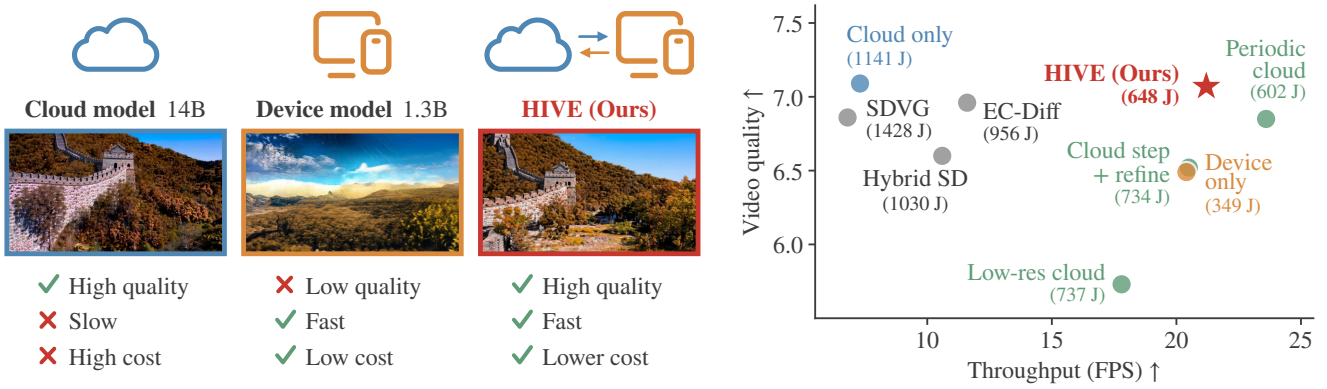
HIVE: Heterogeneous Collaboration for Video World Models

Hyunsoo Kim Guandao Yang Diana Marculescu
The University of Texas at Austin



(a) Either model alone *vs.* their collaboration

(b) Quality *vs.* throughput (J: total energy)

Figure 1. **HIVE splits interactive world generation between an edge device and the cloud.** (a) A large world model in the cloud gives high video quality but is slow and costly. A small model on the edge device is fast and cheap but drifts away from the scene, here 29 s into the same rollout. HIVE lets the two collaborate, so that the video stays close to the quality of the cloud model at the speed of the device model and a lower cost. (b) Video quality, the score of a VLM judge [2] on WorldScore-300 [10], against throughput (neither axis starts at zero), with the total GPU energy per video second under each name. Blue and orange mark the cloud and the device model alone, gray the published two-model methods (Hybrid SD [44], EC-Diff [42], and SDVG [16]), green other collaboration designs, and the red star HIVE. HIVE approaches the quality of the cloud model at 2.9× its throughput and 35% of its cloud energy (398 J).

Abstract

Interactive video world models let users explore generated environments, empowering applications from games to robotics. To achieve good quality, video world models often need to run on cloud GPUs, which can be costly for model deployers. Cloud deployment can also lead to higher response latency. A small on-device model can respond locally, but it loses details and drifts over long rollouts. Motivated by the complementary strengths of cloud and edge computing, we study how to deploy a video world model across heterogeneous edge devices and cloud GPUs. Heterogeneous deployment is challenging because additional cloud computation can improve video quality but increases serving cost and can delay responses to user inputs. We present HIVE, a method that combines one-step cloud prediction with on-device decoding and regeneration after a control change. HIVE achieves judged video quality close to that of the four-step cloud model. Compared with four-step cloud-only generation, it achieves 2.9× *the throughput and uses 65% less cloud GPU energy per second of generated video.*

1. Introduction

Interactive video world models [3, 4, 13, 15] generate environments that users can explore and control through their inputs. These environments can support games, simulation, and the training of embodied agents. Recent models [17, 46] achieve real-time throughput on powerful cloud GPUs, but users typically access them through edge devices. Users expect continuous playback, low response latency, and photometrically and physically realistic videos. Yet models that achieve good quality are often large and require cloud GPUs to run efficiently. This motivates us to study heterogeneous deployment across edge devices and cloud GPUs. Cloud models can provide good quality but incur serving costs and communication latency, while smaller on-device models respond locally but can lose scene details over long rollouts. We study how the two models should collaborate to achieve good quality and responsiveness with lower cloud energy consumption.

We consider models that generate video in short groups of frames called chunks. Each chunk is conditioned on the preceding video and a control input, such as a move-

ment direction or no movement. The input can remain unchanged across multiple frames. A control change occurs when it is updated. Each chunk is generated through denoising steps that predict a clean video latent from a noisy input. A model can run entirely in the cloud or on the user's device. Some work studies how to pair a large and a small diffusion model. Hybrid SD [44] and RISE [20] divide the denoising steps of each sample between the two models. SDVG [16] instead uses the small model to generate drafts and the large model to regenerate rejected drafts. These methods motivate studying which computations should run on each model when every chunk must meet a playback deadline.

In this paper, we systematically study how a large cloud model and a small on-device model should collaborate. We vary the cloud sampling depth, the device's role, and the generation history used by each model. Two findings guide our design. *First, one cloud step retains much of the judged quality on LingBot-World.* Its one-step 14B model scores 7.03 against 7.09 for four steps and 6.49 for the four-step 1.3B model under the VLM evaluation defined below (Table 1). We use this result to reduce cloud sampling depth for LingBot-World. *Second, the small model can regenerate a chunk using cloud-generated history.* On our 20-scene analysis set, its regenerated chunks are sharper with this history than when it generates the rollout alone (Appendix D.2). We therefore use the small model to regenerate chunks after control changes.

Based on these findings, we propose HIVE, which combines one-step cloud lookahead with on-device decoding and regeneration (Figs. 1 and 2). Both pretrained models retain their weights. While the current chunk is displayed, the cloud predicts the next chunk assuming that the current control input persists. It sends the x_0 prediction from its first denoising step, an estimate of the clean video latent. The VAE decoder on the device converts this latent into video frames, so the cloud does not run the decoder. If the input changes, the device discards the prediction made under the previous input. The on-device model regenerates the chunk with all four denoising steps under the new input, without waiting for a cloud response. Every displayed chunk enters both models' histories, so the cloud continues from the regenerated chunk. More frequent control changes reduce the number of cloud predictions that can be reused. A device that cannot run the small model at the playback rate retains only the decoder and leaves regeneration to the cloud.

