Robo-GS: A Physics Consistent Spatial-Temporal Model for Robotic Arm with Hybrid Representation

Haozhe Lou^{1*}, Yurong Liu^{2*}, Yike Pan^{3*}, Yiran Geng^{4*}, Jianteng Chen^{5*}, Wenlong Ma⁶, Chenglong Li⁶, Lin Wang⁶, Hengzhen Feng⁶, Lu Shi⁹, Liyi Luo⁸, Yongliang Shi^{†7}

Abstract—The Real2Sim2Real (R2S2R) paradigm is critical for advancing robotic learning. Existing methods lack a comprehensive solution to accurately reconstruct real-world objects with both spatial representations and their associated physics attributes in the Real2Sim stage.

We propose a Real2Sim pipeline to generate digital assets enabling high-fidelity simulation. We design a hybrid representation model that integrates mesh geometry, 3D Gaussian kernels, and physics attributes to enhance the representation of robotic arms in digital assets. This hybrid representation is implemented through a Gaussian-Mesh-Pixel binding technique, which establishes an isomorphic mapping between mesh vertices and the Gaussian model. This enables a fully differentiable rendering pipeline that can be optimized through numerical solvers, achieves high-fidelity rendering via Gaussian Splatting, and facilitates physically plausible simulation of the robotic arm's interaction with its environment through mesh geometry. With the digital assets, we propose a fully manipulable Real2Sim pipeline that standardizes coordinate systems and scales, ensuring the seamless integration of multiple components. To demonstrate its effectiveness, we include datasets covering various robotic manipulation tasks with their mesh reconstructions. Our model achieves state-of-the-art results in realistic rendering and mesh reconstruction quality for robotic applications. Our code and datasets will be made publicly available at robostudioapp.com.

I. INTRODUCTION

Real2Sim2Real plays a critical role in robotic arm control and reinforcement learning but remains a challenging task due to the complex physical properties of robots and the objects they manipulate. It tackles well-known problems of existing learning frameworks in robot autonomy such as model generalization ability and label availability [1]. Still, it has yet to be fully realized due to significant challenges in spatial and color representation as well as rendering quality within current Real2Sim approaches [2]. These challenges hinder transferring learned policies from simulation to real-world applications and compromise the reliability and performance of robotic systems trained in simulated environments.

In this paper, we target the Real2Sim and demonstrate a **holistic reconstruction** of robotic arm operation scenes, which requires a (i) manipulable robot model, (ii) reconstructed background and objects, (iii) physical parameters such as mass and friction. Our approach employs Unified Robot Description Format(URDF, which is a standard XML

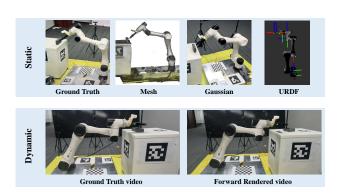


Fig. 1. The reconstructed digital assets include the extracted mesh, Gaussian Models [3], a dynamically consistent kinematic model.

format used to describe the physical configuration, joints, and kinematic structure of robots) [4], [5], as a spatial representation and integrates governing equations with physics parameters as the forward deformation mechanism. This combination enables accurate collision detection and consistent rendering across both simulation and Gaussian Splatting (GS) environments [6], [7].

The core to our method is a **Gaussian-Mesh-Pixel binding**, which establishes an isomorphic relationship between mesh vertices, Gaussian kernels [3], and image pixels. Each Gaussian is assigned a semantic label and a corresponding ID, enabling the precise application of transformation matrices governed by the URDF. This ensures a seamless transfer of trajectories between real-world video, simulation results, and rendered images. The advantages of this binding include (i) End-to-end differentiable gradient passing between each representation, allowing backward optimization to be performed across different backends; (ii) Superior collision detection through our state-of-the-art mesh reconstruction; and (iii) High rendering quality.

Our system ensures a faithful Real2Sim process, allowing learned policies to be effectively deployed in real-world scenarios (Please see the supplementary video). Additionally, it supports editing within the robotic simulator Isaac Sim (Gym) [7], [8] backend, enabling novel-pose and novel-policy adjustments. Optimized for robotic arms in CR3, CR5 [9] and UR5 [10] product sequences, our method is versatile enough to generalize to other robotic arm models. Our method achieves state-of-the-art performance in mesh reconstruction and dynamic rendering compared to current methods [11].

Furthermore, we propose a new format for digital assets, represented by a combination of mesh, Gaussian Splatting,

^{*}Equal contribution, †Corresponding author.

¹University of Southern California, ²National University of Singapore, ³University of Michigan, Ann Arber, ⁴ Peking University, ⁵Hong Kong University of Science and Technology, ⁶Beijing Institute of Technology, ⁷Tsinghua University, ⁸Xiaomi Robotics Lab, ⁹AIR, Tsinghua University.

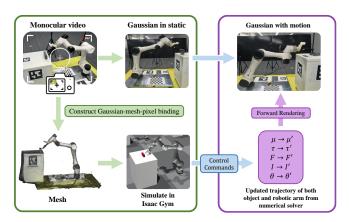


Fig. 2. (Left) Our method converts monocular video to 3D meshes and Gaussians, URDF models, and generates trajectories through Gaussian-Mesh-Pixel binding, and (Right) renders dynamic interactions using forward deformation based on updated object and robotic arm trajectories.

and real-world motion [12], as shown in Fig.1. This approach surpasses traditional textured mesh and material attributes by integrating critical physics parameters, such as mass and friction, extracted from real-world motion videos [13].

