Fast 4D Mesh Generation by Spatio-Temporal Attention Chains

Dvir Samuel[1] **Yuval Atzmon**[1] **Gal Chechik**[1,2] **Yoni Kasten**[1]
[1]NVIDIA Research, Tel-Aviv, Israel
[2]Bar-Ilan University, Ramat-Gan, Israel

Abstract

4D mesh generation has recently emerged as a powerful paradigm for recovering dynamic 3D structure from videos, but existing methods remain slow, computationally expensive, and difficult to scale to longer sequences. We introduce a training-free approach that accelerates 4D mesh generation while improving temporal correspondence quality. Our key observation is that temporal correspondences emerge inside a 4D backbone long before its generated meshes become visually accurate. We exploit this with a general framework we call Spatio-Temporal Attention Chain which propagates information across space and time. Starting from vertices on an anchor mesh, the chain maps vertices to latent tokens. It then follows temporal correspondences in latent space, and recovers frame-specific vertices through latent-to-vertex attention. This design avoids expensive explicit matching while preserving anchor mesh details and thereby improving dynamic mesh geometry and temporal consistency. Compared to state-of-the-art, our method generates a 4D mesh in 9 seconds, achieving a $13\times$ speedup while producing higher-quality results, and it transfers without training to other 4D backbones, improving their quality with up to $87\times$ speedup. Moreover, our approach scales to videos up to $16\times$ longer without degrading mesh quality. Beyond generation, the improved correspondences enable competitive zero-shot performance on two downstream tasks: 2D object tracking and 4D tracking. We further show that our framework enables reliable camera estimation, a capability not supported by prior 4D mesh generation methods. Project Page

1 Introduction

Understanding a dynamic 3D world is a central goal of computer vision, and a foundation for embodied AI, physical reasoning, simulation, and virtual reality. Yet progress on dynamic 3D lags far behind images and videos, because high-quality 4D data must capture both 3D structure and motion over time and is therefore rare and expensive to acquire. This motivates recovering 4D from ordinary videos, a far more scalable source of motion and shape [26, 56, 94, 86, 80]. In this work, we focus on *video-to-dynamic-mesh reconstruction*: recovering a temporally coherent 3D mesh sequence from a video of a moving object.

Reconstructing dynamic geometry from monocular video is challenging because the model must infer detailed 3D geometry in every frame while preserving surface identity across time, a requirement that is hard to learn under scarce 4D supervision [8, 42, 58, 92, 59]. ActionMesh [59] resolves this with a staged design. A 4D generative diffusion backbone first lifts the video into 3D latents with frame-specific topology; a separate network then animates an anchor mesh to enforce shared connectivity. While effective, this design is costly. The generative stage requires significant time to produce high-quality geometry; the second stage adds non-end-to-end training; and the full pipeline remains tied to scarce 4D supervision. Beyond these costs, the output mesh lies in an arbitrary coordinate frame with no link back to input pixels, preventing downstream applications like 4D and

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

2D tracking, camera recovery, or scene composition [14, 31, 85, 33, 75]. Finally, training on short clips makes this drift or collapse on longer videos.

That additional stage nevertheless reveals an important fact: the geometry comes from a simpler image-to-3D anchor, while the heavy 4D generative backbone contributes only a 4D motion prior. How can we apply that prior directly to the anchor, without a separate neural animator?

We show that the answer lies inside the 4D generative backbone itself: our key observation is that useful temporal correspondences already emerge when we run denoising with as few as four denoising steps, where attention patterns already link anchor-frame 3D tokens to matching tokens in later frames. We expose this signal through a *Spatio-Temporal Attention Chain*. At a high level, the chain treats *attention as a soft Markov transport*: each attention row is a probability distribution over latent tokens, so multiplying attention maps gives the probability of moving from one representation to the next. Concretely, it links an anchor-frame vertex to an anchor-frame latent token ($V_a \to Z_a$), transports it across time to frame f via temporal self-attention ($Z_a \to Z_f$), and projects it back to a 3D point in the target frame ($Z_f \to V_f$).

This shifts the role of the backbone: instead of producing a high-resolution mesh per frame with 30 denoising steps, we run the denoiser with four steps and read the correspondence field it already computes. By tracking sparse landmarks through the chain and lifting them to the anchor mesh via geodesic-rigid skinning, we drop the second stage entirely. Generation time therefore falls from nearly two minutes to roughly nine seconds, preserving topology by construction and improving 3D accuracy. Beyond speed, the chain naturally connects 2D patches, latent tokens, and mesh vertices, unlocking three additional capabilities. First, when extending generation from 16 to 240 frames autoregressively, reinforcing the strongest correspondences during denoising substantially reduces geometric drift without retraining. Second, different chain compositions yield competitive zero-shot 3D point trajectories (4D tracking) and 2D point trajectories. Third, the resulting 2D-to-3D matches allow recovering per-frame cameras, placing the mesh back into a reconstructed scene (Fig. 4).

Contributions. (1) We identify spatio-temporal attention chains as a hidden correspondence signal linking pixels, latent tokens, and mesh vertices inside a 4D generative backbone. (2) We propose a training-free framework for 4D generative backbones that tracks sparse landmarks through these chains and lifts them to a full animated mesh, cutting inference from 120s to 9s at favorable quality; the same framework transfers unchanged to Mesh4D [27] and Sculpt4D [93], improving their quality with up to $87\times$ speedup. (3) We improve autoregressive generation by reinforcing these correspondences to reduce drift, enabling coherent $16\times$ longer sequences. (4) We show that the same chains yield several capabilities missing in prior work: competitive zero-shot 4D and 2D point tracking, as well as camera recovery from 2D-3D matches.

2 Related Work

Image-to-3D generative backbones. Our chain reads attention weights off a VecSet-style 3D decoder [98], which reconstructs geometry by cross-attending 3D query points to a compact set of latent tokens encoding the shape. We instantiate on TripoSG [41], a flow-based image-to-3D generator producing high-fidelity meshes from a single image. The same decoder structure underlies CLAY [102], Craftsman [38], Dora-VAE [9], and Hunyuan3D [70], and our chain transfers to generators sharing this structure (Sec. 5.2). Other representation classes include Trellis's [83] sparse structured latents (SLAT), LRM's [23] triplanes, LGM's [68] Gaussian primitives, and AssetGen's [63] PBR-textured meshes.

Video-to-4D generation. Per-scene optimization pipelines [26, 56, 94, 81, 96, 79, 100, 44] distill dynamic 3D from video using diffusion priors, taking minutes to hours per clip. Multi-view video diffusion [86, 89, 80, 25] generates feed-forward novel-view sequences but still needs per-scene optimization for 4D. Feed-forward 4D methods predict spatial primitives directly, typically in topology-free spaces: L4GM [57] and 4DGT [88] produce Gaussian sequences, Motion2VecSets [6] denoises vector sets, and ShapeGen4D [92] adds temporal attention to a 3D generator; all decode frames independently, lacking shared topology. Prior methods therefore add an explicit topology stage: ActionMesh [59] learns a temporal 3D autoencoder that deforms a reference mesh via per-frame anchor displacements, $\mathbf{V}_f = \mathbf{V}_a + \Delta_f$, while optimization-based methods [8, 42, 58] impose topology or temporal consistency through registration, deformation, or optimized implicit representations. In contrast, our approach extracts dense correspondences directly from the temporal

backbone, bypassing any separate topology-enforcing or animation stage. A related line animates a given mesh via predicted skeletons [19, 64, 46, 90] or deformation fields [50, 61, 82]; these methods assume clean input assets and predict explicit skeletons and skinning weights.

Emergent Correspondences in Diffusion Features: Two recent lines tap frozen diffusion models for zero-shot correspondence. One matches features as descriptors: DIFT [69] on UNet activations, Diff3F [15] lifting them onto 3D shapes, MbQ [2] on video-DiT queries for Q-injection, and Track4Gen [24] via an auxiliary tracking loss. The other reads attention weights directly: CAMEO [34] in multi-view 3D attention, DiTFlow [53] as a per-clip optimization loss, and DiffTrack [51] on video-DiT temporal-matching layers; Point Prompting [62] sidesteps both via counterfactual prompting. We instead *compose* three attention maps of a frozen 4D generator – vertex-to-token, temporal token-to-token, and token-to-surface – into a $V_a \to Z_a \to Z_f \to V_f$ chain yielding correspondences tied to an anchor mesh's surface through a forward pass – no optimization, no external tracker.

Attention Control in Diffusion Models: A complementary line manipulates or analyzes frozen attention. Several methods reweight cross/self-attention [22, 60] for editing, inject self-attention features [73] for structure, or share self-attention across images [5, 72] for identity; TiARA [40] suppresses temporal attention weights for extended video generation. A related thread treats attention rows as probability distributions and composes them to trace information flow within a single transformer [1, 7, 16]. Building on this view, we compose attention across separately-trained modules and modalities of a 4D pipeline – vertex-to-token, token-to-token, token-to-surface – and reinforce reliable matches to stabilize long sequences.

Point tracking and monocular 4D geometry. Supervised 2D trackers [20, 14, 13, 31, 29, 11, 36, 21] are driven by standard benchmarks [12, 104]; 3D trackers [85, 84, 97] update point clouds, while 4RC [49], Trace-Anything [47], and TracksTo4D [33] predict motion fields and MegaSaM [43] runs deep visual SLAM. A separate line predicts metric pointmaps, introduced by DUSt3R [78] and extended for dynamic scenes [28, 17, 101, 77, 87]; closest in spirit, Easi3R [10] achieves 4D reconstruction via training-free attention adaptation of DUSt3R. These methods either require per-frame mesh reconstruction from scattered points or depend on pointmap supervision. In contrast, our approach needs no tracker or pointmap supervision, and directly outputs the skinned mesh alongside the 2D-3D matches needed for PnP+RANSAC [37, 4] camera pose estimation.

3 4D Mesh Generation: Preliminaries and Notations

Video-to-dynamic-mesh methods map a video of F frames, to a temporally coherent 4D mesh sequence $\mathcal{M}_{1:F} = \{(\mathbf{V}_f, \mathcal{F})\}_{f=1}^{F}$, where $\mathcal{F}$ and $\mathbf{V}_f \in \mathbb{R}^{|V|\times 3}$ define a fixed topology and shared vertex identities across time.

We denote attention between a query sequence $\mathbf{X}$ and context sequence $\mathbf{Y}$ by:

$$A_{\mathbf{X}\to\mathbf{Y}} = \textit{softmax}((\mathbf{X}\mathbf{W}_Q)(\mathbf{Y}\mathbf{W}_K)^\top/\sqrt{d_k}), \quad \mathrm{Attn}(\mathbf{X},\mathbf{Y}) = A_{\mathbf{X}\to\mathbf{Y}}\cdot(\mathbf{Y}\mathbf{W}_V) \tag{1}$$

where $\mathbf{W}_Q$, $\mathbf{W}_K$, and $\mathbf{W}_V$ project inputs to queries, keys, and values.

Recent pipelines [59, 8, 27] employ a three-staged approach: (0) an image-to-3D model reconstructs initial reference geometry, (I) temporal or independent generators produce per-frame 3D representations, and (II) a final topology-preserving stage aligns the vertices through time so all frames share the same connectivity.

Stage 0: Image-to-3D anchor reconstruction. An image-to-3D diffusion model (TripoSG [41] in our experiments) does not represent a shape directly as mesh vertices. Instead, it denoises a VecSet-style [98] shape latent $\mathbf{z}_a \in \mathbb{R}^{N\times d}$: an unordered set of N tokens of dimension d (N=2048 for TripoSG). Geometry is recovered by the VAE's transformer decoder: given a 3D query point, the decoder cross-attends the point to the N tokens and predicts an implicit field value (signed distance); evaluating this field over 3D space and extracting its zero level set yields the anchor mesh $\mathcal{M}_a = (\mathbf{V}_a, \mathcal{F})$. The decoder's cross-attention weights describe how each query point combines information from the tokens; querying the anchor vertices yields $A_{V_a\to Z_a} \in \mathbb{R}^{|V_a|\times N}$, which expresses each anchor vertex as an attention-weighted combination of latent tokens. Image cross-attention similarly links anchor image patches P_a to the same tokens, giving $A_{P_a\to Z_a}$.

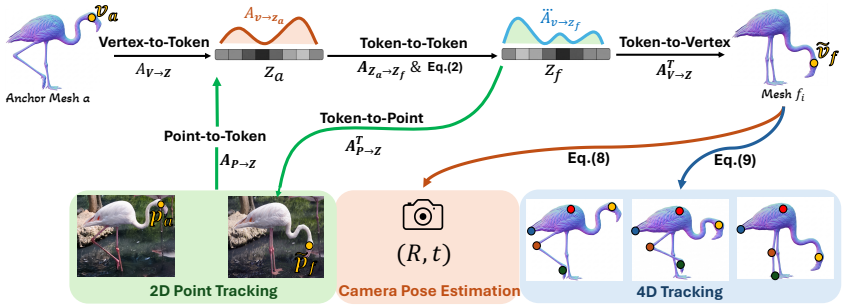


Figure 1: **Method overview.** Our attention chain follows a point through the frozen 4D generator: from an anchor mesh vertex to latent tokens, across time to target-frame tokens, and back to a target mesh vertex. Image-patch endpoints give analogous chains for 2D tracking, camera pose estimation, and 4D tracking, without additional training.

Stage I: Video-to-4D mesh generation. Given the anchor latent $\mathbf{z}_a$ and the input video, a temporal denoiser Φ_θ predicts one latent $\mathbf{z}_f$ for each frame f. Inflated self-attention inside Φ_θ links tokens in $\mathbf{z}_a$ to tokens in $\mathbf{z}_f$, giving $A_{Z_a \to Z_f} \in \mathbb{R}^{N \times N}$.

