Diagnose, Correct, and Learn from Manipulation Failures via Visual Symbols

Supplementary Material

1. Table of Content

This supplementary material contains:

2. Design of Visual Symbols

In this section, we present concrete examples to illustrate the application of individual visual symbols, as well as their usage in combination. As shown in Figure 1, the visual symbols in each instance have the following meanings:

(a) Move the left gripper to the right significantly, forward significantly, and downward significantly, aligning the held object with the green cube for placement.
(b) Rotate the right gripper clockwise to achieve the correct grasp pose for the marker.
(c) The correct target (as indicated by the crosshair) is the top drawer handle rather than the bottom one.
(d) The dual crosshairs linked by a dashed line indicates an alignment requirement between the two targeted objects—in this instance, placing the held object into the transparent box.
(e) Open the right gripper, thereby releasing the spatula into the drawer.
(f) Keep the right gripper closed, preventing the coke from slipping during movement.
(g) Hold the left arm still until the human intervention is removed.
(h) Move the right arm back to its initial pose.
(i) Move the left gripper to the right slightly and rotate it clockwise, achieving the correct grasp pose for the lid.
(j) Move the left gripper to the right slightly and backward slightly to align the vial with the specific slot.
(k) Hold the left arm still and align the right gripper with the edge of the blue plate for grasping.
(l) The left arm targets the wrong object, resets to the initial state, and then grasps the correct target—the doll.

3. Task Design and Data Annotation

3.1. Task Design Details

As mentioned in Section 3.3.1, our designed 100 real-world manipulation tasks cover diverse manipulation skills. All 100 tasks and their corresponding trajectory counts are listed in Figure 2.

3.2. Data Annotation Tools

The front-end annotation tool is illustrated in Figure 3. Annotators can complete the preliminary annotation via click-and-drag mouse operations. Annotations are performed based on the head camera view. Additionally, the wrist camera view (if available) can be loaded to assist annotators in determining spatial positions. Finally, we sample the video at 1 fps and add the selected keyframes to the sampling list.

4. Details of Dataset and Benchmark

4.1. VQA Design and Examples

4.1.1. Design of Closed-ended VQA

The closed-ended VQA question types include failure detection, failure keyframe localization, failure subtask localization, failure type identification, and low-level textual guidance (avoidance and correction). For all tasks excluding low-level textual guidance, we randomly sample the corresponding annotation pools to construct distractor candidates. The question templates are shown in Figure 4.

For low-level textual guidance, we construct three challenging distractors using a hybrid sampling strategy. We combine a static pool of common hard-coded actions (*e.g.*, "Hold still", "Open gripper") with a larger, dynamic pool of all possible commands, ensuring that all sampled distractors are unique and not the ground truth. To increase difficulty, distractors are semantically constrained to match the active gripper (*e.g.*, "left" or "right") of the correct answer, making all options contextually plausible. The question templates are shown in Figure 5.

4.1.2. Design of Open-ended VQA

The open-ended VQA question types include failure reason, CoT form of low-level textual guidance (avoidance and correction), and high-level textual guidance (avoidance and correction). The evaluation of open-ended VQA is introduced in Section 5.2.2. The question templates are shown in Figure 6 and Figure 7.

Additionally, there is a special type of VQA called visual guidance, introduced in Section 3.2 and 4.2, which is used to train and evaluate our VLM model's ability to generate code elements and draw our designed visual symbols. The question and answer templates are shown in Figure 8 and Figure 9.

4.2. Dataset Statistics

All the VQA pairs in our dataset are generated from 5,202 real-world manipulation trajectories of an ALOHA robot.

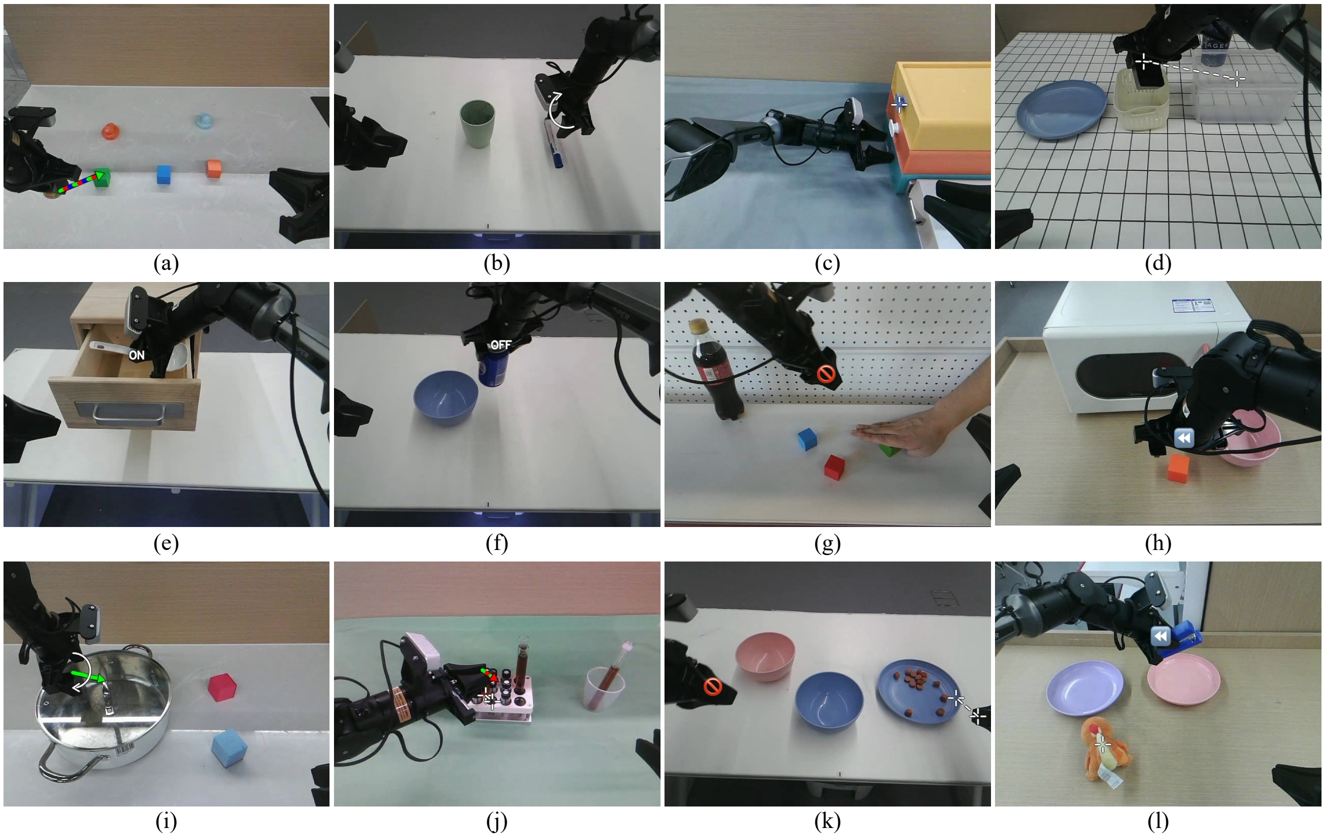


Figure 1. **Examples of individual visual symbols and their combination. Top and Middle Rows:** Instances of individual visual symbols. **Bottom Row:** Instances of their usage in combination.

The distribution of trajectory durations is listed in Table 1, which indicates that our trajectory data encompasses both short-horizon and long-horizon episodes, with a predominance of short-horizon episodes (more conducive to failure analysis).

Table 1. Distribution of trajectory durations in the dataset

Duration (Second)	Count
0 – 4	156
4 – 8	2013
8 – 12	1682
12 – 16	708
16 – 20	412
20 – 24	164
≥ 24	67

4.3. Benchmark Statistics

The tasks within the benchmark and their respective trajectory counts are illustrated in Figure 2. Notably, five tasks (Task ID: 43–47) are entirely OOD.

5. Details of Finetuning and Evaluation

5.1. Training Details

We use LoRA [14] to finetune the Qwen3-VL-8B [2] model for 1 epoch, with a LoRA rank of 32 and a scaling factor α of 64, yielding the model ViFailback-8B. We unfreeze both the LLM backbone and the adapter parameters, and train the model using deepspeed zero2 stage [32] to ensure stable training. Each GPU processes a batch size of 1, with a gradient accumulation step of 4 and a learning rate of 1e-5. The training is performed on 4 NVIDIA Hopper GPUs.

5.2. Evaluation Details

5.2.1. Model Configuration

For consistency, all models are set the temperature to 0 and the maximum generation length to 2,048 tokens.

5.2.2. Open-ended VQA Evaluation

For open-ended questions, we employ the GPT-4o-based evaluator to assess the quality of the generated outputs comprehensively. Specifically, we compare the model outputs with the ground truth across three dimensions: **Semantic**

Similarity, **Content Completeness**, and **Functional Equivalence**.

- **Semantic Similarity:** The degree to which the two texts convey equivalent meanings and intentions, regardless of surface-level wording differences.
- **Content Completeness:** Whether all critical information elements are present in both texts, including gripper specifications, movement directions, and command details.
- **Functional Equivalence:** Whether the described actions achieve the same robotic manipulation goals and operational outcomes.

The exact prompts used are shown in Figure 10.

6. Details of Real-world Experiments

6.1. Experiment Setup

For each task, we collected 20 expert demonstrations to finetune $\pi_{0.5}$ [4] model as the actor. During inference, our ViFailback-8B serves as a supervisor, queried at intervals of 6 action chunks. By analyzing the visual observations of the past 5 seconds, it diagnoses failures and provides multimodal (textual and visual) guidance for failure recovery when necessary.

