PhysRig: Differentiable Physics-Based Skinning and Rigging Framework for Realistic Articulated Object Modeling

Anonymous ICCV submission

Paper ID 637

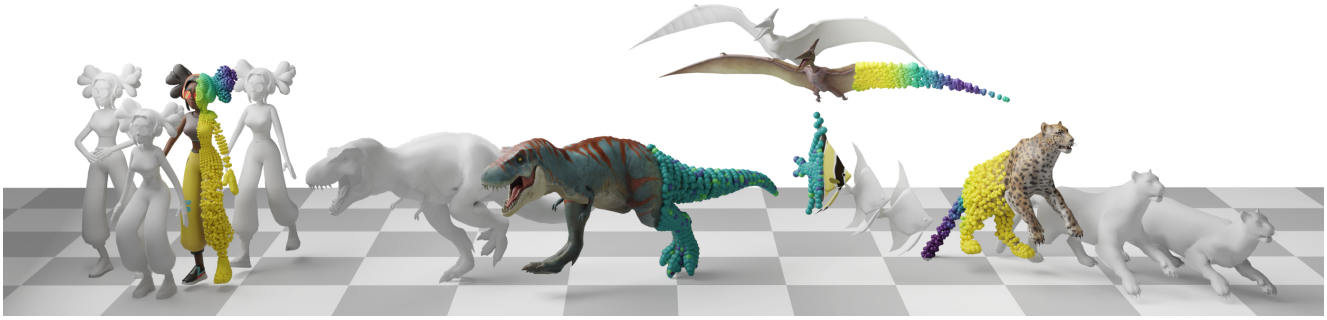

Figure 1. **PhysRig** is a differentiable physics-based skinning approach that models objects as soft-body volumes driven by embedded driving points, enabling realistic deformations and capturing complex dynamics across diverse topologies and motions—from humanoids to dinosaurs and flying creatures.

Abstract

*Skinning and rigging are fundamental components in animation, articulated object reconstruction, motion transfer, and 4D generation. Existing approaches predominantly rely on Linear Blend Skinning (LBS), due to its simplicity and differentiability. However, LBS introduces artifacts such as volume loss and unnatural deformations, and it fails to model elastic materials like soft tissues, fur, and flexible appendages (e.g., elephant trunks, ears, and fatty tissues). In this work, we propose **PhysRig**: a differentiable physics-based skinning and rigging framework that overcomes these limitations by embedding the rigid skeleton into a volumetric representation (e.g., a tetrahedral mesh), which is simulated as a deformable soft-body structure driven by the animated skeleton. Our method leverages continuum mechanics and discretizes the object as particles embedded in an Eulerian background grid to ensure differentiability with respect to both material properties and skeletal motion. Additionally, we introduce material prototypes, significantly reducing the learning space while maintaining high expressiveness. To evaluate our framework, we construct a comprehensive synthetic dataset using meshes from Objaverse [4], The Amazing Animals Zoo [30], and MixaMo [1], covering diverse object categories and motion patterns. Our method consistently outperforms traditional LBS-based approaches, generating more realistic and physically plausible results. Furthermore, we demonstrate the applicability of our framework in the pose transfer task highlighting its versatility for articulated object modeling.*

1. Introduction

Skinning and rigging are essential for animating articulated objects and play a critical role in numerous applications, including *character animation, motion retargeting, 4D reconstruction, and generative modeling*. Among existing approaches, *Linear Blend Skinning (LBS)* remains the dominant method due to its efficiency and differentiability. However, LBS suffers from severe limitations, including unnatural distortions (e.g., *collapsing joints, candy-wrapper artifacts, and volume shrinkage*) and an inability to capture the behavior of elastic materials. These artifacts become especially problematic when modeling characters with highly deformable regions, such as an elephant's trunk, a human's soft tissue, or flexible appendages.

To address these shortcomings, we introduce a differentiable physics-based skinning and rigging framework that models articulated object deformation as a volumetric simu-

lation problem. Instead of directly mapping vertices to rigid skeleton transformations, we embed the skeleton into a deformable soft-body volume (e.g., bounded by a set of Gaussians and tetrahedral meshes), which is driven by skeletal motion while respecting fundamental physical principles. In particular, we leverage continuum mechanics and the material point method to establish a fully differentiable deformation process, ensuring that both the material properties and skeletal motion are incorporated in a physically consistent manner. Unlike LBS, which applies simple linear blending, our approach captures intricate material behaviors by modeling stress-strain relationships and dynamic responses to skeletal forces, allowing us to achieve more realistic and physics-driven deformations.

A major challenge encountered with these physics-based methods is a large number of material parameters and complex particle interactions, which makes optimization challenging. To overcome this, we introduce material prototypes, a vocabulary of primitives that can be combined to represent all material properties, and span common deformation behaviors of articulated objects. This novel approach significantly reduces the learning space while maintaining expressiveness. It provides a structured way to interpolate material properties across different object types, enabling more efficient learning while preserving the diversity of real-world material responses.

Evaluating physics-based skinning models is challenging due to the lack of suitable benchmark datasets. Existing datasets are primarily built via LBS-based deformations and lack sufficient variation in material properties and deformation types. To address this gap, we construct a comprehensive synthetic dataset incorporating meshes from *Objaverse [4], The Amazing Animals Zoo [30], and MixaMo [1]*, covering a diverse range of objects, motion patterns, and material properties. Using this dataset, we demonstrate that our method outperforms LBS-based approaches, producing more realistic deformations across a variety of articulated objects. Additionally, we showcase the effectiveness of our framework in downstream tasks such as pose transfer and 4D object generation, illustrating its broad applicability. Our key contributions can be summarized as follows:

- **A differentiable physics-based skinning/rigging framework**, leveraging continuum mechanics to enable realistic and physically plausible deformations while remaining differentiable.
- **A novel material prototype formulation**, which reduces the learning complexity by introducing a structured interpolation approach while maintaining high material expressiveness.
- **A novel synthetic dataset** for evaluating physics-based skinning models, demonstrating our framework's superiority over LBS-based approaches.

Our approach bridges the gap between physics-based simulation and differentiable learning, providing a powerful tool for articulated object modeling in computer vision and graphics. By introducing a differentiable physics-driven deformation process, our framework enables new opportunities for more accurate, physically consistent skinning and rigging, with broad implications for animation, motion generation, and 4D modeling.

