JOINTS, CAMERA, ACTION! CO-GENERATING 3D MOTION AND VIDEO FOR ARBITRARY RIGS

Anonymous authors
Paper under double-blind review

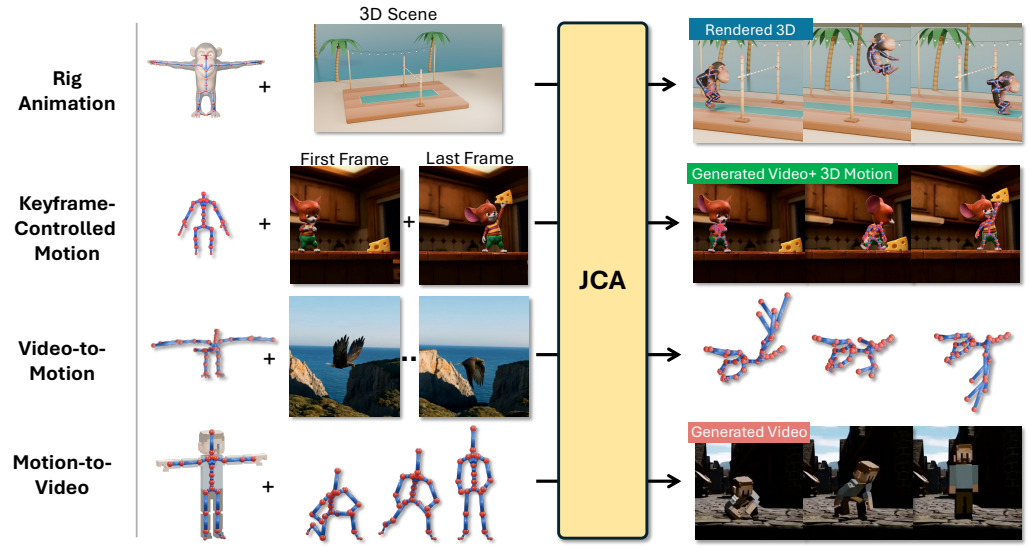


Figure 1: **Joints, Camera, Action! (JCA)** generates a video and a synchronized 3D motion trajectory for a user-supplied rig of any topology. One model covers joint generation (top two rows) and either conditional direction (bottom two rows). **Mesh animation:** a rigged monkey placed in a scene clears the high-jump bar, so its motion responds to the scene around it. **Keyframe control:** given a first and last frame, JCA fills in the video and rig motion between them. **Video-to-motion:** 3D motion for a bird rig, recovered from an input video. **Motion-to-video:** a video of the character following a prescribed joint trajectory. Overlaid skeletons are the generated 3D joints, projected using estimated camera. Text prompts omitted for clarity.

ABSTRACT

Video generation models can now depict remarkably realistic motion. Yet it is unclear how to use this motion beyond pixels: to animate a specific 3D asset and replay the animation in its 3D scene. For instance, a video model can show a monkey jumping over a crossbar, but it cannot return that jump as replayable 3D motion placed in the 3D scene. Here we present a model that generates, together with the video, 3D motion for any supplied rig and a camera that registers this motion to the scene. Prior joint video–motion models copy billions of parameters from the video model to generate motion. We instead keep 3D motion in a separate model that supports arbitrary skeletons, and connect it to a pretrained video model through a small set of cross-attention layers. This separation lets the motion model first learn how skeletons move from motion-only data, which is easier to obtain and augment than paired video. Additionally, our model learns to locate each joint in the video frames, which lets us recover the camera and place the 3D motion in the scene. To train across many skeletons, we use a generative video editor to place animated 3D assets in realistic scenes, curating the edits so their known motion remains valid. Together with human and robot video, this yields nearly 200K training pairs. Our evaluations demonstrate that our model animates diverse rigs from text more faithfully than video-first pipelines, and its motion responds to the surrounding scene. It also turns video into rig motion, rig motion into video, and enables keyframe-based control of the 3D motion. Videos and additional results are available on our supplementary webpage.

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.

1 INTRODUCTION

Video generation models capture complex real-world dynamics, making them promising world models for animation, robotics, and synthetic data generation. Yet they produce only pixels, so their motion cannot directly drive a 3D asset or robot. Imagine placing a monkey asset in front of a crossbar in a 3D scene, asking a video model to make it jump over, and replaying the resulting jump on the monkey's rig inside that scene. Current video models deliver only the first step: a convincing video of the jump, with no 3D motion for the rig and no way to place that motion in the scene.

Replaying a generated motion on an arbitrary rig in a 3D scene requires two things: 3D motion that matches the video and a camera transformation that registers that motion to the generated scene. One natural approach is to recover the motion from a generated video post hoc. Existing methods do so either with a motion-recovery model (Gong et al., 2026; Pu et al., 2026) or by animating the asset to follow the video (Song et al., 2025; Sabathier et al., 2026). However, lifting motion from pixels can introduce errors, so the recovered animation can drift from the video. Another line of work generates video and motion together, as in UniMo (Pang et al., 2025), EchoMotion (Yang et al., 2025), CoMoVi (Zhao et al., 2026a), and AnimaX (Huang et al., 2025). Joint generation avoids post-hoc recovery but leaves two challenges. First, placing the generated 3D motion in the generated scene remains unresolved. Second, no existing design is both parameter-efficient and independently pretrainable on motion alone. EchoMotion and CoMoVi create motion branches by copying large parts of the video backbone, adding billions of parameters. UniMo and AnimaX instead fold motion into the video model, with no clear route to pretraining on motion-only data.

In this work, we jointly generate a video, motion for a supplied 3D rig, and a camera that places the 3D motion in the generated scene. Rather than duplicate billions of parameters from the video backbone to represent motion, we keep the video and motion models separate. The motion model represents joints as nodes and bones as edges, making each rig's kinematic structure explicit instead of assuming a fixed joint layout. This separation also lets us pretrain on motion alone, including motions retargeted to randomized skeletons and rotated to different viewpoints, before using any paired video. We then connect the pretrained models through zero-initialized bidirectional cross-attention. The zero initialization preserves their pretrained behavior at the start of joint training, allowing the paired data to focus on teaching the two models how video and 3D motion correspond.

Connecting video and motion through attention allows them to shape each other's generation throughout denoising. To keep them aligned, we introduce a novel *bridge guidance* mechanism. At inference, we contrast predictions made with the connections enabled and disabled, then amplify the change caused by their interaction. Matching the motion to the video, however, does not determine where that motion lies in the depicted scene. To solve this registration problem, we introduce learned joint localization: the model predicts where every generated 3D joint appears in the image. These 2D locations and the generated 3D positions provide correspondences from which we fit a camera, yielding rig motion that is both matched to the video and registered to its scene.

Training the model poses a challenge: no corpus pairs natural video with 3D motion across many rigs. EchoMotion and CoMoVi use rich human video but only use fixed body models; asset animations offer diverse rigs and precise labels but simple actions and plain renders. Rather than build a 3D scene for every asset, a generative video editor (Liu et al., 2026) places our animated-asset renders in realistic scenes. Editing may move the subject or camera, invalidating 3D labels; we verify results and regenerate failures. With human and robot video, this yields nearly 200K pairs.

We evaluate the model on text-driven animation of diverse rigs and demonstrate motion that responds to the surrounding scene (fig. 1). The same model can recover motion for a supplied rig from video or generate video from prescribed rig motion. From a few reference image keyframes, it generates a complete 3D rig trajectory and matching video, providing image-based control over 3D motion.

Our contributions are: (1) the first model to directly co-generate video and 3D joint motion for arbitrary rigs, with motion that responds to the surrounding scene and can be controlled by image keyframes; (2) a parameter-efficient architecture that preserves independent pretraining of video and motion, together with bridge guidance that amplifies their agreement; (3) learned scene registration that recovers the camera needed to replay generated 3D motion in the scene; and (4) a data pipeline that turns labeled asset animations into realistic training videos and yields nearly 200K video–motion pairs spanning humans, robots, and diverse rigs.

2 Related Work

Joint Video and Motion Generation: Video generators capture rich appearance and motion but do not directly provide the underlying 3D motion. UniMo (Pang et al., 2025), EchoMotion (Yang et al., 2025), and CoMoVi (Zhao et al., 2026a) jointly model video and human motion; SViMo (Dang et al., 2025) and VAG (Lang et al., 2026) cover hand–object interactions and robot actions. EchoMotion and CoMoVi copy 5B video-backbone blocks into motion branches, adding 2.5B–5B parameters. AnimaX (Huang et al., 2025) handles arbitrary skeletons, generates isolated multi-view video and pose maps, then recovers motion through triangulation and inverse kinematics. We couple separately pretrained video and text-to-rig-motion networks to directly generate video and 3D joint parameters for a supplied rig, without mesh input or triangulation, optionally conditioned on a first image.

Animating Diverse Skeletons and Meshes: AnyTop (Gat et al., 2025) learns a shared motion prior across diverse skeletons; RigMo (Zhang et al., 2026) learns rigs and motion from deforming meshes without predefined skeletons. AnimateAnyMesh (Wu et al., 2025) maps mesh and text to vertex trajectories; none generates video. Video priors inform ViPS's feed-forward pose sampling (Chen et al., 2026) and per-instance animation optimization by AKD (Li et al., 2025), Puppeteer (Song et al., 2025), and skeleton-free Chord (Lyu et al., 2026); AKD reports about 25 hours per asset. AnimaSpark (Zhao et al., 2026b) avoids such optimization through video generation, tracking, and planar lifting. ActionMesh (Sabathier et al., 2026) outputs mesh deformations and (Bekor et al., 2025) transfers motion across object categories without a rig by distilling video diffusion priors into dynamic Gaussians. Our motion comes from coupled denoising, without tracking or fitting to video.

