Supplementary Materials
GauSTAR: Gaussian Surface Tracking and Reconstruction

Chengwei Zheng[1] Lixin Xue[1] Juan Zarate[1] Jie Song[1,2,3]

[1]ETH Zürich [2]HKUST(GZ) [3]HKUST

In this supplementary document, we provide implementation details of GauSTAR and other baseline methods in Sec. 1. Then, we demonstrate more experiment results in Sec. 2 with qualitative comparisons and ablation studies that further validate our method's effectiveness. Finally, we include an extensive discussion of ethics and societal impact of our approach in Sec. 3.

In the supplementary video, we demonstrate GauSTAR's overall pipeline and its performance on various dynamic scenes. We also provide visual comparisons against baseline methods.

Contents

1. Implementation Details

1.1. GauSTAR Details

Code releasing. Our code is available at https://eth-ait.github.io/GauSTAR/. We provide our implementation on publicly available datasets, allowing for reproducibility and further research. However, due to licensing restrictions, the data captured for GauSTAR testing will not be released.

Running Time. Our method sequentially processes the video frames. Each frame requires approximately 5 minutes of training time on a single NVIDIA RTX 4090 GPU (running time varies depending on the face number). For frames where no topology changes are detected, we only perform fixed-topology surface reconstruction, reducing the processing time to approximately 2 minutes. Once training is complete, rendering is performed in real time, leveraging CUDA acceleration for core Gaussian splatting operations. A comparison of running times across different methods, using the same sequence, is provided in Tab. 1.

Method	Training (per frame)	Rendering
HumanRF [3]	4.9 min	0.8 fps
Dynamic 3D GS [4]	1.5 min	203 fps
PhysAvatar [6]	1.1 min	218 fps
2DGS [2]	7.1 min	231 fps
GauSTAR (Ours)	4.6 (or 2.1) min	188 fps

Table 1. Runtime comparisons on a single RTX 4090. GauSTAR requires 2.1 min when no topology changes are detected.

Initial Input for the First Frame. GauSTAR requires a mesh as input for each frame. For the first frame, we employ an RGB-D based multi-view reconstruction method [1] to generate the initial mesh. The reconstructed mesh is then down-sampled to contain between 100,000 and 200,000 faces, with the exact count varying according to scene complexity. We attach $N = 6$ Gaussians to each face, resulting in approximately 600,000 to 1,200,000 total Gaussians (note that the number of faces may dynamically change due to topology updates). The Gaussian appearance parameters for the first frame are initialized using the mesh texture, while the initial opacity, scale, and rotation parameters are set to predefined values.

Remeshing Details. As introduced in Sec. 3.5 of the main paper, after generating new faces, we update the underlying mesh topology by integrating newly generated faces with the original mesh. This process involves removing specific faces from the original mesh, identifying corresponding newly generated faces, and seamlessly connecting them.

This method begins by identifying original faces where the unbound weight exceeds a predefined threshold. These faces are then grouped by their connected components. We delete any connected components that contain more faces than a specified threshold. Next, we create a voxel volume to record the positions of unbound Gaussians from deleted faces. Within this volume, we identify newly generated faces and remove isolated faces based on their connected components, preparing them for integration with the remaining original mesh. The connection process involves two steps of vertex matching: first, for each vertex x on the boundary of newly generated faces, we locate its closest vertex y on the original mesh boundary, set their positions to y, and merge them; then, for unmatched vertices on the original mesh boundary, we find their closest vertices on the new face boundary and perform similar alignment and merging operations. Finally, we complete the mesh reconstruction through edge flipping and hole filling operations to ensure a seamless surface.

1.2. Baseline Details

HumanRF [3]. As the official mesh extraction code for HumanRF is not publicly available, we implemented our own version following their paper. While we use Marching Cubes for mesh extraction, the raw outputs often contain undesirable internal surfaces, such as those inside the human body. To address this issue, we additionally generate outer surfaces using TSDF fusion, then remove any mesh faces that are far from these TSDF-extracted surfaces. We also implement light annotations in HumanRF to reduce light bloom artifacts. We capture a background frame to detect intense light sources and mask the affected image regions. While this enhancement improves the overall quality, it does not entirely eliminate the artifacts.

Dynamic3DGS [4]. The original Dynamic3DGS paper primarily focuses on rendering quality rather than geometric reconstruction. As it does not provide a dedicated surface reconstruction method, we employ TSDF fusion techniques similar to our surface generation approach, combining depth images from multiple views to obtain the final mesh. Due to the inherent limitations of Gaussian splatting, the resulting reconstruction exhibits considerable noise in the geometry.

