Appendix

Contents

A. Real-world Execution Sequences for the Eight Tasks

In this section, we demonstrate the execution sequence of the eight real-world tasks in 1.

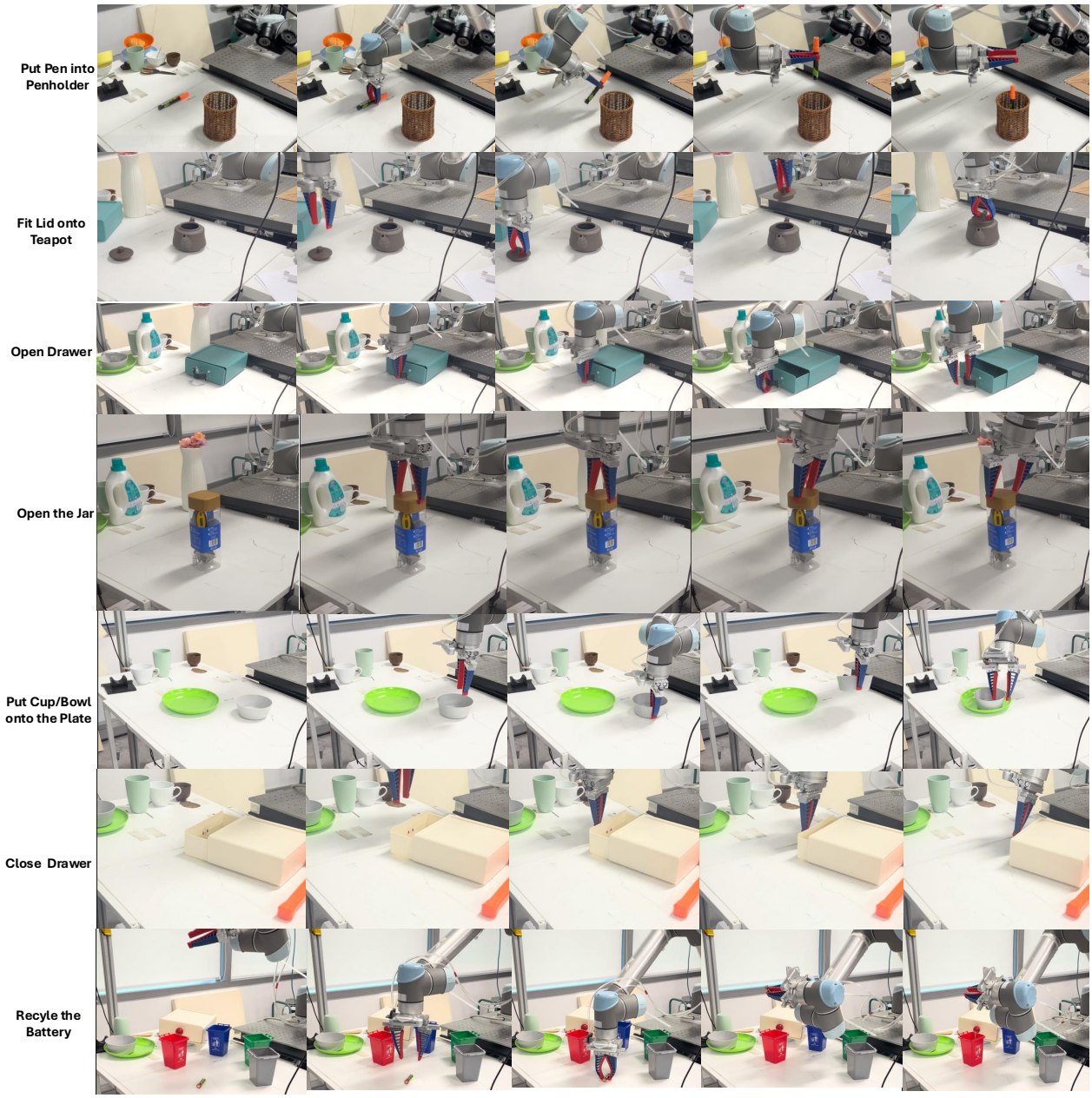

Figure 1. The sequences of execution for the eight real-world tasks.

From the execution sequence, we can see our method can manage diverse tasks, including non-prehensile manipulation (e.g., press red button, open the jar), pick-and-place (e.g., put cup/bowl onto the plate), articulation object manipulation (e.g., open/close drawer), manipulation with alignment (e.g., put pen into penholder), and manipulation with precise localization (e.g., fit lid onto teapot).

B. Robust Experiment Settings on Scenes and Task

We illustrate the ten scene settings for three tasks in Fig 2. For each of the tasks, we fully randomize the object initial positions, object initial orientations, and background object placements under the robot workspace and camera view volume.

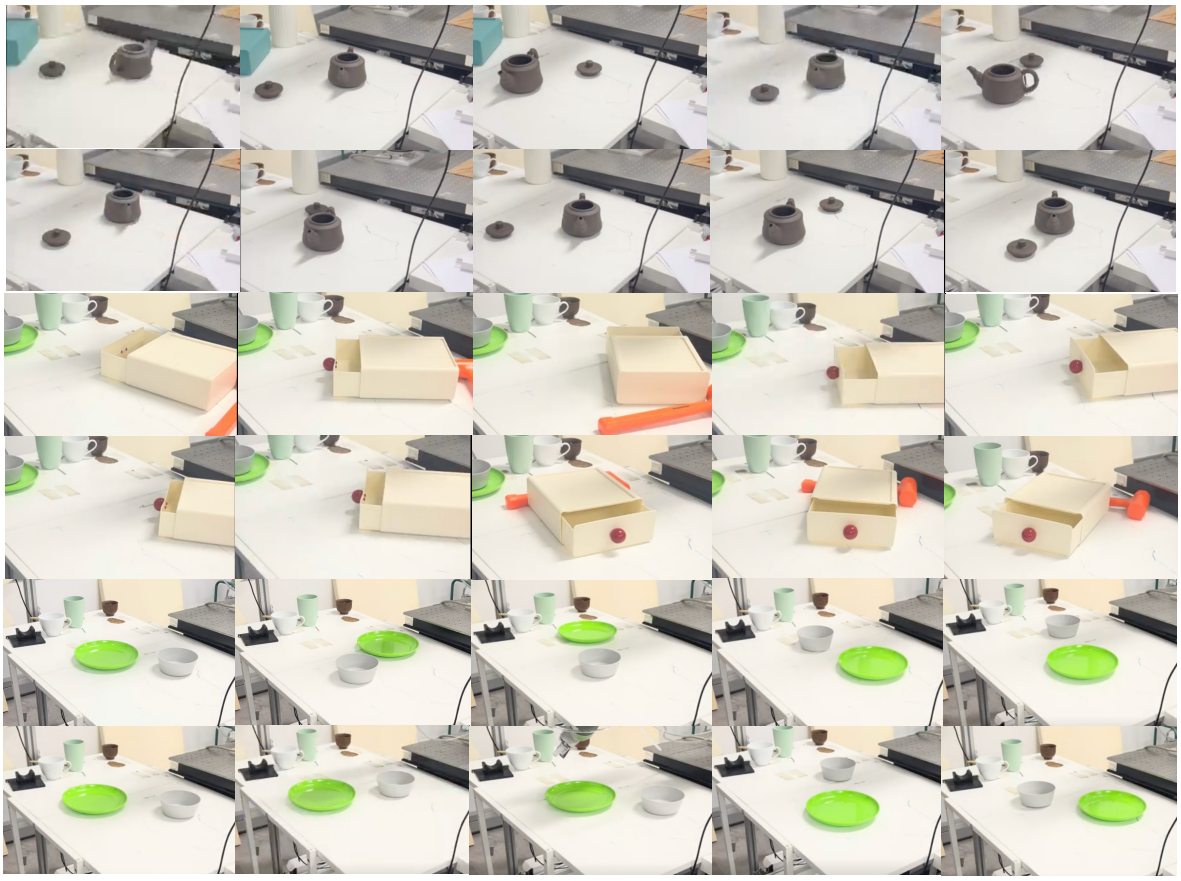

Figure 2. The initial scene settings.

