Zero-Shot Object Manipulation with Semantic 3D Image Augmentation for Perceiver-Actor

***Abstract*— Recent advances in robot learning have shown promise in achieving multitask control and generalisation to novel scenarios—a feat previously difficult to achieve with hand-engineered solutions. However, these results are not as grandiose as those achieved by large models trained on internet-scale data; robot learning is still crucially limited by the bottleneck of real-world data collection. To breach this gap, we propose a data augmentation framework that utilises several large pretrained models to generate additional data from a limited set of human demonstrations. By combining pretrained image segmentation, image inpainting and depth estimation models, we can create new scenarios that are not seen in the dataset, but that are still consistent with the task setup. We demonstrate zero-shot capacity on a real robot, by training an agent on our augmented dataset to successfully manipulate objects that did not exist in the original collected data.**

I. INTRODUCTION

The observed trend in machine learning research is that with more compute and data, generalisation and better performance follow [1]. For example, there have been recent advances in joint vision-language understanding by scaling models on large datasets [2], [3], [4], [5]. While this trend hasn't dominated the field of robot learning, recent studies [6], [7] have shown similar performance gains can be achieved by scaling compute and data.

However, one downside of this approach—the reliance on large amounts of data—is a serious bottleneck for robotics. Unlike vision or language data, collecting data for robot learning is costly. And while the widely-used metric of data-efficiency does account for this somewhat, in robotics we also need to consider time-efficiency (in data collection) [8]. For example, RT-1 used 130,000 human teleoperated demonstrations, collected over the course of 17 months, with a fleet of 13 robots [6]. Even if significant generalisation required a single order of magnitude increase in data collection, this would be infeasible in any reasonable timeline.

However, large models pretrained on various modalities can be used to bootstrap robot learning. For example, the CLIP text-vision model [5] was combined with existing heuristic designs for robotics to create general controllers for manipulation [9], [10]. In another vein, works such as GenAug [11], CACTI [12] and ROSIE [13] have utilised similar large vision-language models, but for data augmentation. This line of work uses pretrained models to generate new scenes, alleviating the cost of collecting data.

Another line of work in robot learning has investigated using 3D representations for learning, which confers a useful inductive bias for spatial awareness [10], [14], [15]. These models can be very data-efficient, learning better policies with a magnitude less data than their 2D counterparts. However, the aforementioned data augmentation pipelines only augment 2D image data [12], [13], [11]. In this work, we extend data augmention to 3D data by combining pretrained vision and vision-language models (Fig. 1). In particular, we make use of pretrained image segmentation [16], [17], image inpainting [3] and depth estimation [18] models, which we show can be used on real robotics data without requiring further fine-tuning. We test our data augmentation pipeline on a real robot controlled with Perceiver-Actor (PerAct) [10], a recent language-conditioned imitation learning model that uses voxel representations. We show that our pipeline improves performance on both tasks seen in the data, as well as tasks only seen in the synthetic, augmented data.

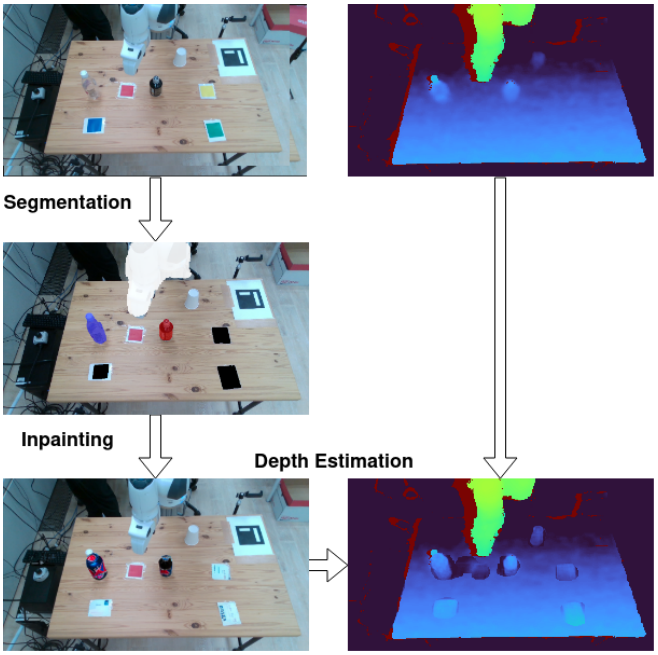

Fig. 1: Our 3D image data augmentation pipeline. Given an RGBD image from the real dataset (top), we first segment objects of interest (e.g., the target object, distractor objects, the goals, and the robot arm itself; middle-left), inpaint novel objects using the segmentation masks, composing objects in order as necessary (bottom-left), perform depth estimation on the new image, and then stitch this together with the original depth information (bottom-right) by using the segmentation mask to overwrite the inpainted novel objects depth, and other objects unmodified (including robot arm, table).

