iManip: Skill-Incremental Learning for Robotic Manipulation

Supplementary Material

1. Experiments on key hyperparameters

We conducte experiments in RLbench on four key hyperparameter: the number of replays per keyframe, the length of each action prompt, the hyperparameter of distillation loss factor λ_{dis}, and the dimension of the appended PerceiverIO weight matrix d_{new}, using the B5-1N1 and B5-5N1 setups.

Exploring the amount of replays per keyframe. For each new manipulation skill, 20 training trajectories are used for training. Specifically, as shown in Figure 1 (a), we vary the replay size from 1 to 5 per keyframe based on the temporal replay strategy. The results demonstrate that increasing the number of replays leads to improved model performance. For memory efficiency, We store 2 samples per old keyframe for replay.

Exploring the length of each action prompt. During the training in the new incremental step, there are new skill-specific action prompts assigned to learn action primitives. Specifically, as shown in Figure 1 (b), we conduct experiments with action prompt lengths varying from 4 to 32. The results show that extending the length of skill-specific action prompts enhances model performance. For better memory efficiency, the action prompt length is set to 16.

Exploring the distillation loss factor λ_{dis}. In our iManip, we propose to use a distillation loss with a factor λ_{dis} to transfer the knowledge from the old model to the agent. We conduct ablation experiments with different values of λ_{dis} to investigate the impact of the distillation loss on model performance, as shown in Figure 1 (c). Results indicate that distillation from the old model contributes to improving agent performance. However, a larger weight of λ_{dis} causes the model to focus too much on old skills, negatively affecting overall performance. We set this parameter to 0.001 to achieve optimal performance.

Exploring the impact of expended PerceiverIO. We propose extending the weights of the PerceiverIO to adapt to learn new action primitives. As shown in Figure 1 (d), we perform ablation experiments on the dimension of the appended PerceiverIO weight matrix d_{new}. The results indicate that the best performance is achieved when d_{new} is set to 8, and larger values hinder overall performance. This is because more new parameters for new skills training can interfere with the retention of old knowledge, leading to forgetting.

2. More exploratory experiments

The effect of skill learning order. Long-horizon skills are more challenging for robotic manipulation [1]. Based on the number of keyframes, we organized robotic skills into three progressively more challenging levels, *i.e.* short, medium, and long. Skills with fewer than five keyframes are considered short-horizon, those with 5 to 10 keyframes are classified as medium, and skills with more than 10 keyframes are regarded as long-horizon. We conducted experiments on the learning sequence of skills with varying levels in the B2-2N2 setting, as shown in Table 1. It is evident that, regardless of the skill learning sequence, our method maintains a balanced average accuracy after completing all skills in the final stage. This highlights the robustness of our approach, which can effectively adapt to learn different new skills. Furthermore, the results reveal that, compared to short-horizon skills, longer-horizon skills are more effective in acquiring general knowledge, thereby mitigating forgetting.

Order	Base	Step 1		Step 2			
		B	S1	B	S1	S2	Average
S-M-L	72	42	44	42	36	10	29.3
S-L-M	72	44	10	40	8	44	30.7
M-S-L	48	42	66	34	42	10	28.7
M-L-S	48	38	8	32	4	58	31.3
L-S-M	10	6	60	4	46	38	29.3
L-M-S	10	8	48	4	34	52	30.0

Table 1. Average success rate of skills with varying levels (Short, Medium, and Long) on different continuous learning orders. The new skills learned in the base step, 1st step, and 2nd step are termed B,S1, and S2 respectively.

Method		B5-1N5	B3-2N3
PerAct [5]	None	15.6	8.4
	+TIB	20.4	22.6
	+Ours	25.6	30.7
GNFactor [6]	None	20.4	15.6
	+TIB	27.4	29.3
	+Ours	33.6	36.9
3DDA [4]	None	42.4	31.6
	+TIB	67.6	68.4
	+Ours	72.8	76.4

Table 2. The results of adapting our iManip to different robotic manipulation pipelines.

