

---

# MPMAvatar: Learning 3D Gaussian Avatars with Accurate and Robust Physics-Based Dynamics – Supplementary Material –

---

Anonymous Author(s)

Affiliation

Address

email

## 1 A Additional Results

2 In the supplementary video (NeurIPS2025\_MPMAvatar\_Supplementary\_Video.mp4), we show the  
3 additional video results of our experiments.

4 **Qualitative Comparisons (Sec. A.1).** We present our video results in comparison with PhysA-  
5 vatar [10], a state-of-the-art physics-based avatar method. Our approach consistently achieves more  
6 accurate garment dynamics and higher rendering quality.

7 **Additional Qualitative Results (Sec. A.2).** We also include additional qualitative results of our  
8 method on (1) novel pose driving and (2) zero-shot scene interactions. For novel pose driving, our  
9 method accurately models garment deformation driven by motions from the AMASS [5] dataset,  
10 which contains relatively more dynamic motions than our training motions in [1, 7]. For novel scene  
11 interactions, our method models plausible interactions between garments and a variety of materials  
12 (e.g., cushion, sand), as well as mesh-based colliders (e.g., chair, rotor) that were unseen during  
13 training. This generalization is attributed to (1) the versatility of MPM [3] and (2) our effective  
14 mesh-based collision handling method.

15 **Ablation Results (Sec. A.3)** In the supplementary video, we additionally validate each of the key  
16 components of our method: (1) the constitutive model for anisotropic elastoplasticity [2], (2) physical  
17 parameter learning, (3) quasi-shadowing, and (4) rest-geometry modeling. In the fourth ablation, we  
18 directly use the canonical geometry as the rest geometry. Specifically, we optimize only Young’s  
19 modulus  $E$  and density  $\rho$ , while keeping the rest geometry parameter  $\alpha$  fixed at 1. Each component  
20 is shown to be critical for accurate dynamics and appearance modeling.

## 21 B Implementation Details

### 22 B.1 Physics-Based Dynamics Modeling

23 **Collision handling.** In Alg. 1, we outline our collision handling algorithm. After parameter  
24 initialization (lines 1-3), we iterate over every face  $f$  in the collider mesh and transfer its velocity  $\mathbf{v}_f$   
25 and normal  $\mathbf{n}_f$  to nearby grid nodes based on B-Spline weights (lines 5-11). This yields the extended  
26 velocity  $\mathbf{v}_i^c$  and normal  $\mathbf{n}_i^c$  of the collider at grid node  $i$ . Then, given the velocity of the simulating  
27 object  $\mathbf{v}_i$  at grid node  $i$ , if the vector  $\mathbf{v}_i - \mathbf{v}_i^c$  points inward, we project out the normal component —  
28 keeping only the tangential part — to model collision (lines 13–24). Note that this entire procedure  
29 runs in  $O(N_f)$  time, as collision checks become simple B-Spline weight lookups rather than costly  
30 level-set queries at all grid nodes.

31

**Algorithm 1** Collision Handling

---

```

1: for each  $i$  in a set of grid node indices do
2:    $\mathbf{v}_i^c, \mathbf{n}_i^c \leftarrow \mathbf{0}, \mathbf{0}$ ; ▷ Initialize the zero-value grids for collider
3: end for
4:
5: for each  $f$  in a set of collider mesh faces do
6:   for  $i$  in a set of neighboring grid nodes of  $\mathbf{x}_f$  do
7:      $w_{if}^c \leftarrow \text{Bspline}(\mathbf{x}_f, \mathbf{x}_i)$ ; ▷ Compute interpolation weights using B-spline kernel
8:      $\mathbf{v}_i^c \leftarrow \mathbf{v}_i^c + w_{if}^c \mathbf{v}_f$ ;
9:      $\mathbf{n}_i^c \leftarrow \mathbf{n}_i^c + w_{if}^c \mathbf{n}_f$ ;
10:   end for
11: end for
12:
13: for each  $i$  in a set of grid node indices do
14:    $w_i^c \leftarrow \sum_f w_{if}^c$ 
15:   if  $w_i^c > 0$  then ▷ Detect whether a collision has occurred
16:      $\mathbf{v}_i^c \leftarrow \mathbf{v}_i^c / w_i^c$ ;
17:      $\mathbf{n}_i^c \leftarrow \mathbf{n}_i^c / \|\mathbf{n}_i^c\|$ ;
18:      $\mathbf{v}_i^{\text{rel}} \leftarrow \mathbf{v}_i - \mathbf{v}_i^c$ ; ▷ Transform velocities into the collider's reference frame
19:     if  $\mathbf{v}_i^{\text{rel}} \cdot \mathbf{n}_i^c < 0$  then ▷ Check if the relative velocity points inward toward the collider
20:        $\mathbf{v}_i^{\text{rel}} \leftarrow \mathbf{v}_i^{\text{rel}} - (\mathbf{v}_i^{\text{rel}} \cdot \mathbf{n}_i^c) \mathbf{n}_i^c$ ; ▷ Project relative velocity onto the collider's tangent space
21:     end if
22:      $\mathbf{v}_i \leftarrow \mathbf{v}_i^{\text{rel}} + \mathbf{v}_i^c$  ▷ Transform velocities back into world frame
23:   end if
24: end for

