Optimizing Mesh Animation from Video via Shape Flow Guidance

Jingqiao Xiu[1], Yicong Li[2⋆], and Angela Yao[1]

[1] National University of Singapore
[2] University of Science and Technology of China

Abstract. Optimizing vertex deformations from video for mesh animation is constrained by rendering-based reconstruction losses. While existing approaches improve mesh representations, supervision signals, or animation paradigms, their supervision remains confined to the 2D domain. Such 2D supervision is limited: it provides no signal for occluded regions and only indirect cues for visible areas. Consequently, these methods often suffer from severe shape and motion artifacts. To address this limitation, we propose Shape Flow Guidance (SFG), a sequence of 3D shapes derived from videos, which serves as explicit 3D supervision for mesh animation. Specifically, SFG is elicited by intervening in the sampling process of a pretrained mesh generator in a training-free manner. We further tailor a skeletal animation model that separates local deformation from global transformation. This model enables SFG to guide complex local motion while reserving rendering-based losses for simple global motion. Extensive experiments confirm that our method significantly outperforms prior work qualitatively, quantitatively, and in terms of processing speed. Qualitative results are available on our project page: `https://sfgmesh.pages.dev/`.

Keywords: Shape Flow Guidance · Video-driven Mesh Animation

1 Introduction

Extracting mesh animations from monocular videos holds significant research and practical value. To ensure compatibility with established graphics pipelines (e.g., skeletal rigging and texture mapping), the animated meshes must maintain strict topological consistency (i.e., identical sets of vertices and faces). A common approach is to optimize per-vertex displacements relative to a static mesh, guided by reconstruction losses that compare the rendered mesh with the reference video. However, mesh representations suffer from inherent topological rigidity and gradient sparsity caused by discontinuous rasterization. As a result, the optimization process often struggles to capture complex deformations.

Existing methods address these optimization challenges through innovations in representation, supervision, and paradigm (see the upper two rows in Figure 1). Some approaches [18] introduce hybrid Gaussian-mesh representations

⋆ Corresponding author.

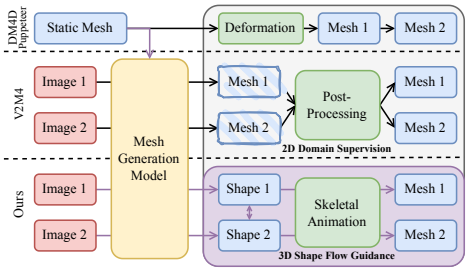

Fig. 1: Comparison of mesh animation paradigms. The optimization process of prior methods relies heavily on 2D supervision. In contrast, as indicated by purple arrows, our method distills shape flow guidance from the mesh generation model to provide direct 3D supervision.

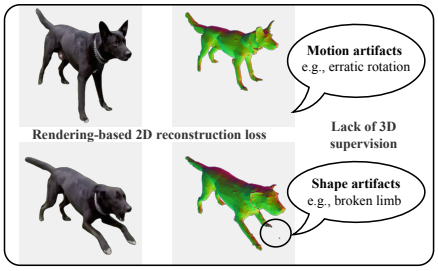

Fig. 2: Due to the inherent ambiguity and indirect nature of 2D rendering supervision, the state-of-the-art method [4] exhibits severe shape and motion artifacts (right), even in regions clearly visible in the video (left), highlighting the need for 3D supervision.

to facilitate the optimization of rendering-based losses, but the volumetric and unstructured nature of Gaussian primitives compromises surface quality during animation. Other methods [41] incorporate video tracking priors as advanced supervision signals, employing projection-based losses to align projected 3D mesh vertices with 2D tracked keypoints. While providing more direct guidance, this 2D supervision still suffers from limitations similar to those of rendering-based losses. Another line of work [4] bypasses deformation optimization by generating per-frame meshes independently and then post-processing them to enforce topological consistency. However, its frame-to-frame registration still relies heavily on 2D rendering losses and remains susceptible to large, non-rigid deformations, resulting in unnatural artifacts. In summary, these advancements offer only partial solutions and do not fully overcome the limitations of 2D supervision.

We argue that projecting 3D structures into 2D to leverage 2D supervision is fundamentally limited. First, occluded or unseen regions of the animated meshes receive no supervision, as variations in these areas do not affect the 2D loss. This often leads to uncontrolled artifacts and breaks geometric consistency with the static mesh. Second, as illustrated in Figure 2, severe shape and motion artifacts emerge even in visible areas. This is because 2D supervision provides only an indirect signal for 3D deformation, where optimization can only infer 3D vertex displacements from their 2D projections. This ambiguity leads to lengthy optimization and unrealistic animations, as optimizing the 2D loss does not guarantee a plausible 3D deformation.

In this work, we introduce Shape Flow Guidance (SFG) to provide explicit 3D shape and motion supervision during optimization, where shape flow is defined as a sequence of temporally consistent 3D shapes. To derive such flow from 2D videos, we intervene in the sampling process of a rectified-flow-based generator [28, 53]. Our intervention forces all shapes to originate from the static mesh and directs them to their respective destinations as dictated by the corresponding video frames, while allowing them to interact with their temporal neighbors. This

is achieved by tailoring a 2D flow field whose velocity can be derived from the learned 1D velocity field of the mesh generator without additional training. While sampling along our flow field yields a sequence of consistent guidance shapes, one issue remains: the guidance shapes inherit the location and orientation of the static mesh, capturing only local deformations while overlooking the global transformations present in the video.

To resolve this issue, we decouple animation into local deformation and global transformation using an automatic rigging module [66]. This yields a skeletal animation model that naturally separates root joint transformation (global motion) from joint deformations in root space (local motion). This decoupling allows us to apply a targeted supervision strategy for each component: SFG drives complex local deformations in 3D, while 2D losses supervise only the simpler global transformation. Consequently, this design prevents 2D supervision from interfering with local deformation and focuses it on global motion. To further enhance the animation model, a post-skinning correction module is integrated to capture fine-grained surface details beyond standard skinning. As shown in the bottom row of Figure 1, our framework integrates shape flow guidance with this enhanced animation model, realistically transferring 2D motion cues from monocular video onto a topology-consistent 4D mesh by optimizing the deformation parameters.

Our contributions can be summarized as follows:

- We propose Shape Flow Guidance (SFG), a novel 3D supervision signal derived from a pretrained generative model, which provides explicit and temporally consistent 3D guidance for animating a static mesh from video.
- We introduce a framework that decouples animation into local and global motion, where shape flow guides local deformations while 2D rendering losses are confined to supervising global transformations.
- Extensive experiments demonstrate that our method significantly outperforms previous methods, establishing a new state of the art. The resulting assets are directly usable in modern 3D engines.

2 Related Work

3D Generation. 3D content creation has advanced rapidly with methods that lift generative priors from pretrained 2D diffusion models [36, 38] to the 3D domain. A seminal work, DreamFusion [33], introduces Score Distillation Sampling (SDS), an optimization-based approach that iteratively refines a 3D representation (e.g., NeRF [32]) to match the output of a 2D model from various viewpoints. This paradigm has inspired follow-up research aimed at either improving 3D representations [5, 6, 21, 45, 46, 61] or enhancing distillation processes [20, 49, 50]. However, lacking inherent 3D information, these methods often suffer from geometric inconsistencies, notably the multi-face Janus problem. To mitigate this issue, another line of work [27, 29, 30, 39, 48] focuses on training multi-view diffusion models on large-scale 3D asset datasets [7, 8].

