Echoes of the Coliseum: Towards 3D Live streaming of Human-centric Events

Junkai Huang

CMU-RI-TR-25-24

May 5

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Professor Fernando De la Torre Frade, *Chair*
Professor Shubham Tulsiani
Jianjin Xu

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

To my parents.

Abstract

Human-centered live events have always played a pivotal role in shaping culture and fostering social connections. Traditional 2D live transmissions fail to replicate the immersive quality of physical attendance. Addressing this gap, this thesis proposes a framework towards real-time, photo-realistic 3D reconstructions of live events using high-performance 3D Gaussian Splatting.

Our solution capitalizes on strong geometric priors to optimize through distributed processing and load balancing, enabling interactive, freely explorable 3D experiences. By dividing scene reconstruction into actor-centric and environment-specific tasks, we employ hierarchical coarse-to-fine optimization to rapidly and accurately reconstruct human actors based on pose data, refining their geometry and appearance with photometric loss. For static environments, we focus on view-dependent appearance changes, streamlining rendering efficiency and maximizing GPU performance. To facilitate evaluation, we introduce (and distribute) a synthetic benchmark dataset of basketball games, offering high visual fidelity as ground truth. In both our synthetic benchmark and publicly available benchmarks, our method consistently outperforms existing approaches.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my advisor, Prof. Fernando De la Torre, for his invaluable guidance, encouragement, and support throughout the course of my Master's research. I am also sincerely thankful for his thoughtful advice on major life decisions beyond my academic journey. I am further grateful to Prof. Shubham Tulsiani and Jianjin Xu for serving on my thesis committee and for their insightful feedback, which has greatly contributed to the improvement of this work.

I would like to extend my heartfelt thanks to Saswat Subhajyoti Mallick for his camaraderie and unwavering support during challenging times. I am also indebted to Dr. Bernhard Kerbl for his technical guidance and valuable assistance throughout the project. My sincere thanks go to Alex Amat, Marc Ruiz, and Francisco Vicente Carrasco for fostering a supportive and stimulating research environment, and for their collaboration and helpful discussions.

Finally, I am profoundly grateful to my family, especially my parents, for their unconditional love, patience, and unwavering support throughout this journey.

Contents

1 Introduction **1**

2 Related works **5**
- 2.1 Radiance Fields . . . 5
- 2.2 3DGS Methods . . . 6
- 2.3 3DGS for avatars . . . 6
- 2.4 NeRF streaming methods . . . 6
- 2.5 3DGS streaming methods . . . 7

3 Method **9**
- 3.1 System overview . . . 9
- 3.2 Dynamic optimization . . . 10
- 3.3 Static optimization . . . 14
  - 3.3.1 SH-only optimization . . . 14
  - 3.3.2 Precomputation . . . 14
  - 3.3.3 CUDA graphs . . . 14
  - 3.3.4 Load Balancing . . . 15
- 3.4 Loss . . . 15

4 BASKET-Multiview Dataset **17**

5 Experiments **21**
- 5.1 Datasets . . . 21
- 5.2 Baselines . . . 21
- 5.3 Metrics . . . 22
- 5.4 Evaluations . . . 23
- 5.5 Limitations . . . 25

6 Conclusion **29**

A Supplementary Materials **31**
- A.1 Per-scene evaluation results. . . . 31
- A.2 Ablations . . . 31
- A.3 BASKET-Multiview dataset . . . 32
  - A.3.1 Config sets . . . 32

A.3.2 Camera setup 38
A.3.3 Render settings 39
A.3.4 Data generating workflow 40

Bibliography **43**

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

1.1 **Task Overview.** Our method takes multi-view video streams of Human-Centered Live Events (HCLE) as input and reconstructs the dynamic 3D scene to allow viewers to watch the events from any novel view with interactive frame rates. 2

3.1 **System Architecture Overview.** Our method first segments multi-view images at t, to obtain per-subject segmentation masks for each subject, and passes the masked RGB images of each subject to the nodes. Each node will then optimize 3D Gaussians at t for that subject, using either dynamic or static optimization logic. The optimized 3D Gaussians are collected from each node and aggregated into a single model, and then served for on-demand renderings from arbitrary viewpoints to clients. 10
3.2 **Qualitative Gains by Skeleton Optimization.** 13

4.1 **BASKET-Multiview Dataset Multi-modal Data Example**. From left to right: segmentation mask, depth map, surface normal image, and RGB image. 17
4.2 **Lighting and Camera Configuration in the Court.** White light: point light. Red light: spotlight. Some lights are not visible in this view due to occlusion. 18

5.1 **Qualitative Results.** (a)-(d) are BASKET-Multiview core scenes; (e)-(g) are BASKET-Multiview development scenes; (h), (i) are CMU-Panoptic scenes; (j) is a BASKET-Multiview static scene. 26
5.2 **Video Qualitative Results.** 27

A.1 **Lighting Layout in the Court.** 36
A.2 **Sample Binding Pose Player.** 36
A.3 **Lighting Layout for the Lighting Change Sequence.** 37
A.4 **Camera Layout in the Court.** 38
A.5 **Camera Layout for Capturing the Binding Pose Pictures and Black Background Sequences.** 39
A.6 **Layer Rendering Illustration.** 40

List of Tables

4.1 **Comparison on available multi-view sports datasets.** 19

5.1 **Full evaluation on BASKET-Multiview core scenes and CMU-Panoptic dataset.** For CMU-Panoptic, the detected noisy 3D skeletons are used and the skeleton optimization is applied. For BASKET-Multiview, skeleton optimization is not always applicable because the cameras are too far in some scenes, so the perfect skeletons are used. 23

5.2 **Evaluation with different camera distances.** Attack 4 noisy skeletons are perturbed from perfect skeletons. Far views in Defense 2 result in no meaningful priors and hence are skipped from the evaluation. 24

5.3 **Evaluation under sudden lighting changes (LC).** 24

5.4 **Quality and speed evaluation of static scene optimization with lighting changes only.** 25

A.1 **CMU-Panoptic Per-scene Evaluation Results.** 32

A.2 **BASKET-Multiview Per-scene Evaluation Results.** 33

A.3 **Ablation Study of Skeleton Optimization.** Max ptb: max degrees of random perturbation. 34

A.4 **Ablation Study of Static Optimization Techniques.** 34

A.5 **BASKET-Multiview Core Scenes Composition.** 35

A.6 **BASKET-Multiview Development Scenes Composition.** . . . 35

Chapter 1

Introduction

"Panem et circenses" coined by Roman poet Juvenal to critize the self-serving nature of Romans who, deprived of political influence, are placated soley by "bread and circuses" provided by the government. Although technology has advanced significantly, we remain the same humans, governed by similar systems. Entertainment continues to be at the heart of societal engagement, illustrating that some aspects of human nature endure across millennia.

Nowadays, human-centered events (HCE) are a core element of modern visual media: sports events, concerts, and face-to-face debates are only a few examples of popular content, drawing the attention of billions of viewers. Witnessing or participating in these events *while they are unfolding* is an integral part of modern social life; the shared experiences create important connections and establish our culture. To allow for widespread participation, live transmissions of such HCE are ubiquitously available for remote, visual consumption in a 2D format. Unfortunately, this does not allow viewers to immerse themselves in the same way that personal attendance would: at best, television services may allow viewers to cycle through cameras at will to enable a basic level of engagement.

In 2001, Takeo Kanade and collaborators [39] developed the "EyeVision" technology for use during Super Bowl 2001. It involved an array of cameras mounted at the Raymond James Stadium, which provided a dynamic, panoramic view of the play. This system synthesized the inputs from multiple cameras into a single, flowing image, allowing viewers to see the action from virtually any angle. This

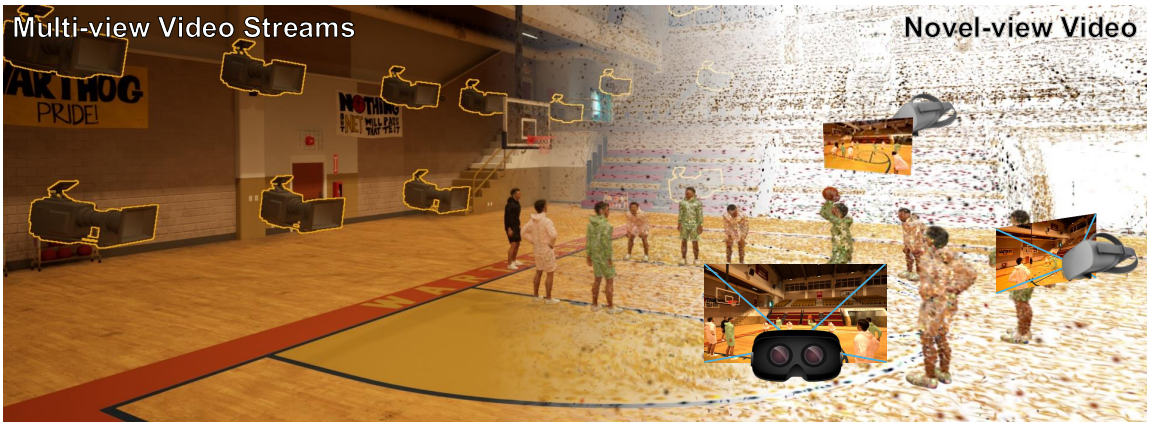

Figure 1.1: **Task Overview.** Our method takes multi-view video streams of Human-Centered Live Events (HCLE) as input and reconstructs the dynamic 3D scene to allow viewers to watch the events from any novel view with interactive frame rates.

innovation significantly enhanced the viewer's experience by offering detailed, 360-degree views of live action, which traditional single-point-of-view cameras could not capture. However, EyeVision did not allow real-time interaction, needed a lot of computing and was extremely costly to set up. Building on this, Intel's TrueView (formerly freeD) technology uses an array of high-resolution cameras around stadiums to create immersive 360-degree replays, allowing viewers to experience key moments from various angles. Spiideo's Multi-Angle Autocasting utilizes AI-powered cameras to capture multiple perspectives of sports events automatically, enabling seamless switching between angles without manual intervention.

Anticipating the next frontier of visual content consumption, this thesis lays the foundation to provide significant leaps forward for the remote experience of HCE, live events: Building on recent advances for high-performance radiance field representations, we describe a wholistic solution toward providing a freely explorable, photo-realistic 3D format for human-centric *live* events (HCLE). Our approach is based on fast-to-train radiance fields from 3D Gaussian Splatting [16] (3DGS) . By focusing on human-centered content, we can exploit strong geometric priors to cut reconstruction time, facilitate distributed and parallel processing, as well as load balancing to produce high-quality 3D reconstructions for live content at interactive rates. Specifically, we separate the job of reconstructing the full scene at a given timestamp into *per-subject* and *environment* optimization tasks.

For each subject, we can seed their high-quality reconstruction at each timestamp from easy-to-obtain pose information, and further refine geometry and appearance from photometric loss via differentiable 3DGS rendering. This work proposes a hierarchical *coarse-to-fine* approach that progressively resolves increasingly fine aspects of subjects' pose and appearance, which forms a lightning-fast solution that's robust to noise, for reconstructing 3D humanoid actors from multi-view images.

