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

M. Nomaan Qureshi[1], Sparsh Garg[1], Francisco Yandun[1], David Held[1], George Kantor[1], Abhisesh Silwal[1]

***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 real-world data.**

I. INTRODUCTION

The Sim2Real problem, a focal challenge in robotics, pertains to the transfer of control policies learned in simulated environments to real world settings. Recently, significant progress has been made in deploying controllers trained in simulation to the real world in a zero-shot manner. Robots have demonstrated the ability to walk on rough terrains [1], [2], perform in-hand object rotation [3], [4], [5], [6], and grasp previously unseen objects [7]. Notably, all of these methods rely on perception modalities like depth, tactile sensing, or point cloud inputs, which have gained significant attention due to the relatively small Sim2Real gap they offer. The reduced discrepancy between simulated and real-world data in these modalities has led to remarkable progress, reinforcing the idea that *modalities that can be simulated well, can be transferred well.*

In contrast, RGB images are rarely used as the primary sensing modality in robot learning applications. RGB images offer several unique advantages over other commonly used modalities in Sim2Real transfer. They capture crucial visual details such as color, texture, lighting, and surface reflectivity, to name a few, which are essential for understanding complex environments. For instance, in a simple task of plucking ripe fruits, color is a key feature for determining ripeness—an inference straightforward in RGB space but difficult and impractical with depth or tactile inputs. Additionally, RGB images are easy to acquire in real-world environments with cameras and align closely with human perception, making them well-suited for interpreting intricate details in dynamic and complex scenes.

But why has it been difficult to deploy policies trained in simulation with RGB information to the real world? The problem lies in the fact that the distribution of images the robot observes in the simulator is very different from the distribution of images it would see in the real world. This makes "vision Sim2Real an out-of-domain generalization problem" [9], a fundamental challenge in machine learning that is still unsolved. For this reason, policies trained on simulated images often struggle when applied to distributions of real-world images.

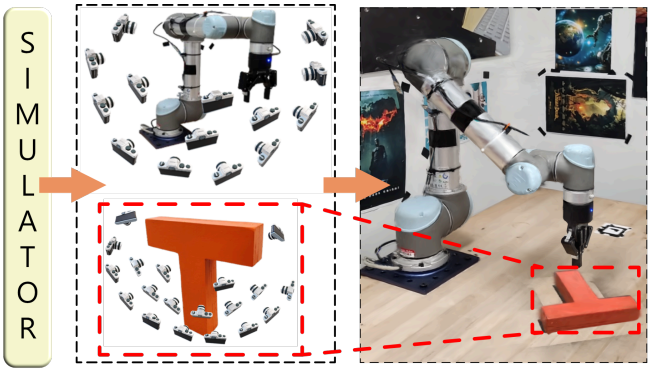

Fig. 1: We employ Gaussian Splatting [8] as the primary rendering primitive within existing simulation environments to generate highly photorealistic synthetic data for robotic manipulation tasks. Our framework retains all the traditional advantages of simulators—including scalability, cost-efficiency, and safety—while enhancing visual realism. Policies trained exclusively on this synthetic data exhibit zero-shot transfer capabilities to real-world scenarios, achieving performance comparable to those trained on real-world datasets.

In this paper, we propose a systematic and novel method to reduce the Sim2Real gap for RGB images, by leveraging Gaussian Splatting [8] as a photorealistic render, using existing simulators as the physics backbone. We propose utilizing Gaussian Splatting [8] as the primary rendering primitive, replacing traditional mesh-based representations in existing simulators, to significantly improve the photorealism 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.

1 All authors are with the Carnegie Mellon University, USA.
Website and code available at https://splatsim.github.io

- 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.

II. Related Works

A. Sim2real

Robotics simulation tools like [10], [11], [12], [13], [14], [15] have become invaluable in scaling up robot learning due to several advantages including parallelization, cost and time efficiency, and safety. Recent advancements in transferring learned policies from simulation to the real world have demonstrated impressive results, particularly in domains that leverage modalities with a low Sim2Real gap, such as depth, point cloud, proprioception, or tactile feedback. These modalities have enabled robots to perform contact-rich tasks like quadruped locomotion [1], [16], [2], [17], [18] and bipedal locomotion [19], [20], dexterous manipulation [3], [4], [5], [6], [7], manipulation of articulated objects [21], [22], among others [23], [24], [25], [26].