6.2. Mask Details of the VSF Method

To enable $\pi_{0.5}$ to perform corrective actions based on visual symbols, we collected a visual symbols following dataset, which consists of trajectories where the robot performs atomic actions following specific visual symbols. To force the model to focus on the symbols, we mask out irrelevant regions in the observations (as shown in Figure 11). Specifically, in the head camera view, we define a Region of Interest (ROI) based on the visual symbol's bounding box. This ROI is expanded by a 50-pixel margin, subject to a minimum dimension constraint of 50 pixels. Any visual content outside this ROI is masked with zeros. Additionally, the wrist camera observations for the idle arm are fully masked with zeros.

6.3. Visualization of Experimental Results

6.3.1. Low-level Textual and Visual Guidance

By analyzing historical observations from the head camera, ViFailback-8B conducts failure diagnosis and correction. As illustrated in Figures 12–17, we show the model's generated responses and the correction outcomes achieved by two types of visual symbol following approaches. Beyond diagnosing common Gripper 6d-pose errors, our model exhibits the capability to identify other types of failure, such as human intervention and gripper state depicted in Figure 18 and 19.

6.3.2. Failure Reason and High-level Guidance

Figure 20–22 illustrate ViFailback-8B identifying the root cause of failures and providing high-level guidance across three distinct tasks. These examples demonstrate its ability to understand failure contexts for effective correction.

Task ID	Task Name	Task Description	Total Trajectories	Benchmark Trajectories	Training Trajectories
1	cut_sausage	Use the right arm to pick up the knife from the knife holder, cut a small piece of sausage and put it back on the knife holder.	33	0	33
2	grasp_bottle_place_bowl	Use the left arm to grasp the bottle and place it into the pink bowl.	15	0	15
3	grasp_cube_place_shelf	Use the right arm to grasp the test tube and place it upright on the shelf.	15	0	15
4	grasp_filter_place_juicer	Use the left arm to grasp the filter and place it into the juicer.	20	0	20
5	grasp_marker_place_in_cup	Use the right arm to grasp the marker and place it in the purple cup.	20	0	20
6	grasp_spatula_put_pot	Use the right arm to grasp the wooden spatula and put it into the pot.	15	0	15
7	lift_lid_and_put_green_duck	Use the right arm to lift the lid off the pot, then use the left arm to place the green duck into the pot.	18	0	18
8	lift_lid_place_pot	Use the left arm to lift the lid of the pot and place it over the pot.	15	0	15
9	open_box_place_cable	Use the right arm to open the paper box. Then use the left arm to grasp and put the data cable into the paper box.	15	0	15
10	open_left_right_door	Use the left arm to grasp the left door handle and open the left door. Then use the right arm to grasp the right door handle and open the right door.	20	0	20
11	open_lid_put_spoon	Use the right arm to open rice cooker lid. Then use the left arm to pick up the white spoon and put it into the rice cooker.	15	0	15
	open_lid_put_spoon_ep2		150	0	150
12	open_microwave_door_put_bowl	Use the left arm to open the microwave oven door. Then use the right arm to put the blue bowl inside the microwave oven.	15	0	15
13	pick_block_drawer_to_plate	Use the right arm to pick up the purple block from inside the drawer and place it on the plate. Then, use the left arm to push the drawer closed.	36	0	36
14	pick_bowl_cube	Use the left arm to pick up the pink bowl and place it in the middle of the table. Then use the right arm to pick up the yellow cube and place it into the pink bowl.	15	0	15
	pick_bowl_cube_ep2		150	0	150
15	pick_bowl_on_top_shelf	Use the left arm to pick up the pink bowl and place it on the top shelf.	15	0	15
16	pick_cube_place_plate	Use the left arm to pick up the blue cube and place it on the green plate.	10	0	10
17	pick_cup_on_left_table	Use the right arm to pick up the blue cup and place it on the left table.	15	0	15
18	pick_marker_place_bowl	Use the right arm to pick up the marker and place it into the white bowl.	15	0	15
19	pick_orange_block_place_into_microwave	Use the right arm to place the orange block into the bowl. Then, use the left arm to open the microwave door, use the right arm to put the bowl into the microwave, and finally, close the microwave door.	250	30	220
20	pick_place_chem_ep1	Use the left arm to pick up the vial and place it in the left empty slot on the rack. Then use the right arm to grasp the test tube and place it in the right empty slot on the rack.	250	30	220
	pick_place_chem_ep2		250	30	220
21	place_block_into_drawer	Use the right arm to place the yellow block inside the drawer, and then push the drawer closed.	32	10	22
22	place_duck_on_block	Use the left arm to place the green duck on top of the green block.	16	0	16
23	pour_sausage_into_pot	Use the left arm to pour the sausages from the plate into the pot, then use the right arm to place the lid on the pot.	34	10	24
24	pull_open_drawer	Use the left arm to pull open the top drawer.	32	10	22
25	stack_cup_cola	Use the left arm to pick up the purple cup and place it in the blue bowl. Then use the right arm to pick up the red coke can and place it into the purple cup.	250	30	220
26	stack_green_cup_into_purple_cup	Use the left arm to stack the green cup into the purple cup.	15	0	15
27	stack_three_block	Use the left arm to stack the orange block on top of the green block, then use the right arm to stack the blue block on top of the orange block.	49	0	49
28	stack_three_color_cubes_ep1	Use the left arm to stack the green cube on top of the blue cube. Subsequently, use the right arm to place the red cube on top of the green one to complete the three-layer structure.	159	20	139
	stack_three_color_cubes_ep2 (VLA Rollout Data)		41	10	31
29	pick_object_place_left_box_ep1	Use the right arm to grasp the object from the table and place it in the black box on the left.	160	20	140
	pick_object_place_left_box_ep2 (VLA Rollout Data)		40	10	30
30	pull_bottom_drawer_put_can_ep1	Use the left arm to pull the bottom drawer open, then use the right arm to place the can on the table into the drawer, and finally close the drawer with the left arm.	162	20	142
	pull_bottom_drawer_put_can_ep2 (VLA Rollout Data)		40	10	30
31	pick_marker_draw_cross_ep1	Use the right arm to grasp the marker in the green cup and draw a black cross mark on the table and then put it into the green cup.	160	20	140
	pick_marker_draw_cross_ep2 (VLA Rollout Data)		18	10	8
32	pick_marker_transfer_place_plate_ep1	Use the left arm to grasp the marker, transfer it to the right arm and then use the right arm to put it in the green plate.	159	0	159
	pick_marker_transfer_place_plate_ep2 (VLA Rollout Data)		42	0	42
33	pour_water_transparent_cup_ep1	Use the right arm to grasp the handle of the pink cup and pour the colored water into the transparent cup.	159	20	139
	pour_water_transparent_cup_ep2 (VLA Rollout Data)		26	10	16
34	lift_lid_and_place_cube_ep1	Use the left arm to lift the lid off the pot. Then use the right arm to grasp and place the red cube into the pot and the left arm put the lid back on the pot.	157	20	137
35	drag_basket_place_cube_ep1	Use the right gripper to drag the basket to the center of the table, then use the left gripper to grasp and place the blue cube into the basket.	157	0	157
36	put_doll_pink_plate_stapler_purple_plate_ep1	Use the left arm to put the yellow doll into the pink plate, and use the right arm to put the blue stapler into the purple plate.	150	20	130
37	put_mouse_basket_from_tray_ep1	Use the left arm to place the mouse from the green tray into the basket.	150	20	130
38	pick_plate_from_drawer_put_pink_cube_ep1	Use the right arm to place the pink block on the drawer onto the table, and use the left arm to place the blue block onto the pink block.	140	20	120
39	grab_socket_unplug_charger_ep1	Grab the socket with the right arm and unplug the charger with the left arm.	150	20	130
40	pick_cube_transfer_place_basket_ep1	Use the right arm to grasp the blue cube, transfer it to the left arm and then use the left arm to put it in the basket.	130	0	130
41	pick_coke_bottle_place_plate_ep1	Use the left arm to grasp the coke bottle and place it upright on the blue plate.	138	0	138
42	pick_pepsi_transfer_place_plate_ep1	Use the left arm to grasp the Pepsi can, transfer it to the right arm and then use the right arm to place it upright on the green plate.	138	20	118
43	pull_top_drawer_put_controller	Use the right arm to pull the top drawer open, then use the left arm to place the remote controller into the drawer, and finally close the drawer with the right arm.	16	16	0
44	pick_pen_transfer_plate	Use the left arm to grasp the pen, transfer it to the right arm and then place it in the blue plate.	16	16	0
45	place_spoon_drawer_push	Use the right arm to place the spoon inside the drawer, and then use the left arm to push the drawer closed.	16	16	0
46	pick_mouse_place_laptop	Use the left arm to grasp the mouse and then place it on the middle of the laptop	16	16	0
47	pick_tape_thread_marker	Use the right arm to grasp the tape and thread the tape's central hole through the upright marker on the table.	16	16	0
48	pick_red_marker_place_tray	Use the right arm to pick up the red marker and place it on the white tray.	16	0	16
49	pour_cubes_into_box	Use the right arm to pour the cubes from the cup into the yellow box.	16	0	16
50	place_knife_in_drawer	Use the right arm to place the green plastic knife into the open top drawer.	16	0	16

Figure 2. Full list of our designed tasks. (Continued on next page)

