FLOWRIG: UNIFIED SKELETON GENERATION FOR 3D AND 4D MESHES VIA FLOW MATCHING

Anonymous authors
Paper under double-blind review

ABSTRACT

Automatic rigging is a fundamental task for both static 3D and dynamic 4D mesh animation, yet existing methods do not provide a unified feedforward solution for these two settings. While static 3D riggers achieve strong quality, they often provide limited control over bone count and become temporally unstable when applied frame by frame to dynamic meshes. Existing 4D methods, in contrast, typically rely on expensive per-sequence optimization. We propose FLOWRIG, a template-free and data-efficient framework for unified 3D and 4D skeleton generation. Built on flow matching, FLOWRIG generates temporally aligned skeletons for an entire sequence in a single feedforward pass, while naturally handling single-mesh 3D rigging. The model further supports explicit skeleton-count control, uncertainty estimation for candidate selection, and masking-based interpolation, forecasting, and backcasting. To address limited paired 4D rigging data, we adopt a curriculum that initializes from large-scale 3D rigging data and adapts to mid-scale 4D mesh-skeleton data. Experiments show that FLOWRIG achieves strong static 3D rigging performance with controllable bone complexity, as well as high-quality and temporally consistent 4D reconstruction and prediction results.

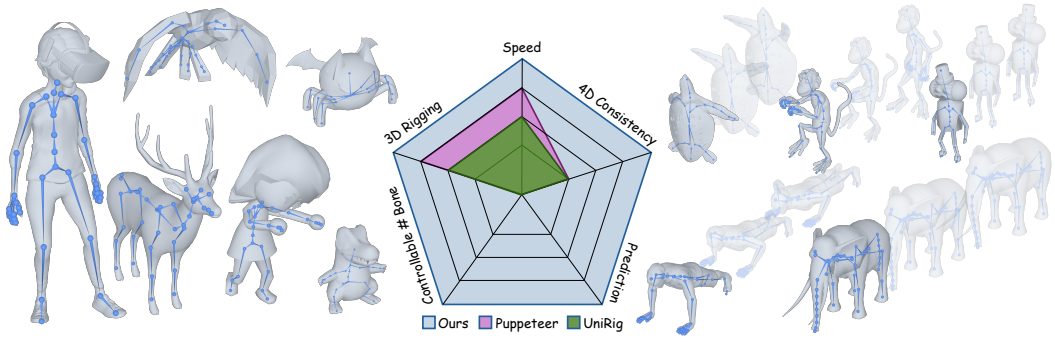


Figure 1: FLOWRIG unifies rigging for both 3D and 4D meshes. Compared with prior methods such as Puppeteer (Song et al., 2025a) and UniRig (Zhang et al., 2025a), our model delivers accurate 3D rigging, preserves 4D temporal consistency, supports controllable bone count, achieves faster inference, and enables 4D skeleton prediction, including interpolation, forecasting, and backcasting.

1 INTRODUCTION

Automatic rigging is a core problem in computer graphics because it turns raw geometry into animatable assets. This need spans both static 3D and dynamic 4D meshes: 3D assets require high-quality, controllable skeletons for downstream animation workflows, while 4D assets additionally require temporal consistency across time. At the same time, temporally coherent 4D meshes are becoming increasingly available from modern reconstruction and generation pipelines. Recent systems can recover or synthesize dynamic 4D geometry directly from videos or generative models, making 4D meshes a practical representation rather than a niche output (Sabathier et al., 2026; Zhang et al., 2025b; Xie et al., 2026). However, geometry alone is not enough for downstream animation,

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

editing, simulation, and control. To make these assets truly useful, we need a unified feedforward rigging model that works for both 3D and 4D meshes, produces informative skeletons, supports controllable bone complexity, and remains stable across diverse poses and motions.

Unfortunately, this capability is still missing. On one side, existing rigging methods are predominantly designed for static 3D inputs, ranging from classical optimization systems such as Pinocchio (Baran & Popović, 2007) to modern feedforward and autoregressive models such as RigNet, UniRig, RigAnything, ARMO, and Puppeteer (Xu et al., 2020; Zhang et al., 2025a; Liu et al., 2025a; Sun et al., 2025a; Song et al., 2025a;b; Zhang et al., 2026a). These methods have substantially improved 3D rigging quality and efficiency, but they are fundamentally static, and many of them provide limited control over skeleton density or bone count for downstream applications. When applied frame by frame to dynamic meshes, they also typically fail to maintain temporal consistency (Wang et al., 2026b). On the other hand, existing 4D methods mainly focus on per-sequence optimization, reconstruction, or motion transfer (Li et al., 2025; Yao et al., 2025b; Wang et al., 2026a; Dabhi et al., 2026; Muralikrishnan et al., 2025; Kim et al., 2026). While these methods are valuable, they do not provide a unified feedforward rigging solution that bridges controllable 3D rigging, dynamic 4D reconstruction, and predictive temporal reasoning within one framework.

Building such a unified model is challenging for two reasons. First, many strong static rigging methods are autoregressive, which makes explicit bone-count control less natural and global temporal reasoning inefficient for dynamic sequences. Second, paired 4D mesh-skeleton data remains far scarcer than static 3D rigging data, so a practical model must learn temporal consistency from limited 4D supervision while preserving strong 3D rigging quality.

In this work, we propose FLOWRIG, a data-efficient feedforward framework for unified skeleton generation on 3D and 4D meshes. Built on flow matching, FLOWRIG predicts temporally aligned skeletons for an entire sequence in one pass, while naturally handling 3D rigging with explicit skeleton-count control. The model also includes a graph head for temporally consistent topology prediction and an uncertainty head for confidence estimation. The masking-based training further enables 4D prediction including interpolation, forecasting, and backcasting. To address limited 4D supervision, we initialize from large-scale 3D rigging data and then adapt to mid-scale 4D rigging data. Experiments show that FLOWRIG achieves strong static 3D rigging performance with controllable bone complexity, as well as high-quality, temporally consistent 4D reconstruction and prediction.

Our main contributions are as follows:

- We propose FLOWRIG, a unified flow-matching model for controllable 3D/4D skeleton generation.
- We introduce a 3D-to-4D curriculum for data-efficient training under limited 4D supervision.
- We achieve strong 3D rigging, high-quality 4D reconstruction, and flexible 4D prediction.

2 RELATED WORK

Static Automatic Rigging and Skinning. Automatic rigging has been studied extensively in the static 3D setting. Early methods relied on template fitting and per-shape optimization, as in Pinocchio (Baran & Popović, 2007), while later data-driven approaches improved both speed and generalization. RigNet (Xu et al., 2020) introduced graph neural networks for direct skeleton and skinning prediction from meshes. Subsequent methods such as UniRig (Zhang et al., 2025a), RigAnything (Liu et al., 2025a), ARMO (Sun et al., 2025a), Puppeteer (Song et al., 2025a), and Skin Tokens (Zhang et al., 2026b) further improved structural validity through autoregressive or language-model-style generation (Song et al., 2025b). These methods are strong baselines for static 3D rigging, but not for general-purpose 4D rigging. In particular, autoregressive generation is less suitable for 4D sequences, where one must reason jointly over skeletal slots and time; applying static models per-frame often leads to temporal flickering and inconsistent topologies (Wang et al., 2026b). Their discrete formulation can also make skeleton density control harder for downstream 4D use.

Dynamic and 4D Rigging. As 4D meshes become more accessible, there is growing interest in rigging dynamic assets directly from videos or mesh sequences. Existing 4D rigging-related methods mainly fall into two categories. The first consists of per-sequence optimization methods, such as AKD (Li et al., 2025), RigGS (Yao et al., 2025b), AniMimic (Xie et al., 2026), GaussiAnimate (Wang et al., 2026a), and RAT4D (Dabhi et al., 2026), which align articulated structure to multi-frame observations or dynamic 3D representations. These methods can achieve high fidelity, but their

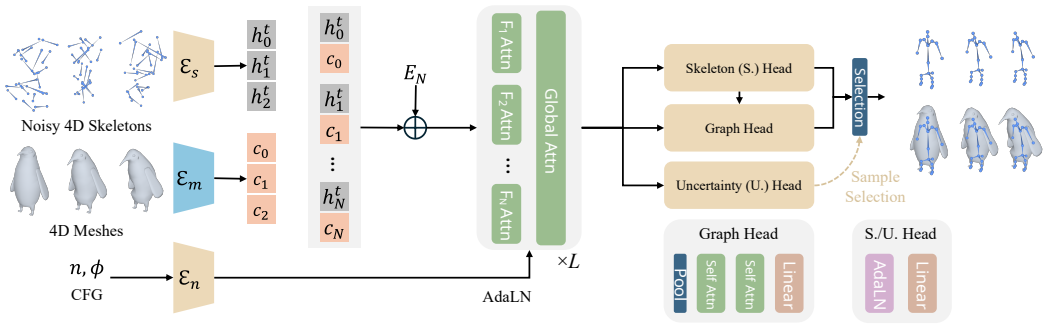


Figure 2: FLOWRIG overview. We encode meshes into conditioning tokens, and initialize noisy skeleton slots, which are processed by a flow matching transformer to predict temporally aligned skeleton representations. The predicted features are decoded by the skeleton, graph, and uncertainty heads to recover 3D/4D rigging outputs, with uncertainty used for sample selection. n denotes the skeleton count, and ϕ denotes the unconditional token.

reliance on per-scene optimization makes them computationally expensive and less suitable for scalable deployment. The second involves consistency-oriented extensions of static riggers, such as SPRig (Wang et al., 2026b), which improve cross-frame consistency through additional self-supervision. While promising, these methods do not yet provide a unified feedforward framework for raw 4D mesh sequences that supports both reconstruction and prediction. Related animation methods, including Gaussian bone representation, zero-shot pose transfer and morphology-aware motion transfer (Guo et al., 2025; Muralikrishnan et al., 2025; Kim et al., 2026; Zhang et al., 2026a), are complementary rather than direct solutions to general-purpose 4D rigging.

4D Mesh Availability and Upstream Geometry Models. Our work is motivated in part by the increasing availability of temporally coherent 4D meshes from upstream reconstruction and generation systems. The field has seen rapid expansion, ranging from text- and image-to-4D generation (Yin et al., 2023; Ling et al., 2024; Bahmani et al., 2024a; Miao et al., 2024; Yuan et al., 2024; Chen et al., 2024; Zeng et al., 2024a; Bahmani et al., 2024b; Sun et al., 2025b; Lin et al., 2025a;b; Zhao et al., 2023a; Gao et al., 2024; Wang et al., 2024; Dai et al., 2025; Nag et al., 2025) to video-based dynamic reconstruction (Jiang et al., 2024; Zeng et al., 2024b; Wu et al., 2024; Zhang et al., 2024b;a; Li et al., 2024; Fu et al., 2024; Zhu et al., 2026; Yao et al., 2025a; Xie et al., 2025; Chen et al., 2025; Ren et al., 2024; 2023; Liang et al., 2024). In particular, recent methods such as ActionMesh (Sabathier et al., 2026) and AniMimic (Xie et al., 2026) show that high-quality dynamic geometry can now be derived from monocular videos and learned priors. While these systems expand the supply of dynamic geometry, they do not solve the downstream rigging problem. Instead, they highlight the need for a feedforward 4D rigging model that converts temporally aligned geometry into informative and animation-ready skeletons. In this sense, our method is complementary: upstream models provide the dynamic geometry, while FLOWRIG provides the rigging for downstream animation and control.

