MARIONETTE: EXPLICIT WORLD STATE FOR INTERACTIVE WORLD MODEL

Anonymous authors
Paper under double-blind review

ABSTRACT

Interactive game world models typically generate video autoregressively in pixel or latent space. As a model repeatedly conditions on its own predictions, its inputs can drift away from the training distribution, causing errors to accumulate over long horizons. Correcting this drift is difficult when the dynamics remain implicit in visual latents. Our key insight is to identify the dynamics relevant to interaction and autoregress them in an explicit representation where geometric constraints can correct implausible predictions. In games with articulated characters, these dynamics take the form of 3D skeletal animation, which makes constraints on bone lengths, joint angles, and terrain contact directly expressible. We introduce MARIONETTE, a world model that decouples explicit dynamics prediction from appearance generation. An action-conditioned model predicts character poses and trajectories; a deterministic graphics bridge applies state constraints and rasterizes the resulting animation into pose-control videos. A video diffusion model then renders the motion into the final frames. Experiments show that the predicted motion responds to action inputs and that state constraints reduce ground penetration while preserving skeletal structure during long rollouts, without changing the video model.

1 INTRODUCTION

Interactive game world models generate a playable video stream directly from a stream of user actions, and recent systems reach high visual quality (Bruce et al., 2024; Valevski et al., 2025; Alonso et al., 2024; Zhang et al., 2025b; Che et al., 2025; Decart et al., 2024; Parker-Holder & Fruchter, 2025; Kanervisto et al., 2025). They do this by autoregressing appearance in pixel or latent space, which requires the model to maintain pose, geometry, occlusion, identity, and action effects within that sequence. Over a long rollout the model is repeatedly fed its own output, so errors in these implicit properties compound (Ross et al., 2011; Bengio et al., 2015), and consistency and controllability degrade as the horizon grows.

Learning $p(\text{video})$ directly entangles physics, identity, and control in pixel space. Assuming the state s suffices to condition observation synthesis, we learn separate dynamics and observation models (Ha & Schmidhuber, 2018; Hafner et al., 2025a):

$$p(\text{video}) = \int \underbrace{p(\text{obs} \mid s)}_{\text{observation model}} \underbrace{p(s_{t+1:} \mid s_{\leq t}, \text{control})}_{\text{dynamics model}} \, ds. \tag{1}$$

Neural generative models are strong at appearance and perceptual plausibility, and weak at exact bookkeeping over long horizons and at discrete logic (Dziri et al., 2023; Ma et al., 2026; Ye et al., 2026; Zhang et al., 2026). Minecraft makes the gap concrete (Guo et al., 2025). It includes redstone, a built-in system of logic circuits that behaves like digital wiring. Whether a redstone lamp turns on is determined by the logical state of that circuit rather than by the appearance of nearby pixels. A generator trained to produce the next frame can render one plausible frame while ignoring that state, which is why such logic is better delegated to a deterministic component. We use a deterministic renderer to compute metric skeleton geometry and pose controls from predicted states, following the broader use of external computation in generative systems (Yao et al., 2023; Schick et al., 2023; Gao et al., 2023). Appendix C details terrain conditioning, state constraints, and the skeletal distortion observed with stronger terrain penalties.

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

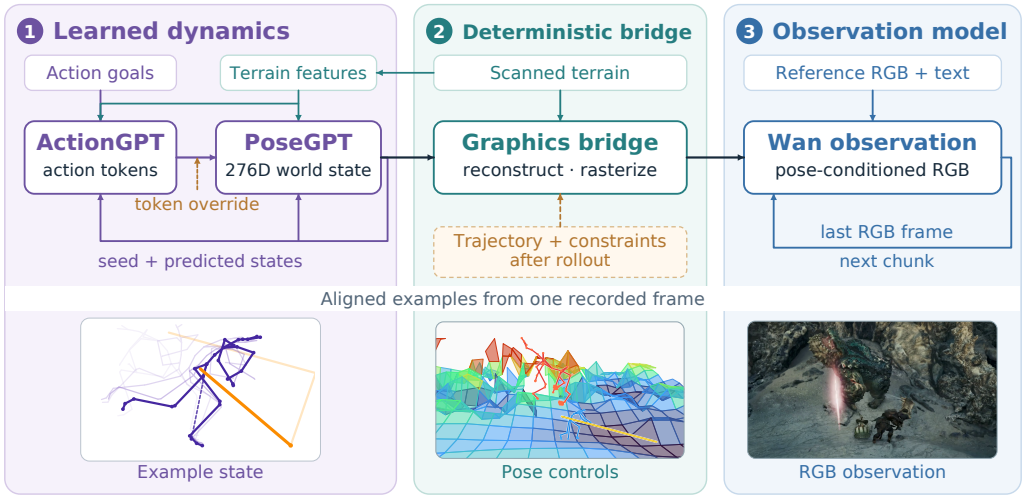

Figure 1: **MARIONETTE overview. (1)** ActionGPT selects actions from state history, action goals, and terrain; PoseGPT predicts the next 276D state. **(2)** The deterministic bridge reconstructs geometry, applies trajectory edits and constraints after rollout, and rasterizes pose controls. **(3)** The observation model synthesizes RGB from pose controls, a reference image, and text; its last frame conditions the next chunk. The lower row shows aligned state, pose-control, and RGB examples from one recorded frame. Dashed arrows mark optional interventions. The state feedback loop excludes trajectory edits and constraints.

MARIONETTE. We instantiate this idea as MARIONETTE, a world model for interactive games with articulated characters, with three components (Figure 1). First, a two-stage autoregressive *dynamics model* predicts an explicit, interpretable world state $s \in \mathbb{R}^{276}$ that encodes two articulated entities, a monster and a player character, as metric root trajectories, root-relative joint positions, and 6D root rotations. A compact *decision* model (ActionGPT) selects a discrete action per frame and entity, and a larger *animation* model (PoseGPT) turns the chosen actions into body pose. Second, a *zero-parameter deterministic bridge* reconstructs metric world-space skeletons from the state and rasterizes them into a pose-control video, computing geometry and occlusion in closed form. Third, a control-conditioned video-diffusion *observation model* renders that video into photorealistic RGB, relayed across chunks for long horizons. Because the action is an explicit token, control is applied by overwriting it, which is the same operation a player's button press performs.

We evaluate action sensitivity by overriding action tokens. With the observation model fixed, we compare recorded and predicted state inputs, then test a terrain collider and a separation cap on predicted-state rollouts.

2 RELATED WORK

Generative game world models in pixel or latent space. A large family of interactive world models autoregresses appearance directly in pixel or latent space, conditioned on user actions, to produce a playable video stream. The line runs from action-conditional prediction in Atari and GAN-based game simulators (Oh et al., 2015; Chiappa et al., 2017; Kim et al., 2020; Menapace et al., 2021; 2024), through neural game engines and interactive generators (Valevski et al., 2025; Alonso et al., 2024; Decart et al., 2024; Guo et al., 2025; Che et al., 2025; Kanervisto et al., 2025; Wu et al., 2024), to foundation-scale systems targeting open worlds, real-time streaming, and instruction following (Bruce et al., 2024; Parker-Holder & Fruchter, 2025; Zhang et al., 2025b; He et al., 2025b; Feng et al., 2025; Li et al., 2025a; Tang et al., 2025; Yu et al., 2025d; Zhu et al., 2026b). Dedicated surveys now cover this area (Yu et al., 2025b; Zhu et al., 2024b), and the action interface is itself widening, through latent actions learned without labels, language interfaces for multi-entity combat, and NPC-directed control (Garrido et al., 2026; Zhu et al., 2026a; Wang et al., 2026b). These systems learn rendering, dynamics, and control in one sequence model. They do not generally

provide explicit metric skeletons that can be inspected and constrained between prediction steps. We include Seedance 2.0 and Grok Imagine 1.5 as qualitative references under the prompt-based protocol in Appendix G.

Separating structure from appearance. One line grounds video generation in explicit 3D or in memory, caching generated content, retrieving past context, or conditioning on reconstructed geometry (Xiao et al., 2025; Wu et al., 2025; Ren et al., 2025; Yu et al., 2025a; Li et al., 2025b; Zhao et al., 2026b; Po et al., 2025; Zhang et al., 2025a). A related thread specifies camera trajectories explicitly (Wang et al., 2024; He et al., 2025a; Xu et al., 2024; Zheng et al., 2024a), and recent benchmarks probe how well a generated world preserves off-screen state and responds to actions (Ma et al., 2026; Ye et al., 2026; Zhang et al., 2026). There the 3D structure is a cache or consistency prior derived from pixels that have already been generated, while the forward dynamics stay in the video latent. A second line predicts structure before rendering it, in driving (Rahimi et al., 2026; Mo et al., 2025), in human video (Wang et al., 2025), and over a persistent semantic-voxel state paired with a learned shader (Garcin et al., 2026). Several position papers argue that a generative world model should expose a structured, rule-governed interface and delegate exact simulation to a deterministic component (Chen & Zhu, 2026; Yu et al., 2025c; Zhao et al., 2026a). A concurrent survey organizes the field around the action-state-observation loop of conventional game engines (Li et al., 2026b), and mechanistic analyses read video generators through the same state-and-dynamics lens (Wang et al., 2026a).

We place the *forward* model in an explicit metric articulated state, and turn that state into appearance with a graphics bridge that has no learnable parameters. Pose-guided character synthesis (Hu, 2024; Zhu et al., 2024a; Zhang et al., 2025c) and the enhancement of engine buffers to photorealistic video (Richter et al., 2023) establish that such a control signal is enough to drive high-quality video, which our observation model relies on. There the driving signal is an *input*, whereas a world model has to predict it and roll it out.

Controllable articulated motion. Character animation now generates articulated motion directly, under text and kinematic constraints. Diffusion and discrete-token sequence models turn a text description into a motion clip (Tevet et al., 2023; Zhang et al., 2024; 2023). The control interface has since narrowed from a whole-clip prompt to a chosen joint at a chosen frame (Xie et al., 2024). A parallel line drives characters through a physics simulator, tracking reference clips and reusing learned skill embeddings (Peng et al., 2018; 2021; 2022; Luo et al., 2023). Ground and scene conditioning has a long history here, from locomotion controllers that take the local terrain profile as input (Holden et al., 2017) to models that place the body against scene geometry (Starke et al., 2019; Hassan et al., 2021). ARDY (Zhao et al., 2026c) streams motion autoregressively with a diffusion denoiser over a hybrid representation that pairs an explicit root with a latent body embedding, and accepts long-horizon constraints such as waypoints, paths, and keyframe poses. Kimodo (Rempe et al., 2026) scales a kinematic motion diffusion model over several hundred hours of professional capture, driving multiple skeleton conventions from text plus end-effector, waypoint, and keyframe constraints. Our animation stage likewise maps control signals to articulated motion.