Plug and play. Our skill-incremental policy can also be seamlessly integrated into other robotic multi-task learning pipelines. We apply our method to three robotic multi-task learning frameworks including Peract[5], GNFactor[6], and 3DDA[4]. we compare our method with pipelines that ei-

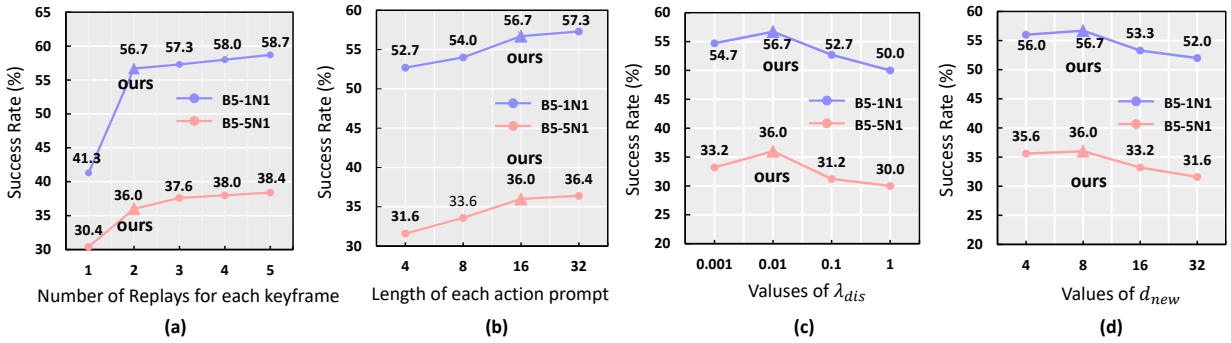

Figure 1. Experiments results of four key hyperparameters.

Manipulation skill	Type	Variations	Keyframes	Instruction Template
close jar	color	20	6.0	"close the _jar"
open drawer	placement	3	3.0	"open the _drawer"
sweep to dustpan	size	2	4.6	"sweep dirt to the _dustpan"
meat off grill	category	2	5.0	"take the _off the grill"
turn tap	placement	2	2.0	"turn _tap"
slide block	color	4	4.7	"slide the block to _target"
put in drawer	placement	3	12.0	"put the item in the _drawer"
drag stick	color	20	6.0	"use the stick to drag the cube onto the _target"
push buttons	color	50	3.8	"push the _button, [then the _button]"
stack blocks	color, count	60	14.6	"stack _blocks"

Table 3. Task Information Table

ther did not include incremental methods or used traditional incremental learning methods. Specifically, for storing old replay data, all the aforementioned pipelines can utilize our temporal replay strategy to sample informative, temporally balanced samples from previous manipulation skills. Furthermore, we can modify the transformer with our extensible self-attention layer by appending action prompts and incorporating a minimal number of trainable parameters to adapt to different action primitives. Notably, following the original setup of each paper, we use 20 demonstrations per manipulation skill for PerAct and GNFactor, and 100 demonstrations for 3DDA. As shown in Table 2, in each pipeline, our method achieves the highest task success rates. This demonstrates the excellent scalability of our incremental strategy.

3. More details about simulation experiments

Manipulation skills in RLbench. We select 10 language-conditioned skills from RLBench [3], each involving at least two variations. An overview of these skills can be found in Table 3. The variations include random sampling of object colors, sizes, quantities, placements, and categories, resulting in a total of 166 distinct combinations. The color set consists of 20 different colors: red, maroon, lime, green, blue, navy, yellow, cyan, magenta, silver, gray, orange, olive, purple, teal, azure, violet, rose, black, and white. The size set includes two options: short and tall. The count set has three possible values: 1, 2, or 3. The placements and object categories are skill-specific. For instance, the "open drawer" skill has three placement options: top, middle, and bottom. Additionally, objects are randomly placed on the tabletop in various poses within a defined range.

