Gaussian Surfel Splatting for Live Human Performance Capture

ZHENG DONG, State Key Laboratory of CAD&CG, Zhejiang University, China
KE XU, City University of Hong Kong, China
YAOAN GAO, State Key Laboratory of CAD&CG, Zhejiang University, China
HUJUN BAO, State Key Laboratory of CAD&CG, Zhejiang University, China
WEIWEI XU*, State Key Laboratory of CAD&CG, Zhejiang University, China
RYNSON W.H. LAU, City University of Hong Kong, China

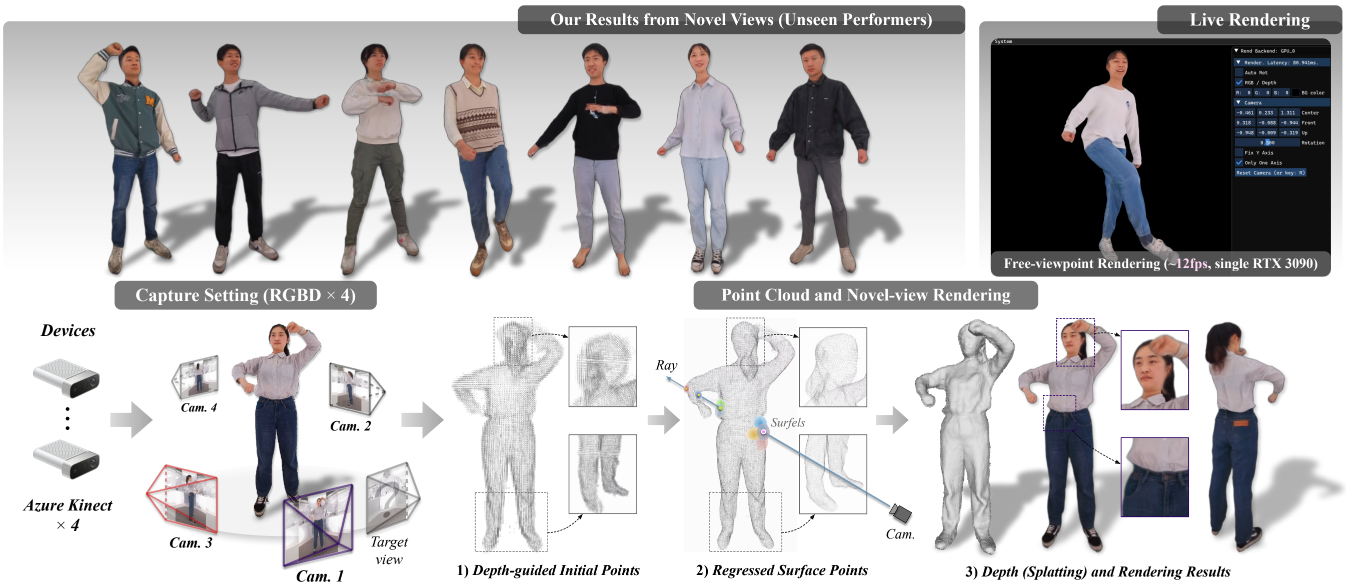

Fig. 1. We propose a novel point-based method for live human performance capture from very sparse (e.g., four) RGBD sensors. Our method learns a hybrid generalizable human representation (PGH), which regresses human surface points and parameterizes their geometry/texture features as 2D Gaussian surfels via a *surface implicit function* and a *Gaussian implicit function*, respectively, and then uses surfel splatting and blending-based appearance enhancement to create geometrically and photometrically correct novel-view videos.

High-quality real-time rendering using user-affordable capture rigs is an essential property of human performance capture systems for real-world applications. However, state-of-the-art performance capture methods may not yield satisfactory rendering results under a very sparse (e.g., four) capture setting. Specifically, neural radiance field (NeRF)-based methods and 3D Gaussian Splatting (3DGS)-based methods tend to produce local geometry errors for unseen performers, while occupancy field (PIFu)-based methods often produce unrealistic rendering results. In this paper, we propose a novel generalizable neural approach to reconstruct and render the performers from very sparse RGBD streams in high quality. The core of our method is a novel point-based generalizable human (PGH) representation conditioned on the pixel-aligned RGBD features. The PGH representation learns a *surface implicit function* for the regression of surface points and a *Gaussian implicit function* for parameterizing the radiance fields of the regressed surface points with 2D Gaussian surfels, and uses surfel splatting for fast rendering. We learn this hybrid human representation via two novel networks. First, we propose a novel point-regressing network (PRNet) with a depth-guided point cloud initialization (DPI) method to regress an accurate surface point cloud based on the denoised depth information. Second, we propose a novel neural blending-based surfel splatting network (SPNet) to render high-quality geometries and appearances in novel views based on the regressed surface points and high-resolution RGBD features of adjacent views. Our method produces free-view human performance videos of 1K resolution at 12 fps on average. Experiments on two benchmarks show that our method outperforms state-of-the-art human performance capture methods.

CCS Concepts: • **Computing methodologies → Image-based rendering**; *Point-based models*.

*Corresponding author

Authors' addresses: Zheng Dong, State Key Laboratory of CAD&CG, Zhejiang University, China, zhengdong@zju.edu.cn; Ke Xu, City University of Hong Kong, China, kkangwing@gmail.com; Yaoan Gao, State Key Laboratory of CAD&CG, Zhejiang University, China, yaoangao@zju.edu.cn; Hujun Bao, State Key Laboratory of CAD&CG, Zhejiang University, China, bao@cad.zju.edu.cn; Weiwei Xu, State Key Laboratory of CAD&CG, Zhejiang University, China, xww@cad.zju.edu.cn; Rynson W.H. Lau, City University of Hong Kong, China, rynson.lau@cityu.edu.hk.

0730-0301/2024/12-ART263 $15.00
https://doi.org/10.1145/3687993

Additional Key Words and Phrases: human performance capture, point-based generalizable human representation, surface implicit function, Gaussian implicit function

ACM Reference Format:
Zheng Dong, Ke Xu, Yaoan Gao, Hujun Bao, Weiwei Xu, and Rynson W.H. Lau. 2024. Gaussian Surfel Splatting for Live Human Performance Capture. *ACM Trans. Graph.* 43, 6, Article 263 (December 2024), 17 pages. https://doi.org/10.1145/3687993

1 INTRODUCTION

Human performance capture is an active research problem in the graphics and vision communities. It aims to reconstruct and render novel views of human-featured scenarios and is fundamental to numerous VR/AR applications. To provide immersive user experience in daily and commercial applications (e.g., remote presence, holographic communication, and teleconferencing), high-quality real-time rendering with user-affordable capture rigs is an essential property of human performance capture systems.

State-of-the-art human performance capture methods are based on neural implicit functions, i.e., pixel-aligned implicit function (PIFu) [Saito et al. 2019] and neural radiance field (NeRF) [Mildenhall et al. 2020]. PIFu-based methods [Li et al. 2020a; Saito et al. 2019; Yu et al. 2021b] combine pixel-aligned image features with the occupancy fields to reconstruct human surfaces and with the surface color fields to model human textures. However, as they only model the colors of the surface points, they often fail to render high-frequency geometry/appearance details and view-dependent effects. NeRF-based methods [Gafni et al. 2021; Gao et al. 2022; Tretschk et al. 2021] model and render humans via colorized volumetric densities and integration of radiance along rays, respectively. However, NeRF-based methods typically suffer from shape-radiance ambiguity under sparse views, and are still costly to train and render due to their dense point sampling strategies. Recently, SAILOR [Dong et al. 2023] incorporates the occupancy fields and pixel-aligned RGBD features into the radiance fields for modeling human surface geometries and textures. While their results are impressive, they may still contain color artifacts and cannot be rendered in real-time. On the other hand, 3D Gaussian Splatting (3DGS) [Kerbl et al. 2023] is efficient for scene rendering by parameterizing scene radiance fields with point cloud-based Gaussians. However, 3DGS also suffers from two limitations in human performance capture. First, it cannot be generalized to unseen performers as Gaussians are optimized per scene. Second, it suffers from the local shape ambiguity problem, which significantly worsens under sparse capture settings since the Gaussian variables exceed the view number. Hence, creating high-quality free-view human videos in real-time with sparse capture rigs is still challenging.

In this work, we aim to address the above challenges (i.e., high-quality, real-time, and generalizable rendering using very sparse capture rigs) based on two observations. First, we observe that while the explicit point clouds are efficient for rendering, they often suffer from geometry/appearance inaccuracies, and neural implicit functions can complement this limitation by learning to identify accurate surface points and model their colors. Second, we observe that the lack of geometric constraints on the surface of 3D Gaussian ellipsoids causes multi-view shape ambiguities of 3D Gaussians. The first observation inspires us to formulate an efficient and accurate human representation by learning a *surface implicit function* to regress accurate surface points and a *Gaussian implicit function* to encode the radiance fields of regressed surface points for rendering via point splatting. The second observation inspires us to parameterize the radiance fields with 2D Gaussian surfels, which allow explicit normal and depth constraints to be derived.

Based on the above two observations, we propose a novel live human performance capture method for high-quality reconstruction and free-view rendering of performers, from sparse (e.g., four) RGBD cameras, as shown in Fig. 1. Our method has two main technical novelties. First, we propose a novel point-regressing network (PRNet) with a depth-guided point cloud initialization (DPI) scheme to regress human surface points. This DPI scheme leverages denoised depth information to obtain near-surface points in the reconstructed visual hull, while the PRNet regresses robust surface points from the initialization of near-surface points by learning to predict the signed distance field (SDF) value and the shifting direction for each sampled point. Second, we propose a novel neural blending-based surfel splatting network (SPNet). Instead of using 3D Gaussians, we parameterize the regressed surface points as 2D Gaussian surfels and explicitly model the normals and depths of 2D surfels. Specifically, SPNet learns to predict the attributes (i.e., scale, normal, opacity, and features) of each 2D Gaussian surfel based on the pixel-aligned RGBD features of input points (where the normals are initialized as the shifting directions predicted by PRNet). Meanwhile, SPNet also learns to predict the depth maps of the target view via the surfel splatting process. We provide explicit supervision for the predicted normals and depths of 2D surfels, significantly reducing geometric ambiguities on human surfaces. Finally, SPNet uses the predicted depth map to query and blend high-resolution features of the adjacent views to render the final result.

We evaluate our method on two standard human novel-view synthesis benchmarks, i.e., the *THuman2.0* dataset [Yu et al. 2021b] and the real-captured dataset of SAILOR [Dong et al. 2023]. Extensive experiments verify the effectiveness of our method for handling diverse gestures, motions and clothing, and its superior efficiency against existing human performance capture methods in terms of rendering accuracy. Our method can render human free-view videos of 1K resolution in real-time/live (12 fps on average) under acceleration on a single RTX 3090 GPU card.

In summary, this work has the following main contributions:

- A novel hybrid human representation that combines the surface implicit function and Gaussian implicit function for point splatting-based rendering. This representation enables a performance capture system to use a very sparse (e.g., four) RGBD capture setting, while being able to handle unseen performers and rendering 1K-resolution videos in high quality.
- A novel point-regressing network (PRNet) with a depth-guided point cloud initialization (DPI) method to regress accurate human surface points by predicting the signed distance values and shifting directions based on the denoised depth information.
- A novel neural blending-based surfel splatting network (SPNet) to explicitly model the normals and depths of 2D Gaussian surfels and incorporate high-resolution features of adjacent views for splatting-based novel-view rendering.

2 RELATED WORK

2.1 Monocular Human Performance Capture

Using monocular videos for human performance capture has become popular since the first marker-free deep method was proposed [Xu et al. 2018], in which a pre-computed T-pose textured template mesh is used for each performer as a reference to model the articulated motions and non-rigid deformations. Follow-up methods adopt the T-pose (or A-pose) mesh as the template mesh [Dou et al. 2017, 2016; Habermann et al. 2021, 2019, 2020; Li et al. 2021; Newcombe et al. 2015a, 2011; Su et al. 2020; Yu et al. 2018] and estimate the deformations from the template mesh for the reconstruction of human motions. Meanwhile, Xiang et al. [2020] use statistical deformation models for textured human reconstruction, and Zhao et al. [2022c] propose to predict the dynamic surface offsets and the texture maps based on SMPL [Loper et al. 2015]. These template-based methods typically fail to generalize well to unseen performers.

Recently, neural implicit functions have benefited monocular human capture significantly. One popular category of methods is the pixel-aligned implicit function (PIFu) [Li et al. 2020a; Saito et al. 2019, 2020], which reconstructs 3D textured surfaces by learning the occupancy and color fields based on pixel-aligned image features. Many methods incorporate PIFu with depth [Li et al. 2020b; Pesavento et al. 2024], human parsing maps [Chan et al. 2022b; Saito et al. 2020], parametric human model (e.g., SMPL [Chan et al. 2022a; Feng et al. 2022; Xiu et al. 2022; Zheng et al. 2021] and 3DMM [Cao et al. 2022]), deformation fields [He et al. 2021; Huang et al. 2020], voxel-alignment [He et al. 2020; Hong et al. 2021; Pesavento et al. 2024; Zheng et al. 2021], and multi-resolution pixel-voxel-aligned features learning [Pesavento et al. 2024]. Despite producing high-resolution reconstruction results of 3D humans with motions, PIFu-based methods may not produce view-dependent and realistic appearances as they only model the basic colors of limited surface points (vertices).

Another popular category of methods is based on the neural radiance field (NeRF) [Mildenhall et al. 2020], which models volumetric density and color fields based on coordinates. The motions in dynamic scenes are typically modeled by learning the deformation fields [Park et al. 2021a; Peng et al. 2023; Pumarola et al. 2021; Tretschk et al. 2021] in a non-rigid reconstruction-and-tracking manner [Newcombe et al. 2015b]. Parametric human models [Joo et al. 2018; Kocabas et al. 2020; Loper et al. 2015] and skeletons [Weng et al. 2022] are used as templates for the construction of deformation fields in [Chen et al. 2021b; Jiang et al. 2022; Peng et al. 2021b]. Another group of NeRF-based methods [Gafni et al. 2021; Hu et al. 2023; Su et al. 2022, 2021; Xian et al. 2021] learn conditional NeRFs to handle motions, e.g., video timestamps [Xian et al. 2021], and latent codes and morphable face/pose models [Gafni et al. 2021]. Park et al. [2021b] combine the deformation field with the conditional NeRF, and Kim et al. [2023] further introduce latent identity and pose-conditioned codes to the HumanNeRF [Weng et al. 2022] for a joint rendering of multiple performers.

2.2 Volumetric Human Performance Capture

Pioneered by [De Aguiar et al. 2008; Vlasic et al. 2008], a line of methods capture human performances in a studio setting [Collet et al. 2015; Guo et al. 2019; Işık et al. 2023; Jiakai et al. 2021; Liu et al. 2009; Vlasic et al. 2009; Wang et al. 2021c, 2022; Zhang et al. 2022; Zhao et al. 2022a], using a dense set (tens up to hundreds) of high-end RGB cameras [Işık et al. 2023; Wang et al. 2021c, 2022] or RGB/IR cameras [Collet et al. 2015; Guo et al. 2019]. Despite their success, these methods are expensive for amateur users.

