Real2Gen: Imitation Learning from a Single Human Demonstration with Generative Foundational Models

Nick Heppert*[1,2], Minh Quang Nguyen*[1], Abhinav Valada[1]

***Abstract*— Imitation learning is a common paradigm for teaching robots new tasks. However, collecting robot demonstrations through teleoperation or kinesthetic teaching can be tedious and time-consuming, slowing down training data collection for policy learning. On the other hand, while transfer to the robot can be non-trivial, directly demonstrating a task using our human embodiment is much easier, and data is available in abundance. In this work, we propose Real2Gen to train a manipulation policy from a single human demonstration. Real2Gen extracts required information from the demonstration, transfers it to a simulation environment, where a programmable expert agent can demonstrate the task arbitrarily many times, generating an unlimited amount of data to train a flow matching policy. We evaluate Real2Gen on human demonstrations from three different real-world tasks and compare it to a recent baseline. Real2Gen shows an average increase in the success rate of 26.6% and better generalization of the trained policy due to the abundance and diversity of training data. We make the data, code, and trained models publicly available at real2gen.cs.uni-freiburg.de.**

I. Introduction

In the future, we want robots to easily be able to perform new manipulation skills with very little to no overhead in teaching them. To achieve this, in recent years, imitation learning has crystallized to be one of the main paradigms to teach a robot skills [1], [2]. Classically, imitation learning uses a dataset consisting of demonstrations of aligned robot observations and actions collected through tele-operation [3] or kinesthetic teaching [4]. While the user operates the robot, the robot actively records its observations and actions to form a training demonstration dataset. We call these type of demonstrations robot demonstrations. One advantage of robot demonstrations is the embodiment alignment of the training data with the robot. On the other hand, while collecting the data itself is not only time-consuming, tele-operating also requires a skilled operator. For kinesthetic teaching, the operator needs to be physically present in the scene and can block cameras, for example.

A recent trend revolves around the tedious collection process by trying to directly learn from demonstrations of humans in their own embodiment [5], [6]. We coin these types of demonstrations human demonstrations. While human demonstrations compared to robot demonstrations are easy to collect or are even widely available on the internet [7], they impose a significant challenge as the embodiment between human and robot differs. This challenge was tackled by researchers, for example, through using hand-crafted heuristics to match a human hand to the robot gripper [8] or through constraining the possible human movements in a demonstration to the robots capabilities [9]. Differently, works like DITTO [10] and ORION [11] proposes to close the embodiment gap through finding explicit object correspondences between the human demonstration and the robot environment at inference time, allowing for the transfer of motion in an object-centric manner.

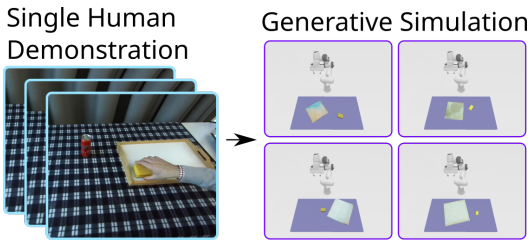

Fig. 1: Overview of Real2Gen. Real2Gen takes a single human demonstration as input, produces simulatable meshes using 3D generative foundational models which can be used in an generative simulation setup.

In this work, we aim to combine the best of human and robot demonstrations and propose a "Reality to Simulation" (Real2Sim) [12] method that processes a single-human human demonstration to set up a simulator that automatically allows us to collect robot demonstrations. Thus, we leverage the ease of collecting demonstrations using our human embodiment but train a policy directly on data using robot embodiment. While previous Real2Sim approaches needed to scan the environment in an offline step [13], [14], we use 3D generative foundational models like Point-E [15] or Zero-1-to-3 [16] to directly generate 3D assets of task-relevant objects given their appearance in the human demonstration. We then use a DINO-like matcher [17] to align the 3D assets to the human demonstration to retrieve their scale and canonical orientation. Similarly to other simulation works [18], [19], we first randomly sample poses of the assets and second, use a scripted expert agent to generate robot demonstrations. Last, we train a flow matching policy [20] on the collected robot demonstrations. We evaluate our policy in a simulation environment consisting of unseen object instances retrieved from a curated large-scale 3D object set such as Objaverse [21]. Compared to the DITTO [10] baseline, we show an average success rate improvement of 26.6%.