C. Tasks List in Quantitative Study on Action-Generalizability and VLM-Comprehensibility

The 33 tasks we employed in the preliminary experiments are listed as follows:

1. Sort the Red Cube: Pick the red cube from a group and place it inside the red circle.
2. Bin the Blue Cylinder: Pick the blue cylinder and drop it into the bin marked with a blue square.
3. Stack Cube on Cube: Pick the green cube and stack it on top of the yellow cube.
4. Move the Soda Can: Pick the soda can from the left table and place it on the right table.
5. Fill the Tray: Pick the AA battery and place it into the empty slot in the plastic tray.
6. Insert the USB Drive: Pick the USB drive from the table and insert it into the laptop's USB port.
7. Assemble the LEGO: Pick the 2x4 LEGO brick and attach it to the red baseplate, connecting it to two other bricks.
8. Place the Ring: Pick the wooden ring and place it onto the vertical post.
9. Put the Lid on the Jar: Pick the plastic jar lid and place it on top of the jar
10. Hang the Key: Pick the key and hang it on the keyhook by its hole. Non-Prehensile Manipulation
11. Push the Dice: Push the white dice across the table until it crosses the black line.
12. Flip the Pancake: Use the spatula to flip the pancake in the frying pan.
13. Close the Drawer: Push the kitchen drawer closed using the flat of the gripper.
14. Press the Doorbell: Press the round, lit doorbell button on the wall.
15. Align the Block: Push the wooden block until it is flush against the corner of the table.
16. Scoop the Rice: Use the metal spoon to scoop rice from the pot into the bowl.
17. Stir the Soup: Use the spoon to stir the liquid in the pot three times clockwise.
18. Hammer the Nail: Use the toy hammer to tap the nail until its head is flush with the board.
19. Screw in the Lightbulb: Pick the lightbulb and screw it into the empty lamp socket.
20. Pour the Water: Pick the pitcher and pour water into the empty glass until it is half-full.
21. Uncoil the Rope: Manipulate the coiled rope until it forms a straight line from start to end.
22. Fold the Washcloth: Fold the small, square washcloth in half.
23. Open the Bag: Use two grippers to pull the handles of the plastic bag apart.
24. Drape the Towel: Drape the hand towel over the horizontal bar.
25. Route the Cable: Route the USB cable around the two posts in an S-shape.
26. Grasp the Marble: Pick the glass marble from a flat surface.
27. Grasp the Coin: Pick the single coin from the table.
28. Re-grip the Screwdriver: Pick the screwdriver by its handle, then place it down and re-grip it by its shaft.
29. Pick the Book: Pick the paperback book from the shelf by its spine.
30. Hook the Mug: Hook a gripper finger through the handle of the coffee mug and lift it.
31. Place the T-Block: Pick the T-shaped block and insert it into the matching T-shaped slot on the board.
32. Assemble the Stack: Pick the large square block and place it on the table, then place the medium block on it, and finally the small block on top.
33. Plug in the Lamp: Pick the power plug from the floor and insert it into the wall outlet.

These tasks cover a wide range of tasks in including basic prehensile manipulation, precise manipulation, tool use, deformable object manipulation, non-prehensile manipulation, and assembly, which provide a comprehensive evaluations on VLM's robustness to generate intermediate representation and reflects the VLM's comprehensibility.

D. Complete Vocabulary and Grammar Table for Different Comparison Methods

To the best of our knowledge, this is the first study on analyzing the intermediate representation for VLM-based robot manipulation. Following the VLM-based robot manipulation works, we reformulate and summarize their prompt designs into vocabulary and grammar. The complete vocabulary and grammar settings for each method are illustrated in Tab. 1.

Method	Vocabulary $\mathcal{V}$	Grammar $\mathcal{G}$
ReKep	get_keypoint, gripper_close, gripper_open, move_to, get_gripper_pos, get_gripper_pose, $\mathcal{V}_{\text{Python}}$	cost $\rightarrow$ cost + cost, cost $\rightarrow$ cost_fns, kpts, kpts$\rightarrow$kpts,keypoint, kpts$\rightarrow$get_keypoint, kpts$\rightarrow$get_end_effector, $\mathcal{G}_{\text{Python}}$
OmniManip	gripper_close, gripper_open, move_to, get_gripper_pos, get_gripper_pose, get_keypoint, get_axis	cost $\rightarrow$ cost + cost, cost $\rightarrow$ angular constraint, p, p, $S \rightarrow$ distance constraint, p, p, p$\rightarrow$ get_keypoint, p$\rightarrow$ get_axis
Instruct2act	gripper_close, gripper_open, get_gripper_pos, get_gripper_pose, find, pick, place, pick_place, insert, push, press, move_above, move_parallel, move_perpendicular, flip, scoop, stir, tap, screw_rotation, controlled_pour, route_around, pull_apart, fold, straighten, $\mathcal{V}_{\text{Python}}$	action $\rightarrow$ action + action, action $\rightarrow$ v, segm, segm $\rightarrow$ find, obj, get_gripper_pos $\rightarrow$ action, find $\rightarrow$ action, pick $\rightarrow$ action, place $\rightarrow$ action, pick_place $\rightarrow$ action, insert $\rightarrow$ action, push $\rightarrow$ action, press $\rightarrow$ action, move_above $\rightarrow$ action, move_parallel $\rightarrow$ action, ... $\mathcal{G}_{\text{Python}}$

Table 1. Vocabulary $\mathcal{V}$ and Grammar $\mathcal{G}$ for each method.

From the vocabulary and grammar table, we can observe that works with high-level intermediate representation (Instruct2act) needs extensive vocabulary to cover our designed 33 tasks. Meanwhile, the vocabulary of the works with low-level intermedate representation(RekKep, OmniManip) is abstract and poor in VLM-comprehensibility.

E. "Atomic Action" and their associated Intermediate Representation

1. move something to something with an offset

```
        move_cost('<source object part>', centroid('<target object part>') + np.array([<x, y, z>]))
    + parallel_cost(get_axis('<source object part>'), vector) + upright_cost(up_part='<up object
    part of the same source object>', down_part='<down object part of the same source object>')

## 0. You can you can use function get_height('<object part>'), get_width('<object part>'),
    get_length('<object part>') to get the dimension of the object part to calculate for the offset
## 1. [x, y, z] is the EXTRA between the <target object part> and the <source object part>
    ## 1.0 If move above the target, [<x, y, z>] = [0, 0, some positive margin (> 0.1) + get_cheight
    ('<target object part>')]
    ## 2.0 If move to contact the target from front, [<x, y, z>] = [0, 0, 0]
## 2 (optional) Keep the parallel cost if <source object part> part need to be aligned with some
    vector direction after the move
    ## 2.0 if <source object part> aligned with the target object part, vector = get_axis('<target
    object part>')
    ## 2.1 if <source object part> aligned vertically, vector = [0, 0, -1]
## 3 (optional) If the <source object> needs to be upright after moved, keep upright_cost. down_part
     and up_part MUST be the same object!
## 4 All object part should follow the format "object part of the object"

```

2. pick something up or put something down when grasped

```
        move_cost_with_offset('<source object part>', offset=[<0, 0, z>])

## 0. You can use function get_height('<object part>'), get_width('<object part>'), get_length('<
    object part>') to get the dimension of the object part to calculate for the offset
## 1. If move up / lift, z= get_height('<object part>') + a positive margin
## 2. If move down / place,  z= - get_height('<object part>') + a negative margin

```

3. press something after aligned

```
        gripper_close_first_cost() + move_cost('gripper', '<target object part>')

```

4. pull to open something after grasped

```
        move_cost('gripper', centroid_last('gripper') + direction_of(start='<object you want to pull
     away from>', end='gripper') * <offset_distance in meters>)

## fill in the <object you want to pull away from> accordingly.
## <offset_distance in meters> > 0.1

```

5. push to close something after grasped

```
        gripper_open_cost()

## directly use gripper_open_cost() to open the gripper and release the grasped item

```

6. release something only

```
        gripper_open_cost()

## directly use gripper_open_cost() to open the gripper and release the grasped item

```

F. The RAG Database for Open-vocabulary Part Segmentation

Below, we visualize example cases from our RAG database used for open-vocabulary part segmentation. The six visualized cases include the microwave hinge, laptop lid affordance, cup spout, flower stem, teapot opening, and teapot spout. Each case has one or more object part segmentation examples, along with the words used to prompt the segmentation.