In addition to subjects in the event, a complete visual experience also requires reacting to changes in the environment. To limit the problem space and allow highly effective run-time optimization, this research focuses on geometrically static environments. In other words, we assume that given an initial reconstruction of an environment, only its appearance (i.e., view-dependent color) may change as actors perform live actions: this includes any changes in global illumination, shadows, or reflections as actors move about the scene. Enforcing this restriction allows for a highly streamlined render pipeline design that alleviates several of the complex—and time-consuming—aspects of 3DGS rendering. We show how this simple assumption enables us to eschew all dynamic resource management challenges and use high-level (e.g., ideal load balancing) and low-level (e.g., CUDA graphs) optimizations to maximize the efficacy of available hardware for training.

A significant challenge in this research is the lack of available data and standardization for live event capture with ground-truth data. To enable an in-depth exploration of our target setting despite this issue, we have designed a comprehensive, synthetic benchmark dataset, focusing on sports activities with multiple actors. Designed with professional 3D authoring and rendering tools, our dataset provides high visual fidelity and multimodal reference outputs for optimization, as well as options for simulating real-live error sources (e.g., in pose detection).

In summary, we provide the following contributions:

1. A scalable system design toward real-time reconstruction of HCLE via distributed and parallel processing.
2. A high-performance, coarse-to-fine solution for reconstructing subjects in the event.
3. A streamlined pipeline for appearance optimization with optimal load balancing.
4. A comprehensive benchmark dataset with multiple actors in HCE settings.

Chapter 2

Related works

2.1 Radiance Fields

Radiance fields are pivotal in computer graphics and vision, representing 3D scenes by modeling light distribution within a volume. Kajiya's rendering equation [14] established the foundation for simulating radiance transfer in scenes. Precomputed Radiance Transfer (PRT) [35] advanced real-time rendering by precomputing light interactions for dynamic lighting. Neural Radiance Fields (NeRF) [28] revolutionized the field, using neural networks to synthesize high-fidelity novel views of complex scenes. Extensions like D-NeRF [30] capture non-rigid deformations over time, while Time-of-Flight Radiance Fields (TöRF) [11] incorporate depth sensing for improved dynamic scene reconstruction.

Despite their dominance in scene reconstruction and novel view synthesis, NeRF and its variants are computationally intensive, requiring significant optimization for each new scene. Recently, 3D Gaussian Splatting (3DGS) [16] has emerged as a more efficient alternative, modeling radiance fields with spatially distributed Gaussian primitives.

2.2 3DGS Methods

Spacetime Gaussians [21] and Deformable 3D Gaussians [46] enhance 3DGS by incorporating temporal opacity, parametric motion, and annealing mechanisms to model dynamic scenes while addressing pose estimation inaccuracies. VideoRF [41] and 4D Gaussian Splatting [44] utilize space-time mappings and multi-resolution HexPlane modules, respectively, to compress temporal redundancies and capture motion and shape changes with deformation fields. SurMo [10] and Gaussian-Flow [23] model temporal dynamics using surface-based triplanes and dual-domain deformation models, to capture complex motions and deformations through frequency and polynomial fitting. SWinGS [34], Katsumata et al. [15], and DynMF [17] represent scene dynamics with sliding windows, Fourier approximations, and basis trajectories, enabling efficient and controllable motion synthesis. Street Gaussians [45] and DualGS [12] focus on human-centric dynamic scenes, leveraging tracked poses, and separate encodings for skeletal motion and surface appearance, with real-time rendering supported by entropy and codec-based compression.

2.3 3DGS for avatars

Several methods on driving human avatars such as GaussianAvatar [9], GoMAvatar [43] and SplattingAvatar [33] achieve high-quality renderings from monocular videos by combining explicit mesh representations with Gaussian primitives. Concurrently, approaches such as HumanGaussian [24], SimAvatar [20], and PSHuman [19] focus on text-driven 3D human generation and photorealistic reconstruction, leveraging diffusion models and cross-scale techniques. V^3 [42] and SqueezeMe [32] address the challenges of streaming volumetric videos on mobile devices and optimizing Gaussian avatars for VR applications, respectively.

2.4 NeRF streaming methods

StreamRF [18] uses an explicit grid-based model with incremental learning to reconstruct streaming radiance fields, updating each frame as a difference from a base

model, and optimizes only critical regions in each frame. ReRF [40] represents dynamic scenes by modeling inter-frame changes with a compact motion grid and residual feature grid, decoded by a lightweight MLP. NeRFPlayer [36] enables efficient streamable representation of dynamic scenes by decomposing them into static, deformable, and unseen areas, each modeled by separate neural fields, while employing a time-dependent sliding window for feature streaming.

2.5 3DGS streaming methods

Dynamic-3DGS [26] (D-3DGS) models dynamic scenes with Gaussians with persistent geometry attributes that can orient freely, and enforcing local rigidity for spatial consistency. 3DGStream [38] uses a neural transformation cache (NTC) for Gaussian transformation and a refinement stage for detail reconstruction. HiCoM [7] adopts hierarchical voxel-based motion modeling, perturbation smoothing, and Gaussian merging for compact, streamable representations. DASS employs Gaussian inheritance, motion-aware alignment, and densification via optimization errors for iterative refinement. QUEEN [8] encodes Gaussian attributes with quantized residuals, sparsifies position data, and disentangles static and dynamic content using viewspace gradients.

Chapter 3

Method

Our aim is to achieve real-time 3D reconstruction of human-centered events from multi-view streaming videos, to enable unrestricted 6-DOF exploration, providing a uniquely interactive experience for (remote) audiences. However, delivering such up-to-date visuals poses a massive engineering challenge, requiring reconstructions to happen thousands of times faster than traditional methods.

To tackle this challenge, we model the reconstruction process using a two-level MapReduce [5] approach and parallelize it across GPU-powered nodes . Nodes in a node-pool process individual frames independently, while GPUs within a node handle the reconstruction of dynamic elements-such as players, the court, or the ball-in parallel. This design allows interactive frame rates, which improve linearly with increase in computational resources.

3.1 System overview

Figure 3.1 provides an overview of the method. At time t, we map the last N unprocessed frames (from t to $t - N$) to each of the N nodes. For each frame, we generate multi-view masks for all subjects using off-the-shelf models (SAM2 [31]) and assign each subject to a dedicated GPU for reconstruction.

Each node optimizes 3D Gaussians at frame t, with a subject assigned to each GPU, employing either dynamic or static optimization, which we detail in Sections 3.2 and 3.3, respectively. After completing the 3D reconstruction for all subjects in the

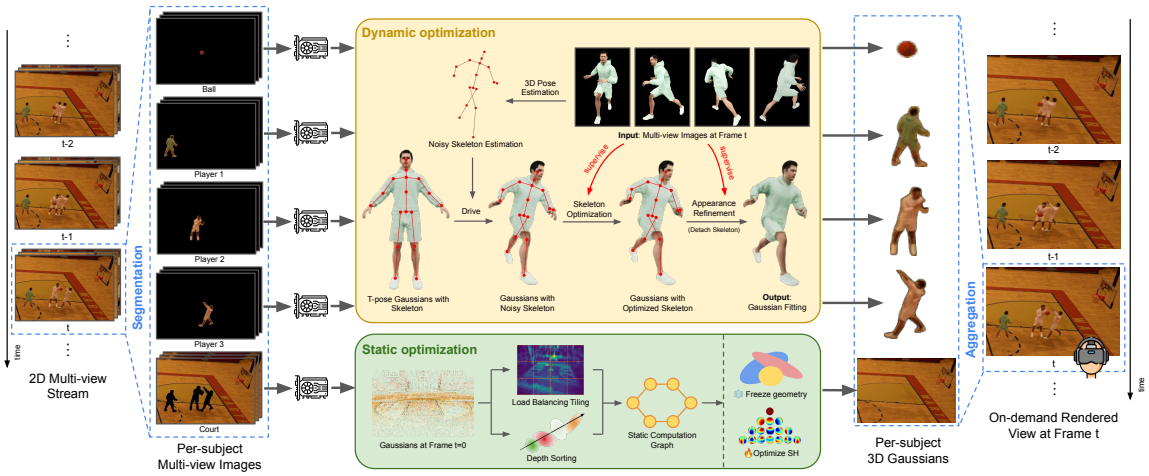

Figure 3.1: **System Architecture Overview.** Our method first segments multi-view images at t, to obtain per-subject segmentation masks for each subject, and passes the masked RGB images of each subject to the nodes. Each node will then optimize 3D Gaussians at t for that subject, using either dynamic or static optimization logic. The optimized 3D Gaussians are collected from each node and aggregated into a single model, and then served for on-demand renderings from arbitrary viewpoints to clients.

frame, the optimized 3D Gaussians from each node are aggregated into a unified model. Our method prioritizes reconstruction in a manner that novel-view synthesis at high-fidelity is possible with just one optimization cycle, thereby amortizing on-demand rendering from arbitrary viewpoints for connected viewers.

A key design feature of our method that enables nearly unbounded scalability is that **scene reconstruction at frame t operates completely independently of other frames**. This added flexibility in distributed processing allows us to *multiplex* the reconstruction of multiple adjacent frames, fully leveraging the available distributed computing resources for real-time 3D reconstruction.

3.2 Dynamic optimization

Incremental frame optimization based on a robust prior is fundamental to several recent methods [7, 26, 34, 38] for dynamic scene reconstruction using Gaussians. However, methods often lose structural detail over time because relying on rigid/no priors accumulates errors, which degrade fine details during motion. To overcome

this limitation, we introduce a novel dynamic coarse-to-fine optimization for humans, capable of moving 3D Gaussians over long distances while maintaining fine-grained details.

In HCE, modeling human body motion relies on high-quality priors, such as 3D human skeletons, to guide the spatial transformations of the 3D Gaussians. We define two key concepts: *skeleton* and *skinning weights*. A *skeleton* is the kinematic tree structure of human body where each node represents a joint, characterized by properties such as global position $\mathbf{x} \in \mathbb{R}^3$, global rotation quaternion $\mathbf{q} \in \mathbb{H}$ in world coordinates, and a pointer p to its parent node. *Skinning weights* are values assigned to mesh vertices, specifying the influence of each joint on the vertex, allowing for realistic deformations as the joints rotate.

Our method processes multi-view RGB images per frame to produce detailed, temporally consistent 3D Gaussian-based surface reconstructions. Initialization involves optimizing vanilla 3DGS [16] using multi-view images of a human in a T-pose (binding pose), to generate a detailed 3DGS reconstruction. Following [2, 3], we fit an SMPL [25] model to the converged Gaussian model of the T-pose, by minimizing the chamfer loss between Gaussian centers and SMPL mesh vertices, yielding the final mesh, skeleton, and linear blend skinning (LBS) weights. Each Gaussian is assigned to its nearest mesh vertex and inherits its skinning weights, enabling skeleton-driven manipulation.