- We introduce a holistic framework for robotic arm control and simulation, combining mesh geometry, 3D Gaussian kernels, and physics attributes to achieve highfidelity scene reconstruction and physically accurate simulation of robotic manipulation tasks. Our method, based on a Gaussian-Mesh-Pixel binding technique, ensures seamless integration between simulation, rendering, and real-world data, allowing for precise policy training and control optimization in complex environments.
- We address the challenges of accurately reconstructing robotic arm movements, including occlusions and inconsistencies between simulated and real-world data.
- Experiments and ablations demonstrate the effectiveness of our method. It achieves state-of-the-art performance in mesh reconstruction and high-quality rendering, significantly outperforming previous approaches in both static and dynamic scenarios.

II. RELATED WORK

Rendering: The Real2Sim2Real paradigm, crucial for bridging physical and simulated environments, relies on advanced rendering techniques. Recent innovations in 3D reconstruction, such as Gaussian Splatting and Neural Radiance Fields (NeRF) [3], [14]–[17], have significantly enhanced the creation of high-fidelity digital twins, essential for accurate simulations.

Robotic simulation has progressed with the use of transformation matrices and Linear Blend Skinning (LBS) [18] control modes. However, these methods often struggle with maintaining geometric consistency in robotic arms due to rigid mesh motions. SC-GS [19] addresses this by employing sparse control points, though it falls short in accurately simulating robotic arm movements.

Further advancements include the use of NeRFs and Gaussian Splatting for reconstructing robotic operation

scenes [11], [20], [21]. MD-Splatting [22] explores Gaussian-based reconstruction of highly deformable objects, utilizing NeRF and particle-based representations. However, these methods often lack high-fidelity rendering capabilities [23]. Our approach enhances 4D Gaussian Splatting quality by replacing traditional MLP-based deformation fields [19], [24], [25] with numerical ODE solvers [1], [26].

Asset Extraction: Significant progress in mesh reconstruction and simulation integration has been made with NeRFmeshing [27] and SUGAR [28], [29]. NeRFmeshing utilizes NeRFs for mesh reconstruction, while SUGAR employs Gaussian Splatting for mesh extraction. However, the complex and sparse nature of robotic arm structures often leads to non-smooth reconstructions due to challenges in normal estimation. The 2DGS method [30] effectively addresses non-smooth surface issues by introducing a planar-like 2DGS representation. After applying cleaning and matching techniques, the quality of meshes generated by 2DGS proves sufficient for robotic simulation purposes.

Control: In the Real2Sim2Real paradigm, robotic arm control typically employs pose-based methods, transferring end-effector pose trajectories [31] between real-world and simulation policies. This approach uses inverse kinematics to derive pre-joint control information [32]. Current control policies include diffusion-based models [33] and trajectoryto-video generation methods like IRASim [34], which utilize generative methods driven by trajectory for simulation [35] and representation. However, these methods are limited in fully representing 3D and 4D real-world scenarios [36], [37]. Traditional robotic arm control connects linkages with predefined joints, using angles for path control [38]-[40]. In a Gaussian setting, where each Gaussian is treated as an independent element [41], this approach can lead to motion inconsistencies. Our method addresses this issue through isomorphism Mesh-Gaussian binding.

Recent innovations include a signed distance field-based approach for robotic arm morphology [42], using joint angles as input with a neural decoder guiding the deformation field. However, this method lacks explicit compatibility with the MOVEIT path-based control pipeline [43] and current reinforcement training toolkits [38], [44] UMI on Legs demonstrates the potential of simulation-based manipulation tasks, though scene and object reconstruction remains a challenge [45].

Physically embodied Gaussian Splatting [46] employs Gaussian distributions and particles derived from pre-built meshes to generate movement [47], tested on simulated data. Our approach advances this concept by utilizing meshes extracted from video data for more accurate representation, coupled with real-world scenario data collection.

III. METHOD

A. Representation

Our digital asset is represented by a mesh, a set of Gaussian primitives, and the spatial-temporal trajectory of the assets. Traditionally, digital asset production focuses on the textured mesh and material attributes [48]. However,

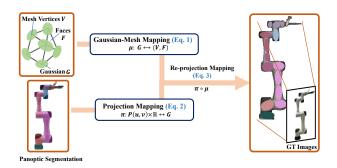


Fig. 3. Gaussian-Mesh-Pixel binding: This binding preserves the structural properties between Gaussian G, mesh (V,F), and pixel location P under affine transformation, which is a composition (right) a Gaussian-Mesh mapping (top-left), and a projective Pixel-Gaussian binding (top-right).

physical parameters such as mass and friction coefficients are necessary to enable Sim2Real policy training for downstream robotic manipulation tasks. Thus we incorporate these physical parameters in our representation.

A polygon mesh M=(V,F) is defined by a set of vertices $V=\{\nu_1,\nu_2,\ldots,\nu_n\},\nu_i\in\mathbb{R}^3$ and faces $F=\{f_1,f_2,\ldots,f_k\}$. Similar to 3DGS [3], we use a set of 3D Gaussian primitives $\{G\}_{p\in P}$, where each Gaussian primitive G includes a center \mathbf{x}_p , an opacity σ_p , a covariance matrix C_p , a set of spherical harmonic (SH) coefficients SH_p , and a semantic category of the Gaussians S_p , respectively [49].

