Supplementary Materials
ODHSR: Online Dense 3D Reconstruction of Humans and Scenes from Monocular Videos

Zetong Zhang[1] Manuel Kaufmann[1] Lixin Xue[1] Jie Song[1,2,3] Martin R. Oswald[4]

[1]ETH Zürich [2]HKUST(GZ) [3]HKUST [4]University of Amsterdam

In this supplementary document, we present the detailed design of our avatar local deformation network, keyframe management module and proposes regularizations on the avatar representation. All the hyperparameters involved in our experiments are also introduced. In addition to the evaluation results in the main paper, we show qualitative results of novel view synthesis task on the NeuMan dataset and human pose estimation comparisons. We further demonstrate the effectiveness of our avatar representation components in the ablation study.

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

Contents

A. Time-pose Dependent Deformation Network

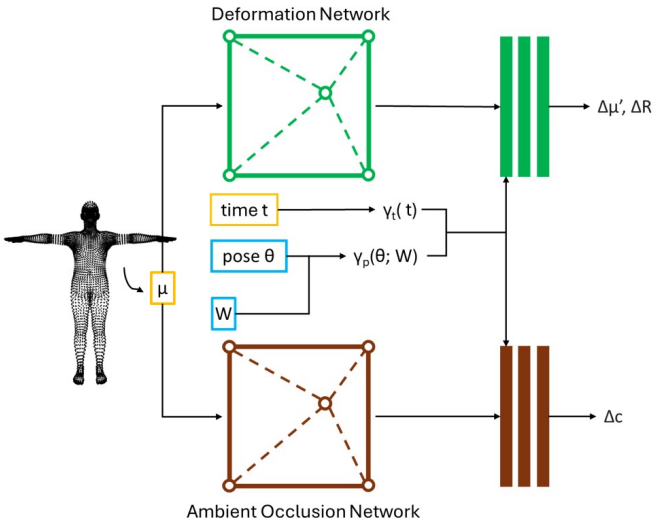

Figure 5. Local deformation and ambient occlusion network. Yellow: Fixed parameters; Blue: Frozen parameters.

In this section, we present the details of our designed time-pose dependent non-rigid deformation and appearance module introduced in Sec. 3.1 in the main paper. [1]

Our network design is shown in Fig.5, where two parallel multiresolution hash encoding networks are utilized to learn geometric and photometric deformation respectively. Given time step t, human pose parameter θ and per-Gaussian LBS

[1]In Sec. A and Sec. C, we omit the subscripts of human Gaussian parameters for simplicity.

weight W, the encodings of time and pose are denoted as $\gamma_t(t)$ and $\gamma_p(\theta; W)$, respectively. Specifically, we use the positional encoding as in [11] to encode the normalized input time, where the max degree is set to be 4. For the pose, we follow the idea of [12] to use an attention-weighting scheme to encode only the pose parameters of joints that are close to the Gaussian center. By doing this, the redundant information in the global pose parameters θ can be removed so that the local deformation around the input Gaussian will be better learned without the spurious correlation of irrelevant joints. This is inspired by SCANimate [16], which uses the LBS weight and a predefined attention map V that limits the propagation of deformation within four neighboring joints in the kinematic tree. The pose encoding is then formulated as follows:

$$\gamma_p(\theta; W) = (V \cdot W) \odot \theta \tag{15}$$

where θ is in quaternion format and $\odot$ denotes element-wise multiplication.

Similar to [8], we use the fixed Gaussian centers μ as the input of the hash grid to compress the size of the hash table and prevent optimization from diverging owing to the unstable Gaussian displacements. The time and pose encodings are concatenated with the hash encoding features queried with the Gaussian center μ as the input of the shallow MLP networks to produce deformation $\Delta\mu'$, ΔR and 1-channel ambient occlusion Δc prediction. With the MLP architecture, we use its smoothness prior and expect good interpolation properties to be learned to facilitate generalization to novel frames and poses. In practice, we parameterize the rotation ΔR in the form of a quaternion vector and limit the ambient occlusion factor within the range of 0-2.

B. Keyframe Management

In this section, we introduce our designed criteria for keyframe selection selection.

▷ **Frame interval**: Only a frame whose time difference from the last keyframe is above a threshold τ_t can be chosen as a new keyframe so that we can avoid registering the keyframes too frequently and idling the main thread for a long period.