The experimental details of iManip in the B5-5N1 setup. In this setup, the agent is first trained on the 5 base skills, and then a new skill is added at each step, with a total of 5 steps. The base skill includes *close jar*, *open drawer*, *sweep to dustpan*, *meat off grill* and *turn tap*. The training sequence for the new skills is *slide block*, *put in drawer*, *drag stick*, *push buttons*, *stack blocks*. Table 4 shows the success rate of each skill at each step. The Old is the average success rate of the old skills. For example, the Old in step 1 is the average success rate of the five base skills. The All is the average success rate of all learned skills at that step.

4. More details about real world experiments

In the real world experiment, we use the B1-4N1 setup, which allows the agent to gradually learn five different skills one by one. The five manipulation skills includes *Silde toy to target*, *Open drawer*, *Pick and place*, *Pour water* and

Robotic skills	Base	Step 1	Step 2	Step 3	Step 4	Step 5
close jar	28	24	40	32	16	20
open drawer	56	72	64	68	60	64
sweep to dustpan	52	52	56	32	40	36
meat off grill	80	76	52	60	52	52
turn tap	64	64	60	60	56	56
slide block	-	52	32	32	44	40
put in drawer	-	-	32	32	4	8
drag stick	-	-	-	64	64	60
push buttons	-	-	-	-	16	12
stack blocks	-	-	-	-	-	12
Old	56.0	57.6	50.7	45.1	42.0	38.7
All	56.0	56.7	48.0	47.5	39.1	36.0

Table 4. Performance of iManip in the setup of B5-5N1.

Close jar. Concretely, The skill of *silde toy to target* requires the agent to move the toy to the color area specified by the instruction. The skill of *Open drawer* requires the agent to open the drawer at the corresponding position, including variations for the top, middle, and bottom positions. The skill of *Pick and place* requires the agent to grasp the specified object and place it at the designated location. The skill of *Pour water* requires the agent to pick up the water-filled cup of a specified color and pour the water into the mug of another specified color. Lastly, the skill of *Close jar* requires the agent to grasp the bottle cap and screw it onto the bottle. We present the keyframes of the five manipulation skills in sequence in Figure 2, visually illustrating the specific steps of the execution process.

5. Model architecture

Voxel Encoder: We employ a compact 3D UNet with only 0.3M parameters to encode the input voxel of size $100^3 \times 10$ (which includes RGB features, coordinates, indices, and occupancy) into our deep 3D volumetric representation, resulting in a size of $100^3 \times 128$.

Original PerceiverIO. The Extendable PerceiverIO in iManip is an improvement upon the original PerceiverIO [2]. A detailed explanation of the original PerceiverIO is provided here to offer a more comprehensive understanding of our Extendable PerceiverIO.

The original PerceiverIO consists of 6 attention blocks designed to process sequences from multiple modalities (such as 3D volumes, language tokens, and robot proprioception) and output a corresponding sequence. To efficiently handle long sequences, the Perceiver Transformer uses a small set of latents to attend to the input, improving computational efficiency. The resulting output sequence is then reshaped back into a voxel representation to predict the robot's actions. The Q-function for translation is predicted using a 3D convolutional layer. For predicting openness, collision avoidance, and rotation, we apply global max pooling and spatial softmax to aggregate 3D volume features, and then project the aggregated feature to the output dimension using a multi-layer perceptron.

Model inference. The agent obtains the current observation state, including RGB-D, proprioception, and textual instructions, and predicts the end-effector pose for the next keyframe. It then uses a predefined motion planner (e.g., RRT-Connect) to solve for the motion path and joint control angles. This approach reduces the sequential decision-making problem to predicting the optimal keyframe action for the next step based on the current observation.

