

Supplementary Materials: Mesh-Centric Gaussian Splatting for Human Avatar Modelling with Real-time Dynamic Mesh Reconstruction

Anonymous Authors

1 TRAINING DETAILS

We introduce training details in this section, including Gaussians initialization, density control, and optimization losses.

For initialization, we uniformly set Gaussians on UV grids of SMPL texture map with a resolution of 256. For better geometry detail, we subdivide SMPL mesh once, resulting in 55104 faces, compared to default 13776 faces of SMPL. Throughout optimization, density control is executed to duplicate Gaussians with higher gradients, and remove those with small opacity α . Given our explicit constraint that each Gaussian should not extend beyond its face, enforced through a sigmoid function in equation (10) of the main paper, we verify if a face not have any Gaussians and assign one accordingly.

The full loss is formulated as follows:

$$L = L_{l1} + \lambda_{perc} L_{perc} + \lambda_{mask} L_{mask} + \lambda_{skin} L_{skin} + \lambda_{mesh} L_{mesh}. \quad (1)$$

Each loss term is defined below:

RGB Loss: We use $l1$ loss to compute pixel-wise error and a perceptual loss to provide robustness to local mis-alignments. Following HumanNeRF [5] and 3DGS-Avatar [2], we optimize LPIPS as perceptual loss with VGG as backbone. We render the whole image via rasterization and thus do not require patch sampling in HumanNeRF.

Mask Loss: Following 3DGS-Avatar [2], we use an explicit mask loss. For each pixel p , we compute the opacity value O_p by summing up the sample weights in alpha blending equation (2) in the main paper, namely:

$$O_p = \sum_i \alpha'_i \prod_{j=1}^{i-1} (1 - \alpha'_j). \quad (2)$$

We then supervise it with ground truth foreground mask via an $l1$ loss.

Skinning Loss: We leverage SMPL prior by sampling 1024 points \mathbf{X}_{skin} on surface of the canonical SMPL mesh and regularizing the forward skinning network with corresponding skinning weights \mathbf{w} interpolated with barycentric coordinates.

$$L_{skin} = \frac{1}{|\mathbf{X}_{skin}|} \sum_{\mathbf{x}_{skin} \in \mathbf{X}_{skin}} \|f_{skinning}(\mathbf{x}_{skin}, \theta) - \mathbf{w}\|^2. \quad (3)$$

Mesh Smoothness Loss: We incorporate regularization losses on mesh smoothness from Pytorch3d [3] concerning edge length, normal consistency, and Laplacian smoothing.

$$L_{mesh} = L_{edge} + 0.01 L_{normal} + L_{laplacian}. \quad (4)$$

We set $\lambda_{perc} = 0.01$, $\lambda_{mask} = 0.1$, $\lambda_{mesh} = 1$ in all experiments. For λ_{skin} , we set it to 10 for the first 1k iterations for fast convergence to a reasonable skinning field, then decreased to 0.1 for soft regularization.

2 MESH RECONSTRUCTION OF 3DGS-AVATAR

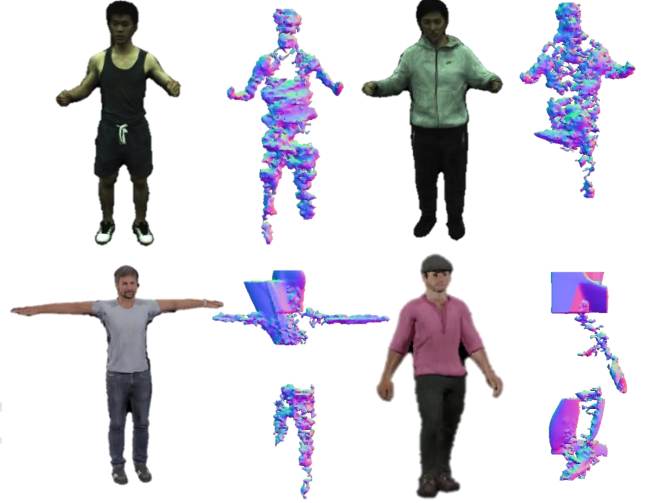


Figure 1: Mesh Reconstruction results of 3DGS-Avatar. The top row is from ZJU-MoCap dataset. The bottom row is from SyntheticHuman dataset.

It is challenging to extract mesh from 3D Gaussian Splatting due to unordered nature and poor alignment with actual scene surfaces of the optimized Gaussians. While we employed the mesh extraction method proposed in DreamGaussian [4] to extract meshes from 3DGS-Avatar [2], we found that it failed to generate reasonable meshes, as illustrated in Figure 1. Several concurrent methods, such as SuGaR [1] and GAvatar [6], focus on mesh extraction from Gaussians by jointly training Gaussian Splatting and implicit SDF, and then using marching cubes or Poisson reconstruction to extract meshes from SDF or point clouds. However, the training of implicit SDF and the reliance on marching cubes or Poisson reconstruction significantly impede their mesh reconstruction efficiency, resulting in a frame rate lower than 1 FPS, which is notably slower than the real-time efficiency of our method (32 FPS).

REFERENCES

- [1] Antoine Guédon and Vincent Lepetit. 2023. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. *arXiv preprint arXiv:2311.12775* (2023).
- [2] Zhiyin Qian, Shaoqi Wang, Marko Mihajlovic, Andreas Geiger, and Siyu Tang. 2023. 3dgs-avatar: Animatable avatars via deformable 3d gaussian splatting. *arXiv preprint arXiv:2312.09228* (2023).
- [3] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. 2020. Accelerating 3D Deep Learning with PyTorch3D. *arXiv:2007.08501* (2020).

- [4] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. 2023. Dream-gaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv preprint arXiv:2309.16653* (2023).
- [5] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 16210–16220.
- [6] Ye Yuan, Xueting Li, Yangyi Huang, Shalini De Mello, Koki Nagano, Jan Kautz, and Umar Iqbal. 2023. Gavatar: Animatable 3d gaussian avatars with implicit mesh learning. *arXiv preprint arXiv:2312.11461* (2023).