2D Gaussian Splatting [2]. 2DGS is designed for reconstructing static scenes, and we use it to process each frame independently. For each frame, we initialize the point cloud using the refined point cloud from [1]. These point clouds are the ones used for rendering depth inputs from IR cameras. We down-sample each frame's point cloud to 600,000 points before processing. To enhance geometry consistency, we incorporate a mask loss similar to our formulation in Eq. (6) during the training stage. Unlike GauSTAR and other baselines that leverage temporal information across frames, 2DGS reconstructs each frame independently. As a result, 2DGS is less robust and more prone to overfitting in our 47-view setting, leading to floater artifacts and notable temporal jittering. A similar overfitting trend is also observed in another experiment, where reducing input views from 200 to 47 on the Mip-NeRF dataset causes the training PSNR to increase (29.2 to 31.5), while the test PSNR declines (28.7 to 26.9).

PhysAvatar [6]. While PhysAvatar's original paper describes a pipeline beginning with mesh tracking followed by clothing reconstruction and simulation, it does not explicitly mention SMPL-X dependency. However, their released implementation utilizes SMPL-X for improved tracking robustness, particularly in hand regions, as confirmed by the authors. Without SMPL-X initialization, their method relies on inertial estimates for full-body initialization between frames, similar to Dynamic3DGS.

For a fair comparison across our diverse sequences containing single humans, multiple humans, and non-human objects, we implement two variants of PhysAvatar: one using inertial initialization and another additionally using SMPL-X deformation for human vertices. As the SMPL-X fitting code was not publicly available at the time of submission, we adopted the approach from X-Avatar [5] to fit SMPL-X using multi-view images and reconstructed meshes. This fitting is done with a multi-stage pipeline: first extracting 2D keypoints using OpenPose and triangulating them to 3D with specific filtering for unstable hand predictions. The SMPL-X parameter optimization then proceeds through three stages: we first initialize the parameters using the filtered 3D keypoints, then refine body pose and shape parameters using the scan geometry, and finally optimize hand poses and facial expressions using 3D landmarks.

2. Additional Experiments

2.1. Comparisons with Baselines

We provide additional qualitative comparisons with HumanRF [3], Dynamic 3D Gaussians [4], PhysAvatar [6], and 2D Gaussian Splatting [2] in Fig. 1.

HumanRF trains each video segment independently. This approach leads to slow rendering times and inconsistent tracking. Due to its independent segment training, HumanRF struggles with strong occlusions where most cameras cannot observe certain regions (Fig. 4 in the main paper). In contrast, our method demonstrates greater robustness to occlusion through consistent tracking and scene flow warping.

2DGS is a static scene reconstruction method, and we use it to process frames independently for reconstruct-

Figure 1. Comparisons of appearance and geometry reconstruction. HumanRF offers overall good visual quality but lacks tracking capabilities. Dynamic 3D Gaussians produces blurry renderings and noisy surfaces. PhysAvatar struggles with handling topology changes, while 2D Gaussian Splatting faces challenges with both tracking and floating artifacts. In contrast, GauSTAR delivers high-quality reconstruction and effectively manages topology changes.

ing dynamic surfaces. However, its reconstruction quality varies significantly across frames, with some frames showing impressive results while others exhibit notable artifacts. Furthermore, as a single-frame method, it produces temporally unstable reconstructions and does not provide tracking capabilities.

Dynamic3DGS employs inertial estimation to initialize Gaussians for subsequent frames. While it supports tracking, the geometric quality is limited. Moreover, without an underlying mesh constraint, Gaussians can move freely in space, resulting in inconsistent tracking.