For dynamic frames, OpenPose [4] is used to extract multi-view 2D skeletons for frame t, which are then triangulated to compute the corresponding 3D skeleton. Gaussians are transformed from the T-pose to the pose at frame t using this derived 3D skeleton. If noise in the posed skeleton leads to misplacement of Gaussians, skeleton optimization is employed to refine their positions. Finally, the Gaussians are decoupled from the skeleton, allowing their parameters to be freely optimized to enhance visual quality.

Skeleton-driven Gaussians. Given the Gaussians $\mathcal{G}_0$ and skeleton $\mathcal{S}_0$ at T-Pose, and posed skeleton $\mathcal{S}_t$ at frame t, one can transform the Gaussian positions and

quaternions from the T-pose to the pose t by:

$$\begin{aligned} \mathbf{x}_{i,t} &= \sum_{j=1}^{|\mathcal{S}_0|} w_{i,j} \left[\mathbf{q}_{j,t} \cdot \mathbf{q}_{j,0}^{*} \cdot (\mathbf{x}_{i,0} - \mathbf{x}_{j,0}) \cdot \mathbf{q}_{j,0} \cdot \mathbf{q}_{j,t}^{*} + \mathbf{x}_{j,t} \right] \\ \mathbf{q}_{i,t} &= \sum_{j=1}^{|\mathcal{S}_0|} w_{i,j} \ \mathbf{q}_{j,t} \cdot \mathbf{q}_{j,0}^{*} \cdot \mathbf{q}_{i,0} \end{aligned} \tag{3.1}$$

where $\mathbf{x}_{i,t}$, $\mathbf{q}_{i,t}$ are the position and quaternion of i^{th} Gaussian at frame t. $\mathbf{x}_{j,t}$, $\mathbf{q}_{j,t}$ are the position and quaternion of j^{th} joint at frame t. It is worth noting that subscript i is the index of Gaussians, and subscript j is that of joints. $w_{i,j}$ is the skinning weight of the i^{th} Gaussian w.r.t. the j^{th} joint.

Skeleton optimization. Skeleton-driven Gaussians assume accurate skeletons, but triangulated skeletons from OpenPose often suffer from noise due to occlusions and resolution limits, leading to suboptimal Gaussian initialization. To mitigate this, we introduce a skeleton optimization algorithm (Algorithm 1), that refines the 3D skeleton by rasterizing transformed Gaussians and minimizing 2D photometric loss against the ground truth image.

In Algorithm 1, drive$(\cdot)$ is the skeleton-driven Gaussian transformation explained by Equation 3.1; rasterize$(\cdot)$ is the Gaussian rasterizing function (implementation adopted from [27]). During the skeleton optimization, all the Gaussians are bound to the skeleton, to ensure their movement aligns with joint motions. Only the joint rotations are updated at each step, while Gaussian opacity, scale, and spherical harmonics (SH) are held constant.

Appearance refinement. Once skeleton optimization converges, the Gaussians are detached from the skeleton, and their parameters are optimized using the photometric loss $\mathcal{L}_{\text{photo}}$, to enhance local structural details and adapt to changes in illumination.

To enhance speed and robustness, skeleton optimization optimizes global joint rotations in the world coordinate system instead of local rotations relative to parent joints, simplifying the computational graph. During appearance refinement, Gaussians, now independent of the skeleton, move freely. To prevent them from drifting into the background due to imperfect player masks, random backgrounds are applied to both the ground truth and rasterizer, penalizing Gaussians that move outside the player's

Algorithm 1 Skeleton Optimization

Input: A list of ground truth multi-view images $\{I_i^*\}$ of the player at frame t along with their corresponding camera poses $\{\text{cam}_i\}$, T-pose Gaussians $\mathcal{G}_0$, T-pose skeleton $\mathcal{S}_0$, frame t skeleton $\mathcal{S}_t$
Output: Skeleton-optimized 3D Gaussians $\mathcal{G}_t$ of the player
function OPTIMIZESKELETON($\{I_i^*\}$, $\{\text{cam}_i\}$, $\mathcal{G}_0$, $\mathcal{S}_0$, $\mathcal{S}_t$)
    for iter = 1 to maxIters **do**
        $\mathcal{G}_t \leftarrow \text{drive}(\mathcal{G}_0, \mathcal{S}_0, \mathcal{S}_t)$
        $i \leftarrow \text{iter} \ \% \ \text{numCams}$
        $I_i \leftarrow \text{rasterize}(\mathcal{G}_t, \text{cam}_i)$
        $\mathcal{L} \leftarrow \mathcal{L}_{\text{photo}}(I_i, I_i^*)$
        $\nabla \mathcal{S}_t \leftarrow \nabla_{\mathcal{S}_t} \mathcal{L}$
        $\mathcal{S}_t \leftarrow \mathcal{S}_t - \eta \nabla \mathcal{S}_t$
    end for
    $\mathcal{G}_t \leftarrow \text{drive}(\mathcal{G}_0, \mathcal{S}_0, \mathcal{S}_t)$
    return $\mathcal{G}_t$
end function

body.

Figure 3.2 demonstrates the qualitative gains by skeleton optimization, and detailed ablation study of our skeleton optimization are provided in the supplemental.

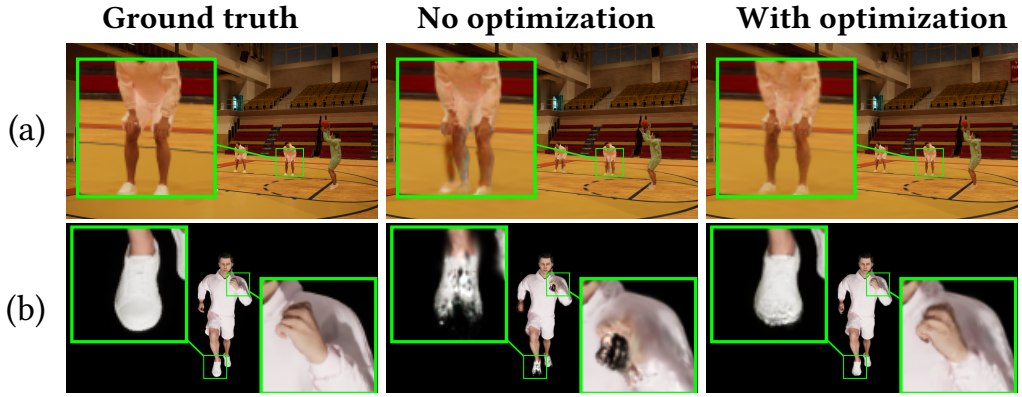

Figure 3.2: **Qualitative Gains by Skeleton Optimization.**

3.3 Static optimization

In most sports, the only significant source of change in the scene arises from the dynamic movements of human participants (players) and the ball. The rest of the environment remains static, with variations limited to global lighting and shadow dynamics influenced by these moving elements.

3.3.1 SH-only optimization

Following [26], each Gaussian provides a soft representation of physical space and can be frozen once the geometry of the scene is accurately captured using vanilla 3DGS on the initial frame. Subsequent optimization focuses exclusively on appearance (SH parameters) to account for lighting changes introduced by dynamic elements. This reduces computational overhead by excluding non-appearance parameters from gradient calculations, minimizing atomic operations during the backward pass, and significantly decreasing training time. Furthermore, decoupling geometry from appearance helps prevent artifacts such as flickering, which can occur when Gaussians shift, resize, or rotate.

3.3.2 Precomputation

Our empirical analysis reveals that depth-sorting of Gaussians is the primary bottleneck in rasterization. For Gaussians associated with static regions, their geometrical parameters (position, scale, rotation) are frozen, making their depth values invariant and cacheable. To address this, we implement a precomputed method that stores the sorted Gaussians for each rasterizer tile. These precomputed values are reused during the optimization of appearance changes in subsequent frames. This precomputation is performed after the optimization of the first frame has converged.

3.3.3 CUDA graphs

The precomputed structures reside in the VRAM which eliminate the expensive synchronization procedures between GPU and CPU. This gives way to implementing a purely CUDA graph-based [1] solution devoid of any CPU routines or system calls.

A CUDA graph is a software rendition that records GPU operations into a Directed Acyclic Graph (DAG) that can be used for asynchronous replays, minus the CPU overheads. Each replay reuses prerecorded execution paths and memory addresses, which removes any final unattended system-level constructs from limiting the speed of our method.

3.3.4 Load Balancing

Due to the non-deterministic nature of 3DGS, the number of Gaussians rendered per thread can vary significantly. To balance the workload across threads, we introduce a load-balancing step after precomputation. For tile i, the ordered set of Gaussians $\mathcal{G}^i = \{g_1, g_2, \ldots, g_L\}$ is partitioned by order into M subtiles, each containing a maximum of N Gaussians ($M = \lceil L/N \rceil$). We have $\mathcal{G}^i = \bigcup_{j=1}^{M} \mathcal{B}_j^i$, where $\mathcal{B}_j^i$ is the set of Gaussians in subtile j with $|\mathcal{B}_j^i| \leq N$. Each $\mathcal{B}_j^i$ is processed by a separate thread block to maximize GPU occupancy.

Since only SH coefficients are optimized, transmittance values can be precomputed during the first frame. This allows independent blending within each subtile without requiring synchronization between thread blocks. Consequently, subtile-distributed alpha blending for pixel p can be expressed as:

$$\mathbf{I}(p) = \sum_{j=1}^{M} T_j^i(p) \cdot \sum_{g_k \in \mathcal{B}_j^i} \prod_{m=1}^{k-1} (1 - \alpha_m(p)) \cdot \alpha_k(p) \cdot \mathbf{C}_k(p) \tag{3.2}$$

where $T_j^i(p)$ is the transmittance of subtile j at pixel p; $\alpha_k(p), \mathbf{C}_k(p)$ are the contributions of opacity and color of Gaussian k at pixel p. A representative flow for the static optimization and an ablation for each of the aforementioned optimizations is attached in the supplementary.

3.4 Loss

We adopt the photometric loss function in 3DGS [16]:

$$\mathcal{L}_{\text{photo}}(I, I^*) = (1 - \lambda)\mathcal{L}_1(I, I^*) + \lambda\mathcal{L}_{\text{D-SSIM}}(I, I^*) + R \tag{3.3}$$

for skeleton optimization, where I and I^* are the predicted and ground truth images, $\lambda = 0.2$. For the static optimization and appearance refinement in the dynamic optimization, only the subject-masked area is used to compute loss to avoid problems from occlusion among subjects: $\mathcal{L}_{\text{masked}} = \mathcal{L}_{\text{photo}}(MI, MI^*)$, where M is the subject mask. R is used to regularize Gaussian movements and scales, and is expressed as:

$$R = \lambda_x ||\mathbf{x} - \mathbf{x}^*|| + \lambda_s ||\mathbf{s} - \mathbf{s}^*||, \tag{3.4}$$

where $\mathbf{x}, \mathbf{x}^*, \mathbf{s}, \mathbf{s}^*$ are the optimized and initial Gaussian positions and scales. This regularization limits the geometrical changes to a Gaussian, thereby arresting any chances of flicker.

