NeRF-Aug: Data Augmentation for Robotics with Neural Radiance Fields

Eric Zhu[1], Mara Levy[1], Matthew Gwilliam[1], Abhinav Shrivastava[1]

***Abstract*— Training a policy that can generalize to unknown objects is a long standing challenge within the field of robotics. The performance of a policy often drops significantly in situations where an object in the scene was not seen during training. To solve this problem, we present NeRF-Aug, a novel method that is capable of teaching a policy to interact with objects that are not present in the dataset. This approach differs from existing approaches by leveraging the speed and photorealism of a neural radiance field for augmentation. NeRF-Aug both creates more photorealistic data and runs 3.83 times faster than existing methods. We demonstrate the effectiveness of our method on 4 tasks with 11 novel objects that have no expert demonstration data. We achieve an average 69.1% success rate increase over existing methods. See video results at `https://nerf-aug.github.io`.**

I. Introduction

Humans have an innate ability to interact with objects they have never encountered before. For instance, a person can intuitively approach an unknown object, pick it up, and interact with it. This is in stark contrast to existing robotic systems. For a robot, even the slightest differences in shape or color from the objects seen during training can prevent the robot from achieving success. This challenge of generalization to out-of-distribution samples is a fundamental issue in machine learning and robotics.

Many prior works have explored methods to develop policies for robots that generalize to different objects. A straightforward approach is to simply collect demonstrations involving the novel object. However, this method has significant drawbacks because creating expert demonstrations is time-consuming and expensive as it requires a human to consciously control the robot's movements. Collecting such human demonstrations is unfeasible at scale as every new object potentially requires many new demonstrations. Another approach is to use image editing tools, e.g., the latest diffusion-based image editing [1–5]. While these models can effectively edit images to insert new objects, they are often slow and struggle to render the exact object that will be encountered by the robot. This inaccuracy means the current object remains out of the domain of the training set which often causes these models to fail. Alternatively, some pipelines use depth images for object manipulation [6, 7]. Unfortunately, depth images in the real world suffer from noise and incompleteness [7]. This issue is exacerbated when using mounted gripper cameras, which amplify noise as they get closer to the object. Moreover, even though depth images disregard texture and color, small geometric differences between the original (training) and novel objects can still result in confusion and failure to complete the intended task.

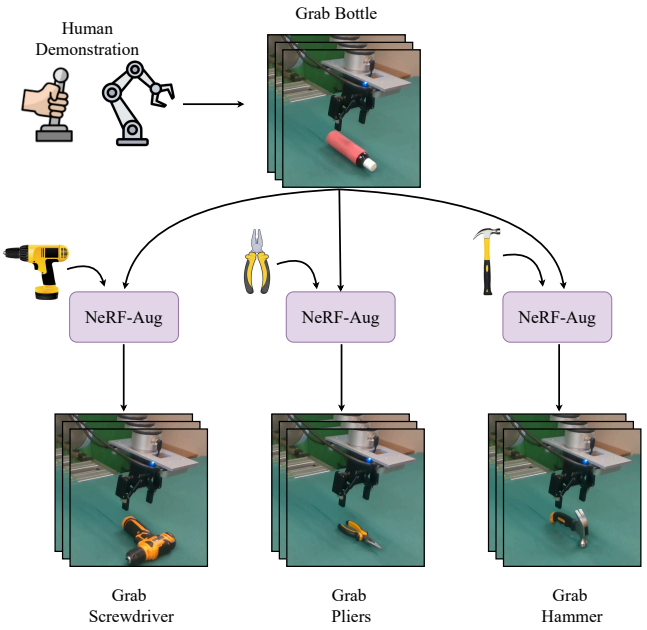

Fig. 1. When a human provides expert demonstrations for training a behavior cloning model, the model is effective for the object in the demonstration, but will fail for novel objects. We propose NeRF-Aug, where we automatically learn NeRFs for the novel objects and inpaint them in the expert data. With this photorealistic synthetic data, the robot can learn to interact successfully with novel objects.

In this work, we propose NeRF-Aug, a lightweight framework that streamlines and automates data collection for a wide range of novel objects. Our goal is to generate data samples for different tasks using novel objects without collecting more human demonstrations. To enable this, we follow the image editing paradigm, but instead of relying on slow generation frameworks, we propose grounding the editing process in the scene using the 3D model of a novel object with a Neural Radiance Field (NeRF) [8] representation. We augment the training data for the robot's policy using this edited scene. Our framework uses existing demonstrations of a different object and generates NeRF-Augmented (NeRF-Aug) synthetic data (Fig. 2) that can be used in imitation learning policies.

We demonstrate that the synthetic data generated by the NeRF-Aug framework is almost indistinguishable from real-world data. Moreover, we show that compared to existing diffusion-based image editing techniques, our method runs significantly faster, creates photorealistic images, and can consistently render objects at a wider degree of viewpoints.