Our main contributions are as follows:

* These authors contributed equally and are ordered alphabetically.

[1]Department of Computer Science, University of Freiburg, Germany, [2]Zuse School ELIZA

This work was partially funded by the Carl Zeiss Foundation with the ReScaLe project. Nick Heppert is supported by the Konrad Zuse School of Excellence in Learning and Intelligent Systems (ELIZA) through the DAAD programme Konrad Zuse Schools of Excellence in Artificial Intelligence, sponsored by the Federal Ministry of Education and Research.
The authors would like to thank Eugenio Chisari and Max Argus for discussion.

- A framework to process human demonstrations for generating robot demonstrations.
- An extensive evaluation of mesh generation rates.
- Code, trained models, and data will be made publicly available at real2gen.cs.uni-freiburg.de.

II. Related Work

This section reviews prior works on learning robotic manipulation policies from human demonstrations and scaling robot learning data through generative or procedural simulations.

A. Robot Manipulation from Human Demonstrations

A key challenge in learning from human demonstrations is bridging the gap between the human and robot embodiment. Approaches range from object trajectories [10], [11], affordances [8], [22], and pixel-level motion [23], [24].

Trajectories: Methods like DITTO [10] and ORION [11] detect and track objects in human demonstrations, extracting object-centric trajectories. ORION additionally segments demonstrations into key frames and encodes object interactions in Open-World Object Graphs.

Affordance: Affordance-based methods, such as RAM [25] and HRP [22], serve as intermediate representations for task transfer. Different from trajectories, affordance formulations usually consist of a region and simple motion direction or horizon [26].

Pixel-Level: Track2Act [24] and Im2Flow2Act [23] infer motion trajectories from pixels, bypassing explicit object detection. These models can be trained on large-scale internet data and transferred to the robot through a learned policy.

Real2Gen builds on these advances by generating diverse training data from extracted trajectories and key frames.

B. Robot Learning via Procedural and Generative Simulation

To execute trajectories using the robot embodiment, the previously discussed works use either motion planning [10], [11] or closed-loop policies [23], [24], but combining both remains a challenge. To combine the advantages of both, the ease and determinism of motion planning and the robustness of closed-loop policy execution, automatic robot demonstration generation in real-world-aligned simulations is a promising candidate [18], [19]. LLMs can enable task generation and reward specification in simulated environments [27], leveraging curated 3D assets [28]. To reduce human effort for curating, Gen2Sim [29] enhances object diversity by generating 3D assets from 2D images, while GRS [30] uses VLMs to match real-world objects to existing 3D datasets.

RoboGen [31] integrates either generating meshes or using a 3D dataset, selecting between reinforcement learning, trajectory optimization, or motion planning for different tasks. Unlike prior works, Real2Gen employs generative 3D models for asset creation and automatic scene alignment, eliminating the need for manual curation or textual task descriptions.

III. Technical Approach

In this section, we detail our approach, Real2Gen. We start by describing the pre-processing procedure (in Sec. III-A), how we generate 3D assets (in Sec. III-B) and robot training data (in Sec. III-C), and lastly detail our policy learning approach (in Sec. III-D). An overview is presented in Fig. 2.

A. Pre-Processing Human Demonstrations

The input to Real2Gen is a single human demonstration consisting of a sequence of T RGB-D images o_h. We pre-process the sequence o_h using an object trajectory extractor DITTO [10], but other options such as ORION [11], are also possible. Similarly to these methods, we assume there is a primary object p moving, either in free-form or depending on another secondary goal object s. The goal of the pre-processing step is to extract segmentation masks of all relevant objects, i.e., m_p and m_s if applicable, in the first RGB image I_0 as well as the object-centric trajectory of the primary object J_p consisting of relative poses. By segmenting the first RGB image I_0 using m_p and m_s masks, we get unobstructed object-centric RGB reference images I_p and I_s of our primary and secondary objects, respectively, before the human interacts with them.

