

4D Sports Scene Reconstruction

Keming Jiao

3D COMPUTER VISION | DEPARTMENT OF DESIGN SCIENCES
FACULTY OF ENGINEERING LTH | LUND UNIVERSITY
2026

MASTER THESIS

SONY



4D Sports Scene Reconstruction

From Motion Capture System to Free-Viewpoint Video
using Gaussian Splatting

Keming Jiao



LUND
UNIVERSITY

4D Sports Scene Reconstruction

From Motion Capture System to Free-Viewpoint Video using Gaussian Splatting

Copyright © 2026 Keming Jiao

Published by

Department of Design Sciences
Faculty of Engineering LTH, Lund University
Box 118
SE-221 00 LUND
Sweden

Subject: Degree Project in Virtual Reality and Augmented Reality (MAMM15)

Division: 3D Computer Vision

Supervisor(s): Viktor Larsson, viktor.larsson@math.lth.se

Ludvig Dillén, ludvig.dillen@math.lth.se

Roberto Franceschi, Roberto.Franceschi@sony.com

Olivier Moliner, Olivier.Moliner@sony.com

Examiner: Mattias Wallergård, mattias.wallergard@design.lth.se

Abstract

Key actions, tactical cooperation, and spatial relationships in sports activities are usually recorded through 2D videos, which limits users' ability to understand the game in three-dimensional space. Although professional sports events can provide Free-Viewpoint Video using large-scale camera arrays, such systems are difficult to deploy in low-cost scenarios such as community games and personal training. To address this problem, this thesis presents a 4D sports scene reconstruction and replay pipeline based on motion capture data and 3D Gaussian Splatting representations. The system takes sparse multi-view videos captured by a small number of fixed cameras as input, together with camera calibration, human segmentation, and estimated human pose parameters. It reconstructs dynamic human Gaussian avatars and the static environment separately, and integrates them in a unified interactive viewer for scene composition and rendering. To improve sparse-view multi-person reconstruction, the method incorporates pose correction, non-uniform Gaussian initialization, two-stage training, and a perceptually oriented loss-weight setting. The avatar reconstruction component is evaluated on the GalaBasketball and Hi4D datasets using objective metrics, subjective assessment, and ablation studies. Environment reconstruction and full scene replay are demonstrated separately using BASKET-Multiview data with more background views. The results validate the effectiveness of sparse-view avatar reconstruction and demonstrate the feasibility of the complete replay pipeline, while a fully low-camera end-to-end system remains a direction for future deployment. The thesis also discusses limitations in detail recovery, training efficiency, and the use of generative teacher models.

Keywords: Gaussian Splatting, Gaussian Avatar, 4D Reconstruction, Motion Capture System

Sammanfattning

Centrala handlingar, taktiskt samspel och rumsliga relationer inom idrottsaktiviteter registreras vanligtvis genom 2D-videor, vilket begränsar användarnas möjlighet att förstå spelet i tredimensionellt rum. Även om professionella idrottsevenemang kan erbjuda Free-Viewpoint Video med hjälp av storskaliga kamerasystem, är sådana system svåra att implementera i lågkostnadsscenarier såsom lokala matcher och individuell träning. För att hantera detta problem presenterar denna avhandling en pipeline för 4D-rekonstruktion och återuppspelning av idrottsscener, baserad på rörelsefångstdata och representationer med 3D Gaussian Splatting. Systemet använder glesa flervyvideor inspelade med ett litet antal fasta kameror som in-data, tillsammans med kamerakalibrering, mänsklig segmentering och uppskattade parametrar för mänsklig pose. Det rekonstruerar dynamiska mänskliga Gaussian avatars och den statiska miljön separat, och integrerar dem i en enhetlig interaktiv visare för scenkomposition och rendering. För att förbättra gles flervybaserad flerpersonekonstruktion inför metoden posekorrigering, icke-uniform Gaussian-initiering, tvåstegsträning och en perceptuellt orienterad inställning av förlustvikter. Avatarrekonstruktionskomponenten utvärderas på datamängderna GalaBasketball och Hi4D med hjälp av objektiva mått, subjektiv bedömning och ablationssstudier. Miljörekonstruktion och fullständig scenåteruppspelning demonstreras separat med BASKET-Multiview-data med fler bakgrundsvyer. Resultaten bekräftar effektiviteten hos gles flervybaserad avatarrekonstruktion och demonstrerar genomförbarheten hos den kompletta återuppspelningspipelinen, medan ett helt lågkamera-baserat end-to-end-system fortfarande är en riktning för framtida implementering. Avhandlingen diskuterar även begränsningar i detaljåtergivning, träningseffektivitet och användningen av generativa lärarmodeller.

Nyckelord: Gaussian Splatting, Gaussian Avatar, 4D-rekonstruktion, rörelse fångst system

Acknowledgements

My two years of study and life in Lund have been a wonderful journey. As this journey is coming to an end, I would like to express my sincere gratitude to everyone who has helped and supported me.

I would like to thank my supervisors, Viktor and Ludvig, for their guidance throughout the completion of my thesis. They helped me clarify my ideas, provided constructive perspectives during our discussions based on their experience and knowledge, and gave detailed suggestions on my thesis writing. With their support, I was able to steadily develop the methodology and writing of my thesis, making it more comprehensive and firmly grounded in theory.

I would also like to thank Roberto and Olivier for their support throughout the one and a half years I spent at Sony as an intern and thesis student. They taught me valuable communication skills and practical techniques required in a professional working environment, provided various forms of hardware and software support, offered application-oriented perspectives on the design of my method, and helped me move forward with the thesis process in an organized way.

I am grateful to Günter, whose connection with industry gave me the opportunity to intern at Sony and complete my thesis there. I would also like to thank Mattias for serving as the examiner of my thesis. I am sincerely grateful to both of them for selecting me for this master's programme and for awarding me the scholarship opportunity.

I would like to thank Andrej for welcoming me to Sony for my internship and thesis work, as well as for his care and support over the past year. I wish him a smooth and speedy recovery.

I am also grateful to my friends, who helped me overcome difficulties in both my studies and daily life during my two years in Lund.

Most importantly, I would like to thank my parents for supporting me in coming to Sweden to pursue my master's studies. Although, due to unexpected circumstances, they were not able to come to Sweden in person to witness my thesis defense and

graduation, I believe they will be proud of everything I have achieved here.
Finally, I would like to thank everyone who attended my thesis defense.

Lund, June 2026
Keming Jiao

Table of contents

List of acronyms and abbreviations	10
1. Introduction	11
1.1 Background	11
1.2 Goal	12
1.3 Research Questions	12
1.4 Contributions	13
1.5 Sustainable Development Goals	13
1.6 Ethics	14
2. Theory	15
2.1 3D Reconstruction	15
2.2 SMPL Model	17
2.3 3D Gaussian Splatting	20
2.4 Evaluation Metrics	22
2.4.1 Objective Metrics	22
2.4.2 Subjective Image Quality	23
3. Related Work	26
3.1 Traditional Free-Viewpoint Video	26
3.2 Motion Capture	27
3.3 Human Avatars	28
3.3.1 Mesh Avatars	29
3.3.2 Gaussian Avatars	29
3.4 Unified Framework for Avatars and Environment	31
4. Method	32
4.1 Data Preparation	34
4.1.1 Datasets	34
4.1.2 SMPL Fitting	36
4.1.3 Preprocessed Data Structure	38
4.2 Human Avatars Reconstruction	39
4.2.1 Pipeline Overview	39
4.2.2 Multi-resolution Hash Grid Encoder	40

4.2.3	3D Gaussian Deformation	42
4.2.4	Pruning and Duplication with a Fixed Number of Gaussians	43
4.2.5	Optimization Details	44
4.3	Environment Reconstruction	46
4.4	Interactive Viewer	48
5.	Results	50
5.1	Human Avatars Reconstruction	50
5.1.1	Experiment Settings	50
5.1.2	Comparison	50
5.1.3	Ablation Study	52
5.1.4	Pose-driven Avatar Rendering	56
5.2	Environment Reconstruction	58
5.3	Sports Scene Replay	59
6.	Discussion	61
6.1	Data Quality of SMPL Pose Parameters	61
6.2	Why a Higher λ_{DSSIM} Improves Subjective Avatars Quality	63
6.3	Failure to Learn from a Teacher Model	66
7.	Conclusion	68
	References	69

List of acronyms and abbreviations

3DGS	3D Gaussian Splatting
FPS	Frames Per Second
FVV	Free-Viewpoint Video
GUI	Graphical User Interface
LPIPS	Learned Perceptual Image Patch Similarity
MLP	Multi-Layer Perceptron
NeRF	Neural Radiance Fields
PSNR	Peak Signal-to-Noise Ratio
SAM	Segment Anything Model
SfM	Structure from Motion
SMPL	Skinned Multi-Person Linear Model
SOTA	State-of-the-Art
SSIM	Structural Similarity Index Measure

1 Introduction

This chapter introduces the motivation, objectives, research questions, and main contributions of the thesis. It first presents the background of low-cost free-viewpoint replay for sports scenes, then defines the scope of the proposed system and the research questions that guide the work.

1.1 Background

Sports have become an important part of modern life. People are not only interested in watching match results, but also in reviewing and analyzing key actions, tactical cooperation, and spatial relationships during sports activities. An offensive play, a defensive movement, or an individual action usually involves relative positions among athletes, body poses, motion trajectories, occlusions, and interactions with the surrounding environment. However, conventional photos and videos record these events only from the original camera viewpoints. They lack depth information and complete three-dimensional structure, which limits their ability to support analysis and replay from novel viewpoints. Although some professional sports events can already provide Free-Viewpoint Video (FVV) [6] through large-scale camera arrays, community games, personal training sessions, and other non-professional sports scenarios usually rely on ordinary recording devices and single-view videos. Therefore, achieving low-cost free-viewpoint sports replay with spatial understanding is a practically meaningful problem.

Recent progress in computer vision has provided new possibilities for addressing this problem. Three-dimensional reconstruction, novel view synthesis, and camera-based motion capture systems make it possible to recover scene geometry, human motion, and appearance from image or video inputs. In particular, representations such as NeRF [24] and 3D Gaussian Splatting [16] have shown strong capability in reconstructing high-quality three-dimensional scenes and rendering realistic novel views. Compared with implicit neural representations, 3D Gaussian Splatting provides an explicit spatial representation and efficient rendering performance, making it suitable for applications that require interactive viewpoint changes and interactive replay. At the same time, the development of markerless motion capture systems has reduced the hardware barrier for obtaining human poses, enabling three-dimensional pose and trajectory estimation from a relatively small number of cameras.

Nevertheless, free-viewpoint reconstruction in sports scenes remains challenging. First, community games and training environments usually allow only a small number of fixed cameras, and such sparse-view inputs make it difficult to recover human details, occluded regions, and plausible novel-view appearance. Second, athletes move quickly and often occlude each other. It causes that direct application of static-scene reconstruction methods can lead to blurred geometry, floaters, or unstable structures. Finally, a complete replay system must not only reconstruct individual athletes, but also handle multiple human avatars, the static environment, and their integration and rendering in a unified three-dimensional space.

Motivated by these challenges, this thesis focuses on low-cost 4D reconstruction and free-viewpoint replay for multi-view sports scenes. Here, 4D reconstruction denotes the reconstruction of time-varying 3D scene representations, where the additional temporal dimension captures the motion and appearance changes of dynamic elements across frames. It combines human pose priors provided by a motion capture system with 3D Gaussian Splatting representations, reconstructs dynamic human avatars and the static environment separately, and integrates them in a unified viewer for scene composition and interactive rendering.

1.2 Goal

The goal of this thesis is to design and evaluate a 3D Gaussian Splatting-based reconstruction pipeline for dynamic sports scenes, with particular focus on sparse-view multi-person avatar reconstruction from a small number of fixed cameras. The system aims to reconstruct and render athletes and the scene environment in a unified 4D representation under sparse multi-view input. By combining human pose priors from a motion capture system, Gaussian avatar representations, and static environment reconstruction, the proposed pipeline supports high-quality novel view synthesis, free-viewpoint replay, and interactive visualization. In this way, it provides a low-cost and deployable spatial replay solution for community games, personal training, and other non-professional sports scenarios.

1.3 Research Questions

Based on the above goal, this thesis addresses the following research questions:

RQ1: How can a complete pipeline for 4D reconstruction and free-viewpoint replay of dynamic sports scenes be built from sparse multi-view images, assuming that camera calibration, human masks, skeletons, pose estimates, or SMPL parameters are available from upstream systems or datasets?

RQ2: In multi-person sports scenes with fast motion, frequent occlusion, and limited camera views, how can the training process of Gaussian avatars be improved to enhance human appearance and novel-view rendering quality?

RQ3: How can dynamic human Gaussian avatars and static environment Gaussians be integrated into a unified interactive viewer that supports free-viewpoint observation, motion replay, and interactive visualization for sports scenes?

1.4 Contributions

The main contributions of this thesis are threefold. First, it presents a complete pipeline for 4D reconstruction and free-viewpoint replay of dynamic sports scenes by using upstream calibration, segmentation, skeleton/pose, and SMPL data together with 3D Gaussian Splatting representations. The pipeline organizes human avatars, the static environment, and the interactive viewer within a unified reconstruction and rendering framework. Second, to address pose errors, insufficient initialization, and training instability under sparse-view multi-person settings, the thesis integrates pose correction, non-uniform Gaussian initialization, two-stage training, and a perceptually oriented loss-weight setting, improving the reconstruction quality and novel-view rendering performance of Gaussian avatars. Third, the proposed method is evaluated through objective metrics, subjective assessment, ablation studies, and application demonstrations, demonstrating the superiority of the proposed improvements over existing methods as well as the necessity of each introduced component. The thesis also discusses its limitations and analyzes an unsuccessful teacher-model strategy, providing empirical evidence for low-cost sports free-viewpoint replay and future research on high-fidelity Gaussian avatars. The contribution therefore lies in how these upstream signals from motion capture systems are organized, optimized, and rendered for sparse-view multi-person sports replay, rather than in the design of a new motion capture algorithm.

1.5 Sustainable Development Goals

This study has practical relevance to several United Nations Sustainable Development Goals (SDGs), with particular connections to Goal 4 (Quality Education), Goal 8 (Decent Work and Economic Growth), and Goal 9 (Industry, Innovation and Infrastructure).

The research proposes a FVV review tool for community basketball games. Compared with conventional analysis systems designed for professional competitions, the proposed approach substantially lowers hardware and technical barriers, allow-

ing community players, coaches, and organizers to access data-driven tools for post-game review.

In relation to Goal 4, the FVV review tool provides an intuitive medium for sports education and coaching. Through free-viewpoint playback, coaches and learners can more clearly examine tactical cooperation, positional decisions, and team coordination, which can improve both the effectiveness and accessibility of community sports education.

Regarding Goal 8, this work creates new opportunities for digital services related to community sports, such as low-cost technical analysis, match content production, and remote coaching support. By reducing dependence on expensive equipment and labor-intensive annotation, the system may improve the efficiency of small organizations and individual practitioners.

Finally, in relation to Goal 9, the study contributes to innovation in digital sports infrastructure. The proposed FVV solution can be deployed using existing camera equipment and integrated with community sports platforms, thereby introducing advanced visual computing capabilities into grassroots sports activities and supporting the digital transformation of traditional sports environments.

1.6 Ethics

This study reconstructs and analyzes humans and environments in sports scenes from multi-view videos. Therefore, it raises ethical concerns related to privacy, content authenticity, and potential misuse of the technology.

One major concern lies in personal privacy. A FVV system captures athletes' appearance and motion information. Even in community sports scenarios, participants may still be identifiable through body shape, clothing, motion patterns, or reconstructed facial details. As a result, the collection, storage, and distribution of such data may create privacy risks.

Another important issue concerns the authenticity of generated content. Since the proposed method can synthesize novel-view images and produce editable 3D results, the generated content may be modified, taken out of context, or mistaken for original video footage. This could lead to incorrect interpretations of match events or athlete behavior.

A further ethical risk is the potential misuse of high-precision human reconstruction technology. Such technology may also be extended to behavioral monitoring or unauthorized tracking, raising ethical issues beyond the intended sports application context.

2 Theory

This chapter explains and clarifies the basic concepts and theories used in this thesis.

2.1 3D Reconstruction

The goal of 3D reconstruction is to recover the geometry and appearance of a real scene from 2D images, videos, or depth observations, and to represent the scene in a computational 3D format such as point clouds, meshes, voxels, neural fields, or 3D Gaussians. In the context of free-viewpoint sports scenes, 3D reconstruction is not only used to recover the shape and texture of the static environment, but also provides a unified spatial coordinate basis for later human avatar composition, occlusion reasoning, and novel-view rendering.

Camera imaging is commonly described using the pinhole camera model [9]. Let a 3D point in the world coordinate system be represented in homogeneous coordinates as $\mathbf{X} = [X, Y, Z, 1]^T$, and let its 2D projection on the image plane be $\mathbf{x} = [x, y, 1]^T$. Their relationship can be written as

$$\lambda \mathbf{x} = \mathbf{P}\mathbf{X} = \mathbf{K}[\mathbf{R}|\mathbf{t}]\mathbf{X}, \quad (2.1)$$

where λ is the depth scale factor and $\mathbf{P}$ is the camera projection matrix. The matrix $\mathbf{K}$ denotes the camera intrinsic matrix, which describes internal camera parameters such as focal length, principal point, and pixel scale. It is commonly written as

$$\mathbf{K} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \quad (2.2)$$

Here, f_x and f_y are the focal lengths measured in pixels, (c_x, c_y) is the principal point, and s is the skew coefficient. The matrices $\mathbf{R}$ and $\mathbf{t}$ are the extrinsic parameters, representing the rotation and translation from the world coordinate system to the camera coordinate system. For a world-space point $\mathbf{X}_w$, its camera-space coordinate can be expressed as

$$\mathbf{X}_c = \mathbf{R}\mathbf{X}_w + \mathbf{t}. \quad (2.3)$$

Therefore, camera intrinsics determine how 3D points are projected from camera coordinates onto the pixel plane, while camera extrinsics determine the position and orientation of each camera view in a shared world coordinate system. In multi-view reconstruction, accurate camera calibration is a prerequisite for triangulation, and rendering alignment.

When the same spatial point is observed by two or more cameras, multi-view geometry can infer its 3D position from correspondences across images. For corresponding points $\mathbf{x}$ and $\mathbf{x}'$ in two views, the epipolar constraint is given by

$$\mathbf{x}'^T \mathbf{F} \mathbf{x} = 0, \quad (2.4)$$

where $\mathbf{F}$ is the fundamental matrix that encodes the relative geometry between the two cameras. This constraint indicates that, given a point in the first image, its corresponding point in the second image does not need to be searched over the entire 2D image plane, but only along the corresponding epipolar line [9]. This significantly reduces the matching search space and provides an important geometric constraint for stereo matching and depth estimation.