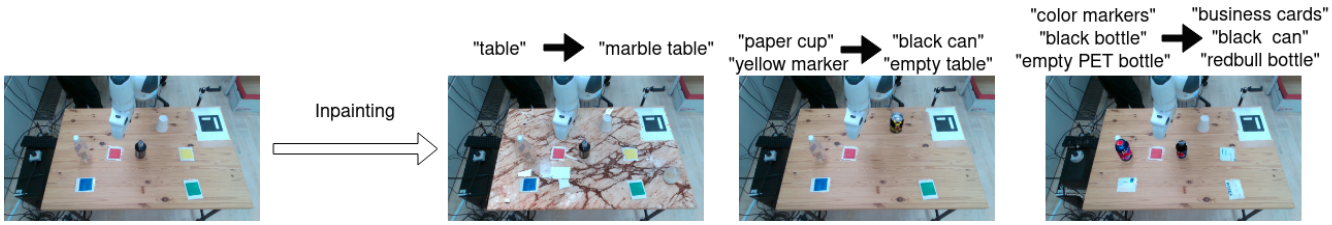

Fig. 2: Given a segmentation mask of an object, we can either change its appearance, or completely replace it with a different object. The text above each inpainted image denotes the segmentation and inpainting prompts, where we can use a composition of prompts.

II. BACKGROUND

In this work we use PerAct [10], a language-conditioned imitation learning model that takes voxels, robot proprioception values, and text that specifies the task to be executed, and outputs the desired gripper pose, gripper open/close status, and motion planner mode. This output fed in to a motion planner, which after execution queries the model for the next pose, in an open loop. The voxel inputs are constructed by taking RGBD images from a camera with known extrinsic parameters, forming a point cloud, and then creating a voxel grid from these with respect to the robot base.

The core of PerAct is the Perceiver IO architecture [19], which is used to mitigate the large memory requirement of processing voxels. All of its parameters are trained from scratch, apart from the language encoder, which uses a pretrained CLIP model [5]. PerAct is trained to predict the next action in a sequence of collected data, similar to behavioural cloning [20]. One key modification is that instead of predicting the next action at every step of the sequence, it instead predicts the next *key* action available in the episode, reducing the credit assignment problem[15].

III. METHOD

Augmenting PerAct's training data requires adjusting the vision and depth information from collected data to construct novel scenarios, as well as changing the text prompt to match the augmentation. We do so by chaining together pretrained models with different purposes, creating different scene representations whilst preserving relevant task-specific information, such as the location of the robot and the predicted action.

Our pipeline first starts with using a text-conditioned image segmentation model [16], [17]. We use text prompts to specify parts of the RGB image we want to segment: the workspace, the object we wish to manipulate, non-target objects, goal markers, and finally the robot arm itself. With these segmentation masks available, we then prompt an image inpainting model [3] with a new object/texture, together with the mask corresponding to the part of image we want to inpaint, repeating this process for each part, and finally composing these together, in order. Finally, we run a depth estimation model [18] on the composed inpainted image, and replace the parts of the depth image that inpainting has modified. The entire process can be seen in Fig. 1.

Since each of the models in the pipeline are independent, they can be interchanged as needed. We tried several pretrained text-conditioned image segmentation models [16], as well as the Segment Anything Model (SAM) [17]. As the pretrained SAM models are not text-conditioned, we used the largest model to propose segmentation masks for the entire image, and then used CLIP's image and text encoders [5] to match extracted objects to text queries. Given multiple masks, we used the Hungarian method to assign masks to our specified set of image segmentation labels. In our experiments we used both CLIPSeg [16] and SAM, as they each produced better masks for different tasks.

One of the strengths of the image segmentation + inpainting pipeline is the ability to change many aspects of the input (Fig. 2). Not only is it possible to change the objects that are manipulated, conferring "zero-shot" capabilities, but it is also possible to change the workspace, which can make the agent more robust to visual changes. We note that we keep the initial random seed of the inpainting model fixed across a trajectory, as it improves consistency of the inpainted objects.