Generative Models for 3D and 4D Assets. A separate line of work studies the co-generation of geometry and rigging information. AniGen (Huang et al., 2026) generates animatable 3D assets together with rigging information, while AKD (Li et al., 2025) uses score distillation to generate articulated 4D assets from images or text. Related directions include skeleton-guided 3D generation (Xu et al., 2025), creative articulated asset generation (Lu et al., 2026), sketch-based rigging (Zhao et al., 2026), and rig-based generative synthesis of dynamic 3D Gaussians (Wu et al., 2026). However, these works are not for general-purpose feedforward rigging of raw 4D mesh sequences. Instead, our focus is to take 3D or 4D geometry as input and produce temporally consistent skeletons with controllable complexity and prediction capability, making our setting complementary to asset generation.

3 PROPOSED METHOD

3.1 PROBLEM FORMULATION

Let $\mathcal{M}_{1:N} = (m_1, \ldots, m_N)$ denote a observation sequence of mesh m_i, where each frame is represented as a sampled point cloud with normals $P_i \in \mathbb{R}^{N_p \times 6}$ and N_p is the size of the point cloud. Our goal is to predict a corresponding skeleton sequence $\mathcal{S}_{1:N} = (S_1, \ldots, S_N)$, where each S_i is a set of bones with 3D endpoints and a tree topology. We consider static and dynamic tasks below.

Static skeleton reconstruction. When $N = 1$, we generate a skeleton set S_1 conditioned on a single mesh observation P_1. Formally, we model the conditional distribution $p(S_1 \mid P_1)$.

Dynamic skeleton reconstruction and prediction. Given observed frames $\mathcal{O} \subseteq \{1, \ldots, N\}$ and observations $\mathcal{P}_\mathcal{O} = \{P_i\}_{i \in \mathcal{O}}$, we generate the full skeleton sequence using $p(\mathcal{S}_{1:N} \mid \mathcal{P}_\mathcal{O})$. Reconstruction corresponds to $\mathcal{O} = \{1, \ldots, N\}$, while prediction corresponds to $\mathcal{O} \subset \{1, \ldots, N\}$.

3.2 Flow Matching on An Optimal-Transport Path

Instead of learning a reverse denoising process as in diffusion models, flow matching learns a continuous-time vector field that transports noise to data. Let $\mathbf{x}_0$ denote a clean sample and $\mathbf{x}_1 \sim \mathcal{N}(0, \sigma^2 I)$ be Gaussian noise. We use the optimal-transport straight-line path $\mathbf{x}_t = (1-t)\mathbf{x}_0 + t\mathbf{x}_1$ for $t \in (0, 1)$, whose target velocity is constant: $\mathbf{v}^\star(\mathbf{x}_t, t) = \mathbf{x}_1 - \mathbf{x}_0$. Given mesh-derived conditioning tokens $\mathbf{c}$, the network $\mathcal{M}$ predicts the velocity field $\hat{\mathbf{v}} = \mathcal{M}(\mathbf{x}_t, t, \mathbf{c}; \Theta)$ and is trained to match $\mathbf{v}^\star$ with mean-squared error, together with our matching and auxiliary losses. At inference time, we start from $\mathbf{x}_{t=1} \sim \mathcal{N}(0, \sigma^2 I)$ and integrate the learned ODE from $t = 1$ to $t = 0$ with an Euler solver: $\mathbf{x}_{t-\Delta t} = \mathbf{x}_t - \Delta t \cdot \hat{\mathbf{v}}(\mathbf{x}_t, t, \mathbf{c})$, yielding the clean sample $\mathbf{x}_0$.

3.3 4D Skeleton Flow Matching Model

Figure 2 shows an overview of FlowRig. Given a static mesh or a mesh sequence, we encode each frame with the frozen Michelangelo encoder (Zhao et al., 2023b) to obtain mesh conditioning tokens $\mathbf{c} \in \mathbb{R}^{N \times L_m \times H}$. In parallel, we initialize a fixed set of noisy skeleton slots $\mathbf{x}_t \in \mathbb{R}^{N \times L_s \times 7}$ and optionally inject a target skeleton-count condition to control the number of generated bones. We set $L_s = 128$ to cover all training instances, similar to prior slot-based methods such as Puppeteer (Song et al., 2025a), and allow N to vary by sampling different window lengths during training. The DiT embeds the skeleton slots into H-dimensional tokens, concatenates them with the mesh tokens, and processes the resulting sequence of length $N \cdot (L_s + L_m)$ with temporal rotational positional encoding (RoPE) using the frame index as the shared temporal reference. It then performs slot-based set prediction over the full sequence and predicts a velocity tensor $\hat{\mathbf{v}} \in \mathbb{R}^{N \times L_s \times 7}$, which is decoded into up to L_s bones per frame as 7D slot states $\mathbf{x}_0 \in \mathbb{R}^{N \times L_s \times 7}$, where each slot is $\mathbf{x}_{0,i,j} = [e_{i,j}, \mathbf{p}_{i,j,0}, \mathbf{p}_{i,j,1}]$ with an existence indicator $e \in \{0, 1\}$ and two 3D endpoints $\mathbf{p}_{i,j,0}, \mathbf{p}_{i,j,1} \in \mathbb{R}^3$. After the transformer predicts the aligned skeleton slots, a graph head infers the kinematic tree and an uncertainty head estimates confidence, producing the final controllable 3D or 4D rigging output.

Besides the skeleton head, we use two additional lightweight heads: a graph head and an uncertainty head. A naive way to build a skeleton tree is to connect bones by distance, but this can lead to unstable topology over time as the joints move. Instead, the graph head predicts parent indices from the hidden features $\{\mathbf{h}_i\}$ and the predicted skeletons $\{\hat{\mathbf{x}}_i\}$, yielding a more temporally consistent tree structure. The uncertainty head takes the hidden features as input and predicts per-frame per-bone uncertainty $\hat{\sigma} \in \mathbb{R}^{N \times L_s \times 7}$. Intuitively, it estimates how reliable each generated bone is, so we can use uncertainty as a score when drawing multiple samples and select the most confident result.

Skeleton-count conditioning and classifier-free guidance (CFG). In addition to mesh tokens, we condition the model on a scalar skeleton or bone count y, which serves as a guidance label. During training we apply classifier-free guidance by randomly dropping the condition with probability p_{drop}, that is, replacing y with an unconditional token $y = -1$ in the embedding layer. The conditional and unconditional predictors share parameters and differ only in the presence of y:

$$\hat{\mathbf{v}}_{\text{cond}} = \mathcal{M}(\mathbf{x}_t, t, \mathbf{c}, y; \Theta), \qquad \hat{\mathbf{v}}_{\text{uncond}} = \mathcal{M}(\mathbf{x}_t, t, \mathbf{c}, -1; \Theta). \tag{1}$$

At inference time we form the guided velocity with CFG scale s as $\hat{\mathbf{v}}_{\text{cfg}} = \hat{\mathbf{v}}_{\text{uncond}} + s \cdot \big(\hat{\mathbf{v}}_{\text{cond}} - \hat{\mathbf{v}}_{\text{uncond}}\big)$, and integrate the ODE with $\hat{\mathbf{v}}_{\text{cfg}}$.

3.4 Training Objective

Our training objective combines flow matching with the auxiliary supervision terms.

One-to-one supervision via sequence Hungarian matching. To handle variable ground-truth bone counts while preserving a fixed tensor interface, we follow the DETR-style one-to-one supervision (Carion et al., 2020). Each S_i is represented using L_s permutation-invariant skeleton slots, and the

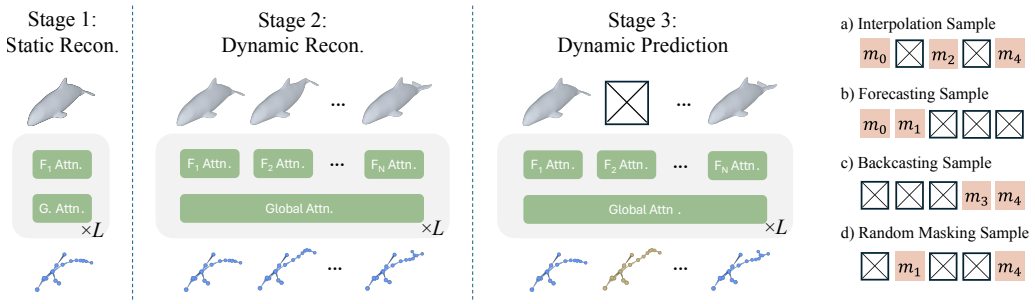


Figure 3: Data-efficient curriculum training. We first train on static 3D rigging, then adapt to 4D reconstruction, and finally enable prediction through masked conditioning. During the final stage, we use diverse masking patterns, including interpolation, forecasting, backcasting, and random masking, to expose the model to different temporal prediction settings. This schedule transfers large-scale 3D supervision to the data-scarce 4D setting.

model is trained with hungarian matching for each predicted and ground-truth skeleton pair. For each frame we extract raw endpoints $\mathbf{g}_i \in \mathbb{R}^{N_{\text{gt}} \times 6}$ and pad them to a fixed tensor size. Since slot indices are permutation-invariant, we compute a single assignment per sequence, shared across frames, using the Hungarian algorithm. We first recover a predicted clean state by

$$\hat{\mathbf{x}}_0 = \mathbf{x}_t - t\,\hat{\mathbf{v}} \in \mathbb{R}^{N \times L_s \times 7}, \tag{2}$$

and build a cost matrix between slot endpoints $\hat{\mathbf{p}}_{j,0}$ and $\hat{\mathbf{p}}_{j,1}$ and the ground-truth endpoints $\mathbf{g}_j$. To align the static and dynamic training objectives, we set the anchor frame $i = 1$ for hungarian matching in all training stages. The matching cost is a weighted sum of endpoint Euclidean distance. The resulting assignment maps each ground-truth bone to one slot, producing matched targets $\tilde{\mathbf{x}}$.

Flow matching loss. The primary objective is a weighted flow-matching mean-squared error on the matched optimal-transport velocity:

$$\mathcal{L}_{\text{FM}} = \frac{1}{NL_s} \sum_{i,j} w_{i,j} \left\| \hat{\mathbf{v}}_{i,j} - \tilde{\mathbf{v}}^{\star}_{i,j} \right\|_2^2, \qquad \tilde{\mathbf{v}}^{\star} = \mathbf{x}_1 - \tilde{\mathbf{x}}_0. \tag{3}$$

where per-slot weights $w \in \mathbb{R}^{N \times L_s}$ that upweight matched slots and control supervision on empty slots with less focus. Intuitively, we set the unmatched slot endpoints with zero weight since they have no contribution to the final results and can make the model focus on the matched pairs.

Uncertainty loss. To quantify the confidence of skeletons, we add a Gaussian negative log-likelihood (NLL) term to estimate the uncertainty of bone existence and the joint positions. Following ID-DPM (Nichol & Dhariwal, 2021), we stop gradients through the mean prediction in the NLL branch:

$$\mathcal{L}_{\text{NLL}} = \frac{1}{NL_s} \sum_{i,j} w_{b,j} \Big(\left\| \text{sg}(\hat{\mathbf{v}}_{b,i,j}) - \tilde{\mathbf{v}}^{\star}_{b,i,j} \right\|_2^2 / (\hat{\boldsymbol{\sigma}}^2_{v,b,i,j} + \epsilon) + \log \left(\hat{\boldsymbol{\sigma}}^2_{v,b,i,j} + \epsilon \right) \Big), \tag{4}$$

where ϵ is a small value and $\text{sg}(\cdot)$ denotes the stop-gradient operator. This way, $\mathcal{L}_{NLL}$ can guide $\hat{\sigma}^2$ without affecting the training of transformer backbone and skeleton head.