2. Related Work

Skinning in 4D Modeling and Animation. Skinning is fundamental to 3D character animation, modeling surface deformations induced by skeletal motion [22, 24]. Among various techniques, Linear Blend Skinning (LBS) remains the most widely used due to its simplicity and computational efficiency [17].

LBS is integral to many vision tasks, including video-to-3D reconstruction and avatar modeling. Parametric models like SCAPE [2] and SMPL [21] rely on predefined skeletons and skinning weights, limiting their adaptability. Neural implicit approaches [6, 16, 25, 37–39, 42–44] improve generalization but still require precise skeletal information. In avatar modeling, explicit methods [10, 33, 34, 40] optimize SMPL parameters, whereas implicit ones [7, 15, 26, 27, 29, 31, 35] leverage neural representations but face challenges in optimization and topological consistency.

LBS has also been applied to pose transfer [18, 28], with MagicPose4D [41] enabling cross-species motion. However, these methods often require recalculating skeletons and skinning weights for novel motions. Since LBS linearly blends external skeletal motion, it fails to capture true internal deformations, prompting research into physically-based skinning [5, 13, 14, 23]. While such methods better model volumetric changes, their non-differentiability limits integration with deep learning. Our approach introduces a differentiable physics-based skinning model, enabling efficient joint optimization via gradient descent.

Physical 4D Generation. In multiphysics simulation, the Material Point Method (MPM) [8, 11, 12] excels in handling topology changes and frictional interactions across various materials. Recent works [19, 32, 46] integrate MPM for physically plausible motion but rely on manual parameter tuning. Differentiable approaches [9, 20, 45] learn material properties but are restricted to simple motions. To bridge this gap, we propose PhysRig, a differentiable physics framework that learns material parameters for articulated objects, ensuring physical consistency across complex motions.

3. Method

In this paper, we introduce PhysRig, a differentiable physics-based skinning framework for 3D object deformation, applicable to meshes, point clouds, and Gaussian representations. If the input is a mesh or a Gaussian representation,

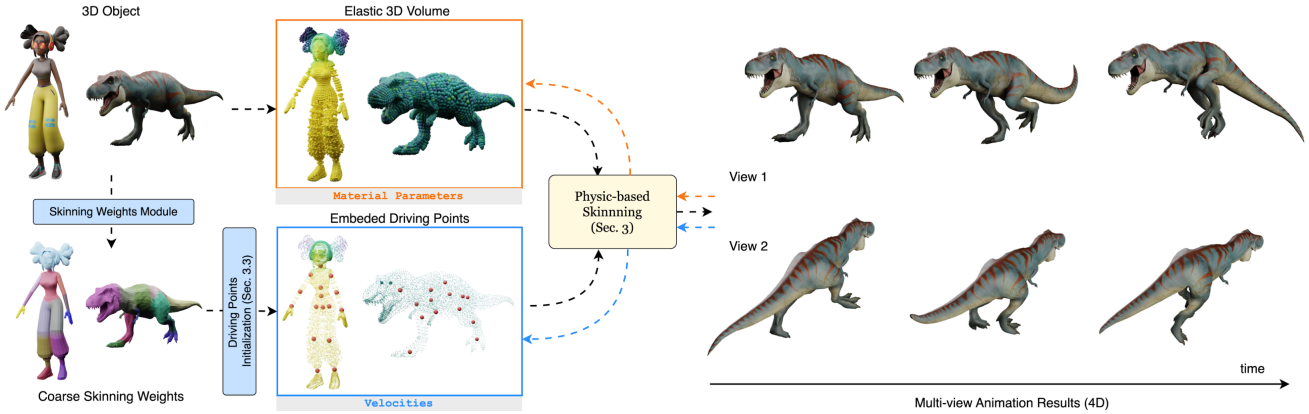

Figure 2. Overview of **PhysRig**. Given a 3D object, we first compute coarse skinning weights, which initialize embedded driving points for local deformation control. These points, assigned velocities, are linked to an elastic 3D volume with material parameters governing deformation. The differentiable physics-based skinning module generates natural deformations, optimizing velocities and material properties via backward propagation. Finally, multi-view animations illustrate physically plausible shape deformations over time.

we first perform a filling operation to obtain a solid volume in the form of a point cloud, and the total number of points is N. As shown in Fig. 2, unlike traditional Linear Blend Skinning (LBS), which applies a weighted sum of bone transformations, PhysRig employs a differentiable physics simulator (Sec.3.1) to model the 3D object as a volumetric structure. Instead of directly manipulating vertex positions, it embeds driving points within the volume to induce deformation. PhysRig optimizes two key components to achieve fine-grained control and produce the desired deformation:

- **Material properties**, including Young's modulus $E \in \mathbb{R}^P$ and Poisson's ratio $\nu \in \mathbb{R}^P$ for P material prototypes. The material properties of all N points are then computed using a function based on the Mahalanobis distance between each point and the material prototypes (Sec. 3.2). These properties govern elasticity and deformation behavior, determining how internal forces propagate through the structure.
- **Driving point velocities**, $v \in \mathbb{R}^{\{l*M,3\}}$, representing the motion of the internal skeletal structure parameterized by transformations $\{\mathbf{t}_0, ..., \mathbf{t}_M\}, \mathbf{t}_i \in SE(3)$, where M is the number of virtual joints. These velocities v drive the deformation, with $l = 8$ set by default and the driving points' positions are initialized by coarse skinning weights or uniform sampling (detailed in Sec. 3.3).

The driving points encode the skeletal motion, propagating movement to the surrounding 3D volume, while the material properties define how internal motion influences the object's outer surface. PhysRig can be formulated as:

$$\mathbf{X}' = \mathcal{F}(\mathbf{X}, E, \nu, v, \Delta t), \tag{1}$$

where $\mathbf{X} \in \mathbb{R}^{N\times 3}$ denotes the initial point positions, and $\Delta t \in \mathbb{R}$ is the time step governing temporal evolution. The function $\mathcal{F}$ computes the deformed positions $\mathbf{X}'$ via a differentiable physics simulation.

3.1. Physics-Based Simulation

To model object deformations under external interactions, we simulate motion using the principles of continuum mechanics. Our approach represents objects as continuous volumetric materials governed by conservation laws, enabling differentiable physics-based deformation modeling.