We here describe the proposed Gaussian-Mesh-Pixel binding (Fig. 3): We first define a Gaussian-Mesh mapping μ that associates each Gaussian primitive with a set of mesh vertices (and faces). Second, we project the Gaussian primitives into 2D via projection mapping, thus establishing an isomorphism between pixel locations P and Gaussian kernels G. By compositing the above mappings, we define an isomorphic binding between Gaussians, mesh, and pixels. Here we define each mapping concretely:

1) **Gaussian-Mesh Mapping:** An order-preserving mapping $\mu : \mathbb{R}^3 \to V$ associates A with a vertex in V:

$$\mu(G(x, y, z)) = \{\nu_n, f_k | \nu_n \in V, f_k \in F\}.$$
 (1)

2) **Projection Mapping:** Given a known camera pose represented in $\mathbb{SE}(3)$ and camera intrinsic, any known 3D point location can be re-projected to the 2D image plane using the perspective projection model:

$$(u,v) = \pi(G(x,y,z)). \tag{2}$$

3) **Re-projection Mapping:** Through the composition of the above mappings, we define an isomorphic relationship ϕ that associates Gaussian G with both pixel location P and vertices V [6]:

$$\phi(G(x,y,z)) = (\pi \circ \mu)(G(x,y,z)). \tag{3}$$

This mapping connects the trajectory between real-world scenes captured on the image plane, simulation results from the mesh-based engine, and rendered scenes [50]–[52]. It unifies the representations of Gaussian Splatting, mesh, and real-world motion, ensuring spatio-temporal consistency and allowing gradients to flow from real-world video to Gaussian

to mesh (backward optimization) and from mesh to Gaussian to rendered video (forward rendering).

B. Mesh extraction

To obtain necessary geometric cues from the video sequences, we first collect a set of static images of the scene and utilize Gaussian Splatting techniques to prepare the meshes. We extract linkage, object, and background mesh through our mesh extraction and cleaning technique. Specifically, we adapt surfel-based Gaussian Splatting [30], along with vertex cleaning and our re-orientation strategy.

We follow 2DGS [30] and represent a Gaussian with a "2D" surfel rather than a full 3D Gaussian kernel, this is characterized by a center vector \mathbf{x}_p , a tangent vector $\mathbf{t}_p = (\mathbf{t}_u, \mathbf{t}_v)$, and scaling $\mathbf{s}_p = (s_u, s_v)$. The normal vector of the Gaussian primitive is thereby defined as $\mathbf{N}_w = \mathbf{t}_u \times \mathbf{t}_v$. A Gaussian surfel is then defined in a local tangent plane as:

$$\mathbf{x}_{n}(u,v) = \mathbf{x}_{n} + s_{u}\mathbf{t}_{u}u + s_{v}\mathbf{t}_{v}v = \mathbf{H}(u,v,1)^{\mathrm{T}}$$
(4)

$$\mathbf{H} = \begin{bmatrix} s_u \mathbf{t}_u & s_v \mathbf{t}_v & \mathbf{0} & \mathbf{p}_k \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{RS} & \mathbf{p}_k \\ 0 & 1 \end{bmatrix}$$
(5)

where $\mathbf{H} \in \mathbb{R}^{4 \times 4}$ is a homogeneous transformation matrix representing the geometry of the Gaussian surfel. For the point $\mathbf{u} = (u,v)$ in uv space, its probability density function can then be expressed as: $\mathcal{G}(\mathbf{u}) = \exp\left(-\frac{u^2+v^2}{2}\right)$.

After the reconstruction process, we can extract a mesh based on the Gaussian kernels using TSDF fusion [53]. However, there are discrepancies between the extracted meshes and the requirements for URDF production.

We identify four major issues and propose solutions for each to achieve alignment between the real-world robotic arm and Gaussian Splatting.

- Define a Unified Coordinate System and Origin:
 A unified coordinate system integrates real-world and simulation scenes using the OpenGL y-up axis, which is consistent in both Gaussian Splatting and the simulation engine. A reference point, such as the base of the robotic arm, must be set as the origin in Gaussian Splatting to ensure a consistent coordinate definition.
- Scale Adjustment: To compensate for discrepancies between the reconstructed scene and the real world—inevitable when reconstructing from monocular videos but essential to address for accurate robotic control and simulation—we manually correct them by establishing a standard based on real-world measurements using MeshLab.
- Orientation Alignment: The reconstructed mesh from Gaussian Splatting may not be axis-aligned due to the settings of Structure-from-Motion (SfM) [54]. To correct this, we use a scene re-orientation technique to align the reconstructed scenes with the simulation engine's coordinate system [17]. This alignment is crucial for accurately reproducing simulated movements and interactions in the real world.

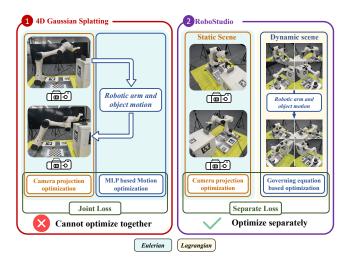


Fig. 4. Comparison between deformable Gaussian Splatting (left) and our solution. (Left) In general deformation-based Gaussian Splatting [11], [19], the dynamics of the scene are embedded in each Gaussian primitive, resulting in a high-dimensional degree of freedom (DoF). This causes failure during the training process in complex dynamic scenes, as observed in our setting. (Right) Our approach decomposes the motion into a small number of linkages and objects driven by governing equations, significantly reducing the degrees of freedom. This allows numerical solvers to optimize more efficiently.

• Generate Physics Parameters: The URDF requires mass, friction, and damping factors for physics simulation. We embed the LLM-inferred physics parameters and governing equation categories into the URDF asset, based on Panoptic information [55], [56].

C. Kinematic Governing Equation

Optimizing 3DGS involves minimizing the re-projection error between the Gaussian representation and pixel data. In 4D Gaussian Splatting, input images include both pixel data and timestamps, aiming to optimize the 4D reconstruction under the XYZT representation. This non-convex problem is challenging due to the lack of multi-view consistency in this setting, as shown in Fig. 4.

