AniGen: Unified S^3 Fields for Animatable 3D Asset Generation

YI-HUA HUANG*, The University of Hong Kong, China
ZI-XIN ZOU, VAST, China
YUTING HE, The Chinese University of Hong Kong, China
CHIRUI CHANG, The University of Hong Kong, China
CHENG-FENG PU, Tsinghua University, China
ZIYI YANG, The University of Hong Kong, China
YUAN-CHEN GUO, VAST, China
YAN-PEI CAO†, VAST, China
XIAOJUAN QI†, The University of Hong Kong, China

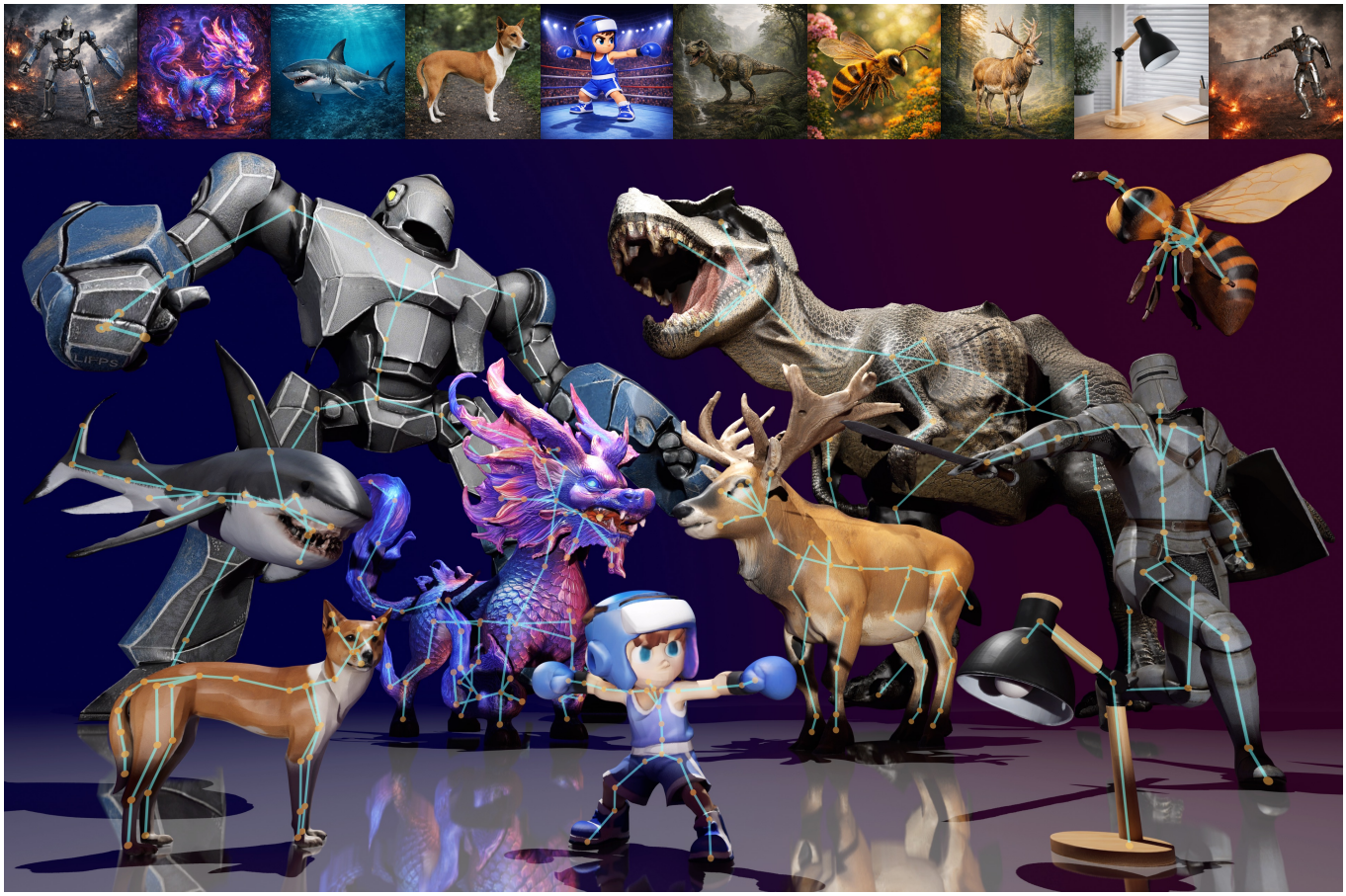


Fig. 1. Given a single conditional image as input, *AniGen* generates a 3D shape along with its skeleton and skinning weights, supporting a wide range of 3D assets, including organic entities such as animals, cartoon characters, humans, and articulated man-made objects.

Animatable 3D assets, defined as geometry equipped with an articulated skeleton and skinning weights, are fundamental to interactive graphics, embodied agents, and animation production. While recent 3D generative models can synthesize visually plausible shapes from images, the results are typically static. Obtaining usable rigs via post-hoc auto-rigging is brittle and often produces skeletons that are topologically inconsistent with the generated geometry. We present *AniGen*, a unified framework that directly generates animate-ready 3D assets conditioned on a single image. Our key insight is to represent shape, skeleton, and skinning as mutually consistent S^3 *Fields* (Shape, Skeleton, Skin) defined over a shared spatial domain. To enable the robust learning of these fields, we introduce two technical innovations: (i) a *confidence-decaying skeleton field* that explicitly handles the geometric ambiguity of bone prediction at Voronoi boundaries, and (ii) a *dual*

*This work is in collaboration with VAST.
†Corresponding Author

skin feature field that decouples skinning weights from specific joint counts, allowing a fixed-architecture network to predict rigs of arbitrary complexity. Built upon a two-stage flow-matching pipeline, *AniGen* first synthesizes a sparse structural scaffold and then generates dense geometry and articulation in a structured latent space. Extensive experiments demonstrate that *AniGen* substantially outperforms state-of-the-art sequential baselines in rig validity and animation quality, generalizing effectively to in-the-wild images across diverse categories including animals, humanoids, and machinery. **Homepage:** https://yihua7.github.io/AniGen-web/

1 Introduction

The rapid advancement of generative 3D models [Chen et al. 2025b; Li et al. 2025c; Wu et al. 2024a; Xiang et al. 2025b; Zhang et al. 2023, 2024b] has begun to democratize 3D content creation, enabling the synthesis of visually stunning geometry and appearance from simple text or image prompts. However, a significant gap remains between visual plausibility and functional utility. In interactive domains such as video games, virtual reality, and embodied AI, a 3D asset's utility depends on its *animatability*. That is, it must be equipped with an articulated skeleton and precise skinning weights to enable posing, motion-capture driving, or physical simulation. Current generative models paradigms primarily yield static statues: models that process high-fidelity surface details but remain functionally inert, serving as mere decorations rather than interactive entities.

A straightforward strategy to bridge this gap is a sequential "generate-then-rig" pipeline: first synthesize a static mesh using a state-of-the-art 3D generator [Xiang et al. 2025b], followed by an off-the-shelf auto-rigging algorithm [Deng et al. 2025; Liu et al. 2025; Song et al. 2025a; Zhang et al. 2025]. Unfortunately, this decoupled approach is brittle. Unlike artist-authored assets, which are modeled with clean topology and articulation in mind, generated meshes often contain "topological variances", e.g., small geometric irregularities, diversely posed limbs, or ambiguous surface topology. While these variances may be visually negligible, they are catastrophic for auto-riggers, which rely on precise topological cues to infer kinematic chains. As shown in Figure 2, this mismatch often leads to anatomically incorrect skelections or skinning artifacts that cause unnatural shearing during animation.

In this work, we argue that this fragility stems from a fundamental *representation mismatch*. Geometry and articulation are not disjoint attributes to be processed sequentially; rather, they are inherently entangled. The shape of a character is functionally determined by its underlying skeleton, and the validity of a skeleton is constrained by the volume of the shape. Therefore, treating rigging as a post-processing step ignores the intrinsic cross-modal priors that exist between shape and function.

To address this, we present *AniGen*, a unified generative framework that treats 3D shape, skeleton, and skinning as mutually consistent fields to be co-generated. Our key insight is to unify these distinct modalities into a common shared continuous representation, which we term S^3 *Fields* (*S*hape, *S*keleton, *S*kin). By representing the skeleton not as a discrete graph but as a dense vector field, and skinning not as a sparse matrix but as a dual feature field, we enable all three components to share the same spatial domain and generative priors. Crucially, this unified representation enables a coherent

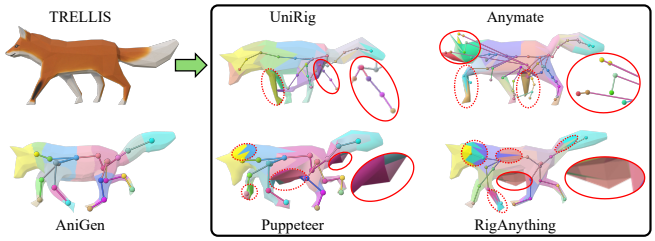


Fig. 2. We present a case study combining TRELLIS [Xiang et al. 2025b] with state-of-the-art auto-rigging baselines for animatable asset generation. Existing methods produce skeletons with missing bones (UniRig [Zhang et al. 2025]), unstructured rigs (Anymate [Deng et al. 2025]), or poor skinning (Puppeteer [Song et al. 2025a], RigAnything [Liu et al. 2025]). The resulting animations expose practical failure modes of this pipeline, whereas *AniGen* generates plausible skeletons and skinning that support stable character animation.

compression-generation pipeline that can effectively "grow" the geometry and rig simultaneously. To make joint generation tractable, we introduce two key designs at the auto-encoding stage to compress the S^3 Fields into structured sparse latents [Xiang et al. 2025b]. First, to address the inherent ambiguity of skeleton regression in regions equidistant to multiple bones (i.e., near Voronoi boundaries), we introduce a *confidence-decaying skeleton field*. This mechanism explicitly down-weights ambiguous regions during auto-encoder training, focusing supervision on high-certainty areas near kinematic chains and thereby enforcing structural coherence. Second, to accommodate the wide variation in joint counts across object categories, we propose a *Dual Skin Field* representation, decoded via a pre-trained Skin Auto-Encoder (SkinAE). This design converts variable-cardinality skinning into a fixed-dimensional latent feature space, enabling consistent compression of rigs with arbitrary complexity.

Built upon these compressed latents, we train a sparse structured flow-matching model [Xiang et al. 2025b] to generate structured latent codes from image-conditioned noise, which are subsequently decoded into geometry and rigging simultaneously. *AniGen* demonstrates strong generalization capability, producing fully rigged, animate-ready 3D assets from single images across a broad range of categories, including humanoid characters, organic animals, and articulated machinery. Extensive experiments show that *AniGen* consistently outperforms sequential baselines in rig validity and animation quality, representing a practical step toward the scalable generation of functional 3D worlds. We additionally show that this joint modeling of geometry and articulation does not compromise geometric fidelity relative to strong geometry-only generators, making the method practical rather than merely structurally correct.

In summary, our contributions are:

- *Unified Generative Formulation:* We propose a holistic framework for the co-generation of shape, skeleton, and skinning as mutually aligned S^3 Fields, effectively eliminating the accumulation of errors inherent in sequential "generate-then-rig" pipelines.

- *Confidence-Aware Skeleton Field:* We introduce a confidence-decaying vector field representation that resolves structural ambiguity, enabling robust graph extraction from noisy volumetric predictions.
- *Joint-Count Agnostic Skinning:* We devise a Dual Skin Field and SkinAE training strategy that allows generative models to synthesize rigs of arbitrary complexity.
- *State-of-the-Art Performance:* We demonstrate that *AniGen* produces high-fidelity, animate-ready assets from in-the-wild images, outperforming existing auto-rigging baselines.