These methods still require costly per-scene reconstruction after generating multi-view images. By comparison, feedforward approaches [9, 14, 16, 17, 25,

26, 44, 56–60, 67] directly generate 3D representations from one or multiple images. Although these methods can efficiently produce implicit fields or Gaussian splats [13], polygonal meshes remain the industry standard for downstream applications, owing to their explicit surface representation and compatibility with established pipelines. To meet this demand, recent works [15,53,68,70] have made significant progress in direct mesh generation in compressed latent spaces [53,64]. While these models excel at creating high-fidelity static meshes, making them animatable remains a key challenge. In this work, we aim to leverage the shape priors learned by these generative models to facilitate mesh animation.

4D Generation. Recent success in 3D generation has inspired a surge of research on dynamic 4D content generation. The initial wave of 4D generation methods [1, 2, 12, 22, 40, 51, 63, 71] focuses on distilling spatio-temporal priors from pretrained image, multi-view, or video diffusion models. However, these distillation-based approaches often impose high computational demands, require prolonged optimization times, and suffer from artifacts such as flickering. A subsequent line of work [11, 19, 35, 54, 55, 62, 65] investigates fine-tuning 3D or video generative models on large-scale 4D data. This strategy improves spatio-temporal consistency by synthesizing multi-view dynamic videos. However, these methods do not directly output dynamic meshes, which are crucial for integration into standard 3D animation pipelines.

Recent approaches [3, 37, 52] explore feedforward 4D mesh generation by training auxiliary animation models. While promising, these methods are resource-intensive and fail to capitalize on the rich priors of 3D generative models. Our work instead follows the setting of optimization-based animation from video [4, 18, 41], which does not require a separate training stage. Specifically, the first two utilize 3D generative models to generate an initial mesh, followed by the optimization of deformation parameters. In contrast, the last one independently generates each frame using a 3D generative model and relies on a complex post-processing pipeline to enforce cross-frame consistency. A fundamental limitation common to these methods [4, 18, 41] is their primary reliance on 2D supervision signals to drive animation. Our method addresses this by leveraging 3D generative models to distill explicit 3D guidance, enabling mesh deformation to be driven directly in 3D space with improved effectiveness and efficiency.

3 Preliminaries

Mesh Animation. A static mesh $\mathcal{M}^0$ is defined by a set of N vertices $\mathcal{V}^0 \in \mathbb{R}^{N\times 3}$, a collection of triangular faces $\mathcal{F}^0 \subset \{1,\dots,N\}^3$, and an associated texture map $\mathcal{T}^0$. Its animation is represented by a sequence of per-vertex deformations, $\{\mathcal{D}^l \in \mathbb{R}^{N\times 3}\}_{l=1}^{L}$, over L frames. Applying these deformations to the static mesh vertices yields a sequence of dynamic meshes $\{\mathcal{M}^l = (\mathcal{V}^l, \mathcal{F}^0, \mathcal{T}^0)\}_{l=1}^{L}$, where the vertex positions at frame l are given by $\mathcal{V}^l = \mathcal{V}^0 + \mathcal{D}^l$. Importantly, these dynamic meshes share the same topology and texture map.

Mesh Rigging. To reduce the large number of degrees of freedom inherent in per-vertex deformations, it is conventional to rig a static mesh to a struc-

turally plausible skeleton. Formally, a skeleton consists of a set of J joint locations, $\mathcal{J} \in \mathbb{R}^{J\times 3}$, and a kinematic hierarchy defined by bone connections, $\mathcal{B} \subset \{1,\ldots,J\}^2$. While traditional rigging approaches are labor-intensive and demand considerable expertise, recent automatic rigging methods [24,42,66] formulate skeleton generation as a sequence modeling problem, training an autoregressive transformer on large-scale 3D datasets with rigging annotations. These models naturally accommodate variable numbers of bones across different 3D models and generalize effectively to diverse object categories, without requiring prior assumptions about shape orientation or topology.

Mesh Skinning. The skeleton must be bound to the mesh vertices to allow its motion to deform the mesh. Automatic rigging frameworks [24, 42, 66] accomplish this by predicting skinning weights, which quantify the influence of each skeleton joint on every mesh vertex. These weights are fundamental to canonical deformation models such as Linear Blend Skinning (LBS) [31], where the deformed position of a vertex is computed as a weighted blend of transformations applied by its influencing joints:

$$\mathrm{LBS}(\mathbf{v}) = \sum_j w_j \mathbf{T}_j \mathbf{B}_j^{-1} \mathbf{v}, \tag{1}$$

where $\mathbf{v}$ denotes the homogeneous coordinate of the vertex in the bind pose. For each joint j, w_j is the skinning weight assigned to vertex $\mathbf{v}$, with the constraint that $\sum_j w_j = 1$. The matrix $\mathbf{B}_j \in \mathbb{R}^{4\times 4}$ is the global transformation of joint j in the bind pose, while $\mathbf{T}_j \in \mathbb{R}^{4\times 4}$ is the global transformation of the same joint in the target pose. Consequently, the product $\mathbf{T}_j\mathbf{B}_j^{-1}$ represents the net transformation applied to the vertex by the motion of joint j.

Mesh Generation. Mesh generation models typically employ a Rectified Flow (RF) framework [23, 28] in latent space [53, 64], which models a velocity field $v(x_t, t, I)$ that defines a straight-line transport path between a standard Gaussian distribution, $x_0 \sim \mathcal{N}(0, \mathbf{I})$, and the target data distribution x_1. Given an image condition I, the corresponding shape latent is generated by integrating the velocity field: $\frac{dx_t}{dt} = v(x_t, t, I)$, which is then decoded into the mesh.

4 Shape Flow Guidance

Task Definition. Given a static mesh $\mathcal{M}^0$ and a monocular video $\mathcal{I} = \{\mathcal{I}^l\}_{l=0}^L$ that demonstrates its animation from a fixed viewpoint, our objective is to optimize the per-frame vertex deformations $\{\mathcal{D}^l\}_{l=1}^L$, forming a consistent dynamic mesh sequence $\{\mathcal{M}^l = (\mathcal{V}^l, \mathcal{F}^0, \mathcal{T}^0)\}_{l=0}^L$. In cases where the static mesh is unavailable, we leverage a pretrained 3D generative model [53] to produce a high-quality static mesh conditioned on the initial frame.

