ASH: Animatable Gaussian Splats for Efficient and Photoreal Human Rendering – Supplemental Document –

Haokai Pang[1,2 †] Heming Zhu[1 †] Adam Kortylewski[1,3] Christian Theobalt[1,4] Marc Habermann[1,4 ✉]
[1] Max Planck Institute for Informatics, Saarland Informatics Campus
[2] ETH Zürich [3] Universität Freiburg
[4] Saarbrücken Research Center for Visual Computing, Interaction and AI
{hpang, hezhu, akortyle, theobalt, mhaberma}@mpi-inf.mpg.de

In this appendix, we provide more details regarding the following aspects: More implementation details (Sec. 1); more ablative studies (Sec. 2); more results with driven poses from novel datasets (Sec. 3); runtime analysis for the major components (Sec. 4); realtime applications built upon ASH (Sec. 5); more detailed discussion on limitations and future directions (Sec. 6).

1. Implementation Details

In the main paper, we mentioned that ASH learns the Gaussian splat parameters in the 2D texture space of an animatable human template $M(\boldsymbol{\theta}_f) = \mathbf{V}_f$. Here, we provide more details regarding the deformable template mesh and the motion-aware decoders.

Deformable Template Mesh. We adopt the formulation introduced in Habermann et al. [3] for modeling the deformable template mesh, which deforms the template mesh vertices $\mathbf{V}_\mathrm{m}$ in the canonical space with a learned embedded deformation [9, 10]:

$$\bar{\mathbf{V}}_{f,i} = \mathbf{D}_i + \sum_{j \in \mathcal{N}_{\mathrm{nv},i}} w_{i,j}(R(\mathbf{A}_j)(\mathbf{V}_{\mathrm{m},i} - \mathbf{V}_{\mathrm{G},j}) + \mathbf{V}_{\mathrm{G},j} + \mathbf{T}_j) \tag{1}$$

where $\bar{\mathbf{V}}_{f,i} \in \mathbb{R}^3$ denotes the deformed template vertices in the rest pose. $\mathcal{N}_{\mathrm{nv},i} \in \mathbb{N}$ indicates the indices for the embedded graph node [10] that are connected to the i th vertex on the template mesh. $\mathbf{V}_{\mathrm{G},j} \in \mathbb{R}^3$, $\mathbf{A}_j \in \mathbb{R}^3$, and $\mathbf{T}_j \in \mathbb{R}^3$ denotes the rest positions, Euler angles, and translations of the embedded graph nodes. Notably, the connectivity of the embedded graph $\mathbf{V}_{\mathrm{G},j}$ can be adopted by simplifying the template mesh M using quadric edge collapse decimation [1, 2]. Moreover, the connection, as well as the connection weights $w_{i,j}$, between the template mesh $\mathbf{V}_\mathrm{m}$ and the embedded graph are generated Sumner et al. [10]. $R(\cdot) \in \mathbb{R}^{3\times3}$ denotes the function that converts the Euler angle to a rotation matrix. $\mathbf{D}_i \in \mathbb{R}^3$ indicates the per-vertex displacement to model an even finer level of geometry details. Specifically, embedded graph parameters $\mathbf{V}_{\mathrm{G},j}$, $\mathbf{A}_j$, and per-vertex displacements $\mathbf{D}_i$ are derived from skeletal motion $\bar{\boldsymbol{\theta}}_f$ with structure-aware graph convolution neural networks. We refer to Harbermann et al. [3] for more details.

Motion-aware Decoders. ASH adopts motion-aware 2D convolutional neural networks, i.e., the geometry network $\mathcal{E}_\mathrm{geo}$, and the appearance network $\mathcal{E}_\mathrm{app}$, predicting the geometry and appearance parameters of the Gaussian splats from the motion-aware textures $(\mathbf{T}_{\mathrm{n},f}, \mathbf{T}_{\mathrm{p},f})$. Both the geometry network $\mathcal{E}_\mathrm{geo}$ and the appearance network $\mathcal{E}_\mathrm{app}$ are U-Nets implemented following the configuration mentioned in Olaf et al. [8]. Specifically, we channel-wise concatenate the global appearance features $\Phi_f \in \mathbb{R}^{16}$ to the bottleneck features of the appearance network $\mathcal{E}_\mathrm{app}$ to account for the lighting variations in the studio. The global appearance feature Φ_f is derived from positional encoded skeleton root translation with a 3-layer shallow MLP, of which the width is set to 32.