3.1.1. Continuum Mechanics Formulation

We describe the motion of a deformable object using a time-dependent mapping function ϕ, which transforms material coordinates $\boldsymbol{X}$ in the undeformed space Ω_0 to world coordinates $\boldsymbol{x}$ in the deformed space Ω_t: $\boldsymbol{x} = \phi(\boldsymbol{X}, t)$. The evolution of ϕ is constrained by fundamental physical laws:

Conservation of Mass. The total mass within a material region remains constant over time:

$$\int_{B_\epsilon^t} \rho(\boldsymbol{x}, t)\, d\boldsymbol{x} = \int_{B_\epsilon^0} \rho(\phi^{-1}(\boldsymbol{x}, t), 0)\, d\boldsymbol{x}, \tag{2}$$

where $\rho(\boldsymbol{x}, t)$ is the density field.

Conservation of Momentum. The motion of the object is dictated by the balance of internal and external forces:

$$\int_{B_\epsilon^t} \rho(\boldsymbol{x}, t)\boldsymbol{a}(\boldsymbol{x}, t)\, d\boldsymbol{x} = \int_{\partial B_\epsilon^t} \sigma \cdot \boldsymbol{n}\, d\boldsymbol{x} + \int_{B_\epsilon^t} \boldsymbol{f}^{\text{ext}}\, d\boldsymbol{x}, \tag{3}$$

where $\boldsymbol{a}(\boldsymbol{x}, t) = \frac{\partial^2 \phi}{\partial t^2}$ represents acceleration, $\boldsymbol{f}^{\text{ext}}$ denotes external forces, and σ is the Cauchy stress tensor, which encodes local deformation behavior.

3.1.2. Material Model and Deformation Representation

To model elastic responses, we use a constitutive model relating the stress tensor σ to the deformation gradient $F = \frac{\partial \phi}{\partial \boldsymbol{X}}$. We adopt a Fixed Corotated hyperelastic material model, which effectively captures nonlinear deformations while maintaining stability.

The Cauchy stress tensor is derived from the strain energy density function $\psi(F)$:

$$\sigma = \frac{1}{\det(F)} \frac{\partial \psi}{\partial F} F^T. \tag{4}$$

Following the Fixed Corotated model, the strain energy function is given by:

$$\psi(F) = \mu \sum_{i=1}^{d} (\sigma_i - 1)^2 + \frac{\lambda}{2} (\det(F) - 1)^2, \tag{5}$$

where σ_i are the singular values of F, and the material parameters μ and λ are related to Young's modulus E and Poisson's ratio ν:

$$\mu = \frac{E}{2(1+\nu)}, \quad \lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}. \tag{6}$$

3.1.3. Simulation via the Material Point Method

We employ the Material Point Method (MPM) [8] to solve the governing equations efficiently. MPM discretizes the object as particles embedded in an Eulerian background grid, enabling robust handling of large deformations while ensuring differentiability.

Particle-to-Grid (P2G) Transfer. At each simulation step, per-particle mass and momentum are transferred to the grid using B-spline interpolation:

$$\begin{aligned} m_i v_i = \sum_p N(x_i - x_p) \Big[m_p v_p + \Big(m_p C_p \\ - \frac{4}{\Delta x^2 \Delta t} V_p \frac{\partial \psi}{\partial F} F_p^T \Big) (x_i - x_p) \Big] + f_i. \end{aligned} \tag{7}$$

where: m_i, v_i are the mass and velocity at grid node i, $N(x_i - x_p)$ is the interpolation kernel, C_p is the velocity gradient at the particle, f_i is the external force, V_p is the volume of the particle, which scales the contribution of the stress force term. The stress-based force term, $V_p \frac{\partial \psi}{\partial F} F_p^T$, represents the internal elastic forces exerted by the particle. The factor V_p ensures that the contribution is properly scaled according to the physical size of the particle, preventing instabilities when transferring forces to the grid.

Grid-to-Particle (G2P) Update. After computing velocity updates on the grid, the velocities are interpolated back to particles, and positions are updated:

$$v_p^{t+1} = \sum_i N(x_i - x_p) v_i, \quad x_p^{t+1} = x_p + \Delta t v_p^{t+1}. \tag{8}$$

Deformation Gradient Update. The velocity gradient and deformation gradient are updated as:

$$\begin{aligned} \nabla v_p^{t+1} &= \frac{4}{\Delta x^2} \sum_i N(x_i - x_p) v_i (x_i - x_p)^T, \\ F_p^{t+1} &= (I + \Delta t \sum_i v_i \nabla N(x_i - x_p)^T) F_p. \end{aligned} \tag{9}$$

where: ∇v_p^{t+1} is the velocity gradient at particle p, describing how velocity varies locally. F_p^{t+1} is the updated deformation gradient, tracking material deformation over time. $\nabla N(x_i - x_p)$ is the spatial gradient of the interpolation function, describing how interpolation weights change with position. I is the identity matrix, ensuring that the deformation gradient starts from an undeformed state. Δt is the time step, controlling how much deformation accumulates per iteration. By iterating these updates, MPM efficiently captures complex material deformations while maintaining differentiability.

Driving Points Gradient Update. Driving points influence the motion of specific object regions by modifying the velocities of nearby grid nodes within their control region. The velocity update for a driving point $v_{\mathrm{d},j}$ is determined by the contributions from the affected grid nodes and is given by:

$$v_{\mathrm{d},j} \leftarrow v_{\mathrm{d},j} + \frac{1}{|R_c|} \sum_{i \in R_c} \nabla v_i, \tag{10}$$

where ∇v_i represents the velocity gradient at grid node i within the control region R_c.

3.1.4. Optimization Strategy for Inverse Skinning

Inverse Skinning is the process of recovering underlying motion parameters, such as material properties and driving point velocities, from observed deformations of a 3D object. Unlike traditional skinning methods (LBS), where deformations are computed from transformations and skinning weights, our inverse skinning aims to estimate the driving point velocities v and material properties (Young's modulus E, Poisson's ratio ν) that best explain a given motion sequence. This requires optimizing physical parameters to minimize discrepancies between simulated and observed motion.

