SplatSim: Zero-Shot Sim2Real Transfer of RGB Manipulation Policies Using Gaussian Splatting

M. Nomaan Qureshi, Sparsh Garg, Francisco Yandun, David Held, George Kantor, Abhisesh Silwal
Robotics Institute, School of Computer Science
Carnegie Mellon University, United States
{mquresh2, sparshg, fyandun, dheld, gkantor, asilwal}@andrew.cmu.edu

Abstract: Sim2Real transfer, particularly for manipulation policies relying on RGB images, remains a critical challenge in robotics due to the significant domain shift between synthetic and real-world visual data. In this paper, we propose *SplatSim*, a novel framework that leverages Gaussian Splatting as the primary rendering primitive to reduce the Sim2Real gap for RGB-based manipulation policies. By replacing traditional mesh representations with Gaussian Splats in simulators, *SplatSim* produces highly photorealistic synthetic data while maintaining the scalability and cost-efficiency of simulation. We demonstrate the effectiveness of our framework by training manipulation policies within *SplatSim* and deploying them in the real world in a zero-shot manner, achieving an average success rate of 86.25%, compared to 97.5% for policies trained on the real-world data. Videos can be found on our project page: https://splatsim.github.io

1 Introduction

In this paper, we propose a systematic and novel method to reduce the Sim2Real gap for RGB images, by leveraging Gaussian Splatting [1] as a photorealistic render, using existing simulators as the physics backbone. We propose utilizing Gaussian Splatting [1] as the primary rendering primitive, replacing traditional mesh-based representations in existing simulators, to significantly improve the photo-realism of rendered scenes. By integrating these renderings of simulated demonstrations with state-of-the-art behavior cloning techniques, we introduce a framework for zero-shot transfer of manipulation policies trained entirely on simulation data, to the real world. Our key contributions are as follows:

- We propose a novel and scalable data generation framework, *SplatSim* for manipulation tasks. *SplatSim* is focused predominantly on bridging the vision Sim2Real gap by leveraging photorealistic renderings generated through Gaussian Splatting, replacing traditional mesh representation in the rendering pipeline of the simulator.
- We show how to leverage Robot Splat Models and Object Splat Models, along with the simulator as a physics backend, to generate photorealistic trajectories of robot-object interactions. Our method eliminates the need for the real-world data collection to learn these interactions, and relies solely on an initial video of the static scene with the robot. We further demonstrate how these renderings, when combined with simulated demonstrations, can be utilized to generate high-quality synthetic datasets for behavior cloning methods.
- We demonstrate the effectiveness of our framework by deploying RGB policies, trained entirely in simulation, to the real world in a zero-shot manner across four tasks, achieving an average success rate of 86.25%, compared to 97.5% for policies trained on the real-world data.

8th Conference on Robot Learning (CoRL 2024), Munich, Germany.

2 Method

The key premise of our method is that if each rigid body in the Gaussian Splat representation of the real-world scene can be accurately segmented, and its corresponding homogeneous transformation relative to the simulator is identified, then it becomes feasible to render the rigid body in novel poses. The rigid bodies can include links of the robot, links of the gripper, articulated objects, or simple non-deformable objects. By applying this process to all rigid bodies interacting with the robot in simulation, we can generate photorealistic renderings for an entire demonstration trajectory. This approach is analogous to traditional rendering in simulators; however, instead of using mesh primitives, we utilize Gaussian Splats as the underlying representation. This approach allows us to be more effective at capturing the detailed visual fidelity of the real-world scenes.

2.1 Problem Statement

We define $\mathcal{S}_{real}$ as the Gaussian Splat of a real-world scene, captured from multiple RGB viewpoints, including the robot. We also define $\mathcal{S}^k_{obj}$ as the splat of the k-th object in the scene, captured from multiple viewpoints. Our goal is to use $\mathcal{S}_{real}$ for generating photorealistic renderings I^{sim} of a robot operating in any simulator (e.g., PyBullet). Then, we can leverage this representation to collect demonstrations using the expert $\mathcal{E}$ for training RGB-based policies.

The expert $\mathcal{E}$ generates a trajectory $\tau_{\mathcal{E}}$ consisting of state-action pairs $\{(s_1, a_1), \ldots, (s_T, a_T)\}$ for a full episode. The state at each time step t is defined as $s_t = (q_t, x^1_t, \ldots, x^n_t)$, where $q_t \in \mathbb{R}^m$ denotes the robot's joint angles and $x^k_t = (p^k_t, R^k_t)$ represents the position $p^k_t \in \mathbb{R}^3$ and orientation $R^k_t \in SO(3)$ of the k-th object in the scene. The corresponding action $a_t = (p^e_t, R^e_t)$ refers to the end effector's position $p^e_t \in \mathbb{R}^3$ and orientation $R^e_t \in SO(3)$.

