PHOTOREALISTIC HUMAN AVATAR GENERATION AND REAL-TIME POSE EDITING WITH GAUSSIAN SPLATTING

A Thesis

by

YI-CHENG HSIAO

Submitted to the Graduate and Professional School of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Li, Xin
Committee Members,	Keyser, John
	McNamara, Ann
Head of Department,	Christine Bergeron

August 2025

Major Subject: Visualization

ABSTRACT

This thesis introduces a two-part framework for photorealistic human avatar reconstruction and real-time pose editing from monocular video. The first module, **GSDHuman**, employs **G**aussian **S**urfels to reconstruct **D**ynamic **Human** avatars within minutes, offering efficient human avatar reconstruction from single-view inputs. The second, **3DFilmPCA**, is a photorealistic avatar system based on rigged **3D Gaussian Splatting**, enabling SMPL-driven **Pre-Composite Action** control. Users can interactively manipulate SMPL joints and instantly visualize updates on the associated Gaussian representation.

Together, these components form a complete pipeline that accelerates the creation of controllable, high-fidelity human models. The proposed methods are tailored for artist-driven workflows in visual effects, digital filmmaking, and virtual human editing. Unlike conventional techniques that rely on complex geometry fitting or canonical space learning, our approach achieves rapid and high-quality results using only monocular video.

Qualitative results on both *in-the-wild* and *lab-captured* video demonstrate that **GSDHuman** efficiently reconstructs photorealistic, dynamic human avatars under diverse conditions. **3DFilmPCA** further enables intuitive and controllable avatar animation, offering a practical, artist-friendly solution for video compositing, virtual production, and digital human performance.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This thesis was supported by a committee consisting of Professor Xin Li (Chair of Committee) and Professor Ann McNamara of the Department of Visualization, and Professor John Keyser of the Department of Computer Science and Engineering, all at Texas A&M University.

Technical discussions with Zhengming Yu, a Ph.D. student in Computer Science and Engineering at Texas A&M University, contributed significantly to the development of **GSDHuman**, particularly in the design of the Gaussian Splatting framework. He provided key insights the implementation of core components, including Outlier Pruning, Normal Consistency Pruning, and 2D Gaussian surfel-based human mesh reconstruction.

The datasets used in this work include the People-Snapshot dataset from TU Braunschweig[1], the Dataset-Demo dataset[2], and selected segments from commercial film footage for demonstration purposes. Additionally, the SMPL template model[3] was used as a human body prior in the reconstruction pipeline.

All remaining work conducted for this thesis was completed independently by the student.

Funding Sources

Graduate study was supported by a fellowship from Texas A&M University and by research funding from the National Science Foundation.

[1] https://graphics.tu-bs.de/people-snapshot
[2] https://chingswy.github.io/Dataset-Demo/
[3] https://smpl.is.tue.mpg.de/

NOMENCLATURE

GSDHuman	Gaussian Surfels for Dynamic Human Reconstruction from Monocular Video
3DFilmPCA	Photorealistic Human Avatars with Rigged 3D Gaussian Splatting for SMPL-Driven Pre-Composite Action Control
3DGS	3D Gaussian Splatting
2DGS	2D Gaussian Splatting
UI	User Interface
NVS	Novel view synthesis
NeRF	Neural Radiance Fields
MLP	Multi-layer Perceptron
GT	Ground Truth

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

1.1 Introduction

The ability to generate photorealistic and animatable 3D human avatars from video is central to numerous applications in computer graphics, computer vision, and immersive media production. From digital stunt doubles in films to personalized avatars in AR/VR environments, high-fidelity digital humans are becoming increasingly essential. In many production scenarios, reshooting scenes due to minor performance issues or continuity errors is both time-consuming and costly. A robust avatar creation pipeline can significantly streamline this process by enabling efficient digital re-enactment. However, most existing systems rely on multi-camera capture setups, controlled studio environments, or expensive motion capture suits—rendering them inaccessible to most creators and impractical in real-world settings.

Recent advances in neural rendering and point-based representations have introduced new ways of modeling deformable objects with compelling realism. Among these, *3D Gaussian Splatting* (3DGS) [1] has emerged as a powerful technique for rendering complex surfaces without requiring dense triangle meshes. Although 3DGS is effective for modeling static scenes and objects, adapting it to dynamic and articulated human bodies presents significant challenges.

To address this, we propose a hybrid approach that combines the strengths of both 2D and 3D Gaussian representations. While 3D Gaussians enable efficient radiance field rendering, they often suffer from multi-view inconsistencies and fail to capture precise surface geometry. In contrast, *2D Gaussians* [2], defined as oriented planar disks, provide view-consistent surface modeling by collapsing volumetric representations into image-aligned splats, making them well-suited for reconstructing detailed and geometrically accurate surfaces from multiview inputs.

The first component, **GSDHuman**, employs *2D Gaussian Surfels (2DGS)* [2] anchored to the surface of a parametric SMPL model [3], enabling efficient and high-fidelity reconstruction of

dynamic human avatars from monocular video within minutes.

The second component, **3DFilmPCA**, introduces a *rigged 3D Gaussian Splatting* (3DGS) framework, in which each 3D Gaussian is associated with SMPL [3] joints and transformed according to skeletal motion. This enables controllable animation and interactive pose editing, supporting real-time visualization and pre-compositing workflows.

Our main contributions for GSDHuman are as follows:

- We present GSDHuman, the method to leverage Gaussian Surfels for reconstructing dynamic 3D human avatars from monocular video in minutes.
- We propose an opacity loss function to address the overlooked role of 2D Gaussian opacity in surface reconstruction, encouraging more faithful geometric representation.
- Extensive experiments demonstrate that GSDHuman can efficiently reconstruct high-quality human avatar surfaces across both controlled datasets (People-Snapshot) and in-the-wild video scenarios.

Our main contributions for 3DFilmPCA are as follows:

- We introduce the first system capable of editing both the shape and pose of human avatars in real-time.
- We develop a custom interactive viewer for real-time rendering and editing, providing a responsive interface for inspecting and manipulating avatar attributes.
- We demonstrate the robustness and versatility of 3DFilmPCA on diverse cinematic sequences, showcasing its adaptability to various motion styles and character appearances.

1.2 Related Works

Our work builds upon recent progress in human reconstruction, Gaussian-based representations, and controllable avatar animation. In this section, we review related literature in four key areas: 3D human reconstruction, Gaussian Splatting for modeling, pose-guided human animation, and animatable Gaussian-based avatars.

1.2.1 3D Human Reconstruction

Reconstructing 3D human avatars has been a long-standing pursuit in both computer vision and graphics. Early approaches focused on recovering 3D geometry from multiview video [4]–[6] by leveraging crossview correspondences or integrating depth information from RGB-D sensors [7]–[10]. In addition, techniques have been developed to incorporate albedo maps for realistic surface appearance [11], [12].

In recent years, neural radiance fields have emerged as a powerful tool for avatar modeling [13]–[20], enabling photorealistic rendering from arbitrary viewpoints using multiview image supervision. However, the reliance on densely calibrated camera rigs in controlled environments limits their broader applicability. To alleviate the dependency on multi-camera systems, methods such as [19]–[21] explore monocular video-based solutions, allowing avatar reconstruction and animation with only a single camera. However, these monocular methods often incur long training times. For instance, InstantAvatar [22] leverages Instant-NGP [23] to accelerate convergence, while InstantNVR [24] adopts voxel-based representations to reduce computational load.

Nonetheless, producing accurate surface geometry remains a challenging task for most NeRF-based pipelines. To resolve this, recent methods [25], [26] have shifted toward implicit surface representations that can better capture fine-grained details such as clothing folds and hair strands. Techniques like ICON [27] and ECON [28] integrate SMPL priors to manage difficult human poses, but they are mainly tailored for static images and show limitations when extended to video sequences. SelfRecon [25] introduces neural surface rendering to maintain temporal consistency

across frames, while Vid2Avatar [26] incorporates a Signed Distance Field (SDF) with surface-guided volume rendering to reconstruct avatars from unconstrained video input.

While these advancements are promising, the training process for most systems remains time-consuming, often taking several hours. This limitation hinders their suitability for real-time or scalable applications.

1.2.2 3D Human Modeling from 3D Gaussian Splatting

The emergence of 3D Gaussian Splatting (3DGS)[1] has recently drawn considerable attention for its ability to achieve both real-time rendering and high visual fidelity. Several avatar reconstruction methods[29]–[32] have adopted this representation, enabling significantly faster training and interactive rendering performance.

Despite these strengths, current 3DGS-based approaches still face challenges in precise surface representation. The lack of explicit surface modeling leads to view inconsistency in novel viewpoints and hinders downstream applications that rely on accurate geometry, such as mesh extraction or animation.

GoMAvatar [32] addresses this by associating 3D Gaussians with a mesh template and jointly optimizing the mesh vertex positions. While this strategy enhances surface alignment, it introduces a trade-off—slower optimization speed and reliance on the fixed SMPL topology, which may cause artifacts or incorrect surface reconstruction under diverse human poses or clothing styles. In contrast, our method allows 2D Gaussian primitives to deform freely without being restricted by mesh connectivity, enabling more flexible surface fitting and significantly accelerating training.

1.2.3 Human Animation and Pose Transfer

Pose transfer and human animation aim to synthesize human images under new poses while preserving identity. GAN-based [33] methods often suffer from occlusions and identity drift. Recent works adopt diffusion models for improved quality and control. MagicPose [34] employs a two-stage training strategy with a pose ControlNet and an appearance-aware attention module,

achieving high-fidelity identity preservation and pose retargeting. DisCo [35] introduces a CLIP-guided pose control mechanism, but requires fine-tuning for out-of-domain generalization. DreamPose [36] and Animate Anyone [37] further extend diffusion frameworks to video synthesis, with varying levels of identity consistency and realism. These methods demonstrate the potential of diffusion-based animation, while highlighting challenges in temporal stability and generalizability.

1.2.4 Animatable 3D Gaussian Splatting Human Avatar

3D Gaussian Splatting has recently been extended beyond static scene representation to enable animatable avatars with high-quality appearance and real-time rendering. These methods typically bind Gaussians to deformable mesh structures, such as triangle surfaces or articulated skeletons, allowing controllable animation and expressive reconstruction. For instance, GaussianAvatars [38] binds each 3D Gaussian to a triangle on a FLAME [39] [40] mesh and introduces a local coordinate system to enable independent optimization of translation, rotation, and scaling, leveraging a binding inheritance strategy to achieve photorealistic and expression-driven head animation in real time.

Additionally, HUGS: Human Gaussian Splats [30] introduces an animatable human avatar with a disentangled scene representation, trained on 50-100 frames of monocular video to achieve novel view rendering and human pose synthesis at 60 FPS, utilizing SMPL to capture body details not modeled by SMPL (e.g., cloth, hair). Likewise, GauHuman: Articulated Gaussian Splatting from Monocular Human Videos [41] presents a 3D human model with Gaussian Splatting, achieving fast training (1-2 minutes) and real-time rendering with 13k Gaussians, incorporating a novel merge operation to further speed up the process by pruning redundant Gaussians.

These approaches clearly demonstrate the effectiveness of combining structured mesh priors with Gaussian representations, it is possible to reconstruct and animate expressive human avatars with fine detail and real-time performance, offering a promising direction for controllable neural rendering.

