

A Experiments

A.1 Physical Scene Recovery

The multiview method is conducted by capturing a short video of the scene using a handheld smartphone, then using Polycam¹ to perform 3D reconstruction from the video. Polycam processes the frames to generate a textured mesh of the environment, which is then used to extract object geometry for simulation. Figure 1 shows the policy learning environment of the task, along with the reconstructed meshes from both the single-view (RoLA) and multiview (Polycam-based) pipelines.

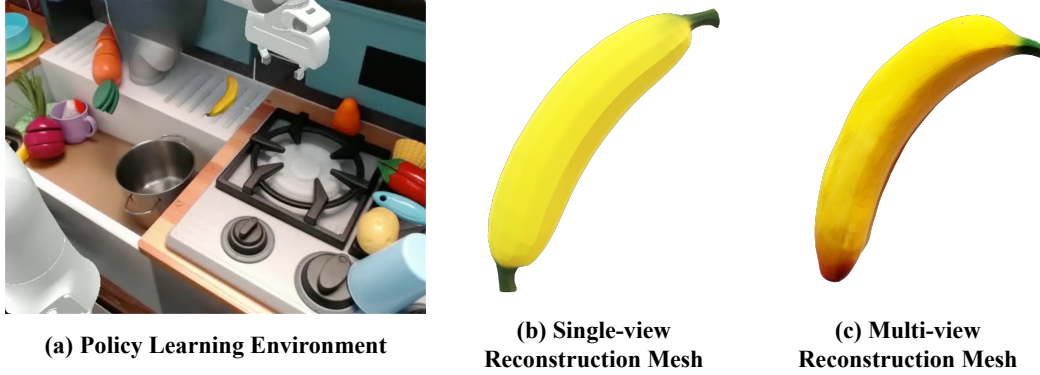


Figure 1: Comparison of the policy learning environment and the reconstructed scene meshes from the single-view RoLA pipeline and multiview Polycam pipeline.

A.2 Robotic Data Generation

(1) ACDC[1] is a retrieval-based, single-image data generation method. We follow its original pipeline with modifications to better suit our task. Given a single RGB image, ACDC performs four sequential steps. First, we extract per-object information by prompting GPT-4o² to generate captions for all visible objects. These captions are then passed to GroundedSAM[2] to produce object masks. Next, we compute the similarity between each masked object and entries in our object database using SigLip2 [3], retrieving the most visually similar “digital cousin” from OmniObject3D [4]. In the third step, we use metric depth information to align each retrieved mesh with the corresponding object and post-process the matched assets to construct a fully interactive simulated scene. Finally, we employ the same demonstration collection pipeline as RoLA to generate 200 demonstrations per task for imitation policy learning, ensuring fair comparison.

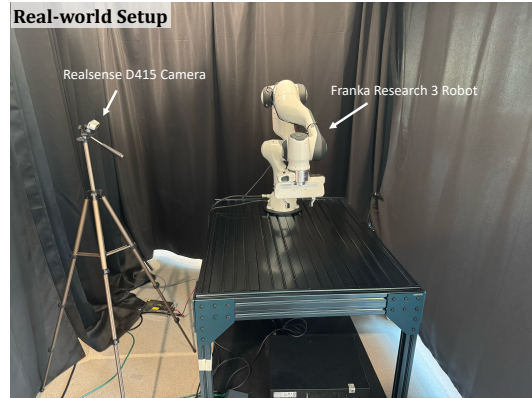


Figure 2: Real-world experiment setup. We employ a Franka Research 3 Robot and a RealSense D415 camera.

(2) RoboEngine [5] is an image-based demonstration augmentation method. We adopt the original codebase without modification. Given a single pre-collected demonstration, RoboEngine segments the robot arm and the manipulated object (i.e., the foreground), then uses a diffusion-based model to synthesize a physically plausible background conditioned on a foreground mask and a scene

¹<https://poly.cam/>

²<https://chatgpt.com/>

description. We apply this process to augment each demonstration into 200 demonstrations per task for imitation policy learning, enabling a consistent comparison with other methods.

Figure 3 shows the evaluation tasks (images on the left) and the comparison with the baseline methods. RoLA consistently outperforms the baselines in different tasks.

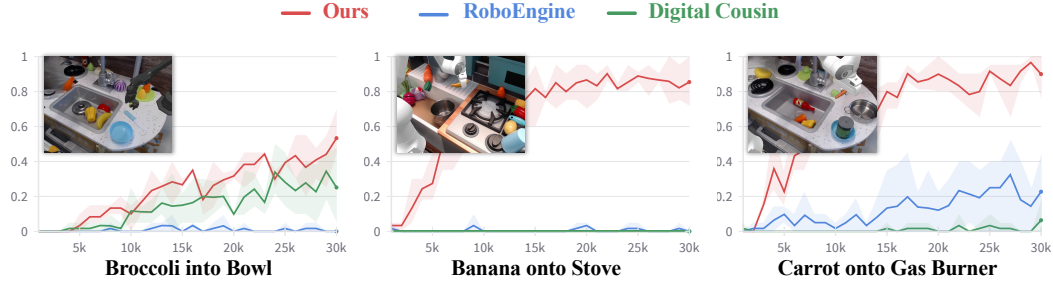


Figure 3: Baseline comparison for robotic data generation.