[1]University of Maryland, College Park

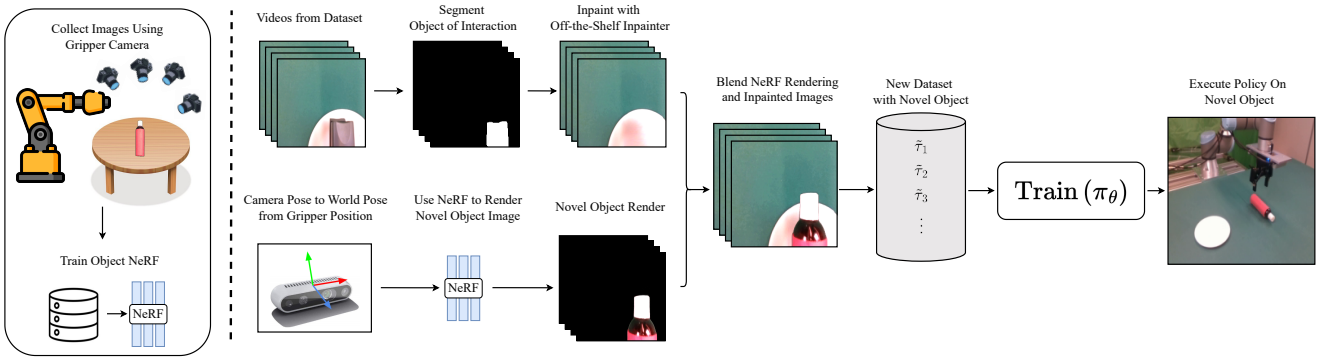

Fig. 2. An illustration of our pipeline from beginning to end. We first train an object-level NeRF of a novel object (**left**). We then simultaneously erase the object in question with an off-the-shelf inpainter (**top**) and leverage NeRF to render images of a new object in the same position as the original object (**bottom**). We use the final synthetic dataset to train a new policy for the robot (**right**).

We test our method on a variety of real world tasks, and achieve a 69.1% increase in success rate while rendering synthetic data 3.83 times faster than the baselines.

To summarize, our contributions are as follows:

- We propose a fast and photorealistic image editing framework to generate synthetic data that can be used in robot policy learning to generalize to novel objects.
- We learn a NeRF of a novel object by using multi-view images of the object captured using a robot arm.
- We edit videos of existing demonstrations by removing the training object via in-painting and blending the NeRF render of a novel object into the inpainted image to generate a synthetic dataset for training the robot policy.
- We demonstrate effective generalization on four diverse tasks using the generated synthetic data for training.

II. Related Works

A. Imitation Leaning

Imitation learning is a common approach to creating robotic policies. Several approaches have been proposed that allow a policy to learn from existing successful trajectories. [1, 9–11] introduce various methods that take observations as input and predict actions. Of these, [1, 10] are multi-input models that input values from different types of data at the same time. [12–15] go a step further and use an expert to actively correct the policy when as it executes trajectories. [16–19] take initial demonstrations and learn a reward function which subsequently trains a reinforcement learning policy.

B. Neural Radiance Fields (NeRFs)

In recent years, there has been a surge in research on NeRFs in both computer graphics and robotics. [20–24] are all NeRF approaches that are capable of rendering high resolution images in real time. Other methods focus on enhancing the robustness of NeRF models against real-world constraints, such as a limited number of images [25], camera pose noise [26], and glare [27]. Finally, other novel-view synthesis techniques [28–31] remove the traditional multi-layer perceptron used to volume render images with other trainable structures.

Within the context of robotics, NeRFs have been used for robot navigation by creating a 3D map of the environment [32]. [33, 34] combine NeRFs and reinforcement learning. [35, 36] explored using robot arms for creating higher quality NeRF models of objects. Conversely, several works have explored using NeRF models for robot decision making, including [37, 38] which creates accurate depth maps for transparent images using NeRF models, and [39] which creates an object level NeRF, imagines a scene with the object in a different place and queries a vision-language model (VLM) for feedback. Finally, [40] adds noisy viewpoints along a demonstration trajectory and take corrective actions to make a more robust behavior cloning model, and [41, 42] combine structure from motion and NeRFs.

C. Synthetic Data in Robotics

Creating synthetic data for robot learning is a popular paradigm for training data hungry machine learning models. [43, 44] champion creating digital twins of the current robot environments and running the robot in these simulations. [45] go a step further and automate the exploration of real-world environments to create the high-fidelity digital twin. [32] combine a simulator with a NeRF model to create a photorealistic policy that could bridge the sim-to-real gap.

Another direction for using synthetic data is using diffusion models in robotics. [1–5] use diffusion models to randomize the texture of objects and swap objects in a scene. [46, 47] use an action-conditioned diffusion model to simulate synthetic videos of a robot performing a task. [48] use a text-to-image diffusion model to synthesize a goal image of objects in a desired location for a goal conditioned reinforcement learning model to subsequently rearrange. [12] uses diffusion to model expert online corrections.