Adapting ARDY or Kimodo to this stage would require matching their skeleton representations and control inputs to our state interface. ARDY splits an explicit metric root from a learned body representation just as our state does, because the trajectory must be controlled in world units while the pose need not be. Our setting adds four things. Two entities fight each other. The model predicts the next action itself, and a control input overwrites that prediction at a single token. Both stages are conditioned on the scanned height field of the shipped game map. Finally, a deterministic bridge carries the state past the skeleton to rendered video.

World-model scope. The term spans distinct goals. One long-standing use is control-oriented, where latent world models learn an abstract state and predict short-horizon futures in it so that an agent can plan (Ha & Schmidhuber, 2018; Hafner et al., 2019; 2025a;b; Bar et al., 2025; Zhou et al., 2025), and for that purpose the state serves the policy and need not be exact or readable by a person. Occupancy world models predict an explicit 3D scene state, chiefly for driving perception (Zheng et al., 2024b; Lu et al., 2024), and general-purpose video foundation models are increasingly framed as world simulators for physical domains (Brooks et al., 2024; NVIDIA et al., 2026; Kang et al., 2025). We keep the classical dynamics-and-observation decomposition, but we target a different

regime. Ours is an interactive and persistent game world, where the objective is consistency and controllability. We use the same articulated state for forward prediction and for the pose controls consumed by the video model.

3 METHOD

3.1 PROBLEM FORMULATION

We represent the world by an explicit, interpretable state $s_t \in \mathbb{R}^{276}$ that describes the physical configuration of the scene at frame t. For each of the two articulated entities, a monster M and a player character (the hunter) N, it holds a per-frame root displacement, joint positions relative to that root, and the root orientation as a continuous 6D rotation, together with a small weapon sub-state:

$$s = [\underbrace{\delta^M_{0:3}}_{\text{root}}, \underbrace{p^M_{3:162}}_{53\times3\text{ joints}}, \underbrace{\delta^N_{162:165}}_{\text{root}}, \underbrace{p^N_{165:258}}_{31\times3\text{ joints}}, \underbrace{w_{258:264}}_{\text{weapon}}, \underbrace{r^M_{264:270}}_{\text{6D rot}}, \underbrace{r^N_{270:276}}_{\text{6D rot}}].$$

Root-relative joints and local root displacements separate articulation from global translation; the root rotations retain each entity's world orientation. It also makes the state *render-ready*, since a fixed operator can reconstruct metric world-space joints from s (§3.3). The dynamics control c_t comprises per-entity action goals and optional action-token overrides. Action ids identify recorded animations such as attacking, dodging, or walking. Root-trajectory edits are a separate operation on generated states before rendering (Appendix D).

Three components then realize Eq. 1. A *dynamics model* $p(s_{t+1:} \mid s_{\le t}, c)$ predicts future states (§3.2), a deterministic render operator $R(s)$ produces a pose-control video (§3.3), and an *observation model* $p(\text{obs} \mid R(s))$ synthesizes RGB (§3.4). Metric geometry, root integration, joint kinematics, and camera projection are all computed by the zero-parameter operator R, and the neural models are responsible only for plausible dynamics and appearance.

3.2 TWO-STAGE DYNAMICS: DECISION THEN ANIMATION

The dynamics model factors the next-state prediction into a *discrete decision* stage and a *continuous animation* stage. The decision of what to do is discrete and low-dimensional; the body motion that carries it out is continuous and high-dimensional.

ActionGPT. A causal transformer (4 layers, width 256, 3.62M parameters in the released checkpoint) predicts per-entity action logits and animation progress. Its inputs are recent 18D root/rotation states, action embeddings, the weapon sub-state, terrain features, and upcoming-action goals. It has no root-output head. Control replaces a predicted action token before PoseGPT uses it; §4.3 shows that overriding the action token changes the predicted pose.

PoseGPT. A larger causal transformer (8 layers, width 512, 25.69M parameters in the released checkpoint) consumes recent body and root states, terrain features, and the chosen actions. It jointly predicts 258D body/weapon coordinates and 18D root displacement/rotation, assembled into the next 276D state.

Autoregressive rollout. At each step ActionGPT proposes actions, optional overrides replace them, and PoseGPT predicts the full next state, which is appended to the context. The released generator obtains upcoming-action goals from the recorded action sequence, including its future portion; this is goal-conditioned generation. Appendix D specifies this conditioning and the 512-frame attention window. Both stages receive an MLP-encoded egocentric height patch concatenated with per-joint ground clearances, and training uses a weak non-penetration penalty and a root anti-drift term to reduce penetration and vertical drift.

3.3 DETERMINISTIC GRAPHICS BRIDGE

The bridge R turns a state sequence into a pose-control video with *zero learnable parameters*. For each frame it converts the 6D root rotations (Zhou et al., 2019) to rotation matrices by Gram–

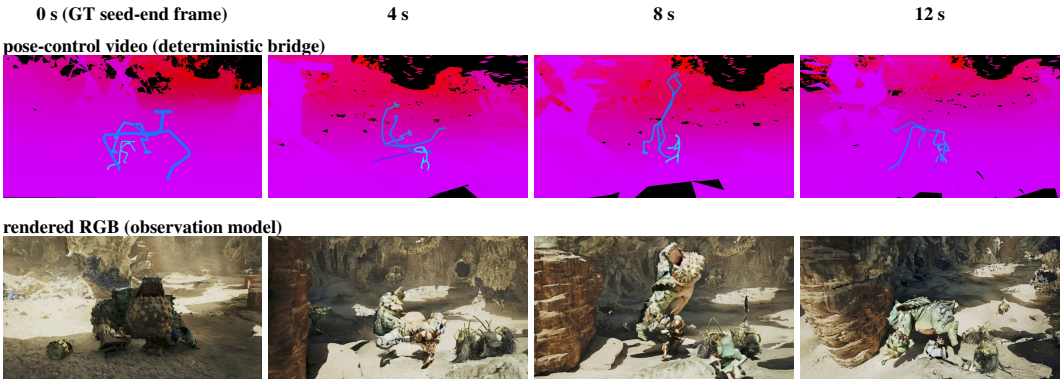

Figure 2: The deterministic bridge on a 12-second generated rollout. Top: pose-control frames rasterized from the predicted state (terrain depth encoding with both entities' skeletons). Bottom: the RGB frames the observation model renders from them.

Schmidt orthonormalization, then rotates each per-frame root delta into the world frame and integrates it by cumulative sum to recover the absolute root trajectory. It places every relative joint by adding the root-frame-rotated offsets to the root position, which yields metric world-space skeletons for both entities. It then views the scene with a camera, by default a deterministic follow-camera that frames the two entities, or a supplied ground-truth view for evaluation, and rasterizes the skeletons into the pose-control frame the observation model consumes. Given a state sequence and camera, the bridge deterministically computes metric skeleton positions and their depth ordering in the pose controls.

The bridge also carries the scanned terrain, the one part of the world that neither recorded view contains and that both the renderer and the dynamics model consume. At inference, a fixed terrain collider adjusts the generated trajectories before rendering, using the scanned height field and rate-limited vertical corrections. The pose-control frame is built as a three-channel geometry buffer, packing height, per-joint identity, and inverse depth into channels the observation model can read back. Appendix C gives the encoding, the terrain representation, and the effect of both on the training objective and on inference. Figure 2 shows the bridge on a real rollout: the rasterized pose-control frames (top) and the RGB frames the observation model renders from them (bottom), frame-aligned over twelve seconds.

3.4 Observation model and long-horizon rollout

The observation model is a control-conditioned video-diffusion model (Wan2.2-Fun-5B) with a DiT backbone (Peebles & Xie, 2023; Wan et al., 2025). It conditions RGB generation on a text prompt, the pose-control video from R, and a reference image for identity and appearance. The observation model can still deviate from the pose controls or change character identity across chunks (§5). This conditioning follows pose-guided video synthesis (Hu, 2024; Zhu et al., 2024a; Zhang et al., 2025c). For horizons far beyond a single diffusion window we use a chunk-relay rollout, in the spirit of autoregressive long-video generation (Yin et al., 2025; Liu et al., 2025): the model generates a fixed-length chunk and uses its final RGB frame, together with the next pose-control chunk, to condition the next one.

4 Experiments

Our experiments answer two questions. **(Q1)** Does a control input have a measurable effect on generated pose? We force an entity's action stream and measure what the body does (§4.3). **(Q2)** What decides how the world behaves as the horizon grows? We hold the observation model fixed and change only the state it is driven with (§4.4). §4.1 fixes the data, baseline, and protocol, and §4.2 defines the two evaluation layers. Additional ablations and full distributions are in the appendix.

4.1 SETUP AND EVALUATION PROTOCOL

Data and rollout settings. Our data is drawn from a commercial action game with articulated player and monster characters. Gameplay corpora with temporally aligned action or state annotations remain scarce (Zhou et al., 2026; Guo et al., 2026; Li et al., 2026a;b), and the recordings used here provide exactly such annotations. The same gameplay recordings yield two synchronized views, an RGB video stream and a per-frame 276D articulated world state, so that ground-truth state and ground-truth appearance are available for the identical footage. This correspondence is what makes the two-layer evaluation and the bridge ablation (§4.4) possible. Dynamics use 20 fps state sequences with monster, hunter, and weapon skeletons of 54, 32, and 2 points. The archived control and state-only evaluations use the earlier 1,395-segment data configuration; the released inference checkpoints use a 2,241-segment configuration. Appendix A separates their action vocabularies and data volumes. The observation model is trained on the RGB view under pose-control conditioning at 704×1280, with 81-frame chunks at 30 fps and chunk-relay for long horizons. Dataset construction, provenance, the terrain scan, the corpus-wide entity diversity we hold fixed, and the current single-monster scope of the dynamics model are detailed in Appendix A.

Evaluation protocol. The control probe, long-horizon state diagnostic, and rendered-window ablation use separate evaluation sets. We report a *state layer* and an *observation layer* (§4.2), with seed contexts, horizons, and control settings specified per experiment. The observation layer applies to any model that emits RGB; the state layer only to models that expose an explicit state.

