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

Eric Zhu, Mara Levy, Matthew Gwilliam, Abhinav Shrivastava
Department of Computer Science
University of Maryland College Park
{ezhu2958,mlevy,mgwillia,abhinav2}@umd.edu

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, photorealism, and 3D consistency of a neural radiance field for augmentation. NeRF-Aug both creates more photorealistic data and runs 63% faster than existing methods. We demonstrate the effectiveness of our method on 5 tasks with 9 novel objects that are not present in the expert demonstrations. We achieve an average performance boost of 55.6% when comparing our method to the next best method. You can see video results at https://nerf-aug.github.io.

Keywords: Imitation Learning, Novel View Synthesis, Generalization

1 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. 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. Therefore, collecting such human demonstrations is infeasible at scale.

Another approach is to use image editing tools, e.g., the latest diffusion-based image editing [1, 2, 3, 4, 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.

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

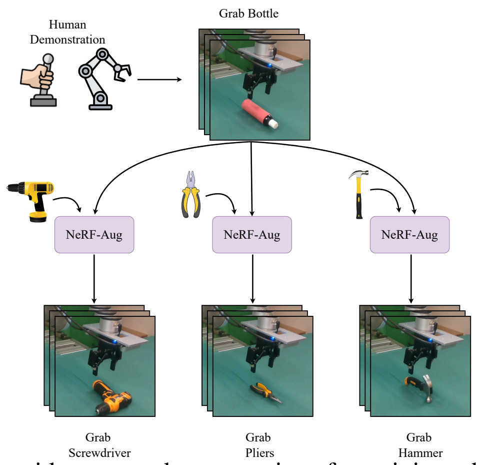

Figure 1: When a human provides expert demonstrations for training a behavior cloning model, the model is effective for the object in the demonstration, but will fail for novel objects. We propose NeRF-Aug, where we automatically learn NeRFs for the novel objects and inpaint them in the expert data.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

the image editing paradigm, but instead of relying on slow generation frameworks, we propose using a 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 wide degree of viewpoints.

We test our method on a variety of real world tasks, and achieve a 55.6% increase in success rate over generation based approaches while rendering the synthetic data 63% 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 existing expert video demos by removing the training object via in-painting and blend the NeRF render of a novel object into the inpainted image to create synthetic data.
- We demonstrate effective generalization on five diverse tasks using the generated synthetic dataset for training.

2 Related Works

2.1 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, 10, 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, 13, 14, 15] go a step further and use an expert to actively correct the policy when as it executes trajectories. [16, 17, 18, 19] take initial demonstrations and learn a reward function to train a reinforcement learning policy.

2.2 Neural Radiance Fields (NeRFs)

In recent years, there has been a surge in research on NeRFs. [20, 21, 22, 23, 24] are all NeRF approaches that are capable of rendering high resolution images in real time. Other methods focus on the robustness of NeRF models against real-world constraints, such as limited number of images [25], camera pose noise [26], and glare [27]. Finally, other novel-view synthesis methods [28, 29, 30, 31] remove the traditional mlp used to 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 and 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.

2.3 Synthetic Data in Robotics

Creating synthetic data for robot learning is a popular paradigm for training data hungry machine learning models. [41, 42] champion creating digital twins of the current robot environments and running the robot in these simulations. [43] 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, 2, 3, 4, 5] use diffusion models to randomize the texture of objects and swap objects in a scene. [44, 45] use a video diffusion model to create synthetic videos of a robot performing a task and then let the robot recreate these videos. [46] 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. Finally [47] uses diffusion to switch the model of robot arm performing an action in a demonstration.

3 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 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 $I_k \in \mathbb{R}^{w\times h\times 3}$. Neural radiance fields are able to render the scene at a novel pose T through volumetric rendering. The result is a photorealistic image of a given scene by querying the NeRF model with a requested camera pose.

