Supplementary: AHOY! Animatable Humans under Occlusion from YouTube Videos with Gaussian Splatting and Video Diffusion Priors

Aymen Mir[1], Riza Alp Guler[3], Xiangjun Tang[4], Peter Wonka[4†], and Gerard Pons-Moll[1,2†]

[1] Tübingen AI Center, University of Tübingen, Germany
[2] Max Planck Institute for Informatics, Saarland Informatics Campus, Germany
[3] Imperial College London, UK
[4] King Abdullah University of Science and Technology (KAUST), Saudi Arabia

This supplementary material provides additional details and results. It is organized as follows:

1 Ethical and Societal Impact Considerations

In Sec. 3 of the main paper, we present a method for reconstructing animatable 3D avatars from in-the-wild monocular video. Here we discuss the ethical considerations this capability raises. The ability to create photorealistic digital replicas of individuals from publicly available footage, without their explicit cooperation, poses risks related to privacy, consent, and potential misuse such as unauthorized deepfakes or identity impersonation.

We emphasize that our method is intended for legitimate research and creative applications, including film production, gaming, and virtual reality. All YouTube videos used in our experiments are publicly available, and we do not reconstruct avatars for the purpose of deception or unauthorized use. We encourage the community to develop and adopt safeguards (such as watermarking, provenance tracking, and consent frameworks) to mitigate potential misuse of avatar reconstruction technologies.

We additionally note that the quality of our reconstructions depends on the underlying video diffusion model, which may inherit biases present in its

[†] Equal advising.

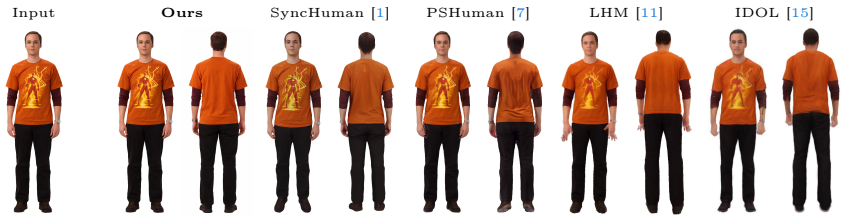

Fig. 1: Additional static reconstruction comparison. All methods receive an unoccluded canonical-pose image. Our animatable avatar is posed to match the target view.

training data. This could lead to uneven reconstruction quality across different demographics, body types, or clothing styles. Future work should audit and address such biases to ensure equitable performance.

2 Additional Static Reconstruction Comparisons

We provide additional qualitative comparisons for static reconstruction beyond those shown in Sec. 4.3 of the main paper. Fig. 1 shows results on an additional subject where all methods receive an unoccluded canonical-pose image as input.

3 Background: 3D Gaussian Splatting

In Sec. 3.1 of the main paper, we build a 3DGS avatar from canonical Gaussian maps. Here we provide additional background on 3D Gaussian Splatting and define the notation used throughout.

3D Gaussian Splatting (3DGS) [5] represents a scene as a set of anisotropic 3D Gaussians. Each Gaussian k is parameterized by a mean (position) $\boldsymbol{\mu}_k \in \mathbb{R}^3$, a covariance matrix $\boldsymbol{\Sigma}_k \in \mathbb{R}^{3\times3}$, an opacity $\alpha_k \in [0,1]$, and color coefficients $\mathbf{c}_k$ (spherical harmonics or RGB). To ensure positive semi-definiteness, the covariance is decomposed as:

$$\boldsymbol{\Sigma}_k = \mathbf{R}_k \,\mathrm{diag}(\mathbf{s}_k)^2\, \mathbf{R}_k^\top, \tag{1}$$

where $\mathbf{R}_k \in \mathbb{R}^{3\times3}$ is a rotation matrix (stored as a unit quaternion $\mathbf{q}_k$) and $\mathbf{s}_k \in \mathbb{R}^3$ is a scale vector. A complete Gaussian is thus defined by the tuple $(\boldsymbol{\mu}_k, \mathbf{s}_k, \mathbf{q}_k, \alpha_k, \mathbf{c}_k)$.

Given a camera $\boldsymbol{\gamma}$, the Gaussians are projected onto the image plane and composited front-to-back via alpha blending. We denote the differentiable rasterization operator as $\mathcal{R}(\mathcal{G};\boldsymbol{\gamma})$, which produces the rendered image for a Gaussian set $\mathcal{G}$.