Iteritively Optimization. To ensure stability, we adopt an iterative training strategy. First, we initialize the positions of the driving points and estimate their approximate velocities for each frame. We then alternate between the following two optimization steps: (1) Material Parameter Optimization: Fix the driving point velocities and update the material parameters using all frames as a single batch. (2) Driving

Method	Humanoid Character										Quadruped Animal							
	Michelle		Ortiz		Mutant		Jellyman		Kaya		Leopard		Mammoth		Stego		Krin	
	UR ↑	CD ↓	UR ↑	CD ↓	UR ↑	CD ↓	UR ↑	CD ↓	UR ↑	CD ↓	UR ↑	CD ↓	UR ↑	CD ↓	UR ↑	CD ↓	UR ↑	CD ↓
LBS-1	2.82	1.426	2.18	2.993	2.02	2.512	3.05	6.532	2.42	2.081	3.03	2.413	2.54	1.782	3.07	0.682	3.03	0.561
LBS-2	3.06	0.891	2.93	2.011	3.37	1.438	3.25	4.130	2.97	1.214	3.13	1.328	2.8	0.891	3.64	0.202	3.38	0.271
LBS-3	3.32	0.372	3.01	1.472	3.47	0.801	3.66	3.811	3.43	0.408	3.21	0.493	3.25	0.346	3.83	0.103	3.47	0.048
Ours-init	3.19	2.105	2.91	12.755	2.98	5.977	3.43	14.161	3.39	2.027	3.43	1.822	3.13	4.991	3.51	1.664	3.63	0.844
Ours	**4.7**	**0.139**	**4.43**	**1.375**	**4.61**	**0.527**	**4.34**	**1.214**	**4.8**	**0.228**	**4.45**	**0.212**	**4.59**	**0.127**	**4.5**	**0.085**	**4.4**	**0.032**

Method	Quadruped Animal				Other Entities													
	Cow		Raccoon		T-rex		Pterosaur		Whale		Angelfish		Cobra		Shark		Ave.	
	UR ↑	CD ↓	UR ↑	CD ↓	UR ↑	CD ↓	UR ↑	CD ↓	UR ↑	CD ↓	UR ↑	CD ↓	UR ↑	CD ↓	UR ↑	CD ↓	UR ↑	CD ↓
LBS-1	2.51	1.351	2.29	1.243	2.88	5.861	2.54	6.722	2.77	0.292	3.11	0.841	2.64	2.712	3.27	0.078	2.72	2.43
LBS-2	3.0	1.019	2.58	0.791	3.21	3.763	2.78	4.512	2.95	0.253	3.57	0.614	2.79	2.133	3.59	0.046	3.12	1.56
LBS-3	2.99	0.781	3.48	0.342	3.25	0.687	3.05	1.082	3.17	0.134	3.61	0.209	3.01	0.607	3.81	0.031	3.35	0.73
Ours-init	3.1	2.244	3.37	4.737	3.17	6.574	2.97	9.522	3.11	4.561	3.81	0.296	3.77	4.049	3.93	0.178	3.34	4.67
Ours	**4.24**	**0.187**	**4.63**	**0.198**	**4.66**	**0.588**	**4.49**	**0.653**	**4.56**	**0.132**	**4.32**	**0.021**	**4.76**	**0.372**	**4.34**	**0.016**	**4.52**	**0.37**

Table 1. **Comparison of different rigging methods for inverse skinning.** UR ↑: User Study Rate, CD ↓: Chamfer Distance. LBS-1, LBS-2, and LBS-3 correspond to using RigNet [36], Pinocchio [3], and ground truth skinning weights, respectively, as initialization for jointly optimizing skinning weights and bone transformation. **PhysRig** utilizes Pinocchio to obtain coarse skinning weights for initializing driving points, and then iteratively learns material parameters and driving point velocities. Our dataset consists of 17 diverse objects among humans, quadrupeds, and other entities, totaling 120 motion sequences. We report the average performance across all motions for objects with multiple motions. The User Study setup is provided in the appendix Sec. A.4

Point Velocity Optimization: Fix the material parameters and sequentially update the velocities of the driving points for each frame. The optimization progresses frame by frame, moving to the next frame once the loss falls below a predefined threshold. These two steps are repeated iteratively until either the overall loss falls below a set threshold or the total number of iterations reaches the stopping criterion.

This strategy is designed to account for the differing requirements of material parameters and velocity optimization. Optimizing material parameters requires information accumulated across multiple frames, as the material properties influence the object's global behavior over time. In contrast, optimizing driving point velocities must be performed sequentially on a per-frame basis. Simultaneously optimizing velocities across multiple frames is ineffective, as accurate simulation of later frames is only meaningful if the preceding frames have already been well-optimized.

3.2. Material Prototype

To efficiently represent material properties across an object's volume, we introduce material prototypes, each characterized by two learnable parameters: Young's modulus and Poisson's ratio. The material properties at any point within the volume are computed as a weighted sum of these prototypes. The weights are determined using a function based on the **Mahalanobis distance** between the query position and the prototypes. Specifically, we define each *material prototype* as a Gaussian ellipsoid, parameterized by its center $\mathbf{C} \in \mathbb{R}^{P\times 3}$, orientation $\mathbf{V} \in \mathbb{R}^{P\times 3\times 3}$, and diagonal scale $\mathbf{\Lambda} \in \mathbb{R}^{P\times 3\times 3}$, where P denotes the number of prototypes. The weight assignment follows:

$$W_{n,p} = \text{softmax}_{p\in P}(d(\mathbf{x}_n, \mathbf{C}_p, \mathbf{Q}_p)) \tag{11}$$

where $d(\mathbf{x}_n, \mathbf{C}_p, \mathbf{Q}_p)$ is the Mahalanobis distance, defined as: $d(\mathbf{x}_n, \mathbf{C}_p, \mathbf{Q}_p) = (\mathbf{x}_n - \mathbf{C}_p)^T \mathbf{Q}_p (\mathbf{x}_n - \mathbf{C}_p), \quad \mathbf{Q}_p = \mathbf{V}_p^T \mathbf{\Lambda}_p \mathbf{V}_p$. Here, $\mathbf{x}_n$ represents the coordinates of a query point n, and the Mahalanobis distance function ensures that weights are assigned based on the spatial relationship between the query position and the material prototypes. This formulation enables an efficient and differentiable material representation that generalizes across diverse volumetric structures.