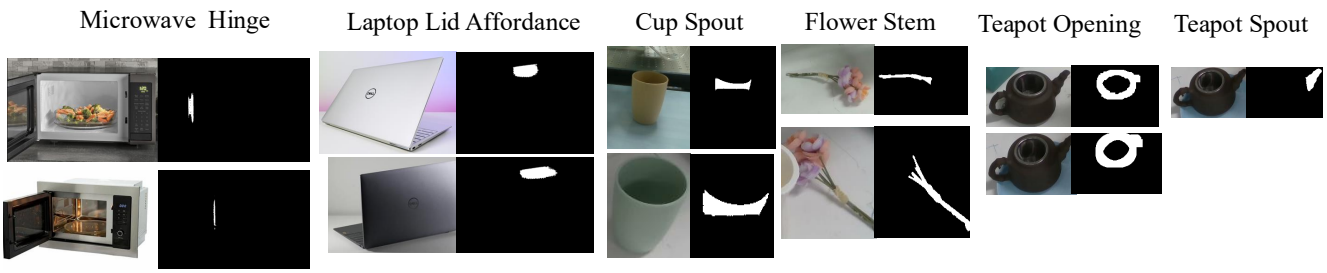

Figure 3. Example cases from the RAG database for open-vocabulary part segmentation.

G. More Results on Open-vocabulary Segmentation

In this section, we present the open-vocabulary segmentation results from our method and three state-of-the-art techniques: LISA, OV-Seg, and Grounded SAM2. We focus on five cases: button, doorbell, hammer cap, key ring, and pen cap, with results highlighted in blue. Our method demonstrates a higher precision in identifying critical object parts essential for robotic manipulation. For instance, it precisely locates the button to be pressed, the specific area on the doorbell, the cap of the hammer used in striking, the key ring for hanging keys, and the cap of the pen. In comparison, Grounded SAM2 often segments entire objects instead of key parts and may misidentify background elements. OV-Seg produces messy masks and typically identifies whole objects. Although LISA can estimate the locations of target parts, its masks can be messy and may occasionally encompass the entire object.

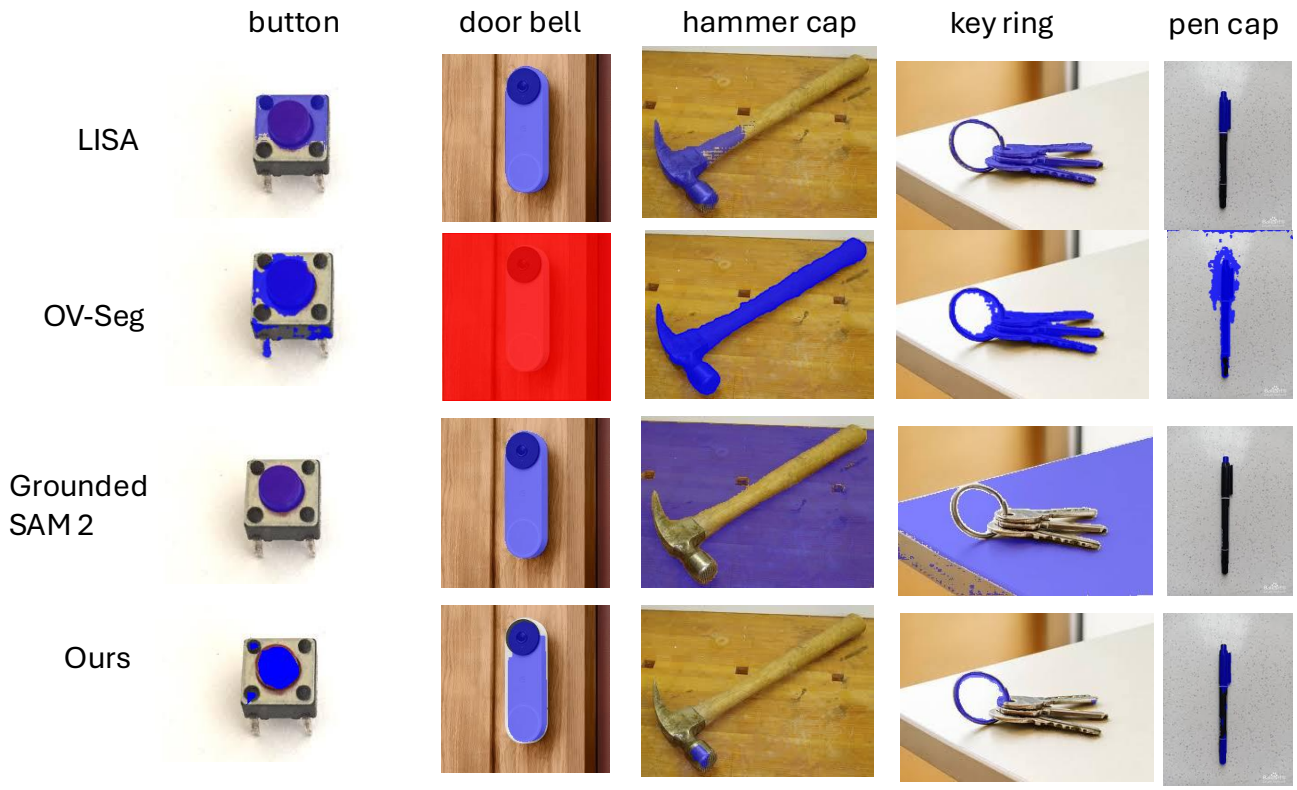

Figure 4. More open-vocabulary segmentation comparisons with other SOTA methods and our method.

H. RAG Database for Open-vocabulary Part Segmentation in Real-world Experiments.

Below, we provide some examples from our RAG database for open-vocabulary part segmentation in real-world experiments. We include 15 cases: button, bowl rim, bin, battery, bowl, basket, teapot opening, teapot spout, pen cap, plate, teapot lid, pen, drawer handle, drawer, and cap. Each case features a text prompt along with one or multiple segmentation results.

