RoBridge: A Hierarchical Architecture Bridging Cognition and Execution for General Robotic Manipulation

Supplementary Material

SUMMARY OF THE APPENDIX

This appendix contains additional details for this paper. The appendix is organized as follows:

- §A provides **Limitations** of our work.
- §B provides **Experiment Details**.
- §C shows more **Visualization**.

A. Limitations

RoBridge still presents opportunities for further improvement. As a hierarchical framework, it is susceptible to the performance of any individual module. Enhancements in the visual understanding capabilities of the high-level planning module and improvements in the precision of execution in the low-level control module could significantly boost RoBridge's overall performance. Furthermore, while currently limited to manipulating objects with simple shapes, RoBridge could be extended to handle a wider variety of shapes in the future, including soft or tiny objects.

B. Experiment Details

B.1. Primitive Actions

The detailed definitions of primitive actions are shown in Table 1.

Table 1. The list of primitive actions and their description.

Action	Description
grasp	Securely hold an object to control its position.
place	Put an object at a specific location.
press	Apply force to an object to activate or transform it.
push	Exert force on an object to move it away from a specific direction.
pull	Apply force to draw an object closer from a specific direction.
open	Adjust an object to allow access or exposure.
close	Adjust an object to restrict access or seal it.
turn	Rotate an object to change its orientation.
reach	Approach an object or a designated location.

B.2. Training Details

In our experiments on MetaWorld, we conducted training for 1M steps, with failure data sampling occurring every 100k steps. We added the RL training curve. As shown in Figure 1, most tasks can be trained in 200k timesteps. The success rate of π_e in the training scenario is 90.8%. During real-world experiments, we finetuned the model for 2k steps. We will resize the image to 168×168 and feed it to GEA for faster speed. Training π_e and π_g on an A100 GPU took 25 and 30 GPU hours, respectively. Finetuning on real data took just 1 GPU hour. Inference needs 6 GB GPU memory (except GPT-4o). Only GEA and Track-Anything run every frame, taking 60-80ms. VLM and SAM+GroundingDINO only run in the first frame of each primitive action, with VLM at 0.3s and SAM+GroundingDINO at 1s per run.

B.3. Robosuite Benchmark Results

Robosuite encompasses a suite of contact-rich robotic manipulation tasks, emphasizing high-fidelity rendering and realistic physical control. This environment allows us to evaluate the potential effectiveness of our approach in real-world scenarios. Table 2 shows the results of Robosuite, RoBridge achieved the highest success rates in all cases, indicating that our method is also effective for tasks involving contact-rich interactions.

B.4. MetaWorld Benchmark Results

We show in detail the specific success rate of our method on MT50 (the 50 tasks of the metaworld) in the Table 3.

C. Visualization

C.1. DINOv2 Visualization

We conducted an in-depth analysis of why DINOv2 performs poorly as a feature extractor. By visualizing DINOv2's attention areas, as shown in Figure 2, we found that it predominantly focuses on common objects, such as drawers, and does not pay much attention to robotic arms or small objects. It only partially attends to slightly more prominent objects, like buttons, or when a robotic arm is near a drawer. Therefore, it is not well-suited for robotic manipulation tasks.

C.2. Results

We show more demonstrations of real-world experiments in the Figure 3.

C.3. Failure Case Analysis

This section investigates the causes of failure in the operation of RoBridge. Our findings indicate that the majority of failures are attributable to the loss of masks due to occlusion or overlap, as well as positional deviations during execution. Fortunately, we observed that RoBridge is capable of correcting errors to a certain extent. As illustrated in Figure 4, when the task failed during the attempt to grasp the second block, RoBridge was able to replan and successfully complete the task on the second attempt.