Graph head loss. To build the kinematic tree for the generated skeleton, we supervise a parent index classifier over slots. To handle temporal aggregation and derive a temporal-consistent tree, we pool time by averaging to obtain a single parent distribution per slot for geometry $\bar{\mathbf{x}}_0 = \frac{1}{N} \sum_{i=1}^{N} \hat{\mathbf{x}}_{0,i}$ and high-level context $\bar{\mathbf{h}} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{h}_i$, we stop gradients from the parent branch into the DiT backbone and concatenate them as the input of the graph head $\mathbf{z} = \text{concat}\big(\text{sg}(\bar{\mathbf{h}}), \text{sg}(\bar{\mathbf{x}}_0)\big) \in \mathbb{R}^{L_s \times (H+7)}$. The graph head G will output $\mathbf{p} = G(\mathbf{z}) \in \mathbb{R}^{L_s \times (L_s+2)}$ where the additional two channels are used to represent the root and non-existing classes. Targets are constructed from the ground-truth joint graph and the Hungarian assignment, including root, non-root, and non-existing classes. The loss is

$$\mathcal{L}_{\text{graph}} = \frac{1}{L_s} \sum_{j} \text{CE}(\mathbf{p}_{b,j,:}, y^{\text{parent}}_{b,j}) \cdot (1-t)^{\gamma_p}, \tag{5}$$

where γ_p is a hyperparameter to control the denosing-step-aware weight $(1-t)^{\gamma_p}$ in flow matching. Intuitively, small t means the noisy state is still close to a coherent skeleton, so parent index targets

(via hungarian matching on $\hat{\mathbf{x}}_0$) are more reliable while large t means slots are dominated by noise, so downweighting parent CE reduces conflicting topology gradients. The exponent $\gamma_p \geq 0$ controls how sharply supervision is concentrated toward $t \approx 0$.

Total objective. The final objective is a weighted sum:

$$\mathcal{L} = \mathcal{L}_{\text{FM}} + \lambda_{\text{NLL}}\mathcal{L}_{\text{NLL}} + \lambda_{\text{graph}}\mathcal{L}_{\text{graph}} + \lambda_{\text{aux}}\mathcal{L}_{\text{aux}}, \tag{6}$$

where λ are the coefficients for each term and $\mathcal{L}_{\text{aux}}$ is used to regularize the geometry and temporal smoothness (defined in the Appendix B). Overall, the losses in our framework serve three purposes: (1) $\mathcal{L}_{\text{FM}}$ learns accurate skeleton generation through flow matching, (2) $\mathcal{L}_{\text{NLL}}$ and $\mathcal{L}_{\text{graph}}$ improve confidence estimation and kinematic topology prediction, and (3) $\mathcal{L}_{\text{aux}}$ regularizes geometric validity and temporal consistency. Together, these terms improve stability and produce controllable, temporally aligned skeletons for both 3D and 4D inputs.

3.5 Curriculum Training from 3D Reconstruction to 4D Prediction

We train the model with a curriculum progressively increasing temporal complexity while keeping the architecture fixed. As shown in Figure 3, we start with static single-frame skeleton generation initializing the model with a strong geometry prior. We then switch to dynamic reconstruction, training on different time windows with full guidance of mesh. Finally, we emphasize dynamic prediction by randomly masking mesh tokens to force the model to simulate missing observations.

4 Experiments

We evaluate FlowRig across diverse rigging scenarios, focusing on its ability to maintain 4D consistency and performance as a static rigger. Unless otherwise specified, we use unconditional guidance at inference time, since the ground-truth bone count is unavailable at test time.

4.1 Experimental Setup

Datasets. To ensure a comprehensive evaluation, we utilize three primary data sources spanning static and dynamic domains. (1) For static rigging, we follow the protocol established in Puppeteer (Song et al., 2025a) and employ Articulation-XL (Song et al., 2025b), a massive collection of 33,000 high-quality rigged 3D models covering humanoids, animals, and mechanical objects. (2) We also evaluate on the classic ModelsResource dataset (Xu et al., 2020), which contains a curated set of character models with professionally designed skeletons, serving as a standard benchmark for topological accuracy. (3) For dynamic 4D rigging, we utilize a curated dynamic dataset derived from TexVerse (Zhang et al., 2025b). We manually filtered the TexVerse animated subset to retain around 2,500 object-centric instances where the skeleton and mesh remain strictly aligned across temporal sequences, providing a high-fidelity ground truth for evaluating consistency and predictive forecasting. More details about the curated dataset can be found in Appendix A.

Evaluation Metrics. We evaluate skeleton quality with three Chamfer Distance (CD) variants: Joint-to-Joint (J2J), Joint-to-Bone (J2B), and Bone-to-Bone (B2B), which measure joint accuracy, joint-to-bone alignment, and structural fidelity, respectively. For 4D tasks, we average these over time. We also evaluate temporal consistency using bone distance deviation (BDD), which measures temporal jitter in bone lengths; Graph Structural Distance (GSD), which measures structural consistency through Hamming distance (HD) (Wang et al., 2026b); and bone number deviation (BND), which measures changes in bone count relative to the anchor frame. More details are provided in Appendix C.1.

Baselines. We compare FlowRig against both classical and modern rigging baselines. Classical methods include Pinocchio (Baran & Popović, 2007), a template-fitting optimization-based rigging system, and RigNet (Xu et al., 2020), a graph-neural-network-based feedforward rigger. Among recent generative or transformer-based methods, we consider MagicArticulate (Song et al., 2025b), Anymate (Deng et al., 2025), UniRig (Zhang et al., 2025a), RigAnything (Liu et al., 2025a), and Puppeteer (Song et al., 2025a). These methods cover a broad spectrum from optimization-based pipelines to modern feedforward sequence models. While several of them are strong for static 3D meshes, our results show that FlowRig remains competitive in the static setting while additionally providing the native 4D consistency and predictive capability required for dynamic sequences.

Table 1: **Static skeleton reconstruction comparison.** Our methods achieves strong reconstruction fidelity across diverse out-of-domain static datasets. *Ours** utilizes uncertainty as selection score to select the best candidate from the 5 samples and outperforms all baselines on Articulation-XL 2.0 and Diverse-Pose. On ModelsResource, its zero-shot performance approaches that of RigAnything trained directly on the same dataset. See the "Baselines" paragraph for citations of these models.

Model	Articulation-XL 2.0			ModelsResource			Diverse-Pose		
	J2J ↓	J2B ↓	B2B ↓	J2J ↓	J2B ↓	B2B ↓	J2J ↓	J2B ↓	B2B ↓
Pinocchio	0.0832	0.0661	0.0549	0.0685	0.0482	0.0409	0.0797	0.0641	0.0515
RigNet	0.0762	0.0608	0.0528	0.0722	0.0599	0.0433	0.0775	0.0639	0.0571
MagicArt.	0.0326	0.0250	0.0212	0.0411	0.0314	0.0269	0.0438	0.0346	0.0296
Anymate	0.0409	0.0301	0.0294	0.0370	0.0261	0.0258	0.0342	0.0255	0.0272
UniRig	0.0331	0.0261	0.0218	0.0396	0.0302	0.0257	0.0325	0.0257	0.0208
RigAnyth.	0.0489	0.0379	0.0306	**0.0343**	**0.0244**	**0.0204**	0.0397	0.0308	0.0241
Puppeteer	0.0311	0.0237	0.0198	0.0377	0.0280	0.0241	0.0251	0.0199	0.0160
Ours	0.0264	0.0188	0.0163	0.0358	0.0259	0.0234	0.0252	0.0188	0.0160
Ours*	**0.0259**	**0.0184**	**0.0160**	0.0354	0.0256	0.0233	**0.0245**	**0.0182**	**0.0155**

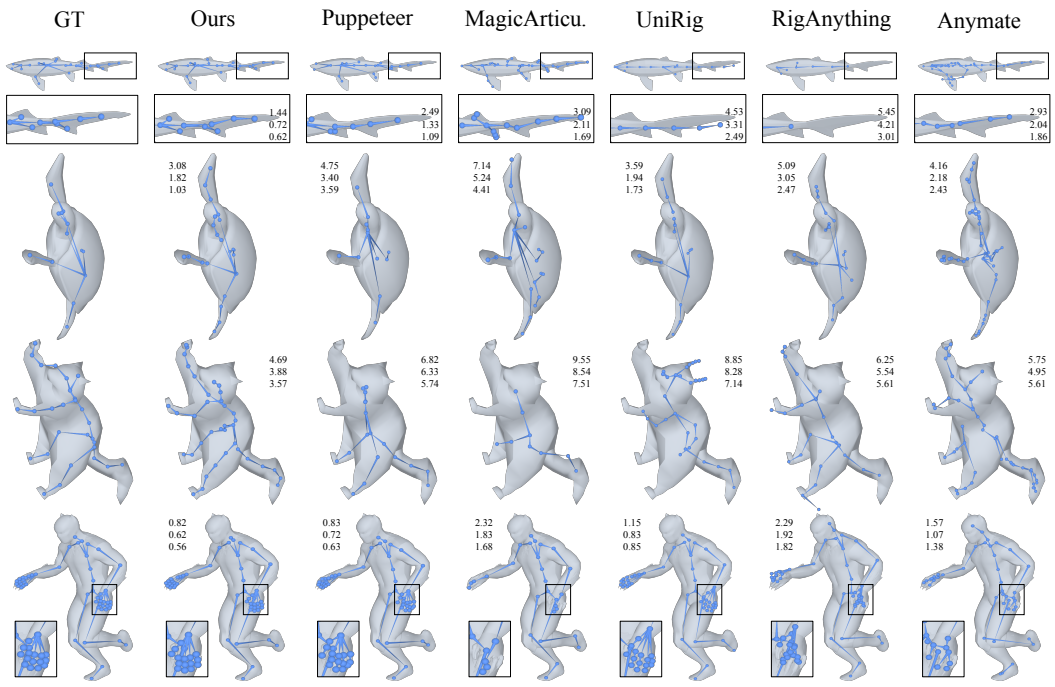
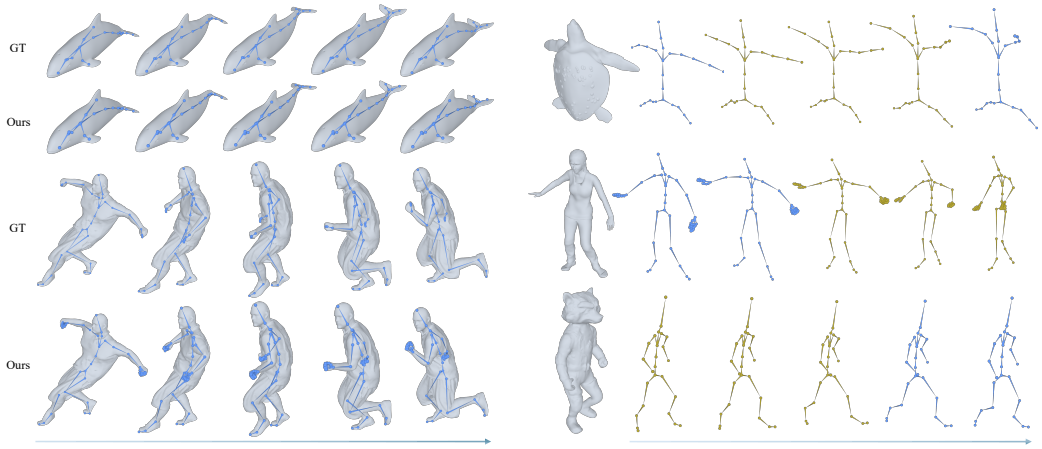


Figure 4: Static Reconstruction. Numbers from top to bottom are J2J, J2B, and B2B for each instance. Compared with baselines, our model preserves finer geometric details and achieves more accurate mesh alignment.