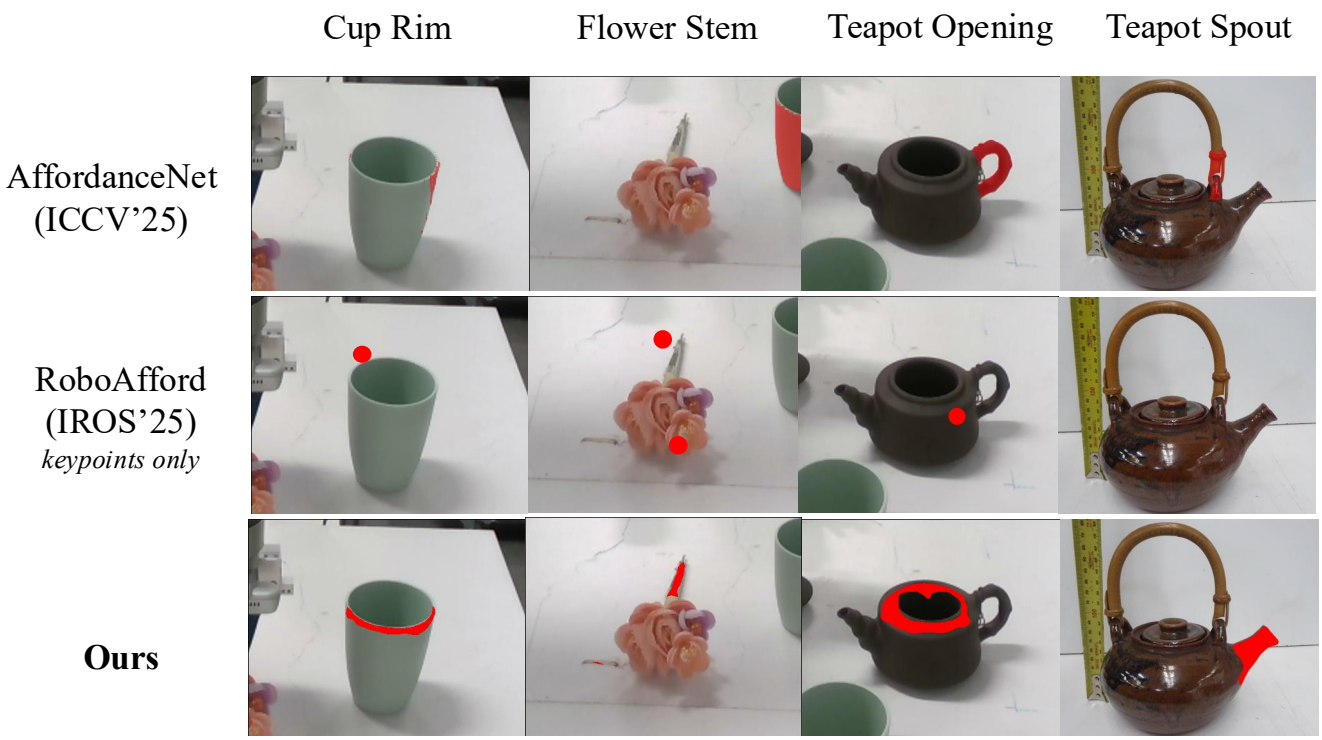

Figure 6. **Comparison with affordance segmentation.**

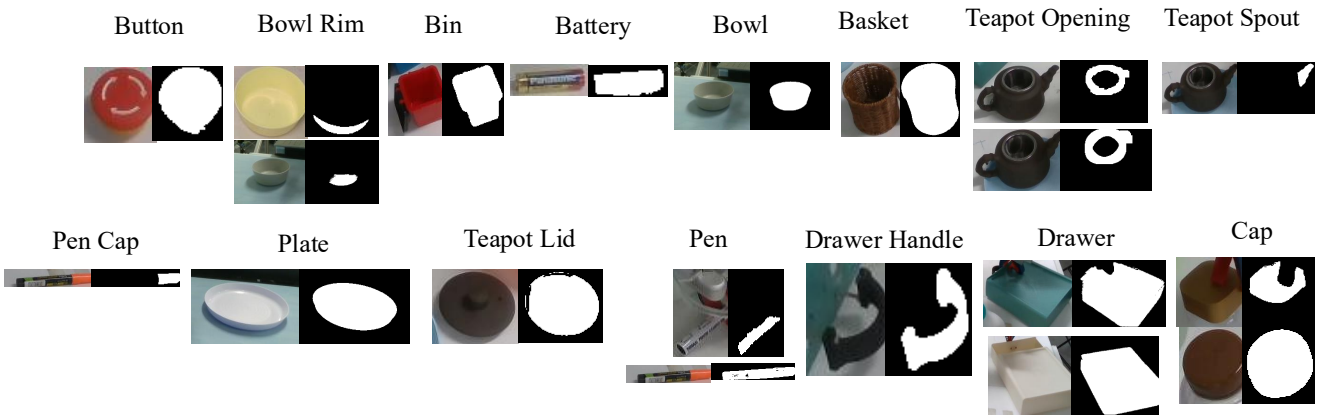

Figure 5. Some example cases from the RAG database for open-vocabulary segmentation in the real-world experiments.

I. Segmentation Comparisons with Affordance Method

J. Prompt and the Generated Outputs Used in the Analytical Experiments

J.1. OmniManip

Prompt:

```
For each of the task, decompose it into stages and write me the Python intermediate code for each
stage. Each code consists of Python basic arithmetic and choose extra function ONLY from the
vocabulary: $[$ get\_keypoint, get\_gripper\_pose, get\_gripper\_xyz, get\_axis, gripper\_close,
gripper\_open, move\_to$]$. get\_keypoint (obj:str) function returns the 3D coordinate (xyz) of the
 point of an object, and get\_axis(obj:str) function returns the three principle axes(pitch, yaw,
or roll axis) of obj. get\_gripper\_pose() returns the XYZ Euler angle of the gripper, get\_gripper
```

```
\_pos() return the xyz coordinate of the gripper. move\_to(x,y,z,X,Y,Z) move the gripper to
location(xyz) and pose(XYZ). Your should organize your output in Latex with enumerate and
lstlisting.
```

Generated Language By Qwen3-VL:

1. **Sort the Red Cube:** Pick the red cube from a group and place it inside the red circle.

```
# Stage 1: Move to red cube
cube_pos = get_keypoint("red_cube")
move_to(cube_pos[0], cube_pos[1], cube_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(cube_pos[0], cube_pos[1], cube_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Move to red circle
circle_pos = get_keypoint("red_circle")
move_to(circle_pos[0], circle_pos[1], circle_pos[2] + 0.1, 0, 0, 0)
move_to(circle_pos[0], circle_pos[1], circle_pos[2], 0, 0, 0)
gripper_open()
```

2. **Bin the Blue Cylinder:** Pick the blue cylinder and drop it into the bin marked with a blue square.

```
# Stage 1: Move to blue cylinder
cyl_pos = get_keypoint("blue_cylinder")
move_to(cyl_pos[0], cyl_pos[1], cyl_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(cyl_pos[0], cyl_pos[1], cyl_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Move to bin
bin_pos = get_keypoint("blue_bin")
move_to(bin_pos[0], bin_pos[1], bin_pos[2] + 0.1, 0, 0, 0)
move_to(bin_pos[0], bin_pos[1], bin_pos[2], 0, 0, 0)
gripper_open()
```

3. **Stack Cube on Cube:** Pick the green cube and stack it on top of the yellow cube.

```
# Stage 1: Move to green cube
green_pos = get_keypoint("green_cube")
move_to(green_pos[0], green_pos[1], green_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(green_pos[0], green_pos[1], green_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Move above yellow cube
yellow_pos = get_keypoint("yellow_cube")
move_to(yellow_pos[0], yellow_pos[1], yellow_pos[2] + 0.2, 0, 0, 0)
move_to(yellow_pos[0], yellow_pos[1], yellow_pos[2] + 0.05, 0, 0, 0)
gripper_open()
```

4. **Move the Soda Can:** Pick the soda can from the left table and place it on the right table.

```
# Stage 1: Move to soda can
can_pos = get_keypoint("soda_can")
move_to(can_pos[0], can_pos[1], can_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(can_pos[0], can_pos[1], can_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Move to right table
right_table_pos = get_keypoint("right_table")
move_to(right_table_pos[0], right_table_pos[1], right_table_pos[2] + 0.1, 0, 0, 0)
move_to(right_table_pos[0], right_table_pos[1], right_table_pos[2], 0, 0, 0)
gripper_open()
```

5. **Fill the Tray:** Pick the AA battery and place it into the empty slot in the plastic tray.

```
# Stage 1: Move to battery
battery_pos = get_keypoint("AA_battery")
```

```
move_to(battery_pos[0], battery_pos[1], battery_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(battery_pos[0], battery_pos[1], battery_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Move to tray slot
slot_pos = get_keypoint("tray_slot")
move_to(slot_pos[0], slot_pos[1], slot_pos[2] + 0.1, 0, 0, 0)
move_to(slot_pos[0], slot_pos[1], slot_pos[2], 0, 0, 0)
gripper_open()
```

6. **Insert the USB Drive:** Pick the USB drive from the table and insert it into the laptop's USB port.

```
# Stage 1: Move to USB drive
usb_pos = get_keypoint("USB_drive")
move_to(usb_pos[0], usb_pos[1], usb_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(usb_pos[0], usb_pos[1], usb_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Move to USB port
port_pos = get_keypoint("laptop_USB_port")
move_to(port_pos[0], port_pos[1], port_pos[2] + 0.05, 0, 0, 0)
move_to(port_pos[0], port_pos[1], port_pos[2], 0, 0, 0)
gripper_open()
```