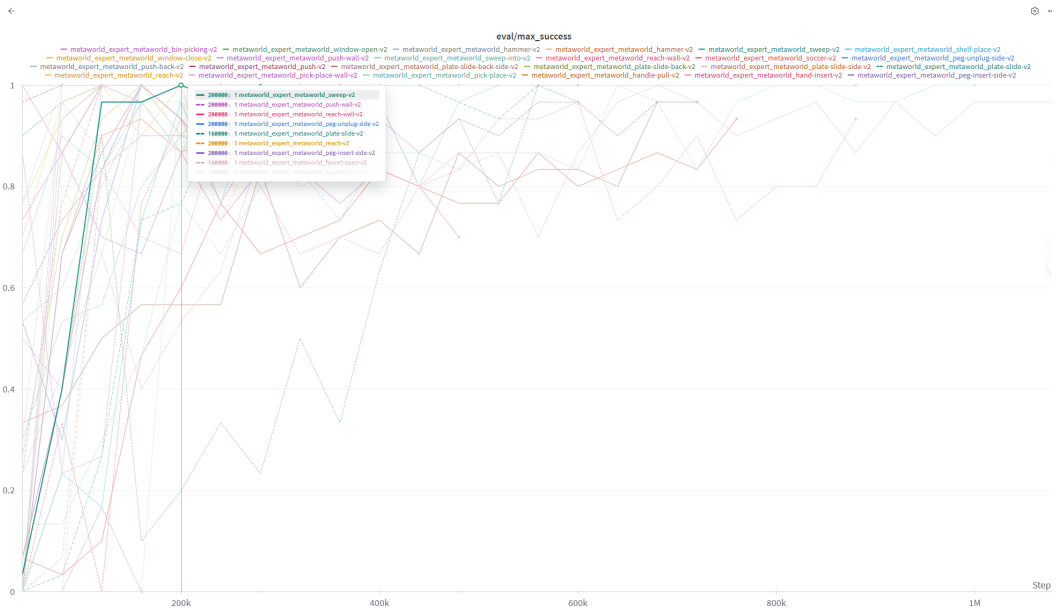

Figure 1. RL training details.

Table 2. Robosuite Benchmark Results(%).

Model	RS-Bread	RS-Can	RS-Milk	RS-Cereal	RS-NutRound	RS-NutSquare	Mean
DRQ-v2	52	32	2	0	6	2	15.7
RAPS	0	0	0	0	0	0	0
TAMP	90	100	85	100	40	35	75
SayCan	93	100	90	63	56	27	71.5
PSL	100	100	100	100	98	97	99.2
RoBridge	100	100	100	100	100	100	100

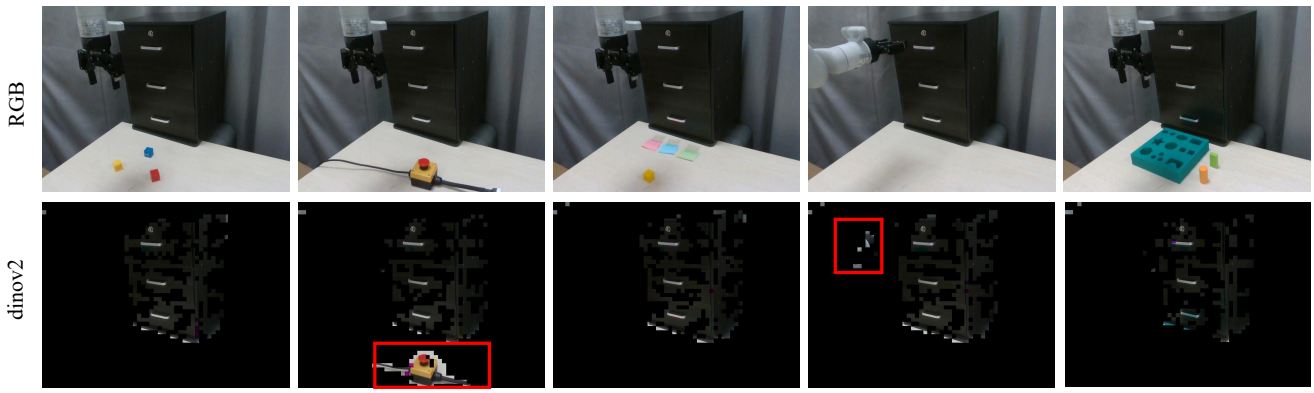

Figure 2. Feature visualization of DINOv2.

Table 3. Detail Results of Metaworld Tasks. Each task is tested 10 times.

