

BIGS: Bimanual Category-agnostic Interaction Reconstruction from Monocular Videos via 3D Gaussian Splatting

Supplementary Material

In this supplemental, we included implementation details, pre-processing, additional quantitative results, additional qualitative results, and ablation study.

A. Implementation Details

Optimization details. For ‘single-subject optimization step’, we optimize the hand Gaussians for 15,000 iterations, and optimize the object Gaussians for 30,000 iterations. For ‘interacting-subjects optimization step’, we optimize parameters for additional 30,000 iterations. Based on the NVIDIA A6000 GPU, it takes 5 hours for ‘single-subject optimization’ and 3 hours for ‘interacting-subjects optimization’, respectively. We use the Adam optimizer for all the optimization and the learning rate is set as 1.6×10^{-4} for μ_i and set as 1.0×10^{-4} for all other \mathbf{P} . For both MLPs and TriplaneNets, we use the learning rate of 1.0×10^{-3} .

Densification details. Shapes of Gaussians often become irregular when fitting 3D Gaussians and this can lead to an uneven image quality: certain regions are overrepresented; while others lack sufficient detail. To address the issue, we further employed the adaptive density control (ADC) [4] strategy. The ADC monitors the local density in 3D space and adaptively prunes or splits Gaussians to maintain a balanced representation. This mechanism ensures that regions with complex geometry receive higher details through denser Gaussian placement; while simpler areas are represented more sparsely. We applied the ADC after 3,000 iterations and used it until 27,000 iterations. Also, we limit the number of total Gaussians as 2.0×10^5 .

B. Pre-processing

In this section, we provide a detailed description of the pre-processing stage in our pipeline. The overall process consists of four main steps: hand pose estimation, object/camera pose estimation, hand-object alignment, and signed distance function-based mesh initialization. In the following section, we provide a more detailed explanation of each step.

Hand pose estimation. As mentioned in Sec. 3.2 of the main paper, we first obtain the poses of both hands (*i.e.*, $\Phi_L, \Phi_R, \theta_L, \theta_R, \Gamma_L, \Gamma_R$) using HaMeR [7]. As bimanual hand-object interactions often involve severe occlusions—either caused by the object itself or by the other hand—the pose estimator occasionally fails to produce valid predictions, resulting in missing frames. To address this issue, we apply linear interpolation (LERP) using the neighboring frames to fill in the missing ones. Since Φ_L, Φ_R, θ_L , and θ_R represent joint rotations, using LERP—which considers only linear

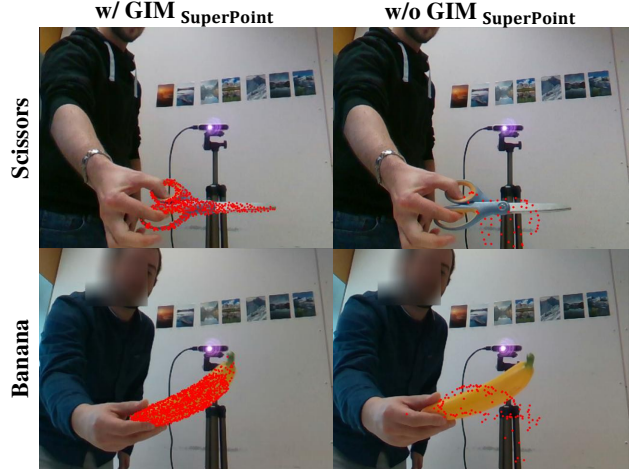


Figure A. Qualitative examples for image matcher. The points describe the initialized object point cloud locations.

relationships—can lead to interpolation errors. To mitigate this, we specifically apply spherical linear interpolation (SLERP) for the missing Φ and θ values, as it captures the geometry of rotational space and yields more accurate results.