Stage II: Topology-consistent decoding. Prior pipelines add a topology-preserving stage that predicts per-frame displacements with learned decoders [27, 59] or test-time optimization [8, 42]. We drop this stage, recovering anchor motion directly from attention-chain correspondences.

4 Method

Current staged pipelines [59, 8] treat geometry generation and animation as separate tasks. However, a dedicated Stage II requires an entirely separate network, adding significant overhead during both training and inference, and these pipelines typically remain restricted to short, drift-prone temporal windows (Fig. 2). We aim to accelerate and scale topology-preserving 4D generation without any additional training. Our core observation is that a frozen pipeline like ActionMesh [59] inherently encodes temporal tracking within its features. Instead of relying on a learned decoder, we extract 3D correspondences directly during the denoising process (stages 0 and I) via an *attention chain*. Conceptually, each attention matrix is a soft transition map, and multiplying them transports probability mass from anchor vertices (V_a), through anchor latent tokens (Z_a) and, via temporal self-attention, frame-f tokens (Z_f), to target surface points V_f:

$$V_a \to Z_a \to Z_f \to V_f$$

These correspondences emerge within just a few denoising steps. We then animate the anchor mesh using a fast closed-form deformation model. By strictly reusing the constant face set $\mathcal{F}$, we guarantee perfect topology consistency by construction.

4.1 Correspondence from the Attention Chain

For each anchor vertex v and target frame f, we seek a target surface point $\tilde{v}_f$. Instead of training a separate deformation network, we establish tracking with an attention chain (Fig. 1). The chain links the anchor and target geometries through intermediate representations by sequentially multiplying the backbone's internal spatial and temporal attention maps. This composes localized attention steps into a dense correspondence map. We assemble the chain from three components.

(1) Vertex-to-token attention ($V_a \to Z_a$). During Stage 0 (Sec. 3), the 3D decoder yields the cross-attention matrix $A_{V_a \to Z_a} \in \mathbb{R}^{|V_a| \times N}$. Since the softmax normalization is applied over the latent key dimension, each row $A_{V_a \to Z_a}[v, :]$ forms a valid probability distribution. This row explicitly describes which latent tokens explain anchor vertex v, where each entry represents the probability that anchor token t relates to that specific vertex.

(2) Token-to-token temporal attention ($Z_a \to Z_f$). During the denoising step of Stage I, the inflated temporal self-attention layers process all frames simultaneously. For a given target frame f, we extract

the attention weights linking anchor-frame tokens to frame-f tokens to yield $A_{Z_a \to Z_f} \in \mathbb{R}^{N \times N}$. We do not select a particular layer: we use *all* inflated temporal self-attention layers of Φ_θ and average the attention weights over heads and layers, which we found more stable than any single layer. This matrix governs the transfer of structural information from the anchor frame to the target frame at the latent token level.

(3) Token-to-surface attention ($Z_f \to V_f$). For target frame f, the 3D decoder turns Z_f into an implicit field. We evaluate this field on a fixed 128^3 grid and retain the grid points that lie near the predicted surface. These points form the set of *candidate surface points* $S_f = \{x_u^{(f)}\}_{u=1}^{|V_f|}$: the possible target locations of an anchor vertex in frame f. We call them candidates because no correspondence has been assigned to them yet. Querying them against the frame-f latent tokens yields the cross-attention matrix $A^T_{Z_f \to V_f} \in \mathbb{R}^{|V_f| \times N}$, which relates each candidate surface point u to these tokens.

Composing the attention chain. We compose the attention matrices above to map anchor vertex v to frame f. The row $A_{V_a \to Z_a}[v, :]$ gives weights over anchor tokens. Multiplying this row by $A_{Z_a \to Z_f}$ transfers those weights to frame-f tokens $t' \in Z_f$:

$$\dddot{A}_{v,Z_f}(t') = \sum_{t=1}^{N} A_{V_a \to Z_a}[v, t]\, A_{Z_a \to Z_f}[t, t'], \tag{2}$$

where t indexes anchor-frame tokens and t' indexes tokens in frame f. The vector $\dddot{A}_{v,Z_f}$ is therefore a probability distribution over frame-f tokens. A candidate surface point $x_u^{(f)} \in S_f$ is likely to match v when its token-level attention agrees with $\dddot{A}_{v,Z_f}$, so we score it by their inner product:

$$s_{v,f}(u) = \sum_{t'=1}^{N} \dddot{A}_{v,Z_f}(t')\, A^T_{Z_f \to V_f}[u, t']. \tag{3}$$

Finally, we obtain the correspondence $\tilde{v}_f$ as a sharp softmax blend over the top-scoring surface points:

$$\tilde{v}_f = \sum_{u \in \mathcal{N}_{v,f}} \pi_{v,f}(u)\, x_u^{(f)}, \qquad \pi_{v,f}(u) = \frac{\exp(s_{v,f}(u)/\tau)}{\sum_{q \in \mathcal{N}_{v,f}} \exp(s_{v,f}(q)/\tau)}. \tag{4}$$

Here $\mathcal{N}_{v,f} \subset S_f$ is the set of the K highest-scoring candidates for vertex v (K=3 in all experiments) and τ is a temperature hyperparameter; rather than selecting only the winner, Eq. (4) converts the top-K scores into weights and returns the weighted average of their 3D positions. We also define a confidence score $c_v^{(f)} = \max_u s_{v,f}(u)$ to be used later for mesh animation.

Step-by-step summary. For an anchor vertex v and frame f: (i) read row v of $A_{V_a \to Z_a}$ (which anchor tokens explain v); (ii) transport it to frame-f tokens with $A_{Z_a \to Z_f}$ (Eq. 2); (iii) score every candidate in S_f by how well its own token distribution agrees with the transported one (Eq. 3); (iv) blend the K best candidates into $\tilde{v}_f$ (Eq. 4) and record $c_v^{(f)}$. All attention maps are read off the frozen backbone during its normal execution.

This construction has two key properties. First, both endpoint attentions come from the same 3D decoder, so anchor vertices and target surface points are compared in a shared token–geometry space. Second, $\tilde{v}_f$ is computed only from the top-scoring surface samples $\mathcal{N}_{v,f}$, keeping the correspondence on the target surface and reducing drift to unrelated regions.

4.2 Topology-Preserving Animation

In early experiments, we observed that directly querying all anchor vertices and simply mapping them to their target positions using our dense correspondences produced noisy results. Instead, we obtain topology-preserving animation by tracking a sparse set of control landmarks and lifting their motion to the full mesh in three steps:

1. Landmark Extraction and Filtering: We sample a sparse set of L control landmarks on the anchor mesh by farthest point sampling. We extract their trajectories across frames via the attention chain, assigning confidence scores and rejecting physically implausible displacements as outliers. The outlier threshold is relative to the median of the *non-negligible* landmark displacements in that

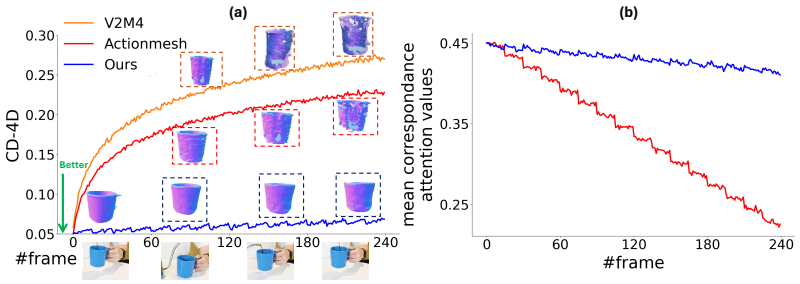


Figure 2: **Long-sequence rollout.** (a) Naive autoregressive 4D generation accumulates errors over time, degrading mesh quality. We find that this drift is driven by weakening latent correspondences across windows. (b) Our correspondence reinforcement preserves these correlations, stabilizing the rollout and maintaining high-quality generation.

frame (displacements below a configurable motion floor are excluded), which avoids a degenerate threshold for mostly static objects with one fast-moving part, e.g., a humanoid moving only one hand. A flagged landmark receives zero confidence for that frame only and is interpolated from neighboring frames in the next step; it is discarded altogether only if flagged repeatedly across the sequence (Appendix I).

2. Temporal Smoothing: To ensure fluid motion, we apply a confidence-weighted 1D Gaussian temporal smoothing to each landmark's trajectory independently. This bridges gaps caused by outlier removal by interpolating each landmark from nearby reliable frames.

3. Mesh Deformation: Finally, we propagate the smoothed landmark motions to the dense mesh using Geodesic Rigid Skinning [67]. For each vertex, we compute a local rigid transformation (rotation and translation) from its closest landmarks under geodesic (on-surface) distance. This prevents motion from leaking between spatially close but disconnected parts, such as an arm and torso, while the local-rigid transform preserves volume and avoids the shrinkage artifacts of linear blend skinning. This pipeline yields a temporally coherent animated mesh $\hat{\mathcal{M}}_f = (\hat{\mathbf{V}}_f, \mathcal{F}_a)$ that strictly maintains the anchor topology. Full details of temporal smoothing and the weighted Procrustes skinning formulation are deferred to Appendix I.

4.3 Scaling to Longer Sequences

Existing 4D generators are trained on short clips, so autoregressive rollout to longer videos quickly drifts: each new 16-frame window is anchored at the final frame of the previous one, and errors accumulate. On long ActionBench [59] sequences (Appendix C) we observe both degrading mesh quality (Fig. 2a) and a steady drop in the correlation of matched latent points across windows (Fig. 2b).

To prevent this drift, we reinforce reliable temporal correspondences from one window to the next (Algorithm 1, Appendix D). After denoising a window, we run the chain for the control landmarks (Sec. 4.1) and keep the matches with confidence $c_v^{(f)} \geq \tau_c$. Each kept match is traced backward to the token pair most responsible for it: the top anchor token $t^\star = \arg\max_t A_{V_a \to Z_a}[v, t]$ and the frame-f token $t'^\star$ maximizing $A_{v,Z_f}(t')\, A^T_{Z_f \to V_f}[u^\star, t']$, where $u^\star$ is the matched surface point. The pairs are collected in $\mathcal{C}_f$ with accumulated weights, normalized to confidences $c^{(f)}_{t,t'} \in (0, 1]$. When denoising the next window, we gate the temporal attention toward these pairs with strength λ and renormalize each row:

$$\tilde{A}_{Z_a \to Z_f}[t, t'] \;\propto\; \left(1 + \lambda\, c^{(f)}_{t,t'}\, \mathbf{1}[(t, t') \in \mathcal{C}_f]\right) A_{Z_a \to Z_f}[t, t']. \tag{5}$$

This stabilizes latent correlations and mesh quality over long sequences (Fig. 2b).

4.4 Extension to 2D and 4D Point Tracking

Beyond mesh animation, attention chaining is a general composition mechanism: any two attention maps sharing an intermediate representation can be linked, e.g., image patches to tokens, tokens to

tokens, and tokens to vertices. We demonstrate this on 2D point tracking in the input video and on 4D point tracking, which recovers world-coordinate 3D trajectories for every visible pixel.

2D point tracking ($P_a \to Z_a \to Z_f \to P_f$). We replace the 3D decoder attention with the denoiser cross-attention between latent tokens and image patches. Let $A_{P_f \to Z_f} \in \mathbb{R}^{P \times N}$ denote the attention from P image patches to N latent tokens in frame f. Given a query patch p_a in the anchor frame, we transport its attention through the temporal block to find its correspondence $\tilde{p}_f$ in frame f:

$$\tilde{p}_f = \arg\max_{p} \sum_{t,t'=1}^{N} A^T_{P_a \to Z_a}[t, p_a]\, A_{Z_a \to Z_f}[t, t']\, A^T_{P_f \to Z_f}[t', p] \tag{6}$$

This reuses the temporal attention of Section 4.1 without any additional computation.

2D to 3D bridge ($P \leftrightarrow V$). By composing anchor-frame attentions, we directly link image patches and mesh vertices. Notably, ActionMesh [59] lacks this capability, leaving no direct way to map their output mesh back to the source image. In contrast, for a given anchor patch p_a or vertex v_a, we establish the following mappings :

$$\tilde{v}_p = \arg\max_{v} \sum_{t=1}^{N} A_{P_a \to Z_a}[p_a, t]\, A_{Z_a \to V_a}[t, v], \quad \tilde{p}_v = \arg\max_{p} \sum_{t=1}^{N} A_{P_a \to Z_a}[p, t]\, A_{Z_a \to V_a}[t, v_a] \tag{7}$$

Camera pose estimation. Using this bridge, we collect 2D to 3D correspondences $\{(\mathbf{u}_v, \mathbf{V}_a[v])\}$ between anchor-frame pixel coordinates $\mathbf{u}_v$ and mesh vertices. Given camera intrinsics K, we estimate the camera pose (R, t) via robust PnP [71, 37, 4]:

$$(R^\star, t^\star) = \arg\min_{R,t} \sum_{v} \rho_v \left(\| \pi_K(R\, \mathbf{V}_a[v] + t) - \mathbf{u}_v \|_2 \right) \tag{8}$$

Here π_K is the perspective projection and ρ_v is a robust weighting factor for each correspondence.