2.1 Method

The overall architecture of our reconstruction framework is illustrated in Fig. 2.1. We first introduce key background concepts—2D Gaussian Splatting and the SMPL model—in Sec. 2.1.1. The placement of 2D Gaussian surfels onto the human template surface is then explained in Sec. 2.1.2, followed by the deformation process using Linear Blend Skinning (LBS), described in Sec. 2.1.3. Our geometry-aware pruning technique and Wasserstein distance-based densification strategy are presented in Sec. 2.1.4. Finally, Sec. 2.1.5 outlines the training loss formulation and regularization techniques that guide our optimization process.

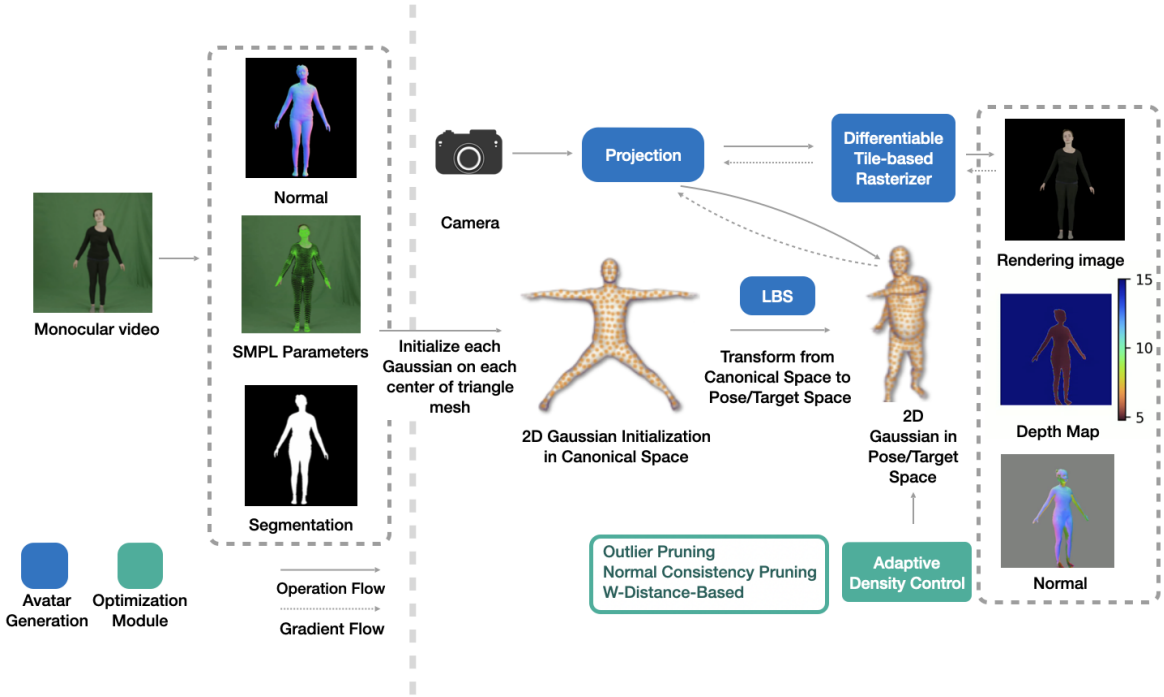

Figure 2.1: Framework of GSDHuman. We begin by initializing a set of 2D Gaussian primitives on the surface of the SMPL template mesh in the canonical space. These Gaussians are then deformed into the posed space using Linear Blend Skinning (LBS) for differentiable rendering. To prevent over-densification and maintain geometric consistency, we apply outlier pruning, normal-consistency pruning, and a Wasserstein distance-based densification strategy.

2.1.1 Preliminaries

2D Gaussian Splatting. Following the formulation in [2], we adopt 2D Gaussian Splatting (2DGS) to represent 3D geometry using a collection of elliptical primitives—referred to as *Gaussian Surfels*. Each surfel can be viewed as a 2D Gaussian distribution embedded on a local plane in 3D space, providing a compact and differentiable representation of surface appearance and structure. Given its planar nature, a Gaussian Surfel is parameterized by a center point $\mathbf{p}_k$, two orthogonal basis directions $\mathbf{t}_u$ and $\mathbf{t}_v$, and their associated scales s_u and s_v, which control its extent. A local coordinate point (u, v) maps to a 3D position via:

$$P(u, v) = \mathbf{p}_k + s_u \mathbf{t}_u v + s_v \mathbf{t}_v u, \quad u, v \in [-1, 1], \tag{2.1}$$

This transformation can also be written in matrix form as:

$$\mathbf{H} = \begin{bmatrix} s_u \mathbf{t}_u & s_v \mathbf{t}_v & 0 & \mathbf{p}_k \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{RS} & \mathbf{p}_k \\ 0 & 1 \end{bmatrix}, \tag{2.2}$$

where $\mathbf{R}$ is the local frame matrix and $\mathbf{S}$ encodes the anisotropic scaling.

The intensity value of a 2D Gaussian evaluated at (u, v) is defined by the standard Gaussian function:

$$\mathcal{G}(\mathbf{u}) = \exp\left(-\frac{u^2 + v^2}{2}\right). \tag{2.3}$$

To render the Gaussian onto an image plane, we determine the intersection between the Gaussian's support plane and the viewing ray from each pixel. Let $\mathbf{x} = (x, y)$ denote a screen-space coordinate. Two auxiliary plane equations aligned with the image axes are defined as:

$$\mathbf{h}_x = (-1, 0, x), \quad \mathbf{h}_y = (0, -1, 0, y). \tag{2.4}$$

These are transformed into the Gaussian's local space using the camera-to-world projection $\mathbf{W}$

and the surfel transformation $\mathbf{H}$:

$$\mathbf{h}_u = (\mathbf{WH})^T \mathbf{h}_x, \quad \mathbf{h}_v = (\mathbf{WH})^T \mathbf{h}_y. \tag{2.5}$$

The coordinates of the intersection point $\mathbf{u}(\mathbf{x})$ can then be computed from the homogeneous parameters as:

$$\mathbf{u}(\mathbf{x}) = \begin{bmatrix} \frac{\mathbf{h}_v^T \mathbf{h}_u^4 - \mathbf{h}_v^T \mathbf{h}_u^2}{\mathbf{h}_u^T \mathbf{h}_v^2 - \mathbf{h}_v^T \mathbf{h}_u^2} \\ \frac{\mathbf{h}_v^T \mathbf{h}_u^4 - \mathbf{h}_v^T \mathbf{h}_u^4}{\mathbf{h}_u^T \mathbf{h}_v^2 - \mathbf{h}_v^T \mathbf{h}_u^1} \end{bmatrix}, \quad \mathbf{v}(\mathbf{x}) = \begin{bmatrix} \frac{\mathbf{h}_v^T \mathbf{h}_u^1 - \mathbf{h}_v^T \mathbf{h}_u^4}{\mathbf{h}_u^T \mathbf{h}_v^2 - \mathbf{h}_v^T \mathbf{h}_u^1} \\ \frac{\mathbf{h}_v^T \mathbf{h}_u^1 - \mathbf{h}_v^T \mathbf{h}_u^4}{\mathbf{h}_u^T \mathbf{h}_v^2 - \mathbf{h}_v^T \mathbf{h}_u^1} \end{bmatrix}. \tag{2.6}$$

Once projected onto the screen, each Gaussian contributes to the pixel color and opacity based on its value at the intersection point. These contributions are composited using alpha blending along the ray direction, sorted by depth:

$$c(\mathbf{x}) = \sum_{i=1}^{s-1} c_i \alpha_i \mathcal{G}_i(\mathbf{u}(\mathbf{x})) \prod_{j=1}^{i-1} \left(1 - \alpha_j \mathcal{G}_j(\mathbf{u}(\mathbf{x}))\right), \tag{2.7}$$

where c_i and α_i represent the color and opacity of the i-th surfel, and the product term models occlusion and transparency along the ray.

For further details, the reader is referred to [2].

SMPL Model. SMPL [3] is a parametric human body model defined as $M(\beta, \theta)$, where β controls body shape and θ governs body pose. To represent dynamic human motion, 3D points defined in the canonical space are transformed into the posed space using the Linear Blend Skinning (LBS) algorithm [3]. Specifically, a point $\mathbf{p}_c$ in canonical space is mapped to its posed counterpart $\mathbf{p}_p$ as follows:

$$\mathbf{p}_p = \sum_{k=1}^{K} w_k \left(\mathbf{R}_k(J, \theta)\, \mathbf{p}_c + \mathbf{t}_k(J, \theta, \beta)\right),$$

where J denotes the joint locations of the K skeletal joints, $\mathbf{R}_k(J, \theta)$ and $\mathbf{t}_k(J, \theta, \beta)$ are the rotation and translation of joint k, and w_k is the skinning weight corresponding to joint k.

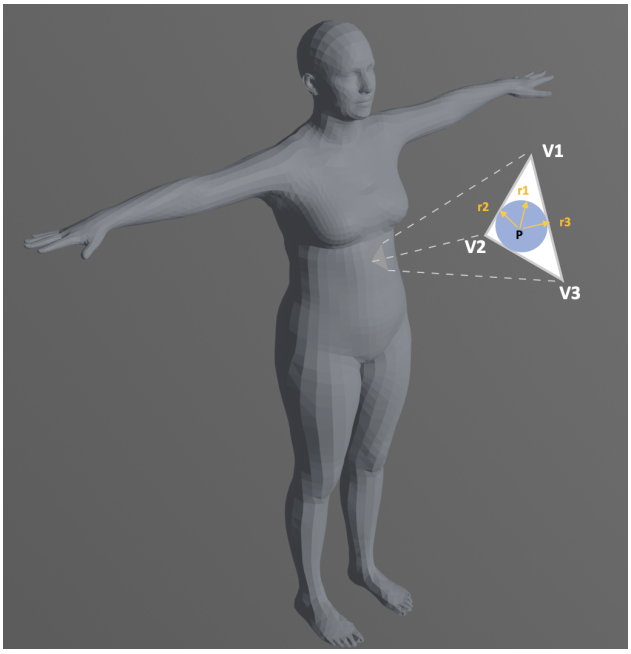

Figure 2.2: Initialization of 2D Gaussian. The 2D Gaussian is initialized at point $\mathbf{p}$, the centroid of the triangle formed by vertices $\mathbf{v}_1$, $\mathbf{v}_2$, and $\mathbf{v}_3$. The covariance matrix is computed from the basis $\mathbf{r}_1$, $\mathbf{r}_2$, and $\mathbf{r}_3$ spanning the triangle's surface.

2.1.2 Data Preprocessing

Figure 2.3: Visualization of the SMPL estimation. From left to right: input image, SMPL overlay, segmentation mask, and surface normal.

Prior to performing 2D Gaussian Splatting, a series of preprocessing steps are applied to the raw input data to extract essential geometric and semantic information. This study employs the SMPL model [3] to estimate human body pose and shape, generating auxiliary signals to support subsequent 3D reconstruction and rendering processes. Specifically, the preprocessing pipeline includes the following steps: first, the human region is extracted from the input image; next, the SMPL

model is applied to estimate the 3D shape and pose of the human body; then, a segmentation mask is generated to separate the human from the background; finally, surface normals are computed to provide geometric details.

Figure 2.3 illustrates the visualization results of the preprocessing steps. From left to right, the sub-images depict: (1) the original input image, showing a person in a real-world scene; (2) the SMPL overlay, presented as a green mesh representing the SMPL model's estimation of body shape and pose; (3) the segmentation mask, shown as a black-and-white binary image highlighting the human silhouette; (4) the surface normals, encoded with colors to indicate the geometric orientation of the body surface. These preprocessing steps provide critical geometric and semantic foundations for the subsequent 2D Gaussian Splatting process.