We find that the optimal solution to this ill-posed problem in robotic scenes is to decompose the original 4D reconstruction problem into a two-stage process: static and dynamic stages. We treat static scenes using the Eulerian representation and dynamic scenes using the Lagrangian representation [51]. The static stage aims to optimize the camera reprojection loss, while the dynamic stage focuses on simulating the motion of objects and the robotic arm based on their corresponding governing equations. Through our Mesh-Pixel-Gaussian binding approach, we can pass gradients and accurately simulate the results.

In our system, control information is aligned with a predefined governing equation for physics simulation and rendering [57]. However, when simulating these scenes in a Gaussian Splatting framework, the joint angle based trajectory representation are incompatible with the discrete Gaussian setting because the spatial-temporal correspondence between joints and Gaussians cannot be accurately determined [42]. To overcome this, we define a kinematic and dynamic system

tailored for Gaussian-based robotic arm and gripper pose control, focusing on controlling the end-effector's pose and using inverse kinematics to generate control signals for each joint in the real world.

The kinematics of the robotic arm are adapted to Gaussian Splatting using Modified Denavit-Hartenberg (MDH) parameters and coordinate transformations across Gaussian Splatting, simulation, and real-world settings. A point on link i in Cartesian coordinates can be described using the MDH parameters [58], which include link length (a_i) , link twist (α_i) , link offset (d_i) , and joint angle (θ_i) . These parameters define the forward kinematic transformation matrix T_i between consecutive coordinate frames, allowing the transformation from link i-1 to link i.

$$T_{i-1}^{i} = R_x(\theta_{i-1})T_x(a_{i-1})R_z(\alpha_i)T_z(d_i)$$
 (6)

$$T_i = \begin{bmatrix} \cos \theta_i & -\sin \theta_i & 0 & a_i \\ \sin \theta_i \cos \alpha_i & \cos \theta_i \cos \alpha_i & -\sin \alpha_i & -\sin \alpha_i d_i \\ \sin \theta_i \sin \alpha_i & \cos \theta_i \sin \alpha_i & \cos \alpha_i & \cos \alpha_i d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The value of MDH parameter can be supplied from either CAD designing, real-world measuring, or manual values.

In this approach, the mesh serves as an intermediary for mapping movements. The motion of each mesh is transferred to the Gaussian bound to it, guiding the movement of the Gaussian [59], [60]. Unlike traditional inverse kinematic control method [61], which represents trajectories under continuous space, we model the system as a discrete state space. In this model, each timestamp is independent but shares a common coordinate system (Fig. 4). To account for deformation between timestamp 0 and timestamp t, we first map the point t0 and t1 to its base coordinate t2 to its base coordinate t3 to t4 to t5 to t6 to t8 first map the point t9 to t9 to its base coordinate t9 to t9 to

$$\mathbf{P}_{\text{base}} = (\mathbf{T}_0^i)^{-1} \mathbf{P_0} \tag{8}$$

Then, map the point forward to the new frame at time t, $P_t = (x_t, y_t, z_t)$:

$$\mathbf{P}_t = \mathbf{T}_0^i \, \mathbf{P}_{\text{base}} \tag{9}$$

The overall transformation from the initial frame to the new frame at time t_1 is:

$$\mathbf{P}_{t1} = \mathbf{T}_{0_t}^i (\mathbf{T}_0^i)^{-1} \mathbf{P_0} \tag{10}$$

Since the initial position of the robotic arm is the default position, we can simply load the trajectory from the policy and transform it into the mesh and Gaussian kernels form.

D. Dynamic Governing Equation

The dynamics of the rigid asset are governed by the Newton-Euler equations. The governing equation transforms the optimization problem from 6n degrees of freedom (DOF) to 6A DOF, where A is the number of movable parts defined by instance segmentation, and n is the number of

Gaussian [56], [62]. This can simplify the representation and optimization of the numerical solver.

Following [50], [52], we define the Newton-Euler equations as

$$\begin{bmatrix} m\mathbf{I}_{3\times3} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{v}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \boldsymbol{\tau} \end{bmatrix} - \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega} \end{bmatrix}$$
(11)

The simulation of forces, inertial, and torque are performed in a physics simulator based on our generated mesh and inferences parameter. We simply perform the semi-implicit Euler integration from Gradsim [50] for numerical solving. Dynamic governing equation generates the transformation matrix of force based control of robotic arm and the motion of rigid object. The simulation of soft body can follow PACnerf, PhysGaussian and Simplicits [51], [60], [63].

We decompose the Gaussian movement into a global transformation matrix under the supervision of kinematics and dynamics. Unlike the deformation-based Gaussian Splatting [19], [64], which treats movement as an MLP deformation field, our approach explicitly updates the position of Gaussians in each timestamp and is particularly effective for objects with rigid and linked motion.

To render a view, Gaussian Splatting projects these 3D Gaussians onto the image plane. The final color of each pixel is computed as [51], [60]

$$C = \sum_{k \in P} \alpha_k SH(d_k; C_k) \prod_{j=1}^{k-1} (1 - \alpha_j).$$
 (12)

Here, α_k represents the z-depth ordered effective opacity, i.e., products of the Gaussian weights and their overall opacity σ_k ; d_k stands for the view direction from the camera to x_k . We only compute transformation for the k_{th} Gaussians that shares same panoptic mask with this pixel.

For the effect of global transformation matrix T_p^{sam} apply on static scenes, we decompose it into the effect of Rotation and translation:

$$f_t(d) = f_0(R^T d). (13)$$

The equation $f_t(d)$ represents the effect of rotation on the view direction of Gaussian [60]. We define τ to represent the position update of each Gaussian center at a new time t, according to a transformation conditioned on the kinematics and dynamics of the moving part,

$$\tilde{\tau}_p(X, t, S) = R_t^S \cdot x_p^S + T_t^S. \tag{14}$$

where S indicates the instance ID of the moving part.