4D point tracking. Finally, we lift pixels to 3D by intersecting anchor-frame rays with the anchor mesh and tracking their barycentric coordinates over time. Because the animated mesh vertices $\{\hat{\mathbf{V}}^{(f)}\}_{f=1}^{F}$ from Section 4.2 reside in a canonical object space, we apply our estimated camera pose to align them with the input video. For a pixel $\mathbf{u}$ with barycentric weights $\mathbf{w}$ on face ϕ, its 3D trajectory in the anchor camera coordinates at frame f is calculated as:

$$\mathbf{X}^{(f)}_{\mathbf{u}} = R^\star \sum_{i=0}^{2} w_i\, \hat{\mathbf{V}}^{(f)}_{\mathcal{F}[\phi,i]} + t^\star \tag{9}$$

This yields dense 3D trajectories in the observer's frame for all foreground pixels.

5 Experiments

We evaluate our approach in three complementary settings that exercise the full attention chain of Sec. 4: **(1)** *4D mesh generation* (Secs. 5.1–5.2), **(2)** *2D point tracking* on dynamic objects (Sec. 5.3), and **(3)** world-coordinate *4D point tracking* (Sec. 5.4). Unless stated otherwise, we build on ActionMesh [59] as our base model (Sec. 3). For 2D and 4D tracking videos longer than 16 frames, we use the autoregressive scaling of Sec. 4.3. All experiments ran on a single H100 GPU.

5.1 4D Mesh Generation

Datasets and metrics. Following [59], we evaluate on *ActionBench* [59] and *Consistent4D* [26]. ActionBench contains 16-frame clips with ground-truth 4D meshes, while Consistent4D is used to evaluate out-of-distribution rendering quality. For geometry, we report end-to-end generation *Time*, *CD-3D* (per-frame), *CD-4D* (full 4D point cloud), *CD-M* (motion-only), and *Normal Consistency*. All predictions are aligned to ground truth with ICP before metric computation; results without ICP are provided in the supplement. For rendering, we report *LPIPS* [103], *CLIP* [55], and *DreamSim* [18] between rendered and ground-truth views.

Baselines. Following [59], we compare against state-of-the-art video-to-4D methods: Step1X-3D [39], L4GM [57], GVFD [99], LIM [58], DreamMesh4D (DM4D) [42], V2M4 [8], ShapeGen4D

Table 1: **Left**: ActionBench results. Our training-free approach is the fastest and achieves SoTA CD-3D, CD-4D, and Normal Consistency. **Right**: Consistent4D rendering results; with camera pose estimation (*Ours + CPE*) we achieve the best LPIPS, CLIP, and DreamSim.

ActionBench	Time (s)	CD-3D ↓	CD-4D ↓	CD-M ↓	Normal Consist. ↑
TRELLIS [83]	900	0.065	0.181	–	–
TripoSG [41]	120	0.056	0.184	–	–
DM4D [42]	2100	0.104	0.152	0.265	–
LIM [58]	900	0.089	0.126	0.243	–
V2M4 [8]	2100	0.068	0.340	0.616	–
SG4D [92]	900	0.060	0.170	0.348	0.91
ActionMesh [59]	120	0.053	0.081	**0.148**	0.85
Ours	**9**	**0.048**	**0.077**	0.163	**0.97**

Consistent4D	Aligned	LPIPS ↓	CLIP ↑	Dream Sim ↓
Step1X-3D [39]	✗	0.1524	0.9040	0.1106
L4GM [57]	✓	0.0988	0.9397	0.0487
GVFD [99]	✗	0.1691	0.8601	0.1467
SG4D [92]	✗	0.1359	0.9009	0.0966
ActionMesh [59]	✗	0.1458	0.9012	0.0939
Ours	✗	0.1423	0.9112	0.0823
Ours + CPE	✓	**0.0823**	**0.9468**	**0.0319**

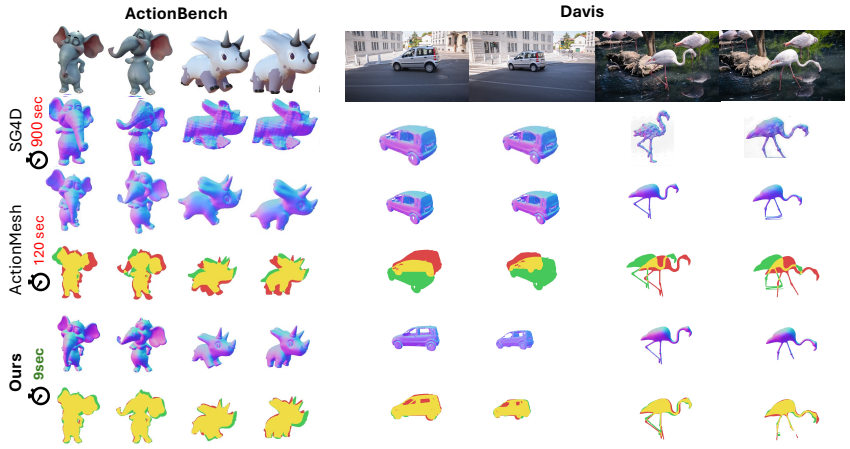


Figure 3: **4D Mesh Generation.** Our method produces sharp, temporally consistent meshes, aligns them to the input camera, and runs in **only** 9 **sec**, compared to 120–900 sec for prior methods. Unlike SG4D and ActionMesh, which generate object-centric meshes, our spatial attention-chain correspondences enable camera recovery and world placement, yielding high foreground overlap (yellow) and fewer mismatch regions (red/green).

(SG4D) [92], and ActionMesh [59]. We also include image-to-3D models applied independently per frame: TripoSG [41] and TRELLIS [83].

Quantitative results. Table 1(left) shows that our method achieves the best performance on three of four geometric ActionBench metrics: CD-3D (0.048), CD-4D (0.077), and Normal Consistency (0.97), while being over an order of magnitude faster: **9** s per 16-frame clip vs. 2 min for ActionMesh [59] ($\sim 14\times$) and 15 min for ShapeGen4D [92] ($\sim 100\times$). ActionMesh is slightly better on CD-M (0.148 vs. 0.163). On Consistent4D (Table 1 (right)), our method outperforms all non-aligned baselines, and with camera-pose estimation (*Ours + CPE*) it surpasses the aligned baseline L4GM [57] on all metrics.

User study. In 2,000 pairwise comparisons against ActionMesh [59] (100 raters judging appearance and motion consistency), ours is preferred in 75% of cases (Appendix I.3).

Qualitative results. Fig. 3 shows that our method produces sharp, temporally consistent meshes while also uniquely aligning them to the input camera: unlike the object-centric meshes of ActionMesh and ShapeGen4D, with clear silhouette mismatch, our dense 2D↔3D matches enable PnP camera recovery and accurate image alignment (yellow regions).

Genuine long sequences. The 240-frame rollouts of Sec. 4.3 loop 16-frame ActionBench clips, which preserves motion continuity artificially (Appendix C). We therefore also evaluate on the MIT mesh-animation dataset [74], whose ground-truth mesh sequences span up to 250 frames of natural motion (Appendix F). Our method remains stable throughout: from the first window to frame 250,

Table 2: **(a,b) Generalization to other backbones.** Applied without training to Sculpt4D [93] and Mesh4D [27] (each paper's protocol, single H100; Appendix E), our chain improves every metric while reducing runtime. **(c) Component ablation** on ActionBench: the attention chain alone closes $\sim 60\%$ of the gap between the static anchor and the full method.

(a) Sculpt4D protocol (16-frame videos)

	Chamfer ↓	IoU ↑	F-Score ↑	Time (s) ↓
Sculpt4D [93]	0.1052	0.3381	0.3137	960
Ours	**0.1032**	**0.3457**	**0.3201**	**11** ($\sim 87\times$)

(b) Mesh4D protocol (6-frame videos)

	Chamfer ↓	IoU ↑	PS2 ↓	Time (s) ↓
Mesh4D [27]	0.0286	0.365	0.0296	30
Ours	**0.0251**	**0.371**	**0.0273**	**16** ($\sim 1.9\times$)

(c) Component ablation (ActionBench)

Configuration	CD-3D ↓	CD-4D ↓
Static anchor	0.160	0.300
Attention chain only	0.092	0.160
+ sparse landmarks	0.059	0.096
+ outlier filtering	0.055	0.089
+ temporal smoothing	0.053	0.084
+ geodesic-rigid skinning	0.050	0.080
Full method	**0.048**	**0.077**

CD-3D/CD-4D rise only from $0.051/0.082$ to $0.063/0.104$, versus $0.053/0.086 \rightarrow 0.121/0.223$ for a naive ActionMesh rollout.

Attention chain vs. post-processing. Table 2(c) isolates each component of Sec. 4. The attention chain supplies the motion signal: applied alone, mapping every anchor vertex to its chain correspondence with no post-processing, it already closes $\sim 60\%$ of the gap between the static anchor and the full method. The remaining steps denoise this signal and transfer it to a topology-consistent mesh: sparse landmarks give the largest additional gain by removing per-vertex discretization noise, while outlier filtering, temporal smoothing, and geodesic-rigid skinning add smaller improvements, mostly on fast or noisy sequences.

5.2 Generalization to Other 4D Backbones

Our method targets staged pipelines whose Stage I is a temporal diffusion model (Sec. 3). ActionMesh was the only such method with public code at submission time; since then, Mesh4D [27] and Sculpt4D [93] have been released. We apply our attention chain to both *without any modification or training*, following each paper's protocol and sequence length (Appendix E). Although these backbones differ from ActionMesh in architecture, latent representation, and VAE design, and hence in their exact attention patterns, they preserve the correspondence structure our chain relies on, and Table 2(a,b) shows consistent gains in both quality and efficiency. On Sculpt4D, replacing its costly per-frame mesh generation with the attention-chain readout cuts generation time from 16 minutes to 11 seconds ($\sim 87\times$); on Mesh4D, where runtime is dominated by the unavoidable anchor-mesh generation, we still obtain a $\sim 1.9\times$ speedup. In both cases all quality metrics improve, indicating that the attention-chain principle transfers across 4D architectures and latent representations rather than being specific to ActionMesh.

5.3 2D Point Tracking

Datasets and metrics. Our pipeline follows ActionMesh [59] in reconstructing a single dynamic object rather than the full scene, so we evaluate 2D tracking on foreground object points only. We use *BADJA* [3] (articulated-animal joint tracking) and a foreground-only version of *TAP-Vid-DAVIS* [12, 54]. Following [12], we report Average Jaccard (*AJ*), average position accuracy ($\langle \delta \rangle_{\text{avg}}$), and Occlusion Accuracy (*OA*). On BADJA, we report segmentation accuracy on joint trajectories (*segA*) and the fraction of predictions within 3 px of ground truth ($\delta^{3\text{px}}$).

Baselines. We compare with *2D-supervised* trackers (BootsTAP [13], TAPIR [14], CoTracker [31], TAP-Net [12], PIPs [20], OmniMotion [76], CowTracker [35]), the *3D-aware* SpatialTracker [85], which lifts 2D tracks into 3D, and *zero-shot* diffusion-feature trackers DiffTrack [51] and Denoise-to-Track [95].

Results. Table 3(a) shows that our method is the strongest zero-shot tracker on DAVIS-foreground, outperforming DiffTrack and Denoise-to-Track on all metrics. It also achieves the best OA overall (90.41), while remaining competitive with supervised trackers. Table 3(b), on the BADJA benchmark, our zero-shot method remains close to supervised trackers, outperforming several of them despite using no tracking supervision.

Table 3: **Point tracking results.** (a) and (b): 2D point tracking on DAVIS-foreground and BADJA. (c): 4D point tracking on PointOdyssey (PO) and Dynamic Replica (DR).

(a) 2D Tracking (DAVIS)

	AJ	$\langle\delta\rangle_{\text{avg}}$	OA
Supervised			
BootsTAP [13]	56.64	69.99	87.88
SpatialTracker [85]	54.97	71.51	85.64
CowTracker [35]	**60.08**	**73.53**	88.64
Zero-shot			
DiffTrack [51]	26.44	43.01	72.10
DenoiseToTrack [95]	35.15	50.12	75.16
Ours	53.33	66.34	**90.41**

(b) 2D Tracking (BADJA)

	segA	$\delta^{3\text{px}}$
Supervised		
TAP-Net [12]	54.4	6.3
PIPs [20]	61.9	13.5
TAPIR [14]	66.9	15.2
OmniMotion [76]	57.2	13.2
CoTracker [31]	63.6	**18.0**
SpatialTracker [85]	**69.2**	17.1
Zero-shot		
Ours	64.8	16.3

(c) 4D Tracking

	PO	DR
Supervised		
St4RTrack [17]	72.04	76.82
TraceAny. [47]	47.02	61.19
Any4D [32]	64.25	70.33
4RC [49]	66.92	76.11
V-DPM [66]	**82.12**	**78.18**
Zero-shot		
ActionMesh (II) [59]	31.5	41.6
Ours	59.9	65.3

Physical vs. semantic correspondence. These evaluations are strictly position-based, with tight thresholds (e.g., 3 px on BADJA) that coarse semantic or part-level matching cannot satisfy, and the 2D tracks are read directly from the raw attention chain (Eq. 6) without the filtering, smoothing, or skinning of Sec. 4.2. The chain therefore captures material-point correspondences, not merely semantic similarity; in Fig. 7, points stay attached to specific surface locations rather than following the same body part.

5.4 4D Point Tracking

Datasets and metrics. We follow the dynamic-only protocol of the WorldTrack benchmark [17] and evaluate on *PointOdyssey (PO)* [104] and *Dynamic Replica (DR)* [30]. We report $APD_{\mathbf{3D}}$, the percentage of predictions within a set of 3D distance thresholds, averaged over thresholds after global median alignment.

Baselines. We compare against ActionMesh Stage II [59] as a zero-shot baseline for per-frame mesh tracking, and five supervised 4D trackers: V-DPM [66], 4RC [49], Any4D [32], TraceAnything [47], and St4RTrack [17].