Compared to directly learning per-point material properties or employing a triplane-based function that maps spatial coordinates to material parameters, our material prototype representation offers a significantly more compact and efficient parameterization. By leveraging a small set of prototypes rather than densely modeling every point, we substantially reduce the optimization space while maintaining high expressiveness. Moreover, the prototype-based formulation naturally enforces smooth material transitions, preventing noisy or abrupt variations that are common in per-point learning approaches. This property aligns more closely with the behavior of real-world materials, where material properties exhibit gradual spatial variations rather than sharp discontinuities.

3.3. Driving Point Initialization

Driving points are a crucial component of PhysRig, as efficiently initializing their positions and velocities significantly improves optimization efficiency. To achieve this, we propose a coarse-to-fine initialization strategy based on skinning weights. We first obtain coarse skinning weights using existing rigging models such as Pinocchio [3] or RigNet [36], which provide an approximate mapping between the object's

surface and skeletal structure. We then place driving points at joint locations, which naturally reside at the boundaries between adjacent parts.

3.3.1. Affinity-Based Seg via Spectral Clustering

Given per-vertex skinning weights $\mathbf{W} \in \mathbb{R}^{N \times B}$, where N is the number of vertices and B is the number of bones, we construct an affinity matrix $\mathbf{A}$ to measure similarity between vertices:

$$A_{i,j} = \exp\left(-\frac{\|\mathbf{W}_i - \mathbf{W}_j\|^2}{\sigma^2}\right), \tag{12}$$

where σ controls the sensitivity of similarity measurement. A larger σ results in smoother clustering, while a smaller σ captures finer-scale differences. Using $\mathbf{A}$, we compute the graph Laplacian: $\mathbf{L} = \mathbf{D} - \mathbf{A}$,where $D_{i,i} = \sum_j A_{i,j}$. We obtain a low-dimensional embedding by computing the k smallest eigenvectors of $\mathbf{L}$ and apply k-means clustering to segment the object into rigid regions, each assigned a cluster label c_i. Note that k could be different with B. Since the coarse skinning weights may not always meet our expectations, our approach allows for flexible control over the number of parts by adjusting k.

3.3.2. Locating Joint via Skinning Weight Variance

To extract joint locations, we analyze the segmentation output to identify transition regions where adjacent rigid components meet. A vertex i is classified as a **boundary vertex** if: $c_i \neq c_j, \quad$ for some $j \in \mathcal{N}(i)$, where $\mathcal{N}(i)$ denotes the set of neighboring vertices in the mesh. These boundary vertices form the primary candidates for joint locations. To further refine the detected joints, we analyze **variance in skinning weights** at boundary vertices. Specifically, we define the joint set $\mathcal{J}$ as: $\mathcal{J} = \left\{ i \mid \sum_b \left(W_{i,b} - \bar{W}_{\mathcal{N}(i),b}\right)^2 > \tau \right\}$, where $\bar{W}_{\mathcal{N}(i),b}$ is the mean skinning weight of neighboring vertices of i, and τ is a threshold for detecting significant weight variations. This step ensures that only regions with meaningful changes in skinning influences are selected as joints.

3.3.3. Driving Points Initialization

At each identified **joint**, we uniformly place l driving points to ensure fine-grained control over the deformation of nearby volumetric regions. Each driving point's initial velocity is computed as the **average velocity of its surrounding volume**, ensuring a smooth and physically consistent initialization: $v_p = \frac{\sum_{i \in \mathcal{N}_p} v_i}{|\mathcal{N}_p|}$, where $\mathcal{N}_p$ represents the set of nearby points influencing the driving point.

Although the coarse skinning weights obtained from pre-existing models may not be highly accurate, they provide a decent starting point. Our method refines these initial estimates (velocity) during optimization, ultimately yielding more accurate motion parameters that adapt to the specific material properties of the object.

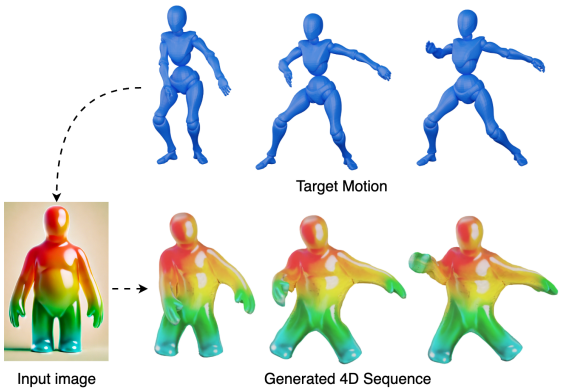

Figure 3. **PhysRig** enables pose transfer for generated objects.

	Michelle		Leopard		Angelfish		Converge
	UR↑	CD↓	UR↑	CD↓	UR↑	CD↓	Iteration↓
Mat Field	3.31	1.93	3.47	1.58	3.93	0.23	-
Per-point	3.15	2.31	3.61	1.77	3.73	0.25	-
w/o Locating	4.31	0.186	4.23	0.358	3.97	0.031	8000
w/o Vel Init	4.08	0.183	4.03	0.301	4.17	0.029	5000
Prototypes: 25	-	0.147	-	0.229	-	0.023	**2000**
Prototypes: 100	**4.7**	0.139	**4.45**	0.212	**4.32**	0.021	2500
Prototypes: 200	-	**0.133**	-	**0.207**	-	**0.019**	2500

Table 2. **Ablation study** on material prototypes vs. material field vs. per-point for material representation, the impact of the number of material prototypes, and the effect of driving point initialization, including (i) joint localization and (ii) velocity initialization.

4. Experiments

In this section, we compare PhysRig with the traditional neural Linear Blend Skinning (LBS) method on the inverse skinning task, which serves as a fundamental component for various applications such as 3D video reconstruction and part decomposition. This comparison highlights PhysRig's strong capability in dynamic modeling and optimization for articulated objects. To facilitate the evaluation, we introduce a new dataset, which is constructed from existing datasets (Objaverse, The Amazing Animals Zoo and Mixamo) and includes entities with diverse structural variations. Additionally, we generate a large amount of synthetic data using PhysRig, enabling a more comprehensive analysis of its optimization performance, particularly in learning material properties and driving point velocities. For more details on the dataset (Sec. A.1) and implementation (Sec. A.2), more experimental (Sec. A.3) results, and video results please refer to the supplementary materials.