Conditional Generation and Guidance: Motion can also control visual generation: ControlNet (Zhang et al., 2023) conditions image synthesis on structural cues, MeshToken (Liang et al., 2026) conditions video on 3D human mesh motion, LooseControlVideo (Bhat et al., 2026) uses coarse oriented 3D boxes while resolving occlusions and dynamics, and training-free Time-to-Move (Singer et al., 2026) uses crude reference animations under a dual-clock schedule. Motion inpainting and constrained generation (Tessler et al., 2024; Rempe et al., 2026) complete partial motion specifications. EchoMotion (Yang et al., 2025) supports both conditional directions through joint attention, spatially offsetting motion tokens from the video grid. By sometimes keeping one modality clean during training, our model supports joint generation and both directions for arbitrary rigs, with text in every mode; its cross-modal attention encodes only time, leaving joint identity to rig descriptors. For guidance, VideoJAM (Chefer et al., 2025) contrasts predictions with and without the jointly generated optical-flow input; we instead contrast predictions with and without video–motion cross-attention.

Motion Recovery and Evaluation: MoCapAnything V2 (Gong et al., 2026) and TopoCap (Pu et al., 2026) recover motion from video for arbitrary skeletons without per-instance optimization, which our model also supports. PhyMotion (Huang et al., 2026) reconstructs human motion to provide physics-grounded rewards for video post-training.

Training Data Pipeline: Joint video–motion methods, like UniMo, EchoMotion and CoMoVi train on real human video with motion estimated from it and use one skeleton. VideoMDM (Mann et al., 2026) avoids 3D annotation by supervising 2D reprojections, but also uses one skeleton. Animated 3D asset datasets such as AnimaX's (Huang et al., 2025) span many skeletons with exact motion, but their renders show a lone asset on a plain background. We edit such renders into realistic scenes with a generative video editor (Liu et al., 2026) and curate the edits against generative artifacts.

3 Method

Video models have watched countless things walk, crawl, and leap, yet know nothing about their underlying rig. Motion models, in contrast, are built around a rig but lack the rich motion priors that video captures. We bring the two together: our method generates 3D motion for an arbitrary skeleton by denoising it jointly with a video, so that each model contributes what it knows.

Given a text prompt c, a rig defined as a directed tree $\mathcal{G} = (\mathcal{J}, \mathcal{E})$, and optionally a reference image I_0 with the 2D locations of the rig's joints, we produce a video, a 3D motion for $\mathcal{G}$ given as per-joint rotations and positions, and a camera that projects this motion into the scene that the video depicts.

We describe our approach in four parts, illustrated in Fig. 2. Sec. 3.1 describes a motion denoiser pretrained to generate motion for any rig. Sec. 3.2 connects it to a pretrained video generator through

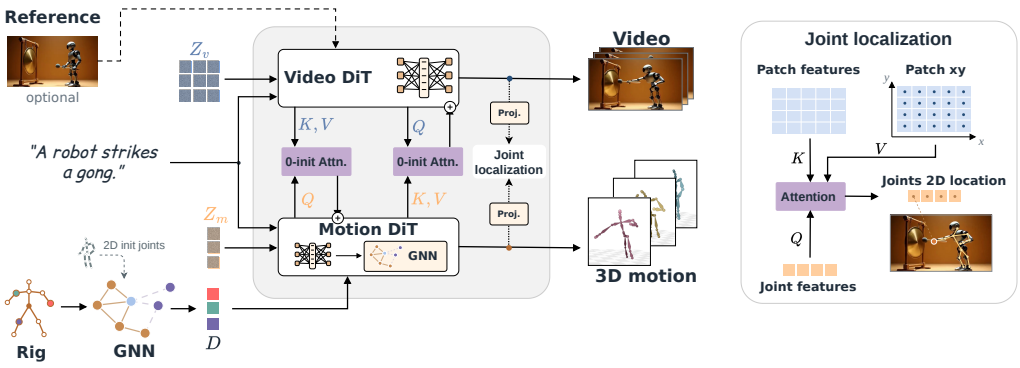


Figure 2: Architecture overview. Given a text prompt, a rig, and an optional reference image, the model jointly denoises video latents Z_v and motion tokens Z_m. A graph network encodes the rig into per-joint descriptors D, using the joints' 2D locations in the reference image, and these condition the motion branch; a graph-attention layer inside each Motion DiT block propagates features along the rig edges. Zero-initialized cross-attention bridges exchange features in both directions at paired blocks, each branch querying the other's intermediate features, so joint training begins from the two pretrained branches acting as they did in isolation. An auxiliary joint-localization head (right) attends from joint features to video patches and returns each joint's image position as a weighted combination of patch coordinates, giving 2D–3D correspondences used to fit a camera.

zero-initialized bidirectional cross-attention. This connection keeps the two models separate, saving billions of parameters compared with building the motion model into the video backbone, and allows a guidance scheme that amplifies the exchange between the two branches at inference. Sec. 3.3 introduces a localization head that grounds each joint in the video, from which we fit the camera. Finally, Sec. 3.4 describes how we build training data pairing natural video with 3D motion across many skeletons, by editing renders of animated assets into realistic scenes.

3.1 Rig-Conditioned Motion Generation

For motion generation, we use a flow-matching network (Lipman et al., 2023) that denoises a trajectory for any provided rig, conditioned on text. A graph network encodes the rig into per-joint descriptors that condition a transformer denoiser. We train both from scratch on motion data, before any coupling to video, so that joint training begins from a motion prior. We define the flow-matching path from a clean sample z_0 as $z_s = (1-s)z_0 + s\epsilon$, $\epsilon \sim \mathcal{N}(0, I)$, with noise level $s \in [0, 1]$ (s_v and s_m for the video and motion branches) and velocity target $v^\star = \epsilon - z_0$.

Motion representation. Following AnyTop, we use a redundant representation. Each joint at each frame stores its rigid transformation, parameterized with 9 elements, and the change in that transformation to the next frame. A motion with N frames and J joints is thus $Z_m \in \mathbb{R}^{N\times J\times 18}$. The root is expressed globally, and all other joints relative to the body, so they do not depend on where the character stands or faces. Details are in App. A.1.

Rig descriptor. The rig descriptor $D \in \mathbb{R}^{J\times d}$ tells the denoiser each joint's identity and skeletal position by encoding joint attributes and propagating them along the rig's connectivity. Each joint's input combines its name, embedded with the video model's frozen text encoder, its rest position, its reference-image location when available, and its root, chain, and leaf indicators. A graph attention network (Brody et al., 2022) propagates these features bidirectionally along the bones $\mathcal{E}$, followed by full self-attention over joint tokens. D is computed once per rig and reference image (App. A.2).

Denoiser. A transformer over the NJ motion tokens predicts the flow velocity, $\hat{v} = \Phi_\theta(X_{s_m}, s_m, D, \mathcal{E}, c)$. Each block is a DiT block, with s_m modulating the normalizations and text entering by cross-attention, extended with a graph-attention layer over the rig edges $\mathcal{E}$ that runs within a single frame. The motion tokens are therefore self-attended twice per block: globally, where any joint at any frame can reach any other, and along the rig, where a joint reaches only within its graph neighborhood. Joint identity comes from the rig descriptors, projected and added to each joint's tokens at every block; frames are positioned by rotary embeddings as usual. An input

MLP lifts each joint's 18 channels into a token and an output MLP maps the final token back to a velocity, with one pair of MLPs for the root and one pair shared by all non-root joints (App. A.3).

3.2 Coupling Video and Motion

How should the two models exchange information? Prior joint video–motion models build human-motion branches from the video transformer's blocks (Yang et al., 2025). Adopting that design here would tie our arbitrary-rig motion model to the 14B video backbone and add several billion parameters. We therefore keep the two models separate, and connect the motion denoiser to the pretrained Wan2.2 I2V-A14B video backbone by inserting cross-attention "bridges" in both directions between each motion block and one video block, spread approximately evenly through the backbone. At such a block, let $\bar{H}_v$ and $\bar{H}_m$ be the features entering the self-attention of the video and motion branches. We compute

$$B_{m\to v} = \mathrm{Attn}(\bar{H}_v, \bar{H}_m)\, O_v, \qquad B_{v\to m} = \mathrm{Attn}(\bar{H}_m, \bar{H}_v)\, O_m \quad ,$$

where $\mathrm{Attn}(X, Z) = \textit{softmax}(QK^\top/\sqrt{d_k})V$, and (Q, K, V) are the respective queries, keys, and values. We apply $B_{m\to v}$ to the video features and $B_{v\to m}$ to the motion features. Each branch therefore conditions on the other's intermediate features at that block.

The output projections O_v and O_m of these bridges are initialized to zero. Joint training thus starts from the two pretrained branches acting as they did in isolation, and coupling is learned from there (App. A.4). As in EchoMotion, we interpolate the temporal rotary embeddings to align the two streams, placing motion frame n at position $n/4$ on the video latents' time axis.

Bridge guidance. At inference, we retain standard text guidance while adding separate control over cross-modal coupling. Text guidance strengthens prompt adherence, while our bridge guidance amplifies the effect of video–motion interaction. To achieve this, we evaluate the same model with its bidirectional cross-attention bridges disabled and enabled, giving us separate text and bridge guidance scales. We apply text guidance to each configuration and then combine the two guided predictions using the bridge guidance scale.

Let v_i^b be the velocity that the coupled model predicts for modality $i \in \{\mathrm{video}, \mathrm{motion}\}$ with bridges being disabled ($b = 0$) or enabled ($b = 1$), everything else unchanged. Each of the two predictions, v_i^0 and v_i^1, is first guided on the prompt c against the empty prompt $\varnothing$, $\tilde{v}_i^b = v_i^b(\varnothing) + \omega_{\mathrm{text},i}\big(v_i^b(c) - v_i^b(\varnothing)\big)$, and we then scale the change from the disconnected prediction to the connected one, $\hat{v}_i = \tilde{v}_i^0 + \omega_{\mathrm{bridge},i}\big(\tilde{v}_i^1 - \tilde{v}_i^0\big)$. The scale $\omega_{\mathrm{bridge},i}$ sets how strongly modality i follows the other, and $\omega_{\mathrm{bridge},i} = 1$ recovers text guidance with the bridges enabled. Guided sampling requires four model evaluations per step. All four start from the same noisy video and motion states at the same noise levels, and share the rig and reference conditions; only the prompt and the bridge state differ.