Baseline. The quantitative baseline is an *end-to-end pixel-autoregressive* world model trained on the same RGB footage, conditioned on the same first frame, and rolled out with the same chunk relay. It receives the scenario as text where our observation model receives the rendered pose video, and it must carry dynamics, geometry, and appearance jointly in its latent. We take this to be the closest available comparison for our claim, because it shares our backbone family and training footage. What differs is *where* the world state is represented: in an explicit metric state with a deterministic renderer, or in an implicit latent. The systems share the observation backbone family and RGB training footage. Our pipeline additionally learns from state annotations and uses pose controls, while the baseline receives text; the comparison measures these complete pipelines under their stated conditioning. The comparison is drawn among systems that share the task as we pose it; a metric computed across mismatched protocols would mainly measure the protocol difference (scope in Appendix G).

Metrics and protocol. State-layer metrics are computed by reconstructing metric world-space joints from the predicted state (§3.3) and comparing to the ground-truth state. Observation-layer metrics are computed on rendered RGB. Because a long rollout of a stochastic world may legitimately diverge from any single reference, we distinguish short-horizon accuracy metrics from long-horizon distributional metrics (§4.2), and never summarize a long rollout by a single reference-based accuracy scalar (§4.4). Full metric definitions are in Appendix E, and the control-injection protocol in Appendix F.

4.2 EVALUATION

State-layer metrics. Because the state is a metric articulated skeleton, the predicted dynamics can be scored directly, in absolute units. Reference-based per-frame accuracy is insufficient at long horizons, because a plausible world may diverge from any single recorded trajectory, so the state-layer numbers we report are physical-plausibility quantities that need no reference to be meaningful: ground penetration, foot-skate, and how far the two entities have drifted apart. Each is compared with the same measurement on the recorded state of the same windows (Table 2). Where a control input is involved, RA-MPJPE against the recorded pose is the measure, compared between control conditions (Table 4).

Observation-layer metrics. In the observation space any model that emits RGB can be compared. We report FVD (Unterthiner et al., 2018) for perceptual and temporal quality, both over short clips and as a function of the chunk index over long rollouts. Against the pixel-autoregressive baseline our observation model reaches an FVD of 831 to its 975 (Table 1). The clip-bootstrap intervals overlap.

Table 1: Observation layer, against the end-to-end pixel-autoregressive baseline. Twelve 16.2-second rollouts per model against real footage; FVD_{16} is over 16-frame clips, eight per rollout (128 real / 96 generated), every pair started from the recorded first frame of its own held-out window, with the bridge camera initialized from the recorded view; the reported rendered rollouts use the terrain collider. The final-chunk row uses the last 81 frames. Bracketed ranges are clip-level bootstrap percentile intervals (Efron, 1979); resampling duplicates clips and inflates a Fréchet distance, which is why they sit above the point estimates. The bias is common to both columns, and only the overlap is read. State-layer behaviour is in Table 2 and Figure 3, where each number is read against the recorded state; controllability is in Table 4.

Metric	MARIONETTE	Pixel-AR baseline
$\text{FVD}_{16} \downarrow$	831 $[963, 1209]$	975 $[1101, 1346]$
FVD, final chunk of long rollout $\downarrow$	1013	1198

FVD is the only automated observation-layer score we report; reference-based per-frame scores can penalize plausible divergence in stochastic long rollouts, and broader perceptual evaluation remains future work (Huang et al., 2024). Whether the intended action happened is evaluated on the state layer, where it can be measured directly.

4.3 Q1: ACTION-TOKEN CONTROLLABILITY

Experimental objective. Our control interface is a discrete action token on the state stream. The claim is that writing that token changes what the body does.

Design. We roll out 48 held-out segments from a common seed context under three action streams for the hunter. Under FREE the model samples its own actions; under FORCE-GT we overwrite them with the recorded ids; under FORCE-SHUF with a temporally shuffled, and therefore mismatched, copy of those ids. Segment, seed and horizon are fixed and decoding is deterministic. Recorded upcoming-action goals are shared across conditions; only the injected token stream changes. Thus FREE means no token override under the same goal conditioning. We score RA-MPJPE against the recorded pose with each entity's root subtracted, so it measures the articulated pose, not the distance travelled. The baseline exposes no interface for these commands, so the probe has no baseline column; Appendix G shows both systems' response to one scripted schedule.

Result. A mismatched stream increases RA-MPJPE by 31%, from 0.272 to 0.357 m, with FORCE-GT $<$ FORCE-SHUF on 33 of the 48 segments. Forcing the correct stream matches the free-run level, 0.272 against 0.281.

Interpretation. The pose error changes when the action stream is replaced. The small mean error gap between FORCE-GT and FREE does not determine how often their action sequences differ. The full protocol is in Appendix F.

4.4 Q2: STATE CONSTRAINTS IN LONG-HORIZON ROLLOUTS

Experimental objective. We test whether terrain and separation constraints reduce their corresponding state-layer violations with the observation model fixed.

Design. We hold the observation model fixed and change only the state that drives it, over the same twelve held-out windows, with checkpoint, per-window seeds, and bridge identical. We evaluate four driving states, the recorded one and three generated ones, differing by which rule is applied between the dynamics model and the bridge. We score the state layer by ground penetration, footskate, and how far the two characters drift apart, and the observation layer by FVD. The recorded state provides the reference level for each state-layer number.

Character separation drift. Over the rollout the generated hunter and monster drift apart, from 4.9 m at the first frame to 21.2 m at the last. The camera frames both characters, so it pulls back as

Table 2: The effect of each rule. Twelve held-out windows, identical checkpoint, bridge commit and seeds; only the rule applied to the state differs. Recorded state is the same measurement on the recorded trajectories of those windows, and is a reference level, not a competing method. **CFR**, collision-frame ratio: frames with any of 11 key joints below terrain at a 0.15 m margin. **MMP**, mean maximum penetration depth. **Skate**, contact-gated foot slide. **Sep**, monster–hunter distance at the final frame; the separation cap bounds it at 6 m by construction.

Driving state	CFR ↓	MMP (m) ↓	Skate (m/s) ↓	Sep (m)
Recorded *(reference)*	*0.082*	*0.023*	*1.13*	*4.8*
Generated, no rule	0.337	0.444	1.14	21.2
+ collider	0.114	0.157	1.14	21.2
+ collider + separation cap	0.114	0.155	1.22	5.1

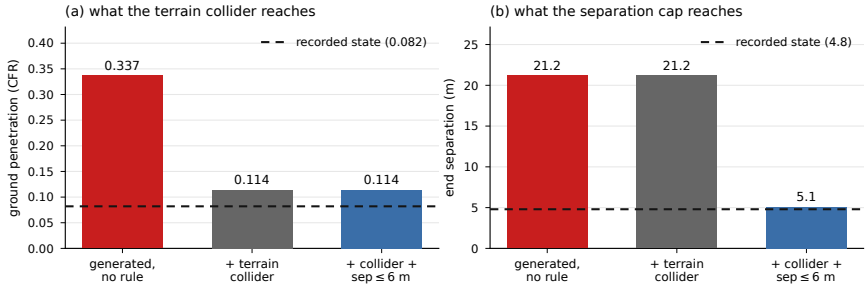

Figure 3: The terrain collider reduces penetration without changing separation; the separation cap reduces separation with similar penetration scores. Dashed line: the same measurement on recorded state over the same windows.

they separate: between the first and the last rendered chunk, each body's share of the pose-control frame falls from 1.81% to 0.57%; under recorded state that share holds at 1.55% to 1.60%. The generated root stays on scanned ground in 100% of frames, so the two characters drift apart inside the map.

Separation constraint. When a frame's motion would carry the two characters more than 6 m apart, we cancel the outward part of that motion and split the correction evenly between them. It takes the same form as the terrain wall-block (Appendix C), its threshold is a fixed constant, and it reads no recorded trajectory, so it adds no information about the future it is scored against.

Constraint effects. The terrain collider cuts the penetration collision-frame ratio from 0.337 to 0.114, a drop of 66%, and leaves separation at 21.2 m. The separation cap holds the pair at 5.1 m, leaves penetration at 0.114, and adds 7% of foot-skate, since cancelled motion makes a walking foot slide. Recorded state scores 0.082 and 4.8 m on the two measures.

Interpretation. The constraints reduce the reported penetration and separation violations with the observation model fixed, while the separation cap increases foot-skate. On the observation layer all four states land in a narrow aggregate FVD band, 747 (with the cap) to 831 (collider alone), and the final-chunk FVD is approximately 4–20% above the first chunk. Rendered differences between them are best judged by eye, and the rollouts are in the supplementary archive.

We also compare recorded and generated driving states in the same observation model. Their FVD scores are 799 and 831, respectively, with overlapping clip-bootstrap intervals; this does not establish perceptual equivalence.

On the state layer, root divergence from the recorded reference grows steadily with horizon. Such divergence alone does not imply implausible motion. The state layer reports whether a commanded action was executed and how far the state has drifted; a per-chunk appearance curve reports neither, so the protocol carries both. Both curves are given in Appendix E, Figure 5.

5 LIMITATIONS

Appearance conditioning. Appearance shifts as the horizon grows, while the geometry underneath it stays exactly what the state prescribes. The pose-control video fixes geometry, depth, and articulation, and the bridge makes those exact, but it says nothing about appearance, so appearance is conditioned on prior generated frames through the chunk-relay seed (§4.4). Reference images of each entity, persistent and re-supplied at every chunk, would give the observation model a persistent appearance reference, playing the role the explicit state plays for the dynamics; our model already accepts the weakest version of this, a single first-frame reference.

Coverage of the recorded state. The recordings used here contain entities that appear in the RGB view but have no recorded joints, notably AI companions and small monsters, so the observation model is trained on pairs in which part of the target is unexplained by the condition. Such pairs teach it to supply those entities itself, which at inference appears as content with no state behind it. This is a property of the capture and could be addressed during recording, since the engine already tracks these entities and recording their skeletons brings them into the state and into the render.

Ground-truth and generated pose. The observation model is trained on pose-control video rendered from *ground-truth* state, because that is the only pose video for which a paired real RGB frame exists, and at inference it is driven by *generated* state. The two differ in distribution, since generated pose carries the dynamics model's characteristic errors, so we rely on generalization across that shift, and the bridge ablation of §4.4 compares the two driving conditions.

One possible approach is to constrain generated states before rendering. The terrain collider and separation cap reduce specific violations (§4.4), while additional constraints on rig limits, contact, and joint ranges require evaluation. A second approach would perturb ground-truth state when rendering training conditions; its effects on robustness and alignment with the RGB targets would need evaluation.