We evaluate HIVE on scripted rollouts with control changes every six seconds that are not provided to the cloud in advance. This tests regeneration when a prediction no longer matches the current input. Recorded action sequences known in advance test the one-step cloud stream without regeneration. A second model pair tests whether the collaboration design extends beyond LingBot-World. On 300 scenes sampled from WorldScore [10], HIVE achieves a mean visual-quality score of 7.07 out of 10 under a Qwen3-VL-32B [2] evaluator, compared with 7.09 for four-step cloud-only generation. Appendix B specifies the scoring protocol. Throughput rises from 7.3 to 21.2 FPS, while cloud GPU energy falls from 1141 to 398 J per video second (Table 1). The ratios are 2.904 and 2.867, respectively, so both round to $2.9\times$. During a network outage, the device also generates chunks that cannot arrive from the cloud. With cloud messages withheld for 20 s, playback stalls for 0.29 s per clip, compared with 19.33 s without the device model (Table 4).

Our contributions are summarized below.

- We study how cloud sampling depth, device computation, and generation history affect the quality and cost of collaborative video generation.
- We propose HIVE, which combines one-step cloud lookahead with on-device decoding and regeneration under updated control inputs.
- We evaluate the design on scripted control sequences, recorded actions, and a second model pair, and measure the role of local generation in maintaining playback during injected network outages.

2. Related Work

Interactive video world models. Video world models [4, 13, 15, 24, 25, 32, 37, 49] generate how an environment evolves in response to actions, and recent ones build on large video diffusion models. They range from neural game engines such as GameNGen [39] and Oasis [9] and generative interactive environments such as Genie [4] and Genie 3 [32] to navigation and open-world exploration under keyboard and camera control, as in Navigation World Models [3], Matrix-Game 2.0 [15], and LingBot-World [13]. Long-horizon memory keeps the scene consistent in Matrix-Game 3.0 [40] and WorldCrafter [48]. Real-time generation builds on few-step chunk-wise autoregressive models. CausVid [46] distills a bidirectional video model into a causal generator, Self Forcing [17] trains on the model's own rollouts, and One-Forcing [11] stabilizes one-step autoregressive generation. A line of work makes one such model cheaper, with feature caching [12, 35], KV-cache compression [21, 41], sparse attention [26, 43], quantization [6], or models sized for local deployment [33]. Several studies also report that denoising steps are not equally useful. DIAMOND [1] finds that deterministic transitions need fewer steps than multimodal ones, DIDO [27] that static content forms in the first step while interaction dynamics form later, and Speculative Forcing [8], which exits tokens early inside one generator, notes that viewpoint changes require more steps. All of these works improve a single generator on a single machine. In contrast, we keep both a small and a large model unchanged and use the observation about

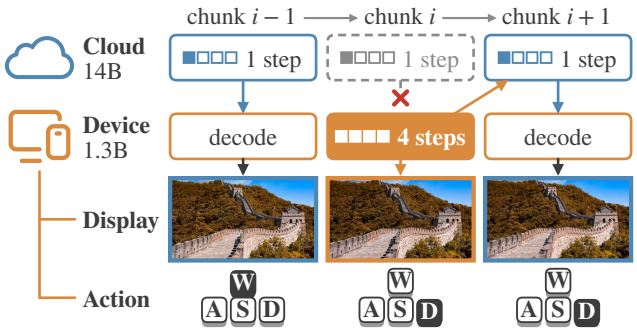


Figure 2. **Overview of HIVE** on three consecutive chunks around an action change. While the action stays the same (chunk $i-1$), the cloud predicts the next chunk under the last action it received with one denoising step, and the device decodes and displays this estimate. When the action changes (chunk i), the estimate that was predicted under the old action is invalid. The device regenerates the chunk under the new action with its own four-step sampler and uploads it, and the cloud continues from it (chunk $i+1$). Both models write only the displayed chunks into their caches. The border of a frame shows which model generated the chunk.

denoising steps to decide which of the two generates each chunk.

Collaborative sampling with a large and a small model. A second line of work lets two diffusion models share the denoising trajectory of every sample. Some of these methods target edge–cloud deployment. Hybrid SD [44] and RISE [20] run the early steps on a large cloud model and the late steps on a small edge model, EC-Diff [42] extrapolates part of the cloud steps, and EcoVideo [5] generates keyframes in the cloud and interpolates the remaining frames on the device, which needs future frames. For autoregressive video, SDVG [16] lets a large model regenerate the drafts of a small model that a reward model rejects. Others share the steps between the two models to cut the sampling cost on one machine. T-Stitch [31] runs the small model first, SRDiffusion [7] switches from the large to the small model adaptively, FlowBlending [34] gives the large model the first and last steps, TRACK [30] calibrates the model of each step offline, and HybridStitch [36] selects the model per token. Except for SDVG, these methods were designed for offline generation with many steps, and each of them keeps the large model involved in every sample or verifies every draft. In contrast, HIVE follows the action stream of an interactive video. It displays the one-step estimate of the large model directly and runs a full sampler only where the action changes.

Speculation in interactive systems. Interactive systems hide latency by computing ahead of the user. Outatime [23] renders cloud-gaming frames for likely inputs and corrects mispredictions. For video generation, speculative sampling [18] reuses predictions from a bidirectional window while their conditioning remains valid, and ActionSplice [38] adapts an in-flight sampler state to a revised action. For language models, edge–cloud speculative decoding [14, 28] over wide-area networks gains little latency and mainly saves server capacity. HIVE also predicts one chunk ahead, but unlike these systems it speculates with the large model and repairs a wrong prediction with a different, smaller model on the device, which receives the action first.

3. Edge–Cloud Collaboration

This section describes the framework (Fig. 2), namely the setting and its design space (Sec. 3.1), the role of the cloud (Sec. 3.2), and the role of the device (Sec. 3.3).

3.1. Preliminaries

Chunk-wise autoregressive world models. We consider an action-conditioned video diffusion model [17, 46] that generates a video chunk by chunk. Chunk i is a latent $z_i \in \mathbb{R}^{F\times H\times W\times C}$ of F latent frames, $H \times W$ positions, and C channels, generated from the action a_i and the previous chunks $z_{<i}$, to which the model attends through a key–value (KV) cache. A VAE decoder maps it to video frames. The model is a few-step generator G_θ with parameters θ and timesteps $t_T > \cdots > t_1$ ($T = 4$ in our models). Starting from noise $z_{i,T} \sim \mathcal{N}(0, I)$, step $j = T, \ldots, 1$ predicts the clean chunk

$$\hat{z}_{i,j} = G_\theta\big(z_{i,j},\, t_j,\, a_i,\, z_{<i}\big), \tag{1}$$

which is re-noised to the next timestep, $z_{i,j-1} = (1 - \sigma_{t_{j-1}})\,\hat{z}_{i,j} + \sigma_{t_{j-1}}\epsilon$ with $\epsilon \sim \mathcal{N}(0, I)$. The estimate of the last step is the chunk, $z_i = \hat{z}_{i,1}$, and one more forward pass at $t = 0$ writes it into the KV cache.