PhysAvatar maintains consistent mesh tracking and

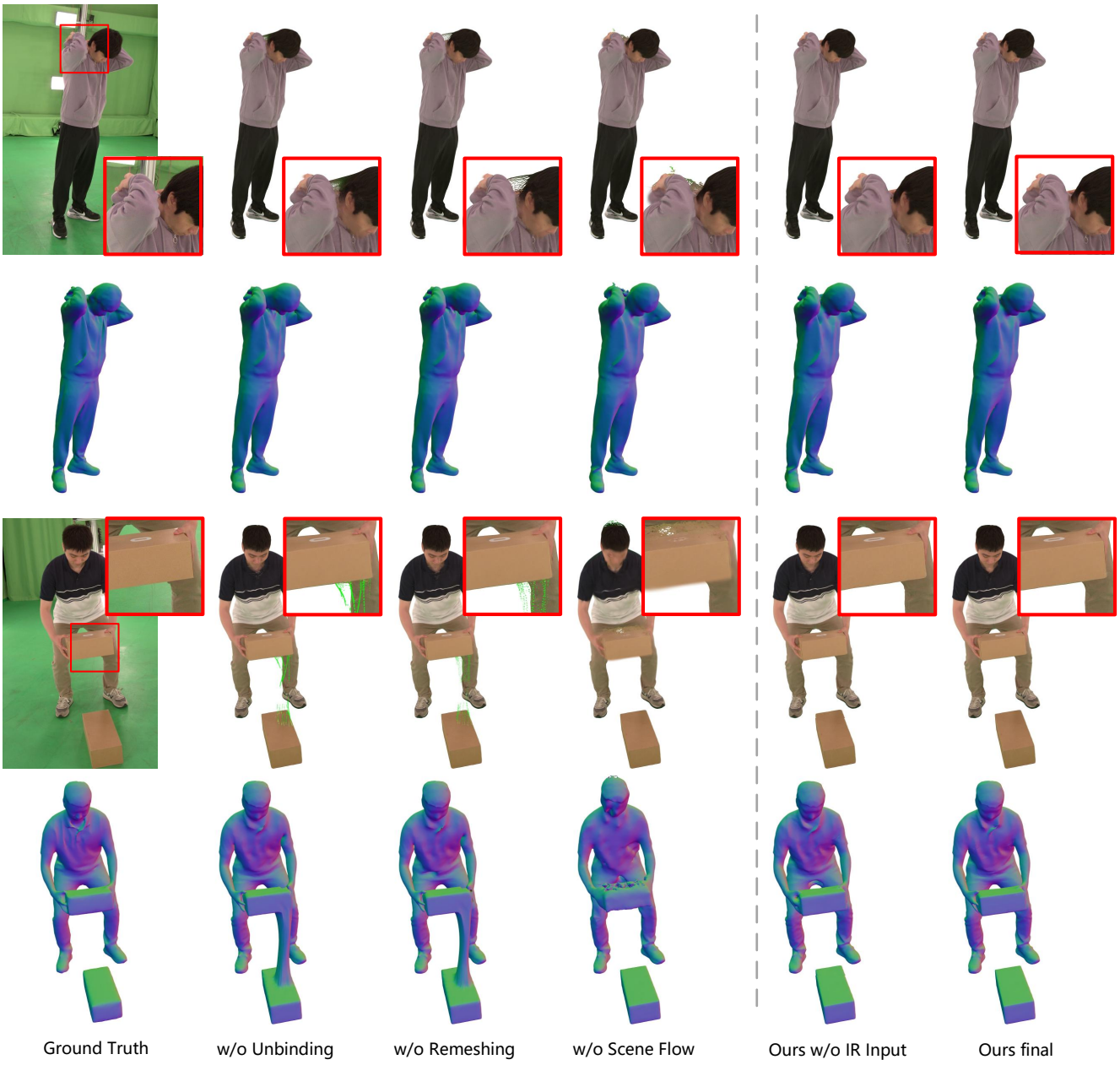

Figure 2. Qualitative results for ablation study. Unbinding and re-meshing are crucial for handling topology changes, and scene flow ensures robust tracking of large movements. Our method without IR input yields a similar quality to the full version of our method but needs additional data prepossessing.

achieves high-quality reconstruction for clothed humans under normal motions. However, its fixed-topology assumption fundamentally limits its ability to handle dynamic scenes where topology changes occur. In such cases, it fails dramatically when encountering topology changes due to its inability to handle such modifications.

2.2. Qualitative Results for Ablations

We illustrate the impact of our key components through qualitative comparisons in Fig. 2. Without the unbinding and remeshing components, our method fails to properly handle topology changes, resulting in incorrectly merged geometries between the human head and hood in the first two rows, and erroneous connections between the separate boxes. The scene flow warping initialization proves crucial

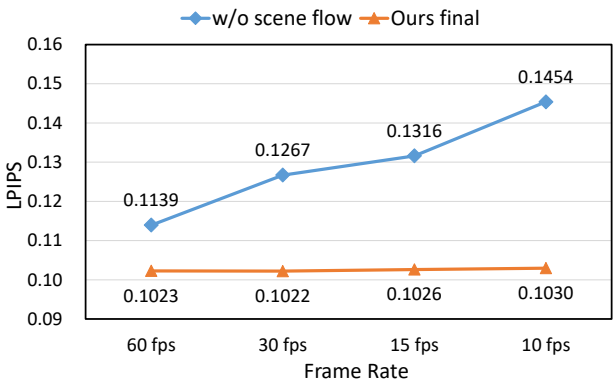

Figure 3. Scene flow warping ablation. We capture a sequence at 60 FPS and down-sample it to 30 FPS, 15 FPS, and 10 FPS, increasing motion between frames. Our method consistently performs well across frame rates, while the method without scene flow shows higher errors as the motion between frames increases.

as well; without it, Gaussians become trapped in local minima and cannot properly redistribute, leading to significant geometric artifacts.

2.3. Additional Scene Flow Ablation

To demonstrate the effectiveness of our scene flow warping in handling large motions, we evaluate reconstruction quality under varying degrees of inter-frame movement. We conduct this experiment by capturing a sequence at 60 FPS and systematically down-sampling it to 30 FPS, 15 FPS, and 10 FPS, effectively increasing the magnitude of motion between consecutive frames. Comparing our full method against a variant without scene flow warping reveals that our approach maintains consistent reconstruction quality across all frame rates, while the ablated version shows progressively deteriorating performance as inter-frame motion increases.