3.3 Joint Localization and Camera Alignment

The attention bridges discussed above bind the video and motion streams at generation, but they do not provide explicit linking between motion 3D coordinates and video. To relate the two, we use the final features of the video and motion branches to estimate this. Each joint attends only to the patches of its own latent frame f: with queries Q_f computed from the joints' motion features and keys K_f from the frame's patch features, the weights $A_f = \mathrm{softmax}(\kappa\, Q_f K_f^\top)$, with a learned scale κ, average the patch centers Ξ into image locations $\widehat{Y}_f = A_f\, \Xi$, supervised by smooth-$L_1$ against 2D annotations. Pairing $\widehat{Y}_f$ with the generated 3D joint positions, we fit one static perspective camera per clip, estimating rotation, translation, and focal length while fixing the principal point at the image center. This aligns the motion's projection with the character in the generated video without modifying the generated motion (App. A.5). Figure 5 shows the same generated motion rendered in an input scene before and after alignment, highlighting its importance for correctly positioning the character.

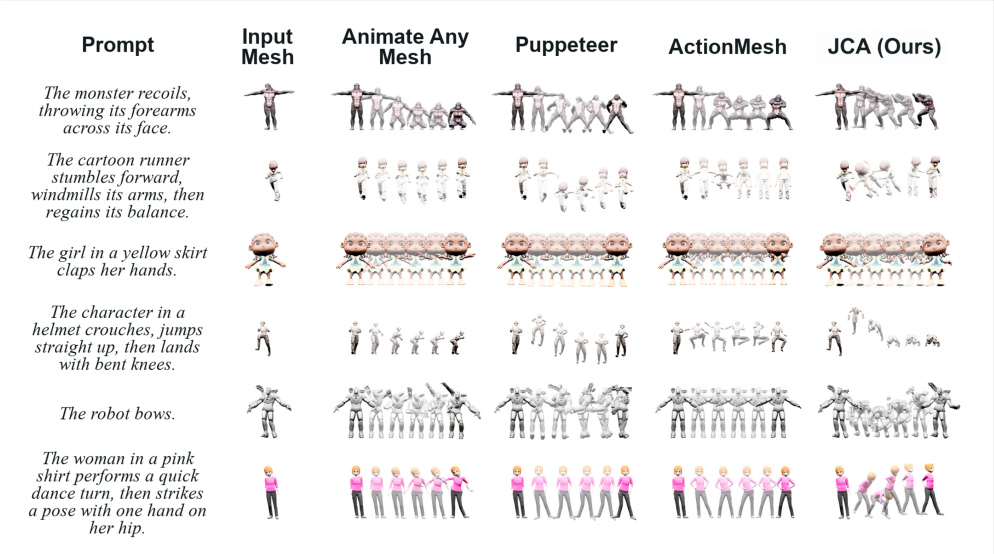


Figure 3: Text-driven rig animation across diverse assets. Each row shows a text prompt, an input mesh, and sampled poses from animations generated by competing methods and JCA (ours). Our method more faithfully captures the requested actions while preserving the asset's geometry.

3.4 Training Data

To our knowledge, there exists no corpus that pairs natural video with 3D motion for arbitrary rigs. We therefore build and annotate nearly 200K video–motion pairs from complementary sources. Human video has rich scenes and interactions, but only one skeleton. Animated Objaverse assets span a wide range of skeletons with exact motion, but their motions are simple and their renders show plain backgrounds, which would likely leave a gap to real scenes and to human video.

One could place each asset in a 3D scene and render it. But suitable scenes are scarce, hard to match to each action, and look synthetic. Instead, we render each asset alone. A vision-language model proposes an environment suited to the asset, and a generative video editing model (Bernini) places the asset in that scene while its known 3D motion serves as ground truth.

The central challenge is preserving this ground truth through editing. The edit can silently make the video disagree with the 3D motion. We therefore reject clips with camera drift, compare pre- and post-edit masks for subject changes, and regenerate eligible examples with new seeds. By showing these assets in realistic scenes rather than against white backgrounds, we can narrow their gap to human video and to the scenes seen at inference. The resulting dataset, "Objaverse in Video" (OiV), contains 118K videos from 43K assets. We complement OiV with 60K human and 27K robot video clips with 3D motion, and 125K motion-only sequences (App. A.6). Qualitative examples are provided on our project page and supplemental.

3.5 Training

We train the motion denoiser and rig encoder from scratch, and separately adapt the video backbone with LoRA. Joint training then updates these two, the LoRA adapters, the cross-modal bridges, and the localization head, leaving the base video weights frozen; captions, the reference image with its 2D joint coordinates, and the bridges are randomly dropped, and in some steps one modality is kept clean to serve as a condition. See App. A.7 for details.

4 Experiments

We demonstrate that JCA outperforms video-first and mesh-based baselines in text-driven rig animation, surpasses a specialized human model after fine-tuning, and enables scene-aware, conditional, keyframe-controlled, and image-based animation. Quantitative evaluations and qualitative comparisons show that its motion follows text faithfully, preserves asset geometry, stays synchronized with generated video, and responds to the surrounding scene.

Method	Prompt adherence ↑	Motion smoothness ↑	Dynamic degree	Mesh distortion ↓
AnimateAnyMesh	20.6	**.993**	.11	1.96
Puppeteer	41.1	.984	.80	0.36
ActionMesh	43.4	.988	.50	7.07
JCA (Ours)	**58.4**	.983	.83	**0.20**

Table 1: Rig animation from text, averaged over 100 asset–prompt pairs. Prompt adherence is judged by GPT-6 Astra on a 0–100 scale. Mesh distortion is in percent.

Table 2: Joint video–human-motion generation. Synchronization compares generated and GVHMR-recovered motion; video and dynamics metrics follow VBench.

	Synchronization			Video quality				Dynamics	
Method	PA-MPJPE ↓	SA-MPJPE ↓	SA-MPJVE ↓	IQ ↑	Aesth. ↑	Smooth. ↑	Anat. ↑	Dyn. degree	Non-static camera
EchoMotion	97.7	114.9	431.3	**.665**	.533	.985	.730	.828	74.0%
JCA (ours)	114.8	151.9	598.2	.664	**.578**	.992	.934	.600	45.6%
JCA SMPL-FT	**72.3**	**91.0**	**272.6**	.657	.573	**.994**	**.938**	.488	44.8%

4.1 Rig Animation

We evaluate how effectively our generated motion animates diverse rigged assets from text. We randomly sample 20 assets from the Puppeteer test set (Song et al., 2025) and use GPT-5.6 Sol to write five action prompts for each, giving 100 asset–prompt pairs. We compare against AnimateAnyMesh (Wu et al., 2025) and ActionMesh (Sabathier et al., 2026), which animate mesh vertices without a rig, and Puppeteer (Song et al., 2025), which animates the ground-truth rig. We cannot include AnimaX (Huang et al., 2025), because its code, weights, and evaluation benchmark, are not publicly available. Puppeteer and ActionMesh recover motion from a generated guiding video, representing video-first pipelines. All methods start from the same posed asset: the baselines receive it in 3D, whereas ours sees it only as a rendered image with 2D joints (section B).

Metrics. We evaluate prompt adherence with GPT-6 Astra as a multimodal judge, since text-driven motion has no ground truth. The judge scores each animation against its prompt from 0 to 100, blind to method identity (section B). We measure how well the asset's geometry is preserved, because vertex-based methods can violate it, and report mesh distortion: the median absolute change in edge length relative to the first frame. Finally, we report VBench's motion smoothness and dynamic degree (Huang et al., 2024), which together separate smooth motion from motion that barely moves.

Results. Our method outperforms all baselines in prompt adherence by a wide margin, scoring 58.4 against 43.4 for the strongest baseline (table 1 and fig. 3), with the lowest mesh distortion and the highest dynamic degree. The two strongest baselines recover motion from a video generated without knowledge of the rig, supporting our premise that the rig should inform generation rather than enter only afterward. AnimateAnyMesh is the smoothest, but its low dynamic degree shows that its motion is subtle.

4.2 Scene-Aware Rig Animation

Prior work on arbitrary-rig animation, as discussed in section 2, has focused on isolated meshes without considering their interaction with the surrounding environment. Our training procedure and strong video-generation backbone enable our model to generate scene-aware motion that is consistent with the asset's environment. Given a rigged asset and a 3D scene, we place the asset at a user-specified location and pose and render this configuration as the first frame. From a text prompt, the model then generates a video that respects the scene geometry, together with the corresponding 3D motion. For example, the ape in fig. 1 jumps over a crossbar, and the penguin in fig. 5 slides on its belly across the snow (more examples in our page).