4.1. Inverse Skinning Evaluation

We evaluate the effectiveness of our inverse skinning method across a diverse set of humanoid characters, quadruped animals, and other articulated entities. We compare against traditional Linear Blend Skinning (LBS) baselines, includ-

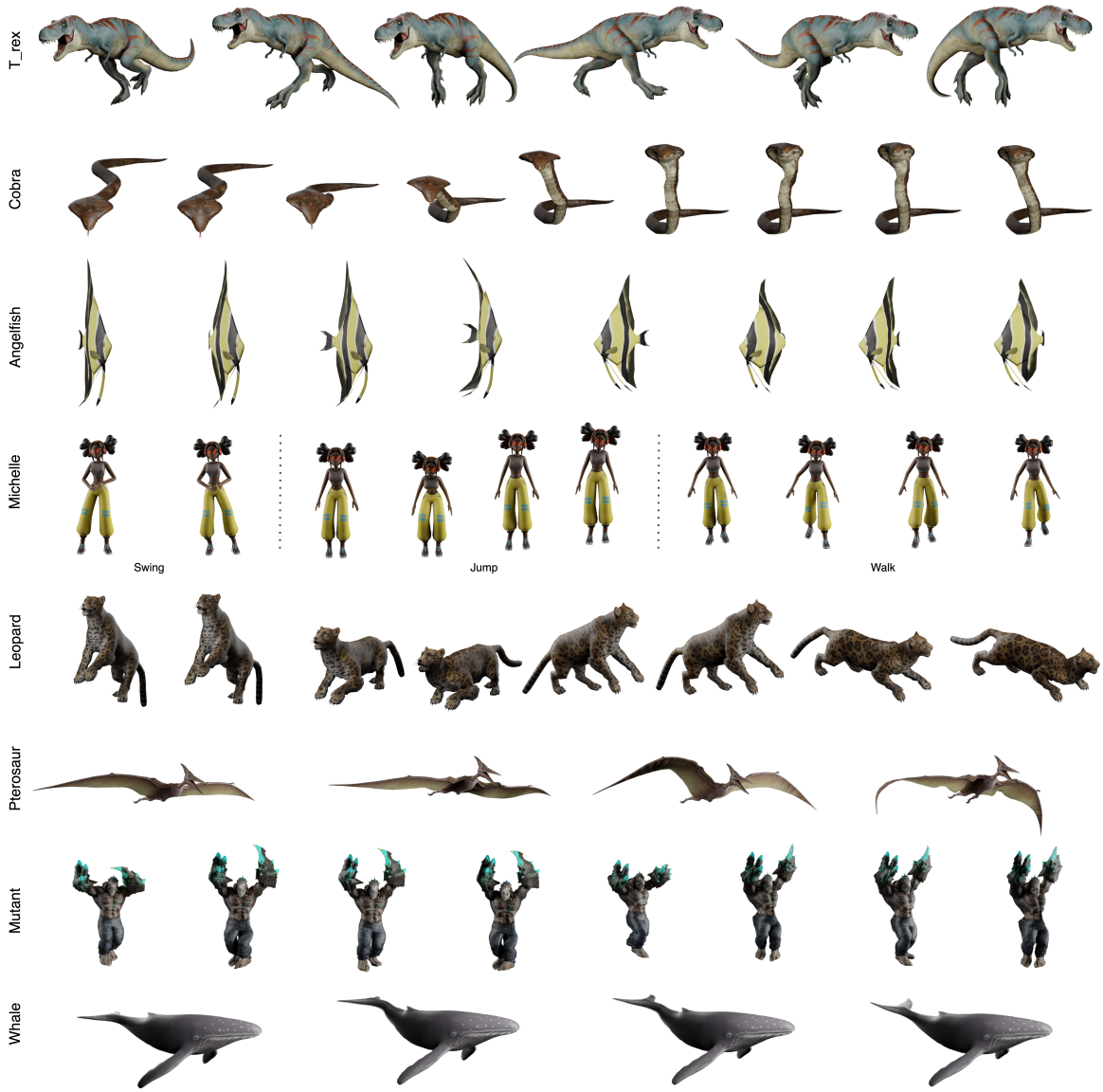

Figure 4. **Animation results** from the **PhysRig** approach. These results are obtained from the inverse skinning problem by optimizing material properties and driving point velocities to minimize the deviation from the ground truth mesh sequence.

ing RigNet [36], Pinocchio [3], and ground truth skinning weight initialization, as well as the results after driving points initialization before optimization (Ours-init). The evaluation metrics include **User Study Rate (UR)**, which quantifies perceptual quality based on user preferences, with scores ranging from 0 to 5 (higher is better), and **Chamfer Distance (CD)**, which evaluates geometric fidelity.

Table 1 presents the results. Our method consistently outperforms all baselines, achieving the highest UR scores and the lowest CD across all evaluated categories. Notably, on humanoid characters, our method achieves a UR of 4.7 on Michelle and a UR of 4.8 on Kaya, surpassing all LBS-based approaches. Similarly, for quadrupeds, our approach demonstrates superior performance, particularly on the Leopard (UR: 4.45, CD: 0.212) and Stego (UR: 4.5, CD: 0.085), highlighting its robustness across diverse morphologies. Our approach also generalizes well to other articulated entities, such as the Angelfish (UR: 4.32, CD: 0.021) and Pterosaur (UR: 4.49, CD: 0.653), showcasing its effectiveness beyond conventional character rigging. These results indicate that our inverse skinning formulation not only improves perceptual quality but also significantly reduces geometric error

Figure 5. Comparison of the learned material properties with ground truth using our method.

compared to existing baselines. For qualitative comparisons and analysis, please refer to the appendix (Sec. A.3).

4.2. Ablation Study

We conduct an ablation study to analyze the impact of different components in our method, particularly focusing on material representation and driving point initialization. The results are summarized in Table 2.

Material Representation: We compare our prototype-based material representation against material fields and per-point assignments. The per-point approach leads to higher geometric error (CD: 2.31 on Michelle, 1.77 on Leopard), indicating that it struggles to find the optimal solution. The material field (triplane) method also underperforms, demonstrating increased Chamfer Distance across all test cases. In contrast, our prototype-based representation significantly reduces CD and achieves high UR scores.