51	pick_green_cube_put_basket	Use the left arm to pick up the green cube and place it into the woven basket.	16	0	16
52	put_blackboard_eraser_in_transparent_box	Use the right arm to put the blackboard eraser into the transparent box.	16	0	16
53	stack_cup_on_book	Use the left arm to pick up the cup and place it on top of the closed book.	16	0	16
54	put_two_objects_into_plate_and_box	Use the right arm to put the doll on the blue plate and the left arm to put the stapler on the box.	16	0	16
55	place_pen_in_drawer	Use the left arm to place the pen into the open drawer.	16	0	16
56	grasp_duck_put_cup	Use the right arm to grasp the yellow duck and place it on the cup.	16	0	16
57	put_pen_in_basket	Use the right arm to put the black pen into the basket.	16	0	16
58	pick_cube_put_pot	Use the left arm to pick up the red cube and place it into the red pot.	16	0	16
59	grasp_coke_can_put_tray	Use the right arm to grasp the Coke can and place it on the green tray.	16	0	16
60	put_mouse_on_pot	Use the right arm to place the computer mouse on the red pot.	16	0	16
61	pick_ball_put_drawer	Use the right arm to pick up the table tennis ball and place it in the open drawer.	16	0	16
62	place_spoon_on_plate	Use the left arm to place the spoon on the purple plate.	16	0	16
63	open_drawer_put_marker	Use the left arm to open the drawer, then use the right arm to put the marker inside.	16	0	16
64	open_microwave_door_put_green_duck	Use the left arm to open the microwave oven door. Then use the right arm to put the green duck inside the microwave oven.	16	0	16
65	place_spoon_into_pot_cover_lid	Use the left arm to place the green spoon into the red pot, and then use the right arm to grasp the lid and cover the pot.	16	0	16
66	put_stapler_top_drawer_and_close	Use the left arm to put the blue stapler into the top drawer, and then use the right arm to pull close the top drawer.	16	0	16
67	place_green_duck_from_drawer_to_pot	Use the right arm to pick up the green duck from the drawer and place it on the red pot.	16	0	16
68	hold_socket_plug	Use the right arm to hold the socket, left arm to insert the plug into it.	16	0	16
69	pick_spoon_place_rice_cooker	Use the right arm to pick up the spoon and place it into the rice cooker.	16	0	16
70	grasp_knife_place_plate	Use the left arm to grasp the knife from the holder and place it on the plate.	16	0	16
71	grasp_test_tube_place_bowl	Use the right arm to grasp the test tube and place it in the blue bowl.	16	0	16
72	pour_sausage_into_bowl	Use the right arm to pour the sausages from the plate into the pink bowl.	16	0	16
73	pick_cylinder_place_drawer	Use the left arm to pick up the green cylinder from the plate and place it in the drawer.	16	0	16
74	grasp_spatula_place_basket	Use the left arm to grasp the wooden spatula and place it in the blue basket.	16	0	16
75	lift_lid_place_duck	Use the right arm to lift the lid off the pot, then use the left arm to place the green duck inside.	16	0	16
76	open_box_place_marker	Use the left arm to open the paper box, then use the right arm to place the marker inside.	16	0	16
77	pull_bottom_drawer_put_duck_into_pot	Use the right arm to pull the bottom drawer, then use the left arm to take out the orange duck in it and put the orange duck into the red pot.	16	0	16
78	pick_marker_place_cup	Use the left arm to pick up the marker and place it in the purple cup.	16	0	16
79	pick_orange_block_place_pot	Use the left arm to place the orange block into the pot.	16	0	16
80	pick_vial_place_right_slot	Use the right arm to pick up the vial and place it in the right empty slot on the rack.	16	0	16
81	pick_test_tube_place_left_slot	Use the left arm to grasp the test tube and place it in the left empty slot on the rack.	16	0	16
82	place_cube_pot_lid	Use the right arm to place the blue cube into the pot and then place the lid on top.	16	0	16
83	place_doll_stapler_swap	Use the left arm to put the white and blue doll into the green plate, and use the right arm to put the stapler into the blue plate.	16	0	16
84	pick_mouse_basket	Use the right arm to pick up the mouse from the tray and place it in the basket.	16	0	16
85	pick_cube_drawer_plate	Use the left arm to pick up the pink cube from the drawer and place it on the plate.	16	0	16
86	stack_pepsi_can_cup	Use the left arm to pick up the blue pepsi can and place it into the blue cup.	16	0	16
87	pull_open_bottom_drawer	Use the right arm to pull open the bottom drawer.	15	0	15
88	pick_marker_draw_line	Use the left arm to grasp the marker from the cup, draw a line on the table and put it back into the cup.	16	0	16
89	pick_eraser_place_right_box	Use the left arm to grasp the eraser from the table and place it in the black box on the right.	16	0	16
90	pick_marker_transfer_place_cup	Use the right arm to grasp the marker, transfer it to the left arm and then place it in the cup.	16	0	16
91	pick_pepsi_place_bowl	Use the right arm to grasp the pepsi bottle and place it upright in the blue bowl.	16	0	16
92	grasp_knife_place_drawer	Use the right arm to grasp the knife from the holder and place it in the drawer.	16	0	16
93	open_lid_place_filter	Use the left arm to open the rice cooker lid, then use the right arm to place the filter inside.	16	0	16
94	pour_sausage_into_basket	Use the left arm to pour the sausages from the plate into the white basket.	16	0	16
95	push_middle_drawer	Use the left arm to push to close the middle drawer.	16	0	16
96	pick_vial_place_middle_slot	Use the left arm to pick up the vial and place it in the middle empty slot on the rack.	16	0	16
97	pick_test_tube_place_middle_slot	Use the right arm to grasp the test tube and place it in the middle empty slot on the rack.	16	0	16
98	pick_remote_controller_transfer_board	Use the right arm to grasp the remote controller, transfer it to the left arm, and then place it on the chopping board.	15	0	15
99	put_stapler_blue_plate	Use the left arm to put the blue stapler into the blue plate.	16	0	16
100	open_drawer_put_stapler	Use the right arm to open the drawer, then use the left arm to put the stapler inside.	16	0	16

Figure 2. Full list of our designed tasks (continued).

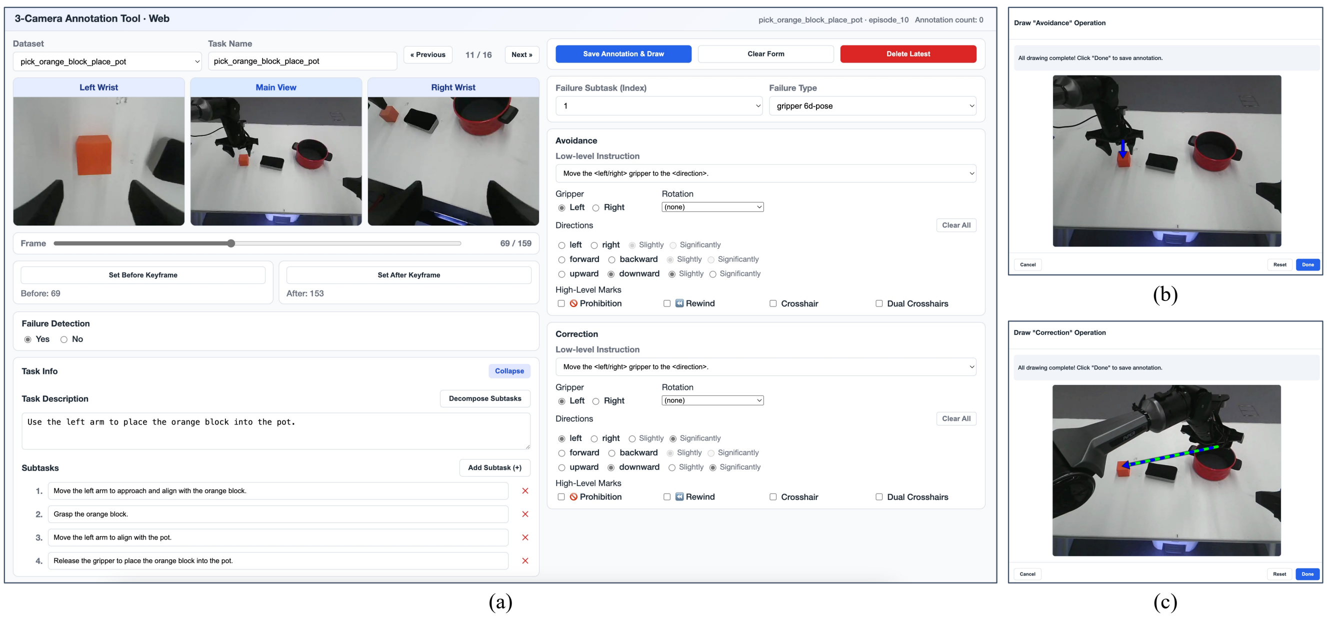


Figure 3. The front-end UI for annotation, including (a) the main annotation interface, (b) the visual symbol drawing interface for avoiding failure, and (c) the visual symbol drawing interface for recovering from failure.

VQA question template for failure detection:

{task_image_sequence}These images are captured by the head camera of the dual-arm robot in chronological order.
Given the robot's actions in the images, with the specified task being to '**{task_description}**', can it be determined whether the dual-arm robot successfully completed the task?
You should answer based on the following definitions:
Successful: The task is fully and correctly completed.
Failed: Any failure occurred during execution, leaving the task incomplete.

Your answer must be only one of the two options: ['Successful' or 'Failed'].

VQA question template for failure keyframe localization:

{task_image_sequence} These images are captured by the head camera of the dual-arm robot in chronological order.
Based on the visual evidence of the robot executing the task: '**{task_description}**', select the frame where the failure starts to happen.