Object/camera pose estimation. To obtain the object pose (*i.e.*, Φ_O and Γ_O) and the camera pose \mathbf{C} , which consists of extrinsic matrix \mathbf{R} and intrinsic matrix \mathbf{K} (*i.e.*, $\mathbf{C} = \{\mathbf{R}, \mathbf{K}\}$), we use HLoc [10]. Before applying the HLoc, we apply the SuperPoint [2] and SuperGlue [11] as feature extractor and image matcher, respectively, to obtain the 3D object point cloud. However, since SuperGlue is designed for large-scale scene matching, it often fails to perform reliably on object-only images with no background. The issue becomes more severe in cases where the object is small, textureless, or frequently rotated (as in HO3Dv3’s BB12, BB13, GSF12 and GSF13 sequences). In such cases, SuperGlue struggles with image matching, which in turn results in a highly sparse object point cloud and a large number of missing frames. To address the limitations of conventional image matchers, we adopt GIM [13], a framework specifically designed to enhance image matching robustness across diverse domains. GIM improves the performance via self-training on internet videos, enabling better generalization beyond the training domain. By leveraging the pre-trained $\text{GIM}_{\text{SuperGlue}}$ as our image matcher, we are able to obtain reasonable object and camera poses even from object-only images with no background. However, we empirically observed that initializing Gaussians from the point cloud obtained via GIM leads to an excessive number of Gaussians during training,

leading to the inefficiency in fitting and rendering phases. To avoid this, we apply GIM only when the number of initial SfM-based point clouds falls below a certain threshold (*i.e.*, 100). For qualitative examples regarding this, see Fig. A.

Since SfM fails to estimate valid object poses and produces noise 3D reconstructions in some frames, we apply linear interpolation (LERP) to Γ_O and spherical linear interpolation (SLERP) to Φ_O . We follow the strategy proposed in HOLD [3] for this: we first compute the center and overall diameter of the object point cloud. Then, we calculate the distance corresponding to the 20th percentile from the center, and all points exceeding 1.5 times this distance are filtered out. Since sparse point cloud can negatively impact on the final performance, if the number of initial point cloud is below 10^4 , we perform upsampling until the point count exceeds the threshold. Here, upsampling means generating new points via linear interpolation between randomly selected point pairs.

As the final step, we define the initial canonical object Gaussian by shifting the centroid of the processed object point cloud to the origin.

Hand-object alignment. Poses of two hands and an object have been obtained in previous stage, while there exist a few issues: First, the translations of the hands, Γ_L and Γ_R , are located in the MANO [9] space, which is not aligned with camera poses c obtained from HLoc [10]. Secondly, as SfM relies only on image correspondences to reconstruct the 3D geometry, it loses information about real-world scale (*i.e.*, absolute scale s). To address these issues, we perform a simple optimization step that aligns Γ_L and Γ_R with the HLoc camera C, following the initialization step of HOLD [3] (written in Sec. A of their supplemental). The only difference is that we have both hands; while HOLD only tackles the one hand case (*i.e.*, right hand). We extend the formula by additionally involving the left hand. After the optimization, we can get hand-object aligned parameters (*i.e.*, $\Gamma_L, \Gamma_R, \Gamma_O, s$).

Signed distance function-based mesh initialization. To obtain the final meshes of both the hand and the object, given point clouds, we apply three implicit functions (f_L , f_R and f_O) [1] which are trained for left hand, right hand and objects. Specifically, we cast a ray from each pixel of the input image via the obtained camera C and sample 3D points in the camera space. These sampled points are then mapped to the hand-object aligned space using pre-processed initial poses (*i.e.*, $\theta_L, \theta_R, \Phi_L, \Phi_R, \Phi_O, \Gamma_L, \Gamma_R, \Gamma_O, s$). These points are then used as inputs to the implicit function, which predicts their signed distance and color values. Then, we apply the Marching Cubes algorithm [6] to their respective learned implicit fields, f_L , f_R , and f_O . For each, we evaluate the signed distance function over a uniform 3D grid within a pre-defined bounding volume in the canonical space, and extract the zero level set ($d = 0$) as the surface mesh. The extracted mesh vertices are then used to initialize the 3D means (*i.e.*, μ_i) of our Gaussian representations.