B. Asset Generation

Given one of the reference object RGB images I_f, i.e., either I_p or I_s if applicable, we use an off-the-shelf 3D generative foundational model, specifically Point-E [15] combined with marching cubes, to generate a raw 3D mesh m^c. We decided on this combination because of its ease of use, but other models such as Zero-1-To-3 [16], Stable-Dreamfusion [32], or Shape-E [33] are also utilizable. As these models return object meshes in a canonical non-metric frame, they are not directly usable in a robot simulation. Although VLMs were previously used to verify the mesh and guess a metric dimension [29], we use our provided human demonstration and match the generated mesh to it. As our generated mesh and the object in the human demonstration share semantic similarities but are not the exact instance, methods like FoundationPose [34] are not applicable here. Instead, we propose to use a DINO [35] based feature matcher Zero-Shot-Pose (ZSP) [17] to first generate correspondences and then subsequently align the correspondences through a 7-DoF affine transformation with a closed form solution [36]. We detail this approach in the following.

Given a generated mesh from our 3D generative foundational model m^c, we render N_v views V of it using spherical Fibonacci sampling [37] to sample the polar and azimuthal angles, similarly as done by Giammaorino *et al.* [38]. The Fibonacci sampling procedure ensures maximum diversity in views compared to densely sampling the polar and azimuthal angles. For each view $v_r = \{I_r, D_r\}$, we render an RGB image I_r and a depth image D_r. We collectively give all views V and the initial reference image I_f to ZSP [17]. ZSP [17] then first extracts descriptors for each view's RGB image I_r, matches these descriptors to the reference image I_f to select the view $\hat{v}$ with the top-K similar descriptors. Second, ZSP [17] uses the most similar descriptors as correspondences and projects them in 3D using the selected views $\hat{v}$ depth image $\hat{D}$ and the reference depth image D_f. Lastly, ZSP [17] solves a closed-form seven degree of freedoms (7-DoF) least squares problem [36] to estimate

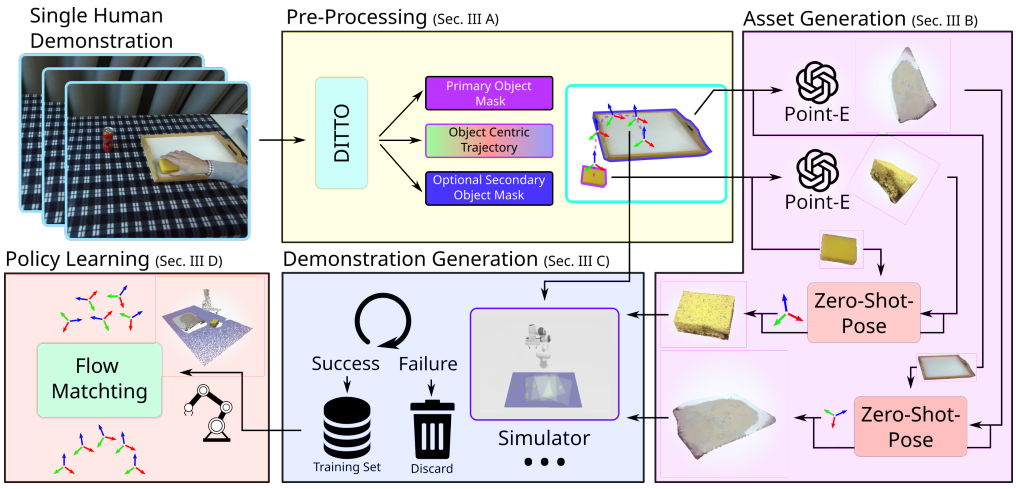

Fig. 2: Technical Approach of Real2Gen. Real2Gen uses a single human demonstration as input, consisting of a sequence of RGB-D images. We pre-process (Sec. III-A) these images using DITTO [10] to retrieve a primary and, if applicable, a secondary object mask as well as an object-centric trajectory of the object. In the second step, asset generation (Sec. III-B), we pass object images to Point-E [15] to generate 3D meshes in a canonical space. We then use Zero-Shot-Pose (ZSP) [17] to scale and align the meshes to the human demonstration. We then use the generated meshes combined with object-centric trajectories to set up a simulation (Sec. III-C). Using grasp and motion planning, we use the simulation to generate an expert dataset of policy rollouts. In the last step, policy learning (Sec. III-D), we use the collected dataset to train a conditional flow matching policy [20].

a full affine transformation ${}^{c}\mathbf{T}_{m}$ to transform the canonical mesh m^c to a metric mesh m^m of which the up-orientation matches the human demonstration. Despite using the top-K similar descriptors across all views, there might be outliers present. Thus, the estimate is done multiple times using RANSAC. This process can be repeated arbitrarily many times to continuously generate new and unseen meshes usable in a robotic simulation. In practice, though, we generate a fixed amount of N_m meshes M before continuing with our demonstration generation. We additionally verify all matches before proceeding, which, with improvements in 3D generative foundational models, will become obsolete. Throughout our experiments, we sampled $N_v = 80$ views and used the top-30 descriptors in ZSP [17].