Framework Overview. As illustrated in Figure 3, our optimization process centers on the proposed *Shape Flow Guidance (SFG)*. We elicit SFG from the mesh generation model to lift 2D motion from an input video to 3D motion on the static shape, yielding a temporally coherent shape flow. This guidance is

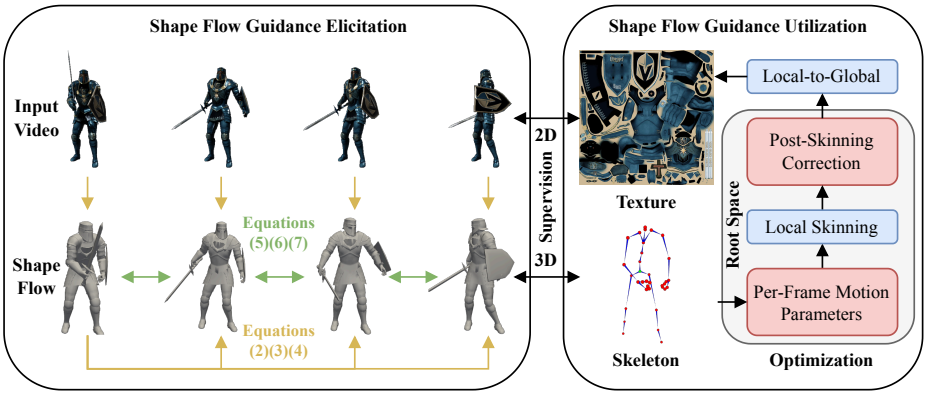

Fig. 3: Overview of our shape-flow-guided mesh animation framework. Shape flow is elicited from a 3D generative model conditioned on the input video. The yellow arrows indicate that sampling trajectories originate from the static shape, while the green arrows represent interactions among adjacent-frame trajectories during sampling. This shape flow then serves as the primary guidance for animation optimization within a skeleton-based model that decouples global and local motion. The supervision consists of a 3D component that guides the deforming meshes in root space toward SFG, and a 2D component that aligns mesh renderings with the input video.

then incorporated through a tailored skeletal animation pipeline that decouples local and global motion, where the skeleton and skinning weights are obtained from the static mesh using an automatic rigging model. The framework optimizes parameters for both per-frame skeletal motion and a post-skinning correction network. First, we factor out the root transformation from the skeletal motion, and the transformations of the non-root joints drive the mesh deformation via LBS. Subsequently, the post-skinning correction network compensates for residual deformation in root space. The deformation before applying the root transformation is supervised by bidirectional point-mesh distances to the SFG. Finally, the globally transformed mesh, which inherits a consistent texture, is differentiably rendered into color images and silhouettes to apply 2D reconstruction supervision. We first elaborate on the SFG elicitation process in Section 4.1, followed by its utilization within our optimization framework in Section 4.2.

4.1 Shape Flow Guidance Elicitation

In this section, we introduce a training-free method for eliciting a temporally coherent shape flow from pretrained 3D generative models, which serves as 3D supervision for mesh deformation. While the 3D generator can create 3D shapes from individual video frames, the results lack temporal consistency. Shapes often differ arbitrarily, especially in regions occluded from the image condition. Moreover, each shape is individually centered and randomly oriented, irrespective of its pose in the image. Directly concatenating these shapes thus leads to significant inter-frame jitter.

To encourage consistency, we anchor the generation of all subsequent shapes to the initial shape. This strategy compels each subsequent shape to evolve from this common origin, guided by its corresponding image. Consequently, the generated shapes inherit the base geometry, location, and orientation of the initial frame, while still capturing the local animations depicted in the video. Specifically, we construct a set of synchronized latent paths, $\{s_t^l\}_{l=1}^L$, that all originate from the initial shape x_1^0 and evolve toward their respective target shapes x_1^l, for $l \in \{1, \ldots, L\}$. These paths are formulated as:

$$s_t^l = x_1^0 + x_t^l - x_t^0, \tag{2}$$

With shared initial noise across frames (i.e., $x_0^l = x_0^0$), each path s_t^l transitions smoothly from $s_0^l = x_1^0$ at $t = 0$ to its target $s_1^l = x_1^l$ at $t = 1$. The velocity field governing each path can be derived as the difference between the original velocity fields $v(x_t, t, I)$ defined in the 3D generative model:

$$\frac{ds_t^l}{dt} = v(x_t^l, t, I^l) - v(x_t^0, t, I^0). \tag{3}$$

While evolving along the trajectories in Equation (3) yields shapes that are individually consistent with the initial shape, temporal jitter may still arise between adjacent frames, as there is no interaction mechanism or smoothness enforcement across the frame dimension. To address this, we extend the 1D flow of Equation (3) to a unified 2D flow field over both sampling time t and frame index l. Inspired by transport phenomena, we model the inter-path dynamics using a partial differential equation (PDE), specifically an advection-diffusion equation (ADE):

$$\frac{\partial s_t^l}{\partial t} + \lambda_1 \mathrm{A}(s_t^l) - \lambda_2 \mathrm{D}(s_t^l) = v(x_t^l, t, I^l) - v(x_t^0, t, I^0), \tag{4}$$

where $\mathrm{A}(\cdot)$ and $\mathrm{D}(\cdot)$ denote the advection and diffusion operators along the frame dimension, respectively, and λ_1, λ_2 control their strengths.

Advection, represented by the first-order derivative $\frac{\partial s_t^l}{\partial l}$, models directional propagation. This biases s_t^l toward its predecessor s_t^{l-1}.

Diffusion, represented by the second-order derivative $\frac{\partial^2 s_t^l}{\partial l^2}$, models isotropic smoothing. This encourages a continuous transition among s_t^{l-1}, s_t^l, and s_t^{l+1}.

By discretizing the frame dimension l, we approximate the first-order derivative using finite differences:

$$\mathrm{A}(s_t^l) = \frac{\partial s_t^l}{\partial l} \approx s_t^l - s_t^{l-1}, \tag{5}$$

and approximate the second-order derivative using the discrete Laplacian:

$$\mathrm{D}(s_t^l) = \frac{\partial^2 s_t^l}{\partial l^2} \approx s_t^{l+1} - 2s_t^l + s_t^{l-1}. \tag{6}$$

Substituting these approximations into the continuous PDE yields a system of coupled ordinary differential equations (ODEs). The resulting velocity field, incorporating both advective and diffusive terms, is given by:

$$\frac{ds_t^l}{dt} = \underbrace{v(x_t^l, t, I^l) - v(x_t^0, t, I^0)}_{\text{Time Evolution}} - \underbrace{\lambda_1(s_t^l - s_t^{l-1})}_{\text{Advection Term}} + \underbrace{\lambda_2(s_t^{l+1} - 2s_t^l + s_t^{l-1})}_{\text{Diffusion Term}}. \quad (7)$$

For boundary handling, we use the latent trajectory of the static mesh as temporal padding at the first animation frame, i.e., setting s_t^0 to the static trajectory. At the final frame, where s_t^{L+1} is unavailable, we omit the diffusion term.

While sampling from this coupled 2D flow field yields a sequence of shapes with strong temporal consistency, they are unsuitable as final animated meshes due to two key limitations of the 3D generative model. First, it operates in latent space, making direct control over the explicit mesh structure difficult. Thus, the resulting guidance shapes still exhibit topological variations, such as different vertex counts and connectivity. Second, the output shapes are confined to a unit cube, which inevitably discards global motion. Therefore, rather than using this sequence as the final output, we employ it as direct 3D guidance to optimize the vertex displacements of the static mesh.