Traditional image-based 3D reconstruction usually consists of two stages: Structure from Motion (SfM) [30] and Multi-View Stereo (MVS) [31]. SfM first extracts local image features and establishes cross-view correspondences. It then estimates relative camera poses and triangulates sparse 3D points. The reconstruction is usually further refined by bundle adjustment [34], which jointly optimizes camera parameters and 3D point positions by minimizing the reprojection error between observed 2D feature points and projected 3D points. The reprojection error can be written as

$$E = \sum_{i,j} \|\mathbf{x}_{ij} - \pi(\mathbf{K}_i, \mathbf{R}_i, \mathbf{t}_i, \mathbf{X}_j)\|^2, \quad (2.5)$$

where $\mathbf{x}_{ij}$ is the observation of the j -th 3D point in the i -th camera, and $\pi(\cdot)$ denotes the projection function defined by the camera parameters. MVS then uses the estimated camera poses to perform dense matching, estimate depth maps for each view, and fuse multi-view depth information into a denser point cloud or surface model.

In addition to traditional geometric methods, 3D reconstruction can also use depth sensors such as LiDAR, structured-light cameras, or ToF (Time-of-Flight) cameras to directly acquire depth information. These methods reduce the uncertainty of inferring depth from multi-view images, but they usually require additional hardware

and may still be limited in scenes containing transparent or reflective materials, long-distance observations, or fast motion. In recent years, neural rendering methods such as NeRF [24] and 3D Gaussian Splatting [16] have further formulated reconstruction as a differentiable view-synthesis optimization problem: by minimizing the difference between rendered images and training images, the model learns both scene geometry and appearance.

2.2 SMPL Model

SMPL (Skinned Multi-Person Linear Model) is a parametric deformable 3D human body model proposed by Loper et al. in 2015 [23]. Its goal is to represent human body shape and pose with a small number of interpretable parameters while producing a consistent human mesh. This type of model is useful in vision and graphics pipelines for several reasons. First, the number of parameters is fixed and controllable, which makes learning and optimization easier. Second, a given set of parameters corresponds to a deterministic mesh, which is suitable for rendering, reconstruction, and animation. Third, the model is compatible with skeletal skinning systems and can therefore be directly used for motion-driven animation.

The core idea of SMPL is to represent the human body as a deformable triangular mesh with fixed topology. The standard SMPL model contains $N = 6890$ vertices, and its mesh topology remains consistent across different identities and poses. As a result, each vertex has a stable semantic meaning, which is important for texture learning, Gaussian initialization, frame-to-frame correspondence, and animation. Given the shape parameters β and pose parameters θ , SMPL outputs a human mesh:

$$\mathbf{M}(\beta, \theta) = \{\mathbf{V}(\beta, \theta), \mathbf{F}\}, \quad (2.6)$$

where $\mathbf{V}$ denotes the set of mesh vertices and $\mathbf{F}$ denotes the fixed triangular face connectivity. The shape parameters β are usually represented by low-dimensional PCA coefficients and control identity-dependent body variations, such as height, body mass, shoulder width, and limb proportions. The pose parameters θ describe the rotations of the joints in the body skeleton. Common SMPL implementations use 24 joints, where each joint rotation can be represented in axis-angle form. Therefore, the full pose is usually represented by $24 \times 3 = 72$ parameters, including the global orientation of the root joint and the local rotations of the remaining body joints.

The skeletal structure of SMPL is shown in Figure 2.1. The joints start from the pelvis and extend along the legs, spine, head, and arms, forming a tree structure suitable for modeling human motion. The pelvis joint usually acts as the root, controlling global translation and orientation. The hip, knee, ankle, shoulder, elbow,

and wrist joints control limb motion, while the spine, neck, and head joints represent torso bending and head orientation. Since this joint definition is compatible with skeletal skinning, SMPL can naturally generate continuous human motion from pose sequences.

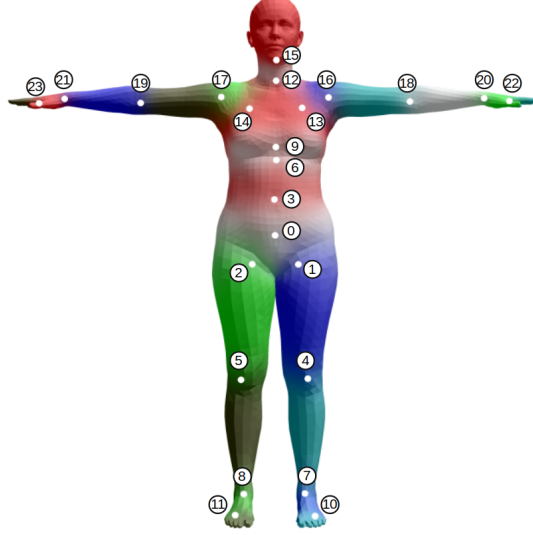


Figure 2.1 The 24-joint skeleton configuration of the SMPL model [23].

To generate the final human mesh from the parameters, SMPL can be understood as first constructing the body shape of a specific person in a standard pose, and then transforming this body into the target pose according to the skeleton. The first step is performed in the canonical pose, or rest pose. Let $\mathbf{T}$ be the mean body template, $\mathbf{B}_S(\beta)$ be the shape blend shapes, and $\mathbf{B}_P(\theta)$ be the pose blend shapes. The template after shape and pose-dependent corrections can be written as

$$\mathbf{T}_P(\beta, \theta) = \mathbf{T} + \mathbf{B}_S(\beta) + \mathbf{B}_P(\theta). \quad (2.7)$$

Here, $\mathbf{T}_P$ is still defined in the standard coordinate system and has not yet been articulated into the target motion by the skeleton. The term $\mathbf{B}_S(\beta)$ adds vertex offsets controlled by the shape parameters, modeling static body-shape differences between individuals, such as height, body mass, shoulder width, and limb proportions. The term $\mathbf{B}_P(\theta)$ is pose-dependent and compensates for local non-rigid deformations that cannot be represented by simple skeletal rotations. For example, when the elbow, knee, or shoulder bends, the body surface does not behave as a purely rigid structure; local changes around muscles, skin, and clothing must also be approximated. The pose blend shapes are used to model these effects.

After the body template has been adapted to the target shape, SMPL also needs to determine the skeletal joint locations for this body shape. Since joint positions vary with body proportions, they are not fixed constants from the mean template. Instead, they are estimated from the shape-deformed mesh by a linear joint regressor:

$$\mathbf{J}(\beta) = \mathcal{J}(\mathbf{T} + \mathbf{B}_S(\beta)), \quad (2.8)$$

where $\mathcal{J}$ denotes a linear mapping from mesh vertices to skeletal joints. Intuitively, each joint location can be regarded as a weighted combination of several related mesh vertices. For instance, the knee joint is determined by vertices around the transition between the thigh and the lower leg. This allows the skeleton to change consistently with the body shape instead of using the same fixed joint positions for all individuals.

Given the joint locations $\mathbf{J}(\beta)$ and pose rotations θ , SMPL applies Linear Blend Skinning (LBS) to transform the template vertices from the standard pose into the final posed mesh. The main idea of LBS is that a vertex is not controlled by only one joint, but is usually influenced by several nearby joints. For example, vertices around the elbow are affected by both the upper arm and the forearm, while vertices around the shoulder are affected by both the torso and the upper arm. For the i -th vertex, its final position can be expressed as

$$\tilde{\mathbf{v}}'_i = \sum_{j=1}^K w_{ij} \mathbf{G}_j(\theta, \mathbf{J}) \tilde{\mathbf{v}}_i, \quad (2.9)$$

where $\tilde{\mathbf{v}}_i$ is the homogeneous coordinate of the i -th vertex in $\mathbf{T}_P$, $K = 24$ is the number of joints, and w_{ij} is the skinning weight describing the influence of joint j on vertex i , typically satisfying $\sum_j w_{ij} = 1$. The matrix $\mathbf{G}_j(\theta, \mathbf{J})$ denotes the global rigid transformation of the j -th joint under the current pose. It is computed by accumulating transformations along the kinematic tree. Therefore, the rotation and translation of the root joint affect the entire body, while the rotation of a local joint mainly affects the body parts in its subtree. The weighted summation in the equation linearly blends the transformed positions from multiple joints, resulting in smooth surface motion.

The equation above gives the core form of LBS. In practical SMPL implementations, each joint transformation is usually also multiplied by the inverse transformation in the rest pose. This removes the initial offset caused by the skeleton configuration of the template and ensures that each vertex is affected only by the relative transformation from the standard pose to the target pose. In this way, SMPL can generate animatable 3D human meshes from a compact set of shape and pose parameters while preserving a fixed mesh topology and stable vertex semantics.

2.3 3D Gaussian Splatting

3D Gaussian Splatting (3DGS) is an explicit differentiable 3D scene representation method proposed by Kerbl et al [16]. It represents a scene as a set of 3D Gaussians, where each Gaussian stores attributes such as center position, scale, rotation, opacity, and color. Compared with volume-rendering methods that require dense sampling along camera rays, 3DGS directly projects Gaussians onto the image plane and synthesizes images efficiently through rasterization. This makes it particularly effective for novel-view rendering.

The original 3DGS pipeline is shown in Figure 2.2. A sparse point cloud obtained from SfM is first used to initialize a set of 3D Gaussians. Given camera parameters, these Gaussians are projected onto the 2D image plane. The projected Gaussians are then processed by a differentiable tile rasterizer, where they are sorted, blended, and alpha-composited to render the current view. The error between the rendered image and the ground-truth training image is back-propagated to the Gaussian parameters, updating their positions, shapes, colors, and opacities.

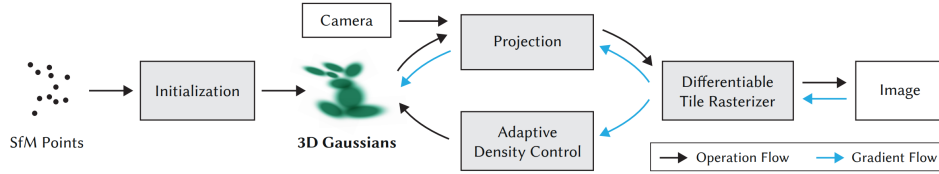


Figure 2.2 Overall optimization and rendering pipeline of 3D Gaussian Splatting [16]. Black arrows indicate the forward operation flow, while blue arrows indicate the gradient flow produced by image reconstruction error.

Each 3D Gaussian can be understood as an anisotropic ellipsoid. Its spatial shape is determined by scale and rotation, and is commonly parameterized as

$$\Sigma = \mathbf{R}\mathbf{S}\mathbf{S}^T\mathbf{R}^T, \quad (2.10)$$

where $\mathbf{R}$ represents rotation and $\mathbf{S}$ represents scale. With this formulation, a Gaussian is not only a dimensionless point, but also covers a local spatial region, allowing it to represent local geometry and surface orientation. During rendering, Gaussians that contribute to the same image region are sorted according to their depth in the current viewing direction and composited in front-to-back order according to their opacity. The color of a pixel can be simplified as

$$\mathbf{C} = \sum_{i=1}^N T_i \alpha_i \mathbf{c}_i, \quad T_i = \prod_{j=1}^{i-1} (1 - \alpha_j), \quad (2.11)$$

where $\mathbf{c}_i$ is the color of the i -th Gaussian, α_i is its opacity contribution to the pixel, and T_i is the accumulated transmittance from the Gaussians in front of it.

During training, 3DGS uses multi-view images as supervision and optimizes the Gaussians so that the rendered image $\hat{\mathbf{I}}$ matches the ground-truth image $\mathbf{I}$. A common loss combines an L1 term and a D-SSIM term:

$$\mathcal{L} = (1 - \lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\text{D-SSIM}}. \quad (2.12)$$

The L1 term encourages pixel-level color accuracy, while the D-SSIM term constrains local structural similarity. Here, $\mathcal{L}_{\text{D-SSIM}}$ denotes the dissimilarity derived from the Structural Similarity Index, commonly defined as $(1 - \text{SSIM})/2$. Since the rendering process is differentiable with respect to Gaussian position, scale, rotation, color, and opacity, the loss can be directly back-propagated to these explicit parameters.

Because SfM initialization is usually sparse, and because the distribution of initial points is mainly determined by image feature matching, the spatial density of the initial Gaussians is often insufficient for high-quality rendering. For example, texture-rich regions may contain many initial points, while low-texture surfaces, object boundaries, and fine structures may lack enough Gaussians. Some initialized points may also lie at incorrect positions or contribute very little during training. Therefore, 3DGS introduces adaptive density control during optimization, allowing the number and distribution of Gaussians to change according to the reconstruction error.

Density control mainly consists of densification and pruning. Densification is usually guided by the magnitude of the gradients of Gaussian projected positions in the image plane. If the rendering error around a Gaussian is large, its position, scale, or color receives stronger gradients, indicating that the current representation is insufficient in that region and that more Gaussians are needed to model local geometry or appearance. For Gaussians with small scale but large gradients, the system typically applies cloning, which creates a new Gaussian near the original one to increase local sampling density. For Gaussians with large scale and large gradients, the system applies splitting, where one overly large Gaussian is divided into several smaller Gaussians to represent boundaries, texture changes, or curved geometry more accurately.

Pruning is mainly based on opacity and effective contribution. If some Gaussians keep very low opacity or barely participate in the final image composition, they are treated as redundant or erroneous points and removed from the scene representation. In this way, 3DGS allocates more representational capacity to difficult regions while reducing Gaussians that contribute little to the rendered image, balancing reconstruction quality, model complexity, and rendering efficiency.

For the sports-scene reconstruction task in this thesis, 3DGS is useful for two main reasons. First, it is an explicit representation, which makes it convenient to combine with SMPL, Gaussian avatars, and scene Gaussians, and also enables different Gaussian sources to be rendered in a unified viewer. Second, it is well suited to free-viewpoint replay applications that require interactive viewpoint switching. Therefore, both the static environment reconstruction and the dynamic human avatar representation in this thesis use Gaussians as their core representation units.

2.4 Evaluation Metrics

2.4.1 Objective Metrics

L1 Loss

L1 loss measures the average absolute pixel difference between the reconstructed image $\hat{I}$ and the ground-truth image I :

$$\mathcal{L}_1 = \frac{1}{N} \sum_{i=1}^N |I_i - \hat{I}_i|. \quad (2.13)$$

Here, N is the total number of pixels. A lower L1 loss indicates that the reconstruction is closer to the ground truth at the pixel level. It is simple and stable, but it mainly evaluates per-pixel accuracy and does not fully reflect structural or perceptual quality.

Peak Signal-to-Noise Ratio (PSNR) [11]

PSNR is a commonly used pixel-level error metric for image reconstruction. It measures how close the reconstructed image is to the ground-truth image in terms of pixel intensity. The metric first computes the mean squared error (MSE) between the two images and then converts this error into a logarithmic signal-to-noise ratio:

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N (I_i - \hat{I}_i)^2, \quad \text{PSNR} = 10 \log_{10} \left(\frac{L^2}{\text{MSE}} \right) \quad (2.14)$$

where I and $\hat{I}$ denote the ground-truth and reconstructed images, respectively, N is the total number of pixels, and L is the maximum possible pixel value. A higher PSNR indicates a smaller pixel-wise reconstruction error. However, since PSNR mainly focuses on per-pixel differences, it is less sensitive to structural consistency and perceptual image quality.

Structural Similarity Index Measure (SSIM) [37]

SSIM is designed based on the observation that human perception is sensitive to structural information in images. It compares two images in terms of local luminance, contrast, and structure. Compared with pure pixel-wise error metrics, SSIM better reflects whether local textures, edges, and structural relationships are preserved:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (2.15)$$

where μ , σ^2 , and σ_{xy} denote the local mean, variance, and covariance, respectively, and C_1 and C_2 are constants used for numerical stability. SSIM is typically in the range of $[0, 1]$, and a value closer to 1 indicates that the reconstructed image is more structurally similar to the ground truth. Therefore, SSIM is suitable for evaluating the consistency of boundaries, textures, and overall visual structures.

Learned Perceptual Image Patch Similarity (LPIPS) [41]

LPIPS is a learned perceptual quality metric that measures the visual difference between two images from the perspective of human perception. Instead of directly comparing pixel values, LPIPS feeds images into a pretrained deep neural network and computes the distance between normalized feature activations across multiple layers:

$$\text{LPIPS}(x, y) = \sum_l \frac{1}{H_l W_l} \sum_{h, w} \|w_l \odot (\phi_l(x)_{h, w} - \phi_l(y)_{h, w})\|_2^2 \quad (2.16)$$

where $\phi_l(\cdot)$ denotes the feature representation at the l -th layer of the network, H_l and W_l are the spatial dimensions of the feature map, and w_l is a learned channel-wise weight. A lower LPIPS value indicates that the two images are more similar in deep feature space, which usually corresponds to higher perceptual similarity. Since LPIPS can capture perceptual differences in texture, semantics, and local details, it complements PSNR and SSIM, whose descriptions of perceptual quality are limited.

2.4.2 Subjective Image Quality

Although PSNR, SSIM, and LPIPS are widely used evaluation metrics in image reconstruction, they have inherent limitations in capturing visual differences at the level of human perception. These metrics quantify image quality through pixel-wise errors or local patch-level similarities. As a result, they are sensitive to low-level signals such as slight misalignment, boundary sharpness, and high-frequency texture, but they may not effectively reflect quality dimensions that are more important to human observers, such as the naturalness of skin texture, the coherence of details within local regions, and the absence of subtle structural artifacts. This discrepancy is particularly evident in Gaussian Avatar reconstruction: methods optimized for perceptual realism may only produce minor improvements in these metrics, or even slight numerical regressions, while still showing clearly improved visual quality.

To address this issue, this thesis introduces subjective image quality evaluation as a complement to objective metrics, directly collecting perceptual judgments from human observers. The experiment is conducted in the form of an AB test. Each question presents three images simultaneously: one reference image, which is the original input photograph and helps participants understand the real appearance of the scene, and two rendered outputs generated by two different configurations. Participants then choose whether the left image is better, the right image is better, or the two images are similar, corresponding to the “Similar / No Preference” option. For each participant, the order of questions and the left-right positions of the two rendered images are independently randomized to reduce the influence of ordering effects and positional bias. One example of the questions is shown in Fig 2.3.