4.2 3D Skeleton Reconstruction

FlowRig supports both static and dynamic meshes, and with a single mesh as input it becomes a strong 3D rigging model. As shown in Table 1, it consistently ranks among the best methods across all three datasets, clearly outperforming classical baselines such as Pinocchio and RigNet. Among recent methods, Puppeteer (Song et al., 2025a) and RigAnything (Liu et al., 2025a) are the strongest competitors, but FlowRig achieves the best overall results on Articulation-XL 2.0 and Diverse-Pose and remains competitive on ModelsResource in the zero-shot setting, even if RigAnything is trained directly on that training dataset. Overall, these results show that our unified 3D/4D formulation preserves strong static rigging performance and improves further with uncertainty-based selection in *Ours**. Figure 4 shows that our method recovers more detailed and better mesh-aligned skeletons compared with baselines; more results are shown in Appendix E and the demo video.



(a) Dynamic Reconstruction (b) Dynamic prediction.

Figure 5: Dynamic reconstruction and prediction. Blue indicates reconstruction w/ mesh guidance, while yellow indicates predictions w/o mesh guidance. In (b), from top to bottom, it shows interpolation, forecasting, and backcasting, respectively. Refer to the demo video for a clearer comparison.

Table 2: Dynamic quantitative comparisons on TexVerse dynamic dataset. N is time window length. *Puppeteer-D** means we finetuned Puppeteer-diverse model (Song et al., 2025a) on TexVerse training dataset. *Ours** utilizes uncertainty as selection score to select the best candidate from the 5 samples.

Method	N	J2J-W ↓	J2B-W ↓	B2B-W ↓	BND ↓	BDD ↓	GSD ↓	HD ↓
Puppeteer-D*	4	0.0161	0.0130	0.0118	2.8279	0.0075	0.1189	6.2173
Puppeteer-D*	8	0.0162	0.0131	0.0118	2.8753	0.0091	0.1575	8.3745
Puppeteer-D*	24	0.0163	0.0132	0.0119	3.1763	0.0150	0.2654	14.1894
Ours	4	0.0161	0.0125	0.0104	0	0.0021	0	0
Ours	8	0.0152	0.0119	0.0099	0	0.0026	0	0
Ours	24	0.0142	0.0112	0.0093	0	0.0038	0	0
Ours*	4	0.0144	0.0112	0.0093	0	0.0019	0	0
Ours*	8	0.0144	0.0113	0.0094	0	0.0025	0	0
Ours*	24	0.0136	0.0107	0.0089	0	0.0037	0	0

4.3 4D Skeleton Reconstruction and Prediction

Our model naturally handles 4D reconstruction and prediction. For dynamic meshes, we study two settings. In 4D reconstruction, the full mesh sequence is observed with the goal of recovering a temporally consistent skeleton for every frame. In 4D prediction, only part of the sequence is observed and the model must infer the missing skeletons for interpolation, forecasting, or backcasting. Table 2 shows that FlowRig consistently outperforms the fine-tuned Puppeteer baseline *Puppeteer-D** on both reconstruction accuracy and 4D consistency. As the temporal window N increases, our model achieves lower J2J-W, J2B-W, and B2B-W errors, where W represents averaging over the time window, indicating more accurate per-frame skeleton reconstruction with stronger temporal context. More importantly, FlowRig maintains perfect structural consistency across the full sequence, with zero BND, GSD, and HD for all reported settings, meaning that the predicted bone count and kinematic topology remain stable over time. In contrast, *Puppeteer-D** exhibits clear temporal degradation as N grows, with increasing BND, BDD, GSD, and HD, indicating both geometric drift and topological flickering. The uncertainty-based selection in *Ours** further improves reconstruction quality while preserving the same structural behavior, showing that FlowRig produces not only accurate per-frame skeletons but also temporally coherent 4D rigs. Figure 5a provides qualitative examples showing that our model preserves 4D consistency over time while also recovering finer skeletal details, such as the structure of the human hand. In addition, FlowRig supports 4D prediction, enabling interpolation, forecasting, and backcasting within a unified framework. Figure 5b presents qualitative evaluations of these settings and demonstrates FlowRig's flexibility for downstream tasks. More qualitative results are shown in Appendix E and the demo video.

Table 3: Ablation studies on data sources. Metrics are evaluated on the TexVerse test set with $T = 24$.

Static	Dynamic	J2J-W ↓	J2B-W ↓	B2B-W ↓	BDD ↓	GSD ↓
✗	✓	0.0213	0.0173	0.0147	0.0049	0
✓	✓	**0.0142**	**0.0112**	**0.0093**	**0.0038**	0

4.4 Further Analysis and Ablation Studies

Efficient Training Curriculum. We apply curriculum training: pre-train on large-scale static assets and fine-tune on dynamic sequences. Table 3 shows that static pre-training improves accuracy and reduces temporal instability compared with training on dynamic data alone. This suggests that large-scale static supervision provides a strong geometric prior for learning skeletal dynamics.

Ablation of the Graph Head. The graph head is introduced to maintain a temporally consistent kinematic hierarchy across the full 4D sequence. We evaluate its effect by replacing it with a simple 1-Nearest Neighbor tree-building heuristic. As shown in Table 4, removing the graph head significantly increases Hamming distance, indicating frequent topological changes and unstable parent assignments over time. The corresponding rise in GSD further shows that the graph head is essential for preserving structural coherence and avoiding topological flickering in dynamic rigging.

Table 4: Ablation studies on graph head. The metrics are evaluated on the TexVerse test set with time window $T = 24$.

Graph Head	GSD ↓	HD ↓
✗	0.0375	2.2844
✓	0	0

Controllable Skeletal Complexity. A unique advantage of our flow-matching framework is the ability to explicitly control skeletal density via Classifier-Free Guidance. While our model can autonomously determine a plausible bone count through unconditional generation, it also supports bone-count conditioning as a user-defined input. This allows FlowRig to synthesize skeletons with arbitrary bone counts ranging from 1 to 128, providing direct and flexible control over the output granularity that is absent in existing autoregressive methods. As illustrated in Figure 6, FlowRig can adapt the skeletal detail to satisfy specific animation requirements, varying from sparse armatures for background characters to high-density rigs for complex manual dexterity, whereas other baselines remain restricted to a single, uncontrolled output density.

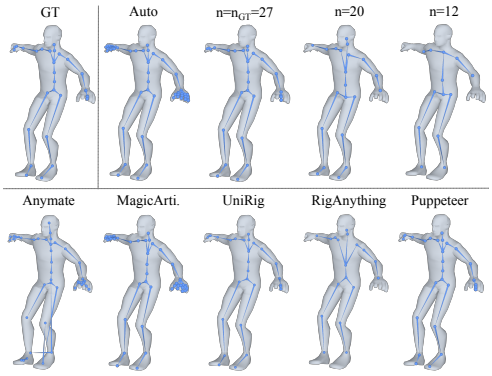


Figure 6: Analysis of bone count control. Our model can freely control the number of bones to adjust skeletal detail, while existing baselines do not provide this level of controllable output granularity.

Additional Notes. We report inference speed in Appendix C.2, where FlowRig is faster than all baselines. We also develop a feedforward skinning model for downstream use in Appendix F, which is competitive with prior baselines.

5 Conclusion and Future Works

We propose FlowRig, a unified framework for 3D and 4D skeleton rigging from static meshes and dynamic mesh sequences. Built on flow matching, FlowRig generates temporally aligned skeletons while maintaining strong static reconstruction quality. By combining mesh-conditioned skeleton generation with a graph head for consistent topology and an uncertainty head for confidence-based sample selection, the model achieves competitive 3D performance and clear gains in 4D consistency and controllability. For future work, we plan to build stronger benchmarks for dynamic prediction under partial observations, extend the framework to jointly predict skinning weights and deformation, and introduce richer user controls such as semantic constraints or text-guided rig specification. We hope FlowRig serves as a step toward a more general and controllable rigger for 3D and 4D content.

AI USE STATEMENT

In this work, we used generative AI tools to assist in curating datasets (see Appendix A and Appendix D.1 for details), to help edit the paper to improve readability, and to assist with coding. We have reviewed all AI-assisted work. For example, 2 authors verified and tested LLM-generated code, and we manually verified all results from the AI assisted dataset curation.

We did not use generative AI tools for any of the required disclosure tasks as defined in the ICLR 2027 AI Policy for Authors. Specifically, we did not use generative AI tools to do any of the following: generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, or interpret results. This was either because we did the task ourselves without AI tool assistance, or the task was not applicable to our work. We take responsibility for the final content of this work, including text and code produced with the aid of generative AI.

ETHICS STATEMENT

Our work advances automatic rigging for both static 3D meshes and dynamic 4D mesh sequences, with potential applications in animation, game production, digital content creation, AR/VR, and human-computer interaction. By improving temporal consistency, controllability, and efficiency, FLOWRIG can reduce the manual effort required to create articulated assets and make rigging tools more accessible to artists and non-expert users.

At the same time, the method could be misused to accelerate the creation of synthetic animated content, including deceptive or misleading media. Our work does not use sensitive personal data or human-subject annotations, but downstream applications should still be deployed responsibly to ensure the appropriate usage.

REPRODUCIBILITY STATEMENT

In this work, we propose FLOWRIG, a unified framework for feed-forward rigging of 3D and 4D meshes using flow matching. Code for this model, its unique training curriculum, and associated weights will be released with the camera-ready version of the paper if it is accepted. Datasets we manually curated from TexVerse (Zhang et al., 2025b) and Objaverse (Deitke et al., 2023) will also be released with the camera-ready version of this paper if it is accepted. Further details regarding the composition of these datasets, the design and training of the rigging and skinning models introduced here, and more experimental results can be found in the Appendix.

REFERENCES

Sherwin Bahmani, Xian Liu, Wang Yifan, Ivan Skorokhodov, Victor Rong, Ziwei Liu, Xihui Liu, Jeong Joon Park, Sergey Tulyakov, Gordon Wetzstein, et al. Tc4d: Trajectory-conditioned text-to-4d generation. In *European Conference on Computer Vision*, pp. 53–72. Springer, 2024a.

Sherwin Bahmani, Ivan Skorokhodov, Victor Rong, Gordon Wetzstein, Leonidas Guibas, Peter Wonka, Sergey Tulyakov, Jeong Joon Park, Andrea Tagliasacchi, and David B Lindell. 4d-fy: Text-to-4d generation using hybrid score distillation sampling. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7996–8006. IEEE, 2024b.

Ilya Baran and Jovan Popović. Automatic rigging and animation of 3d characters. *ACM Transactions on graphics (TOG)*, 26(3):72–es, 2007.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pp. 213–229. Springer, 2020.

Ce Chen, Shaoli Huang, Xuelin Chen, Guangyi Chen, Xiaoguang Han, Kun Zhang, and Mingming Gong. Ct4d: Consistent text-to-4d generation with animatable meshes. *arXiv preprint arXiv:2408.08342*, 2024.

Jianqi Chen, Biao Zhang, Xiangjun Tang, and Peter Wonka. V2m4: 4d mesh animation reconstruction from a single monocular video. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11643–11653. IEEE, 2025.

Mosam Dabhi, Simon Lucey, and László A Jeni. Rat4d: Rig and animate objects without surface templates in 4d. In *2026 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 393–401. IEEE, 2026.

Sisi Dai, Xinxin Su, Boyan Wan, Ruizhen Hu, and Kai Xu. Textmesh4d: High-quality text-to-4d mesh generation. *arXiv e-prints*, pp. arXiv–2506, 2025.

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13142–13153, 2023.

Yufan Deng, Yuhao Zhang, Chen Geng, Shangzhe Wu, and Jiajun Wu. Anymate: A dataset and baselines for learning 3d object rigging. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–10, 2025.