2. Ablations

In this section, we provide more ablative studies to demonstrate the effectiveness of ASH.

Number of Camera Views. To assess the robustness of ASH against sparser camera view supervision, we conducted ablative experiments that take multi-view videos from 12, 30, and 60 cameras as supervision, termed as **w/ 12.cam**, **w/ 30.cam**, and **w/ 60.cam**. Note that the selected camera views are evenly distributed in the studio. As illustrated in Fig. 1 and Tab. 1, ASH can still accurately synthesize the animatable characters when training with sparser input views.

The Impact of 2D Learning. To validate the efficacy of the

† Joint first authors.
✉ Corresponding author.
Project page: vcai.mpi-inf.mpg.de/projects/ash

Methods	Training Pose		Testing Pose	
	PSNR	**LPIPS**	**PSNR**	**LPIPS**
w/ MLP	28.05	34.93	26.79	35.00
w/ 12.cam	33.50	10.76	26.96	21.66
w/ 30.cam	35.37	8.52	27.08	20.76
w/ 60.cam	35.49	8.32	27.14	20.42
Ours	35.47	8.30	27.13	20.22

Table 1. **Ablation Study**. We further assess our design choices on the image synthesis tasks with the subject wearing loose outfits in the DynaCap [3] dataset. We highlight the best and the second-best scores.

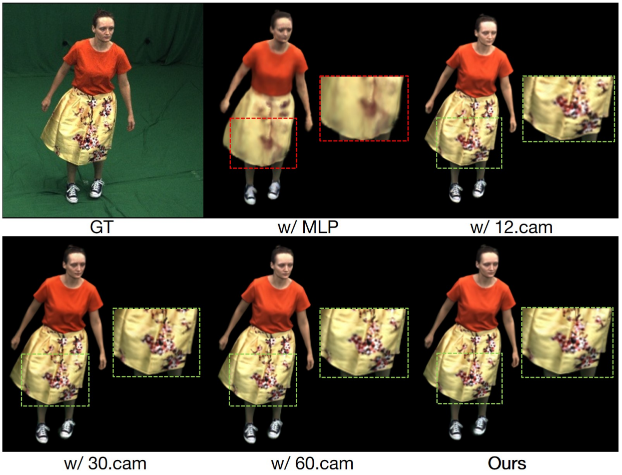


Figure 1. **Qualitative Ablation.** We compare ASH with the models that take alternative design choices. ASH excels in rendering quality than the model directly learns the Gaussian parameters from 3D canonical space (**w/ MLP**). Moreover, ASH exhibits robustness against fewer training views (**w/ 12.cam**, **w/ 30.cam**, **w/ 60.cam**).

Methods	Training Pose		Testing Pose	
	PSNR	**LPIPS**	**PSNR**	**LPIPS**
SMPL	34.93	14.90	28.75	24.87
Ours	35.47	8.30	27.13	20.22

Table 2. **ASH conditioned on SMPL.** ASH achieves significantly better quantitative performance than the SOTA real-time methods.

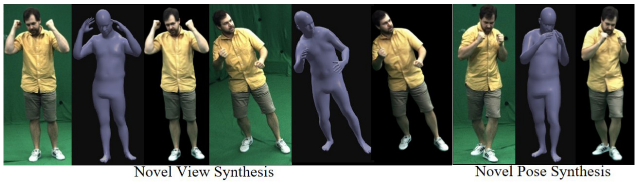


Figure 2. **ASH conditioned on SMPL.** Despite large deviations between the underlying template and the real surface, ASH generates visually plausible results.

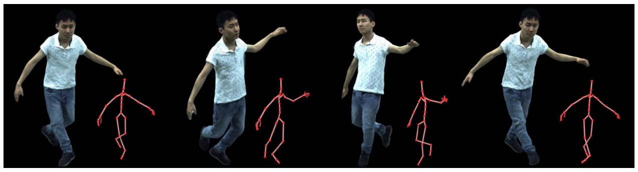


Figure 3. **Results with AMASS DanceDB motion.** ASH produces photorealistic rendering given the motion from an entirely different dataset.