Edge–cloud setting. The edge device, which we call the device from here on, runs a small generator G_{dev} and the cloud a large generator G_{cloud} of the same family. Both operate in the same latent space, and each keeps its own KV cache. The user issues actions online, and the device must display a new chunk every Δ seconds ($\Delta = 0.75$ s for 12 frames at 16 FPS). The goal is low cloud GPU usage with high quality and responsiveness.

Design space. A collaboration design fixes, for every chunk, (i) which model runs which denoising steps, (ii) where the chunk is decoded, and (iii) which chunks enter the history of each model. The two-model samplers of Sec. 2 choose (i) once and apply it to every chunk. We instead let it depend on the action. The large model contributes one step to every chunk, and the small model contributes its full

sampler only to the chunks that follow an action change. Sec. 4.3 evaluates other points of this design space under the same protocol.

3.2. One-Step Lookahead in the Cloud

While chunk $i-1$ is displayed, the device requests chunk i under the current action, which serves as the predicted action $\hat{a}_i$. The cloud computes only the first estimate,

$$\hat{z}_i = G_{\text{cloud}}\big(z_{i,T},\, t_T,\, \hat{a}_i,\, z_{<i}\big), \tag{2}$$

and returns it with its chunk index and action. Because the request is issued one chunk ahead, the cloud step and the network latency are hidden behind the chunk that is being displayed. A cloud chunk therefore costs one denoising pass and one cache update instead of T passes and an update. The first estimate is trained to be a plausible chunk on its own, because these models are distilled with distribution matching [17, 45], which supervises the clean estimate of every denoising step rather than only the last. The cloud never decodes. It sends latents, at 3.6 Mbit/s in FP16 and about half of that with 8-bit quantization (see Appendix E).

3.3. Decoding and Regeneration on the Device

Decoding. If $\hat{a}_i$ equals the action a_i that is in effect when chunk i starts, the device decodes $\hat{z}_i$ and writes it into its own cache. Decoding overlaps the next request. The VAE decoder, a substantial part of the per-chunk work, thus stays off the cloud.

Regeneration at action changes. If $\hat{a}_i \neq a_i$, the prediction is invalid and is discarded. The device regenerates the chunk with all T steps of G_{dev} under a_i, decodes it, and uploads the latent. The full sampler thus runs where a single cloud step is least adequate, on the machine that receives the action first and already holds the displayed video. It adds no cloud cost and does not depend on the network, so the same path covers cloud chunks that are late or missing, *e.g.*, during a network outage. As the new action usually arrives while the previous chunk is displayed, the device starts regenerating as soon as the action arrives.

Shared history. Before predicting chunk $i+1$, the cloud writes chunk i into its cache, namely its own estimate if the prediction was valid and the uploaded latent otherwise. Both models therefore condition on exactly the displayed chunks $z_{\le i}$, and no cache tensor is ever transmitted. We verify this property for both of our implementations (see Appendix G).

Decode-only devices. Device regeneration requires that the regenerated chunk is ready when it is due. The T steps of G_{dev} and the decoding must fit into one chunk interval Δ, *i.e.*, the device must run its model at least at the playback rate. A device that does not meet this condition, *e.g.*, one that only has room for the decoder, falls back to *cloud regeneration*. It asks the cloud for a new one-step estimate under a_i and decodes it, while lookahead, decoding on the device, and the shared history stay the same. This fallback is HIVE without the device model, and we evaluate it as an ablation in Secs. 4.4 and 5.

4. Experiments

4.1. Setup

Models and hardware. We use LingBot-World 2.0 causal-fast [13], with the 1.3B model on the device and the 14B model in the cloud. Both are four-step distilled and generate three latent frames (12 video frames) per chunk at about 624×352. Each model runs on one NVIDIA RTX PRO 6000 Blackwell GPU (96 GB), which is built on the same processor as the consumer RTX 5090 (32 GB). By edge device we thus mean the user's own machine with one GPU of this class, such as a workstation or gaming PC, not a phone or headset. The device role uses little of this memory, at most 12.1 GiB with the 1.3B model and 8.8 GiB for decoding alone, against 52.5 GiB in the cloud. What it needs is speed, as it must run the 1.3B model above the 16-FPS playback rate, which this GPU does at 20.4 FPS. A 24 GB RTX A5000 does not, so it only decodes and leaves regeneration to the cloud (see Appendix E). Sec. 5 repeats the evaluation with another backbone.

Benchmarks. We evaluate on two public benchmarks. From *WorldScore* [10], we take ten scenes from each of its 30 categories and roll out each of the 300 scenes for 30 seconds under an action sequence of five six-second segments. The four action changes per rollout are *not announced* to the cloud, so HIVE regenerates four of the 39 chunks on the device. From *WBench* [47], we use the 158 navigation cases with recorded actions (WBenchNavi), in which all actions are known in advance (see Appendix A). The design study, the ablation, and the analyses of Secs. 4.3–5 need several seeds per scene. They use a set of 20 scenes under the same protocol with four or ten seeds, and the costs of Table 1 are measured on this set (see Appendix A).

Baselines. We compare three groups of methods. *(i) Single models* are the device and cloud models alone, the cloud model with one step per chunk, the 4-bit quantized 14B model on the device, and Speculative Forcing [8]. *(ii) Two models on two GPUs* are Hybrid SD [44] and EC-Diff [42], which divide every denoising trajectory between a cloud and an edge model, and SDVG [16], in which a large model regenerates the rejected drafts of a small one. In all three the cloud waits for the device on every chunk, so it cannot work