▷ **Camera motion**: If the displacement of the current camera from that of the last keyframe is larger than a threshold τ_c, we will add the current frame to the keyframe set to span a wide baseline.

▷ **Human motion**: We measure the averaged human joint displacement from the last keyframe for each frame to estimate the pose change. Large human motion is likely to lead to unobserved local non-rigid deformation and appearance change. Thus, we register frames with drastic pose change, where the joint displacement is above τ_j, to better model the garment deformation.

▷ **Gaussian covisibility**: 3D Gaussians respect visibility ordering since the rasterizer will sort Gaussians along the camera ray. Similar to [10], we mark Gaussians as visible if they contribute to the rendering from the camera view. We then compute the covisibility of all the Gaussians (human + scene) by computing the IOU value of visible ones between the current frame and the last keyframe. If the covisibility is below a threshold τ_v, the current frame will be selected as the new keyframe to reduce redundant visual overlap between keyframes.

When a new keyframe is added or the size of the local keyframe window is larger than τ_s, we update the window with the new keyframe. Previous keyframe whose overlap with the latest keyframe is below a threshold τ_r or the frame whose camera distance from other keyframes is the farthest will be removed from the current keyframe window. By doing this, we update the Gaussians and networks with the knowledge from the new keyframes, which can be generalized better to a subsequent frame.

C. Losses

We describe in this section the detailed formulation of proposed regularizations applied on the avatar representation that are introduced in Sec. 3.3.

Local Deformation Loss. We constrain the local deformation to be as small as possible and encourage the frame-generic model to best learn the average shape and average. The local deformation loss L_{deform} is composed of three parts that respectively penalize the displacement $\Delta\mu'$, push the rotation offset ΔR to be close to the identity matrix, and enforce the ambient occlusion factor Δc to stay close to 1.

$$L_{\text{deform}} = \|\Delta\mu'\|_2 + \|\Delta R - R_{id}\|_1 + \|\Delta c - 1\|_2 \tag{16}$$

In practice, we use quaternions to represent the rotations.

LBS Weights Loss. For each Gaussian in the canonical space, we use the K-Nearest Neighbor (KNN) algorithm to find the k nearest SMPL vertices v_{NN} and take the weighted sum of their LBS weights $\tilde{W}$ as the label to supervise the Gaussian LBS weights with $L_{\text{LBS}} = \|W - \tilde{W}\|_F$. Inspired by [7], the displacements between the nearest k SMPL vertices and the Gaussian center, formulated as $\Delta v_{NN} = \mu + \Delta\mu - v_{\text{NN}}$, are used to weigh each element so that SMPL vertices closer to the corresponding Gaussian will contribute more to the supervision. Differently, we propose a novel distance-based weighting that takes account of the shape and scale of each Gaussian and calculate $\tilde{W}$ as fol-

lows.

$$\tilde{W} = \sum_{i=1}^{k} \frac{w_i}{w} W_{\text{NN},i} \tag{17}$$

$$w_i = \exp\left(-\frac{1}{2} \Delta v_{\text{NN},i}{}^T \Sigma^{-1} \Delta v_{\text{NN},i}\right) \tag{18}$$

$$w = \sum_{i=1}^{k} w_i \tag{19}$$

where $\Delta v_{\text{NN},i}$ and $W_{\text{NN},i}$ are respectively the relative position and LBS weight of the i-th nearest vertice on the SMPL mesh from the Gaussian center, and Σ is the Gaussian covariance matrix that is defined as $\Sigma = RSS^TR^T$.

In our experiments, we set $k = 3$.

Canonical Center Loss. In our online pipeline, where the local window is typically small, each Gaussian is only visible to limited training views and is thus largely unconstrained. To prevent Gaussians from moving and growing arbitrarily along the camera ray, we softly regularize the geometry of the reconstructed human with the underlying SMPL model. Because the garment can lead to large displacements from naked-body SMPL to the reconstructed avatar, we do not directly regularize the magnitudes of the Gaussian displacements but instead enforce the nearest vertex on the SMPL mesh from each Gaussian to be the vertex used to initialize the Gaussian. The regularization is applied on the canonical Gaussian centers before local deformation as follows:

$$L_{\text{center}} = \text{ReLU}(\|\mu + \Delta\mu - v_{\text{init}}\|_2 - \|\mu + \Delta\mu - v_{\text{NN}}\|_2) \tag{20}$$

where v_{NN} is the nearest SMPL vertex from the Gaussian in the canonical space, and v_{init} is the corresponding vertex position initially. We run the K-Nearest Neighbor algorithm via the efficient CUDA implementation in PyTorch3D [14].