C. Demonstration Generation

After we generate a set of meshes M_p for the primary object and if applicable, a set M_s for the secondary object, we use them in a simulation environment to generate large-scale robot data. Specifically, we use SAPIEN [39] as our simulator with a Franka Panda Robot mounted on a table. To observe the scene, we set up an external RGB-D camera as well as attach an RGB-D wrist camera to the robot.

To generate a robot demonstration, we first sample random meshes, one for the primary object $m_p^m \in M_p$ and if needed, one for the secondary object $m_s^m \in M_s$. We then sample random poses, constraining the meshes to be on the table and randomly rotated around the up-axis. Next, using our privileged simulation information, we analytically calculate grasps on the primary object mesh using antipodality constraints[1] and pick a random grasp. At this point, we explicitly differentiate between tasks only moving the primary object and tasks with an additional secondary object. For tasks that only involve a primary object, we apply the extracted object-centric trajectory J_p to the grasp. For tasks consisting of an additional secondary object, we constrain the primary object to be placed in the center of the secondary object[2] approached through a bottleneck pose above the table. To finally execute the task, we use a motion planner[3]. During execution, we record all observations o_r and the robots proprioceptive states as end-effector poses. After the roll-out is completed, using the proposed pose error metric by GraspNet [40], we compare the distance between the actual pose of the primary object and the expected pose to determine whether the roll-out was a success or failure. Only successful roll-outs d are added to our demonstration dataset D.

D. Policy Learning

The last component of Real2Gen consists of learning a manipulation policy given our previously generated demonstration dataset D. As our policy model we chose to use PointFlowMatch [20] a flow matching based imitation learning method. Similar to PointFlowMatch [20], our policy learns to move a Gaussian distribution to a target distribution being our ground truth actions where the process is conditioned on the current robot observations. The robot observations are the last two point clouds encoded using a PointNet-encoder [41] as well as the last two robot proprioceptive states, in our case, the end-effector pose in $SE(3)$. We use an aggregated point cloud from the wrist camera and one external camera by concatenating the points and randomly downsample it to a fixed size. Our inferred actions are future end-effector poses with a fixed horizon length H. See App. A for our hyperparameter setup.

[1]In practice, this process was done in a pre-processing step for each mesh and stored. At demonstration generation time we only perform collision checks to filter out grasps.

[2]We plan to extend our work to use the final relative poses between the primary and secondary object.

[3]https://motion-planning-lib.readthedocs.io/latest/

Method	Sponge on Tray	Coke on Tray	Paperroll upright	Mean SR (↑)
DITTO [10]	$6.3^{\pm 2.1}$	$26.0^{\pm 3.6}$	$0.3^{\pm 0.5}$	$10.9^{\pm 2.1}$
DITTO [10] w/ ZSP [17]	$4.3^{\pm 1.2}$	$19.7^{\pm 3.8}$	$0.7^{\pm 0.6}$	$8.2^{\pm 1.8}$
Real2Gen (ours)	$\mathbf{41.3}^{\pm 4.5}$	$\mathbf{46.3}^{\pm 6.4}$	$\mathbf{25.0}^{\pm 1.0}$	$\mathbf{37.5}^{\pm 3.0}$

TABLE I: Performance comparison of Real2Gen against different baseline methods. We evaluate all methods on three different tasks and report per-task success rate as well as overall mean success rate [%] (↑).

Source	Available Meshes	100 Mesh Pre-Selection	Matching Successful and Task Relevant†
Point-E [15] (ours)	∞	Random	**54%**
Objaverse [21]	690	Most viewed or All	19%
		Random or All	18%

TABLE II: Comparison of Mesh Generation. We compare the number of resulting meshes when using our proposed generative way vs. using a large-scale object dataset. For an extended version, see App. C. †*We report the total percentage from the 100 selected meshes per category, or if fewer than 100 meshes are available from all available ones.*

IV. Experimental Evaluation

In our experiments, we quantitatively evaluate how well Real2Gen can transfer the human demonstration to a robot compared to a DITTO [10] baseline. We also investigate the difference in quality and effort when using a 3D generative foundational model over a large-scale object dataset, such as Objaverse [21].