You only need to choose the answer from the following four options provided: [A. B. C. D.]
A. **{keyframe_option_1}**
B. **{keyframe_option_2}**
C. **{keyframe_option_3}**
D. **{keyframe_option_4}**

VQA question template for failure subtask localization:

{task_image_sequence}These images are captured by the head camera of the dual-arm robot in chronological order.
The task the dual-arm robot in the images is doing is: '**{task_description}**'. From the recorded execution of the dual-arm robot shown in the images, determine which subtask failed during the process. Here are all the subtasks of this task
{subtasks}

You only need to choose the answer from the following four options provided: [A. B. C. D.]
A. **{subtask_option_1}**
B. **{subtask_option_2}**
C. **{subtask_option_3}**
D. **{subtask_option_4}**

VQA question template for failure type identification:

{task_image_sequence} These images are captured by the head camera of the dual-arm robot in chronological order.
The task the dual-arm robot in the images is doing is '**{task_description}**'. Given the task execution images of the dual-arm robot, determine the failure type. Choose the correct answer from the options provided.
task planning
gripper state
gripper 6d-pose
human intervention
The failure modes are defined as follows: 'gripper state' indicates that the gripper does not close or open properly or its level of closure or opening is insufficient; 'gripper 6d-pose' means the end-effector fails to reach its correct position or orientation; 'task planning' signifies that a subtask is skipped or the wrong target object is selected; 'human intervention' means that human interference makes the task impossible to complete.

You only need to answer one of the four types: ['gripper state', 'gripper 6d-pose', 'task planning', 'human intervention’]

Figure 4. VQA question templates for failure detection, failure keyframe localization, failure subtask localization, and failure type identification.

VQA question template for low-level avoidance:

Your task is to analyze the behavior of a dual-arm robot based on images provided, and to suggest adjustments to correct any failures. Your responses should specifically address spatial directions. The spatial directions are defined as follows: the images are captured from the head camera of the dual-arm robot. The front of the robot is the positive X-axis direction(the vertical component towards the top of the image), the left side is the positive Y-axis direction, and the above side is the positive Z-axis direction, which is a right-handed coordinate system. In summary: front, left, and above are considered positive directions, while back, right, and below are negative directions.

{task_image_sequence}These images are captured by the head camera of the dual-arm robot in chronological order.

The task the dual-arm robot in the images is doing is: '**{task_description}**'.

{avoidance_keyframe_image} This is the keyframe just before the failure happened.

The failure is categorized as a '**{failure_type}**' error. Based on the keyframe before a failure happened, please provide low-level corrective commands that could help the robot avoid this failure and complete the task successfully. The command must clearly include the following elements:

1. Target Gripper: Specify either 'left gripper' or 'right gripper'.
2. Specific Action: Choose one or more actions from the list below to combine:
   - Move: The direction must be one of [upward, downward, left, right, forward, backward], optionally with a magnitude [slightly, significantly].
   - Rotate: The direction must be one of [clockwise, counter-clockwise].
   - Grasp: The state must be one of [open, close].

You only need to choose the answer from the following four options provided: [A. B. C. D.]

A. **{low_level_command_option_1}**
B. **{low_level_command_option_2}**
C. **{low_level_command_option_3}**
D. **{low_level_command_option_4}**

VQA question template for low-level correction:

Your task is to analyze the behavior of a dual-arm robot based on images provided, and to suggest adjustments to correct any failures. Your responses should specifically address spatial directions. The spatial directions are defined as follows: the images are captured from the head camera of the dual-arm robot. The front of the robot is the positive X-axis direction(the vertical component towards the top of the image), the left side is the positive Y-axis direction, and the above side is the positive Z-axis direction, which is a right-handed coordinate system. In summary: front, left, and above are considered positive directions, while back, right, and below are negative directions.

{task_image_sequence}These images are captured by the head camera of the dual-arm robot in chronological order.

The task the dual-arm robot in the images is doing is: '**{task_description}**'.

{correction_keyframe_image} This is the keyframe after the failure happened.

The failure is categorized as a '**{failure_type}**' error. Based on the keyframe after the failure happened, please provide low-level corrective commands that could help the robot correct this failure and complete the task successfully. The command must clearly include the following elements:

1. Target Gripper: Specify either 'left gripper' or 'right gripper'.
2. Specific Action: Choose one or more actions from the list below to combine:
   - Move: The direction must be one of [upward, downward, left, right, forward, backward], optionally with a magnitude [slightly, significantly].
   - Rotate: The direction must be one of [clockwise, counter-clockwise].
   - Grasp: The state must be one of [open, close].

You only need to choose the answer from the following four options provided: [A. B. C. D.]

A. **{low_level_command_option_1}**
B. **{low_level_command_option_2}**
C. **{low_level_command_option_3}**
D. **{low_level_command_option_4}**

Figure 5. VQA question templates for low-level textual guidance (low-level avoidance and low-level correction).

VQA question template for failure reason:

You are a senior expert in robotic manipulation failure analysis and task-level planning. Your task is to analyze the behavior of a dual-arm robot based on images provided and provide a strategic failure diagnosis.
{task_image_sequence}These images are captured by the head camera of the dual-arm robot in chronological order.
The task the dual-arm robot in the images is doing is: **'{task_description}'**. You should give me a concise root-cause analysis (2–3 sentences, $\leq$ 60 words).

VQA question template for high-level avoidance:

You are a senior expert in robotic manipulation failure analysis and task-level planning. Your task is to analyze the behavior of a dual-arm robot based on images provided and suggest adjustments to help the robot avoid the failure before it happening.
{task_image_sequence}These images are captured by the head camera of the dual-arm robot in chronological order.
The task the dual-arm robot in the images is doing is: **'{task_description}'**. You should give me brief and feasible high-level commands.

VQA question template for high-level correction:

You are a senior expert in robotic manipulation failure analysis and task-level planning. Your task is to analyze the behavior of a dual-arm robot based on images provided and suggest adjustments to help the robot recover from the failure and continue to finish the task after the failure happening.
{task_image_sequence}These images are captured by the head camera of the dual-arm robot in chronological order.
The task the dual-arm robot in the images is doing is: **'{task_description}'**. You should give me brief and feasible high-level commands.

Figure 6. VQA question templates for failure reason and high-level textual guidance (avoidance and correction).

VQA question template for low-level avoidance (CoT):
{task_image_sequence}These images are captured by the head camera of the dual-arm robot in chronological order.
The task is to: '**{task_description}'**.

Please provide a complete analysis of the robot's execution.

Your analysis should follow these steps in your reasoning:
1. First, determine if the task was completed successfully (Your answer must be only one of the two options: ['Successful' or 'Failed']).
2. If the task execution is failed, perform **Failure Localization**: Identify the numerical position of the key image frame(1-N) just before the failure began.
3. Next, perform **Failure Classification**: Choose one of the following failure types. The types are defined as follows:
   - 'gripper state': The gripper does not close or open properly or its level of closure or opening is insufficient.
   - 'gripper 6d-pose': The end-effector fails to reach its correct position or orientation.
   - 'task planning': A subtask is skipped or the wrong target object is selected.
   - 'human intervention': Human interference makes the task impossible to complete.
4. Finally, provide a **Low-Level Avoidance Command**: This command must be a clear natural language instruction to help the robot avoid the failure. It must include the following elements:
   - **Target Gripper**: Specify either 'left gripper' or 'right gripper'.
   - **Specific Action**: Choose one or more from:
     - Move: Direction must be one of [upward, downward, left, right, forward, backward], optionally with a magnitude [slightly, significantly].
     - Rotate: Direction must be one of [clockwise, counter-clockwise].
     - Grasp: State must be one of [open, close].
Present your response using the following format:
- Your step-by-step reasoning must be enclosed in <Think></Think> tags.
- Your final answer must be enclosed in <Answer></Answer> tags.
- The answer block should contain the 'success detection' status, the 'keyframe' and the 'low level commands'.

VQA question template for low-level correction (CoT):
{task_image_sequence}These images are captured by the head camera of the dual-arm robot in chronological order.
The task is to: '**{task_description}'**.

Please provide a complete analysis of the robot's execution.

Your analysis should follow these steps in your reasoning:
1. First, determine if the task was completed successfully (Your answer must be only one of the two options: ['Successful' or 'Failed']). If the task execution is failed, execute step 2 and 3.
2. Next, perform **Failure Classification**: Choose one of the following failure types. The types are defined as follows:
   - 'gripper state': The gripper does not close or open properly or its level of closure or opening is insufficient.
   - 'gripper 6d-pose': The end-effector fails to reach its correct position or orientation.
   - 'task planning': A subtask is skipped or the wrong target object is selected.
   - 'human intervention': Human interference makes the task impossible to complete.
3. Finally, provide a **Low-Level Correction Command**: This command must be a clear natural language instruction to help the robot correct and recover from the failure on the frame number **{correction_keyframe}**. It must include the following elements:
   - **Target Gripper**: Specify either 'left gripper' or 'right gripper'.
   - **Specific Action**: Choose one or more from:
     - Move: Direction must be one of [upward, downward, left, right, forward, backward], optionally with a magnitude [slightly, significantly].
     - Rotate: Direction must be one of [clockwise, counter-clockwise].
     - Grasp: State must be one of [open, close].
Present your response using the following format:
- Your step-by-step reasoning must be enclosed in <Think></Think> tags.
- Your final answer must be enclosed in <Answer></Answer> tags.
- The answer block should contain the 'success detection' status and the 'low level commands'.

Figure 7. VQA question templates for CoT form of low-level textual guidance (avoidance and correction).

VQA question template for generating low-level visual guidance of avoidance (CoT):

{task_image_sequence}These images are captured by the head camera of the dual-arm robot in chronological order.
The task is to: '**{task_description}**'.

Please provide a complete analysis of the robot's execution.