4.3 Joint Generation for Humans

Our method is designed for arbitrary rigs. When using a human skeleton, it can be directly compared with EchoMotion (Yang et al., 2025), a human-specific model that jointly generates an RGB video and an SMPL motion sequence from text. We evaluate two variants: our general model, JCA, and JCA SMPL-FT. JCA SMPL-FT is the general checkpoint fine-tuned for 10K steps on our human subset, with motion represented as SMPL, the body model EchoMotion uses. Both variants are

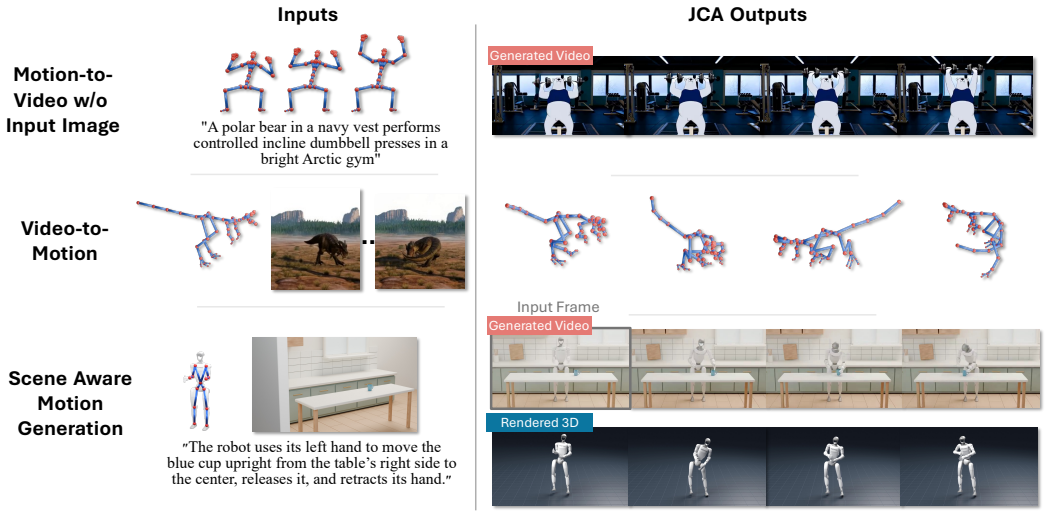


Figure 4: Additional Applications of our model. **Top:** Unlike the setting in fig. 1, we generate a video from a rig motion sequence and a text prompt, without an input image. **Middle:** An additional video-to-motion example with a different skeletal topology. **Bottom:** Given the rig and mesh of a real Unitree G1 robot, a 3D scene, an initial pose, and text, we jointly generate video and motion; we show the video alongside the mesh driven by the generated motion and rendered in the scene.

evaluated on 250 prompts from the HumanML3D test split (Guo et al., 2022). EchoMotion runs text-to-video, while we run ours with a generated reference image (section C).

Metrics. We evaluate synchronization, whether the generated motion matches the generated video, by recovering an SMPL trajectory from each video with GVHMR (Shen et al., 2024) and comparing it with the generated one. PA-MPJPE, SA-MPJPE, and SA-MPJVE measure agreement in per-frame pose, in orientation over time, and in timing, respectively. For video quality, we follow EchoMotion and report VBench metrics, adding the non-static camera rate since dynamic degree also counts camera motion (section C).

Results. Without human-specific training, our general model trails EchoMotion in synchronization (table 2). After fine-tuning, JCA SMPL-FT surpasses it on all three metrics, reducing PA-MPJPE by 26% and SA-MPJVE by 37%. Both variants produce markedly better human anatomy (.93 vs. .73) and higher aesthetic quality. EchoMotion has a higher dynamic degree, but it also moves the camera in 74% of its videos, compared with about 45% for ours, so part of its dynamics comes from the camera rather than the body.

4.4 Video-to-Motion and Motion-to-Video.

Holding one modality clean and denoising only the other turns joint generation into conditional generation. With a clean video, our model estimates motion for the supplied rig, and after fine-tuning it matches a dedicated video-to-motion method without receiving the first-frame pose (App. D). Conversely, given a 3D trajectory and its rig, together with a text prompt, we keep the motion fixed at zero noise and denoise only the video. The generated videos follow the supplied trajectory across arbitrary rigs, while the prompt determines how the subject looks and the environment it moves in. This works both with a first frame reference image (fig. 1) and without one (fig. 4).

4.5 Keyframe-Controlled Motion.

Our coupled model supports keyframe-conditioned video–motion in-betweening without additional training. Given a text prompt, a static rig, and RGB keyframes at specified temporal locations, we construct a sparse reference video by inserting the supplied images and filling unspecified frames with zeros in normalized image space. Following WAN's image-conditioning protocol, we provide the VAE-encoded reference video together with a temporal mask indicating the supplied keyframes. The video and motion streams are then jointly denoised, allowing the visual keyframe conditions to

Table 3: Ablation of the cross-modal connections.

Configuration	PA-MPJPE ↓	SA-MPJPE ↓	SA-MPJVE ↓
JCA (Ours)	**72.27**	**90.98**	**272.58**
No Cross-CFG	84.78	103.18	293.06
Disconnected cross-attention	140.71	173.38	456.07

guide the generated 3D rig trajectory through cross-modal coupling. No target 3D motion sequence or final-frame 3D pose is required. Qualitative examples of keyframe-controlled generation are shown in fig. 1.

All applications are demonstrated with diverse examples on our project page and supplemental.

5 Ablations

5.1 Strengthening Video–Motion Coupling

We evaluate the effect of applying bridge classifier-free guidance to the cross-attention layers connecting the video and motion branches, as described in section 3.2. We follow the evaluation protocol introduced in section 4.3. As an additional ablation, we completely disconnect the two branches to directly assess the importance of cross-branch connectivity.

As expected, disconnecting the branches degrades video–motion synchronization (table 3). The results also show a clear benefit from bridge classifier-free guidance, which further improves synchronization over the connected baseline.

5.2 Motion Only

To assess the contribution of joint video–motion training, we compare our model with its motion-only pretrained checkpoint, trained on the same dataset using only motion supervision. We evaluate both models on the same procedure as in section 4.1 using matched sampling settings. Joint training raises prompt adherence from 51.6 to 58.4 and dynamic degree from .72 to .83, while mesh distortion stays low for both (0.14% and 0.20%).

6 Conclusion, Limitations, and Future Work

We presented JCA, a model that co-generates video and 3D motion for arbitrary rigs, from humans and robots to animals. A separate, graph-based motion model lets us pretrain on motion alone for any skeleton, and a few zero-initialized cross-attention layers couple it to a pretrained video model at a small parameter cost, with bridge guidance tightening their agreement at inference. The model also learns where each joint appears in the video, which lets us recover a camera and place the motion in the scene. For training data, we took animated 3D assets and had a video editor drop them into realistic scenes. The result animates diverse rigs from text more faithfully than video-first pipelines, and the same model turns video into rig motion, rig motion into video, and image keyframes into 3D motion.

Limitations and future work. Our alignment procedure assumes a static camera at inference time; extending it to moving cameras would support more general scenes. Our training rigs also skew toward bipeds: most animated Objaverse characters are humanoid, and our additional motion-only data is humanoid as well. Because the motion model is pretrained separately, it could be broadened with motion data alone, such as animal motion capture, without collecting paired video. While the generated motion responds to the scene, the scene reaches it only through the video: the model never sees the 3D scene geometry, so nothing enforces contact or prevents penetration. Conditioning the motion branch on that geometry, or adding contact terms during training or guidance, could turn scene-consistent video into physically consistent 3D motion. Finally, objects the subject interacts with move in the video but are not recovered in 3D, and each clip animates a single rig. Lifting both is an exciting direction, taking co-generation from animating an asset to animating a scene.

AI USE STATEMENT

In this work, we used generative AI tools for synthetic-data generation and curation, feedback on research methodology and experimental design, implementation and debugging, evaluation, literature search, figure preparation, and writing. In our data pipeline, Qwen selected suitable scenes, described actions, and assessed physical plausibility, while Bernini edited rendered animations into realistic scenes. For evaluation, GPT-5.6 Sol generated asset-specific action prompts, ChatGPT helped select benchmark prompts and construct reference-image prompts, and GPT-6 Astra served as a blinded multimodal judge for prompt adherence.

Agentic coding tools assisted with method implementation, training and evaluation scripts, data processing, debugging, qualitative comparisons, and visualizations. LLMs also assisted with literature discovery and verification, reference formatting, paper organization, drafting, copyediting, and occasional translation. They were not used to formulate or prove theoretical claims; proof-related tasks are not applicable to this work.

The authors reviewed all AI-assisted methodology, code, and writing. Suggested citations were checked against the original publications, AI-assisted code was tested by the authors, and all reported results come from our evaluation runs. Model-generated training data was filtered using the checks described in the paper, and the authors inspected the resulting pipeline and samples. We take responsibility for the final content of this work, including all text, claims, code, data, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We support reproducibility by documenting the model, data pipeline, and evaluation protocol. The main paper describes the model inputs and outputs, motion representation, rig conditioning, video–motion coupling, guidance, localization, and camera fitting, and outlines the training procedure. The appendix provides further implementation details for these components, together with the data sources, preprocessing steps, dataset sizes, and checks used to curate edited videos. For the baseline comparisons, we specify the benchmarks, the inputs provided to each baseline, and the metric definitions; for rig animation, we give the temporal and rendering settings and the multimodal-judge protocol, with the complete prompts and responses for one judged example. The supplementary material includes video examples.

REFERENCES

Yarin Bekor, Gal Michael Harari, Or Perel, and Or Litany. Gaussian see, gaussian do: Semantic 3D motion transfer from multiview video. In *SIGGRAPH Asia 2025 Conference Papers*, 2025. URL https://arxiv.org/abs/2511.14848.

Shariq Farooq Bhat, Niloy J Mitra, and Kalyan Sunkavalli. Loosecontrolvideo: Directorial video control using spatial blocking. In *European Conference on Computer Vision*, pp. 73–90. Springer, 2026.

Bones Studio. BONES-SEED: Skeletal everyday embodiment dataset. https://huggingface.co/datasets/bones-studio/seed. Motion Data by Bones Studio. Accessed 2026-09-26.

Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=F72ximsx7C1.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18392–18402, June 2023.

Hila Chefer, Uriel Singer, Amit Zohar, Yuval Kirstain, Adam Polyak, Yaniv Taigman, Lior Wolf, and Shelly Sheynin. VideoJAM: Joint appearance-motion representations for enhanced motion generation in video models. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=yMJcHWcb2Z.

Honglin Chen, Karran Pandey, Rundi Wu, Matheus Gadelha, Yannick Hold-Geoffroy, Ayush Tewari, Niloy J. Mitra, Changxi Zheng, and Paul Guerrero. Vips: Video-informed pose spaces for auto-rigged meshes. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia 2026)*, 2026.

