RISE: 3D Perception Makes Real-World Robot Imitation Simple and Effective

Chenxi Wang [1] **Hongjie Fang** [2] **Hao-Shu Fang** [2] **Cewu Lu** [2]

Abstract

Precise robot manipulations require rich spatial information in imitation learning, which remains a challenge in both 2D and 3D based policies. To tackle this problem, we present RISE, an end-to-end baseline for real-world imitation learning, which predicts continuous actions directly from single-view point clouds. It compresses the point cloud to tokens with a sparse 3D encoder. After adding sparse positional encoding, the tokens are featurized using a transformer. Finally, the features are decoded into robot actions by a diffusion head. Trained with 50 demonstrations for each real-world task, RISE surpasses currently representative 2D and 3D policies by a large margin, showcasing significant advantages in both accuracy and efficiency. Project website: rise-policy.github.io.

1. Introduction

Recent work has made significant strides in imitation learning in an end-to-end fashion (Brohan et al., 2023; Chi et al., 2023; Fang et al., 2024a;b; Zhao et al., 2023), which opens new possibilities for addressing complex manipulation tasks and drives research in the field of manipulation (Rahmatizadeh et al., 2018).

Spatial information is crucial for precise manipulations. Image-based imitation learning tends to learn implicit spatial representations from fixed camera views (Brohan et al., 2023; Chi et al., 2023; Fang et al., 2024a; Ha et al., 2023; Team et al., 2023; Zhao et al., 2023). Many of these approaches utilize distinct image encoders for each view and increase the number of cameras to enhance stability and precision, consequently increasing the number of network parameters and computational overhead.

Recently, imitation learning based on point clouds is drawing increasing interest in our community (Chen et al., 2023; Gervet et al., 2023; Goyal et al., 2023; Guhur et al., 2022; James et al., 2022; Shridhar et al., 2022; Xian et al., 2023; Ze et al., 2023). Most of the 3D-based methods learn to predict the next keyframe as opposed to continuous actions, which often struggle with tasks involving frequent contacts and abrupt environmental changes. Meanwhile, addressing the annotation of keyframes at scale for real-world data necessitates additional manual effort.

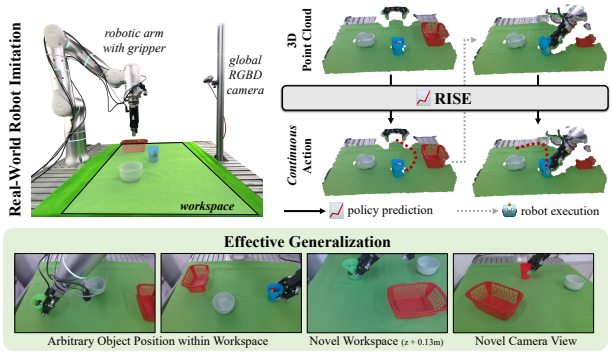


Figure 1. **RISE** focuses on real-world robot imitation settings with a noisy single-view partial point cloud as input, and outputs continuous robot actions. While simple, it shows effective generalization ability across object locations, novel workspaces, and novel camera views.

In this work, we propose an end-to-end imitation baseline, **RISE**, a method leveraging 3D perception to make real-world robot imitation simple and effective. RISE takes point clouds from a single-view RGB-D camera as input directly, and outputs continuous action trajectories.

We test RISE in 6 real-world tasks, where all the objects are randomly arranged throughout the entire workspace. Trained on 50 demonstrations for each task, RISE significantly outperforms other representative methods and keeps stable when the number of objects increases. We also find that RISE is more robust to environmental disturbance, which enhances error tolerance of real-world deployment.

2. Method

Given a point cloud $\mathcal{O}^t = \{P_i^t = (x_i^t, y_i^t, z_i^t, r_i^t, g_i^t, b_i^t)\}$ as the observation at time t, RISE aims to predict the next n-step robot actions $\mathcal{A}^t = \{A_{t+1}, A_{t+2}, \cdots, A_{t+n}\}$, where

[1]Shanghai Noematrix Intelligence Technology Ltd. [2]Shanghai Jiao Tong University. Correspondence to: Hao-Shu Fang <fhaoshu@gmail.com>, Cewu Lu <lucewu@sjtu.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

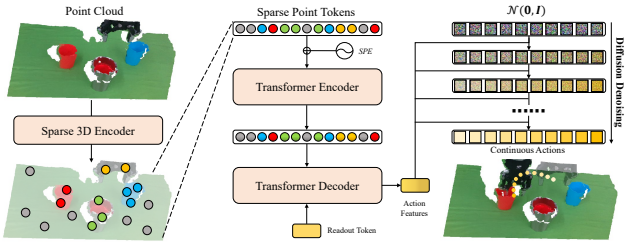


Figure 2. Overview of RISE architecture. The input of RISE is a noisy point cloud captured from the real world. A 3D encoder built with sparse convolution is employed to compress the point cloud into tokens. The tokens are fed into the transformer encoder after adding sparse positional encoding. A readout token is used to query the action features from the transformer decoder. Conditioned on the action features, the Gaussian samples are denoised into continuous actions iteratively using a diffusion head.