IV. EXPERIMENT

A. Datasets:

Current datasets [65]–[68] contain only videos, language commands, and robot trajectories. Building digital assets based on these datasets would result in a significant Real2Sim gap [65]. To address this gap, we propose that a comprehensive representation for a Real2Sim dataset should include monocular video with panoptic labeling, bounding

box annotations, physics parameters, mesh geometry, policy, and trajectory information [69], [70], and real-world motion data of both the object and the robotic arm during policy implementation, along with URDF files for CR and UR robotic arm sequences. We find that this dataset format provides all the necessary information for policy training while ensuring millimeter and millisecond-level precision.

B. Rendering Results:

We compare the current state-of-the-art in 4D Gaussian Splatting and 4D-NeRF, including SC-GS [19] and K-Planes [71], with our method. Fig. 5 illustrates the reconstruction of a robotic arm performing a series of trajectories. We observe that both K-Planes and SC-GS struggle to accurately optimize the transformation of the robotic arm and object motion, resulting in poor rendering outcomes. As discussed in section III-D, unlike SC-GS and K-Planes, which optimize the representation across the entire space, our method simplifies the dynamic Gaussian Splatting optimization process by employing Newton-Euler equations, focusing specifically on spatiotemporal variations. This approach makes our rendering more robust to rigid and linkage motion changes, and the results demonstrate our method's ability to handle complex trajectories and motions.

For a dynamic robotic manipulation scene simulator, rendering the manipulated object is equally crucial. Fig. 6 illustrates the reconstruction of a robotic arm pushing a box. As shown, both KPlanes and SC-GS fail to accurately capture the dynamic motion of the robotic arm and the rigid body, consistent with the findings from Robo360 [65]. In contrast, our method successfully preserves motion and geometric consistency during robotic manipulation tasks.

C. Mesh Quality:

We compare our method against SOTA video-based mesh reconstruction approaches, including 2DGS [30], Gaustudio [16], SUGAR [28], as well as ground truth data obtained from a commercial 3D scanner. As shown in Fig.7, compared to SUGAR and Gaustudio, our approach results in better mesh quality with texture. Mean Square Error (MSE) and F-score, with a tolerance of 1e-5, were employed to evaluate reconstruction precision and spatial consistency, respectively. Building on 2DGS for our mesh reconstruction, the surface quality remains largely consistent. However, the suboptimal opacity optimization in 2DGS can lead to the inner shell within the robotic arm's mesh, adversely affecting the quantitative results (Table I). This issue also compromises collision detection, reducing the realism of the simulation. Our meshcleaning technique eliminates redundant meshes generated by TSDF, enhancing alignment and sampling and yielding improved quantitative outcomes. Therefore, our model effectively reconstructs the collision groups of various small modules of the robotic arm in the URDF implementation.

D. Manipulable Robotic Model:

In our approach, the simulation generates the movement of objects and the robotic arm, resulting in simulated trajectories for each object and robotic linkages. These trajectories



Fig. 5. Qualitative result for Novel-pose

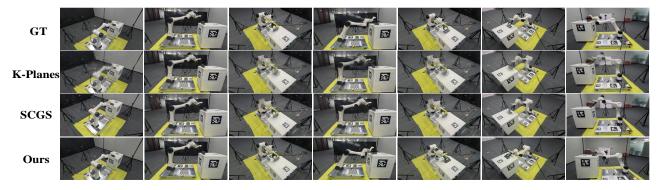


Fig. 6. Qualitative result for Push-box



Fig. 7. Qualitative result for mesh reconstruction, and the white inner shell of 2DGS mesh is raised by the TSDF sampling strategy, whereas our mesh solves this issue.

TABLE I
MESH RECONSTRUCTION ABLATION STUDY

Methods	MSE	F-score
Gaustudio	0.153	0.268
Sugar	0.159	0.375
2DGS	0.153	0.228
Ours	0.151	0.721

are then transformed into the pre-timestamp global transformation matrix. We enable editing in the Isaac simulation backend [7] and seamless rendering in Dynamic Gaussian Splatting (details are shown in the attached video). Additionally, our approach also supports novel-policy editing [72]. Novel-policy allows training a position-based policy, distill-

ing a set of trajectory of end-effector pose from policy [48] based on the position of object from the reconstruction of our digital asset with rendering and simulation in our method. We currently support custom trajectory editing in Isaac Gym and policy implementation, demonstrating the ability for Sim2Real and Sim2Render applications based on this paradigm (details are shown on part *Policy Implementation* of our webpage).

V. CONCLUSION

Our goal was to develop a robust Real2Sim framework that significantly reduces the gap between real-world robotic manipulation tasks and their simulated counterparts. We accomplished this by introducing a hybrid representation model that integrates mesh geometry, Gaussian kernels, and physics attributes. This approach ensures high-quality, realistic, and physics-consistent rendering of robotic arm manipulation scenes. Our model was trained and validated across CR [9] and UR [10] product, demonstrating its effectiveness in constructing accurate URDFs from video data. This method not only enhances the fidelity of simulated environments but also generalizes well to other robotic applications, advancing the state-of-the-art in robotic learning and control. Our current grasping approach uses a position-based policy. However, Gaussian kernels serves as a world representation, enabling accurate rendering from any in-scene camera pose [73]. Our engine also supports vision-based grasping policies using our assets and model, allowing for the setup of the render camera and providing precise rendering.