2.1.3 Initialization

As part of the GSDHuman framework, we initialize 2D Gaussian surfels directly on the surface of a template mesh such as SMPL. For each triangle face defined by vertices $[v_1, v_2, v_3] \in R^3$, the surface normal is computed as:

$$\mathbf{n} = \frac{(v_2 - v_1) \times (v_3 - v_1)}{\|(v_2 - v_1) \times (v_3 - v_1)\|}. \tag{2.8}$$

A Gaussian is placed at the centroid of the triangle, given by:

$$\mathbf{p}_k = \frac{1}{3}v_1 + \frac{1}{3}v_2 + \frac{1}{3}v_3, \tag{2.9}$$

and oriented using a local orthonormal frame constructed as:

$$\mathbf{r}_1 = \frac{v_1 - \mathbf{p}_k}{\|v_1 - \mathbf{p}_k\|}, \quad \mathbf{r}_3 = \mathbf{n}, \quad \mathbf{r}_2 = \mathbf{r}_3 \times \mathbf{r}_1, \tag{2.10}$$

where the rotation matrix is defined as $R = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3]$.

The initial scale $S = \{s_1, s_2\}$ of each Gaussian is set isotropically, with both axes initialized to half the distance between v_1 and the centroid:

$$s_1 = s_2 = \frac{1}{2}\|v_1 - \mathbf{p}_k\|. \tag{2.11}$$

2.1.4 2D Gaussian Surfels Deformation

To enable articulated deformation of our 2D Gaussian surfels, we adopt a motion-aware transformation strategy rooted in the SMPL body model [3]. Inspired by the success of NeRF- and 3DGS-based avatar animation [19]–[21], [29]–[31], [42], we transform each surfel from the canonical space to the posed configuration by leveraging skinning weights estimated from SMPL.

The deformation process is governed by Linear Blend Skinning (LBS), where both position and orientation are linearly blended from joint transformations. Given a canonical position $\mathbf{p}_c$ and rotation matrix R_c, the posed counterparts $\mathbf{p}_t$ and R_t are computed as:

$$\mathbf{p}_t = R(\mathbf{J}_t, \boldsymbol{\theta}_t)\,\mathbf{p}_c + \mathbf{b}(\mathbf{J}_t, \boldsymbol{\theta}_t, \boldsymbol{\beta}_t), \quad R_t = R(\mathbf{J}_t, \boldsymbol{\theta}_t)\,R_c, \tag{2.12}$$

where $\mathbf{J}_t$ denotes the joint locations, and $\boldsymbol{\theta}_t$, $\boldsymbol{\beta}_t$ are the pose and shape parameters, respectively. The blended rotation and translation are computed by summing over the contributions from all K joints, modulated by their skinning weights w_k:

$$R(\mathbf{J}_t, \boldsymbol{\theta}_t) = \sum_{k=1}^{K} w_k\, R_k(\mathbf{J}_t, \boldsymbol{\theta}_t), \tag{2.13}$$

$$\mathbf{b}(\mathbf{J}_t, \boldsymbol{\theta}_t, \boldsymbol{\beta}_t) = \sum_{k=1}^{K} w_k\, \mathbf{b}_k(\mathbf{J}_t, \boldsymbol{\theta}_t, \boldsymbol{\beta}_t). \tag{2.14}$$

While the blend weights can be directly extracted from the SMPL skinning map, we observe that such weights may be suboptimal for fine-grained surfel-level deformation. Following [19], [26], [29], we incorporate a lightweight MLP to learn residual offsets to the initial weights, allowing for smoother and more accurate deformations.

This formulation enables each surfel to follow the body motion in a temporally coherent manner, preserving geometric fidelity even under complex pose changes.

2.1.5 Pruning and Densification

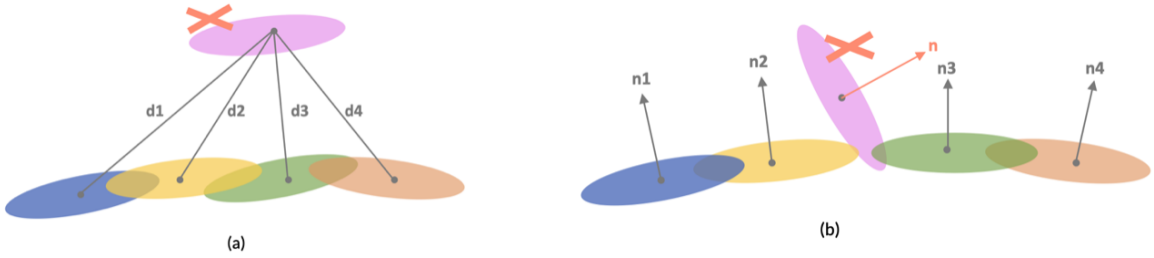

Figure 2.4: Outlier Pruning (a) and Normal Consistency Pruning (b). We prune the 2D Gaussian primitives that are far away from the collective and not aligned with the surface.

Pruning and densification play a vital role in achieving effective 3D Gaussian Splatting [1]. The 2D Gaussian Splatting method [2] adopts similar strategies. However, in our experiments with dynamic human reconstruction, we found that this approach may result in excessive Gaussian concentration. Such over-densification negatively affects both rendering quality and surface fidelity. To mitigate this, we propose an improved pruning and densification strategy.

Outlier Pruning.

We remove Gaussian Surfels that lie far from the main distribution. As shown in Fig. 2.4, we compute the average distance from each Gaussian to its k nearest neighbors. If this average exceeds a threshold ϵ_o, the Gaussian is considered an outlier and pruned. This step helps eliminate noise introduced by isolated Gaussians.

Normal Consistency Pruning.

We also discard Gaussians whose surface normals deviate significantly from their neighbors. For a given Gaussian with normal $\mathbf{n}$, we evaluate its alignment with the normals $\mathbf{n}_i$ of its k nearest

neighbors using average cosine similarity:

$$\frac{1}{k}\sum_{i=1}^{k}\left|\frac{\mathbf{n}\cdot\mathbf{n}_i}{\|\mathbf{n}\|\|\mathbf{n}_i\|}\right|.$$

If this similarity is below a threshold ϵ_n, the Gaussian is pruned. This ensures that the retained Gaussians conform to the local surface orientation.

Wasserstein Distance-Based Densification.

Typical densification strategies in 3D and 2D Gaussian Splatting are gradient-driven. Yet, we observe that this can lead to redundant Gaussians clustered in already well-sampled regions, as also noted by [29]. This not only reduces rendering efficiency but also impairs optimization. To avoid such issues, we use the **Wasserstein distance** [43], [44] to regulate densification. The distance between two Gaussians μ_1 and μ_2 is defined as:

$$\mathcal{W}(\mu_1,\mu_2) = \|\mathbf{p}_1-\mathbf{p}_2\|^2 + \mathrm{tr}(\mathbf{C}_1+\mathbf{C}_2-2(\mathbf{C}_2^{\frac{1}{2}}\mathbf{C}_1\mathbf{C}_2^{\frac{1}{2}})^{\frac{1}{2}}) \tag{2.15}$$

where $\mathbf{p}_1$ and $\mathbf{p}_2$ are the mean positions, and $\mathbf{C}_1$, $\mathbf{C}_2$ are the covariance matrices. A Gaussian is allowed to densify only if its Wasserstein distance $\mathcal{W}$ to its closest neighbor exceeds a threshold ϵ_w.

2.1.6 Training Loss and Regularization

To enhance the reconstruction of human body geometry, we adopt a similar loss design and regularization strategy as in 2D Gaussian Splatting [2]. In addition, by incorporating human-specific priors, we introduce a normal prediction module to guide and regularize the surface normals of the Gaussians.

Depth Distortion.

Similar to 3D Gaussian Splatting, the rendering process does not account for the distances between intersected Gaussians along the ray, unlike NeRF. This can lead to spatially spread-out Gaussians

producing comparable color and depth values [2]. To mitigate this, we adopt a depth distortion loss following Mip-NeRF360 [45] and 2D Gaussian Splatting [2], which concentrates the weight distribution along each ray by minimizing the distance between intersections:

$$\mathcal{L}_d = \sum_{i,j} \omega_i \omega_j |z_i - z_j|, \tag{2.16}$$

where $\omega_i = \alpha_i \hat{G}_i(u(x)) \prod_{j=1}^{i-1}(1-\alpha_j \hat{G}_j(u(x)))$ is the blending weight of the i-th splat intersection, and z_i is its corresponding depth. This loss penalizes depth spread and encourages splats to cluster more tightly around the actual surface.

Normal Consistency.

Unlike 3D Gaussians, 2D Gaussian primitives inherently contain surface normals. Since our representation is based on 2D Gaussian surfels, we can enforce normal consistency directly. Instead of aligning the normals with depth gradients as in prior work, we align them with normals predicted by a pretrained monocular human normal estimator. The loss is defined as:

$$\mathcal{L}_n = \sum_i \omega_i (1 - \mathbf{n}_i^\top \mathbf{N}), \tag{2.17}$$

where i indexes over the splats intersected by a ray, ω_i is the corresponding blending weight, $\mathbf{n}_i$ is the view-aligned normal of the splat, and $\mathbf{N}$ is the predicted surface normal from the dataset. This encourages the Gaussian primitives to better align with human body surfaces.

Opacity Regularization.

To ensure stable rendering and meaningful surface reconstruction, we regularize the opacity values of Gaussian primitives. Without this regularization, optimization may collapse into degenerate solutions such as highly transparent Gaussians (causing under-reconstruction) or overly opaque ones (leading to incorrect layering or hard-edged artifacts). Therefore, we encourage the opacity values to converge to binary states—either fully opaque ($p = 1$) or fully transparent ($p = 0$)—which improves compositing stability and surface extraction.

Among the tested opacity regularization weights, we empirically found that a value of **0.005** achieves the best balance between *surface fidelity* and *rendering stability*. Larger weights (e.g., 0.1, 0.5, 1) often lead to training collapse due to overly aggressive regularization that pushes most opacities toward binary values too early, resulting in sparse, incomplete, or noisy reconstructions. Conversely, very small weights (e.g., 0.0001) allow excessive flexibility, producing semi-transparent Gaussian splats that blur surface boundaries and hinder mesh extraction.

Through extensive fine-tuning, we observed that **0.005** consistently yields clean and connected surface reconstructions with minimal artifacts. It encourages opacities to converge to interpretable binary states while preserving sufficient gradient flow for stable optimization. This empirical result is corroborated by the visual and quantitative comparisons shown in Figure 2.5 and Table **??**.

We introduce an opacity regularization term that penalizes uncertain (semi-transparent) values. The loss is defined as:

$$\mathcal{L}_{\text{opacity}} = -E\left[\log(p) + \log(1-p)\right] = -E\left[\log(p(1-p))\right] \tag{2.18}$$

This function reaches its maximum at $p = 0.5$, and decreases toward $-\infty$ as p approaches 0 or 1. Minimizing this loss thus pushes opacity values away from intermediate states, promoting clearer separation and more interpretable rendering.

To ensure numerical stability and avoid undefined values from $\log(0)$ or $\log(1)$, we add a small constant ϵ (e.g., 10^{-7}):

$$\mathcal{L}_{\text{opacity}} = -E\left[\log(p+\epsilon) + \log(1-p+\epsilon)\right] \tag{2.19}$$

Final Loss.