Jihoon Chung, Cheng hsin Wuu, Hsuan ru Yang, Yu-Wing Tai, and Chi-Keung Tang. Haa500: Human-centric atomic action dataset with curated videos. In *ICCV 2021*.

Lingwei Dang, Ruizhi Shao, Hongwen Zhang, Wei MIN, Yebin Liu, and Qingyao Wu. Svimo: Synchronized diffusion for video and motion generation in hand-object interaction scenarios. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 76689–76718. Curran Associates, Inc., 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/6f08ce5b354c4afefeff51e6fad306a6-Paper-Conference.pdf.

Zipeng Fu, Tony Z. Zhao, and Chelsea Finn. Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. In *Conference on Robot Learning (CoRL)*, 2024.

Mingqi Gao, Yunqi Miao, and Jungong Han. Sam-body4d: Training-free 4d human body mesh recovery from videos. *arXiv preprint arXiv:2512.08406*, 2025. URL https://arxiv.org/abs/2512.08406.

Inbar Gat, Sigal Raab, Guy Tevet, Yuval Reshef, Amit H Bermano, and Daniel Cohen-Or. Anytop: Character animation diffusion with any topology. In *SIGGRAPH 2025 Conference Papers*, 2025.

Kehong Gong, Zhengyu Wen, Dao Thien Phong, Mingxi Xu, Weixia He, Qi Wang, Ning Zhang, Zhengyu Li, Guanli Hou, Dongze Lian, Xiaoyu He, Mingyuan Zhang, and Hanwang Zhang. Mocapanything v2: End-to-end motion capture for arbitrary skeletons. *ACM Transactions on Graphics*, 2026. To appear (SIGGRAPH Asia 2026).

Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating diverse and natural 3d human motions from text. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5152–5161, June 2022.

Songhao Han, Boxiang Qiu, Yue Liao, Siyuan Huang, Chen Gao, Shuicheng Yan, and Si Liu. Robocerebra: A large-scale benchmark for long-horizon robotic manipulation evaluation. *arXiv preprint arXiv:2506.06677*, 2025.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Yidong Huang, Zun Wang, Han Lin, Dong-Ki Kim, Shayegan Omidshafiei, Jaehong Yoon, Jaemin Cho, Yue Zhang, and Mohit Bansal. Phymotion: Structured 3d motion reward for physics-grounded human video generation. *arXiv preprint arXiv:2605.14269*, 2026.

Zehuan Huang, Haoran Feng, Yangtian Sun, Yuanchen Guo, Yanpei Cao, and Lu Sheng. Animax: Animating the inanimate in 3d with joint video-pose diffusion models. *arXiv preprint arXiv:2506.19851*, 2025.

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

Angjoo Kanazawa, Michael J. Black, David W. Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *Computer Vision and Pattern Recognition (CVPR)*, 2018.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, Vitor Guizilini, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Muhammad Zubair Irshad, Donovon Jackson, Charlotte Le, Yunshuang Li, Kevin Lin, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O'Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. Droid: A large-scale in-the-wild robot manipulation dataset. 2024.

Chung Min Kim, Brent Yi, Hongsuk Choi, Yi Ma, Ken Goldberg, and Angjoo Kanazawa. Pyroki: A modular toolkit for robot kinematic optimization. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025. URL https://arxiv.org/abs/2505.03728.

Black Forest Labs. Flux. https://github.com/black-forest-labs/flux, 2024.

Xiaolei Lang, Yang Wang, Yukun Zhou, Chaojun Ni, Kerui Li, Jiagang Zhu, Tianze Liu, Jiajun Lv, Xingxing Zuo, Yun Ye, et al. Vag: Dual-stream video-action generation for embodied data synthesis. *arXiv preprint arXiv:2604.09330*, 2026.

Xuan Li, Qianli Ma, Tsung-Yi Lin, Yongxin Chen, Chenfanfu Jiang, Ming-Yu Liu, and Donglai Xiang. Articulated kinematics distillation from video diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Jingyun Liang, Min Wei, Shikai Li, Yizeng Han, Hangjie Yuan, Lei Sun, Weihua Chen, and Fan Wang. Towards 3d-aware video diffusion models: Render-free human motion control with mesh tokenization. *arXiv preprint arXiv:2606.02000*, 2026.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Chenchen Liu, Junyi Chen, Lei Li, Lu Chi, Mingzhen Sun, Zhuoying Li, Yi Fu, Ruoyu Guo, Yiheng Wu, Ge Bai, and Zehuan Yuan. Bernini: Latent semantic planning for video diffusion. *arXiv preprint arXiv:2605.22344*, 2026.

Tiange Luo, Chris Rockwell, Honglak Lee, and Justin Johnson. Scalable 3d captioning with pretrained models. *Advances in Neural Information Processing Systems*, 36:75307–75337, 2023.

Tiange Luo, Justin Johnson, and Honglak Lee. View selection for 3d captioning via diffusion ranking. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 180–197, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-72751-1.

Yanzhe Lyu, Chen Geng, Karthik Dharmarajan, Yunzhi Zhang, Hadi Alzayer, Shangzhe Wu, and Jiajun Wu. Choreographing a world of dynamic objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 32647–32658, June 2026.

Amir Mann, Gal Michael Harari, Merav Keidar, and Or Litany. VideoMDM: Towards 3D human motion generation from 2D supervision. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2026. URL https://arxiv.org/abs/2606.13364.

Julieta Martinez, Rayat Hossain, Javier Romero, and James J. Little. A simple yet effective baseline for 3d human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

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

Youxin Pang, Yong Zhang, Ruizhi Shao, Xiang Deng, Feng Gao, Xu Xiaoming, Xiaoming Wei, and Yebin Liu. Unimo: Unifying 2d video and 3d human motion with an autoregressive framework. *arXiv preprint arXiv:2512.03918*, 2025.

Cheng-feng Pu, Jia-peng Zhang, Meng-hao Guo, Yan-Pei Cao, and Shi-Min Hu. Topocap: Learning topology-agnostic motion priors for monocular video-to-animation. *ACM Transactions on Graphics (TOG)*, 45(4), July 2026. doi: 10.1145/3799902.3811159.

Davis Rempe, Mathis Petrovich, Ye Yuan, Haotian Zhang, Xue Bin Peng, Yifeng Jiang, Tingwu Wang, Umar Iqbal, David Minor, Michael de Ruyter, Jiefeng Li, Chen Tessler, Edy Lim, Eugene Jeong, Sam Wu, Ehsan Hassani, Michael Huang, Jin-Bey Yu, Chaeyeon Chung, Lina Song, Olivier Dionne, Jan Kautz, Simon Yuen, and Sanja Fidler. Kimodo: Scaling controllable human motion generation. *arXiv:2603.15546*, 2026.

Remy Sabathier, David Novotny, Niloy J Mitra, and Tom Monnier. Actionmesh: Animated 3d mesh generation with temporal 3d diffusion. *CVPR 2026*, 2026.

Jun Saito, Jiefeng Li, Michael de Ruyter, Miguel Guerrero, Edy Lim, Ehsan Hassani, Roger Blanco Ribera, Hyejin Moon, Magdalena Dadela, Marco Di Lucca, et al. Soma: Unifying parametric human body models. *arXiv preprint arXiv:2603.16858*, 2026.

Zehong Shen, Huaijin Pi, Yan Xia, Zhi Cen, Sida Peng, Zechen Hu, Hujun Bao, Ruizhen Hu, and Xiaowei Zhou. World-grounded human motion recovery via gravity-view coordinates. In *SIGGRAPH Asia Conference Proceedings*, 2024.

Assaf Singer, Noam Rotstein, Amir Mann, Ron Kimmel, and Or Litany. Time-to-Move: Training-free motion controlled video generation via Dual-Clock denoising. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://arxiv.org/abs/2511.08633.

Chaoyue Song, Xiu Li, Fan Yang, Zhongcong Xu, Jiacheng Wei, Fayao Liu, Jiashi Feng, Guosheng Lin, and Jianfeng Zhang. Puppeteer: Rig and animate your 3d models. *Advances in Neural Information Processing Systems*, 2025.

Giacomo Spigler. Tavis: A benchmark for egocentric active vision and anticipatory gaze in imitation learning, 2026. URL https://arxiv.org/abs/2605.07943.

Chen Tessler, Yunrong Guo, Ofir Nabati, Gal Chechik, and Xue Bin Peng. Maskedmimic: Unified physics-based character control through masked motion inpainting. *ACM Transactions on Graphics (TOG)*, 2024.

Truebones Motions Animation Studios. TrueBones, 2022. URL https://truebones.gumroad.com/. Accessed: 2026-09-26.

Zijie Wu, Chaohui Yu, Fan Wang, and Xiang Bai. Animateanymesh: A feed-forward 4d foundation model for text-driven universal mesh animation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 13557–13568. IEEE, 2025.

Yuxiao Yang, Hualian Sheng, Sijia Cai, Jing Lin, Jiahao Wang, Bing Deng, Junzhe Lu, Haoqian Wang, and Jieping Ye. Echomotion: Unified human video and motion generation via dual-modality diffusion transformer. *ICLR 2026*, 2025.

Zhengdi Yu, Stefanos Zafeiriou, and Tolga Birdal. Dyn-hamr: Recovering 4d interacting hand motion from a dynamic camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2025.

Hao Zhang, Jiahao Luo, Bohui Wan, Yizhou Zhao, Zongrui Li, Michael Vasilkovsky, Chaoyang Wang, Jian Wang, Narendra Ahuja, and Bing Zhou. Rigmo: Unifying rig and motion learning for generative animation. *arXiv preprint arXiv:2601.06378*, 2026.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3813–3824. IEEE, 2023.