2 Related Work

2.1 Conditional 3D Generation

Early text-to-3D generation methods are predominantly optimization-based. DreamField [Jain et al. 2022] leverages CLIP [Radford et al. 2021] to optimize NeRF [Mildenhall et al. 2021] renderings such that the reconstructed 3D scene aligns with a given text prompt. Building upon this paradigm, DreamFusion [Poole et al. 2023] introduces Score Distillation Sampling (SDS), which employs a pretrained image diffusion model [Ho et al. 2020; Song et al. 2021] to supervise NeRF optimization. Subsequent "dreamer" variants [Liu et al. 2024; Long et al. 2024; Wang et al. 2023b; Yu et al. 2023] further extend DreamFusion by improving the SDS formulation [Wang et al. 2023b; Yu et al. 2023], incorporating multi-view SDS supervision [Liu et al. 2024; Shi et al. 2023], and introducing normal-domain constraints [Long et al. 2024].

Despite their effectiveness, optimization-based approaches are computationally expensive, often requiring hours to generate a single asset. To address this limitation, feed-forward methods have been proposed to directly infer 3D representations. Methods such as Zero-1-to-3 [Liu et al. 2023], Instant3D [Li et al. 2024], DMV3D [Xu et al. 2024], and CAT3D [Gao et al. 2024] first synthesize pose-conditioned multi-view images and then reconstruct 3D geometry using NeRF or 3D Gaussian Splatting (3DGS) [Kerbl et al. 2023]. In contrast, LRM [Hong et al. 2023] formulates 3D reconstruction as a direct regression problem by employing a transformer [Vaswani et al. 2017] to predict triplane representations from partial image observations. Building on this idea, TriplaneGaussian [Zou et al. 2024] and LGM [Tang et al. 2024] replace volumetric representations with 3DGS to improve rendering quality and geometric fidelity. TripoSR [Tochilkin et al. 2024] further scales up LRM by leveraging larger datasets and refined architectural designs.

However, under partial observations, such as a single image or a small number of views, recovering a complete 3D shape is inherently a generative task rather than a pure reconstruction problem. Consequently, regression-based feed-forward methods tend to produce overly smoothed geometry, as they implicitly learn the data average and struggle to preserve fine-grained details. To mitigate this issue, VecSet [Zhang et al. 2023] proposes encoding 3D shapes into fixed-length token sequences and training diffusion models in the token space. Direct3D [Wu et al. 2024a] adopts VecSet tokens to predict triplanes that encode geometric structure, while Dora [Chen et al. 2025c] enhances detail preservation by adaptively sampling more points around sharp edges. Subsequent works, including TripoSG [Li et al. 2025c] and Clay [Zhang et al. 2024b], further improve performance by scaling up model capacity and leveraging larger, more carefully curated datasets.

More recently, TRELLIS [Xiang et al. 2025b] has emerged as a leading framework for high-quality 3D generation, demonstrating the effectiveness of a two-stage generation paradigm. In the first stage, TRELLIS generates a sparse voxel representation that captures the global structure of the shape. Conditioned on this sparse structure, the second stage refines geometry and details using rectified flow [Liu et al. 2022] operating on sparse tensors. Several follow-up works improve upon this framework. SparseFlex [He et al. 2025] increases the voxel resolution in the first stage to 512^3 to enhance geometric accuracy. Ultra3D [Chen et al. 2025b] replaces the voxel-based first stage with VecSet representations and feeds voxelized outputs into the second-stage diffusion model. CUPID [Huang et al. 2025a] improves image-conditioned generation by predicting per-voxel UV coordinates in the first stage, leading to better alignment with input images. Recently proposed TRELLIS 2 [Xiang et al. 2025a] further advances this paradigm by substantially increasing the first-stage resolution to 1536^3 and incorporating high-quality PBR material prediction in the second stage, achieving state-of-the-art results and highlighting the advantages of two-stage 3D generation. Although these approaches produce geometry of exceptional quality, they are limited to static representations and lack the articulation capabilities essential for interactive 3D applications.

2.2 Automatic Rigging and Skinning

Automatic rigging [Bærentzen et al. 2014; Borosán et al. 2012; Pandey et al. 2022; Xu et al. 2019] aims to equip static 3D meshes with skeletons and skinning weights to enable animation. To address animation and rigging challenges, structural representations for 3D shapes were initially proposed by Marr et al.[Marr and Nishihara 1978], focusing on spatial organization rather than surface geometry. Pinocchio[Baran and Popović 2007] introduced a fully automated method for embedding template skeletons into arbitrary 3D meshes, enabling animation-ready rigs with minimal manual intervention. RigNet [Xu et al. 2020] further advanced rigging through deep learning but was limited to templates and T-poses. Subsequent works, such as Neural Blend Shapes [Li et al. 2021] and TARig [Ma and Zhang 2023], tackled rigging and skinning for humanoid characters, while MoRig [Xu et al. 2022] used deforming point clouds for guidance. Other methods, such as NeuroSkinning [Liu et al. 2019] and DRiVE [Sun et al. 2025], also focus on skeleton prediction and skinning for human characters.

More recently, general-purpose rigging methods have emerged. UniRig [Zhang et al. 2025] was one of the first to rig arbitrary shapes with an end-to-end auto-regressive model, followed by RigAnything [Liu et al. 2025], which applied a joint diffusion model for sequential joint generation. Anymate [Deng et al. 2025] introduced a modular approach to predict joints, connectivity, and skinning weights, while Puppeteer [Song et al. 2025a] leveraged an auto-regressive transformer to predict skeletons and drive motion via optimization. However, these methods typically function as post-processing steps that depend heavily on the distribution of the

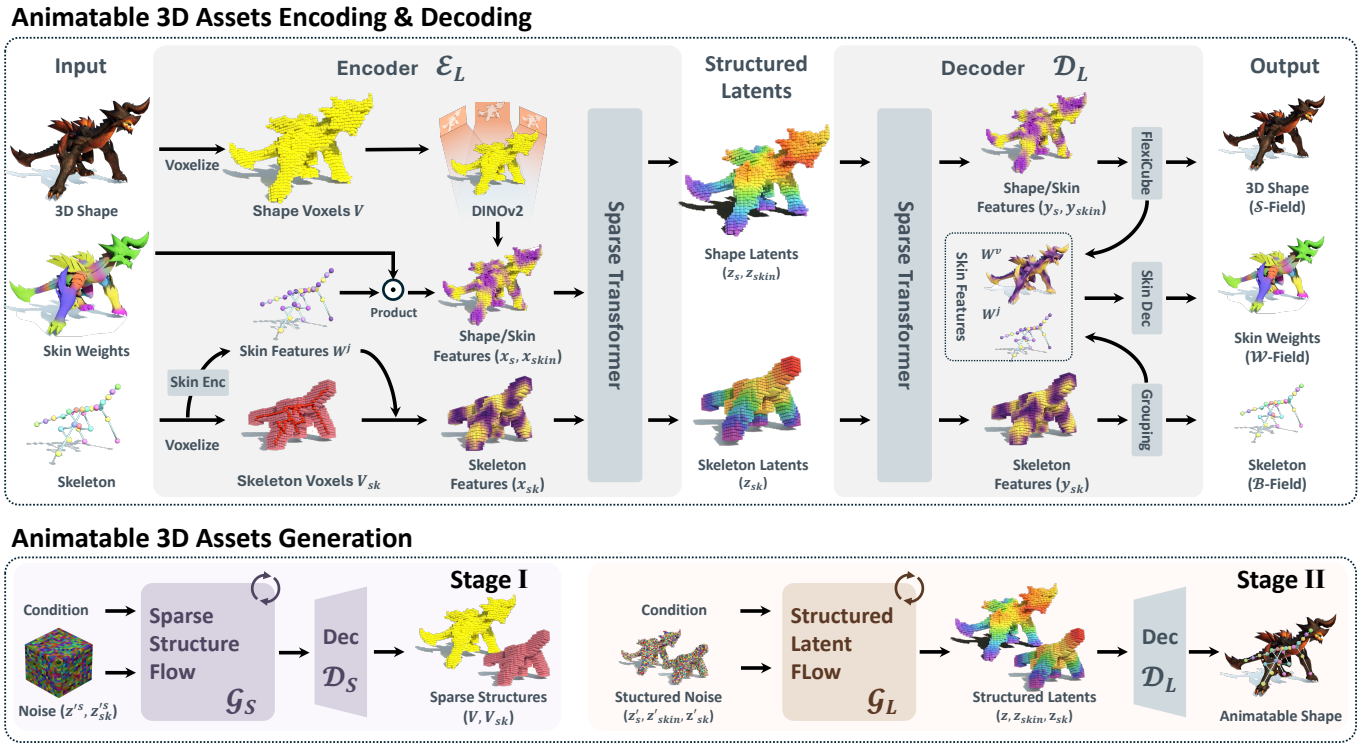


Fig. 3. Overview of the *AniGen* framework. The top panel illustrates the encoding and decoding of the S^3 fields using the structured latent autoencoder $\mathcal{E}_L$ and $\mathcal{D}_L$. From top to bottom, the three rows correspond to the shape, skin, and skeleton branches, respectively. The bottom panel depicts the generation pipeline for animatable 3D assets. Given an input image, the sparse structure flow model $\mathcal{G}_S$ predicts voxel scaffolds that serve as supports for the shape and skeleton. Conditioned on these supports, a structured latent flow $\mathcal{G}_L$ synthesizes the corresponding S^3 fields. Finally, the decoded outputs yield a 3D shape equipped with skeleton rigging and associated skinning, producing an animatable 3D asset.

input mesh. They often struggle with topological, shape-, and pose-variances common in generative outputs, motivating our end-to-end approach where geometry and rigging are co-generated.

2.3 Generative Dynamic & Articulated Assets

Beyond generating static 3D content, researchers have also explored creating animatable or dynamic 3D objects. One direction involves 4D generation. DreamGaussian4D [Ren et al. 2023] extends optimization-based methods to dynamic generation. Cat4D [Wu et al. 2025a] and SVG [Dai et al. 2025] use diffusion models to generate spatial-temporal frame matrices for dynamic scenes but are limited in view range. SMRNet [Zhang et al. 2024a] and HMC [Wang et al. 2023a] animate existing 3D shapes by retargeting them with reference animations, while SC4D [Wu et al. 2024b] and DeformSplat [Kim et al. 2025] optimize 3D shapes into dynamic motion with sparse control and local rigidity [Huang et al. 2024]. AnimateAnyMesh [Wu et al. 2025b] generates deformed mesh sequences using a feedforward model, and SS4D [Li et al. 2025b] predicts sequential sparse structures and latent representations to produce dynamic scenes. While these methods generate dynamic content, they often lack flexible controllers to freely adjust pose and motion.

Another direction explores combining rigging with motion synthesis or part-level articulation. Anytop [Gat et al. 2025] animates skeletons based on semantic joint names using a transformer with skeletal and temporal attention. AnimaX [Huang et al. 2025b] generates multi-view videos of animated characters and derives pose parameters via triangulation and inverse kinematics. Similarly, AnimaMimic [Xie et al. 2025] combines UniRig with video priors to drive motion. Make-it-Poseable [Guo et al. 2025] bypasses skinning by directly animating meshes with their associated skeletons. For articulated objects, ArtiLatent [Chen et al. 2025a] generates sparse structures with articulation labels (e.g., part types and joint types), enabling dynamic objects like drawers or cabinets. Particulate [Li et al. 2025a] applies a part articulation transformer to predict articulation labels, supporting downstream embodied AI tasks. Unlike these works, which often focus on rigid parts or video-driven priors, our framework unifies shape, skeleton, and skinning generation to produce fully animatable organic assets.

3 Method

3.1 Overview

Our goal is to generate a fully functional, animatable 3D asset from a single RGB image $\mathbf{I}$. Formally, an animatable asset is a tuple $\mathcal{A} = (\mathcal{M}, \mathcal{K}, \mathcal{W})$, consisting of a 3D mesh geometry $\mathcal{M}$, a hierarchical

skeleton $\mathcal{K}$ (joints and connectivity), and skinning weights $\mathcal{W}$ that bind the geometry to the skeleton.