6 CONCLUSION

We argued that an interactive game world model should fit the *world state* rather than the pixel distribution, delegating the parts that must be exact to a fixed renderer and leaving appearance to the neural model. Instantiated for articulated multi-character games, this makes geometry and occlusion exact given the state, and supports control by replacing the action stream; a shuffled stream raises RA-MPJPE by 31% relative to recorded actions. The FVD clip-bootstrap intervals overlap (831 for generated pose versus 799 for recorded pose), which does not establish perceptual equivalence. The experiments support applying targeted constraints to generated states before rendering to reduce specific rollout violations. Our own negative result illustrates a limitation of soft penalties, since a constraint expressed as a differentiable penalty on the body reduced penetration at the cost of skeleton deformation.

AI USE STATEMENT

We used AI tools to assist with literature search, drafting and revising the manuscript, and developing code for data processing and experiments. AI tools also assisted with LaTeX preparation, figure layout, and analysis of experimental outputs. The central research idea originated with the authors. The training data and evaluation references were recorded from the game; we did not use generative AI tools to synthesize these datasets. We checked the cited references against primary sources. We take responsibility for the final text, code, experimental results, and claims.

REPRODUCIBILITY STATEMENT

The recordings, frame-level action labels, and explicit per-entity state come from WILDWORLD (Li et al., 2026a); Appendix A describes the data configurations. §3 and Appendix D specify the state layout, model interfaces, conditioning, and rollout procedure. The evaluation protocols are detailed in §4 and Appendices E–G. The supplementary project page provides paired pose-control and RGB sequences, action and trajectory-control demonstrations, twelve-window constraint comparisons, and the qualitative system comparison with its text schedule.

REFERENCES

Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos J. Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in Atari. In *Advances in Neural Information Processing Systems*, pp. 58757–58791, 2024.

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*, pp. 15791–15801, 2025. arXiv:2412.03572.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, 2015. arXiv:1506.03099.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. `https://openai.com/research/video-generation-models-as-world-simulators`, 2024. OpenAI technical report.

Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *International Conference on Machine Learning*, pp. 4603–4623. PMLR, 2024.

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.

Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *Advances in Neural Information Processing Systems*, 2024.

Zhikang Chen and Tingting Zhu. From generative engines to actionable simulators: The imperative of physical grounding in world models. *arXiv preprint arXiv:2601.15533*, 2026.

Silvia Chiappa, Sébastien Racanière, Daan Wierstra, and Shakir Mohamed. Recurrent environment simulators. In *International Conference on Learning Representations*, 2017.

Decart, Julian Quevedo, Quinn McIntyre, Spruce Campbell, Xinlei Chen, and Robert Wachen. Oasis: A universe in a transformer. `https://oasis-model.github.io/`, 2024. Decart and Etched technical blog post.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Sean Welleck,

Xiang Ren, Allyson Ettinger, Zaid Harchaoui, and Yejin Choi. Faith and fate: Limits of transformers on compositionality. In *Advances in Neural Information Processing Systems*, 2023. arXiv:2305.18654.

B. Efron. Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1):1–26, 1979.

Ruili Feng, Han Zhang, Zhilei Shu, Zhantao Yang, Longxiang Tang, Zhicai Wang, Andy Zheng, Jie Xiao, Zhiheng Liu, Ruihang Chu, Yukun Huang, Yu Liu, and Hongyang Zhang. The matrix: Infinite-horizon world generation with real-time moving control. In *Advances in Neural Information Processing Systems*, pp. 87318–87344, 2025.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. PAL: Program-aided language models. In *International Conference on Machine Learning*, 2023. arXiv:2211.10435.

Samuel Garcin, Thomas Walker, Steven McDonagh, Tim Pearce, Hakan Bilen, Tianyu He, Kaixin Wang, and Jiang Bian. Beyond pixel histories: World models with persistent 3D state. *arXiv preprint arXiv:2603.03482*, 2026.

Quentin Garrido, Tushar Nagarajan, Basile Terver, Nicolas Ballas, Yann LeCun, and Michael Rabbat. Learning latent action world models in the wild. *arXiv preprint arXiv:2601.05230*, 2026.

Junliang Guo, Yang Ye, Tianyu He, Haoyu Wu, Yushu Jiang, Tim Pearce, and Jiang Bian. MineWorld: a real-time and open-source interactive world model on Minecraft. *arXiv preprint arXiv:2504.08388*, 2025.

Rongjin Guo, Dong Liang, Yuhao Liu, Fang Liu, Tianyu Huang, Gerhard P. Hancke, and Rynson W. H. Lau. EgoCS-400K: An egocentric gameplay dataset for world models. *arXiv preprint arXiv:2606.18180*, 2026.

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

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, pp. 2555–2565. PMLR, 2019.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, 2025a.

Danijar Hafner, Wilson Yan, and Timothy Lillicrap. Training agents inside of scalable world models. *arXiv preprint arXiv:2509.24527*, 2025b.

Mohamed Hassan, Duygu Ceylan, Ruben Villegas, Jun Saito, Jimei Yang, Yi Zhou, and Michael J. Black. Stochastic scene-aware motion prediction. In *IEEE/CVF International Conference on Computer Vision*, 2021.

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*, 2025a.

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*, 2025b.

Daniel Holden, Taku Komura, and Jun Saito. Phase-functioned neural networks for character control. *ACM Transactions on Graphics*, 36(4), 2017.

Li Hu. Animate anyone: Consistent and controllable image-to-video synthesis for character animation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. arXiv:2311.17117.

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. arXiv:2311.17982.

Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, 2014.

Anssi Kanervisto, Dave Bignell, Linda Yilin Wen, Martin Grayson, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Tabish Rashid, Tim Pearce, Yuhan Cao, et al. World and human action models towards gameplay ideation. *Nature*, 638(8051):656–663, 2025.

Bingyi Kang, Yang Yue, Rui Lu, Zhijie Lin, Yang Zhao, Kaixin Wang, Gao Huang, and Jiashi Feng. How far is video generation from world model: A physical law perspective. In *International Conference on Machine Learning*. PMLR, 2025. arXiv:2411.02385.

Seung Wook Kim, Yuhao Zhou, Jonah Philion, Antonio Torralba, and Sanja Fidler. Learning to simulate dynamic environments with GameGAN. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1228–1237, 2020.

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*, 2025a.

Runjia Li, Philip Torr, Andrea Vedaldi, and Tomas Jakab. VMem: Consistent interactive video scene generation with surfel-indexed view memory. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025b.

Zhen Li, Zian Meng, Shuwei Shi, Wenshuo Peng, Yuwei Wu, Bo Zheng, Chuanhao Li, and Kaipeng Zhang. WildWorld: A large-scale dataset for dynamic world modeling with actions and explicit state toward generative ARPG. *arXiv preprint arXiv:2603.23497*, 2026a.

Zhen Li, Zian Meng, Shuwei Shi, Mingliang Zhai, Jiaming Tan, Chuanhao Li, and Kaipeng Zhang. From pixels to states: Rethinking interactive world models as game engines. *arXiv preprint arXiv:2607.14076*, 2026b.

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

Jiachen Lu, Ze Huang, Zeyu Yang, Jiahui Zhang, and Li Zhang. Wovogen: World volume-aware diffusion for controllable multi-camera driving scene generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. arXiv:2312.02934.

Zhengyi Luo, Jinkun Cao, Alexander Winkler, Kris Kitani, and Weipeng Xu. Perpetual humanoid control for real-time simulated avatars. In *IEEE/CVF International Conference on Computer Vision*, 2023.

Ziqi Ma, Mengzhan Liufu, and Georgia Gkioxari. Out of sight, out of mind? evaluating state evolution in video world models. *arXiv preprint arXiv:2603.13215*, 2026.

Willi Menapace, Stéphane Lathuilière, Sergey Tulyakov, Aliaksandr Siarohin, and Elisa Ricci. Playable video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10061–10070, 2021.

Willi Menapace, Aliaksandr Siarohin, Stéphane Lathuilière, Panos Achlioptas, Vladislav Golyanik, Sergey Tulyakov, and Elisa Ricci. Promptable game models: Text-guided game simulation via masked diffusion models. *ACM Transactions on Graphics*, 43(2):1–16, 2024.

Sicheng Mo, Ziyang Leng, Leon Liu, Weizhen Wang, Honglin He, and Bolei Zhou. Dreamland: Controllable world creation with simulator and generative models. *arXiv preprint arXiv:2506.08006*, 2025.

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

Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in Atari games. In *Advances in Neural Information Processing Systems*, pp. 2863–2871, 2015.

Jack Parker-Holder and Shlomi Fruchter. Genie 3: A new frontier for world models. `https://deepmind.google/discover/blog/genie-3-a-new-frontier-for-world-models/`, 2025. Google DeepMind blog, August 5, 2025.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics*, 37(4), 2018.

Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. AMP: Adversarial motion priors for stylized physics-based character control. *ACM Transactions on Graphics*, 40 (4), 2021.

Xue Bin Peng, Yunrong Guo, Lina Halper, Sergey Levine, and Sanja Fidler. ASE: Large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Transactions on Graphics*, 41(4), 2022.

Ryan Po, Yotam Nitzan, Richard Zhang, Berlin Chen, Tri Dao, Eli Shechtman, Gordon Wetzstein, and Xun Huang. Long-context state-space video world models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8733–8744, 2025.

Ahmad Rahimi, Valentin Gerard, Eloi Zablocki, Matthieu Cord, and Alexandre Alahi. MAD: Motion appearance decoupling for efficient driving world models. *arXiv preprint arXiv:2601.09452*, 2026.

Davis Rempe, Mathis Petrovich, Ye Yuan, Haotian Zhang, Xue Bin Peng, Yifeng Jiang, et al. Kimodo: Scaling controllable human motion generation. *arXiv preprint arXiv:2603.15546*, 2026.

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*, pp. 6121–6132, 2025.

Stephan R. Richter, Hassan Abu AlHaija, and Vladlen Koltun. Enhancing photorealism enhancement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):1700–1715, 2023. arXiv:2105.04619.

Stéphane Ross, Geoffrey J. Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 627–635, 2011. arXiv:1011.0686.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, 2023. arXiv:2302.04761.

Seedance Team. Seedance 2.0: Advancing video generation for world complexity. *arXiv preprint arXiv:2604.14148*, 2026. ByteDance Seed.

Sebastian Starke, He Zhang, Taku Komura, and Jun Saito. Neural state machine for character-scene interactions. *ACM Transactions on Graphics*, 38(6), 2019.

Junshu Tang, Jiacheng Liu, Jiaqi Li, Longhuang Wu, Haoyu Yang, Penghao Zhao, Siruis Gong, Xiang Yuan, Shuai Shao, Linfeng Zhang, and Qinglin Lu. Hunyuan-GameCraft-2: Instruction-following interactive game world model. *arXiv preprint arXiv:2511.23429*, 2025.

Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H. Bermano. Human motion diffusion model. In *International Conference on Learning Representations*, 2023. arXiv:2209.14916.

Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. Diffusion models are real-time game engines. In *International Conference on Learning Representations*, 2025. arXiv:2408.14837.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Haoyu Wang, Hao Tang, Donglin Di, Zhilu Zhang, Wangmeng Zuo, Feng Gao, Siwei Ma, and Shiliang Zhang. MoSA: Motion-coherent human video generation via structure-appearance decoupling. *arXiv preprint arXiv:2508.17404*, 2025.

Luozhou Wang, Zhifei Chen, Yihua Du, Dongyu Yan, Wenhang Ge, Guibao Shen, Xinli Xu, Leyi Wu, Man Chen, Tianshuo Xu, Peiran Ren, Xin Tao, Pengfei Wan, and Ying-Cong Chen. A mechanistic view on video generation as world models: State and dynamics. *arXiv preprint arXiv:2601.17067*, 2026a.

Zeqing Wang, Danze Chen, Zhaohu Xing, Zizhao Tong, Yinhan Zhang, Xingyi Yang, and Yeying Jin. ReactiveGWM: Steering NPC in reactive game world models. *arXiv preprint arXiv:2605.15256*, 2026b.

Zhouxia Wang, Ziyang Yuan, Xintao Wang, Yaowei Li, Tianshui Chen, Menghan Xia, Ping Luo, and Ying Shan. MotionCtrl: A unified and flexible motion controller for video generation. In *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–11, 2024.

Jialong Wu, Shaofeng Yin, Ningya Feng, Xu He, Dong Li, Jianye Hao, and Mingsheng Long. iVideoGPT: Interactive VideoGPTs are scalable world models. In *Advances in Neural Information Processing Systems*, pp. 68082–68119, 2024.

Tong Wu, Shuai Yang, Ryan Po, Yinghao Xu, Ziwei Liu, Dahua Lin, and Gordon Wetzstein. Video world models with long-term spatial memory. In *Advances in Neural Information Processing Systems*, 2025.

xAI. Grok imagine. `https://x.ai/news/grok-imagine-api`, 2026. Commercial video-generation product; accessed through the xAI API.

Zeqi Xiao, Yushi Lan, Yifan Zhou, Wenqi Ouyang, Shuai Yang, Yanhong Zeng, and Xingang Pan. WorldMem: Long-term consistent world simulation with memory. In *Advances in Neural Information Processing Systems*, 2025.

Yiming Xie, Varun Jampani, Lei Zhong, Deqing Sun, and Huaizu Jiang. OmniControl: Control any joint at any time for human motion generation. In *International Conference on Learning Representations*, 2024. arXiv:2310.08580.

Dejia Xu, Weili Nie, Chao Liu, Sifei Liu, Jan Kautz, Zhangyang Wang, and Arash Vahdat. CamCo: Camera-controllable 3D-consistent image-to-video generation. *arXiv preprint arXiv:2406.02509*, 2024.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. arXiv:2210.03629.

Yixuan Ye, Xuanyu Lu, Yuxin Jiang, Yuchao Gu, Rui Zhao, Qiwei Liang, Jiachun Pan, Fengda Zhang, Weijia Wu, and Alex Jinpeng Wang. MIND: Benchmarking memory consistency and action control in world models. *arXiv preprint arXiv:2602.08025*, 2026.

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*, pp. 22963–22974, 2025.

Jiwen Yu, Jianhong Bai, Yiran Qin, Quande Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Xihui Liu. Context as memory: Scene-consistent interactive long video generation with memory retrieval. In *SIGGRAPH Asia 2025 Conference Papers*, pp. 1–11, 2025a.

Jiwen Yu, Yiran Qin, Haoxuan Che, Quande Liu, Xintao Wang, Pengfei Wan, Di Zhang, Kun Gai, Hao Chen, and Xihui Liu. A survey of interactive generative video. *arXiv preprint arXiv:2504.21853*, 2025b.

Jiwen Yu, Yiran Qin, Haoxuan Che, Quande Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Xihui Liu. Position: Interactive generative video as next-generation game engine. *arXiv preprint arXiv:2503.17359*, 2025c.

Jiwen Yu, Yiran Qin, Xintao Wang, Pengfei Wan, Di Zhang, and Xihui Liu. GameFactory: Creating new games with generative interactive videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025d.

Jianrong Zhang, Yangsong Zhang, Xiaodong Cun, Yong Zhang, Hongwei Zhao, Hongtao Lu, Xi Shen, and Ying Shan. Generating human motion from textual descriptions with discrete representations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14730–14740, 2023.

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

Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. MotionDiffuse: Text-driven human motion generation with diffusion model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(6):4115–4128, 2024.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.

Shengjun Zhang, Zhang Zhang, Simin Huang, Zhenyu Tang, Hanyang Wang, Chensheng Dai, Min Chen, Yifan Li, Yuxin Li, Yingjie Chen, et al. MBench: A comprehensive benchmark on memory capability for video world models. *arXiv preprint arXiv:2606.00793*, 2026.

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*, 2025b.

Yuang Zhang, Jiaxi Gu, Li-Wen Wang, Han Wang, Junqi Cheng, Yuefeng Zhu, and Fangyuan Zou. Mimicmotion: High-quality human motion video generation with confidence-aware pose guidance. In *International Conference on Machine Learning*, 2025c. arXiv:2406.19680.

Hongyu Zhao, Siyu Zhou, Haolin Yang, Zengyi Qin, and Tianyi Zhou. Neuro-symbolic synergy for world modeling. *arXiv preprint arXiv:2602.10480*, 2026a.

Jinjing Zhao, Fangyun Wei, Zhening Liu, Hongyang Zhang, Chang Xu, and Yan Lu. Spatia: Video generation with updatable spatial memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026b.

Kaifeng Zhao, Mathis Petrovich, Haotian Zhang, Tingwu Wang, Siyu Tang, and Davis Rempe. ARDY: Autoregressive diffusion with hybrid representation for interactive human motion generation. *ACM Transactions on Graphics (SIGGRAPH)*, 2026c. arXiv:2607.08741.

Guangcong Zheng, Teng Li, Rui Jiang, Yehao Lu, Tao Wu, and Xi Li. CamI2V: Camera-controlled image-to-video diffusion model. *arXiv preprint arXiv:2410.15957*, 2024a.

Wenzhao Zheng, Weiliang Chen, Yuanhui Huang, Borui Zhang, Yueqi Duan, and Jiwen Lu. Occworld: Learning a 3d occupancy world model for autonomous driving. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024b. arXiv:2311.16038.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. In *International Conference on Machine Learning*, pp. 79115–79135. PMLR, 2025.

Yang Zhou, Yifan Wang, Jianjun Zhou, Wenzheng Chang, Haoyu Guo, Zizun Li, Kaijing Ma, Xinyue Li, Yating Wang, Haoyi Zhu, et al. OmniWorld: A multi-domain and multi-modal dataset for 4D world modeling. In *International Conference on Learning Representations*, 2026.

Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5745–5753, 2019.

Shangwen Zhu, Qianyu Peng, Zhao Pu, Zhilei Shu, Xiangrui Ke, Zhaohu Xing, Zizhao Tong, Zeqing Wang, Xinyu Cui, Zian Zheng, Huangji Wang, Jian Zhao, Yeying Jin, Fan Cheng, and Ruili Feng. Incantation: Natural language as the action interface for multi-entity video world models. *arXiv preprint arXiv:2605.18601*, 2026a.

Shenhao Zhu, Junming Leo Chen, Zuozhuo Dai, Zilong Dong, Yinghui Xu, Xun Cao, Yao Yao, Hao Zhu, and Siyu Zhu. Champ: Controllable and consistent human image animation with 3d parametric guidance. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024a. arXiv:2403.14781.

Yixuan Zhu, Jiaqi Feng, Wenzhao Zheng, Yuan Gao, Xin Tao, Pengfei Wan, Jie Zhou, and Jiwen Lu. Astra: General interactive world model with autoregressive denoising. In *International Conference on Learning Representations*, 2026b.

Zheng Zhu, Xiaofeng Wang, Wangbo Zhao, Chen Min, Bohan Li, Nianchen Deng, Min Dou, Yuqi Wang, Botian Shi, Kai Wang, et al. Is Sora a world simulator? a comprehensive survey on general world models and beyond. *arXiv preprint arXiv:2405.03520*, 2024b.

A Dataset and Data Integrity

Data source and synchronized views. All data derives from recordings of a single commercial action game with articulated player (hunter) and monster characters. It is drawn from WildWorld (Li et al., 2026a), a previously released gameplay corpus. The three quantities a world model needs, actions, state, and observations, are produced at three different points of a running game engine, and WildWorld records each at its source. The engine consumes player *actions*, maintains and updates the *world state*, and renders the state into *observations*. The recorded action ids identify the animations played by the engine, providing frame-aligned labels for training the dynamics model. Each recording is processed into two *synchronized* views of the same footage, an RGB video stream and a per-frame 276-dimensional articulated world state extracted from the game's own pose data. Because both views come from the identical frames, we have ground-truth appearance and ground-truth state in correspondence, which is what enables the two-layer evaluation (§4.2) and the bridge ablation (§4.4). The observation model's pose-control input is produced by rendering the state view through the deterministic bridge (§3.3), so training and evaluation use exactly the same state→pose operator.

State corpus. The archived control and state-only evaluations use the earlier configuration with 1,395 motion segments and 16,389,139 frames at 20 fps. Its metadata contains 189 monster and 688 hunter action ids; reserving slot 0 for padding gives 190 and 689 model slots. The released inference configuration contains 2,241 segments and 6,554,748 frames, with 167 monster and 976 hunter action ids (168 and 977 slots including padding). Metadata and checkpoints must be paired within a configuration. Both configurations operate over articulated skeletons of 54 (monster) and 32 (hunter) joints plus 2 weapon points; each root is carried as a separate displacement channel, leaving 53 and 31 root-relative positions in the state. The state concatenation is given in Table 3. Action ids are derived from the game's animation-bank and motion identifiers, which gives a frame-accurate, zero-offset action label.

Scope: single-monster dynamics, multi-monster appearance. The dynamics models considered here use gameplay against a *single* monster type, the most-represented one. The action vocabularies above enumerate the observed animation ids in each data configuration. The multi-monster action space is strongly heterogeneous, with near-disjoint per-monster vocabularies, and we train the dynamics model on one monster's vocabulary in this study. The RGB observation model, by contrast, is trained across 27 monster identities and multiple stages and weapon types. Consequently our end-to-end interactive results are single-monster on the dynamics side. Scaling the dynamics model to the full multi-monster vocabulary is future work (§5).