III. Preliminaries

Imitation Learning. In imitation learning, we assume a premade list of N expert trajectories $D = \{\tau_i\}_{i=1}^{N}$ where each trajectory consists of state-action pairs $\tau_i = \{(s_k, a_k)\}_{k=1}^{K}$. In

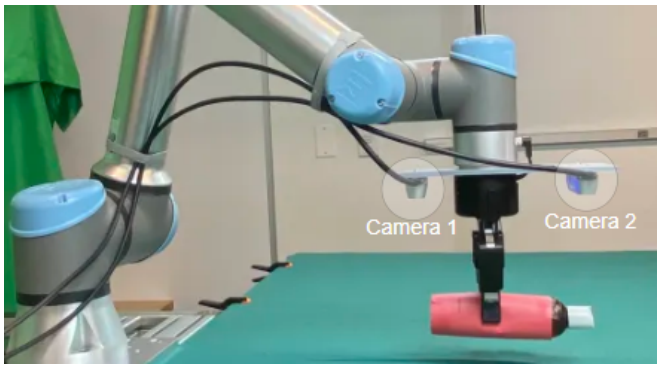

Fig. 3. Our setup. We have two Realsense D435 cameras, angled parallel to the table. The robot arm is an UR5e from Universal Robotics.

visual imitation learning, we assume access to n cameras where each contributes an image to the state. Thus, our state can be denoted as $s_k = (I_1, I_2, ..., I_n)$ where I_k is an image. In vanilla behavior cloning, the policy π is trained offline and learns a mapping between states and actions from our expert dataset by minimizing the loss $\mathcal{L}(\theta) = \mathbb{E}_{(s,a)\sim D}\left[\ell\left(\pi(s), a\right)\right]$ for a given distance metric ℓ.

Neural Radiance Fields. NeRF is a method used for rendering novel views of a scene or object. Formally, we assume a dataset of images and corresponding camera poses $D = \{(I_k, T_k)\}_{k=1}^{K}$ where $T_k \in \mathrm{SE}(3)$ and $I_k \in \mathbb{R}^{w\times h\times 3}$. At test time, we are able to render a photorealistic view at an unknown pose T through volumetric rendering. The result is a photorealistic image of a given scene through querying the NeRF model with a requested camera pose.

IV. Our Approach: NeRF-Aug

An illustrative overview of generating synthetic data using our proposed NeRF-Aug approach is shown in Fig. 2. First, we capture multi-view images of a novel object and learn a corresponding NeRF model. Next, we use the robot arm's gripper position to calculate the camera pose at each image in our trajectory, and we use these to query the NeRF model to render an object-level image at the same position and orientation as our original object in the training example. Then, we combine the NeRF rendering and the original image to create a new synthetic image of the robot handling a different object. Finally, we train our policy on this new dataset and evalaute it on a task consisting of the new object.

A. Creating a NeRF model of the Novel Object

Given a novel object for which there exists no training data, we train a NeRF model for this object. Using a gripper mounted camera, we are able to collect a dataset of image-pose pairs $\left\{(I_k, T_k)\right\}_{k=1}^{M}$ by moving the gripper to various viewpoints both close and far away from the novel object. We use these images to train a NeRF model for the object. We found that due to real-world noise, our NeRF renders came out blurry from the millimeter-level imprecision in our measurements. To account for this, we used NerfStudio's [49] Nerfacto model, which refines the camera intrinsic and extrinsic parameters of all images to denoise measurements.

Fig. 4. A top-down view of the objects used in our real-world study. We used a variety of common household items of varying shapes, sizes and colors to show several kinds of generalization.

B. Calculating Camera Pose relative to Object

Our method relies on accurately rendering a new object in the same position as the original training object at each frame. If the position of the object is not known ahead of time, the pose of the object can be determined by using the built-in grasp detector of a robot's gripper. The pose of the gripper at the first frame that the object is grasped is the same as the pose of the object at the start of the trajectory. We show in Section V-F that using the gripper for calculating the object pose does not significantly decrease the success rate. Once grasped, we assume that the object does not move with respect to the gripper.

Once we know the object pose, we need to find the camera position relative to the object. Using the gripper position with respect to the world given by $T_{\text{gripper}}(t) \in \mathrm{SE}(3)$, we are able to find the camera-to-world matrix of the gripper camera. We multiply the position of the gripper by the offset of the camera from the gripper center

$$T_{\text{camera-to-world}}(t) = T_{\text{gripper}}(t) \cdot T_{\text{camera-offset}}. \tag{1}$$

We have that the relative position of the camera with respect to the object coordinate system is denoted by

$$T_{\text{camera-to-object}}(t) = T^{-1}_{\text{object-to-world}}(t) \cdot T_{\text{camera-to-world}}(t). \tag{2}$$

If the object is already grasped, the relative position of the camera and the object stays the same, denoted by

$$T_{\text{camera-to-object}}(t) = T_{\text{camera-to-object}}(t_{\text{grasp}}) \text{ for } t \geq t_{\text{grasp}}. \tag{3}$$

C. NeRF Renderings

For each frame in the trajectory, we produce a rendering of our novel object at the same position and orientation as the original object by plugging our camera-to-object matrix into the NeRF model, $I_{\text{NeRF}} = \text{NeRF\_Render}\left(T_{\text{camera-to-object}}(t)\right)$. Because we are only interested in the novel object rendered from the NeRF and do not need the background of the NeRF renderings, we find a mask for the object as M_{NeRF}.