In our setting, we densely sample N_H points on the SMPL mesh in canonical pose $\boldsymbol{\theta}_*$ to define a canonical Gaussian template:

$$\mathcal{G}_*^H = \{(\boldsymbol{\mu}_{*,k},\, \mathbf{s}_{*,k},\, \mathbf{q}_{*,k},\, \alpha_{*,k},\, \mathbf{c}_{*,k})\}_{k=1}^{N_H}. \tag{2}$$

A StyleUNet [13] S predicts per-pixel offsets from this template, yielding $\mathcal{G}^H = S(\boldsymbol{\theta}_*)$, along with per-Gaussian skinning weights $\mathbf{w}^H_{k,j}$ for LBS deformation (see Sec. 5).

4 Texture Mapping to Canonical Pose

In Sec. 3.1.1 of the main paper, we refer to the supplementary for full details on texture mapping and inpainting. We provide them here.

For each frame t, DensePose [3] provides a mapping $\mathsf{D}_t : \mathbb{R}^2 \to \mathbb{R}^2$ from image pixels to UV coordinates on the SMPL body surface. We define a canonical texture map $\mathbf{T} \in \mathbb{R}^{H_{\text{uv}} \times W_{\text{uv}} \times 3}$ and an occupancy mask $\mathbf{O} \in \{0,1\}^{H_{\text{uv}} \times W_{\text{uv}}}$ that tracks which texels have been filled. For each visible pixel $\mathbf{p}$ (where $\mathbf{M}_t(\mathbf{p}) = 1$), we compute its UV coordinate $\mathbf{u} = \mathsf{D}_t(\mathbf{p})$ and apply a first-write-wins rule:

$$\mathbf{T}(\mathbf{u}) \leftarrow \begin{cases} \mathbf{I}_t(\mathbf{p}) & \text{if } \mathbf{O}(\mathbf{u}) = 0, \\ \mathbf{T}(\mathbf{u}) & \text{otherwise,} \end{cases} \tag{3}$$

setting $\mathbf{O}(\mathbf{u}) \leftarrow 1$ after each write. By iterating over all T frames, we accumulate a partial canonical texture that covers as much of the body surface as the video observations allow.

We define a fixed front-facing canonical pose $\boldsymbol{\theta}_*$ and compute its DensePose map D_*, which assigns a UV coordinate to every pixel of a person standing in $\boldsymbol{\theta}_*$. The canonical-pose image is obtained by an inverse warp [10]: for each canonical pixel $\mathbf{p}_*$ with UV coordinate $\mathbf{u}_* = \mathsf{D}_*(\mathbf{p}_*)$, we look up the color from the accumulated texture atlas, $\mathbf{I}^{\text{part}}_*(\mathbf{p}_*) = \mathbf{T}(\mathbf{u}_*)$. Pixels whose UV texels were never filled ($\mathbf{O}(\mathbf{u}_*) = 0$) are marked as missing in a binary mask $\mathbf{M}^{\text{miss}}_*$. A pretrained FLUX [2] inpainting model fills in the missing regions:

$$\mathbf{I}_* = \mathsf{Inpaint}(\mathbf{I}^{\text{part}}_*,\ \mathbf{M}^{\text{miss}}_*), \tag{4}$$

yielding a complete canonical-pose image $\mathbf{I}_*$.

5 LBS Deformation

In Sec. 3.1 of the main paper, we refer to the supplementary for details on LBS deformation. We provide them here.

For each frame t with estimated pose $\boldsymbol{\theta}_t$, we deform the canonical Gaussians into posed space via Linear Blend Skinning (LBS). The per-Gaussian rotation is:

$$\mathbf{R}^H_{k,t} = \sum\nolimits_{j=1}^{J} \mathbf{w}^H_{k,j}\, \mathbf{R}^{\text{SMPL}}_j(\boldsymbol{\theta}_t), \tag{5}$$

where J is the number of SMPL joints and $\mathbf{R}^{\text{SMPL}}_j(\boldsymbol{\theta}_t)$ is the j-th joint rotation matrix under pose $\boldsymbol{\theta}_t$. The canonical Gaussian center $\boldsymbol{\mu}_k$ and covariance $\boldsymbol{\Sigma}_k$ are then transformed as:

$$\boldsymbol{\mu}'_{k,t} = \mathbf{R}^H_{k,t}\boldsymbol{\mu}_k + \mathbf{t}^H_{k,t}, \quad \boldsymbol{\Sigma}'_{k,t} = \mathbf{R}^H_{k,t}\boldsymbol{\Sigma}_k\big(\mathbf{R}^H_{k,t}\big)^\top, \tag{6}$$

where $\mathbf{t}^H_{k,t}$ is the translational component of the per-Gaussian LBS transform.