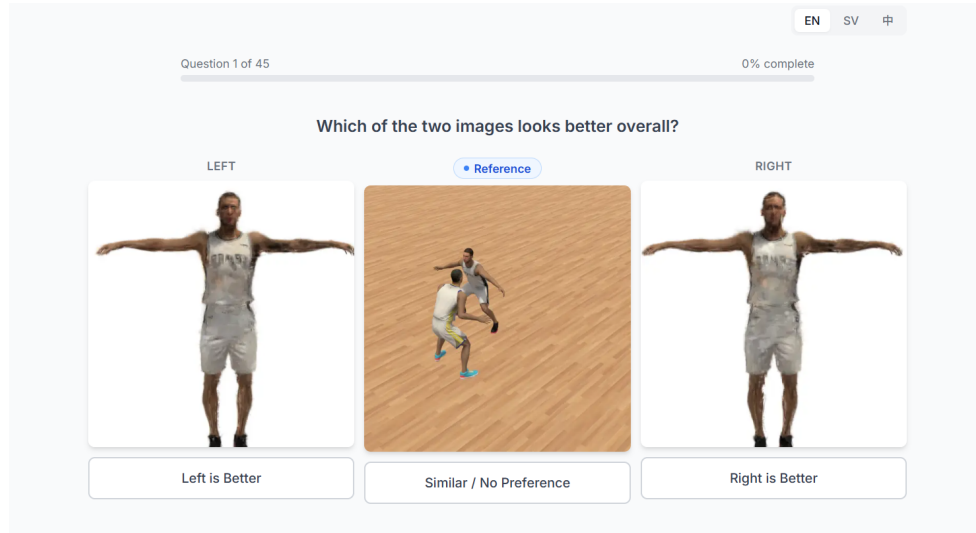


Figure 2.3 One example of the subjective test questions.

The evaluation focuses on the overall visual quality of the rendered outputs, including the naturalness and realism of the appearance, the fidelity of fine surface details, and the absence of visible artifacts, blur, or structural inconsistencies. The reference image is used only as background context for scene content and human appearance, and is not part of the pairwise comparison.

For the experimental results, this thesis uses three types of quantitative analyses. **Win Rate**: the proportion of times each configuration is selected as better among all comparisons in which it appears. When participants select “Similar / No Preference”, the comparison is not counted as a win for either side. This metric measures the overall perceptual preference for each configuration. **Pairwise Win Rate**: for each pair of configurations, wins and losses are counted separately, and the conditional win rate of each configuration is computed after excluding “Similar / No Preference” responses. This metric reflects the relative preference between any two configurations

under direct comparison. **Bradley-Terry (BT) Model** [2]: the BT model provides a probabilistic way to convert pairwise comparison results into a global ranking. Instead of only counting how often a configuration wins, it assumes that each configuration has an underlying perceptual strength parameter. For two configurations i and j , the probability that i is preferred over j is modeled as

$$P(i \succ j) = \frac{\theta_i}{\theta_i + \theta_j} \quad (2.17)$$

where θ_i and θ_j denote their latent strengths. A configuration with a larger strength value is therefore more likely to be preferred in pairwise comparisons. Based on the win-loss data from all pairwise comparisons, these strength parameters are estimated using a minorization-maximization (MM) iterative algorithm. The resulting BT scores provide globally comparable measures of relative perceptual quality, even when different configurations are not compared equally often. In addition, 95% confidence intervals are constructed using 500 bootstrap resampling runs to evaluate the stability of the ranking results.

3 Related Work

This chapter reviews and summarizes research related to 4D sports scene reconstruction, including traditional Free-Viewpoint Video methods, motion capture systems, human avatar reconstruction, and unified rendering frameworks that combine avatars with reconstructed environments.

3.1 Traditional Free-Viewpoint Video

Free-Viewpoint Video (FVV) is a technique in modern video media that provides viewers with an immersive experience by removing the constraints of fixed camera positions and allowing them to freely select viewpoints by 6 DoF (Degrees of Freedom) control. In sports scenarios such as football and basketball, traditional FVV research has generally developed along several complementary directions. The first direction centers on multi-camera geometric reconstruction and fusion. Early Virtualized Reality systems [15] established the basic paradigm of array-based capture, per-frame dense depth or visible-surface reconstruction, and novel-view rendering, demonstrating a complete path from concept to system implementation. The second direction focuses on depth-assisted image-based rendering (IBR) and view interpolation, where depth reprojection, layered representations, and video matting are used to mitigate occlusion holes and boundary artifacts, enabling interactive viewpoint switching even when the number of cameras is limited [44]. The third direction exploits structural priors and wide-baseline strategies for sports scenes. For example, methods may use the approximate planarity of the court, stable line markings, and fixed camera positions, together with techniques such as plane sweeping, to estimate usable depth and synthesize common sports effects such as orbiting or freeze-frame shots [8]. At the same time, sports broadcasting has also motivated workflows that are more closely aligned with production pipelines. For instance, MasterCam FVV uses a static high-resolution main camera as an anchor and robustly registers multiple camera views into a unified framework to support long-term alignment and free-viewpoint generation [1]. At the system level, streamable free-viewpoint video [6] represents a more mature and representative line of work, generating dynamic textured surfaces through dense RGB/IR multi-view stereo and fusion and compressing them into transmissible data streams. In addition, point-based primitives and dy-

dynamic point cloud compression have been explored to improve rendering efficiency and bitrate efficiency [39].

However, in real-world sports scenes, these traditional methods often face similar limitations. A primary limitation is that deployment and acquisition costs are high, while the achievable free-viewpoint range is constrained. Sports scenes require wide-area coverage, synchronization, and accurate calibration, but camera placement is restricted by physical location, occlusion, and operational cost, making it difficult to obtain dense sampling comparable to studio environments [33]. Another major challenge lies in the strong dependence of these methods on the quality of segmentation and depth estimation. Distant players occupy only small image regions, while rapid motion and frequent mutual occlusion can significantly amplify stereo matching errors and layered boundary errors, leading to holes, floating artifacts, geometric jitter, and temporal inconsistency. As a result, stable visual quality often depends on more intensive post-processing or denser camera coverage. A further constraint concerns the structural trade-off among photorealism, latency, and bitrate. Explicit geometry and volumetric video representations tend to impose high computational and transmission costs when high quality is pursued, whereas depth-based or layered IBR methods often produce visible synthesis artifacts under large viewpoint changes and complex occlusions. Consequently, subsequent research has increasingly turned to neural representations [24], using learned geometric and appearance priors to alleviate the ill-posed nature of sparse-camera and occluded settings, while improving viewpoint generalization and temporal stability for deployable, high-quality FVV in sports scenes. Meanwhile, with the emergence of explicitly renderable neural field representations such as Gaussian Splatting [16], related work has further sought more efficient solutions for key requirements such as real-time rendering and streaming.

3.2 Motion Capture

Motion capture is an important foundation for free-viewpoint video, human reconstruction, and digital human animation. Its core objective is to recover human body pose and motion trajectories in 3D space from observed data. Early high-precision motion capture systems mainly relied on optical marker-based setups. In such systems, reflective markers are attached to key body locations, and multiple synchronized cameras capture the marker positions from different views. The observed markers are then matched, triangulated, and fitted to a human skeleton to obtain 3D motion data. These systems usually provide high spatial accuracy and temporal stability, and have therefore long been regarded as a standard solution for 3D human motion measurement. However, they depend strongly on dedicated capture spaces, camera arrays, marker suits, marker placement, and manual calibration procedures,

which limits their flexibility in natural scenes and large-scale applications [7].

With the development of computer vision methods, motion capture has gradually evolved from professional systems that rely on physical markers toward markerless vision-based motion capture. In recent years, progress in 2D multi-person pose estimation has further accelerated this transition. For example, methods based on Part Affinity Fields [4] can detect real-time multi-person 2D keypoints from a single RGB image, providing reliable observations for subsequent multi-view 3D reconstruction and human model fitting. Building on this, multi-view 3D pose estimation methods further fuse image features or 2D detections from multiple camera views into a unified 3D space, enabling the recovery of multi-person 3D skeletal motion without markers [35]. Moreover, in response to the limited number of cameras available in practical deployments, recent work has begun to explore lightweight full-body multi-person motion capture under sparse multi-view camera setups, showing that robust estimates of body, hand, and face motion can still be obtained with relatively few camera views [42].

Overall, motion capture technology has clearly trended from dense, highly constrained, marker-based professional capture systems toward lightweight, sparse-view, markerless visual-perception frameworks. In this thesis, motion capture is treated as an upstream source of motion and geometric priors rather than as a proposed algorithmic contribution. Depending on the dataset, the pipeline uses available camera calibration, segmentation masks, skeleton data, pose parameters, or SMPL parameters provided by the dataset annotations or external preprocessing tools. These signals are used to drive Gaussian avatar reconstruction, scene composition, and novel-view rendering. The thesis therefore focuses on how to build a 4D reconstruction and rendering pipeline on top of such inputs, not on estimating the motion-capture data itself.

3.3 Human Avatars

Human avatars aim to recover digital human representations from images, videos, or multi-view observations. These representations should be reconstructable, renderable, and controllable by human pose. According to the underlying representation of human geometry and appearance, existing methods can be broadly divided into mesh-based avatars, neural implicit avatars, and the recently emerging Gaussian avatars. Since the system in this thesis needs to support multi-person reconstruction, pose-driven animation, and free-viewpoint rendering in sports scenes, this section mainly reviews mesh avatars and Gaussian avatars, which are most closely related to the proposed work.

3.3.1 Mesh Avatars

Mesh avatars usually use SMPL or its extended models as the underlying human topology, and recover clothing, hair, and local details through geometric refinement, texture prediction, or material modeling. Early single-image human reconstruction methods have already shown the ability to recover relatively complete clothed human geometry. For example, PIFuHD [29] uses multi-level pixel-aligned implicit functions to reconstruct detailed clothing wrinkles and human surfaces from high-resolution images. For sports scenarios, Zhu et al. proposed Reconstructing NBA Players [43], which introduces an animatable human mesh database derived from NBA2K game characters. This provides stronger priors for challenging basketball poses, clothing, and self-occlusion, enabling the recovery of poseable player meshes from single game images. Liao et al. further proposed High-Fidelity Clothed Avatar Reconstruction from a Single Image [21], using a coarse-to-fine strategy that combines learned canonical shape prediction with non-rigid optimization in posed space to reconstruct high-quality clothed avatars from a single image.

Compared with single-image input, multi-view video provides more stable geometric constraints and appearance observations. MeshAvatar [5] learns high-quality triangular-mesh human avatars from multi-view videos by combining an explicit triangular mesh extracted from an implicit SDF with a pose-conditioned material field, and introduces physically based rendering to support material decomposition, editing, and relighting. More recently, UP2YOU [3] focuses on more relaxed input conditions. It converts unstructured and uncontrolled personal photo collections into more regular orthogonal multi-view observations, and then quickly reconstructs textured human meshes. Overall, mesh avatars have clear advantages in explicit topology, editability, animation, and physical interaction, and they can be naturally integrated into traditional graphics pipelines. However, in scenarios involving fast motion, complex clothing, multi-person occlusion, and wide-range free-viewpoint rendering, mesh surfaces and textures often require additional detail compensation.

3.3.2 Gaussian Avatars

Before the emergence of Gaussian avatars, NeRF-based human avatars had already demonstrated the potential of neural representations for free-viewpoint human rendering. For example, HumanNeRF [38] represents the human body in canonical space and transforms it to each posed frame through a motion field, enabling novel-view rendering of dynamic humans from monocular videos. InstantAvatar [13] further reduces training time with acceleration structures, bringing monocular avatar reconstruction closer to interactive applications. However, NeRF-based methods usually rely on dense sampling and volume rendering, leading to high training and inference costs. This remains a clear limitation in multi-person scenes and real-time

applications. The emergence of 3D Gaussian Splatting provides a new direction for avatar representation: it represents geometry and appearance with explicit Gaussian primitives and enables real-time rendering through efficient rasterization, making it well suited to FVV systems that require interactive viewpoint switching.

Early Gaussian avatar methods mainly focused on how to connect Gaussians in canonical space with human pose deformation. 3DGS-Avatar [28] learns deformable 3D Gaussians from monocular videos, and drives canonical Gaussians into posed space through a non-rigid deformation network and LBS. It also introduces as-isometric-as-possible regularization to improve generalization to unseen poses. Animatable 3D Gaussian [22] combines skinned 3D Gaussians with a skeleton, uses a multi-head hash encoder to represent pose-dependent shape and appearance, and models dynamic shadows through time-dependent ambient occlusion, allowing the method to extend to multi-person scenes. Animatable and Relightable Gaussians [19] represents high-fidelity human appearance using canonical Gaussian maps combined with 2D CNNs, and further introduces physically based rendering to support relighting.

Another line of work explicitly combines Gaussians with mesh geometry. SplattingAvatar [32] embeds Gaussians onto a triangular mesh surface, using the mesh to handle low-frequency motion and surface deformation while using Gaussians to represent high-frequency appearance details. This combines traditional animation control with real-time rendering. RMAvatar [27] also uses mesh-embedded Gaussians and introduces a Gaussian rectification module to compensate for non-rigid clothing and detail deformations that are difficult to represent with LBS alone. Recent work has further focused on clothing disentanglement, sparse-view input, spatiotemporal consistency, and occlusion. STG-Avatar [12] combines spacetime Gaussians with LBS to enhance dynamic regions and non-rigid clothing details. AniGaussian [18] makes fuller use of SMPL pose priors through pose-guided deformation. AHOY [25] further addresses human reconstruction from in-the-wild YouTube videos with occlusions caused by objects or other people, and introduces video diffusion priors to complete invisible regions.

These works show that Gaussian avatars have gradually developed from single-person, monocular, and controlled settings toward multi-view, sparse-input, occluded, and multi-person scenarios. However, many methods still primarily focus on the reconstruction quality of a single avatar. In contrast, this thesis focuses on multiple dynamic humans in sports scenes and further explores training and optimization based on Gaussian avatar representations under this specific setting.

3.4 Unified Framework for Avatars and Environment

Unifying human avatars with environment representations is a key problem in free-viewpoint sports scene reconstruction. If only human avatars are reconstructed, novel-view rendering and pose-driven animation of the humans can be achieved, but the real ground plane, court, background structures, and occlusion relationships are missing. If the entire dynamic scene is reconstructed as a whole, the result can become unstable around fast-moving humans, mutual occlusions, and local detail changes. Therefore, recent work has started to model humans and scenes within a shared framework.

For sports scenes, GS-SFS [14] combines Gaussian Splatting with Shape-from-Silhouette for multi-person reconstruction in large-scale sports environments. It uses 3DGS to generate additional silhouette observations from more viewpoints, and then combines them with an improved SFS method to obtain higher-quality human meshes. This demonstrates the potential of Gaussian representations for strengthening multi-view geometric constraints in sports scene reconstruction. Dynamic Avatar-Scene Rendering from Human-centric Context [36] addresses dynamic rendering of human-scene interaction in real environments and proposes a Separate-then-Map strategy: the human and the scene are first modeled separately, and then a shared information mapping mechanism is used to reduce inconsistencies at their spatial and visual boundaries. AHA! [26] further combines 3DGS scenes with animatable human avatars. Through a Gaussian-aligned motion module and human-scene refinement, it enables humans to perform more natural motion and contact in diverse 3DGS scenes. A work closer to sports FVV is Echoes of the Coliseum [10], which proposes LiveSplats as an end-to-end 3D live streaming framework for sports events. It reconstructs dynamic 3D Gaussian scenes in real time from multi-view video streams and supports interactive-frame-rate viewing from novel viewpoints.

4 Method

This chapter introduces the proposed method for reconstructing multi-person sports scenes, including data preparation, the training framework for multi-person Gaussian Avatars, the reconstruction of scene Gaussians, and the design of a viewer for interactive replay of human motion within the reconstructed scene.

The overall framework is shown in Figure 4.1. The system first obtains skeleton sequences of multi-person motion from the motion capture system and fits them to frame-level SMPL parameters, which describe the pose, shape, and temporal motion of each person. At the same time, the scene images captured by multi-view cameras are processed by a data preparation pipeline, where segmentation is used to separate the static environment from the dynamic humans. The environment component is used to train environment Gaussians that represent the geometry and appearance of the scene, while the human component is combined with the corresponding SMPL parameters to train a Gaussian avatar for each human. After training, the environment Gaussians provide the static scene representation, and the Gaussian avatars are driven by the SMPL motion. These two components are then merged in the viewer, enabling interactive free-viewpoint replay in which multiple human motions are placed back into the reconstructed scene.

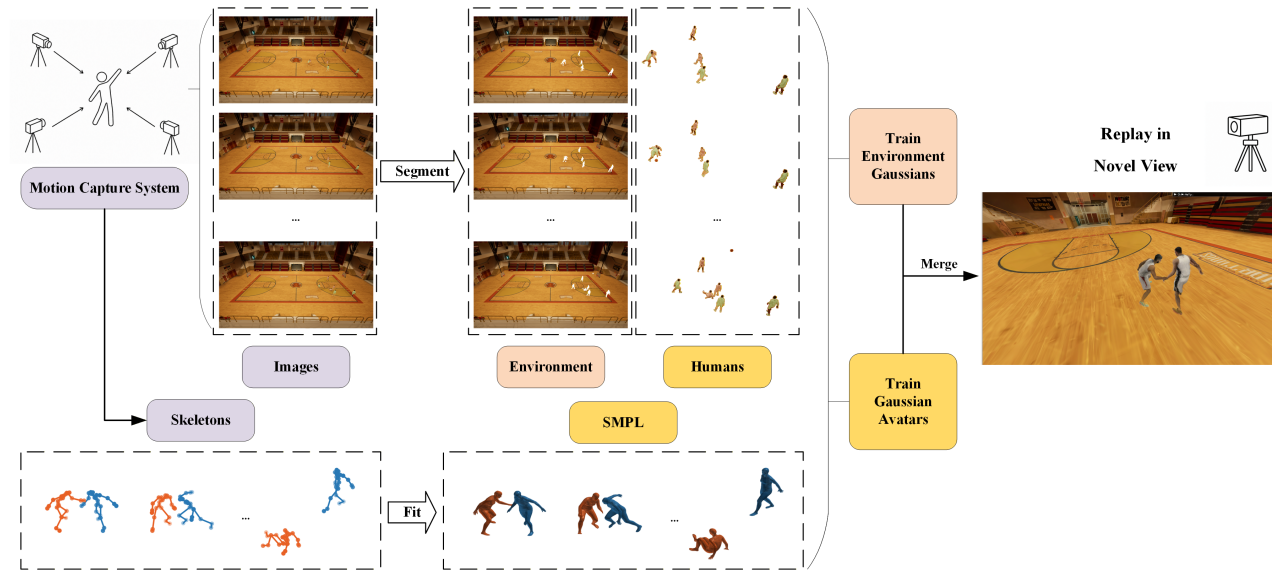


Figure 4.1 Overview of the framework for sports scene reconstruction and rendering. The motion capture system captures images from multi-view and generates skeletons of humans. The skeletons are fitted to SMPL parameters. The images are segmented to static environment and dynamic humans. The environment component is used to train environment Gaussians, while the human component is combined with the corresponding SMPL parameters to train a Gaussian avatar for each human. After training, these two components are then merged in the viewer, enabling interactive free-viewpoint replay.