D. Combining NeRF Renderings and Original Image

Once we have the NeRF rendering of the new object and the original image, we need to combine these two images. Our new object is potentially smaller than the original object,

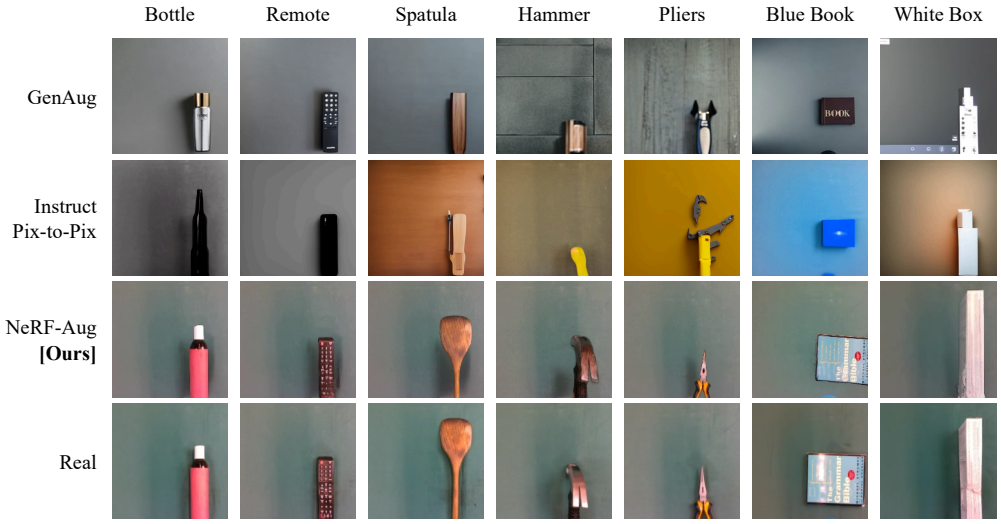

Fig. 5. Comparison of inpainting results when replacing the original object with various objects in our training set. Instruct Pix-to-Pix and GenAug create images that show a new object, but do not recreate our specific object with appropriate dimensions and texture, this generally causes these methods to fail.

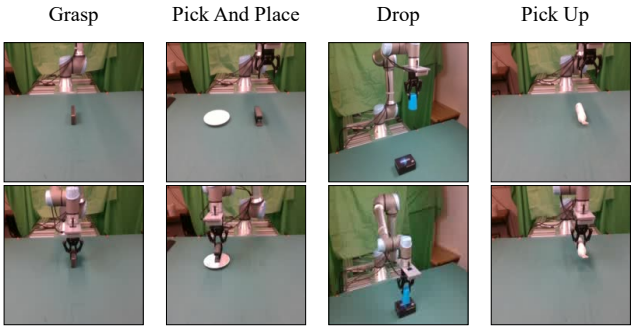

Fig. 6. Where the robot is initialized and the final state of the environment for each of our tasks.

so we cannot simply copy over all the object pixels from the NeRF rendering onto our original image. If we were to do so, pixels from the original object could still remain in our new synthetic image because our new object does not completely cover the original one. Thus, we choose to erase the original object using an off-the-shelf object eraser. To do this, we use a pretrained Segment-Anything model [50] to segment the original object, Cutie video object tracker [51] to track the segmentation mask through the video frames, and ProPainter [52] to erase the object from the image using the segmentation mask. This results in a frame where the original object is removed from the frame and only the background objects/scene remains. From there on, we blend the object pixels from the NeRF renderings onto the image where the object was removed. We combine I_{NeRF} and $I_{\text{no-object}}$ to get a new image with the novel object at the same pose as the original object while still keeping the background of the original image. This is done by computing

$$I_{\text{final}} = I_{\text{NeRF}} \odot M_{\text{NeRF}} + I_{\text{no-object}} \odot \left(1 - M_{\text{NeRF}}\right). \quad (4)$$

These frames and corresponding actions can then be trained using a behavior cloning model and run on a scenario involving the new object.

V. Experiments

We test our method on 4 real-world tasks: **grasp**, **pick and place**, **pick up**, and **drop**. In grasp, place and place, and pick up, we swap the object that is grasped by the gripper. In drop, we switch the object that we drop a cup onto.

A. Our Setup

In our real-world experiments, we use a Universal Robotics UR5e robot arm with a Robotiq-85 gripper. The arm is equipped with two Realsense D435 cameras with one mounted to the front of the gripper and the other mounted to the back (see Fig. 3). We run all experiments on a single Nvidia RTX A4000 GPU. We train the NeRF models with 300 images of resolution 480×640 that are taken with the robot arm, parallel to the ground. We use the default Nerfacto model from Nerf-Studio for training without any hyperparameter tuning.

For our behavior cloning model, we use BAKU [10]. The model takes image inputs from both gripper cameras to predict the next action for the robot to take. We do not allow BAKU to access gripper position for any of the trials. For the drop, pick and place, and pick up tasks, we add color jitter, which is necessary due to auto-exposure and shadows.

B. Baselines

While there has been substantial work with diffusion-based models in robot data augmentation [1–5], to the best of our knowledge, we are only aware of one project that made their augmentation code available to the public: **GenAug** [2]. As such, we are only able to compare to GenAug's diffusion approach. We also compare to **Instruct Pix-to-Pix** [53], where we prompt the model to edit the image by placing a novel object at the same position as the original object.