Results. Tab. 3(c) shows that our method substantially improves over the zero-shot ActionMesh Stage II baseline on both datasets (+28.4 APD_{3D} on PO and +23.7 on DR): attention-chain correspondences, together with camera estimation, provide a much stronger signal for world-coordinate 4D tracking. Despite using no 4D tracking supervision, our method is competitive with supervised trackers: it outperforms TraceAnything and approaches Any4D and 4RC, closing a large part of the gap without task-specific training.

6 Conclusion

We introduced a training-free framework for fast video-to-4D-mesh reconstruction. By exposing spatio-temporal attention chains inside a frozen 4D backbone, we recover correspondences that animate an anchor mesh without a learned deformation network, yielding *topology-consistent meshes*, *faster inference*, *longer rollouts*, and *2D/4D tracking* with *camera recovery*, and transferring unchanged to other staged 4D backbones.

Limitations. Mesh quality is bounded by the frozen image-to-3D model and denoiser, and an anchor artifact that the denoiser preserves is faithfully tracked by the chain (Appendix G). Intrinsic to our design, sparse landmarks with locally rigid skinning can damp subtle non-rigid motion; the fixed anchor connectivity rules out topology changes and may propagate motion across erroneously connected parts; and over long rollouts errors still accumulate as attention to generated anchors becomes diffuse. Structured latents (e.g., TRELLIS [83]) could simplify the chain's endpoint steps (Appendix H).

Acknowledgments and Disclosure of Funding

This work was supported solely by the authors' primary affiliations. The authors declare no competing interests.

References

[1] S. Abnar and W. Zuidema. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4197. Association for Computational Linguistics, July 2020.

[2] Y. Atzmon, R. Gal, Y. Tewel, Y. Kasten, and G. Chechik. Identity-motion trade-offs in text-to-video generation. In *36th British Machine Vision Conference 2025, BMVC*, 2025.

[3] B. Biggs, T. Roddick, A. Fitzgibbon, and R. Cipolla. Creatures great and SMAL: Recovering the shape and motion of animals from video. In *ACCV*, 2018.

[4] R. C. Bolles and M. A. Fischler. A ransac-based approach to model fitting and its application to finding cylinders in range data. In *Ijcai*, volume 1981, pages 637–643, 1981.

[5] M. Cao, X. Wang, Z. Qi, Y. Shan, X. Qie, and Y. Zheng. Masactrl: Tuning-free mutual self-attention control for consistent image synthesis and editing. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 22560–22570, 2023.

[6] W. Cao, C. Luo, B. Zhang, M. Nießner, and J. Tang. Motion2vecsets: 4d latent vector set diffusion for non-rigid shape reconstruction and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20496–20506, 2024.

[7] H. Chefer, S. Gur, and L. Wolf. Generic attention-model explainability for interpreting bi-modal and encoder-decoder transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 397–406, October 2021.

[8] J. Chen, B. Zhang, X. Tang, and P. Wonka. V2m4: 4d mesh animation reconstruction from a single monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11643–11653, 2025.

[9] R. Chen, J. Zhang, Y. Liang, G. Luo, W. Li, J. Liu, X. Li, X. Long, J. Feng, and P. Tan. Dora: Sampling and benchmarking for 3d shape variational auto-encoders. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 16251–16261, June 2025.

[10] X. Chen, Y. Chen, Y. Xiu, A. Geiger, and A. Chen. Easi3r: Estimating disentangled motion from dust3r without training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9158–9168, 2025.

[11] S. Cho, J. Huang, J. Nam, H. An, S. Kim, and J.-Y. Lee. Local all-pair correspondence for point tracking. In *European conference on computer vision*, pages 306–325. Springer, 2024.

[12] C. Doersch, A. Gupta, L. Markeeva, A. Recasens, L. Smaira, Y. Aytar, J. Carreira, A. Zisserman, and Y. Yang. Tap-vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems*, 35:13610–13626, 2022.

[13] C. Doersch, P. Luc, Y. Yang, D. Gokay, S. Koppula, A. Gupta, J. Heyward, I. Rocco, R. Goroshin, J. Carreira, et al. Bootstap: Bootstrapped training for tracking-any-point. In *Proceedings of the Asian Conference on Computer Vision*, pages 3257–3274, 2024.

[14] C. Doersch, Y. Yang, M. Vecerik, D. Gokay, A. Gupta, Y. Aytar, J. Carreira, and A. Zisserman. Tapir: Tracking any point with per-frame initialization and temporal refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10061–10072, 2023.

[15] N. S. Dutt, S. Muralikrishnan, and N. J. Mitra. Diffusion 3d features (diff3f): Decorating untextured shapes with distilled semantic features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4494–4504, June 2024.

[16] Y. Erel, O. Dünkel, R. Dabral, V. Golyanik, C. Theobalt, and A. H. Bermano. Attention (as discrete-time markov) chains. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026.

[17] H. Feng, J. Zhang, Q. Wang, Y. Ye, P. Yu, M. J. Black, T. Darrell, and A. Kanazawa. St4rtrack: Simultaneous 4d reconstruction and tracking in the world. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8503–8513, 2025.

[18] S. Fu, N. Tamir, S. Sundaram, L. Chai, R. Zhang, T. Dekel, and P. Isola. Dreamsim: Learning new dimensions of human visual similarity using synthetic data. In *Advances in Neural Information Processing Systems*, volume 36, pages 50742–50768, 2023.

[19] Z. Guo, J. Xiang, K. Ma, W. Zhou, H. Li, and R. Zhang. Make-it-animatable: An efficient framework for authoring animation-ready 3d characters. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10783–10792, 2025.

[20] A. W. Harley, Z. Fang, and K. Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *European Conference on Computer Vision*, pages 59–75. Springer, 2022.

[21] A. W. Harley, Y. You, X. Sun, Y. Zheng, N. Raghuraman, Y. Gu, S. Liang, W.-H. Chu, A. Dave, S. You, et al. Alltracker: Efficient dense point tracking at high resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5253–5262, 2025.

[22] A. Hertz, R. Mokady, J. Tenenbaum, K. Aberman, Y. Pritch, and D. Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022.

[23] Y. Hong, K. Zhang, J. Gu, S. Bi, Y. Zhou, D. Liu, F. Liu, K. Sunkavalli, T. Bui, and H. Tan. Lrm: Large reconstruction model for single image to 3d. *arXiv preprint arXiv:2311.04400*, 2023.

[24] H. Jeong, C.-H. P. Huang, J. C. Ye, N. J. Mitra, and D. Ceylan. Track4gen: Teaching video diffusion models to track points improves video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[25] Y. Jiang, C. Yu, C. Cao, F. Wang, W. Hu, and J. Gao. Animate3d: Animating any 3d model with multi-view video diffusion. *Advances in Neural Information Processing Systems*, 37:125879–125906, 2024.

[26] Y. Jiang, L. Zhang, J. Gao, W. Hu, and Y. Yao. Consistent4d: Consistent 360° dynamic object generation from monocular video. In *The Twelfth International Conference on Learning Representations*, 2024.

[27] Z. Jiang, C. Zheng, I. Laina, D. Larlus, and A. Vedaldi. Mesh4d: 4d mesh reconstruction and tracking from monocular video. *arXiv preprint arXiv:2601.05251*, 2026.

[28] L. Jin, R. Tucker, Z. Li, D. Fouhey, N. Snavely, and A. Holynski. Stereo4d: Learning how things move in 3d from internet stereo videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10497–10509, 2025.

[29] N. Karaev, Y. Makarov, J. Wang, N. Neverova, A. Vedaldi, and C. Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6013–6022, 2025.

[30] N. Karaev, I. Rocco, B. Graham, N. Neverova, A. Vedaldi, and C. Rupprecht. Dynamicstereo: Consistent dynamic depth from stereo videos. *CVPR*, 2023.

[31] N. Karaev, I. Rocco, B. Graham, N. Neverova, A. Vedaldi, and C. Rupprecht. Cotracker: It is better to track together. In *European conference on computer vision*, pages 18–35. Springer, 2024.

[32] J. Karhade, N. Keetha, Y. Zhang, T. Gupta, A. Sharma, S. Scherer, and D. Ramanan. Any4d: Unified feed-forward metric 4d reconstruction. *arXiv preprint arXiv:2512.10935*, 2025.

[33] Y. Kasten, W. Lu, and H. Maron. Fast encoder-based 3d from casual videos via point track processing. *Advances in Neural Information Processing Systems*, 37:96150–96180, 2024.

[34] M. Kwon, J. Choi, J. Park, S. Jeon, J. Jang, J. Seo, M.-S. Kwak, J.-H. Kim, and S. Kim. Cameo: Correspondence-attention alignment for multi-view diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[35] Z. Lai, E. Insafutdinov, E. Sucar, and A. Vedaldi. Cowtracker: Tracking by warping instead of correlation. *arXiv preprint arXiv:2602.04877*, 2026.

[36] G. Le Moing, J. Ponce, and C. Schmid. Dense optical tracking: Connecting the dots. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19187–19197, 2024.

[37] V. Lepetit, F. Moreno-Noguer, and P. Fua. Epnp: Efficient perspective-n-point camera pose estimation. *Int. J. Comput. Vis*, 81(2):155–166, 2009.

[38] W. Li, J. Liu, H. Yan, R. Chen, Y. Liang, X. Chen, P. Tan, and X. Long. Craftsman3d: High-fidelity mesh generation with 3d native diffusion and interactive geometry refiner. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5307–5317, 2025.

[39] W. Li, X. Zhang, Z. Sun, D. Qi, H. Li, W. Cheng, W. Cai, S. Wu, J. Liu, Z. Wang, et al. Step1x-3d: Towards high-fidelity and controllable generation of textured 3d assets. *arXiv preprint arXiv:2505.07747*, 2025.

[40] X. Li, F. Zhang, J. Pan, Y. Hou, V. Y. F. Tan, and Z. Yang. Enhancing long video generation consistency without tuning. *arXiv preprint arXiv:2412.17254*, 2024. ICML 2025 Workshop on Building Physically Plausible World Models.

[41] Y. Li, Z.-X. Zou, Z. Liu, D. Wang, Y. Liang, Z. Yu, X. Liu, Y.-C. Guo, D. Liang, W. Ouyang, et al. Triposg: High-fidelity 3d shape synthesis using large-scale rectified flow models. *arXiv preprint arXiv:2502.06608*, 2025.

[42] Z. Li, Y. Chen, and P. Liu. Dreammesh4d: Video-to-4d generation with sparse-controlled gaussian-mesh hybrid representation. *Advances in Neural Information Processing Systems*, 37:21377–21400, 2024.

[43] Z. Li, R. Tucker, F. Cole, Q. Wang, L. Jin, V. Ye, A. Kanazawa, A. Holynski, and N. Snavely. Megasam: Accurate, fast and robust structure and motion from casual dynamic videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10486–10496, 2025.

[44] H. Liang, Y. Yin, D. Xu, H. Liang, Z. Wang, K. N. Plataniotis, Y. Zhao, and Y. Wei. Diffusion4d: Fast spatial-temporal consistent 4d generation via video diffusion models. *arXiv preprint arXiv:2405.16645*, 2024.

[45] H. Lin, S. Chen, J. H. Liew, D. Y. Chen, Z. Li, G. Shi, J. Feng, and B. Kang. Depth anything 3: Recovering the visual space from any views. *arXiv preprint arXiv:2511.10647*, 2025.

[46] I. Liu, Z. Xu, Y. Wang, H. Tan, Z. Xu, X. Wang, H. Su, and Z. Shi. Riganything: Template-free autoregressive rigging for diverse 3d assets. *ACM Transactions on Graphics (TOG)*, 44(4):1–12, 2025.

[47] X. Liu, Y. Xiao, D. Y. Chen, J. Feng, Y.-W. Tai, C.-K. Tang, and B. Kang. Trace anything: Representing any video in 4d via trajectory fields. *arXiv preprint arXiv:2510.13802*, 2025.

[48] R. Lu, Y. Chen, Y. Liu, J. Tang, J. Ni, D. Wan, G. Zeng, and S. Huang. Taco: Taming diffusion for in-the-wild video amodal completion. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

[49] Y. Luo, S. Zhou, Y. Lan, X. Pan, and C. C. Loy. 4rc: 4d reconstruction via conditional querying anytime and anywhere. *arXiv preprint arXiv:2602.10094*, 2026.

[50] S. Muralikrishnan, N. S. Dutt, and N. J. Mitra. Smf: Template-free and rig-free animation transfer using kinetic codes. *ACM Transactions on Graphics (TOG)*, 44(6), 2025.

[51] J. Nam, S. Son, D. Chung, J. Kim, S. Jin, J. Hur, and S. Kim. Emergent temporal correspondences from video diffusion transformers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[52] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. B. Huang, S.-W. Li, I. Misra, M. G. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jégou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski. Dinov2: Learning robust visual features without supervision. *TMLR*, 2024.

[53] A. Pondaven, A. Siarohin, S. Tulyakov, P. Torr, and F. Pizzati. Video motion transfer with diffusion transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[54] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017.

[55] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[56] J. Ren, L. Pan, J. Tang, C. Zhang, A. Cao, G. Zeng, and Z. Liu. Dreamgaussian4d: Generative 4d gaussian splatting. *arXiv preprint arXiv:2312.17142*, 2023.

[57] J. Ren, K. Xie, A. Mirzaei, H. Liang, X. Zeng, K. Kreis, Z. Liu, A. Torralba, S. Fidler, S. W. Kim, et al. L4gm: Large 4d gaussian reconstruction model. *Advances in Neural Information Processing Systems*, 37:56828–56858, 2024.