Method	FPS ↑	VLM judge		VBench		Camera / motion		Energy (J / video s) ↓			Latency	Network
		Quality ↑	Drift ↓	Quality ↑	Imag. ↑	RPE ↓	Resp. ↑	Device / GPU 1	Cloud / GPU 2	Total	(s) ↓	(kB / frame) ↓
Single model												
Device only (1.3B)	20.4	6.49	64.8	76.2	61.9	0.653	86.6	349	0	349	0.583	0
Cloud only (14B)	7.3	7.09	53.5	77.0	66.4	0.483	84.0	0	1141	1141	1.632	–
Cloud only (14B), 1-step†	14.2	7.03	48.2	77.5	65.2	0.371	83.5	0	551	551	0.842	–
14B on the device, 4-bit	5.0	6.99	51.2	76.4	65.4	0.454	85.2	1649	0	1649	2.364	0
Speculative Forcing [8]	22.7	5.37	71.1	73.2	61.0	0.834	85.9	349	0	349	0.561	0
Two models on two GPUs												
Hybrid SD [44]	10.6	6.60	58.7	76.1	61.8	0.507	84.2	383	647	1030	1.102	86.2
EC-Diff [42]	11.6	6.96	51.3	76.9	64.8	0.435	85.4	329	627	956	1.028	57.7
SDVG [16] ($\tau = -1.2$)	6.8	6.86	58.1	76.7	64.3	0.564	86.2	686	742	1428	2.192	**29.2**
Edge–cloud collaboration												
HIVE w/o device model	21.3	7.03	48.2	77.5	65.2	0.371	83.5	209	413	622	0.788	32.0
HIVE (Ours)	**21.2**	**7.07**	**46.6**	**77.9**	**65.3**	**0.382**	**87.2**	**250**	**398**	**648**	**0.586**	32.0

Table 1. **Interactive rollouts with unannounced action changes.** Quality is measured on WorldScore-300 (300 scenes, one seed) and cost on the 20-scene set. Every method runs on one RTX PRO 6000 per model, and cloud-only rows also decode in the cloud. Network is the data exchanged between the device and the cloud per displayed frame, in both directions. Cloud-only rows would stream encoded video, 6 to 10 kB per frame with H.264 (see Appendix E). VBench quality aggregates six dimensions. Bold and underline mark the best and second-best value among the two-model methods and HIVE. †Actions known in advance. Details are in Appendix A.

a chunk ahead. *(iii) Edge–cloud collaboration* is HIVE. All methods use the same model pair and hardware, and each published method is instantiated on our four steps with its configuration fixed before the evaluation (see Appendix A).

Metrics. For visual quality, we report the VBench [19] quality score and its imaging-quality dimension (Imag.), the MUSIQ [22] score of the frames, a no-reference measure of image quality between 0 and 100. For action following, we estimate camera poses from the generated video and report the rotation error (RPE) to the commanded trajectory and a Response score that rewards an early onset of motion after an action change. A VLM judge additionally rates visual quality from 1 to 10 and scene drift from 0 to 100, which captures the gradual degradation of a scene over a long rollout (see Appendix B). For efficiency, we report the throughput in decoded frames per second, the energy of each GPU per second of generated video, the latency from an action to its first decoded frame, and the data exchanged between the device and the cloud per displayed frame. The video plays at 16 FPS, so a method runs in real time only if its throughput reaches this rate. All costs are per video stream, with one cloud GPU per device. A difference is significant when its paired 95% bootstrap interval excludes zero, and Appendix B adds a Holm correction.

4.2. Main Results

Comparison with single models. HIVE uses both models unchanged, so the 14B cloud model is its quality reference (Table 1). HIVE is statistically equivalent to this reference in judged quality, at 7.07 against 7.09 (margin 0.17, see Appendix B), while it uses 2.9× less cloud energy and raises the throughput from 7.3 to 21.2 FPS, above the playback rate. The factors agree because the cloud GPU draws about 525 W in both systems. Against the one-step cloud model the cloud saving is 1.4×, and the total energy of HIVE is higher, 648 against 551 J. Its drift and rotation error are significantly lower than those of the device model and of the four-step cloud model, and its latency is that of the device model, 0.586 against 0.583 s, whereas cloud-only generation needs 1.63 s. HIVE exchanges 32 kB per frame, 4.1 Mbit/s, three to five times an H.264 stream of the same frames and less than EC-Diff and Hybrid SD (see Appendix E). Compared with the 1.3B device model, it raises the judged quality from 6.49 to 7.07 at a similar throughput. The price is a significantly lower imaging quality than that of the cloud model, 65.3 against 66.4 MUSIQ. Speculative Forcing, which accelerates the device model alone, falls below the device model at its published thresholds (see Appendix F).

Comparison with two-model methods. The two-model methods give the device a part of every chunk. Hybrid SD lets the device run the last two steps of every chunk and loses quality, at 6.60. EC-Diff leaves only the last step to the device and comes closest to HIVE, at 6.96 against 7.07, a difference that is significant only before the Holm correction. With two cloud steps per chunk it runs at 11.6 FPS with 1.6× the cloud energy. SDVG lets the cloud regenerate about 40% of the device drafts on WorldScore-300 and about half on the 20-scene set, which gives a quality of 6.86 at 6.8 FPS and more than twice the total energy. HIVE is the only one of these methods above the 16-FPS playback rate, and it has the best value among them in every column

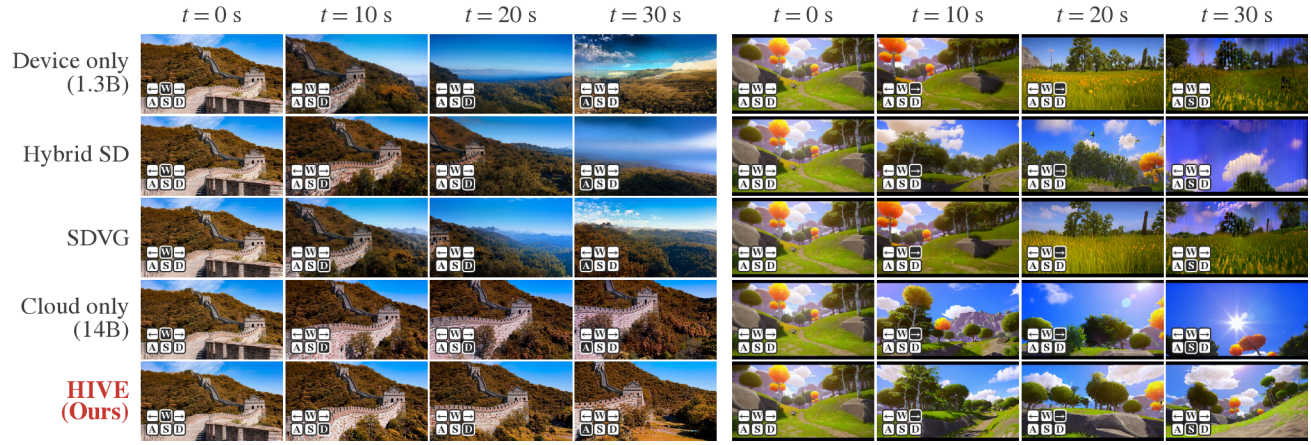


Figure 3. **Qualitative comparison on interactive rollouts.** Two 30-second rollouts with unannounced action changes. Every method receives the same input image, actions, and seed, and the keys in each frame show the active action (W and S for forward and backward, A and D for left and right, arrows for turns, no key for no input). The device model, Hybrid SD, and SDVG lose the wall (left) and the stylized trees (right) during the rollout, whereas the cloud model and HIVE keep both scenes to the end.

except the network traffic, where SDVG is lower.