C. Real World Tasks

For **pick and place**, our initial object is a stapler which is grasped, lifted into the air, and placed on a white plate

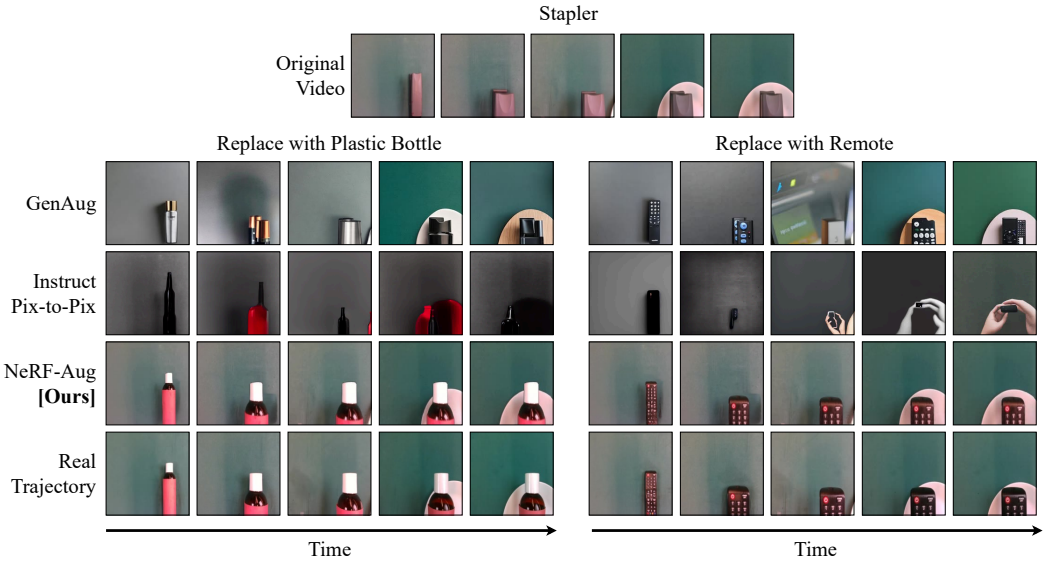

Fig. 7. A comparison of data augmentation results for the "pick and place" task. The top shows the original trajectory. Below that we show three methods that generate an expert synthetic trajectory for two novel objects (plastic bottle on the left, remote on the right). The last row ground-truth trajectories are created by swapping the original object (stapler) with these novel objects in the real world. GenAug is effective at swiping the texture of the object, but the color stays consistent with the original object. Instruct Pix-to-Pix is better at switching the color but also heavily changes the shading. Our method creates nearly identical trajectories to those collected in the real world.

TABLE I

TASK SUCCESS RATES FOR NOVEL OBJECTS (↑)

	Grasping			**Pick and Place**				**Drop Cup**			**Pick Up**
Methods	Screwdriver	Hammer	Pliers	Bottle	Remote	Eraser	Spatula	Green Plate	Book	Pan	White Box
Ours	**100%**	**90%**	**80%**	**100%**	**70%**	**40%**	**50%**	**100%**	**90%**	**100%**	**100%**
Default	0%	0%	0%	10%	0%	0%	0%	0%	0%	0%	0%
GenAug	0%	0%	0%	0%	50%	10%	0%	0%	0%	40%	60%
Instruct Pix-to-Pix	0%	40%	30%	10%	0%	0%	0%	40%	0%	0%	30%

about 2 feet to the left of the stapler. An attempt is deemed successful if the object is released onto and stays on the plate. Our new objects consist of a bottle of lotion, a small remote, a dry-erase eraser, and a wooden spatula. Collecting the data requires 1.6 hours.

For **pick up**, our initial object is a dispenser bottle that was grasped and lifted in the air. Success is determined if the object was not in contact with the table and stayed steady within the gripper. Our novel object is a white cardboard box. The data is collected in 1.2 hours.

For **grasp**, our initial object is a whiteboard eraser that is grasped. Success is determined by if the object is grasped and tight so that it could not move. Our novel objects include a small hammer, an electric screwdriver, and pliers. Collecting the data takes 1.4 hours.

For **drop**, we drop a blue cup onto a flat object. Success is determined if the cup is successfully dropped on top of the object in question. Our initial object is a box that the robot would drop this cup onto. Our novel objects include a small black frying pan, a green plate, and a blue book. We collect the necessary data in 1.1 hours.

We show examples of start and end states for these 4 tasks in Fig. 6. We show all objects in Fig. 4. For each task, we create 200 trajectories using a hard coded expert that has access to the position of objects. To add variation to trajectories, we add seeded random noise to the output of this expert. For re-rendering, we start out with the ground truth object pose instead of calculating it.

D. Comparison to Baselines

As seen in Table I, our method consistently outperforms other methods on all four tasks. Fig. 7 and Fig. 5 demonstrate that by leveraging the photorealism of a NeRF blended with inpainted background, our method is able to create synthetic data with the same geometry and color as real-world data of a different object.

In contrast to our method, GenAug and Instruct Pix-to-Pix require intensive prompting to achieve a realistic looking image. We observe that the two methods are highly inconsistent from frame to frame because the random noise inputted into the diffusion model that leads to very different results each time a frame is rendered. The inconsistency can be detrimental to multi-camera systems where GenAug and Instruct Pix-to-Pix could render a completely different looking object for each camera image at the same time-step. Additionally, both GenAug and Instruct Pix-to-Pix

TABLE II
TIME TO CREATE NEW DATA IN HOURS (↓)