Zhoujie Fu, Jiacheng Wei, Wenhao Shen, Chaoyue Song, Xiaofeng Yang, Fayao Liu, Xulei Yang, and Guosheng Lin. Sync4d: Video guided controllable dynamics for physics-based 4d generation. *arXiv preprint arXiv:2405.16849*, 2024.

Quankai Gao, Qiangeng Xu, Zhe Cao, Ben Mildenhall, Wenchao Ma, Le Chen, Danhang Tang, and Ulrich Neumann. Gaussianflow: Splatting gaussian dynamics for 4d content creation. *arXiv preprint arXiv:2403.12365*, 2024.

Zhiyang Guo, Ori Zhang, Jax Xiang, Alan Zhao, Wengang Zhou, and Houqiang Li. Make-it-poseable: Feed-forward latent posing model for 3d humanoid character animation. *arXiv e-prints*, pp. arXiv–2512, 2025.

Yi-Hua Huang, Zi-Xin Zou, Yuting He, Chirui Chang, Cheng-Feng Pu, Ziyi Yang, Yuan-Chen Guo, Yan-Pei Cao, and Xiaojuan Qi. Anigen: Unified S^3 fields for animatable 3d asset generation. *arXiv preprint arXiv:2604.08746*, 2026.

Yanqin Jiang, Li Zhang, Jin Gao, Weiming Hu, and Yao Yao. Consistent4d: Consistent 360 dynamic object generation from monocular video. In *International Conference on Learning Representations*, volume 2024, pp. 51844–51861, 2024.

Taeyeon Kim, Youngju Na, Jumin Lee, Sebin Lee, Minhyuk Sung, and Sung-Eui Yoon. Morphgs: Morphology-adaptive articulated 3d motion transfer from videos. In *European Conference on Computer Vision*, pp. 210–229. Springer, 2026.

Xuan Li, Qianli Ma, Tsung-Yi Lin, Yongxin Chen, Chenfanfu Jiang, Ming-Yu Liu, and Donglai Xiang. Articulated kinematics distillation from video diffusion models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17571–17581. IEEE, 2025.

Zhiqi Li, Yiming Chen, and Peidong Liu. Dreammesh4d: Video-to-4d generation with sparse-controlled gaussian-mesh hybrid representation. *Advances in Neural Information Processing Systems*, 37:21377–21400, 2024.

Hanwen Liang, Yuyang Yin, Dejia Xu, Hanxue Liang, Zhangyang Wang, Konstantinos N Plataniotis, Yao Zhao, and Yunchao Wei. Diffusion4d: Fast spatial-temporal consistent 4d generation via video diffusion models. *arXiv preprint arXiv:2405.16645*, 2024.

Jiajing Lin, Zhenzhong Wang, Shu Jiang, Yunpeng Gong, Yongjie Hou, Bowen Yu, Lili Qu, and Min Jiang. Phy124: Fast physics-driven 4d content generation from a single image. In *2025 International Conference on Machine Intelligence and Nature-Inspired Computing (MIND)*, pp. 74–79. IEEE, 2025a.

Jiajing Lin, Zhenzhong Wang, Dejun Xu, Shu Jiang, Yunpeng Gong, and Min Jiang. Phys4dgen: Physics-compliant 4d generation with multi-material composition perception. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 10398–10407, 2025b.

Huan Ling, Seung Wook Kim, Antonio Torralba, Sanja Fidler, and Karsten Kreis. Align your gaussians: Text-to-4d with dynamic 3d gaussians and composed diffusion models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8576–8588. IEEE, 2024.

Isabella Liu, Zhan Xu, Wang Yifan, Hao Tan, Zexiang Xu, Xiaolong Wang, Hao Su, and Zifan Shi. Riganything: Template-free autoregressive rigging for diverse 3d assets. *ACM Transactions on Graphics (TOG)*, 44(4):1–12, 2025a.

Minghua Liu, Mikaela Angelina Uy, Donglai Xiang, Hao Su, Sanja Fidler, Nicholas Sharp, and Jun Gao. Partfield: Learning 3d feature fields for part segmentation and beyond. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9704–9715. IEEE, 2025b.

Hexiao Lu, Xiaokun Sun, Zeyu Cai, Hao Guo, Ying Tai, Jian Yang, and Zhenyu Zhang. Muses: Designing, composing, generating nonexistent fantasy 3d creatures without training. *arXiv preprint arXiv:2601.03256*, 2026.

Qiaowei Miao, Jinsheng Quan, Kehan Li, and Yawei Luo. Pla4d: Pixel-level alignments for text-to-4d gaussian splatting. *arXiv preprint arXiv:2405.19957*, 2024.

Sanjeev Muralikrishnan, Niladri Shekhar Dutt, and Niloy J Mitra. Smf: Template-free and rig-free animation transfer using kinetic codes. *arXiv preprint arXiv:2504.04831*, 2025.

Sauradip Nag, Daniel Cohen-Or, Hao Zhang, and Ali Mahdavi Amiri. In-2-4d: Inbetweening from two single-view images to 4d generation. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, pp. 1–12, 2025.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pp. 8162–8171. PmLR, 2021.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Jiawei Ren, Liang Pan, Jiaxiang Tang, Chi Zhang, Ang Cao, Gang Zeng, and Ziwei Liu. Dreamgaussian4d: Generative 4d gaussian splatting. *arXiv preprint arXiv:2312.17142*, 2023.

Jiawei Ren, Kevin Xie, Ashkan Mirzaei, Hanxue Liang, Xiaohui Zeng, Karsten Kreis, Ziwei Liu, Antonio Torralba, Sanja Fidler, Seung W Kim, et al. L4gm: Large 4d gaussian reconstruction model. *Advances in Neural Information Processing Systems*, 37:56828–56858, 2024.

Remy Sabathier, David Novotny, Niloy J Mitra, and Tom Monnier. Actionmesh: Animated 3d mesh generation with temporal 3d diffusion. *arXiv preprint arXiv:2601.16148*, 2026.

Chaoyue Song, Xiu Li, Fan Yang, Zhongcong Xu, Jiacheng Wei, Fayao Liu, Jiashi Feng, Guosheng Lin, and Jianfeng Zhang. Puppeteer: Rig and animate your 3d models. *Advances in neural information processing systems*, 38:72152–72184, 2025a.

Chaoyue Song, Jianfeng Zhang, Xiu Li, Fan Yang, Yiwen Chen, Zhongcong Xu, Jun Hao Liew, Xiaoyang Guo, Fayao Liu, Jiashi Feng, et al. Magicarticulate: Make your 3d models articulation-ready. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15998–16007. IEEE, 2025b.

Mingze Sun, Shiwei Mao, Keyi Chen, Yurun Chen, Shunlin Lu, Jingbo Wang, Junting Dong, and Ruqi Huang. Armo: Autoregressive rigging for multi-category objects. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 7721–7730. IEEE, 2025a.

Qi Sun, Zhiyang Guo, Ziyu Wan, Jing Nathan Yan, Shengming Yin, Wengang Zhou, Jing Liao, and Houqiang Li. Eg4d: Explicit generation of 4d object without score distillation. In *International Conference on Learning Representations*, volume 2025, pp. 30713–30738, 2025b.

Jiaxin Wang, Dongxin Lyu, Zeyu Cai, Zhiyang Dou, Cheng Lin, Anpei Chen, and Yuliang Xiu. Gaussianimate: Reconstruct and rig animatable categories with level of dynamics. *arXiv preprint arXiv:2604.08547*, 2026a.

Ruipeng Wang, Langkun Zhong, and Miaowei Wang. Sprig: Self-supervised pose-invariant rigging from mesh sequences. *arXiv preprint arXiv:2602.12740*, 2026b.

Xinzhou Wang, Yikai Wang, Junliang Ye, Fuchun Sun, Zhengyi Wang, Ling Wang, Pengkun Liu, Kai Sun, Xintong Wang, Wende Xie, et al. Animatabledreamer: Text-guided non-rigid 3d model generation and reconstruction with canonical score distillation. In *European Conference on Computer Vision*, pp. 321–339. Springer, 2024.

Lifan Wu, Ruijie Zhu, Yubo Ai, and Tianzhu Zhang. Skeletongaussian: Editable 4d generation through gaussian skeletonization. *arXiv preprint arXiv:2602.04271*, 2026.

Zijie Wu, Chaohui Yu, Yanqin Jiang, Chenjie Cao, Fan Wang, and Xiang Bai. Sc4d: Sparse-controlled video-to-4d generation and motion transfer. In *European Conference on Computer Vision*, pp. 361–379. Springer, 2024.

Tianyi Xie, Yunuo Chen, Yaowei Guo, Yin Yang, Bolei Zhou, Demetri Terzopoulos, Ying Jiang, and Chenfanfu Jiang. Animimic: Imitating 3d animation from video priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 40266–40276, 2026.

Yiming Xie, Chun-Han Yao, Vikram Voleti, Huaizu Jiang, and Varun Jampani. Sv4d: Dynamic 3d content generation with multi-frame and multi-view consistency. In *International Conference on Learning Representations*, volume 2025, pp. 33421–33441, 2025.

Yuanyou Xu, Zongxin Yang, and Yi Yang. Skdream: Controllable multi-view and 3d generation with arbitrary skeletons. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 314–325. IEEE, 2025.

Zhan Xu, Yang Zhou, Evangelos Kalogerakis, Chris Landreth, and Karan Singh. Rignet: Neural rigging for articulated characters. *arXiv preprint arXiv:2005.00559*, 2020.

Chun-Han Yao, Yiming Xie, Vikram Voleti, Huaizu Jiang, and Varun Jampani. Sv4d 2.0: Enhancing spatio-temporal consistency in multi-view video diffusion for high-quality 4d generation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 13248–13258. IEEE, 2025a.

Yuxin Yao, Zhi Deng, and Junhui Hou. Riggs: Rigging of 3d gaussians for modeling articulated objects in videos. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5592–5601. IEEE, 2025b.

Yuyang Yin, Dejia Xu, Zhangyang Wang, Yao Zhao, and Yunchao Wei. 4dgen: Grounded 4d content generation with spatial-temporal consistency. *arXiv preprint arXiv:2312.17225*, 2023.

Yu-Jie Yuan, Leif Kobbelt, Jiwen Liu, Yuan Zhang, Pengfei Wan, Yu-Kun Lai, and Lin Gao. 4dynamic: Text-to-4d generation with hybrid priors. *arXiv preprint arXiv:2407.12684*, 2024.

Bohan Zeng, Ling Yang, Siyu Li, Jiaming Liu, Zixiang Zhang, Juanxi Tian, Kaixin Zhu, Yongzhen Guo, Fu-Yun Wang, Minkai Xu, et al. Trans4d: Realistic geometry-aware transition for compositional text-to-4d synthesis. *arXiv preprint arXiv:2410.07155*, 2024a.

Yifei Zeng, Yanqin Jiang, Siyu Zhu, Yuanxun Lu, Youtian Lin, Hao Zhu, Weiming Hu, Xun Cao, and Yao Yao. Stag4d: Spatial-temporal anchored generative 4d gaussians. In *European Conference on Computer Vision*, pp. 163–179. Springer, 2024b.

Haiyu Zhang, Xinyuan Chen, Yaohui Wang, Xihui Liu, Yunhong Wang, and Yu Qiao. 4diffusion: Multi-view video diffusion model for 4d generation. *Advances in Neural Information Processing Systems*, 37:15272–15295, 2024a.

Hao Zhang, Di Chang, Fang Li, Mohammad Soleymani, and Narendra Ahuja. Magicpose4d: Crafting articulated models with appearance and motion control. *arXiv preprint arXiv:2405.14017*, 2024b.