The renderings I^{sim}, derived from these simulated states s_t, are used as inputs to train the policy $\pi_{\mathcal{I}}$. The policy relies solely on real-world RGB images I^{real} at test time.

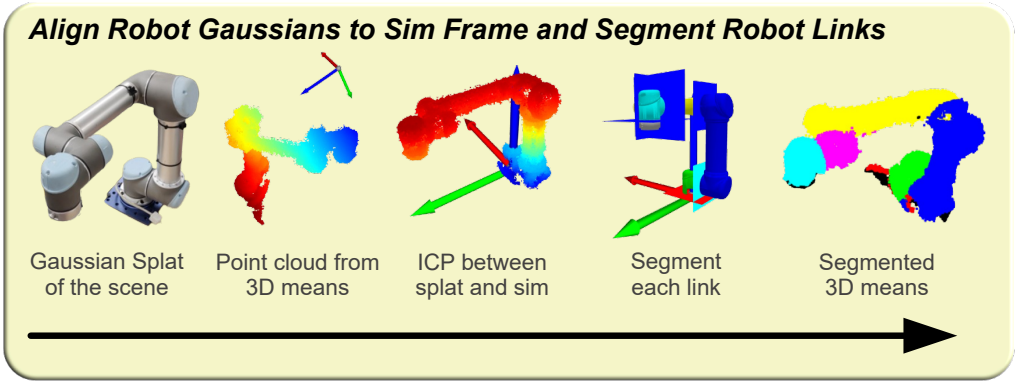

Figure 1: The robot is visualized in a static scene by first creating a Gaussian splat of the scene with the robot in its home position. The robot's point cloud is manually segmented and aligned with the canonical robot frame using the ICP algorithm. Each robot link is then segmented, and forward kinematics transformations are applied, enabling the rendering of the robot at arbitrary joint configurations.

2.2 Robot Splat Models

Our method for obtaining robot renderings at novel joint poses is summarized in Fig. 1. It follows a three-step approach:

2.2.1 Alignment of Gaussian Splat Robot Frame to the Simulator Frame

In order to combine the Gaussian Splat representation $\mathcal{S}_{real}$ with the simulator, we first manually segment out the 3D Gaussians associated with the robot. The means of these 3D Gaussians form

a point cloud which is aligned with the ground truth point cloud obtained from the simulator. To achieve this, we use the Iterative Closest Point (ICP) algorithm, which produces the desired transformation $T_{\mathcal{F}_{robot}}^{\mathcal{F}_{splat}}$.

2.2.2 Segmentation of the Robot Links

To associate the 3D Gaussians with their respective links in $\mathcal{S}_{real}$, we leverage the ground truth bounding boxes of the robot's links, provided by its CAD model. This method allows us to isolate the 3D Gaussians corresponding to each link in the real-world scene, denoted as $\mathcal{S}_{real}^{l}$, where l refers to the l-th link of the robot.

2.2.3 Forward Kinematics Transformation

Once we have the 3D Gaussians for individual links and the frames aligned, we can use the robot's forward kinematics to get the robot pose at arbitrary joint angles $q_t \in s_t$, given by the simulator. In this work, we use the forward kinematics routine from PyBullet to get the Transformation T_{fk}^{l} for link l in the robot's canonical frame $\mathcal{F}_{sim}$. The transformation of the 3D Gaussians can be calculated as :

$$T = (T_{\mathcal{F}_{robot}}^{\mathcal{F}_{splat}})^{-1} \cdot T_{fk}^{l} \cdot T_{\mathcal{F}_{robot}}^{\mathcal{F}_{splat}} \tag{1}$$

where $T_{\mathcal{F}_{robot}}^{\mathcal{F}_{splat}}$ is the transformation matrix to get the robot from splat frame to the simulation frame. Once the transformation for each link is calculated, we transform the 3D Gaussians related to individual links of the robot. The robot at novel poses is then rendered by the standard Gaussian Splatting rendering framework [1].

2.3 Rendering Simulated Trajectories using SplatSim

Now that we are able to render individual rigid bodies in the scene, we can use this to represent any simulated trajectory $\tau_{\mathcal{E}}$ with photorealistic accuracy. We use these state-based transformations along with methods described in Sec. 2.2 to get the demonstration for our policy to learn from $\tau_{\mathcal{G}} = \{(I_1^{sim}, a_1), (I_2^{sim}, a_2), \ldots, (I_T^{sim}, a_T)\}$. This data is used by policy to predict actions from the synthetically generated images.

2.4 Policy Training and Deployment

For learning from the generated demonstrations $\tau_{\mathcal{G}}$ in the simulator, we employ Diffusion Policy [2, 3], which is the state of the art for behavior cloning. Although our method significantly mitigates the vision Sim2Real gap, discrepancies between the simulated and real-world environments remain. For instance, simulated scenes lack shadows, and rigid body assumptions can lead to improper rendering of flexible components such as robot cables. To address these issues, we incorporate image augmentations similar to [4] during policy training, which includes adding gaussian noise, random erasing and adjusting brightness and contrast of the image. These augmentations notably enhance the robustness of the policy and improve its performance during real-world deployment.