Existing approaches typically treat this as a sequential pipeline: generating a static mesh $\mathcal{M}$ first, and then predicting $\mathcal{K}$ and $\mathcal{W}$ via post-processing. This separation often leads to topological mismatches, where the generated geometry lacks the structural integrity required for articulation (e.g., fused limbs).

AniGen fundamentally departs from this paradigm by modeling *S*hape, *S*keleton, and *S*kinning as a unified, spatially aligned representation which we term S^3 Fields (Sec. 3.2). By representing articulation properties as continuous fields sharing the same spatial domain as the geometry, we ensure mutual consistency.

To synthesize these fields from a single image, we devise a two-stage generative pipeline (see Fig. 3):

- *Representation & Compression:* We first learn to compress the continuous S^3 Fields into a compact, sparse structural representation. We employ a sparse structure auto-encoder ($\mathcal{E}_S, \mathcal{D}_S$) to capture the coarse spatial layout and a structured latent auto-encoder ($\mathcal{E}_L, \mathcal{D}_L$) to encode fine-grained geometry and skinning features into a low-dimensional latent space (Sec. 3.3).
- *Generative Flow:* We then train two flow-matching models to synthesize these representations from the input image. A sparse structure flow model first predicts the spatial occupancy and skeleton graph, followed by a structured latent flow model that "paints" the geometry and skinning details onto this structure (Sec. 3.4).

3.2 S^3 Fields

A core challenge in generating animatable assets is that skeletons and skinning weights are traditionally irregular data structures (graphs and sparse matrices), which are difficult to generate using standard convolutional or transformer architectures that assume fixed grid topologies. We resolve this by lifting them into continuous fields defined over the shared 3D domain $\mathbb{R}^3$. We collectively term this representation S^3 Fields.

3.2.1 Shape Field $\mathcal{S}$. Following recent state-of-the-art 3D generation method [Xiang et al. 2025b], we define the shape field over a set of active sparse voxels $\mathcal{V} \subset \{0,1\}^{N^3}$ that track the mesh surface. Formally, the Shape Field $\mathcal{S}$ maps each active voxel coordinate $v \in \mathcal{V}$ to a set of local geometric and appearance parameters required by FlexiCubes [Shen et al. 2023] for mesh extraction:

$$\mathcal{S}(v) = [\mathbf{d}, \mathbf{n}, \mathbf{c}, \mathbf{f}_{\text{flex}}] \in \mathbb{R}^{C_S}, \tag{1}$$

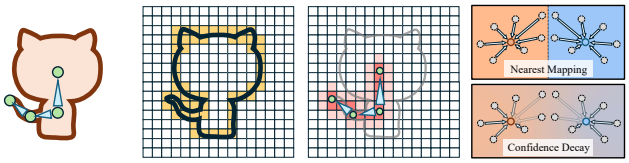
where $\mathbf{d} \in \mathbb{R}^8$ denotes the signed distances at the voxel corners, $\mathbf{n} \in \mathbb{R}^{8\times3}$ and $\mathbf{c} \in \mathbb{R}^{8\times3}$ represent corner normals and colors, and $\mathbf{f}_{flex}$ contains the FlexiCubes-specific interpolation and splitting weights [Shen et al. 2023]. This explicit parameterization enables the extraction of high-quality, watertight meshes $\mathcal{M}$ directly from the field components. Since color is stored in the same field, the final textured mesh can be obtained during surface extraction.

3.2.2 Skeleton Field $\mathcal{B}$. A skeleton is conventionally represented as a set of joints organized in a tree structure in Euclidean space. However, such a discrete graph representation is ill-suited for generative modeling: it lacks a fixed spatial support, does not align naturally with volumetric or grid-based shape representations, and is difficult to co-generate with geometry using shared model structures. To this end, we represent the skeleton $\mathcal{K}$ not as a graph, but as a dense vector field, allowing it to be co-generated with the shape using the same model architecture. Let $\mathcal{K} = \{J_1, \ldots, J_M\}$ be the set of discrete joints in Euclidean space. The Skeleton Field $\mathcal{B} : \mathbb{R}^3 \to \mathbb{R}^6$ maps any point $x \in \mathbb{R}^3$ to a vector pair pointing to its nearest joint $j(x) \in \mathcal{K}$ and that joint's parent $p(x) \in \mathcal{K}$:

$$\mathcal{B}(x) = [(j(x) - x) \oplus (p(x) - x)], \tag{2}$$

where $\oplus$ denotes concatenation. This relative parameterization ensures the field is translation-invariant and local.

Voxel Support $\mathcal{V}_{sk}$. To model this field efficiently, we parameterize it using latent features distributed across a specific set of sparse voxels $\mathcal{V}_{sk}$. Critically, we cannot simply reuse the shape voxels $\mathcal{V}$ for this purpose, as $\mathcal{V}$ tracks the surface boundary, whereas skeleton joints often reside deep within the object's volume (far from surface voxels). Therefore, we define $\mathcal{V}_{sk}$ as the set of voxels intersected by the skeleton bones. To ensure robust coverage and prevent disconnection during generation, we dilate $\mathcal{V}_{sk}$ with a radius of 2 voxels. This ensures the field is defined on a continuous, volumetric structure enveloping the kinematic chains (see Fig. 4).



(a) Shape & Skeleton (b) Shape Voxels (c) Skeleton Voxels (d) Skeleton Field

Fig. 4. Illustration of the (b) shape voxels $\mathcal{V}$, (c) skeleton voxels $\mathcal{V}_{sk}$, and the (d) confidence-decaying nearest-neighbor joint field.

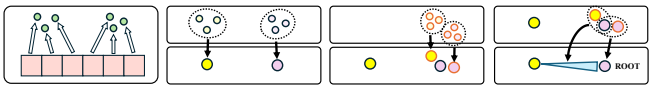
Confidence-Aware Prediction. A major challenge in regression-based skeleton prediction arises near Voronoi boundaries between joints, where the identity of the nearest bone changes abruptly. At these locations, the regression target becomes discontinuous, causing ambiguous supervision and unstable gradients during training. Standard Bayesian uncertainty learning strategies [Kendall and Gal 2017] often struggle here, as they rely on a learned variance that is difficult to weight against the regression loss and cannot guarantee low confidence scores in ambiguous regions (see Sec. 4.6). To mitigate this, we augment the field with a scalar *confidence score* $c(x) \in [0,1]$. We explicitly supervise the confidence using a geometric metric that reflects the ambiguity of joint assignment. For a voxel center v_c, let j^{gt} be the nearest joint and j^{gt}_{2nd} be the second nearest. We define the ground truth (GT) confidence as:

$$c_{gt}(v_c) = 1 - \frac{\|v_c - j^{gt}\|^2}{\|v_c - j^{gt}_{2nd}\|^2} \in [0,1]. \tag{3}$$

The model predicts an extra scalar field to fit this target. Crucially, we use c_{gt} to weight the regression loss for joint and parent predictions. This forces the model to focus on the high-certainty regions near bones while suppressing gradients from ambiguous

boundaries. Additionally, the learned confidence field facilitates the next joint recovery stage.

From Continuous Field to Discrete Skeleton. During inference, we can recover the discrete skeleton $\mathcal{K}$ from the predicted field $\mathcal{B}$ and confidence c. For every center $v_i \in \mathcal{V}_{sk}$, we derive a voting joint $\hat{j}_i$ and corresponding parent $\hat{p}_i$ using the predicted Skeleton Field $\hat{\mathcal{B}}(v)$. We then employ a confidence-weighted iterative clustering algorithm (detailed in Alg. 1) to enhance the joint prediction accuracy: we iteratively shift the voting points toward the centroid of their local neighborhood, weighted by their predicted confidence. This effectively concentrates votes into tight clusters. The final confidence-weighted centroids are identified as joints, and isolated low-confidence points are filtered out to remove noise. Additionally, the clustering process estimates the parent location for each cluster. By connecting the cluster centroid to its closest estimated parent, the joints are sequentially linked to form the skeleton. Fig. 5 provides an intuitive illustration of this conversion.



(a) Continuous Joint Field over V_{sk} (b) Joint Vote Clustering (c) Parent Vote Clustering (d) Skeleton Construction

Fig. 5. Illustration of the conversion from a continuous field to a discrete skeleton. The voting results of the joint field in (a) are clustered to obtain discrete joints in (b) and parent nodes in (c). The skeletal topology is then determined by assigning each parent to its nearest joint as in (d).

3.2.3 Dual Skin Field $\mathcal{W}$. Skinning weights represent a normalized probability distribution indicating how each vertex follows the motion of the underlying skeleton. Parameterizing this relationship directly with a neural network is non-trivial because the number of joints, N_j, varies significantly across different asset categories. This variability precludes the use of standard regression layers with fixed output dimensions.

To overcome this, we propose a *Dual Skin Field* formulation that factorizes the skinning function into two implicit feature fields defined over the shape voxels $\mathcal{V}$ and skeleton voxels $\mathcal{V}_{sk}$, respectively:

$$\mathcal{W}, \mathcal{W}_{sk} : \mathbb{R}^3 \to \mathbb{R}^{D_{\text{skin}}}, \tag{4}$$

where D_{skin} denotes the dimension of the shared latent embedding space.

- The *Surface Skin Field* $\mathcal{W}$ encodes local deformability and segmentation cues of the geometry. This field is instantiated as y_{skin} in the auto-encoder, as illustrated in Fig. 3.
- The *Skeleton Skin Field* $\mathcal{W}_{sk}$ encodes the influence characteristics of the skeletal structure. This field is instantiated as parts of y_{sk} in the auto-encoder, as illustrated in Fig. 3.

This representation effectively decouples the network architecture from the specific joint count. To recover the explicit skinning weights $w_v \in \mathbb{R}^{N_j}$ for a query vertex v and a generated set of joints $\{j_i\}_{i=1}^{N_j}$, we query the fields to obtain the vertex feature $\mathcal{W}(v)$ and the joint features $\{\mathcal{W}_{sk}(j_i)\}_{i=1}^{N_j}$. These features are mapped to a shared compatibility space via a lightweight MLP, and the final weights are computed using a cross-attention operation, ensuring that w_v forms a valid partition of unity (sums to 1). For more implementation details, please refer to Sec.3.3.1 and Fig.6. This allows our generative model to predict fixed-size feature maps that naturally generalize to rigs of arbitrary complexity.

ALGORITHM 1: Field-to-Skeleton Clustering

Input : $\mathbf{J}, \mathbf{P} \in \mathbb{R}^{N\times 3}$; confidences $\mathbf{c} \in \mathbb{R}^{N\times 1}$ (default ones); threshold τ; bandwidth h; min cluster size $s_{\min}$.
Output: Grouped joints $\bar{\mathbf{J}} \in \mathbb{R}^{M\times 3}$; parent indices $\boldsymbol{\pi} \in \{-1, \dots, M-1\}^M$.