4.2 Shape Flow Guidance Utilization

The elicited shape flow is consistently posed with the initial frame, meaning it exclusively captures local deformations but does not contain global translations or rotations. To fully utilize this guidance, it is necessary to decouple global transformations from local deformations. We achieve this by articulating the static mesh and adopting an enhanced skeletal animation model. Specifically, we parameterize the per-frame animation using the global transformation of a root joint, $[\mathbf{R}_0, \mathbf{t}_0]$, and the local rotations of all non-root joints, $\{\mathbf{R}_j\}_{j=1}^{J-1}$. We then reformulate standard LBS in Equation (1) by factoring out the root transformation matrix $\mathbf{T}_0 = [\mathbf{R}_0|\mathbf{t}_0]$:

$$\text{LBS}(\mathbf{v}) = \sum_{j=0}^{J-1} w_j \mathbf{T}_j \mathbf{B}_j^{-1} \mathbf{v} = \mathbf{T}_0 \sum_{j=0}^{J-1} w_j \mathbf{T}'_j \mathbf{B}_j^{-1} \mathbf{v}, \quad (8)$$

where $\mathbf{T}'_j$ denotes the joint transformation relative to the root joint's coordinate system (i.e., root space). These relative transformations are computed using forward kinematics from the local joint rotations along the kinematic tree. This formulation effectively isolates the local deformation in root space, allowing it to be supervised independently of the global transformation.

However, the LBS model inherently lacks the non-linear details required to capture localized deformations. To address this, we propose a *Post-Skinning Correction* network, an MLP that represents a pose-dependent, per-vertex residual offset field to compensate for the missing effects of skeletal animation. Importantly, the correction is also performed in per-frame root space. Our animation model, parameterized by per-frame motion parameters and an MLP deformation field, is optimized using the following three types of loss functions.

For the local deformation, we define a shape loss, $\mathcal{L}_{\text{shape}}$, as the bidirectional distance between the deformed meshes and the shape flow:

$$\mathcal{L}_{\text{shape}}(\mathcal{P}, \mathcal{F}) = \underbrace{\frac{1}{N}\sum_{p\in\mathcal{P}} \min_{f\in\mathcal{F}} \mathrm{d}^2(p, f)}_{\text{point-to-mesh}} + \underbrace{\frac{1}{M}\sum_{f\in\mathcal{F}} \min_{p\in\mathcal{P}} \mathrm{d}^2(p, f)}_{\text{mesh-to-point}}, \tag{9}$$

where $\mathcal{P} \in \mathbb{R}^{N\times 3}$ is the set of points sampled on the deformed mesh in root space, $\mathcal{F}$ is the set of M faces of shape flow meshes, and $\mathrm{d}(\cdot,\cdot)$ measures the closest distance between a point and a mesh face.

Equation (9) requires surface samples on the deformed mesh to remain consistent across optimization steps. Otherwise, resampling causes noisy gradients, leading to erratic updates and poor convergence. To avoid this issue, we introduce a *consistent surface sampling* strategy. Specifically, we pre-sample N barycentric anchors, $\{(\alpha_n, \beta_n)\}_{n=1}^{N}$, on the static mesh. Here, $\alpha_n = (\alpha_n^1, \alpha_n^2, \alpha_n^3)$ denotes the vertex indices of a triangle face sampled in proportion to its surface area, and β_n denotes the barycentric coordinates, which are computed using random variables $u, v \sim \mathcal{U}(0, 1)$ to ensure uniform coverage:

$$\beta_n = (1 - \sqrt{u}, \sqrt{u}(1 - v), \sqrt{u}v). \tag{10}$$

These barycentric anchors remain fixed throughout optimization. Given vertex positions derived from the current deformation parameters, we determine the position of each anchor $\mathbf{p}_n$ on the deformed mesh using barycentric interpolation:

$$\mathbf{p}_n = \beta_n^1 \mathbf{v}_{\alpha_n^1} + \beta_n^2 \mathbf{v}_{\alpha_n^2} + \beta_n^3 \mathbf{v}_{\alpha_n^3}. \tag{11}$$

The deformed anchors serve as sampled surface points that consistently track the same local surface regions across optimization steps, which stabilizes gradients and promotes reliable convergence.

For global motion, we incorporate rendering-based 2D supervision that computes a reconstruction loss, $\mathcal{L}_{\text{recon}}$, between the rendered color images and silhouettes of the animated mesh and those of the reference video. In cases where the static mesh is generated, we estimate the camera parameters such that the rendered static mesh aligns with the initial video frame, analogous to V2M4 [4].

We introduce several regularization terms: A joint smoothness loss, $\mathcal{L}_{\text{joint}}$, penalizes abrupt changes in joint transformations between consecutive frames, effectively reducing temporal jitter and ensuring smooth motion over time. An As-Rigid-As-Possible (ARAP) loss [43], $\mathcal{L}_{\text{arap}}$, preserves local rigidity by discouraging non-uniform stretching and compression, thereby promoting the structural integrity of the mesh under deformation. In addition, a normal consistency (NC) loss, $\mathcal{L}_{\text{nc}}$, enforces coherent normals between adjacent faces, suppressing surface foldovers and wrinkles and producing visually smooth surfaces.

Collectively, the optimization objective is a weighted sum of all components:

$$\mathcal{L} = \mathcal{L}_{\text{shape}} + \mathcal{L}_{\text{recon}} + \lambda_{\text{reg}} \left(\mathcal{L}_{\text{joint}} + \mathcal{L}_{\text{arap}} + \mathcal{L}_{\text{nc}}\right), \tag{12}$$

where λ_{reg} controls the strength of the regularization terms. Details of these loss functions are provided in the supplementary material.

5 Experiments

5.1 Experimental Setup

Datasets. Our benchmark comprises 40 animation videos: 20 videos sourced from existing works, including Consistent4D [12], STAG4D [63], and V2M4 [4], and the remaining 20 videos curated from Sketchfab. The Sketchfab videos include a more diverse and challenging set of motions for a more rigorous evaluation. As our setting assumes a fixed topology, we exclude cases that exhibit topological changes, such as the emergence of new surfaces in later frames that are absent from the initial mesh.

Metrics. In the absence of ground-truth meshes, we follow prior works [4,18] to evaluate performance by comparing rendered videos of the animated meshes with the reference videos. For frame-by-frame assessment, we compute the LPIPS score [69] to measure perceptual similarity and the CLIP score [34] to assess semantic consistency. To capture overall quality and temporal coherence, we employ the Fréchet Video Distance (FVD) [47] on the video sequences. To reduce the impact of color variations caused by mesh textures, these video metrics are computed on grayscale frames, emphasizing structural information that reflects the animation. Given that video-based metrics alone cannot adequately reflect 3D geometric fidelity, we further introduce the Uni3D score [72]. This metric evaluates the 3D geometry of the animated meshes by comparing 8,192 uniformly sampled surface points from each mesh with the corresponding reference images.

Baselines. We compare our method against three state-of-the-art approaches, DreamMesh4D [18], Puppeteer [41], and V2M4 [4], all of which produce animated meshes while enforcing consistent topology through optimization.

Implementation Details. Textured static mesh generation and shape flow elicitation are based on Trellis [53]. We preprocess each input video by segmenting the foreground object from the background. UniReg [66] is employed as the automatic rigging module. The hyperparameters λ_1 and λ_2 in Equation (7) are both set to 0.1, while λ_{reg} in Equation (12) is set to 10. The optimization runs for 200 epochs. All experiments are conducted on an NVIDIA A100 GPU.