A few methods are proposed to reconstruct humans using a sparse set (less than ten) of RGB(D) cameras, based on point cloud volume [Pang et al. 2021; Wang et al. 2024; Wu et al. 2020] and PIFu [Dong et al. 2022, 2023; Saito et al. 2019, 2020; Shao et al. 2022a; Yu et al. 2021b]. Among them, Function4D [Yu et al. 2021b] combines a depth-based local tracking and fusion scheme with detail-preserving PIFu for surface reconstruction/texturing. FNHR [Pang et al. 2021] uses depth to optimize a global 3D skeleton to select keyframes for few-shot learning, and combines point rendering and classical mesh texturing for rendering. Wang et al. [2024] propose to render portraits and backgrounds by constructing the depth-tolerable Multi-layer Point Cloud volume and leveraging volumetric rendering in novel view synthesis [Fridovich-Keil and Yu et al. 2022]. However, noisy depth or unreliable point cloud data typically introduces inaccuracy into geometric/texture modeling. In contrast, our PGH representation learns an accurate surface implicit function through DPI and the derived supervision for PRNet. The regressed surface points facilitate subsequent surfel-based rendering.

There are also many human rendering methods based on the generalizable NeRF [Chen et al. 2021a; Yu et al. 2021a]. Some methods focus on modeling the deformations in NeRF by leveraging 3D body parametric models [Gao et al. 2022; Kwon et al. 2021; Liu et al. 2021], neural blending field and skeleton estimation [Peng et al. 2021a], image-based rendering [Cheng et al. 2022; Kwon et al. 2023; Wang et al. 2021b; Zhao et al. 2022b], 3D keypoint detection [Mihajlovic et al. 2022], and pose estimation [Liu et al. 2021; Remelli et al. 2022; Zhao et al. 2022b]. Some recent methods propose to improve the geometry accuracy of NeRF, by jointly regressing occupancy and densities [Shao et al. 2022b], and incorporating the depth probability distribution [Lin et al. 2022]. Most recently, Dong et al. [2023] propose to combine PIFu and NeRF representations based on the pixel-aligned RGBD features for surface reconstruction and appearance rendering. Nonetheless, the costly training and inference overheads are a fundamental limitation of using NeRF for rendering.

2.3 3D Gaussian Splatting

3D Gaussian Splatting (3DGS) [Kerbl et al. 2023] has recently become a very popular alternative to NeRF [Mildenhall et al. 2020], due to its high efficiency. Instead of querying a dense set of points along the ray for rendering a pixel in NeRF, 3DGS rasterizes a few Gaussian points with independent attributes, and renders the pixel via point-based alpha blending. As a result, 3DGS can render images of comparable quality but is significantly faster than NeRF.

Many works immediately follow 3DGS. Some methods propose to extend 3DGS by, e.g., improving surface reconstruction accuracy [Dai et al. 2024; Huang et al. 2024; Yu et al. 2024b], incorporating the deformation field for handling dynamic scenes [Wu et al. 2024; Yang et al. 2023], focusing on removing aliasing artifacts [Yan et al. 2023; Yu et al. 2024a], and determining the minimum number of Gaussians via Markov Chain Monte Carlo (MCMC) sampling [Kheradmand et al. 2024]. 3DGS has also been widely adopted for various

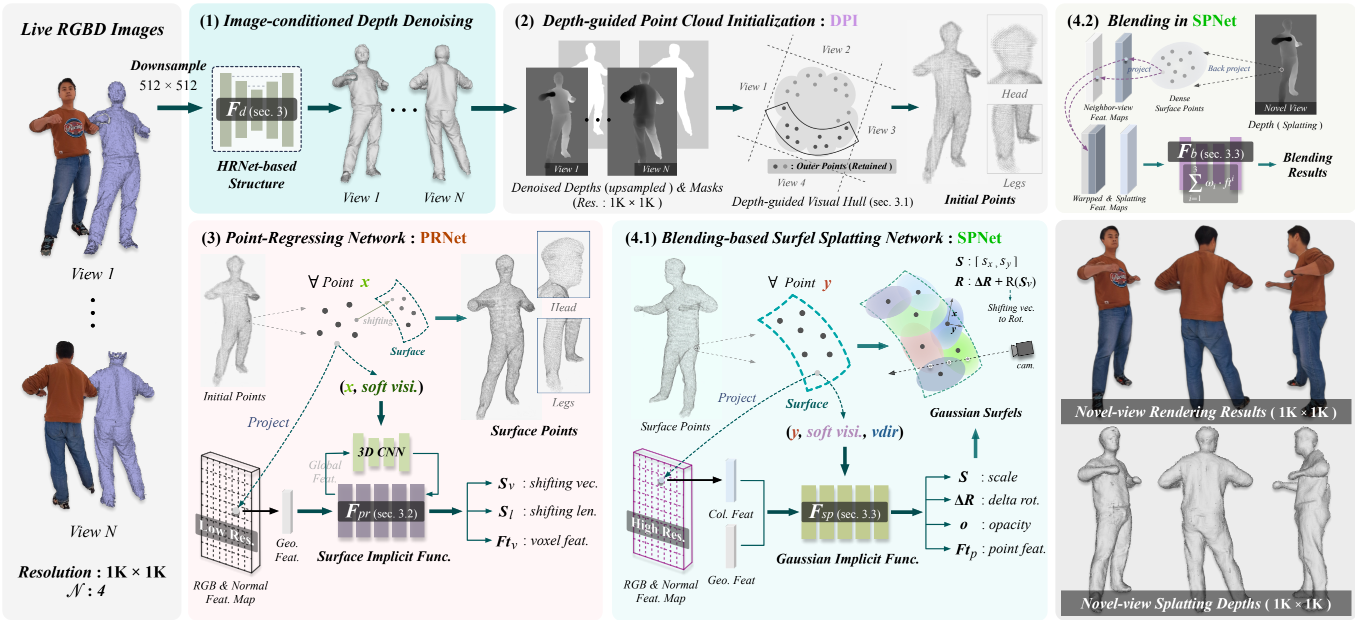

Fig. 2. Given a 4-view RGBD live stream captured via Azure Kinects as input: **(1)** we first apply a *depth denoising* module [Dong et al. 2023] to reduce the noise in raw depth; **(2)** a *Depth-guided Point Cloud Initialization* (DPI) method then leverages visual hull with depth guidance to construct a volume (closed) point set, which is near to the surface; **(3)** a novel *Point-Regressing Network* (PRNet) is proposed to learn a surface implicit function to regress the surface points; **(4.1)** A *Surfel Splatting* network (SPNet) is proposed to parameterize the radiance field as Gaussian Surfels via learning a Gaussian implicit function; and **(4.2)** the splatting outputs are further enhanced by an *Appearance Blending* scheme to render novel-view images in 1K resolution. Geo., Col., visi., Feat., vec., len., rot., cam., Func., Res. are abbreviations for Geometric, Colorimetric, visibility, Feature, vector, length, rotation, camera, Function, and Resolution, respectively.

applications, e.g., (text/image driven) 3D content generation [Abdal et al. 2023; Liu et al. 2024; Tang et al. 2024; Yinghao et al. 2024; Zhou et al. 2024], animatable head/human avatar reconstruction [Kocabas et al. 2024; Lei et al. 2024; Li et al. 2024; Moreau et al. 2024; Qian et al. 2024; Shao et al. 2024; Xu et al. 2024a; Zielonka et al. 2023], and controllable portrait generation [Rivero et al. 2024].

There are some concurrent 3DGS-based methods proposed for human performance capture [Hu and Liu 2024; Xu et al. 2024b; Zheng et al. 2024]. Hu and Liu [2024] propose a monocular performance capture method, which uses SMPL [Loper et al. 2015] vertex points to initialize 3D Gaussian points in the canonical space and transforms the initialized Gaussians to the target space via linear blend skinning (LBS) predictions. Like all the other monocular-based methods, this method suffers from serious occlusion problems. Xu et al. [2024b] propose a multi-view performance capture method, which first reconstructs a point cloud using space carving [Kiriakos and Steven 2000] and then maps it to 4D feature space using the K-Planes method [Fridovich-Keil et al. 2023]. While they use 3DGS to model the dynamic geometry, a hybrid appearance model combining image blending and spherical harmonics is proposed to support pre-computation for rendering efficiency. However, this method requires dozens of cameras to capture multi-views for reconstruction, which is expensive for casual applications. Zheng et al. [2024] propose the GPS-Gaussian, which captures human performance using a sparse set of eight RGB cameras. GPS-Gaussian uses RGB images of two adjacent source views and the derived depth images as the 3D position and color maps of 3DGS, and predicts other 3DGS attributes (i.e., scaling factor, opacity, and rotation) in a pixel-wise manner. The predicted Gaussian maps of source views are then deprojected to 3D space and aggregated for rendering the target view. However, their method tends to produce results of inaccurate local geometry under very sparse (i.e., four views in our case) capture settings as their stereo matching-based depth estimator assumes that adjacent views are similar to each other in order to produce sufficiently accurate depth maps.

This work proposes a novel point-based generalizable human (PGH) representation for fast and high-quality human rendering using four RGBD cameras. PGH explores the point representation for fast rendering, and addresses its geometry/appearance ambiguities by incorporating two novel functions, a surface implicit function and a Gaussian implicit function, as two networks, PRNet and SPNet.

3 PROPOSED METHOD

We aim to explore Gaussian Splatting to address the challenges of achieving high-quality, near real-time, and generalizable rendering with very sparse capture rigs. As illustrated in Fig. 2, our system generates photometrically correct free-view videos in near real-time performances, from live $\mathcal{N}$-view ($\{\mathbf{I}^i, \mathbf{D}^i, \mathbf{M}^i\}_{i=1,\ldots,\mathcal{N}}$) RGBD stream captured by Kinect-V4 sensors, where $\mathbf{I}$, $\mathbf{D}$ and $\mathbf{M}$ denote the RGB, depth and mask images, respectively, and $\mathcal{N}$ is set to four in our implementation.

Our system consists of four steps: **(1)** *Image-conditioned Depth Denoising* ($\mathcal{F}_d$) [Dong et al. 2023] removes the undesirable noise and holes in the raw captured depths. The denoised depths are denoted as $\mathbf{D}^i_{rf}$; **(2)** *Depth-guided Point Cloud Initialization* (DPI) constructs the initial point cloud $\mathbf{P}_{init}$, by leveraging the multi-view

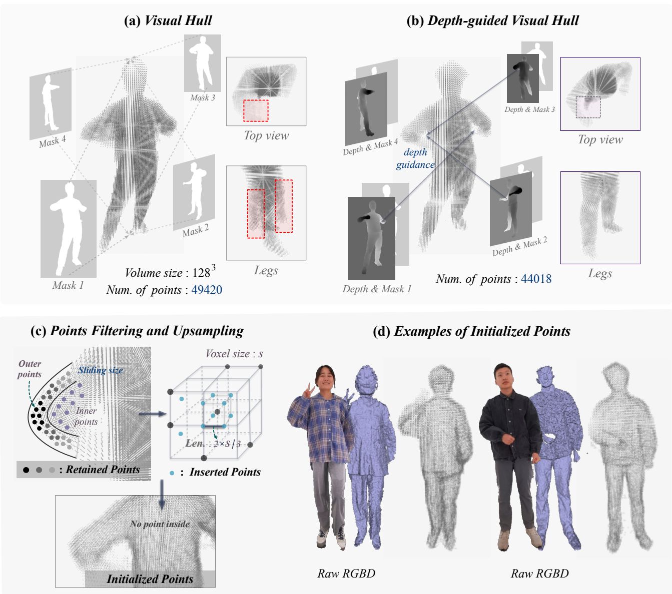

Fig. 3. The *Depth-guided point Cloud Initialization* method. (a) Building a closed visual hull from 4-view body mask images; (b) Leveraging depth data to remove the extra points (purple boxes) inside the visual hull; (c) Using a sliding voxel to filter out the inner points and obtain the outer points. Interpolating sub-points with a length of 1/3 voxel size in the 8-diagonal directions of each point, to upsample the filtered outer points; (d) Given raw RGBD data, the initial surface points of two examples.

denoised depths and masks; **(3)** *Point-Regressing Network* (PRNet) learns a surface implicit function to predict an accurate surface point cloud $\mathbf{P}_{sf}$, with inputs $\mathbf{P}_{init}$ and low-resolution RGB-N features; **(4)** *Blending-based Surfel Splatting Network* (SPNet) learns a Gaussian implicit function to represent the geometry and texture features as Gaussian surfels for each surface point in $\mathbf{P}_{sf}$, and then use Gaussian surfel splatting [Dai et al. 2024] to render the feature ($\tilde{\mathbf{Ft}}$), depth ($\tilde{\mathbf{D}}$) and normal ($\tilde{\mathbf{N}}$) maps at the target view. Finally, an *Appearance Blending* module decodes the feature maps to a coarse RGB image, and then uses $\tilde{\mathbf{D}}$ to aggregate the neighbor-view pixel-aligned RGB-N features to enhance rendering details.

3.1 Depth-guided Point Cloud Initialization (DPI)

Initializing a high-quality point cloud is important as it provides effective geometric constraints and guidance for improving the rendering efficiency of the point-based rendering methods. To this end, we propose DPI, which builds a near-surface volume point cloud $\mathbf{P}_{init}$ based on the reliable denoised depth and the visual hull. It leverages $\mathbf{D}^i_{rf}$ to remove the far-from-surface points (Fig. 3(a)) contained in the rough visual hull constructed by masks $\mathbf{M}$.

Since the visual hull can express a closed geometry like the human body, thus avoiding the missing-region problem during rendering, we first adopt the silhouette-based visual hull [Laurentini 1994] to convert the masks $\mathbf{M}^i$ ($i = 1, ..., \mathcal{N}$) into a full-body solid point set, where each point p belongs to the grid vertices of a volume $\mathbf{V}_c$ at a resolution of 128, as shown in Fig. 3(a). However, $\mathbf{V}_c$ cannot accurately represent complex surface details and contains many extra points (red dot boxes), especially under a very sparse capture

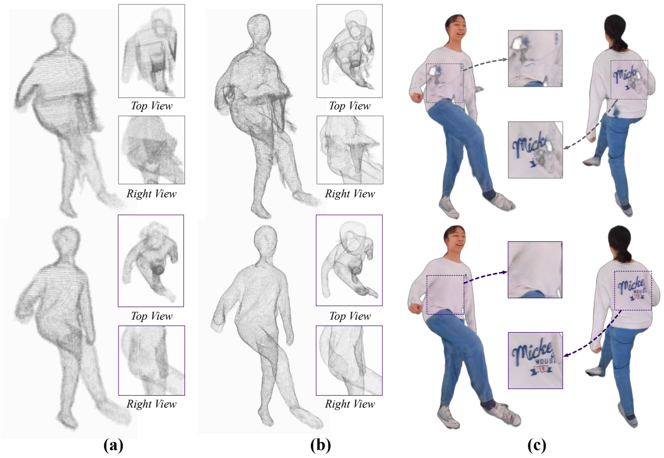

Fig. 4. Visual examples of initial point clouds (a), regressed surface points (b), and rendering results (c) without (first row) and with (second row) *Depth-guided Visual Hull*. Our DPI can remove the far-from-surface points, improving the quality of the regressed surface points and novel-view rendering.

setting. To effectively remove these extra points and build a fine volume $\mathbf{V}_f$, we utilize the depth $\mathbf{D}^i_{rf}$ to determine whether point p needs to be removed according to the relative depth fetched by projecting p into the i-th input view, as:

$$\mathbf{V}_f[\mathbf{x}] = \begin{cases} \prod_{i=1}^{\mathcal{N}} (d^i_{rf}(p) - z^i(p) < \tau) & \mathbf{V}_c[\mathbf{x}] = 1 \\ 0 & \text{otherwise} \end{cases}, \quad (1)$$

where $\mathbf{x} = idx(p)$ is the 3D index of p in the volume $\mathbf{V}_c$. $z^i(p)$ and $d^i_{rf}(p)$ are the projected z-value and the sampled depth-value of $\mathbf{D}^i_{rf}$ in the i-th view, respectively. τ is a depth threshold, set to 0.02 in our implementation [1].

