Bokai Yan, Xinyuan Li, Xiao Yang, Chongxuan Li, and Jun Zhu. Causal Forcing++: Scalable few-step autoregressive diffusion distillation for real-time interactive video generation. *arXiv preprint arXiv:2605.15141*, 2026. URL https://arxiv.org/abs/2605.15141.

114 Yuyang Zhao, Yicheng Pan, Qiyuan He, Jincheng Yu, Junsong Chen, Tian Ye, Haozhe Liu, Enze Xie, and Song Han. SANA-Streaming: Real-time Streaming Video Editing with Hybrid Diffusion Transformer. *arXiv preprint arXiv:2605.30409*, 2026. URL https://arxiv.org/abs/2605.30409.

115 Zhuoran Zhao, Shengju Qian, Tongtong Liang, Xianghao Kong, Songchun Zhang, Junchao Huang, Guian Fang, Xin Wang, Pan Hui, and Anyi Rao. Mask Forcing: Improving autoregressive video diffusion distillation via dual-noise masking rollout. *arXiv preprint arXiv:2609.09123*, 2026.

116 Kaiwen Zheng, Guande He, Min Zhao, Jintao Zhang, Huayu Chen, Jianfei Chen, Chen-Hsuan Lin, Ming-Yu Liu, Jun Zhu, and Qianli Ma. Causal-rCM: A unified teacher-forcing and self-forcing open recipe for autoregressive diffusion distillation in streaming video generation and interactive world models. *arXiv preprint arXiv:2606.25473*, 2026.

117 Zanwei Zhou, Jiazhong Cen, Jiemin Fang, Yumeng He, Chen Yang, Sikuang Li, Fanpeng Meng, Zhikuan Bao, Wei Shen, and Qi Tian. ProxyUp: Training-Free Proxy-Conditioned Video Generation for Controllable Dynamics. *arXiv preprint arXiv:2607.03732*, 2026. URL https://arxiv.org/abs/2607.03732.

118 Haoyi Zhu, Haozhe Liu, Yuyang Zhao, Tian Ye, Junsong Chen, Jincheng Yu, Tong He, Song Han, and Enze Xie. SANA-WM: Efficient Minute-Scale World Modeling with Hybrid Linear Diffusion Transformer. *arXiv preprint arXiv:2605.15178*, 2026. URL https://arxiv.org/abs/2605.15178.

119 Hongzhou Zhu, Min Zhao, Guande He, Hang Su, Chongxuan Li, and Jun Zhu. Causal forcing: Autoregressive diffusion distillation done right for high-quality real-time interactive video generation. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2602.02214.

120 Junhao Zhuang, Shiyi Zhang, Yuxuan Bian, Yaowei Li, Yawen Luo, Yijun Liu, Weiyang Jin, Songchun Zhang, Xianglong He, Xuying Zhang, Haoran Li, Haoyang Huang, Zeyue Xue, and Nan Duan. Self gradient forcing: Native long video extrapolation. *arXiv preprint arXiv:2607.20368*, 2026. URL https://arxiv.org/abs/2607.20368.