However, Sim2Real transfer for RGB-based manipulation policies remains challenging. Previous attempts in this domain have primarily focused on tasks like navigation [27], where high-fidelity collision meshes from scans [28], [29] can enhance visual realism but fail to capture dynamic interactions with the environment. Furthermore, existing approaches such as domain adaptation [30], often rely on extensive offline data collection of real-world object interactions.

In this work, we address the challenge of transferring RGB-based policies for manipulation tasks, which require rendering complex interactions between objects and the robot. Our method requires only an initial video of the static scene without the need for additional real-world data collection. A work notably related to ours is RialTo [31] which uses a Real2Sim2Real approach similar to ours. However, their policy is still trained on point clouds, which requires depth during execution time. In contrast, *SplatSim* only uses RGB images for learning and policy deployment. Another recent work Maniwhere [32] does large-scale reinforcement learning in simulation and shows generalization to the real world, however, their method still requires depth at test time and cannot work with just RGB images in the real world.

B. Gaussian Splatting for Robotics

Gaussian Splatting [8] is a state-of-the-art rendering technique that models scenes using 3D Gaussian primitives, offering an efficient and photorealistic representation of complex geometries. In contrast to NeRF [33] and its derivatives [34], [35], [36], the explicit, point cloud-like structure of Gaussian Splats enables easier manipulation, which has led to numerous subsequent works focused on dynamic Gaussian Splatting models [37], [38], [39], [40], [41]. This explicit nature of Gaussian Splatting has also garnered interest in the robotics community, with recent studies applying it to language-guided manipulation [42], object grasping [43], and deformable object manipulation [44]. The two related works, Embodied Gaussians [45] and RoboStudio [46], focus on learning from real-world data. Embodied Gaussians [45] directly learns a forward model for robot-object interactions, requiring real-world data for each new robot and object, while we offload dynamics to a physics engine and focus on RGB-based policy deployment. RoboStudio [46] combines simulation with Gaussian Splatting but focuses on system identification, whereas our approach generates synthetic data for real-world deployment using existing simulators. Another closely related work to our method is [47], which combines Gaussian Splatting with a simulator, but it is focused on navigation. Unlike manipulation tasks, the agent in their approach does not interact with or manipulate the environment.

III. Preliminary

A. Rigid Body Transformations in Gaussian Splatting

In Gaussian Splatting, segmented objects within a scene can undergo rigid body transformations, such as translation and rotation, while still maintaining high-quality renderings. Each object, represented by a set of 3D Gaussians, can be transformed using a homogeneous transformation matrix T, defined by the rotation R and translation t. For a 3D Gaussian with mean position μ and covariance matrix Σ, the transformed position μ' and covariance Σ' under the rigid transformation are given by:

$$\mu' = R\mu + t \tag{1}$$
$$\Sigma' = R\Sigma R^T \tag{2}$$

Applying these transformations updates the position and orientation of the 3D Gaussians of the object while preserving the corresponding geometric properties. Despite making these changes in the object's configuration, the Gaussian representation enables smooth and accurate renderings.