Corpus coverage and study subset. The source corpus is broader than the subset we use, and we narrow it deliberately. WildWorld spans 29 monster species, four player characters, four weapon types, five stages, and party encounters of up to four hunters against one or two monsters (Li et al., 2026a). Our own RGB training manifest covers 27 of those monster species, and spans 5 weapon types, 6 hunter appearances, and 5 stages, including recordings made after the release. The monster species matter most here, because they are not skin variations of one rig. Joint counts, bone lengths, proportions, and locomotion modes differ from species to species, so a skeleton, an action vocabulary, and a learned animation prior fitted to one may require adaptation for another species. For the experiments in this paper we therefore hold most of that variation fixed and study one-on-one combat with a fixed weapon type, so that the state layout, the action vocabulary, and every metric mean the same thing across all of our ablations. Extension to other monsters, weapons, or party sizes remains untested. Another weapon or another monster is a different slice of the same corpus, re-fitted with the same two-stage recipe, and the bridge needs only that species' rig because it is driven by the state layout and has no learned weights. Widening to a co-operative party is the one case that also touches the layout, since each additional hunter adds its own root, joints, and action stream to the state vector and to both stages' inputs. This extension requires retraining both stages.

Terrain. One component of the state that the two views do not carry is the ground itself. A pose sequence says where a body is without saying what it is standing on, and neither an RGB frame nor a joint list tells the dynamics model that a slope rises ahead. We therefore record a third product alongside the two views: a scanned height field of each stage, accumulated from the geometry the

engine reports beneath the characters as they traverse it. This is our own addition, produced for this work and absent from the released corpus. The representation, how it enters both the bridge and the dynamics model, and what it does to the losses and to inference are given in Appendix C.

Dataset duration. The RGB training manifest contains $4{,}008$ unique clips with 673.8 hours of decodable footage at native frame rate. State volume is counted from segment frames before training-window sampling: the earlier configuration contains 16.4M frames (227.6 hours at 20 fps), while the released configuration contains 6.55M frames (91.0 hours). These state durations do not measure unique source-video coverage. Multiplying overlapping training windows by their length would overstate either volume.

B World-State Representation

Table 3: The 276-dimensional world state s. Monster M and hunter N are each stored as a per-frame root displacement (local), root-relative joint positions, and a 6D root rotation; a small weapon sub-state is stored relative to the hunter. All positions are metric (metres); the world frame is Y-up, right-handed.

Indices	Component	Dimension
$[0{:}3]$	Monster root displacement (local)	3
$[3{:}162]$	Monster root-relative joint positions	159
$[162{:}165]$	Hunter root displacement (local)	3
$[165{:}258]$	Hunter root-relative joint positions	93
$[258{:}264]$	Weapon points (hunter-relative)	6
$[264{:}270]$	Monster root rotation (6D)	6
$[270{:}276]$	Hunter root rotation (6D)	6
	Total	**276**

Joint positions are root-relative and root displacements are local, while the 6D root rotations retain world orientation. Absolute world positions are recovered only through the deterministic bridge (Appendix C). The 6D rotation parameterization stores the first two columns of the root rotation matrix, following the continuous rotation representation of Zhou et al. (2019).

C The Deterministic Graphics Bridge

The bridge R maps a state sequence $\{s_t\}$ to metric world-space skeletons and then to a rendered pose-control video, with no learnable parameters. For each entity and frame t:

1. **Rotation.** Convert the 6D root rotation r_t to a matrix $\mathbf{R}_t \in SO(3)$ by Gram–Schmidt orthonormalization of its two 3-vectors ($\mathbf{b}_1 = \hat{r}_t^{(1)}$, $\mathbf{b}_2 = r_t^{(2)} - \widehat{(\mathbf{b}_1 \cdot r_t^{(2)})}\mathbf{b}_1$, $\mathbf{b}_3 = \mathbf{b}_1 \times \mathbf{b}_2$).
2. **Root integration.** Rotate the local root displacement into the world frame, $\Delta_t^{\text{world}} = \mathbf{R}_t\, \delta_t$, and integrate by cumulative sum, $\mathbf{x}_t^{\text{root}} = \sum_{\tau \le t} \Delta_\tau^{\text{world}}$.
3. **Joint placement.** Place each root-relative joint $p_{t,k}$ by $\mathbf{x}_{t,k} = \mathbf{x}_t^{\text{root}} + \mathbf{R}_t\, p_{t,k}$, giving metric world-space joints for the monster and hunter skeletons (and the weapon points).
4. **Terrain.** Bind the stage's scanned height field and tessellate it into a triangle mesh in the same world frame, so that the ground the characters stand on is drawn from measured geometry.
5. **Camera.** View the scene with a deterministic follow-camera (a fixed geometric function of both entities' roots, framing the pair) by default, or with a supplied ground-truth camera for evaluation.
6. **Projection and rasterization.** Project joints to image space and rasterize the skeletons (bones and joints, with a fixed palette and depth ordering) into the pose-control frame consumed by the observation model.

Every step is a closed-form geometric operation, so world-space consistency, metric scale, and occlusion ordering hold by construction. The rendered geometry can drift only if the predicted *state* drifts, which the state-layer metrics measure directly. The same operator is used to build the pose-control training targets and to render rollouts, so there is no train/test mismatch in the bridge.

Pose-control frame format. The pose-control frame is a three-channel geometry buffer, chosen to encode height, categorical identity, and inverse depth in colour, and formatted for the video codec used in the training pipeline. At 704×1280, each fragment is written as

$$\begin{aligned} R &= \text{height:} \quad 0.5 + 0.5\,\hat{h} \ \text{ for terrain,} \qquad 0.2\,\hat{h} \ \text{ for a bone,} \\ G &= \text{a categorical identity, } \mathrm{id}/255, \\ B &= \text{inverse depth, } 1 - (z - z_{\text{near}})/(z_{\text{far}} - z_{\text{near}}), \end{aligned}$$

where $\hat{h}$ is the world Y coordinate of the fragment normalized over the stage's measured height range and z is the metric distance in front of the camera, so near geometry is bright. Three details make this work in practice. First, terrain and bone occupy *disjoint* R ranges. With a single shared range, ground and a limb at the same height receive the same colour and the two collapse into one hue. Splitting them guarantees a separation of at least 0.3 in R, and the mapping stays invertible, since the G channel says which branch to undo. Second, G resolves all the way down to individual joints, with ids drawn from reserved bands: 0 for terrain, 60–99 for weapon segments, 105–194 for the monster, 195–254 for the hunter. Each child joint holds a fixed id across the whole rollout. The observation model therefore sees a stable label for "this limb" and can use this label for correspondence across frames, and the gaps between bands absorb the residual quantization noise of the 10-bit encoding. Terrain uses $G \approx 0$, separating its identity range from those of the characters. Third, occlusion is resolved by the depth buffer. Terrain fragments closer to the camera than a few metres (3 m by default) are discarded per frame, so a wall or boulder the follow-camera passes through cannot occlude the character it is meant to show. Bones are never culled. The frame is rendered into a half-float buffer and encoded at 10 bits per channel with no chroma subsampling, since R and B are measurements and quantizing them to 8 bits discards geometric precision that the encoding is there to carry.

Terrain representation. We record terrain as a scanned height field, accumulated from the geometry the engine reports beneath the characters as they traverse a stage, and store it as a sparse set of 100 m $\times$ 100 m chunks, each a 100×100 grid of 1 m cells. Each cell holds a short list of *layers* in place of a single height, each layer carrying a height, a surface normal, a hit count, and a validity flag. A single height per cell cannot express a bridge, a ledge, or the floor of a building with terrain above it, and a model querying such a grid under an overhang gets the roof instead of the floor. A query therefore takes a reference height as well as a position and returns the valid layer nearest it, which is the surface the body is actually standing on. Two more operations make the field smooth enough to condition on and to clamp against. Layers within 0.5 m of each other in the same cell are merged, so scanning noise does not present as a staircase of near-duplicate surfaces. An unobserved cell is filled by dilation when at least three of its eight neighbours agree to within 2 m, so gaps the characters never walked over do not read as holes in the floor. Finally, the height query bilinearly interpolates the four surrounding cell centres. The per-cell lookup stepped by up to 20 cm at 1 m cell boundaries, and because that height clamps the character at inference, the steps appeared in the output as the body visibly snapping upward as it walked.

Terrain conditioning. Terrain features condition motion prediction on the local surface. Both stages therefore consume the same two terrain features, re-derived per frame from the body's own root and heading (Figure 4). In the configuration we report, both are built for the monster entity, whose ground interaction dominates the scene. The construction is per-entity and applies unchanged to the hunter. The first is an *egocentric height patch*: an 11×11 grid of 0.4 m cells, 4.4 m across, centred 1 m ahead of the root along the body's forward axis and rotated with its heading. It is sampled from the height field and stored *relative to the root height*. Making it root-relative and yaw-aligned is what makes it a reusable feature, since the same rising slope produces the same patch anywhere on the map and at any heading. The second is a vector of *clearances*, the signed vertical distance from each of 11 key joints of the monster skeleton to the surface beneath it, which tells the model which parts of the body are currently in contact. A two-layer MLP encodes the height patch into 32 features for ActionGPT and 64 for PoseGPT; the eleven clearances are concatenated