3 Experiments

To evaluate the effectiveness of our framework in bridging the Sim2Real gap for RGB-based manipulation tasks, we conducted extensive experiments across four real-world manipulation tasks. We begin by detailing the data collection process in both the simulator and real-world environments. We then compare the performance of policies trained on our synthetic data with Real2Real policies—those trained on real-world data and deployed in real-world environments. This comparison demonstrates the high fidelity of our synthetic data, showing that policies trained within our framework can be deployed to real-world tasks without fine-tuning on the real-world data. Additionally, we assess Sim2Sim performance by training and evaluating policies entirely within the *SplatSim* framework, allowing us to quantify the degradation in performance during Sim2Real transfer.

3.1 Demonstrations in the Real World and Simulation

In the real world, demonstrations for each task were manually collected by a human expert. In contrast, the simulator streamlines this process by employing privileged information-based motion planners, which automatically generate data using privileged information, such as the position and orientation of each rigid body in the scene. The simulator not only reduces effort by automating resets between demonstrations when a human expert is involved but more importantly, it leverages motion planners that eliminate the need for human intervention entirely. This enables the generation of large-scale, high-quality demonstration datasets with minimal manual input. As a result, the simulator drastically reduces the time and effort required for data collection. As shown in Table 1, while real-world demonstration collection required about 20.5 hours, the same tasks were completed in just 3 hours in the simulator, underscoring the efficiency and scalability of our approach.

3.2 Zero-Shot Policy Deployment Results

We evaluate the zero-shot deployment of our policies across four contact-rich real-world tasks, using task success rate as the primary metric. As shown in Table 1, our method achieves an average success rate of 86.25% for zero-shot Sim2Real transfer, compared to 97.5% for policies trained directly on real-world data, highlighting the effectiveness of our approach. All experiments were conducted using a UR5 robot equipped with a Robotiq 2F-85 gripper and 2 Intel Realsense D455 cameras [5] with deployment on an NVIDIA RTX 3080Ti GPU for the Diffusion Policy [2].

Task	**Successful Trials** (Out of 40 Trials)			**Human Effort to Collect Data** (hours)	
	Sim2Sim	Real2Real	Sim2Real (**SplatSim**)	Simulator	Real World
T-Push	100%	100%	90%	3.0	3.5
Pick-Up-Apple	100%	100%	95%	0.0*	3.5
Orange-On-Plate	97.5%	95%	90%	0.0*	6.0
Assembly	85%	90%	70%	0.0*	7.5
Total	95.62%	97.5%	**86.25**%	3.0	20.5

* Automated process

Table 1: Comparison of task success rates and data collection times across various manipulation tasks. Our policies trained solely on synthetic data achieve an 86.25% zero-shot Sim2Real performance, comparable to those trained on real-world data. By leveraging the automation capabilities of simulators, we significantly reduce the human effort required for data generation.

4 Conclusion

In this work, we tackled the challenge of reducing the Sim2Real gap for RGB-based manipulation policies by leveraging Gaussian Splatting as a photorealistic rendering technique, integrated with existing simulators for physics-based interactions. Our framework enables zero-shot transfer of RGB-based manipulation policies trained in simulation to real-world environments. While our framework advances the current state-of-the-art, it is still limited to rigid body manipulation and cannot handle complex objects such as cloth, liquids, or plants. We will also further improve our system to train and deploy robots in highly complex and contact-rich tasks in the real world. Specifically, agricultural tasks such as pruning and harvesting, which require data that is challenging to obtain under field conditions.

Acknowledgement

We would like to express our gratitude to Prof. Shubham Tulsiani for his valuable insights during the early development of this idea. This work is in part supported by NSF/USDA-NIFA AIIRA AI Research Institute 2021-67021-35329 and USDA-NIFA/NSF National Robotics Initiative 2021-67021-35974.

Acknowledgments

References

[1] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/.

[2] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[3] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 2024.

[4] A. Byravan, J. Humplik, L. Hasenclever, A. Brussee, F. Nori, T. Haarnoja, B. Moran, S. Bohez, F. Sadeghi, B. Vujatovic, and N. Heess. Nerf2real: Sim2real transfer of vision-guided bipedal motion skills using neural radiance fields. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9362–9369, 2023. doi:10.1109/ICRA48891.2023.10161544.

[5] L. Keselman, J. I. Woodfill, A. Grunnet-Jepsen, and A. Bhowmik. Intel(r) realsense(tm) stereoscopic depth cameras. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1267–1276, 2017. doi:10.1109/CVPRW.2017.167.