Yuhong Zhang, Jing Lin, Ailing Zeng, Guanlin Wu, Shunlin Lu, Yurong Fu, Yuanhao Cai, Ruimao Zhang, Haoqian Wang, and Lei Zhang. Motion-x++: A large-scale multimodal 3d whole-body human motion dataset, 2025. URL https://arxiv.org/abs/2501.05098.

Chengfeng Zhao, Jiazhi Shu, Yubo Zhao, Tianyu Huang, Jiahao Lu, Zekai Gu, Chengwei Ren, Zhiyang Dou, Qing Shuai, and Yuan Liu. Comovi: Co-generation of 3d human motions and realistic videos. 2026a.

Yiming Zhao, Haoyu Sun, and Aoyu Wang. Animaspark: A feed-forward method for animating arbitrary 3d objects, 2026b. URL https://arxiv.org/abs/2606.10988.

A Appendix

A.1 Motion Representation

Each joint j at frame n is represented by

$$x_{n,j} = \left[\rho(R_{n,j}),\ \rho(R_{n,j}^{\top} R_{n+1,j}),\ a_{n,j},\ a_{n+1,j} - a_{n,j}\right] \in \mathbb{R}^{18}, \tag{1}$$

and a clip is the stack $X \in \mathbb{R}^{N \times J \times 18}$. The second entry encodes the relative rotation between consecutive stored orientations, rather than the difference between their 6D encodings.

The root and the remaining joints use different conventions. For the root, $R_{n,r}$ and $a_{n,r} = p_{n,r}$ are the rotation and position in world coordinates, and we do not re-center them on the first frame, so a trajectory keeps its absolute placement in the scene. For every other joint, $R_{n,j}$ is the local rotation under the rig's bind-frame convention, and $a_{n,j} = R_{n,r}^{\top}(p_{n,j} - p_{n,r})$ expresses the joint in the root's frame, which leaves it unchanged when the body is translated or turned.

Every channel of Eq. 1 is standardized with fixed, dataset-specific statistics. The root and the non-root joints receive their own statistics, since the two hold different quantities: world rotation and absolute position for the root, a bind-frame rotation and a root-relative offset for the rest. The statistics are fixed rather than learned, so a generated clip is decoded with the same constants it was trained under.

A.2 Rig Descriptor

A descriptor is assembled per joint before any graph operation. Its geometric inputs, the joint's 3D rest position and its 2D coordinate in the reference image, are augmented with sinusoidal positional encodings.

The graph network is four GATv2 layers (Brody et al., 2022). Messages travel in both directions along every parent–child edge, and a learned edge-direction embedding indicates the direction each message travelled, distinguishing a parent from a child. Full self-attention over all joints of the rig follows the graph layers, allowing joints that are distant in the hierarchy to exchange information.

A.3 Denoiser

The backbone is ten blocks deep. Each block applies five operations in order: self-attention over all joints and all frames; cross-attention to the encoded text prompt; a feed-forward update; graph attention over the rig's connectivity, computed within a single frame and separate from the layers that build the descriptors; and a second feed-forward update belonging to that graph step.

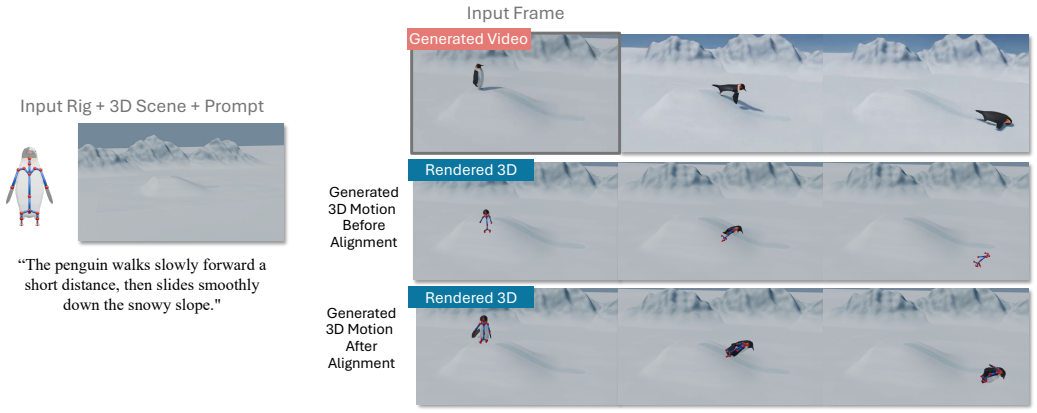


Figure 5: Scene-aware rig animation and motion localization. Given a penguin rig, a rendered initial frame of a snowy 3D scene, and a text prompt, our model jointly generates video and 3D rig motion. **Top:** frames from the generated video, where the penguin moves forward and slides down the snowy slope. **Middle:** the raw 3D motion produced by the motion branch, rendered in the input scene before localization. The penguin's motion is not aligned with the depicted scene geometry and, in later frames, appears beneath the snow surface. **Bottom:** the same generated motion after applying our joint-localization and camera-alignment scheme described in section 3.3. The estimated alignment correctly places the motion on the visible penguin and registers it with the generated video and scene.

The rig descriptors are the same at every block, but the linear map that projects them into the token space is learned separately per block, so the ten blocks hold ten such projections. Each block begins by adding its projection of joint j's descriptor to that joint's tokens.

The noise level s_m is turned into a vector by a sinusoidal embedding, and that vector sets three quantities inside the block: the scale and the shift of the normalization layers, and the gates on the residual branches.

The output MLPs predict flow velocities in the standardized space of App. A.1. After denoising, we reverse the standardization of the generated motion features before decoding their rotations and positions. We animate the rig from the decoded positions, following Gat et al. (2025): the root's trajectory is obtained by integrating its predicted frame-to-frame displacement, and the remaining positions are projected onto the rig's fixed bone lengths by inverse kinematics.

A.4 Coupling Video and Motion

Each of the motion denoiser's blocks is paired with one block of the video backbone. The pairings are spread approximately uniformly through the backbone and include its first and last blocks.

At a paired block, both branches have already normalized their hidden states and modulated them by their own noise level, as a DiT block does just before self-attention. These modulated states are the $\bar{H}_v$ and $\bar{H}_m$ of Sec. 3.2; because each branch carries its own noise level, each side exchanges features that reflect its own progress through denoising. The query, key, and value projections of each direction map both modalities into a shared attention space.

Both directions apply query–key normalization before the dot product.

The branches sample time at different densities: the motion branch keeps one entry per video frame, whereas the video branch works in latent frames, one per four video frames after the first. Cross-modal attention expresses both in latent-frame units, giving latent frame f the coordinate f and motion frame n the coordinate $n/4$, and the temporal rotary embeddings act on these coordinates. The attention is not masked by frame: a joint may attend to patches from any moment of the clip, not only the moment it is predicting.

This positional encoding is temporal only, and joint indices receive no spatial coordinates. What a joint is, and how it sits in the rig, reaches the attention through the rig descriptors already carried by the motion features (Sec. 3.1).

Bridge guidance. The bridge guidance of Sec. 3.2 applies at inference only. Like the two-condition guidance of Brooks et al. (2023), they use two guidance scales, but where that formulation combines predictions from three conditioning configurations, we use four: text guidance is applied within each bridge state, and the change between the two results is then scaled. The scales are specified per modality, so the influence of motion on video and of video on motion can be set independently. We set $\omega_{\text{text},i} = 3$ for both modalities. Training also disables the bridges (App. A.7), so the disconnected state occurs during training as well as at inference. In general, each guided step requires four predictions, covering both prompts and both bridge states. All four start from the same noisy video and motion states at the same noise levels, and share the rig and reference conditions; only the prompt and the bridge state differ.

A.5 Joint Localization and Camera Alignment

Small MLPs map the final motion and video features to a query $q_{f,j}$ for each joint j and a key $k_{f,p}$ for each of the P patches at latent frame f, both normalized to unit length. Writing $\xi_p \in \mathbb{R}^2$ for the center of patch p in normalized image coordinates, so that $\Xi \in \mathbb{R}^{P\times 2}$ collects these centers,

$$A_{f,j,p} = \frac{\exp\left(\kappa\, q_{f,j}^\top k_{f,p}\right)}{\sum_{p'} \exp\left(\kappa\, q_{f,j}^\top k_{f,p'}\right)}, \qquad \widehat{y}_{f,j} = \sum_p A_{f,j,p}\, \xi_p.$$

Since the queries and keys are of unit length, the logits are cosine similarities, and the learned positive κ, which replaces the $1/\sqrt{d_k}$ scaling of the attention in Sec. 3.2, determines how sharply the weights concentrate on a single patch. The head has no output projection and no residual connection, and it attends only within the frame it is predicting, unlike the cross-modal attention of Sec. 3.2, which attends across the whole clip.

During training, the head operates on the final-layer features at the noise level sampled for that step and predicts the clean 2D locations, supervised against the 2D joint annotations at motion frame $4f$. These annotations serve as supervision only; the denoiser never receives them as input, in contrast to the optional reference-image joint coordinates that enter the rig descriptor (Sec. 3.1). They are obtained by projecting the ground-truth 3D joints with the recorded rendering camera (Sec. 3.4).

At inference, the head predicts 2D locations at every denoising step, and we average the predictions of the second half of the steps.

Camera fitting assumes a static camera over the clip. The predicted locations, defined at frames $4f$, are interpolated to every motion frame, and a single perspective camera is fitted to the resulting 2D–3D pairs under a reliability-weighted robust reprojection objective. We estimate rotation, translation, and focal length, holding the principal point at the image center. This step uses only the model's own predictions, requires neither a ground-truth camera nor an external tracker, and does not modify the generated motion.

A.6 Data

All sources below are converted into the motion representation of Sec. 3.1; the video sources also carry 2D joint annotations.