	Grasping			Pick and Place				Drop Cup			Pick Up
Methods	Screwdriver	Hammer	Pliers	Bottle	Remote	Eraser	Spatula	Green Plate	Book	Pan	Box
Ours (first novel object)	5.2	5.3	5.0	5.3	5.3	5.7	5.3	4.3	4.2	4.6	4.0
Ours (subsequent novel objects)	2.1	2.2	1.9	2.5	2.5	2.9	2.5	2.0	1.9	2.3	1.8
GenAug	43.3	42.3	42.6	50.0	49.1	50.3	49.3	34.7	35.1	35.0	27.0
Instruct Pix-to-Pix	19.2	19.3	19.1	20.8	21.5	21.4	20.9	16.8	16.9	16.7	14.5

TABLE III
BREAKDOWN OF HOURS TAKEN FOR EACH PART OF OUR METHOD AVERAGED PER NOVEL OBJECT

Dataset	NeRF Training	Segmentation + Object Eraser	NeRF Rendering
Grasp	0.69	3.06	1.37
Pick And Place	0.90	2.83	1.64
Drop	0.74	2.34	1.30
Pick Up	0.62	2.20	1.13

TABLE IV
ABLATION ON SUCCESS RATES WITH INFERRING AND NOT INFERRING OBJECT POSE

	Pick and Place			
Methods	Bottle	Remote	Eraser	Spatula
Known Object Pose	**100%**	**70%**	**40%**	50%
Inferred Object Pose	70%	50%	30%	**70%**

seem to change the shading of the scene. Instruct Pix-to-Pix especially makes the background of the frames the same color as the object we prompted for.

Despite GenAug explicitly switching the texture of the object, many of the renders came out with an object that was the same color as the original object. We also notice that GenAug inpaints a new object with the exact geometry as the original object, which can be problematic because the new object is usually slightly bigger or smaller than the original one. This behavior also explains why GenAug works most effectively on picking and placing a remote which has the same dimensions and color as the original object (stapler). Finally, we argue that a major drawback of diffusion models in general is that they are not trained to render objects that appear very close to the camera. Gripper camera robotic systems, like our setup, will move the camera to only a few inches away from the object it is interacting with. Often only part of the object is in frame at that distance. We believe the camera was too close to the object for GenAug and Instruct Pix-to-pix to synthesize a recognizable bottle or remote in Fig. 7, especially in the later frames which look the most unrecognizable.

E. Data augmentation speed

We show that our method is significantly faster than other methods, as seen in Table II. Existing diffusion-based methods are slow because diffusion models innately require running the same network many times for a single frame whereas ours runs only once per frame. On the other hand, because our image resolution is only 180×180 pixels, our NeRF model is actually able to render at twice the fps as the original video. Note that our times are doubled because we used a two-camera system, so for a typical single-camera system, we expect our numbers to be half of what we report. We also notice that the majority of the time taken was from the segmentation and object eraser component (refer to Table III). This module only needs to run once per task as opposed to once per novel object because it deals with erasing the original object in the training videos which is shared across all novel objects. In Table II, we show the time taken including the segmentation and object eraser ('first novel object') and excluding this component ('subsequent novel objects'). We measure all times on a single Nvidia RTX A4000 graphics card.

F. Is Ground-truth Object Position necessary?

Often, it is hard to precisely estimate the position of objects in robot demonstration data. As mentioned in Section IV-B, one way to estimate the object position is by recording the gripper position at the time the object was grasped as the starting object position. We conduct an ablation study on the pick and place task to show that estimating object pose by using a gripper is a reasonable approximation and does not immensely deteriorate the quality of the policy compared to using ground-truth object position. We report results in Table IV. Interestingly, the performance with the wooden spatula actually increases, likely because using the gripper to infer the object pose is more accurate than measuring it beforehand.

VI. CONCLUSION

We introduce NeRF-Aug, a data augmentation framework for gripper-camera robotic systems. Our method leverages the photorealism of NeRFs to replace training objects in expert demonstrations with novel objects. This allows us to create synthetic training data for novel objects that is virtually indistinguishable from data that would otherwise require a human to demonstrate. Through extensive quantitative and qualitative experiments on four real-world tasks with 11 different objects, we show that policies trained using NeRF-Aug data are consistently more successful at tasks involving novel objects than the baselines, while only requiring a fraction of the time to collect. Future research could explore other novel-view synthesis methods such as Gaussian Splatting and Plenoxels to generate similar augmentation frameworks.

References

[1] H. Bharadhwaj, J. Vakil, M. Sharma, A. Gupta, S. Tulsiani, and V. Kumar, "Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking," 2023. [Online]. Available: https://arxiv.org/abs/2309.01918

[2] Z. Chen, S. Kiami, A. Gupta, and V. Kumar, "Genaug: Retargeting behaviors to unseen situations via generative augmentation," 2023. [Online]. Available: https://arxiv.org/abs/2302.06671

[3] Z. Mandi, H. Bharadhwaj, V. Moens, S. Song, A. Rajeswaran, and V. Kumar, "Cacti: A framework for scalable multi-task multi-scene visual imitation learning," 2023. [Online]. Available: https://arxiv.org/abs/2212.05711

[4] T. Yu, T. Xiao, A. Stone, J. Tompson, A. Brohan, S. Wang, J. Singh, C. Tan, M. Dee, J. Peralta, B. Ichter, K. Hausman, and F. Xia, "Scaling robot learning with semantically imagined experience," *ArXiv*, vol. abs/2302.11550, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:257079001

[5] I. Kapelyukh, V. Vosylius, and E. Johns, "Dall-e-bot: Introducing web-scale diffusion models to robotics," *IEEE Robotics and Automation Letters*, vol. 8, no. 7, p. 3956–3963, July 2023. [Online]. Available: http://dx.doi.org/10.1109/LRA.2023.3272516

[6] X. Zhu, L. Sun, Y. Fan, and M. Tomizuka, "6-dof contrastive grasp proposal network," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6371–6377, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:232417362

[7] C. Sweeney, G. Izatt, and R. Tedrake, "A supervised approach to predicting noise in depth images," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 796–802.