Chapter 4

BASKET-Multiview Dataset

We introduce the BASKET (Basketball Synthetic benchmarK for Enhanced Telepresence)-Multiview Dataset, a synthetic collection of scenarios representing common basketball plays generated using Unreal Engine 5 [6] and EasySynth [47], leveraging the basketball court assets provided by [37] and player models from [29]. For each scene, we provide comprehensive annotations that include calibrated cameras parameters, animations, RGB images, segmentation masks, depth maps, surface normal images and animations, as illustrated in Figure 4.1. All scenes are rendered at 1080p and 30 fps, with the exception of sequence *Attack 4*, which has been rendered in 4K resolution. The lighting and camera configuration in the court is illustrated in Figure 4.2.

Figure 4.1: **BASKET-Multiview Dataset Multi-modal Data Example**. From left to right: segmentation mask, depth map, surface normal image, and RGB image.

Figure 4.2: **Lighting and Camera Configuration in the Court.** White light: point light. Red light: spotlight. Some lights are not visible in this view due to occlusion.

The dataset is divided into two partitions: *Core* and *Development*. The *Core* partition contains 7 scenes designed for evaluating reconstruction methods for HCLE. Each *Core* scene is created within a basketball court environment, consisting of an 83-camera setup, optimized for capturing game-play during matches. More details on lighting and camera configurations are provided in the supplementary. The *Development* partition contains 9 simpler sequences containing varying lighting conditions and backgrounds. *Development* sequences are designed to isolate features such as lighting dynamics, complex movements, close/far camera settings, and more, in order to test the capabilities of the method during development. The full list of scenes and their detailed specifications is provided in the supplementary.

Dataset Name	Nature	Cameras	Annotations			
			Semantic Mask	Depth Map	Normal Map	Animation
TeamTrack	Real	3	✗	✗	✗	✗
SoccerNet	Real	1	✗	✗	✗	✗
SportsMOT	Real	1	✗	✗	✗	✗
KTH Multiview Football II	Real	3	✗	✗	✗	✗
FineSports	Real	1	✗	✗	✗	✗
APIDIS	Real	7	✗	✗	✗	✗
EPFL Basketball	Real	4	✗	✗	✗	✗
SoccerNet-Depth	Synthetic	1	✗	✓	✗	✗
Soccer on your tabletop	Synthetic	1	✗	✓	✗	✗
BASKET-Multiview	Synthetic	83	✓	✓	✓	✓

Table 4.1: **Comparison on available multi-view sports datasets.**

Chapter 5

Experiments

This section describes the experiments we carry out for benchmarking the performance of our method. We post the results of our ablation studies on skeleton optimization and static optimization speed-up techniques in the accompanying supplemental.

5.1 Datasets

We performed experiments on the BASKET-Multiview (described in Section 4) and the CMU-Panoptic dataset [13]. For the CMU-Panoptic dataset, we selected three subsequences from the *171204_pose1* sequence and conduct experiments on these.

To evaluate the robustness of our method against imperfect priors, we generate skeletons, meshes, and skinning weights using OpenPose [4] and SMPL [25], and further introduce random perturbations to the skeletons (referred to as *noisy skl*). We mimic different noise levels in the skeletons by randomly perturbing the limb joints (shoulders, elbows, hips, and knees) by 10° to 20°. These scenarios provide a controlled testbed to validate the ability of our skeleton optimization method to correct errors introduced by traditional off-the-shelf models.

5.2 Baselines

We compared our method against all known state-of-the-art approaches that perform online training and have publicly available implementations, to the best of

our knowledge. We consider 3DGS-based 3DGStream [38](CVPR'24 Highlight), HiCoM [7](Neurips'24), and NeRF-based StreamRF [18](Neurips'22). Additionally, we included Dynamic-3DGS (D-3DGS) [26](3DV'24) as a baseline, prioritizing quality despite its lack of real-time training capabilities. In our experiments, we adhered to the recommended settings for each method wherever possible, adjusting hyperparameters only when necessary to achieve optimal results.

All experiments are conducted on an Nvidia RTX A4500 GPU, with training and evaluation performed at a resolution of 960×540. Our method optimizes each scene component for 500 iterations, with the initial 60 iterations dedicated to skeleton optimization, if applicable. Since the preprocessing requirements vary across methods, we follow the protocol in [26, 38] to ensure a fair comparison by reporting only the training time.

5.3 Metrics

We evaluated the per-frame reconstruction quality using established image-based metrics: PSNR, SSIM, and LPIPS. Additionally, we introduced masked PSNR (M-PSNR), which calculates the PSNR exclusively in dynamic regions to more accurately assess image quality in motion-intensive areas. The image metrics were averaged across all frames and views of each sequence.

For video evaluation, we adopted VMAF [22], a widely used metric that integrates spatial and temporal quality measures, to account for multi-resolution quality and temporal coherence. We reported the VMAF averaged across all views of each sequence. The averaged training time (in seconds) per frame (SPF) was also reported as a measure of computational cost. Since our approach processed each scene component independently in parallel, we reported the largest SPF across all components. Furthermore, we introduced VMAF efficiency (VE) and PSNR efficiency (PE), defined as $\text{VE} = \frac{\text{VMAF}}{\text{SPF}}$ and $\text{PE} = \frac{\text{PSNR}}{\text{SPF}}$, respectively, to highlight the quality achieved per unit time for each method.

5.4 Evaluations

Table 5.1 shows the evaluation results on BASKET-Multiview and the CMU-Panoptic dataset. Figure 5.1 houses all qualitative results and Figure 5.2 illustrates results on consecutive frames. From Table 5.1, we can see that our method outperforms other methods in terms of speed and quality on BASKET-Multiview. On CMU-Panoptic, our method aces in both video and image quality as well as reconstruction speed. This is particularly noteworthy given the imperfect segmentation masks and noisy priors in the dataset—conditions under which most methods degrade significantly—highlighting our method's robustness and minimal reliance on perfect priors.

Dataset	Method	VMAF↑	PSNR↑	M-PSNR ↑	SSIM↑	LPIPS↓	SPF↓	VE↑	PE↑
BASKET-Multiview	StreamRF	36.30 ± 2.47	26.82 ± 0.42	13.45 ± 1.20	0.756 ± 0.040	0.198 ± 0.028	48.6 ± 0.5	0.75 ± 0.06	0.55 ± 0.01
	3DGStream	51.43 ± 6.05	28.36 ± 0.90	20.79 ± 1.52	0.875 ± 0.019	0.174 ± 0.024	12.6 ± 0.7	4.10 ± 0.43	2.26 ± 0.16
	D-3DGS	65.41 ± 0.87	28.75 ± 0.85	23.20 ± 1.64	0.901 ± 0.003	0.149 ± 0.009	74.3 ± 3.5	0.88 ± 0.05	0.39 ± 0.01
	HiCoM	54.69 ± 10.61	28.08 ± 2.71	19.03 ± 1.26	0.888 ± 0.060	0.157 ± 0.077	6.1 ± 0.1	8.89 ± 1.65	4.57 ± 0.40
	Ours	67.30 ± 2.41	29.75 ± 0.21	25.24 ± 0.86	0.912 ± 0.007	0.119 ± 0.006	5.2 ± 0.1	13.02 ± 0.67	5.75 ± 0.12
CMU-Panoptic	StreamRF	34.49 ± 2.93	23.46 ± 0.34	18.20 ± 1.09	0.852 ± 0.057	0.404 ± 0.042	9.0 ± 0.0	3.83 ± 0.33	2.61 ± 0.04
	3DGStream	46.00 ± 6.03	25.68 ± 1.28	21.37 ± 1.79	0.902 ± 0.011	0.338 ± 0.008	5.0 ± 0.0	9.20 ± 1.21	5.13 ± 0.26
	D-3DGS	50.66 ± 4.30	26.41 ± 0.24	24.60 ± 0.94	0.908 ± 0.008	0.286 ± 0.006	88.3 ± 3.5	0.57 ± 0.04	0.30 ± 0.01
	HiCoM	38.37 ± 4.10	25.35 ± 0.93	21.85 ± 1.83	0.904 ± 0.009	0.301 ± 0.009	4.0 ± 0.0	9.60 ± 1.03	6.34 ± 0.23
	Ours	57.33 ± 3.31	26.53 ± 1.27	26.55 ± 1.22	0.879 ± 0.009	0.274 ± 0.008	4.7 ± 0.0	12.20 ± 0.70	5.64 ± 0.27

Table 5.1: **Full evaluation on BASKET-Multiview core scenes and CMU-Panoptic dataset.** For CMU-Panoptic, the detected noisy 3D skeletons are used and the skeleton optimization is applied. For BASKET-Multiview, skeleton optimization is not always applicable because the cameras are too far in some scenes, so the perfect skeletons are used.

In the BASKET-Multiview dataset, three camera views (close, mid, and far) simulate court-side and spectator seat placements by varying focal lengths and camera-to-player distances. From Table 5.2 and Figure 5.1 (a), (d), and (e), we observe that our method achieves the best quality and convergence speed in mid views, excelling even with noisy skeletons. In close views, it maintains high reconstruction quality with minor speed trade-offs. For far views, our method significantly outperforms others in both quality and speed, though skeleton detection and optimization become unreliable at extreme distances.

In real-world applications, lighting changes can come from difference in illumination between the animated player and the T-pose reference, or due to the dynamic, unpredictable movements of players. We evaluate our method's ability to handle these lighting variations using the Running Player sequence with and without light

Sequence	Dribbling Player - close view					Attack 4 - mid view					Defense 2 - far view				
Method — Metric	VMAF↑	PSNR↑	M-PSNR↑	SPF↓	VE↑	VMAF↑	PSNR↑	M-PSNR↑	SPF↓	VE↑	VMAF↑	PSNR↑	M-PSNR↑	SPF↓	VE↑
3DGStream	36.08	30.54	19.73	3	12.03	57.01	26.75	23.63	14	4.07	56.57	29.19	19.81	12.3	4.60
D-3DGS	55.84	31.03	19.41	54	1.03	65.37	26.82	24.33	67	0.98	65.25	29.18	23.06	76.0	0.86
HiCoM	33.24	28.64	18.24	3	11.08	31.47	22.03	18.65	6	5.25	55.08	28.59	17.81	6.1	9.03
Ours (perfect skl)	76.16	35.85	25.74	3.8	20.04	72.68	30.17	25.60	5	14.54	66.31	29.75	26.15	5.2	12.75
Ours (noisy skl)	62.13	35.72	25.60	5.8	10.71	72.67	30.17	25.60	5.7	12.75	-	-	-	-	-

Table 5.2: **Evaluation with different camera distances.** Attack 4 noisy skeletons are perturbed from perfect skeletons. Far views in Defense 2 result in no meaningful priors and hence are skipped from the evaluation.

changes. Results are reported in Table 5.3 and illustrated in Figure 5.1 (e) (f). The results show that our method outperforms all baselines in quality, even with noisy priors while maintaining comparable training speeds, which demonstrates the effectiveness of our skeleton optimization.