All three-dimensional quantities in this work share a common right-handed world frame with Y pointing upward and units in metres, consistent with the SMPL coordinate convention. Table 4.1 summarises the five coordinate frames used throughout the pipeline. The SMPL global translation $\mathbf{t}$ positions each person’s root joint directly in the world frame, so avatar Gaussians and environment Gaussians are automatically co-located in the same space and can be merged in a single rasterisation pass without any additional coordinate transform. Camera extrinsics are stored as the world-to-camera transform $[\mathbf{R} \mid \mathbf{t}]$ (3×4) in `cameras.json`; the rasteriser receives this matrix in column-major (transposed) layout. GalaBasketball is generated with Unity, which uses a left-handed convention; the camera matrices are converted to the common right-handed frame by negating their first row.

Table 4.1 Coordinate frames used in the pipeline. Handedness: RH = right-handed. The camera frame defines orientation only and carries no metric unit.

Frame	Convention	Unit	Role in pipeline
World / mocap	RH, Y-up	m	common reference for all 3-D entities
SMPL canonical	RH, Y-up (= world)	m	rest-pose template; Gaussians initialised here
Camera (OpenCV)	RH; X right, Y down, Z forward	—	image projection; extrinsic = $W2C [\mathbf{R} \mid \mathbf{t}] 3 \times 4$
Environment Gaussian	= world	m	3DGS scene; trained and rendered in world frame
Viewer	= world	m	interactive rendering in the same world frame

4.1 Data Preparation

4.1.1 Datasets

Hi4D [40] is a dataset designed for the study of close human interaction. It contains sequences of 20 different pairs of people performing various physical contact actions, as shown in Figure 4.2. For each sequence, the dataset provides frame-by-frame images from seven video views, segmentation masks for each person, and camera calibration for each view. In addition, it provides frame-level SMPL parameters for the two people in each sequence, which removes the need to estimate SMPL parameters from the original videos or keypoint data. Although Hi4D is not a sports-scene dataset, it satisfies the requirements of multi-person and multi-view data, making it suitable for rapid implementation and model validation.

GalaBasketball [22] is a synthetic dataset generated using human body models with different shapes and appearances. It contains seven sequences, including four single-

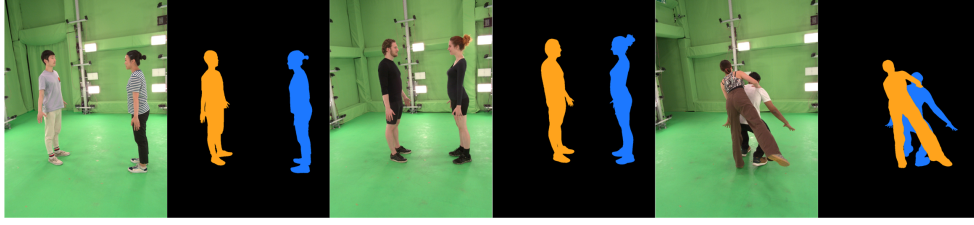


Figure 4.2 Example images and masks of Hi4D dataset.

person sequences, two two-person sequences, and one sequence with ten players, corresponding to the number of players in a standard basketball game, as shown in Figure 4.3. For each sequence, the dataset provides frame-by-frame images from seven video views and camera calibration for each view. It also provides frame-level pose data for each athlete in each sequence. Although the scenes in this dataset are relatively simple, it provides the basic conditions required for avatar training and testing.

BASKET-Multiview [10] is a dataset composed of common basketball scenarios synthesized in UE5. It provides two scenes and seven sequences, each containing 11 characters: ten athletes and one referee, as shown in Figure 4.4. For each sequence, the dataset provides frame-by-frame RGB images from 89 video views, together with detailed annotations for each view, including camera calibration, segmentation masks, depth maps, and surface normal maps. In addition, it provides animation-related parameters for each character. The dataset has a high level of scene realism, all sequences contain the number of players required for a standard game, and its detailed annotations and large number of viewpoints provide rich possibilities for model training and testing.



Figure 4.3 Example images of GalaBasketball dataset.



Figure 4.4 Example images and masks of BASKET-Multiview dataset.

4.1.2 SMPL Fitting

Because the raw data formats differ across datasets, corresponding SMPL parameter extraction and optimization pipelines are designed for BASKET-Multiview and GalaBasketball.

BASKET-Multiview provides per-frame human pose information in skeleton files. For each joint, the file records the parent joint index, the local offset, and the local rotation quaternion $\mathbf{q}_j = (x_j, y_j, z_j, w_j)$. Since these quaternions already represent local rotations with respect to the parent joint coordinate system, no additional relative-rotation computation is required. They can be directly converted to the axis-angle representation required by SMPL.

After extracting the pose parameters, the difference between the SMPL pelvis joint position under zero shape parameters and zero translation, J_0^{SMPL} , and the pelvis position recorded in the skeleton file, $\mathbf{p}_{\text{skel}}$, is used as the initial translation:

$$\mathbf{t}_{\text{init}} = \mathbf{p}_{\text{skel}} - J_0^{\text{SMPL}}(\beta = \mathbf{0}, \theta, \mathbf{t} = \mathbf{0}). \quad (4.1)$$

The pose parameters θ are then fixed, and the shape parameters β and translation $\mathbf{t}$ are optimized. Using the differentiability of the SMPL model, gradients of the loss with respect to β and $\mathbf{t}$ are computed by automatic differentiation and solved iteratively with Adam. The objective function is

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^N \|\mathbf{v}_i^{\text{SMPL}}(\beta, \theta, \mathbf{t}) - \mathbf{v}_i^{\text{OBJ}}\|^2 + \lambda_\beta \|\beta\|^2. \quad (4.2)$$

Here, $\mathbf{v}_i^{\text{SMPL}}$ denotes the i -th vertex of the SMPL mesh generated by the current

parameters, while $\mathbf{V}_i^{\text{OBJ}}$ denotes the corresponding i -th vertex of the target OBJ mesh. $N = 6890$ is the number of SMPL vertices, and $\lambda_\beta = 0.001$ is the weight of the L2 regularization term that prevents the shape parameters from drifting too far from the zero-mean prior. The optimization is performed in two stages. In the first stage, only $\mathbf{t}$ is optimized for 300 iterations with $lr = 0.01$ to align the global position of the model with the target mesh. In the second stage, β and $\mathbf{t}$ are jointly optimized for 1000 iterations with $lr = 0.005$ to further fit the individual body shape after positional alignment.

GalaBasketball stores the local rotations of 27 joints in each frame as Euler angles in YXZ order. Compared with BASKET-Multiview, fitting this dataset introduces two additional challenges. First, the Gala skeleton contains 27 joints, whereas SMPL contains only 24 joints, so a cross-skeleton mapping must be established. Second, the bone orientations of the two skeletons are inconsistent in the rest pose, which means that the Euler angles cannot be directly converted and assigned to the SMPL joints.

Pose extraction is performed in the world coordinate system through forward kinematics (FK). FK is applied separately to the T-pose and to the current frame, producing the world rotations $\mathbf{R}_j^w$ for both cases. The deformation matrix of each joint relative to the T-pose is then computed as

$$\mathbf{D}_j = \mathbf{R}_j^{w, \text{frame}} \cdot \left(\mathbf{R}_j^{w, \text{tpose}} \right)^\top. \quad (4.3)$$

To compensate for the difference between the rest-pose bone orientations of the two skeletons, a correction rotation $\mathbf{R}_j^{\text{corr}}$ is precomputed for each pair of corresponding bones. This correction is the minimum-arc rotation between the T-pose bone directions of the two skeletons and is applied to the deformation matrix during mapping:

$$\mathbf{R}_j^{w, \text{SMPL}} = \mathbf{D}_{g(j)} \cdot \mathbf{R}_j^{\text{corr}}, \quad (4.4)$$

where $g(j)$ denotes the corresponding Gala joint index. Finally, the world rotations are decomposed layer by layer into SMPL local rotations,

$$\mathbf{R}_j^{l, \text{SMPL}} = \left(\mathbf{R}_{\text{pa}(j)}^{w, \text{SMPL}} \right)^\top \cdot \mathbf{R}_j^{w, \text{SMPL}}. \quad (4.5)$$

where $\text{pa}(j)$ denotes the parent joint index of joint j in the SMPL kinematic tree. These local rotations are then converted to axis-angle representation to obtain the SMPL pose parameters.

For shape and translation optimization, the joint position error is used instead of the vertex error because the characters wear clothing and the SMPL mesh vertices cannot be directly aligned with the clothed surface mesh. Since the two skeletons have

inconsistent topology around the spine, with SMPL containing three spine joints and GalaBasketball containing four, lower weights w_j are assigned to joints such as the spine and clavicles. The objective function is

$$\mathcal{L} = \frac{\sum_j w_j \|\mathbf{J}_j^{\text{SMPL}}(\boldsymbol{\beta}, \boldsymbol{\theta}, \mathbf{t}) - \mathbf{J}_j^{\text{Gala}}\|^2}{\sum_j w_j} + \lambda_\beta \|\boldsymbol{\beta}\|^2. \quad (4.6)$$

The pose parameters are fixed, while $\boldsymbol{\beta}$ and $\mathbf{t}$ are treated as optimization variables. Gradients are computed by automatic differentiation, and Adam is used for two-stage optimization: first, $\mathbf{t}$ is optimized alone for 200 iterations; then $\boldsymbol{\beta}$ and $\mathbf{t}$ are jointly optimized for 500 iterations.

4.1.3 Preprocessed Data Structure

After preprocessing, each sequence is organized into a unified directory structure:

```
<sequence>/
|-- cameras.json           # camera parameters
|-- images/               # RGB frames by view
|   |-- CAM00/
|   |   |-- 0000.jpg
|   |   '-- ...
|   '-- ...
|-- masks/                # segmentation masks by view
|   |-- CAM00/
|   |   |-- 0000_mask.png
|   |   '-- ...
|   '-- color_map.json    # person ID to mask color
'-- SMPLpose/             # SMPL parameters by person
    |-- player00.npz      # betas(10,), body_pose(T,69),
    |                     # root_orient(T,3), transl(T,3)
    '-- ...
```

The file SMPLpose/playerXX.npz stores the shape parameters $\boldsymbol{\beta}$, which describe the sequence-independent body shape of each person, as well as the frame-wise pose parameters $\boldsymbol{\theta}$, represented by `body_pose` and `root_orient`, and the global translation $\mathbf{t}$, stored as `transl`. Here, T denotes the number of frames in the sequence. GalaBasketball does not provide segmentation masks, so the masks are generated using Segment Anything Model (SAM) [17]. Different humans have different mask colors,

and the color is consistent across all frames and views. The image frames, masks, and SMPL pose files are assumed to be temporally synchronized by the dataset. For a given frame index, all camera views, segmentation masks, and skeleton/SMPL parameters correspond to the same time instant.

4.2 Human Avatars Reconstruction

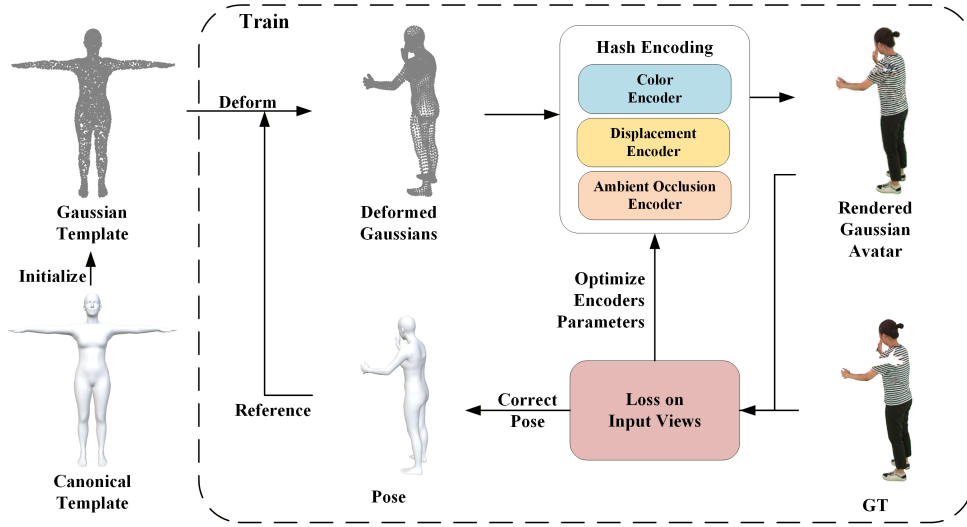


Figure 4.5 Overview of the Gaussian Avatars training pipeline. 3D Gaussians are initialized on the vertices of the SMPL canonical mesh, forming a Gaussian Template. During training, for each input view, the Gaussians are deformed into pose space using SMPL pose as reference, and all the attributes are predicted by 3 hash encoders. Then avatar images are rendered by a rasterizer and used to compute the loss based on GT images. Then the loss is back-propagated for optimization.

4.2.1 Pipeline Overview

The overall training pipeline is shown in Figure 4.5. A set of 3D Gaussians is first initialized on the vertices of the SMPL canonical mesh, forming a Gaussian Template. During training, for each frame from an input view, the SMPL pose parameters of the current frame are used as reference. Linear blend skinning (LBS) is applied to deform the Gaussians from the canonical space to the posed world space, producing the deformed Gaussians.

At the same time, a set of encoders based on multi-resolution hash-grid encoding predicts three types of attributes for each Gaussian. The color encoder outputs view-dependent color represented by spherical harmonic coefficients, the displacement encoder predicts positional offsets in the canonical space, and the ambient occlusion

(AO) encoder predicts ambient occlusion coefficients. These predicted attributes, together with the deformed Gaussian positions, are sent to a differentiable rasterizer to render the Gaussian avatar image from the current viewpoint. The reconstruction error between the rendered result and the ground-truth image, denoted as the loss on input views, is back-propagated along two paths. One path optimizes the parameters of the three encoders to improve the appearance and geometric representation of the Gaussians. The other path corrects the input SMPL pose parameters to compensate for errors in the initial pose estimates. These two gradient paths form a closed training loop, allowing the model to progressively fit the appearance and motion of the human subject with a fixed number of Gaussians.

The following subsections describe the main components of this avatar reconstruction framework in detail. Section 4.2.2 introduces the multi-resolution hash-grid encoders used to predict Gaussian attributes. Section 4.2.3 explains how the canonical Gaussian template is initialized from SMPL and deformed to the posed world space through LBS. Section 4.2.4 presents the fixed-size pruning and duplication strategy used to adapt the Gaussian distribution without changing the total number of Gaussian slots. Finally, Section 4.2.5 describes the optimization details, including pose correction, non-uniform Gaussian initialization, two-stage training, and the photometric loss design.

4.2.2 Multi-resolution Hash Grid Encoder

As shown in Figure 4.6, the three encoders share the same structural backbone, whose core component is a multi-resolution hash grid. The normalized 3D coordinate (x, y, z) of each Gaussian in canonical space is used as input. The AO encoder additionally concatenates a time encoding t , while the displacement encoder additionally takes pose information as input.

The hash grid consists of L independent resolution levels, which cover spatial details from coarse to fine scales. The grid resolution of the l -th level is defined as

$$N_l = \lfloor N_{\min} \cdot s^{l-1} \rfloor, \quad s = 1.5. \quad (4.7)$$

Here, $N_{\min} = 4$ is the resolution of the coarsest level, and the resolution increases with a common ratio of $s = 1.5$. Each level maintains an independent hash table with $2^{T_{\text{hash}}}$ entries, where T_{hash} is the base-2 logarithm of the hash table size. In this work, $T_{\text{hash}} = 14$, so each table contains 16384 entries. Each entry stores a $D_{\text{feat}} = 4$ dimensional learnable feature vector. Given a query coordinate, the voxel containing the point is first located at the l -th resolution level. The eight corner vertices enclosing the point are then found, and their integer coordinates are mapped to hash table indices through a spatial hash function to retrieve the corresponding features. Trilinear

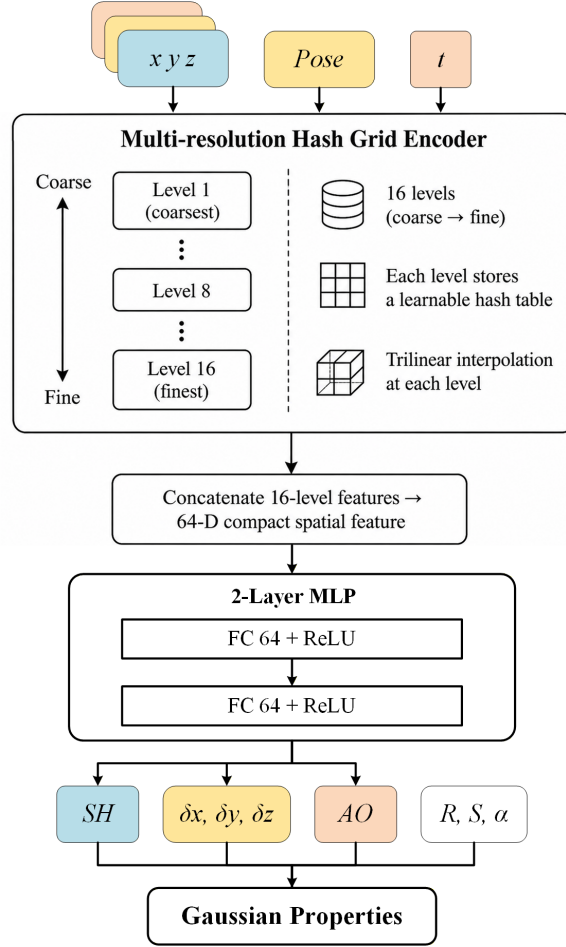


Figure 4.6 Overview of the Multi-resolution Hash Grid Encoder. Color, displacement and AO encoders share the same backbone structure and use 3D coordinates as input. Color encoder outputs spherical harmonic coefficients. Displacement encoder additionally uses pose information and outputs a 3D positional offset. AO encoder takes time as input and outputs ambient occlusion coefficients. These attributes together with rotation, scale and opacity, are then sent to the rasterizer for rendering.

interpolation over the eight corner features produces the D_{feat} dimensional output of this level. The features from all L levels are concatenated into a compact spatial feature vector of dimension $L \times D_{\text{feat}}$, which is then passed to the following MLP decoder.

Encoder Configuration. The three encoders use $L = 16$ levels, resulting in a feature dimension of $16 \times 4 = 64$, which is decoded by a two-layer MLP. The three encoders produce different Gaussian attributes: the color encoder outputs spherical harmonic coefficients for view-dependent appearance, the displacement encoder outputs a 3D positional offset $(\delta x, \delta y, \delta z)$ so that Gaussians can move away from the SMPL

surface to fit non-rigid details such as clothing, and the AO encoder outputs pose-dependent ambient occlusion coefficients to model illumination changes caused by occlusion. For multi-person scenes, each encoder maintains independent hash-grid weights for each subject, enabling parallel training of different body shapes and appearances.