A.3 Single-Image Robot Learning

Figure 2 illustrates the setup of our real-world experiment. Figure 4 illustrates the comparison between RoLA-generated simulation data and the corresponding sim-to-real deployment in the real world on two tasks: *Manipulation in Cluttered Scenes* and *Pouring Water*.



Figure 4: **RoLA-Generated Data vs. Sim2Real Deployment.** Each row pair shows the simulated RoLA-generated scenes (top) and corresponding real-world executions (bottom), demonstrating strong sim-to-real consistency in object placement, robot behavior, and overall task execution.

41 A.4 VLA Model Training & Evaluation

42 To demonstrate the scalability of our data gen-
 43 eration pipeline, we train a vision-language-
 44 action (VLA) model by first capturing real-
 45 world photographs and then transforming them
 46 into robotic training data. We target the same
 47 scale as BridgeData V2 [6]. Specifically, we
 48 collect 2,000 real-world images across 12 dis-
 49 tinct environments (compared to 24 in Bridge-
 50 Data V2), a process completed within 2 hours
 51 of human effort—the only human interven-
 52 tion required. Next, we use GPT-4o to pro-
 53 pose approximately 10 tasks per image, yield-
 54 ing around 20,000 unique (image, language in-
 55 struction) pairs. Our pipeline RoLA is then
 56 used to generate approximately 60,000 robotic
 57 trajectories, matching the scale of BridgeData V2. Figure 6 displays example images captured in
 58 various environments.

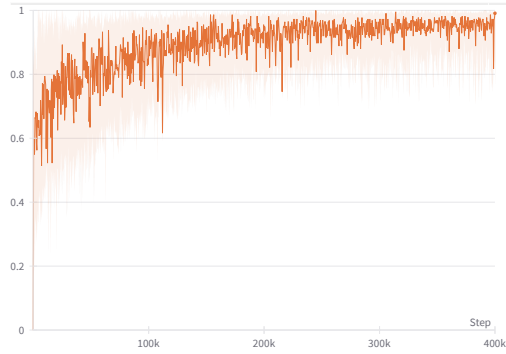


Figure 5: **Training Curve of VLA.** Action token accuracy steadily increases and surpasses 95% within 400K steps.



Figure 6: **Example Real-World Scenes.** Sample images captured across diverse environments for vision-language task specification.

59 For model architecture and training, we strictly follow the MiniVLA [7] setup. The model is trained
 60 for 400,000 steps until the action token accuracy exceeds 95%. Training is performed on 8xH100
 61 GPUs over four days (i.e., 768 GPU-hours). Figure 5 shows the accuracy progression over training
 62 steps.

63 **Evaluation.** For simulation-based evaluation, we closely follow the SimplerEnv [8] setup, with
 64 necessary modifications to the background image, manipulated object, camera pose, and language
 65 instruction to align with our task settings. We do not directly evaluate on the original SimplerEnv
 66 tasks because our model is not trained on any Open X-Embodiment [9] real-world data, making
 67 such a comparison unfair. For the same reason, we do not compare our model against models
 68 like OpenVLA [10], which are trained on Open X-Embodiment but not on any of our generated
 69 data—such a comparison would be unfair to OpenVLA.

Language Instruction	Success Rate	Language Instruction	Success Rate
pick up the banana (scene 1)	10/10	pick up the banana (scene 2)	7.5/10

Table 1: Simulation evaluation of our VLA model trained on RoLA-generated data. Each task is tested over 10 trials with randomized object poses, under SimplEnv [8] setting. Note that *partial success* (score of 0.5) is possible for some tasks, following OpenVLA [10].

We showcase two additional tasks in Table 1, both using the same language instruction: *pick up the banana*. These tasks are evaluated in distinct background environments to assess the model’s ability to generalize. The results show that our model can robustly handle varying visual contexts while correctly executing the instructed behavior.

Below are descriptions of all tasks that requires *partial success* (i.e., a score of 0.5):

- Pick up the Banana (Scene 2):** Partial credit is given when the robot successfully *grasps the banana* but fails to *lift* it.
- Put the Green Pepper beside the Lemon:** Partial credit is given when the robot successfully *picks up the green pepper* but does not perform the *place* action.
- Take the Banana and Place it beside the Potato:** Partial credit is given when the robot successfully *picks up the banana* but does not perform the *place* action.
- Pick up the Red Tomato and Place it beside the Yellow Lemon:** Partial credit is given when the robot successfully *picks up the red tomato* but does not perform the *place* action.
- Take the Grey Object beside the Lemon and Place it beside the Yellow Apple:** Partial credit is given when the robot successfully *picks up the grey object* but does not perform the *place* action.