A_i contains the translation, rotation and width of the gripper. Due to the large domain gap between point clouds and robot actions, it is challenging to learn the approximation $f : \mathcal{O}^t \to \mathcal{A}^t$ directly. To model the process, RISE is decomposed into three functions: a sparse 3D encoder $h_{\mathrm{E}} : \mathcal{O}^t \to \mathcal{F}_{\mathrm{P}}^t$, a transformer $h_{\mathrm{T}} : \mathcal{F}_{\mathrm{P}}^t \to \mathcal{F}_{\mathrm{A}}^t$ and an action decoder $h_{\mathrm{D}} : \mathcal{F}_{\mathrm{A}}^t \to \mathcal{A}^t$, where $\mathcal{F}_{\mathrm{P}}^t$ and $\mathcal{F}_{\mathrm{A}}^t$ denote the features of point clouds and actions respectively. The overview of RISE architecture is illustrated in Fig. 2.

2.1. Modeling Point Clouds using Sparse 3D Encoder

The most significant difference between point cloud data and images is that point clouds are sparse and unorganized, which makes CNNs unsuitable to be applied to the points. For inputs at different scales, the computation efficiency and flexibility of a model should be taken into consideration. We employ a 3D encoder built on sparse convolution (Choy et al., 2019). It keeps most of the standard convolution, while only computes outputs on predefined coordinates. Such an operator saves computation and inherits the core advantage of conventional convolution.

The sparse 3D encoder h_{E} adopts a shallow ResNet architecture (He et al., 2016). It is composed of one initial convolution layer, four residual blocks, and one final convolution layer, with five $2\times$ sparse pooling layers between every two components. The number of layers can be freely increased, while the evaluation results demonstrate that a shallow encoder is sufficient for our experiments.

By h_{E}, the voxelized point cloud $\mathcal{O}^t$ is encoded to sparse point features $\mathcal{F}_{\mathrm{P}}^t$ in an efficient way, avoiding redundant computing on huge empty space. $\mathcal{F}_{\mathrm{P}}^t$ is then fed into the transformer h_{T} as sparse tokens. For $\mathcal{O}^t$ cropped in $1 \times 1 \times 1\mathrm{m}^3$ space, $\mathcal{F}_{\mathrm{P}}^t$ contains only $60 \sim 80$ tokens. Although the token number is less than the one in ACT (Zhao et al., 2023) (300 per image), experiments in §3.3 show that point cloud based ACT still outperforms the original implementation.

2.2. Transformer with Sparse Point Tokens

We adopt transformer (Vaswani et al., 2017) to implement the mapping from point features $\mathcal{F}_{\mathrm{P}}^t$ to action features $\mathcal{F}_{\mathrm{A}}^t$. While the positional encoding for image tokens is dense and natural, sparse point tokens could not be processed in the same manner. We instead introduce sparse positional encoding for point tokens.

Let (x, y, z) be the coordinate of the point token P with d-dimension, the position of P is defined as

$$\begin{aligned} pos_k &= \frac{k}{v} + c, \quad k \in \{x, y, z\}, \\ pos &= [pos_x, pos_y, pos_z], \end{aligned} \tag{1}$$

where c and v are fixed offsets, and $[\cdot]$ stands for vector concatenation. The encoding dimension along each axis is $d_x = d_y = \lfloor d/3 \rfloor$, $d_z = d - d_x - d_y$. The position encoding of P is computed by $SPE = [SPE^x, SPE^y, SPE^z]$ where

$$\begin{cases} SPE^k_{(pos,2i)} = \sin \dfrac{pos_k}{10000^{2i/d_k}} \\ SPE^k_{(pos,2i+1)} = \cos \dfrac{pos_k}{10000^{2i/d_k}} \end{cases}, \quad k \in \{x, y, z\} \tag{2}$$

With the help of sparse positional encoding, we effectively capture intricate 3D spatial relationships among unordered points, which enables seamless embedding of the 3D features into conventional transformers.

The transformer h_{T} utilizes an encoder-decoder architecture, taking point features $\mathcal{F}_{\mathrm{P}}^t$ as input tokens without other proprioceptive signals. In the transformer decoding step, we use one readout token to query action features $\mathcal{F}_{\mathrm{A}}^t$.

2.3. Diffusion as Action Decoder

The action decoder h_D is implemented as a denoising process by diffusion (Chi et al., 2023; Ho et al., 2020; Janner et al., 2022). Conditioning on $\mathcal{F}_{\mathrm{A}}^t$, h_{D} denoises the Gaussian noises $\mathcal{N}(0, \sigma^2 I)$ to actions $\mathcal{A}^t$ iteratively. The denoising process of step k is

$$\mathcal{A}_{k-1}^t = \alpha(\mathcal{A}_k^t - \gamma\epsilon_\theta(\mathcal{O}^t, \mathcal{A}_k^t, k) + \mathcal{N}(0, \sigma^2 I)), \tag{3}$$

where ϵ_θ is a network predicting noises with parameters θ, α, γ and σ are hyperparameters related to k in noise schedule. The objective function is the simplified objective in (Ho et al., 2020). We use the DDIM scheduler (Song et al., 2021) to accelerate the inference speed in real-world experiments.

The regression head is also frequently employed due to its simplicity (Gervet et al., 2023; Guhur et al., 2022; Jang et al., 2021; Zhao et al., 2023), whereas the diffusion head excels in handling scenes with multiple targets. Moreover,

diffusion produces diverse trajectories to the same target, as opposed to averaging learned trajectories (Chi et al., 2023).

For all tasks in our experiments, RISE adopts a unified action representation in the camera coordinate system, which is composed of translations, rotations, and gripper widths. We opt for absolute position for translation and 6D representation (Zhou et al., 2019) for rotation with continuity considerations.