4.2.3 3D Gaussian Deformation

Before training, the canonical template is initialized from the SMPL model. Specifically, the standard SMPL template vertices are first deformed according to the subject-specific shape parameters β , producing an individualized canonical mesh with 6890 vertices. A pre-set number of 3D Gaussians are placed on each vertex as their initial positions. The initial scale of each Gaussian is determined by the distance to its K -nearest neighbors, the opacity is uniformly initialized to 0.2, and the rotation is initialized as the identity quaternion. In canonical space, each Gaussian carries three groups of learnable attributes: color represented by spherical harmonic (SH) coefficients, canonical displacement $\Delta\mathbf{x}$, and an AO coefficient. These attributes are dynamically predicted from the canonical coordinates of each Gaussian by the hash grid encoders, rather than being stored as independent parameters.

At each training step, the displacement predicted by the Displacement Encoder is first added to the canonical template position, producing the displaced canonical coordinate that will later be mapped to world space by the LBS transformation matrix:

$$\mathbf{v}_{\text{disp}} = \mathbf{v}_{\text{template}} + \Delta\mathbf{x}. \quad (4.8)$$

Meanwhile, the current-frame SMPL pose parameters, including body pose θ , global orientation θ_0 , and global translation $\mathbf{t}$, are used to drive Linear Blend Skinning. For each Gaussian, a blended transformation matrix $\mathbf{T}_i$ is computed as the weighted combination of the joint transformations:

$$\mathbf{T}_i = \sum_{j=1}^{24} w_{ij} \mathbf{A}_j(\theta, \mathbf{t}). \quad (4.9)$$

Here, w_{ij} denotes the learnable skinning weight, and $\mathbf{A}_j$ is the global rigid transformation of the j -th joint under the current pose. Applying $\mathbf{T}_i \in \mathbb{R}^{3 \times 4}$ to the displaced canonical coordinate $\mathbf{v}_{\text{disp},i}$ gives the world-space position of the Gaussian:

$$\mathbf{p}_i = \mathbf{T}_i [\mathbf{v}_{\text{disp},i}; 1]^\top. \quad (4.10)$$

The linear part of the same transformation is also applied to the Gaussian rotation and covariance, ensuring that its orientation remains consistent with local skeletal

deformation. Finally, the world-space attributes of all Gaussians are sent to the differentiable rasterizer for rendering.

4.2.4 Pruning and Duplication with a Fixed Number of Gaussians

Unlike the original 3DGS [16] introduced in Section 2.3, which controls adaptive density by dynamically increasing and decreasing the number of Gaussians, the proposed method uses a fixed-size Gaussian template initialized from the SMPL vertices. Each SMPL vertex is associated with a predefined group of Gaussian slots initialized around that vertex, and these slots retain fixed indices during training. The skinning-weight tensor $\mathbf{W} \in \mathbb{R}^{N_{\text{person}} \times N_{\text{gauss}} \times 24}$ is stored according to these Gaussian indices, and multiple subjects share the same template structure. Therefore, the total number of Gaussian slots cannot be changed during training. To address this constraint, a fixed-size adaptive density control mechanism based on slot reallocation is designed.

Opacity Decay and Slot Identification. During training, the opacity of Gaussians that contribute little to the scene naturally decays toward zero under gradient optimization. Every fixed number of densification steps, denoted by I_{densify} , the maximum opacity of each Gaussian over all views and all subjects is computed as

$$\alpha_{\max,i} = \max_r \sigma(o_{r,i}). \quad (4.11)$$

Gaussians satisfying $\alpha_{\max,i} < \alpha_{\min}$ are regarded as inactive points, where $\alpha_{\min} = 0.05$ in this work. Their parameter slots are marked as reusable free slots.

Selection of High-gradient Candidates. At the same time, the norm of the loss gradient with respect to the projected 2D position of each Gaussian, $\|\nabla_{\mathbf{u}_{2D}} \mathcal{L}\|$, is used to measure the degree of under-fitting in the image region covered by that point. When the region covered by a Gaussian contains large reconstruction errors, its projected coordinate receives gradient forces from multiple neighboring pixels. If these gradients are mostly consistent in direction, their resultant norm becomes large, indicating that the Gaussian is trying to cover a spatial region beyond its representational capacity and that local geometry should be refined through densification. For multi-person scenes, the maximum gradient over all subject instances, indexed by r , is used as the aggregated gradient signal of the i -th Gaussian:

$$g_i = \max_r \|\nabla_{\mathbf{u}_{2D,r,i}} \mathcal{L}\|. \quad (4.12)$$

A Gaussian is selected as a densification candidate if it is still active and satisfies $g_i > \tau_g$, where $\tau_g = 2 \times 10^{-4}$. The selected candidates are then divided according to their scale:

$$s_{\max,i} = \max_r \max_k e^{\ell_{r,i,k}}. \quad (4.13)$$

Here, $\ell_{r,i,k}$ denotes the log-scale parameter along axis k . Large-scale Gaussians satisfying $s_{\max,i} \geq \tau_s$ are assigned to the split candidate set $\mathcal{S}$, while smaller Gaussians are assigned to the clone candidate set $\mathcal{C}$. In this work, $\tau_s = 0.01$ m. Both candidate sets are sorted in descending order by gradient magnitude.

Slot Assignment and Parameter Update. Free slots are assigned with priority given to split candidates. The highest-gradient points in the split candidate set are assigned first, and any remaining free slots are then assigned to the clone candidate set. If the number of free slots is N_{free} , the actual numbers of split and clone operations are

$$n_{\text{split}} = \min(|\mathcal{S}|, N_{\text{free}}), \quad n_{\text{clone}} = \min(|\mathcal{C}|, N_{\text{free}} - n_{\text{split}}). \quad (4.14)$$

For a split operation, an offset is sampled in the local rotation frame of the source Gaussian according to its scale parameters and then rotated back to world space:

$$\Delta \mathbf{x}_{\text{split}} \sim \mathcal{N}(\mathbf{0}, \text{diag}(\mathbf{s}_i^2)), \quad \delta_{\text{split}} = \mathbf{R}_i \Delta \mathbf{x}_{\text{split}}. \quad (4.15)$$

The canonical template coordinate of the target slot is initialized as $\mathbf{v}_{\text{src}} + \delta_{\text{split}}$, and the skinning weights are copied from the source Gaussian. The scale parameters of both the source and target Gaussians are reduced by $\ln 1.6$, which is equivalent to shrinking the scale by a factor of $1/1.6$, while opacity and rotation are kept unchanged.

For a clone operation, all attributes of the target slot, including coordinate, skinning weights, scale, rotation, and opacity, are directly copied from the source Gaussian. A random perturbation with standard deviation determined by the source scale is then added to the canonical coordinate to break the gradient symmetry caused by exact overlap.

After the parameter update, the first- and second-order Adam moments of the opacity and scale parameters in the target slots are reset, preventing historical gradient accumulation from interfering with the newly generated points.

4.2.5 Optimization Details

Pose Correction.

Because the SMPL pose parameters estimated from the raw videos inevitably contain noise, inaccurate poses can cause the Gaussian positions after LBS deformation to misalign with the true human boundary, reducing reconstruction quality. To address

this issue, a learnable pose correction is introduced for each frame to compensate for errors in the initial pose estimates. Specifically, for every frame in the training sequence, a set of incremental parameters $(\Delta\theta, \Delta\theta_0, \Delta\mathbf{t})$ is learned. During each forward pass, these increments are added to the input SMPL parameters before LBS is performed:

$$\theta' = \theta + \Delta\theta, \quad \theta'_0 = \theta_0 + \Delta\theta_0, \quad \mathbf{t}' = \mathbf{t} + \Delta\mathbf{t}. \quad (4.16)$$

All correction terms are initialized to zero, ensuring that the training behavior at the beginning is identical to the case without pose correction. To prevent the corrections from drifting too far from the original pose, an L2 regularization term is added to the loss:

$$\mathcal{L}_{\text{pose}} = \|\Delta\theta\|^2 + \|\Delta\theta_0\|^2 + \|\Delta\mathbf{t}\|^2. \quad (4.17)$$

Gaussian Initialization.

To match the Gaussian density with the detail requirements of different body parts, a non-uniform Gaussian initialization strategy is used. The dominant joint of each SMPL vertex is determined by the argmax of its SMPL skinning weights. Vertices are then divided into three detail levels according to their dominant joints and assigned different duplication counts. High-detail regions, including the head joint 15, both hand joints 22 and 23, and both foot joints 10 and 11, receive 30 Gaussians per vertex. Medium-detail regions, including the pelvis joint 0, spine joints 3, 6, and 9, neck joint 12, clavicle joints 13 and 14, ankle joints 7 and 8, and wrist joints 20 and 21, receive 20 Gaussians per vertex. Low-detail regions, including hip joints 1 and 2, knee joints 4 and 5, shoulder joints 16 and 17, and elbow joints 18 and 19, receive 12 Gaussians per vertex. Since the SMPL template contains 6890 vertices, this strategy initializes more than one hundred thousand Gaussians for each subject. Regions with complex appearance changes, such as the face and hands, are covered more densely, while unnecessary computation is avoided in large and relatively smooth regions such as the torso.

Two-stage Training.

In multi-person scenes, if all Gaussians from all subjects are mixed and jointly optimized from the beginning, occlusions and soft boundaries between different people can interfere with one another, making it difficult for the Gaussians of each subject to converge to the correct region during early training. Therefore, a two-stage training strategy is adopted: alternating training followed by joint training, denoted as `alternating_then_joint`.

In the first stage, alternating training is performed for the first E_{alt} epochs. In each iteration, only the Gaussians of one subject are optimized, while the Gaussians of the other subjects do not participate in gradient updates for that step. During rendering, the segmentation mask of the current subject is used as the ground-truth mask, and the loss is computed only within that subject’s region, excluding interference from the background and other people. At the same time, a per-player crop strategy is enabled. Cropped camera parameters are precomputed for each subject, so the rendering resolution is concentrated on the corresponding bounding-box region, improving both optimization efficiency and accuracy.

In the second stage, joint training is performed. After alternating training, the Gaussians of each subject have independently converged to their own appearance and geometry. The training then switches to joint mode, where the Gaussians of all subjects participate together in each forward pass and loss computation. Under full multi-person rendering supervision, the model is further refined to improve the overall consistency of occluded regions in multi-person scenes.

Loss Function.

The photometric loss is a weighted combination of the L1 loss and D-SSIM:

$$\mathcal{L}_{\text{photo}} = (1 - \lambda_{\text{dssim}}) \mathcal{L}_1 + \lambda_{\text{dssim}} (1 - \text{SSIM}). \quad (4.18)$$

In the original 3DGS [16] formulation, $\lambda_{\text{dssim}} = 0.2$, so the L1 term dominates and SSIM acts as a secondary term. In this work, the weight is changed to $\lambda_{\text{dssim}} = 0.8$, making the SSIM term dominant. During avatar training, Gaussians are attached to the SMPL surface, and their spatial distribution is strongly constrained. Pixel-wise L1 loss is therefore overly sensitive to Gaussian position noise, whereas SSIM measures image quality from the perspective of local structural similarity and focuses more on the global consistency of textures and boundaries. With a limited number of Gaussians, this encourages the model to prioritize perceptually meaningful structural details rather than purely pixel-wise fitting.

4.3 Environment Reconstruction

The goal of environment reconstruction is to extract static background information from multi-view images and train environment Gaussians that can render the scene containing the humans from arbitrary viewpoints in real time. This provides a high-quality scene representation for the later merged rendering with human avatars. As shown in Figure 4.7, the overall pipeline consists of three stages: data preprocessing, sparse point cloud reconstruction, and Gaussian Splatting optimization.

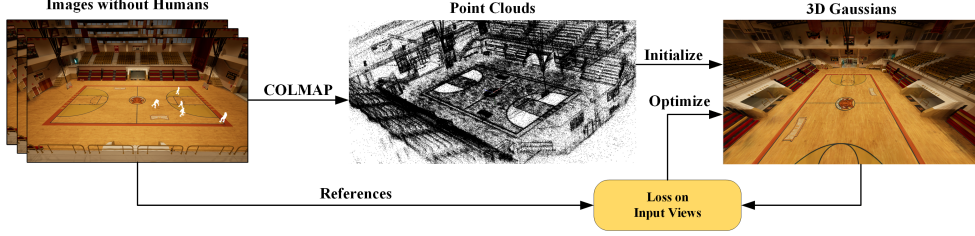


Figure 4.7 Overview of the Environment Reconstruction. The background-only images are used for point cloud extraction. During training, the loss is only computed on the background part of images.

Since the training images contain both dynamic humans and the static background, directly using the original multi-view images for 3D reconstruction would introduce inconsistencies across camera views and frames. The appearance variation in human regions would interfere with feature matching and with the appearance fitting of the Gaussians. To avoid this problem, the obtained human segmentation mask $\mathbf{M} \in \{0, 1\}^{H \times W}$ is first used to mask out the human regions in each training image, producing human-free images that retain only the background content. These images are then fed into the COLMAP [30] [31] pipeline with the dataset-provided camera intrinsics and extrinsics. In this setting, COLMAP is not used to estimate new camera poses. Instead, feature extraction and matching are used to establish image correspondences under the known camera calibration, and triangulation is then performed with the provided camera poses to generate a sparse 3D point cloud. The resulting point cloud is used as the spatial initialization for the 3D Gaussians.

The Gaussian representation of the scene is optimized end-to-end under the 3DGS [16] framework. Given the camera parameters of a training view, the current set of Gaussians is rendered with differentiable rasterization to obtain the rendered image $\hat{\mathbf{I}}$, which is compared with the ground-truth image $\mathbf{I}_{\text{gt}}$ using a reconstruction loss.

In standard 3DGS, the loss is computed over all pixels in the full image. However, in scenes containing humans, the human regions in the training images are not intrinsic scene content. If these pixels participate in the loss computation, the Gaussians are forced to fit an incorrect appearance for the background that is actually occluded by humans, degrading the quality of scene reconstruction. Therefore, a segmentation-mask-based loss masking strategy is introduced. Before computing the loss, the rendered image is modified using the mask $\mathbf{M}$:

$$\tilde{\mathbf{I}} = \mathbf{M} \odot \hat{\mathbf{I}} + (1 - \mathbf{M}) \odot \mathbf{I}_{\text{gt}}. \quad (4.19)$$

Here, $\odot$ denotes pixel-wise multiplication, $\mathbf{M}(p) = 1$ corresponds to background pixels, and $\mathbf{M}(p) = 0$ corresponds to human pixels. In the modified image $\tilde{\mathbf{I}}$, the human regions are identical to the ground-truth image, so their contribution to the

loss is always zero. The loss computation is therefore effectively restricted to the background regions:

$$\mathcal{L}_{\text{env}} = (1 - \lambda_{\text{env}}) \mathcal{L}_1(\tilde{\mathbf{I}}, \mathbf{I}_{\text{gt}}) + \lambda_{\text{env}} \mathcal{L}_{\text{D-SSIM}}(\tilde{\mathbf{I}}, \mathbf{I}_{\text{gt}}). \quad (4.20)$$

With this mask constraint, the gradient signal is provided only by reconstruction errors in the background regions. Throughout optimization, the positions, shapes, and appearance attributes of the Gaussians are therefore guided toward accurate reconstruction of the static scene background, avoiding interference from human occlusions during scene Gaussian fitting.

4.4 Interactive Viewer

To enable unified visualization of the reconstructed environment and the reconstructed human avatars, a standalone interactive viewer is implemented with a unified rendering framework for heterogeneous Gaussian data. The goal of this framework is not to render the static scene Gaussians and dynamic human Gaussians separately and then composite the rendered images. Instead, both types of Gaussians are first converted into a consistent runtime representation and are then rendered within the same rasterization process. Specifically, the background scene is represented by the static Gaussian point cloud produced by standard 3DGS, whose attributes mainly include center position, anisotropic scale, rotation, opacity, and spherical harmonic coefficients. The human avatars are generated frame by frame by the Gaussian avatar models under given pose conditions. In addition to position, scale, rotation, opacity, and color coefficients, their outputs also include auxiliary quantities related to skeletal deformation. To remove the difference in data organization between the two representations, the viewer wraps both background Gaussians and avatar Gaussians into a shared data structure, abstracting each Gaussian as a combination of position $\mathbf{x}$, scale $\mathbf{s}$, rotation $\mathbf{q}$, opacity α , spherical harmonic color coefficients $\mathbf{c}$, and additional transformation parameters. In this way, both the static background and the dynamic humans can enter the same rendering pipeline.

Based on this representation, let the set of scene background Gaussians be $\mathcal{G}^{\text{env}}$, and let the Gaussian set of the k -th avatar at time t be $\mathcal{G}_{k,t}^{\text{ava}}$. The full set of Gaussians rendered in the current frame is then

$$\mathcal{G}_t = \mathcal{G}^{\text{env}} \cup \bigcup_{k=1}^K \mathcal{G}_{k,t}^{\text{ava}}. \quad (4.21)$$

In implementation, the viewer does not call the renderer separately for different objects. Instead, it first concatenates the Gaussian attributes of all visible objects along

the tensor dimension to form a unified batched input, and then performs Gaussian rasterization once. For the static background, this set remains unchanged over time. For dynamic avatars, the Gaussian parameters are recomputed for each frame according to the current pose. The single-rasterization strategy ensures that the background and all avatars share the same depth sorting and opacity accumulation process. As a result, occlusion relationships among different objects can be handled in a physically consistent framework, avoiding occlusion errors, boundary discontinuities, and inconsistent semi-transparent regions that may arise from rendering objects separately and compositing them at the image level.

For animation driving of the human avatars, the viewer adopts a two-input design consisting of a model parameter file and a pose sequence file. Each avatar first loads its trained checkpoint to recover the canonical-space Gaussian representation and deformation network. It then loads the corresponding pose sequence, where each frame contains body joint rotations θ_t , global orientation $\theta_{0,t}$, and root translation $\mathbf{t}_t$. At runtime, the viewer reads the pose corresponding to the current playback frame index and feeds it, together with the time encoding, into the avatar model to obtain the human Gaussian distribution for that frame:

$$\mathcal{G}_{k,t}^{\text{ava}} = F_k(\theta_{k,t}, \theta_{0,k,t}, \mathbf{t}_{k,t}, \gamma(t)). \quad (4.22)$$

Here, $F_k(\cdot)$ denotes the dynamic Gaussian generation function of the k -th avatar, and $\gamma(t)$ denotes the time encoding. This process effectively couples pose driving with Gaussian deformation. The SMPL pose parameters control the skeletal and surface deformation, while the model further outputs the position, scale, rotation, and color attributes of each Gaussian under the given pose, allowing the human body to be displayed as a continuous animation. In multi-avatar scenes, each object is evaluated independently according to its own pose sequence and is then merged into the unified Gaussian set $\mathcal{G}_t$ for joint rendering. In addition, the viewer supports applying rigid translation offsets on top of the pose-driven results, making it possible to interactively adjust the layout of multiple avatars in the scene.