[58] R. Sabathier, N. J. Mitra, and D. Novotny. Lim: Large interpolator model for dynamic reconstruction. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 6154–6164, 2025.

[59] R. Sabathier, D. Novotny, N. J. Mitra, and T. Monnier. Actionmesh: Animated 3d mesh generation with temporal 3d diffusion. *arXiv preprint arXiv:2601.16148*, 2026.

[60] D. Samuel, M. Levy, N. Darshan, G. Chechik, and R. Ben-Ari. Omnimattezero: Fast training-free omnimatte with pre-trained video diffusion models. In *SIGGRAPH Asia 2025 Conference Papers*, 2025.

[61] Y. Shi, Y. Liu, Y. Wu, X. Liu, C. Zhao, J. Luo, and B. Zhou. Drive any mesh: 4d latent diffusion for mesh deformation from video. *arXiv preprint arXiv:2506.07489*, 2025.

[62] A. Shrivastava, S. Mehta, D. Geng, and A. Owens. Point prompting: Counterfactual tracking with video diffusion models. In *International Conference on Learning Representations*, 2026.

[63] Y. Siddiqui, T. Monnier, F. Kokkinos, M. Kariya, Y. Kleiman, E. Garreau, O. Gafni, N. Neverova, A. Vedaldi, R. Shapovalov, et al. Meta 3d assetgen: Text-to-mesh generation with high-quality geometry, texture, and pbr materials. *Advances in Neural Information Processing Systems*, 37:9532–9564, 2024.

[64] C. Song, J. Zhang, X. Li, F. Yang, Y. Chen, Z. Xu, J. H. Liew, X. Guo, F. Liu, J. Feng, et al. Magicarticulate: Make your 3d models articulation-ready. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 15998–16007, 2025.

[65] O. Sorkine and M. Alexa. As-rigid-as-possible surface modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, 2007.

[66] E. Sucar, E. Insafutdinov, Z. Lai, and A. Vedaldi. V-DPM: 4d video reconstruction with dynamic point maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[67] R. W. Sumner, J. Schmid, and M. Pauly. Embedded deformation for shape manipulation. In *ACM siggraph 2007 papers*. 2007.

[68] J. Tang, Z. Chen, X. Chen, T. Wang, G. Zeng, and Z. Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *European Conference on Computer Vision*, pages 1–18. Springer, 2024.

[69] L. Tang, M. Jia, Q. Wang, C. P. Phoo, and B. Hariharan. Emergent correspondence from image diffusion. *Advances in neural information processing systems*, 36:1363–1389, 2023.

[70] T. H. Team. Hunyuan3d 2.5: Towards high-fidelity 3d assets generation with ultimate details, 2025.

[71] G. Terzakis and M. Lourakis. A consistently fast and globally optimal solution to the perspective-n-point problem. In *European Conference on Computer Vision*, pages 478–494. Springer, 2020.

[72] Y. Tewel, O. Kaduri, R. Gal, Y. Kasten, L. Wolf, G. Chechik, and Y. Atzmon. Training-free consistent text-to-image generation. *ACM Transactions on Graphics (TOG)*, 43(4):1–18, 2024.

[73] N. Tumanyan, M. Geyer, S. Bagon, and T. Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1921–1930, June 2023.

[74] D. Vlasic, I. Baran, W. Matusik, and J. Popović. Articulated mesh animation from multi-view silhouettes. 2008.

[75] J. Wang, M. Chen, N. Karaev, A. Vedaldi, C. Rupprecht, and D. Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5294–5306, 2025.

[76] Q. Wang, Y.-Y. Chang, R. Cai, Z. Li, B. Hariharan, A. Holynski, and N. Snavely. Tracking everything everywhere all at once. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19795–19806, 2023.

[77] Q. Wang, Y. Zhang, A. Holynski, A. A. Efros, and A. Kanazawa. Continuous 3d perception model with persistent state. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10510–10522, 2025.

[78] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20697–20709, 2024.

[79] Y. Wang, X. Wang, Z. Chen, Z. Wang, F. Sun, and J. Zhu. Vidu4d: Single generated video to high-fidelity 4d reconstruction with dynamic gaussian surfels. *Advances in Neural Information Processing Systems*, 37:131316–131343, 2024.

[80] R. Wu, R. Gao, B. Poole, A. Trevithick, C. Zheng, J. T. Barron, and A. Holynski. Cat4d: Create anything in 4d with multi-view video diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26057–26068, 2025.

[81] Z. Wu, C. Yu, Y. Jiang, C. Cao, F. Wang, and X. Bai. Sc4d: Sparse-controlled video-to-4d generation and motion transfer. In *European Conference on Computer Vision*, pages 361–379. Springer, 2024.

[82] Z. Wu, C. Yu, F. Wang, and X. Bai. Animateanymesh: A feed-forward 4d foundation model for text-driven universal mesh animation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13557–13568, 2025.

[83] J. Xiang, Z. Lv, S. Xu, Y. Deng, R. Wang, B. Zhang, D. Chen, X. Tong, and J. Yang. Structured 3d latents for scalable and versatile 3d generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 21469–21480, 2025.

[84] Y. Xiao, J. Wang, N. Xue, N. Karaev, Y. Makarov, B. Kang, X. Zhu, H. Bao, Y. Shen, and X. Zhou. Spatialtrackerv2: 3d point tracking made easy. *arXiv preprint arXiv:2507.12462*, 2025.

[85] Y. Xiao, Q. Wang, S. Zhang, N. Xue, S. Peng, Y. Shen, and X. Zhou. Spatialtracker: Tracking any 2d pixels in 3d space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20406–20417, 2024.

[86] Y. Xie, C.-H. Yao, V. Voleti, H. Jiang, and V. Jampani. SV4D: Dynamic 3d content generation with multi-frame and multi-view consistency. *arXiv preprint arXiv:2407.17470*, 2024.

[87] T.-X. Xu, X. Gao, W. Hu, X. Li, S.-H. Zhang, and Y. Shan. Geometrycrafter: Consistent geometry estimation for open-world videos with diffusion priors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6632–6644, 2025.

[88] Z. Xu, Z. Li, Z. Dong, X. Zhou, R. Newcombe, and Z. Lv. 4dgt: Learning a 4d gaussian transformer using real-world monocular videos. 2025.

[89] C.-H. Yao, Y. Xie, V. Voleti, H. Jiang, and V. Jampani. Sv4d 2.0: Enhancing spatio-temporal consistency in multi-view video diffusion for high-quality 4d generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13248–13258, 2025.

[90] Y. Yao, Z. Deng, and J. Hou. Riggs: Rigging of 3d gaussians for modeling articulated objects in videos. In *CVPR*, 2025.

[91] J. Yenphraphai, J. Chen, J. Wang, G. G. Qian, S. Tulyakov, R. Abdal, R. A. Yeh, P. Wonka, and C. Wang. Helix4d: Complex 4d mesh generation. *arXiv preprint arXiv:2605.26109*, 2026.

[92] J. Yenphraphai, A. Mirzaei, J. Chen, J. Zou, S. Tulyakov, R. A. Yeh, P. Wonka, and C. Wang. Shapegen4d: Towards high quality 4d shape generation from videos. *arXiv preprint arXiv:2510.06208*, 2025.

[93] M. Yin, W. Hu, J. Xu, Y. Shan, and K. Han. Sculpt4d: Generating 4d shapes via sparse-attention diffusion transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[94] Y. Yin, D. Xu, Z. Wang, Y. Zhao, and Y. Wei. 4dgen: Grounded 4d content generation with spatial-temporal consistency. *arXiv preprint arXiv:2312.17225*, 2023.

[95] T. Yuan, Y. Yang, L.-Z. Chen, Y. Yao, and Z. Qian. Denoise to track: Harnessing video diffusion priors for robust correspondence. *arXiv preprint arXiv:2512.04619*, 2025.

[96] Y. Zeng, Y. Jiang, S. Zhu, Y. Lu, Y. Lin, H. Zhu, W. Hu, X. Cao, and Y. Yao. Stag4d: Spatial-temporal anchored generative 4d gaussians. In *European Conference on Computer Vision*, pages 163–179. Springer, 2024.

[97] B. Zhang, L. Ke, A. W. Harley, and K. Fragkiadaki. Tapip3d: Tracking any point in persistent 3d geometry. *arXiv preprint arXiv:2504.14717*, 2025.

[98] B. Zhang, J. Tang, M. Niessner, and P. Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Transactions On Graphics (TOG)*, 42(4):1–16, 2023.

[99] B. Zhang, S. Xu, C. Wang, J. Yang, F. Zhao, D. Chen, and B. Guo. Gaussian variation field diffusion for high-fidelity video-to-4d synthesis. *arXiv preprint arXiv:2507.23785*, 2025.

[100] H. Zhang, X. Chen, Y. Wang, X. Liu, Y. Wang, and Y. Qiao. 4diffusion: Multi-view video diffusion model for 4d generation. *Advances in Neural Information Processing Systems*, 37:15272–15295, 2024.

[101] J. Zhang, C. Herrmann, J. Hur, V. Jampani, T. Darrell, F. Cole, D. Sun, and M.-H. Yang. Monst3r: A simple approach for estimating geometry in the presence of motion. *arXiv preprint arXiv:2410.03825*, 2024.

[102] L. Zhang, Z. Wang, Q. Zhang, Q. Qiu, A. Pang, H. Jiang, W. Yang, L. Xu, and J. Yu. Clay: A controllable large-scale generative model for creating high-quality 3d assets. *ACM Transactions on Graphics (TOG)*, 43(4):1–20, 2024.

[103] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[104] Y. Zheng, A. W. Harley, B. Shen, G. Wetzstein, and L. J. Guibas. Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19855–19865, 2023.

Supplementary Material

A Additional Qualitative Results

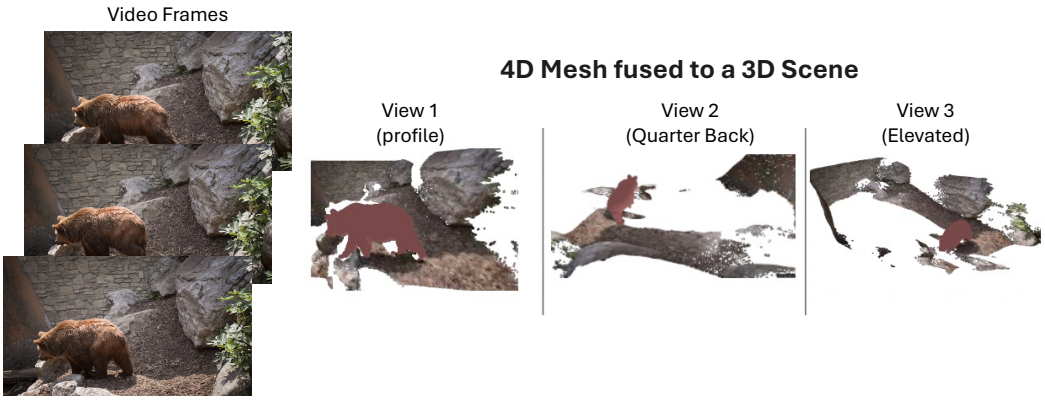


Figure 4: **Video-to-4D Scene Alignment.** Given our generated 4D object and a reconstructed background scene, we align the object to the environment using our dense 2D-to-3D correspondences. While this example utilizes Depth Anything V3 [45], our alignment framework is completely agnostic and functions seamlessly with any underlying scene reconstruction method. Please refer to our supplementary website for interactive, dynamic visualizations.

We provide additional qualitative comparisons that complement the quantitative evaluation in the main paper. These results further demonstrate the advantages of our training-free attention-chain framework across four settings: 4D mesh generation, long-sequence rollout, 2D point tracking, and 4D mesh placement in reconstructed scenes.

4D mesh generation. Figure 5 presents additional side-by-side comparisons between ActionMesh [59] and our method on sequences from ActionBench and in-the-wild videos. Our approach consistently produces comparable to better geometry with fewer temporal artifacts, while running in only 9 s per clip compared to ActionMesh's 120 s. Unlike ActionMesh, which generates object-centric meshes, our spatial attention-chain correspondences enable camera recovery and world placement, yielding high foreground overlap (yellow) and fewer mismatch regions (red/green).

Long-sequence generation. Figure 6 shows autoregressive rollouts spanning 240 frames. ActionMesh accumulates drift over successive windows, leading to progressive geometry degradation and loss of recognizable structure. In contrast, our correspondence reinforcement (Sec. 4.3) maintains stable mesh quality throughout the sequence, preserving both global pose and fine detail even at frame 240.

2D point tracking. Figure 7 compares our zero-shot 2D point tracking against Denoise-to-Track [95], the current zero-shot state of the art. Each colored dot represents a tracked query point, with trails showing the predicted trajectory. Our method produces smoother, more accurate trajectories that better follow the underlying surface motion, particularly on articulated body parts where attention-chain correspondences provide geometrically grounded matches rather than purely appearance-based ones.

4D mesh placement. Figure 4 demonstrates that our camera pose estimation enables fusing the generated 4D mesh directly into a reconstructed 3D scene. Given only the input video frames, we recover the camera trajectory via PnP and place the animated mesh into a scene point cloud, allowing it to be viewed from arbitrary novel viewpoints. The mesh correctly occupies the physical space of the subject across all views, confirming that our attention-chain correspondences provide accurate world-space localization without any pose supervision. An interactive version with full camera control is available on the supplementary website.