D. Implementation Details

D.1. Model Configurations

We initialize the canonical Gaussian positions by creating r replicates of each SMPL vertex in the canonical space and injecting Gaussian noises. r is set to be 5 for the EMDB dataset and 3 for the NeuMan dataset. The Gaussian opacities are initialized to be 0.9. We use the anisotropic Gaussians for both the scene and human parts.

For the LBS weights per Gaussian, we directly optimize an offset vector to sum to the original SMPL weights and use SoftMax as the activation function to apply to each element of the optimized weights to ensure that their values are all positive and sum to one.

For the time-pose dependent deformation network, the canonical points μ are first normalized with a bounding box that tightly encloses the canonical SMPL mesh. The detailed network hyperparameters are listed in Tab. 4.

Parameter	Value
Number of levels	16
Number of features per level	2
Hash table size	2^{17}
Coarsest resolution	4
Per Level Scale	1.5
MLP Width	128
MLP Number of hidden layers	3

Table 4. Local deformation and ambient occlusion network hyperparameters

D.2. Training Strategies

Hash Encoding Network Pretraining. Random initial values of the hash encoding network can produce incorrect output on the fly when there are insufficient training frames and the training iterations are limited. This is the typical situation in the online training pipeline. Good interpolation and extrapolation properties are required to quickly fit the novel keyframe with the knowledge learned from previous frames. Otherwise, the Gaussian parameters could also get optimized in the wrong direction.

Considering these issues, we propose to pre-train the local deformation and ambient occlusion networks introduced in Sec. A at the very beginning. This is achieved by randomly sampling input time and poses to obtain the deformation outputs from the hash encoding network and minimize the deformation loss L_{deform}. We sample the input time from a uniform distribution between 0 and 1. As for the human pose, we sample from a combination of the pose of the first frame and poses stored in a large-scale human database AMASS [9] so that the network is pre-trained with realistic poses of large variations. Gaussian noises with a standard deviation of 0.1 are added to the input pose to augment the data. In our experiments, the poses in the BMLmovi dataset [1] are used for sampling. We use Adam optimizer with learning rate 10^{-4} to run the optimization for 5000 iterations.

Multi-stage Training. We evenly divide the mapping process into two stages and choose not to include the time-pose-dependent deformation and ambient occlusion in avatar Gaussians in the first stage while later activate them in the second stage. This multi-stage training strategy is employed in both the online mapping and final color refinement steps.

D.3. Training Configurations

We use Adam Optimizer to optimize the camera and human pose parameters. The learning rates in the tracking thread are 3×10^{-3} for camera rotation, 10^{-3} for camera translation, 10^{-2} for the human root translation and orientation, and 10^{-3} for other local pose parameters. In the mapping thread where we simultaneously perform local bundle ad-

justment on the keyframe window, the learning rates are reduced to 1.5×10^{-3}, 5×10^{-4}, 10^{-4} and 10^{-5} respectively. The learning rates of all the Gaussian parameters are exactly the same as the original implementation from [6]. For our additionally designed time-pose dependent network, we set learning rate of all its parameters to be 10^{-4}.

In the tracking thread, we iteratively run camera and human pose optimization for 100 iterations with $\lambda_{\text{rgb}} = 1, \lambda_{\text{flow}} = 1, \lambda_{\text{disp}} = 0.001, \lambda_{\text{sil}} = 0.1, \lambda_{\text{kp}} = 0.0001$ in Eq. (14). While for mapping, we set $\lambda_{\text{rgb}} = 1, \lambda_{\text{sil}} = 1, \lambda_{\text{depth}} = 0.001, \lambda_{\text{LBS}} = 100, \lambda_{\text{center}} = 10, \lambda_{\text{deform}} = 0.001$ in Eq. (15). Optimized Gaussians in the mapping thread are synchronized with the tracking thread every 20 mapping iterations. Finally when we iterate over the whole sequence, we finetune the Gaussians with all the selected keyframes for 100 epochs.