4 Our Approach: NeRF-Aug

The first step in our approach is capturing images of a novel object from multiple viewpoints and training a corresponding NeRF model. Next, we use the robot arm's gripper position to calculate the camera pose at each image in our trajectory. Using these camera poses we query the NeRF model to render an image of the novel object 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

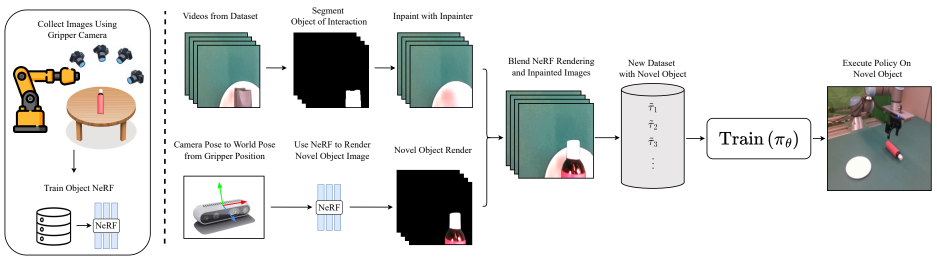
Collect Images Using Gripper Camera
Train Object NeRF
NeRF
Videos from Dataset
Segment Object of Interaction
Inpaint with Inpainter
Camera Pose to World Pose from Gripper Position
Use NeRF to Render Novel Object Image
NeRF
Novel Object Render
Blend NeRF Rendering and Inpainted Images
New Dataset with Novel Object
$\tilde{\tau}_1$
$\tilde{\tau}_2$
$\tilde{\tau}_3$
$\vdots$
Train (π_θ)
Execute Policy On Novel Object

Figure 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 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**).

new synthetic image of the robot handling the novel object. Finally, we train our policy on this new dataset and evaluate the original task on the novel object. An illustrative overview of our proposed NeRF-Aug approach is shown in Fig. 2.

4.1 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 $\{(I_k, T_k)\}_{k=1}^{M}$ 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. In our experiments, we used NerfStudio's [48] default Nerfacto model, which refines the camera parameters of all images to denoise measurements.

4.2 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. For simplicity we use a set position in this project, but the original object position can easily be calculated from the gripper position at the time the object is first grasped.

Next, NeRF-Aug finds the camera position relative to the object. Using the gripper position with respect to the world given by $T_{\text{gripper}}(t) \in \text{SE}(3)$, we are able to find the camera-to-world matrix of the gripper camera. We multiply the gripper position by the offset of the camera from the gripper

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

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

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

We also add a small amount of noise at each timestep into the camera-to-object matrix before rendering in the NeRF model. This slightly shifts the object from its original position to simulate a larger degree of grasp positions.

4.3 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 denoted as M_{NeRF}.

4.4 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, so we cannot

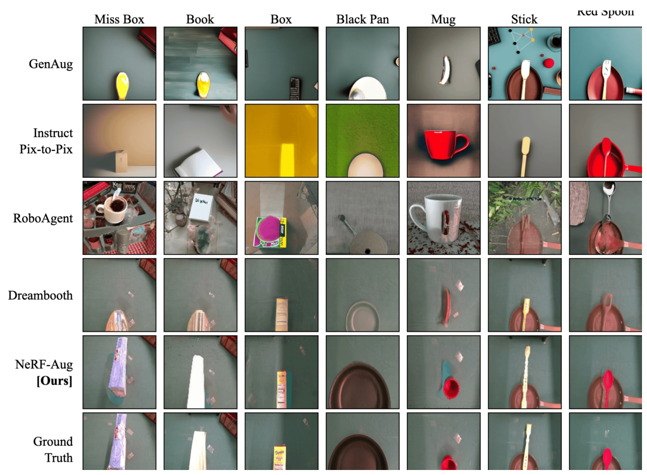

Figure 3: Inpainting results when replacing the original object with various objects in our training set. Instruct P2P, GenAug, and RoboAgent create images that show a new object, but do not recreate our specific object with appropriate dimensions and texture, causing these methods to fail. Dreambooth has trouble with inpainting complex masks and does not recreate the right shade of colors.

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 the new synthetic image. Thus, we choose to erase the original object using an inpainter. While any inpainter would work, we chose to inpaint using a NeRF model trained on the background of the scene. To do this, we use a pretrained Segment-Anything model [49] to segment the original object, Cutie video object tracker [50] to track the segmentation mask through the video frames, and a background NeRF model to switch the pixels in this mask with pixels from the background.

This results in a frame where the original object is removed from the frame and only the background scene remains. From there, 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

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

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

5 Experiments

We test our method on 5 real-world tasks: 1) **Remove Object**, 2) **Place in Chest**, 3) **Serve On Plate**, 4) **Cook with Pot**, and 5) **Lift up High**.