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

Jia-Peng Zhang, Cheng-Feng Pu, Meng-Hao Guo, Yan-Pei Cao, and Shi-Min Hu. One model to rig them all: Diverse skeleton rigging with unirig. *ACM Transactions on Graphics (ToG)*, 44(4):1–18, 2025a.

Jia-peng Zhang, Cheng-Feng Pu, Meng-Hao Guo, Yan-Pei Cao, and Shi-Min Hu. Skin tokens: A learned compact representation for unified autoregressive rigging. *arXiv preprint arXiv:2602.04805*, 2026b.

Yibo Zhang, Li Zhang, Rui Ma, and Nan Cao. Texverse: A universe of 3d objects with high-resolution textures. *arXiv preprint arXiv:2508.10868*, 2025b.

Ruisi Zhao, Haoren Zheng, Zongxin Yang, Hehe Fan, and Yi Yang. Stroke3d: Lifting 2d strokes into rigged 3d model via latent diffusion models. *arXiv preprint arXiv:2602.09713*, 2026.

Yuyang Zhao, Zhiwen Yan, Enze Xie, Lanqing Hong, Zhenguo Li, and Gim Hee Lee. Animate124: Animating one image to 4d dynamic scene. *arXiv preprint arXiv:2311.14603*, 2023a.

Zibo Zhao, Wen Liu, Xin Chen, Xianfang Zeng, Rui Wang, Pei Cheng, Bin Fu, Tao Chen, Gang Yu, and Shenghua Gao. Michelangelo: Conditional 3d shape generation based on shape-image-text aligned latent representation. *Advances in neural information processing systems*, 36:73969–73982, 2023b.

Hanxin Zhu, Tianyu He, and Zhibo Chen. Ar4d: Autoregressive 4d generation from monocular videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 88–98, 2026.

Table of Contents in Appendix

A Curated Dataset **16**

B Training and model details **16**

C Evaluation Details **17**

C.1 Evaluation Metrics 17

C.2 Inference Details. 18

D More Quantitative Rigging Results **18**

D.1 Generalization and Per-Sequence Optimization Comparison 18

E More Qualitative Rigging Results **19**

F Skinning Model **19**

A CURATED DATASET

TexVerse provides a large-scale collection of animated mesh-skeleton pairs (Zhang et al., 2025b). However, the raw dataset is not limited to clean object-centric motions, and many sequences exhibit quasi-static motion, inconsistent temporal alignment, or incorrect skeleton annotations. To construct a reliable evaluation set, we first use Qwen3.5 (Qwen Team, 2026) for preliminary filtering and then perform additional manual verification, resulting in 2,500 clean object-centric animated instances with correct mesh-skeleton alignment. The bone count distribution is shown in Figure S1. For training and evaluation, we divide into a train set with 2000 instances and a test set with 500 instances.

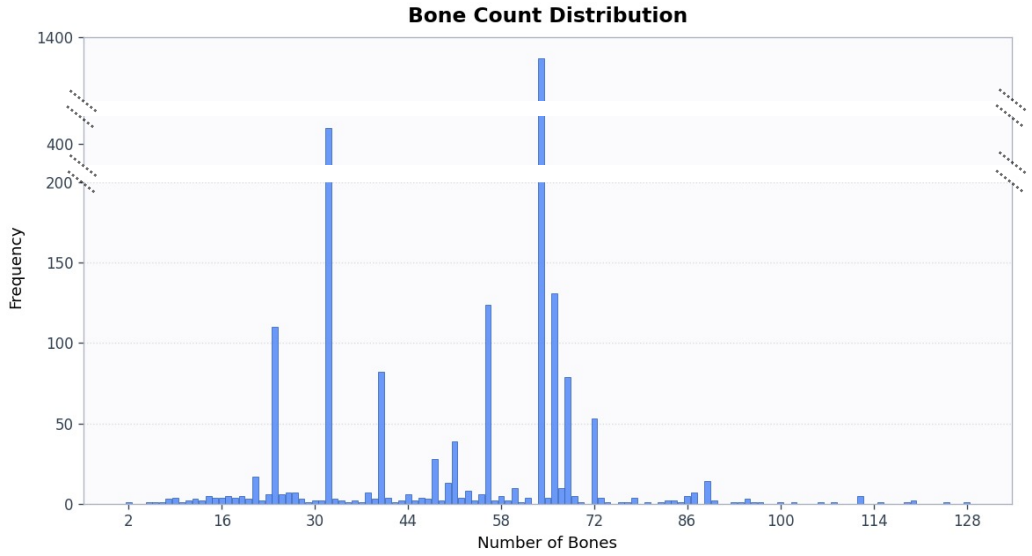


Figure S1: Bone Count Histogram of the filtered TexVerse dataset.

B TRAINING AND MODEL DETAILS

Mesh-to-token conditioning We first sample N_p surface points and normals from each input mesh to form point-cloud frames $P_i \in \mathbb{R}^{N_p \times 6}$. We encode each frame independently using a off-the-shelf Michelangelo point-cloud VAE to obtain a compact set of mesh tokens (Zhao et al., 2023b). Concretely, we flatten the sequence to $P_{flat} \in \mathbb{R}^{N \times N_p \times 6}$ and pass $\{P_i\}$ independently through the point cloud encoder. The resulting conditioning tokens are reshaped to $\mathbf{C} \in \mathbb{R}^{N \times L_m \times H}$. For partially observed sequences, we mask the tokens of unobserved frames and replace with a learnable token for all masked frames.

Model architecture. We use a DiT-L backbone with depth 24, hidden size $H = 1152$, and 16 attention heads. Following the notation in the main method section, for each frame i, the input token sequence concatenates (a) L_s skeleton-slot tokens obtained by linearly projecting the 7D state $\mathbf{x}_{t,i,:,:} \in \mathbb{R}^{L_s \times 7}$ to $\mathbb{R}^{L_s \times H}$ and (b) L_m mesh conditioning tokens $\mathbf{C}_i \in \mathbb{R}^{L_m \times H}$ produced by the Michelangelo point-cloud VAE encoder. Here L_s is the maximum number of skeleton slots, L_m is the number of mesh tokens per frame, and N is the sequence length. This yields a per-frame token set of size $(L_s + L_m)$ and a packed transformer stream of length $N \cdot (L_s + L_m)$. We employ an alternating attention schedule, denoted frame-global alternating, that interleaves frame-wise attention on frame-indexed sequences with global attention on the packed sequence, and we apply temporal RoPE using the frame index across the concatenated token stream.

Training auxiliary loss. The auxiliary loss includes geometry loss and temporal smooth loss. First, we apply the geometry loss to regularize the joints outside the mesh by including a signed distance

(SD) penality term:

$$\mathcal{L}_{\text{geo}} = \frac{1}{NL_s} \sum_{i,j} \text{ReLU}(\text{SD}(\hat{\mathbf{x}}_{0,i,j}, M_i)) \cdot (1-t)^{\gamma_{in}}, \tag{S1}$$

where γ_{in} is another hyperparameter for denoising-step-aware weight.

Second, to encourage coherent dynamics and reduce temporal jitter, we additionally applies lightweight temporal regularizers on slot sequences, including bone-length consistency across adjacent frames and velocity smoothness across frame triplets:

$$\mathcal{L}_{\text{smooth}} = \lambda_{\text{bl}} \frac{1}{N-1} \sum_{i=2}^{N} \ell_{\text{bl}}(\mathbf{s}_i, \mathbf{s}_{i-1}) + \lambda_{\text{vel}} \frac{1}{N-2} \sum_{i=3}^{N} \ell_{\text{vel}}(\mathbf{s}_i, \mathbf{s}_{i-1}, \mathbf{s}_{i-2}), \tag{S2}$$

where $\mathbf{s}_i \in \mathbb{R}^{L_s \times 7}$ denotes the slot tensor at frame i, and λ_{bl} and λ_{vel} are the bone length and velocity smoothness coefficients. As a result, the total auxiliary loss is:

$$\mathcal{L}_{\text{aux}} = \lambda_{\text{geo}} \mathcal{L}_{\text{geo}} + \lambda_{\text{smooth}} \mathcal{L}_{\text{smooth}} \tag{S3}$$

Curriculum training. In the first stage, we set $N = 1$ and optimize the full objective with one-to-one matching. In the second stage, we train on variable-length windows and apply full-frame conditioning. In the third stage, we apply masked-conditioning mechanism: for each training sample we draw an observed index set $\mathcal{O}$ and replace mesh tokens for unobserved frames with a mask token while injecting an observed-frame pooled summary into those positions. In practice, we increase the frequency of structured masks such as biasing the masking distribution toward structured patterns corresponding to forecasting and backcasting, including forecast masks with $\mathcal{O} = \{1, \ldots, \lfloor N/2 \rfloor\}$, backcast masks with $\mathcal{O} = \{\lfloor N/2 \rfloor + 1, \ldots, N\}$, both-ends masks with $\mathcal{O} = \{1, N\}$, and full observation with $\mathcal{O} = \{1, \ldots, N\}$. These strategies align the training distribution with inference-time use cases without changing the underlying objective.

Optimization and guidance. We optimize the model with AdamW under mixed precision using bf16, together with cosine learning-rate decay and warmup. We set the starting learning rate to $1e-4$, $5e-5$, $3e-5$ for the three training stages of static reconstruction, dynamic reconstruction, and dynamic prediction. Skeleton-count conditioning is trained with classifier-free guidance by dropping the count label y with probability $p_{\text{drop}} = 0.5$, which forces the model to predict the reasonable skeleton count when no bone count guidance is provided. At inference time we apply CFG with scale $s = 5$ to control the trade-off between fidelity and diversity.

C Evaluation Details

C.1 Evaluation Metrics

Temporal topological consistency. We evaluate temporal topology with graph structure distance (GSD). Following the notation in the main method, each sequence contains N frames and each frame is represented by at most L_s skeleton slots. For methods without persistent slots, we first recover per-frame predicted skeletons $\{\hat{\mathbf{x}}_{0,i}\}_{i=1}^{N}$ and use the sequence-level Hungarian assignment defined in Section 3 to establish bone correspondence across time. Let $\mathcal{J} \subseteq \{1, \ldots, L_s\}$ denote the set of matched and existing bones in the anchor frame $i = 1$, and let $J = |\mathcal{J}|$.

For each frame i, let $\pi_i(j)$ denote the parent index of matched bone $j \in \mathcal{J}$. We compare the predicted topology at frame i against the anchor-frame topology $\pi_1(j)$ using the Hamming distance

$$H_i = \sum_{j \in \mathcal{J}} \mathbb{1}\big[\pi_i(j) \neq \pi_1(j)\big], \tag{S4}$$

and normalize it to obtain the per-frame graph structure distance

$$\text{GSD}_i = \frac{H_i}{J}. \tag{S5}$$

Table S1: Comparison of inference time of skeleton generation on static mesh.

Method	Pinocchio	RigNet	UniRig	MagicArti.	Puppeteer	Ours
Inference time (s)	3.9	4.5	2.9	2.4	1.5	**1.1**

Table S2: Inference time performance across different frame counts.

Time window size	1	2	4	8	16	24
Inference Time (s)	1.1	1.7	2.2	4.4	11.9	23.4

We then report the sequence-level averages

$$\overline{H} = \frac{1}{N-1}\sum_{i=2}^{N} H_i, \qquad \overline{\mathrm{GSD}} = \frac{1}{N-1}\sum_{i=2}^{N} \mathrm{GSD}_i. \tag{S6}$$

For fixed-slot methods, the same definition applies directly because slot correspondence is given by the slot index.