IV. 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

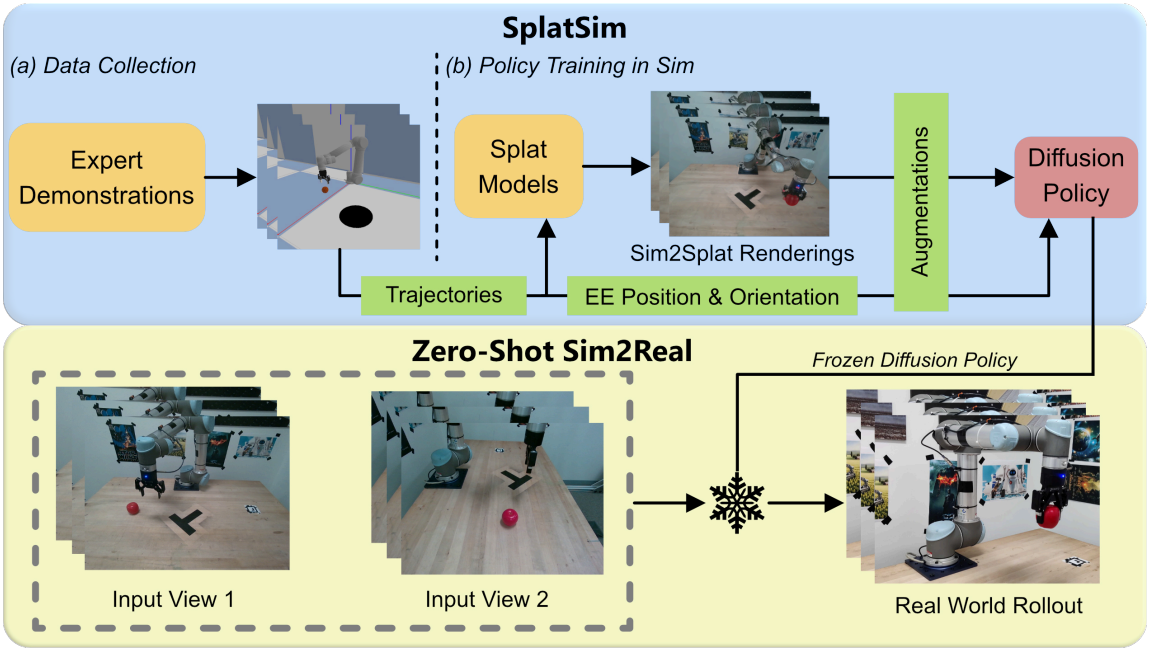

Fig. 2: **Top:** Our proposed SplatSim framework. Expert demonstrations are collected (a) in a physics simulator (PyBullet). In our case, these demonstrations come either from human experts (teleoperation via Gello [48]) or through a privileged information-based motion planner. The trajectories from the simulator are then fed to the simulator-aligned splat models of the scene and the object (b). We transform the 3D Gaussians to manipulate the static Gaussian Splat models, as delineated in Sec. IV-C, to extract photorealistic renderings of the scene at novel joint and object poses, which serve as the RGB state observations for the diffusion policy. Along with these RGB observations, diffusion policy [49] also takes the end effector position and orientation as the input. We augment the end effector states as well. **Bottom:** Once trained with the sim data, we freeze the policy and directly deploy it to the real-world setting.

utilize Gaussian Splats as the underlying representation. This approach allows us to be more effective at capturing the detailed visual fidelity of real-world scenes.

The following subsections describe our method. We begin by formalizing the problem statement in Sec. IV-A and notations in Sec. IV-B. Next, we detail the segmentation and rendering of each robot link at novel joint poses in Sec. IV-C, individual objects at new positions in Sec. IV-D, and grippers in Sec. IV-E. Sec. IV-F covers the rendering of complete robot-object interaction trajectories, followed by the policy training protocol in Sec. IV-G.

A. 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.

B. Definitions of Coordinate Frames and Transformations

We define several coordinate frames to clarify the relationships between the real-world scene, the simulator, and the splat point clouds. The real-world coordinate frame, denoted as $\mathcal{F}_{real}$, serves as the primary reference frame. Both the simulator coordinate frame, $\mathcal{F}_{sim}$, and the real-world robot frame, $\mathcal{F}_{robot}$, are aligned with $\mathcal{F}_{real}$. This alignment ensures that the robot's base in the simulator and the real world share the same coordinate system.

Additionally, the splat coordinate frame, denoted as $\mathcal{F}_{splat}$, represents the frame of the base of the robot in the Gaussian Splat of the scene $\mathcal{S}_{real}$. The robot base in the splat point cloud has a different frame from $\mathcal{F}_{real}$, and we account for this difference by using the transformation matrix $T^{\mathcal{F}_{splat}}_{\mathcal{F}_{robot}}$.

We also define the k-th object frame in the simulator, $\mathcal{F}_{k-obj,sim}$, where objects are initialized in $\mathcal{SIM}$ at the origin with no rotation. The k-th object frame in the splat, $\mathcal{F}_{k-obj,splat}$, represents the object's position and orientation in its Gaussian splat $\mathcal{S}_{obj}$. The object frames $\mathcal{F}_{k-obj,sim}$ and $\mathcal{F}_{k-obj,splat}$ are later aligned during the simulation and splat process using the transformation matrix $T^{\mathcal{F}_{k-obj,sim}}_{\mathcal{F}_{k-obj,splat}}$.

C. Robot Splat Models

Our method for obtaining robot renderings at novel joint poses is summarized in Fig. 3. It follows a three-step approach:

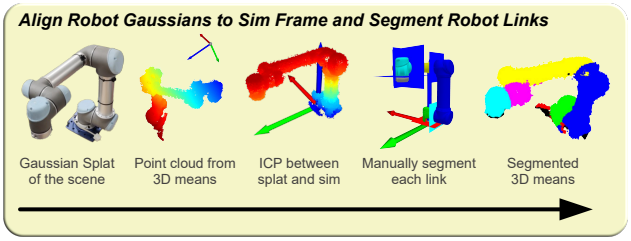

Fig. 3: The robot is visualized in a static scene by first creating a Gaussian splat of the scene with the robot in its home position. The robot's point cloud is manually segmented and aligned with the canonical robot frame using the ICP algorithm. Each robot link is then segmented, and forward kinematics transformations are applied, enabling the rendering of the robot at arbitrary joint configurations.