6 Coarse Avatar Training Objective

In Sec. 3.1 of the main paper, we refer to the supplementary for details on the coarse avatar training objective including the perceptual loss. We provide them here.

The coarse avatar is supervised by two sources. Let $\hat{\mathbf{I}}_t = \mathcal{R}(\mathcal{T}(\mathcal{G}^H; \boldsymbol{\theta}_t); \boldsymbol{\gamma}^{\text{obs}})$ denote the rendering at frame t, where $\boldsymbol{\gamma}^{\text{obs}}$ is the observed camera. The coarse reconstruction loss is:

$$\mathcal{L}_{\text{coarse}} = \sum_{v=1}^{V} \|\mathbf{I}_*^v - \mathcal{R}(\mathcal{G}^H; \boldsymbol{\gamma}_v)\|_1 \; + \; \sum_{t=1}^{T} \|(\mathbf{I}_t - \hat{\mathbf{I}}_t) \odot \mathbf{M}_t\|_1, \tag{7}$$

where $\odot$ denotes element-wise multiplication and $\mathbf{M}_t$ masks out occluded regions. The first term provides canonical-view supervision; the second supervises only the visible portions of each observed frame. The total objective is:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{coarse}} + \lambda_{\text{per}} \mathcal{L}_{\text{per}} + \lambda_{\text{reg}} \mathcal{L}_{\text{reg}}, \tag{8}$$

where $\mathcal{L}_{\text{per}}$ is a perceptual loss [14] with the same masking strategy, and $\mathcal{L}_{\text{reg}}$ encourages the learned skinning weights to stay close to SMPL-derived values.

7 PCA Pose Projection

In Sec. 3.4 of the main paper, we refer to the supplementary for details on PCA-based pose projection. We provide them here.

While the training pose distribution is broad (Sec. 3.3.4 of the main paper), novel driving poses may still lie outside the span of training position maps, causing the StyleUNet to produce artifacts. Following [8], we apply PCA-based pose projection at inference time. We perform PCA on the position map features $\{\mathbf{x}_t\}$ collected during training, retaining N_{pca} principal components $\mathbf{S}$ with standard deviations $\{\sigma_i\}$. For a novel driving pose with position map feature $\mathbf{x}$, we project and clip:

$$\boldsymbol{\xi} = \mathbf{S}^\top (\mathbf{x} - \bar{\mathbf{x}}), \quad \xi_i \leftarrow \text{clip}(\xi_i, -2\sigma_i, 2\sigma_i), \tag{9}$$

and reconstruct $\mathbf{x}_{\text{recon}} = \mathbf{S}\boldsymbol{\xi} + \bar{\mathbf{x}}$, where $\bar{\mathbf{x}}$ is the training mean. The clipped reconstruction constrains novel poses to lie within the training distribution, ensuring that the StyleUNet receives in-distribution inputs.

8 Network Architecture

In Sec. 3 of the main paper, we describe our pipeline components. Here we provide detailed architecture specifications.

StyleUNet. Following AnimatableGaussians [8] and GASPACHO [9], we employ the StyleUNet architecture (a U-Net augmented with StyleGAN2 [4] modulated convolutions) as the backbone for Gaussian map prediction. An overview of the architecture is shown in Fig. 2.