Comparison with model-stitching samplers. Five model-stitching samplers from recent work, adapted to our model pair, spend more than one cloud step on every chunk. The best two, T-Stitch [31] and FlowBlending [34], match HIVE in judged quality, at 7.03 and 7.12 against 7.07, but run at 12.0 and 12.8 FPS with about 1.5× the total energy, and none of the five reaches the playback rate (see Appendix C).

Recorded actions (WBenchNavi). On WBenchNavi [47], every action is known in advance, so HIVE never regenerates and reduces to the one-step cloud stream decoded on the device. This stream stays close to the cloud model, with a judged quality of 8.29 against 8.37, a difference that is not significant, and lower drift. Only T-Stitch is significantly better, at 8.41 with two cloud steps per chunk and 12.0 FPS (see Appendix C).

4.3. Collaboration Design Choices

Figure 4 varies the design of HIVE in two directions, with quality on WorldScore-300 and cost on 20 scenes.

Role of the device. If the device upsamples a low-resolution cloud chunk, the quality falls below that of the device model, to 5.73 against 6.49, and if it refines every one-step cloud estimate with three guided steps, the quality drops from 7.07 to 6.52 at a similar cost. Periodic cloud, in which the device generates every second chunk, has the lowest cloud energy, 307 against 398 J for HIVE, but its quality falls to 6.85 and its rotation error is significantly

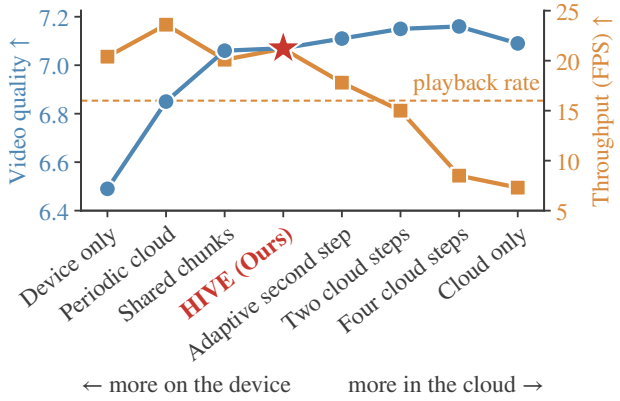


Figure 4. **Collaboration design choices.** Video quality on WorldScore-300 (blue) changes little beyond HIVE, whereas throughput (orange) falls below the 16-FPS playback rate (dashed) once the cloud runs two steps per chunk. See Appendices C and D.

higher. Letting the device generate every fourth chunk lowers the cloud energy from 398 to 370 J at preserved quality (7.06) but with a significantly higher rotation error (see Appendix C). Which chunks the device generates thus matters more than how many.

Cloud computation. More cloud steps per chunk change the quality little. A second cloud step on every chunk reaches 7.15 against 7.07, which is not significant, although it costs 1.5× the cloud energy and lowers the throughput to 15.0 FPS, below the playback rate. Four cloud steps reach 7.16, a significant difference, at 2.5× the cloud energy and 8.5 FPS and with significantly higher drift and rotation error. A second step on the half of the chunks that a learned

Variant	FPS ↑	Quality ↑	Imag. ↑	RPE ↓	Lat. ↓	Cloud J ↓
Components removed cumulatively						
– device model	**21.3**	7.23	65.0	0.521	0.79	413
– lookahead	18.9	7.23	65.0	0.521	0.63	403
– device decoding	14.2	7.23	65.0	0.521	0.84	551
– one-step sampling	7.3	**7.33**	**68.7**	0.729	1.63	1141
Chunk invalidated by an action change						
No regeneration†	21.1	7.23	65.5	0.690	–	400
Guided regeneration	20.1	7.31	66.5	0.462	0.80	401
HIVE (Ours)	21.2	7.28	67.0	**0.461**	**0.59**	**398**

Table 2. **Ablation on 20 scenes × 4 seeds.** The upper block removes components one after another, and the lower block changes how the invalidated chunk is handled. Rows 1–3 give identical videos, row 4 is the four-step cloud model. Lat. is the latency in seconds (†quality on 40 clips, the new action is never shown). The scores differ from Table 1, which uses WorldScore-300. Best in bold, second-best underlined.

predictor selects reaches 7.11 at 1.3× the cloud energy, again not significant, although it is significantly better than HIVE on the 20-scene set with ten seeds (see Appendix C). All three raise the imaging quality significantly, from 65.3 to between 65.8 and 66.5.

4.4. Ablation Study

Table 2 removes the components of HIVE one after another. Without the device model, the cloud regenerates the invalidated chunks with one step and the device only decodes. A new action then waits for the cloud, and the latency rises from 0.59 to 0.79 s (0.60 to 0.80 s on WorldScore-300, see Appendix D), so the device model is what lets HIVE respond as quickly as the on-device model. Removing lookahead as well lowers the throughput to 18.9 FPS but also the latency, to 0.63 s, because no chunk is predicted under an old action. Lookahead thus buys throughput with latency, and the device model returns the latency. Without it the player stalls under delay (Table 4). Without device decoding the throughput falls to 14.2 FPS, below the playback rate, and the cloud energy rises from 403 to 551 J. Four cloud steps per chunk then give a quality of 7.33, not significantly above HIVE, at 7.3 FPS and 2.9× the cloud energy of HIVE. If the invalid chunk is shown instead of regenerated, the camera moves the wrong way in 58.6% of the first chunks after a change, against 7.1% with HIVE, and guiding the device sampler with the invalid estimate lowers the Response score from 87.1 to 82.3 (see Appendix D). Guidance also makes the device wait for the invalid chunk, raising the latency to 0.80 s.