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 $T^{\mathcal{F}_{splat}}_{\mathcal{F}_{robot}}$.

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 $\mathcal{S}^l_{real}$, where l refers to the l-th link of the robot.

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^l_{fk} for link l in the robot's canonical frame $\mathcal{F}_{sim}$. The transformation of the 3D Gaussians can be calculated as :

$$T = (T^{\mathcal{F}_{splat}}_{\mathcal{F}_{robot}})^{-1} \cdot T^l_{fk} \cdot T^{\mathcal{F}_{splat}}_{\mathcal{F}_{robot}} \tag{3}$$

where $T^{\mathcal{F}_{splat}}_{\mathcal{F}_{robot}}$ is the transformation matrix to get the robot from splat frame to the simulation frame. Once the transformation for each link is calculated, we use Eq. 1 and Eq. 2 to 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 [8].

D. Object Splat Models

Similar to the robot rendering, we use ICP to align each object's 3D Gaussians $\mathcal{S}^k_{obj}$ to its simulated ground truth point cloud. In this way, we get the transformation $T^{\mathcal{F}_{k-obj,splat}}_{\mathcal{F}_{k-obj,sim}}$, which transforms the splat in $\mathcal{S}^k_{obj}$ frame to simulator frame. Given the position $p^k_t \in s_t$ and orientation $R^k_t \in s_t$ can be used to calculate the transformation of object T^{k-obj}_{fk} from its original simulator frame $\mathcal{F}_{k-obj,sim}$. Using T^{k-obj}_{fk} we can get the object's 3D Gaussians in $\mathcal{S}_{real}$ frame with the transformation :

$$T = (T^{\mathcal{F}_{splat}}_{\mathcal{F}_{robot}})^{-1} \cdot T^{k-obj}_{fk} \cdot T^{\mathcal{F}_{k-obj,splat}}_{\mathcal{F}_{k-obj,sim}} \tag{4}$$

Once the transformation for the object is calculated, we can again use Eq. 1 and Eq. 2 to transform the 3D Gaussians related to the object. Then we use the Gaussian Splatting rendering framework [8] to render the object at its new position and orientation.

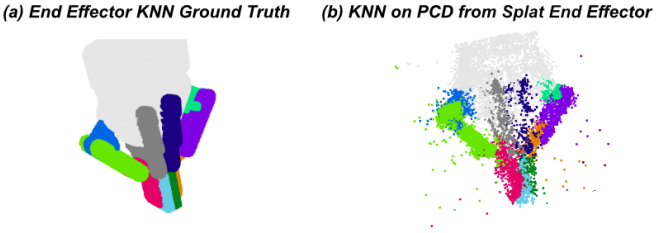

Fig. 4: We use a KNN-based classifier for segmenting links for articulated objects like parallel jaw grippers. We train a KNN model with the ground truth point labeling from the URDF model of the end effector.

E. Articulated Object

While CAD axis-aligned bounding boxes allow straightforward segmentation of robot links, certain objects, such as parallel jaw grippers, present challenges due to their misalignment with standard axes, that is, the gripper links are not neatly segmented out by just using bounding boxes in the 3D space. To address this, we employ a ground truth K-Nearest Neighbour (KNN) classifier trained on labeled simulator point clouds as in Fig. 4 (a), which enables inference of the link class for each 3D Gaussian in the aligned splat as shown in Fig. 4 (b).

F. 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. IV-C, IV-D to get the demonstration for our policy to learn from $\tau_{\mathcal{G}} = \{(I^{sim}_1, a_1), (I^{sim}_2, a_2), \ldots, (I^{sim}_T, a_T)\}$. This data is used by policy to predict actions from the synthetically generated images.

G. Policy Training and Deployment

For learning from the generated demonstrations $\tau_{\mathcal{G}}$ in the simulator, we employ Diffusion Policy [49], [50], 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 [51] 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.

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 I: 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.

V. 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. Additionally, we assess Sim2Sim performance by training and evaluating policies entirely within the *SplatSim*, allowing us to quantify the degradation in performance during Sim2Real transfer. Lastly, we investigate the effects of data augmentation on the transfer process and evaluate the visual fidelity of the photorealistic renderings generated by the *SplatSim* framework.