Your analysis should follow these steps in your reasoning:
1. First, determine if the task was completed successfully (Your answer must be only one of the two options: ['Successful' or 'Failed']).
2. If the task execution is failed, perform **Failure Localization**: Identify the numerical position(1-N) of the key image frame just before the failure began.
3. Next, perform **Failure Classification**: Choose one of the following failure types. The types are defined as follows:
   - 'gripper state': The gripper does not close or open properly or its level of closure or opening is insufficient.
   - 'gripper 6d-pose': The end-effector fails to reach its correct position or orientation.
   - 'task planning': A subtask is skipped or the wrong target object is selected.
   - 'human intervention': Human interference makes the task impossible to complete.
4. Then, provide a **Low-Level Avoidance Command**: This command must be a clear natural language instruction to help the robot avoid the failure. It must include the following elements:
   - **Target Gripper**: Specify either 'left gripper' or 'right gripper'.
   - **Specific Action**: Choose one or more from:
     - Move: Direction must be one of [upward, downward, left, right, forward, backward], optionally with a magnitude [slightly, significantly].
     - Rotate: Direction must be one of [clockwise, counter-clockwise].
     - Grasp: State must be one of [open, close].
5. Finally, generate the **Visual Prompt Instructions** based on the low-level command.

When generating the visual prompts, you must follow these drawing rules:
1. Use **Straight Arrows** to indicate movement direction:
   - The color indicates which axis the movement is related to: Red for x-axis, green for y-axis, and blue for z-axis.
   - The spatial orientation of the arrow indicates positive/negative direction.
   - Arrows with multiple color segments mean movement along multiple axes.
2. Use **Rotating Arrows** to indicate rotation direction.
3. Use **ON/OFF** text to indicate the gripper state ('ON' for open, 'OFF' for close).
4. Use **Dual Crosshairs** to indicate that the object at the first crosshair should be aligned with the position of the second crosshair.
5. Use a **Crosshair** to indicate the correct target object.
6. Use a **Lock** icon to indicate that the selected object should remain stationary.
7. Use a **Rewind** icon to indicate that the selected object should return to its previous position or state.

Present your response using the following format:
- Your step-by-step reasoning must be enclosed in <Think></Think> tags.
- Your final answer must be enclosed in <Answer></Answer> tags.
- The answer block should contain the 'success detection' status, the 'keyframe', the 'low level commands', and the 'visual prompts'.

VQA answer template for generating low-level visual guidance of avoidance (CoT):

<Think>
First, I need to determine if the task was successful. The provided data indicates **{failure_detection}**.
Next, I must localize when the failure began. The keyframe just before the failure is image number **{failure_keyframe}**.
Now I need to classify the failure. The failure type is identified as '**{failure_type}**'. Based on this, I will formulate a low-level command to avoid the failure. The corrective action is: '**{low_level_commands}**'.
Finally, I will generate the visual prompt commands corresponding to this low-level correction. The required drawings are:
{visual_symbols_code_elements}
</Think>
<Answer>
"success detection": "**{failure_detection}**"
"keyframe": "**{failure_keyframe}**"
"low level commands": "**{low_level_commands}**"
"visual prompts": "**{visual_symbols_code_elements}**"
</Answer>

Figure 8. VQA templates for generating low-level visual guidance of avoidance (CoT).

VQA question template for generating low-level visual guidance of correction (CoT):
{task_image_sequence}These images are captured by the head camera of the dual-arm robot in chronological order.
The task is to: '**{task_description}**'.

Please provide a complete analysis of the robot's execution.

Your analysis should follow these steps in your reasoning:
1. First, determine if the task was completed successfully (Your answer must be only one of the two options: ['Successful' or 'Failed']). If the task execution is failed, execute following steps.
2. Next, perform **Failure Classification**: Choose one of the following failure types. The types are defined as follows:
   - 'gripper state': The gripper does not close or open properly or its level of closure or opening is insufficient.
   - 'gripper 6d-pose': The end-effector fails to reach its correct position or orientation.
   - 'task planning': A subtask is skipped or the wrong target object is selected.
   - 'human intervention': Human interference makes the task impossible to complete.
3. Then, provide a **Low-Level Correction Command**: This command must be a clear natural language instruction to help the robot correct and recover from the failure based on the frame number **{correction_keyframe}**. It must include the following elements:
   - **Target Gripper**: Specify either 'left gripper' or 'right gripper'.
   - **Specific Action**: Choose one or more from:
     - Move: Direction must be one of [upward, downward, left, right, forward, backward], optionally with a magnitude [slightly, significantly].
     - Rotate: Direction must be one of [clockwise, counter-clockwise].
     - Grasp: State must be one of [open, close].
4. Finally, generate the **Visual Prompt Instructions** based on the low-level command.

When generating the visual prompts, you must follow these drawing rules:
1. Use **Straight Arrows** to indicate movement direction:
   - The color indicates which axis the movement is related to: Red for x-axis, green for y-axis, and blue for z-axis.
   - The spatial orientation of the arrow indicates positive/negative direction.
   - Arrows with multiple color segments mean movement along multiple axes.
2. Use **Rotating Arrows** to indicate rotation direction.
3. Use **ON/OFF** text to indicate the gripper state ('ON' for open, 'OFF' for close).
4. Use **Dual Crosshairs** to indicate that the object at the first crosshair should be aligned with the position of the second crosshair.
5. Use a **Crosshair** to indicate the correct target object.
6. Use a **Lock** icon to indicate that the selected object should remain stationary.
7. Use a **Rewind** icon to indicate that the selected object should return to its previous position or state.

Present your response using the following format:
- Your step-by-step reasoning must be enclosed in <Think></Think> tags.
- Your final answer must be enclosed in <Answer></Answer> tags.
- The answer block should contain the 'success detection' status, the 'low level commands', and the 'visual prompts'.

VQA answer template for generating low-level visual guidance of correction (CoT):
<Think>
First, I need to determine if the task was successful. The provided data indicates **{failure_detection}**.
Now I need to classify the failure. The failure type is identified as '**{failure_type}**'. Based on this, I will formulate a low-level command to correct and recover from the failure, based on the frame number **{correction_keyframe}**. The corrective action is: '**{low_level_commands}**'.
Finally, I will generate the visual prompt commands corresponding to this low-level correction. The required drawings are: **{visual_symbols_code_elements}**
</Think>
<Answer>
"success detection": "**{failure_detection}**"
"low level commands": "**{low_level_commands}**"
"visual prompts": "**{visual_symbols_code_elements}**"
</Answer>

Figure 9. VQA templates for generating low-level visual guidance of correction (CoT).

Prompt for scoring open-ended high-level VQA pairs:
You are an expert evaluator specializing in robotic manipulation tasks, capable of understanding and judging the semantic accuracy of task descriptions involving robot actions, objects, and spatial reasoning.

You will rate how semantically consistent Text B is compared to Text A on a continuous scale from 0.0 to 1.0.

Evaluation criteria:
- Focus on whether the two descriptions convey the same manipulation **intent, sequence, and outcome**.
- Key aspects to consider:
1. **Gripper usage**: left vs. right gripper, open/close actions.
2. **Action correctness**: pick, place, move, align, push, lift, etc.
3. **Object and target consistency**: same object names and corresponding targets.
4. **Order and causality**: whether the sequence of steps is preserved.
5. **Success condition**: whether the goal or outcome matches.

Scoring guidance:
- 1.0 → Exactly the same meaning and correct execution description.
- 0.8–0.9 → Minor paraphrasing differences but semantically identical in robot actions and outcomes.
- 0.6–0.7 → Mostly correct but with small errors (e.g., gripper swapped, one step missing).
- 0.3–0.5 → Partially correct but with notable mismatches in object, action, or order.
- 0.1–0.2 → Only slightly related, mostly incorrect.
- 0.0 → Completely unrelated or contradicting actions.

Return **only** the numeric score between 0.0 and 1.0.

Text A (reference): **{ground_truth}**
Text B (candidate): **{model_raw_response}**

ATTENTION: You MUST ONLY give me a score number. The score MUST be from 0.0 to 1.0.

Prompt for scoring open-ended low-level VQA pairs:
You are an expert evaluator of semantic similarity between texts especially in robotic manipulation tasks.

Your evaluation should consider the following aspects:
1. **Semantic Similarity**: The degree to which the two texts convey equivalent meanings and intentions, regardless of surface-level wording differences.
2. **Content Completeness**: Whether all critical information elements are present in both texts, including gripper specifications, movement directions, and command details.
3. **Functional Equivalence**: Whether the described actions achieve the same robotic manipulation goals and operational outcomes.

Scoring Scale
- 1.0: Exact semantic equivalence with complete functional match
- 0.0: Completely unrelated or contradictory meanings
- Fractional scores between 0.0 and 1.0: Partially correct in proportion, with primary scoring points focused on low-level commands.

You should pay attention to the description of left/right gripper, low-level commands(move, rotate, open, close, hold still). They are the key criteria of the evaluation, if the predicted has the opposite answer to the ground truth, you should give 0.0. If the answer has no direction but the object, it's not an expected answer, but you can still give a lower score. If there is other info like keyframe, success detection, the success detection(Successful/Failed) between the predicted and the ground truth must be the same, or you should give 0.0. The keyframe can have a tolerance of 1(for example, the predicted keyframe is 5 and the ground truth keyframe is 4 or 6, it's still correct, and then you should check the low-level commands). If the predicted includes all the information in the ground truth, you can also give a score of 1.0.
Return ONLY the numeric score.

Text A(Ground Truth): **{ground_truth}**
Text B(Predicted): **{model_raw_response}**

You must ONLY answer a number. (**A fractional score ranging from 0.0 to 1.0**)

Figure 10. Prompts for scoring open-ended VQA pairs.