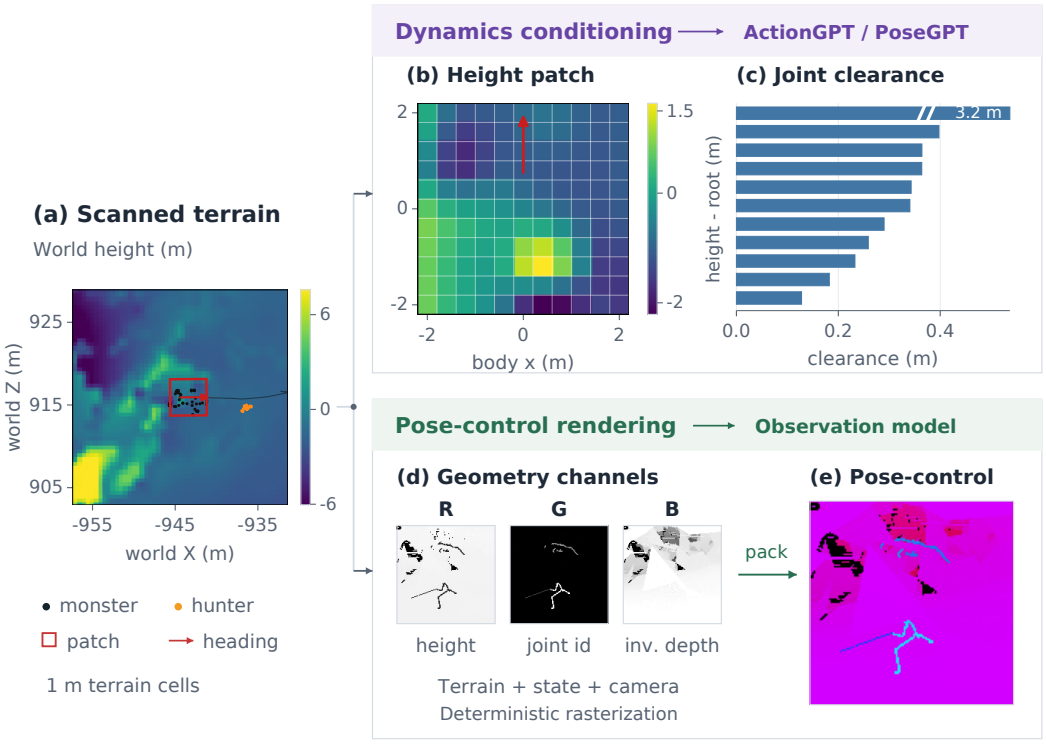

Figure 4: Terrain conditioning and pose-control rendering. All five panels show the same frame of a generated rollout. **(a)** The scanned height field, with the local patch and body heading marked. **(b,c)** Dynamics conditioning uses an 11×11 root-relative, yaw-aligned height patch (0.4 m cells, centred 1 m ahead) encoded by MLPs, together with the directly concatenated clearances of 11 joints, for ActionGPT and PoseGPT. The largest clearance is 3.2 m and is truncated in (c). **(d,e)** The deterministic bridge uses terrain, state, and camera to produce height, joint-identity, and inverse-depth channels and the composite pose-control frame for the observation model. Channel visualizations retain the original contrast windows.

directly. ActionGPT sees them because the decision of where to move next depends on what is walkable. PoseGPT sees them because the pose that realizes a step depends on where the ground is.

Terrain losses and weighting. Terrain also supplies two auxiliary losses. Neither needs the height field at training time. For every frame we precompute the terrain height under each key joint, expressed relative to the root, together with the world-up row of the body rotation, which is enough to turn a predicted root-relative joint into a predicted world height. A one-sided hinge $\mathcal{L}_{\text{pen}} = \text{ReLU}(H_j + \epsilon - y_j)^2$ penalizes a joint predicted below the surface, and $\mathcal{L}_{\text{contact}} = \mathbf{1}[\text{contact}]\,(y_j - H_j)^2$ pins joints that the ground truth marks as touching the ground, contact being defined as clearance within 0.25 m. A third term acts on the root alone and opposes the cumulative downward drift that an autoregressive root integrator otherwise develops. In the final training configuration, $\epsilon = -0.05$ m.

The final weighting was set by a failure mode we did not anticipate. With the penetration and contact terms strong (weights 2.0 and 0.5), the model did reduce penetration, and it did so by *stretching the skeleton*. Lifting a foot out of the ground is cheaper for gradient descent if the leg simply gets longer. At frame 300, the archived bone-length diagnostic reports a mean relative error of 3.2% for the terrain-free baseline and 13.7% for the strong-penalty model, using mean ground-truth seed-frame bone lengths as the reference. The constraint was satisfied only by deforming the rig. The configuration described here keeps the patch as an *input* and demotes the hinge to a small regularizer (weight 0.1), drops the contact term entirely, and retains the root-level anti-drift term (weight 1.0), which does not act on joint positions.

Terrain at inference. Inference uses the following two terrain operations. First, the patch and the clearances are re-sampled at every step from the model's *own* predicted root and rotation, no longer from a recorded trajectory, so the conditioning stays consistent with the rollout as it drifts and keeps describing the ground the character is actually over. Second, in the released implementation the generator applies a collider to reconstructed world-space trajectories after the neural rollout and frame-rate conversion, before rendering; §4.4 disables it only as an ablation. It is a simple, fixed rule with no learned parts. A frame's horizontal motion is cancelled if it would step up more than 1.5 m onto what is effectively a wall, unless the root is rising fast enough to be a jump. The collider computes a rigid vertical correction from the terrain height queried at the root and the body's lowest joint height. The target lift is capped at 0.4 m, and the correction is rate-limited to 4 cm per frame. No part of the collider is learned, and it cannot deform the body, because the same rigid shift is applied to every joint and to the weapon. These corrections change the trajectories rendered into pose controls; the released implementation does not feed the corrected trajectories back into the neural rollout.

State constraints during rollout. A model trained on recorded frames is fed its own outputs at rollout time, so the input distribution it sees is one it was never trained on. Each small error moves the next input further out of distribution, and the error compounds (Ross et al., 2011; Bengio et al., 2015). Pixel-space video generators inherit this directly, and much recent work attacks it inside the generator, through longer or packed context, drift-prevention objectives, and modified forcing schedules (Zhang et al., 2025a; Yin et al., 2025; Liu et al., 2025; Chen et al., 2024).

If the quantity fed back is an interpretable, metric state in place of an appearance latent, the drift becomes something a rule can act on, and we can inspect or constrain generated trajectories before rendering. (i) The bridge preserves the specified joint connectivity and number of limbs in the rendered pose-control sequence. (ii) The state can be checked against an external structure between steps. Terrain features are re-sampled from predicted states during neural rollout. The collider subsequently adjusts the reconstructed trajectories, without changing the earlier neural predictions. (iii) The state exposes measurable geometric quantities. Penetration depth is measured in metres and foot-skate in metres per second; bone-length stability is measured by a dimensionless coefficient of variation. These metrics distinguish different forms of state drift.

Related constraints are used in character animation, where controllers have long been conditioned on the local terrain profile and constrained by a physics engine rather than asked to learn ground contact from data alone (Holden et al., 2017; Peng et al., 2018). What the offloading view adds is that the same reasoning applies to a *world* model, and that it gives a criterion for choosing the intermediate representation.

D Dynamics Model Details

Both stages are causal transformers with sinusoidal position encodings, 8 attention heads, and a 512-frame context. The released **ActionGPT** has 4 layers, width 256, and $3{,}616{,}091$ parameters. It consumes 18D root/rotation states, 6D weapon states, action embeddings, upcoming-action goals, and terrain features, and predicts action logits and two progress scalars. Its root-output dimension is zero. The released **PoseGPT** has 8 layers, width 512, and $25{,}693{,}844$ parameters. It consumes 258D body/weapon states, 18D root/rotation states, terrain features, and actions aligned to the next frame. Its output concatenates 258 body/weapon coordinates and 18 root/rotation values; these are reordered into the state layout of Table 3. Parameter counts exclude positional-encoding and attention-mask buffers. The saved training configuration uses a non-penetration weight of 0.1, a root anti-drift weight of 1.0, and zero contact-loss weight; Appendix C describes the earlier strong-penalty setting. **Control injection.** At inference, replacing an entity's sampled action id with a target id at a given frame steers that entity. This is a single tensor assignment with no gradient or auxiliary network, and is the mechanism used by the controllability probe (Appendix F).

Action-goal conditioning. A goal is the action id at the next transition in the recorded sequence; after the last transition it is the current id. The released generator computes these goals from the seed and future action sequence, then samples next-frame actions conditioned on them. This provides future action information without supplying future poses. The saved control and state-only

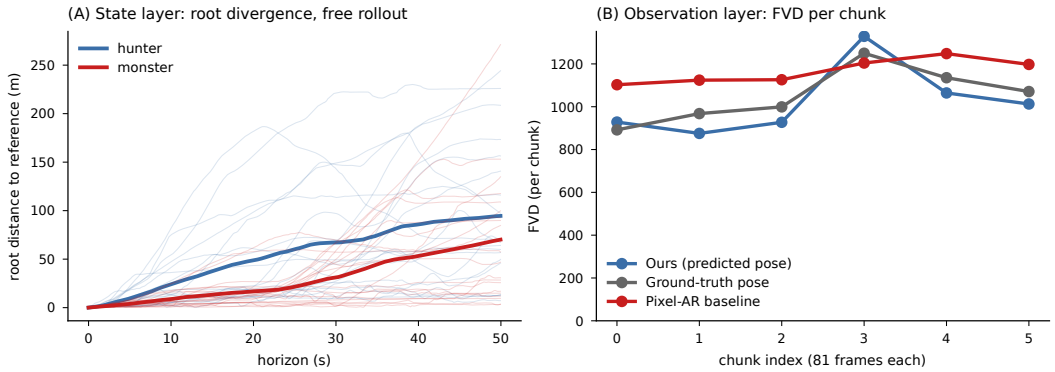

Figure 5: Long-horizon behaviour on both evaluation layers (§4.4). (A) State layer: root distance to the recorded reference for 20 held-out goal-conditioned rollouts without token overrides (faint) and their means (bold). A plausible stochastic world legitimately diverges from any single reference, so this panel is a diagnostic of divergence in absolute units and not a score. (B) Observation layer: FVD per 81-frame chunk for predicted-pose driving, recorded-pose driving, and the Pixel-AR baseline.

evaluation scripts use the same construction. In the control probe, all three conditions share these goals; FREE denotes absence of a token override, rather than unconditional action prediction.

Root-trajectory editing. The released movement demonstration first generates neural states and reconstructs world-space joints. A scripted operation then translates and rotates the hunter and weapon along a specified trajectory before pose rendering. It preserves the generated articulated motion rather than regenerating limb motion for the edited path, so sliding can occur. Action overrides act inside the neural rollout; root edits act on its output. The terrain collider runs after trajectory editing and frame-rate conversion, without feedback into the dynamics model.

E EVALUATION PROTOCOL DETAILS

State-layer metrics. *RA-MPJPE* adapts the standard mean per-joint position error (Ionescu et al., 2014) by subtracting each entity's root before comparing joints, isolating articulated-pose error from global translation. *Root drift* is the Euclidean distance between predicted and reference root position as a function of horizon; the drift curves of Figure 5 use twenty held-out goal-conditioned rollouts without token overrides of 1,000 frames at 20 fps from a 64-frame seed context, a longer state-only horizon than the twelve rendered windows. *Bone-length stability* is the coefficient of variation of each bone length over a rollout (a rig-consistency check). *Foot-skate* measures horizontal motion of a foot joint while it is in ground contact, using a rolling-window local floor estimate (radius 1 s, contact within 8 cm of the local floor) and a vertical-velocity gate ($<$ 3 cm per frame) to define contact; a global percentile floor fails for roaming characters. The recorded-state value is reported beside every generated one, so only the comparison is read. *Ground penetration* measures joints below the local surface. Table 2 uses all 512 world-joint frames at 30 fps for each of the twelve windows; the corresponding six RGB chunks contain 486 frames (16.2 s). The eleven scoring joints have zero-based indices $[53, 52, 45, 51, 44, 37, 43, 27, 36, 35, 48]$ in the monster skeleton.