For keyframe selection, we set $\tau_t = 0.1$s, $\tau_c = 0.05$m, $\tau_j = 0.1$m, $\tau_v = 0.9$ as the thresholds. As for the local keyframe update, we set $\tau_s = 10$ and $\tau_r = 0.3$.

For the scene representation, we periodically perform Gaussian densification and pruning as originally described by 3DGS [6]. In contrast, for the fixed-size human, we disable the adaptive seeding during the online mapping since the complicated topology of the human body and the limited training viewpoints can lead to noisy gradients, especially in the occluded human parts. The densification and pruning module will be later activated for humans in the final color refinement step to capture richer details.

D.4. Baselines

When assessing the performance of novel view synthesis, we optimize human poses across all test frames for the baseline methods to eliminate the impact of pose errors on rendering. In contrast, for our approach, this step is omitted because the test poses are already optimized dynamically during the process. By adopting this strategy, we provide an advantage to the baselines, as their test poses are refined against the final reconstruction to minimize re-rendering errors. Conversely, our test poses are optimized using the online reconstructed model, which may be incomplete, suboptimally refined, and therefore more susceptible to errors.

For consistency, we fix the Gaussian and network parameters across all methods and utilize each method's specific pose estimation module, applying the same loss functions used during their training to perform test pose optimization. This ensures that the evaluation of novel view synthesis reflects the robustness of the respective pose optimization designs as well. For direct pose estimation modules, as implemented in [2, 7, 13, 19], we employ a uniform learning rate of 10^{-3}. For the pose correction MLP network used in [3], we maintain the same learning rate as during training. To ensure fairness, pose optimization is conducted for 100 steps on each frame across all baselines.

	ATE RMSE [m]↓
DROID-SLAM	**0.079**
MonoGS (human masked)	0.459
Ours (human masked)	0.247
Ours (full model)	0.084

Table 5. Camera tracking evaluation on the EMDB dataset.

E. Additional Evaluation Results

E.1. Novel View Synthesis

Qualitative results on the NeuMan dataset [4] are presented in Fig. 6. Despite performing online tracking and mapping, our method surpasses most offline reconstruction approaches in terms of background scene fidelity and clarity, even though those methods leverage ground truth camera poses. Furthermore, our approach achieves superior quality in the reconstruction of critical and challenging human features, such as faces and hands. However, a limitation of our method is that geometry near contact points between the human and the scene may not always be precisely recovered, occasionally resulting in blurry reconstructions, as seen in areas like shoes and the ground.

E.2. Camera Tracking

We demonstrate that our camera tracker achieves on-par performance with the state-of-the-art SLAM approach DROID-SLAM[18] in Tab. 5. Without knowing the true scale, the output from DROID-SLAM cannot be seamlessly integrated with human pose estimates unless ground truth depth or trajectory information is provided, limiting its applicability in dynamic scenes. However, by explicitly building the dynamic human and modeling human-scene spatial correlation, our method handles the scaling well. Moreover, to further inspect the impact of human on the tracking, we run MonoGS[10] and our method while using pre-estimated human masks to completely remove the human in the input images and the model. As shown in Tab. 5, our method significantly enhances the accuracy of predicted camera trajectories by explicitly modeling the human, as it provides additional spatial cues and aids in scaling the monocular depth signal.

E.3. Human Pose Estimation

We evaluate our human pose estimations and compare them with WHAM[17] in Tab. 6 and Fig. 7. Our reconstruction-based pose optimization module achieves slightly enhanced local poses that align more accurately with the 2D image. For global motion, our holistic human-scene reconstruction supplies the essential spatial context, enabling the human tracker to significantly reduce globally aligned joint errors. In contrast, WHAM, lacking explicit scene awareness, fails to adapt to terrain changes, resulting in substantial trajectory errors. However, the increased jitter observed in our

(a) GT (b) Ours (c) Vid2Avatar (d) HSR (e) HUGS (f) 3DGS-Avatar (g) GauHuman

Figure 6. Qualitative comparison of novel view synthesis task on the NeuMan dataset [4].

method indicates a limitation: the gradient descent optimization approach becomes ineffective for occluded body parts that are not visible in the 2D image.