LC	Method	VMAF↑	PSNR↑	SSIM↑	LPIPS↓	SPF↓	VE↑
✗	3DGStream	32.61	29.89	0.973	0.037	2	16.31
	D-3DGS	56.12	31.60	0.976	0.024	57	0.98
	HiCoM	31.03	27.22	0.965	0.038	3	10.34
	Ours (perfect skl)	82.52	39.43	0.990	0.008	3.8	21.72
	Ours (noisy skl)	70.80	35.22	0.987	0.013	5.8	12.21
✓	3DGStream	25.81	30.33	0.971	0.041	2	12.91
	D-3DGS	39.25	29.31	0.972	0.034	57	0.69
	HiCoM	16.31	27.44	0.963	0.044	3	5.44
	Ours (perfect skl)	77.59	38.95	0.990	0.012	3.8	20.42
	Ours (noisy skl)	63.61	35.23	0.984	0.018	5.8	10.97

Table 5.3: **Evaluation under sudden lighting changes (LC).**

In Section 3.3, we propose several quality-preserving, performance optimizations for static regions under lighting changes, which we test using three such scenes. As shown in Table 5.4, we achieve significant reduction in SPF with comparable VMAF and PSNR against other methods. Since different methods require different numbers of iterations, in addition to the quality and performance metrics, we report the milliseconds taken per iteration (ms/iter) to better enunciate the speedup of our optimizations.

Scene	Method	VMAF↑	PSNR↑	SPF↓	VE↑	PE↑	ms/iter
Day loop	3DGStream	30.42	23.84	10	3.04	2.38	41
	D-3DGS	62.19	27.16	29	2.14	0.94	29
	HiCoM	40.74	24.85	7	5.82	3.55	35
	Ours	53.31	26.69	2	26.66	13.35	18
Opera	3DGStream	32.71	21.14	11	2.97	1.92	47
	D-3DGS	50.97	24.47	29	1.76	0.84	29
	HiCoM	42.28	22.89	7	6.04	3.27	35
	Ours	51.79	25.16	2	25.90	12.58	18
Factory	3DGStream	11.61	27.91	9	1.29	3.10	36
	D-3DGS	45.94	28.62	31	1.48	0.92	32
	HiCoM	35.49	28.61	6	5.92	4.77	28
	Ours	41.61	28.82	2	20.81	14.41	13

Table 5.4: **Quality and speed evaluation of static scene optimization with lighting changes only.**

5.5 Limitations

While the proposed framework achieves state-of-the-art performance, several limitations remain that present opportunities for future improvement. First, the method depends on multi-view consistent 2D instance segmentation masks as priors, a challenging open problem that continues to be actively explored. Second, although the framework demonstrates strong results on the BASKET-Multiview and CMU-Panoptic datasets, the range of human motions captured in these datasets is relatively limited. Future work will involve applying the method to more complex human interactions involving physical contact, such as wrestling and boxing, to further assess its robustness. Third, the framework heavily relies on human-specific priors and has only been validated on datasets featuring human motion. Extending the concept of binding Gaussians to skeletons of general objects beyond humans is a promising direction for future research. Finally, although the BASKET-Multiview dataset offers a strong benchmark with highly accurate ground-truth annotations, its synthetic nature may not fully capture the complexities encountered in real-world environments. Therefore, further validation on diverse and real-world datasets is essential to fully establish the framework's generalizability.

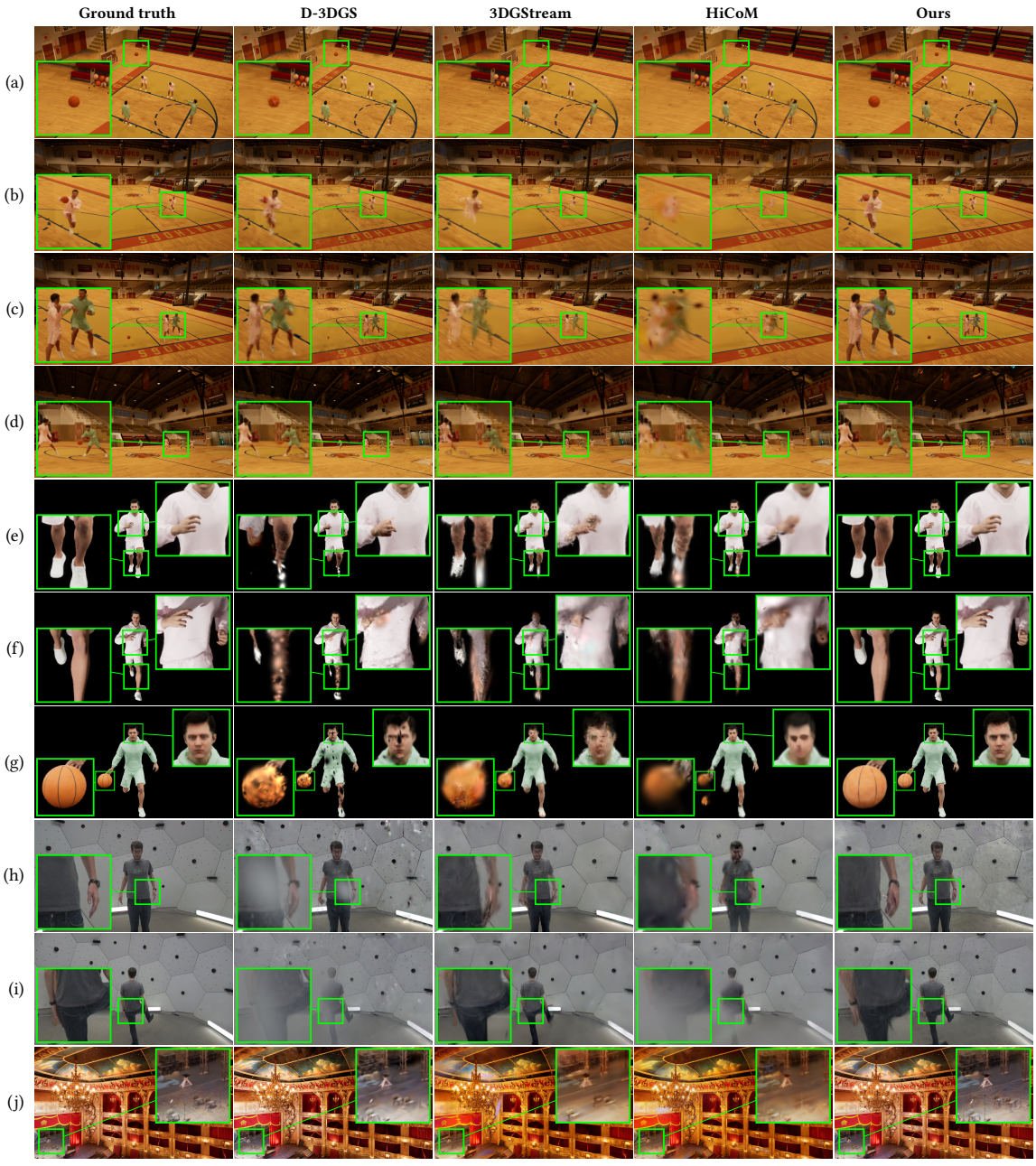

Figure 5.1: **Qualitative Results.** (a)-(d) are BASKET-Multiview core scenes; (e)-(g) are BASKET-Multiview development scenes; (h), (i) are CMU-Panoptic scenes; (j) is a BASKET-Multiview static scene.

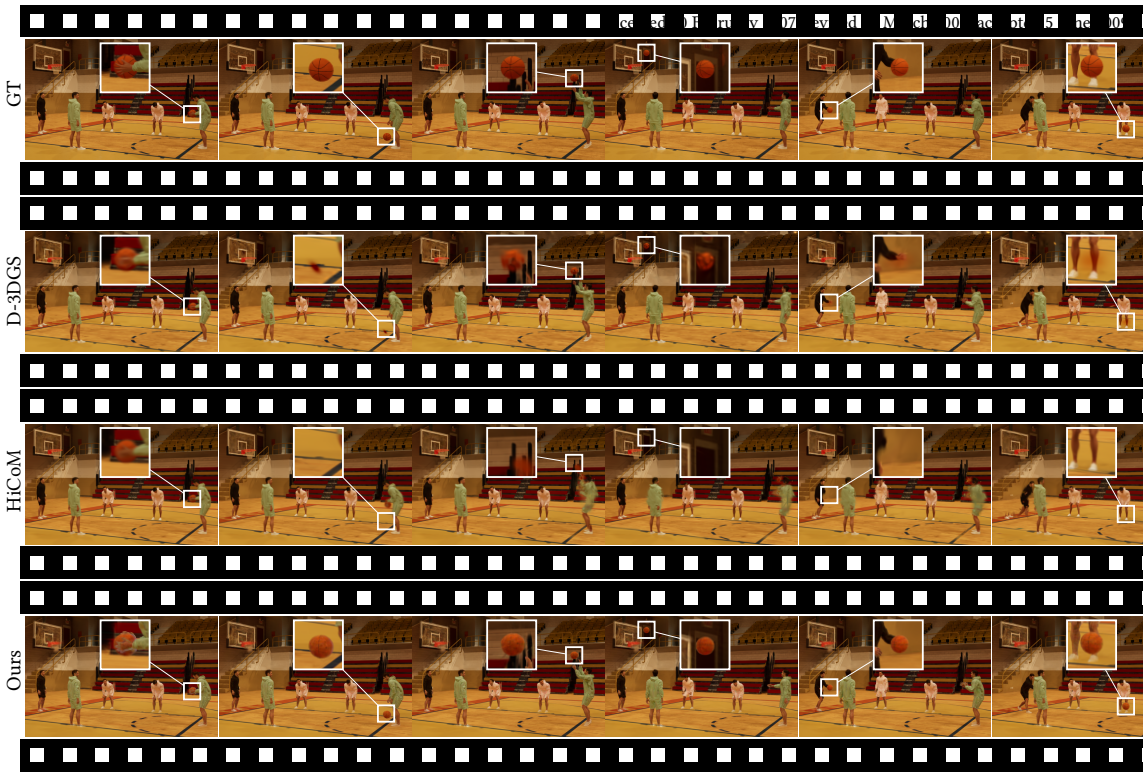

Figure 5.2: **Video Qualitative Results.**

Chapter 6

Conclusion

We propose a framework towards real-time, photo-realistic 3D reconstruction of HCLE, using 3DGS and distributed optimization. By computationally factorizing subjects and environment reconstruction, and employing an hierarchical optimization strategy with performance enhancements to the rasterization pipeline, we achieve scalable, high fidelity results at interactive frame rates. Our BASKET-Multiview dataset, establishes a benchmark for evaluating methods in complex dynamic scenarios. Experiments demonstrate our approach surpasses SOTA methods in quality, robustness, and efficiency, setting the stage for immersive, real-time live event streaming that bridges physical and digital experiences.