We optimize our model to minimize the following loss function:

$$\mathcal{L} = \mathcal{L}_{color} + \lambda_1 \mathcal{L}_{mask} + \lambda_2 \mathcal{L}_{SSIM} + \lambda_3 \mathcal{L}_{LPIPS} + \lambda_4 \mathcal{L}_d + \lambda_5 \mathcal{L}_n + \lambda_6 \mathcal{L}_{opacity} \tag{2.20}$$

where $\mathcal{L}_{mask} = \left\| \sum_i \omega_i - \hat{M} \right\|_2^2$ minimizes the difference between the rendered alpha map and the human body mask $\hat{M}$. $\mathcal{L}_{color}$ is the L1 loss between the rendered image and the ground truth. $\mathcal{L}_{SSIM}$ is the structural similarity loss, and $\mathcal{L}_{LPIPS}$ is a perceptual loss computed using a pre-trained network. $\mathcal{L}_d$ and $\mathcal{L}_n$ refer to the depth distortion loss and normal alignment loss as described above. $\mathcal{L}_{opacity}$ is opacity loss, which ensures stable rendering and meaningful surface reconstruction.

These terms are all standard image-space losses that guide the optimization across both photometric and geometric fidelity. Empirically, we set: $\lambda_1 = 0.1$, $\lambda_2 = \lambda_3 = 0.01$, $\lambda_4 = 1000$, $\lambda_5 = 0.08$, and $\lambda_6 = 0.005$.

Table 2.1: Mesh extraction results under different opacity weights. Large weights (e.g., 1, 0.5) lead to training failure, while smaller weights (≤ 0.01) allow successful reconstruction.

Weight	1	0.5	0.1	0.05	0.01	0.005	0.001	0.0005	0.0001
Result	fail	fail	fail	fail	ok	ok	ok	ok	ok

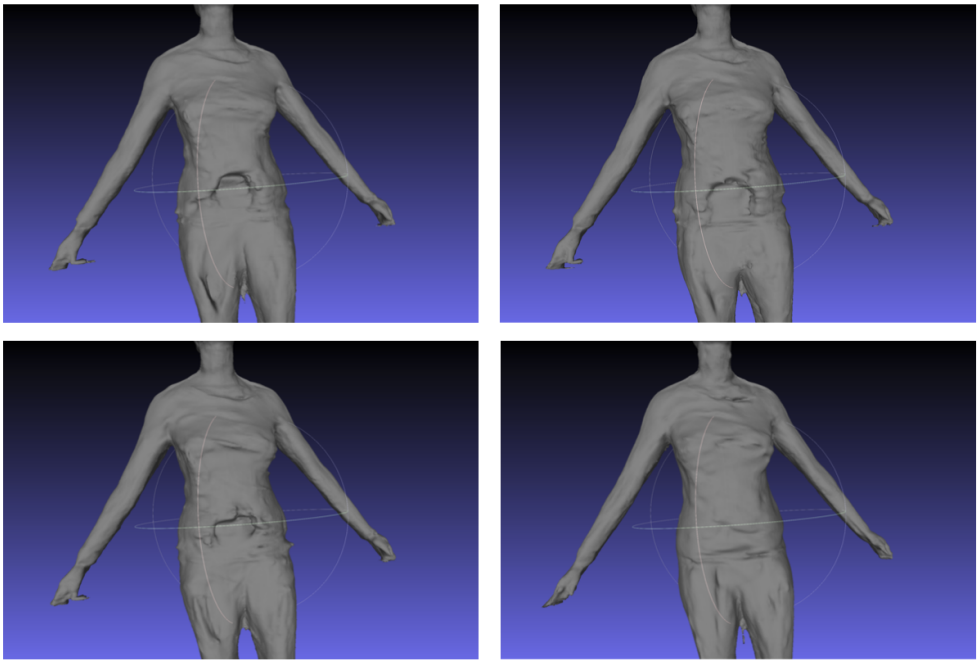

Figure 2.5: Mesh reconstruction results under different opacity weights are shown (Top-Left: 0.0001, Top-Right: 0.001, Bottom-Left: 0.0005, Bottom-Right: 0.005). Among these settings, our experiments indicate that an opacity weight of 0.005 yields the best reconstruction quality.

2.2 Experiments

Assuming camera parameters for the monocular videos are calibrated, we evaluated our method using the ZJU_Mocap Dataset [13], in line with previous works [19], [21], [29], [32], and demonstrated novel view synthesis and 3D reconstruction results. Moreover, we also tried People Snapshot Dataset [46] and in-the-wild video.

2.2.1 Experimental Setup

The NeRF-based novel view synthesis methods, Instant-NVR [24] and InstantAvatar [22], adopt multi-resolution hash encoding to reduce training time to several minutes. The 3DGS-based method GauHuman [29] leverages 3D Gaussian Splatting [1] for human avatar representation, enabling efficient training and high-quality rendering. For geometry reconstruction, GomAvatar [32] builds on 3D Gaussians distributed over mesh surfaces to recover human avatar geometry.

We adopt the same training and evaluation protocol as GauHuman [29] on the ZJU_Mocap dataset. Specifically, we use the provided human masks, camera parameters, SMPL annotations, and estimated surface normals. One camera is selected for training, while the remaining 22 cameras are reserved for novel view synthesis evaluation. For each subject, we follow GauHuman and subsample 1 out of every 5 frames to obtain 100 training frames. For testing, we sample 1 out of every 30 frames from the remaining views to evaluate generalization to unseen viewpoints.

For the People Snapshot dataset [46], where only a single camera is available, we utilize all available images from that camera for training and evaluation. This setup ensures full utilization of the monocular sequence for robust learning in the absence of multi-view supervision.

Implementation details. We used Adam [47] as our optimizer and ran our framework on a single RTX 4090 GPU. We set the threshold ϵ_o for outlier pruning to 0.015 and ϵ_n for normal consistency pruning to 0.3, and ϵ_w for the threshold of Wasserstein distance to 0.012. The number k of neighbors in Sec. 2.1.5 was set to 5.

2.2.2 Extract 3D Mesh

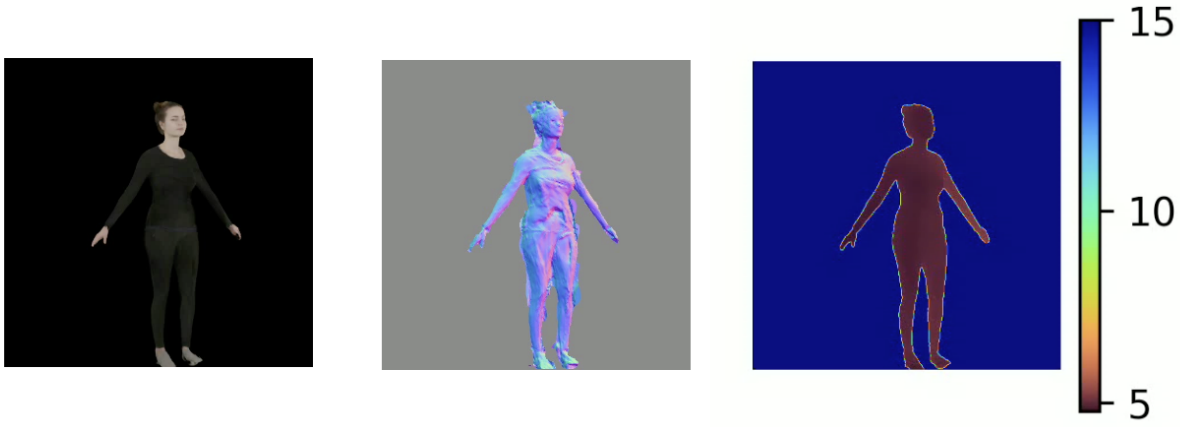

Figure 2.6: 2DGS rendering results of subject female-3-sport from People Snapshot dataset. From left to right: rendered RGB image, surface normal map, and depth map. The depth visualization uses a colormap where warmer colors indicate shallower regions and cooler colors represent farther distances.

To extract a 3D mesh from the rendered output of our Gaussian-based human model, we adopt a volumetric fusion method based on Open3D's GPU-accelerated `VoxelBlockGrid`.

During the testing phase, a set of camera viewpoints is rendered using the optimized Gaussian Splatting model. Each rendered frame provides an RGB image, a depth map, and surface normals. In order to achieve a consistent canonical representation, the subject's pose is normalized to a T-pose by removing the SMPL global rotation and translation.

The rendered depth maps are filtered using the alpha channel to remove low-confidence regions. These processed depth images, along with their corresponding RGB images and calibrated camera intrinsics and extrinsics, are integrated into a shared TSDF (truncated signed distance function) volume. We utilize the `VoxelBlockGrid` module in Open3D to perform efficient depth fusion across views on the GPU. Each RGB-D frame is inserted using its camera pose, which is formatted as a 4×4 extrinsic matrix. After integration, the final mesh is extracted via marching cubes. To

remove disconnected components and floating artifacts, we apply post-processing using connected triangle clustering. Only the largest connected region is retained for surface mesh export. The resulting mesh provides a clean, geometry-consistent reconstruction of the canonical subject.

Figure 2.7: Textured mesh reconstructed from the People-Snapshot scene male-2-outdoor.

In the wild videos.

To enhance the diversity and robustness of our experiments across various real-world conditions,

we include sequences collected from publicly available in-the-wild videos. These videos capture a wide range of lighting, background, and subject variations, providing challenging scenarios for evaluating mesh reconstruction and rendering quality in unconstrained environments.

Figure 2.8: Textured mesh reconstructed from in-the-wild video.