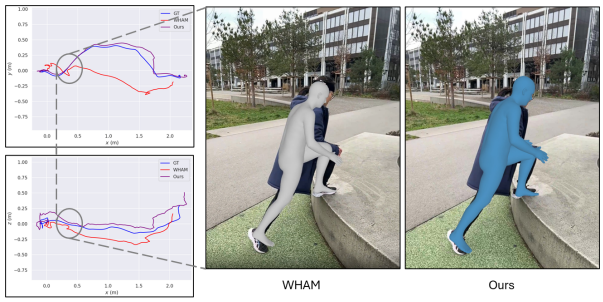

Figure 7. Comparison of global human trajectory estimations on the EMDB dataset. Left: Human trajectories of GT, WHAM predictions and our predictions on the x-y and x-z plane. The global trajectories are globally aligned. Right: Estimated SMPL mesh on one selected frame.

F. Ablation Study

F.1. Ablation of Avatar Module Designs

Input and output components of the avatar deformation module are ablated in Tab. 7. On the challenging EMDB dataset where drastic garment deformation and illumination change exist, jointly modeling the per-Gaussian deformation and ambient occlusion significantly improves all the re-rendering metrics. As for the input, we achieve the best performance by taking both the pose and time features compared to using either one of them.

F.2. Ablation of Hash Encoding Network Pretraining Strategy

In Tab. 7, we also present the evaluation results without pretraining the hash encoding network. Due to the randomized initial network parameters, the local deformation network produces noisy outputs, resulting in failed learning of garment deformation and shadows, particularly at unseen timesteps and poses. The bad interpolation and extrapolation properties lead to an overall degraded performance.

	Local Pose				Global Motion	
	PA-MPJPE↓	MPJPE↓	MVE↓	Jitter↓	WA-MPJPE↓	W-MPJPE↓
WHAM	40.845	72.964	83.254	**14.765**	636.001	2990.746
Ours	**40.571**	**69.162**	**79.463**	32.183	**175.215**	**449.036**

Table 6. Human pose estimation evaluation on the EMDB dataset. Jitter is in the unit of $10m/s^{-3}$ and others in mm.

	PSNR ↑	SSIM ↑	LPIPS ↓
w/o ambient occlusion	28.201	0.958	0.034
w/o deformation	27.927	0.959	0.036
w/o pose encoding	28.741	0.962	0.033
w/o time encoding	27.779	0.957	0.037
w/o HE pretraining	27.801	0.958	0.041
Full model	**28.955**	**0.966**	**0.031**

Table 7. Ablation study on avatar module designs and hash encoding (HE) network pretraining strategy. The performance is evaluated on the **human-only** rendering on the EMDB dataset.

Figure 8. Results in the scene occlusion scenario. Our generated human mask is compared against the prediction from the Segment Anything Model(SAM).

G. Discussions

G.1. Online Training

We follow existing dense SLAM works[5, 10, 18] to perform a final refinement step to finetune the Gaussian representation with all the selected keyframes. The refinement process can be seen as a traditional global bundle adjustment (BA) step, in which case it does not conflict with the online nature of ODHSR. Unlike other approaches, we do not perform full BA but instead refine only the Gaussians, allowing us to distribute the refinement into the online optimization rather than applying it as a post-processing step—though this comes at the cost of lower training FPS. By distributing refinement into the online pipeline after each keyframe tracking step and training for ten epochs per refinement operation, we achieve a final PSNR of 23.013 for the whole image and 28.814 for human-only regions, which is slightly worse than the full model and increases runtime (reducing FPS by 0.06). The final refinement step is designed to prevent catastrophic forgetting, and we showcase that without this, ODHSR still largely overperforms baselines in novel view synthesis and runtime efficiency.

G.2. Challenging Cases

Scene Occlusion. We demonstrate the impact of our occlusion-aware human silhouette design in Fig. 8. For body parts occluded by scene components, such as legs, ODHSR consistently generates smooth and precise boundary silhouettes. In contrast, the state-of-the-art general segmentation model SAM[15], while capable of predicting occlusions, occasionally produces results with missing parts. By explicitly modeling occlusions, ODHSR effectively models spatial correlations without losing human features.

Long Trajectories. In Fig. 9, we showcase the results of our method in a long-trajectory scenario, where repetitive background patterns pose challenges for camera tracking. Overall, ODHSR delivers decent results and effectively captures camera motion trends with small trajectory errors. However, the sparse features on the wall and ground increase the challenge of accurate geometric reconstruction, introducing some surface noise that subsequently leads to additional errors in the estimated camera poses for certain frames. DROID-SLAM performs better in such scenarios by leveraging cleverer bundle adjustment and graph-based optimization strategy, highlighting a promising direction for further improvements.