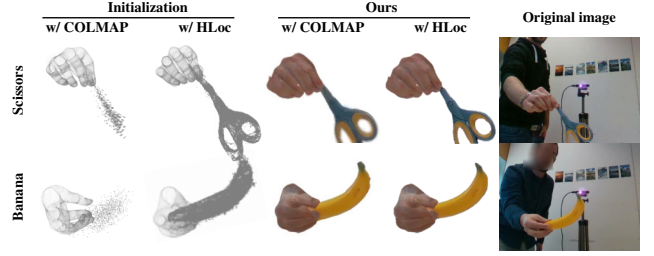


Figure B. Ablation study on SfM/hand initialization.



Figure C. Results on challenging objects.

C. Additional Quantitative Results

Fitting and inference speed comparison. During model fitting, ‘HOLD’ takes 30 hours; while ‘Ours’ takes around 1.5 hours on NVIDIA RTX3090 GPU. During inference, ‘HOLD’ runs with 0.1 FPS; while ‘Ours’ runs with 42.4 FPS. This is more efficient than ‘HOISDF’, whose inference speed is 30.7 FPS.

Quantitative results on HO3Dv2 dataset. Throughout the main paper, we used the HO3Dv3 dataset. Tab. A presents quantitative results on the HO3Dv2 dataset, comparing our method with prior works HOLD [3] and HOISDF [8]. We evaluate performance using CD_o and MPJPE, where lower values indicate better accuracy. Our method achieves the best performance on both metrics, with a CD_o of 0.37 cm² and an MPJPE of 23.68 mm, outperforming HOLD (0.39 / 24.47) and HOISDF (0.39 / 23.73). We run their public code to get the results. The evaluation was conducted across 13 HO3Dv2’s evaluation sequences (*i.e.*, SM2, SM4, MC1, MC4, ShSu10, GPMF12, GPMF14, SMu40, SMu1, ABF12, ABF14, MDF12, MDF14), demonstrating the robustness and accuracy of our approach in diverse hand-object interaction scenarios.

D. Additional Qualitative Results

Novel pose rendering. As explained in Fig. 1 of the main paper, we build the BIGS (Bimanual Interaction 3D Gaussian Splatting) and this can be used to animate hand-object interactions with novel poses of hands, objects and camera. Fig. F and Fig. G show the example rendered images of 2 objects (capsule machine and mixer), each with 4 different hand and object poses, rendered from 8 different camera viewpoints. Each column denotes 8 different hand and object interaction cases and each row denotes 8 different camera viewpoints. Using the optimized 3D hand and object Gaussians, we can reliably animate hand-object animations for novel poses of

	CD _o (cm ²) ↓	MPJPE (mm) ↓
HOLD [3]	0.39	24.47
HOISDF [8]	0.39	23.73
Ours	0.37	23.68

Table A. Quantitative results on HO3Dv2 evaluation set.

	Rendering		
	PSNR ↑	SSIM ↑	LPIPS ↓
Ours w/o <i>share</i> *	19.22	0.61	0.21
Ours w/o \mathcal{L}_{SDS}	23.81	0.76	0.13
Ours w/o \mathcal{L}_{mask}	16.84	0.48	0.28
Ours*	24.87	0.96	0.05

Table B. **Ablation study on loss functions and architecture.** ‘*share*’ denotes the Gaussian sharing scheme for two hands. Without *share**, we build two Gaussians for each hand; while with *share**, we build one Gaussian for a right hand and share it across two hands.

hand, object and camera. We further made the supplementary video for the novel pose rendering sequences.

Results on challenging objects. We visualized results on challenging samples (*i.e.*, banana, scissor) of HO3Dv3 in Fig. B and also results on the in-the-wild transparent object in Fig. C. In Fig. C, textures are unrealistic due to transparency in the ‘Novel view’.