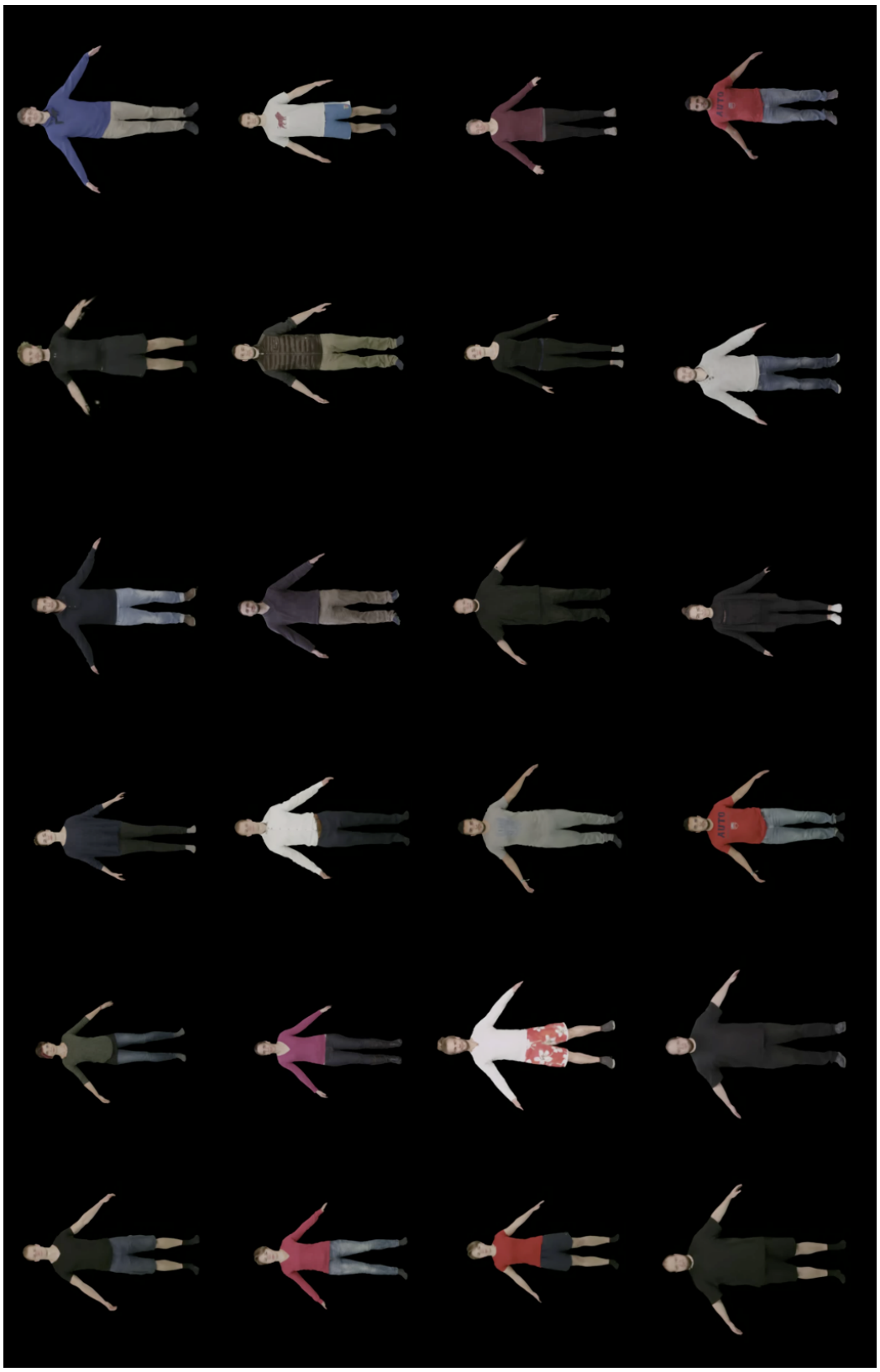

Figure 2.9: 2D Gaussians rendering images on People Snapshot dataset.

Figure 2.10: Textured mesh reconstructed on People Snapshot dataset.

3.1 Method

The overall architecture of our framework is illustrated in Fig. 3.7. We begin by introducing the key background concepts of Binding-Aware 3D Gaussian Parameterization and Binding Inheritance in Sec. 3.1.3. To ensure scale consistency, we propose a strategy that accounts for triangle area, as detailed in Sec. 3.1.4. The interactive viewer is presented in Sec. 3.2.2, followed by shape editing demonstrations in Sec. 3.2.3. Finally, pose manipulation is introduced in Sec. 3.2.4.

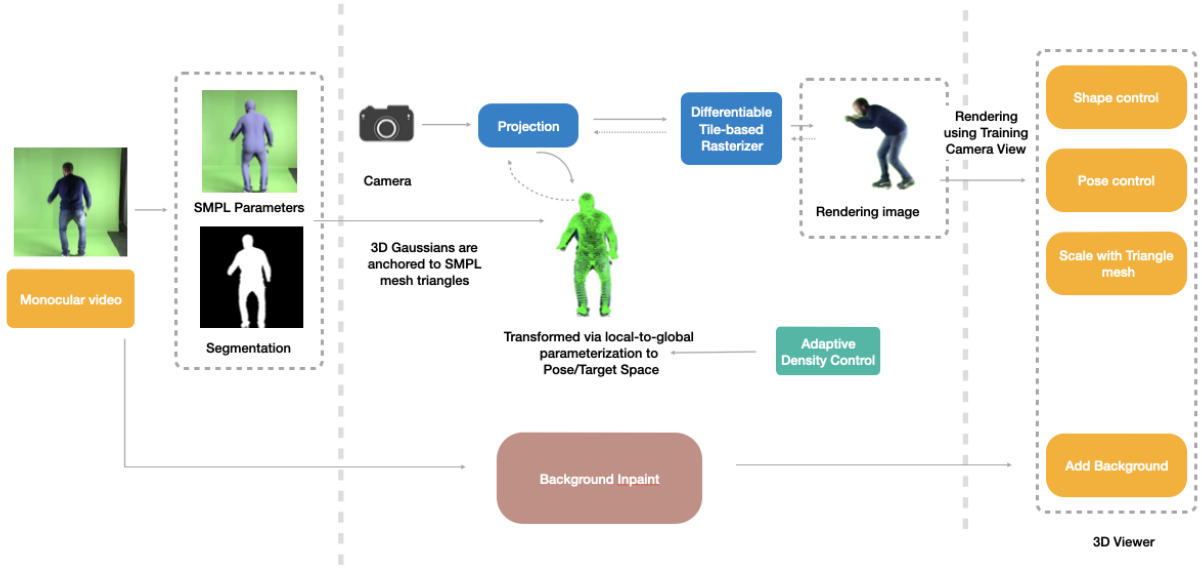

Figure 3.1: Illustration of the 3DFilmPCA pipeline. The orange parts show modules implemented in our system, including monocular video input and interactive controls. The rest of the pipeline follows GaussianAvatars [38], where 3D Gaussians are anchored to SMPL mesh triangles and transformed using a local-to-global parameterization for temporally coherent rendering. Adaptive density control and background inpainting further enhance the rendering quality.

3.1.1 Preliminaries

3D Gaussian Splatting.

3D Gaussian Splatting [1] builds upon traditional point-based rendering and volumetric radiance field techniques, aiming to enable real-time, high-quality novel view synthesis. Unlike mesh or voxel-based representations, 3D Gaussians offer a continuous and differentiable volumetric representation that can be efficiently projected to 2D and rendered using α-blending. Each Gaussian is defined by the tuple:

$$\mathcal{G} = (\mu, \Sigma, \alpha, \mathbf{c}, \mathbf{s})$$

where $\mu \in R^3$ is the 3D mean (position), $\Sigma \in R^{3\times3}$ is the covariance matrix, α is the opacity, $\mathbf{c}$ is the spherical harmonics (SH) color coefficient vector, and $\mathbf{s}$ is the scale of the Gaussian.
The projection of a 3D Gaussian onto the image plane results in a 2D elliptical footprint, which is rendered using differentiable rasterization. The opacity and color contribution to the pixel is accumulated via the α-compositing rule:

$$\mathbf{C}(x, y) = \sum_i \alpha_i \mathbf{c}_i \cdot T_i, \quad T_i = \prod_{j=1}^{i-1}(1 - \alpha_j)$$

where T_i represents the accumulated transparency, ensuring proper visibility handling.
Initialization is a critical step: Gaussian positions μ are derived from a Structure-from-Motion (SfM) sparse point cloud, while scale and opacity are initialized with small constants (e.g., $\sigma = 0.01$, $\alpha = 0.1$). The covariance matrix Σ is typically initialized as a diagonal matrix aligned with camera axes, later optimized during training. SH coefficients c are sampled from input image colors. This setup ensures that 3D Gaussians are differentiable, compact, and flexible, enabling efficient projection, tile-based splatting, and compatibility with gradient descent optimization. **SMPL.**

As discussed in Section 2.1.1 Preliminaries, we have previously introduced the foundational concepts of SMPL.

3.1.2 Data Preprocessing

Our data preprocessing follows the same procedure described in Section 2.1.1. However, in cinematic sequences where the camera poses change rapidly and exhibit large inter-frame motions, conventional segmentation methods tend to produce unstable results. To address this, we replace our segmentation pipeline with one based on **SAPIEN** [48], which offers a comprehensive suite for human-centric vision tasks and enables more precise and temporally stable segmentation masks.

3.1.3 Initialization

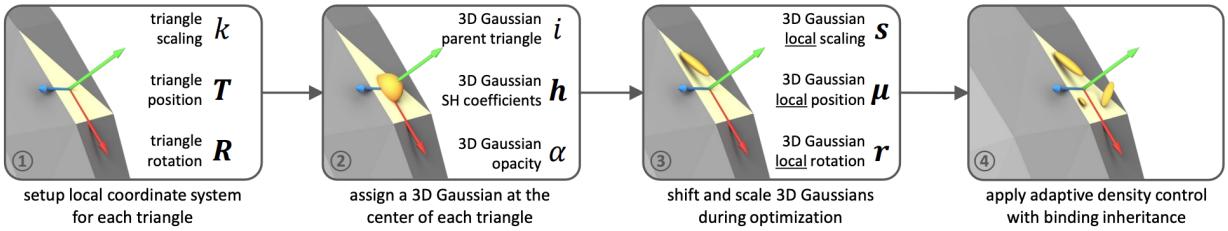

Figure 3.2: Overview of the binding-aware 3D Gaussian parameterization pipeline. (1) For each triangle on the SMPL mesh, a local coordinate system is defined using its rotation R, position T, and scaling factor k. (2) A 3D Gaussian is initialized at the triangle's centroid, with opacity alpha, spherical harmonics (SH) coefficients h, and a parent triangle index i. (3) During optimization, local parameters including position mu, rotation r, and scaling s are learned. (4) Adaptive density control is applied, with binding inheritance to maintain spatial consistency. This illustration is adapted from GaussianAvatars [38].

Binding-Aware 3D Gaussian Parameterization

To ensure that 3D Gaussians deform coherently with dynamic human geometry, we parameterize each Gaussian in the local coordinate system of a triangle on the SMPL mesh, following the strategy proposed in GaussianAvatars [38], which depicted in Figure 3.2. Specifically, for each triangle, we define a local frame using its rotation $\mathbf{R}$, position $\mathbf{T}$, and scale k. A 3D Gaussian is then initialized at the triangle's centroid with a parent-triangle index i, spherical harmonics (SH) coefficients $\mathbf{h}$, and opacity α.

During optimization, the Gaussian's local parameters—including local scale $\mathbf{s}$, local position $\boldsymbol{\mu}$, and local rotation $\mathbf{r}$—are updated. These are converted to global parameters using the triangle's transformation:

$$\mathbf{s}' = k\mathbf{s}, \quad \boldsymbol{\mu}' = k\mathbf{R}\boldsymbol{\mu} + \mathbf{T}, \quad \mathbf{r}' = \mathbf{R}\mathbf{r} \tag{3.1}$$

This formulation allows the Gaussians to move consistently with the underlying mesh as the SMPL parameters (shape $\boldsymbol{\beta}$, pose $\boldsymbol{\theta}$, and translation $\mathbf{t}$) vary over time. The mesh-based anchoring ensures spatial coherence across frames and significantly improves animation quality and temporal consistency. To further enhance rendering performance and fidelity, we employ adaptive density control with binding inheritance. This mechanism allows the model to increase detail in complex regions while maintaining efficiency by reducing density in flat or less important areas.

Binding Inheritance

To capture fine details in dynamic human geometry, simply matching the number of Gaussian splats to the triangles on the SMPL mesh is insufficient. For example, a single triangle on the body may intersect with multiple detailed features, such as hair strands or clothing folds, requiring more splats to represent these complexities. Inspired by Qian et al. [38], we adopt an adaptive density control strategy to dynamically adjust the number of 3D Gaussians based on the view-space positional gradient and opacity of each Gaussian.

For a 3D Gaussian exhibiting a large view-space positional gradient, we either split it into two smaller Gaussians if its scale is large, or clone it if its scale is small. This densification process is performed in the local coordinate system of the associated SMPL triangle, ensuring that newly created Gaussians remain spatially close to their parent Gaussian.