$\mathbf{S} \leftarrow \mathbf{J}$
for $t \leftarrow 1$ **to** *10* **do**
  for $i \leftarrow 1$ **to** $|\mathbf{S}|$ **do**
    $\mathcal{N}_i \leftarrow$ neighbors of $\mathbf{S}_i$ within radius h
    $w_{ij} \leftarrow c_j^J \exp(-\frac{\|\mathbf{S}_j - \mathbf{S}_i\|^2}{2h^2}) \;\; \forall j \in \mathcal{N}_i$
    $\mathbf{S}'_i \leftarrow \frac{\sum_{j\in\mathcal{N}_i} w_{ij}\mathbf{S}_j}{\sum_{j\in\mathcal{N}_i} w_{ij} + 10^{-8}}$
  end
  if $\max_i \|\mathbf{S}'_i - \mathbf{S}_i\| \le \frac{1}{10}h$ **then**
    break
  end
  $\mathbf{S} \leftarrow \mathbf{S}'$
end
Cluster $\mathbf{S}$ with merge radius $r = \frac{h}{2}$ to get labels $\ell_i \in \{1, \dots, N_c\}$
for $k \leftarrow 1$ **to** N_c **do**
  $\tilde{\mathbf{J}}_k \leftarrow \frac{\sum_{i:\ell_i=k} c_i^J \mathbf{J}_i}{\sum_{i:\ell_i=k} c_i^J}$; $\tilde{\mathbf{P}}_k \leftarrow \frac{\sum_{i:\ell_i=k} c_i^P \mathbf{P}_i}{\sum_{i:\ell_i=k} c_i^P}$
end
Keep clusters with $|\{i : \ell_i = k\}| \ge s_{\min}$ to obtain $\bar{\mathbf{J}}, \bar{\mathbf{P}}$
for $k \leftarrow 1$ **to** M **do**
  $\pi_k \leftarrow \arg\min_u \|\bar{\mathbf{P}}_k - \bar{\mathbf{J}}_u\|_2$
end
return $\bar{\mathbf{J}}, \boldsymbol{\pi}$

3.3 Latent Representation of S^3 Fields

To enable scalable generative modeling of complex S^3 fields, we must project them onto a compact, lower-dimensional manifold. We achieve this via a hierarchical compression strategy that disentangles coarse structural topology from fine-grained geometry and articulation. Unlike standard Variational Auto-Encoders (VAEs) [Kingma and Welling 2014] commonly used in latent diffusion, we employ *Denoising Auto-Encoders (DAEs).* This design choice, aligned with recent findings in generative modeling [Yang et al. 2025; Yao et al. 2025], provides a latent space that is structurally better suited for the linear interpolation trajectory of flow matching.

To ensure training stability and prevent the model from exploiting unbounded feature magnitudes to trivialize the reconstruction task (which leads to singularities in the flow field), we strictly normalize all latent features onto a unit ℓ_1-norm hypersphere. Without this constraint, we observe that the DAE can artificially inflate the latent norm to enlarge pairwise sample distances, which improves reconstruction shortcuts but makes the subsequent flow-matching objective substantially harder. During training, we perturb the clean

encoded latent z with standard Gaussian noise n, mixed via a coefficient $t \in [0, t_{\max}]$, i.e., $z_{\text{sample}} = t \cdot n + (1-t) \cdot z$. To further stabilize the learning process, we implement a curriculum schedule where $t_{\max}$ is linearly increased from 0 to 0.75 over the course of training.

3.3.1 Prerequisite: Skin Auto-Encoder (SkinAE). Before introducing the main auto-encoder that compresses the full S^3 Fields, we first describe how skinning information is encoded into latent features. Skinning weights $W \in \mathbb{R}^{N_v \times N_j}$ inherently depend on the number of joints N_j, which varies significantly across asset categories (e.g., $N_j = 10$ for a fish versus $N_j = 52$ for a humanoid). This variable cardinality makes explicit skinning matrices incompatible with standard neural networks that require fixed input channel dimensions.

To address this issue, we introduce *SkinAE*, which learns a *joint-count–agnostic* representation of skinning. Rather than representing skinning as a joint-indexed matrix, SkinAE factorizes it into fixed-dimensional *joint embeddings* and *vertex embeddings*, thereby decoupling the representation from the number of joints while preserving articulation structure.

Architecture. SkinAE comprises a Transformer-based encoder and a lightweight MLP decoder, as shown in Fig. 6.

(1) *Joint Encoding:* Given a set of skeleton joints $j_{i}{}_{i=1}^{N_j}$, we compute parent-relative edge vectors $e_i = j_i - p_i$, apply positional encoding (PE), and concatenate them with parent features to incorporate structural information. A Transformer encoder processes these to generate joint skin features $W_i^j \in \mathbb{R}^C$, $(C = 4)$.
(2) *Vertex Encoding:* Corresponding vertex embeddings $\{W_k^v \in \mathbb{R}^C\}$ are computed via the *skin-weighted average* of the joint embeddings. It makes the vertex skin features of a fixed channel dimension independent of N_j. This compresses the sparse skinning matrix into a dense and fixed-channel vertex field.
(3) *Decoding:* A lightweight decoder MLP lifts these features to a higher dimension ($\mathbb{R}^{64}$) and reconstructs the original skinning weights via *channel-wise compatibility*:

$$w_k(v) = \text{Softmax}_i\left(\frac{1}{\tilde{T}_k^v}\langle \tilde{W}_k^v, \tilde{W}_i^j \rangle\right). \tag{5}$$

Here, $\tilde{W}$ denotes output features produced by the decoder MLPs (see Fig. 6). We pre-train SkinAE using 3D points sampled directly from the mesh surface and freeze the network weights for the subsequent stages. Empirically, we find this pre-training strategy is essential for the convergence and performance of the structured latent auto-encoder (as demonstrated in Sec. 4.6). This step effectively converts the variable-cardinality skinning regression task into a *fixed-channel feature matching task*, enabling the generative models to synthesize skinning fields for rigs of any complexity.

3.3.2 Sparse Structure Auto-Encoder $\mathcal{E}_S$ & $\mathcal{D}_S$. The first stage of our pipeline captures the coarse spatial layout. The Sparse Structure Auto-Encoder learns compact discrete representations from two aligned volumetric inputs: the shape occupancy grid $\mathcal{V} \in \{0,1\}^{64^3}$ and the skeleton occupancy grid $\mathcal{V}_{sk}$ (defined in Sec. 3.2.2).

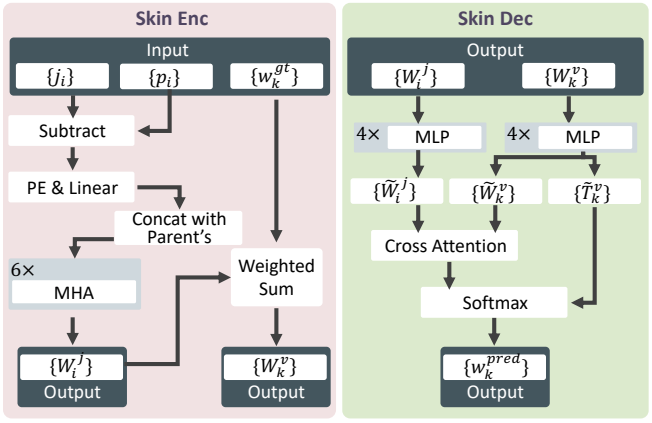


Fig. 6. SkinAE encodes skeleton joints j_i and parents p_i into joint skin features W_i^j, which are averaged by GT skin weights w_k^{gt} to obtain vertex skin features W_k^v. The decoder processes W_i^j and W_k^v via MLPs, producing $\tilde{W}_i^j$, $\tilde{W}_v^k$, and vertex temperatures $\tilde{T}_v^k$. Final skin weights are obtained through cross-attention and Softmax with temperature adjustment.

The encoder $\mathcal{E}_S$ processes each input with a lightweight 3D convolutional network. We utilize strided 3D convolutions to downsample the spatial resolution, followed by multi-resolution 3D residual blocks to capture multi-scale structural dependencies. This process yields two distinct compressed latent volumes: $z^s \in \mathbb{R}^{16^3 \times 8}$ for the shape structure and $z_{sk}^s \in \mathbb{R}^{16^3 \times 4}$ for the skeleton structure. Conversely, the decoder $\mathcal{D}_S$ maps these latents back to the original resolution using progressive 3D upsampling and residual blocks. The network terminates in two separate output heads that reconstruct the binary occupancy probabilities for the shape and skeleton volumes, respectively.

3.3.3 Structured Latent Auto-Encoder $\mathcal{E}_L$ & $\mathcal{D}_L$. The Structured Latent Auto-Encoder ($\mathcal{E}_L$, $\mathcal{D}_L$) serves as the core engine for high-fidelity reconstruction (see Fig. 3, top). Using the sparse voxels $\mathcal{V}$ and $\mathcal{V}_{sk}$ as scaffolds, it encodes the fine-grained S^3 fields into a continuous latent space.

Input Feature Construction. We construct voxel-aligned sparse inputs from both multi-view and geometry observations:

- *Shape (x_s):* Defined on $\mathcal{V}$. We back-project multi-view DINOv2 [Oquab et al. 2024] features onto the voxel grid following TRELLIS [Xiang et al. 2025b].
- *Skinning (x_{skin}):* Defined on $\mathcal{V}$. For each occupied voxel, we query the nearest point on the ground-truth mesh and assign it the corresponding vertex embedding W^v derived from the frozen SkinAE.
- *Skeleton (x_{sk}):* Defined on $\mathcal{V}_{sk}$. We concatenate the positional embeddings of the nearest joint and its parent, along with the joint embedding W^j from SkinAE.

Encoding & Decoding. The encoder $\mathcal{E}_L$ employs a multi-stream sparse Transformer backbone (12 Swin-style blocks [Liu et al. 2021]) to process these inputs, producing three latent volumes ($z_s, z_{\text{skin}}, z_{\text{sk}}$) with channel dimensions 8, 4, and 4, respectively. The decoder $\mathcal{D}_L$

applies the same multi-stream sparse Transformer backbone to obtain hidden features ($h_s, h_{\text{skin}}, h_{\text{sk}}$). It then upsamples the shape and skin latents to high resolution (256^3) to predict: (i) the Shape Field y_s for FlexiCubes extraction, and (ii) the Vertex Feature Field y_{skin} for predicting $\{W_k^v\}$. Simultaneously, the skeleton branch decodes the Skeleton Field y_{sk} (vectors and confidence) and Joint Feature Field for predicting $\{W_i^j\}$. The final asset is assembled by extracting the mesh via Marching FlexiCubes, clustering the skeleton field into discrete joints (Sec. 3.2.2), and decoding the skinning weights using the SkinAE decoder on the predicted features. The reconstructed shape is thus rigged with a predicted skeleton, as illustrated in Fig. 3.

Confidence-Weighted Supervision. We supervise the geometry with standard rendering losses (depth, normal, color). For the skeleton and skeleton-side skinning fields ($\mathcal{W}_{sk}$), which are both represented by y_{sk}, we employ confidence-weighted supervision to mitigate border ambiguity (see Sec. 3.2.2). The regression loss for a joint prediction j_{v_c} at voxel v_c is:

$$L_J = \mathbb{E}_{v_c}\left[c_{gt}(v_c)\,\|j_{v_c} - j_{v_c}^{gt}\|_2^2\right], \tag{6}$$

where c_{gt} is the ground-truth confidence (Eq. 3). The same weighting is applied to parent predictions and skin feature W^j predictions in the skeleton branch. Furthermore, the structured latent autoencoder is trained with feature reconstruction losses for the surface-side skinning embeddings. The sparse-structure auto-encoder, augmented with an additional branch to encode $\mathcal{V}sk$, is optimized using binary occupancy reconstruction losses on both $\mathcal{V}$ and $\mathcal{V}sk$.

BVH-Accelerated Skin Transfer. To supervise the predicted vertex skin features $\{W_k^v\}$, we must match them to the ground truth (GT). Since the predicted mesh topology differs from the GT mesh, we transfer GT skin features via nearest-surface interpolation. To make this efficient during training, we implement a CUDA-based Bounding Volume Hierarchy (BVH) that caches the GT geometry. This reduces average query time from 48.6 ms to 2.6 ms (18.6× speedup). Crucially, as shown in Fig. 7, BVH-based barycentric transfer is more robust to uneven vertex sampling than simple nearest-vertex matching.