A. 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 I, 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.

B. 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 I, 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 [52] with deployment on an NVIDIA RTX 3080Ti GPU for the Diffusion Policy [49].

1) T-Push Task: The T-Push task, popularized by Diffusion Policy [49], captures the dynamics of non-prehensile manipulation, which involves controlling both object motion and contact forces. For training, a human expert collected 160 demonstrations in simulation using the Gello teleoperation [48]. While testing, the robot started from a random location and achieved a 90% success rate (36/40 trials) in zero-shot Sim2Real transfer as shown in Table I. This result shows the effectiveness of our framework in handling the dynamics of pushing without fine-tuning on real-world demonstrations. Additionally, the performance of our method is comparable to Real2Real (40/40) and Sim2Sim (40/40).

2) Pick-Up-Apple Task: The Pick-Up-Apple task involves grasping and manipulating the full pose of an object (i.e., position and orientation) in 3D. This task was designed to evaluate the robot's grasping capabilities when trained using our simulated renderings. A motion planner, leveraging privileged state information from the simulator (accurate position and orientation of each rigid body in the scene), generated 400 demonstrations with randomized end-effector positions and orientations. During real-world trials, our policy achieved a 95% success rate (38/40 trials) in zero-shot Sim2Real transfer, as shown in Table I.

3) Orange on Plate Task: In this task the robot has to pick up an orange and place it on a plate. In simulation, a motion planner with access to privileged information, generated 400 demonstrations. The end-effector position and initial gripper state were randomized during training. During testing, the robot always started from a home position. We achieved a 90% success rate (36/40 trials) in zero-shot Sim2Real transfer.

4) Assembly Task: In this task the robot has to put a cuboid block on top of another cuboid. The robot starts at the home position with the green cube already grasped and has to place it on top of the red cube. The task is particularly tough since the robot has to make a precise placement otherwise the