Appendix A

Supplementary Materials

A.1 Per-scene evaluation results.

We report the quality and performance metrics per scene on CMU-Panoptic dataset in Table A.1 and BASKET-Multiview in Table A.2.

A.2 Ablations

To evaluate skeleton optimization with noisy priors, we mimic different noise levels in the skeletons by randomly perturbing the limb joints (shoulders, elbows, hips, and knees) by maximum 10° or 20°, and then compare the reconstruction quality with and without skeleton optimization. We choose two scenes: Running Player (with close views) and Attack 4 (with mid views) to experiment on. To quantify the skeleton quality, we report the skeleton error (SE), computed by $\text{SE} = \frac{1}{N}\sum_{i=1}^{N}||\mathbf{x}_i - \mathbf{x}_i^*||$, where $\mathbf{x}_i, \mathbf{x}_i^*$ are the optimized and ground truth joint positions respectively, and N is the number of joints in the skeleton. From the results in Table A.3, we can see that the skeleton optimization can effectively reduce the SE and improve the video quality under various noise levels and camera settings.

We evaluate the speed-up contributions of techniques in our static optimization method: optimizing SH parameters (Only SH), precomputing Gaussian sortings (Precomp), CUDA computation graph (Graph), and load balancing (LB) through an

Scene	Method	VMAF↑	PSNR↑	M-PSNR↑	SSIM↑	LPIPS↓	SPF↓	VE↑	PE↑
Scene 1	StreamRF	37.86	23.82	19.31	0.819	0.412	9.0	4.21	2.65
	3DGStream	50.69	26.25	22.58	0.909	0.331	5.0	10.14	5.25
	D-3DGS	53.84	26.69	25.51	0.914	0.281	88.0	0.61	0.30
	HiCoM	42.58	26.08	23.42	0.909	0.297	4.0	10.65	6.52
	Ours	53.80	25.58	25.89	0.876	0.274	4.7	11.45	5.44
Scene 2	StreamRF	33.10	23.39	17.14	0.819	0.359	9.0	3.68	2.60
	3DGStream	39.19	24.21	19.31	0.889	0.347	5.0	7.84	4.84
	D-3DGS	52.38	26.23	23.64	0.899	0.292	92.0	0.57	0.29
	HiCoM	34.38	24.31	19.84	0.894	0.311	4.0	8.60	6.08
	Ours	57.82	26.03	25.81	0.873	0.282	4.7	12.30	5.54
Scene 3	StreamRF	32.51	23.16	18.14	0.917	0.441	9.0	3.61	2.57
	3DGStream	48.11	26.57	22.22	0.908	0.335	5.0	9.62	5.31
	D-3DGS	45.77	26.32	24.64	0.912	0.284	85.0	0.54	0.31
	HiCoM	38.15	25.67	22.28	0.909	0.295	4.0	9.54	6.42
	Ours	60.36	27.97	27.96	0.889	0.267	4.7	12.84	5.95

Table A.1: **CMU-Panoptic Per-scene Evaluation Results.**

ablation study on the empty stadium scene with SH1 and SH3. Table A.4 shows that each technique provides substantial speed improvements for both.

A.3 BASKET-Multiview dataset

The full dataset specifications are shown in Table A.5 and Table A.6. Our dataset does not simulate any crowd dynamics among the audience.

A.3.1 Config sets

We use several illumination setups (denoted by *Config set* in A.6) to better simulate environments for diverse benchmarking of our method.

1. Set 1: The map consists of a basketball court with the following illumination setup. (refer A.3)

   Point lights: 46 point lights are strategically placed through the scene to simulate localized lightning. These lights are placed near the assets that give illumination to the scene as fluorescent lights and the intensity of these point lights are adjusted for realism.

Scene	Method	VMAF↑	PSNR↑	M-PSNR↑	SSIM↑	LPIPS↓	SPF↓	VE↑	PE↑
	StreamRF	33.79	26.49	12.15	0.816	0.192	49.0	0.69	0.54
	3DGStream	50.44	28.42	20.67	0.874	0.173	12.5	4.04	2.27
Attack 1	D-3DGS	64.87	29.07	22.03	0.901	0.147	76.0	0.85	0.38
	HiCoM	61.21	29.77	21.49	0.917	0.125	6.1	10.03	4.88
	Ours	65.69	29.50	24.05	0.908	0.123	5.2	12.63	5.67
	StreamRF	34.12	26.82	13.67	0.755	0.259	49.0	0.70	0.55
	3DGStream	41.34	27.73	19.36	0.844	0.213	12.3	3.36	2.25
Attack 2	D-3DGS	66.03	29.08	21.48	0.901	0.146	78.0	0.85	0.37
	HiCoM	62.18	29.48	18.06	0.914	0.127	6.3	9.87	4.68
	Ours	66.44	29.65	24.22	0.910	0.121	5.2	12.78	5.70
	StreamRF	40.95	27.14	11.86	0.721	0.193	48.0	0.85	0.57
	3DGStream	55.97	29.21	21.59	0.896	0.148	12.5	4.48	2.34
Attack 3	D-3DGS	63.99	29.01	25.38	0.901	0.146	74.0	0.86	0.39
	HiCoM	56.37	28.64	19.81	0.906	0.134	6.2	9.09	4.62
	Ours	66.74	29.80	26.01	0.911	0.122	5.2	12.83	5.73
	StreamRF	37.86	27.42	14.75	0.721	0.182	48.0	0.79	0.57
	3DGStream	57.01	26.75	23.63	0.869	0.179	14.0	4.07	1.91
Attack 4	D-3DGS	65.37	26.82	24.33	0.894	0.169	67.0	0.98	0.40
	HiCoM	31.47	22.03	18.65	0.751	0.331	6.0	5.25	3.67
	Ours	72.68	30.17	25.60	0.927	0.105	5.0	14.54	6.03
	StreamRF	35.81	26.41	13.84	0.787	0.189	49.0	0.73	0.54
	3DGStream	53.19	28.98	21.06	0.884	0.159	12.2	4.36	2.38
Defense 1	D-3DGS	66.74	29.04	24.77	0.903	0.143	74.0	0.90	0.39
	HiCoM	56.14	28.53	18.45	0.907	0.131	6.1	9.20	4.68
	Ours	66.97	29.68	25.78	0.909	0.121	5.2	12.88	5.71
	StreamRF	36.46	26.36	14.96	0.784	0.177	48.0	0.76	0.55
	3DGStream	56.57	29.19	19.81	0.895	0.151	12.3	4.60	2.37
Defense 2	D-3DGS	65.25	29.18	23.06	0.903	0.145	76.0	0.86	0.38
	HiCoM	55.08	28.59	17.81	0.905	0.131	6.1	9.03	4.69
	Ours	66.31	29.75	26.15	0.910	0.121	5.2	12.75	5.72
	StreamRF	35.14	27.11	12.89	0.711	0.195	49.0	0.72	0.55
	3DGStream	45.46	28.22	19.38	0.862	0.195	12.1	3.76	2.33
Interval 1	D-3DGS	65.59	29.05	21.36	0.901	0.146	75.0	0.87	0.39
	HiCoM	60.39	29.49	18.94	0.914	0.121	6.2	9.74	4.76
	Ours	66.28	29.69	24.86	0.910	0.123	5.2	12.75	5.71

Table A.2: **BASKET-Multiview Per-scene Evaluation Results.**

Scene	Max ptb	Skl opt	SE↓	VMAF↑	PSNR↑	M-PSNR↑	SSIM↑	LPIPS↓
Running Player (LC)	10°	✗	2.356	58.83	33.06	22.17	0.981	0.023
		✓	**0.593**	**73.77**	**37.43**	**26.61**	**0.987**	**0.014**
	20°	✗	4.519	43.78	29.77	18.55	0.976	0.031
		✓	**1.258**	**65.11**	**34.84**	**24.27**	**0.989**	**0.019**
Attack 4	10°	✗	2.588	71.33	29.88	24.15	0.924	0.107
		✓	**1.334**	**72.59**	**30.17**	**26.22**	**0.927**	**0.106**
	20°	✗	5.134	69.88	29.59	22.90	0.922	0.110
		✓	**2.563**	**72.27**	**30.10**	**25.72**	**0.926**	**0.106**

Table A.3: **Ablation Study of Skeleton Optimization.** Max ptb: max degrees of random perturbation.

SH Only	Precomp	Graph	LB	ms / iter SH1	ms / iter SH3
				24	34
✓				20	30
✓	✓			13	25
✓	✓	✓		9	21
✓	✓	✓	✓	**6**	**18**

Table A.4: **Ablation Study of Static Optimization Techniques.**

Spot Lights: 21 Spot Lights, with two types of intensity (15 lux and 10 lux) are placed on the top of the court and stands. The use of two types of illumination aims to enhance visibility on the court while creating a realistic ambiance by reducing light intensity in the stands, thereby directing spectators' focus toward the court.

Sky Light: The Sky Light setup uses the SLS Specified Cubemap source type with a cubemap of Tx_HDRI_07a at an angle of 175.0 and a distance threshold of $150,000$. The light intensity is set to 5.0, with a white color (FFFFFFFF). It affects the world and casts shadows, with indirect lighting intensity and volumetric scattering intensity both set to 1.0.

Post Process Volume: These settings include an Auto Exposure Histogram with an exposure compensation of -1.24, a minimum brightness of 0.5, and a maximum brightness of 8.0. The color temperature is set to 6036.0.

For global illumination, the *Lumen* method is used with a scene lighting quality

Sequence Name	# Players	#Cameras	Shot	#Frames	Resolution
Attack 1	11	83	Far	321	1080p
Attack 2	6	83	Far	130	1080p
Attack 3	4	83	Far	319	1080p
Attack 4	5	83	Mid	321	4k
Defense 1	2	83	Far	160	1080p
Defense 2	3	83	Far	220	1080p
Interval 1	5	83	Far	185	1080p

Table A.5: **BASKET-Multiview Core Scenes Composition.**

Sequence Name	Animation name	Background	#Players	Ball	#Cameras	Shot	#Frames	Config set
Running Player w/o ball	Running Player	Black	1	✗	40	Close	57	Set 2
Running Player w/o ball with light dynamics		Black	1	✗	40	Close	57	Set 3
Running Player w/o ball		Court	1	✗	83	Far	105	Set 1
Running Player w/o ball long path		Court	1	✗	83	Far	600	Set 1
Running Player w/o ball with light dynamics		Court	1	✗	83	Far	105	Set 4
Dribbling Player w/o ball	Dribbling Player	Black	1	✗	40	Close	48	Set 2
Dribbling Player with ball		Black	1	✓	40	Close	48	Set 2
Dribbling Player w/o ball		Court	1	✗	83	Far	160	Set 1
Day Cycle		Court	0	✗	83	Far	76	Set 1

Table A.6: **BASKET-Multiview Development Scenes Composition.**

of 5.0, scene detail of 4.0, and a scene view distance of 100.0. Reflections use a quality setting of 2.0, hit lighting for reflections, high-quality translucency reflections enabled, and a maximum of 3 reflection bounces.