5.2 Evaluation Results

Quantitative Comparisons. As shown in Table 1, our approach significantly outperforms the baselines across all evaluation metrics. Because DreamMesh4D relies heavily on Gaussian splatting in both its static generation and animation stages, its exported meshes are of relatively lower quality than those produced by the other methods. Since we adopt the same mesh generation model as Puppeteer and V2M4, the superior performance of our method on video metrics (LPIPS, CLIP, FVD) indicates more accurate motion generation. This improvement can be attributed to SFG, which provides a reliable driving signal for animation. Notably, our method achieves a substantially higher Uni3D score, demonstrating the superior geometric fidelity of our animated meshes. This advantage arises

Table 1: Comparison of methods on 2D rendering and 3D metrics.

Method	LPIPS ↓	CLIP ↑	FVD ↓	Uni3D ↑
DreamMesh4D [18]	0.1213	0.8643	931.50	0.2761
Puppeteer [41]	0.1143	0.8964	805.31	0.3009
V2M4 [4]	0.1025	0.8791	763.89	0.2963
Ours	**0.0791**	**0.9442**	**544.16**	**0.3486**

Table 2: Comparison of average runtime and user preference rate.

Method	DreamMesh4D [18]	Puppeteer [41]	V2M4 [4]	**Ours**
Preference ↑	0.0%	6.8%	8.5%	**84.8%**
Runtime ↓	60 min	10 min	20 min	**5 min**

because SFG introduces comprehensive 3D shape constraints during mesh deformation, rather than relying heavily on 2D rendering-based or projection-based losses, thereby reducing geometric artifacts in the deformed meshes.

User Study. We also conduct a user study involving 20 participants across 40 benchmark videos. For each case, participants are presented with a reference video and animated meshes from four methods in randomized order, and are asked to select the best result in terms of motion fidelity and geometric quality. The results in Table 2 demonstrate a strong user preference for our method.

Runtime. We also report the runtime for animating a 32-frame mesh sequence in Table 2. Our entire pipeline takes an average of only 5 minutes, which is 2× faster than Puppeteer, 4× faster than V2M4, and 12× faster than DreamMesh4D. This efficiency arises because SFG requires fewer optimization epochs to converge, supports full-frame parallelization across both elicitation and utilization, and incurs less computational overhead than Puppeteer's projection-based tracking losses. Although V2M4 adopts an interpolation-based strategy that processes only keyframes, its runtime remains bottlenecked by registration between adjacent meshes, which involves many optimization iterations and cannot be parallelized. In contrast, DreamMesh4D is the slowest, as both its static mesh generation and dynamic mesh animation rely on SDS.

Qualitative Comparisons. As shown in Figure 4, our method generates meshes whose motion is consistent with the input video while preserving the shape quality of the static mesh throughout deformation. In contrast, DreamMesh4D primarily emphasizes the rendering quality of Gaussian splatting while overlooking surface fidelity. Puppeteer suffers from the intrinsic depth ambiguity of 2D-3D projection in its projection-based optimization. This often induces physically implausible axial stretching or compression to satisfy 2D alignment, sacrificing 3D geometric integrity, as seen in the distorted bodies of the girl and boy characters. V2M4 is vulnerable to large non-rigid motions, stemming from its reliance on frame-to-frame registration to enforce topological consistency. This

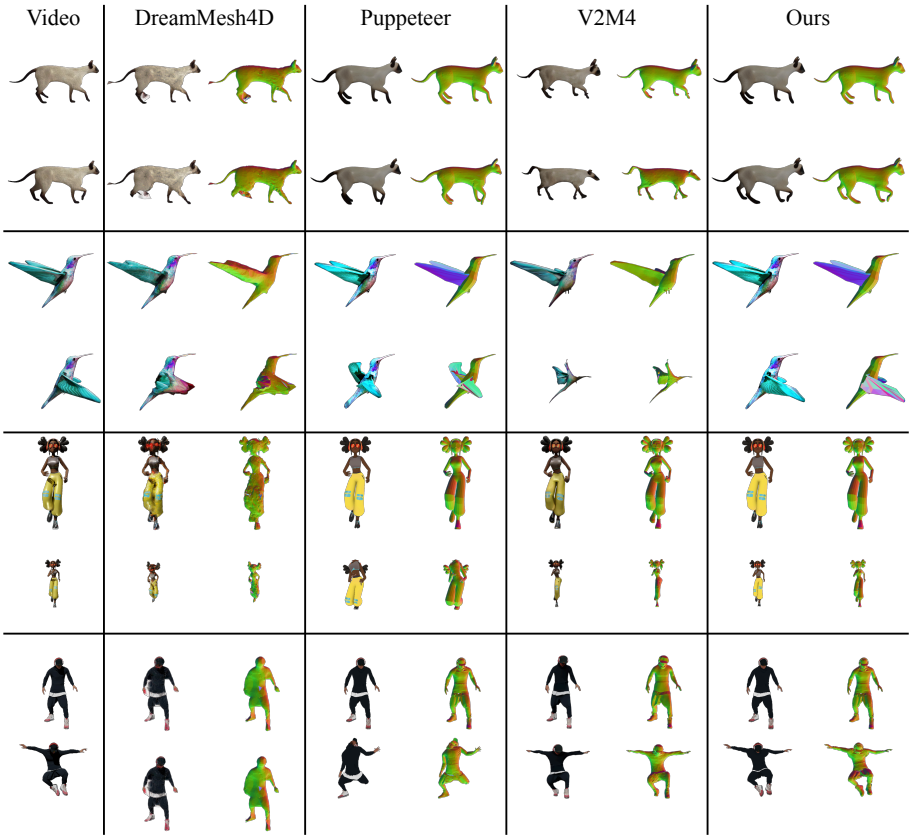

Fig. 4: Comparison of animated meshes from different methods at two timestamps, shown in both color and normal views. The animations (from top to bottom) show: a walking cat, a bird flapping its wings, a girl running backward, and a boy jumping up.

brittleness is particularly pronounced in challenging scenarios such as rapid wing flapping, where the bird mesh undergoes severe collapse. Furthermore, the video demonstrations in our supplementary material show that the baseline animations exhibit visible artifacts. These shortcomings stem from their heavy reliance on 2D supervision, which struggles to transfer motion from video to mesh while preserving geometry. By contrast, our approach provides 3D supervision for both shape and motion, thereby maintaining structural integrity and motion quality.

5.3 Ablation Study

We conduct ablation studies to assess the contribution of each component, with quantitative and qualitative results presented in Table 3 and Figure 5.

Impact of Shape Flow Guidance. We remove $\mathcal{L}_{\text{shape}}$ in Equation (12) while keeping the remaining loss terms and the animation framework unchanged. The results labeled as "w/o SFG" show a substantial performance degradation across

Table 3: Comparison of our full method and its ablative variants.