The final part of our pipeline, depth estimation, does produce a consistent depth image, but it is often slightly misaligned with the original depth image. We correct for this by transforming the average estimated depth of the inpainted area to match the average real depth of the inpainted area. Finally, we re-paint parts of the image we do not wish to change as needed. This is task- and inpaint-dependent, but, for example, for most tasks we wish to keep the robot arm unaltered in the image.

IV. EXPERIMENTS

For our experiments we use a Franka Panda robot with its standard gripper, with an Intel D435i camera with resolution 720, 1280 to capture RGBD images. To integrate each modality we use ROS [21] and the MoveIt package [22] for motion planning, with the default RRT-Connect [23] set to Cartesian path planning. Demos were collected by specifying gripper positions for the motion planner with an handheld controller. The camera extrinsics parameters were calibrated using ArUco markers [24] and the default hand-eye calibration package from MoveIt.

All 3D information was processed with the robot in the center. Each sensory input was synchronised and sampled to 30Hz, and the data is collected under motion execution

and paused during human deciding the next end-effector goal position.

The original PerAct [10] model uses an heuristic algorithm to compute *key points* from the dataset, and uses the action from that data point as the learning target. This heuristic is ill-suited in the real-world data collection setting, when there's noise inherent in every sensor input. Instead, we record key points while collecting the data with the handheld controller. The keypoints are recorded after each execution, and when we open/close the gripper. Comparing the key-points generated this way to the heuristic algorithm, we could observe that the heuristic algorithm miss-classified certain data points as keypoints, compared to the hand-collected key points.

Due to hardware limitations for training PerAct, we changed some of the hyperparameters. We chose voxel grid size of 60^3, with hidden size of 512, a latent size of 512 due to hardware limitation. The original PerAct uses 100^3 [10] voxel grid size, but in the ablation, 64^3 had a close performance. To compensate for the noisy sensory input inherent in the real-world application, we added two further regularisation methods to improve PerAct's performance. Dropout on the point clouds at a rate of 30%, and additive noise on the robot joint positions ($\sim \mathcal{N}(0, 0.1)$) were added.

The task mimics that of the RLBench[25] push box to goal.

The environment for the collected data is as shown in the top left of Fig. 1: the robot is placed in front of a tabletop with 4 coloured markers and objects placed on the table. The goal of the task is to push a specified target object, to a given target goal marker, e.g., "push the black can to the green marker". We conduct training and evaluation in the following setup: "black can", a "empty bottle" and a "white cup" are placed, and demonstration is collected for target object "black can". We then augment this training dataset, by augmenting the "black can" to a "coca cola can", including its depth. We then evaluate two models, one trained with the original data only, and one with the additional augmented data.

We evaluate both models on the original setting, with the same objects. Then, we replace the target object with the unseen object, same prompt used for the data augmentation.

Both when collecting training data and evaluating the agent, the objects are randomly placed on the table such that the target object are not occluded, and not in the collision path with other objects to a given goal marker. Each setting were tested 20 times, 5 per target marker.

TABLE I: Success rate (%) of each experiment. Original target object is a PET bottle shaped, black (coffee) can. The novel target is a cola (red) can. Distractor object was kept the same (Empty plastic bottle and a white cup)

	Peract	with Data Augmentation
Original target object	75%	80%
Novel target object (zeroshot)	25 %	75 %

The result of the experiment is shown in table I. The model trained with the additional augmented data outperforms the original model in the zero-shot setting indicating that the additional augmented data does improve the models capacity, without lowering the performance on the original task.

During the experiment we noticed difference in behaviour between the two models. When the model trained only on the original data was evaluated with the novel object, it often directly moved the arm to the goal, ignoring the target object all together. The model trained with the additional data sometimes did go towards the distractor object (empty bottle) but to a position compensated for the height difference between the objects. Both models did struggle to push the target object to the goal when the target objects initial position was close to the goal, but not on top of it.

V. DISCUSSION

In this work, we set out to find out if 3D data augmentation, achieved via pretrained vision-language and RGB/D vision models could be applied to the robot learning setting, and furthermore if this could be used to achieve zero-shot transfer to scenarios unseen in the original training data. From our preliminary experiments, we can observe that data augmentation does slightly improve the models performance and also allows the model to adapt to tasks with unseen objects. The model struggled to identify a proper position when the target object was close to the goal, which could be caused by the smaller voxel size of 60^3 we decided to use for PerAct. While we show the capacity to inpaint more than the target object in figure 2, we couldn't test this capacity due to office constraints (no other table with different shape and colour available).