To maintain consistency, each new Gaussian inherits the parent triangle index i of its predecessor, enabling binding inheritance during densification. This approach ensures that the newly added Gaussians enhance the fidelity of the local region while remaining anchored to the same SMPL

triangle.

In addition to densification, we implement a pruning mechanism as part of the adaptive density control strategy [38]. This involves periodically resetting the opacity α of all Gaussians to near-zero values and removing those with opacity below a predefined threshold, effectively suppressing floating artifacts in the rendering. However, in dynamic scenes, excessive pruning can lead to issues, such as the loss of Gaussians in frequently occluded regions (e.g., inner arm triangles). To mitigate this, we track the number of Gaussians attached to each triangle and enforce a minimum of one Gaussian per triangle, ensuring consistent representation across the SMPL mesh during animation sequences.

3.1.4 Optimization

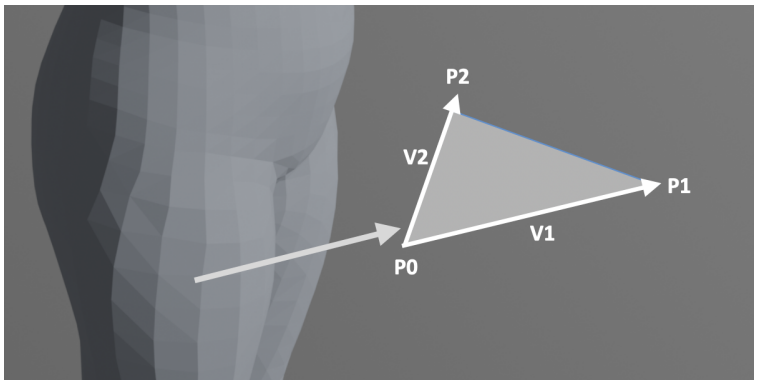

Figure 3.3: Adaptive scaling of 3D Gaussian Splatting based on triangle area changes.

Adaptive Scaling of 3D Gaussian Splatting with SMPL Shape Changes

When modifying the shape parameters $\boldsymbol{\beta}$ of the SMPL model, the geometry of the underlying mesh undergoes deformation, resulting in variations in the area of individual triangles. To maintain consistency and fidelity in the 3D Gaussian Splatting (3DGS) representation, the scale of Gaussians associated with each triangle must be adaptively adjusted. As depicted in Figure 3.3, the area of a

triangle defined by vertices P_0, P_1, and P_2 is calculated as:

$$\text{Area}(\triangle P_0 P_1 P_2) = \frac{1}{2}\|\mathbf{v}_1 \times \mathbf{v}_2\|, \tag{3.2}$$

where $\mathbf{v}_1 = P_1 - P_0$ and $\mathbf{v}_2 = P_2 - P_0$ are the edge vectors. The scale coefficient is determined as the square root of the ratio of the current triangle mesh area to the previous triangle mesh area:

$$\text{Scale coefficient} = \sqrt{\frac{\text{Area of current triangle mesh}}{\text{Area of previous triangle mesh}}}. \tag{3.3}$$

The updated scale of each 3D Gaussian is then computed by multiplying its original scale by this coefficient:

$$\text{Updated 3DGS scale} = \text{Original 3DGS scale} \times \text{Scale coefficient}. \tag{3.4}$$

This adaptive scaling ensures that the Gaussian splats remain proportionate to the deformed mesh geometry, preserving detail in complex regions and maintaining computational efficiency as the SMPL shape evolves.

3.2 Experiments

3.2.1 Experimental Setup

To validate the effectiveness and generalization of our approach, we conduct a series of qualitative experiments on monocular video inputs sourced from real-world cinematic sequences. Our goal is to demonstrate how our method reconstructs animatable avatars and enables novel video synthesis in diverse environments.

Implementation details.

We used GaussianAvatars [38] as our base framework and modified it to use the SMPL model instead of FLAME. We trained on a single RTX 4090 GPU with the number of iterations reduced to 30,000 (compared to the original 600,000 iterations). Furthermore, while the original setting used 11 multi-view images per time frame for training, we drastically simplified the setup by using

only a single camera view per frame, thus improving training efficiency and memory footprint.

In the original GaussianAvatars setting, the authors conducted experiments on the NeRSemble [49] dataset, which contains 9 subjects recorded across 11 action sequences and 16 camera views. Among them, 10 sequences follow predefined expressions or motions, while the 11th is a free-performance sequence used for visual evaluation. Their training setup uses 9 sequences and 15 cameras for each subject, representing a high-fidelity multi-view capture setting.

In contrast, our approach is designed for monocular human performance scenarios. We adapt the pipeline to operate on the PeopleSnapshot [46] dataset and custom in-the-wild monocular videos, where only one camera view is available. For each frame, we fit the SMPL body model and anchor Gaussians directly on the SMPL mesh to enable controllable rendering and animation from monocular input. This significantly reduces hardware requirements while preserving high-quality reconstruction performance.

We retained most of the original hyperparameters from GaussianAvatars [38] to ensure reconstruction quality. Specifically, we adopted the same learning rates as the original method: 5×10^{-3} for Gaussian positions and 1.7×10^{-2} for scaling, with exponential decay applied to the position learning rate throughout training. For regularization, we followed the same loss weights and thresholds: $\lambda_{\text{position}} = 0.01$, $\lambda_{\text{scaling}} = 1$, with thresholds $\epsilon_{\text{position}} = 1$ and $\epsilon_{\text{scaling}} = 0.6$.

Figure 3.4: Motion Transfer on movie clips: La La Land

Motion Transfer.

We utilize CameraHMR [50] to estimate SMPL poses of characters from iconic movie scenes. These pose sequences are then applied to our trained, editable human avatar represented by a 3D Gaussian model, enabling realistic rendering with controllable poses.

To recover clean backgrounds from the same movie scenes, we first perform foreground human segmentation and then use the Anything model [51] for inpainting. This allows us to generate consistent and visually coherent motion transfer results by compositing the rendered avatar with

the reconstructed background.

Free-Viewpoint Rendering.

The bottom row of the results demonstrates that our avatar can be rendered from arbitrary camera angles with a consistent surface appearance, leveraging the advantages of Gaussian splatting in 3D. These experiments demonstrate the applicability of our approach in video-driven avatar animation, virtual production, and VFX compositing, where controllable rendering and free-view synthesis are essential.

3.2.2 Viewer

Figure 3.5: 3DFilmPCA Interavtive Viewer. An interactive interface for controlling SMPL-based 3D Gaussian avatars. Users can adjust rendering settings, manipulate joint poses, and edit body shape parameters, with real-time visual feedback in the central viewport.

To facilitate the visualization and manipulation of 3D Gaussian Splatting within the context of dynamic human geometry, we developed a custom interface, the *FilmEdit3D - Local Viewer*. This tool enables real-time interaction with the SMPL-based human model and its associated Gaussian representations. As illustrated in Figure 3.5, the viewer provides several key functionalities:

- **Rendering Controls**: The interface displays real-time performance metrics, such as a frame rate of 20 FPS, and the total number of Gaussian points. Users can toggle the visibility of splatting and mesh representations, adjust the scale modifier (e.g., 1.00), and modify the vertical field of view (FOV) (e.g., 20 degrees) to optimize the viewing perspective.
- **Body Poses Control**: The SMPL pose parameters are adjustable via a joint-based interface, allowing manipulation of the pelvis and other joints with Euler angles. This enables dynamic pose editing consistent with the underlying SMPL mesh deformation.
- **Body Shape Control**: Users can modify SMPL shape parameters (e.g., height, chest, waist) to alter the human model's geometry, ensuring the Gaussians adapt to changes in shape parameters β.
- **Time Frame and Scale Adjustment**: The viewer supports changing the time frame and scale of Gaussian Splatting, facilitating the analysis of temporal consistency and spatial coherence across animation sequences.

3.2.3 Shape Editing

Figure 3.6: Shape editing results based on the original video. The leftmost image is a frame extracted from the original input video. The subsequent images illustrate the results of shape editing applied to the 3D human avatar reconstructed from that video. By manipulating the shape parameters of the model, we generate variations representing different body types—including shorter, taller, thinner, and heavier forms.

This module allows real-time editing of the SMPL body shape using PCA-based shape parameters (commonly denoted as $\boldsymbol{\beta}$). Users can control individual body attributes—such as *height*, *chest*, *waist*, and *leg length*—through interactive sliders in the user interface. The shape deformations are immediately reflected in the rendered Gaussian avatar, enabling intuitive visual feedback. This functionality is essential for customizing avatars to match different body types or identities.

Mathematically, the body shape is defined by a linear blendshape function:

$$\mathbf{B}_S(\boldsymbol{\beta}) = \sum_{n=1}^{|\boldsymbol{\beta}|} \beta_n \mathbf{S}_n, \tag{3.5}$$

where $\mathbf{S}_n \in R^{3N}$ is the n-th principal shape direction (learned from registered scans), and β_n is the shape coefficient.

Figure 3.7: Add original background

3.2.4 Poses Editing

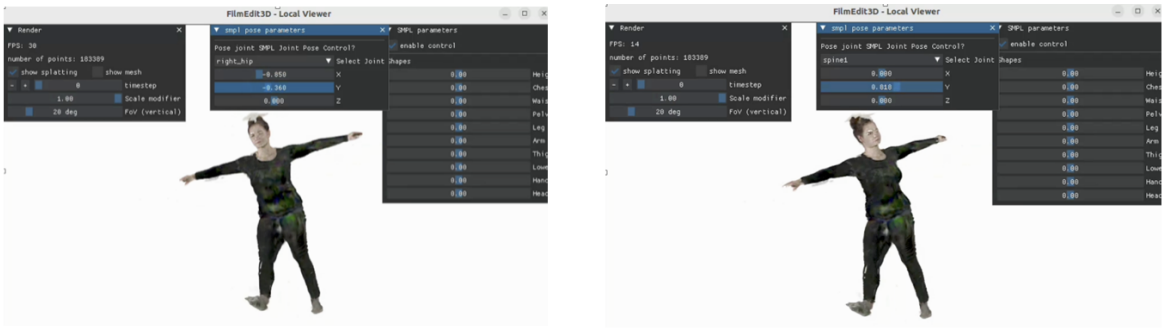

Figure 3.8: Interactive pose editing with the 3DFilmPCA viewer. The interface enables real-time manipulation of SMPL joint rotations. Users can select individual joints and adjust their orientations along the X, Y, and Z axes, with immediate visual feedback shown on the rendered Gaussian avatar.

The pose editing module provides joint-level control over the SMPL model's kinematic hierarchy. Users can select specific joints (e.g., pelvis, arms, head) and adjust their 3D rotation parameters along the X, Y, and Z axes. Internally, this modifies the SMPL pose vector $\boldsymbol{\theta}$, and the associated 3D Gaussian splats are updated accordingly via linear blend skinning (LBS). This feature supports both fine-grained pose tuning and expressive re-targeting for animation purposes.

The pose-dependent blend shapes are expressed as:

$$\mathbf{B}_P(\boldsymbol{\theta}) = \sum_{n=1}^{9K} \left(R_n(\boldsymbol{\theta}) - R_n(\boldsymbol{\theta}^*)\right) \mathbf{P}_n, \tag{3.6}$$

where $R_n(\boldsymbol{\theta})$ are elements of the relative joint rotation matrices (in the kinematic tree), $\boldsymbol{\theta}^*$ is the rest pose, and $\mathbf{P}_n$ are pose blend shape directions. Note that K denotes the number of joints (typically 24 in SMPL), and each 3×3 rotation matrix is flattened into 9 components, resulting in a total of $9K = 216$ pose-dependent terms.