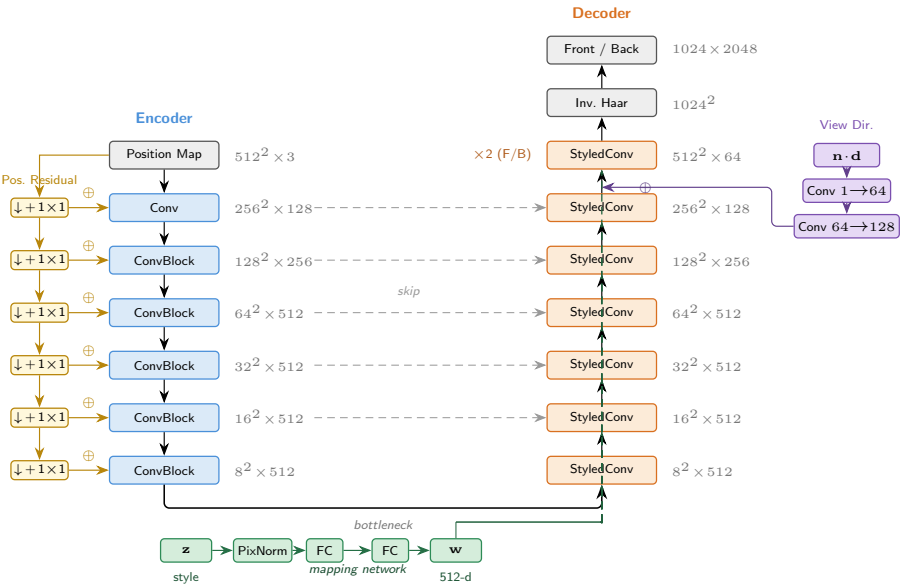

Fig. 2: StyleUNet architecture overview. The encoder processes position maps through progressive downsampling with residual position injection at each scale. The decoder uses StyleGAN2 modulated convolutions with skip connections from the encoder. A mapping network transforms a style vector $\mathbf{z}$ into $\mathbf{W}$ space, which modulates all decoder layers. View-direction features are injected at the 256×256 level for view-dependent color prediction. An inverse Haar transform converts the 512×512 wavelet output to 1024×1024 resolution. Two parallel decoder branches (front/back) produce the final 1024×2048 output map.

The encoder processes 512×512 position maps (3 channels) through a series of downsampling blocks. An initial convolution maps the input from 3 to 128 channels at 256×256 resolution. Five subsequent `ConvBlock` stages progressively downsample to an 8×8 bottleneck with 512 channels. At each stage, the position map is also independently downsampled and projected via a 1×1 convolution, then added as a residual to the main feature path. The channel widths are: 128 at 256×256, 256 at 128×128, then 512 at all remaining resolutions (64×64 down to 8×8).

Two parallel decoder branches (front and back UV maps) upsample from the 8×8 bottleneck to 512×512 in the Haar wavelet domain, which is converted to 1024×1024 spatial resolution via an inverse Haar wavelet transform. Each branch consists of six upsampling stages using 3×3 modulated convolutions with weight demodulation and per-pixel noise injection, following StyleGAN2. The decoder channel widths are: 512 from 16×16 to 64×64, then 256 at 128×128, 128 at 256×256, and 64 at 512×512. Standard U-Net skip connections concatenate encoder features at each corresponding decoder level.

A mapping network (pixel normalization + two equalized-learning-rate linear layers, $512 \to 512$, lr_mul $= 0.01$) maps a 512-dimensional style vector to $\mathbf{W}$ space for decoder weight modulation. Style vectors are initialized as constant unit vectors and kept fixed during inference.

The output comprises three heads: (1) a color network ($C_{\text{out}} = 3$) predicting RGB color offsets, (2) a position network ($C_{\text{out}} = 3$) predicting 3D position offsets (scaled by 0.05), and (3) an attribute network ($C_{\text{out}} = 8$) predicting opacity (1), scales (3), and rotation quaternion (4). The StyleUNet additionally predicts per-Gaussian skinning weights $\mathbf{w}_{k,j}^{H}$ for LBS deformation (Sec. 5).

For the color head, a small CNN (`Conv2d`$(1 \to 64, 4{\times}4, \text{stride } 2)$ with LeakyReLU, followed by `Conv2d`$(64 \to 128, 4{\times}4, \text{stride } 2)$) encodes a view-direction dot-product map (surface normal dotted with the viewing direction), injected additively at the 256×256 decoder level for view-dependent color prediction.

LoRA. For identity-specific video diffusion finetuning (Sec. 3.2.1), we apply LoRA with rank $r = 128$ to the Wan 2.2 [12] velocity field. We additionally learn a text token [v] that binds to the subject's identity in the text encoder's embedding space.

9 Training Procedure

In Sec. 3 of the main paper, we describe our three-stage pipeline. Here we provide training hyperparameters and implementation choices for all stages.

9.1 Stage 1: Coarse Avatar

Initialization. Since the coarse avatar uses a time-invariant canonical position map (Sec. 3.1 of the main paper), we first train the StyleUNet for approximately 2,000 iterations on this canonical map with a target of zero offsets, forcing the network to learn the identity mapping before any deformation.

Optimizer and learning rate. We use Adam [6] with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and an initial learning rate of 10^{-4}. We apply a cosine annealing schedule:

$$\text{lr}(i) = \text{lr}_{\text{init}} \cdot \left[\frac{1}{2} (\cos(\pi \cdot i/N) + 1)(1 - \eta_{\min}) + \eta_{\min} \right], \tag{10}$$

where i is the current iteration, N is the total number of iterations, and $\eta_{\min} = 0.05$ is the minimum learning rate factor, ensuring the learning rate never drops below 5% of the initial value.

Training duration. The coarse StyleUNet is trained for 400,000 iterations with a batch size of 1. At each iteration, we randomly sample one camera view and render a 512×512 patch for supervision.

Loss weights. The coarse training objective (Eq. 8) uses $\lambda_1 = 1.0$, $\lambda_{\text{per}} = 0.1$, and $\lambda_{\text{reg}} = 0.005$.