As shown in Fig. 3(b) and Fig. 4, our depth-guided visual hull can effectively remove the extra points (purple box, top and right views), especially for some regions with serve occlusion, such as arms and legs, thus reducing the number of points and producing a better rendering result. To further retain the near-surface points and increase the density of these points, we first propose to use a sliding voxel of size $\tau/s(\mathbf{V}_f)$ to filter the outer points of $\mathbf{V}_f$, where $s(\cdot)$ is the voxel size of $\mathbf{V}$, i.e., for each point p with $\mathbf{V}_f[\mathbf{x}] = 1$, if the sliding voxel contains a point with $\mathbf{V}_f[\cdot] = 0$, then the point is determined as an outer point and retained. We then propose to upsample the point cloud to obtain $\mathbf{P}_{init}$ (Fig. 3(d)) by interpolating sub-points with the step size of 1/3 voxel size in 8-diagonal directions for each filtered point, to enrich the rendering details, as shown in Fig. 3(c). We have implemented *DPI* with CUDA acceleration (≈6ms), which supports processing multiple point clouds simultaneously.

3.2 Point-Regressing Network (PRNet)

To obtain an accurate surface point cloud for high-quality point-based rendering, we propose PRNet to learn a generalizable human signed distance field (SDF) for the initial point cloud $\mathbf{P}_{init}$, which

[1] We empirically found that it does not affect the performance when τ is set to [0.02, 0.04], which tolerates outside-surface points and covers the majority of errors in the denoised depth maps. The performance drops when $\tau < 0.02$ as surface points may be discarded, while $\tau > 0.04$ does not improve the performance (causing more far-from-surface points) but reduces the rendering speed.

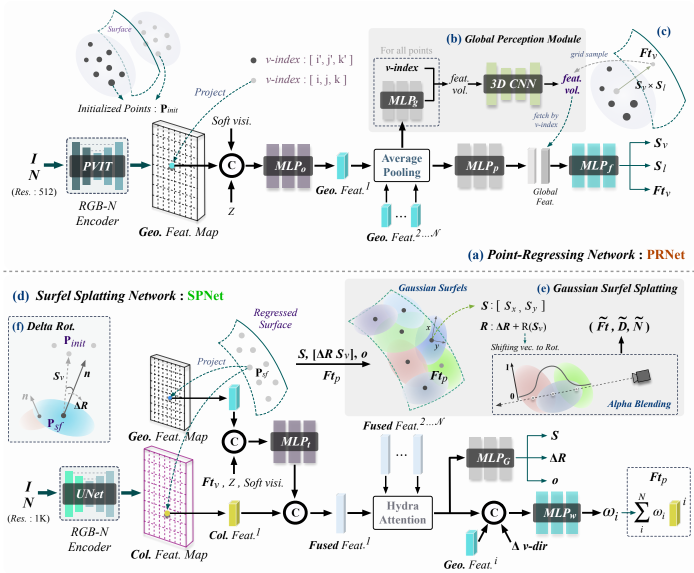

Fig. 5. The structures of our *PRNet* (a) and *SPNet* (d). *PRNet* takes multi-view RGB and Normal (denoted as RGB-N) images at 512 resolution as inputs, and predicts the shifting direction and shifting length for each point of the initialized point cloud. It also provides the voxel features of the regressed surface points (c) by re-sampling the encoded global feature volume (b). *SPNet* predicts the attributes (i.e., scale, Δrotation, opacity, and features) of the regressed surface points from RGB-N images at 1K resolution to build 2D Gaussian surfels, for rendering the feature, depth, and normal maps of the novel view via surfel splatting (e). Note that the final normal vector of a Gaussian surfel is expressed as the resulting direction after rotating shifting direction by a delta quaternion (f).

regresses the human-body surface points utilizing the signed distance values and the point-shifting directions. Our PRNet learns a *surface implicit function* ($\mathcal{F}_{pr}$) conditioned on the pixel-aligned RGB-N features, which provide texture and local geometric information for fitting the surface correctly, where the input normal map $\mathbf{N}_{rf}$ is computed from the points converted by the depth $\mathbf{D}_{rf}$. As a result, we obtain the regressed surface points $\mathbf{P}_{sf}$ by moving the initial points $\mathbf{P}_{init}$ following the predicted shifting vectors. Fig. 5 (a) illustrates the network structure of our PRNet.

3.2.1 Surface Implicit Function: $\mathcal{F}_{pr}$. For a point $\mathbf{x}$ in point cloud $\mathbf{P}_{init}$, $\mathcal{F}_{pr}$ predicts its shifting length (i.e., signed distance value) $S_l(\mathbf{x})$, and shifting direction $\mathrm{S}_v(\mathbf{x})$ (unit vector outward along the human surface), by aggregating the pixel-aligned RGB-N features. Hence, the regressed surface point $\tilde{\mathbf{x}}$, corresponding to $\mathbf{x}$, can be written as: $\tilde{\mathbf{x}} = \mathbf{x} + \mathrm{S}_v(\mathbf{x}) \cdot S_l(\mathbf{x})$. In addition to the point-independent features, considering that the point-shifting information of $\mathbf{x}$ is also affected by its surrounding points, we introduce a *Global Perception Module* (Fig. 5(b)) to extract the volume features $\mathbf{G}_{init}$ of point cloud $\mathbf{P}_{init}$ using a 3D convolution network. We can then obtain the global features of a point by sampling $\mathbf{G}_{init}$ according to the point index. We describe the implicit function $\mathcal{F}_{pr}$ as:

$$\mathcal{F}_{pr}(\mathbf{x}, \mathbf{I}, \mathbf{N}_{rf}) = f_3(\mathbf{G}_{init}(\mathbf{x}), f_2(ft_{geo}(\mathbf{x}))) := S_l(\mathbf{x}), \mathrm{S}_v(\mathbf{x}), \mathbf{Ft}(\tilde{\mathbf{x}}), \tag{2}$$

where $ft_{geo}(\mathbf{x}) = Avg(\{f_1(\mathbf{W}^i(\mathbf{x}), \mathbf{c}^i(\mathbf{x}))\}_{i=1,\ldots,\mathcal{N}})$ are the multi-view aggregated features of $\mathbf{x}$. $\mathbf{W}^i = E_{geo}(\{\mathbf{I}^i, \mathbf{N}^i_{rf}\})$ is the RGB-N feature map of the i-th view, and $E_{geo}(\cdot)$ is the PVIT network [Wang et al. 2021d] to encode RGB-N images. For sampling the pixel-aligned RGB-N features, we project point $\mathbf{x}$ to the image space to obtain the coordinate $\pi^i(\mathbf{x})$ and fetch z-value z^i of $\mathbf{x}$ in view i. $\mathbf{W}^i(\mathbf{x})$ is the fetched RGB-N feature vector at $\pi^i(\mathbf{x})$ and $\mathbf{c}^i(\mathbf{x}) = [z^i, p^i(\mathbf{x})]$, where $p^i(\mathbf{x}) = tanh(\sigma_v \cdot (d^i_{rf} - z^i))$ is a soft-visibility signal and σ_v is set to 200 in our implementation. $\mathbf{W}^i(\mathbf{x})$ along with $\mathbf{c}^i(\mathbf{x})$ are then fed into an MLP_o (noted as f_1), and then processed by an average pooling operation to obtain the geometric features $ft_{geo}(\mathbf{x})$. We then use an MLP_g to prepare the input volume of *3D-CNN* to obtain $\mathbf{G}_{init}$. For sampling the voxel-aligned features, we first obtain the 3D-index of $\mathbf{x}$ in volume $\mathbf{V}_f$, and then use the 3D grid-sampling operation to fetch the interpolated feature vector in $\mathbf{G}_{init}$. The sampled features are referred to as $\mathbf{G}_{init}(\mathbf{x})$. Finally, the post-processed point geometric features $f_2(ft_{geo}(\mathbf{x}))$ by MLP_p (noted as f_2), along with the point global features $\mathbf{G}_{init}(\mathbf{x})$ are fed into the last MLP_f (noted as f_3) for shifting vector querying. For the regressed surface point $\tilde{\mathbf{x}}$, we also obtain the voxel-aligned features $\mathbf{Ft}(\tilde{\mathbf{x}})$ by sampling the learned feature volume $\mathbf{G}_{init}(\tilde{\mathbf{x}})$, which serves as a geometric cue for the subsequent rendering process.

3.2.2 Loss Functions for PRNet. We adopt SDF and chamfer-distance loss functions to supervise the learning of surface implicit function $\mathcal{F}_{pr}$. We first sample point $\mathbf{y}$ around the human-body surface (denoted as PIFuHD [Saito et al. 2020]), and compute the ground-truth SDF value (denoted as S^*_l) between $\mathbf{y}$ and the 3D scanned mesh. We then measure the difference between the predicted length S_l and the ground-truth SDF value S^*_l to learn the SDF field. Besides, we enhance the similarity of the regressed point cloud $\mathbf{P}_{sf}$ to the ground-truth point set $\mathbf{P}_{gt}$ (vertices of the 3D scanned mesh) by measuring the chamfer distance between $\mathbf{P}_{sf}$ and $\mathbf{P}_{gt}$. The overall loss function can be written as:

$$L_{pr} = \mu_S \cdot \sum_{\mathbf{y} \in T} \mathcal{L}_1(S_l, S^*_l) + \mu_P \cdot \mathcal{L}_c(\mathbf{P}_{sf}, \mathbf{P}_{gt}), \tag{3}$$

where T denotes the sampled point set. $\mathcal{L}_1$ and $\mathcal{L}_c$ denote the smooth L1 loss and chamfer-distance loss, respectively. μ_S and μ_P are the balancing weights, which are set to 1.0 and 10.0, respectively.

3.3 Blending-based Surfel Splatting Network (SPNet)

We propose SPNet to exploit the geometry friendliness and rendering efficiency of Gaussian surfel splatting [Dai et al. 2024], and then enhance the rendering details based on appearance blending. Our SPNet learns a generalizable *Gaussian implicit function* ($\mathcal{F}_{sp}$) for the sparse surface points $\mathbf{P}_{sf}$, to express Gaussian-surfel attributes conditioned on the high-resolution RGB-N features. Based on surfel splatting, SPNet outputs the target-view depth ($\tilde{\mathbf{D}}$), normal ($\tilde{\mathbf{N}}$) and feature ($\tilde{\mathbf{Ft}}$) maps. Appearance blending then enhances the texture features of $\tilde{\mathbf{Ft}}$ pixel by pixel through the dense surface points (denoted as $\mathbf{P}_{\tilde{\mathbf{D}}}$), converted from $\tilde{\mathbf{D}}$, to generate the final rendering results. Fig. 5(d) and Fig. 6 illustrate our SPNet.

3.3.1 Gaussian Implicit Function: $\mathcal{F}_{sp}$. For a surface point in the set $\mathbf{P}_{sp}$, $\mathcal{F}_{sp}$ predicts the attributes of Gaussian surfel $\tilde{\mathbf{x}}_g$, i.e., 2D-scale $\mathrm{S}_g \in \mathbb{R}^2$, opacity value o, delta rotation $\Delta\mathbf{R} \in \mathbb{R}^4$ (quaternion) and feature vector $\mathbf{Ft}_p$, by aggregating the pixel-aligned RGB-N features.

Considering that the normal vector of the surfel $\tilde{\mathbf{x}}_g$ is close to the previous shifting direction $\mathbf{S}_v(\mathbf{x})$, we propose to express its normal vector as the resulting direction after rotating $\mathbf{S}_v(\mathbf{x})$ by $\Delta\mathbf{R}$ (refer to Fig. 5(f)), as: $M(\Delta\mathbf{R}) \cdot \mathbf{S}_v(\mathbf{x})$, where $M(\cdot)$ denotes the function of quaternion to rotation matrix. Learning a residual rotation facilitates the PRNet via reducing the complexity of the implicit function $\mathcal{F}_{sp}$ and suppressing the geometry and rendering errors (Fig. 13(d)).

In addition to use $\mathbf{S}_v$, our $\mathcal{F}_{sp}$ also uses the encoded feature map $\mathbf{W}^i$ and the resampled point global features $\mathbf{Ft}(\tilde{\mathbf{x}})$, to introduce the geometric constraints for further reducing the model complexity of learning Gaussian attributes. $\mathcal{F}_{sp}$ can be written as:

$$\mathcal{F}_{sp}(\tilde{\mathbf{x}}, \mathbf{I}, \mathbf{N}_{rf}) = [f_5(ft_u), f_6(\{ft_u, ft_g^i, \Delta\mathbf{d}^i\})] := [\{\mathbf{S}_g, \Delta\mathbf{R}, o\}, \omega^i], \tag{4}$$

where $ft_g^i = f_4(\{\mathbf{W}^i(\tilde{\mathbf{x}}), \mathbf{c}^i(\tilde{\mathbf{x}}), \mathbf{Ft}(\tilde{\mathbf{x}})\})$ are the resampled geometric features of point $\tilde{\mathbf{x}}$. $ft_u = \mathcal{H}(Concat(\{ft_g^i, \mathbf{C}^i(\tilde{\mathbf{x}})\})_{i=1,\ldots,\mathcal{N}})$ are the fused geometric and colorimetric features of $\tilde{\mathbf{x}}$, where $\mathbf{C}^i = E_c(\{\mathbf{I}^i, \mathbf{N}_{rf}^i\})$ is the high-resolution RGB-N feature map in view i, and $E_c(\cdot)$ is a UNet-like encoder. $\mathbf{C}^i(\tilde{\mathbf{x}})$ is the fetched RGB-N feature vector, and $\mathcal{H}$ denotes the transformer encoder [Vaswani et al. 2017] with hydra attention blocks [Bolya et al. 2023]. The MLP_G (noted as f_5) and MLP_ω (noted as f_6) are used to decode Gaussian attributes and fusion weight $\omega^i(\tilde{\mathbf{x}})$, respectively. The surface point features are the weighted sum of $\mathbf{C}^i(\tilde{\mathbf{x}})$, i.e., $\sum_i \omega^i \cdot \mathbf{C}^i(\tilde{\mathbf{x}})$, denoted as $\tilde{\mathbf{F}}$.

For novel-view rendering, we use the alpha-blending function, consistent with that of 3DGS [Kerbl et al. 2023], to obtain the target feature vector of each pixel $\mathbf{u}$ in image space, as:

$$\tilde{\mathbf{Ft}}(\mathbf{u}) = \sum_{i=1}^{n} T_i \alpha_i \tilde{\mathbf{F}}_i, \quad T_i = \prod_{j<i}(1-\alpha_j), \tag{5}$$

where α_i is the alpha-blending weight, which is the product of the predicted opacity value o_i and a Gaussian weight [Kerbl et al. 2023]. n is the number of Gaussian surfels hit by the emitted ray of pixel $\mathbf{u}$. $\tilde{\mathbf{Ft}}(\mathbf{u})$ is the feature vector of feature map $\tilde{\mathbf{Ft}}$ located at pixel $\mathbf{u}$.