E. Ablation Study

Ablation study on SfM/hand initialization. We conducted ablations on different SfMs and hand reconstructors: When initializing objects with ‘COLMAP’ [12] and ‘HLoc’ [10], their initial CD_o (↓) is 80.4 and 57.1, respectively; while respective final result is 1.82 and 1.25, which are already better than ‘HOLD [3]’ (2.04). When initializing hands with ‘METRO’ [5] and ‘HaMeR [7]’, their initial MPJPE (↓) is 21.58 and 21.57, respectively, which result in similar final results. We also interpolate the object vertices until it has over 10^4 vertices from init vertices, if the number of vertices is too small. We provide the qualitative examples for small and textureless objects like scissors and bananas using different initializations in Fig. B: With incomplete initializations, ‘Ours’ reliably produce final results.

Ablation results for the image quality. In Table B, we additionally showed the ablation study results for the image quality (measured in PSNR, SSIM, and LPIPS) depending on the different configurations of the loss functions and architecture. We involved four baselines of ours: ‘Ours’, ‘Ours w/o \mathcal{L}_{mask} ’, ‘Ours w/o \mathcal{L}_{SDS} ’ and ‘Ours w/o *share**’. ‘Ours’ is our full model using the full loss functions, ‘Ours w/o \mathcal{L}_{mask} ’ is our model trained without the use of the mask loss \mathcal{L}_{mask} , ‘Ours w/o \mathcal{L}_{SDS} ’

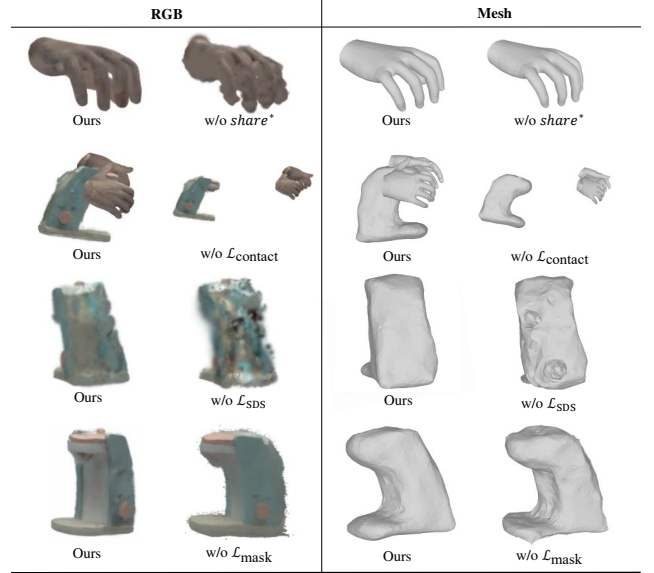


Figure D. Qualitative examples for ablation study.

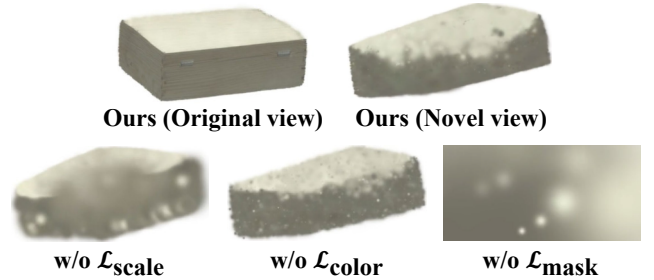


Figure E. More detailed ablation results.

is our model trained without the use of the SDS loss \mathcal{L}_{SDS} . ‘Ours w/o *share**’ is our model using two Gaussians for each hand (without using the hand Gaussian sharing scheme). We can observe that without \mathcal{L}_{mask} , the rendering quality becomes significantly inferior to the original baseline, since Gaussians are generated in non-object regions. Also, we could observe that the sharing scheme contributes a lot for the rendered image quality. Especially, when this scheme is not involved, the occluded hands are not properly learnt. The SDS loss \mathcal{L}_{SDS} also affect the rendering quality; while we did not include the experiments regarding $\mathcal{L}_{contact}$ and \mathcal{L}_{smt} losses, since their effect is not significant in the rendering quality (cf. \mathcal{L}_{smt} is the part of hand loss \mathcal{L}_{hand} : $\mathcal{L}_{smt}^t = \|\theta^t - \theta^{t-1}\|_2^2 + \sum_{H \in \{L, R\}} \|\Gamma_H^t - \Gamma_H^{t-1}\|_2^2$).