Task	Success	Task	Success	Task	Success
assembly	10	button-press-topdown	10	door-unlock	10
basketball	0	button-press-topdown-wall	8	hand-insert	10
bin-picking	9	button-press	10	drawer-close	10
box-close	5	button-press-wall	10	drawer-open	10
coffee-button	10	coffee-pull	10	faucet-open	10
coffee-push	10	dial-turn	9	faucet-close	10
disassemble	6	door-close	10	hammer	2
door-lock	8	door-open	10	handle-press-side	10
handle-press	10	handle-pull-side	5	handle-pull	10
lever-pull	2	peg-insert-side	9	pick-place-wall	10
pick-out-of-hole	5	reach	10	push-back	10
push	7	pick-place	10	plate-slide	9
plate-slide-side	9	plate-slide-back	8	plate-slide-back-side	9
peg-unplug-side	9	soccer	7	stick-push	8
stick-pull	8	push-wall	10	reach-wall	10
shelf-place	9	sweep-into	10	sweep	7
window-open	10	window-close	10		
Mean Success Rate	**85.4**				

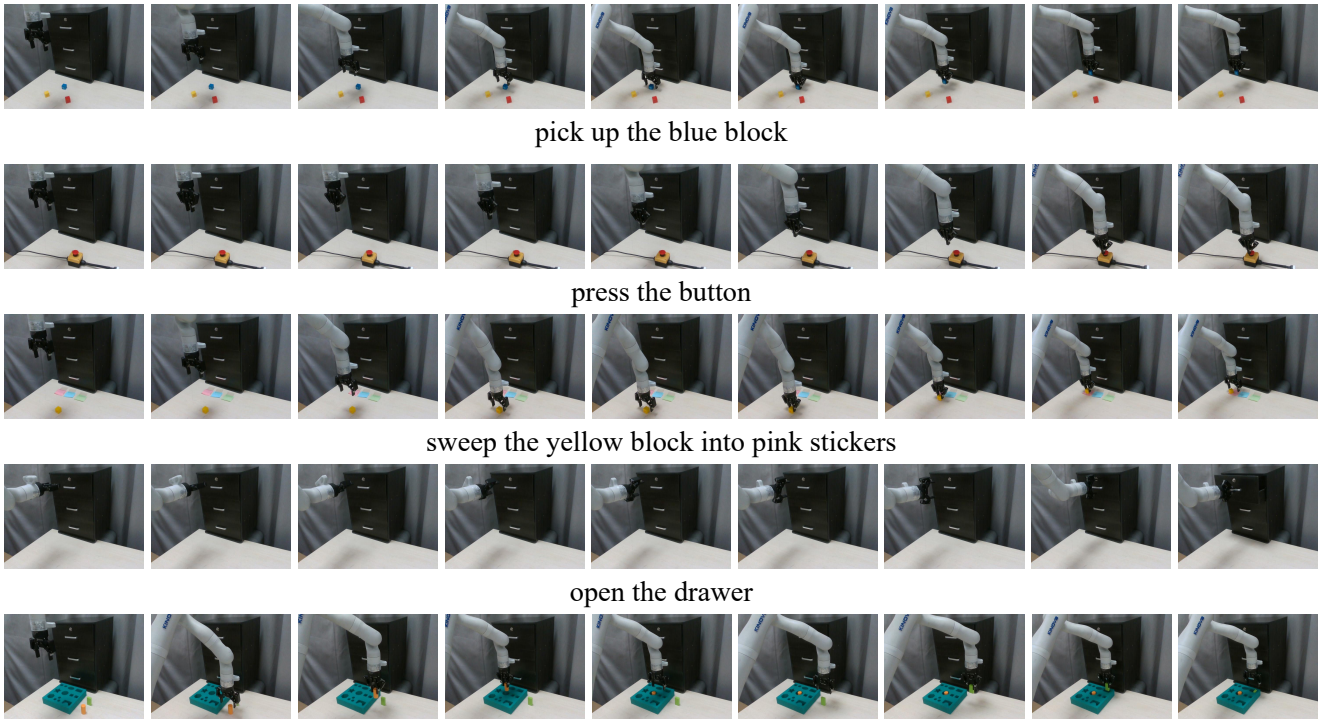

Figure 3. Demonstrations of real-world experiments.

Figure 4. A demonstration shows how RoBridge recovers from failure. When it fails, it re-plans and executes.