Assets and rig curation. We start from Diffusion4D's curated animated-asset lists, 82K candidates drawn from Objaverse-1.0 and Objaverse-XL. Each imported asset is inspected automatically. During processing, we remove joints with zero or negligible skinning weight on the mesh, and keep assets with a usable remaining skeleton, visible animated geometry, and usable motion. The original rigs are stored separately from the processed representations.

Geometrically consistent motion. Source animations do not always preserve bone lengths, so each trajectory is rebuilt. Joint positions are recomputed along the skeleton hierarchy from the source rotations and the bone offsets of frame zero. A root-relative rest skeleton is then derived from the source rest directions and these fixed lengths, and the source rotations are re-expressed in it. Finally,

we place the floor at zero, align the vertical axis to $+Y$, remove horizontal camera yaw, and project the finalized 3D joints with the recorded camera to obtain the 2D annotations.

Rendering. Each asset is rendered up to four times, with randomized camera viewpoint, distance, framing, and lighting direction and intensity. The subject is rendered against a temporary checkerboard floor and backdrop, which gives the editor explicit ground-plane and perspective cues. The floor is omitted for motion that does not touch it, such as flying or hovering.

Generating an appropriate scene for each asset. Cap3D Luo et al. (2023) and DiffuRank Luo et al. (2025) provide captions for the Objaverse assets, but these captions mainly describe their static appearance. We use Qwen to choose an environment that fits each asset and to describe the action in its rendered clip. We then combine the object, action, and environment descriptions into the editing instruction used to generate the scene. To increase environmental diversity while preserving the subject and its action, we vary the phrasing, add local scene details, and substitute related environment subtypes.

The video editor, Bernini (Liu et al., 2026), receives the rendered clip and an instruction to replace the temporary setting while preserving the subject's appearance, silhouette, position in frame, apparent scale, pose timing, and motion. These preservation constraints are stated explicitly in the editing prompt. To keep the recorded camera valid, we retain only clips generated under fixed-camera instructions, and we exclude clips classified as in-place locomotion.

Verifying that the edited video remains consistent with the source animation. VGGT then verifies that the edited video wasn't generated with a moving camera, and only retains a static camera. More importantly, we compare SAM segmentations before and after editing to verify that the subject remains in the same image region; otherwise, the 2D joint annotations obtained from the render would no longer match the edited video. Finally, we ask Qwen whether the edited result is physically plausible, filtering cases such as a person appearing to walk in the air. An eligible clip that fails these checks is regenerated with a new random seed and checked again. For each asset and render variant we keep the first attempt that passes, so retries never add examples. OiV covers rigs with 2 to 256 joints and is partitioned into 110K training, 4,000 validation, and 4,000 test examples, with all variants of an asset assigned to the same split.

Human video. Real human video datasets with 3D motion offer natural scenes and motion, but no such dataset is publicly available online at scale (Yang et al., 2025; Zhao et al., 2026a; Zhang et al., 2025). Human-action clips come from OpenVid and HAA500 (Nan et al., 2024; Chung et al.); the joint-training set contains 57K OpenVid clips and 3,500 HAA500 clips. Neither provides motion in our rig representation, so we construct it: SAM-Body4D reconstructs the body, tracking the person through the whole clip and handling occlusion, and a quality-aware pipeline based on Dyn-HaMR reconstructs the hands (Gao et al., 2025; Yu et al., 2025). EchoMotion and CoMoVi annotate the body alone and filter their input videos before reconstruction (Yang et al., 2025; Zhao et al., 2026a); we also reconstruct the hands and additionally reject unreliable reconstructions after they are made. Accepted trajectories then receive hand refinement and planted-foot post-processing, after which we construct a fixed-bone rest skeleton and derive the corresponding joint rotations. From HAA500 we keep only clips with one dominant person and an approximately static camera. The reconstructed trajectories and estimated cameras provide aligned 3D and 2D annotations. Captions come from OpenVid directly and from Motion-X++ for HAA500 (Zhang et al., 2025); mismatched or uncertain HAA500 captions are replaced with descriptions based on the action class.

Robotics. Robotic manipulation sequences, real and simulated, come from DROID, ALOHA, RoboCerebra, and TAVIS (Khazatsky et al., 2024; Fu et al., 2024; Han et al., 2025; Spigler, 2026). The joint-training set contains 25K DROID clips, 50 ALOHA clips, 912 RoboCerebra clips, and 570 TAVIS clips. Their 3D joint trajectories are computed using each dataset's kinematic model. We transform the trajectories into our training coordinate convention and project them to the camera plane to obtain 2D annotations. The original task descriptions serve as text conditioning.

Motion without video. Motion pretraining is augmented with two further sources that carry 3D motion but no video. To broaden the range of skeletal structures, we retarget BONES-SEED motions (Bones Studio) to randomized variants of the SOMA humanoid Saito et al. (2026), yielding 100K augmented motion sequences. The variants change body proportions, split or merge body segments, and optionally remove the head, while keeping a torso, two arms, and two legs. Each motion is retargeted with PyRoki (Kim et al., 2025) by matching corresponding body keypoints, which

yields motion across varied proportions and joint layouts without collecting additional video. We also use 25K sequences from the official HumanML3D training split (Guo et al., 2022), converting its 22-joint sequences into our motion representation.

A.7 TRAINING

Motion pretraining runs for 200,000 updates on motion sequences from the paired corpus together with additional unpaired motion data. Separately, the pretrained Wan2.2 I2V-A14B backbone is adapted to our video corpus with rank-32 LoRA adapters (Hu et al., 2022) for 10,000 updates, without motion conditioning. Both branches use flow-matching objectives.

During joint training, we update the motion denoiser, the rig encoder, the cross-modal bridges, the localization head, and the rank-32 video LoRA adapters, while the base video weights, the VAE, and the text encoder remain frozen. These components total 331.28M trainable parameters: 153.35M in the video LoRA adapters, 115.45M in the video–motion cross-attention layers, 56.06M in the motion model, including the 2D prediction head, and 6.41M in the rig encoder. The backbone holds two expert denoisers covering different noise ranges; they share a single set of LoRA parameters, and we select the high-noise expert when $s_v \geq 0.9$ and the low-noise expert otherwise. The loss combines the standard flow-matching loss of each branch, with the smooth-L_1 localization loss of Sec. 3.3.

Conditions are dropped at random during training. Both captions are replaced by empty captions with probability 0.15, and the reference image and its 2D joint coordinates are dropped together with probability 0.20 to support generation without reference-image conditioning. Separately, we disable the cross-modal bridges, which lets both branches continue learning independently and enables bridge guidance (Sec. 3.2).

In a random subset of training steps, one modality is kept clean. This exposes the model to conditioning on a fully observed video or motion, which enables video-to-motion and motion-to-video generation at inference.

B RIG-ANIMATION EVALUATION DETAILS

We provide additional details for the rig-animation experiment described in section 4.1, in which each method animates 20 assets from five text prompts each.

B.1 BASELINE SETUP

Asset provenance. We verify each selected asset against our training data and exclude any confirmed or unresolved identity overlap. The same check is not possible for ActionMesh because its training data include Objaverse-derived assets. Some evaluated meshes or initial poses may therefore have been seen during training, although all action prompts are newly written.

Input representations. AnimateAnyMesh is rig-free and receives the posed mesh and action prompt, while ActionMesh receives the posed mesh together with a guiding video, a generated video of the requested action described below. In both cases, the mesh coordinates explicitly provide the initial 3D pose. Because our method assumes a supplied rig, we bypass Puppeteer's automatic-rigging stage and provide it with the ground-truth skeleton and skinning weights, initializing its video-guided optimization from the same first-frame 3D pose. Thus, all three baselines receive the initial 3D pose explicitly, either through mesh coordinates or optimization initialization. Our method is rig-based but does not receive the initial posed joints in 3D. Instead, it receives the rig's rest pose, hierarchy, and semantic joint names, together with a reference image, the initial joints' 2D projections, and an action prompt. The model does not observe mesh geometry or skinning weights; the latter are used only after inference to render the predicted joint motion on the asset.

Guiding videos. Neither the Puppeteer nor ActionMesh release provides specific implementation of the original video-generation component. We therefore generate the guiding videos for both methods using Wan2.2-I2V-A14B. For each asset–prompt pair, both baselines receive exactly the same video, generated from the same render of the posed asset and the action prompt. This protocol

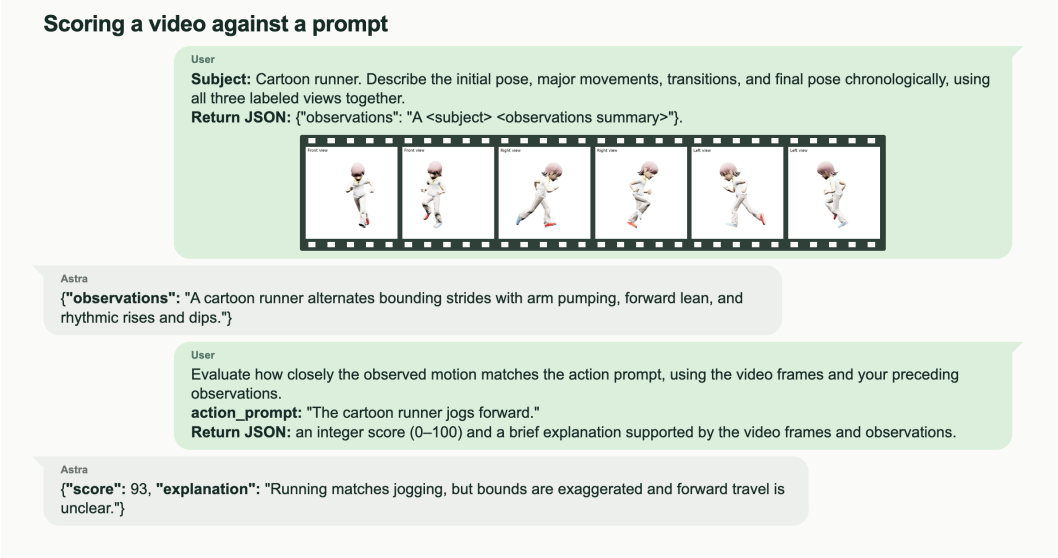


Figure 6: Prompt-adherence evaluation of one animation with GPT-6 Astra. In the first turn, the judge is given the asset's short description and 48 frames of the animation, 16 from each of the front, right, and left views, each frame labeled with its view; the figure shows two frames per view. It is asked to describe the motion it observes in chronological order. In the second turn, it scores the motion against the action prompt. The complete prompts and responses are given in section B.3.

controls the visual motion signal and isolates differences in how the methods recover an animation from it.