A. Quantitative Evaluation

For our evaluation, we selected three tasks from DITTO [10] and, following DITTO [10], a single human demonstration for each task. We generate five meshes and 800 demonstrations. We use a simulator with the same fixed seeds across all evaluations for a fair and reproducible comparison. We evaluate Real2Gen against DITTO [10] baselines as well as ablate Real2Gen.

Evaluation Setup: To set up and align our evaluation simulator with our tasks, we use unseen realistic 3D meshes from Objaverse [21]. All objects were manually checked and scaled to have reasonable geometry and realistic sizes. For the `Sponge on Tray` and `Can on Tray`-task we select five sponge, five can and seven tray meshes. For the `Paperroll upright`-task, we select five paperroll meshes. As for generating robot demonstrations (see Sec. III-B), during each simulation run, we randomly select meshes according to the task and spawn them at random poses. To perform the evaluation, we select three random seeds. For each seed, we evaluate each method 100 times and record the successes of the roll-outs. We evaluate the successes as described in Sec. III-C. As done in [20], [42], we report the mean and standard deviation across all seeds.

Baseline Comparisons: We compare Real2Gen against two variants of DITTO [10]. First, the original variant using LoFTR [43] to match the live observation to the demonstration. This is an unfair comparison as DITTO [10] was designed to be faced with the same object instance as in the demonstration. Thus, we extend DITTO [10] by replacing the LoFTR [43] matching step with ZSP [17] which should enable DITTO [10] to work across categories. The results of our experiment are reported in Tab. I. When comparing Real2Gen to DITTO [10] with and without ZSP [17], we see that the overall performance is better. We assume the difference stems from having only a single image available, which increases difficulty for matching. To our surprise, the original DITTO [10] baseline outperforms the variant with ZSP [17]. Additionally, we observe that grasp and motion planning fail quite often, aligning with the reported results in DITTO [10], whereas our learned policy produces more robust results.

Ablation Studies: We additionally perform ablation studies of Real2Gen to study the effect of the number of generated meshes and demonstrations. Results are reported in the appendix in App. B.

B. Comparison of Mesh Generation

To highlight the effectiveness of our proposed automatic mesh generation and matching procedure, we compare the human effort needed to perform manual mesh retrieval from a large database, as done, for instance, in Scaling Up And Distill Down [19]. To have representative results, we set the goal to generate 100 object meshes. For Real2Gen, this process is straightforward, and we run the asset generation a hundred times. For the object set comparison, we query Objaverse [21] to retrieve meshes with the tag of either `sponge`, `coke/coca can`, `tray`, or `paperroll`. If there are more than a hundred meshes available, we evaluate using the hundred most viewed meshes and a hundred random meshes. We additionally report the total meshes available. We then apply the same matching procedure as for the generated meshes. Lastly, we manually go through all of the generated and retrieved object set meshes and classify whether they are task relevant. For the retrieved meshes, this is particularly important as they are not anchored to the human demonstration, e.g., for `sponge`, sometimes the cartoon character `Sponge Bob` is returned. We report the results in Tab. II. Overall, following our proposed way to generate meshes through a generative model results in almost 3x more available meshes on average while also being able to generate an infinite amount.

V. Conclusion

In this work, we investigated the use of 3D generative foundational models to transfer a single human demonstration to simulation. We showed that with Real2Gen, more robust policies are learned with an increase in success rate of 26.6% against baselines. We additionally analyzed the effort needed to generate task-relevant and usable simulation CAD models.

In the future, we see potential to extend our work to enable more tasks with constrained movement like interacting with articulated objects, e.g., sampled from an object prior like in CARTO [44]. Additionally, we want to further study the difference between generated meshes and large-scale object sets by comparing Real2Gen against a policy trained on objects from an object set. We are also interested in testing the capabilities of VLMs as described in Gen2Sim [29].