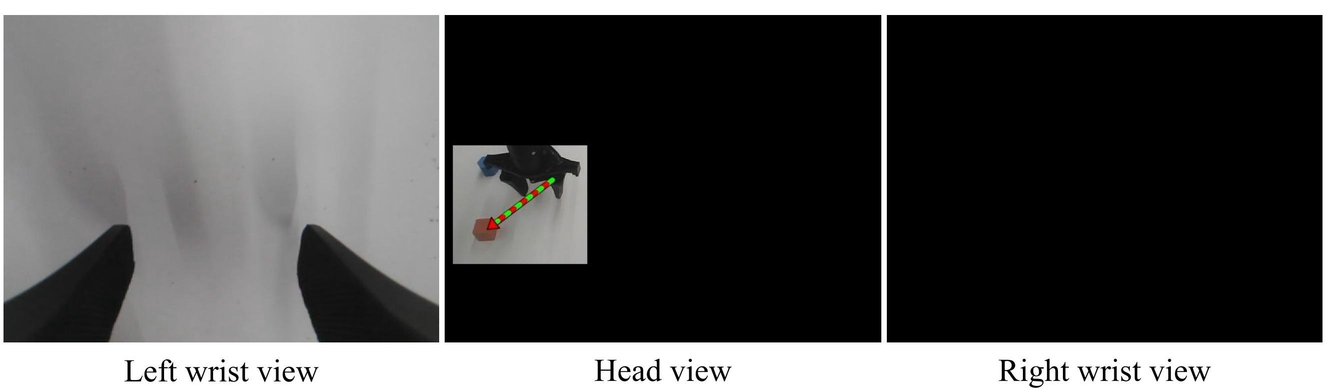


Figure 11. The mask in the visual symbols following dataset.

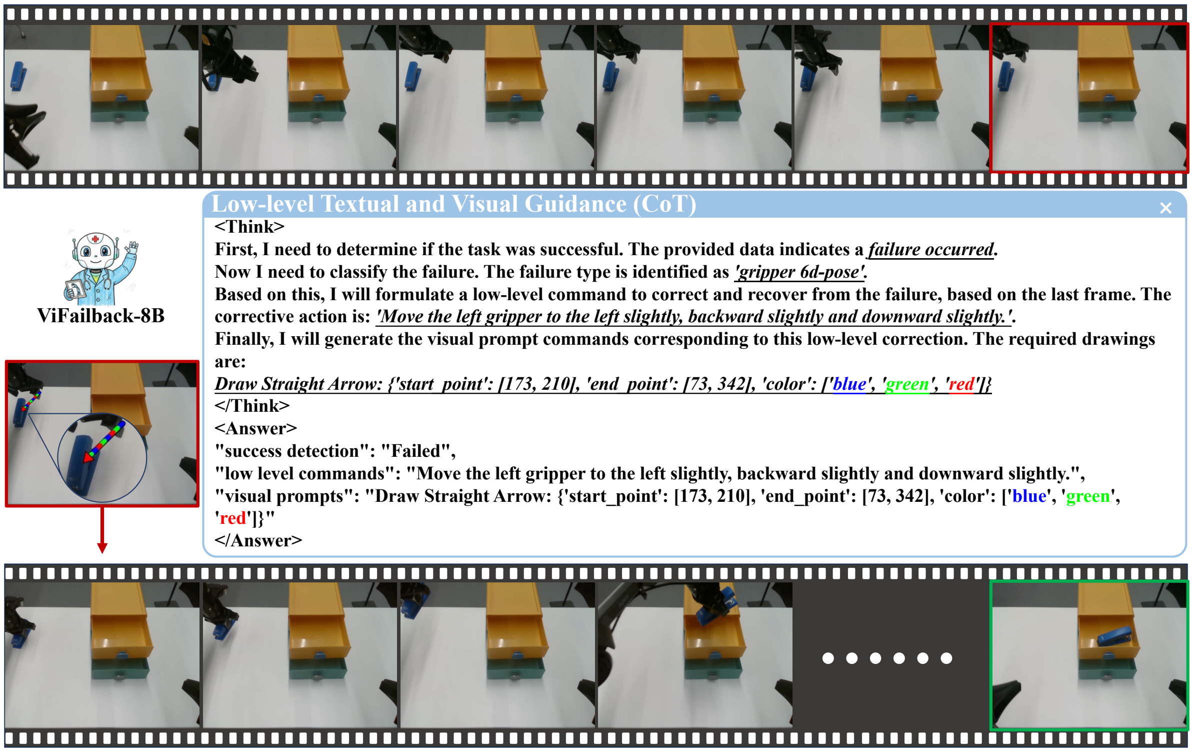


Figure 12. **Visualization of the PlaceOne task. Top & Middle:** ViFailback-8B diagnoses failures and provides low-level multimodal (textual and visual) guidance. **Bottom:** The corrective actions are executed by the **PMC method** to recover from the failure.

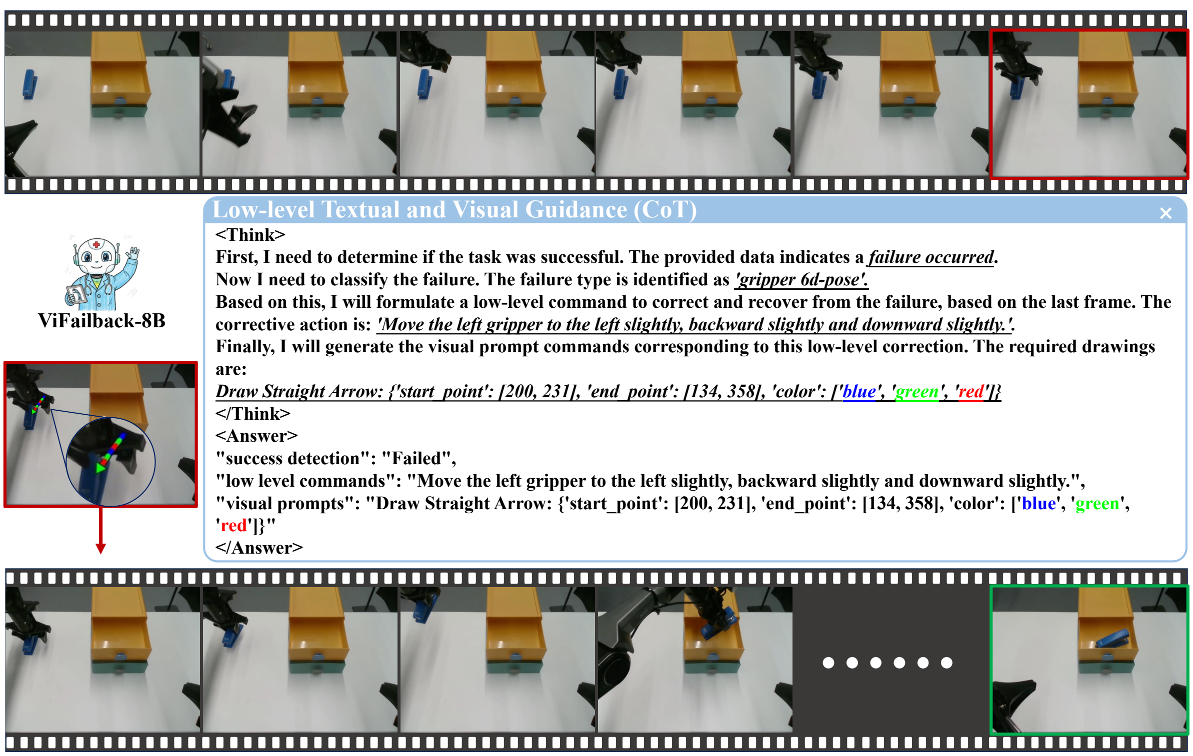


Figure 13. **Visualization of the PlaceOne task. Top & Middle:** ViFailback-8B diagnoses failures and provides low-level multimodal (textual and visual) guidance. **Bottom:** The corrective actions are executed by the **VSF method** to recover from the failure.

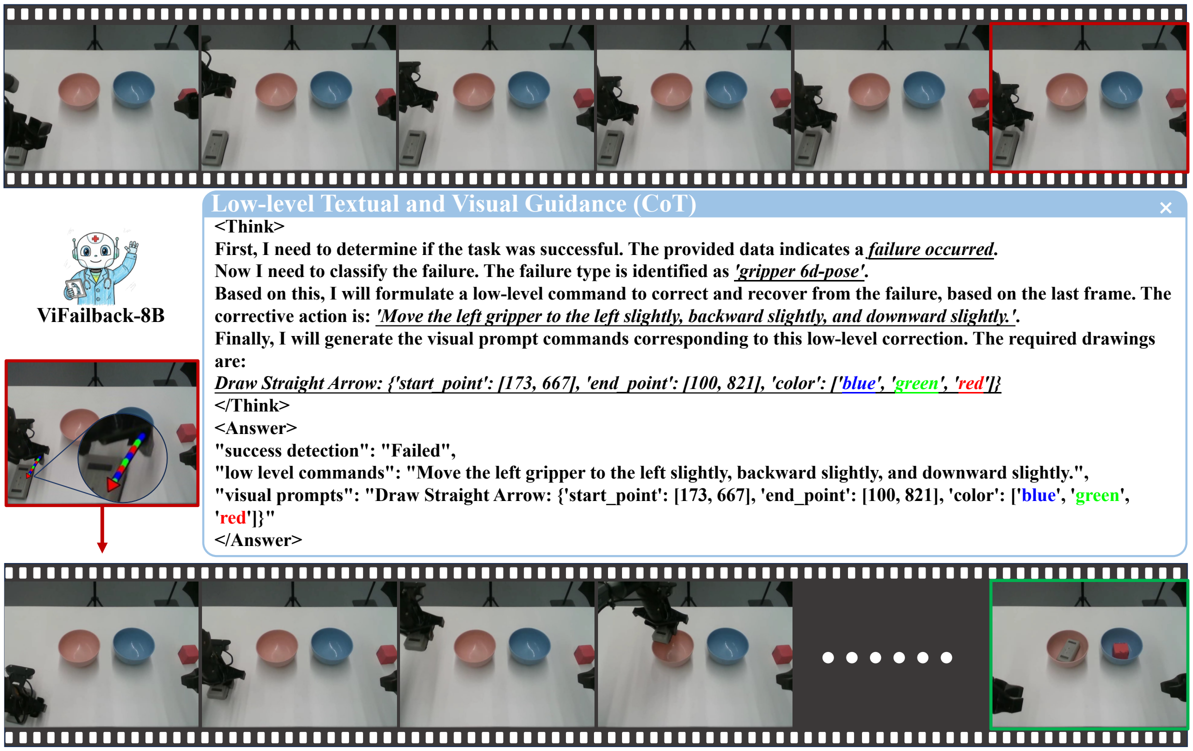


Figure 14. **Visualization of the PlaceTwo task. Top & Middle:** ViFailback-8B diagnoses failures and provides low-level multimodal (textual and visual) guidance. **Bottom:** The corrective actions are executed by the **PMC method** to recover from the failure.