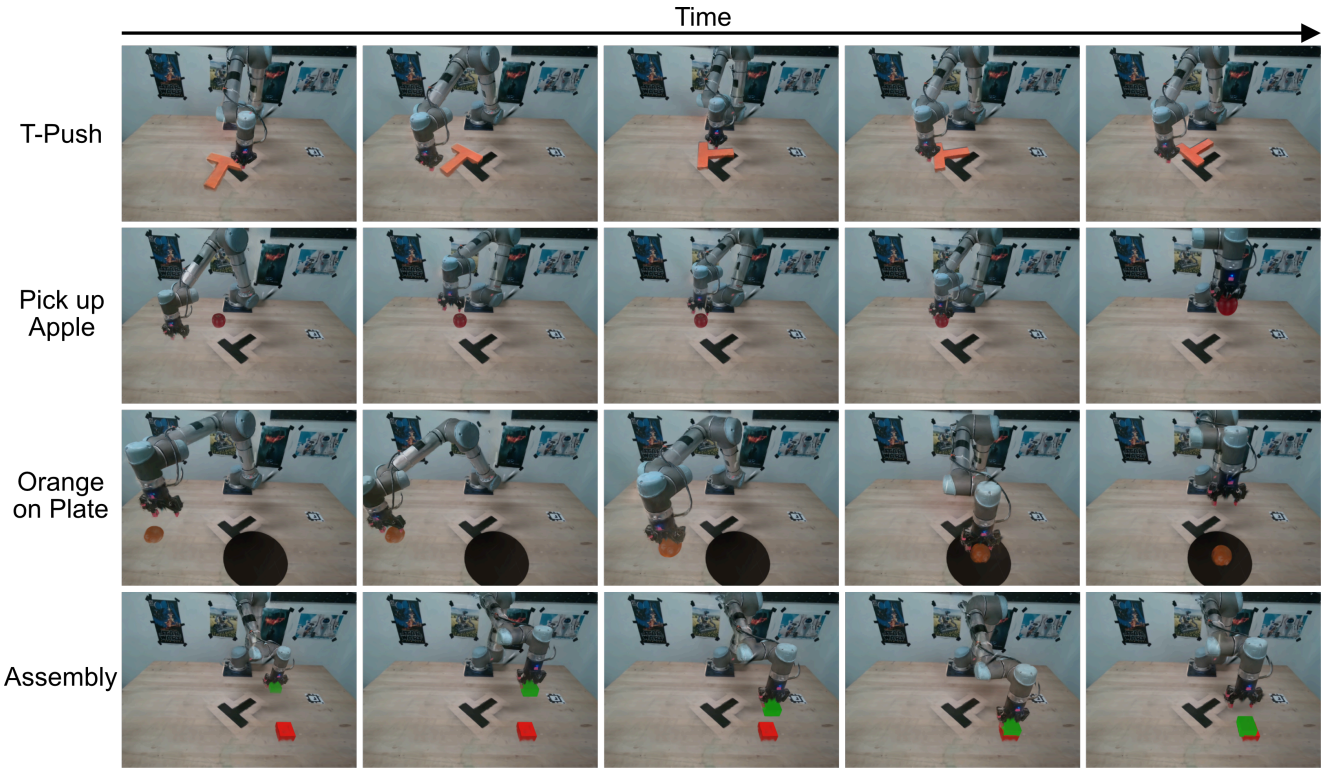

Fig. 5: SplatSim Rollout: Renderings from our SplatSim framework across four different manipulation tasks.

cube will fall and will lead to a failure case. Our Sim2Real policy achieved a performance of 70% (28/40 trials) on this task, compared to 95% on Sim2Sim and 90% on Real2Real.

C. Quantifying Robot Renderings

We quantitatively evaluate the accuracy of rendered robot images at various joint configurations by comparing them with the real-world images. We assess the quality of the robot's renderings across 300 different robot joint angles. To measure the similarity between the rendered and real-world images, we employ two metrics commonly used in image rendering assessment: Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM). Despite the variations in joint configurations, the renderings achieve an average PSNR of 22.62 and an SSIM of 0.7845, indicating that the simulated images closely approximates the visual quality of the real-world RGB observations.

D. Effect of Augmentations

To quantify the impact of data augmentations on the Sim2Real performance of our policy, we conducted experiments comparing policies trained with and without augmentations. While the Diffusion-Policy performs effectively without augmentations in consistent environments (e.g., Sim2Sim or Real2Real scenarios), transferring a policy trained in simulation to the real world introduces domain shifts that necessitate additional robustness as the renderings can't capture dynamic details like changing reflections and shadows. We incorporated augmentations such as random noise addition, Color Jitter, and random erasing during training to address these shifts. These augmentations improve the performance of the policy from 21% to 86.25% across four tasks in Sec. V-B.

VI. 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. In the future, our plan is to combine our framework with reinforcement learning-based methods to acquire more dynamic skills. 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, could greatly benefit from our proposed method.

Acknowledgement

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.

REFERENCES

[1] A. Agarwal, A. Kumar, J. Malik, and D. Pathak, "Legged locomotion in challenging terrains using egocentric vision," *CoRL*, 2022.

[2] X. Cheng, K. Shi, A. Agarwal, and D. Pathak, "Extreme parkour with legged robots," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 11 443–11 450.

[3] H. Qi, A. Kumar, R. Calandra, Y. Ma, and J. Malik, "In-hand object rotation via rapid motor adaptation," in *6th Annual Conference on Robot Learning*, 2022. [Online]. Available: https://openreview.net/forum?id=Xux9gSS7WE0

[4] Y. Yuan, H. Che, Y. Qin, B. Huang, Z.-H. Yin, K.-W. Lee, Y. Wu, S.-C. Lim, and X. Wang, "Robot synesthesia: In-hand manipulation with visuotactile sensing," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 6558–6565.

[5] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. Van Wyk, A. Zhurkevich, B. Sundaralingam, and Y. Narang, "Dextreme: Transfer of agile in-hand manipulation from simulation to reality," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 5977–5984.

[6] Z.-H. Yin, B. Huang, Y. Qin, Q. Chen, and X. Wang, "Rotating without seeing: Towards in-hand dexterity through touch." in *Robotics: Science and Systems*, K. E. Bekris, K. Hauser, S. L. Herbert, and J. Yu, Eds., 2023. [Online]. Available: http://dblp.uni-trier.de/db/conf/rss/rss2023.html#YinHQCW23

[7] T. G. W. Lum, M. Matak, V. Makoviychuk, A. Handa, A. Allshire, T. Hermans, N. D. Ratliff, and K. V. Wyk, "Dextrah-g: Pixels-to-action dexterous arm-hand grasping with geometric fabrics," 2024. [Online]. Available: https://arxiv.org/abs/2407.02274

[8] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, July 2023. [Online]. Available: https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

[9] H. Yu, "On sim2real transfer in robotics," https://www.haonanyu.blog/post/sim2real/, 2024, blog post.

[10] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2021.

[11] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg, "Orbit: A unified simulation framework for interactive robot learning environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3740–3747, 2023.