A.5 Learning from Internet Images

We investigate how Internet images can serve as a data source for robot learning, using apple picking as a case study due to the resource constraints. Our motivation is that Internet images exhibit diverse apple types, sizes, environments, and camera viewpoints. If we can generate grasping data from these images, the learned policy could generalize across object variations, different viewpoints, and scenarios. However, due to the huge domain gap between Internet images and real-world robotic environments, direct zero-shot deployment of Internet-learned policy to the real world is infeasible. Instead, we adopt a pretraining–finetuning paradigm: we first pretrain a grasping policy on Internet-generated apple grasping data, then finetune it using a small number of real robot demonstrations for alignment and fast policy adaptation. An illustration of the whole process is shown in Figure 7.

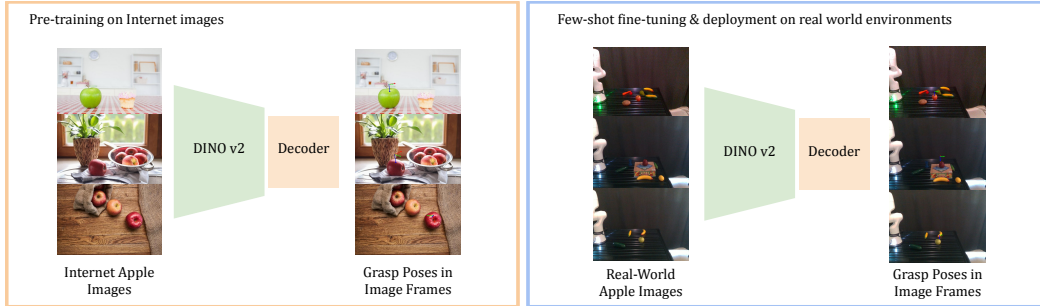


Figure 7: The pretraining-finetuning paradigm for learning from Internet images.

Pretraining. We collect massive Internet apple images, and leverage RoLA to generate robotic data for picking up the apples in images. We filter out those imperfect images and unsuccessful demonstrations, resulting in over 3000 high-quality demonstrations from Internet apple images. Since Internet images are captured from different camera viewpoints and the demonstrations are collected

100 by different robot positions, to unify the data representation, we extract a data sample of an image
101 and the corresponding 6-DoF apple grasp pose in the image frame for each demonstration, which
102 is agnostic to robot positions. Then, we pretrain a neural network that takes an image as input and
103 outputs the 6-DoF grasp pose in the image frame. The neural network contains a DINO v2 image
104 encoder and a small head for decoding the apple grasp pose. The trained network will serve as an
105 apple grasping prior for the subsequent real robot deployment.

106 **Finetuning.** In the real robotic environment, we collect 10/20/50 real robotic demonstrations for
107 picking up an apple, extract the respective camera image and apple grasping pose in the image frame,
108 and leverage these data to fine-tune the pretrained neural network. We set up a baseline by leveraging
109 the same network architecture (DINO v2 + decoder) and directly training the network on the real
110 robotic demonstrations without pretraining on the Internet apple images. During deployment, both
111 the baseline and our network take in the image from the RealScene camera and output the apple
112 grasping pose in the image frame. The grasping pose is then transformed into the robot base frame,
113 and we leverage a motion planner to move the robot gripper to the estimated grasping pose to pick
114 up the apple. We test the real-world grasping success rates and change the apple types, positions,
115 and camera viewpoints during evaluation.

116 **Robotic Demonstrations from Internet Images.** Figure 8 shows more examples of robotic demon-
117 strations generated from other Internet images. Our approach is generalizable to different types of
118 objects and images and enables robotic manipulation on in-the-wild images.

119 B Method

120 B.1 Background Geometry Modeling

121 Generating background meshes from images is challenging, as removing objects will leave a hole
122 in the background point clouds, and how to in-paint the geometry behind the object remains a prob-
123 lem. One solution is to have another depth estimation on the inpainted background image, but this
124 introduces geometry errors due to the inconsistency of depth estimations. In this paper, we leverage
125 the supported plane to complete the background geometry. Specifically, for pixels in the masked
126 object regions, we shoot a ray from each pixel and check the intersection point of the ray and the
127 supported plane. The intersection points will be used for creating background meshes. We also
128 check the intersections of the ray and the scene’s axis-aligned bounding boxes in case the ray does
129 not intersect with the supported plane.