2. Set 2: This is simpler than Set 1 as it only consists of a Skylight. Here the scenario is completely empty as we want to place the player in a black environment with a sky light for global illumination.

   Sky Light: The Sky Light setup uses the SLS Specified Cubemap source type with a cubemap GrayLightTextureCube. The light intensity is set to 3.0, with a white color (FFFFFFFF).

   Post Process Volume: This sets up the lightning and reflection methods to use Lumen.

3. Set 3: This is similar to Set 2 with an additional point light moving around the player, to better understand the effect of lighting changes on our method's ability.

   Point light: It is placed at 170.0 units from the player and rotates around it. It has an intensity of 10000.0 unitless, with a white color (FFFFFFFF) and an

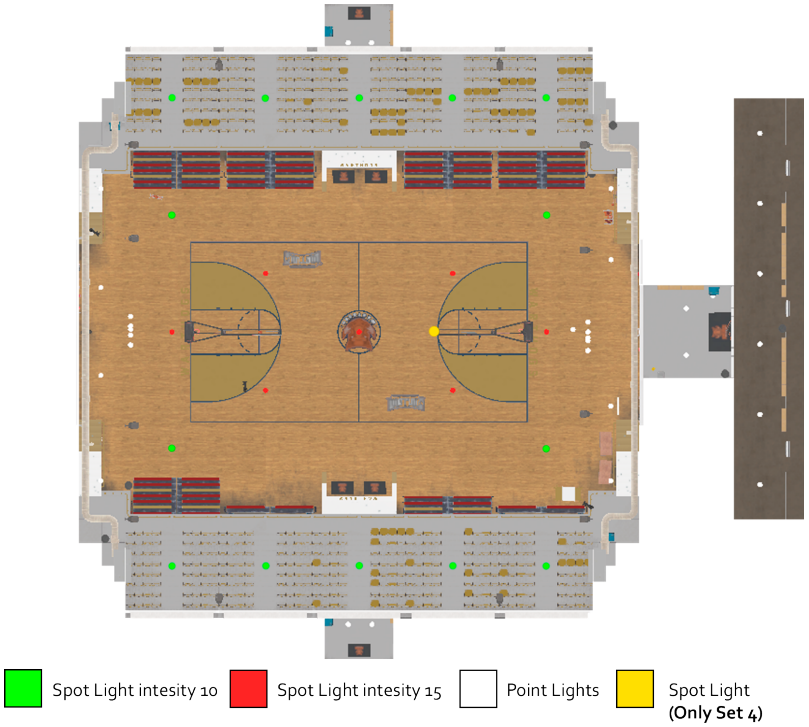

Figure A.1: **Lighting Layout in the Court.**

Figure A.2: **Sample Binding Pose Player.**

attenuation radius of 1000.0.

Sky light: This setup is same as Set 2 but with an intensity of 0.5.

Figure A.3: **Lighting Layout for the Lighting Change Sequence.**

4. Set 4: The map is same as Set 1, a basketball court. The lighting configuration is the same as Set 1, with an additional spotlight on the middle of the path that the player is following. **Spot light**: It has an intensity of 2000.0 cd(candelas), it has a white color (FFFFFFFF), an attenuation radius of 1140.0 and an outer cone angle of 52.0, the rest of the values are set at 0.0.

A.3.2 Camera setup

We use two different camera setups, one for all the scenes in the stadium (Config Set 1 & 4) and another for scenes with a black background (Config Set 2 & 3)

1. **83 Camera Setup**: The camera rig consists of 83 cameras designed to capture all views of the stadium. Using the stadium's square shape, we place cameras across multiple levels: the highest focuses on the play, while the others capture the stadium's structure.

   All cameras have the same configuration, with a sensor width of 23.76 mm and a sensor height of 13.366 mm. They recreate a 16:9 Digital film and capture with a lens of focal length of 12 mm, with a min FStop of 2.8 and a Max FStop of 22.0.

   The rig for Attack 4 has a slight change in the orientation. We place an empty actor on the basket and activate the *look at* function on the camera settings. The camera configurations are similar with a focal length of 22.00 mm for simulating a zoomed-in view.

Figure A.4: **Camera Layout in the Court.**

2. **40 Camera Setup**: This setup is designed to capture multiple angles of the player, and consists of 40 cameras. The cameras are arranged in a sphere and

the player is located at the center. To improve the view quality, we activate the *look at* function with an offset of 94.0 on the Z axis. The camera configurations are similar to the 83 Camera setup, but with a sensor width of 24 mm and a sensor height of 13.5 mm.

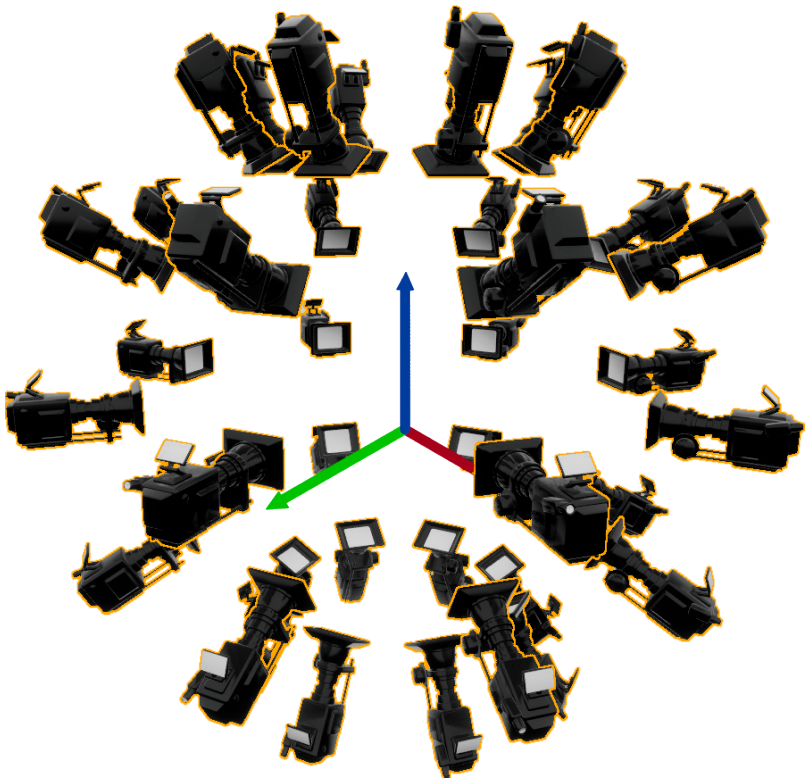

Figure A.5: **Camera Layout for Capturing the Binding Pose Pictures and Black Background Sequences.**

A.3.3 Render settings

1. EasySynth [47]: This setup is configured to generate RGB (Color) images, depth maps, and normal maps. The color images are output in PNG format, depth maps in EXR format with a 16-bit precision, and normal maps in PNG format. The resolution is set to 1920x1080 and depth range configured to 100 meters.
2. Movie Render Queue: Anti-aliasing is turned on with spatial sample count set to 4 and temporal sample count set to 1. Anti-aliasing is overridden and the

anti-aliasing method is set to *None* to ensure no additional smoothing artifacts are introduced. The render warm-up count is set to 32, and the engine warm-up count is set to 4. Both "Use Camera Cut for Warm-Up" and "Render Warm-Up Frames" are disabled.

In the console variables section, Temporal AA upsampling (`r.TemporalAA.Upsampling`) is set to 0.0 to disable any upscaling associated with temporal anti-aliasing. Motion blur quality (`r.MotionBlurQuality`) is also set to 0.0 to remove motion blur effects from the render. The screen percentage (`r.ScreenPercentage`) is set to 70.0, likely to balance performance and output quality. For the output settings, the image size is set to $1920x1080$.

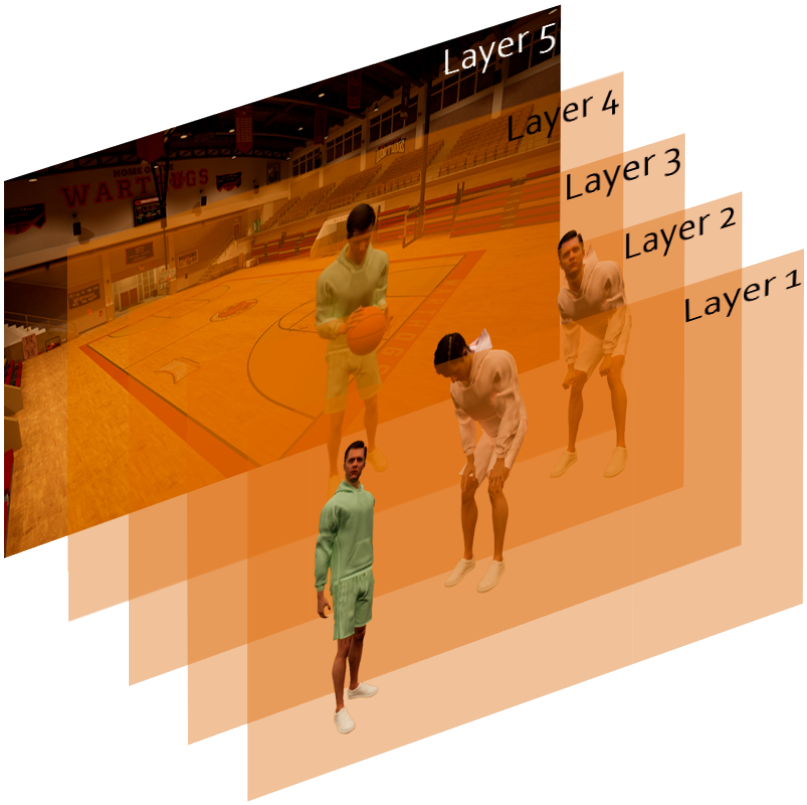

Figure A.6: **Layer Rendering Illustration.**

A.3.4 Data generating workflow

We outline a brief sketch of the steps taken to design a sequence in order.

1. The character skeleton is exported from Unreal Engine 5 (UE5) to Maya. In Maya, a plugin is applied to generate control rigs, which facilitate precise control over the animation process and result in realistic, fluid movements.
2. Using references from real sports plays, animations are carefully created to replicate the movements accurately. Once the animations are completed, they are exported from Maya and re-imported into UE5 as animated skeletons.
3. The play sequences are created in UE5 by combining player animations with elements like the ball. These sequences are assembled and organized in the sequence editor.
4. To generate detailed visual data, Easy Synth is configured. It generates multiple types of images, such as Base Color renders, Depth maps (up to 100 meters, exported in EXR format for improved precision), and Normal maps capturing surface geometry.
5. Finally, semantic images are generated to enhance accuracy. Each player, the ball, and the background are separated into individual layers with transparency. This layered approach results in more precise semantic images.