REFERENCES

[1] M. Zare, P. M. Kebria, A. Khosravi, and S. Nahavandi, "A survey of imitation learning: Algorithms, recent developments, and challenges," *IEEE Transactions on Cybernetics*, 2024.

[2] C. Celemin, R. Pérez-Dattari, E. Chisari, G. Franzese, L. de Souza Rosa, R. Prakash, Z. Ajanović, M. Ferraz, A. Valada, J. Kober *et al.*, "Interactive imitation learning in robotics: A survey," *Foundations and Trends® in Robotics*, vol. 10, no. 1-2, pp. 1–197, 2022.

[3] J. O. von Hartz, T. Welschehold, A. Valada, and J. Boedecker, "The art of imitation: Learning long-horizon manipulation tasks from few demonstrations," *IEEE Robotics and Automation Letters*, 2024.

[4] D. Honerkamp, H. Mahesheka, J. O. von Hartz, T. Welschehold, and A. Valada, "Whole-body teleoperation for mobile manipulation at zero added cost," *IEEE Robotics and Automation Letters*, 2025.

[5] T. Welschehold, C. Dornhege, and W. Burgard, "Learning manipulation actions from human demonstrations," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 3772–3777.

[6] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg, "Concept2robot: Learning manipulation concepts from instructions and human demonstrations," *Int. J. Rob. Res.*, vol. 40, no. 12-14, pp. 1419–1434, 2021.

[7] D. Shan, J. Geng, M. Shu, and D. F. Fouhey, "Understanding human hands in contact at internet scale," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 9869–9878.

[8] G. Papagiannis, N. Di Palo, P. Vitiello, and E. Johns, "R+ x: Retrieval and execution from everyday human videos," *arXiv preprint arXiv:2407.12957*, 2024.

[9] H. Xiong, Q. Li, Y.-C. Chen, H. Bharadhwaj, S. Sinha, and A. Garg, "Learning by watching: Physical imitation of manipulation skills from human videos," in *Proc. IEEE Int. Conf. on Intel. Rob. and Syst.* IEEE, 2021, pp. 7827–7834.

[10] N. Heppert, M. Argus, T. Welschehold, T. Brox, and A. Valada, "Ditto: Demonstration imitation by trajectory transformation," in *IROS*. IEEE, 2024, pp. 7565–7572.

[11] Y. Zhu, A. Lim, P. Stone, and Y. Zhu, "Vision-based manipulation from single human video with open-world object graphs," *arXiv preprint arXiv:2405.20321*, 2024.

[12] X. Li, J. Li, Z. Zhang, R. Zhang, F. Jia, T. Wang, H. Fan, K.-K. Tseng, and R. Wang, "Robogsim: A real2sim2real robotic gaussian splatting simulator," *arXiv preprint arXiv:2411.11839*, 2024.

[13] M. N. Qureshi, S. Garg, F. Yandun, D. Held, G. Kantor, and A. Silwal, "Splatsim: Zero-shot sim2real transfer of rgb manipulation policies using gaussian splatting," in *Proc. IEEE Int. Conf. on Rob. and Auto.*, 2025.

[14] 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," in *Proc. Rob.: Sci. and Syst.*, 2024.

[15] A. Nichol, H. Jun, P. Dhariwal, P. Mishkin, and M. Chen, "Point-e: A system for generating 3d point clouds from complex prompts," *arXiv preprint arXiv:2212.08751*, 2022.

[16] R. Liu, R. Wu, B. Van Hoorick, P. Tokmakov, S. Zakharov, and C. Vondrick, "Zero-1-to-3: Zero-shot one image to 3d object," in *Proc. Int. Conf. Comput. Vis.*, 2023, pp. 9298–9309.

[17] W. Goodwin, S. Vaze, I. Havoutis, and I. Posner, "Zero-shot category-level object pose estimation," in *Proc. Springer Eur. Conf. Comput. Vis.* Springer, 2022, pp. 516–532.

[18] S. James, Z. Ma, D. Rovick Arrojo, and A. J. Davison, "Rlbench: The robot learning benchmark & learning environment," *IEEE Robotics and Automation Letters*, 2020.

[19] H. Ha, P. Florence, and S. Song, "Scaling up and distilling down: Language-guided robot skill acquisition," in *Proc. Conf. on Rob. Learn.* PMLR, 2023, pp. 3766–3777.