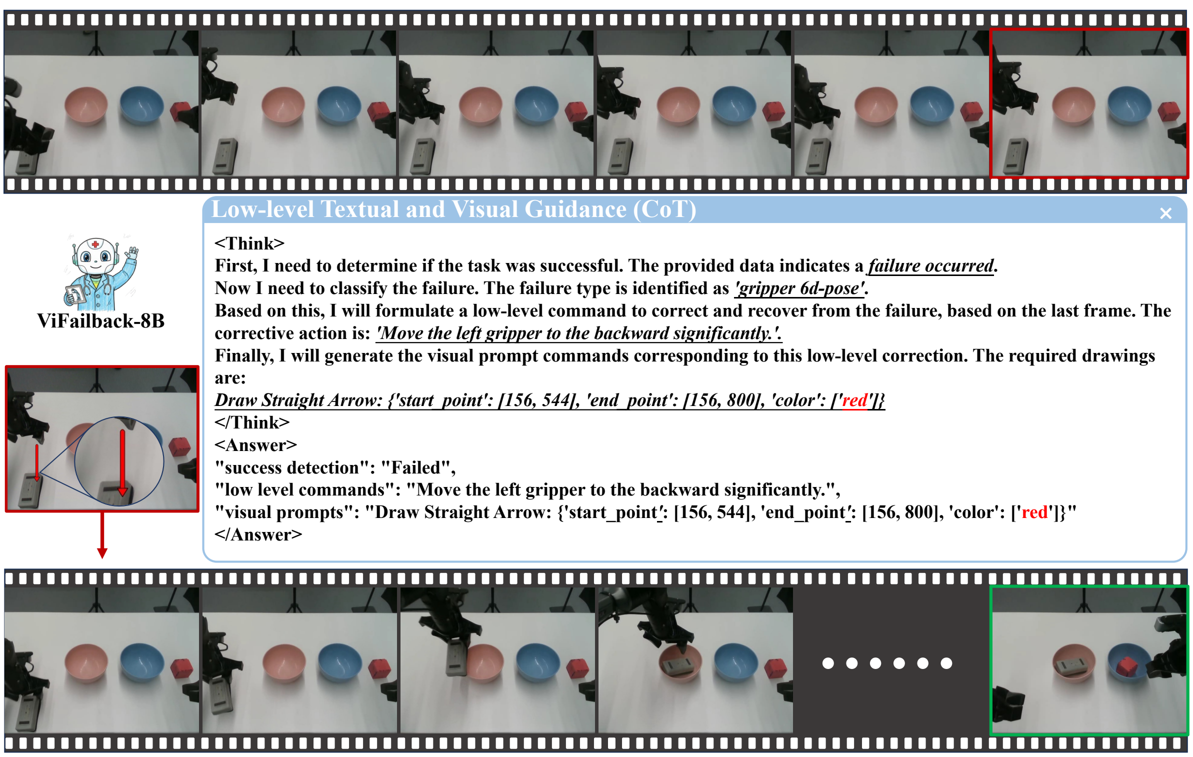


Figure 15. **Visualization of the PlaceTwo task. Top & Middle:** ViFailback-8B diagnoses failures and provides low-level multimodal (textual and visual) guidance. **Bottom:** The corrective actions are executed by the **VSF method** to recover from the failure.

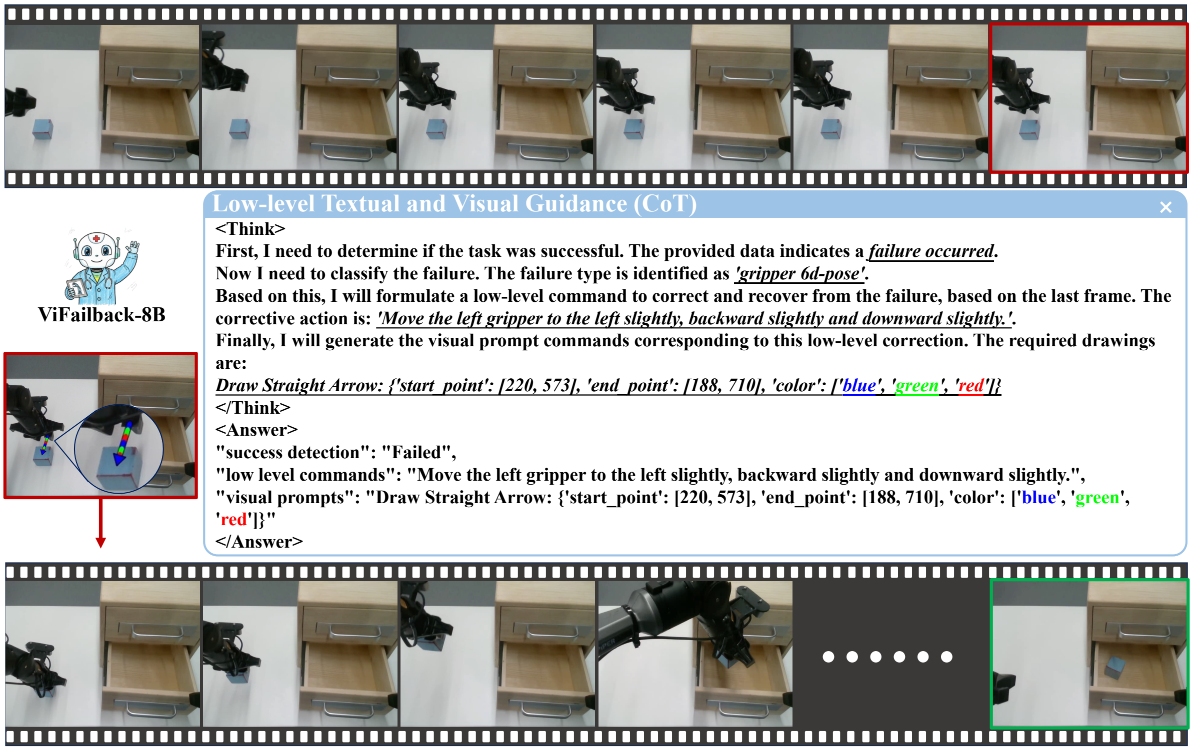


Figure 16. **Visualization of the Pull&Place task. Top & Middle:** ViFailback-8B diagnoses failures and provides low-level multimodal (textual and visual) guidance. **Bottom:** The corrective actions are executed by the **PMC method** to recover from the failure.

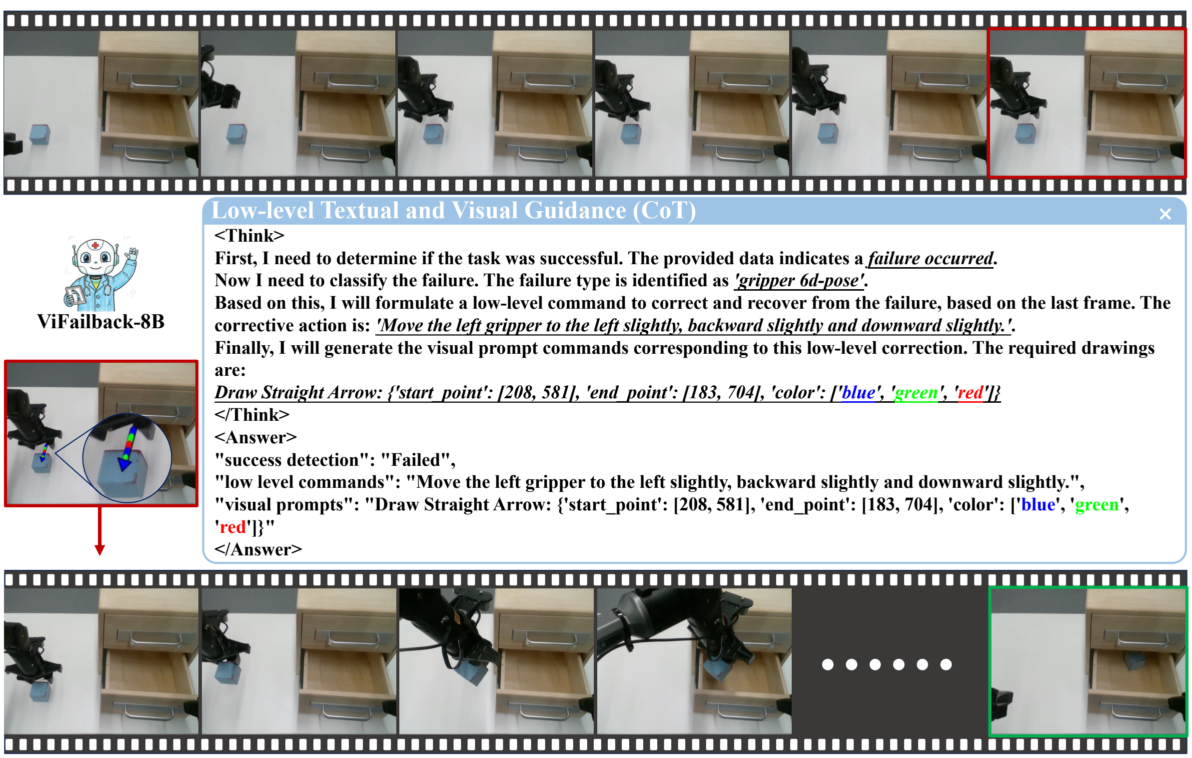
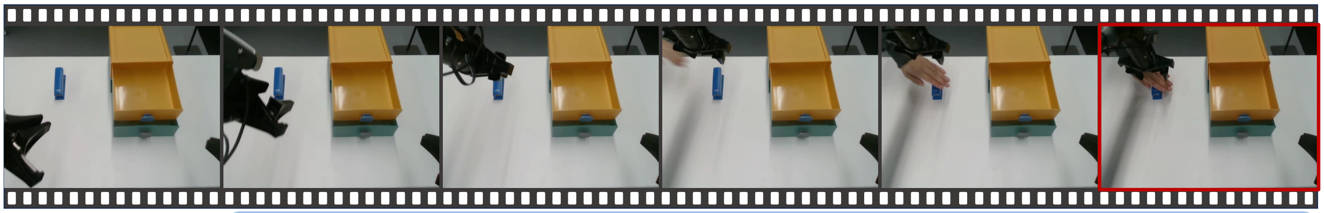
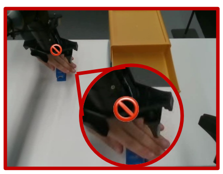