The final skinned mesh is computed using the linear blend skinning (LBS) formulation:

$$\mathbf{t}'_i = \sum_{k=1}^{K} w_{k,i} \mathbf{G}_k^0(\boldsymbol{\theta}, J(\boldsymbol{\beta})) \left(\bar{\mathbf{t}}_i + \mathbf{B}_{S,i}(\boldsymbol{\beta}) + \mathbf{B}_{P,i}(\boldsymbol{\theta})\right), \tag{3.7}$$

where $\mathbf{t}'_i$ is the transformed vertex, $w_{k,i}$ is the skinning weight of joint k for vertex i, and $\mathbf{G}_k^0$ is the global transformation matrix relative to the rest pose.

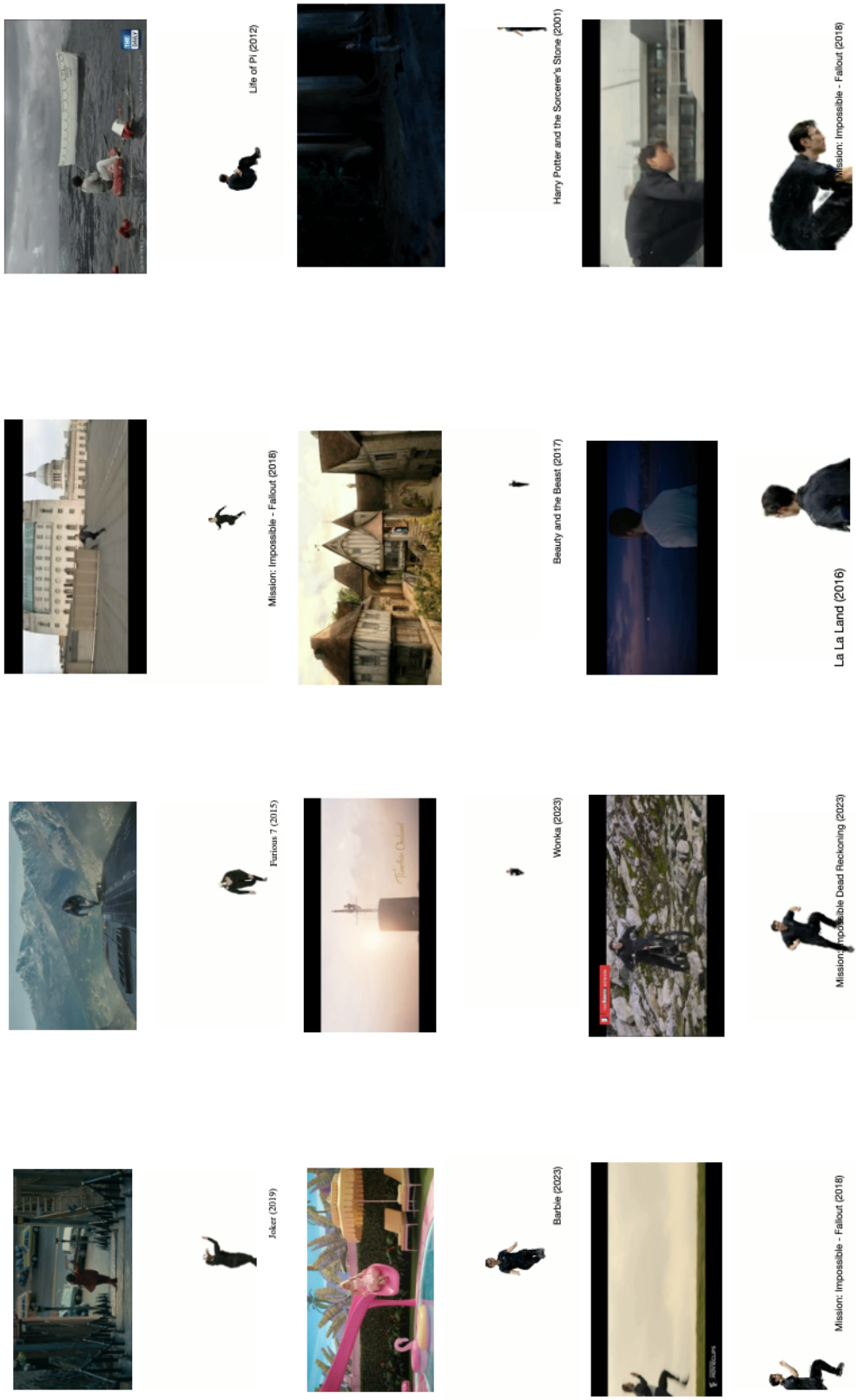

Figure 3.9: Rendering result of novel view synthesis based on movie clips.

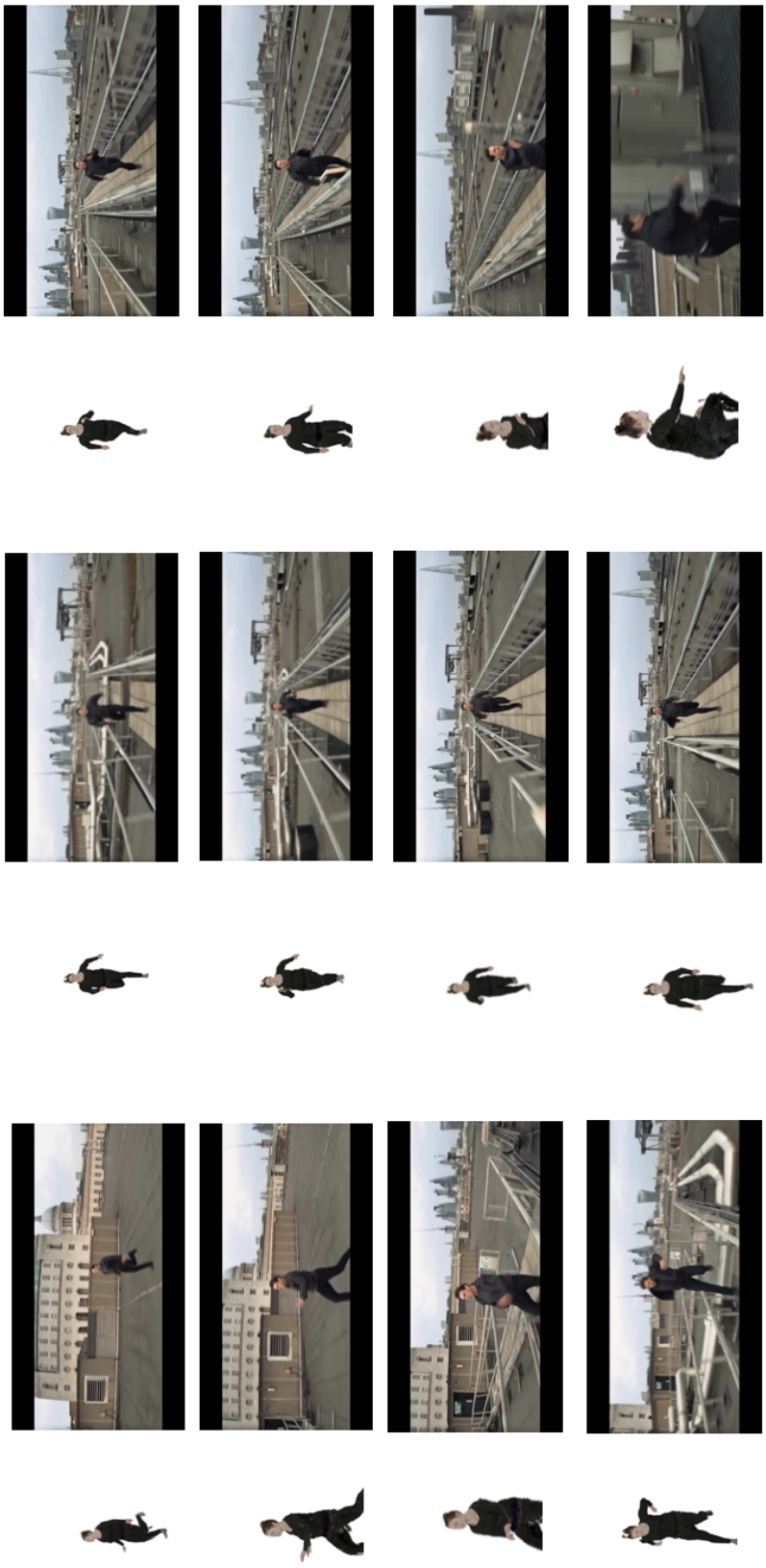

Figure 3.10: Rendering result of novel view synthesis based on a scene from Mission: Impossible – Fallout (2018).

4. DISCUSSION AND CONCLUSIONS

4.1 Challenges

Despite the promising results achieved by our system, several technical and practical challenges remain in the development and deployment of real-time human avatar reconstruction and reanimation tools.

First, handling real-world monocular video remains a complex issue due to diverse lighting, motion blur, and occlusions. Although the system is capable of segmenting and reconstructing dynamic humans, outlier Gaussians often appear in regions with ambiguous or noisy visual information, which can degrade visual quality.

Second, Gaussian stability and control present difficulties during pose deformation. When Gaussian points are rigged to deform with the underlying SMPL mesh, subtle inaccuracies in mesh topology or rapid pose transitions may cause drifting or distortion. Additionally, managing the degrees of freedom of each Gaussian while maintaining rendering efficiency requires careful design of optimization constraints.

Third, while interactive editing of body poses is achievable in 3DFilmPCA, supporting high-fidelity reanimation across frames without breaking temporal coherence is challenging, especially in fast-motion sequences. Further work is needed to ensure smoother transitions and consistent appearance across frames during editing.

Lastly, there is a trade-off between mesh quality and reconstruction speed. Although GSDHuman significantly outperforms previous methods in reconstruction time, the resulting mesh resolution and topology may be suboptimal for certain downstream applications like cloth simulation or facial tracking, which demand higher fidelity.

Overcoming these challenges is key to expanding the system's applicability in both academic and professional VFX pipelines.

4.2 Further Study

While GSDHuman successfully accelerates the reconstruction of dynamic human avatars using 2D Gaussian surfels and achieves high-fidelity textured meshes within minutes, and 3DPreCompAction enables interactive pose editing of Gaussian avatars before video compositing, several future directions remain open to advance this unified pipeline.

On the research side, we plan to further reduce visual artifacts caused by outlier Gaussians, enhance the stability and control of Gaussian behavior through UV-based initialization and local offset learning, and upgrade the model from SMPL to SMPL-X to support finer-grained control of facial expressions and hand gestures. We also aim to improve the extracted mesh quality by integrating more accurate normal estimation and surface refinement techniques.

From an application perspective, future work will focus on enabling multi-person editing, expanding the pipeline to support scenes with multiple interacting characters. The viewer system will also be extended to provide more intuitive user interaction, such as gesture-based joint manipulation or timeline-based animation editing. Additionally, incorporating semantic priors and camera-aware UV mapping could significantly improve reanimation fidelity when deploying in real-world cinematic production.

Together, these future studies will further bridge the gap between real-time human avatar control and artist-friendly tools in professional VFX workflows, solidifying this system's value in both academic research and industry production.

4.3 Conclusion

In this thesis, we presented two complementary approaches for photorealistic human avatar reconstruction and controllable animation from monocular video. The first approach, **GSDHuman**, reconstructs dynamic human avatars efficiently by leveraging 2D Gaussian surfels anchored to the SMPL model, achieving high-fidelity geometry and texture in minutes. The second approach, **3DFilmPCA**, builds on a rigged 3D Gaussian splatting framework, enabling real-time pose editing

and shape manipulation through interactive SMPL control.