Because each component in our pipeline is independent, this type of data augmentation can utilise future advancement in large, pretrained models. However, we have to note that the pipeline is sensitive to the quality of the segmentation model. In our setup, the target object (bottle shaped coffee aluminium can) was a region specific object, and some segmentation models had trouble identifying the object.

To our surprise, the used segmentation models was good at identifying the robot arm. Since the camera position creates heavy occlusion of the target object during manipulation, the robust detection of the robot arm was utilised not just for task-consistency, but for preventing the depth estimate and inpaint of the target object to mistakenly modify unwanted regions. Therefore guaranteeing that the data augmentation won't create unrealistic situations, but sometimes create augmentation that doesn't differ too much from the original.

REFERENCES

[1] R. Sutton, "The bitter lesson," *Incomplete Ideas (blog)*, vol. 13, no. 1, 2019.

[2] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, "Hierarchical text-conditional image generation with clip latents," *arXiv preprint arXiv:2204.06125*, 2022.

[3] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 10 684–10 695.

[4] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, *et al.*, "Photorealistic text-to-image diffusion models with deep language understanding," *Advances in Neural Information Processing Systems*, vol. 35, pp. 36 479–36 494, 2022.

[5] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[6] 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. 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," in *arXiv preprint arXiv:2212.06817*, 2022.

[7] Y. Jiang, A. Gupta, Z. Zhang, G. Wang, Y. Dou, Y. Chen, L. Fei-Fei, A. Anandkumar, Y. Zhu, and L. Fan, "Vima: General robot manipulation with multimodal prompts," *arXiv preprint arXiv:2210.03094*, 2022.

[8] E. Johns, "Back to reality for imitation learning," in *Proceedings of the 5th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, A. Faust, D. Hsu, and G. Neumann, Eds., vol. 164. PMLR, 08–11 Nov 2022, pp. 1764–1768. [Online]. Available: https://proceedings.mlr.press/v164/johns22a.html

[9] M. Shridhar, L. Manuelli, and D. Fox, "CLIPort: What and where pathways for robotic manipulation," in *CoRL*, 2022, pp. 894–906.

[10] ——, "Perceiver-Actor: A multi-task transformer for robotic manipulation," in *CoRL*, 2023, pp. 785–799.

[11] Z. Chen, S. Kiami, A. Gupta, and V. Kumar, "Genaug: Retargeting behaviors to unseen situations via generative augmentation," *arXiv preprint arXiv:2302.06671*, 2023.

[12] 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," *arXiv preprint arXiv:2212.05711*, 2022.

[13] T. Yu, T. Xiao, A. Stone, J. Tompson, A. Brohan, S. Wang, J. Singh, C. Tan, D. M, J. Peralta, B. Ichter, K. Hausman, and F. Xia, "Scaling robot learning with semantically imagined experience," in *arXiv preprint arXiv:2302.11550*, 2023.

[14] S. James, K. Wada, T. Laidlow, and A. J. Davison, "Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 13 739–13 748.

[15] S. James and A. J. Davison, "Q-attention: Enabling efficient learning for vision-based robotic manipulation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1612–1619, 2022.

[16] T. Lüddecke and A. Ecker, "Image segmentation using text and image prompts," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 7086–7096.

[17] 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," *arXiv:2304.02643*, 2023.

[18] D. Kim, W. Ga, P. Ahn, D. Joo, S. Chun, and J. Kim, "Global-local path networks for monocular depth estimation with vertical cutdepth," *arXiv preprint arXiv:2201.07436*, 2022.

[19] A. Jaegle, S. Borgeaud, J.-B. Alayrac, C. Doersch, C. Ionescu, D. Ding, S. Koppula, D. Zoran, A. Brock, E. Shelhamer, *et al.*, "Perceiver IO: A general architecture for structured inputs & outputs," *arXiv:2107.14795*, 2021.

[20] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," *Advances in neural information processing systems*, vol. 1, 1988.

[21] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, vol. 3, no. 3.2, 2009, p. 5.

[22] S. Chitta, I. Sucan, and S. Cousins, "Moveit!" *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, pp. 18–19, 2012.

[23] J. J. Kuffner and S. M. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *ICRA*, vol. 2, 2000, pp. 995–1001.

[24] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J.

Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

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