We also compute depth value $\tilde{\mathbf{D}}(\mathbf{u})$ and normal vector $\tilde{\mathbf{N}}(\mathbf{u})$ as:

$$\tilde{\mathbf{N}}(\mathbf{u}) = w \sum_{i=1}^{n} T_i \alpha_i \cdot M(\Delta\mathbf{R}_i) \cdot \mathbf{S}_v(\mathbf{x}_i), \quad \tilde{\mathbf{D}}(\mathbf{u}) = w \sum_{i=1}^{n} T_i \alpha_i d_i(\mathbf{u}), \tag{6}$$

where $w = 1/(1 - T_{n+1})$ is the normalizing weight for the blending weight $T_i\alpha_i$, referring to [Dai et al. 2024]. $M(\Delta\mathbf{R}_i) \cdot \mathbf{S}_v(\mathbf{x}_i)$ is the final normal vector of the Gaussian surfel $\tilde{\mathbf{x}}_g$. $d_i(\mathbf{u})$ is the camera-space depth value of the ray-surfel intersected point.

3.3.2 Appearance Blending: $\mathcal{B}$. We propose an appearance blending scheme in SPNet to use the dense surface points $\mathbf{P}_{\tilde{\mathbf{D}}}$, which is converted from depth $\tilde{\mathbf{D}}$ using back-projection, to enhance the target-view texture details pixel by pixel. Fig. 6 illustrates the detailed structure of our blending scheme.

For a target view t, we first obtain its two adjacent high-resolution RGB-N feature maps, $\mathbf{C}^{n_0}$ and $\mathbf{C}^{n_1}$, where n_0 and n_1 are the view indices. We then project a point $\mathbf{y} \in \mathbf{P}_{\tilde{\mathbf{D}}}$ (corresponding to the pixel $\mathbf{u}$ in view t) into the two views, to fetch the pixel-aligned RGB-N features ($\mathbf{C}^{n_0}(\mathbf{y})$ and $\mathbf{C}^{n_1}(\mathbf{y})$), z-values ($z_{\mathbf{y}}^{n_0}$ and $z_{\mathbf{y}}^{n_1}$), and depth values ($\mathbf{D}_{sp}^{n_0}(\mathbf{y})$ and $\mathbf{D}_{sp}^{n_1}(\mathbf{y})$). Here, $z_{\mathbf{y}}^i$ is the projected depth of the point $\mathbf{y}$ in view i, and $\mathbf{D}_{sp}^i(\mathbf{y})$ is the depth value sampled from the

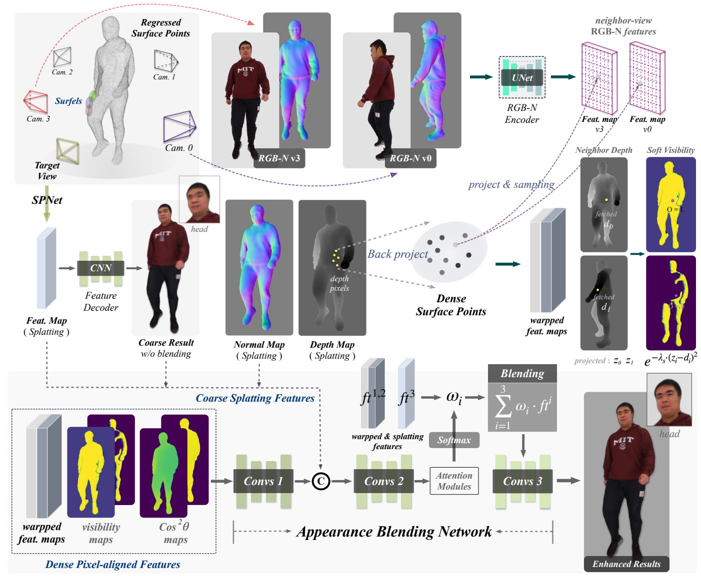

Fig. 6. Overview of our *Appearance Blending* scheme in *SPNet*. We fetch two neighbor-view high-resolution RGB-N feature maps for each target view (1st row). The surfel splatting predicts the feature, normal, and depth maps of the target view, and a CNN-based decoder outputs the coarse rendering results. The rendered depth is back-projected to neighbor views to get the pixel-aligned features and depth values to form the warped feature maps and soft-visibility maps, respectively (2nd row). The blending network takes the feature maps, two visibility maps, and viewing weighted maps as inputs, to produce pixel-by-pixel weights to blend the warped and the splatting features, for decoding the final rendering results (3rd row).

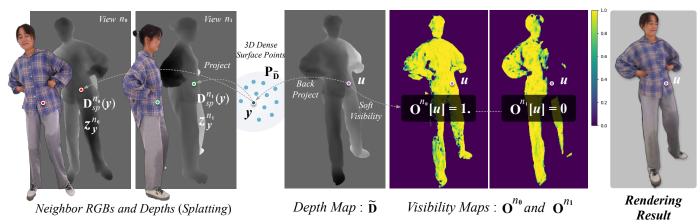

Fig. 7. A visualization example of calculating the soft-visibility maps and rendering result. The point $\mathbf{y} \in \mathbf{P}_{\tilde{\mathbf{D}}}$ corresponding to the pixel $\mathbf{u}$ in the novel view is visible in the viewport n_0, but invisible in the viewport n_1.

2D coordinate $\pi^i(\mathbf{y})$ in the depth map $\mathbf{D}_{sp}^i$, where $\mathbf{D}_{sp}^i$ is computed using the surfel splatting equation (Eq. 6).

Based on the obtained depths $z_{\mathbf{y}}^i$ and $\mathbf{D}_{sp}^i(\mathbf{y})$, we can compute a soft visibility map $\mathbf{O}^i$, as: $\mathbf{O}^i[\mathbf{u}] = \exp(-\lambda_s \cdot (z_{\mathbf{y}}^i - \mathbf{D}_{sp}^i(\mathbf{y}))^2)$, where λ_s is a weight coefficient determined by depth units (set to 800 in our implementation). As shown in Fig. 7, $\mathbf{O}^i[\mathbf{u}]$ tends to be 1 when $\mathbf{y}$ is visible in view i, and 0 otherwise. Based on the fetched features $\mathbf{C}^i(\mathbf{y})$, we can get two warped feature maps, ft^i, where $ft^i[\mathbf{u}] = \mathbf{C}^i(\mathbf{y})$. Considering that the blending weight is affected by the angle distances between views n_0 and t and between views n_1 and t, we also introduce the *cos* map, denoted as $\mathbf{Cos}^i$, to model this view-sweeping effect, and $\mathbf{Cos}^i[\mathbf{u}] = (\vec{\mathbf{v}}_{\mathbf{y}}^t \cdot \vec{\mathbf{v}}_{\mathbf{y}}^i)^2$, where $\vec{\mathbf{v}}_{\mathbf{y}}^i$ is the normalized viewing direction of $\mathbf{y}$ in view i. For the appearance blending network, we first decode the feature map $\tilde{\mathbf{Ft}}$ using a convolution network D_c to obtain the coarse RGB image $\tilde{\mathbf{C}}$, and

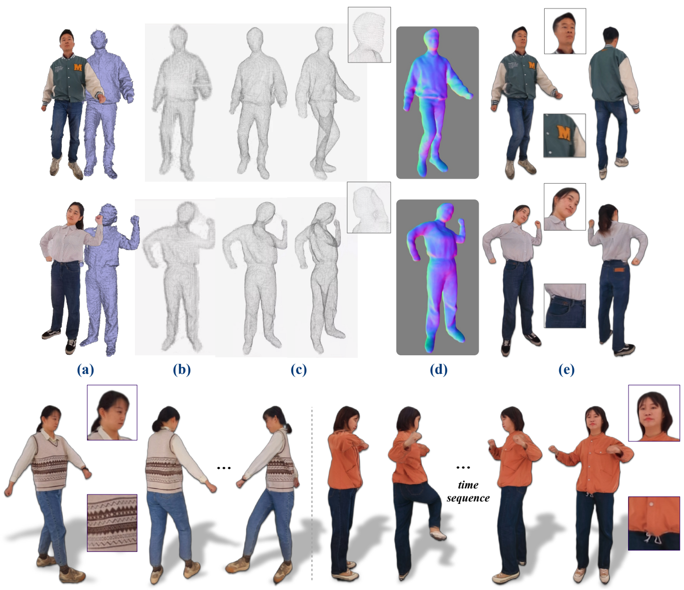

Fig. 8. Visualization of our point cloud and rendering results. One-of-four raw RGBD inputs (a), our initialized point cloud (b), the surface point cloud obtained through our *PRNet* (c), the normal maps obtained by surfel splatting in our *SPNet* (d), our final rendering results (e). The novel-view rendering gallery over a time period (bottom row).

then feed two-view maps ft^i, $\mathbf{O}^i$ and $\mathbf{Cos}^i$ to a CNN to obtain the neighbor-view weighing features. The features are fed into another CNN with spatial and channel attention [Woo et al. 2018]. After being enhanced by the target-view features, the CNN predicts the blending weight maps $\mathbf{W} = [\omega_t, \omega_{n_0}, \omega_{n_1}]$, expressed as:

$$\mathbf{W} = c_1(\ Concat(c_0(\{ft^i, \mathbf{O}^i, \mathbf{Cos}^i\}_{i=n_0,n_1}), \{\tilde{\mathbf{F}}\mathbf{t}, \tilde{\mathbf{C}}, \tilde{\mathbf{N}}\})\ ). \quad (7)$$

$\mathbf{W}$ is then used to blend the warped features and the splatting features. The enhanced rendering result $\tilde{\mathbf{C}}_e$ is obtained via: $c_2(\mathbf{W} \cdot [ft^{n_0}, ft^{n_1}, \tilde{\mathbf{F}}\mathbf{t}])$, where c_0, c_1, c_2 are three convolutional networks.

3.3.3 Loss Functions for SPNet. We adopt the photometric, depth-prior, and normal-prior loss functions to supervise our SPNet.

1) Photometric Loss L_p. We penalize the per-pixel color error between the ground-truth RGB image (denoted as $\mathbf{C}^*$) and the coarse RGB image $\tilde{\mathbf{C}}$, and between $\mathbf{C}^*$ and the enhanced RGB image $\tilde{\mathbf{C}}_e$, as:

$$L_p = \lambda_c \cdot \mathcal{L}_g(\tilde{\mathbf{C}}, \mathbf{C}^*) + \lambda_e \cdot (\ \mathcal{L}_g(\tilde{\mathbf{C}}_e, \mathbf{C}^*) + \lambda_{vgg} \cdot \mathcal{L}_{vgg}(\tilde{\mathbf{C}}_e, \mathbf{C}^*)\ ), \quad (8)$$

where $\mathcal{L}_g$ is the rendering loss following 3DGS [Kerbl et al. 2023]. $\mathcal{L}_{vgg}$ is a perceptual loss used for $\tilde{\mathbf{C}}_e$, which computes the L_1 loss between VGG features. λ_c and λ_e are two-stage weights, set to 0.85 and 1.0, respectively. λ_{vgg} is set to 0.01.

2) Depth Consistency Loss L_d. L_d penalizes the per-pixel depth error between the ground-truth depth map (denoted as $\mathbf{D}^*$) and $\tilde{\mathbf{D}}$:

$$L_d = \mathcal{L}_1(\ \tilde{\mathbf{D}}, \mathbf{D}^*\ ), \quad (9)$$

where $\mathcal{L}_1$ denotes the L1 loss. L_d is an important term that provides geometric constraints for $\mathcal{F}_{sp}$ and appearance blending (Fig. 13(b)).

3) Normal Consistency Loss L_n. L_n penalizes the per-pixel normal error between the $\tilde{\mathbf{N}}$ and monocular normal map $\mathbf{N}^*$ (transformed from $\mathbf{D}^*$) with angular and L1 losses, as:

$$L_n = \mathcal{L}_1(\ \tilde{\mathbf{N}}, \mathbf{N}^*\ ) + \mathcal{L}_1(\ 1,\ \tilde{\mathbf{N}}(\mathbf{u})^T\mathbf{N}^*(\mathbf{u})\ ), \quad (10)$$

where $\mathbf{N}^*(\mathbf{u})$ and $\mathbf{N}(\mathbf{u})$ denote the ground-truth (denoted as GT) and predicted normal vector, respectively, at pixel $\mathbf{u}$. The overall loss for our SPNet is defined as: $Lp + \lambda_d L_d + \lambda_n L_n$, where the balancing weights λ_d and λ_n are set to 0.8 and 0.5, respectively.

Table 1. The running time for each stage of our rendering system w/o and $w/$(using a single RTX 3090 GPU) acceleration is reported. The overall speed increased by ≈ 4 times with acceleration.

Stages	Operations	Time w/o acc.	Time $w/$ acc.
$\mathcal{F}_d$	Depth denoising	≈ 55.4ms	≈ 16.5ms
DPI	Depth-guided point cloud $\mathbf{P}_{init}$ initialization	≈ 5.9ms	-
PVIT	Encoding down-sampled RGB-N images in PRNet	≈ 20.4ms	≈ 9.5ms
UNet	Encoding high-resolution RGB-N images in SPNet	≈ 42.3ms	≈ 9.7ms
$\mathcal{F}_{pr}$	Predicting shifting lengths and directions for $\mathbf{P}_{init}$	≈ 28.8ms	≈ 9.2ms
1) *UNet3D*	Encoding 3D volume features	≈ 23.9ms	≈ 5.6ms
2) MLP_{geo}	Decoding the pixel-aligned features and outputting shifting properties to regress $\mathbf{P}_{sf}$	≈ 4.9ms	≈ 3.6ms
$\mathcal{F}_{sp}$	Predicting Gaussian surfel attributes for $\mathbf{P}_{sf}$ based on pixel-aligned features	≈ 16.9ms	≈ 14.8ms
Splatting	Gaussian surfel splatting to obtain feature, depth and normal maps in target view	≈ 1.9ms	-
$\mathcal{D}_c$	Decoding coarse RGB image	≈ 2.2ms	≈ 0.7ms
$\mathcal{B}$	Appearance enhancement via neural blending	≈ 157.9ms	≈ 17.6ms
Total	-	≈ 331.7ms	≈ 83.9ms

4 EXPERIMENTS

4.1 Implementation Details

We have implemented our human performance capture system under the Pytorch 1.8.0 framework [Paszke et al. 2017] and CUDA 11.1 acceleration. Our model is trained by using two RTX 3090 GPU cards with the Adam [Kingma and Ba 2014] optimizer. The resolution of the input RGBD videos and the rendering results is 1K.

Training and Evaluation Details. We train our model (i.e., PRNet and SPNet) using the public *THuman2.0* [Yu et al. 2021b] dataset, which is split into training and test sets with a ratio of 4:1. We render 60-view RGBD and mask images uniformly for each 3D scan, and randomly select 4-view images spaced approximately 90 degrees as input. We then select 1-view images from the remaining views as the target. For the input raw depth maps, we follow [Dong et al. 2023] to simulate sensor noises on GT depth $\mathbf{D}^*$ to generate $\mathbf{D}$.