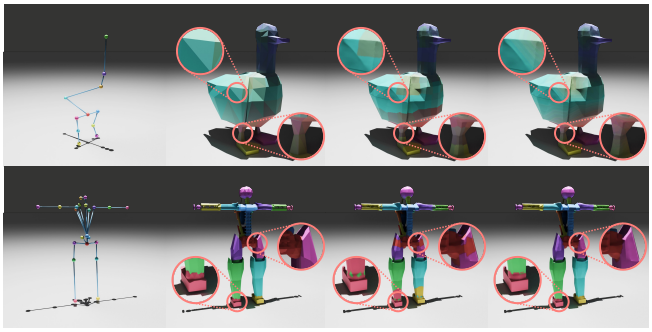


Fig. 7. BVH-based skin transfer vs. nearest-vertex (NN) transfer. We implement a CUDA BVH supporting multi-device deployment and save/restore.

3.4 Generative Flow Model

We model the generation of animatable assets as a conditional flow matching problem. Formally, we aim to learn a velocity field v_t that transports a standard Gaussian distribution $\pi_0 = \mathcal{N}(0, I)$ to the data distribution π_1 of our compressed representations. To handle the complex interplay between topological structure and dense attributes (i.e., geometry and skinning weights), we decompose the generation process into two cascaded stages: *Sparse Structure Flow* ($\mathcal{G}_S$) and *Structured Latent Flow* ($\mathcal{G}_L$). Fig. 8 illustrates our flow model architecture.

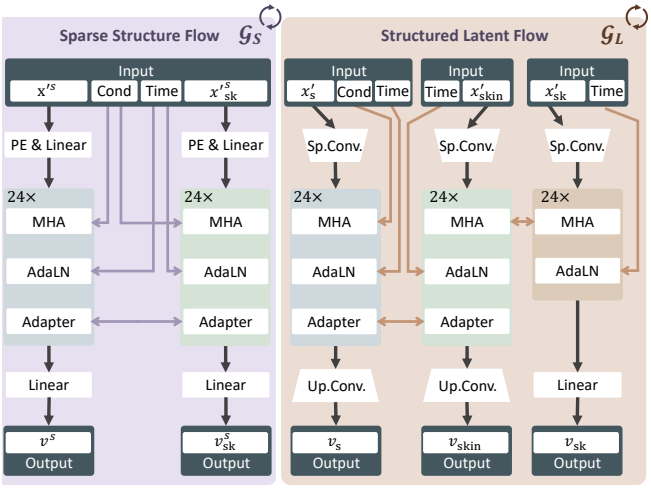


Fig. 8. The architectures of the AniGen flow models, $\mathcal{GS}$ and $\mathcal{GL}$. The input to $\mathcal{GS}$ comprises noisy volumetric features encoding geometry, z'^s, and skeleton information, z'^sskl. $\mathcal{GL}$ processes noisy structured latents representing geometry, z's, skin, z'skin, and skeleton, z'skl. These flow models predict the velocities of the noisy features and iteratively denoise them using the Euler method.

3.4.1 Stage I: Sparse Structure Flow $\mathcal{G}_S$. The first stage constructs the *scaffold* of the asset. Conditioned on image features, $\mathcal{G}_S$ predicts the active sparse voxel sets for both the shape ($\mathcal{V}$) and the skeleton ($\mathcal{V}_{sk}$). We instantiate $\mathcal{G}_S$ as a dual-stream Transformer. Rather than concatenating shape and skeleton into a single sequence (which obscures their distinct topological roles), we process them in parallel branches:

- *Shape Branch:* Predicts the binary occupancy of surface-crossing voxels.
- *Skeleton Branch:* Predicts the binary occupancy of bone-containing voxels.

Cross-Structural Adapters. A naïve dual-stream approach risks generating a skeleton that drifts outside the mesh. To enforce spatial compatibility, we introduce lightweight linear adapters that exchange information between the two branches at every Transformer block. This bidirectional fusion ensures the skeleton "grows" strictly within the geometric bounds of the shape.

3.4.2 Stage II: Structured Latent Flow $\mathcal{G}_L$. Given the generated scaffolds, the second stage synthesizes the S^3 field latent features. $\mathcal{G}_L$ is trained to denoise the latent codes z_{s}, z_{skin}, and z_{sk}.

Architecture. Similar to Stage I, we employ a multi-branch architecture. Since geometry and skinning share the same spatial domain ($\mathcal{V}$), we process them in a primary branch, while the skeleton latent z_{sk} (defined on $\mathcal{V}_{sk}$) is processed in a secondary branch. We again utilize adapter layers to enforce consistency between the predicted skinning features and the underlying skeletal structure.

Controllable Joint Density. A key advantage of our joint-count agnostic representation (Sec. 3.3.1) is that the network is not bound to a specific rig topology. We exploit this by introducing *joint density* as an explicit conditional input. During training, we compute the normalized joint count of the ground truth asset. This scalar is embedded and injected into the flow model via AdaLN modulation [Peebles and Xie 2023]. At inference, users can adjust this scalar to control the granularity of the rig without changing the underlying geometry (visualized in Fig. 11).

4 Experiment

4.1 Experimental Setup

4.1.1 Dataset and Implementation. We adopt ArticulationXL [Song et al. 2025b] as our evaluation dataset, which contains approximately 33K rigged 3D shapes curated from Objaverse-XL [Deitke et al. 2023a,b]. We randomly sample 1K shapes to form the test set. To increase motion diversity, we augment the dataset in two ways. For assets with existing animation sequences, we generate additional samples by interpolating within the asset's own motion. Otherwise, we apply stochastic joint perturbations: each joint has 80% probability of being rotated around a random axis by a jitter of up to 60°. Because ArticulationXL is relatively small (in contrast to the ∼10M shapes in Objaverse-XL) and is insufficient for training a generative flow model from scratch, we adopt a warm-start initialization strategy. Specifically, we initialize the shared branches of the autoencoders and flow modules in *AniGen* with pre-trained TRELLIS parameters [Xiang et al. 2025b], effectively leveraging large-scale geometric priors to facilitate the learning of joint representation. In implementation, we first pre-train SkinAE and then freeze it when training the structured latent auto-encoder.

4.1.2 Baselines. To the best of our knowledge, no existing method directly generates fully rigged 3D shapes – comprising geometry, articulated skeleton, and skinning – in a unified manner. Therefore, we construct strong baselines by coupling state-of-the-art automatic rigging methods with the recent 3D generative model TRELLIS [Xiang et al. 2025b]: we first generate a shape using TRELLIS, and then apply an off-the-shelf rigging algorithm to infer the skeleton and skinning weights. For a fair comparison under the same data domain and pose distribution, we further fine-tune TRELLIS on ArticulationXL using the same pose-augmented training set described above. We refer to this variant as TRELLIS* in Tab.1. The rigging methods we evaluate in this pipeline include UniRig[Zhang et al. 2025], Anymate [Deng et al. 2025], Puppeteer [Song et al. 2025a], and RigAnything [Liu et al. 2025].

4.1.3 Metrics. Following UniRig [Zhang et al. 2025], we adopt three Chamfer-style metrics to quantify skeletal geometric accuracy: *joint-to-joint*, *joint-to-bone*, and *bone-to-bone* distances. While these metrics capture local Euclidean proximity, they are insufficient for comprehensive evaluation of skeleton structure. In particular, they do not treat a skeleton as a *metric measure space* and therefore can be insensitive to structural/topological errors. For example, inserting an extra joint along a ground-truth (GT) bone may not change the joint-to-bone or bone-to-bone distances, and adding a duplicated branch extremely close to a correct branch can still yield negligible Chamfer distances. This many-to-one matching bias inherently masks critical errors in kinematic connectivity and hierarchy.

To address this limitation, we incorporate Optimal Transport-based metrics, specifically the *Wasserstein distance* [Givens and Shortt 1984] and *Gromov–Wasserstein (GW) distance* [Mémoli 2011], which explicitly account for the global distribution. Formally, we use the L_2-Wasserstein distance (Earth Mover's Distance) between joint measures:

$$\mathrm{D}_W(\mu,\nu) = \left(\min_{\pi \in \Pi(a,b)} \sum_{i=1}^{n} \sum_{k=1}^{m} \pi_{ik} \, \| j_i - j_k^{gt} \|_2^2 \right)^{\frac{1}{2}}, \tag{7}$$

where $\pi \in \mathbb{R}_+^{n \times m}$ is a transport plan with $\sum_k \pi_{ik} = \sum_i \pi_{ik} = 1$. Compared with Chamfer distances, the Wasserstein distance enforces a global mass-preserving matching, which mitigates many-to-one correspondences. However, it still relies on the ambient Euclidean cost and therefore does not explicitly encode skeletal topology.

To make the metric topology/structure-aware, we further compute the GW distance between the predicted skeleton graph and the GT skeleton graph by comparing *intrinsic* pairwise distances. Let $d_{\text{p}}(i,i')$ denote the geodesic distance along the predicted skeleton graph between predicted joints i and i', and let $d_{\text{g}}(k,k')$ be defined analogously on the GT skeleton. The GW objective is

$$\mathrm{D}_{GW}(\mu,\nu) = \left(\min_{\pi \in \Pi(a,b)} \sum_{i,i'=1}^{n} \sum_{k,k'=1}^{m} \left| d_{\text{p}}(i,i') - d_{\text{gt}}(k,k') \right|^2 \pi_{ik} \, \pi_{i'k'} \right)^{\frac{1}{2}}. \tag{8}$$

By matching geodesic structures rather than only Euclidean coordinates, GW penalizes topological inconsistencies such as spurious branches or incorrect connectivity even when the geometry is locally close. In practice, the OT problems in Eq. (7) and Eq. (8) can be efficiently solved via iterative Sinkhorn updates. Given the optimal transport plan from Wasserstein distance, we align predicted skinning weights to GT and then report ℓ_1, ℓ_2, and KL divergence between the aligned skinning distributions.

4.2 Quantitative Evaluation

We summarize the quantitative evaluation results in Tab. 1, including Chamfer distances, mm-space distances, and skin metrics. Note that TRELLIS* refers to the finetuned TRELLIS model trained on the split train set. Because every method is image-conditioned rather than GT-conditioned, the generated asset is not expected to be an exact replica of the GT shape. We therefore normalize scale, center the prediction, and then align it to the GT with ICP using 100

Table 1. Quantitative evaluation of skeleton accuracy and skin metrics for various methods. Results include Chamfer distances, mm-space distances, Wasserstein-based distances, and skin metrics (ℓ_1, ℓ_2, KL divergence). The best scores are highlighted in bold, and the second-best scores are underlined. Note that TRELLIS* refers to the finetuned TRELLIS model trained on the split train set, while GT-Mesh combined with rigging methods serves as a reference.