G.3. Limitations

While ODHSR achieves state-of-the-art rendering quality on the challenging in-the-wild dataset, its performance heavily depends on single-frame pre-estimations, such as monocular depth and human keypoints—particularly in the first frame, which initializes the system. Although we incorporate a pairwise flow loss in camera and human pose optimization, we argue that this alone is insufficient for constructing a globally consistent scene and pose representation. Also, despite producing high-quality renderings, our method introduces surface noise due to the nonsmooth depth characteristics of 3D Gaussian Splatting. Additionally, our method could suffer from potential human-scene interpenetrations around the contact points, such as feet. Due to the noisy surfaces 3D Gaussians produce, it is not yet resolved. Finally, our model-based camera and human pose optimization primarily relies on pixel-level errors, which can lead to local optima in textureless regions or areas with uniform features, such as walls and clothing.

Figure 9. Results in the long trajectory scenario. Left: Our human-scene reconstruction with tracked cameras. Right: Estimated trajectories from ours and DROID-SLAM, compared with the ground truth on the EMDB dataset. Colors of the curve segments indicate trajectory error, ranging from 0. to 1.

References

[1] Saeed Ghorbani, Kimia Mahdaviani, Anne Thaler, Konrad Kording, Douglas James Cook, Gunnar Blohm, and Nikolaus F. Troje. Movi: A large multipurpose motion and video dataset, 2020. 3

[2] Chen Guo, Tianjian Jiang, Xu Chen, Jie Song, and Otmar Hilliges. Vid2avatar: 3d avatar reconstruction from videos in the wild via self-supervised scene decomposition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 4

[3] Shoukang Hu and Ziwei Liu. Gauhuman: Articulated gaussian splatting from monocular human videos. *arXiv preprint arXiv:*, 2023. 4

[4] Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan. Neuman: Neural human radiance field from a single video. In *Proceedings of the European conference on computer vision (ECCV)*, 2022. 4, 5

[5] Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. Splatam: Splat, track and map 3d gaussians for dense rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 6

[6] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023. 4

[7] Muhammed Kocabas, Rick Chang, James Gabriel, Oncel Tuzel, and Anurag Ranjan. Hugs: Human gaussian splats, 2023. 2, 4

[8] Yang Liu, Xiang Huang, Minghan Qin, Qinwei Lin, and Haoqian Wang. Animatable 3d gaussian: Fast and high-quality reconstruction of multiple human avatars. *arXiv preprint arXiv:2311.16482*, 2023. 2

[9] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *International Conference on Computer Vision*, pages 5442–5451, 2019. 3

[10] Hidenobu Matsuki, Riku Murai, Paul H. J. Kelly, and Andrew J. Davison. Gaussian Splatting SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. 2, 4, 6

[11] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 2

[12] Arthur Moreau, Jifei Song, Helisa Dhamo, Richard Shaw, Yiren Zhou, and Eduardo Pérez-Pellitero. Human gaussian splatting: Real-time rendering of animatable avatars, 2024. 2

[13] Zhiyin Qian, Shaofei Wang, Marko Mihajlovic, Andreas Geiger, and Siyu Tang. 3dgs-avatar: Animatable avatars via deformable 3d gaussian splatting. In *CVPR*, 2024. 4

[14] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 3

[15] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, Zhaoyang Zeng, Hao Zhang, Feng Li, Jie Yang, Hongyang Li, Qing Jiang, and Lei Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 2024. 6

[16] Shunsuke Saito, Jinlong Yang, Qianli Ma, and Michael J. Black. SCANimate: Weakly supervised learning of skinned

clothed avatar networks. In *Proceedings IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[17] Soyong Shin, Juyong Kim, Eni Halilaj, and Michael J. Black. Wham: Reconstructing world-grounded humans with accurate 3d motion. In *Computer Vision and Pattern Recognition (CVPR)*, 2024. 4

[18] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras, 2022. 4, 6

[19] Lixin Xue, Chen Guo, Chengwei Zheng, Fangjinhua Wang, Tianjian Jiang, Hsuan-I Ho, Manuel Kaufmann, Jie Song, and Hilliges Otmar. HSR: holistic 3d human-scene reconstruction from monocular videos. In *European Conference on Computer Vision (ECCV)*, 2024. 4