We train our PRNet using the sampled 3D points ($2e^4$) around each scan with ground-truth SDF values, to calculate the SDF loss, and together with the initial point set $\mathbf{P}_{init}$ from DPI, to calculate the chamfer-distance loss (see Eq. 3). The input RGB-N images for encoder *PVIT* are of resolution 512^2. The PRNet is trained for ten epochs with a batch size of two, and a learning rate of $5e^{-4}$ (the exponential decay rate is 0.95). After regressing the surface point set $\mathbf{P}_{sf}$, we train our SPNet with the ground truth target-view RGBD-N images (i.e., $\mathbf{C}^*, \mathbf{D}^*$ and $\mathbf{N}^*$). The input RGB-N images are of resolution 1K^2 for the *UNet* encoder. We first warm up the Gaussian implicit function $\mathcal{F}_{sp}$ without appearance blending for ten epochs with a batch size of two and learning rate of $2e^{-4}$. We then fine-tune the SPNet with the blending for five epochs, under a batch size of one and a learning rate of $1e^{-4}$ (0.95 for the exponential decay rate).

Fig. 9. Visual comparisons with SOTA methods, including KeypointNeRF [Mihajlovic et al. 2022], IBRNet [Wang et al. 2021b], NPBG++ [Rakhimov et al. 2022], PIFu(RGBD) [Saito et al. 2019], SHERF [Hu et al. 2023], GPS-Gaussian [Zheng et al. 2024], and SAILOR [Dong et al. 2023], on the *THuman2.0* dataset.

Table 2. Comparisons of rendering results on the *THuman2.0* Dataset [Yu et al. 2021b]. All competing methods are re-trained or fine-tuned for a fair evaluation. We report the average rendering time of 1K frames using a single RTX 3090 GPU. *: we report GPS-Gaussian's inference speed w/o acceleration (no available accelerated model). The best and second best results are marked in **bold** and underline, respectively.

Methods	Avg Time (s)↓	*THuman2.0* Dataset [Yu et al. 2021b]			
		PSNR ↑	**SSIM** ↑	**LPIPS** $\times10^{-1}$ ↓	**MAE**$\times10^{-2}$ ↓
PixelNeRF	≈ 390.0	30.215	0.938	1.179	0.865
IBRNet	≈ 25.7	34.469	0.963	0.742	0.497
MPSNeRF	≈ 32.2	30.317	0.945	0.866	0.754
NHP	≈ 102.5	31.488	0.957	0.851	0.647
KeypointNeRF	≈ 52.3	31.590	0.953	0.746	0.658
SHERF (impl.)	≈ 1.95	32.339	0.957	0.692	0.575
SHERF (expl.)	≈ 4.2	32.085	0.953	0.604	0.596
LGM (N=4)	≈ 0.15	29.844	0.943	0.508	0.844
GPS-Gaussian* (N=6)	≈ 0.17	33.132	0.969	0.468	0.427
NPBG++	≈ 5.5	32.136	0.962	0.558	0.533
PIFu(RGBD)	≈ 8.5	33.296	0.967	**0.270**	0.543
SAILOR	≈ 0.2	34.882	0.969	0.354	0.392
Ours	**≈0.08**	**35.158**	**0.972**	0.365	**0.336**

We evaluate our rendering performance on the test set of *THuman2.0* [Yu et al. 2021b] dataset and the real-captured dataset of SAILOR [Dong et al. 2023]. For the *THuman2.0* test set, we generate noise of five different degrees on the ground truth depth, following SAILOR [2023], to evaluate rendering quality. For the real-captured dataset, we use RGBD images of four fixed perspective views (the view indexes are 4,6,7,0) as inputs. The remaining four views (indexes of 1,2,3,5) are used to evaluate the rendering quality. Fig. 8 shows some of our point clouds and rendering results on this dataset.

Network Details. The depth denoising network $\mathcal{F}_d$ is implemented based on *HRNetV2-W18-Small-v2* [Wang et al. 2021a], following SAILOR [Dong et al. 2023]. In PRNet, we use the *PVTv2-B0* [Wang et al. 2021d] as backbone (trained from scratch), along with four *CBAM* [Woo et al. 2018] blocks and two convolutional layers to extract geometric features (of dimension $128 \times 128 \times 32$). The *UNet* encoder in SPNet uses two convolutional layers with bilinear upsamplings and skip connections, to obtain texture features (of dimension $1K \times 1K \times 16$). The numbers of neurons in MLPs of the *Surface* and *Gaussian Implicit Functions* are set as follows: $MLP_o \in (37, 64, 32)$, $MLP_g \in (32, 16)$, $MLP_p \in (32, 32, 16)$, $MLP_f \in (32, 5)$, $MLP_t \in (51, 64, 16)$, $MLP_G \in (32, 16, 7)$, and $MLP_w \in (52, 64, 32, 1)$. The 3D-CNN in PRNet uses the *3D UNet* structure with $3 \times 3 \times 3$ convolutional kernels. The feature dimensions of the blending network are set to $(36, 32, 16)$, $(35, 32, 16)$, and $(19, 16, 3)$, respectively.

Accelerated Rendering System. We follow SAILOR [Dong et al. 2023] to accelerate our method to achieve a near-real-time rendering speed on a single RTX 3090 GPU card. Specifically, for all the encoders/decoders based on convolutional network, i.e., $\mathcal{F}_d$ (Depth Denoising), E_d (PVIT-based RGB-N encoder), E_c (UNet-based RGB-N encoder), D_c (Coarse RGB decoder), $\mathcal{B}$ (appearance blending network), *3D-CNN* (volume feature decoder), we use TensorRT with half-precision to accelerate and convert them into the executable models. Besides, we adopt the fully-fused [Müller et al. 2021] technique to quantify all the MLPs and the hydra attention block into independent GPU kernels for acceleration. Finally, we implemented DPI, point projection, feature warping, and other operations using CUDA. Tab. 1 reports the time costs of all the operations in our rendering system, and Fig. 15 visualizes our running system. Our method under acceleration takes about 84ms (around 12 fps) to render a novel view image from GPU-loaded 4-view RGBD frames.

4.2 Main Results

4.2.1 Comparisons on the THuman2.0 Dataset. We compare to eleven state-of-the-art methods, on our test dataset (test part of the *THuman2.0* dataset [Yu et al. 2021b]), including six RGB-based methods (i.e., PixelNeRF [Yu et al. 2021a], IBRNet [Wang et al. 2021b], MPSNeRF [Gao et al. 2022], NHP [Kwon et al. 2021], KeypointNeRF [Mihajlovic et al. 2022], and SHERF [Hu et al. 2023]), three RGBD-based methods (i.e., PIFu(RGBD) [Saito et al. 2019], NPBG++ [Rakhimov et al. 2022], and SAILOR [Dong et al. 2023]), and two concurrent 3DGS [Kerbl et al. 2023]-based works (i.e., LGM [Tang et al. 2024] and GPS-Gaussian [Zheng et al. 2024]). For fair comparisons, we either re-train (unavailable pre-trained weights) or fine-tune (available pre-trained weights) all these methods on the training set of *THuman2.0*. As SHERF [Hu et al. 2023] is a monocular method, we use SHERF to render novel-view images in two ways. First, we always select one view (out of four) that is closest to the target view as the input, which implicitly leverages multi-view information (denoted as "SHERF (impl.)"). Second, we modify and re-train SHERF to aggregate 4-view explicitly (denoted as "SHERF (expl.)"). Since GPS-Gaussian [Zheng et al. 2024] does not work under our very sparse (i.e., four views) setting (input images are over-cropped due to stereo rectify), we fine-tune GPS-Gaussian [Zheng et al. 2024] to use six (at least) input views. We measure the rendering accuracy using the PSNR, SSIM, MAE, and LPIPS [Zhang et al. 2018] metrics.

Tab. 2 reports the quantitative results. We can see that our method generally outperforms existing rendering methods on all three objective metrics (i.e., PSNR, SSIM, and MAE), while achieving a slightly lower LPIPS score compared with PIFu (RGBD) [Saito et al. 2019] and SAILOR [Dong et al. 2023]. We summarize the average inference times of all methods in rendering an image of 1K resolution using a single RTX 3090 GPU, which shows that our method runs faster than existing rendering approaches.

In addition, we demonstrate the accurate geometry of our regressed surface points in Tab. 3, by comparing to the triangle vertices of reconstructed mesh (GTPIFu [Dong et al. 2022] and SAILOR [Dong et al. 2023]), and the fused point clouds (GPS-Gaussian [Zheng et al. 2024] and LGM [Tang et al. 2024]), based on the Chamfer/*P2S* distance (the lower the better). Our results perform better than theirs. The visual comparisons in Fig. 10 show that our method produces more robust surface points, mainly benefiting from SDF/Chamfer supervision. Although the vertices provided by SAILOR [Dong et al. 2023] and GTPIFu [Dong et al. 2022] contain more high-frequency details, they may introduce larger geometric deviations (e.g., clothes in boxes), thus affecting subsequent surfel-based rendering.

Fig. 9 shows the qualitative rendering accuracy comparisons. IBRNet [Wang et al. 2021b] relies on the colorimetric inputs from adjacent views. It tends to produce obvious topological and texture errors, for querying views far from the input views. The results of KeypointNeRF [Mihajlovic et al. 2022] suffer from incomplete shapes and incorrect textures in the regions with occlusions or large/complex motions, due to their insufficient topological constraints by using sparse 3D keypoints. By using point clouds to represent 3D humans, NPBG++ [Rakhimov et al. 2022] may produce slightly better results. However, they still suffer from missing parts, inaccurate shapes, and low-quality textures due to the noisy and unstructured point clouds. PIFu (RGBD) [Saito et al. 2019] learns surface fields with pixel-aligned RGBD features for reconstruction. However, inaccurate raw depths can cause missing parts and the lack of high-frequency details, further degrading their texture quality. SHERF (impl.) [Hu et al. 2023] combines SMPL [Loper et al. 2015] with a hierarchical feature bank to represent 3D humans. Their results may contain incorrect/incomplete local shapes (e.g., clothes) and wrong textures, due to the limited SMPL representation. SAILOR [Dong et al. 2023] combines PIFu and NeRF for reconstruction and rendering. While their results are generally better than previous methods, we can still observe over-smoothed surfaces and blurry textures in local regions. On the other hand, despite using six input views, the 3DGS-based GPS-Gaussian [Zheng et al. 2024] may produce obvious incomplete or distorted shapes and sometimes color artifacts in their textures. In contrast, our method can render results with accurate shapes and high-quality texture details.

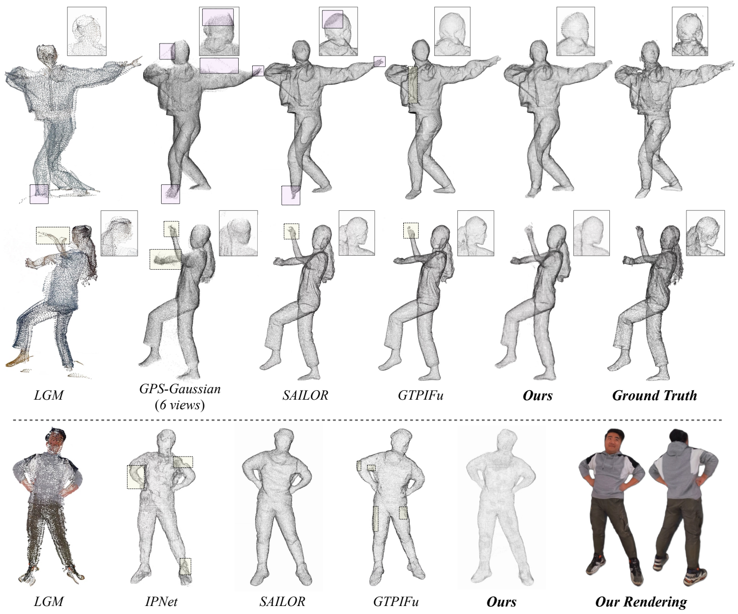

Fig. 10. Visual comparisons between our regressed surface points and those points/vertices produced by existing methods, on the *THuman2.0* (first two rows) and real-captured (bottom row) dataset. Zoom in to view details.

Table 3. Geometric comparisons between our regressed surface points and those produced by GPS-Gaussian [Zheng et al. 2024], LGM [Tang et al. 2024], GTPIFu [Dong et al. 2022] and SAILOR [Dong et al. 2023]. The best and second best results are marked in **bold** and underline, respectively.

Index	*Methods* (*THuman2.0* Dataset [Yu et al. 2021b])				
	GPS-Gaussian (N=6)	LGM	GTPIFu (N=4)	SAILOR	**Ours**
Chamfer $\times 10^{-2}$ ↓	0.982	3.996	<u>0.922</u>	0.975	**0.801**
P2S $\times 10^{-2}$ ↓	0.920	3.942	0.817	<u>0.807</u>	**0.715**

4.2.2 Comparisons on the SAILOR Dataset. Tab. 4 reports the quantitative results of our method and eight best-performing methods based on their efficiencies on the real-captured dataset (containing ten independent performers with different motions) of SAILOR [Dong et al. 2023]. They are RGB-based methods (including IBRNet [Wang et al. 2021b], MPSNeRF [Gao et al. 2022], NHP [Kwon et al. 2021],

Fig. 11. Visual comparisons of our method with SOTA methods, including MPSNeRF [Gao et al. 2022], NHP [Kwon et al. 2021], SHERF [Hu et al. 2023], NPBG++ [Rakhimov et al. 2022], and SAILOR [Dong et al. 2023], on the real-captured SAILOR dataset [Dong et al. 2023].

and SHERF [Hu et al. 2023]), and RGBD-based methods (including NPBG++ [Rakhimov et al. 2022], PIFu (RGBD) [Saito et al. 2019], the re-implemented Function4D [Yu et al. 2021b] (denoted as F4D (re-impl.)), and SAILOR [Dong et al. 2023]). Note that GPS-Gaussian [Zheng et al. 2024] cannot be trained on this dataset, as its stereo rectify during depth estimation requires any two adjacent input views to have no more than 60°, while the training part of SAILOR dataset contains at least two adjacent views with 90°. We implemented F4D (re-impl.) [Yu et al. 2021b] by tracking the former and latter frames of the current frame and fusing the multi-frame point clouds to produce new input depth maps, which are fed into a PIFu model for geometric/texture modeling. Tab. 4 demonstrates that our method generally outperforms these competing methods.

Fig. 11 shows four examples of the visual comparisons. We can see that the SMPL-based methods, MPSNeRF [Gao et al. 2022], NHP [Kwon et al. 2021], and SHERF [Hu et al. 2023] (expl.), tend to produce topological errors and incorrect textures, as SMPL-based models can only represent naked human bodies. Meanwhile, for the RGBD-based methods, NPBG++ [Rakhimov et al. 2022] tends to produce obvious geometry errors, including incomplete and distorted shapes, due to the unavoidable noise in raw point clouds, and the results from SAILOR [Dong et al. 2023] contain blurry textures and sometimes color artifacts. In contrast, our method combines surface implicit function and Gaussian implicit function to enable effective reconstruction and rendering of 3D humans with accurate geometries and fine-grained textures. These experiments overall demonstrate that our method can render high-quality novel-view images for diverse performers.