Method	Joint-to-Joint ↓	Joint-to-Bone ↓	Bone-to-Bone ↓	Wasserstein ↓	Gromov–Wasserstein ↓	Skin ℓ_1 ↓	Skin ℓ_2 ↓	Skin KL ↓
GT-Mesh + UniRig	0.190	0.180	0.166	0.260	0.389	0.0973	0.224	6.117
GT-Mesh + Anymate	**0.169**	**0.161**	**0.145**	**0.232**	0.368	0.0918	0.204	4.388
GT-Mesh + Puppeteer	0.235	0.228	0.212	0.239	**0.322**	**0.0823**	**0.197**	**4.133**
GT-Mesh + RigAnything	0.193	0.181	0.165	0.268	0.392	0.1018	0.223	6.161
TRELLIS * + UniRig	0.205	0.192	0.179	0.269	0.397	0.0966	0.221	5.903
TRELLIS * + Anymate	0.179	0.172	0.157	0.232	0.349	0.0919	0.203	4.221
TRELLIS * + Puppeteer	0.245	0.237	0.219	0.241	0.326	0.0873	0.202	4.135
TRELLIS * + RigAnything	0.273	0.264	0.252	0.285	0.383	0.1026	0.224	6.451
AniGen	**0.174**	**0.164**	**0.143**	**0.229**	**0.286**	**0.0793**	**0.186**	**2.919**

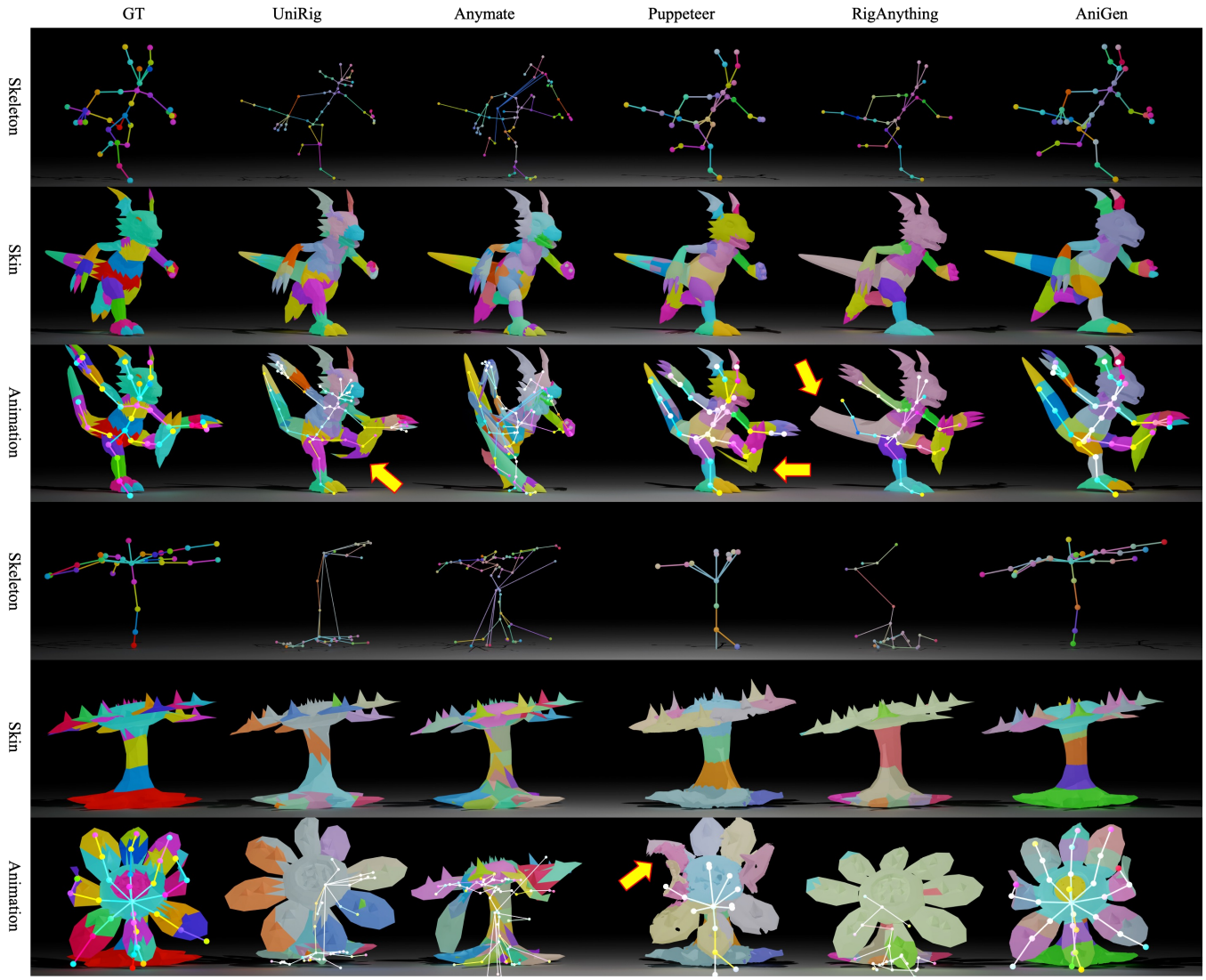


Fig. 9. Qualitative comparison of skeleton and skin results across different methods. We visualize the predicted skeletons, corresponding skins, and demonstrate animations to evaluate the practical usability of the rigged assets. The dragon case represents a relatively simple example with a clear identity and four-limbed structure, while the flower case poses a more complex challenge due to its intricate and non-standard structure.

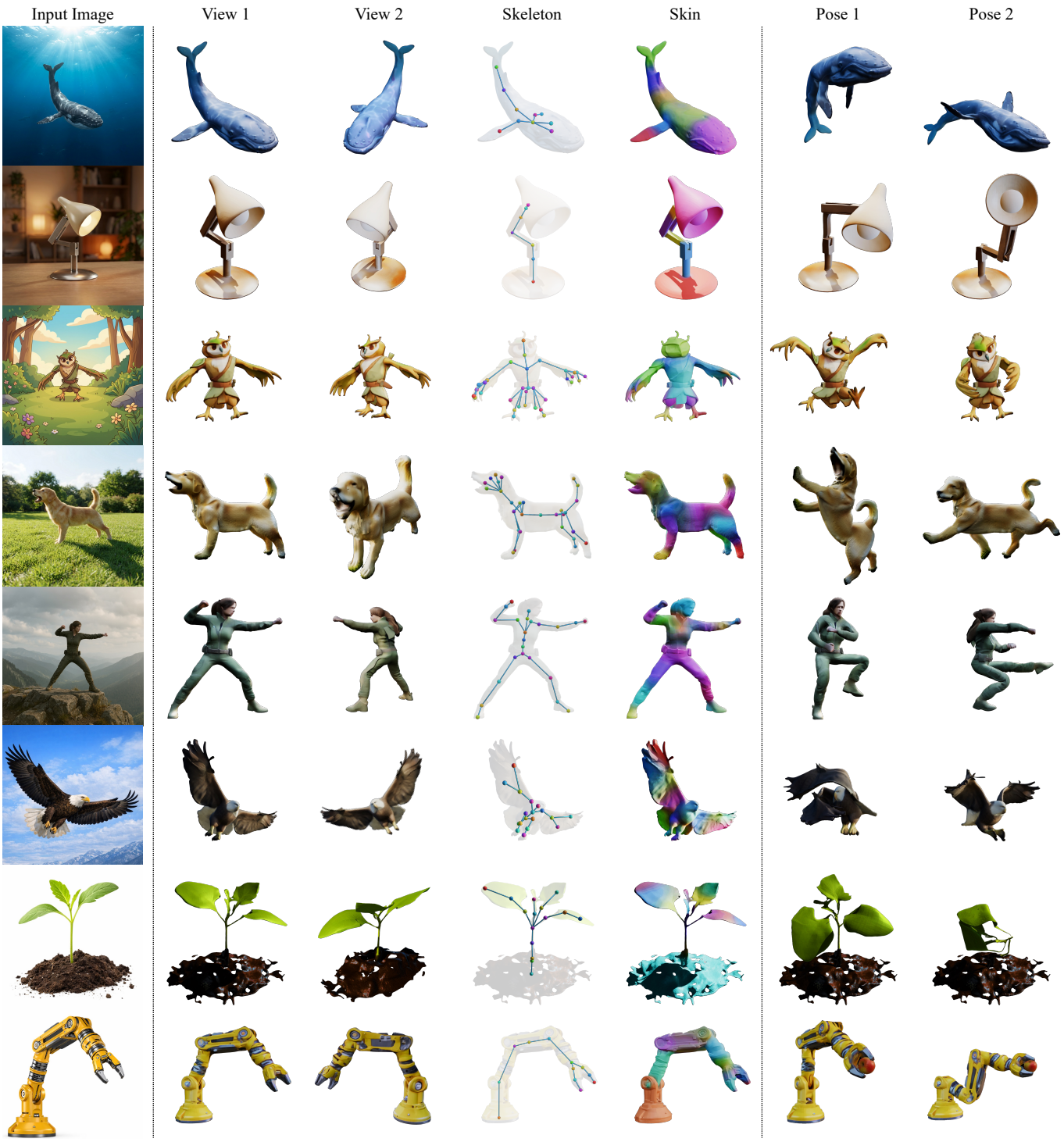


Fig. 10. Qualitative results on in-the-wild images, including examples from real-world photographs, web content, and AI-generated imagery. The showcased objects span a wide range of categories, such as sea animals, household items, cartoon characters, pets, humans, birds, plants, and machinery. Edited poses are demonstrated to illustrate animation capabilities, with examples such as a whale swimming, a lamp adjusting its angle, a dog running, and a robotic arm picking up an apple. These results highlight the versatility and practical usability of *AniGen* for applications in animation, games, image editing, VR, and more.

randomly initialized rotations. The results demonstrate that *AniGen* achieves the best performance in skeleton structure prediction and skin accuracy across all baselines. This establishes *AniGen* as the leading end-to-end image-conditioned rigged shape generation solution. Notably, *AniGen* excels in Gromov-Wasserstein distance and skin KL divergence, achieving significant advantages over the other baselines in the accuracy of skeleton topology and skin weights. Additionally, we provide results from coupling GT meshes with off-the-shelf rigging methods to serve as an upper-bound reference. While these GT-input baselines naturally exhibit better results, it is worth noting that generation models often produce geometries that deviate slightly from the GT due to minor variations in scale, rotation, and pose.

Geometry Evaluation. We further report geometry-and-fidelity metrics in Tab. 2. *AniGen* remains competitive with TRELLIS*, it shows only a small gap, while substantially improving the rigging-related metrics in Tab. 1. This confirms that jointly modeling shape, skeleton, and skin does not substantially degrade geometry quality. Pure geometry generation can still be slightly stronger under geometry-only metrics, which we consider a minor limitation, but the gap is small relative to the articulation gains.

Table 2. Geometry evaluation on the rigged-domain test set. We report surface Chamfer distance, F-score, and image-space PSNR & LPIPS.

Method	Chamfer ↓	F-Score ↑	PSNR ↑	LPIPS ↓
TRELLIS	0.0475	0.77	24.25	0.126
TRELLIS*	**0.0394**	**0.89**	**25.23**	**0.104**
AniGen	0.0409	0.88	25.12	0.108

Inference Cost Evaluation. We report end-to-end inference cost in Tab. 3. *AniGen* is comparable to the fastest sequential baseline while avoiding the heavy post-hoc rigging overhead of methods such as UniRig and RigAnything.

Table 3. Inference cost runtime comparison.

Method	TRELLIS*	TRELLIS*+UniRig	TRELLIS*+Anymate
Time (s) ↓	15	146	19
Method	TRELLIS*+Puppeteer	TRELLIS*+RigAnything	*AniGen*
Time (s) ↓	36	127	19

4.3 Qualitative Evaluation

To provide a more intuitive comparison, we present qualitative results across various baselines in Fig. 9. We visualize the generated skeletons and skins and perform similar animations on the outputs to better demonstrate the practical usability of each rigged asset. For brevity, we omit TRELLIS*+ in the following discussion.

In the case of the dragon, which is a relatively simple example with a clear identity and a four-limbed structure, *AniGen*, UniRig, Puppeteer, and RigAnything generate skeletons that are overall very similar to the ground truth. However, there are notable differences in the details. UniRig generates redundant bones in the head, while Puppeteer and RigAnything fail to produce detailed finger joints. Anymate produces a skeleton with joint distributions very close to the GT, but the connections between bones are incorrect. Regarding skin results, UniRig, Puppeteer, and RigAnything exhibit regional artifacts, particularly in the feet or tail.

The flower example is more challenging than the dragon. UniRig and RigAnything fail to generate skeletons that adequately cover the full structure of the flower. In contrast, Anymate produces joints that closely match the GT joint distribution, but some bone connections are incorrect. Puppeteer creates a coarse yet overall suitable skeleton to support the flower, but its skin results are inadequate for practical animation, resulting in broken animations. Some compared results also exhibit pose discrepancies after animation. This is not because different target poses are used; instead, all methods are driven toward the same target motion. When a baseline predicts topologically broken bones or erroneous skin influences, it cannot physically realize the target pose without catastrophic distortion, so the final animated pose remains visibly mismatched.