3. Experiments

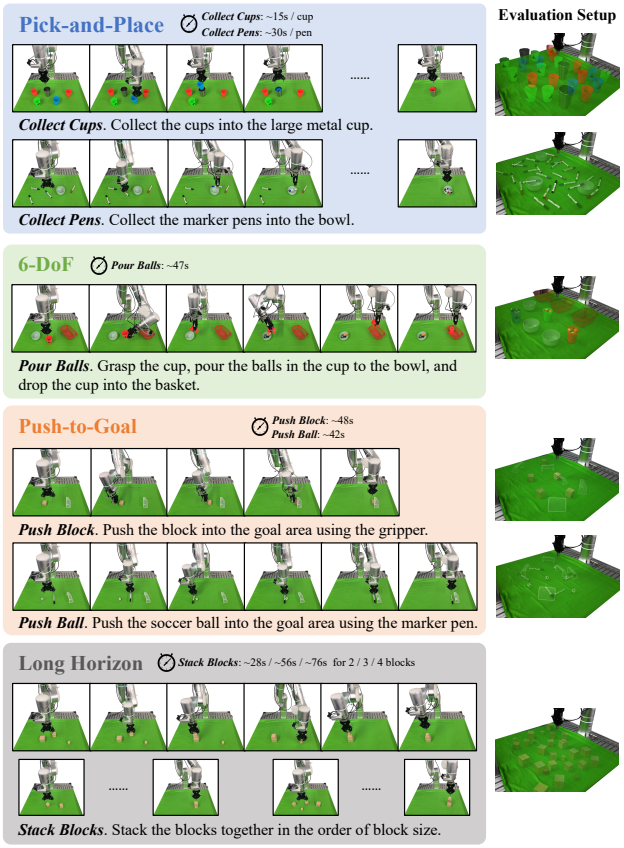


Figure 3. Definition of the tasks in the experiments. During evaluation, each task is randomly initialized within the robot workspace. For each task, only 3 to 5 evaluation setups are depicted for clarity.

3.1. Setup

Tasks. We designed 6 tasks for the experiments in Fig. 3.

Hardware. We use a Flexiv Rizon robotic arm with a Dahuan AG-95 gripper for interacting with objects. Two Intel RealSense D435 RGB-D cameras are installed for scene perception (One global, one inhand).

Baselines. We employ two representative image-based policies as our baselines: ACT (Zhao et al., 2023) and Diffusion Policy (Chi et al., 2023). We also evaluate a keyframe-based 3D policy Act3D (Gervet et al., 2023), the current state-of-the-art policy on RLBench (James et al., 2020).

Protocols. For 3D perception, only the global camera is used to generate a noisy single-view partial point cloud; while for image-based policies, both cameras are used for a better understanding of spatial geometries. We gathered 50 expert demonstrations for each task for training, and each policy was tested for 20 consecutive trials. During evaluations, objects in the task are randomly initialized within the robot workspace of approximately 50cm $\times$ 70cm. The evaluation time limit for each task is sufficient for each method to accomplish the task.

3.2. Results

Pick-and-place tasks are crucial in robotics, focusing on precise object manipulations and efficient policy generalization. The evaluation in Fig. 4 for ***Collect Cups*** and ***Collect Pens*** reveals RISE consistently outperforming all baselines, demonstrating its ability to not only predict the translation part but also accurately forecast planner rotation. We also discover that Act3D performs comparably to image-based baselines. Moreover, given that Act3D requires specially designed motion planners for more complicated actions and cannot provide immediate responses to sudden changes in the environment, we therefore only employ ACT and Diffusion Policy as baselines in our subsequent experiments.

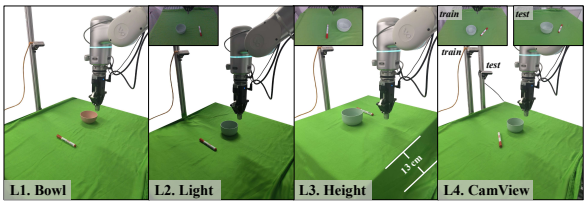
The *6-DoF* ***Pour Balls*** task assesses robot policies' capability in forecasting actions involving complex spatial rotations, unlike the simpler planner rotations in *pick-and-place* tasks. Tab. 1 presents the experimental results. RISE demonstrates superior learning of actions with intricate spatial rotations compared to image-based policies, as evidenced by higher action success rates. Moreover, its precision in pouring positions leads to increased task completion rates, highlighting the effectiveness of 3D perception in capturing accurate spatial object relationships.

For effective task completion, robot policies must promptly respond to environmental changes and adapt to object movements. We designed *push-to-goal* tasks, ***Push Block*** and ***Push Ball*** (Fig. 3), to assess this ability. Evaluation results in Tab. 1 show RISE slightly surpassing Diffusion Policy in the ***Push Block*** task, while significantly outperforming Diffusion Policy in the ***Push Ball*** task, demonstrating its adeptness in 3D perception for object positioning adjustments and swift policy action modifications.