Effect of Driving Point Initialization: Removing joint localization (w/o Locating) results in increased CD values (e.g., 0.186 on Michelle), requiring 8000 iterations for convergence. Similarly, excluding velocity initialization (w/o Vel Init) leads to a higher CD (0.183 on Michelle) and slower convergence (5000 iterations). These findings suggest that both joint localization and velocity initialization are crucial for improving optimization efficiency and accuracy.

Effect of Material Prototype Count: We also investigate the impact of the number of material prototypes. Reducing the prototype count to 25 does not degrade performance and instead accelerates convergence, achieving the fastest convergence at 2000 iterations while maintaining competitive accuracy (CD: 0.147 on Michelle). Increasing the prototype count to 100 strikes a good balance between performance and convergence time (UR: 4.7, CD: 0.139 on Michelle, convergence: 2500 iterations). Further increasing the prototypes to 200 yields a marginal improvement in CD (0.133 on Michelle) but does not significantly affect UR, suggesting diminishing returns.

Overall, these results demonstrate that our material prototype representation, combined with joint localization and velocity initialization, leads to improved inverse skinning accuracy and faster optimization convergence.

4.3. Apply PhysRig for Pose Transfering

As shown in Figure 3, PhysRig enables pose transfer by taking a mesh sequence as input. Inspired by MagicPose4D [41], we first extract the skeleton from the input mesh and align it with the generated mesh. By transferring the bone angles at each frame, we obtain the skeleton sequence for the generated object. This allows us to compute joint velocities between consecutive frames, which serve as the driving point velocities for deforming the generated mesh (volume). Unlike traditional methods that rely on skinning weight, PhysRig achieves more realistic deformations while significantly improving generalization, as it eliminates the need for explicit skinning weight prediction.

5. Conclusion

We introduced PhysRig, a differentiable physics-based skinning framework that addresses the limitations of Linear Blend Skinning (LBS) by modeling deformations through volumetric simulation. By embedding skeletons into a soft-body representation and leveraging continuum mechanics, our approach achieves realistic, physically plausible deformations while remaining fully differentiable. To enhance efficiency, we introduced material prototypes, reducing learning complexity while maintaining expressiveness. Our evaluation of a diverse synthetic dataset demonstrated superior performance over traditional LBS-based methods. Additionally, PhysRig enables applications such as pose transfer, motion retargeting, and 4D generation, bridging the gap between physics-based simulation and differentiable learning. Future work includes integrating real-world priors and optimizing for real-time applications, expanding PhysRig's potential in animation and simulation.

References

[1] Adobe. Mixamo. 1, 2

[2] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. Scape: shape completion and animation of people. *ACM Trans. Graph.*, 24 (3):408–416, 2005. 2

[3] Ilya Baran and Jovan Popović. Automatic rigging and animation of 3d characters. 26(3):72–es, 2007. 5, 7

[4] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. *Advances in Neural Information Processing Systems*, 36:35799–35813, 2023. 1, 2

[5] Nico Galoppo, Miguel A Otaduy, Serhat Tekin, Markus Gross, and Ming C Lin. Soft articulated characters with fast contact handling. In *Computer Graphics Forum*, pages 243–253. Wiley Online Library, 2007. 2

[6] Simon Giebenhain, Tobias Kirschstein, Markos Georgopoulos, Martin Rünz, Lourdes Agapito, and Matthias Nießner. Learning neural parametric head models. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[7] Shoukang Hu, Tao Hu, and Ziwei Liu. Gauhuman: Articulated gaussian splatting from monocular human videos. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20418–20431, 2024. 2

[8] Yuanming Hu, Yu Fang, Ziheng Ge, Ziyin Qu, Yixin Zhu, Andre Pradhana, and Chenfanfu Jiang. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Transactions on Graphics (TOG)*, 37(4):1–14, 2018. 2, 4

[9] Tianyu Huang, Haoze Zhang, Yihan Zeng, Zhilu Zhang, Hui Li, Wangmeng Zuo, and Rynson WH Lau. Dreamphysics: Learning physical properties of dynamic 3d gaussians with video diffusion priors. *arXiv preprint arXiv:2406.01476*, 2024. 2

[10] Boyi Jiang, Yang Hong, Hujun Bao, and Juyong Zhang. Selfrecon: Self reconstruction your digital avatar from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5605–5615, 2022. 2

[11] Chenfanfu Jiang, Craig Schroeder, Andrew Selle, Joseph Teran, and Alexey Stomakhin. The affine particle-in-cell method. *ACM Transactions on Graphics (TOG)*, 34(4):1–10, 2015. 2

[12] Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. The material point method for simulating continuum materials. In *Acm siggraph 2016 courses*, pages 1–52. 2016. 2

[13] Junggon Kim and Nancy S Pollard. Fast simulation of skeleton-driven deformable body characters. *ACM Transactions on Graphics (TOG)*, 30(5):1–19, 2011. 2

[14] Theodore Kim and Doug L James. Physics-based character skinning using multi-domain subspace deformations. In *Proceedings of the 2011 ACM SIGGRAPH/eurographics symposium on computer animation*, pages 63–72, 2011. 2

[15] Muhammed Kocabas, Jen-Hao Rick Chang, James Gabriel, Oncel Tuzel, and Anurag Ranjan. Hugs: Human gaussian splats. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 505–515, 2024. 2

[16] Jiahui Lei, Yufu Wang, Georgios Pavlakos, Lingjie Liu, and Kostas Daniilidis. Gart: Gaussian articulated template models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19876–19887, 2024. 2

[17] J. P. Lewis, Matt Cordner, and Nickson Fong. Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, page 165–172, USA, 2000. ACM Press/Addison-Wesley Publishing Co. 2

[18] Zhouyingcheng Liao, Jimei Yang, Jun Saito, Gerard Pons-Moll, and Yang Zhou. Skeleton-free pose transfer for stylized 3d characters. In *European Conference on Computer Vision*, pages 640–656. Springer, 2022. 2

[19] Jiajing Lin, Zhenzhong Wang, Shu Jiang, Yongjie Hou, and Min Jiang. Phys4dgen: A physics-driven framework for controllable and efficient 4d content generation from a single image. *arXiv preprint arXiv:2411.16800*, 2024. 2