4.4 In-the-Wild Results

We present the robust generalization of *AniGen* on diverse "in-the-wild" images in Fig. 10, including real-world photographs, web-sourced imagery, and AI-generated content. These examples span a diverse set of object categories: sea animals, household items, cartoon characters, pets, humans, birds, plants, and machinery, demonstrating the versatility of our approach across both natural and synthetic visual domains. To highlight the functional utility of the generated assets, we further provide edited poses and motion variants tailored to the identity, structure, and expected behavior of the underlying subject. As illustrated, a whale can be posed to swim freely through an ocean scene; the lamp can be reoriented so its head and body direct light toward different targets; and the cartoon character can be driven through a variety of full-body motions. Likewise, the dog can open and close its mouth while running across a lawn, the woman can be animated performing kung-fu movements, and the eagle can be posed to hug or capture a sheep in a dynamic interaction. Beyond animals and humans, we also demonstrate controllable state changes and functional motions: the plant can transition between blooming and withering, and the robotic arm can grasp an apple, lift it, and transport it to a new location.

These diverse results underscore the flexibility and versatility of *AniGen*, demonstrating its capacity to operate effectively and robustly across a wide variety of subjects, visual styles, and real-world scenarios. This breadth suggests that *AniGen* serves as a unified, category-agnostic foundation for controllable asset synthesis and animation, rather than being confined to narrow context domains. As a result, *AniGen* enables a wide range of downstream applications, including embodied AI (e.g., interactive agents that require consistent, controllable visual assets), image and video editing (e.g., pose- and motion-aware content modification), animation and gaming pipelines (e.g., rapid prototyping of characters, props, and actions), and creative production workflows such as cartoon creation and stylized storytelling. Moreover, it can support immersive and

simulation-centric settings, including virtual reality experiences, digital-twin systems, and game character development.

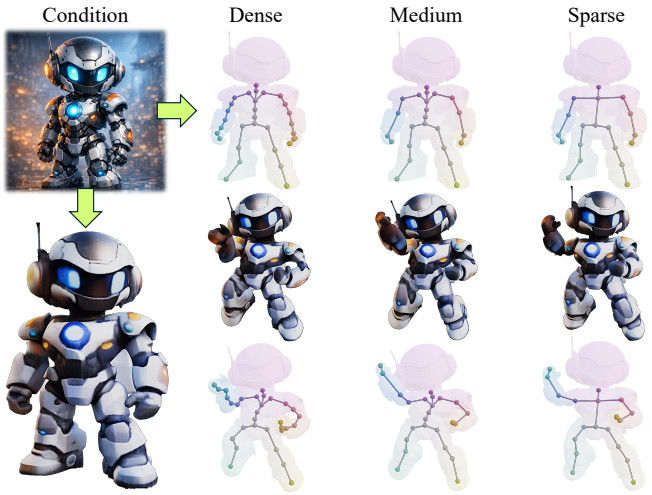


Fig. 11. Skeleton generation with different joint density levels. Higher joint density enables flexible, human-like motions, while medium and sparse densities result in reduced flexibility, resembling real robots. This demonstrates the adaptability of the method to varying motion requirements.

4.5 Joint Number Control

As discussed in Sec. 3.4.2, we introduce joint density as a conditional input to control the number of joints in the generated skeleton, allowing it to adapt to varying flexibility requirements. The GT joint count is normalized to the range $[0, 1]$ (by dividing by 60 and clamping), positionally embedded, and encoded with MLPs before being injected into the flow model via AdaLN modulation. During inference, the joint density condition can be adjusted to control the final number of skeleton joints using classifier-free guidance (CFG). We illustrate the results of joint number control in Fig. 11, using a CFG scale of 3.0. By adjusting the joint density, the model can synthesize skeletons with distinct degrees of freedom: a "high-density" can perform smooth, human-like motions – such as bending arms, twisting its head, and clenching its fists – while a "medium-density" variant retains limb flexibility but loses fine-grained manual dexterity. At the "sparse" extreme, the model yields a rigid, minimalist armature with significantly constrained motion, like a "real robot".

4.6 Ablation Study

In this section, we conduct an ablation study on the design choices discussed in the method section and explain why we selected the current technical approach. First, we investigate the confidence design, as detailed in Sec.3.2.2. High ambiguity in border regions makes confidence learning essential to ensure clean final generation results. Without confidence, the model fails to refine noisy predictions into a clean skeleton, as shown in the 3rd column of Fig.12. Even with Bayesian confidence [Kendall and Gal 2017], the self-adaptive learning approach cannot effectively resolve ambiguous regions, resulting in noisy and redundant bones and joints. In contrast, our method explicitly defines ambiguity in Eq.3 using a prior and supervises the confidence field directly during training, rather than relying on Bayesian learning's adaptive reconstruction loss. This results in a more effective confidence field, which integrates seamlessly with the confidence-weighted grouping algorithm (Alg.1) to consistently achieve accurate grouping results. The quantitative results in Tab. 4 confirm this trend: explicit confidence supervision improves structural correctness over both removing confidence and replacing it with Bayesian uncertainty learning, while SkinAE pretraining is critical for high-quality skin prediction.

Table 4. Quantitative ablation on confidence modeling and SkinAE pretraining. Lower is better for both metrics.

Method	Bayesian	w/o-Conf	w/o-SkinAE	Ours
Joint-GW ↓	0.310	0.337	0.383	**0.286**
Skin-KL ↓	3.174	3.187	5.138	**2.919**

We also analyze the impact of pretraining SkinAE. Without pretraining ("w/o SkinAE"), SkinAE becomes part of the structured latent auto-encoder ($\mathcal{E}_L$ and $\mathcal{D}_L$) and is trained jointly from scratch. This joint optimization lacks skin information in the input to the structured latent encoder, leading to sub-optimal feature alignment and hindered convergence. Consequently, the reconstruction of the joint field is negatively affected. As shown in the 4th column of Fig. 12, this results in poor skin quality for the tissue character and a broken skeleton for the goat. Our pretraining strategy ensures convergence of the structured latent auto-encoder and achieves sufficient accuracy for the generation decoding task.

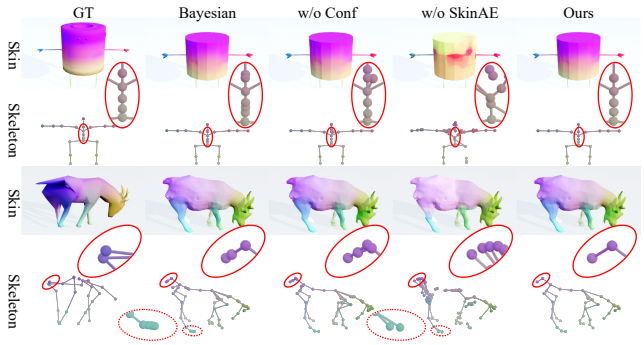


Fig. 12. Ablation study on confidence design and SkinAE pretraining. Without confidence learning or with Bayesian confidence, skeleton predictions are noisy and redundant (column 2,3). Without pretraining SkinAE, skin quality and skeleton structure degrade significantly (column 4). Our method ensures clean skeletons and accurate skin generation (column 5).

5 Conclusion

We introduced *AniGen*, a method for image-conditioned generation of animatable 3D assets that couples geometry synthesis with rig prediction in a single generative model. In contrast to "generate-then-rig" pipelines, *AniGen* learns a joint distribution over shape,

skeleton, and skin, and represents all three as fields to promote mutual consistency. Our core contribution, the S^3 field formulation, together with the confidence-decaying nearest joint-parent field and the dual skin feature field, enables reliable modeling of kinematic structure and skinning despite ambiguity and category diversity. *AniGen* leverages a structured latent auto-encoder and flow-based generation over sparse structures and structured latents to produce coherent rigged assets directly from images. Empirically, *AniGen* achieves clear gains over prior approaches in rigging quality and usability, and generalizes well to in-the-wild images spanning a wide range of object categories and visual styles. The revised experiments further show that these gains are achieved without materially sacrificing geometry fidelity relative to strong geometry-only generators. We believe that *AniGen* establishes a powerful foundation for the next generation of controllable 3D content creation, with broad applications for interactive graphics, embodied AI, VR/AR, digital twins, and animation/editing workflows.

Limitations and future work. One limitation of *AniGen* is its current focus on image-conditioned generation. Although effective, this setting does not fully reflect many practical use cases, where animatable shapes are often derived from captured videos in which motion dynamics and skeletal constraints are more explicitly observed. Extending *AniGen* to support video input could enable more robust and temporally consistent shape and skeleton generation, and also enable the production of animatable shapes directly aligned with the motions and structural cues present in the input video. Addressing this challenge is an important direction for future work and could substantially broaden the applicability of the method. A second limitation arises for articulated objects that require strict geometric alignment between rigid parts, such as the lid and base of a laptop. Although *AniGen* can infer a plausible hinge skeleton for such objects, small geometric mismatches may still produce visible gaps in closed configurations. Finally, the skeletons predicted by our current model primarily reflect the medial-axis tendencies present in the training data, rather than fully anatomically inspired production rigs. We emphasize that this limitation stems from the available data rather than from the representation itself. Since the proposed S^3 fields are continuous and volumetric, they are, in principle, capable of encoding richer sub-surface control structures, including production-style anatomical rigs, given access to suitable data.

References

J. A. Bærentzen, R. Abdrashitov, and K. Singh. 2014. Interactive Shape Modeling using a Skeleton-Mesh Co-Representation. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)* 33, 4 (2014).

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

Péter Borosán, Ming Jin, Doug DeCarlo, Yotam Gingold, and Andrew Nealen. 2012. RigMesh: automatic rigging for part-based shape modeling and deformation. *ACM Trans. Graph.* 31, 6, Article 198 (Nov. 2012), 9 pages. doi:10.1145/2366145.2366217

Honghua Chen, Yushi Lan, Yongwei Chen, and Xingang Pan. 2025a. ArtiLatent: Realistic Articulated 3D Object Generation via Structured Latents. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*. 1–11.

Rui Chen, Jianfeng Zhang, Yixun Liang, Guan Luo, Weiyu Li, Jiarui Liu, Xiu Li, Xiaoxiao Long, Jiashi Feng, and Ping Tan. 2025c. Dora: Sampling and benchmarking for 3d shape variational auto-encoders. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 16251–16261.

Yiwen Chen, Zhihao Li, Yikai Wang, Hu Zhang, Qin Li, Chi Zhang, and Guosheng Lin. 2025b. Ultra3d: Efficient and high-fidelity 3d generation with part attention. *arXiv preprint arXiv:2507.17745* (2025).

Peng Dai, Feitong Tan, Qiangeng Xu, David Futschik, Ruofei Du, Sean Fanello, XIAOJUAN QI, and Yinda Zhang. 2025. SVG: 3D Stereoscopic Video Generation via Denoising Frame Matrix. In *ICLR*.

Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. 2023a. Objaverse-xl: A universe of 10m+ 3d objects. *Advances in Neural Information Processing Systems* 36 (2023), 35799–35813.

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

Yufan Deng, Yuhao Zhang, Chen Geng, Shangzhe Wu, and Jiajun Wu. 2025. 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*. 1–10.

Ruiqi Gao, Aleksander Holynski, Philipp Henzler, Arthur Brussee, Ricardo Martin Brualla, Pratul Srinivasan, Jonathan Barron, and Ben Poole. 2024. CAT3D: Create Anything in 3D with Multi-View Diffusion Models. *Advances in Neural Information Processing Systems* 37 (2024), 75468–75494.

Inbar Gat, Sigal Raab, Guy Tevet, Yuval Reshef, Amit Haim Bermano, and Daniel Cohen-Or. 2025. Anytop: Character animation diffusion with any topology. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*. 1–10.