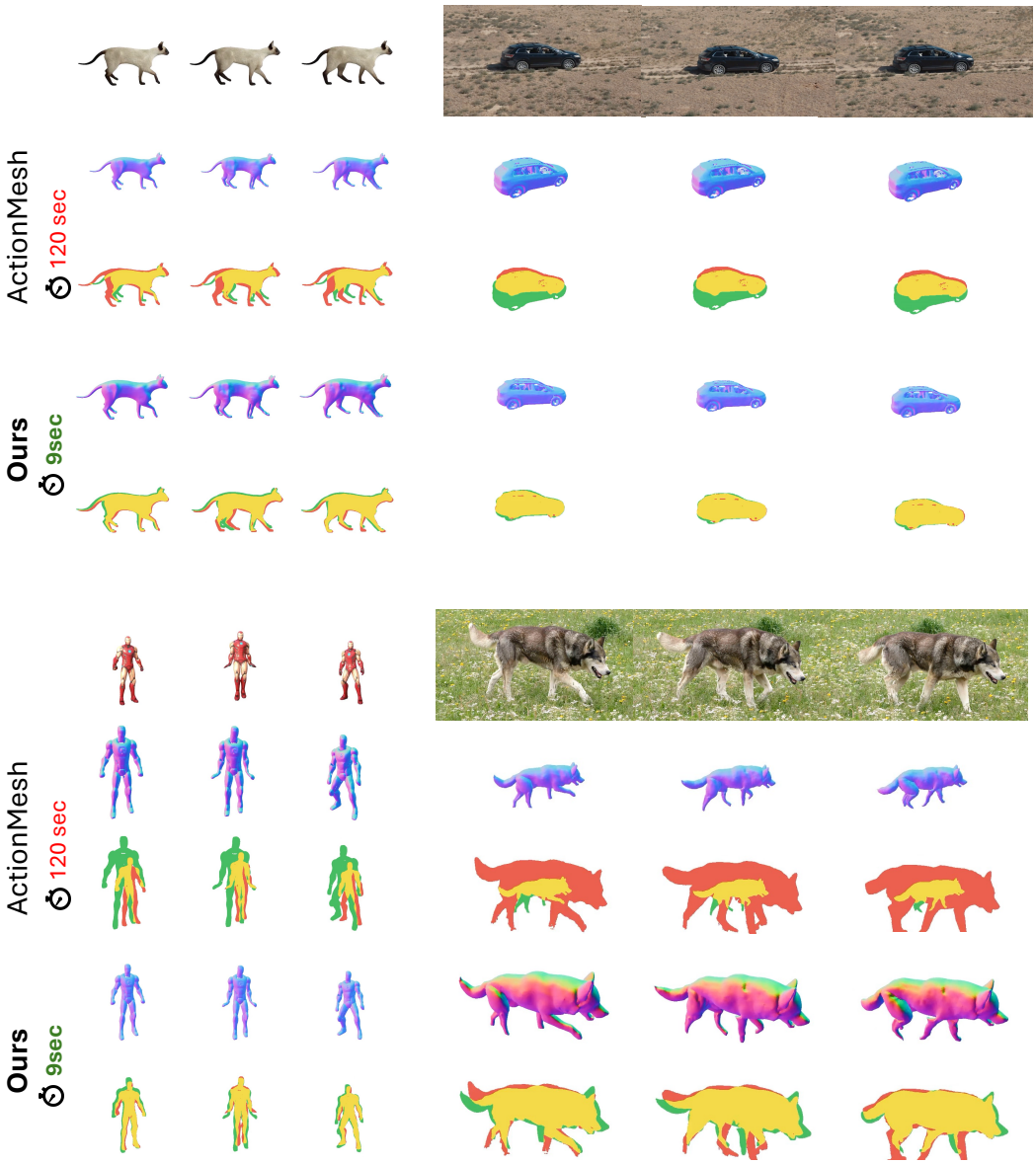


Figure 5: **Additional 4D mesh generation results.** Side-by-side comparison of ActionMesh [59] (left of each pair) and our method (right) on diverse ActionBench sequences. Each row shows a different input video. Our method produces sharper, temporally consistent meshes with fewer distortions while requiring only 9 s per clip vs. 120 s for ActionMesh.

B Ablation Study

We probe three design choices: (i) the number of Stage I denoising steps, (ii) the resulting per-clip wall-clock vs. ActionMesh, and (iii) the contribution of each component on long, 240-frame sequences. Unless stated otherwise we evaluate on ActionBench with the same protocol as Section 5.1.

Denoising steps vs. quality. Both methods share the same Stage I temporal denoiser Φ_θ; ActionMesh further runs a learned Stage II that regularises the per-frame latents Z, while ours reads the attention of the *last* denoising step and propagates it through the chain $\mathbf{V} \to \mathbf{T} \to \mathbf{T} \to \mathbf{V}$. Figure 8 plots CD-3D, CD-4D and CD-Motion as a function of the number of denoising steps. Ours plateaus by step 4 on every metric, whereas ActionMesh keeps improving up to $\sim$ 10–20 steps. On CD-3D and CD-4D the gap is large in the few-step regime that matters for inference cost: at 4 steps ours is roughly $2\times$ better on CD-3D (0.048 vs. 0.095) and CD-4D (0.077 vs. 0.125), and stays strictly better

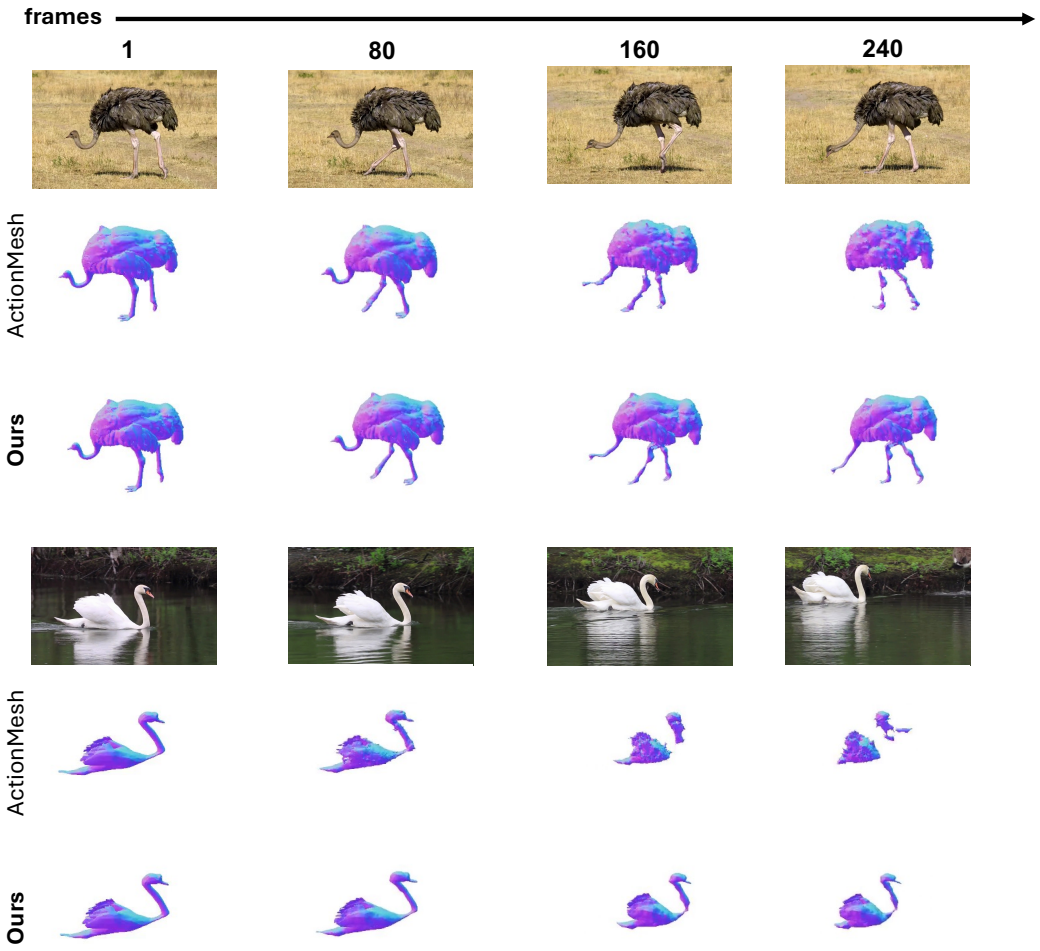


Figure 6: **Long-sequence autoregressive generation.** Mesh quality over 240 frames (sampled at frames 1, 80, 160, 240). ActionMesh progressively degrades due to accumulated drift across autoregressive windows, eventually losing recognizable geometry. Our correspondence reinforcement maintains stable, high-quality meshes throughout the entire sequence.



Figure 7: **Zero-shot 2D point tracking.** Comparison with Denoise-to-Track [95] on challenging sequences with articulated motion. Colored dots mark tracked query points; trails show predicted trajectories. Our attention-chain correspondences yield geometrically grounded tracks that more faithfully follow the underlying surface motion, particularly on limbs and fine structures.

even at 30 steps. On CD-Motion ActionMesh is consistently slightly better – its learned Stage II

yields smoother frame-to-frame motion than our closed-form correspondence chain (0.148 vs. 0.152 at 30 steps; 0.161 vs. 0.163 at 4 steps), with the gap growing at very few steps where Stage II is most needed to clean up noisy latents. The remaining headroom on motion smoothness is therefore in the temporal regulariser, not in the underlying correspondence. All other experiments use 4 denoising steps.

Inference time breakdown. We accelerate 4D mesh generation in two complementary ways (Table 4). **(i) Many fewer denoising steps.** Our correspondence is robust to slightly noisy latents and saturates by step 4 on every metric (Fig. 8), whereas ActionMesh requires $\sim$30 steps for its Stage II network to operate at peak quality – shrinking Stage I from $\sim$100 s to $\sim$7.5 s on the same backbone. **(ii) No additional network.** We drop ActionMesh's learned Stage II ($\sim$10 s) entirely and replace it with a lightweight, training-free pipeline of four closed-form operations: a single batched VAE decode for all frames at once (1.0 s), FPS landmark sampling on the anchor mesh (0.46 s), the V$\rightarrow$T$\rightarrow$T$\rightarrow$V correspondence computation (0.16 s), and a geodesic, topology-preserving animation (0.8 s) – altogether $\sim$2.4 s. End-to-end our full pipeline takes $\sim$9 s per 16-frame clip, vs. $\sim$110 s for ActionMesh: an order-of-magnitude speed-up without any loss of quality on CD-3D / CD-4D (Sec. 5.1).

Component contribution on long videos. Table 5 ablates the three components of our long-video pipeline on the 240-frame split of ActionBench, evaluated *without* ICP so that the metrics reflect the *intrinsic* alignment of each variant. ActionMesh in this regime is unaligned: its Stage II is trained on 16-frame clips and its autoregressive extension drifts in absolute pose, leading to large CD and motion errors. Replacing Stage II with our attention-chain correspondence (Sec. 4.1) tightens both metrics. Periodically re-encoding the anchor across windows (Sec. 4.3) further removes intra-AR drift. Finally, PnP-based camera-pose estimation (Sec. 4.4) folds out the remaining global rigid mismatch and gives the largest jump on CD-3D and CD-Motion. Our full pipeline more than halves both CD and motion error on the same backbone, *without ICP*.

C Implementation Details

Backbone and pipeline. We inherit the frozen, off-the-shelf two-stage backbone of `ActionMesh` [59]: Stage 0 is `TripoSG` [41], an image-to-3D diffusion model that produces the anchor mesh $\mathcal{M}_a$ ($\sim$20k vertices after decimation) and its latent code $z_a \in \mathbb{R}^{N\times d}$ with $N = 2048$, $d = 64$; Stage I is a flow-matching temporal denoiser Φ_θ that jointly produces per-frame latents $Z = \{z_f\}_{f=0}^{F-1}$ in a window of $F = 16$ frames, conditioned on per-frame DINOv2 [52] patch features and anchored at z_a. We **never train or fine-tune any model**: all weights are frozen, and our entire pipeline reduces to (i) reading the V$\rightarrow$T cross-attention of the TripoSG VAE decoder and the inflated self-attention of Φ_θ (all temporal self-attention layers, averaged over heads and layers) during its 4 denoising steps, and (ii) a sequence of closed-form correspondence and skinning operations (Sec. 4.1). The same attention chain serves 2D point tracking, 4D point tracking, and camera-pose estimation (Sec. 4.4). For 2D and 4D tracking under occlusions or unstable object masks, we first apply amodal object completion using a video diffusion model [48] before running our correspondence and deformation pipeline. For sequences longer than F frames we run a sliding 16-frame window with periodic anchor re-encoding (Sec. 4.3).

Hyperparameters. After a coarse grid search we fixed a single configuration that we use across *every* experiment in this paper – 4D mesh generation, 2D and 4D point tracking, and the long-video stress tests – with no per-scene tuning. The values are: 4 Stage I denoising steps (justified in Fig. 8); attention averaged over all heads and all inflated temporal self-attention layers; a 128^3 grid for the candidate surface points S_f and $K = 3$ top-scoring candidates in Eq. (4); 1000 FPS landmarks on the anchor mesh with curvature boost $\alpha = 2.0$; outlier rejection at $5\times$ the per-frame median landmark displacement (motion floor $\delta_{\min} = 0$, Appendix I) and a $\geq 95\%$ valid-frame requirement; Gaussian smoothing of landmark displacements with $\sigma_{\text{landmark}} = 1.5$ and of the full vertex displacement field with $\sigma_{\text{final}} = 1.0$; a geodesic-rigid skinning with $k_{\text{NN}} = 120$ Dijkstra neighbours (auto σ); and, for long-video reinforcement (Algorithm 1), a confidence threshold τ_c and strength λ.