Bibliography

[1] Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, Brian Hirsh, Sherlock Huang, Kshiteej Kalambarkar, Laurent Kirsch, Michael Lazos, Mario Lezcano, Yanbo Liang, Jason Liang, Yinghai Lu, CK Luk, Bert Maher, Yunjie Pan, Christian Puhrsch, Matthias Reso, Mark Saroufim, Marcos Yukio Siraichi, Helen Suk, Michael Suo, Phil Tillet, Eikan Wang, Xiaodong Wang, William Wen, Shunting Zhang, Xu Zhao, Keren Zhou, Richard Zou, Ajit Mathews, Gregory Chanan, Peng Wu, and Soumith Chintala. PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation. In *29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '24)*. ACM, April 2024. doi: 10.1145/3620665.3640366. URL https://pytorch.org/assets/pytorch2-2.pdf. 3.3.3

[2] Bharat Lal Bhatnagar, Cristian Sminchisescu, Christian Theobalt, and Gerard Pons-Moll. Combining implicit function learning and parametric models for 3d human reconstruction. In *European Conference on Computer Vision (ECCV)*. Springer, aug 2020. 3.2

[3] Bharat Lal Bhatnagar, Cristian Sminchisescu, Christian Theobalt, and Gerard Pons-Moll. Loopreg: Self-supervised learning of implicit surface correspondences, pose and shape for 3d human mesh registration. In *Advances in Neural Information Processing Systems (NeurIPS)*, December 2020. 3.2

[4] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 3.2, 5.1

[5] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, January 2008. ISSN 0001-0782. doi: 10.1145/1327452.1327492. URL https://doi.org/10.1145/1327452.1327492. 3

[6] Epic Games. Unreal engine 5, 2025. URL https://www.unrealengine.com. 4

[7] Qiankun Gao, Jiarui Meng, Chengxiang Wen, Jie Chen, and Jian Zhang. Hicom: Hierarchical coherent motion for dynamic streamable scenes with 3d gaussian splatting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. 2.5, 3.2, 5.2

[8] Sharath Girish, Tianye Li, Amrita Mazumdar, Abhinav Shrivastava, David Luebke, and Shalini De Mello. QUEEN: QUantized efficient ENcoding for streaming free-viewpoint videos. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=7xhwE7VH4S. 2.5

[9] Liangxiao Hu, Hongwen Zhang, Yuxiang Zhang, Boyao Zhou, Boning Liu, Shengping Zhang, and Liqiang Nie. Gaussianavatar: Towards realistic human avatar modeling from a single video via animatable 3d gaussians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 634–644, 2024. 2.3

[10] Tao Hu, Fangzhou Hong, and Ziwei Liu. Surmo: Surface-based 4d motion modeling for dynamic human rendering, 2024. 2.2

[11] CMU Imaging Group. Time-of-flight radiance fields (törf): A method for integrating depth sensing into dynamic radiance field modeling. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. 2.1

[12] Yuheng Jiang, Zhehao Shen, Yu Hong, Chengcheng Guo, Yize Wu, Yingliang Zhang, Jingyi Yu, and Lan Xu. Robust dual gaussian splatting for immersive human-centric volumetric videos, 2024. URL https://arxiv.org/abs/2409.08353. 2.2

[13] Hanbyul Joo, Tomas Simon, Xulong Li, Hao Liu, Lei Tan, Lin Gui, Sean Banerjee, Timothy Scott Godisart, Bart Nabbe, Iain Matthews, Takeo Kanade, Shohei Nobuhara, and Yaser Sheikh. Panoptic studio: A massively multiview system for social interaction capture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 5.1

[14] James T Kajiya. The rendering equation. *ACM SIGGRAPH computer graphics*, 20(4):143–150, 1986. 2.1

[15] Kai Katsumata, Duc Minh Vo, and Hideki Nakayama. A compact dynamic 3d gaussian representation for real-time dynamic view synthesis, 2024. URL https://arxiv.org/abs/2311.12897. 2.2

[16] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/. 1, 2.1, 3.2, 3.4

[17] Agelos Kratimenos, Jiahui Lei, and Kostas Daniilidis. Dynmf: Neural motion factorization for real-time dynamic view synthesis with 3d gaussian splatting, 2023. URL https://arxiv.org/abs/2312.00112. 2.2

[18] Lingzhi Li, Zhen Shen, Zhongshu Wang, Li Shen, and Ping Tan. Streaming radiance fields for 3d video synthesis. *Advances in Neural Information Processing Systems*, 35:13485–13498, 2022. 2.4, 5.2

[19] Peng Li, Wangguandong Zheng, Yuan Liu, Tao Yu, Yangguang Li, Xingqun Qi, Mengfei Li, Xiaowei Chi, Siyu Xia, Wei Xue, et al. Pshuman: Photorealistic single-view human reconstruction using cross-scale diffusion. *arXiv preprint arXiv:2409.10141*, 2024. 2.3

[20] Xueting Li, Ye Yuan, Shalini De Mello, Gilles Daviet, Jonathan Leaf, Miles Macklin, Jan Kautz, and Umar Iqbal. Simavatar: Simulation-ready avatars with layered hair and clothing. *arXiv preprint arXiv:2412.09545*, 2024. 2.3

[21] Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime gaussian feature splatting for real-time dynamic view synthesis. *arXiv preprint arXiv:2312.16812*, 2023. 2.2

[22] Zhi Li, Christos G Bampis, Ioannis Katsavounidis, Anne Aaron, and Alan C Bovik. Toward a practical perceptual video quality metric. *Netflix Technology Blog*, June 2016. URL https://medium.com/netflix-techblog/toward-a-practical-perceptual-video-quality-metric-653f208b9652. 5.3

[23] Youtian Lin, Zuozhuo Dai, Siyu Zhu, and Yao Yao. Gaussian-flow: 4d reconstruction with dynamic 3d gaussian particle, 2023. URL https://arxiv.org/abs/2312.03431. 2.2

[24] Xian Liu, Xiaohang Zhan, Jiaxiang Tang, Ying Shan, Gang Zeng, Dahua Lin, Xihui Liu, and Ziwei Liu. Humangaussian: Text-driven 3d human generation with gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6646–6657, 2024. 2.3

[25] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, October 2015. 3.2, 5.1

[26] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *3DV*, 2024. 2.5, 3.2, 3.3.1, 5.2

[27] Saswat Subhajyoti Mallick, Rahul Goel, Bernhard Kerbl, Francisco Vicente Carrasco, Markus Steinberger, and Fernando De La Torre. Taming 3dgs: High-quality radiance fields with limited resources, 2024. URL https://arxiv.org/abs/2406.15643. 3.2

[28] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2020. 2.1

[29] Nice Pictures. Men's and women's sport outfits 3 - modular - rigged, 2024. URL https://www.fab.com/listings/2eebf211-df72-4daf-a8c4-24bfadba9e7a. 3D asset compatible with Metahuman skeletons, available at Fab.com, accessed on January 20, 2025. 4

[30] Albert Pumarola, Enrique Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10318–10327, 2021. 2.1

[31] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. URL https://arxiv.org/abs/2408.00714. 3.1

[32] Shunsuke Saito, Stanislav Pidhorskyi, Igor Santesteban, Forrest Iandola, Divam Gupta, Anuj Pahuja, Nemanja Bartolovic, Frank Yu, Emanuel Garbin, and Tomas Simon. Squeezeme: Efficient gaussian avatars for vr. *arXiv preprint arXiv:2412.15171*, 2024. 2.3

[33] Zhijing Shao, Zhaolong Wang, Zhuang Li, Duotun Wang, Xiangru Lin, Yu Zhang, Mingming Fan, and Zeyu 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*, pages 1606–1616, 2024. 2.3

[34] Richard Shaw, Michal Nazarczuk, Jifei Song, Arthur Moreau, Sibi Catley-Chandar, Helisa Dhamo, and Eduardo Perez-Pellitero. Swings: Sliding windows for dynamic 3d gaussian splatting, 2024. URL https://arxiv.org/abs/2312.13308. 2.2, 3.2

[35] Peter-Pike Sloan, Jan Kautz, and John Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *ACM SIGGRAPH 2002 Papers*, pages 527–536, 2002. 2.1

[36] Liangchen Song, Anpei Chen, Zhong Li, Zhang Chen, Lele Chen, Junsong Yuan, Yi Xu, and Andreas Geiger. Nerfplayer: A streamable dynamic scene representation with decomposed neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics*, 29(5):2732–2742, 2023. doi: 10.1109/TVCG.2023.3247082. 2.4

[37] Decagon Studios. High school basketball gym - (day/night/afternoon/midnight lighting), 2025. URL https://www.fab.com/listings/03e76034-abbf-4fc2-aa05-b025996eeb1d. 3D asset available at Fab.com, accessed on January 20, 2025. 4

[38] Jiakai Sun, Han Jiao, Guangyuan Li, Zhanjie Zhang, Lei Zhao, and Wei Xing. 3dgstream: On-the-fly training of 3d gaussians for efficient streaming of photo-realistic free-viewpoint videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20675–20685, June 2024. 2.5, 3.2, 5.2

[39] Carnegie Mellon University. Carnegie mellon professor's unique new vision technology will be used to present replays in super bowl xxxv. ScienceDaily, January 2001. URL https://www.sciencedaily.com/releases/2001/01/010124075009.htm. Accessed: 2025-01-19. 1

[40] Liao Wang, Qiang Hu, Qihan He, Ziyu Wang, Jingyi Yu, Tinne Tuytelaars, Lan Xu, and Minye Wu. Neural residual radiance fields for streamably free-viewpoint videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 76–87, June 2023. 2.4

[41] Liao Wang, Kaixin Yao, Chengcheng Guo, Zhirui Zhang, Qiang Hu, Jingyi Yu, Lan Xu, and Minye Wu. Videorf: Rendering dynamic radiance fields as 2d feature video streams, 2023. 2.2

[42] Penghao Wang, Zhirui Zhang, Liao Wang, Kaixin Yao, Siyuan Xie, Jingyi Yu, Minye Wu, and Lan Xu. Vˆ 3: Viewing volumetric videos on mobiles via streamable 2d dynamic gaussians. *ACM Transactions on Graphics (TOG)*, 43 (6):1–13, 2024. 2.3

[43] Jing Wen, Xiaoming Zhao, Zhongzheng Ren, Alexander G Schwing, and Shenlong Wang. Gomavatar: Efficient animatable human modeling from monocular video using gaussians-on-mesh. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2059–2069, 2024. 2.3

[44] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20310–20320, June 2024. 2.2

[45] Yunzhi Yan, Haotong Lin, Chenxu Zhou, Weijie Wang, Haiyang Sun, Kun Zhan, Xianpeng Lang, Xiaowei Zhou, and Sida Peng. Street gaussians: Modeling dynamic urban scenes with gaussian splatting, 2024. URL https://arxiv.org/abs/2401.01339. 2.2

[46] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin.

Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. *arXiv preprint arXiv:2309.13101*, 2023. 2.2

[47] YDRIVE. Easysynth: An unreal engine plugin for image dataset creation, 2023. URL https://github.com/ydrive/EasySynth. 4, 1