These two approaches address distinct yet related challenges in human modeling: fast avatar creation from limited input and intuitive control of avatar motion. Together, they demonstrate the feasibility of using Gaussian-based representations to support both high-quality reconstruction and artist-friendly editing.

Our experiments on both lab-captured and in-the-wild data highlight the flexibility, efficiency, and visual quality of these techniques. We believe this work lays a solid foundation for future research in human avatar modeling, bridging the gap between advanced rendering algorithms and practical tools for virtual production and immersive content creation.

REFERENCES

[1] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis, *Torch: 3d gaussian splatting for real-time radiance field rendering*, arXiv preprint arXiv:2308.04079, 2023.

[2] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao, "2d gaussian splatting for geometrically accurate radiance fields," *arXiv preprint arXiv:2403.17888*, 2024.

[3] Matthew Loper, Javier Romero, Gerard Pons-Moll, and Michael J. Black, "Smpl: A skinned multi-person linear model," in *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, vol. 34, 2015, 248:1–248:16. DOI: 10.1145/2816795.2818013.

[4] Feng Xu, Yebin Liu, Carsten Stoll, *et al.*, "Video-based characters: Creating new human performances from a multi-view video database," in *ACM SIGGRAPH*, 2011.

[5] Edilson De Aguiar, Carsten Stoll, Christian Theobalt, Naveed Ahmed, Hans-Peter Seidel, and Sebastian Thrun, "Performance capture from sparse multi-view video," in *ACM SIGGRAPH 2008 papers*, 2008, pp. 1–10. DOI: 10.1145/1399504.1401959.

[6] Leonid Sigal, Alexander O. Balan, and Michael J. Black, "Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion," *International journal of computer vision*, vol. 87, no. 1, pp. 4–27, 2010. DOI: 10.1007/s11263-009-0273-6.

[7] Richard A. Newcombe, Dieter Fox, and Steven M. Seitz, "Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 343–352. DOI: 10.1109/CVPR.2015.7298631.

[8] Ari Shapiro, Andrew Feng, Hao Li, Mark Bolas, Gerard Medioni, and Evan Suma, "Rapid avatar capture and simulation using commodity depth sensors," *Computer Animation and Virtual Worlds*, vol. 25, no. 3-4, pp. 201–211, 2014. DOI: 10.1002/cav.1588.

[9] Tao Yu, Zerong Zheng, Kaiwen Guo, *et al.*, "Doublefusion: Real-time capture of human performances with inner body shapes from a single depth sensor," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7287–7296. DOI: 10.1109/CVPR.2018.00762.

[10] Lan Xu, Wei Cheng, Kaiwen Guo, Lei Han, Yebin Liu, and Lu Fang, "Flyfusion: Realtime dynamic scene reconstruction using a flying depth camera," in *IEEE transactions on visualization and computer graphics*, vol. 27, 2019, pp. 68–82. DOI: 10.1109/TVCG.2019.2934321.

[11] Kaiwen Guo, Feng Xu, Tao Yu, Xiaoyang Liu, Qionghai Dai, and Yebin Liu, "Real-time geometry, albedo, and motion reconstruction using a single rgb-d camera," *ACM Transactions on Graphics (ToG)*, vol. 36, no. 4, pp. 1–13, 2017.

[12] Kaiwen Guo, Peter Lincoln, Philip Davidson, *et al.*, "The relightables: Volumetric performance capture of humans with realistic relighting," *ACM Transactions on Graphics (ToG)*, vol. 38, no. 6, pp. 1–19, 2019.

[13] Sida Peng, Yuanqing Zhang, Yinghao Xu, *et al.*, "Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9054–9063. DOI: 10.1109/CVPR46437.2021.00894.

[14] Sida Peng, Junting Dong, Qianqian Wang, *et al.*, "Animatable neural radiance fields for modeling dynamic human bodies," in *International Conference on Computer Vision (ICCV)*, 2021. DOI: 10.1109/ICCV48922.2021.00926.

[15] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, *et al.*, "Nerfies: Deformable neural radiance fields," in *International Conference on Computer Vision (ICCV)*, 2021. DOI: 10.1109/ICCV48922.2021.00578.

[16] Keunhong Park, Utkarsh Sinha, Peter Hedman, *et al.*, "Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields," in *SIGGRAPH Asia*, 2021. DOI: 10.1145/3478513.3480520.

[17] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang, "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction," *Advances in Neural Information Processing Systems*, vol. 34, pp. 27 171–27 183, 2021.

[18] Wei Cheng, Su Xu, Jingtan Piao, *et al.*, "Generalizable neural performer: Learning robust radiance fields for human novel view synthesis," *arXiv preprint arXiv:2204.11798*, 2022.

[19] Fuqiang Zhao, Wei Yang, Jiakai Zhang, *et al.*, *Humannerf: Generalizable neural human radiance field from sparse inputs*, arXiv preprint arXiv:2112.02789, 2021.

[20] Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan, "Neuman: Neural human radiance field from a single video," *arXiv preprint arXiv:2203.12575*, 2022.

[21] Zhengming Yu, Wei Cheng, Xian Liu, Wayne Wu, and Kwan-Yee Lin, "Monohuman: Animatable human neural field from monocular video," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 16 922–16 932. DOI: 10.1109/CVPR52729.2023.01623.

[22] Tianjian Jiang, Xu Chen, Jie Song, and Otmar Hilliges, "Instantavatar: Learning avatars from monocular video in 60 seconds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 16 922–16 932. DOI: 10.1109/CVPR52729.2023.01623.

[23] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Transactions on Graphics (ToG)*, vol. 41, no. 4, pp. 1–15, 2022. DOI: 10.1145/3528223.3530069.

[24] Chen Geng, Sida Peng, Zhen Xu, Hujun Bao, and Xiaowei Zhou, "Learning neural volumetric representations of dynamic humans in minutes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 8759–8770.

[25] 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 (CVPR)*, 2022, pp. 5605–5615.

[26] Chen Guo, Tianjian Jiang, Xu Chen, Jie Song, and Otmar Hilliges, "Vid2avatar: 3d avatar reconstruction from videos in the wild via self-supervised scene decomposition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 12 858–12 868.

[27] Yuliang Xiu, Jinlong Yang, Dimitrios Tzionas, and Michael J. Black, "Icon: Implicit clothed humans obtained from normals," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 13 286–13 296. DOI: 10.1109/CVPR52688.2022.01294.

[28] 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*, 2023, pp. 512–523. DOI: 10.1109/CVPR52729.2023.00056.

[29] Shoukang Hu and Ziwei Liu, "Gauhuman: Articulated gaussian splatting from monocular human videos," *arXiv preprint arXiv:2312.02973*, 2023.

[30] Muhammed Kocabas, Jen-Hao Rick Chang, James Gabriel, Oncel Tuzel, and Anurag Ranjan, *Hugs: Human gaussian splats*, arXiv preprint arXiv:2311.17910, 2023.

[31] Zhijing Shao, Zhaolong Wang, Zhuang Li, *et al.*, *Splattingavatar: Realistic real-time human avatars with mesh-embedded gaussian splatting*, arXiv preprint arXiv:2403.05087, 2024.

[32] Jing Wen, Xiaoming Zhao, Zhongzheng Ren, Alexander G. Schwing, and Shenlong Wang, *Gomavatar: Efficient animatable human modeling from monocular video using gaussians-on-mesh*, arXiv preprint arXiv:2404.07991, 2024.

[33] Alec Radford, Luke Metz, and Soumith Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1406.2661*, 2015.

[34] Di Chang, Yichun Shi, Quankai Gao, *et al.*, *Magicpose: Realistic human poses and facial expressions retargeting with identity-aware diffusion*, 2024. arXiv: 2311.12052 [cs.CV].

[35] Tan Wang, Linjie Li, Kevin Lin, *et al.*, "Disco: Disentangled control for realistic human dance generation," *arXiv preprint arXiv:2307.00040*, 2023.

[36] Johanna Karras, Aleksander Holynski, Ting-Chun Wang, and Ira Kemelmacher-Shlizerman, "Dreampose: Fashion image-to-video synthesis via stable diffusion," 2023, arXiv preprint. eprint: arXiv:2304.06025.

[37] Li Hu, Xin Gao, Peng Zhang, Ke Sun, Bang Zhang, and Liefeng Bo, "Animate anyone: Consistent and controllable image-to-video synthesis for character animation," *arXiv preprint arXiv:2311.17117*, 2023.

[38] Shenhan Qian, Tobias Kirschstein, Liam Schoneveld, Davide Davoli, Simon Giebenhain, and Matthias Nießner, "Gaussianavatars: Photorealistic head avatars with rigged 3d gaussians," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 20 299–20 309.

[39] Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero, "Learning a model of facial shape and expression from 4D scans," *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, vol. 36, no. 6, 2017. [Online]. Available: https://doi.org/10.1145/3130800.3130813.

[40] Soubhik Sanyal, Timo Bolkart, Haiwen Feng, and Michael Black, "Learning to regress 3d face shape and expression from an image without 3d supervision," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2019.

[41] Shoukang Hu and Zhiwei Liu, *Gaussian: Articulated gaussian splatting from monocular human videos*, arXiv preprint arXiv:2312.02973, 2023.

[42] Yiming Wang, Qin Han, Marc Habermann, Kostas Daniilidis, Christian Theobalt, and Lingjie Liu, "Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction," in *International Conference on Computer Vision (ICCV)*, 2023. DOI: 10.1109/ICCV51070.2023.00337.

[43] Martin Arjovsky, Soumith Chintala, and Léon Bottou, "Wasserstein gan," in *International Conference on Learning Representations (ICLR)*, 2017.

[44] FHNHL Hörmander, NSB Totaro, A Vershik, and M Waldschmidt, *Grundlehren der mathematischen wissenschaften 332*. Springer, 2006, vol. 5.

[45] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman, "Mip-nerf 360: Unbounded anti-aliased neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5470–5479. DOI: 10.1109/CVPR52688.2022.00539.

[46] Thiemo Alldieck, Marcus Magnor, Weipeng Xu, Christian Theobalt, and Gerard Pons-Moll, "Video based reconstruction of 3d people models," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR Spotlight Paper, Jun. 2018, pp. 8387–8397, ISBN: 978-1-5386-6420-9. DOI: 10.1109/{CVPR}.2018.00875.

[47] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[48] Rawal Khirodkar, Timur Bagautdinov, Julieta Martinez, *et al.*, "Sapiens: Foundation for human vision models," *arXiv preprint arXiv:2408.12569*, 2024.

[49] Tobias Kirschstein, Shenhan Qian, Simon Giebenhain, Tim Walter, and Matthias Nießner, "Nersemble: Multi-view radiance field reconstruction of human heads," *ACM Trans. Graph.*, vol. 42, no. 4, Jul. 2023, ISSN: 0730-0301. DOI: 10.1145/3592455. [Online]. Available: https://doi.org/10.1145/3592455.

[50] Priyanka Patel and Michael J. Black, "CameraHMR: Aligning people with perspective," in *International Conference on 3D Vision (3DV)*, 2025.

[51] Tao Yu, Runseng Feng, Ruoyu Feng, *et al.*, "Inpaint anything: Segment anything meets image inpainting," *arXiv preprint arXiv:2304.06790*, 2023.