Variant	LPIPS ↓	CLIP ↑	FVD ↓	Uni3D ↑
w/o SFG	0.1447	0.8232	1057.73	0.2907
w/o ADT	0.0954	0.9215	654.82	0.3289
w/o PSC	0.1015	0.9106	689.27	0.3205
Full	**0.0791**	**0.9442**	**544.16**	**0.3486**

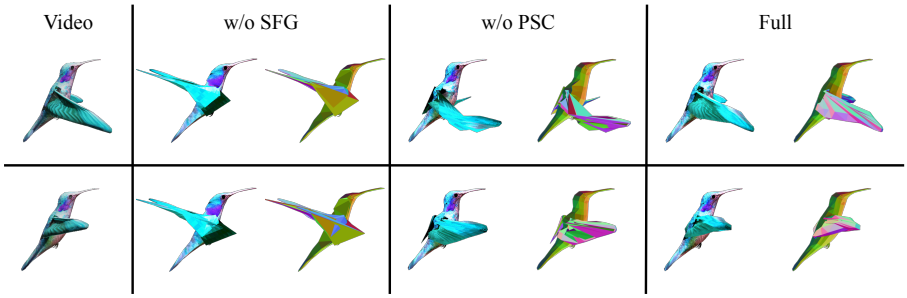

Fig. 5: Comparison of animated meshes from different ablative variants at two timestamps, shown in color and normal views.

all metrics. This suggests that the rendering-based reconstruction loss is insufficient for capturing meaningful mesh animation. As illustrated in Figure 5, without SFG, the rendering-based reconstruction loss alone fails to recover plausible animations and instead leads to severe geometric collapse.

Impact of Advection & Diffusion Terms. We omit the advection and diffusion terms (ADT) from Equation (7), which reduces our 2D flow field to 1D. This hinders interaction between trajectories across different frames and introduces temporal jitter in the shape flow. Consequently, the optimization process is adversely affected, leading to degraded animation quality ("w/o ADT").

Impact of Post-Skinning Correction. We disable the post-skinning correction (PSC) and revert to the standard LBS model. Due to its rigid and linear nature, LBS is limited in capturing fine-scale surface details and shape variations. As a result, the optimization process cannot fully leverage the geometric cues provided by SFG, leading to a noticeable degradation in animation quality ("w/o PSC"). As illustrated in Figure 5, without PSC, the animation model is constrained by the limitations of the rigging priors, making it difficult to reproduce fine-grained animation details in SFG.

5.4 In-depth Analysis

Robustness to Rigging Errors. Since our animation framework relies on an automatic rigging module, the results depend on the quality of the estimated skeleton and skinning weights. Current auto-rigging methods may produce rigs that deviate substantially from the natural kinematic structure. Nevertheless,

Table 4: Experiments on feedforward methods and alternative mesh generators.

Method	LPIPS ↓	CLIP ↑	FVD ↓	Uni3D ↑	Runtime ↓
Motion324 [3]	0.1078	0.8691	732.15	0.3172	40 sec
ActionMesh [37]	0.0971	0.9048	713.29	0.3335	3 min
Ours (Trellis)	0.0791	0.9442	544.16	0.3486	5 min
Ours (HY3D)	0.0774	0.9465	524.13	0.3531	5 min

our framework exhibits a notable degree of resilience to such errors. This robustness is enabled by our PSC network, which compensates for skinning inaccuracies by optimizing vertex-wise correction offsets. The effectiveness of this design is validated in Table 3 and Figure 5, where the performance gain of our full method over the "w/o PSC" variant confirms that PSC helps recover high-quality animations despite imperfect rigs that would otherwise lead to suboptimal results.

Trade-offs with Feedforward Methods. Beyond training-free optimization approaches, we also compare our framework against training-based feedforward methods. Specifically, Motion324 is evaluated on the standard 32-frame sequences, whereas ActionMesh employs two sliding windows spanning the first 31 frames. As reported in Table 4, despite requiring longer runtime, our method yields superior animation fidelity, showing a favorable quality-efficiency trade-off.

Generality across Mesh Generators. SFG can be readily integrated into other rectified-flow-based mesh generators. To validate this generality, we elicit SFG from Hunyuan3D 2.1 [10] using the same algorithm and apply it in the same animation stage. As shown in Table 4, the HY3D-based variant achieves comparable performance, indicating the model-agnostic nature of SFG.

Limitations and Future Work. Following the standard animation setting, our framework assumes fixed-topology animation from fixed-viewpoint videos, which inherently excludes topological evolution and camera motion. We leave topology-changing animation and moving-camera videos for future work.

6 Conclusion

In this work, we show that the heavy reliance on rendering-based or projection-based 2D supervision to drive mesh animation leads to severe shape and motion artifacts. To address this limitation, we introduce Shape Flow Guidance (SFG), a temporally consistent shape sequence distilled from a mesh generation model, to provide explicit 3D guidance for the animation process. We further propose a skeleton-based animation model that disentangles local deformation from global transformation. This separation allows SFG to supervise complex local motion while retaining rendering-based supervision for simple global motion. Extensive experiments show that our method significantly outperforms prior work, both quantitatively and qualitatively, while requiring substantially less runtime. We hope our work will inspire future research on leveraging mesh generation models to produce mesh animations from monocular video.

Acknowledgements

We would like to acknowledge that computational work involved in this research is partially supported by NUS IT's Research Computing group using grant number NUSREC-HPC-00001.

References

1. Bahmani, S., Liu, X., Yifan, W., Skorokhodov, I., Rong, V., Liu, Z., Liu, X., Park, J.J., Tulyakov, S., Wetzstein, G., et al.: Tc4d: Trajectory-conditioned text-to-4d generation. In: European Conference on Computer Vision. pp. 53–72 (2024)
2. Bahmani, S., Skorokhodov, I., Rong, V., Wetzstein, G., Guibas, L., Wonka, P., Tulyakov, S., Park, J.J., Tagliasacchi, A., Lindell, D.B.: 4d-fy: Text-to-4d generation using hybrid score distillation sampling. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7996–8006 (2024)
3. Chen, H., Chen, X., Xu, Z., Chen, A.: Motion 3-to-4: 3d motion reconstruction for 4d synthesis. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 28947–28958 (2026)
4. Chen, J., Zhang, B., Tang, X., Wonka, P.: V2m4: 4d mesh animation reconstruction from a single monocular video. In: IEEE/CVF International Conference on Computer Vision (2025)
5. Chen, R., Chen, Y., Jiao, N., Jia, K.: Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In: IEEE/CVF International Conference on Computer Vision. pp. 22246–22256 (2023)
6. Chen, Z., Wang, F., Wang, Y., Liu, H.: Text-to-3d using gaussian splatting. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21401–21412 (2024)
7. Deitke, M., Liu, R., Wallingford, M., Ngo, H., Michel, O., Kusupati, A., Fan, A., Laforte, C., Voleti, V., Gadre, S.Y., et al.: Objaverse-xl: A universe of 10m+ 3d objects. Advances in Neural Information Processing Systems **36**, 35799–35813 (2023)
8. Deitke, M., Schwenk, D., Salvador, J., Weihs, L., Michel, O., VanderBilt, E., Schmidt, L., Ehsani, K., Kembhavi, A., Farhadi, A.: Objaverse: A universe of annotated 3d objects. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13142–13153 (2023)
9. Hong, Y., Zhang, K., Gu, J., Bi, S., Zhou, Y., Liu, D., Liu, F., Sunkavalli, K., Bui, T., Tan, H.: Lrm: Large reconstruction model for single image to 3d. In: International Conference on Learning Representations (2024)
10. Hunyuan3D, T., Yang, S., Yang, M., Feng, Y., Huang, X., Zhang, S., He, Z., Luo, D., Liu, H., Zhao, Y., et al.: Hunyuan3d 2.1: From images to high-fidelity 3d assets with production-ready pbr material. arXiv preprint arXiv:2506.15442 (2025)
11. Jiang, Y., Yu, C., Cao, C., Wang, F., Hu, W., Gao, J.: Animate3d: Animating any 3d model with multi-view video diffusion. Advances in Neural Information Processing Systems **37**, 125879–125906 (2024)
12. Jiang, Y., Zhang, L., Gao, J., Hu, W., Yao, Y.: Consistent4d: Consistent 360 {\deg} dynamic object generation from monocular video. In: International Conference on Learning Representations (2024)
13. Kerbl, B., Kopanas, G., Leimkuehler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**, 1–14 (2023)