Clark R Givens and Rae Michael Shortt. 1984. A class of Wasserstein metrics for probability distributions. *Michigan Mathematical Journal* 31, 2 (1984), 231–240.

Zhiyang Guo, Ori Zhang, Jax Xiang, Alan Zhao, Wengang Zhou, and Houqiang Li. 2025. Make-It-Poseable: Feed-forward Latent Posing Model for 3D Humanoid Character Animation. *arXiv preprint arXiv:2512.16767* (2025).

Xianglong He, Zi-Xin Zou, Chia-Hao Chen, Yuan-Chen Guo, Ding Liang, Chun Yuan, Wanli Ouyang, Yan-Pei Cao, and Yangguang Li. 2025. SparseFlex: High-Resolution and Arbitrary-Topology 3D Shape Modeling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 14822–14833.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems* 33 (2020), 6840–6851.

Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. 2023. LRM: Large Reconstruction Model for Single Image to 3D. In *The Twelfth International Conference on Learning Representations*.

Binbin Huang, Haobin Duan, Yiqun Zhao, Zibo Zhao, Yi Ma, and Shenghua Gao. 2025a. CUPID: Generative 3D Reconstruction via Joint Object and Pose Modeling. *arXiv preprint arXiv:2510.20776* (2025).

Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. 2024. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 4220–4230.

Zehuan Huang, Haoran Feng, Yang-Tian Sun, Yuan-Chen Guo, Yan-Pei Cao, and Lu Sheng. 2025b. Animax: Animating the inanimate in 3d with joint video-pose diffusion models. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*. 1–13.

Ajay Jain, Ben Mildenhall, Jonathan T Barron, Pieter Abbeel, and Ben Poole. 2022. Zero-shot text-guided object generation with dream fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 867–876.

Alex Kendall and Yarin Gal. 2017. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems* 30 (2017).

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3D Gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.* 42, 4 (2023), 139–1.

Jinhyeok Kim, Jaehun Bang, Seunghyun Seo, and Kyungdon Joo. 2025. Rigidity-Aware 3D Gaussian Deformation from a Single Image. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*. 1–11.

Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *ICLR*.

Jiahao Li, Hao Tan, Kai Zhang, Zexiang Xu, Fujun Luan, Yinghao Xu, Yicong Hong, Kalyan Sunkavalli, Greg Shakhnarovich, and Sai Bi. 2024. Instant3D: Fast Text-to-3D with Sparse-view Generation and Large Reconstruction Model. In *ICLR*.

Peizhuo Li, Kfir Aberman, Rana Hanocka, Libin Liu, Olga Sorkine-Hornung, and Baoquan Chen. 2021. Learning Skeletal Articulations with Neural Blend Shapes. *ACM Transactions on Graphics (TOG)* 40, 4 (2021), 130.

Ruining Li, Yuxin Yao, Chuanxia Zheng, Christian Rupprecht, Joan Lasenby, Shangzhe Wu, and Andrea Vedaldi. 2025a. Particulate: Feed-Forward 3D Object Articulation. *arXiv preprint arXiv:2512.11798* (2025).

Yangguang Li, Zi-Xin Zou, Zexiang Liu, Dehu Wang, Yuan Liang, Zhipeng Yu, Xingchao Liu, Yuan-Chen Guo, Ding Liang, Wanli Ouyang, et al. 2025c. Triposg: High-fidelity 3d shape synthesis using large-scale rectified flow models. *arXiv preprint arXiv:2502.06608* (2025).

Zhibing Li, Mengchen Zhang, Tong Wu, Jing Tan, Jiaqi Wang, and Dahua Lin. 2025b. SS4D: Native 4D Generative Model via Structured Spacetime Latents. *ACM Transactions on Graphics (TOG)* 44, 6 (2025), 1–12.

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

Lijuan Liu, Youyi Zheng, Di Tang, Yi Yuan, Changjie Fan, and Kun Zhou. 2019. Neuroskinning: Automatic skin binding for production characters with deep graph networks. *ACM Transactions on Graphics (ToG)* 38, 4 (2019), 1–12.

Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. 2023. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF international conference on computer vision*. 9298–9309.

Xingchao Liu, Chengyue Gong, and Qiang Liu. 2022. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003* (2022).

Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. 2024. SyncDreamer: Generating Multiview-consistent Images from a Single-view Image. In *ICLR*.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*. 10012–10022.

Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, et al. 2024. Wonder3d: Single image to 3d using cross-domain diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 9970–9980.

Jing Ma and Dongliang Zhang. 2023. TARig: Adaptive template-aware neural rigging for humanoid characters. *Computers & Graphics* 114 (2023), 158–167.

David Marr and Herbert Keith Nishihara. 1978. Representation and recognition of the spatial organization of three-dimensional shapes. *Proceedings of the Royal Society of London. Series B. Biological Sciences* 200, 1140 (1978), 269–294.

Facundo Mémoli. 2011. Gromov–Wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics* 11, 4 (2011), 417–487.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. 2021. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* 65, 1 (2021), 99–106.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. 2024. DINOv2: Learning Robust Visual Features without Supervision. *Transactions on Machine Learning Research Journal* (2024).

Karran Pandey, J. Andreas Bærentzen, and Karan Singh. 2022. Face Extrusion Quad Meshes. In *ACM SIGGRAPH 2022 Conference Proceedings* (Vancouver, BC, Canada) *(SIGGRAPH '22)*. Association for Computing Machinery, New York, NY, USA, Article 10, 9 pages. doi:10.1145/3528233.3530754

William Peebles and Saining Xie. 2023. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*. 4195–4205.

Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. 2023. DreamFusion: Text-to-3D using 2D Diffusion. In *ICLR*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PmLR, 8748–8763.

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

Tianchang Shen, Jacob Munkberg, Jon Hasselgren, Kangxue Yin, Zian Wang, Wenzheng Chen, Zan Gojcic, Sanja Fidler, Nicholas Sharp, and Jun Gao. 2023. Flexible isosurface extraction for gradient-based mesh optimization. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–16.

Yichun Shi, Peng Wang, Jianglong Ye, Long Mai, Kejie Li, and Xiao Yang. 2023. MVDream: Multi-view Diffusion for 3D Generation. In *ICLR*.

Chaoyue Song, Xiu Li, Fan Yang, Zhongcong Xu, Jiacheng Wei, Fayao Liu, Jiashi Feng, Guosheng Lin, and Jianfeng Zhang. 2025a. Puppeteer: Rig and Animate Your 3D Models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Chaoyue Song, Jianfeng Zhang, Xiu Li, Fan Yang, Yiwen Chen, Zhongcong Xu, Jun Hao Liew, Xiaoyang Guo, Fayao Liu, Jiashi Feng, and Guosheng Lin. 2025b. MagicArticulate: Make Your 3D Models Articulation-Ready. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*. 15998–16007.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021. Denoising Diffusion Implicit Models. In *ICLR*.

Mingze Sun, Junhao Chen, Junting Dong, Yurun Chen, Xinyu Jiang, Shiwei Mao, Puhua Jiang, Jingbo Wang, Bo Dai, and Ruqi Huang. 2025. Drive: Diffusion-based rigging empowers generation of versatile and expressive characters. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 21170–21180.

Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. 2024. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In *European Conference on Computer Vision*. Springer, 1–18.

Dmitry Tochilkin, David Pankratz, Zexiang Liu, Zixuan Huang, , Adam Letts, Yangguang Li, Ding Liang, Christian Laforte, Varun Jampani, and Yan-Pei Cao. 2024. TripoSR: Fast 3D Object Reconstruction from a Single Image. *arXiv preprint arXiv:2403.02151* (2024).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

Haoyu Wang, Shaoli Huang, Fang Zhao, Chun Yuan, and Ying Shan. 2023a. Hmc: Hierarchical mesh coarsening for skeleton-free motion retargeting. *arXiv preprint arXiv:2303.10941* (2023).

Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. 2023b. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *Advances in neural information processing systems* 36 (2023), 8406–8441.

Rundi Wu, Ruiqi Gao, Ben Poole, Alex Trevithick, Changxi Zheng, Jonathan T Barron, and Aleksander Holynski. 2025a. Cat4d: Create anything in 4d with multi-view video diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 26057–26068.

Shuang Wu, Youtian Lin, Feihu Zhang, Yifei Zeng, Jingxi Xu, Philip Torr, Xun Cao, and Yao Yao. 2024a. Direct3d: Scalable image-to-3d generation via 3d latent diffusion transformer. *Advances in Neural Information Processing Systems* 37 (2024), 121859–121881.

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

Zijie Wu, Chaohui Yu, Fan Wang, and Xiang Bai. 2025b. AnimateAnyMesh: A Feed-Forward 4D Foundation Model for Text-Driven Universal Mesh Animation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 13557–13568.

Jianfeng Xiang, Xiaoxue Chen, Sicheng Xu, Ruicheng Wang, Zelong Lv, Yu Deng, Hongyuan Zhu, Yue Dong, Hao Zhao, Nicholas Jing Yuan, et al. 2025a. Native and Compact Structured Latents for 3D Generation. *arXiv preprint arXiv:2512.14692* (2025).

Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. 2025b. Structured 3d latents for scalable and versatile 3d generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 21469–21480.

Tianyi Xie, Yunuo Chen, Yaowei Guo, Yin Yang, Bolei Zhou, Demetri Terzopoulos, Ying Jiang, and Chenfanfu Jiang. 2025. AnimaMimic: Imitating 3D Animation from Video Priors. *arXiv preprint arXiv:2512.14133* (2025).

Yinghao Xu, Hao Tan, Fujun Luan, Sai Bi, Peng Wang, Jiahao Li, Zifan Shi, Kalyan Sunkavalli, Gordon Wetzstein, Zexiang Xu, et al. 2024. DMV3D: Denoising Multi-view Diffusion Using 3D Large Reconstruction Model. In *ICLR*.

Zhan Xu, Yang Zhou, Evangelos Kalogerakis, Chris Landreth, and Karan Singh. 2020. RigNet: neural rigging for articulated characters. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 58–1.

Zhan Xu, Yang Zhou, Evangelos Kalogerakis, and Karan Singh. 2019. Predicting Animation Skeletons for 3D Articulated Models via Volumetric Nets. *2019 International Conference on 3D Vision (3DV)* (2019), 298–307. https://api.semanticscholar.org/CorpusID:201309034

Zhan Xu, Yang Zhou, Li Yi, and Evangelos Kalogerakis. 2022. Morig: Motion-aware rigging of character meshes from point clouds. In *SIGGRAPH Asia 2022 conference papers*. 1–9.

Jiawei Yang, Tianhong Li, Lijie Fan, Yonglong Tian, and Yue Wang. 2025. Latent denoising makes good visual tokenizers. *arXiv preprint arXiv:2507.15856* (2025).

Jingfeng Yao, Yuda Song, Yucong Zhou, and Xinggang Wang. 2025. Towards Scalable Pre-training of Visual Tokenizers for Generation. *arXiv preprint arXiv:2512.13687* (2025).

Xin Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Song-Hai Zhang, and Xiaojuan Qi. 2023. Text-to-3D with Classifier Score Distillation. In *ICLR*.

Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 2023. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Transactions On Graphics (TOG)* 42, 4 (2023), 1–16.

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

Jia-Qi Zhang, Miao Wang, Fu-Cheng Zhang, and Fang-Lue Zhang. 2024a. Skinned motion retargeting with preservation of body part relationships. *IEEE Transactions on Visualization and Computer Graphics* (2024).

Longwen Zhang, Ziyu Wang, Qixuan Zhang, Qiwei Qiu, Anqi Pang, Haoran Jiang, Wei Yang, Lan Xu, and Jingyi Yu. 2024b. Clay: A controllable large-scale generative model for creating high-quality 3d assets. *ACM Transactions on Graphics (TOG)* 43, 4 (2024), 1–20.

Zi-Xin Zou, Zhipeng Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Yan-Pei Cao, and Song-Hai Zhang. 2024. Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10324–10335.