7. **Assemble the LEGO:** Pick the 2x4 LEGO brick and attach it to the red baseplate, connecting it to two other bricks.

```
# Stage 1: Move to Lego brick
lego_pos = get_keypoint("2x4_Lego_brick")
move_to(lego_pos[0], lego_pos[1], lego_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(lego_pos[0], lego_pos[1], lego_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Move to baseplate connection point
baseplate_pos = get_keypoint("red_baseplate_connection")
move_to(baseplate_pos[0], baseplate_pos[1], baseplate_pos[2] + 0.05, 0, 0, 0)
move_to(baseplate_pos[0], baseplate_pos[1], baseplate_pos[2], 0, 0, 0)
gripper_open()
```

8. **Place the Ring:** Pick the wooden ring and place it onto the vertical post.

```
# Stage 1: Move to ring
ring_pos = get_keypoint("wooden_ring")
move_to(ring_pos[0], ring_pos[1], ring_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(ring_pos[0], ring_pos[1], ring_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Move to post
post_pos = get_keypoint("vertical_post")
move_to(post_pos[0], post_pos[1], post_pos[2] + 0.1, 0, 0, 0)
move_to(post_pos[0], post_pos[1], post_pos[2], 0, 0, 0)
gripper_open()
```

9. **Put the Lid on the Jar:** Pick the plastic jar lid and place it on top of the jar.

```
# Stage 1: Move to lid
lid_pos = get_keypoint("jar_lid")
move_to(lid_pos[0], lid_pos[1], lid_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(lid_pos[0], lid_pos[1], lid_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Move to jar top
jar_pos = get_keypoint("jar_top")
move_to(jar_pos[0], jar_pos[1], jar_pos[2] + 0.05, 0, 0, 0)
move_to(jar_pos[0], jar_pos[1], jar_pos[2], 0, 0, 0)
gripper_open()
```

10. **Hang the Key:** Pick the key and hang it on the keyhook by its hole.

```
# Stage 1: Move to key
key_pos = get_keypoint("key")
move_to(key_pos[0], key_pos[1], key_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(key_pos[0], key_pos[1], key_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Move to keyhook
hook_pos = get_keypoint("keyhook")
move_to(hook_pos[0], hook_pos[1], hook_pos[2] + 0.05, 0, 0, 0)
move_to(hook_pos[0], hook_pos[1], hook_pos[2], 0, 0, 0)
gripper_open()
```

11. **Push the Dice:** Push the white dice across the table until it crosses the black line.

```
# Stage 1: Move gripper to push position
dice_pos = get_keypoint("white_dice")
line_pos = get_keypoint("black_line")
push_x = dice_pos[0] - 0.05  # slightly behind dice
move_to(push_x, dice_pos[1], dice_pos[2] + 0.05, 0, 0, 0)

# Stage 2: Push forward to cross line
move_to(line_pos[0] + 0.05, dice_pos[1], dice_pos[2] + 0.05, 0, 0, 0)
```

12. **Flip the Pancake:** Use the spatula to flip the pancake in the frying pan.

```
# Stage 1: Move spatula under pancake
pancake_pos = get_keypoint("pancake")
spatula_pos = get_keypoint("spatula")
move_to(spatula_pos[0], spatula_pos[1], spatula_pos[2] + 0.05, 0, 0, 0)
move_to(pancake_pos[0], pancake_pos[1], pancake_pos[2] - 0.01, 0, 0, 0)

# Stage 2: Flip motion (simulated as upward then rotate)
move_to(pancake_pos[0], pancake_pos[1], pancake_pos[2] + 0.1, 0, 0, 0)
move_to(pancake_pos[0], pancake_pos[1], pancake_pos[2] - 0.01, 0, 0, 0)
```

13. **Close the Drawer:** Push the kitchen drawer closed using the flat of the gripper.

```
# Stage 1: Move to drawer front
drawer_pos = get_keypoint("kitchen_drawer_front")
move_to(drawer_pos[0], drawer_pos[1], drawer_pos[2] + 0.05, 0, 0, 0)

# Stage 2: Push drawer closed
closed_pos = get_keypoint("drawer_closed_position")
move_to(closed_pos[0], closed_pos[1], closed_pos[2] + 0.05, 0, 0, 0)
```

14. **Press the Doorbell:** Press the round, lit doorbell button on the wall.

```
# Stage 1: Move to doorbell
doorbell_pos = get_keypoint("doorbell_button")
move_to(doorbell_pos[0], doorbell_pos[1], doorbell_pos[2] + 0.05, 0, 0, 0)

# Stage 2: Press button
move_to(doorbell_pos[0], doorbell_pos[1], doorbell_pos[2], 0, 0, 0)
move_to(doorbell_pos[0], doorbell_pos[1], doorbell_pos[2] + 0.05, 0, 0, 0)
```

15. **Align the Block:** Push the wooden block until it is flush against the corner of the table.

```
# Stage 1: Move to block side
block_pos = get_keypoint("wooden_block")
corner_pos = get_keypoint("table_corner")
push_pos = [block_pos[0] - 0.05, block_pos[1], block_pos[2] + 0.05]
move_to(push_pos[0], push_pos[1], push_pos[2], 0, 0, 0)

# Stage 2: Push to corner
move_to(corner_pos[0], corner_pos[1], corner_pos[2] + 0.05, 0, 0, 0)
```

16. **Scoop the Rice:** Use the metal spoon to scoop rice from the pot into the bowl.

```
# Stage 1: Move spoon to pot
spoon_pos = get_keypoint("metal_spoon")
pot_pos = get_keypoint("rice_pot")
move_to(pot_pos[0], pot_pos[1], pot_pos[2] + 0.1, 0, 0, 0)
move_to(pot_pos[0], pot_pos[1], pot_pos[2], 0, 0, 0)

# Stage 2: Scoop and move to bowl
bowl_pos = get_keypoint("bowl")
move_to(bowl_pos[0], bowl_pos[1], bowl_pos[2] + 0.1, 0, 0, 0)
move_to(bowl_pos[0], bowl_pos[1], bowl_pos[2], 0, 0, 0)
```

17. **Stir the Soup:** Use the spoon to stir the liquid in the pot three times clockwise.

```
# Stage 1: Move spoon to center of pot
spoon_pos = get_keypoint("spoon")
pot_center = get_keypoint("soup_pot_center")
move_to(pot_center[0], pot_center[1], pot_center[2] + 0.05, 0, 0, 0)
move_to(pot_center[0], pot_center[1], pot_center[2], 0, 0, 0)

# Stage 2: Simulate stirring (three circular motions)
for _ in range(3):
    move_to(pot_center[0] + 0.05, pot_center[1], pot_center[2], 0, 0, 0)
    move_to(pot_center[0], pot_center[1] + 0.05, pot_center[2], 0, 0, 0)
    move_to(pot_center[0] - 0.05, pot_center[1], pot_center[2], 0, 0, 0)
    move_to(pot_center[0], pot_center[1] - 0.05, pot_center[2], 0, 0, 0)
```

18. **Hammer the Nail:** Use the toy hammer to tap the nail until its head is flush with the board.

```
# Stage 1: Move hammer to nail
hammer_pos = get_keypoint("toy_hammer")
nail_pos = get_keypoint("nail")
move_to(nail_pos[0], nail_pos[1], nail_pos[2] + 0.1, 0, 0, 0)

# Stage 2: Tap motion (simulated as up-down)
for _ in range(3):  # simulate 3 taps
    move_to(nail_pos[0], nail_pos[1], nail_pos[2] + 0.05, 0, 0, 0)
    move_to(nail_pos[0], nail_pos[1], nail_pos[2], 0, 0, 0)
```

19. **Screw in the Lightbulb:** Pick the lightbulb and screw it into the empty lamp socket.

```
# Stage 1: Move to lightbulb
bulb_pos = get_keypoint("lightbulb")
move_to(bulb_pos[0], bulb_pos[1], bulb_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(bulb_pos[0], bulb_pos[1], bulb_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Move to socket and screw
socket_pos = get_keypoint("lamp_socket")
move_to(socket_pos[0], socket_pos[1], socket_pos[2] + 0.05, 0, 0, 0)
move_to(socket_pos[0], socket_pos[1], socket_pos[2], 0, 0, 0)
gripper_open()
```