REFERENCES

- [1] C. Wang, K. Ji, J. Geng, Z. Ren, T. Fu, F. Yang, Y. Guo, H. He, X. Chen, Z. Zhan, et al., "Imperative learning: A self-supervised neural-symbolic learning framework for robot autonomy," arXiv preprint arXiv:2406.16087, 2024.
- [2] Y. Ze, G. Yan, Y.-H. Wu, A. Macaluso, Y. Ge, J. Ye, N. Hansen, L. E. Li, and X. Wang, "Multi-task real robot learning with generalizable neural feature fields," *CoRL*, 2023.
- [3] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," ACM Transactions on Graphics, vol. 42, July 2023.
- [4] C. Li, F. Xia, R. Martín-Martín, M. Lingelbach, S. Srivastava, B. Shen, K. Vainio, C. Gokmen, G. Dharan, T. Jain, et al., "igibson 2.0: Objectcentric simulation for robot learning of everyday household tasks," arXiv preprint arXiv:2108.03272, 2021.
- [5] "Urdf." http://wiki.ros.org/urdf, 2024.
- [6] C. Jambon, B. Kerbl, G. Kopanas, S. Diolatzis, T. Leimkühler, and G. Drettakis, "Nerfshop: Interactive editing of neural radiance fields," Proceedings of the ACM on Computer Graphics and Interactive Techniques, vol. 6, no. 1, 2023.
- [7] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, et al., "Isaac gym: High performance gpu-based physics simulation for robot learning," arXiv preprint arXiv:2108.10470, 2021.
- [8] C. K. Liu and D. Negrut, "The role of physics-based simulators in robotics," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 4, no. 1, pp. 35–58, 2021.
- [9] CR5, "Dobot cr5 robotic arm key to the flexible and safe future." https://www.dobot-robots.com/products/ cr-series/cr5.html.
- [10] U. Robots, "Collaborative robotic automation— cobots from universal robots," 2019.
- [11] J. Sun, H. Jiao, G. Li, Z. Zhang, L. Zhao, and W. Xing, "3dgstream: On-the-fly training of 3d gaussians for efficient streaming of photorealistic free-viewpoint videos," in *Proceedings of the IEEE/CVF* Conference on Computer Vision and Pattern Recognition, pp. 20675– 20685, 2024.
- [12] M. Ding, Z. Chen, T. Du, P. Luo, J. Tenenbaum, and C. Gan, "Dynamic visual reasoning by learning differentiable physics models from video and language," *Advances In Neural Information Processing Systems*, vol. 34, pp. 887–899, 2021.
- [13] D. Shim, S. Lee, and H. J. Kim, "Snerl: Semantic-aware neural radiance fields for reinforcement learning," in *International Conference* on *Machine Learning*, pp. 31489–31503, PMLR, 2023.
- [14] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99– 106, 2021.
- [15] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction," arXiv preprint arXiv:2106.10689, 2021.
- [16] C. Ye, Y. Nie, J. Chang, Y. Chen, Y. Zhi, and X. Han, "Gaustudio: A modular framework for 3d gaussian splatting and beyond," arXiv preprint arXiv:2403.19632, 2024.
- [17] M. Tancik, E. Weber, E. Ng, R. Li, B. Yi, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, A. Ahuja, et al., "Nerfstudio: A modular framework for neural radiance field development," in ACM SIGGRAPH 2023 Conference Proceedings, pp. 1–12, 2023.
- [18] T. Jeruzalski, D. I. Levin, A. Jacobson, P. Lalonde, M. Norouzi, and A. Tagliasacchi, "Nilbs: Neural inverse linear blend skinning," arXiv preprint arXiv:2004.05980, 2020.
- [19] Y.-H. Huang, Y.-T. Sun, Z. Yang, X. Lyu, Y.-P. Cao, and X. Qi, "Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4220–4230, 2024.
- [20] E. Ślapak, E. Pardo, M. Dopiriak, T. Maksymyuk, and J. Gazda, "Neural radiance fields in the industrial and robotics domain: applications, research opportunities and use cases," *Robotics and Computer-Integrated Manufacturing*, vol. 90, p. 102810, 2024.
- [21] T. Chen, O. Shorinwa, W. Zeng, J. Bruno, P. Dames, and M. Schwager, "Splat-nav: Safe real-time robot navigation in gaussian splatting maps," arXiv preprint arXiv:2403.02751, 2024.
- [22] Y. Y. Bardienus P. Duisterhof, Zhao Mandi, "Deformgs: Scene flow in highly deformable scenes for deformable object manipulation," in