Observation-layer metrics. We report FVD (Unterthiner et al., 2018) for perceptual and temporal quality. For long rollouts we report FVD per chunk against the chunk index (§4.4). Beyond the seed chunk we score distributionally rather than per frame, since a stochastic world may legitimately diverge from the one recorded reference and reference-based scores (PSNR, LPIPS (Zhang et al., 2018)) can penalize such divergence.

Controllability. We use the state-layer forced-action probe of Appendix F.

F STATE-LAYER CONTROLLABILITY: FULL PROTOCOL AND RESULTS

We test whether forcing an entity's action token moves the generated pose. For each held-out segment we seed the two-stage model with a common context and then roll out for a fixed horizon under three action streams for the hunter entity, with everything else identical. Under FREE the model predicts actions without token overrides while retaining recorded upcoming-action goals; under FORCE-GT the actions are overridden with the ground-truth ids, and under FORCE-SHUF they are overridden with a temporally shuffled and therefore mismatched GT stream. We report hunter RA-MPJPE against the ground-truth pose, and because the root is aligned away this measures whether the *action shape* follows the forced token. We decode deterministically (temperature 0) so the three conditions differ only in the injected token stream.

Table 4 reports the probe over 48 held-out segments. The probe shows two distinct effects. First, forcing a *mismatched* stream degrades the pose by 31% ($0.272 \rightarrow 0.357$ mean RA-MPJPE, with the pairwise ordering FORCE-GT $<$ FORCE-SHUF holding on $33/48$ segments), indicating that the injected action stream affects the generated pose under this probe. Second, FORCE-GT is only marginally better than FREE (0.272 against 0.281). The per-class breakdown shows larger relative error increases for stationary and in-place attack frames than for locomotion: the FORCE-SHUF/FORCE-GT mean-error ratios are $2.3\times$ for stationary frames ($n = 12$), $1.7\times$ for in-place attacks ($n = 262$), and $1.3\times$ for locomotion ($n = 5{,}478$). These frame counts apply to each forced condition and make the stationary estimate particularly sparse.

Table 4: State-layer controllability probe. Hunter root-aligned MPJPE (m) at horizon 120, seed context 64 frames, deterministic decoding; 48 held-out segments (mean $\pm$ std). Bottom: per-frame error bucketed by the class of the driving token.

	FREE	FORCE-GT	FORCE-SHUF
RA-MPJPE (m)	0.281 ± 0.141	$\mathbf{0.272 \pm 0.135}$	0.357 ± 0.120
per class (GT→SHUF)	stationary $0.24 \rightarrow 0.55$	attack $0.26 \rightarrow 0.46$	locomotion $0.27 \rightarrow 0.35$

Figure 6 shows the probe's mechanism directly on the state layer: three rollouts share an identical seed and an identical prefix, differ only in the token injected from the 5-second mark, and the generated body executes the respective action within a few frames of the switch. Figure 7 plots the per-class breakdown of Table 4.

G BASELINE SCOPE

Our quantitative reference is a pixel-autoregressive model trained on the same RGB footage and evaluated with the same first-frame and chunk-relay protocol. Its text conditioning differs from our structured state and pose controls. Many interactive video world models evaluate under protocols that differ from ours in ways that preclude a direct numerical comparison. Examples are a fixed single camera, a single controllable entity with the rest left as passive scenery, and control inputs such as device signals or per-engine animation ids that our data does not provide. Forcing such systems onto our protocol would require re-implementing and re-training them under assumptions they were not designed for, and the resulting scores could conflate model performance with changes to the task and implementation. We compare the two pipelines with their respective conditioning interfaces, and position the broader landscape qualitatively in §2.

Commercial general-purpose video generators. A second class of systems enters as a qualitative reference: the commercial general-purpose video generators. We condition the commercial systems on a text prompt and the same first frame used by our model. This differs from the structured pose and action-token conditioning in our quantitative protocol. We compare the resulting clips qualitatively. Each is driven with the same first frame and a normalized action prompt, in which the intended events are written as an explicit timestamped schedule for consistency across systems. The clips are presented side by side. Quantitative tables (§4) are reserved for models we run under our own protocol, which at present is the pixel-autoregressive baseline above. Figure 8 shows the comparison for one action schedule, with Seedance 2.0 (Seedance Team, 2026) and Grok Imagine 1.5

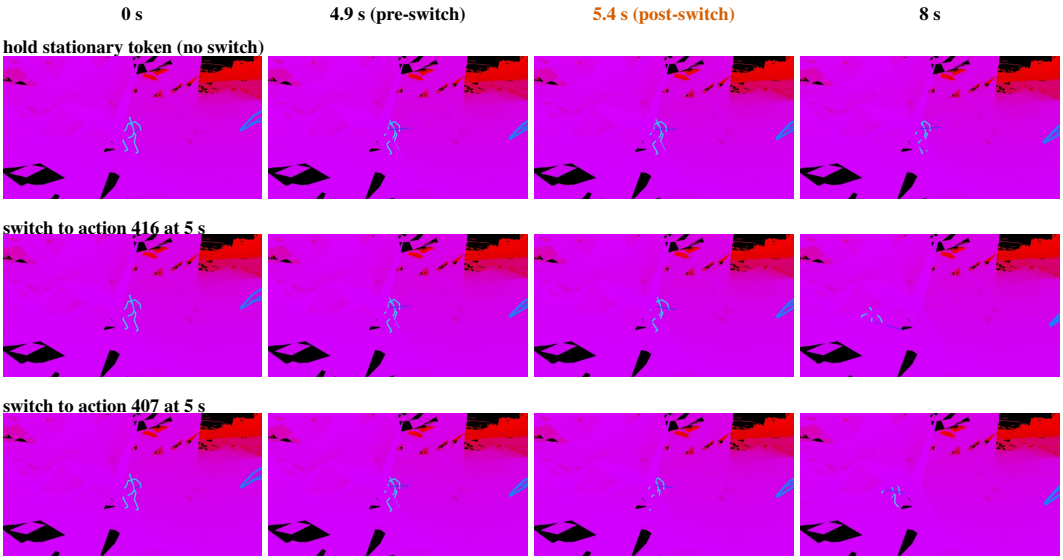

Figure 6: Token switching executes at the commanded frame (state layer, pose-control rendering, fixed camera). All three rollouts share the seed and the sampling seed, so the frames are pixel-identical until the switch. At 5 s the hunter's token is held (top) or replaced by two different action ids (middle, bottom), and the generated pose diverges accordingly within a few frames.

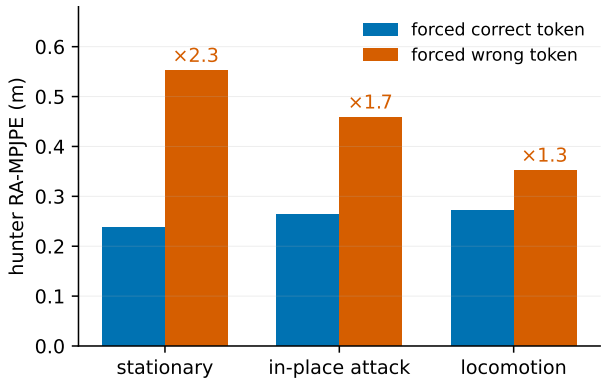

Figure 7: Forced-token pose error by action class (48 held-out segments, deterministic decoding). The relative mean-error increase under FORCE-SHUF is larger for stationary and attack frames than for locomotion. The classes contain 12, 262, and 5,478 frames, respectively, per forced condition.

(xAI, 2026) as the commercial systems. Each receives the schedule described above (the full prompt is shown on the supplementary project page). Both systems produce high-fidelity footage and respect the first frame. Seedance keeps the scene and characters faithful; Grok executes the attack roughly three seconds late and meanwhile exchanges both characters' appearance. In the example shown, our model changes its injected token at 5 s and the pose responds within a few frames. Seedance returns only 5 s of video, so its last column repeats its final frame.

H LICENSES AND USE OF ASSETS

We list every external asset this work uses, together with its verified license terms.

Wan2.2-Fun-5B-Control. We use the Wan2.2-Fun-5B-Control checkpoint as the backbone of our observation model. The weights are released by Alibaba-PAI under the Apache License 2.0, at



Figure 8: Qualitative comparison under one timestamped action schedule (stand still until 5 s, then one large melee attack), all systems starting from the same recorded first frame. Our model receives the schedule as injected action tokens. The baseline and the commercial systems receive it as text. The commercial clips show detailed appearance. Grok attacks seconds late and changes both characters' appearance; the Seedance clip ends at the 5 s command time, so it does not show the post-command response. The pixel-autoregressive baseline changes the monster's identity.

`https://huggingface.co/alibaba-pai/Wan2.2-Fun-5B-Control`. The released inference pipeline loads the umT5-xxl text encoder and Wan2.2 VAE from that repository under the same license.

VideoX-Fun. Our observation-model training and inference code extends the VideoX-Fun framework, released by aigc-apps under the Apache License 2.0, at `https://github.com/aigc-apps/VideoX-Fun`. Our dynamics model, deterministic bridge, terrain pipeline, and evaluation code will be released under the same license.

WildWorld. The gameplay recordings, the frame-level action ids, and the per-entity state we build on come from WILDWORLD (Li et al., 2026a), a previously released dataset used here under its published terms. The scanned terrain height fields (Appendix C) are not part of that release. They are produced by this work and will be distributed with our code.

Monster Hunter Wilds. *Monster Hunter Wilds* (© Capcom Co., Ltd.) is a commercial video game, and unlike the assets above it is not distributed under an open licence. All in-game footage used in this work was recorded on purchased retail copies of the game and is used solely for non-commercial academic research. Capcom publishes a *Capcom Video Policy* (last updated March 15, 2021, `https://www.capcomusa.com/video-policy/`) which permits players to capture and share footage of its titles provided the use is not monetized outside the channels it enumerates. The policy also states explicitly that it is not itself an express permission or licence, and we do not claim one. The corpus this work builds on (Li et al., 2026a) contains recorded gameplay video, together with the derived artifacts we add here: state sequences, terrain height fields, metric implementations, and pose-control renders produced by our own bridge. All of it is made available for non-commercial academic research only, under the terms published with the release, and we ask that anyone using it observe the same restriction. We will promptly comply with any request from the rights holder concerning material released here.