References

[1] Ricardo Garcia, Shizhe Chen, and Cordelia Schmid. Towards generalizable vision-language robotic manipulation: A benchmark and llm-guided 3d policy. *arXiv preprint arXiv:2410.01345*, 2024. 1

[2] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, 2021. 3

[3] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 2020. 2

[4] Tsung-Wei Ke, Nikolaos Gkanatsios, and Katerina Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. *arXiv preprint arXiv:2402.10885*, 2024. 1

[5] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, 2023. 1

[6] 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*, 2023. 1

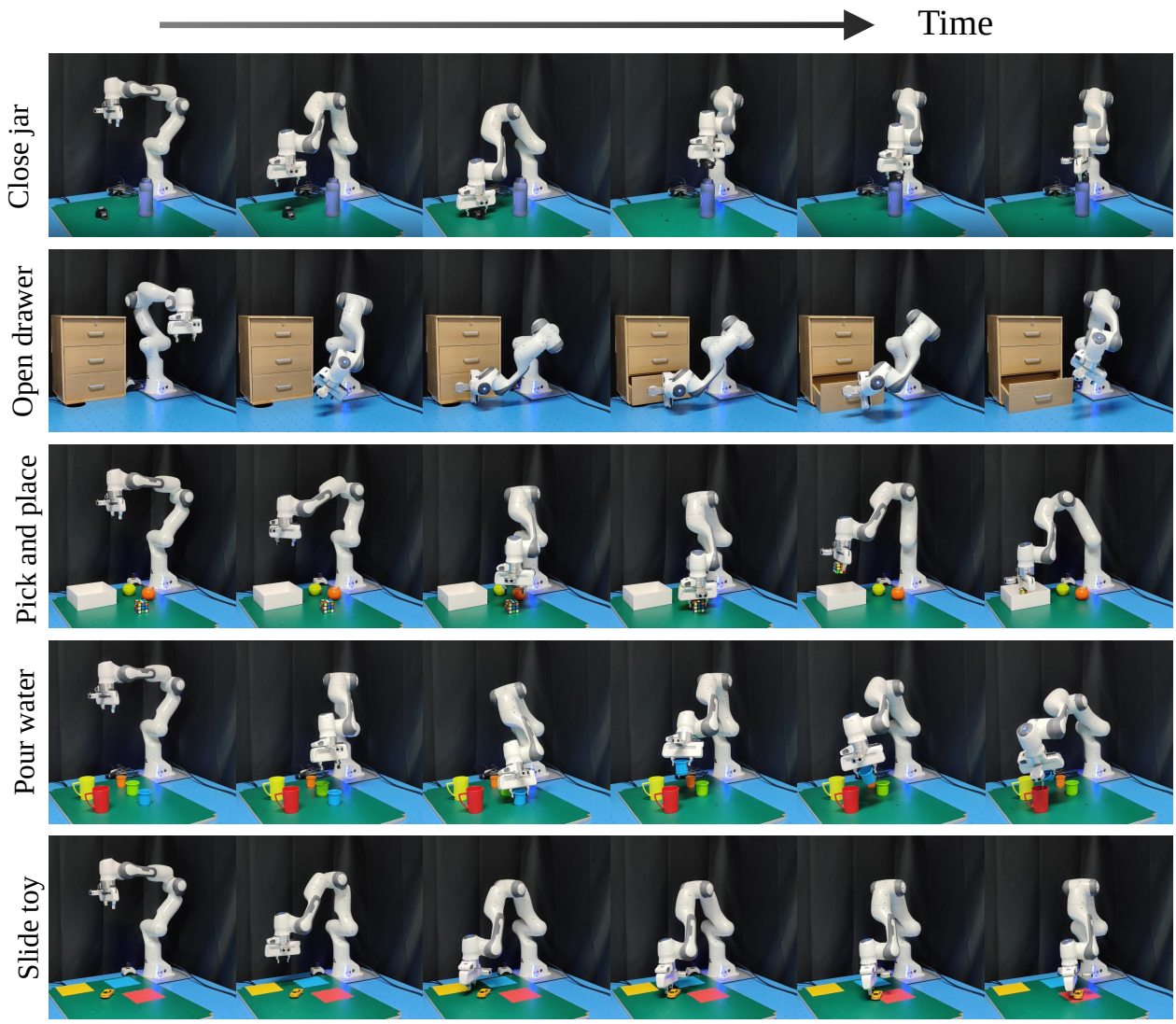

Figure 2. Keyframes for real robot manipulation skills.