```

---

33 **Physical parameters learning.** As discussed in Sec. 4.3.1 in the paper, we optimize Young's  
34 modulus  $E$ , density  $\rho$ , and rest geometry parameter  $\alpha$  by simulating the first-frame canonical  
35 mesh  $\mathcal{M}_1 = (\mathbf{V}_1, \mathbf{F})$  and minimizing the vertex-wise  $L_2$  error with respect to the tracked meshes  
36  $(\mathcal{M}_i)_{i=2, \dots, T}$ , where  $\mathcal{M}_i = (\mathbf{V}_i, \mathbf{F})$ .

37 In particular, let the initial mesh vertices be  $\hat{\mathbf{V}}_1 = \mathbf{V}_1$ . Then, for each subsequent frame, we obtain  
38 the simulated mesh vertices via

$$\hat{\mathbf{V}}_{i+1} = \text{MPM}(\hat{\mathbf{V}}_i, \mathcal{P}), \quad (1)$$

39 where  $\mathcal{P} = (E, \nu, \gamma, \kappa, \rho, \alpha)$  are our physical parameters. We then perform gradient-based optimiza-  
40 tion for  $\rho$ ,  $E$  and  $\alpha$ , such that they minimize the loss

$$\mathcal{L}_{\text{phys}}(\mathcal{P}) = \sum_{i=2}^T \|\hat{\mathbf{V}}_i - \mathbf{V}_i\|^2. \quad (2)$$

41 Here, each parameter's gradient is approximated using finite differences – following PhysAvatar [10].  
42 For example, the gradient with respect to density  $\rho$  is computed as

$$\frac{\partial \mathcal{L}_{\text{phys}}}{\partial \rho} \approx (\mathcal{L}_{\text{phys}}(E, \nu, \gamma, \kappa, \rho + \Delta\rho, \alpha) - \mathcal{L}_{\text{phys}}(E, \nu, \gamma, \kappa, \rho, \alpha)) / \Delta\rho, \quad (3)$$

43 where  $\Delta\rho$  is the perturbation size.

44 **Training details.** Our MPM simulation uses a time step of  $\Delta t = 0.04$  with  $N = 400$  substeps and  
45 a grid resolution of 200. We optimize the physical parameters over 200 iterations using the Adam  
46 optimizer. For finite-difference gradient estimation, the perturbation sizes are set to  $\Delta\rho = 0.05$ ,  
47  $\Delta E = 5$ , and  $\Delta\alpha = 0.005$ . The corresponding learning rates are 0.01 for  $\rho$ , 0.3 for  $E$ , and 0.01 for  
48  $\alpha$ . All parameters are initialized as  $\rho = 1.0$ ,  $E = 100$ , and  $\alpha = 1.0$  for physical parameter learning,  
49 while  $\nu$ ,  $\gamma$ , and  $\kappa$  are fixed at their default values of 0.3, 500, and 500, respectively.

50 **Simulation time and computing resource.** As noted in the main paper (Sec. 5.2), our simulation  
51 runs at approximately 1.1 seconds per frame on a single NVIDIA GeForce RTX 4090.

## 52 B.2 Appearance Learning

53 For the appearance learning (Sec. 4.3.2 in the paper), we leverage the dense temporal correspondences  
 54 from the tracked meshes  $(\mathcal{M}_i)_{i=1\dots T}$  to transform the canonical Gaussians  $\mathcal{G}$  (defined in the first  
 55 frame) into each subsequent frame. Following [6], we compute the transformations that carries every  
 56 Gaussian from its parent triangle in the canonical mesh to the corresponding triangle in the target  
 57 frame. This allows us to render all training frames  $[1, \dots, T]$  using a single shared appearance model  
 58  $\mathcal{G}$ , such that it can be learned jointly from all input views and frames by minimizing:

$$\mathcal{L}_{\text{app}} = \mathcal{L}_{\text{rgb}} + \lambda_p \mathcal{L}_{\text{position}} + \lambda_s \mathcal{L}_{\text{scaling}}. \quad (4)$$