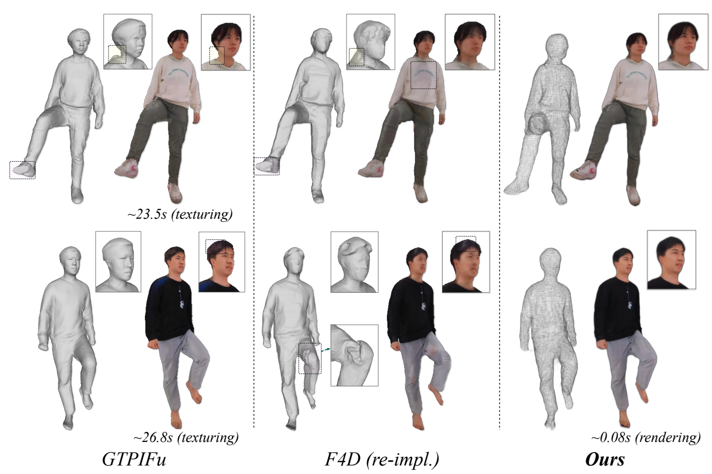

Fig. 12. Comparisons of our 3D reconstruction (surface points) and novel-view rendering results, with GTPIFu [Dong et al. 2022] and the re-implemented F4D [Yu et al. 2021b] on the real-captured dataset.

Table 4. Comparisons of rendering results produced by our method and existing methods on the real-captured dataset of SAILOR [Dong et al. 2023]. All competing methods are re-trained or fine-tuned for fair evaluation. The best and second best results are marked in **bold** and underline, respectively.

Dataset (size) of SAILOR	Index	IBRNet	MPSNeRF	NHP	NPBG++	PIFu(RGBD)	SHERF(impl.)	SHERF(expl.)	F4D(re-impl.)	SAILOR	**Ours**
Rocking & Walking (230)	**PSNR** ↑	28.344	27.930	28.486	27.769	28.448	27.936	28.504	28.446	30.920	**30.952**
	SSIM ↑	0.934	0.928	0.938	0.932	0.948	0.934	0.937	0.950	**0.965**	0.961
	LPIPS ↓	0.121	0.114	0.113	0.0964	0.0688	0.118	0.0997	0.0686	**0.0451**	0.0555
Kung Fu (230)	**PSNR** ↑	24.131	22.760	22.709	24.148	27.147	23.054	23.869	26.849	28.624	**28.711**
	SSIM ↑	0.926	0.913	0.927	0.913	0.953	0.924	0.925	0.951	**0.958**	0.953
	LPIPS ↓	0.110	0.113	0.110	0.0879	0.0422	0.122	0.0976	0.0453	**0.0350**	0.0484
Rocking & Undressing (150)	**PSNR** ↑	29.026	29.265	29.325	28.898	28.133	27.992	28.979	27.996	32.139	**32.312**
	SSIM ↑	0.942	0.942	0.946	0.943	0.955	0.945	0.945	0.955	0.968	**0.969**
	LPIPS ↓	0.126	0.111	0.116	0.0976	0.0808	0.112	0.0997	0.0813	**0.0453**	0.0468
Swinging_1 (110)	**PSNR** ↑	23.706	22.350	20.363	23.879	27.833	22.678	23.036	27.630	28.389	**28.836**
	SSIM ↑	0.925	0.921	0.930	0.900	0.954	0.929	0.928	0.952	0.962	**0.967**
	LPIPS ↓	0.109	0.122	0.118	0.0833	**0.0252**	0.118	0.0968	0.0266	0.0259	0.0298
Swinging_2 (120)	**PSNR** ↑	24.669	24.055	22.986	24.665	27.434	24.137	24.702	27.456	29.065	**29.565**
	SSIM ↑	0.913	0.910	0.919	0.915	0.938	0.914	0.914	0.940	0.948	**0.952**
	LPIPS ↓	0.103	0.108	0.0993	0.0693	**0.0317**	0.109	0.0849	0.0321	0.0344	0.0443
Punching (120)	**PSNR** ↑	26.737	26.054	25.440	27.256	29.338	26.261	27.008	29.410	29.931	**30.394**
	SSIM ↑	0.933	0.926	0.936	0.935	0.947	0.933	0.930	0.948	0.966	**0.971**
	LPIPS ↓	0.098	0.102	0.0961	0.0677	0.0320	0.102	0.0849	0.0317	0.0294	**0.0291**
Swinging & Walking (126)	**PSNR** ↑	23.640	22.266	21.021	24.025	27.302	23.438	23.798	27.361	**30.036**	28.874
	SSIM ↑	0.932	0.928	0.937	0.918	0.954	0.936	0.935	0.953	0.968	**0.971**
	LPIPS ↓	0.104	0.117	0.106	0.0677	0.0280	0.111	0.0900	0.0277	**0.0275**	0.0276
Lifting Legs (120)	**PSNR** ↑	24.387	23.906	22.612	24.960	27.572	23.403	24.653	27.624	**29.060**	28.715
	SSIM ↑	0.917	0.915	0.921	0.915	0.938	0.915	0.917	0.939	0.955	**0.957**
	LPIPS ↓	0.107	0.111	0.108	0.0741	**0.0319**	0.116	0.0895	0.0320	0.0348	0.0351
Stretching_1 (106)	**PSNR** ↑	25.853	25.259	24.173	25.508	29.062	25.569	26.246	29.138	30.224	**30.241**
	SSIM ↑	0.924	0.922	0.928	0.918	0.953	0.926	0.925	0.955	0.956	**0.960**
	LPIPS ↓	0.106	0.105	0.109	0.0760	0.0298	0.101	0.0835	**0.0288**	0.0360	0.0359
Stretching_2 (200)	**PSNR** ↑	26.927	25.939	24.991	26.879	30.050	26.050	26.757	29.708	**30.597**	30.578
	SSIM ↑	0.936	0.938	0.941	0.935	0.953	0.940	0.939	0.951	0.967	**0.971**
	LPIPS ↓	0.102	0.102	0.104	0.0707	**0.0308**	0.101	0.0819	0.0319	0.0353	0.0400
Total (1512)	**PSNR** ↑	25.946	25.172	24.568	25.949	28.254	25.052	25.755	28.157	29.969	**29.988**
	SSIM ↑	0.929	0.925	0.933	0.924	0.950	0.930	0.929	0.950	0.962	**0.963**
	LPIPS ↓	0.110	0.110	0.108	0.0809	0.0428	0.111	0.0909	0.0435	**0.0359**	0.0413

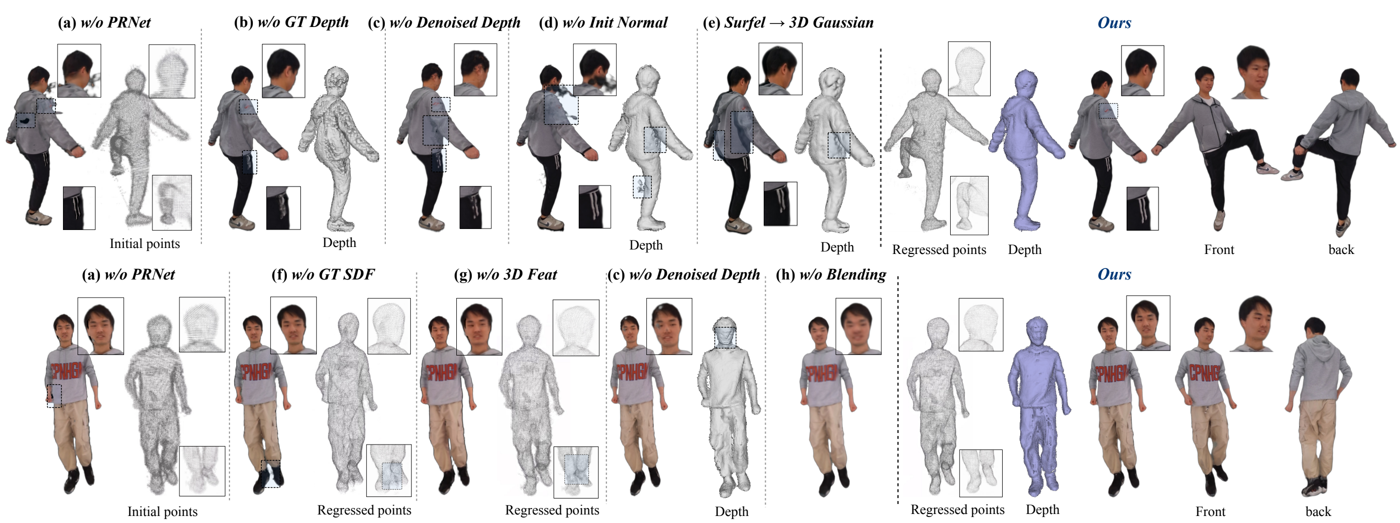

Fig. 13. Visualization of our ablated models on the real-captured SAILOR dataset [Dong et al. 2023]. We show the initial point clouds, the regressed surface point clouds, and the rendered RGB/depth images of novel views, under different ablation settings of Tab. 5.

In addition, we compared our geometric/rendering results with two RGBD-based methods, i.e., GTPIFu [Dong et al. 2022] and F4D (re-impl.) [Yu et al. 2021b], qualitatively. Fig. 12 shows two comparison examples on the real-captured dataset. We can see that GT-PIFu provides detailed 3D meshes and comparable rendering results to ours. However, while its texturing requires approximately 25.2s on average, ours takes about 0.08s, which is significantly faster. Although the F4D (re-impl.) can reconstruct some details (e.g., hair), it may produce notable geometric errors in regions with motions (e.g., leg lifting), due to the unstable sliding fusion.

Table 5. Ablation study on the *THuman2.0* dataset [Yu et al. 2021b] (upper part) and the real-captured dataset [Dong et al. 2023] (lower part). The best and second best results are marked in **bold** and underline, respectively.

Methods	*THuman2.0* Dataset [Yu et al. 2021b]			
	PSNR ↑	**SSIM** ↑	**LPIPS** $\times 10^{-1}$ ↓	**MAE**$\times 10^{-2}$ ↓
w/o PRNet	32.468	0.960	0.537	0.521
w/o GT SDF	34.311	0.969	0.432	0.395
w/o 3D Feat.	33.381	0.957	0.582	0.561
Surfel → 3D Gaussian	33.174	0.963	0.428	0.427
w/o GT Depth	35.142	0.970	0.388	0.337
w/o Init. Normal	34.926	0.971	0.401	0.360
w/o Appearance Blending	33.588	0.962	0.522	0.469
w/o Denoised Depth	34.531	0.967	0.381	0.394
Ours	**35.158**	**0.972**	**0.365**	**0.336**

Methods	*Real-captured Dataset [Dong et al. 2023]*			
	PSNR ↑	**SSIM** ↑	**LPIPS** $\times 10^{-1}$ ↓	**MAE**$\times 10^{-2}$ ↓
w/o PRNet	27.766	0.946	0.544	0.732
w/o GT SDF	28.733	0.951	0.521	0.750
w/o 3D Feat.	28.895	0.951	0.555	0.732
Surfel → 3D Gaussian	27.038	0.948	0.627	0.733
w/o GT Depth	28.287	**0.953**	0.507	0.686
w/o Init. Normal	28.846	0.951	0.491	0.735
w/o Appearance Blending	28.165	0.949	0.486	0.712
w/o Denoised Depth	29.168	0.950	0.481	**0.652**
Ours	**29.334**	0.952	**0.449**	0.680

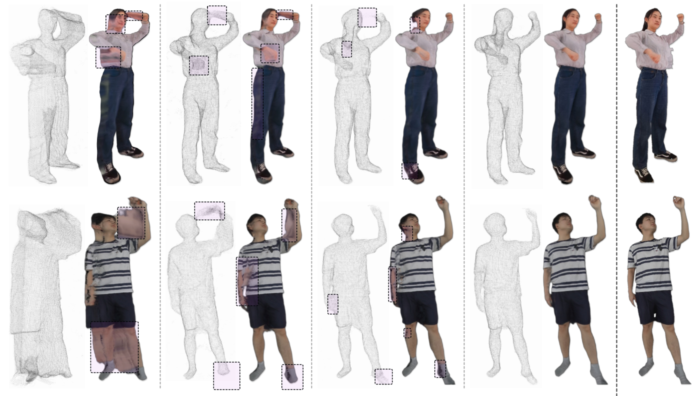

Fig. 14. Visualization of the ablated results for the novel view, with different numbers of input views, on the real-captured dataset [Dong et al. 2023] (upper row) and *THuman2.0* dataset [Yu et al. 2021b] (bottom row).

Table 6. Quantitative results of texture/geometric metrics on the *THuman2.0* dataset [Yu et al. 2021b] of using different numbers of input views.

Models	Photometric				Geometric
	PSNR ↑	**SSIM** ↑	**LPIPS** $\times 10^{-1}$ ↓	**MAE**$\times 10^{-2}$ ↓	**Chamfer**$\times 10^{-2}$ ↓
1-view	31.514	0.962	0.412	0.626	10.482
2-view	31.770	0.962	0.424	0.608	2.880
3-view	33.483	0.966	0.374	0.487	1.108
5-view	**35.144**	**0.973**	**0.363**	**0.343**	1.152
Ours (4-view)	35.081	**0.973**	0.371	0.359	**0.801**

4.3 Ablation Study

We conduct the ablation study on both the *THuman2.0* dataset [Yu et al. 2021b] (upper part) and the real-captured SAILOR [Dong et al. 2023] dataset (lower part), as shown in Tab. 5.

4.3.1 Design Choices of PRNet. We first study the effectiveness of the proposed PRNet. Specifically, we remove the proposed PRNet and train our model directly using the initial points $\mathbf{P}_{init}$ (denoted as "*w/o* PRNet"). We then remove the supervision of ground truth SDF values and use the point cloud chamfer loss to train our PRNet (denoted as "*w/o* GT SDF"). We also remove the *Global Perception Module* and its corresponding 3D volume features from the PRNet (denoted as "*w/o* 3D Feat."). The first three rows in both sub-tables of Tab. 5 show that the performances of all three ablated models degrade on all four metrics, where "*w/o* PRNet" and "*w/o* 3D Feat." generally suffer more significantly. Fig. 13(a) shows that the initial points $\mathbf{P}_{init}$ lack geometry accuracy and contain obvious noise in rendering results, while Fig. 13(f) shows that using ground truth SDF values as supervision helps correct local shape ambiguities (see the foot region), also for rendering. We can also see from Fig. 13(g) that without 3D global features of the point cloud as guidance, the points tend to be noisy, while the rendering results are blurry and contain artifacts. These results verify the necessity of using PRNet and SDF supervision to regress accurate surface points and incorporate global point cloud context features via our *Global Perception Module*.

4.3.2 Design Choices of SPNet. We further verify the design choices of the proposed SPNet. First, we replace our 2D Gaussian surfels with the original 3D Gaussians [Kerbl et al. 2023] (denoted as "Surfel → 3D Gaussian"), to explore the effectiveness of surfel-based rendering. Second, we study the effectiveness of the depth and normal constraints enabled by 2D Gaussian surfels. Specifically, we remove the supervision of ground truth depth (denoted as "*w/o* GT Depth"). Third, we cancel the proposed residual normal (i.e., estimation of $\Delta\mathbf{R}$) and regress the final normal vector of a surfel directly (denoted as "*w/o* Init. Normal"). Fourth, we remove the *Appearance Blending* $\mathcal{B}$ in SPNet, denoted as "*w/o* Appearance Blending".