Long-horizon tasks are essential in robotics, revealing how errors accumulate over extended actions and showcasing a policy's robustness and adaptability. Hence, we introduced the ***Stack Blocks*** task to evaluate this aspect, especially since block stacks are prone to toppling as they grow. Tab. 1 shows the experimental results. Initially, with just two blocks, all policies performed similarly. However, as the block

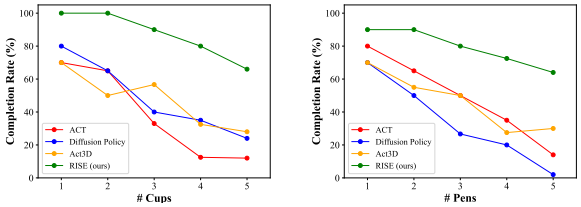
Method	*Pour Balls*					*Push Block*	*Push Ball*	*Stack Blocks*		
	Success Rate (%)			Completion Rate (%)		Success Rate (%)	Success Rate (%)	Completion Rate (%)		
	Grasp	Pour	Place	Overall	If Poured			1 block	2 blocks	3 blocks
ACT	30	30	0	13.0	43.3	-	-	60.0	25.0	10.0
Diffusion Policy	55	55	35	30.5	55.5	50	30	70.0	25.0	16.7
RISE (ours)	**80**	**80**	**70**	**49.0**	**61.3**	**55**	**60**	**80.0**	**75.0**	**30.0**

Table 1. Experimental results of the ***Pour Balls***, ***Push Block***, ***Push Ball*** and ***Stack Blocks*** task.



Method	Completion Rate (%)				
	Original	Disturbance			
		Bowl	Light	Height	CamView
ACT	80	70 ↓10	40 ↓40	0 ↓80	0 ↓80
Diffusion Policy	70	50 ↓20	30 ↓40	0 ↓70	0 ↓70
Act3D	70	40 ↓30	60 ↓10	50 ↓20	10 ↓60
RISE (ours)	**90**	**80** ↓10	**80** ↓10	**80** ↓10	**50** ↓40

Table 2. Generalization test setup and experimental results of the ***Collect Pens*** task with 1 pen (10 trials).



(a) ***Collect Cups*** (b) ***Collect Pens***

Figure 4. Experimental results of the *pick-and-place* tasks.

Method	3D	# Cameras	Completion Rate (%)
ACT		2	12
	✓	1	32 ↑20
Diffusion Policy		2	24
	✓	1	36 ↑12
DP3*	✓	1	-
Act3D	✓	1	28
RISE (ours)	✓	1	**66**

Table 3. Effectiveness test of 3D perception on the ***Collect Cups*** task with 5 cups (10 trials). 2D version of policies take images from both global and in-hand cameras as input. * DP3 fails to learn in our setting, see appendix for a more detailed analysis.

count increased, RISE notably outperformed the baselines, demonstrating its strong adaptability to *long-horizon* tasks and ability to effectively control accumulated errors.

3.3. Effectiveness of 3D Perception

In this section, we explore how 3D perception enhances the performance of robot manipulation policies on the ***Collect Cups*** task with 5 cups. We replace the image encoder of the image-based policies ACT and Diffusion Policy with the sparse 3D encoder used in RISE. The experiment results are shown in Tab. 3. We observe a significant improvement in the performance of ACT and Diffusion Policy after applying 3D perception even with fewer camera views, surpassing the 3D policy Act3D, which reflects the effectiveness of our 3D perception module in manipulation policies.

We also evaluate the recently proposed DP3 (Ze et al., 2024) in this experimental setting. However, DP3 appears to struggle to learn meaningful actions from our demonstration data. Please refer to the appendix for detailed analyses.

3.4. Generalization Test

We assess the generalization abilities of different methods using the ***Collect Pens*** task with 1 pen under various environmental disturbances detailed in Tab. 2, including different objects (L1), different light conditions (L2), different workspaces (L3) and different camera viewpoints (L4). The results in Tab. 2 indicate that image-based policies achieve decent L1 and some L2-level generalizations but fall short in L3 and L4-level generalizations involving spatial transformations. Act3D, as a 3D policy, shows good generalization up to L3-level disturbances but struggles significantly in L4-level tests. On the contrary, RISE demonstrates strong generalization across all testing levels, even excelling in the challenging L4-level tests involving camera view changes.

4. Conclusion

In this paper, we present RISE, an efficient end-to-end policy utilizing 3D perception for real-world robot manipulation. RISE compresses point clouds with a sparse 3D encoder, followed by sparse positional encoding and a transformer to obtain action features. The features are decoded into continuous actions by a diffusion head. RISE significantly outperforms currently representative 2D and 3D policies in multiple tasks, demonstrating great advantages in both accuracy and efficiency. Our ablations verify the effectiveness of 3D perception and the generalization of RISE under different levels of environmental disturbances. We hope our baseline inspires the integration of 3D perception into real-world policy learning.

References

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J. Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael S. Ryoo, Grecia Salazar, Pannag R. Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: robotics transformer for real-world control at scale. In *Robotics: Science and Systems*, 2023.

Shizhe Chen, Ricardo Garcia Pinel, Cordelia Schmid, and Ivan Laptev. Polarnet: 3d point clouds for language-guided robotic manipulation. In *Conference on Robot Learning*, pages 1761–1781. PMLR, 2023.

Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems*, 2023.

Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019.

Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5828–5839, 2017.

Danny Driess, Ingmar Schubert, Pete Florence, Yunzhu Li, and Marc Toussaint. Reinforcement learning with neural radiance fields. *Advances in Neural Information Processing Systems*, 35:16931–16945, 2022.

Hao-Shu Fang, Hongjie Fang, Zhenyu Tang, Jirong Liu, Chenxi Wang, Junbo Wang, Haoyi Zhu, and Cewu Lu. Rh20t: A comprehensive robotic dataset for learning diverse skills in one-shot. In *IEEE International Conference on Robotics and Automation*, 2024a.

Hongjie Fang, Hao-Shu Fang, Yiming Wang, Jieji Ren, Jingjing Chen, Ruo Zhang, Weiming Wang, and Cewu Lu. Airexo: Low-cost exoskeletons for learning whole-arm manipulation in the wild. In *IEEE International Conference on Robotics and Automation*, 2024b.