[20] Fangfu Liu, Hanyang Wang, Shunyu Yao, Shengjun Zhang, Jie Zhou, and Yueqi Duan. Physics3d: Learning physical properties of 3d gaussians via video diffusion. *arXiv preprint arXiv:2406.04338*, 2024. 2

[21] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Smpl: a skinned multi-person linear model. *ACM Trans. Graph.*, 34(6), 2015. 2

[22] N. Magnenat-Thalmann and D. Thalmann. Complex models for animating synthetic actors. *IEEE Computer Graphics and Applications*, 11(5):32–44, 1991. 2

[23] Aleka McAdams, Yongning Zhu, Andrew Selle, Mark Empey, Rasmus Tamstorf, Joseph Teran, and Eftychios Sifakis. Efficient elasticity for character skinning with contact and collisions. In *ACM SIGGRAPH 2011 papers*, pages 1–12. 2011. 2

[24] Alex Mohr and Michael Gleicher. Building efficient, accurate character skins from examples. *ACM Trans. Graph.*, 22(3): 562–568, 2003. 2

[25] Pablo Palafox, Aljaž Božič, Justus Thies, Matthias Nießner, and Angela Dai. Npms: Neural parametric models for 3d deformable shapes. In *Proceedings - 2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021*, pages 12675–12685. Institute of Electrical and Electronics Engineers Inc., 2021. Publisher Copyright: © 2021 IEEE; 18th IEEE/CVF International Conference on Computer Vision, ICCV 2021 ; Conference date: 11-10-2021 Through 17-10-2021. 2

[26] Panwang Pan, Zhuo Su, Chenguo Lin, Zhen Fan, Yongjie Zhang, Zeming Li, Tingting Shen, Yadong Mu, and Yebin Liu. Humansplat: Generalizable single-image human gaussian splatting with structure priors. *Advances in Neural Information Processing Systems*, 37:74383–74410, 2024. 2

[27] Zhiyin Qian, Shaofei Wang, Marko Mihajlovic, Andreas Geiger, and Siyu Tang. 3dgs-avatar: Animatable avatars

via deformable 3d gaussian splatting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5020–5030, 2024. 2

[28] Chaoyue Song, Jiacheng Wei, Ruibo Li, Fayao Liu, and Guosheng Lin. 3d pose transfer with correspondence learning and mesh refinement. *Advances in Neural Information Processing Systems*, 34:3108–3120, 2021. 2

[29] Shih-Yang Su, Frank Yu, Michael Zollhöfer, and Helge Rhodin. A-nerf: Articulated neural radiance fields for learning human shape, appearance, and pose. *Advances in neural information processing systems*, 34:12278–12291, 2021. 2

[30] Truebones Motions Animation Studios. The Amazing Animals Zoo Dataset. 1, 2

[31] Chung-Yi Weng, Brian Curless, Pratul P Srinivasan, Jonathan T Barron, and Ira Kemelmacher-Shlizerman. Humannerf: Free-viewpoint rendering of moving people from monocular video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern Recognition*, pages 16210–16220, 2022. 2

[32] Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4389–4398, 2024. 2

[33] Yuliang Xiu, Jinlong Yang, Xu Cao, Dimitrios Tzionas, and Michael J Black. Econ: Explicit clothed humans optimized via normal integration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 512–523, 2023. 2

[34] Hongyi Xu, Eduard Gabriel Bazavan, Andrei Zanfir, William T Freeman, Rahul Sukthankar, and Cristian Sminchisescu. Ghum & ghuml: Generative 3d human shape and articulated pose models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6184–6193, 2020. 2

[35] Hongyi Xu, Thiemo Alldieck, and Cristian Sminchisescu. H-nerf: Neural radiance fields for rendering and temporal reconstruction of humans in motion. *Advances in Neural Information Processing Systems*, 34:14955–14966, 2021. 2

[36] Zhan Xu, Yang Zhou, Evangelos Kalogerakis, Chris Landreth, and Karan Singh. Rignet: Neural rigging for articulated characters. *arXiv preprint arXiv:2005.00559*, 2020. 5, 7

[37] Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Huiwen Chang, Deva Ramanan, William T Freeman, and Ce Liu. Lasr: Learning articulated shape reconstruction from a monocular video. In *CVPR*, 2021. 2

[38] Gengshan Yang, Deqing Sun, Varun Jampani, Daniel Vlasic, Forrester Cole, Ce Liu, and Deva Ramanan. Viser: Video-specific surface embeddings for articulated 3d shape reconstruction. In *NeurIPS*, 2021.

[39] Gengshan Yang, Minh Vo, Natalia Neverova, Deva Ramanan, Andrea Vedaldi, and Hanbyul Joo. Banmo: Building animatable 3d neural models from many casual videos. In *CVPR*, 2022. 2

[40] Ilya Zakharkin, Kirill Mazur, Artur Grigorev, and Victor Lempitsky. Point-based modeling of human clothing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14718–14727, 2021. 2

[41] Hao Zhang, Di Chang, Fang Li, Mohammad Soleymani, and Narendra Ahuja. Magicpose4d: Crafting articulated models with appearance and motion control. *arXiv preprint arXiv:2405.14017*, 2024. 2, 8

[42] Hao Zhang, Fang Li, and Narendra Ahuja. Open-nerf: Towards open vocabulary nerf decomposition. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3456–3465, 2024. 2

[43] Hao Zhang, Fang Li, Samyak Rawlekar, and Narendra Ahuja. Learning implicit representation for reconstructing articulated objects. *arXiv preprint arXiv:2401.08809*, 2024.

[44] Hao Zhang, Fang Li, Samyak Rawlekar, and Narendra Ahuja. S3o: A dual-phase approach for reconstructing dynamic shape and skeleton of articulated objects from single monocular video. In *International Conference on Machine Learning*, pages 59191–59209. PMLR, 2024. 2

[45] Tianyuan Zhang, Hong-Xing Yu, Rundi Wu, Brandon Y Feng, Changxi Zheng, Noah Snavely, Jiajun Wu, and William T Freeman. Physdreamer: Physics-based interaction with 3d objects via video generation. In *European Conference on Computer Vision*, pages 388–406. Springer, 2024. 2

[46] Haoyu Zhao, Hao Wang, Xingyue Zhao, Hongqiu Wang, Zhiyu Wu, Chengjiang Long, and Hua Zou. Automated 3d physical simulation of open-world scene with gaussian splatting, 2024. 2