For the camera intrinsics, we assume a fixed focal length of 2.1875 defined in canonical Normalized Device Coordinates (NDC). Because the NDC image extent spans [-1, 1], this value is independent of pixel resolution and corresponds to a full vertical field-of-view of roughly 49.2°. Camera poses are then estimated using a robust PnP solver implemented as a RANSAC loop with EPnP as the minimal

Table 4: Per-clip wall-clock breakdown (16 frames). Ours accelerates generation in two ways: (i) Stage I with 4 denoising steps instead of 30; (ii) replacement of ActionMesh's learned Stage II network with a lightweight, training-free pipeline of four cheap closed-form operations. The two together reduce per-clip latency from $\sim$110 s to $\sim$9 s.

	ActionMesh [59]	Ours
Stage I (per-frame latents Z)	100.00 *(30 steps)*	7.50 *(4 steps)*
Stage II (per-frame meshes)	15.00 *(learned)*	1.49 *(training-free)*
batched VAE decode (all frames at once)	–	0.87
FPS landmarks (anchor mesh)	–	0.46
correspondence computation (attention chain)	–	0.16
geodesic topology-preserving animation	–	0.005
Total	110.00s	**9.0s**

Table 5: Component ablation on 240-frame sequences, *no ICP*. We start from ActionMesh's unaligned predictions and add our components one-by-one. Lower is better.

Configuration	CD-3D ↓	CD-4D ↓	CD-M ↓
ActionMesh [59] (unaligned)	0.260	0.260	0.373
+ temporal correspondence (Sec. 4.1)	0.190	0.195	0.310
+ long-video AR (Sec. 4.3)	0.155	0.162	0.250
+ camera-pose estimation (Sec. 4.4)	**0.108**	**0.115**	**0.198**

solver, an 8-pixel reprojection-error inlier threshold, 400 iterations at 0.999 confidence, an SQPnP consistent-solver fallback when EPnP returns a degenerate (near-camera-at-infinity) translation, and a final Levenberg–Marquardt refinement on the inlier set.

Synthetic Long Sequences In Section 4.3, we generate extended ActionBench [59] sequences by employing a *ping-pong* looping scheme. Because standard ActionBench sequences are limited to exactly 16 frames, we artificially lengthen the temporal duration of the videos by continuously playing the frames forward and then in reverse. Specifically, the sequence ordering follows the alternating pattern $(f_0, f_1, \ldots, f_{14}, f_{15}, f_{14}, \ldots, f_1, f_0, f_1, \ldots)$. This progression allows us to generate arbitrarily long sequences while preserving smooth and continuous motion transitions. Because looping preserves motion continuity artificially, we complement it with an evaluation on genuine long sequences in Appendix F.

D Confidence-Reinforced Temporal Attention

Algorithm 1 details the correspondence reinforcement of Sec. 4.3 for one autoregressive window w; its output is applied while denoising window $w{+}1$, whose anchor is the final frame of window w. It has three phases. **Phase 1 (match)** runs the forward attention chain of Sec. 4.1 (Eqs. 2–4) for the control landmarks $\mathcal{L}$ of Sec. 4.2, yielding for every landmark v and frame f the target position $\tilde{v}_f$, the matched surface point $u^{\star}_{v,f}$, and the confidence $c_v^{(f)}$. **Phase 2 (backward trace)** attributes each confident match ($c_v^{(f)} \geq \tau_c$) to a single anchor–target token pair: the *responsibility* $r_{v,f}(t')$ of a frame-f token is its contribution to the score of the matched point, the most responsible token $t'^{\star}$ is paired with the top anchor token $t^{\star}$ of v, and the pair is added to $\mathcal{C}_f$ with weight $r_{v,f}(t'^{\star})$. Weights accumulate over landmarks that vote for the same pair and are normalized so that the strongest pair in each frame has confidence 1. **Phase 3 (reinforce)** multiplies the temporal attention of every pair in $\mathcal{C}_f$ by $(1 + \lambda\, c^{(f)}_{t,t'})$ and renormalizes each row, so that every row remains a probability distribution; attention within the anchor frame is left unchanged. The reinforced attention replaces $A_{Z_a \to Z_f}$ in all denoising steps of the next window, which is otherwise denoised normally; the matching itself uses the layer- and head-averaged attention of Sec. 4.1. The confidence threshold τ_c and strength λ are fixed across all experiments (Appendix C).

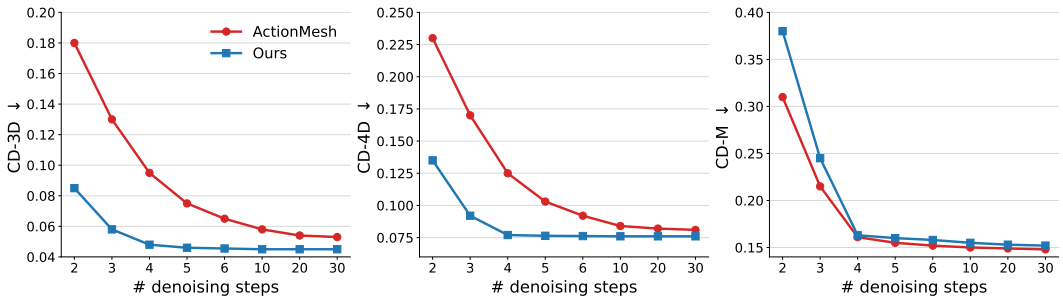


Figure 8: CD-3D / CD-4D / CD-Motion vs. Stage I denoising steps on ActionBench. Ours plateaus by step 4 and decisively wins CD-3D and CD-4D at every step count; ActionMesh's learned Stage II is consistently smoother on CD-Motion.

E Generalization to Other 4D Backbones: Protocols

This section provides the implementation and evaluation details for the experiments of Sec. 5.2 (Table 2a,b). Our method applies to staged video-to-4D pipelines in which Stage I is a temporal diffusion model operating on a set of per-frame shape latents (Sec. 3). In all cases we keep the backbone's pretrained weights frozen, run its Stage 0 to obtain the anchor mesh and anchor latent, read the three attention maps of Sec. 4.1 from the backbone's own decoder and temporal self-attention layers (averaged over heads and layers, as for ActionMesh), and animate the anchor mesh with the pipeline of Sec. 4.2 using the *same* hyperparameters as in Appendix C, without any per-backbone tuning. All timings were measured on a single H100 GPU.

Sculpt4D [93]. Sculpt4D builds a temporal diffusion transformer with block-sparse attention on top of a pretrained image-to-3D generator, and produces a mesh for every frame by running its full sampling schedule and decoding each frame independently. We follow its evaluation protocol and sequence length (16-frame videos), and report Chamfer distance, volumetric IoU, and F-Score as defined in the Sculpt4D paper. Our variant keeps Sculpt4D's Stage 0 anchor and runs its temporal denoiser only for the few steps needed to read stable attention (Fig. 8), replacing the costly per-frame mesh generation and decoding with the attention-chain readout and the topology-preserving animation of the anchor mesh. This reduces end-to-end generation time from 960 s to 11 s ($\sim 87\times$) while improving all three metrics (Table 2a).

Mesh4D [27]. Mesh4D follows the three-stage design of Sec. 3, with a learned topology-preserving decoding stage. We evaluate on 6-frame videos, as in the Mesh4D paper, and report Chamfer distance, IoU, and the PS2 metric used there. Since Mesh4D does not release its evaluation code or data, we reproduced its protocol from the description in the paper; absolute numbers may therefore differ slightly from those reported by the authors, but both rows of Table 2(b) use the identical reproduced protocol. In Mesh4D the overall runtime is dominated by the unavoidable cost of generating the anchor mesh, so the attainable speedup is bounded; our method still reduces end-to-end time from 30 s to 16 s ($\sim 1.9\times$) while improving Chamfer, IoU, and PS2.

Discussion. The two backbones differ from ActionMesh in architecture (dense vs. block-sparse temporal attention), latent representation, and VAE design, and their exact attention patterns differ accordingly. What they share is the structure our chain depends on: a decoder whose query points cross-attend to a set of shape tokens, and a temporal denoiser whose self-attention links anchor-frame tokens to target-frame tokens. The consistent gains in Table 2 suggest that the attention-chain principle transfers across this class of models rather than being a property of one particular network.

F Evaluation on Genuine Long Sequences

To evaluate long-sequence generation on natural rather than looped motion, we use the MIT mesh-animation dataset of Vlasic et al. [74], which provides ground-truth, topology-consistent mesh sequences of articulated human motion with up to 250 frames. We render each sequence to obtain the input video, run both methods autoregressively with the 16-frame sliding window of Sec. 4.3, and report CD-3D and CD-4D over the first window and over the window ending at frame 250,

Algorithm 1 Confidence-reinforced temporal attention for one autoregressive window (Sec. 4.3)

Require: Control landmarks $\mathcal{L}$; anchor vertex–token attention $A_{V_a \to Z_a} \in \mathbb{R}^{|V_a| \times N}$; per-frame temporal attention $A_{Z_a \to Z_f} \in \mathbb{R}^{N \times N}$; per-frame point–token attention $A^{\top}_{Z_f \to V_f} \in \mathbb{R}^{|V_f| \times N}$ over candidate surface points $S_f = \{x_u^{(f)}\}_{u=1}^{|V_f|}$; temperature τ, neighbourhood size K, confidence threshold τ_c, strength λ.

Ensure: Reinforced temporal attention $\widetilde{A}_{Z_a \to Z_f}$ for every frame f, applied during the next window.

▷ **Phase 1 — Match: forward attention chain (Eqs. 2–4)**
for each landmark $v \in \mathcal{L}$ and target frame f **do**
  $\ddot{A}_{v,Z_f}(t') \leftarrow \sum_{t=1}^{N} A_{V_a \to Z_a}[v,t]\, A_{Z_a \to Z_f}[t,t']$ for all t' ▷ distribution over frame-f tokens
  $s_{v,f}(u) \leftarrow \sum_{t'=1}^{N} \ddot{A}_{v,Z_f}(t')\, A^{\top}_{Z_f \to V_f}[u,t']$ for all u
  $\mathcal{N}_{v,f} \leftarrow \mathrm{TopK}_K\big(\{s_{v,f}(u)\}_{u=1}^{|V_f|}\big)$
  $\pi_{v,f}(u) \leftarrow \exp(s_{v,f}(u)/\tau) \big/ \sum_{q \in \mathcal{N}_{v,f}} \exp(s_{v,f}(q)/\tau)$ for $u \in \mathcal{N}_{v,f}$
  $\tilde{v}_f \leftarrow \sum_{u \in \mathcal{N}_{v,f}} \pi_{v,f}(u)\, x_u^{(f)}$ ▷ target position
  $u^{\star}_{v,f} \leftarrow \arg\max_u s_{v,f}(u), \quad c_v^{(f)} \leftarrow s_{v,f}(u^{\star}_{v,f})$ ▷ matched surface point and confidence
end for

▷ **Phase 2 — Backward trace: attribute each match to a token pair**
$\mathcal{C}_f \leftarrow \emptyset$ and $w^{(f)}_{t,t'} \leftarrow 0$ for all f, t, t'
for each (v, f) with $c_v^{(f)} \geq \tau_c$ **do**
  $r_{v,f}(t') \leftarrow \ddot{A}_{v,Z_f}(t')\, A^{\top}_{Z_f \to V_f}[u^{\star}_{v,f}, t']$ for all t' ▷ responsibility of token t'
  $t'^{\star}_{v,f} \leftarrow \arg\max_{t'} r_{v,f}(t')$
  $t^{\star}_{v,f} \leftarrow \arg\max_t A_{V_a \to Z_a}[v,t]$ ▷ single top anchor token of v
  $\mathcal{C}_f \leftarrow \mathcal{C}_f \cup \{(t^{\star}_{v,f}, t'^{\star}_{v,f})\}$
  $w^{(f)}_{t^{\star}_{v,f}, t'^{\star}_{v,f}} \leftarrow w^{(f)}_{t^{\star}_{v,f}, t'^{\star}_{v,f}} + r_{v,f}(t'^{\star}_{v,f})$ ▷ accumulate pair weight
end for
for each frame f and $(t,t') \in \mathcal{C}_f$ **do**
  $c^{(f)}_{t,t'} \leftarrow w^{(f)}_{t,t'} \big/ \max_{(a,b) \in \mathcal{C}_f} w^{(f)}_{a,b}$ ▷ normalised so that $\max_{(t,t') \in \mathcal{C}_f} c^{(f)}_{t,t'} = 1$
end for

▷ **Phase 3 — Reinforce temporal attention for the next window**
for each frame f and anchor token $t = 1, \ldots, N$ **do**
  $\widetilde{A}_{Z_a \to Z_f}[t,t'] \leftarrow \big(1 + \lambda\, c^{(f)}_{t,t'}\, \mathbf{1}[(t,t') \in \mathcal{C}_f]\big)\, A_{Z_a \to Z_f}[t,t']$ for all t' ▷ gate pairs in $\mathcal{C}_f$
  $\widetilde{A}_{Z_a \to Z_f}[t,:] \leftarrow \widetilde{A}_{Z_a \to Z_f}[t,:] \big/ \sum_{k=1}^{N} \widetilde{A}_{Z_a \to Z_f}[t,k]$ ▷ row-renormalise
end for
return $\{\widetilde{A}_{Z_a \to Z_f}\}_f$ ▷ anchor block left unchanged; applied during window $w{+}1$

using the metric protocol of Sec. 5.1. Table 6 summarizes the results. Both methods start from comparable accuracy, but the naive ActionMesh rollout more than doubles its error by frame 250, whereas our correspondence reinforcement keeps the increase to roughly 25%, consistent with the qualitative rollouts of Fig. 6 and with the 2D/4D tracking evaluations on sequences longer than 16 frames (Sec. 5.3, 5.4).