Théophile Gervet, Zhou Xian, Nikolaos Gkanatsios, and Katerina Fragkiadaki. Act3d: 3d feature field transformers for multi-task robotic manipulation. In *Conference on Robot Learning*, pages 3949–3965. PMLR, 2023.

Ankit Goyal, Jie Xu, Yijie Guo, Valts Blukis, Yu-Wei Chao, and Dieter Fox. RVT: robotic view transformer for 3d object manipulation. In *Conference on Robot Learning*, pages 694–710. PMLR, 2023.

Benjamin Graham, Martin Engelcke, and Laurens Van Der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9224–9232, 2018.

Pierre-Louis Guhur, Shizhe Chen, Ricardo Garcia Pinel, Makarand Tapaswi, Ivan Laptev, and Cordelia Schmid. Instruction-driven history-aware policies for robotic manipulations. In *Conference on Robot Learning*, pages 175–187. PMLR, 2022.

Huy Ha, Pete Florence, and Shuran Song. Scaling up and distilling down: Language-guided robot skill acquisition. In *Conference on Robot Learning*, pages 3766–3777. PMLR, 2023.

Abdullah Hamdi, Silvio Giancola, and Bernard Ghanem. Mvtn: Multi-view transformation network for 3d shape recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1–11, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

Stephen James, Kentaro Wada, Tristan Laidlow, and Andrew 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*, pages 13739–13748, 2022.

Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. BC-Z: zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2021.

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pages 9902–9915. PMLR, 2022.

Bo Li, Tianlei Zhang, and Tian Xia. Vehicle detection from 3d lidar using fully convolutional network. In *Robotics: Science and Systems*, 2016.

Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. LIV: language-image representations and rewards for robotic control. In *International Conference on Machine Learning*, pages 23301–23320. PMLR, 2023a.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. VIP: towards universal visual reward and representation via value-implicit pre-training. In *International Conference on Learning Representations*, 2023b.

Arjun Majumdar, Karmesh Yadav, Sergio Arnaud, Yecheng Jason Ma, Claire Chen, Sneha Silwal, Aryan Jain, Vincent-Pierre Berges, Pieter Abbeel, Jitendra Malik, et al. Where are we in the search for an artificial visual cortex for embodied intelligence? In *ICRA 2023 Workshop on Pretraining for Robotics*, 2023.

Ajay Mandlekar, Yuke Zhu, Animesh Garg, Jonathan Booher, Max Spero, Albert Tung, Julian Gao, John Emmons, Anchit Gupta, Emre Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pages 879–893. PMLR, 2018.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning*, pages 1678–1690. PMLR, 2021.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3M: A universal visual representation for robot manipulation. In *Conference on Robot Learning*, pages 892–909. PMLR, 2022.

Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

Xuran Pan, Zhuofan Xia, Shiji Song, Li Erran Li, and Gao Huang. 3d object detection with pointformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7463–7472, 2021.

Jyothish Pari, Nur Muhammad (Mahi) Shafiullah, Sridhar Pandian Arunachalam, and Lerrel Pinto. The surprising effectiveness of representation learning for visual imitation. In *Robotics: Science and Systems*, 2022.

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems*, volume 1, 1988.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Advances in Neural Information Processing Systems*, 35:23192–23204, 2022.

Ilija Radosavovic, Tete Xiao, Stephen James, Pieter Abbeel, Jitendra Malik, and Trevor Darrell. Real-world robot learning with masked visual pre-training. In *Conference on Robot Learning*, pages 416–426. PMLR, 2022.

Rouhollah Rahmatizadeh, Pooya Abolghasemi, Ladislau Bölöni, and Sergey Levine. Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration. In *IEEE International Conference on Robotics and Automation*, pages 3758–3765. IEEE, 2018.

Scott E. Reed, Konrad Zolna, Emilio Parisotto, Sergio Gómez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando

de Freitas. A generalist agent. *Transactions on Machine Learning Research*, 2022.

Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3577–3586, 2017.

William Shen, Ge Yang, Alan Yu, Jansen Wong, Leslie Pack Kaelbling, and Phillip Isola. Distilled feature fields enable few-shot language-guided manipulation. In *Conference on Robot Learning*, pages 405–424. PMLR, 2023.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2022.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in Neural Information Processing Systems*, 28, 2015.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *The International Conference on Learning Representations*, 2021.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, et al. Octo: An open-source generalist robot policy, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Chen Wang, Haochen Shi, Weizhuo Wang, Ruohan Zhang, Li Fei-Fei, and C. Karen Liu. Dexcap: Scalable and portable mocap data collection system for dexterous manipulation. *arXiv preprint arXiv:2403.07788*, 2024.

Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions On Graphics*, 36(4):1–11, 2017.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1912–1920, 2015.

Zhou Xian, Nikolaos Gkanatsios, Théophile Gervet, Tsung-Wei Ke, and Katerina Fragkiadaki. Chaineddiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation. In *Conference on Robot Learning*, pages 2323–2339. PMLR, 2023.

Jianglong Ye, Naiyan Wang, and Xiaolong Wang. Featurenerf: Learning generalizable nerfs by distilling foundation models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8962–8973, 2023.

Yanjie Ze, Ge Yan, Yueh-Hua Wu, Annabella Macaluso, Yuying Ge, Jianglong Ye, Nicklas Hansen, Li Erran Li, and Xiaolong Wang. Gnfactor: Multi-task real robot learning with generalizable neural feature fields. In *Conference on Robot Learning*, pages 284–301. PMLR, 2023.