In our real-world experiments, we use an UR5e robot arm with a Robotiq-85 gripper. The arm is equipped with a Realsense D455 mounted to the front of the gripper (see Fig. 5). We run all experiments on a single Nvidia RTX A4000 GPU. We use the default Nerfacto model from Nerfstudio [48] for training without any hyperparameter tuning aside from stopping the training at 4,000 timesteps for object NeRFs and 20,000 timesteps for background NeRFs.

For our behavior cloning model, we use BAKU [10]. The model takes image inputs 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 some tasks, we also added color jitter to combat auto-exposure and shadows.

5.1 Baselines

While there has been substantial work with diffusion-based models in robot data augmentation [1, 2, 3, 4, 5], to the best of our knowledge, we are only aware of one project that made their augmentation

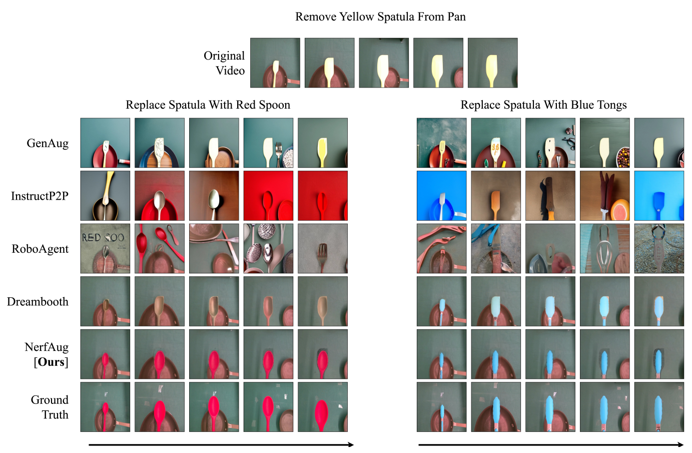
Remove Yellow Spatula From Pan
Original Video
Replace Spatula With Red Spoon
Replace Spatula With Blue Tongs
GenAug
InstructP2P
RoboAgent
Dreambooth
NerfAug [Ours]
Ground Truth

Figure 4: A comparison of data augmentation results for the "Remove Object" task. The top shows various techniques and how they compare to a original trajectory. Below that we show four methods that generate an expert synthetic trajectory for two novel objects (red spoon on the left, blue tongs on the right). 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. RoboAgent adds a bunch of distractor objects that could confuse a behavior cloning model. Dreambooth inpaints an object at the same dimensions at the original object and in a different shade. Our method creates nearly identical trajectories to those collected in the real world.

Table 1: Task success rates for novel objects (↑)

	Remove Object			**Place in Chest**		**Serve on Plate**	**Cook with Pot**	**Lift up High**	
Methods	Blue Tongs	Red Spoon	Wood Stick	Swiss Miss Box	White Book	Large Pan	Sugar Box	Red Mug	Hammer
Ours	**70%**	**100%**	**80%**	**90%**	**60%**	**100%**	**80%**	**90%**	**100%**
Default	0%	0%	0%	10%	10%	0%	0%	0%	90%
GenAug	10%	0%	0%	50%	0%	0%	0%	0%	0%
Instruct Pix-to-Pix	40%	50%	10%	50%	0%	**100%**	0%	0%	0%
Roboagent	30%	20%	0%	80%	**60%**	0%	0%	0%	10%
Dreambooth	60%	0%	60%	20%	40%	0%	0%	**90%**	0%

code available to the public: **GenAug** [2] which uses a diffusion inpainter to randomize textures and change objects. As such, we compare directly to GenAug's diffusion approach. We also compare to a common image-editing model **Instruct Pix-to-Pix** [51]. In this baseline we prompt the model to edit the image by placing a novel object at the same position as the original object. We also wrote our own implementation of **Roboagent** [1]'s data augmentation method and compare to that. Finally, inspired by [47], we finetune a Stable Diffusion Inpainter using Dreambooth [52] to each of our novel objects with the same images used to train our NeRF models.

5.2 Real World Tasks

For the **Remove Object** task, our initial object is a yellow spatula that rests on top of a small frying pan. The robot grasps the spatula and lifts it from the pan then drops it to the left of the pan. For this task we test on 3 novel objects: blue tongs, a red spoon, and a long wooden stick. These objects show how well NeRF-Aug can generalize to different colors and lengths of objects. For this task, we used 30 demonstrations.

For the **Place in Chest** task, our initial object is a mustard bottle which the robot grasps and places into a wooden chest. We tested on two novel objects: a Swiss Miss box and a paperback book. These objects differ in geometry and texture from the original object. We used 30 demonstrations.