20. **Pour the Water:** Pick the pitcher and pour water into the empty glass until it is half-full.

```
# Stage 1: Move to pitcher
pitcher_pos = get_keypoint("pitcher")
move_to(pitcher_pos[0], pitcher_pos[1], pitcher_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(pitcher_pos[0], pitcher_pos[1], pitcher_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Pour into glass
glass_pos = get_keypoint("empty_glass")
move_to(glass_pos[0], glass_pos[1], glass_pos[2] + 0.1, 0, 0, 0)
move_to(glass_pos[0], glass_pos[1], glass_pos[2] + 0.05, 0, 0, 0)
gripper_open()
```

21. **Uncoil the Rope:** Manipulate the coiled rope until it forms a straight line from start to end.

```
# Stage 1: Move to rope coil
rope_start = get_keypoint("rope_start")
rope_end = get_keypoint("rope_end")
move_to(rope_start[0], rope_start[1], rope_start[2] + 0.05, 0, 0, 0)

# Stage 2: Pull rope straight
move_to(rope_end[0], rope_end[1], rope_end[2] + 0.05, 0, 0, 0)
```

22. **Fold the Washcloth:** Fold the small, square washcloth in half.

```
# Stage 1: Move to washcloth center
cloth_pos = get_keypoint("washcloth_center")
move_to(cloth_pos[0], cloth_pos[1], cloth_pos[2] + 0.05, 0, 0, 0)

# Stage 2: Simulate folding (move one edge to meet opposite edge)
edge1 = get_keypoint("washcloth_edge1")
edge2 = get_keypoint("washcloth_edge2")
move_to(edge1[0], edge1[1], edge1[2] + 0.05, 0, 0, 0)
move_to(edge2[0], edge2[1], edge2[2] + 0.05, 0, 0, 0)
```

23. **Open the Bag:** Use two grippers to pull the handles of the plastic bag apart.

```
# Stage 1: Move left gripper to left handle
left_handle = get_keypoint("bag_left_handle")
move_to(left_handle[0], left_handle[1], left_handle[2] + 0.05, 0, 0, 0)

# Stage 2: Move right gripper to right handle
right_handle = get_keypoint("bag_right_handle")
move_to(right_handle[0], right_handle[1], right_handle[2] + 0.05, 0, 0, 0)

# Stage 3: Pull apart (simulated by moving handles outward)
move_to(left_handle[0] - 0.1, left_handle[1], left_handle[2] + 0.05, 0, 0, 0)
move_to(right_handle[0] + 0.1, right_handle[1], right_handle[2] + 0.05, 0, 0, 0)
```

24. **Drape the Towel:** Drape the hand towel over the horizontal bar.

```
# Stage 1: Move to towel center
towel_pos = get_keypoint("hand_towel_center")
move_to(towel_pos[0], towel_pos[1], towel_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(towel_pos[0], towel_pos[1], towel_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Move to bar and drape
bar_pos = get_keypoint("horizontal_bar")
move_to(bar_pos[0], bar_pos[1], bar_pos[2] + 0.05, 0, 0, 0)
move_to(bar_pos[0], bar_pos[1], bar_pos[2], 0, 0, 0)
gripper_open()
```

25. **Route the Cable:** Route the USB cable around the two posts in an S-shape.

```
# Stage 1: Move to cable start
cable_start = get_keypoint("USB_cable_start")
move_to(cable_start[0], cable_start[1], cable_start[2] + 0.05, 0, 0, 0)

# Stage 2: Route around first post
post1 = get_keypoint("first_post")
move_to(post1[0], post1[1], post1[2] + 0.05, 0, 0, 0)

# Stage 3: Route around second post
post2 = get_keypoint("second_post")
move_to(post2[0], post2[1], post2[2] + 0.05, 0, 0, 0)

# Stage 4: End at final position
cable_end = get_keypoint("USB_cable_end")
move_to(cable_end[0], cable_end[1], cable_end[2] + 0.05, 0, 0, 0)
```

26. **Grasp the Marble:** Pick the glass marble from a flat surface.

```
# Stage 1: Move to marble
marble_pos = get_keypoint("glass_marble")
move_to(marble_pos[0], marble_pos[1], marble_pos[2] + 0.05, 0, 0, 0)
gripper_open()
move_to(marble_pos[0], marble_pos[1], marble_pos[2], 0, 0, 0)
gripper_close()
```

27. **Grasp the Coin:** Pick the single coin from the table.

```
# Stage 1: Move to coin
coin_pos = get_keypoint("single_coin")
move_to(coin_pos[0], coin_pos[1], coin_pos[2] + 0.05, 0, 0, 0)
gripper_open()
move_to(coin_pos[0], coin_pos[1], coin_pos[2], 0, 0, 0)
gripper_close()
```

28. **Re-grip the Screwdriver:** Pick the screwdriver by its handle, then place it down and re-grip it by its shaft.

```
# Stage 1: Grip by handle
handle_pos = get_keypoint("screwdriver_handle")
move_to(handle_pos[0], handle_pos[1], handle_pos[2] + 0.05, 0, 0, 0)
gripper_open()
move_to(handle_pos[0], handle_pos[1], handle_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Place down
place_pos = get_keypoint("screwdriver_placement")
move_to(place_pos[0], place_pos[1], place_pos[2] + 0.05, 0, 0, 0)
gripper_open()

# Stage 3: Re-grip by shaft
shaft_pos = get_keypoint("screwdriver_shaft")
move_to(shaft_pos[0], shaft_pos[1], shaft_pos[2] + 0.05, 0, 0, 0)
gripper_open()
move_to(shaft_pos[0], shaft_pos[1], shaft_pos[2], 0, 0, 0)
gripper_close()
```

29. **Pick the Book:** Pick the paperback book from the shelf by its spine.

```
# Stage 1: Move to book spine
book_spine = get_keypoint("book_spine")
move_to(book_spine[0], book_spine[1], book_spine[2] + 0.05, 0, 0, 0)
gripper_open()
move_to(book_spine[0], book_spine[1], book_spine[2], 0, 0, 0)
gripper_close()
```

30. **Hook the Mug:** Hook a gripper finger through the handle of the coffee mug and lift it.

```
# Stage 1: Move to mug handle
mug_handle = get_keypoint("mug_handle")
move_to(mug_handle[0], mug_handle[1], mug_handle[2] + 0.05, 0, 0, 0)

# Stage 2: Hook and lift
move_to(mug_handle[0], mug_handle[1], mug_handle[2], 0, 0, 0)
gripper_close()
move_to(mug_handle[0], mug_handle[1], mug_handle[2] + 0.1, 0, 0, 0)
```

31. **Place the T-Block:** Pick the T-shaped block and insert it into the matching T-shaped slot on the board.

```
# Stage 1: Move to T-block
tblock_pos = get_keypoint("T_block")
move_to(tblock_pos[0], tblock_pos[1], tblock_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(tblock_pos[0], tblock_pos[1], tblock_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Move to T-slot
tslot_pos = get_keypoint("T_slot")
move_to(tslot_pos[0], tslot_pos[1], tslot_pos[2] + 0.05, 0, 0, 0)
move_to(tslot_pos[0], tslot_pos[1], tslot_pos[2], 0, 0, 0)
gripper_open()
```