[20] E. Chisari, N. Heppert, M. Argus, T. Welschehold, T. Brox, and A. Valada, "Learning robotic manipulation policies from point clouds with conditional flow matching," *Proc. Conf. on Rob. Learn.*, 2024.

[21] M. Deitke, D. Schwenk, J. Salvador, L. Weihs, O. Michel, E. VanderBilt, L. Schmidt, K. Ehsani, A. Kembhavi, and A. Farhadi, "Objaverse: A universe of annotated 3d objects," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2023, pp. 13 142–13 153.

[22] M. K. Srirama, S. Dasari, S. Bahl, and A. Gupta, "Hrp: Human affordances for robotic pre-training," in *Proc. Rob.: Sci. and Syst.*, Delft, Netherlands, 2024.

[23] M. Xu, Z. Xu, Y. Xu, C. Chi, G. Wetzstein, M. Veloso, and S. Song, "Flow as the cross-domain manipulation interface," in *Proc. Conf. on Rob. Learn.*, 2024.

[24] H. Bharadhwaj, R. Mottaghi, A. Gupta, and S. Tulsiani, "Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation," in *Proc. Springer Eur. Conf. Comput. Vis.*, 2024.

[25] Y. Kuang, J. Ye, H. Geng, J. Mao, C. Deng, L. Guibas, H. Wang, and Y. Wang, "RAM: Retrieval-based affordance transfer for generalizable zero-shot robotic manipulation," in *CORL*, 2024.

[26] S. Bahl, R. Mendonca, L. Chen, U. Jain, and D. Pathak, "Affordances from human videos as a versatile representation for robotics," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2023, pp. 13 778–13 790.

[27] L. Wang, Y. Ling, Z. Yuan, M. Shridhar, C. Bao, Y. Qin, B. Wang, H. Xu, and X. Wang, "Gensim: Generating robotic simulation tasks via large language models," in *Int. Conf. on Learn. Repr.*, 2024.

[28] L. Downs, A. Francis, N. Koenig, B. Kinman, R. Hickman, K. Reymann, T. B. McHugh, and V. Vanhoucke, "Google scanned objects: A high-quality dataset of 3d scanned household items," in *Proc. IEEE Int. Conf. on Rob. and Auto.* IEEE, 2022, pp. 2553–2560.

[29] P. Katara, Z. Xian, and K. Fragkiadaki, "Gen2sim: Scaling up robot learning in simulation with generative models," in *Proc. IEEE Int. Conf. on Rob. and Auto.* IEEE, 2024, pp. 6672–6679.

[30] A. Zook, F.-Y. Sun, J. Spjut, V. Blukis, S. Birchfield, and J. Tremblay, "Grs: Generating robotic simulation tasks from real-world images," *arXiv preprint arXiv:2410.15536*, 2024.

[31] Y. Wang, Z. Xian, F. Chen, T.-H. Wang, Y. Wang, K. Fragkiadaki, Z. Erickson, D. Held, and C. Gan, "Robogen: Towards unleashing infinite data for automated robot learning via generative simulation," in *Int. Conf. on Mach. Learn.* PMLR, 2024, pp. 51 936–51 983.

[32] J. Tang, "Stable-dreamfusion: Text-to-3d with stable-diffusion," 2022, https://github.com/ashawkey/stable-dreamfusion.

[33] H. Jun and A. Nichol, "Shap-e: Generating conditional 3d implicit functions," *arXiv preprint arXiv:2305.02463*, 2023.

[34] B. Wen, W. Yang, J. Kautz, and S. Birchfield, "Foundationpose: Unified 6d pose estimation and tracking of novel objects," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2024, pp. 17 868–17 879.

[35] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2021, pp. 9650–9660.

[36] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 04, pp. 376–380, 1991.

[37] Á. González, "Measurement of areas on a sphere using fibonacci and latitude–longitude lattices," *Mathematical geosciences*, vol. 42, pp. 49–64, 2010.

[38] L. Di Giammarino, B. Sun, G. Grisetti, M. Pollefeys, H. Blum, and D. Barath, "Learning where to look: Self-supervised viewpoint selection for active localization using geometrical information," in *Proc. Springer Eur. Conf. Comput. Vis.*, 2024, pp. 188–205.