5. Analysis

Number of cloud steps. One 14B step is not equal to four. With every action known, the one-step cloud model scores 7.03 against 7.09 for four steps on WorldScore-300 and 7.21 against 7.40 over ten seeds on the 20-scene set, and its imaging quality is lower throughout the rollout. It nevertheless closes most of the gap from the 1.3B model to the four-step 14B model, and its first estimate is closer to its own four-step output than that of the 1.3B model (see Appendix D). The one-step videos also contain about 20% less motion, which may explain part of their lower drift and rotation error (see Appendix B), and on the other backbone the one-step cloud model falls below the device model.

Method	FPS ↑	Quality ↑	Drift ↓	Imag. ↑	Cloud J ↓
Device only (1.3B)	**14.5**	7.72	29.0	67.7	0
Cloud only (14B)	3.8	**8.13**	19.0	**69.7**	2406
Cloud only (14B), 1-step	8.0	7.60	21.5	65.2	1098
HIVE (Ours)	9.8	7.95	**15.4**	66.6	**844**

Table 3. **Generalization to another backbone.** Self-Forcing 1.3B on the device and Krea Realtime 14B in the cloud, both based on Wan2.1. Quality and Drift are VLM judge scores, Cloud J per video second. Best in bold, second-best underlined.

Device *vs*. cloud regeneration. The alternative to device regeneration is cloud regeneration, in which the cloud regenerates the invalid chunk with one step and the device needs no generative model. The device model does not raise the judged quality, which is equivalent without it at 7.03. It changes the response to a new action. The device model regenerates where the action arrives, without the cloud or the network, so the first frame of a new action is decoded after 0.39 instead of 0.59 s, the cloud spends 4% less energy, and a 2-s network outage stalls the video for 0.29 instead of 1.32 s per clip (Table 4). The cloud saving grows with the action rate, to 9% at an action change every 3 s. With a cloud that we emulate to be 1.2 to 1.3 times slower and a playback buffer of 0.3 s, HIVE stalls at most 0.06 s per clip, against 0.36 to 0.38 s without the device model (see Appendix E). The device model also changes the video. On WorldScore-300 it significantly raises the VBench quality, from 77.5 to 77.9, through subject consistency and aesthetic quality, as a second implementation confirms, and the Response score after a change to a translation, from 84.4 to 88.3. After a turn the Response score is lower, the VLM judge does not separate the two variants, and the imaging and rotation gains of the 20-scene set do not carry over (see Appendix D). Throughput and displayed latency without network delay are unchanged. The device model costs 4% more total energy (see Appendix D), and after an outage the device keeps generating until the cloud leads again, 16 of the 39 chunks of a clip on average. A late cloud chunk alone can trigger it (see Appendix E).

Method	End-to-end (s) ↓	First frame (s) ↓	Stall (s/clip) ↓	Cloud J (J) ↓
Network delay 0 ms				
No lookahead, 1-step	1.08	1.08	0.31	416
HIVE w/o device model	**0.78**	0.59	**0.00**	431
HIVE (Ours)	**0.77**	**0.39**	**0.00**	**414**
Network delay 150 ms				
No lookahead, 1-step	4.43	4.43	6.92	436
HIVE w/o device model	**1.07**	0.90	**0.00**	431
HIVE (Ours)	**1.07**	**0.49**	**0.00**	**415**
Delay 50 ms, jitter ±30 ms, loss 2%				
HIVE w/o device model	0.95	0.73	0.01	431
HIVE (Ours)	**0.87**	**0.41**	**0.00**	**415**
2-s network outage				
HIVE w/o device model	2.01	0.76	1.32	430
HIVE (Ours)	**1.00**	**0.62**	**0.29**	**405**
20-s network outage				
HIVE w/o device model	20.02	17.37	19.33	479
HIVE (Ours)	**1.01**	**0.40**	**0.29**	**339**

Table 4. **Real-time inputs.** The 20-scene set with actions issued at wall-clock times, a one-way network delay, and a 16-FPS video player. End-to-end is the median latency from an input to its display, First frame the median time until its first frame is decoded, Stall the pauses of the player per clip, and Cloud J the cloud energy per video second. In the outage rows, all messages are held for 2 or 20 s once per clip, and the device model also generates cloud chunks that are late. With 20 s the cloud does not return within the clip. Bold marks the best per condition (more in Appendix E).

Real-time inputs and network delay. The latency of Table 1 releases each action at a chunk boundary. We repeat the comparison with actions issued at wall-clock times, an injected one-way network delay, and a 16-FPS video player that pauses whenever the next chunk is not ready (Table 4). Lookahead is what keeps the system responsive. Without it, the one-step cloud model displays a new action after 1.08 s without delay and after 4.43 s at 150 ms, because every chunk waits for the network. With lookahead, HIVE and its variant without the device model display a new action equally fast, after 0.77 s and after 1.07 s at 150 ms, and neither pauses the player, which keeps the delay of its first chunks as a buffer that hides the regeneration. With a buffer of 0.3 s, the variant stalls at every action change at 150 ms, 0.68 s per clip, and HIVE 0.17 s. At 300 ms the cloud chunks arrive after their slots, and HIVE is slower than its variant, with a latency of 1.68 against 1.37 s, unless the cloud updates its cache ahead of the next request. When all messages are held for 1 to 20 s, the stall of HIVE stays between 0.29 and 0.33 s per clip, whereas that of the variant grows from 0.32 to 19.33 s (see Appendix E).

Generalization to another backbone. We apply HIVE unchanged to Self-Forcing 1.3B [17] on the device and Krea Realtime 14B [29] in the cloud, with text prompts as actions that switch four times per rollout without being announced. With HIVE this pair runs at 9.8 FPS on our GPUs, below the playback rate, so it tests the quality more than the runtime. Without the device model it runs at 8.8 FPS, because the cloud regenerates a chunk more slowly than the device. As shown in Table 3, HIVE again stays close to the four-step cloud model in judged quality at 2.9× less cloud energy and 2.6× the throughput, has the lowest drift, and significantly exceeds the one-step cloud model that knows every prompt in advance. Here one-step chunks lose imaging quality over the rollout, and each four-step chunk that the device regenerates stops this decline (see Appendix F).

6. Conclusion

We studied how a large cloud model and a small device model can generate one interactive video together. In HIVE, the cloud streams one-step lookahead chunks, and the device decodes them and regenerates the chunks invalidated by a new action with its full sampler. Under unannounced action changes, HIVE stays close to the 14B model in judged quality at 2.9× lower cloud energy. The collaboration designs that we compared show the trade-off of edge–cloud collaboration. More cloud computation per chunk costs the playback rate, and more device computation per chunk costs quality. Regenerating on the device needs neither the cloud nor the network. It saves 4% of the cloud energy, and network outages of 2 to 20 s stall the video for 0.3 s per clip instead of 1.3 to 19.3 s.