[12] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control." in *IROS*. IEEE, 2012, pp. 5026–5033. [Online]. Available: http://dblp.uni-trier.de/db/conf/iros/iros2012.html#TodorovET12

[13] C. Aguero, N. Koenig, I. Chen, H. Boyer, S. Peters, J. Hsu, B. Gerkey, S. Paepcke, J. Rivero, J. Manzo, E. Krotkov, and G. Pratt, "Inside the virtual robotics challenge: Simulating real-time robotic disaster response," *Automation Science and Engineering, IEEE Transactions on*, vol. 12, no. 2, pp. 494–506, April 2015.

[14] J. Gu, F. Xiang, X. Li, Z. Ling, X. Liu, T. Mu, Y. Tang, S. Tao, X. Wei, Y. Yao, X. Yuan, P. Xie, Z. Huang, R. Chen, and H. Su, "Maniskill2: A unified benchmark for generalizable manipulation skills," in *International Conference on Learning Representations*, 2023.

[15] T. Mu, Z. Ling, F. Xiang, D. Yang, X. Li, S. Tao, Z. Huang, Z. Jia, and H. Su, "Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations." in *NeurIPS Datasets and Benchmarks*, J. Vanschoren and S.-K. Yeung, Eds., 2021. [Online]. Available: http://dblp.uni-trier.de/db/conf/nips/neurips2021db.html#MuLXYLLTHJ021

[16] D. Hoeller, N. Rudin, D. Sako, and M. Hutter, "Anymal parkour: Learning agile navigation for quadrupedal robots," 2023. [Online]. Available: https://arxiv.org/abs/2306.14874

[17] Z. Zhuang, Z. Fu, J. Wang, C. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao, "Robot parkour learning," in *Conference on Robot Learning (CoRL)*, 2023.

[18] Z. Fu, X. Cheng, and D. Pathak, "Deep whole-body control: Learning a unified policy for manipulation and locomotion," in *Proceedings of The 6th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, K. Liu, D. Kulic, and J. Ichnowski, Eds., vol. 205. PMLR, 14–18 Dec 2023, pp. 138–149. [Online]. Available: https://proceedings.mlr.press/v205/fu23a.html

[19] I. Radosavovic, T. Xiao, B. Zhang, T. Darrell, J. Malik, and K. Sreenath, "Learning humanoid locomotion with transformers," *arXiv:2303.03381*, 2023.

[20] A. Kumar, Z. Li, J. Zeng, D. Pathak, K. Sreenath, and J. Malik, "Adapting rapid motor adaptation for bipedal robots," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 1161–1168.

[21] B. Eisner, H. Zhang, and D. Held, "FlowBot3D: Learning 3D Articulation Flow to Manipulate Articulated Objects," in *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, June 2022.

[22] H. Zhang, B. Eisner, and D. Held, "Flowbot++: Learning generalized articulated objects manipulation via articulation projection," in *7th Annual Conference on Robot Learning*, 2023.

[23] E. Su, C. Jia, Y. Qin, W. Zhou, A. Macaluso, B. Huang, and X. Wang, "Sim2real manipulation on unknown objects with tactile-based reinforcement learning," 2024. [Online]. Available: https://arxiv.org/abs/2403.12170

[24] W. Zhou, B. Jiang, F. Yang, C. Paxton, and D. Held, "Hacman: Learning hybrid actor-critic maps for 6d non-prehensile manipulation," in *7th Annual Conference on Robot Learning*, 2023.

[25] A. Loquercio, E. Kaufmann, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza, "Deep drone racing: From simulation to reality with domain randomization," *IEEE Transactions on Robotics*, 2019.

[26] B. Huang, Y. Chen, T. Wang, Y. Qin, Y. Yang, N. Atanasov, and X. Wang, "Dynamic handover: Throw and catch with bimanual hands," in *7th Annual Conference on Robot Learning*, 2023. [Online]. Available: https://openreview.net/forum?id=dgwvY3H8PAS

[27] K.-H. Zeng, Z. Zhang, K. Ehsani, R. Hendrix, J. Salvador, A. Herrasti, R. Girshick, A. Kembhavi, and L. Weihs, "Poliformer: Scaling on-policy rl with transformers results in masterful navigators," *arXiv*, 2024.

[28] B. Jia, Y. Chen, H. Yu, Y. Wang, X. Niu, T. Liu, Q. Li, and S. Huang, "Sceneverse: Scaling 3d vision-language learning for grounded scene understanding," *arXiv preprint arXiv:2401.09340*, 2024.