130 B.2 Visual Blending

131 Figure 9 illustrates the detailed visual blending process. We generate foreground masks by compar-
132 ing the rendered and background depths. For foreground regions, we set the pixels to those in the
133 rendered frames. For background regions, we directly use the background image. This mitigates the
134 rendering artifacts and leads to more photo-realistic results.



Figure 8: Robotic demonstrations from Internet images.

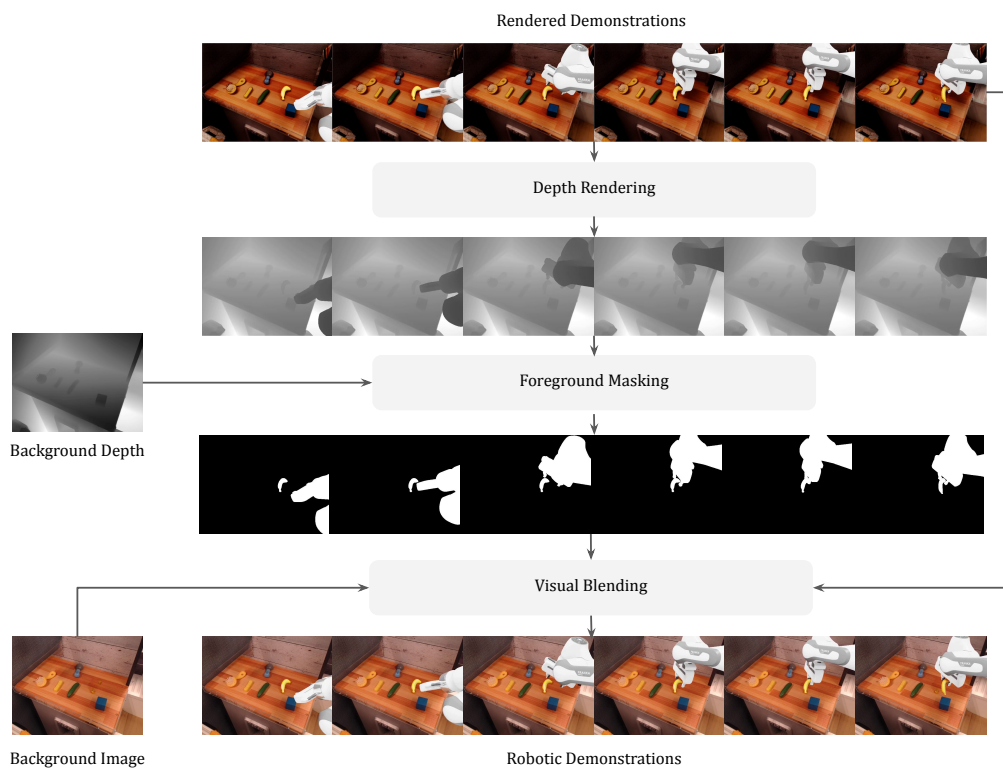


Figure 9: An illustration of visual blending.

References

- [1] T. Dai, J. Wong, Y. Jiang, C. Wang, C. Gokmen, R. Zhang, J. Wu, and L. Fei-Fei. Automated creation of digital cousins for robust policy learning. *CoRL*, 2024.
- [2] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, Z. Zeng, H. Zhang, F. Li, J. Yang, H. Li, Q. Jiang, and L. Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 2024.
- [3] M. Tschannen, A. Gritsenko, X. Wang, M. F. Naeem, I. Alabdulmohsin, N. Parthasarathy, T. Evans, L. Beyer, Y. Xia, B. Mustafa, et al. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.
- [4] T. Wu, J. Zhang, X. Fu, Y. Wang, L. P. Jiawei Ren, W. Wu, L. Yang, J. Wang, C. Qian, D. Lin, and Z. Liu. Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.
- [5] C. Yuan, S. Joshi, S. Zhu, H. Su, H. Zhao, and Y. Gao. Roboengine: Plug-and-play robot data augmentation with semantic robot segmentation and background generation. *arXiv preprint arXiv:2503.18738*, 2025.
- [6] H. R. Walke, K. Black, T. Z. Zhao, Q. Vuong, C. Zheng, P. Hansen-Estruch, A. W. He, V. Myers, M. J. Kim, M. Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *CoRL*, 2023.
- [7] S. Belkhale and D. Sadigh. Minivla: A better vla with a smaller footprint. 2024.
- [8] X. Li, K. Hsu, J. Gu, K. Pertsch, O. Mees, H. R. Walke, C. Fu, I. Lunawat, I. Sieh, S. Kirmani, et al. Evaluating real-world robot manipulation policies in simulation. *CoRL*, 2024.
- [9] A. O’Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.
- [10] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn. Openvla: An open-source vision-language-action model, 2024. URL <https://arxiv.org/abs/2406.09246>.