With this design, the implemented viewer has three main properties. First, it unifies static background Gaussians and dynamic human Gaussians at the representation level, allowing both to share the same rendering interface. Second, at the temporal level, it drives avatar animation through frame-wise pose evaluation and supports independent playback control for multiple people. Finally, at the image formation level, it preserves correct occlusion relationships in complex scenes through single-pass rasterization, providing a unified and stable foundation for interactive browsing, follow-camera control, and offline video export.

5 Results

This chapter provides a comprehensive evaluation of the proposed human avatar reconstruction method, including comparisons with existing approaches, ablation studies, and the results of animation. Then, it presents qualitative results for environment reconstruction and sports scene replay.

5.1 Human Avatars Reconstruction

5.1.1 Experiment Settings

All experiments in this thesis are conducted on a machine equipped with an Intel i9-13950HX CPU and an NVIDIA RTX 4090 GPU. The per-sequence dataset configurations are summarised in Table 5.1; all sequences involve two persons. Training hyperparameters shared across all sequences are listed in Table 5.2.

Table 5.1 Per-sequence dataset configuration. GalaBasketball training frames are sampled with a stride of 2 from the full sequence; Hi4D uses every frame.

Sequence	Dataset	Train Cameras	Test Camera	Train Frames	Resolution
dance00	Hi4D	CAM00–06 (7)	CAM07	110	470×640
hug17	Hi4D	CAM00–06 (7)	CAM07	90	470×640
piggy27	Hi4D	CAM00–06 (7)	CAM07	110	470×640
block	GalaBasketball	CAM0–5 (6)	CAM6	40	512×512
boxout	GalaBasketball	CAM0–5 (6)	CAM6	71	512×512

5.1.2 Comparison

Three methods are included for comparison. Animatable 3D Gaussian [22] is treated as the main and closest baseline, because it natively supports multi-view and multi-person Gaussian avatar training and is therefore the most directly comparable method. In addition, 3DGS-Avatar [28] and Animatable Gaussians [20] are included as adapted reference methods. These methods were originally designed mainly for single-avatar settings, so for multi-person evaluation they are trained separately for visible subjects and then composed in the same scene during testing. Their results should therefore be interpreted as reference comparisons rather than fully equivalent baselines.

Table 5.2 Training hyperparameters shared across all sequences.

Parameter	Value
GPU	NVIDIA RTX 4090
Optimizer	Adam
Total epochs	20
Training strategy	alternating (10 ep.) $\rightarrow$ joint (10 ep.)
λ_{dssim}	0.8
Pose correction LR	1×10^{-4}
Opacity LR	5×10^{-2}
Scaling LR	5×10^{-3}
SH LR	2.5×10^{-3}
Displacement encoder LR	1×10^{-3}
Densification interval	500 steps
Densification epochs	2–15
Gradient threshold	2×10^{-4}
Min. opacity threshold	0.05

For 3DGS-Avatar, one training view is used to train separate avatars for the visible subjects. Animatable Gaussians uses the same training views as the proposed method and is trained for 50,000 epochs, but it is still evaluated through the same separate-avatar composition strategy. Animatable 3D Gaussian can be directly applied to the processed datasets because it supports multi-view and multi-person training.

As shown in Table 5.3, the proposed method achieves clear improvements over previous methods across most metrics in multi-person scenes. Averaged over the five test sequences, the proposed method obtains a PSNR of 31.84 dB, an SSIM of 0.9703, and an LPIPS of 0.0241. Compared with the closest baseline, Animatable 3D Gaussian, the average PSNR improves by about 1.40 dB, while LPIPS decreases from 0.0290 to 0.0241, corresponding to an approximately 17% reduction in perceptual error. The improvement is more pronounced on the two GalaBasketball sequences: PSNR increases by about 2.89 dB on gala-block and 2.78 dB on gala-boxout, while LPIPS decreases by about 0.0040 and 0.0042, respectively. This indicates that the proposed method fits human boundaries and appearance details more stably in the synthetic basketball scenes. Compared with 3DGS-Avatar and Animatable Gaussians, the proposed method improves average PSNR by about 9.71 dB and 8.91 dB, and reduces average LPIPS by about 66% and 54%, respectively, showing that directly extending single-avatar methods to multi-person scenes leads to substantial quality degradation.

The qualitative results in Figure 5.1 further illustrate these differences. Since 3DGS-Avatar uses only a single training view, many Gaussians in the reconstructed avatars become nearly transparent in under-observed body regions. Animatable Gaussians

Table 5.3 Quantitative comparison with existing methods on selected GalaBasketball and Hi4D sequences. Best results are shown in bold, and second-best results are underlined. Time for SMPL fitting is not included in the training time.

Sequence	Method	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	Training Time $\downarrow$
gala-block	3DGS-Avatar	32.52	0.9858	0.0294	1532 s
	Animatable Gaussians	34.03	0.9846	0.0150	17h04m
	Animatable 3D Gaussian	<u>36.48</u>	<u>0.9896</u>	<u>0.0092</u>	453 s
	Ours	39.37	0.9938	0.0052	683 s
gala-boxout	3DGS-Avatar	31.92	0.9851	0.0236	1485 s
	Animatable Gaussians	34.20	0.9896	0.0126	17h25m
	Animatable 3D Gaussian	<u>36.55</u>	<u>0.9903</u>	<u>0.0075</u>	601 s
	Ours	39.33	0.9945	0.0033	864 s
hi4d-dance00	3DGS-Avatar	16.27	0.9164	0.0970	2490 s
	Animatable Gaussians	15.46	0.9098	0.0820	18h36m
	Animatable 3D Gaussian	<u>26.47</u>	<u>0.9557</u>	<u>0.0382</u>	1137 s
	Ours	27.01	0.9592	0.0316	2515 s
hi4d-hug17	3DGS-Avatar	13.55	0.8971	0.1196	2601 s
	Animatable Gaussians	14.95	0.9178	0.0787	17h15m
	Animatable 3D Gaussian	<u>26.20</u>	0.9485	<u>0.0494</u>	930 s
	Ours	26.48	<u>0.9450</u>	0.0463	<u>2483 s</u>
hi4d-piggy27	3DGS-Avatar	16.36	0.9162	0.0884	2649 s
	Animatable Gaussians	15.99	0.9200	0.0707	17h54m
	Animatable 3D Gaussian	<u>26.49</u>	<u>0.9569</u>	<u>0.0406</u>	1134 s
	Ours	26.99	0.9592	0.0340	<u>2554 s</u>

uses multiple views, but it does not explicitly handle occlusion in multi-person scenes, which still leads to missing body parts. Animatable 3D Gaussian is designed for multi-person scenes, but lacks optimization for sparse-view input, resulting in overfitting to observed views and missing or incorrect details in novel views. In terms of training time, the proposed method is much faster than 3DGS-Avatar and Animatable Gaussians. Although it requires more training time than Animatable 3D Gaussian, this cost is exchanged for improved reconstruction quality. Overall, the proposed method provides a better balance between quality and training time for sparse-view multi-person reconstruction.

5.1.3 Ablation Study

An ablation study is conducted to evaluate the effectiveness of the main components proposed in this method, including pose correction, non-uniform Gaussian initialization, the two-stage training strategy, and different values of λ_{dssim} in the loss function. The quantitative results are summarized in Table 5.4.

Pose Correction

After removing pose correction, the average PSNR over the five test sequences drops from 31.84 dB to 30.67 dB, while the average LPIPS increases from 0.0241 to

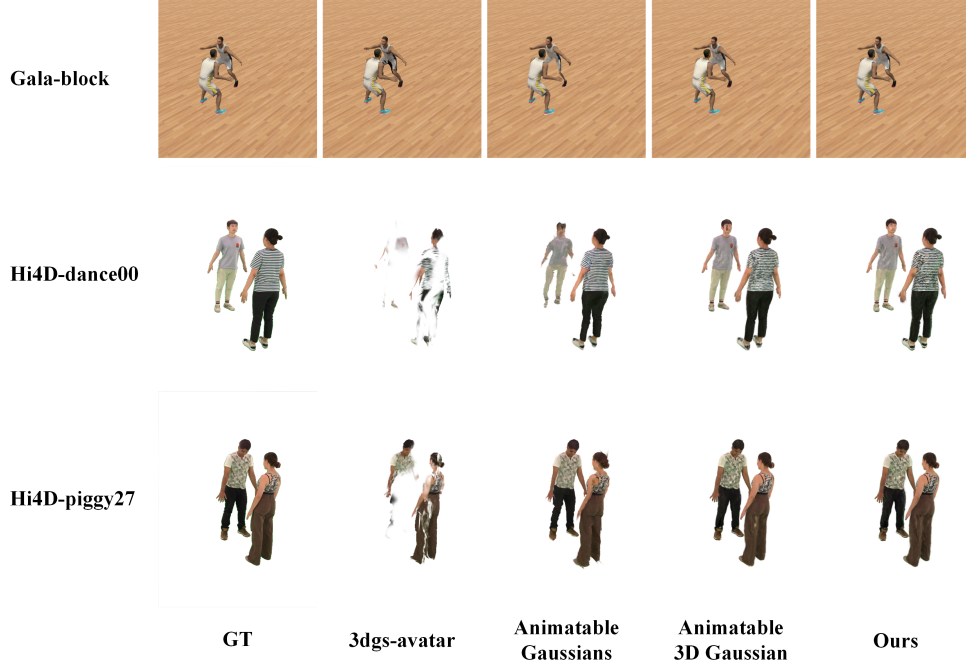


Figure 5.1 Visualization comparison of selected reconstruction and rendering results.

0.0281. This indicates that errors in the initial SMPL pose directly affect the geometric alignment and appearance fitting of the Gaussian avatars. The degradation is most obvious on the GalaBasketball gala-boxout sequence, where PSNR decreases from 39.33 dB to 36.09 dB and LPIPS increases from 0.0033 to 0.0119. This sequence contains clear body contact and occlusion, where inaccurate poses cause Gaussians to deviate from the real human silhouettes, leading to blurred boundaries and missing local details. Therefore, frame-wise pose correction effectively compensates for pose-estimation noise and improves alignment between the body surface and input images. A visual example is shown in Figure 5.2.

Non-uniform Gaussian Initialization

Without non-uniform Gaussian initialization, the average PSNR decreases from 31.84 dB to 31.43 dB, the average SSIM decreases from 0.9703 to 0.9670, and the average LPIPS increases from 0.0241 to 0.0266. Although the overall degradation is smaller than that caused by removing pose correction, all sequences show some quality loss. In particular, the LPIPS on hi4d-hug17 increases from 0.0463 to 0.0512. This result indicates that different body parts require different Gaussian densities. Detail-rich regions such as the head, hands, and feet require denser representations, whereas smoother regions such as the torso can be represented with fewer Gaussians. Non-uniform initialization allocates the limited number of Gaussians to regions where detailed representation is more important, improving local structure

Table 5.4 Full ablation results on the selected test sequences. Best results are shown in bold, and second-best results are underlined.

Sequence	Configuration	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$
gala-block	Ours	39.37	0.9938	0.0052
	w/o pose correction	38.10	0.9926	0.0065
	w/o non-uniform init.	<u>38.98</u>	<u>0.9932</u>	0.0062
	Alternating training only	14.75	0.8561	0.2409
	Joint training only	36.77	0.9910	0.0066
	$\lambda_{\text{dssim}} = 0.2$	38.45	0.9920	<u>0.0061</u>
	$\lambda_{\text{dssim}} = 0.5$	38.20	0.9923	<u>0.0061</u>
gala-boxout	Ours	39.33	0.9945	0.0033
	w/o pose correction	36.09	0.9870	0.0119
	w/o non-uniform init.	38.92	0.9925	0.0062
	Alternating training only	10.76	0.6874	0.4461
	Joint training only	37.52	0.9927	0.0043
	$\lambda_{\text{dssim}} = 0.2$	39.09	0.9938	0.0037
	$\lambda_{\text{dssim}} = 0.5$	<u>39.28</u>	<u>0.9943</u>	<u>0.0036</u>
hi4d-dance00	Ours	<u>27.01</u>	0.9592	0.0316
	w/o pose correction	26.59	0.9568	0.0346
	w/o non-uniform init.	26.65	0.9554	0.0343
	Alternating training only	26.66	0.9578	0.0325
	Joint training only	25.46	0.9484	0.0689
	$\lambda_{\text{dssim}} = 0.2$	27.05	<u>0.9591</u>	0.0328
	$\lambda_{\text{dssim}} = 0.5$	26.98	0.9590	<u>0.0320</u>
hi4d-hug17	Ours	<u>26.48</u>	0.9450	<u>0.0463</u>
	w/o pose correction	26.02	0.9424	0.0500
	w/o non-uniform init.	26.02	0.9397	0.0512
	Alternating training only	24.07	0.9355	0.0541
	Joint training only	26.16	0.9428	0.0437
	$\lambda_{\text{dssim}} = 0.2$	26.70	0.9476	0.0474
	$\lambda_{\text{dssim}} = 0.5$	26.70	<u>0.9467</u>	0.0472
hi4d-piggy27	Ours	<u>26.99</u>	0.9592	0.0340
	w/o pose correction	26.55	0.9560	0.0377
	w/o non-uniform init.	26.59	0.9541	<u>0.0352</u>
	Alternating training only	26.46	0.9441	0.0451
	Joint training only	26.78	0.9561	0.0360
	$\lambda_{\text{dssim}} = 0.2$	26.92	0.9585	0.0371
	$\lambda_{\text{dssim}} = 0.5$	27.02	<u>0.9589</u>	0.0365

and appearance reconstruction.

Two-stage Training

The ablation results show that using only alternating training or only joint training is inferior to the proposed “alternating-then-joint” strategy. When only alternating training is used, the PSNR values on the two GalaBasketball sequences are only 14.75 dB and 10.76 dB, respectively. This suggests that if the persons are always

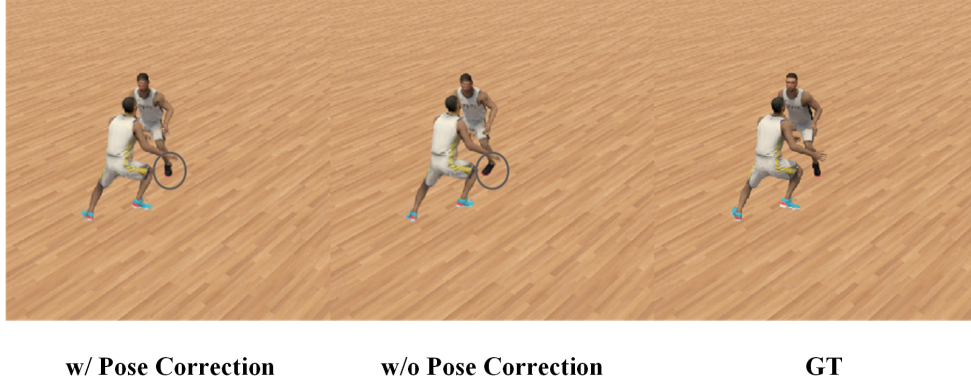


Figure 5.2 Visual example of the effect of pose correction.

optimized separately, the model cannot learn correct global occlusion and color consistency for the final multi-person composition. Joint-only training is more stable, but the average PSNR is only 30.54 dB and the average LPIPS is 0.0319, which is still clearly worse than the full method. This shows that direct joint optimization in the early stage is affected by multi-person occlusion and boundary mixing. Alternating training first allows the Gaussians of each person to converge to reasonable regions independently, while the following joint training further improves occlusion handling and global consistency when multiple people are rendered together.

Value of λ_{dssim}

From the objective metrics, the differences among $\lambda_{\text{dssim}} = 0.2, 0.5$, and 0.8 are relatively small in terms of PSNR and SSIM. The average PSNR values remain around 31.6–31.8 dB, and the average SSIM values are all close to 0.970. This suggests that pixel-level and patch-level objective metrics are not sufficient to clearly distinguish the perceptual differences introduced by different D-SSIM weights. Nevertheless, $\lambda_{\text{dssim}} = 0.8$ achieves the best average LPIPS value of 0.0241, slightly better than 0.5 (0.0251) and 0.2 (0.0254), indicating that a higher structural-similarity weight helps improve perceptual texture and boundary quality. A visual example is shown in Figure 5.3.

PSNR, SSIM, and LPIPS all measure image differences at the pixel or patch level, and are not sufficiently sensitive to the perceptual changes brought by increasing λ_{dssim} , such as improved texture coherence and structural clarity. Therefore, in addition to objective metrics, a subjective image quality evaluation is conducted to directly collect human perceptual preferences.

The subjective study collects 585 valid pairwise comparison records from 13 participants over 45 test samples. Each candidate value appears in 390 comparisons. In terms of overall selection rate, $\lambda = 0.8$ ranks first with 56.4% (220 wins out of 390 appearances), followed by $\lambda = 0.5$ with 33.1% (129/390), and $\lambda = 0.2$ with

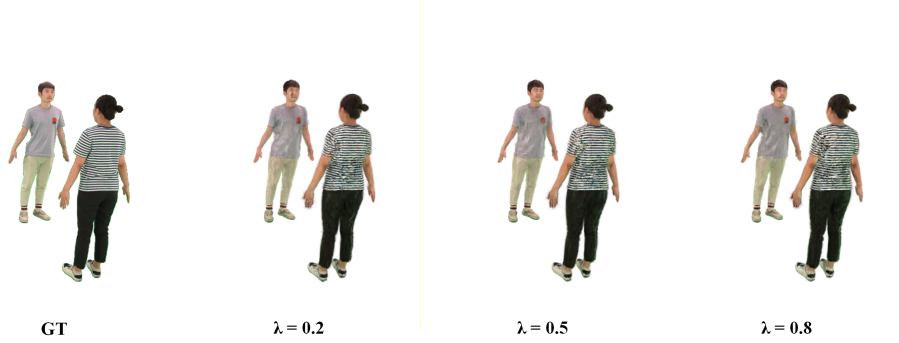


Figure 5.3 Visual comparison of different λ_{dssim} values.

15.9% (62/390). The pairwise comparison results show the same preference trend: after excluding “Similar / No Preference” responses, $\lambda = 0.8$ achieves a conditional win rate of 80.3% against $\lambda = 0.2$ (114/(114+28)) and 75.2% against $\lambda = 0.5$ (106/(106+35)); $\lambda = 0.5$ achieves a conditional win rate of 73.4% against $\lambda = 0.2$ (94/(94+34)). The Bradley-Terry model further confirms this ranking: $\lambda = 0.8$ obtains a BT score of 0.628 (95% CI: [0.568, 0.695]), $\lambda = 0.5$ obtains 0.256 (95% CI: [0.205, 0.304]), and $\lambda = 0.2$ obtains 0.116 (95% CI: [0.084, 0.148]). The confidence intervals do not overlap, indicating a statistically stable ranking. These results show that increasing λ_{dssim} provides a consistent and significant improvement in perceptual quality, supporting the choice of $\lambda_{\text{dssim}} = 0.8$ in this thesis. The subjective evaluation results are visualized in Figures 5.4, 5.5, and 5.6.