The next fourth rows in both sub-tables of Tab. 5 show that the performances of using 3D Gaussian ellipsoids instead of 2D Gaussian surfels drop significantly, as the multi-view geometry inconsistency of 3D Guassians can be amplified under very sparse views. As shown in Fig. 13(e), the depth of 3D Gaussian contains obvious local geometry ambiguities, and the corresponding rendered image is prone to blurring or distortion. The results of the fifth and sixth rows in both sub-tables of Tab. 5 show that removing either the depth constraint or the normal cue can degrade the performance, causing noisy depth and color artifacts (e.g., floating), as shown in Fig. 13(b,d). The results of the seventh rows in both sub-tables of Tab. 5 show that without appearance blending, the rendered image tends to be over-smoothed, as shown in Fig. 13(h).

4.3.3 Image-conditioned Depth Denoising $\mathcal{F}_d$. Last, we remove the depth denoising model $\mathcal{F}_d$ and use the raw depths directly to train our method (denoted as "*w/o* Denoised Depth"). The last rows in both sub-tables of Tab. 5 show that the performance tends to decrease, and we note that the results on the *THuman2.0* dataset are affected more significantly than those on the real-captured SAILOR dataset. We speculate that our simulated raw depth $\mathbf{D}$ in *THuman2.0* dataset [Yu et al. 2021b] contains more severe synthetic noise (e.g., holes). Fig. 13(c) shows that noise in the raw depth can cause geometry errors and color artifacts, mainly in the face region of the second row, and detail loss in the white line pattern of the first row.

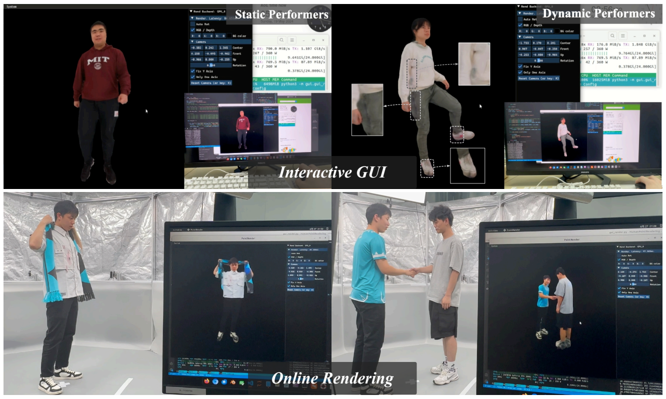

Fig. 15. Free-view rendering in our interactive GUI (upper row), and the online demo (bottom row) for topology-change and multi-person settings. Our results may have ghosting effects as indicated with white boxes.

4.4 More Results

4.4.1 Evaluation of Input View Numbers. We evaluate the texture (in the 45-degree adjacent views of input views) and geometric results, using different numbers (i.e., 1-to-5) of input views on the *THuman2.0* dataset, as shown in Tab. 6. Our 4-view setup achieves the best scores in SSIM and Chamfer metrics, while is slightly lower than the 5-view setup in other metrics. Fig. 14 shows two examples. While 1-view and 2-view setups suffer from severe geometry errors and texture artifacts, the results of 3-view may exhibit incomplete and ambiguous shapes. Hence, we chose the 4-view setup for its lighter cost and support for near-real-time rendering.

4.4.2 Visualization of our Running System. We show our interactive GUI and online rendering demo in Fig. 15. After loading a 4-view RGBD frame (we follow SAILOR [Dong et al. 2023] to preprocess the Kinect-V4 raw captured data into the RGBD inputs of our system), our GUI can render a novel-view image at interactive speed (12 fps) on a single RTX 3090 GPU card. Our online rendering system can handle some topology changes (e.g., wearing scarves) and interactions between two persons.

5 CONCLUSION

In this paper, we have proposed a novel human performance capture method, which learns a novel point-based generalizable human (PGH) representation from very sparse live RGBD videos. The PGH representation contains a surface implicit function for regressing accurate surface points, and a Gaussian implicit function for parameterizing and rendering the radiance fields of the regressed surface points. We have proposed a novel point-regressing network (PRNet) with a depth-guided point cloud initialization (DPI) method, and a novel neural blending-based surfel splatting network (SPNet), for the implementations of the two implicit functions. Our method can produce free-view human performance videos with high-quality geometries and appearances in 1K resolution at 12 fps using a single RTX 3090 GPU. Experiments on two datasets show that our method outperforms existing human performance capture methods.

Our method does have some limitations. First, our results may exhibit temporal flickers or jitters, and sometimes lose high-frequency details (e.g., hair and hands), due to inaccurate matting and the lack of temporal constraints. Second, our rendering results may produce ghosting effects (e.g., dot boxes and their zoomed-in boxes in Fig. 15) mainly due to inaccurate point cloud registration during camera calibration. Last, our results may also exhibit abrupt changes with ghosting effects, caused by the inconsistent changes of view-dependent blending weights during the target view sweeping. Incorporating powerful matting algorithms, temporal constraints, high-frequency view encoding, and a more effective blending network, into a more compact point-based 3D human representation can be interesting for future research.

ACKNOWLEDGMENTS

We thank all the anonymous reviewers for their professional and constructive comments. Weiwei Xu is supported by NSFC (No. 61732016). This work was partially supported by two GRF grants from the RGC of Hong Kong (Ref.: 11205620 and 11211223). This paper is supported by Ant Group and Information Technology Center and State Key Lab of CAD&CG, Zhejiang University.

REFERENCES

Rameen Abdal, Wang Yifan, Zifan Shi, Yinghao Xu, Ryan Po, Zhengfei Kuang, Qifeng Chen, Dit-Yan Yeung, and Gordon Wetzstein. 2023. Gaussian Shell Maps for Efficient 3D Human Generation. *arXiv:2311.17857* (2023).

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, and Judy Hoffman. 2023. Hydra attention: Efficient attention with many heads. In *Eur. Conf. Comput. Vis.*

Yukang Cao, Guanying Chen, Kai Han, Wenqi Yang, and Kwan-Yee K Wong. 2022. JIFF: Jointly-aligned Implicit Face Function for High Quality Single View Clothed Human Reconstruction. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Kennard Chan, Guosheng Lin, Haiyu Zhao, and Weisi Lin. 2022a. S-PIFu: Integrating Parametric Human Models with PIFu for Single-view Clothed Human Reconstruction. In *Adv. Neural Inform. Process. Syst.*

Kennard Yanting Chan, Guosheng Lin, Haiyu Zhao, and Weisi Lin. 2022b. IntegratedPIFu: Integrated Pixel Aligned Implicit Function for Single-view Human Reconstruction. In *Eur. Conf. Comput. Vis.*

Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. 2021a. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Int. Conf. Comput. Vis.*

Jianchuan Chen, Ying Zhang, Di Kang, Xuefei Zhe, Linchao Bao, Xu Jia, and Huchuan Lu. 2021b. Animatable Neural Radiance Fields from Monocular RGB Videos. *arXiv:2106.13629* (2021).

Wei Cheng, Su Xu, Jingtan Piao, Chen Qian, Wayne Wu, Kwan-Yee Lin, and Hongsheng Li. 2022. Generalizable Neural Performer: Learning Robust Radiance Fields for Human Novel View Synthesis. *arXiv:2204.11798* (2022).

Alvaro Collet, Ming Chuang, Pat Sweeney, Don Gillett, Dennis Evseev, David Calabrese, Hugues Hoppe, Adam Kirk, and Steve Sullivan. 2015. High-Quality Streamable Free-Viewpoint Video. *ACM Trans. Graph.* (2015).

Pinxuan Dai, Jiamin Xu, Wenxiang Xie, Xinguo Liu, Huamin Wang, and Weiwei Xu. 2024. High-quality Surface Reconstruction using Gaussian Surfels. In *ACM SIGGRAPH Conference Proceedings.*

Edilson De Aguiar, Carsten Stoll, Christian Theobalt, Naveed Ahmed, Hans-Peter Seidel, and Sebastian Thrun. 2008. Performance capture from sparse multi-view video. In *ACM SIGGRAPH Conference Proceedings.*

Zheng Dong, Ke Xu, Ziheng Duan, Hujun Bao, Weiwei Xu, and Rynson Lau. 2022. Geometry-aware Two-scale PIFu Representation for Human Reconstruction. In *Adv. Neural Inform. Process. Syst.*

Zheng Dong, Ke Xu, Yaoan Gao, Qilin Sun, Hujun Bao, Weiwei Xu, and Rynson WH Lau. 2023. SAILOR: Synergizing Radiance and Occupancy Fields for Live Human Performance Capture. *ACM Trans. Graph.* (2023).

Mingsong Dou, Philip L. Davidson, S. Fanello, S. Khamis, Adarsh Kowdle, Christoph Rhemann, Vladimir Tankovich, and Shahram Izadi. 2017. Motion2fusion: real-time volumetric performance capture. *ACM Trans. Graph.* (2017).

Mingsong Dou, Sameh Khamis, Yury Degtyarev, Philip Davidson, Sean Ryan Fanello, Adarsh Kowdle, Sergio Orts Escolano, Christoph Rhemann, David Kim, Jonathan Taylor, et al. 2016. Fusion4d: Real-time performance capture of challenging scenes. *ACM Trans. Graph.* (2016).

Qiao Feng, Yebin Liu, Yu-Kun Lai, Jingyu Yang, and Kun Li. 2022. FOF: Learning Fourier Occupancy Field for Monocular Real-time Human Reconstruction. In *Adv. Neural*

Inform. Process. Syst.
Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. 2023. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 12479–12488.
Fridovich-Keil and Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. 2022. Plenoxels: Radiance Fields without Neural Networks. In *CVPR.*
Guy Gafni, Justus Thies, Michael Zollhofer, and Matthias Nießner. 2021. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *IEEE Conf. Comput. Vis. Pattern Recog.*
Xiangjun Gao, Jiaolong Yang, Jongyoo Kim, Sida Peng, Zicheng Liu, and Xin Tong. 2022. Mps-nerf: Generalizable 3d human rendering from multiview images. *IEEE Trans. Pattern Anal. Mach. Intell.* (2022).
Kaiwen Guo, Peter Lincoln, Philip Davidson, Jay Busch, Xueming Yu, Matt Whalen, Geoff Harvey, Sergio Orts-Escolano, Rohit Pandey, Jason Dourgarian, et al. 2019. The relightables: Volumetric performance capture of humans with realistic relighting. *ACM Trans. Graph.* (2019).
Marc Habermann, Lingjie Liu, Weipeng Xu, Michael Zollhoefer, Gerard Pons-Moll, and Christian Theobalt. 2021. Real-time deep dynamic characters. *ACM Trans. Graph.* (2021).
Marc Habermann, Weipeng Xu, Michael Zollhoefer, Gerard Pons-Moll, and Christian Theobalt. 2019. Livecap: Real-time human performance capture from monocular video. *ACM Trans. Graph.* (2019).
Marc Habermann, Weipeng Xu, Michael Zollhofer, Gerard Pons-Moll, and Christian Theobalt. 2020. Deepcap: Monocular human performance capture using weak supervision. In *IEEE Conf. Comput. Vis. Pattern Recog.*
Tong He, John Collomosse, Hailin Jin, and Stefano Soatto. 2020. Geo-PIFu: Geometry and Pixel Aligned Implicit Functions for Single-view Human Reconstruction. In *NeurIPS.*
Tong He, Yuanlu Xu, Shunsuke Saito, Stefano Soatto, and Tony Tung. 2021. ARCH++: Animation-ready clothed human reconstruction revisited. In *ICCV.*
Yang Hong, Juyong Zhang, Boyi Jiang, Yudong Guo, Ligang Liu, and Hujun Bao. 2021. StereoPIFu: Depth Aware Clothed Human Digitization via Stereo Vision. In *IEEE Conf. Comput. Vis. Pattern Recog.*
Shoukang Hu, Fangzhou Hong, Liang Pan, Haiyi Mei, Lei Yang, and Ziwei Liu. 2023. Sherf: Generalizable human nerf from a single image. In *Int. Conf. Comput. Vis.*
Shoukang Hu and Ziwei Liu. 2024. GauHuman: Articulated Gaussian Splatting for Real-Time 3D Human Rendering.
Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2024. 2D Gaussian Splatting for Geometrically Accurate Radiance Fields. In *ACM SIGGRAPH Conference Proceedings.*
Zeng Huang, Yuanlu Xu, Christoph Lassner, Hao Li, and Tony Tung. 2020. Arch: Animatable reconstruction of clothed humans. In *CVPR.*
Mustafa Işık, Martin Rünz, Markos Georgopoulos, Taras Khakhulin, Jonathan Starck, Lourdes Agapito, and Matthias Nießner. 2023. HumanRF: High-Fidelity Neural Radiance Fields for Humans in Motion. *ACM Trans. Graph.* (2023).
Zhang Jiakai, Liu Xinhang, Ye Xinyi, Zhao Fuqiang, Zhang Yanshun, Wu Minye, Zhang Yingliang, Xu Lan, and Yu Jingyi. 2021. Editable Free-Viewpoint Video using a Layered Neural Representation. In *Proc. of SIGGRAPH.*
Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan. 2022. NeuMan: Neural Human Radiance Field from a Single Video. In *Eur. Conf. Comput. Vis.*
Hanbyul Joo, Tomas Simon, and Yaser Sheikh. 2018. Total capture: A 3D deformation model for tracking faces, hands, and bodies. In *IEEE Conf. Comput. Vis. Pattern Recog.*
Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Trans. Graph.* (2023).
Shakiba Kheradmand, Daniel Rebain, Gopal Sharma, Weiwei Sun, Jeff Tseng, Hossam Isack, Abhishek Kar, Andrea Tagliasacchi, and Kwang Moo Yi. 2024. 3D Gaussian Splatting as Markov Chain Monte Carlo. *arXiv:2404.09591* (2024).
Jaehyeok Kim, Dongyoon Wee, and Dan Xu. 2023. You Only Train Once: Multi-Identity Free-Viewpoint Neural Human Rendering from Monocular Videos. *arXiv:2303.05835* (2023).
Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv:1412.6980* (2014).
Kutulakos Kiriakos and Seitz Steven. 2000. A Theory of Shape by Space Carving. *Int. J. Comput. Vis.* (2000).
Muhammed Kocabas, Nikos Athanasiou, and Michael J Black. 2020. Vibe: Video inference for human body pose and shape estimation. In *IEEE Conf. Comput. Vis. Pattern Recog.*
Muhammed Kocabas, Rick Chang, James Gabriel, Oncel Tuzel, and Anurag Ranjan. 2024. HUGS: Human Gaussian Splats. In *IEEE Conf. Comput. Vis. Pattern Recog.*
Youngjoong Kwon, Dahun Kim, Duygu Ceylan, and Henry Fuchs. 2021. Neural human performer: Learning generalizable radiance fields for human performance rendering. In *Adv. Neural Inform. Process. Syst.*
Youngjoong Kwon, Dahun Kim, Duygu Ceylan, and Henry Fuchs. 2023. Neural image-based avatars: Generalizable radiance fields for human avatar modeling. In *ICLR.*
Aldo Laurentini. 1994. The visual hull concept for silhouette-based image understanding. *IEEE Transactions on pattern analysis and machine intelligence* 16, 2 (1994), 150–162.
Jiahui Lei, Yufu Wang, Georgios Pavlakos, Lingjie Liu, and Kostas Daniilidis. 2024. Gart: Gaussian articulated template models. In *IEEE Conf. Comput. Vis. Pattern Recog.*
Ruilong Li, Yuliang Xiu, Shunsuke Saito, Zeng Huang, Kyle Olszewski, and Hao Li. 2020a. Monocular real-time volumetric performance capture. In *Eur. Conf. Comput. Vis.*
Yue Li, Marc Habermann, Bernhard Thomaszewski, Stelian Coros, Thabo Beeler, and Christian Theobalt. 2021. Deep Physics-aware Inference of Cloth Deformation for Monocular Human Performance Capture. In *3DV.*
Zhe Li, Tao Yu, Chuanyu Pan, Zerong Zheng, and Yebin Liu. 2020b. Robust 3d self-portraits in seconds. In *IEEE Conf. Comput. Vis. Pattern Recog.*
Zhe Li, Zerong Zheng, Lizhen Wang, and Yebin Liu. 2024. Animatable Gaussians: Learning Pose-dependent Gaussian Maps for High-fidelity Human Avatar Modeling. In *IEEE Conf. Comput. Vis. Pattern Recog.*
Haotong Lin, Sida Peng, Zhen Xu, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. 2022. Efficient Neural Radiance Fields with Learned Depth-Guided Sampling. In *ACM SIGGRAPH Asia.*
Lingjie Liu, Marc Habermann, Viktor Rudnev, Kripasindhu Sarkar, Jiatao Gu, and Christian Theobalt. 2021. Neural actor: Neural free-view synthesis of human actors with pose control. *ACM Trans. Graph.* (2021).
Xian Liu, Xiaohang Zhan, Jiaxiang Tang, Ying Shan, Gang Zeng, Dahua Lin, Xihui Liu, and Ziwei Liu. 2024. HumanGaussian: Text-Driven 3D Human Generation with Gaussian Splatting.
Yebin Liu, Qionghai Dai, and Wenli Xu. 2009. A point-cloud-based multiview stereo algorithm for free-viewpoint video. *IEEE Trans. Vis. Comput. Graph.* (2009).
Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael Black. 2015. SMPL: A skinned multi-person linear model. *ACM Trans. Graph.* (2015).
Marko Mihajlovic, Aayush Bansal, Michael Zollhoefer, Siyu Tang, and Shunsuke Saito. 2022. KeypointNeRF: Generalizing Image-based Volumetric Avatars using Relative Spatial Encoding of Keypoints. In *Eur. Conf. Comput. Vis.*
Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *Eur. Conf. Comput. Vis.*
Arthur Moreau, Jifei Song, Helisa Dhamo, Richard Shaw, Yiren Zhou, and Eduardo Pérez-Pellitero. 2024. Human gaussian splatting: Real-time rendering of animatable avatars. *IEEE Conf. Comput. Vis. Pattern Recog.*
Thomas Müller, Fabrice Rousselle, Jan Novák, and Alexander Keller. 2021. Real-time Neural Radiance Caching for Path Tracing. *ACM Trans. Graph.* (2021).
Richard A Newcombe, Dieter Fox, and Steven M Seitz. 2015a. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *IEEE Conf. Comput. Vis. Pattern Recog.*
Richard A Newcombe, Dieter Fox, and Steven M Seitz. 2015b. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *IEEE Conf. Comput. Vis. Pattern Recog.*
Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. 2011. Kinectfusion: Real-time dense surface mapping and tracking. In *IEEE ISMAR.*
Anqi Pang, Xin Chen, Haimin Luo, Minye Wu, Jingyi Yu, and Lan Xu. 2021. Few-shot neural human performance rendering from sparse RGBD videos. In *IJCAI.*
Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. 2021a. Nerfies: Deformable Neural Radiance Fields. In *Int. Conf. Comput. Vis.*
Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. 2021b. HyperNeRF: A Higher-Dimensional Representation for Topologically Varying Neural Radiance Fields. *ACM Trans. Graph.* (2021).
Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in Pytorch. In *Adv. Neural Inform. Process. Syst.*
Bo Peng, Jun Hu, Jingtao Zhou, Xuan Gao, and Juyong Zhang. 2023. IntrinsicNGP: Intrinsic Coordinate based Hash Encoding for Human NeRF. *arXiv:2302.14683* (2023).
Sida Peng, Junting Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Xiaowei Zhou, and Hujun Bao. 2021a. Animatable neural radiance fields for modeling dynamic human bodies. In *Int. Conf. Comput. Vis.*
Sida Peng, Yuanqing Zhang, Yinghao Xu, Qianqian Wang, Qing Shuai, Hujun Bao, and Xiaowei Zhou. 2021b. Neural body: Implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In *IEEE Conf. Comput. Vis. Pattern Recog.*
Marco Pesavento, Yuanlu Xu, Nikolaos Sarafianos, Robert Maier, Ziyan Wang, Chun-Han Yao, Marco Volino, Edmond Boyer, Adrian Hilton, and Tony Tung. 2024. ANIM: Accurate Neural Implicit Model for Human Reconstruction from a single RGB-D