[39] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, L. Yi, A. X. Chang, L. J. Guibas, and H. Su, "SAPIEN: A simulated part-based interactive environment," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, June 2020.

[40] H.-S. Fang, C. Wang, M. Gou, and C. Lu, "Graspnet-1billion: A large-scale benchmark for general object grasping," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 11 444–11 453.

[41] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 652–660.

[42] N. Funk, J. Urain, J. Carvalho, V. Prasad, G. Chalvatzaki, and J. Peters, "Actionflow: Equivariant, accurate, and efficient policies with spatially symmetric flow matching," *arXiv preprint arXiv:2409.04576*, 2024.

[43] J. Sun, Z. Shen, Y. Wang, H. Bao, and X. Zhou, "LoFTR: Detector-free local feature matching with transformers," *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2021.

[44] N. Heppert, M. Z. Irshad, S. Zakharov, K. Liu, R. A. Ambrus, J. Bohg, A. Valada, and T. Kollar, "Carto: Category and joint agnostic reconstruction of articulated objects," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2023, pp. 21 201–21 210.

APPENDIX

A. Policy Hyper Parameters Setup

We adopt the training settings and hyperparameters from PointFlowMatch [20]. We randomly downsample the input

Category	Source	Available Meshes	100 Mesh Pre-Selection	Matching Successful†	Matching Successful and Task Relevant†
Sponge	Point-E [15] (ours)	∞	Random	94%	58%
	Objaverse [21]	351	Most viewed	65%	2%
			Random	65%	2%
Tray	Point-E [15] (ours)	∞	Random	60%	34%
	Objaverse [21]	278	Most viewed	20%	6%
			Random	15%	3%
Coke Can	Point-E [15] (ours)(ours)	∞	Random	100%	54%
	Objaverse [21]	41	All	66%	54%
Paper Roll	Point-E [15] (ours)(ours)	∞	Random	95%	70%
	Objaverse [21]	20	All	25%	15%
All	Point-E [15] (ours)	∞	Random	**87%**	**54%**
	Objaverse [21]	690	Most viewed or All	44%	19%
			Random or All	43%	18%

TABLE III: Comparison of Mesh Generation. We compare the number of resulting meshes when using our proposed generative way vs. using a large-scale object dataset. †*We report the total percentage from the 100 selected meshes, or if fewer than 100 meshes are available from all available ones.* d

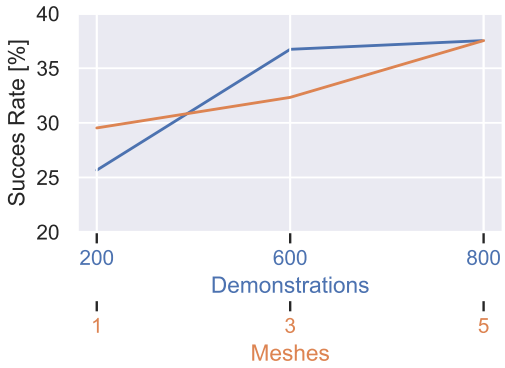

Fig. 3: Results of Ablation Study. We show the average success rate [%] (↑) across all tasks. We either vary the number of demonstrations while using five meshes or we vary the number of meshes using 800 demonstrations.

point cloud to a size of 4096 points. The policy is trained using AdamW optimizer with a learning rate of $3e^{-5}$ and weight decay of $1e^{-6}$. We schedule the learning rate with cosine annealing and linear warmup of 5000 steps. The batch size is 128, and we apply EMA on the weights of the model. Throughout our experiments, we use 20 denoising steps for flow matching and predict actions with a horizon of $H = 32$.

B. Ablation Study

In our ablation study, we investigate the effect of the amount of used meshes and the demonstration. Specifically, we change the number of demonstrations from 800 to 600 and 200, while still using five meshes, as well as changing the number of meshes to three and one with 800 demonstrations. We visualize the results in Fig. 3. As one would expect, the more meshes and demonstrations, the better. Nonetheless, the average performance increase diminishes rather soon when going from 600 to 800 (1.1%) demonstrations and three to five meshes (5.2%), respectively.

C. Extended Comparison of Mesh Generation

In Tab. III we show extended results from our mesh filtering experiment in Sec. IV-B. We show results split by categories and individual pipeline steps.