G Anchor-Mesh Quality: Failure and Recovery Cases

Our method operates on the temporal correspondences produced inside the frozen pipeline; it does not alter the anchor topology and does not reconstruct missing geometry. The effect of an anchor artifact is therefore decided by the backbone: Stage 0 fixes the initial geometry and topology, and the Stage I denoiser determines whether the artifact is preserved or suppressed over time. Fig. 9

Table 6: Error accumulation on genuine 250-frame sequences from the MIT mesh-animation dataset [74]. CD-3D / CD-4D (↓) over the first 16-frame window and over the window ending at frame 250.

	First window		Frame 250	
Method	CD-3D	CD-4D	CD-3D	CD-4D
ActionMesh [59] (naive rollout)	0.053	0.086	0.121	0.223
Ours	**0.051**	**0.082**	**0.063**	**0.104**

illustrates both regimes on the same input, a sit-up sequence whose image-to-3D anchor fuses the subject's two forearms into a single bar in front of the chest.

Failure: artifacts the denoiser preserves. With the ActionMesh backbone, the Stage I denoiser treats the fused bar as part of the object and keeps encoding it in the per-frame latents. Our chain then establishes, and during reinforcement (Sec. 4.3) strengthens, temporal correspondences for that region, so the artifact persists through the entire sequence and moves rigidly with the torso, since geodesic skinning follows the anchor connectivity (second row of Fig. 9). ActionMesh's learned Stage II animates the same anchor and exhibits the identical failure; the two outputs coincide, ours being $13\times$ faster. Our confidence weighting reduces the influence of regions with weak or inconsistent matches, but it cannot correct geometry that the backbone consistently treats as valid.

Recovery: artifacts the denoiser suppresses. With the stronger Sculpt4D denoiser (Sec. 5.2), the fused bar is not supported by the generated per-frame latents: no candidate surface point in S_f agrees with the token distribution of the bar's anchor vertices, so these vertices receive low chain confidence $c_v^{(f)}$, their landmarks are rejected, and the region is not reinforced or propagated to later frames. The animated arms follow the input video and separate as in the input (fourth row). Notably, Sculpt4D's own output retains the artifact in every frame (third row), whereas our attention-chain readout on the very same frozen denoiser recovers. In this regime our pipeline degrades gracefully and the anchor error does not cascade through the sequence.

H Structured Latents

Our chain is formulated for VecSet-style latents [98], where tokens are unordered and carry no explicit position. Locating a token in 3D therefore requires the token-to-surface step of Sec. 4.1: decoding a dense candidate point set S_f for every frame and estimating the target position as a softmax-weighted blend of the top-K candidates. A structured latent representation such as the SLAT representation of TRELLIS [83], in which each token is attached to an active voxel with an explicit coordinate, would simplify both endpoints of the chain. The vertex-to-token step becomes close to a direct spatial lookup of the voxels around a vertex, and the token-to-vertex step can use the voxel coordinate of the transported token directly. This would eliminate the per-frame dense decoding and the top-K blending, and with them the interpolation and grid-quantization errors they introduce, while leaving the temporal token-to-token step unchanged. We expect the correspondence signal in the temporal attention to be at least as clean in such models, since our experiments on three backbones with different latent designs (Sec. 5.2) show that the same correspondence structure emerges across representations. The natural testbed is a structured-latent 4D backbone such as Helix4D [91], which extends TRELLIS-2 to video-conditioned 4D mesh generation with first-frame-anchored cross-frame attention; however, no implementation of such a model is publicly available at the time of writing, so we leave this extension to future work.

I Topology-Preserving Animation Details

This section provides the mathematical details for the temporal smoothing and mesh deformation steps introduced in Section 4.2.

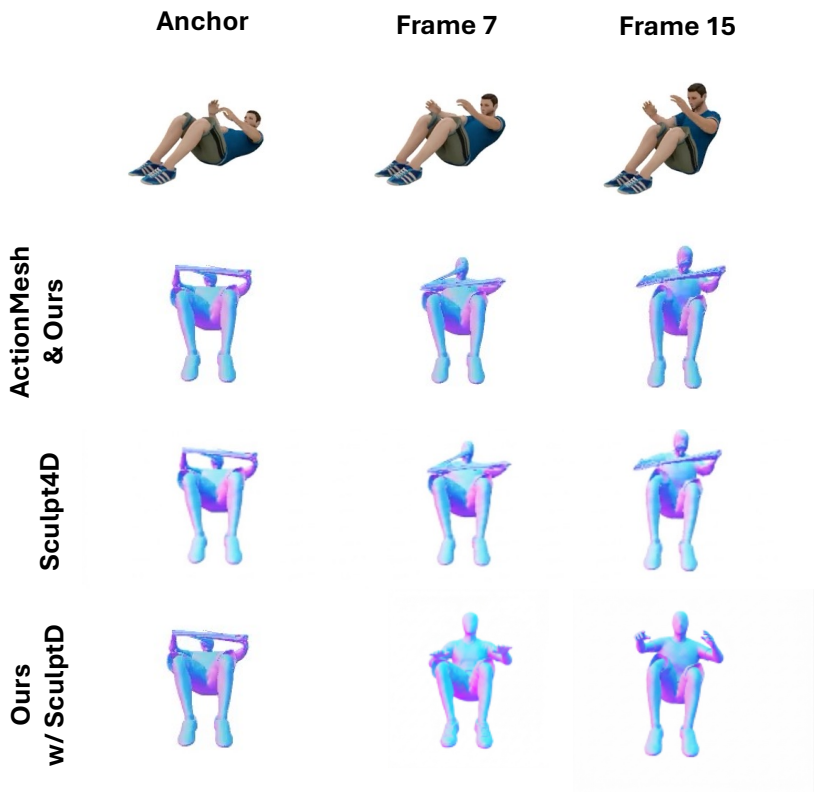


Figure 9: **Anchor-mesh artifacts: failure and recovery.** *Top row:* input video (anchor frame, frames 7 and 15). The image-to-3D anchor (left column) fuses the subject's forearms into a single bar. *Second row:* with the ActionMesh denoiser the artifact is preserved, so ActionMesh's Stage II and our chain produce the same failed result (ours $13\times$ faster). *Third row:* Sculpt4D's own output also retains the artifact in every frame. *Fourth row:* our chain on the frozen Sculpt4D denoiser, which does not support the artifact, assigns it low confidence and does not propagate it; the arms follow the input video.

I.1 1D Gaussian Temporal Smoothing

The raw trajectories of the control landmarks, extracted from the attention chain, are denoted by $\tilde{\mathbf{v}}_\ell^{(f)}$ for landmark ℓ at frame f. To reject physically implausible predictions, we apply an outlier filtering step. Specifically, we flag a landmark's displacement as an outlier if its magnitude exceeds a relative threshold ($5\times$ the median displacement magnitude of the landmarks in that frame) or an absolute distance threshold. We assign a confidence score $c_\ell^{(f)}$ to each prediction, where $c_\ell^{(f)} = 0$ if the displacement is flagged as an outlier, and $c_\ell^{(f)} = 1$ otherwise.

Motion floor and localized motion. The median in the relative threshold is computed over the *non-negligible* displacements only: displacements whose magnitude falls below a configurable motion floor $\delta_{\min}$ are excluded from the median computation. Without this, a mostly static object with a single fast-moving part (e.g., a humanoid moving only one hand) would have a near-zero median, and the valid motion of the moving part would be discarded as an outlier. In all experiments of this paper the motion was sufficiently distributed across the object that we could set $\delta_{\min} = 0$; for domains with highly localized motion, $\delta_{\min}$ can be increased accordingly. Note also that an outlier flag is *per frame*: the flagged landmark keeps its trajectory, its position in that frame is interpolated from neighbouring reliable frames by the confidence-weighted smoothing below, and the landmark is removed from the control set only if it fails the valid-frame requirement of Appendix C, i.e., if it is flagged in a large fraction of the frames of the sequence.

To bridge the gaps caused by rejected outliers and ensure fluid motion, we apply a confidence-weighted 1D Gaussian temporal smoothing independently to each landmark's trajectory. Crucially, we smooth the *displacements* from the anchor pose rather than the absolute positions. This ensures that stationary landmarks do not artificially drift over time.

The smoothed position $\hat{\mathbf{v}}_\ell^{(f)}$ for landmark ℓ at frame f is computed as:

$$\hat{\mathbf{v}}_\ell^{(f)} = \mathbf{v}_\ell^a + \frac{\sum_j G_\sigma(f,j)\, c_\ell^{(j)} \left(\tilde{\mathbf{v}}_\ell^{(j)} - \mathbf{v}_\ell^a\right)}{\sum_j G_\sigma(f,j)\, c_\ell^{(j)} + \epsilon},$$

where $\mathbf{v}_\ell^a$ is the position of the landmark in the anchor mesh, $G_\sigma(f,j) = \exp\left(-\frac{(f-j)^2}{2\sigma^2}\right)$ is a 1D Gaussian kernel with standard deviation σ (in frames), and ϵ is a small constant for numerical stability. The anchor frame itself is pinned to its original geometry to prevent any smoothing leakage.

I.2 Local Rigid Deformation

To propagate the smoothed landmark motions to the dense mesh, we use a local rigid deformation formulation related to As-Rigid-As-Possible (ARAP) surface modeling [65]. Standard linear blend skinning interpolates displacement vectors, which famously causes volume loss when regions undergo rotation. By instead solving for a local rigid transformation (rotation and translation) for each vertex, we preserve the local volume of the mesh.

For each free vertex v on the anchor mesh, we identify its K_{nn} geodesically closest landmarks, denoted as the neighborhood $\mathcal{N}_v$. We assign a Gaussian weight to each landmark $\ell \in \mathcal{N}_v$ based on its geodesic distance $d_{v\ell}$ on the anchor mesh:

$$w_{v\ell} = \exp\left(-\frac{d_{v\ell}^2}{2\sigma_{geo}^2}\right),$$

where σ_{geo} is a scaling factor proportional to the mean spacing between landmarks. The weights are normalized such that $\sum_{\ell\in\mathcal{N}_v} w_{v\ell} = 1$. Using geodesic distances ensures that the deformation respects the surface topology and articulations (e.g., preventing a landmark on the torso from inappropriately influencing the arm).

For each frame f, we solve a weighted Procrustes alignment to find the optimal rotation $R_v^{(f)} \in SO(3)$ that maps the anchor landmark positions to their smoothed target positions:

$$R_v^{(f)} = \arg\min_{R\in SO(3)} \sum_{\ell\in\mathcal{N}_v} w_{v\ell} \left\| R(\mathbf{v}_\ell^a - \boldsymbol{\mu}_v^a) - (\hat{\mathbf{v}}_\ell^{(f)} - \boldsymbol{\mu}_v^{(f)}) \right\|_2^2,$$

where $\boldsymbol{\mu}_v^a$ and $\boldsymbol{\mu}_v^{(f)}$ are the weighted centroids of the neighborhood in the anchor and target frames, respectively:

$$\boldsymbol{\mu}_v^a = \sum_{\ell\in\mathcal{N}_v} w_{v\ell}\mathbf{v}_\ell^a, \qquad \boldsymbol{\mu}_v^{(f)} = \sum_{\ell\in\mathcal{N}_v} w_{v\ell}\hat{\mathbf{v}}_\ell^{(f)}.$$

Once the optimal rotation $R_v^{(f)}$ is found via Singular Value Decomposition (SVD), we apply the rigid transformation to the vertex's anchor position to obtain its final animated position:

$$\hat{\mathbf{v}}_v^{(f)} = R_v^{(f)}(\mathbf{v}_v^a - \boldsymbol{\mu}_v^a) + \boldsymbol{\mu}_v^{(f)}.$$

Applying this transformation to all vertices yields the final animated mesh $\hat{\mathcal{M}}_f = (\hat{\mathbf{V}}^{(f)}, \mathcal{F}_a)$.

I.3 User Study Details

To complement the geometric metrics reported in Sec. 5.1, we conducted a perceptual user study that directly compares our 4D mesh generation against ActionMesh [59], the strongest baseline in this setting. We randomly sampled 20 clips for the study and for each clip, both methods were run with their default settings on the same input frames; the resulting 4D meshes were rendered from the input camera and assembled into a single side-by-side video that places the two renderings next to the

reference input. Each trial shows this triplet to the rater, with the left/right ordering of the two methods randomised per trial to remove positional bias, and asks: *"Which result better matches the reference video in terms of appearance and motion consistency (i.e., fewer temporal mesh distortions)?"* The rater must pick exactly one of the two options. We recruited 100 independent raters, each judging all 20 clips, which yields $2{,}000$ pairwise comparisons in total. Across this entire pool, 75% of the judgments favour our method, and the preference is consistent across clips and categories. This perceptual margin mirrors the quantitative findings of Tab. 1: our renderings exhibit sharper geometry, more accurate silhouette alignment with the input view, and noticeably fewer temporal jitters and mesh distortions than ActionMesh.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Abstract/intro claims match experiments (13x speedup, quality gains, long-sequence scaling, tracking)

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the method's limitations.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: No formal theorems or proofs;

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Full method description, all hyperparameters, backbone details, and evaluation protocols provided.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: Code and Datasets will be released upon acceptance.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Datasets, metrics, GPU type (A100), all hyperparameters from single grid search, ablations provided.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: All methods were evaluated with the same seed, thus the method is deterministic.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: A100 GPU specified; detailed per-component timing breakdown in appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We use public datasets/models, user study posed no risk

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: No social impact.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: No new models or scraped datasets released.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All models and datasets properly cited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: No new datasets or models released.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: User study details provided in appendix.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Low-risk perceptual preference task viewing synthetic renderings.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: No LLMs used in the core methodology.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.