Figure 17. **Visualization of the Pull&Place task. Top & Middle:** ViFailback-8B diagnoses failures and provides low-level multimodal (textual and visual) guidance. **Bottom:** The corrective actions are executed by the **VSF method** to recover from the failure.



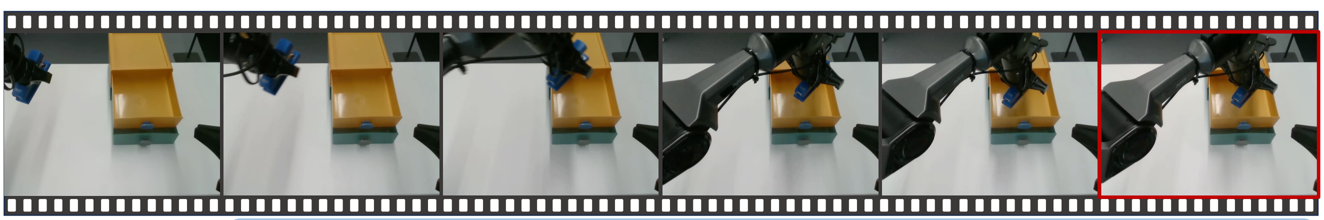
Low-level Textual and Visual Guidance (CoT)

ViFailback-8B



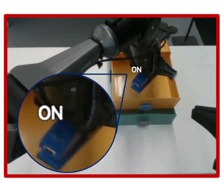
<Think>
First, I need to determine if the task was successful. The provided data indicates a *failure occurred*.
Now I need to classify the failure. The failure type is identified as '*human intervention*'.
Based on this, I will formulate a low-level command to correct and recover the failure. Thus, *I cannot continue to finish the task due to the inevitable human intervention and should hold still*.
Finally, I will generate the visual prompt commands corresponding to this low-level correction. The required drawings are:
Draw Lock: {'position': [220, 288]}
</Think>
<Answer>
"success detection": "Failed",
"low level commands": "Hold still."
"visual prompts": "Draw Lock: {'position': [220, 288]}"
</Answer>

Figure 18. **Failure Type: Human Intervention.** When human intervention is detected, ViFailback-8B provides guidance to pause execution, awaiting the removal of the intervention.



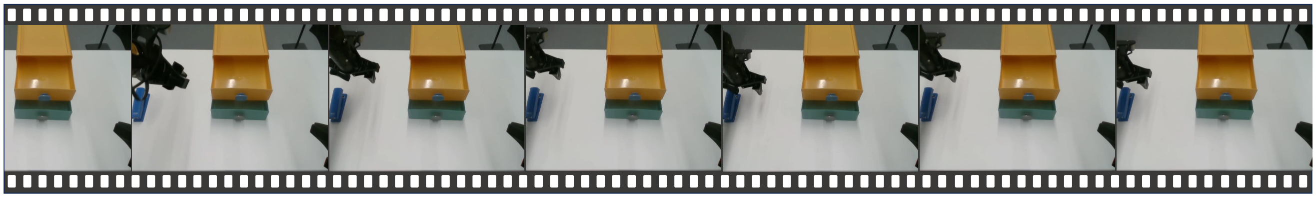
Low-level Textual and Visual Guidance (CoT)

ViFailback-8B



<Think>
First, I need to determine if the task was successful. The provided data indicates a *failure occurred*.
Now I need to classify the failure. The failure type is identified as '*gripper state*'.
Based on this, I will formulate a low-level command to correct and recover from the failure, based on the last frame. The corrective action is: '*Open the left gripper*'.
Finally, I will generate the visual prompt commands corresponding to this low-level correction. The required drawings are:
Draw ON: {'position': [620, 331]}
</Think>
<Answer>
"success detection": "Failed",
"low level commands": "Open the left gripper.",
"visual prompts": "Draw ON: {'position': [620, 331]}"
</Answer>

Figure 19. **Failure Type: Gripper State.** The gripper failed to release the object during placement. ViFailback-8B generates guidance to open the gripper for successful placement.



Failure Reason

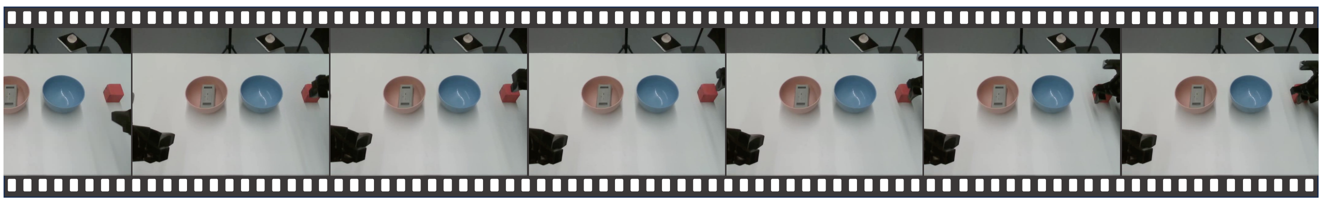
The task failed because **the left gripper approached the stapler with a significant lateral translation error, misaligning its grasp pose**. This occurred during the approach and alignment subtask. As a result, the gripper failed to engage the stapler correctly, preventing pickup. This indicates a fundamental gap in the robot's 6D pose control for fine manipulation.

High-level Guidance

Adjust the left gripper to complete 'Move the left arm to approach and align the stapler.' by **repositioning it vertically above the stapler** and executing a slow, controlled descent with rotational alignment to match the stapler's orientation before attempting grasp.

ViFailback-8B

Figure 20. Failure reason and High-level guidance of the PlaceOne task.



Failure Reason

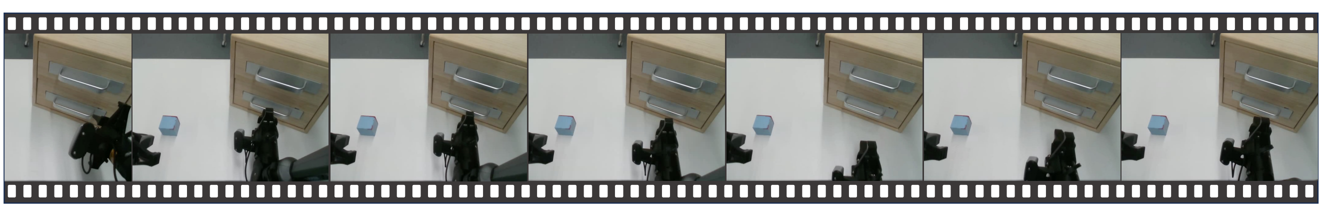
The task failed because **the right gripper approached the red cube with a significant 6D-pose error, misaligning its grasp axis**. This occurred during the subtask to align the cube for transfer. As a result, the gripper failed to securely grasp the cube, preventing placement into the blue bowl. This indicates a fundamental gap in real-time pose estimation and corrective motion control for fine manipulation.

High-level Guidance

Adjust the right gripper to complete 'Move the right arm to approach and align the red cube.' by **retracting slightly, re-estimating the cube's 6D pose, and executing a new approach trajectory** with tighter position and orientation tolerances to achieve a successful grasp.

ViFailback-8B

Figure 21. Failure reason and High-level guidance of the PlaceTwo task.



Failure Reason

The task failed because **the right gripper's 6D pose was misaligned with the bottom drawer handle during approach, causing a translation error.** As a result, the gripper could not engage the handle to pull the drawer. This indicates a fundamental gap in spatial pose calibration for fine manipulation tasks.

High-level Guidance

Adjust the right gripper to complete 'Move the right arm to align with the bottom drawer handle.' by **repositioning it vertically and horizontally to match the handle's centerline**, then attempt a controlled grasp to initiate drawer opening.

ViFailback-8B

Figure 22. Failure reason and High-level guidance of the Pull&Place task.