Bone displacement deviation (BDD). We evaluate temporal geometric stability with bone displacement deviation (BDD). Using the same correspondence set $\mathcal{J}$, let $d_j^{(i)}$ denote the length of matched bone $j \in \mathcal{J}$ at frame i. We measure how much each corresponding bone deviates from its anchor-frame length and average over all matched bones and non-anchor frames:

$$\mathrm{BDD} = \frac{1}{J(N-1)}\sum_{j\in\mathcal{J}}\sum_{i=2}^{N} \left|d_j^{(i)} - d_j^{(1)}\right|. \tag{S7}$$

This correspondence-based definition ensures that BDD is computed on aligned bones rather than unmatched slots.

C.2 Inference Details.

To accurately evaluate the model performance, we sample 5 times for each instance and report the average evaluation metrics to reduce the variance of the metrics.

Inference Time. We compare average inference times among baselines. FlowRig requires 1.1s for single-mesh inference. As shown in Table S1, it is 3.5× faster than Pinocchio, 4.1× faster than RigNet, 2.6× faster than UniRig, 2.2× faster than MagicArticulate, and 1.4× faster than Puppeteer. Table S2 further shows how inference time scales with the temporal window size. These results indicate that FlowRig remains efficient for single-frame 3D rigging while scaling reasonably to longer 4D windows.

D More Quantitative Rigging Results

D.1 Generalization and Per-Sequence Optimization Comparison

Data. To evaluate the generalization capability of FlowRig in dynamic reconstruction, we built a second dynamic evaluation set from Objaverse (Deitke et al., 2023), a different source from the TexVerse data the model trains on, using the same VLM pre-filter, manual verification, and quality predicates detailed in Appendix A. This new dataset contains 1,656 verified animated instances across ten categories. Details of the per-category breakdown can be seen in Table S3.

Per-Sequence Optimization Baseline. To perform a more comprehensive evaluation of FlowRig in dynamic reconstruction, we include a per-sequence optimization 4D baseline, RigGS (Yao et al., 2025b). This model takes video as input and produces a skeleton together with 3D Gaussians, so we render each mesh sequence to video and supply the ground-truth masks its pipeline requires. Additionally, its output lives in its own coordinate frame, so we align the generated skeleton to the

Table S3: Category composition of the new Objaverse-derived 4D evaluation set. No instance of this set is used for training; the model is trained only on the TexVerse split described in A

Category	n	Share
bipedal mammal	453	27.4%
fish	301	18.2%
bird	204	12.3%
dinosaur	193	11.7%
quadruped mammal	183	11.1%
insect	154	9.3%
crustacean	62	3.7%
arachnid	58	3.5%
amphibian	35	2.1%
mollusk	13	0.8%
Total	**1,656**	**100%**

ground truth in scale, position, and orientation before computing metrics. Both steps are concessions in RigGS's favour.

Experiment Setup. RigGS optimizes every sequence from scratch and takes 17 minutes per 8-frame instance on an H200, so running it on the full 1,656-instance Objaverse set of W3 would cost 469 H200 GPU hours, against about 2 GPU hours for FLOWRIG. We therefore sample up to 20 instances per category, giving 193 instances across the ten categories and roughly 57 H200 GPU hours. All methods are evaluated on the same subsample at $T = 8$.

Results. We conclude the following based on the results in Tables S4, S5, and S6:

- FLOWRIG **leads in 28 of the 30 category-metric cells**. In aggregate, Ours* is 19.3% below RigGS and 19.2% below Puppeteer-D* on MACRO J2J, 20.2% and 25.5% below on J2B, and 22.5% and 26.6% below on B2B; MICRO gives the same picture. The two exceptions are fish on J2J and J2B, where RigGS leads by 0.4% and 0.5%, plausibly because fish motion is smooth and low in degrees of freedom, which suits per-sequence fitting.
- **Per-sequence optimization is not uniformly stronger than a fine-tuned static rigger.** RigGS and Puppeteer-D* are level on MACRO J2J (0.0636 vs. 0.0635), and RigGS leads only on J2B and B2B (6.7% and 5.2%). Optimizing each sequence individually does not by itself buy joint accuracy.
- **The cost gap is over two orders of magnitude.** RigGS needs 17 minutes per 8-frame instance where FLOWRIG needs 4.4s in a single feedforward pass (Table S2), roughly 230 times faster, which is what makes evaluation at the scale of the full set practical at all.

E MORE QUALITATIVE RIGGING RESULTS

More static and dynamic instances are shown in Figure S2 and Figure S3, respectively.

F SKINNING MODEL

To animate mesh by the generated skeleton, we additionally train an attention-based skinning network, denoted *SkinningTransformer*, to predict soft skinning weights from a mesh observation and its associated skeletal rig. Given a mesh and a rig with $J \le J_{\max}$ joints, our goal is to predict, for each queried surface point $q \in \{1, \dots, N_q\}$, a categorical distribution over valid joints. Each sample contains N_q sampled query points with normals, a padded bone tensor storing parent and child joint coordinates for each bone slot, a validity mask over joint slots, and a ground-truth skinning distribution over valid joints. Formally, for a query point q we model

$$p(\mathbf{w}_q \mid \mathcal{M}, \mathbf{B}, \mathbf{p}_q, \mathcal{G}), \qquad \mathbf{w}_q \in \mathbb{R}^{J_{\max}}, \qquad \sum_{j=1}^{J_{\max}} w_{q,j} = 1, \tag{S8}$$

Table S4: **Per-category J2J on the Objaverse (Deitke et al., 2023) 4D evaluation set.** Stratified subsample, $T = 8$. Bold is best in the row, underlined second best. MACRO averages the ten categories equally, MICRO weights by instance count.

Category	n	Puppeteer-D*	RigGS	Ours	Ours*
amphibian	20	0.0524	0.0675	0.0389	**0.0361**
arachnid	20	0.0503	0.0567	0.0474	**0.0468**
bipedal mammal	20	0.0291	0.0536	0.0238	**0.0222**
bird	20	0.0759	0.0699	**0.0627**	0.0646
quadruped mammal	20	0.0722	0.0686	0.0564	**0.0540**
crustacean	20	0.0695	0.0637	0.0560	**0.0558**
dinosaur	20	0.0644	0.0653	**0.0554**	0.0619
fish	20	0.0733	**0.0513**	0.0515	0.0525
insect	20	0.0692	0.0632	0.0591	**0.0571**
mollusk	13	0.0784	0.0758	0.0663	**0.0623**
MACRO (mean of 10)	193	0.0635	0.0636	0.0517	**0.0513**
MICRO (instance-weighted)	193	0.0629	0.0631	0.0512	**0.0509**

Table S5: **Per-category J2B**. Same protocol and conventions as Table S4.

Category	n	Puppeteer-D*	RigGS	Ours	Ours*
amphibian	20	0.0439	0.0535	0.0289	**0.0272**
arachnid	20	0.0406	0.0423	0.0323	**0.0316**
bipedal mammal	20	0.0233	0.0391	0.0183	**0.0171**
bird	20	0.0647	0.0560	**0.0507**	0.0520
quadruped mammal	20	0.0594	0.0558	0.0432	**0.0400**
crustacean	20	0.0573	0.0475	0.0416	**0.0408**
dinosaur	20	0.0553	0.0525	**0.0461**	0.0523
fish	20	0.0589	**0.0379**	0.0385	0.0381
insect	20	0.0583	0.0476	0.0469	**0.0444**
mollusk	13	0.0640	0.0589	0.0535	**0.0484**
MACRO (mean of 10)	193	0.0526	0.0491	0.0400	**0.0392**
MICRO (instance-weighted)	193	0.0522	0.0487	0.0395	**0.0388**

Table S6: **Per-category B2B**. Same protocol and conventions as Table S4.

Category	n	Puppeteer-D*	RigGS	Ours	Ours*
amphibian	20	0.0399	0.0467	0.0258	**0.0237**
arachnid	20	0.0368	0.0369	0.0276	**0.0268**
bipedal mammal	20	0.0203	0.0367	0.0150	**0.0138**
bird	20	0.0588	0.0496	**0.0437**	0.0440
quadruped mammal	20	0.0506	0.0511	0.0372	**0.0348**
crustacean	20	0.0506	0.0421	0.0384	**0.0372**
dinosaur	20	0.0480	0.0455	**0.0388**	0.0453
fish	20	0.0453	0.0314	0.0320	**0.0307**
insect	20	0.0529	0.0424	0.0413	**0.0383**
mollusk	13	0.0563	0.0521	0.0473	**0.0422**
MACRO (mean of 10)	193	0.0459	0.0435	0.0347	**0.0337**
MICRO (instance-weighted)	193	0.0456	0.0431	0.0343	**0.0334**

where $\mathbf{p}_q \in \mathbb{R}^6$ is the queried point-and-normal feature, $\mathcal{M}$ denotes the frozen mesh encoders, and $\mathcal{G}$ denotes the skeletal graph structure.

Frozen geometry encoders. We condition the skinning model on two frozen geometry pathways. First, a Michelangelo shape encoder (Zhao et al., 2023b) produces a compact set of global shape

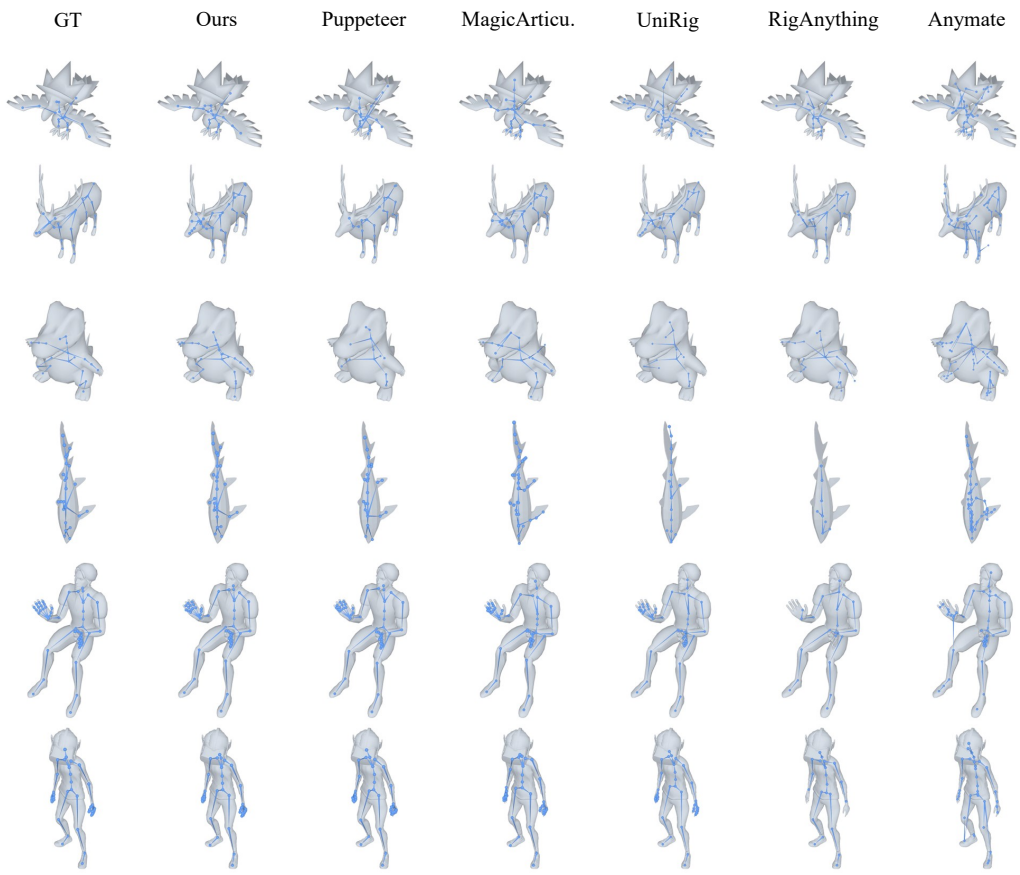


Figure S2: More comparsion on static reconstruction.

tokens $\mathbf{F}_{\text{shape}} \in \mathbb{R}^{257\times768}$. Second, the PartField encoder (Liu et al., 2025b) produces dense query features on sampled surface locations, $\mathbf{F}_{\text{pf}}^{\text{surf}} \in \mathbb{R}^{N_q\times448}$, where N_q is the number of sampled surface queries and $\text{PF}(\cdot)$ denotes the frozen part-aware field encoder. When skeleton-side part features are enabled, we also evaluate this field at the parent and child joint locations $\mathbf{x}_j^{\text{par}}, \mathbf{x}_j^{\text{chd}} \in \mathbb{R}^3$ of bone j and average them per bone,

$$\mathbf{F}_{\text{pf},j}^{\text{skel}} = \tfrac{1}{2}\big(\text{PF}(\mathbf{x}_j^{\text{par}}) + \text{PF}(\mathbf{x}_j^{\text{chd}})\big) \in \mathbb{R}^{448}. \tag{S9}$$

These encoders remain frozen throughout training.