For the **Serve on Plate** task, our initial object is a small white plate that we place a pear onto. We then replace the white plate with a black pan about 2 times larger than the plate. This tests how well augmentation techniques can adapt to different sizes of objects. We used 23 demonstrations.

Table 2: Time To Create New Data in Minutes Including NeRF Model Training (↓)

	Remove Object			Place in Chest		Serve on Plate	Cook with Pot	Lift up High	
Methods	Blue Tongs	Red Spoon	Wood Stick	Swiss Miss Box	White Book	Large Pan	Sugar Box	Red Mug	Hammer
Ours (w/o inpainter)	12.77	15.0	12.8	13.2	15.2	12.8	11.4	5.87	5.36
Ours (w/ inpainter)	30.9	33.1	31.0	44.5	46.5	44.7	38.7	24.9	24.4
GenAug	556	548	555	572	566	480	419	116	117
Instruct Pix-to-Pix	73.5	73.6	70.4	75.4	76.5	64.7	55.9	15.7	15.8
Roboagent	1858	1835	1843	1913	1906	1633	1423	402	408
Dreambooth	751	738	746	777	781	660	526	160	156

For the **Cook with Pot** task, the robot removes the lid from a small pot and places it down on the table. It then picks up a stapler and places it in the pot. Our novel object is a sugar box that replaces the stapler. The sugar box has a more complicated texture than the stapler allowing us to evaluate how different textures affect policy performance as well as how well NeRF-Aug works in multi-step tasks. We use 10 demonstrations.

For the **Lift up High** task, the robot grasps a banana and lifts it into the air. Our novel objects were a red mug and a small yellow-black hammer in place of the banana. The mug is much wider but not as long as the banana. The hammer is much thinner but heavier than the banana. In this case, we tested how different geometries and weights can affect policy performance. We use 10 demonstrations.

5.3 Comparison to Baselines

As seen in Table 1, our method consistently outperforms other methods on all five tasks. Over the next best method, we achieve an average of 43% increase in the ”Remove Objects” task, a 5% increase in the ”Place In Chest” task, an 80% increase on the ”Cook with Pot” task, and a 50% increase on the ”Lift up High” task. Our method achieved a 85.6% success rate compared to the next best method (Dreambooth) with a 30% success rate.

Which methods synthesize object locations and sizes accurately? GenAug, Instruct Pix-to-Pix, and RoboAgent often output images of an object in either the wrong orientation or position. As seen in fig. 3, we prompted the baselines to inpaint a ”red mug,” only for the output image to consist of a sideways view of a mug that is much larger than the actual mug. In contrast, our method was able to accurately synthesize the mug from a top-down view. Being able to render an object at unconventional viewpoints, such as when an object is very close to the camera or in a birds-eye-view, is a major reason why our method performs better than the baselines (such as being able to lift the mug 90% more often than any other method). Dreambooth seems more effective at synthesizing the correct object viewpoint, but will fail when the original object is significantly different in shape from the old object (see fig. 3 where Dreambooth fails to inpaint a swiss miss box and a book when the original object was a mustard bottle). A wide degree of viewpoints is important because images where the object is very close to the camera or from a birds-eye-view form the majority of an expert demonstration. Additionally, all of the baselines fail to inpaint the right size of the novel object.

Which method has higher image consistency? Moreover, we observe that aside from Dreambooth, the baselines are highly inconsistent in the color and presence of distractor objects from frame to frame. The inconsistency can be detrimental to multi-camera systems where these methods may render a different looking object for each camera image at the same time-step. See Fig. 4 for examples where the diffusion models inconsistently adds distractor objects in one image and no distractor objects later on. Roboagent especially seems to add many distractors to the background. On the other hand, NeRF-Aug keeps the same colors and does not add distractor objects.

How do the methods compare in synthesizing texture and color? We see that NeRF-Aug is able to accurately synthesize the exact color and texture of the novel object. Our most complicatedly

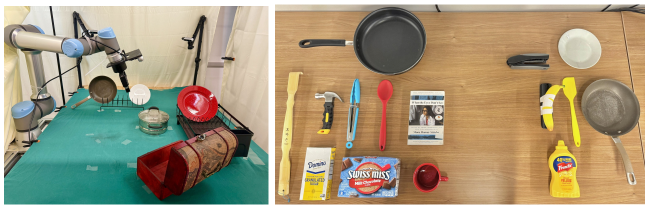

Figure 5: Our setup. We have a Realsense D455 camera, During our tests, we use a large number of background objects such as the two dish racks and the wooden chest to reflect a real world scenario. The right image shows all objects used in experiments. Left side are novel objects, right side are original objects.