- The 16th International Workshop on the Algorithmic Foundations of Robotics (WAFR), 2024.
- [23] W. F. Whitney, T. Lopez-Guevara, T. Pfaff, Y. Rubanova, T. Kipf, K. Stachenfeld, and K. R. Allen, "Learning 3d particle-based simulators from rgb-d videos," arXiv preprint arXiv:2312.05359, 2023.
- [24] L. Xie, J. Julin, K. Niinuma, and L. A. Jeni, "Gaussian splatting lk," arXiv preprint arXiv:2407.11309, 2024.
- [25] R. Yang, Z. Zhu, Z. Jiang, B. Ye, X. Chen, Y. Zhang, Y. Chen, J. Zhao, and H. Zhao, "Spectrally pruned gaussian fields with neural compensation," 2024.
- [26] Z. Zhan, X. Li, Q. Li, H. He, A. Pandey, H. Xiao, Y. Xu, X. Chen, K. Xu, K. Cao, et al., "Pypose v0. 6: The imperative programming interface for robotics," arXiv preprint arXiv:2309.13035, 2023.
- [27] M.-J. Rakotosaona, F. Manhardt, D. M. Arroyo, M. Niemeyer, A. Kundu, and F. Tombari, "Nerfmeshing: Distilling neural radiance fields into geometrically-accurate 3d meshes," in 2024 International Conference on 3D Vision (3DV), pp. 1156–1165, IEEE, 2024.
- [28] A. Guédon and V. Lepetit, "Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5354–5363, 2024.
- [29] I. Liu, H. Su, and X. Wang, "Dynamic gaussians mesh: Consistent mesh reconstruction from monocular videos," arXiv preprint arXiv:2404.12379, 2024.
- [30] B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao, "2d gaussian splatting for geometrically accurate radiance fields," in ACM SIGGRAPH 2024 Conference Papers, pp. 1–11, 2024.
- [31] Z. Wu, T. Liu, L. Luo, Z. Zhong, J. Chen, H. Xiao, C. Hou, H. Lou, Y. Chen, R. Yang, Y. Huang, X. Ye, Z. Yan, Y. Shi, Y. Liao, and H. Zhao, "Mars: An instance-aware, modular and realistic simulator for autonomous driving," CICAI, 2023.
- [32] Q. Dai, Y. Zhu, Y. Geng, C. Ruan, J. Zhang, and H. Wang, "Graspnerf: Multiview-based 6-dof grasp detection for transparent and specular objects using generalizable nerf," in 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 1757–1763, IEEE, 2023.
- [33] I. Kapelyukh, V. Vosylius, and E. Johns, "Dall-e-bot: Introducing web-scale diffusion models to robotics," *IEEE Robotics and Automation Letters*, vol. 8, no. 7, pp. 3956–3963, 2023.
- [34] F. Zhu, H. Wu, S. Guo, Y. Liu, C. Cheang, and T. Kong, "Irasim: Learning interactive real-robot action simulators," arXiv:2406.12802, 2024.
- [35] G. Lu, S. Zhang, Z. Wang, C. Liu, J. Lu, and Y. Tang, "Manigaussian: Dynamic gaussian splatting for multi-task robotic manipulation," arXiv preprint arXiv:2403.08321, 2024.
- [36] Y. Jiang, C. Wang, R. Zhang, J. Wu, and L. Fei-Fei, "Transic: Sim-to-real policy transfer by learning from online correction," in *Conference on Robot Learning*, 2024.
- [37] J. Whitman, M. Travers, and H. Choset, "Learning modular robot control policies," *IEEE Transactions on Robotics*, vol. 39, no. 5, pp. 4095–4113, 2023.
- [38] D. Coleman, I. Sucan, S. Chitta, and N. Correll, "Reducing the barrier to entry of complex robotic software: a moveit! case study," arXiv preprint arXiv:1404.3785, 2014.
- [39] Y.-L. Qiao, J. Liang, V. Koltun, and M. C. Lin, "Efficient differentiable simulation of articulated bodies," in *International Conference on Machine Learning*, pp. 8661–8671, PMLR, 2021.
- [40] Y. Zheng, X. Chen, Y. Zheng, S. Gu, R. Yang, B. Jin, P. Li, C. Zhong, Z. Wang, L. Liu, et al., "Gaussiangrasper: 3d language gaussian splatting for open-vocabulary robotic grasping," arXiv preprint arXiv:2403.09637, 2024.
- [41] K. Wang, W. R. Johnson, S. Lu, X. Huang, J. Booth, R. Kramer-Bottiglio, M. Aanjaneya, and K. Bekris, "Real2sim2real transfer for control of cable-driven robots via a differentiable physics engine," in 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2534–2541, IEEE, 2023.
- [42] B. Chen, R. Kwiatkowski, C. Vondrick, and H. Lipson, "Fully body visual self-modeling of robot morphologies," *Science Robotics*, vol. 7, no. 68, p. eabn1944, 2022.
- [43] Z. Zhong, J. Cao, S. Gu, S. Xie, W. Gao, L. Luo, Z. Yan, H. Zhao, and G. Zhou, "Assist: Interactive scene nodes for scalable and realistic indoor simulation," arXiv preprint arXiv:2311.06211, 2023.
- [44] M. Bauza, A. Bronars, Y. Hou, I. Taylor, N. Chavan-Dafle, and A. Rodriguez, "Simple, a visuotactile method learned in simulation to precisely pick, localize, regrasp, and place objects," *Science Robotics*, vol. 9, no. 91, p. eadi8808, 2024.