2D texel paradigm for 3D Gaussian splats, we conducted an ablative experiment that predicts the 3D Gaussian parameters directly from 3D, termed as **w/ MLP**. We adopted an 8-layer MLP that consumes the skeletal motion $\bar{\boldsymbol{\theta}}_f$ and the positional-encoded canonical Gaussian position $\bar{\boldsymbol{\mu}}_i$, predicting the Gaussian splat parameters $\{\mathcal{G}_i\}$ in the canonical space. Specifically, the width for the hidden layers of the MLP is set to 256. Similar to ASH, the canonical Gaussian splats are transformed to observation space through Dual Quaternion skinning [4]. As illustrated in Fig. 1 and Tab. 1, directly learning the Gaussian parameters in 3D will lead to blurry rendering and cannot preserve the motion-dependent wrinkle details. In contrast, ASH, which formulates the learning of 3D Gaussian splats as image translation in 2D texel space, delivers high-quality rendering with delicate details.

The Reliance on the Accuracy of the Template. Although our method is conditioned on a template mesh, it can compensate for tracking errors with learnable motion-aware residual deformations for the Gaussian splats. To validate our method's robustness against errors in mesh tracking, we replaced the original template mesh with *SMPL body meshes* [6]. Despite large deviations between the template and ”the real surface,” our method generates visually plausible results (Fig. 2) and achieves significantly better quantitative performance than the SOTA real-time methods (Tab. 2), which heavily relies on an accurate deformable template [3].

3. More Results

In Tab. 2 in the main paper, we report the quantitative and qualitative performance on the testing set of the DynaCap dataset, which is an established and challenging benchmark, with the testing set containing more than 7000 frames showing strongly varying poses.

To further highlight the pose generalization ability of ASH, we retarget our skeleton to SMPL motions from the AMASS dataset (DanceDB) [7] to drive our character. Fig. 3 illustrates that even for motions from an entirely different dataset, ASH could generate photoreal rendering with delicated wrinkle details.

Method	Stg.1	Stg.2	Stg.3	Stg.4	Time	FPS
w/ 128.res.	20.28	1.93	5.57	1.91	29.69	33.68
w/ 512.res.	24.25	16.34	18.45	4.31	63.35	15.79
Ours	**21.03**	**3.60**	**7.00**	**2.11**	**33.74**	**29.64**

Table 3. **Runtime Analysis**. We present detailed runtime for each stage in ASH measured in milliseconds. We also report the runtime of the models that take halved and doubled texel resolution, termed as **w/ 128.res.** and **w/ 512.res.**, respectively. Note that ASH can render high-quality animatable humans in a real-time frame rate.

4. Runtime Analysis

In this section, we conduct a detailed runtime analysis for each major component in ASH. Specifically, we record the runtime for each major component when rendering a 1K (1285×940) image on a single Nvidia Tesla A100 graphics device. Additionally, the runtime analysis is benchmarked on models with different texture space resolution, specifically at 128, 512, and 256, referred to as **w/ 128.res**, **w/ 512.res**, and **Ours**, respectively. Here, we divide the rendering pipeline of ASH into four steps:

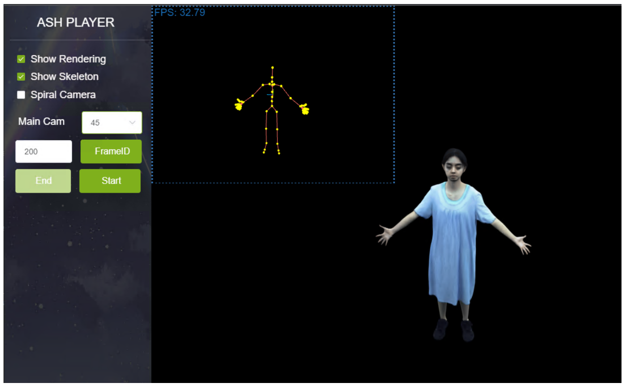
- Creating the deformable template meshes $M(\boldsymbol{\theta}_f)$ from skeletal motions $\boldsymbol{\theta}_f$ with structure-aware graph convolution networks, termed as **Stg.1**.
- Computing motion-aware texture maps $(\mathbf{T}_{\mathrm{n},f}, \mathbf{T}_{\mathrm{p},f})$ from deformable template meshes $M(\boldsymbol{\theta}_f)$, termed as **Stg.2**.
- Predicting the canonical Gaussian splats $\{\mathcal{G}_i\}$ with motion-ware geometry decoder $\mathcal{E}_{\mathrm{geo}}$ and appearance decoder $\mathcal{E}_{\mathrm{app}}$, termed as **Stg.3**.
- Performing tile-based rasterization with the predicted Gaussian splats $\{\mathcal{G}_i\}$, termed as **Stg.4**.