Table 3: Breakdown of Minutes Taken for Each Part of Our Method

Dataset	**Train Novel Object NeRF**	**Segmentation**	**Background NeRF Training + Inpainting**	**Novel Object NeRF Rendering**
Remove Object	3.40	4.75	13.38	9.72
Place In Chest	5.08	5.50	25.83	10.07
Serve On Plate	3.51	5.15	26.68	9.35
Lift Up High	5.07	1.45	17.61	2.07
Cook with Pot	3.03	4.47	22.92	8.36

textured objects were the Swiss Miss Box and Sugar Box, both of which NeRF-Aug was able to accurately synthesize down to the text on the box. In this area, GenAug fails by often incorporating the color of the original object into the modified image. Instruct Pix-to-Pix frequently makes the background of the frames the same color as the object we prompted for. Dreambooth seems to synthesize the texture correctly, but often fails to get the exact shade of the novel object.

5.4 Data augmentation speed

We show that our method is significantly faster than other methods, as seen in Table 2. Our NeRF model is actually able to render at twice the fps as the original video because of our 128 by 128 resolution. Namely, our method is 63.6% faster than the second fastest method (P2P).

We also notice that the majority of the time taken was from the segmentation and background in-painter component (refer to Table 3). 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. For this reason, we also calculated the time without the training and running of the inpainter, resulting in NeRF-Aug running 4.99 times faster than P2P. In Table 2, we show the time taken including the segmentation, training the background NeRF model, and in-painting of the background NeRF ('w/ inpainter') and excluding this component ('w/o inpainter'). We measure all times on a single Nvidia RTX A4000 graphics card.

6 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 5 real-world tasks with 9 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 augment. 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. URL 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. URL 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. URL 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*, abs/2302.11550, 2023. URL 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*, 8(7):3956–3963, July 2023. ISSN 2377-3774. doi:10.1109/lra.2023.3272516. URL 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)*, pages 6371–6377, 2021. URL 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)*, pages 796–802, 2019. doi:10.1109/ICRA.2019.8793820.

[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. URL 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. URL https://arxiv.org/abs/2212.06817.

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

[11] F. Torabi, G. Warnell, and P. Stone. Behavioral cloning from observation, 2018. URL 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. URL 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. URL 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. URL 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. URL 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. URL https://arxiv.org/abs/2303.14623.

[17] J. Ho and S. Ermon. Generative adversarial imitation learning, 2016. URL 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. URL https://arxiv.org/abs/1905.11108.

[19] J. Fu, K. Luo, and S. Levine. Learning robust rewards with adversarial inverse reinforcement learning, 2018. URL 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. URL 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)*, pages 14315–14325, 2021. URL 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)*, pages 5855–5864, 2021. URL 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)*, pages 12417–12427, 2023. URL 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. URL 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. URL https://arxiv.org/abs/2302.10109.

[26] N. Pearl, T. Treibitz, and S. Korman. Nan: Noise-aware nerfs for burst-denoising, 2022. URL 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. URL 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)*, pages 5491–5500, 2021. URL 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)*, pages 5732–5741, 2021. URL 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)*, pages 5449–5459, 2021. URL 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. URL 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. URL 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. URL 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. URL 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. URL 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. URL 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. URL 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. URL 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. URL 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. URL https://arxiv.org/abs/2301.08556.

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

[42] 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. URL https://arxiv.org/abs/2403.03949.

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

[44] 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. URL https://arxiv.org/abs/2302.00111.

[45] 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. URL https://arxiv.org/abs/2406.13301.

[46] 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. URL https://arxiv.org/abs/2310.10639.

[47] L. Y. Chen, C. Xu, K. Dharmarajan, M. Z. Irshad, R. Cheng, K. Keutzer, M. Tomizuka, Q. Vuong, and K. Goldberg. Rovi-aug: Robot and viewpoint augmentation for cross-embodiment robot learning, 2024. URL https://arxiv.org/abs/2409.03403.

[48] 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*, SIGGRAPH '23. ACM, July 2023. doi:10.1145/3588432.3591516. URL http://dx.doi.org/10.1145/3588432.3591516.

[49] 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. URL https://arxiv.org/abs/2304.02643.

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

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

[52] N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, and K. Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. 2022.