Shape-skeleton fusion encoder. All conditioning features are projected into a shared transformer width d. Let $\mathbf{B} \in \mathbb{R}^{J_{\max}\times6}$ denote the padded bone tensor, where each row stores the concatenated parent and child joint coordinates of one bone. The learned projection matrices satisfy $\mathbf{W}_s \in \mathbb{R}^{768\times d}$, $\mathbf{W}_j \in \mathbb{R}^{6\times d}$, $\mathbf{W}^{sp} \in \mathbb{R}^{448\times d}$, and $\mathbf{W}_p \in \mathbb{R}^{448\times d}$. We form projected shape and bone tokens as

$$\mathbf{F}'_{\text{shape}} = \mathbf{F}_{\text{shape}}\mathbf{W}_s, \qquad \mathbf{H}_{\text{bone}}^{(0)} = \mathbf{B}\mathbf{W}_j + \boldsymbol{\mathcal{E}}_{:J_{\max}}^{pos} + [\mathbf{F}_{\text{pf}}^{\text{skel}}\mathbf{W}^{sp}], \tag{S10}$$

where $\boldsymbol{\mathcal{E}}_{:J_{\max}}^{pos} \in \mathbb{R}^{J_{\max}\times d}$ is a learnable positional embedding for bone slots and the last term is optional. The encoder input is then

$$\mathbf{Z}^{(0)} = \text{Concat}(\mathbf{F}'_{\text{shape}}, \mathbf{H}_{\text{bone}}^{(0)}) \in \mathbb{R}^{L_{\text{enc}}\times d}, \qquad L_{\text{enc}} = 257 + J_{\max}. \tag{S11}$$

A stack of transformer encoder blocks updates $\mathbf{Z}$, while invalid joint slots are masked so that padded joints do not participate in attention.

Topology-aware joint attention. To encode skeletal structure explicitly, we inject a topology-aware bias into self-attention over the bone tokens (Song et al., 2025a). Let $\mathbf{D} \in \mathbb{R}^{J_{\max}\times J_{\max}}$ denote the pairwise graph-distance matrix on the skeleton. Distances are clamped into a small set of buckets and

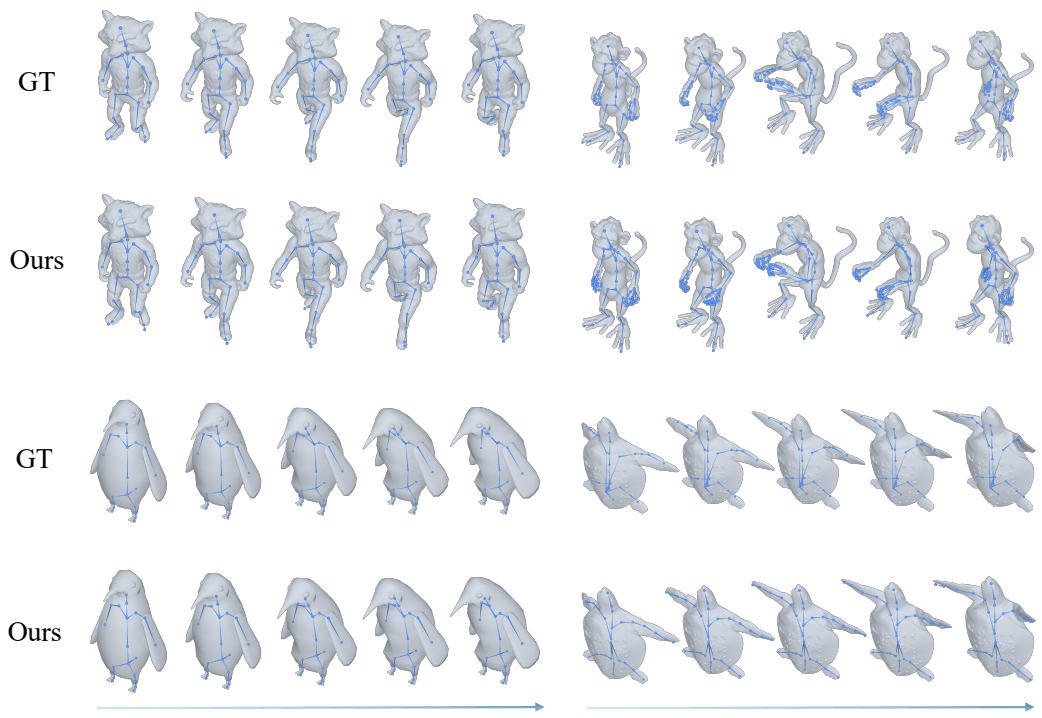


Figure S3: More dynamic reconstruction results.

projected into per-head biases. The attention score for head h, with query/key head dimension d_h, becomes

$$\text{score}_{ij}^{(h)} = \frac{\mathbf{Q}_i^{(h)} \mathbf{K}_j^{(h)\top}}{\sqrt{d_h}} + \lambda_{\text{taja}}\, b_{ij}^{(h)}(\mathbf{D}), \tag{S12}$$

for bone-token indices i and j. This topology-aware joint attention encourages joints that are nearby in the kinematic tree to exchange information more strongly.

Point-wise cross-attention decoder. Surface query tokens are initialized from projected part-aware point features,

$$\mathbf{H}_{\text{pt}}^{(0)} = \mathbf{F}_{\text{pf}}^{\text{surf}} \mathbf{W}_p \in \mathbb{R}^{N_q \times d}. \tag{S13}$$

Optionally, Fourier positional encodings of the queried coordinates and normals are added to these tokens. The decoder then refines point tokens purely through cross-attention to the encoder memory. There is no decoder self-attention; interaction among queries is mediated through the shared shape-skeleton memory.

Scaled cosine prediction head. The final logits are computed by cosine similarity between normalized point embeddings and normalized joint embeddings, scaled by a learnable factor α, where z_{qj} is the logit for assigning query point q to joint j,

$$z_{qj} = (|\alpha| + \varepsilon) \cdot \frac{\mathbf{F}_{\text{pt}}^{\text{out}}(q)}{\|\mathbf{F}_{\text{pt}}^{\text{out}}(q)\|_2} \cdot \frac{\mathbf{F}_{\text{bone}}^{\text{joint}}(j)}{\|\mathbf{F}_{\text{bone}}^{\text{joint}}(j)\|_2}. \tag{S14}$$

Invalid joint logits are masked before normalization, and skinning probabilities are obtained by

$$\mathbf{w}_q = \text{softmax}(\mathbf{z}_q). \tag{S15}$$

This design makes the prediction explicitly probabilistic and naturally handles variable valid-joint sets through masking.

Training objective. We optimize the skinning network with a masked soft cross-entropy objective together with an L_1 loss in probability space. Let $\mathcal{Y}$ denote the target weight distributions and let $\mathbf{Z}$ denote the predicted logits. The overall loss is

$$\mathcal{L} = \mathcal{L}_{\text{CE}}(\mathcal{Y}, \mathbf{Z}; \mathbf{m}) + \lambda_{\ell_1}\, \mathcal{L}_{L_1}(\sigma(\mathbf{Z}), \mathcal{Y}; \mathbf{m}), \tag{S16}$$

Table S7: Quantitative comparison of skinning weight prediction. Precision (Prec.) and Recall (Rec.) quantify the accuracy and coverage of joint influence regions, respectively, where higher values (↑) denote superior performance. Numerical fidelity is assessed via the mean L_1 error, where lower values (↓) indicate closer alignment with the ground truth.

Method	Articulation-XL2.0			ModelsResource			Diverse-pose		
	Prec. ↑	Rec. ↑	L1 ↓	Prec. ↑	Rec. ↑	L1 ↓	Prec. ↑	Rec. ↑	L1 ↓
RigNet	73.7%	66.1%	0.729	65.7%	80.2%	0.707	74.7%	65.4%	0.746
MagicArti.	74.6%	71.3%	0.451	68.1%	80.7%	0.642	74.9%	68.4%	0.479
Puppeteer	87.6%	74.0%	0.335	79.7%	**81.6%**	0.443	83.6%	72.2%	0.405
Puppeteer-D	87.9%	73.8%	0.333	79.8%	81.5%	**0.442**	86.4%	72.8%	0.353
Ours	**89.7%**	**75.0%**	**0.311**	**81.8%**	76.2%	0.449	**88.3%**	**73.6%**	**0.348**

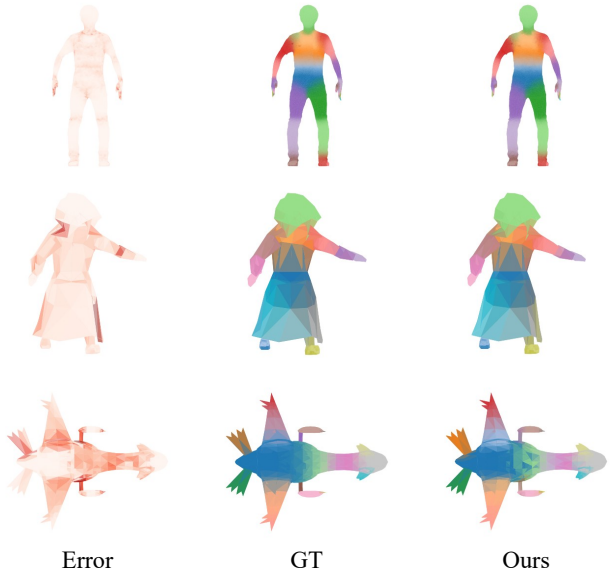


Figure S4: Qualitative results of our skinning model.

where $\mathbf{m} \in \{0,1\}^{N_q \times J_{\max}}$ is the joint-validity mask, $\sigma(\cdot)$ denotes the row-wise softmax operation, and λ_{ℓ_1} is the weight on the probability-space L_1 loss. Only the SkinningTransformer parameters are updated during training; all mesh encoders remain frozen.

Evaluation protocol. At test time, predictions on sampled query points are transferred to mesh vertices through nearest-neighbor interpolation or Gaussian-weighted k-NN blending. Optional post-processing suppresses low-confidence weights and enforces a sparse set of active joints per vertex. This produces the final per-vertex skinning weights used for downstream deformation and evaluation.

Experiments. Table S7 shows that our skinning model achieves the strongest overall performance across the evaluated datasets. On Articulation-XL2.0 and Diverse-pose, it obtains the best precision, recall, and L1 error, indicating more accurate and more complete skinning weight prediction. On ModelsResource, our method achieves the best precision and remains competitive in L1 error, while SPRig attains the highest recall but has poor precision and L1 error. Overall, these results suggest that the proposed skinning model generalizes well across diverse object categories and preserves strong prediction quality in both in-domain and out-of-domain settings. Figure S4 shows qualitative results.