59 Here,  $\mathcal{L}_{\text{rgb}}$  measures the photometric discrepancy between the rendered and the ground truth images  
 60 over all frames and views, and is defined as a weighted sum of L1 loss  $\mathcal{L}_1$ , SSIM [8] loss  $\mathcal{L}_{\text{SSIM}}$ , and  
 61 LPIPS [9] loss  $\mathcal{L}_{\text{LPIPS}}$ :

$$\mathcal{L}_{\text{rgb}} = \lambda_1 \mathcal{L}_1 + \lambda_{\text{SSIM}} \mathcal{L}_{\text{SSIM}} + \lambda_{\text{LPIPS}} \mathcal{L}_{\text{LPIPS}}. \quad (5)$$

62 In Eq. 4,  $\mathcal{L}_{\text{position}} = \|\max(\mu, \epsilon_p)\|_2$  and  $\mathcal{L}_{\text{scaling}} = \|\max(s, \epsilon_s)\|_2$  regularize the location offset  $\mu$   
 63 and the scale  $s$  of each Gaussian not to exceed the thresholds  $\epsilon_p = 1.0$  and  $\epsilon_s = 0.6$ . This is to  
 64 encourage the Gaussians to remain closely aligned with their parent triangle structures.

65 When optimizing the parameters of  $\mathcal{G}$ , we follow the standard 3DGS optimization procedure [4]  
 66 and employ adaptive density control to increase the number of Gaussians in regions with high  
 67 reconstruction error. For the loss weighting hyperparameters, we set  $\lambda_1 = 0.8$ ,  $\lambda_{\text{SSIM}} = 0.2$ ,  
 68  $\lambda_{\text{LPIPS}} = 0.2$ ,  $\lambda_p = 1.0$ , and  $\lambda_s = 1.0$ .

## 69 C Societal Impact

70 Our method enables physically accurate dynamic human avatar reconstruction from multi-view  
 71 videos, supporting a wide range of applications in virtual reality, digital fashion, and entertainment.  
 72 However, the capability to generate lifelike avatars also introduces potential risks, such as the misuse  
 73 of the technology for creating deepfakes or other forms of deceptive content. When publishing  
 74 our code, we will consider embedding traceable digital watermarks or developing authentication  
 75 mechanisms to ensure the responsible use of generated avatars.

## 76 References

- 77 [1] Mustafa Işık, Martin Rünz, Markos Georgopoulos, Taras Khakhulin, Jonathan Starck, Lourdes Agapito,  
 78 and Matthias Nießner. Humanrf: High-fidelity neural radiance fields for humans in motion. *ACM TOG*,  
 79 42(4):1–12, 2023.
- 80 [2] Chenfanfu Jiang, Theodore Gast, and Joseph Teran. Anisotropic elastoplasticity for cloth, knit and hair  
 81 frictional contact. *ACM TOG*, 36(4):1–14, 2017.
- 82 [3] Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. The material point  
 83 method for simulating continuum materials. In *Acm siggraph 2016 courses*. 2016.
- 84 [4] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for  
 85 real-time radiance field rendering. *ACM TOG*, 42(4):139–1, 2023.
- 86 [5] Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. Amass:  
 87 Archive of motion capture as surface shapes. In *Proceedings of the IEEE/CVF international conference on*  
 88 *computer vision*, pages 5442–5451, 2019.
- 89 [6] Shenhan Qian, Tobias Kirschstein, Liam Schoneveld, Davide Davoli, Simon Giebenhain, and Matthias  
 90 Nießner. Gaussianavatars: Photorealistic head avatars with rigged 3d gaussians. In *CVPR*, pages 20299–  
 91 20309, 2024.
- 92 [7] Wenbo Wang, Hsuan-I Ho, Chen Guo, Boxiang Rong, Artur Grigorev, Jie Song, Juan Jose Zarate, and  
 93 Otmar Hilliges. 4d-dress: A 4d dataset of real-world human clothing with semantic annotations. In *CVPR*,  
 94 pages 550–560, 2024.

- 95 [8] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error  
96 visibility to structural similarity. *TIP*, 13(4):600–612, 2004.
- 97 [9] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable  
98 effectiveness of deep features as a perceptual metric. In *CVPR*, pages 586–595, 2018.
- 99 [10] Yang Zheng, Qingqing Zhao, Guandao Yang, Wang Yifan, Donglai Xiang, Florian Dubost, Dmitry Lagun,  
100 Thabo Beeler, Federico Tombari, Leonidas Guibas, et al. Physavatar: Learning the physics of dressed 3d  
101 avatars from visual observations. In *ECCV*, pages 262–284. Springer, 2024.