9.2 Stage 2: Identity-Specific LoRA Finetuning

Optimizer and learning rate. We use AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.99$, and weight decay 0.01. The learning rate is 10^{-4} for the LoRA parameters and 10^{-5} for the learnable text token. We apply gradient clipping with a maximum norm of 0.05.

Training duration. The LoRA and learnable text token [v] are trained for 600 iterations with a batch size of 8. We apply dropout regularization with rate 0.8 to the LoRA $\mathbf{B}$ matrices, and text prompt dropout with probability 0.1.

9.3 Stage 3: Full Avatar

Initialization. The canonical Gaussians are initialized to match the final coarse avatar Gaussians. Since Stage 3 uses pose-dependent position maps (Sec. 3.3.1 of the main paper), we pretrain the re-initialized StyleUNet for ∼2,000 iterations on randomly sampled pose-dependent maps, with the coarse avatar's final Gaussians as the target. This forces the network to directly predict the coarse avatar for any input pose before further training refines it. Pose and camera corrections $\{\Delta\boldsymbol{\theta}_{n,t}\}$, $\{\Delta\boldsymbol{\gamma}_{n,t}\}$ are initialized to zero.

Optimizer and learning rate. We use Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and an initial learning rate of 10^{-4}, with the same cosine annealing schedule (Eq. 10).

Training duration. The full avatar StyleUNet is trained for 500,000 iterations with a batch size of 1. At each iteration, we randomly sample one camera view and render a 512×512 patch for supervision.

Loss weights. The full training objective (Eq. 8) uses $\lambda_{\text{body}} = 1.0$, $\lambda_{\text{obs}} = 1.0$, $\lambda_{\text{head}} = 0.5$, $\lambda_{\text{per}} = 0.1$, and $\lambda_{\text{reg}} = 0.005$.

References

1. Chen, W., Li, P., Zheng, W., Zhao, C., Li, M., Zhu, Y., Dou, Z., Wang, R., Liu, Y.: Synchuman: Synchronizing 2D and 3D generative models for single-view human reconstruction. In: Advances in Neural Information Processing Systems (NeurIPS) (2025)
2. Esser, P., Kulal, S., Blattmann, A., Entezari, R., Müller, J., Saini, H., Levi, Y., Lorenz, D., Sauer, A., Boesel, F., Podell, D., Dockhorn, T., English, Z., Rombach, R.: Scaling rectified flow transformers for high-resolution image synthesis. In: International Conference on Machine Learning (ICML) (2024)
3. Güler, R.A., Neverova, N., Kokkinos, I.: DensePose: Dense human pose estimation in the wild. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
4. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of StyleGAN. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
5. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (2023)
6. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (ICLR) (2015)
7. Li, P., Zheng, W., Liu, Y., Yu, T., Li, Y., Qi, X., Chi, X., Xia, S., Cao, Y.P., Xue, W., Luo, W., Guo, Y.: Pshuman: Photorealistic single-image 3D human reconstruction using cross-scale multiview diffusion and explicit remeshing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2025)
8. Li, Z., Zheng, Z., Wang, L., Liu, Y.: Animatable gaussians: Learning pose-dependent gaussian maps for high-fidelity human avatar modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2024)
9. Mir, A., Moreau, A., Dhamo, H., Zhang, Z., Pons-Moll, G., Pérez-Pellitero, E.: Gaspacho: Gaussian splatting for controllable humans and objects. arXiv preprint arXiv:2503.09342 (2025)
10. Neverova, N., Güler, R.A., Kokkinos, I.: Dense pose transfer. In: European Conference on Computer Vision (ECCV) (2018)
11. Qiu, L., Gu, X., Li, P., Zuo, Q., Shen, W., Zhang, J., Qiu, K., Yuan, W., Chen, G., Dong, Z., Bo, L.: LHM: Large animatable human reconstruction model for single image to 3D in seconds. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2025)
12. Wan Team: Wan: Open and advanced large-scale video generative models. arXiv preprint arXiv:2503.20314 (2025)
13. Wang, L., Zhao, X., Sun, J., Zhang, Y., Zhang, H., Yu, T., Liu, Y.: StyleAvatar: Real-time photo-realistic portrait avatar from a single video. In: ACM SIGGRAPH Conference Proceedings (2023)
14. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
15. Zhuang, Y., Lv, J., Wen, H., Shuai, Q., Zeng, A., Zhu, H., Chen, S., Yang, Y., Cao, X., Liu, W.: IDOL: Instant photorealistic 3D human creation from a single image. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2025)