2.4. GauSTAR with RGB input

Our method does not necessarily require the IR depth input. For example, we can use multiview stereo to create a rough depth map. Here we use rendered depth from HumanRF for its robustness and smoothness, which has fewer artifacts. As shown in the last two columns of Fig. 2, GauSTAR works almost equally well compared to the version with depth input. This means GauSTAR can be utilized for capture setup with RGB input only.

2.5. Failure Cases

We present failure cases in Fig. 4, highlighting the challenges faced by our method. In particular, transparent and specular surfaces remain difficult for most surface reconstruction techniques, including Gaussian splitting, due to

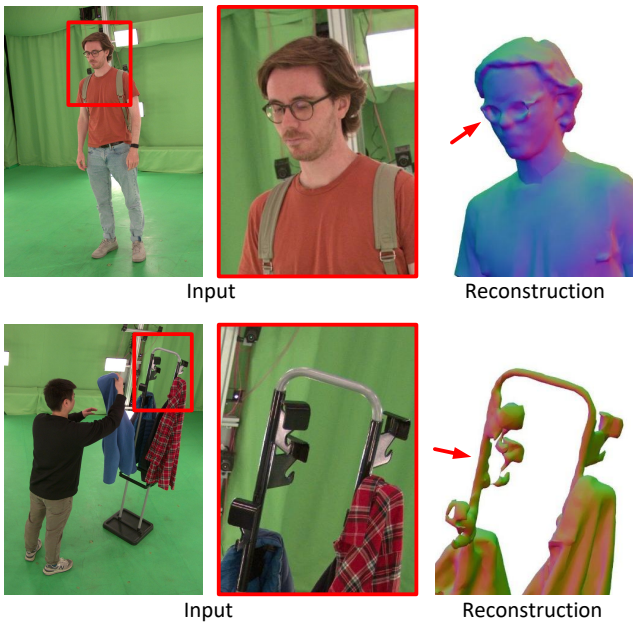

Figure 4. Failure cases on transparent and specular surfaces.

their complex light interactions and lack of reliable depth cues. Accurately reconstructing such surfaces requires more advanced strategies, which we leave as future works.

3. Ethics and Societal Impact Discussion

Our data collection procedure has been reviewed and approved by the responsible Institutional Review Board. All subjects voluntarily participated in the data collection process and were fully informed about the intended use of the data in research.

GauSTAR enables the digitization of general dynamic scenes from multi-view captures, which has broad applications in visual effects, robotics, and virtual production. As our method can reconstruct and track detailed surface changes, there are potential concerns about privacy and surveillance when applied to scenes involving human activities. Such concerns must be addressed before deploying this technology in commercial products. Our goal with this work is to enable beneficial applications such as human-robot interaction, markerless motion analysis, and immersive telepresence. Our system represents a technical advancement in computer vision that can benefit numerous fields from industrial automation to cultural preservation. While we cannot prevent potential misuse of such technology, we believe in transparent research practices, including detailed technical discussions and code release. This openness allows the research community to better understand both the capabilities and limitations of such systems, and to develop appropriate safeguards against concerning applications.

References

[1] Alvaro Collet, Ming Chuang, Pat Sweeney, Don Gillett, Dennis Evseev, David Calabrese, Hugues Hoppe, Adam Kirk, and Steve Sullivan. High-quality streamable free-viewpoint video. *ACM Transactions on Graphics (ToG)*, 34(4):1–13, 2015. 1, 2

[2] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 1, 2

[3] Mustafa Işık, Martin Rünz, Markos Georgopoulos, Taras Khakhulin, Jonathan Starck, Lourdes Agapito, and Matthias Nießner. Humanrf: High-fidelity neural radiance fields for humans in motion. *ACM Transactions on Graphics (TOG)*, 42(4):1–12, 2023. 1, 2

[4] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *2024 International Conference on 3D Vision (3DV)*, pages 800–809. IEEE, 2024. 1, 2

[5] Kaiyue Shen, Chen Guo, Manuel Kaufmann, Juan Zarate, Julien Valentin, Jie Song, and Otmar Hilliges. X-avatar: Expressive human avatars. *Computer Vision and Pattern Recognition (CVPR)*, 2023. 2

[6] Yang Zheng, Qingqing Zhao, Guandao Yang, Wang Yifan, Donglai Xiang, Florian Dubost, Dmitry Lagun, Thabo Beeler, Federico Tombari, Leonidas Guibas, et al. Physavatar: Learning the physics of dressed 3d avatars from visual observations. *arXiv preprint arXiv:2404.04421*, 2024. 1, 2