14. Li, J., Tan, H., Zhang, K., Xu, Z., Luan, F., Xu, Y., Hong, Y., Sunkavalli, K., Shakhnarovich, G., Bi, S.: Instant3d: Fast text-to-3d with sparse-view generation and large reconstruction model. In: International Conference on Learning Representations (2024)
15. Li, Y., Zou, Z.X., Liu, Z., Wang, D., Liang, Y., Yu, Z., Liu, X., Guo, Y.C., Liang, D., Ouyang, W., et al.: Triposg: High-fidelity 3d shape synthesis using large-scale rectified flow models. arXiv preprint arXiv:2502.06608 (2025)
16. Li, Y., Chen, Y., Ma, Z., Xiao, J., Wang, X., Yao, A.: Intermediate connectors and geometric priors for language-guided affordance segmentation on unseen object categories. In: IEEE/CVF International Conference on Computer Vision. pp. 22836–22845 (2025)
17. Li, Y., Zhao, N., Xiao, J., Feng, C., Wang, X., Chua, T.s.: Laso: Language-guided affordance segmentation on 3d object. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14251–14260 (2024)
18. Li, Z., Chen, Y., Liu, P.: Dreammesh4d: Video-to-4d generation with sparse-controlled gaussian-mesh hybrid representation. Advances in Neural Information Processing Systems **37**, 21377–21400 (2024)
19. Liang, H., Yin, Y., Xu, D., Wang, Z., Plataniotis, K.N., Zhao, Y., Wei, Y., et al.: Diffusion4d: Fast spatial-temporal consistent 4d generation via video diffusion models. Advances in Neural Information Processing Systems **37**, 110854–110875 (2024)
20. Liang, Y., Yang, X., Lin, J., Li, H., Xu, X., Chen, Y.: Luciddreamer: Towards high-fidelity text-to-3d generation via interval score matching. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6517–6526 (2024)
21. Lin, C.H., Gao, J., Tang, L., Takikawa, T., Zeng, X., Huang, X., Kreis, K., Fidler, S., Liu, M.Y., Lin, T.Y.: Magic3d: High-resolution text-to-3d content creation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 300–309 (2023)
22. Ling, H., Kim, S.W., Torralba, A., Fidler, S., Kreis, K.: Align your gaussians: Text-to-4d with dynamic 3d gaussians and composed diffusion models. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8576–8588 (2024)
23. Lipman, Y., Chen, R.T., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. In: International Conference on Learning Representations (2023)
24. Liu, I., Xu, Z., Yifan, W., Tan, H., Xu, Z., Wang, X., Su, H., Shi, Z.: Riganything: Template-free autoregressive rigging for diverse 3d assets. ACM Transactions on Graphics **44**, 1–12 (2025)
25. Liu, M., Shi, R., Chen, L., Zhang, Z., Xu, C., Wei, X., Chen, H., Zeng, C., Gu, J., Su, H.: One-2-3-45++: Fast single image to 3d objects with consistent multi-view generation and 3d diffusion. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10072–10083 (2024)
26. Liu, M., Xu, C., Jin, H., Chen, L., Varma T, M., Xu, Z., Su, H.: One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. Advances in Neural Information Processing Systems **36**, 22226–22246 (2023)
27. Liu, R., Wu, R., Van Hoorick, B., Tokmakov, P., Zakharov, S., Vondrick, C.: Zero-1-to-3: Zero-shot one image to 3d object. In: IEEE/CVF International Conference on Computer Vision. pp. 9298–9309 (2023)
28. Liu, X., Gong, C., Liu, Q.: Flow straight and fast: Learning to generate and transfer data with rectified flow. In: International Conference on Learning Representations (2023)

29. Liu, Y., Lin, C., Zeng, Z., Long, X., Liu, L., Komura, T., Wang, W.: Syncdreamer: Generating multiview-consistent images from a single-view image. In: International Conference on Learning Representations (2024)
30. Long, X., Guo, Y.C., Lin, C., Liu, Y., Dou, Z., Liu, L., Ma, Y., Zhang, S.H., Habermann, M., Theobalt, C., et al.: Wonder3d: Single image to 3d using cross-domain diffusion. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9970–9980 (2024)
31. Magnenat-Thalmann, N., Laperrière, R., Thalmann, D.: Joint-dependent local deformations for hand animation and object grasping. In: Graphics Interface Conference. pp. 26–33 (1989)
32. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: European Conference on Computer Vision. pp. 405–421 (2020)
33. Poole, B., Jain, A., Barron, J.T., Mildenhall, B.: Dreamfusion: Text-to-3d using 2d diffusion. In: International Conference on Learning Representations (2023)
34. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning. pp. 8748–8763 (2021)
35. Ren, J., Xie, C., Mirzaei, A., Kreis, K., Liu, Z., Torralba, A., Fidler, S., Kim, S.W., Ling, H., et al.: L4gm: Large 4d gaussian reconstruction model. Advances in Neural Information Processing Systems **37**, 56828–56858 (2024)
36. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10684–10695 (2022)
37. Sabathier, R., Novotny, D., Mitra, N.J., Monnier, T.: Actionmesh: Animated 3d mesh generation with temporal 3d diffusion. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 34312–34321 (2026)
38. Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E.L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al.: Photorealistic text-to-image diffusion models with deep language understanding. Advances in Neural Information Processing Systems **35**, 36479–36494 (2022)
39. Shi, Y., Wang, P., Ye, J., Long, M., Li, K., Yang, X.: Mvdream: Multi-view diffusion for 3d generation. In: International Conference on Learning Representations (2024)
40. Singer, U., Sheynin, S., Polyak, A., Ashual, O., Makarov, I., Kokkinos, F., Goyal, N., Vedaldi, A., Parikh, D., Johnson, J., et al.: Text-to-4d dynamic scene generation. In: International Conference on Machine Learning. pp. 31915–31929 (2023)
41. Song, C., Li, X., Yang, F., Xu, Z., Wei, J., Liu, F., Feng, J., Lin, G., Zhang, J.: Puppeteer: Rig and animate your 3d models. Advances in Neural Information Processing Systems **38**, 72152–72184 (2025)
42. Song, C., Zhang, J., Li, X., Yang, F., Chen, Y., Xu, Z., Liew, J.H., Guo, X., Liu, F., Feng, J., et al.: Magicarticulate: Make your 3d models articulation-ready. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15998–16007 (2025)
43. Sorkine, O., Alexa, M.: As-rigid-as-possible surface modeling. In: Symposium on Geometry Processing. vol. 4, pp. 109–116 (2007)
44. Tang, J., Chen, Z., Chen, X., Wang, T., Zeng, G., Liu, Z.: Lgm: Large multi-view gaussian model for high-resolution 3d content creation. In: European Conference on Computer Vision. pp. 1–18 (2024)