Tab. 3 illustrates the runtime for each component in ASH for models with different 2D texel resolutions. While halving the texel resolution (**w/ 128.res.**) speeds up the image synthesis of the animatable humans, it may produce blurry details in the rendered images. Doubling the texel resolution (**w/ 512.res.**) results in comparable rendering quality. Nevertheless, it significantly increases computational complexity, preventing the model from being real-time compatible. In contrast, ASH can generate high-fidelity renderings of animatable characters in a real-time frame rate.

5. Application

In this section, we introduce ASH Player, a real-time application built upon ASH.

Fig. 4 presents a screenshot of ASH Player, which runs in the web browser on a personal computer. The backend model of ASH Player, i.e., ASH, is deployed on the GPU cluster server. Once users specify the skeletal poses and virtual camera views, ASH Player will present the photoreal rendering of animatable characters, which is real-time computed and streamed from the GPU cluster server. Moreover, ASH Player allows users to inspect the animatable characters with spiral camera views. Please refer to the supplementary video for a more comprehensive visualization.


FPS: 32.79

ASH PLAYER

☑ Show Rendering

☑ Show Skeleton

□ Spiral Camera

Main Cam: 45

200

Figure 4. **System Overview.** ASH Player is an interface runs in the browser, visualizing the imagery and skeletal poses of animatable characters. The renderings of the animatable humans are computed in real time from the GPU cluster server and streamed to the ASH Player front-end interface on a personal computer.

6. Limitations

Although ASH enables high-fidelity, real-time rendering of animatable human characters, it has certain limitations that we hope to address in the future. Firstly, ASH does not extract detailed explicit geometry from the Gaussian splats. We will explore refining the explicit template meshes by backpropagating the gradient from image space into the template meshes using splatting. Additionally, ASH does not model topological changes like opening a jacket. Future research might focus on modeling the topological changes with the adaptive adding and removal of Gaussian splats introduced in the original 3D Gaussian splatting paper [5]. Lastly, as various factors could affect the appearance of dynamic clothed humans, it is unfeasible to establish a one-to-one correspondence between the skeletal motions and the dynamic clothed human appearance. Future research will explore different types of fine-grained control to define human rendering, e.g., the external physical forces.

References

[1] Paolo Cignoni, Guido Ranzuglia, M Callieri, M Corsini, F Ganovelli, N Pietroni, M Tarini, et al. Meshlab. 2011. 1

[2] Michael Garland and Paul S Heckbert. Surface simplification using quadric error metrics. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 209–216, 1997. 1

[3] Marc Habermann, Lingjie Liu, Weipeng Xu, Michael Zollhoefer, Gerard Pons-Moll, and Christian Theobalt. Real-time deep dynamic characters. *ACM Trans. Graph.*, 40(4), 2021. 1, 2

[4] Ladislav Kavan, Steven Collins, Jiří Žára, and Carol O'Sullivan. Skinning with dual quaternions. In *Proceedings of the 2007 symposium on Interactive 3D graphics and games*, pages 39–46, 2007. 2

[5] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):1–14, 2023. 3

[6] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM Transactions on Graphics*, 34(6), 2015. 2

[7] 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. 2

[8] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, Oct 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015. 1

[9] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*. Eurographics Association, 2007. 1

[10] Robert W. Sumner, Johannes Schmid, and Mark Pauly. Embedded deformation for shape manipulation. *ACM Trans. Graph.*, 26(3):80–es, 2007. 1