Limitations and future work. One-step chunks are softer than four-step chunks, so the imaging quality of HIVE stays below that of the cloud model. Device regeneration requires a device that keeps the small model ready at the playback rate. In our experiments this is a workstation GPU. An RTX A5000 does not and leaves regeneration to the cloud (see Appendix E). After a network outage the device keeps generating until the cloud leads again, 16 of the 39 chunks of a clip on average, and these chunks have the quality of the device model. With an action change every 3 s the quality falls significantly below that of the four-step cloud model, to the level of the one-step cloud model with known actions (see Appendix C). The latency is bounded by the 0.75-s chunk, and continuous inputs would leave most chunks to the device model. The action sequences are scripted, the visual scores come from automatic metrics and a VLM judge, and the cost is measured with both GPUs in one host, an injected network delay, and one cloud GPU per device without batching across users (see Appendix G). A human study, closed-loop play, a wide-area deployment, and models trained for their roles are left for future work.

References

[1] 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 (NeurIPS)*, 2024. 2

[2] 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. 1, 2

[3] 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 (CVPR)*, 2025. 1, 2

[4] 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, et al. Genie: Generative interactive environments. In *International Conference on Machine Learning (ICML)*, 2024. 1, 2

[5] Jiayu Chen, Hengyi Zhang, Maoliang Li, Minyu Li, Zihao Zheng, Xuanzhe Liu, Guojie Luo, and Xiang Chen. EcoVideo: Entropy-orchestrated video generation paradigm in cloud-edge dynamics. In *European Conference on Computer Vision (ECCV)*, 2026. 3

[6] Yukang Chen, Luozhou Wang, Wei Huang, Shuai Yang, Bohan Zhang, Yicheng Xiao, Ruihang Chu, Weian Mao, Qixin Hu, Shaoteng Liu, Yuyang Zhao, Huizi Mao, Ying-Cong Chen, Enze Xie, Xiaojuan Qi, and Song Han. LongLive-2.0: An NVFP4 parallel infrastructure for long video generation. *arXiv preprint arXiv:2605.18739*, 2026. 2

[7] Shenggan Cheng, Yuanxin Wei, Lansong Diao, Yong Liu, Bujiao Chen, Lianghua Huang, Yu Liu, Wenyuan Yu, Jiangsu Du, Wei Lin, and Yang You. Srdiffusion: Accelerate video diffusion inference via sketching-rendering cooperation. *arXiv preprint arXiv:2505.19151*, 2025. 3

[8] Hyunwook Choi, Seungryong Kim, Dongyoon Han, and Sangdoo Yun. Speculative forcing for efficient autoregressive video generation. *arXiv preprint arXiv:2610.22347*, 2026. 2, 4, 5

[9] Decart AI, Julian Quevedo, Quinn McIntyre, Spruce Campbell, Xinlei Chen, and Robert Wachen. Oasis: A universe in a transformer. https://oasis-model.github.io/, 2024. 2

[10] Haoyi Duan, Hong-Xing Yu, Sirui Chen, Li Fei-Fei, and Jiajun Wu. Worldscore: A unified evaluation benchmark for world generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. 1, 2, 4

[11] Jiaqi Feng, Justin Cui, Yuanhao Ban, and Cho-Jui Hsieh. One-Forcing: Towards stable one-step autoregressive video generation. *arXiv preprint arXiv:2605.23458*, 2026. 2

[12] Weilun Feng, Guoxin Fan, Haotong Qin, Mingqiang Wu, Yuqi Li, Xiangqi Li, Zhulin An, Libo Huang, Dingrui Wang, Longlong Liao, Michele Magno, Yongjun Xu, and Chuanguang Yang. WorldCache: Accelerating world models for free via heterogeneous token caching. In *International Conference on Machine Learning (ICML)*, 2026. 2

[13] 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. 1, 2, 4

[14] Yunhe Han, Yunqi Gao, Bing Hu, Mahdi Boloursaz Mashhadi, Yitong Duan, Pei Xiao, and Yanfeng Zhang. PipeSD: An efficient cloud-edge collaborative pipeline inference framework with speculative decoding. *arXiv preprint arXiv:2605.13319*, 2026. 3

[15] Xianglong He, Chunli Peng, Zexiang Liu, Boyang Wang, Yifan Zhang, Qi Cui, Fei Kang, Biao Jiang, Mengyin An, Yangyang Ren, Baixin Xu, Hao-Xiang Guo, Kaixiong Gong, Size Wu, Wei Li, Xuchen Song, Yang Liu, Yangguang Li, and Yahui Zhou. Matrix-Game 2.0: An open-source, real-time, and streaming interactive world model. *arXiv preprint arXiv:2508.13009*, 2025. 1, 2

[16] Yuezhou Hu and Jintao Zhang. Speculative decoding for autoregressive video generation. *arXiv preprint arXiv:2604.17397*, 2026. 1, 2, 3, 4, 5

[17] Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. 1, 2, 3, 4, 8

[18] Yubo Huang, Yuxuan Zhang, Jiaming Liu, Sirui Zhao, and Enhong Chen. Speculative sampling in interactive video generation. *arXiv preprint arXiv:2610.20556*, 2026. 3

[19] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. VBench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21807–21818, 2024. 5

[20] Zilan Huang, Zhiqing Tang, Hanshuai Cui, Tian Wang, Yuan Wu, Weijia Jia, and Wei Zhao. RISE: Relay inference and online scheduling for efficient edge-device collaborative diffusion model services. *arXiv preprint arXiv:2606.17378*, 2026. 2, 3

[21] Yicheng Ji, Zhizhou Zhong, Jun Zhang, Qin Yang, XiTai Jin, Ying Qin, Wenhan Luo, Shuiyang Mao, Wei Liu, and Huan Li. Forcing-KV: Hybrid KV cache compression for efficient autoregressive video diffusion models. *arXiv preprint arXiv:2605.09681*, 2026. 2

[22] Junjie Ke, Qifei Wang, Yilin Wang, Peyman Milanfar, and Feng Yang. MUSIQ: Multi-scale image quality transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 5

[23] Kyungmin Lee, David Chu, Eduardo Cuervo, Johannes Kopf, Yury Degtyarev, Sergey Grizan, Alec Wolman, and Jason Flinn. Outatime: Using speculation to enable low-latency continuous interaction for mobile cloud gaming. In

Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services, 2015. 3

[24] 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. 2

[25] Jiuming Liu, Chaojun Ni, Mengmeng Liu, Chensheng Peng, Fangjinhua Wang, et al. Towards interactive video world modeling: Frontiers, challenges, benchmarks, and future trends. *arXiv preprint arXiv:2606.01164*, 2026. 2

[26] Chengtao Lv, Yumeng Shi, Yushi Huang, Ruihao Gong, Shen Ren, and Wenya Wang. Light forcing: Accelerating autoregressive video diffusion via sparse attention. In *International Conference on Machine Learning (ICML)*, 2026. 2

[27] Jing Lyu, Shuanghao Bai, Runze Xiao, Zhenyu Liao, Wenxing Tan, Zihan Tang, Ruochuan Shi, Cheng Peng, Yuheng Ji, Yihao Wang, Badong Chen, Pengwei Wang, Zhongyuan Wang, and Xiaoguang Zhao. DIDO: Distilling interaction-centric dynamics into one-step denoising for world action models. *arXiv preprint arXiv:2609.15570*, 2026. 2

[28] Yuan Lyu, Bharath Irukulapati, and Jaya Prakash Champati. Speculation at a distance: Where edge-cloud speculative decoding actually pays off. *arXiv preprint arXiv:2606.25091*, 2026. 3

[29] Erwann Millon. Krea realtime 14b: Real-time video generation. https://github.com/krea-ai/realtime-video, 2025. 8

[30] Mustafa Munir, Huy Vu, Shreyas Misra, Rohit Jena, Sajad Norouzi, Ali Taghibakhshi, Anis Ahmad, Anjul Patney, Pavlo Molchanov, and Nima Tajbakhsh. Accelerating video diffusion via training-free trajectory routing. *arXiv preprint arXiv:2609.30096*, 2026. 3

[31] Zizheng Pan, Bohan Zhuang, De-An Huang, Weili Nie, Zhiding Yu, Chaowei Xiao, Jianfei Cai, and Anima Anandkumar. T-Stitch: Accelerating sampling in pre-trained diffusion models with trajectory stitching. In *International Conference on Learning Representations (ICLR)*, 2025. 3, 6

[32] Jack Parker-Holder and Shlomi Fruchter. Genie 3: A new frontier for world models. Google DeepMind blog, https://deepmind.google/blog/genie-3-a-new-frontier-for-world-models/, 2025. 2

[33] Rajit Rajpal, Shahbuland Matiana, Liew Wei Pyn, Anmol Agarwal, Ryan Craig, Andrew Lapp, Mithun Hunsur, Sami BuGhanem, Scottie Fox, Aaron Sanders, Carson Poole, Irene Park, Dave Rossi, Spencer Frazier, and Louis Castricato. Waypoint-1.5: A real-time video world model for consumer hardware. *arXiv preprint arXiv:2609.37107*, 2026. 2

[34] Jibin Song, Mingi Kwon, Jaeseok Jeong, and Youngjung Uh. Flowblending: Stage-aware multi-model sampling for fast and high-fidelity video generation. *arXiv preprint arXiv:2512.24724*, 2025. 3, 6

[35] Shangye Song, Dong Gong, Hong Jia, Yun Sing Koh, and Xinyu Zhang. CtrlCache: Accelerating interactive video world models with control-aware caching. *arXiv preprint arXiv:2610.08777*, 2026. 2

[36] Desen Sun, Jason Hon, Jintao Zhang, and Sihang Liu. HybridStitch: Token-adaptive model stitching for efficient diffusion generation. *arXiv preprint arXiv:2603.07815*, 2026. 3

[37] 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. 2

[38] Pardis Taghavi, Tingyu Guo, Jonas Lossner, Gaurav Pandey, and Reza Langari. ActionSplice: In-flight action editing for interactive world models. *arXiv preprint arXiv:2609.08230*, 2026. 3

[39] Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. Diffusion models are real-time game engines. In *International Conference on Learning Representations (ICLR)*, 2025. 2

[40] 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. 2

[41] Haocheng Xi, Shuo Yang, Yilong Zhao, Muyang Li, Han Cai, Xingyang Li, Yujun Lin, Zhuoyang Zhang, Jintao Zhang, Xiuyu Li, Zhiying Xu, Jun Wu, Chenfeng Xu, Ion Stoica, Song Han, and Kurt Keutzer. Quant VideoGen: Autoregressive long video generation via 2-bit KV-Cache quantization. In *International Conference on Machine Learning (ICML)*, 2026. 2

[42] Jiajian Xie, Shengyu Zhang, Zhou Zhao, Fan Wu, and Fei Wu. EC-Diff: Fast and high-quality edge-cloud collaborative inference for diffusion models. *arXiv preprint arXiv:2507.11980*, 2025. 1, 3, 4, 5

[43] Boxun Xu, Yuming Du, Zichang Liu, Siyu Yang, Ziyang Jiang, Siqi Yan, Rajasi Saha, Albert Pumarola, Wenchen Wang, and Peng Li. Sparse forcing: Native trainable sparse attention for real-time autoregressive diffusion video generation. *arXiv preprint arXiv:2604.21221*, 2026. 2

[44] Chenqian Yan, Songwei Liu, Hongjian Liu, Xurui Peng, Xiaojian Wang, Fangmin Chen, Lean Fu, and Xing Mei. Hybrid SD: Edge-cloud collaborative inference for stable diffusion models. *arXiv preprint arXiv:2408.06646*, 2024. 1, 2, 3, 4, 5

[45] Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Frédo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 4

[46] Tianwei Yin, Qiang Zhang, Richard Zhang, William T. Freeman, Frédo 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 (CVPR)*, 2025. 1, 2, 3

[47] Kaining Ying, Hengrui Hu, Siyu Ren, Jiamu Li, Fengjiao Chen, Ziwen Wang, Xuezhi Cao, Xunliang Cai, and Henghui Ding. WBench: A comprehensive multi-turn benchmark for interactive video world model evaluation. *arXiv preprint arXiv:2605.25874*, 2026. 4, 6

[48] Wangbo Yu, Kunhao Liu, Wenbo Hu, Shenghai Yuan, Chaoran Feng, et al. WorldCrafter: Consistent video world model with implicit 3D-aware memory. *arXiv preprint arXiv:2609.24984*, 2026. 2

[49] Jie Zhang, Xiaoyue Chen, Anzhe Chen, Dayiheng Liu, Deqing Li, et al. Qwen-RobotWorld technical report: Unifying embodied world modeling through language-conditioned video generation. *arXiv preprint arXiv:2606.17030*, 2026. 2