[8] 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," 2020. [Online]. Available: https://arxiv.org/abs/2003.08934

[9] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich, "Rt-1: Robotics transformer for real-world control at scale," 2023. [Online]. Available: https://arxiv.org/abs/2212.06817

[10] S. Haldar, Z. Peng, and L. Pinto, "Baku: An efficient transformer for multi-task policy learning," 2024. [Online]. Available: https://arxiv.org/abs/2406.07539

[11] F. Torabi, G. Warnell, and P. Stone, "Behavioral cloning from observation," 2018. [Online]. Available: https://arxiv.org/abs/1805.01954

[12] X. Zhang, M. Chang, P. Kumar, and S. Gupta, "Diffusion meets dagger: Supercharging eye-in-hand imitation learning," 2024. [Online]. Available: https://arxiv.org/abs/2402.17768

[13] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer, "Hg-dagger: Interactive imitation learning with human experts," 2019. [Online]. Available: https://arxiv.org/abs/1810.02890

[14] R. Hoque, L. Y. Chen, S. Sharma, K. Dharmarajan, B. Thananjeyan, P. Abbeel, and K. Goldberg, "Fleet-dagger: Interactive robot fleet learning with scalable human supervision," 2022. [Online]. Available: https://arxiv.org/abs/2206.14349

[15] X. Sun, S. Yang, M. Zhou, K. Liu, and R. Mangharam, "Mega-dagger: Imitation learning with multiple imperfect experts," 2024. [Online]. Available: https://arxiv.org/abs/2303.00638

[16] G. Swamy, S. Choudhury, J. A. Bagnell, and Z. S. Wu, "Inverse reinforcement learning without reinforcement learning," 2024. [Online]. Available: https://arxiv.org/abs/2303.14623

[17] J. Ho and S. Ermon, "Generative adversarial imitation learning," 2016. [Online]. Available: https://arxiv.org/abs/1606.03476

[18] S. Reddy, A. D. Dragan, and S. Levine, "Sqil: Imitation learning via reinforcement learning with sparse rewards," 2019. [Online]. Available: https://arxiv.org/abs/1905.11108

[19] J. Fu, K. Luo, and S. Levine, "Learning robust rewards with adversarial inverse reinforcement learning," 2018. [Online]. Available: https://arxiv.org/abs/1710.11248

[20] P. Wang, Y. Liu, Z. Chen, L. Liu, Z. Liu, T. Komura, C. Theobalt, and W. Wang, "F^2-nerf: Fast neural radiance field training with free camera trajectories," 2023. [Online]. Available: https://arxiv.org/abs/2303.15951

[21] C. Reiser, S. Peng, Y. Liao, and A. Geiger, "Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 14 315–14 325, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:232352619

[22] P. Hedman, P. P. Srinivasan, B. Mildenhall, J. T. Barron, and P. E. Debevec, "Baking neural radiance fields for real-time view synthesis," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5855–5864, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:232379923

[23] F. Rivas-Manzaneque, J. S. Acosta, A. P. Sánchez, F. Moreno-Noguer, and Á. Ribeiro, "Nerflight: Fast and light neural radiance fields using a shared feature grid," *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12 417–12 427, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:260876753

[24] H. Lin, S. Peng, Z. Xu, Y. Yan, Q. Shuai, H. Bao, and X. Zhou, "Efficient neural radiance fields for interactive free-viewpoint video," *SIGGRAPH Asia 2022 Conference Papers*, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:254044754

[25] J. Gu, A. Trevithick, K.-E. Lin, J. Susskind, C. Theobalt, L. Liu, and R. Ramamoorthi, "Nerfdiff: Single-image view synthesis with nerf-guided distillation from 3d-aware diffusion," 2023. [Online]. Available: https://arxiv.org/abs/2302.10109

[26] N. Pearl, T. Treibitz, and S. Korman, "Nan: Noise-aware nerfs for burst-denoising," 2022. [Online]. Available: https://arxiv.org/abs/2204.04668

[27] Y.-C. Guo, D. Kang, L. Bao, Y. He, and S.-H. Zhang, "Nerfren: Neural radiance fields with reflections," 2022. [Online]. Available: https://arxiv.org/abs/2111.15234

[28] A. Yu, S. Fridovich-Keil, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, "Plenoxels: Radiance fields without neural networks," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5491–5500, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:245006364

[29] A. Yu, R. Li, M. Tancik, H. Li, R. Ng, and A. Kanazawa, "Plenoctrees for real-time rendering of neural radiance fields," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5732–5741, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:232352425

[30] C. Sun, M. Sun, and H.-T. Chen, "Direct voxel grid optimization: Super-fast convergence for radiance fields

reconstruction," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5449–5459, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:244477646