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. *arXiv preprint arXiv:2403.03954*, 2024.

Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *IEEE International Conference on Robotics and Automation*, pages 5628–5635. IEEE, 2018.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16259–16268, 2021.

Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems*, 2023.

Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019.

Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018.

Yifeng Zhu, Zhenyu Jiang, Peter Stone, and Yuke Zhu. Learning generalizable manipulation policies with object-centric 3d representations. In *7th Annual Conference on Robot Learning*, 2023.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng

Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183, 2023.

Appendix

1. Related Work

1.1. Imitation Learning for Robotics

Imitation learning is a machine learning paradigm where a robot learns to operate by observing and mimicking expert demonstrations. Behavior cloning (BC) (Pomerleau, 1988), as the most direct form of imitation learning, aims to identify a mapping from observations to corresponding robot actions with the supervision of the given demonstrations. Despite its simplicity, BC has shown promising potential in learning robotic manipulations (Brohan et al., 2023; Chi et al., 2023; Jang et al., 2021; Mandlekar et al., 2021; Shridhar et al., 2022; Team et al., 2023; Zhao et al., 2023).

2D Imitation Learning. 2D image data is commonly used in imitation learning. One intuitive approach is to utilize pre-trained representation models for images (Ma et al., 2023a;b; Majumdar et al., 2023; Nair et al., 2022; Radosavovic et al., 2022) to convert them into 1D representations and map these transformed observations to the action space either through a BC policy (Zhang et al., 2018) or non-parametric nearest neighbour (Pari et al., 2022). Unfortunately, current pre-trained representation models are not general enough to handle diverse experimental environments, encountering trouble achieving satisfactory results in real-world settings. Thus, many researchers learn such mapping in an end-to-end manner (Brohan et al., 2023; Chi et al., 2023; Jang et al., 2021; Mandlekar et al., 2021; Padalkar et al., 2023; Reed et al., 2022; Team et al., 2023; Zhao et al., 2023; Zitkovich et al., 2023) and have demonstrated impressive performance across many tasks. Specifically, ACT (Zhao et al., 2023) adopts a CVAE scheme (Sohn et al., 2015) with transformer backbones (Vaswani et al., 2017) and ResNet image encoders (He et al., 2016) to model the variability of human data, while Diffusion Policy (Chi et al., 2023) directly utilizes diffusion process (Ho et al., 2020) to express multimodal action distributions generatively. Nonetheless, these policies are sensitive to camera positions and often fail to capture 3D spatial information about the objects in the environments.

3D Imitation Learning. The formulation of incorporating 3D information in the imitation learning framework is under active exploration. The most straightforward method is to apply projections to transform the 3D point cloud to several 2D image views and transfer the task to multi-view image-based policy learning (Goyal et al., 2023; Guhur et al., 2022), which requires the virtual viewpoints to be carefully designed to ensure performance. Moreover, due to sparse and noisily sensed point clouds, (Goyal et al., 2023) fails to grasp slim objects like marker pens in real-world experiments. (James et al., 2022; Shridhar et al., 2022; Ze et al., 2023) process point clouds to dense voxel grids and apply 3D convolutions. Since high-resolution 3D feature maps require expensive computes, these methods have to trade off performance against cost. (Zhu et al., 2023) proposed an object-centric representation for learning which requires an additional segmentation process. (Gervet et al., 2023; Xian et al., 2023) featurize point clouds by projecting multi-view image features to 3D world to avoid dense convolutions. However, such feature fusion techniques struggle to capture the consistent 3D representation from different views accurately. Recently, a concurrent work DP3 (Ze et al., 2024) also leverages 3D perception in robotic manipulation policies, but our real-world evaluations in §3.3 demonstrate that it cannot handle demonstrations with various representations limited by its network capacity. DexCap (Wang et al., 2024) also proposed a PointNet (Qi et al., 2017) with diffusion head architecture for dexterous manipulation.

As mentioned before, most of the current 3D robotic imitation learning methods predict keyframes instead of continuous action, which makes it hard to annotate and limits their capacity. Besides, many of these methods only show results in simulation environments like RLBench (James et al., 2020) and CALVIN (Mees et al., 2022). In this work, we aim to evaluate our method in a more challenging setting: continuous action control with a noisy single-view partial point cloud in the real world.

1.2. 3D Perception

3D perception has received considerable attention from researchers in the computer vision and robotics communities. It can be roughly divided into the following three categories:

Projection-based. This approach initially projects the 3D point cloud onto multiple images on different planes and then employs traditional multi-view image perception techniques. It is widely applied in shape recognition (Hamdi et al., 2021), object detection (Chen et al., 2017; Li et al., 2016) and robotic manipulations (Goyal et al., 2023; Guhur et al., 2022) due to its simplicity. However, the projections can lead to the geometric information loss of the 3D data, and the sensitivity to the choice of projection planes may result in inferior performance (Zhao et al., 2021).

Point-based. Early researchers directly utilized 3D convolutional neural networks (CNNs) to process 3D point cloud data based on dense volumetric representations (Dai et al., 2017; Wu et al., 2015; Zhou and Tuzel, 2018). Still, the sparsity of 3D data makes the vanilla approaches inefficient and memory-intensive. To solve this problem, researchers have explored using octrees for memory footprint reduction (Riegler et al., 2017; Wang et al., 2017), utilizing sparse convolutions to minimize unnecessary computations in inactive regions to improve efficiency and effectiveness (Choy et al., 2019; Graham et al., 2018), and aggregating features across point sets directly using different network architectures (Pan et al., 2021; Qi et al., 2017; Qian et al., 2022; Zhao et al., 2021).