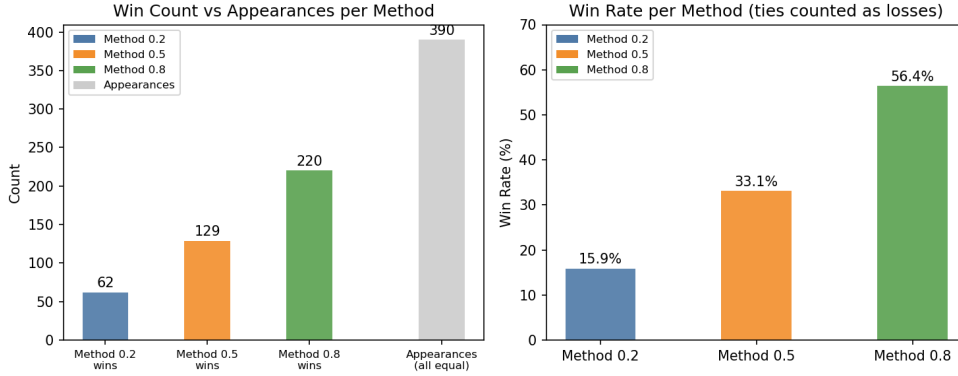


Figure 5.4 Overall win rate of different λ_{dssim} settings in the subjective evaluation.

5.1.4 Pose-driven Avatar Rendering

This subsection presents a qualitative experiment on pose-driven avatar rendering. The purpose is to examine whether a trained Gaussian avatar can be animated by an external pose sequence. The trained Gaussian avatars are not fixed Gaussian point

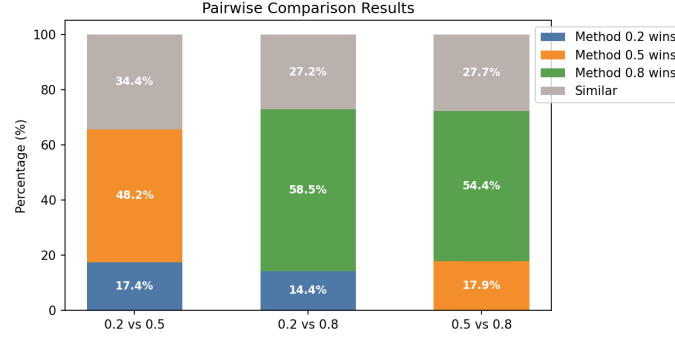


Figure 5.5 Pairwise comparison results of different λ_{dssim} settings.

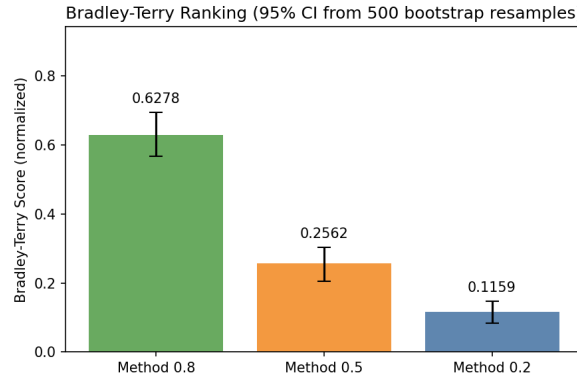


Figure 5.6 Bradley-Terry scores of different λ_{dssim} settings with 95% confidence intervals.

sets at static positions. Instead, they can be driven by pose data to render dynamic motion. The driving pose is not limited to the original pose sequence used during avatar training; external pose sequences can also be used to make the avatar perform specified actions. This suggests that, once the avatar appearance has been learned, the model can be reused with new pose streams as long as the poses are represented in the same SMPL-compatible format. Figure 5.7 shows the rendering result of avatars trained on Hi4D data and driven by motion sequences from BASKET-Multiview.

The trained Gaussian avatars are not fixed Gaussian point sets at static positions. Instead, they can be driven by pose data to render dynamic motion. The driving pose is not limited to the original pose sequence used during avatar training; external pose sequences can also be used to make the avatar perform specified actions. Figure 5.7 shows the rendering result of avatars trained on Hi4D data and driven by motion sequences from BASKET-Multiview.

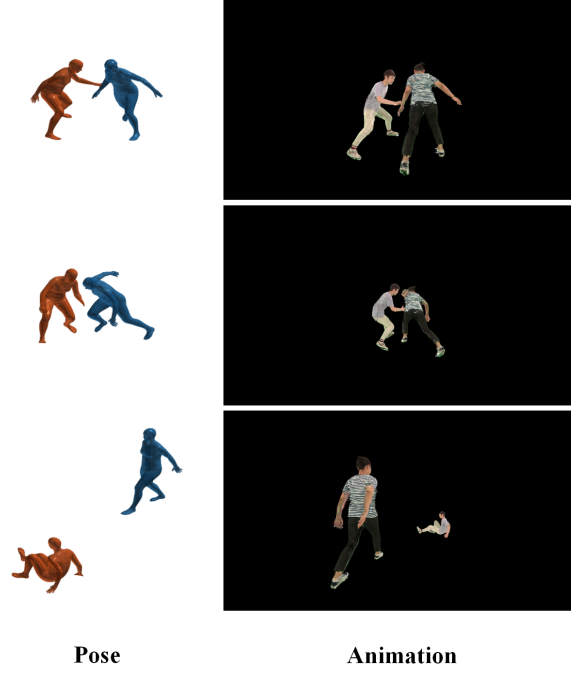


Figure 5.7 Animation results of Gaussian avatars driven by external motion sequences.

5.2 Environment Reconstruction

To reconstruct the sports environment, images from the BASKET-Multiview dataset are used to reconstruct a basketball court. Two different sequences are selected, where the humans are located on different sides of the court and do not overlap. This helps recover a more complete scene. For each sequence, one image is selected from each of the 89 views, resulting in 178 images in total, together with their corresponding masks, for training the scene Gaussians. Since the reconstructed scene is mainly used as a background, no additional improvement is applied to the training process, and no dedicated test views are held out. Evaluation on the training views gives an L1 error of 0.01356, a PSNR of 32.42 dB, and an SSIM of 0.9458. The training time for the basketball court is 45 minutes. Because no held-out views are used for this experiment, the reported metrics only measure reconstruction quality on the training views and should not be interpreted as novel-view generalization performance. Figure 5.8 shows a comparison between the training images and the Gaussian-rendered results.

This environment reconstruction experiment is not intended to validate sparse-camera background reconstruction. Instead, it uses the dense-view annotations of BASKET-Multiview to obtain a complete static court representation for demonstrating the integrated replay system.



Figure 5.8 Environment reconstruction results of the basketball court.

5.3 Sports Scene Replay

Using the viewer introduced in Section 4.4, one or more Gaussian avatars and the environment Gaussians of the basketball court are placed in the same scene. The avatars are driven by the corresponding motion pose sequences, producing a reconstructed dynamic sports scene that can be replayed from arbitrary viewpoints. Figure 5.9 shows several replay results from novel viewpoints.

The rendering performance of the viewer is summarized in Table 5.5. The results show that large-scale scene content has a significant impact on frame rate. In addition, because the Gaussians of dynamic avatars need to be recomputed under different poses, human motion also reduces the rendering frame rate.

Table 5.5 Rendering frame rates of the viewer under different scene compositions.

Components in the Scene	FPS
1 static avatar	119.0
1 dynamic avatar	102.5
2 static avatars	67.3
2 dynamic avatars	55.4
5 static avatars	26.6
5 dynamic avatars	20.9
Basketball court	18.0
Basketball court + 1 dynamic avatar	15.3
Basketball court + 2 dynamic avatars	11.0
Basketball court + 5 dynamic avatars	7.7

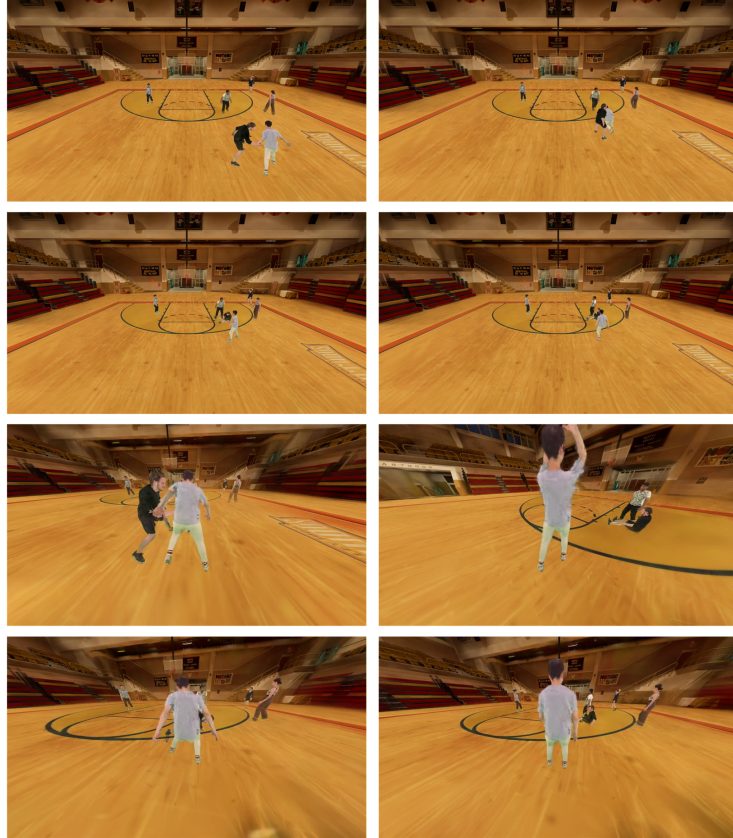


Figure 5.9 Sports scene replay results from novel viewpoints.

6 Discussion

This chapter discusses several observations that are important for understanding the method, including the influence of input pose quality, the effect of the D-SSIM loss weight, and an unsuccessful attempt to learn from a teacher model.

6.1 Data Quality of SMPL Pose Parameters

The quality of SMPL pose parameters has a direct influence on avatar reconstruction. Inaccurate poses cause the Gaussian positions deformed by LBS to become misaligned with the real human silhouettes, which in turn reduces the overall reconstruction quality. This section uses the GalaBasketball dataset as an example to discuss the evolution of the pose extraction strategy and its effect on the reconstruction results.

When extracting SMPL pose parameters for GalaBasketball, an initial attempt used a numerical fitting strategy based on 3D joint positions, similar in spirit to SMPLify. The procedure was as follows. First, forward kinematics (FK) was applied to the Gala skeleton animation data, converting local Euler angles of the joints into global joint positions in the world coordinate system. Then, 14 keypoints corresponding to the COCO-14 format were extracted, including the head, neck, shoulders, elbows, wrists, hips, knees, and ankles. These 3D keypoints were used as supervision, and gradient descent was used to optimize the SMPL pose parameters (θ, θ_0) , shape parameters β , and global translation $\mathbf{t}$, minimizing the Huber error between the corresponding SMPL joints and the observed keypoints.

However, this strategy has several fundamental limitations, which lead to systematic distortions in limb rotations.

Lossy reduction of information. The Gala dataset directly provides complete joint rotations in the form of Euler angles. The initial strategy first converted these rotations into joint positions and then attempted to recover rotations from the positions. This is a lossy reduction of the original information. 3D joint positions cannot uniquely determine joint rotations, especially the twist degree of freedom around the bone axis. As a result, the optimization can easily fall into unreasonable local minima and produce degenerate rotation parameters.

Severe under-constrained optimization. The SMPL model has 24 joints and 72 rotational degrees of freedom, including 69 dimensions for body pose and 3 dimensions for root orientation. In contrast, the 14 3D keypoints provide only 42 positional constraints. The problem is therefore strongly under-constrained. Many joints that are not directly observed, such as internal spine joints, can only be implicitly constrained by regularization terms, making their estimated rotations unreliable.

Unresolved skeleton topology mismatch. The Gala skeleton has 27 joints and contains six spine segments, while SMPL has only 24 joints and five spine-related segments. The two skeletons therefore have different topologies. The initial fitting strategy did not explicitly compensate for this mismatch, and directly mapped Gala joints to the nearest SMPL joints, which produced large fitting residuals around the spine.

Due to these issues, the SMPL pose parameters obtained by numerical fitting from joint positions were limited in quality and introduced additional pose noise into avatar training. The improved strategy described in the method chapter, which directly extracts rotations and applies bone-orientation correction, makes fuller use of the original rotation information provided by GalaBasketball. It avoids the under-constrained fitting and information loss of the previous approach, significantly improving the quality of the pose parameters, as shown in Figure 6.1.

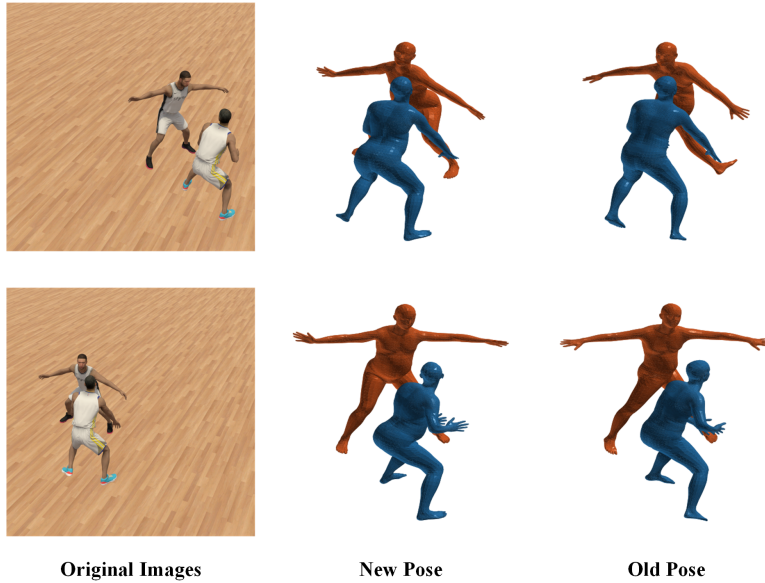


Figure 6.1 Comparison of avatar reconstruction quality using different SMPL pose extraction strategies.

After updating the pose fitting strategy from the old method to the improved method, the Gala block sequence obtains improvements of PSNR +4.16, SSIM +0.0099, and LPIPS -0.0096. This improvement is larger than that brought by any other individual

modification in the method, indicating that input data quality is crucial for learning-based avatar reconstruction.

6.2 Why a Higher λ_{DSSIM} Improves Subjective Avatars Quality

From the perspective of optimization, increasing λ_{dssim} does not only change the numerical weight between two loss terms. More importantly, it changes the gradient structure back-propagated to the Gaussian parameters. For L1 loss, the gradient of the rendered image I at pixel p can be written as

$$\frac{\partial \mathcal{L}_1}{\partial I(p)} = \frac{1}{N} \text{sign}(I(p) - I_{\text{gt}}(p)). \quad (6.1)$$

This gradient depends only on the error of the individual pixel and has no direct relationship with neighboring pixels. Its magnitude also remains almost constant, even when the error becomes small. Therefore, L1 loss tends to continuously push the model to fit independent pixel-level color differences in the training views. For sparse-view Gaussian Avatar reconstruction, this pixel-wise constraint can make the optimization focus too much on local errors in the training views, while lacking constraints on neighborhood structure and cross-view consistency.

In contrast, SSIM depends on statistics computed within a local window W , including the mean, variance, and covariance. Although its exact gradient is more complex, its core effect can be expressed as the joint influence of several local statistical terms on each pixel:

$$\frac{\partial \mathcal{L}_{\text{SSIM}}}{\partial I(p)} = \sum_{W \ni p} \left(\frac{\partial \text{SSIM}_W}{\partial \mu_x} \frac{\partial \mu_x}{\partial I(p)} + \frac{\partial \text{SSIM}_W}{\partial \sigma_x^2} \frac{\partial \sigma_x^2}{\partial I(p)} + \frac{\partial \text{SSIM}_W}{\partial \sigma_{xy}} \frac{\partial \sigma_{xy}}{\partial I(p)} \right). \quad (6.2)$$

This means that the gradient at a pixel reflects not only the color error of that pixel, but also the luminance distribution, contrast variation, and structural correlation in its neighborhood. In other words, SSIM provides an optimization signal for matching local image patterns, rather than independent pixels. When local texture, edges, or silhouette structures do not match the ground truth, SSIM can still produce meaningful gradients through local statistical differences, even if some pixel values are already close to the target.

This difference is particularly important after the gradient is propagated to Gaussian parameters. Suppose a Gaussian g_i contributes to pixel p with weight $\alpha_i(p)T_i(p)$,

where α_i denotes opacity contribution and T_i denotes transmittance. The gradient of its color parameter c_i can be written as

$$\frac{\partial \mathcal{L}}{\partial c_i} = \sum_p \frac{\partial \mathcal{L}}{\partial I(p)} \alpha_i(p) T_i(p). \quad (6.3)$$

Under L1 loss, pixels from different training views independently ask the same Gaussian to adjust its color. When sparse views contain occlusions, pose errors, or local misalignment, these pixel-wise gradients can conflict with each other, pulling the color parameter toward a compromise across views. SSIM first introduces local-window statistics in image space, so the gradient propagated to the Gaussian already contains a certain structural consistency constraint. This reduces the influence of single-pixel noise on color updates.

The difference is even more apparent for geometric parameters such as position, scale, and rotation. Taking the Gaussian center μ_i as an example, its gradient can be written as

$$\frac{\partial \mathcal{L}}{\partial \mu_i} = \sum_p \frac{\partial \mathcal{L}}{\partial I(p)} \frac{\partial [\alpha_i(p) T_i(p)]}{\partial \mu_i} c_i. \quad (6.4)$$

Geometric parameters do not directly change pixel colors. Instead, they change the projected region and weight distribution of the Gaussian on the image plane. If strong L1 supervision is used alone, a degenerate elongated Gaussian may fit local pixel colors in the training view, causing the L1 gradient in its covered region to become small. As a result, its geometric parameters may not receive enough signal to correct the degenerate shape. In contrast, SSIM continues to compare variance, contrast, and covariance patterns within local windows. If an elongated projection causes unnatural edges, abnormal local contrast, or broken texture patterns, SSIM can still generate gradients and propagate them to the Gaussian position, scale, and rotation, even when pixel-wise color error is small. This helps suppress overfitting through geometric degeneration.

From the perspective of optimization stability, the sign gradient of L1 loss may continue to oscillate around the true pixel value near convergence. In sparse-view training, the same Gaussian may explain colors or geometry differently across views. Constant-magnitude pixel-wise gradients can amplify these conflicts and may encourage the model to reduce scale, elongate shape, or shift position to avoid cross-view color inconsistency. The local-window statistics of SSIM smooth the neighborhood gradients to some extent, and once local structures become matched, the gradients tend to stabilize as the structural error decreases.