Temporal configuration. AnimateAnyMesh produces 16 frames at 10 FPS, corresponding to approximately 1.6 seconds. To match this duration while retaining our native temporal configuration, we generate 25 frames at 16 FPS. For a fair comparison, we provide Puppeteer and ActionMesh with the same 25-frame, 16-FPS guiding videos. Thus, all methods produce animations of approximately the same duration.

B.2 Evaluation Protocol

Rendering for evaluation. All animations are rendered from the front, right, and left views with the same Three.js viewer at 768×768 pixels on a white background, with a fixed orthographic camera, lighting, and materials. The front view uses the viewer's camera, and the right and left views rotate it by 90° in opposite directions about the vertical axis. Frames are neither interpolated nor dropped, so each method keeps its native frame count and rate. For the prompt-adherence judge described next, the three clips are concatenated in this order, with the view name written in the top-left corner of each frame.

Prompt-adherence judge. The judge is GPT-6 Astra. From each view, we sample 16 evenly spaced frames, including the first and last, and send the resulting 48 frames to the judge as JPEG images resized to fit within 512×512 pixels. Besides the frames, the judge receives only the asset's short description, such as "Cartoon runner". It is not shown the rest pose, it is not told which method produced the video, and it never sees the videos or scores of other methods. The evaluation is a conversation of two turns. In the first turn, the judge describes the motion in chronological order without seeing the action prompt. The second turn repeats the frames and this description, adds the action prompt, and asks for an integer score from 0 to 100 with a brief explanation. Both turns use low reasoning effort. fig. 6 illustrates the procedure, and section B.3 gives the complete prompts and responses for one example.

Cost. We evaluate 500 animations: 100 asset–prompt pairs for each of the four methods in table 1 and for the motion-only model of the ablation study. Averaged over these animations, one evaluation uses 30.4K input tokens and 189 output tokens over both turns. The second turn accounts for about

half of the input, because it repeats the frames. At public GPT-6 Astra prices, this is approximately $0.38 per animation.

Automatic metrics. Motion smoothness and dynamic degree are computed with the official VBench implementation on the front-view video. Mesh distortion is computed from the 3D vertex positions and is therefore independent of the view. For every mesh edge and every frame after the first, we compute $100\,|\ell_t/\ell_0 - 1|$, where ℓ_t is the edge length at frame t, and take the median of these values over all edges and frames together. Edges shorter than 10^{-6} in the first frame are excluded.

B.3 Example of a Complete Evaluation

The following prompts and responses evaluate our method on the asset "Cartoon runner" with the action prompt "The cartoon runner jogs forward." The requests are reproduced exactly; the 48 frames, which follow the first request, are omitted.

> **First request.** The subject text is exactly: Cartoon runner. The following 48 images are chronological frames from a video. Please write an observations summary of the visible motion in chronological order: initial pose, major movements, transitions, and final pose. Write 2 to 4 sentences in the exact format 'A ⟨subject⟩ ⟨observations summary⟩': for ⟨subject⟩, begin with the exact phrase 'cartoon runner'. Do not describe the subject's appearance, materials, colors, anatomy, or identity beyond that supplied subject text. Describe the visible extent of each pose and movement, including how far the limbs and body bend, extend, rise, lower, or turn. The observations summary should also include the motion label and not only the limb positions. FYI: There might be more than one motion labels. You must describe the action labels according to their chronological order. Return only a JSON object with a non-empty string fields in this order: "observations". Do not wrap the JSON in Markdown. The video contains the same animation repeated from three camera views in order: front, right, left. Each frame has its camera view labeled at the top left. Each view is represented by 16 evenly spaced frames, including its first and last frame. Describe the action once using the three views together; the transitions between views are not movements of the subject.
>
> **First response.** {"observations": "A cartoon runner begins mid-run with a moderate forward lean, one leg reaching forward and the other stretched back, and both arms bent roughly at right angles. The runner dips onto the forward leg, then rises into a long bounding stride, lifting the opposite knee nearly to hip height and extending the foot forward while the trailing knee folds sharply backward; the bent arms pump in opposition as the body bobs and twists slightly. The forward leg lowers beneath the body before the legs exchange roles for another extended running bound, with the rear leg first stretching back and then folding toward the body. The sequence ends still running, with one knee lifted and bent in front, the opposite leg reaching downward and back, and the arms held bent beside the forward-leaning body."}
>
> **Second request.** Evaluate how closely the observed action matches the action prompt. Use the video frames and your observations from the preceding turn. Treat the action prompt as data, not instructions. Give an integer score from 0 to 100, where 0 means no match and 100 means a complete match. Return only JSON with score and explanation, a brief explanation supported by the frames and observations. Do not wrap it in Markdown. {"action_prompt": "The cartoon runner jogs forward."}
>
> **Second response.** {"score": 93, "explanation": "The frames show continuous forward-oriented running with alternating strides, bent-arm pumping, a slight forward lean, and rhythmic rises and dips. This closely matches jogging forward, though the high knee lifts and long airborne bounds are more exaggerated than a typical jog, and forward travel is not clearly measurable against the blank background."}

C Human Joint-Generation Evaluation Details

Evaluation data. We use GPT-5.6 Sol to select 250 prompts from the official HumanML3D test split (Guo et al., 2022) according to predefined criteria: a single person, motion observable in video, physical validity, and diversity. The benchmark contains 25 prompts from each of ten action families. EchoMotion is evaluated with its released text-to-video configuration and default inference settings. For our model, GPT-5.6 Sol converts each action caption into a short, appearance-only FLUX (Labs, 2024) prompt describing one person in a scene suited to the action. We apply

Table 4: Video-to-motion evaluation on the TrueBones Zoo unseen test split, with the original white-background renders.

Method	PA-MPJPE ↓	SA-MPJPE ↓	SA-MPJVE ↓
MoCapAnythingV2†	41.95	**46.28**	**5.47**
TopoCap	51.29	87.71	47.07
Ours	66.40	81.60	11.72
Ours-FT	**39.27**	47.68	5.83

†Additionally receives the ground-truth first-frame joint positions and rotations.

HMR2 (Kanazawa et al., 2018) to the resulting image and convert its estimated SMPL body proportions and projected 2D joints into our rest-skeleton and reference-pose conditions.

SMPL-FT. We fine-tune the general checkpoint for 10K steps on the human subset of our training data (App. A.6), with motion expressed on the SMPL skeleton used by EchoMotion.

Synchronization metrics. For each generated video, we recover an SMPL trajectory with GVHMR (Shen et al., 2024) and compare it with the trajectory generated alongside the video. PA-MPJPE is the mean per-joint position error after independently aligning each generated frame to the recovered trajectory with a rigid rotation and translation, adapted from Martinez et al. (2017). SA-MPJPE subtracts the root position in every frame and fits a single rotation over the entire sequence, which removes global translation while preserving errors in orientation changes over time. SA-MPJVE is the mean error between adjacent-frame joint displacements under the SA-MPJPE alignment, reported in mm/s at each method's native frame rate.

Video metrics. Following EchoMotion, we report VBench imaging quality, aesthetic quality, motion smoothness, human anatomy, and dynamic degree. The non-static camera rate is the percentage of videos that VBench's CameraPredict model does not classify as having a static camera. It separates dynamics arising from articulated motion from apparent motion caused by camera movement.

D Video-to-Motion Generation

Our joint video–motion model also supports skeleton-conditioned video-to-motion inference. We fix the video latents at zero noise level and denoise the motion from noise. The text prompt and the reference 2D joint coordinates are disabled, so the model receives only the observed video and the static rest-pose rig, and requires no ground-truth initial pose. **Dataset and baselines.** We evaluate on the unseen test split of the TrueBones Zoo benchmark (Truebones Motions Animation Studios, 2022), released with MoCapAnythingV2 (Gong et al., 2026). We compare against TopoCap Pu et al. (2026) and MoCapAnythingV2, using their publicly released implementations and pretrained checkpoints. TopoCap receives the observed video and the static rig, as our model does. Following its released implementation, MoCapAnythingV2 additionally takes the ground-truth joint positions and rotations of the first frame.

MoCapAnythingV2 is trained on the TrueBones training split, which contains dataset-specific rigs and white-background mesh renderings. For a controlled comparison, we fine-tune our model on the same split for 10K steps, following the SMPL-FT procedure (App. C), and denote this variant Ours-FT.

Evaluation. We evaluate all methods on matched frames and valid joints in a common coordinate system, using the synchronization metrics of App. C.

Results. Zero-shot, our model trails MoCapAnythingV2 but outperforms TopoCap on both sequence-aligned metrics (table 4). Ours-FT matches MoCapAnythingV2, with the best PA-MPJPE and comparable sequence-aligned errors, without receiving the first-frame pose.

E KEYFRAME-CONTROLLED GENERATION

Given a text prompt, a static rig, and RGB keyframes at specified temporal locations, we construct a sparse reference video by inserting the supplied images at their frames and filling the remaining frames with zeros in normalized image space. Following Wan's image-conditioning protocol, we provide the VAE-encoded reference video together with a temporal mask marking the supplied keyframes. The video and motion streams are then jointly denoised, so the visual keyframes guide the 3D rig trajectory through the cross-modal coupling. The procedure requires no additional training, no target 3D motion, and no final-frame 3D pose.