[29] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. Chaplot, O. Maksymets, A. Gokaslan, V. Vondrus, S. Dharur, F. Meier, W. Galuba, A. Chang, Z. Kira, V. Koltun, J. Malik, M. Savva, and D. Batra, "Habitat 2.0: Training home assistants to rearrange their habitat," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[30] K. Rao, C. Harris, A. Irpan, S. Levine, J. Ibarz, and M. Khansari, "Rl-cyclegan: Reinforcement learning aware simulation-to-real," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 154–11 163.

[31] M. T. Villasevil, 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," in *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024.

[32] Z. Yuan, T. Wei, S. Cheng, G. Zhang, Y. Chen, and H. Xu, "Learning to manipulate anywhere: A visual generalizable framework for reinforcement learning," in *8th Annual Conference on Robot Learning*, 2024. [Online]. Available: https://openreview.net/forum?id=jart4nhCQr

[33] 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," in *ECCV*, 2020.

[34] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, vol. 41, no. 4, pp. 102:1–102:15, Jul. 2022. [Online]. Available: https://doi.org/10.1145/3528223.3530127

[35] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan, "Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields," *ICCV*, 2021.

[36] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, "pixelNeRF: Neural radiance fields from one or few images," in *CVPR*, 2021.

[37] J. Luiten, G. Kopanas, B. Leibe, and D. Ramanan, "Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis," in *3DV*, 2024.

[38] G. Wu, T. Yi, J. Fang, L. Xie, X. Zhang, W. Wei, W. Liu, Q. Tian, and X. Wang, "4d gaussian splatting for real-time dynamic scene rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2024, pp. 20 310–20 320.

[39] Y. Chen, C. Gu, J. Jiang, X. Zhu, and L. Zhang, "Periodic vibration gaussian: Dynamic urban scene reconstruction and real-time rendering," *arXiv:2311.18561*, 2023.

[40] J. Bae, S. Kim, Y. Yun, H. Lee, G. Bang, and Y. Uh, "Per-gaussian embedding-based deformation for deformable 3d gaussian splatting," in *European Conference on Computer Vision (ECCV)*, 2024.

[41] T. Xie, Z. Zong, Y. Qiu, X. Li, Y. Feng, Y. Yang, and C. Jiang, "Physgaussian: Physics-integrated 3d gaussians for generative dynamics," *arXiv preprint arXiv:2311.12198*, 2023.

[42] O. Shorinwa, J. Tucker, A. Smith, A. Swann, T. Chen, R. Firoozi, M. D. Kennedy, and M. Schwager, "Splat-MOVER: Multi-stage, open-vocabulary robotic manipulation via editable gaussian splatting," in *8th Annual Conference on Robot Learning*, 2024. [Online]. Available: https://openreview.net/forum?id=8XFT1PatHy

[43] M. Ji, R.-Z. Qiu, X. Zou, and X. Wang, "Graspsplats: Efficient manipulation with 3d feature splatting," in *8th Annual Conference on Robot Learning*, 2024. [Online]. Available: https://openreview.net/forum?id=pPhTsonbXq

[44] B. P. Duisterhof, Z. Mandi, Y. Yao, J.-W. Liu, J. Seidenschwarz, M. Z. Shou, D. Ramanan, S. Song, S. Birchfield, B. Wen, and J. Ichnowski, "Deformgs: Scene flow in highly deformable scenes for deformable object manipulation," in *The 16th International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2024.

[45] J. Abou-Chakra, K. Rana, F. Dayoub, and N. Suenderhauf, "Physically embodied gaussian splatting: A realtime correctable world model for robotics," in *8th Annual Conference on Robot Learning*, 2024. [Online]. Available: https://openreview.net/forum?id=AEq0onGrN2

[46] H. Lou, Y. Liu, Y. Pan, Y. Geng, J. Chen, W. Ma, C. Li, L. Wang, H. Feng, L. Shi, L. Luo, and Y. Shi, "Robo-gs: A physics consistent spatial-temporal model for robotic arm with hybrid representation," 2024. [Online]. Available: https://arxiv.org/abs/2408.14873

[47] A. Quach, M. Chahine, A. Amini, R. Hasani, and D. Rus, "Gaussian splatting to real world flight navigation transfer with liquid networks," in *8th Annual Conference on Robot Learning*, 2024. [Online]. Available: https://openreview.net/forum?id=ubq7Co6Cbv

[48] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel, "Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators," 2023.

[49] 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.

[50] 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.

[51] 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)*, 2023, pp. 9362–9369.

[52] 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)*, 2017, pp. 1267–1276.