Therefore, increasing λ_{dssim} can be understood as shifting the optimization target from pixel-wise color fitting toward local structure matching. L1 more easily encourages the model to memorize independent pixel values in the training views, whereas SSIM emphasizes local patterns that are more stable under viewpoint changes, such as boundary shape, texture continuity, and contrast relationships within a region. These local structures are usually closer to the projections of real 3D surface properties than to accidental pixel observations from a single training view. To verify this interpretation, λ_{dssim} was set to the extreme values of 0 and 1, and the Gaussian gradient magnitudes during training was visualized. As shown in Figure 6.2, when $\lambda_{\text{dssim}} = 1$, the edge regions of the avatar and high-frequency texture regions receive clearly stronger gradient than when $\lambda_{\text{dssim}} = 0$. This indicates that these Gaussians are optimized more actively during training.

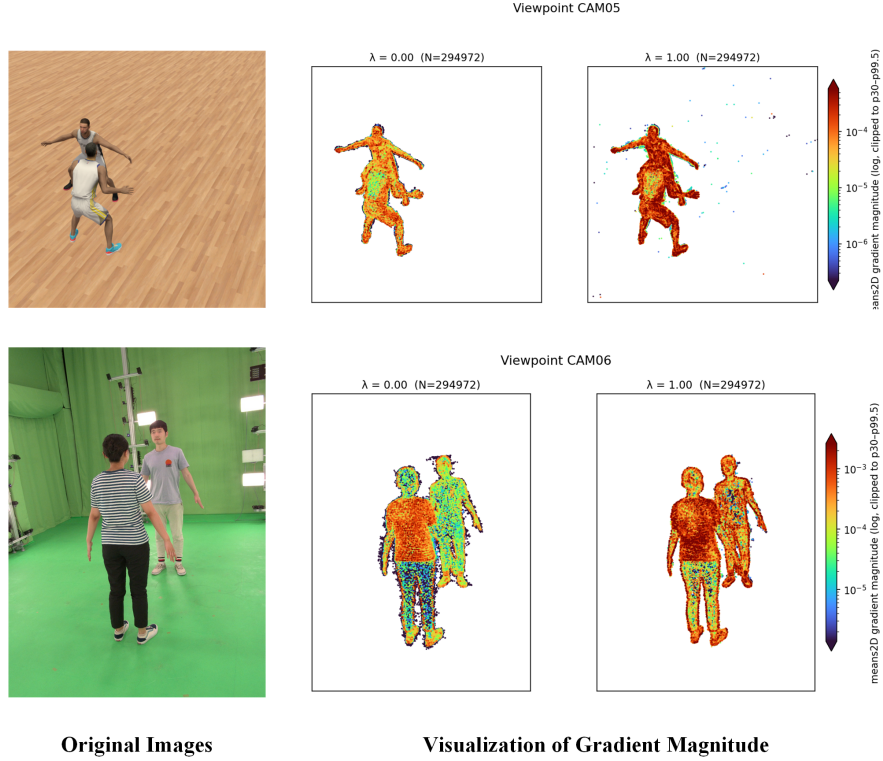


Figure 6.2 Visualization of Gaussian gradient magnitudes under different λ_{dssim} settings.

Thus, in sparse-view Gaussian Avatar reconstruction, increasing the D-SSIM weight helps reduce training-view overfitting and geometric degeneration, making novel-view renderings more consistent in structure, boundaries, and texture stability from the perspective of human observers.

6.3 Failure to Learn from a Teacher Model

During the experimental process, this thesis also attempted to introduce a static teacher model to assist Gaussian Avatar training. The idea is shown in Figure 6.3. Specifically, UP2YOU [3] was used to generate a vertex-colored clothed mesh from multiple human images with arbitrary poses. The mesh was stored in PLY format and contained XYZ coordinates and RGB colors, and was treated as a geometry and color prior in canonical space. In implementation, this strategy consisted of two independent supervision mechanisms.

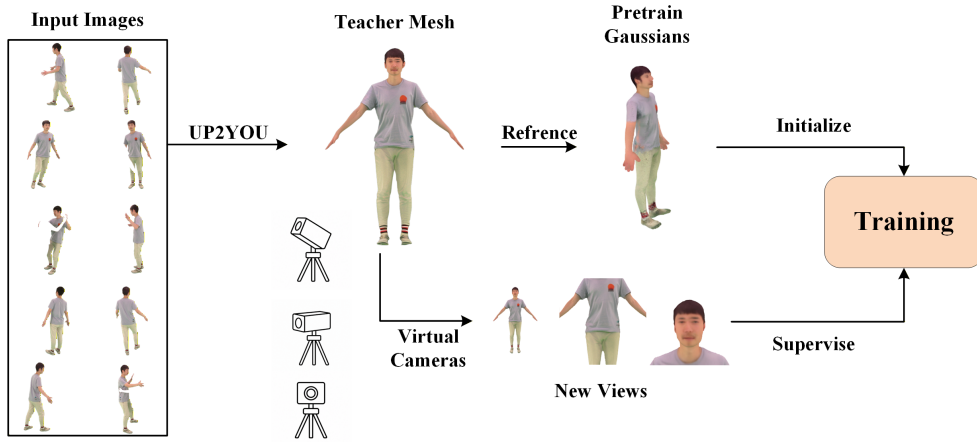


Figure 6.3 Attempted teacher model strategy for Gaussian Avatar training.

The first mechanism was **teacher initialization**. Before the formal multi-view training stage, several initialization iterations were performed. The teacher point cloud and the Gaussian A-pose positions were matched in 3D space using nearest-neighbor search, and four losses were applied to each Gaussian: a position loss based on Smooth L1, a base spherical harmonic color loss on SH0, a Gaussian scale loss, and an opacity loss. These losses encouraged the Gaussians to move toward the teacher geometry and color distribution at the early stage of training.

The second mechanism was **teacher supervision**. During the main training process, the teacher mesh was rendered from virtual cameras at fixed intervals using the pyrender offscreen renderer. At the same time, the Gaussians were rasterized from the same virtual viewpoints, and a photometric loss between the two rendered images was used as additional supervision. Two types of virtual cameras were used. The first type sampled full-body views on a hemisphere around the human body, with the camera placed approximately 2.2 m from the body center and the elevation angle randomly sampled from -15° to $+30^\circ$. The second type used close-up patch cameras for individual body regions, including the head, front and back torso, left and right arms, left and right legs, and both feet. These close-up views were intended

to provide additional supervision for body regions that are difficult to observe under sparse real camera views.

To reduce the spatial discrepancy between the UP2YOU output and the SMPL canonical body, the teacher mesh was uniformly scaled according to the body height along the Y axis and translated to the center of the student A-pose bounding box. This similarity transformation used a scale factor constrained between 0.8 and 1.2. In addition, PCA was used to estimate the orientation angles of the left and right arms of the teacher mesh, and the body pose parameters of the student A-pose were adjusted to match the teacher arm angles. This avoided introducing additional errors by directly deforming the teacher mesh. In principle, this two-stage auxiliary supervision strategy could provide reasonable geometry and color initialization for the Gaussians at the early stage of training and continuously provide gradients for poorly observed regions during training, thereby alleviating training instability caused by sparse gradients and the small image area occupied by the human body under sparse-view input.

However, this strategy did not provide stable improvement in practice and was therefore not included in the final method. There are two main reasons. First, the teacher avatar cannot be perfectly aligned with the target avatar in 3D space. Even when both are in canonical pose, the mesh generated by UP2YOU still differs from the canonical body used by SMPL/A3DG in scale, limb position, and surface details. This mismatch introduces incorrect geometric constraints during encoder warmup or virtual-camera supervision, pulling Gaussians toward inaccurate positions and reducing the convergence quality of subsequent real multi-view training. Second, the appearance of the teacher mesh is not fully reliable. In particular, hallucinated details may appear in clothing patterns, texture boundaries, and occluded regions. These generated appearance details do not necessarily correspond to the real person. If they are used directly as supervision, the avatar may learn incorrect colors and textures. Therefore, although the teacher model can provide additional geometric and appearance information, in this task the negative effects caused by misalignment and appearance hallucination outweigh the potential benefits. This direction was ultimately abandoned.

7 Conclusion

This chapter summarizes the main contributions and limitations of the thesis, and outlines possible directions for future work.

This thesis addressed the problem of low-cost free-viewpoint replay for sports scenes by developing a 4D reconstruction pipeline that combines data from a motion capture system with 3D Gaussian Splatting representations. The proposed pipeline organizes dynamic human Gaussian avatars, static environment reconstruction, and an interactive viewer within a unified framework. The dynamic avatar component was evaluated under sparse-view settings, while the environment reconstruction and integrated replay were demonstrated using denser background observations. To improve reconstruction under multi-person settings with pose errors, limited initialization, and training instability, this work introduced pose correction, non-uniform Gaussian initialization, two-stage training, and an appropriate loss-weight setting. The effectiveness of these design choices was evaluated through objective metrics, subjective assessment, ablation studies, and application demonstrations.

At the same time, the proposed method still has several limitations. Since the input views are sparse, the appearance and geometry of insufficiently observed body regions cannot always be fully recovered. As a result, some novel views may contain blurred textures, unstable local structures, or missing details. In addition, although the system can reconstruct and combine multiple human avatars with the environment, the overall training process still requires a considerable amount of time. Further optimization is therefore needed before the method can be deployed more efficiently or scaled to larger applications.

Future work can proceed in two main directions. First, based on the unsuccessful teacher-model strategy explored in this thesis, generative methods could be further investigated for high-fidelity Gaussian avatar training. A key challenge is to reduce spatial misalignment and appearance hallucination between the generated prior and the target avatar, so that generative priors can provide reliable additional supervision under sparse-view settings. Second, at the system level, the viewer can be improved through more structured Gaussian data management, more efficient temporal data loading, and rendering-side acceleration. These improvements could increase the frame rate and enhance the interactivity and usability of the system in real sports replay scenarios.

References

- [1] F. Angehrn, O. Wang, Y. Aksoy, M. Gross, and A. Smolić. “Mastercam fvv: robust registration of multiview sports video to a static high-resolution master camera for free viewpoint video”. In: *2014 IEEE International Conference on Image Processing (ICIP)*. IEEE. 2014, pp. 3474–3478.
- [2] R. A. Bradley and M. E. Terry. “Rank analysis of incomplete block designs: i. the method of paired comparisons”. *Biometrika* **39**:3/4 (1952), pp. 324–345.
- [3] Z. Cai, Z. Li, X. Li, B. Li, Z. Wang, Z. Zhang, and Y. Xiu. “Up2you: fast reconstruction of yourself from unconstrained photo collections”. *arXiv preprint arXiv:2509.24817* (2025).
- [4] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. “Realtime multi-person 2d pose estimation using part affinity fields”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 7291–7299.
- [5] Y. Chen, Z. Zheng, Z. Li, C. Xu, and Y. Liu. “Meshavatar: learning high-quality triangular human avatars from multi-view videos”. In: *European Conference on Computer Vision*. Springer. 2024, pp. 250–269.
- [6] A. Collet, M. Chuang, P. Sweeney, D. Gillett, D. Evseev, D. Calabrese, H. Hoppe, A. Kirk, and S. Sullivan. “High-quality streamable free-viewpoint video”. *ACM Transactions on Graphics (ToG)* **34**:4 (2015), pp. 1–13.
- [7] K. Das, T. de Paula Oliveira, and J. Newell. “Comparison of markerless and marker-based motion capture systems using 95% functional limits of agreement in a linear mixed-effects modelling framework”. *Scientific Reports* **13**:1 (2023), p. 22880.
- [8] P. Goorts, S. Maesen, M. Dumont, S. Rogmans, and P. Bekaert. “Free viewpoint video for soccer using histogram-based validity maps in plane sweeping”. In: *2014 International Conference on Computer Vision Theory and Applications (VISAPP)*. Vol. 3. IEEE. 2014, pp. 378–386.
- [9] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [10] J. Huang, S. Subhajiya Mallick, A. Amat, M. Ruiz Olle, A. Mosella-Montoro, B. Kerbl, F. Vicente Carrasco, and F. De la Torre. “Echoes of the coliseum: towards 3d live streaming of sports events”. *ACM Transactions on Graphics (TOG)* **44**:4 (2025), pp. 1–17.
- [11] Q. Huynh-Thu and M. Ghanbari. “Scope of validity of psnr in image/video quality assessment”. *Electronics letters* **44**:13 (2008), pp. 800–801.

- [12] G. Jiang, T. Zhang, D. Li, Z. Zhao, H. Li, M. Li, and H. Wang. “Stg-avatar: animatable human avatars via spacetime gaussian”. In: *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2025, pp. 20058–20065.
- [13] T. Jiang, X. Chen, J. Song, and O. 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. 16922–16932.
- [14] Y. Jiang, J. Li, H. Qin, Y. Dai, J. Liu, G. Zhang, C. Zhang, and T. Yang. “Gs-sfs: joint gaussian splatting and shape-from-silhouette for multiple human reconstruction in large-scale sports scenes”. *IEEE Transactions on Multimedia* **26** (2024), pp. 11095–11110.
- [15] T. Kanade, P. Rander, and P. Narayanan. “Virtualized reality: constructing virtual worlds from real scenes”. *IEEE multimedia* **4:1** (1997), pp. 34–47.
- [16] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. “3d gaussian splatting for real-time radiance field rendering”. *ACM Transactions on Graphics* **42:4** (2023), pp. 1–14.
- [17] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick. “Segment anything”. *arXiv:2304.02643* (2023).
- [18] M. Li, S. Yao, C. Kai, Z. Xie, K. Chen, and Y.-G. Jiang. “Anigaussian: animatable gaussian avatar with pose-guided deformation”. *arXiv preprint arXiv:2502.19441* (2025).
- [19] Z. Li, Y. Sun, Z. Zheng, L. Wang, S. Zhang, and Y. Liu. “Animatable and reliable gaussians for high-fidelity human avatar modeling”. *arXiv preprint arXiv:2311.16096* (2023).
- [20] Z. Li, Z. Zheng, L. Wang, and Y. Liu. “Animatable gaussians: learning pose-dependent gaussian maps for high-fidelity human avatar modeling”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2024, pp. 19711–19722.
- [21] T. Liao, X. Zhang, Y. Xiu, H. Yi, X. Liu, G.-J. Qi, Y. Zhang, X. Wang, X. Zhu, and Z. Lei. “High-fidelity clothed avatar reconstruction from a single image”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 8662–8672.
- [22] Y. Liu, X. Huang, M. Qin, Q. Lin, and H. Wang. “Animatable 3d gaussian: fast and high-quality reconstruction of multiple human avatars”. In: *Proceedings of the 32nd ACM International Conference on Multimedia*. 2024, pp. 1120–1129.
- [23] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. “Smpl: a skinned multi-person linear model”. *ACM Transactions on Graphics* **34:6** (2015).
- [24] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. “Nerf: representing scenes as neural radiance fields for view synthesis”. *Communications of the ACM* **65:1** (2021), pp. 99–106.
- [25] A. Mir, R. A. Guler, X. Tang, P. Wonka, and G. Pons-Moll. “Ahoy! animatable humans under occlusion from youtube videos with gaussian splatting and video diffusion priors”. *arXiv preprint arXiv:2603.17975* (2026).

- [26] A. Mir, J. Wang, R. A. Guler, C. Guo, G. Pons-Moll, and B. Zhou. “Aha! animating human avatars in diverse scenes with gaussian splatting”. *arXiv preprint arXiv:2511.09827* (2025).
- [27] S. Peng, W. Xie, Z. Wang, X. Guo, Z. Chen, B. Yang, and X. Dong. “Rmavatar: photorealistic human avatar reconstruction from monocular video based on rectified mesh-embedded gaussians”. *Graphical Models* **139** (2025), p. 101266.
- [28] Z. Qian, S. Wang, M. Mihajlovic, A. Geiger, and S. Tang. “3dgs-avatar: animatable avatars via deformable 3d gaussian splatting”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2024, pp. 5020–5030.
- [29] S. Saito, T. Simon, J. Saragih, and H. Joo. “Pifuhd: multi-level pixel-aligned implicit function for high-resolution 3d human digitization”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 84–93.
- [30] J. L. Schönberger and J.-M. Frahm. “Structure-from-motion revisited”. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.
- [31] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm. “Pixelwise view selection for unstructured multi-view stereo”. In: *European Conference on Computer Vision (ECCV)*. 2016.
- [32] Z. Shao, Z. Wang, Z. Li, D. Wang, X. Lin, Y. Zhang, M. Fan, and Z. Wang. “Splattingavatar: realistic real-time human avatars with mesh-embedded gaussian splatting”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 1606–1616.
- [33] A. Smolic. “3d video and free viewpoint video—from capture to display”. *Pattern recognition* **44**:9 (2011), pp. 1958–1968.
- [34] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. “Bundle adjustment—a modern synthesis”. In: *International workshop on vision algorithms*. Springer. 1999, pp. 298–372.
- [35] H. Tu, C. Wang, and W. Zeng. “Voxelpose: towards multi-camera 3d human pose estimation in wild environment”. In: *European conference on computer vision*. Springer. 2020, pp. 197–212.
- [36] W. Wang, H. Yang, J. Kittler, and X. Zhu. “Dynamic avatar-scene rendering from human-centric context”. *arXiv preprint arXiv:2511.10539* (2025).
- [37] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. “Image quality assessment: from error visibility to structural similarity”. *IEEE transactions on image processing* **13**:4 (2004), pp. 600–612.
- [38] C.-Y. Weng, B. Curless, P. P. Srinivasan, J. T. Barron, and I. 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*. 2022, pp. 16210–16220.
- [39] S. Würmlin, E. Lamboray, and M. Gross. “3d video fragments: dynamic point samples for real-time free-viewpoint video”. *Computers & Graphics* **28**:1 (2004), pp. 3–14.
- [40] Y. Yin, C. Guo, M. Kaufmann, J. J. Zarate, J. Song, and O. Hilliges. “Hi4d: 4d instance segmentation of close human interaction”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 17016–17027.

- [41] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. “The unreasonable effectiveness of deep features as a perceptual metric”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 586–595.
- [42] Y. Zhang, Z. Li, L. An, M. Li, T. Yu, and Y. Liu. “Lightweight multi-person total motion capture using sparse multi-view cameras”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 5560–5569.
- [43] L. Zhu, K. Rematas, B. Curless, S. M. Seitz, and I. Kemelmacher-Shlizerman. “Reconstructing nba players”. In: *European conference on computer vision*. Springer. 2020, pp. 177–194.
- [44] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski. “High-quality video view interpolation using a layered representation”. *ACM transactions on graphics (TOG)* **23**:3 (2004), pp. 600–608.