image. In *CVPR*.

Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. 2021. D-nerf: Neural radiance fields for dynamic scenes. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Shenhan Qian, Tobias Kirschstein, Liam Schoneveld, Davide Davoli, Simon Giebenhain, and Matthias Nießner. 2024. Gaussianavatars: Photorealistic head avatars with rigged 3d Gaussians. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Ruslan Rakhimov, Andrei-Timotei Ardelean, Victor Lempitsky, and Evgeny Burnaev. 2022. NPBG++: Accelerating Neural Point-Based Graphics. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Edoardo Remelli, Timur Bagautdinov, Shunsuke Saito, Chenglei Wu, Tomas Simon, Shih-En Wei, Kaiwen Guo, Zhe Cao, Fabian Prada, Jason Saragih, et al. 2022. Drivable volumetric avatars using texel-aligned features. In *ACM SIGGRAPH Conference Proceedings.*

Alfredo Rivero, ShahRukh Athar, Zhixin Shu, and Dimitris Samaras. 2024. Rig3DGS: Creating Controllable Portraits from Casual Monocular Videos. *arXiv:2402.03723* (2024).

Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. 2019. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Int. Conf. Comput. Vis.*

Shunsuke Saito, Tomas Simon, Jason Saragih, and Hanbyul Joo. 2020. PIFuHD: Multi-Level Pixel-Aligned Implicit Function for High-Resolution 3D Human Digitization. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Ruizhi Shao, Liliang Chen, Zerong Zheng, Hongwen Zhang, Yuxiang Zhang, Han Huang, Yandong Guo, and Yebin Liu. 2022a. FloRen: Real-Time High-Quality Human Performance Rendering via Appearance Flow Using Sparse RGB Cameras. In *ACM SIGGRAPH Asia Conference Proceedings.*

Ruizhi Shao, Hongwen Zhang, He Zhang, Mingjia Chen, Yanpei Cao, Tao Yu, and Yebin Liu. 2022b. DoubleField: Bridging the Neural Surface and Radiance Fields for High-fidelity Human Reconstruction and Rendering. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Zhijing Shao, Zhaolong Wang, Zhuang Li, Duotun Wang, Xiangru Lin, Yu Zhang, Mingming Fan, and Zeyu Wang. 2024. SplattingAvatar: Realistic Real-Time Human Avatars with Mesh-Embedded Gaussian Splatting. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Shih-Yang Su, Timur Bagautdinov, and Helge Rhodin. 2022. DANBO: Disentangled Articulated Neural Body Representations via Graph Neural Networks. In *Eur. Conf. Comput. Vis.*

Shih-Yang Su, Frank Yu, Michael Zollhöfer, and Helge Rhodin. 2021. A-nerf: Articulated neural radiance fields for learning human shape, appearance, and pose. *Adv. Neural Inform. Process. Syst.* (2021).

Zhuo Su, Lan Xu, Zerong Zheng, Tao Yu, Yebin Liu, and Lu Fang. 2020. RobustFusion: Human Volumetric Capture with Data-Driven Visual Cues Using a RGBD Camera. In *Eur. Conf. Comput. Vis.*

Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. 2024. LGM: Large Multi-View Gaussian Model for High-Resolution 3D Content Creation. *arXiv:2402.05054* (2024).

Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. 2021. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *Int. Conf. Comput. Vis.*

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Adv. Neural Inform. Process. Syst.*

Daniel Vlasic, Ilya Baran, Wojciech Matusik, and Jovan Popović. 2008. Articulated Mesh Animation from Multi-View Silhouettes. *ACM Trans. Graph.* (2008).

Daniel Vlasic, Pieter Peers, Ilya Baran, Paul Debevec, Jovan Popović, Szymon Rusinkiewicz, and Wojciech Matusik. 2009. Dynamic shape capture using multi-view photometric stereo. In *ACM SIGGRAPH Asia.*

Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. 2021a. Deep High-Resolution Representation Learning for Visual Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* (2021).

Liao Wang, Ziyu Wang, Pei Lin, Yuheng Jiang, Xin Suo, Minye Wu, Lan Xu, and Jingyi Yu. 2021c. IButter: Neural Interactive Bullet Time Generator for Human Free-Viewpoint Rendering. In *ACM Int. Conf. Multimedia.*

Liao Wang, Jiakai Zhang, Xinhang Liu, Fuqiang Zhao, Yanshun Zhang, Yingliang Zhang, Minye Wu, Jingyi Yu, and Lan Xu. 2022. Fourier PlenOctrees for Dynamic Radiance Field Rendering in Real-Time. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. 2021b. Ibrnet: Learning multi-view image-based rendering. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Shengze Wang, Ziheng Wang, Ryan Schmelzle, Liujie Zheng, YoungJoong Kwon, Roni Sengupta, and Henry Fuchs. 2024. Learning View Synthesis for Desktop Telepresence with Few RGBD Cameras. *IEEE TVCG* (2024).

Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. 2021d. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF international conference on computer vision.* 568–578.

Chung-Yi Weng, Brian Curless, Pratul P. Srinivasan, Jonathan T. Barron, and Ira Kemelmacher-Shlizerman. 2022. HumanNeRF: Free-Viewpoint Rendering of Moving People From Monocular Video. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. 2018. Cbam: Convolutional block attention module. In *Eur. Conf. Comput. Vis.*

Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Wang Xinggang. 2024. 4D Gaussian Splatting for Real-Time Dynamic Scene Rendering. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Minye Wu, Yuehao Wang, Qiang Hu, and Jingyi Yu. 2020. Multi-view neural human rendering. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. 2021. Space-time neural irradiance fields for free-viewpoint video. In *IEEE Conf. Comput. Vis. Pattern Recog.*

D. Xiang, F. Prada, C. Wu, and J. Hodgins. 2020. MonoClothCap: Towards Temporally Coherent Clothing Capture from Monocular RGB Video. In *3DV.*

Yuliang Xiu, Jinlong Yang, Dimitrios Tzionas, and Michael J. Black. 2022. ICON: Implicit Clothed humans Obtained from Normals. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Weipeng Xu, Avishek Chatterjee, Michael Zollhöfer, Helge Rhodin, Dushyant Mehta, Hans-Peter Seidel, and Christian Theobalt. 2018. MonoPerfCap: Human Performance Capture From Monocular Video. *ACM Trans. Graph.* (2018).

Yuelang Xu, Benwang Chen, Zhe Li, Hongwen Zhang, Lizhen Wang, Zerong Zheng, and Yebin Liu. 2024a. Gaussian Head Avatar: Ultra High-fidelity Head Avatar via Dynamic Gaussians. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Zhen Xu, Sida Peng, Haotong Lin, Guangzhao He, Jiaming Sun, Yujun Shen, Hujun Bao, and Xiaowei Zhou. 2024b. 4K4D: Real-Time 4D View Synthesis at 4K Resolution. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Zhiwen Yan, Weng Fei Low, Yu Chen, and Gim Hee Lee. 2023. Multi-Scale 3D Gaussian Splatting for Anti-Aliased Rendering. *arXiv:2311.17089* (2023).

Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. 2023. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. *arXiv:2309.13101* (2023).

Xu Yinghao, Shi Zifan, Yifan Wang, Chen Hansheng, Yang Ceyuan, Peng Sida, Shen Yujun, and Wetzstein Gordon. 2024. GRM: Large Gaussian Reconstruction Model for Efficient 3D Reconstruction and Generation. *arXiv:2403.14621* (2024).

Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. 2021a. pixelnerf: Neural radiance fields from one or few images. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Tao Yu, Zerong Zheng, Kaiwen Guo, Pengpeng Liu, Qionghai Dai, and Yebin Liu. 2021b. Function4d: Real-time human volumetric capture from very sparse consumer rgbd sensors. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Tao Yu, Zerong Zheng, Kaiwen Guo, Jianhui Zhao, Qionghai Dai, Hao Li, Gerard Pons-Moll, and Yebin Liu. 2018. Doublefusion: Real-time capture of human performances with inner body shapes from a single depth sensor. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Zehao Yu, Anpei Chen, Binbin Huang, Torsten Sattler, and Andreas Geiger. 2024a. Mip-Splatting: Alias-free 3D Gaussian Splatting. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Zehao Yu, Torsten Sattler, and Andreas Geiger. 2024b. Gaussian Opacity Fields: Efficient High-quality Compact Surface Reconstruction in Unbounded Scenes. *arXiv:2404.10772* (2024).

Jiakai Zhang, Liao Wang, Xinhang Liu, Fuqiang Zhao, Minzhang Li, Haizhao Dai, Boyuan Zhang, Wei Yang, Lan Xu, and Jingyi Yu. 2022. NeuVV: Neural Volumetric Videos with Immersive Rendering and Editing. *arXiv:2202.06088* (2022).

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wan. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Fuqiang Zhao, Yuheng Jiang, Kaixin Yao, Jiakai Zhang, Liao Wang, Haizhao Dai, Yuhui Zhong, Yingliang Zhang, Minye Wu, Lan Xu, and Jingyi Yu. 2022a. Human Performance Modeling and Rendering via Neural Animated Mesh. *ACM Trans. Graph.* (2022).

Fuqiang Zhao, Wei Yang, Jiakai Zhang, Pei Lin, Yingliang Zhang, Jingyi Yu, and Lan Xu. 2022b. HumanNeRF: Efficiently Generated Human Radiance Field From Sparse Inputs. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Hao Zhao, Jinsong Zhang, Yu-Kun Lai, Zerong Zheng, Yingdi Xie, Yebin Liu, and Kun Li. 2022c. High-Fidelity Human Avatars from a Single RGB Camera. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Shunyuan Zheng, Boyao Zhou, Ruizhi Shao, Boning Liu, Shengping Zhang, Liqiang Nie, and Yebin Liu. 2024. GPS-Gaussian: Generalizable Pixel-wise 3D Gaussian Splatting for Real-time Human Novel View Synthesis. In *IEEE Conf. Comput. Vis. Pattern Recog.*

Zerong Zheng, Tao Yu, Yebin Liu, and Dai Qionghai. 2021. PaMIR: Parametric Model-Conditioned Implicit Representation for Image-based Human Reconstruction. *IEEE Trans. Pattern Anal. Mach. Intell.* (2021).

Zhenglin Zhou, Fan Ma, Hehe Fan, and Yi Yang. 2024. HeadStudio: Text to Animatable Head Avatars with 3D Gaussian Splatting. *arXiv:2402.06149* (2024).

Wojciech Zielonka, Timur Bagautdinov, Shunsuke Saito, Michael Zollhöfer, Justus Thies, and Javier Romero. 2023. Drivable 3D Gaussian Avatars. *arXiv:2311.08581* (2023).