Fig. D further visualizes the qualitative results for the ablation studies. Especially, we further showed cases for our results with and without the *share** scheme, our results with and without the contact loss $\mathcal{L}_{contact}$, our results with and without the SDS loss \mathcal{L}_{SDS} and our results with and without the mask loss \mathcal{L}_{mask} , respectively. We observe that the rendering results significantly reduced when the *share** scheme is not used. We also observe that

‘Ours w/o \mathcal{L}_{SDS} ’ exhibits noised results in object parts since ours cannot reliably reconstruct object Gaussians for unseen viewpoints, however with the SDS loss, ours can reconstruct the missing pixels. We also observe that ‘Ours w/o $\mathcal{L}_{\text{mask}}$ ’ shows jittered pixels around the objects; while when using the mask loss $\mathcal{L}_{\text{mask}}$, ours can reduce such pixel jitterings. Also, we can observe that without the contact loss $\mathcal{L}_{\text{contact}}$, 3D locations between hands and objects become implausible especially for unseen views. However, when using the contact loss, we observe that hands and objects become tightly contacted even in the unseen viewpoints. **More detailed ablation results.** In Fig. E, we present more ablations on losses: Without $\mathcal{L}_{\text{scale}}$, images could become blurred since the scale of a certain Gaussian could dominate. Without $\mathcal{L}_{\text{color}}$, novel surface’s color could become irregular. $\mathcal{L}_{\text{mask}}$ ensures the stable training of 3DGS using foreground masks.

References

- [1] Bimanual category-agnostic reconstruction. Available at: <https://github.com/zc-alexfan/hold/blob/master/docs/arctic.md>. 2
- [2] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPRw*, 2018. 1
- [3] Zicong Fan, Maria Parelli, Maria Eleni Kadoglou, Muhammed Kocabas, Xu Chen, Michael J Black, and Otmar Hilliges. HOLD: Category-agnostic 3D reconstruction of interacting hands and objects from video. In *CVPR*, 2024. 2, 3
- [4] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM TOG*, 2023. 1
- [5] Kevin Lin, Lijuan Wang, and Zicheng Liu. End-to-end human pose and mesh reconstruction with transformers. In *CVPR*, 2021. 3
- [6] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *SIGGRAPH*, 1987. 2
- [7] Georgios Pavlakos, Dandan Shan, Ilija Radosavovic, Angjoo Kanazawa, David Fouhey, and Jitendra Malik. Reconstructing Hands in 3D with Transformers. In *CVPR*, 2024. 1, 3
- [8] Haozhe Qi, Chen Zhao, Mathieu Salzmann, and Alexander Mathis. HOISDF: Constraining 3D Hand-Object Pose Estimation with Global Signed Distance Fields. In *CVPR*, 2024. 2, 3
- [9] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied Hands: Modeling and capturing hands and bodies together. *ACM TOG*, 2017. 2
- [10] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From Coarse to Fine: Robust Hierarchical Localization at Large Scale. In *CVPR*, 2019. 1, 2, 3
- [11] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning Feature Matching with Graph Neural Networks. In *CVPR*, 2020. 1
- [12] Johannes L. Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 3
- [13] Xuelun Shen, Zhipeng Cai, Wei Yin, Matthias Müller, Zijun Li, Kaixuan Wang, Xiaozhi Chen, and Cheng Wang. GIM: Learning Generalizable Image Matcher From Internet Videos. In *ICLR*, 2024. 1

































cam 0				
cam 1				
cam 2				
cam 3				
cam 4				
cam 5				
cam 6				
cam 7				

Figure F. Examples for animatable hand-object sequence rendering for novel poses of hand, object and camera.

































cam 0				
cam 1				
cam 2				
cam 3				
cam 4				
cam 5				
cam 6				
cam 7				

Figure G. Examples for animatable hand-object sequence rendering for novel poses of hand, object and camera.