NeRF-based. Neural radiance fields (NeRFs) (Mildenhall et al., 2021) have demonstrated impressive performance on high-fidelity 3D scene synthesis and scene representation extractions. In recent years, some studies (Driess et al., 2022; Shen et al., 2023; Ye et al., 2023; Ze et al., 2023) have employed features extracted from pre-trained 2D foundational models as additional supervisory signals in NeRF training for scene feature extraction and distillation. Nevertheless, NeRF training requires image data from multiple views, which poses obstacles for scaling up in real-world environments. Additionally, it does not align with our single-view setting.

2. Experiment Details

2.1. Tasks Parameters

We list the parameters of the demonstrations for different tasks in this paper in Tab. 4. The axis-wise action representation is implemented with keyboard teleoperation (one key to control movement, or rotation, or gripper action in each direction). We observe that although the axis-wise action representation results in fewer steps during demonstrations, its teleoperation time was approximately 3x as long as that of the natural teleoperation, aligning with the findings in (Mandlekar et al., 2018).

The evaluation settings for different tasks are summarized in Tab. 5. Compared to the average steps in demonstrations, the maximum steps in evaluations prove to be sufficient.

2.2. Implementation Details

Data Processing. The point cloud is created from a single-view RGB-D image. Both input point clouds and output actions are in the camera coordinate system. We crop the point clouds with the range of $x, y \in [-0.5\text{m}, 0.5\text{m}]$, $z \in [0\text{m}, 1\text{m}]$, and normalize the translation values to $[-1, 1]$ with the range of $x, y \in [-0.35\text{m}, 0.35\text{m}]$, $z \in [0\text{m}, 0.7\text{m}]$. The gripper width is normalized to $[-1, 1]$ using the range of $[0\text{m}, 0.11\text{m}]$.

Task Name	Notes	# Demos	Avg. Steps	Avg. Teleop. Time (s)
Collect Cups	1 cup	10	117.4	19.37
	2 cups	10	225.0	34.73
	3 cups	10	345.3	54.84
	4 cups	10	451.4	71.07
	5 cups	10	520.0	76.02
	* 1 cup, natural action	50	102.7	17.06
	* 1 cup, axis-wise action	50	30.2	45.93
Collect Pens	1 pen	10	179.4	52.47
	2 pens	10	278.2	62.71
	3 pens	10	411.5	91.88
	4 pens	10	556.1	124.22
	5 pens	10	694.1	157.15
Pour Balls		50	185.4	50.69
Push Block		50	204.3	51.72
Push Ball		50	223.1	46.00
Stack Blocks	2 blocks	10	148.6	32.06
	3 blocks	20	286.2	59.22
	4 blocks	20	401.8	79.83

Table 4. Parameters of the collected demonstrations for different tasks. "Avg. Teleop. Time" stands for the average teleoperation time for collecting one demonstration. * denotes that these data are only used for the comparison experiments with DP3.

Task Name	Notes	# Trials	Max. Steps	Max. Keyframes
Collect Cups	1 cup	10	300	20
	2 cups	10	600	40
	3 cups	10	900	60
	4 cups	10	1200	80
	5 cups	10	1500	100
Collect Pens	1 pen	10	300	20
	2 pens	10	600	40
	3 pens	10	900	60
	4 pens	10	1200	80
	5 pens	10	1500	100
Pour Balls		20	1200	N/A
Push Block		20	1200	N/A
Push Ball		20	1200	N/A
Stack Blocks	2 blocks	10	600	N/A
	3 blocks	10	1200	N/A
	4 blocks	10	1800	N/A

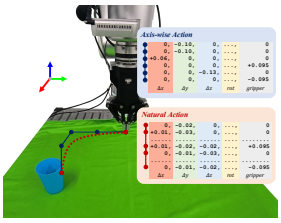
Table 5. Evaluation settings for different tasks.

Network. The sparse 3D encoder is implemented based on MinkowskiEngine (Choy et al., 2019) with a voxel size of 5mm, which outputs a set of point feature vectors at the dimension of 512. For sparse positional encoding, we set $v = 5\text{mm}$ and $c = 400$. The transformer contains 4 encoding blocks and 1 decoding block, with $d_{\text{model}} = 512$ and $d_{\text{ff}} = 2048$. The dimension of the readout token is 512. We employ a CNN-based diffusion head (Chi et al., 2023) with 100 denoising iterations for training and 20 iterations for inference. The output action horizon is 20.

Training. RISE is trained on 2 Nvidia A100 GPUs with a batch size of 240, an initial learning rate of 3e-4, and a warmup step of 2000. The learning rate is decayed by a cosine scheduler. During training, the point clouds are randomly translated by $[-0.2\text{m}, 0.2\text{m}]$ along X/Y/Z-axis, and randomly rotated by $[-30°, 30°]$ around X/Y/Z-axis.

Baseline. ACT (Zhao et al., 2023), Diffusion Policy (Chi et al., 2023), Act3D (Gervet et al., 2023) and DP3 (Ze et al., 2024) are trained based on the official implementations. The Diffusion Policy baseline takes ResNet18 as the visual encoder and adopts a CNN-based backbone. For the Act3D baseline, we implement a simple planner for *pick-and-place* tasks to avoid collisions, which decouples an action into a horizontal one and a vertical one. It follows a heuristic rule: the horizontal action precedes the downward one while it follows the upward one. For ACT (3D), we replace the image tokens with the point tokens. For Diffusion Policy (3D), we employ an AvgPooling layer to get the observation embedding from point features.