45. Tang, J., Ren, J., Zhou, H., Liu, Z., Zeng, G.: Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. In: International Conference on Learning Representations (2024)
46. Tang, J., Wang, T., Zhang, B., Zhang, T., Yi, R., Ma, L., Chen, D.: Make-it-3d: High-fidelity 3d creation from a single image with diffusion prior. In: IEEE/CVF International Conference on Computer Vision. pp. 22819–22829 (2023)
47. Unterthiner, T., Van Steenkiste, S., Kurach, K., Marinier, R., Michalski, M., Gelly, S.: Towards accurate generative models of video: A new metric & challenges. arXiv preprint arXiv:1812.01717 (2018)
48. Voleti, V., Yao, C.H., Boss, M., Letts, A., Pankratz, D., Tochilkin, D., Laforte, C., Rombach, R., Jampani, V.: Sv3d: Novel multi-view synthesis and 3d generation from a single image using latent video diffusion. In: European Conference on Computer Vision. pp. 439–457 (2024)
49. Wang, H., Du, X., Li, J., Yeh, R.A., Shakhnarovich, G.: Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12619–12629 (2023)
50. Wang, Z., Lu, C., Wang, Y., Bao, F., Li, C., Su, H., Zhu, J.: Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. Advances in Neural Information Processing Systems **36**, 8406–8441 (2023)
51. Wu, Z., Yu, C., Jiang, Y., Cao, C., Wang, F., Bai, X.: Sc4d: Sparse-controlled video-to-4d generation and motion transfer. In: European Conference on Computer Vision. pp. 361–379 (2024)
52. Wu, Z., Yu, C., Wang, F., Bai, X.: Animateanymesh: A feed-forward 4d foundation model for text-driven universal mesh animation. In: IEEE/CVF International Conference on Computer Vision (2025)
53. Xiang, J., Lv, Z., Xu, S., Deng, Y., Wang, R., Zhang, B., Chen, D., Tong, X., Yang, J.: Structured 3d latents for scalable and versatile 3d generation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21469–21480 (2025)
54. Xie, Y., Yao, C.H., Voleti, V., Jiang, H., Jampani, V.: Sv4d: Dynamic 3d content generation with multi-frame and multi-view consistency. In: International Conference on Learning Representations (2025)
55. Xiu, J., Hong, F., Li, Y., Li, M., Wang, W., Han, S., Pan, L., Liu, Z.: Egotwin: Dreaming body and view in first person. In: International Conference on Learning Representations (2026)
56. Xiu, J., Li, M., Ji, W., Chen, J., Zhao, H., Satoh, S., Zimmermann, R.: Hierarchical debiasing and noisy correction for cross-domain video tube retrieval. In: ACM International Conference on Multimedia. pp. 9271–9280 (2024)
57. Xiu, J., Li, M., Yang, Z., Ji, W., Yin, Y., Zimmermann, R.: Few-shot incremental learning via foreground aggregation and knowledge transfer for audio-visual semantic segmentation. In: AAAI Conference on Artificial Intelligence. pp. 8788–8796 (2025)
58. Xiu, J., Li, Y., Zhao, N., Fang, H., Wang, X., Yao, A.: Geometric alignment and prior modulation for view-guided point cloud completion on unseen categories. In: IEEE/CVF International Conference on Computer Vision. pp. 27435–27444 (2025)
59. Xiu, J., Wang, C., Xu, D.: Mllmsplat: A 2d mllm-powered framework for 3d gaussian splatting understanding, generation, and editing. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 33301–33311 (2026)
60. Xu, Y., Shi, Z., Yifan, W., Chen, H., Yang, C., Peng, S., Shen, Y., Wetzstein, G.: Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. In: European Conference on Computer Vision. pp. 1–20 (2024)

61. Yi, T., Fang, J., Wang, J., Wu, G., Xie, L., Zhang, X., Liu, W., Tian, Q., Wang, X.: Gaussiandreamer: Fast generation from text to 3d gaussians by bridging 2d and 3d diffusion models. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6796–6807 (2024)
62. Yu, H., Wang, C., Zhuang, P., Menapace, W., Siarohin, A., Cao, J., Jeni, L., Tulyakov, S., Lee, H.Y.: 4real: Towards photorealistic 4d scene generation via video diffusion models. Advances in Neural Information Processing Systems **37**, 45256–45280 (2024)
63. Zeng, Y., Jiang, Y., Zhu, S., Lu, Y., Lin, Y., Zhu, H., Hu, W., Cao, X., Yao, Y.: Stag4d: Spatial-temporal anchored generative 4d gaussians. In: European Conference on Computer Vision. pp. 163–179 (2024)
64. Zhang, B., Tang, J., Niessner, M., Wonka, P.: 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. ACM Transactions on Graphics **42**, 1–16 (2023)
65. Zhang, H., Chen, X., Wang, Y., Liu, X., Wang, Y., Qiao, Y.: 4diffusion: Multi-view video diffusion model for 4d generation. Advances in Neural Information Processing Systems **37**, 15272–15295 (2024)
66. Zhang, J.P., Pu, C.F., Guo, M.H., Cao, Y.P., Hu, S.M.: One model to rig them all: Diverse skeleton rigging with unirig. ACM Transactions on Graphics **44**, 1–18 (2025)
67. Zhang, K., Bi, S., Tan, H., Xiangli, Y., Zhao, N., Sunkavalli, K., Xu, Z.: Gs-lrm: Large reconstruction model for 3d gaussian splatting. In: European Conference on Computer Vision. pp. 1–19 (2024)
68. Zhang, L., Wang, Z., Zhang, Q., Qiu, Q., Pang, A., Jiang, H., Yang, W., Xu, L., Yu, J.: Clay: A controllable large-scale generative model for creating high-quality 3d assets. ACM Transactions on Graphics **43**, 1–20 (2024)
69. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 586–595 (2018)
70. Zhao, Z., Lai, Z., Lin, Q., Zhao, Y., Liu, H., Yang, S., Feng, Y., Yang, M., Zhang, S., Yang, X., et al.: Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation. arXiv preprint arXiv:2501.12202 (2025)
71. Zheng, Y., Li, X., Nagano, K., Liu, S., Hilliges, O., De Mello, S.: A unified approach for text-and image-guided 4d scene generation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7300–7309 (2024)
72. Zhou, J., Wang, J., Ma, B., Liu, Y.S., Huang, T., Wang, X.: Uni3d: Exploring unified 3d representation at scale. In: International Conference on Learning Representations (2024)