[31] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," 2023. [Online]. Available: https://arxiv.org/abs/2308.04079

[32] 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," 2022. [Online]. Available: https://arxiv.org/abs/2210.04932

[33] D. Shim, S. Lee, and H. J. Kim, "Snerl: Semantic-aware neural radiance fields for reinforcement learning," 2023. [Online]. Available: https://arxiv.org/abs/2301.11520

[34] D. Driess, I. Schubert, P. Florence, Y. Li, and M. Toussaint, "Reinforcement learning with neural radiance fields," 2022. [Online]. Available: https://arxiv.org/abs/2206.01634

[35] S. Lee, L. Chen, J. Wang, A. Liniger, S. Kumar, and F. Yu, "Uncertainty guided policy for active robotic 3d reconstruction using neural radiance fields," 2022. [Online]. Available: https://arxiv.org/abs/2209.08409

[36] L. Chen, Y. Song, H. Bao, and X. Zhou, "Perceiving unseen 3d objects by poking the objects," 2023. [Online]. Available: https://arxiv.org/abs/2302.13375

[37] J. Kerr, L. Fu, H. Huang, Y. Avigal, M. Tancik, J. Ichnowski, A. Kanazawa, and K. Goldberg, "Evo-neRF: Evolving neRF for sequential robot grasping of transparent objects," in *6th Annual Conference on Robot Learning*, 2022. [Online]. Available: https://openreview.net/forum?id=Bxr45keYrf

[38] J. Ichnowski, Y. Avigal, J. Kerr, and K. Goldberg, "Dex-nerf: Using a neural radiance field to grasp transparent objects," 2021. [Online]. Available: https://arxiv.org/abs/2110.14217

[39] I. Kapelyukh, Y. Ren, I. Alzugaray, and E. Johns, "Dream2real: Zero-shot 3d object rearrangement with vision-language models," 2024. [Online]. Available: https://arxiv.org/abs/2312.04533

[40] A. Zhou, M. J. Kim, L. Wang, P. Florence, and C. Finn, "Nerf in the palm of your hand: Corrective augmentation for robotics via novel-view synthesis," 2023. [Online]. Available: https://arxiv.org/abs/2301.08556

[41] C.-M. Chung, Y.-C. Tseng, Y.-C. Hsu, X.-Q. Shi, Y.-H. Hua, J.-F. Yeh, W.-C. Chen, Y.-T. Chen, and W. H. Hsu, "Orbeez-slam: A real-time monocular visual slam with orb features and nerf-realized mapping," 2023. [Online]. Available: https://arxiv.org/abs/2209.13274

[42] A. Rosinol, J. J. Leonard, and L. Carlone, "Nerf-slam: Real-time dense monocular slam with neural radiance fields," 2022. [Online]. Available: https://arxiv.org/abs/2210.13641

[43] Z. Jiang, C.-C. Hsu, and Y. Zhu, "Ditto: Building digital twins of articulated objects from interaction," 2022. [Online]. Available: https://arxiv.org/abs/2202.08227

[44] 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," 2024. [Online]. Available: https://arxiv.org/abs/2403.03949

[45] C.-C. Hsu, Z. Jiang, and Y. Zhu, "Ditto in the house: Building articulation models of indoor scenes through interactive perception," 2023. [Online]. Available: https://arxiv.org/abs/2302.01295

[46] Y. Du, M. Yang, B. Dai, H. Dai, O. Nachum, J. B. Tenenbaum, D. Schuurmans, and P. Abbeel, "Learning universal policies via text-guided video generation," 2023. [Online]. Available: https://arxiv.org/abs/2302.00111

[47] S. Huang, M. Levy, Z. Jiang, A. Anandkumar, Y. Zhu, L. Fan, D.-A. Huang, and A. Shrivastava, "Ardup: Active region video diffusion for universal policies," 2024. [Online]. Available: https://arxiv.org/abs/2406.13301

[48] K. Black, M. Nakamoto, P. Atreya, H. Walke, C. Finn, A. Kumar, and S. Levine, "Zero-shot robotic manipulation with pretrained image-editing diffusion models," 2023. [Online]. Available: https://arxiv.org/abs/2310.10639

[49] M. Tancik, E. Weber, E. Ng, R. Li, B. Yi, T. Wang, A. Kristoffersen, J. Austin, K. Salahi, A. Ahuja, D. Mcallister, J. Kerr, and A. Kanazawa, "Nerfstudio: A modular framework for neural radiance field development," in *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings*, ser. SIGGRAPH '23. ACM, July 2023. [Online]. Available: http://dx.doi.org/10.1145/3588432.3591516

[50] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," 2023. [Online]. Available: https://arxiv.org/abs/2304.02643

[51] H. K. Cheng, S. W. Oh, B. Price, J.-Y. Lee, and A. Schwing, "Putting the object back into video object segmentation," 2024. [Online]. Available: https://arxiv.org/abs/2310.12982

[52] S. Zhou, C. Li, K. C. K. Chan, and C. C. Loy, "Propainter: Improving propagation and transformer for video inpainting," 2023. [Online]. Available: https://arxiv.org/abs/2309.03897

[53] T. Brooks, A. Holynski, and A. A. Efros, "Instructpix2pix: Learning to follow image editing instructions," 2023. [Online]. Available: https://arxiv.org/abs/2211.09800