2.3. Analyses of DP3 Failure



Method	Completion Rate (%)	
	Axis-wise	Natural
DP3, hor. 4	0	0
DP3, hor. 8	0	0
DP3, hor. 16	20	0
DP3, hor. 24	40	0
RISE	**80**	**100**

Table 6. Analysis of the failures of DP3 in our experiments. (left) Illustrations of the axis-wise and natural action. (right) Results of the ***Collect Cups*** task with 1 cup when using demonstrations with different action representations for training (10 trials).

After communicating with the authors of DP3, one potential reason is that they are using RealSense L515 in their original experiments, and we adopted RealSense D435 in our experiments. The point cloud from D435 is noisier, making it more challenging for networks to learn. By using sparse convolution, RISE is more robust to the noise in the point cloud. Besides, after delving into their real robot experiments, we found that instead of natural actions, axis-wise actions are used in their demonstration data, as illustrated in Tab. 6 (left). Hence, we collect 50 demonstrations on the ***Collect Cups*** task with 1 cup using axis-wise and natural action representations respectively. These demonstrations are then used for DP3 policy training. After carefully tuning hyper-parameters such as horizons and color utilizations, we report the evaluation results in Tab. 6, with the best completion rate of 40%. We suspect that the limited network capacity of the 3D encoder of DP3 prevents it from modeling the diverse state-action pairs present in the real-world human teleoperated demonstrations, leading it to only handle a smaller set of state-action pairs under the axis-wise action representations. On the contrary, RISE can handle real-world demonstrations with various action representations and maintain satisfactory performances. Lastly, compared to the evaluation setup in the DP3 paper, we allow objects to be placed anywhere in the entire workspace. This results in a greater variation of object locations, making the task more challenging.

2.4. Discussions about Action Representations

Axis-wise. (Tab. 6 (blue)) Axis-wise action representation assumes that only one axis-wise movement is conducted in each step (typically one of the translations along the X/Y/Z-axis, the rotation around the X/Y/Z-axis, and the gripper opening/closing). Demonstrations with axis-wise action representations are usually collected via teleoperation with low frequency, like keyboard teleoperations.

Natural. (Tab. 6 (red)) Natural action representation allows composite movement patterns in each step (that is, the robot can simultaneously translate, rotate, and open or close the gripper in one step). Demonstrations with natural action representations are usually collected via teleoperations with high frequency, like teleoperations with haptic devices.

Discussions. Due to only one non-zero value for each action at any step, axis-wise action representations are easy to learn. However, this ease of learning can introduce noticeable induction biases in the learned policy, resulting in a lack of action diversity. Moreover, the axis-wise action representation increases the difficulty of representing complex trajectories, resulting in the limited generalization capability of the learned policy. On the contrary, the natural action representation is more challenging to learn than the axis-wise action representation, the learned policy can exhibit more natural action trajectories. Furthermore, the natural action representation aligns more closely with the patterns of human action execution, thus adopting natural actions can enhance data collection efficiency, as illustrated in Tab. 4. Therefore, we adopt natural action representation in our collected real-world demonstrations.

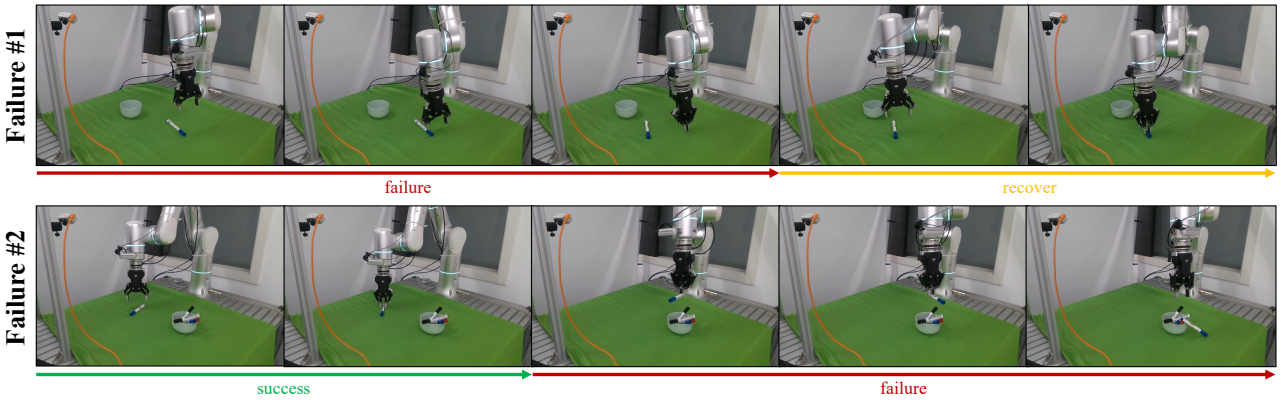


Figure 5. Failure cases of the ***Collect Pens*** task in the experiments.

2.5. Failure Cases and Recovery

In this section, we take the ***Collect Pens*** task as an example and illustrate the failure cases of RISE during experiments in Fig. 5. We observe that failure cases are mainly caused by inaccurate positions during picking (Failure #1) and placing (Failure #2). We found that RISE can automatically correct some failure scenarios, such as instances where the pen is inadvertently moved due to imprecise positioning during grasping (Failure #1). In contrast, many keyframe-based methods (Gervet et al., 2023; Shridhar et al., 2022; Xian et al., 2023) lack the ability to offer immediate recovery actions for failures, potentially leading to the exacerbation of errors.