- [45] H. Ha, Y. Gao, Z. Fu, J. Tan, and S. Song, "Umi on legs: Making manipulation policies mobile with manipulation-centric whole-body controllers," arXiv preprint arXiv:2407.10353, 2024.
- [46] J. Abou-Chakra, K. Rana, F. Dayoub, and N. Sünderhauf, "Physically embodied gaussian splatting: A realtime correctable world model for robotics," in 8th Annual Conference on Robot Learning, 2024.
- [47] V. Lim, H. Huang, L. Y. Chen, J. Wang, J. Ichnowski, D. Seita, M. Laskey, and K. Goldberg, "Real2sim2real: Self-supervised learning of physical single-step dynamic actions for planar robot casting," in 2022 International Conference on Robotics and Automation (ICRA), pp. 8282–8289, IEEE, 2022.
- [48] M. Torne, A. Simeonov, Z. Li, A. Chan, T. Chen, A. Gupta, and P. Agrawal, "Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation," arXiv preprint arXiv:2403.03949, 2024.
- [49] D. Qu, Q. Chen, P. Zhang, X. Gao, B. Zhao, D. Wang, and X. Li, "Livescene: Language embedding interactive radiance fields for physical scene rendering and control," arXiv preprint arXiv:2406.16038, 2024
- [50] K. M. Jatavallabhula, M. Macklin, F. Golemo, V. Voleti, L. Petrini, M. Weiss, B. Considine, J. Parent-Lévesque, K. Xie, K. Erleben, et al., "gradsim: Differentiable simulation for system identification and visuomotor control," arXiv preprint arXiv:2104.02646, 2021.
- [51] X. Li, Y.-L. Qiao, P. Y. Chen, K. M. Jatavallabhula, M. Lin, C. Jiang, and C. Gan, "Pac-nerf: Physics augmented continuum neural radiance fields for geometry-agnostic system identification," arXiv preprint arXiv:2303.05512, 2023.
- [52] E. Sifakis and J. Barbic, "Fem simulation of 3d deformable solids: a practitioner's guide to theory, discretization and model reduction," in Acm siggraph 2012 courses, pp. 1–50, 2012.
- [53] J. Choe, S. Im, F. Rameau, M. Kang, and I. S. Kweon, "Volumefusion: Deep depth fusion for 3d scene reconstruction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16086–16095, 2021.
- [54] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4104–4113, 2016.
- [55] Y. Weng, B. Wen, J. Tremblay, V. Blukis, D. Fox, L. Guibas, and S. Birchfield, "Neural implicit representation for building digital twins of unknown articulated objects," in *Proceedings of the IEEE/CVF* Conference on Computer Vision and Pattern Recognition, pp. 3141– 3150, 2024.
- [56] R.-Z. Qiu, G. Yang, W. Zeng, and X. Wang, "Feature splatting: Language-driven physics-based scene synthesis and editing," arXiv preprint arXiv:2404.01223, 2024.
- [57] K. Ehsani, T. Gupta, R. Hendrix, J. Salvador, L. Weihs, K.-H. Zeng, K. P. Singh, Y. Kim, W. Han, A. Herrasti, et al., "Spoc: Imitating shortest paths in simulation enables effective navigation and manipulation in the real world," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16238–16250, 2024.
- [58] P. I. Corke, "A simple and systematic approach to assigning denavit– hartenberg parameters," *IEEE transactions on robotics*, vol. 23, no. 3, pp. 590–594, 2007.
- [59] Y. Jiang, C. Yu, T. Xie, X. Li, Y. Feng, H. Wang, M. Li, H. Lau, F. Gao, Y. Yang, et al., "Vr-gs: a physical dynamics-aware interactive gaussian splatting system in virtual reality," in ACM SIGGRAPH 2024 Conference Papers, pp. 1–1, 2024.
- [60] T. Xie, Z. Zong, Y. Qiu, X. Li, Y. Feng, Y. Yang, and C. Jiang, "Phys-gaussian: Physics-integrated 3d gaussians for generative dynamics," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4389–4398, 2024.
- [61] S. Kucuk and Z. Bingul, "The inverse kinematics solutions of industrial robot manipulators," in *Proceedings of the IEEE International Conference on Mechatronics*, 2004. ICM '04., pp. 274–279, 2004.
- [62] C. M. Kim, M. Wu, J. Kerr, K. Goldberg, M. Tancik, and A. Kanazawa, "Garfield: Group anything with radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21530–21539, 2024.
- [63] V. Modi, N. Sharp, O. Perel, S. Sueda, and D. I. Levin, "Simplicits: Mesh-free, geometry-agnostic elastic simulation," ACM Transactions on Graphics (TOG), vol. 43, no. 4, pp. 1–11, 2024.
- [64] Z. Yang, X. Gao, W. Zhou, S. Jiao, Y. Zhang, and X. Jin, "Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision* and Pattern Recognition, pp. 20331–20341, 2024.

- [65] L. Liang, L. Bian, C. Xiao, J. Zhang, L. Chen, I. Liu, F. Xiang, Z. Huang, and H. Su, "Robo360: A 3d omnispective multi-material robotic manipulation dataset," arXiv preprint arXiv:2312.06686, 2023.
- [66] Z. Fu, T. Z. Zhao, and C. Finn, "Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation," arXiv preprint arXiv:2401.02117, 2024.
- [67] Q. Vuong, S. Levine, H. R. Walke, K. Pertsch, A. Singh, R. Doshi, C. Xu, J. Luo, L. Tan, D. Shah, et al., "Open x-embodiment: Robotic learning datasets and rt-x models," in *Towards Generalist Robots: Learning Paradigms for Scalable Skill Acquisition @ CoRL2023*, 2023.
- [68] H. Liu, C. Ye, Y. Nie, Y. He, and X. Han, "Lasa: Instance reconstruction from real scans using a large-scale aligned shape annotation dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20454–20464, 2024.
- [69] Q. Wang, Y.-Y. Chang, R. Cai, Z. Li, B. Hariharan, A. Holynski, and N. Snavely, "Tracking everything everywhere all at once," in Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 19795–19806, 2023.
- [70] Y. Xiao, Q. Wang, S. Zhang, N. Xue, S. Peng, Y. Shen, and X. Zhou, "Spatialtracker: Tracking any 2d pixels in 3d space," in *Proceedings* of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 20406–20417, 2024.
- [71] S. Fridovich-Keil, G. Meanti, F. R. Warburg, B. Recht, and A. Kanazawa, "K-planes: Explicit radiance fields in space, time, and appearance," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12479–12488, 2023.
- [72] F. Barthel, A. Beckmann, W. Morgenstern, A. Hilsmann, and P. Eisert, "Gaussian splatting decoder for 3d-aware generative adversarial networks," 2024.
- [73] B. Wen, W. Yang, J. Kautz, and S. Birchfield, "Foundationpose: Unified 6d pose estimation and tracking of novel objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17868–17879, 2024.