32. **Assemble the Stack:** Pick the large square block and place it on the table, then place the medium block on it, and finally the small block on top.

```
# Stage 1: Place large block
large_pos = get_keypoint("large_square_block")
table_pos = get_keypoint("table_surface")
move_to(large_pos[0], large_pos[1], large_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(large_pos[0], large_pos[1], large_pos[2], 0, 0, 0)
gripper_close()
move_to(table_pos[0], table_pos[1], table_pos[2] + 0.1, 0, 0, 0)
move_to(table_pos[0], table_pos[1], table_pos[2], 0, 0, 0)
gripper_open()

# Stage 2: Place medium block
medium_pos = get_keypoint("medium_square_block")
move_to(medium_pos[0], medium_pos[1], medium_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(medium_pos[0], medium_pos[1], medium_pos[2], 0, 0, 0)
gripper_close()
move_to(table_pos[0], table_pos[1], table_pos[2] + 0.1, 0, 0, 0)
move_to(table_pos[0], table_pos[1], table_pos[2] + 0.05, 0, 0, 0)
gripper_open()

# Stage 3: Place small block
small_pos = get_keypoint("small_square_block")
move_to(small_pos[0], small_pos[1], small_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(small_pos[0], small_pos[1], small_pos[2], 0, 0, 0)
gripper_close()
move_to(table_pos[0], table_pos[1], table_pos[2] + 0.1, 0, 0, 0)
move_to(table_pos[0], table_pos[1], table_pos[2] + 0.1, 0, 0, 0)
gripper_open()
```

33. **Plug in the Lamp:** Pick the power plug from the floor and insert it into the wall outlet.

```
# Stage 1: Move to plug
plug_pos = get_keypoint("power_plug")
move_to(plug_pos[0], plug_pos[1], plug_pos[2] + 0.1, 0, 0, 0)
gripper_open()
move_to(plug_pos[0], plug_pos[1], plug_pos[2], 0, 0, 0)
gripper_close()

# Stage 2: Move to outlet
outlet_pos = get_keypoint("wall_outlet")
move_to(outlet_pos[0], outlet_pos[1], outlet_pos[2] + 0.05, 0, 0, 0)
move_to(outlet_pos[0], outlet_pos[1], outlet_pos[2], 0, 0, 0)
gripper_open()
```

Evaluation generated by DeepSeek:

1. **Sort the Red Cube:**

```
# Analysis: Correct and sufficient. Properly sequences approach, grasp, lift, transport, and
    placement using available functions.
```

2. **Bin the Blue Cylinder:**

```
# Analysis: Correct and sufficient. Same pattern as task 1, properly handles cylindrical object.
```

3. **Stack Cube on Cube:**

```
# Analysis: Correct and sufficient. Accounts for stacking height by placing green cube slightly
    above yellow cube.
```

4. **Move the Soda Can:**

```
# Analysis: Correct and sufficient. Clear transport between two defined locations.
```

5. **Fill the Tray:**

```
# Analysis: Correct and sufficient. Proper battery insertion into specific slot.
```

6. **Insert the USB Drive:**

```
# Analysis: Partially correct but insufficient. Missing alignment with USB port orientation and
    insertion force/motion.
```

7. **Assemble the LEGO:**

```
# Analysis: Partially correct but insufficient. Missing precise alignment for Lego stud connection
    and pressing motion.
```

8. **Place the Ring:**

```
# Analysis: Correct and sufficient. Simple placement task well handled.
```

9. **Put the Lid on the Jar:**

```
# Analysis: Correct and sufficient. Proper lid placement.
```

10. **Hang the Key:**

```
# Analysis: Correct and sufficient. Appropriate for hook-based hanging.
```

11. **Push the Dice:**

```
# Analysis: Correct and sufficient. Good use of relative positioning for pushing.
```

12. **Flip the Pancake:**

```
# Analysis: Incorrect and insufficient. Missing spatula pickup, proper flipping motion, and
    orientation changes.
```

13. **Close the Drawer:**

```
# Analysis: Correct and sufficient. Proper pushing motion to close drawer.
```

14. **Press the Doorbell:**

```
# Analysis: Correct and sufficient. Proper press and release motion.
```

15. **Align the Block:**

```
# Analysis: Correct and sufficient. Good corner alignment strategy.
```

16. **Scoop the Rice:**

```
# Analysis: Incorrect and insufficient. Missing spoon pickup, proper scooping motion, and pouring
    transfer.
```

17. **Stir the Soup:**

```
# Analysis: Partially correct but insufficient. Missing spoon pickup and proper circular stirring
    trajectory.
```

18. **Hammer the Nail:**

```
# Analysis: Incorrect and insufficient. Missing hammer pickup and proper hammering motion with wrist
     action.
```

19. **Screw in the Lightbulb:**

```
# Analysis: Incorrect and insufficient. Missing rotational motion for screwing and proper alignment.
```

20. **Pour the Water:**

```
# Analysis: Partially correct but insufficient. Missing pitcher orientation change for pouring and
    controlled flow.
```

21. **Uncoil the Rope:**

```
# Analysis: Incorrect and insufficient. Single pull motion insufficient for uncoiling; needs
    multiple grasp points.
```

22. **Fold the Washcloth:**

```
# Analysis: Incorrect and insufficient. Missing grasp points, folding motion, and proper
    manipulation sequence.
```

23. **Open the Bag:**

```
# Analysis: Incorrect and insufficient. Cannot handle dual grippers with current function set;
    missing handle grasping.
```

24. **Drape the Towel:**

```
# Analysis: Correct and sufficient. Proper draping motion over bar.
```

25. **Route the Cable:**

```
# Analysis: Partially correct but insufficient. Missing cable grasping and proper S-shaped routing
    with tension management.
```

26. **Grasp the Marble:**

```
# Analysis: Correct and sufficient. Proper small object grasping.
```

27. **Grasp the Coin:**

```
# Analysis: Correct and sufficient. Proper flat object grasping.
```

28. **Re-grip the Screwdriver:**

```
# Analysis: Correct and sufficient. Clear regripping sequence with proper placement.
```

29. **Pick the Book:**

```
# Analysis: Correct and sufficient. Proper spine grasping for book extraction.
```

30. **Hook the Mug:**

```
# Analysis: Correct and sufficient. Good hooking motion through handle.
```

31. **Place the T-Block:**

```
# Analysis: Partially correct but insufficient. Missing precise orientation alignment for T-slot
    insertion.
```

32. **Assemble the Stack:**

```
# Analysis: Correct and sufficient. Clear sequential stacking with proper height adjustments.
```

33. **Plug in the Lamp:**

```
# Analysis: Partially correct but insufficient. Missing plug orientation alignment and insertion
    force.
```

J.2. Instruct2Act

Prompt:

```
For each of the task, decompose it into stages and write me the Python intermediate code for each
stage. Each code consists of Python basic arithmetic and choose extra function ONLY from the
vocabulary: [get_gripper_pose, get_gripper_xyz, gripper_close, gripper_open, find, pick, place,
pick_place, insert, push, press, move_above, move_perpendicular, move_parallel, flip, scoop, stir,
tap, screw_rotation, controlled_pour, route_around, pull_apart, fold, straighten]. get_gripper_pose
() returns the XYZ Euler angle of the gripper, get_gripper_pos() return the xyz coordinate of the
gripper. find(obj_part:str) return points of the object part. pick(obj_part:str) pick the object
part. place(obj_part:str) place the object part. place_place(obj_part:str) pick and place the
object part. insert(obj_part1:str, obj_part2:str) insert the object part 1 to object part 2 after
grasped. push(object_part:str) push the object part. press(object_part:str) press the object part.
move_above(object_part1:str,object_part2:str, offset) moves the object_part1 above the object_part2
 by offset(unit: meter).
```

```
    move_parallel(object_part1:str, object_part2:str) moves the object part1 to a pose that's parallel
    to object part 2. move_perpendicular(object_part1:str, object_part2:str) moves the object part1 to
    a pose that's perpendicular to object part 2. flip(object_part:str) flip the object part. scoop(
    object:str, object_part:str) scoop object_part with object. stir(object:str, object_part:str) stir
    object_part with object. tap(object_part:str) tap object_part, screw_rotation(object:str,
    object_part:str) screw object_part with object.controlled_pour(object1:str, object2:str) pour the
    object1 onto object2 after aligned. route_around(object_part1:str, object_part2:str) route
    object_part1 to object_part2. pull_apart(object_part1:str, object_part2:str) pull object_part1 away
     from object_part2. fold(object_part:str, object:str) fold object_part on the object to the object
    itself. straighten(object_part:str, object:str) straighten the object by pulling object_part. Your
    output should be in Latex with enumerate and lstlisting.
```

Intermediate Language Generated by Qwen3-VL:

1. **Sort the Red Cube:** Pick the red cube from a group and place it inside the red circle

```
red_cube = find("red cube")
pick(red_cube)
move_above(red_cube, "red circle", 0.05)
place(red_cube)
```

2. **Bin the Blue Cylinder:** Pick the blue cylinder and drop it into the bin marked with a blue square

```
blue_cylinder = find("blue cylinder")
pick(blue_cylinder)
move_above(blue_cylinder, "bin with blue square", 0.05)
place(blue_cylinder)
```

3. **Stack Cube on Cube:** Pick the green cube and stack it on top of the yellow cube

```
green_cube = find("green cube")
yellow_cube = find("yellow cube")
pick(green_cube)
move_above(green_cube, yellow_cube, 0.02)
place(green_cube)
```

4. **Move the Soda Can:** Pick the soda can from the left table and place it on the right table

```
soda_can = find("soda can")
pick(soda_can)
move_above(soda_can, "right table", 0.1)
place(soda_can)
```

5. **Fill the Tray:** Pick the AA battery and place it into the empty slot in the plastic tray

```
aa_battery = find("AA battery")
empty_slot = find("empty slot in tray")
pick(aa_battery)
move_above(aa_battery, empty_slot, 0.02)
place(aa_battery)
```

6. **Insert the USB Drive:** Pick the USB drive from the table and insert it into the laptop's USB port

```
usb_drive = find("USB drive")
usb_port = find("laptop USB port")
pick(usb_drive)
insert(usb_drive, usb_port)
```

7. **Assemble the LEGO:** Pick the 2x4 LEGO brick and attach it to the red baseplate, connecting it to two other bricks

```
lego_brick = find("2x4 Lego brick")
baseplate = find("red baseplate")
pick(lego_brick)
move_above(lego_brick, baseplate, 0.01)
insert(lego_brick, baseplate)
```

8. **Place the Ring:** Pick the wooden ring and place it onto the vertical post

```
wooden_ring = find("wooden ring")
vertical_post = find("vertical post")
pick(wooden_ring)
move_above(wooden_ring, vertical_post, 0.02)
place(wooden_ring)

```

9. **Put the Lid on the Jar:** Pick the plastic jar lid and place it on top of the jar

```
jar_lid = find("plastic jar lid")
jar = find("jar")
pick(jar_lid)
move_above(jar_lid, jar, 0.01)
place(jar_lid)

```

10. **Hang the Key:** Pick the key and hang it on the keyhook by its hole.

```
key = find("key")
keyhook = find("keyhook")
pick(key)
move_perpendicular(key, keyhook)
place(key)

```

11. **Push the Dice:** Push the white dice across the table until it crosses the black line

```
white_dice = find("white dice")
black_line = find("black line")
push(white_dice)

```

12. **Flip the Pancake:** Use the spatula to flip the pancake in the frying pan

```
spatula = find("spatula")
pancake = find("pancake")
flip(pancake)

```

13. **Close the Drawer:** Push the kitchen drawer closed using the flat of the gripper

```
kitchen_drawer = find("kitchen drawer")
push(kitchen_drawer)

```

14. **Press the Doorbell:** Press the round, lit doorbell button on the wall

```
doorbell_button = find("doorbell button")
press(doorbell_button)

```

15. **Align the Block:** Push the wooden block until it is flush against the corner of the table

```
wooden_block = find("wooden block")
table_corner = find("table corner")
push(wooden_block)

```

16. **Scoop the Rice:** Use the metal spoon to scoop rice from the pot into the bowl

```
metal_spoon = find("metal spoon")
rice = find("rice")
bowl = find("bowl")
scoop(metal_spoon, rice)
move_above(rice, bowl, 0.05)
place(rice)

```

17. **Stir the Soup:** Use the spoon to stir the liquid in the pot three times clockwise

```
spoon = find("spoon")
soup = find("soup")
stir(spoon, soup)
```

18. **Hammer the Nail:** Use the toy hammer to tap the nail until its head is flush with the board

```
toy_hammer = find("toy hammer")
nail = find("nail")
tap(nail)
```

19. **Screw in the Lightbulb:** Pick the lightbulb and screw it into the empty lamp socket

```
lightbulb = find("lightbulb")
lamp_socket = find("lamp socket")
pick(lightbulb)
screw_rotation(lightbulb, lamp_socket)
```

20. **Pour the Water:** Pick the pitcher and pour water into the empty glass until it is half-full

```
pitcher = find("pitcher")
empty_glass = find("empty glass")
controlled_pour(pitcher, empty_glass)
```

21. **Uncoil the Rope:** Manipulate the coiled rope until it forms a straight line from start to end

```
coiled_rope = find("coiled rope")
straighten(coiled_rope, "rope")
```

22. **Fold the Washcloth:** Fold the small, square washcloth in half

```
washcloth = find("washcloth")
fold(washcloth, "washcloth")
```

23. **Open the Bag:** Use two grippers to pull the handles of the plastic bag apart

```
bag_handle_left = find("left bag handle")
bag_handle_right = find("right bag handle")
pull_apart(bag_handle_left, bag_handle_right)
```

24. **Drape the Towel:** Drape the hand towel over the horizontal bar

```
hand_towel = find("hand towel")
horizontal_bar = find("horizontal bar")
move_above(hand_towel, horizontal_bar, 0.05)
place(hand_towel)
```

25. **Route the Cable:** Route the USB cable around the two posts in an S-shape

```
usb_cable = find("USB cable")
post1 = find("first post")
post2 = find("second post")
route_around(usb_cable, post1)
route_around(usb_cable, post2)
```

26. **Grasp the Marble:** Pick the glass marble from a flat surface

```
glass_marble = find("glass marble")
pick(glass_marble)
```

27. **Grasp the Coin:** Pick the single coin from the table

```
coin = find("coin")
pick(coin)
```

28. **Re-grip the Screwdriver:** Pick the screwdriver by its handle, then place it down and re-grip it by its shaft

```
screwdriver_handle = find("screwdriver handle")
screwdriver_shaft = find("screwdriver shaft")
pick(screwdriver_handle)
place(screwdriver_handle)
pick(screwdriver_shaft)

```

29. **Pick the Book:** Pick the paperback book from the shelf by its spine

```
book_spine = find("book spine")
pick(book_spine)

```

30. **Hook the Mug:** Hook a gripper finger through the handle of the coffee mug and lift it

```
mug_handle = find("mug handle")
pick(mug_handle)

```

31. **Place the T-Block:** Pick the T-shaped block and insert it into the matching T-shaped slot on the board

```
t_block = find("T-shaped block")
t_slot = find("T-shaped slot")
pick(t_block)
insert(t_block, t_slot)

```

32. **Assemble the Stack:** Pick the large square block and place it on the table, then place the medium block on it, and finally the small block on top

```
large_block = find("large square block")
medium_block = find("medium block")
small_block = find("small block")
pick(large_block)
move_above(large_block, "table", 0.02)
place(large_block)
pick(medium_block)
move_above(medium_block, large_block, 0.02)
place(medium_block)
pick(small_block)
move_above(small_block, medium_block, 0.02)
place(small_block)

```

33. **Plug in the Lamp:** Pick the power plug from the floor and insert it into the wall outlet

```
power_plug = find("power plug")
wall_outlet = find("wall outlet")
pick(power_plug)
insert(power_plug, wall_outlet)

```

Evaluation by DeepSeek:

1. **Sort the Red Cube**

```
red_cube = find("red cube")
pick(red_cube)
move_above(red_cube, "red circle", 0.05)
place(red_cube)
```

Judgment: Correct and sufficient. Uses proper sequence: find, pick, move above target, place.

2. **Bin the Blue Cylinder**

```
blue_cylinder = find("blue cylinder")
pick(blue_cylinder)
move_above(blue_cylinder, "bin with blue square", 0.05)
place(blue_cylinder)
```

Judgment: Correct and sufficient. Same pattern as Task 1, appropriate for bin placement.

3. **Stack Cube on Cube**

```
green_cube = find("green cube")
yellow_cube = find("yellow cube")
pick(green_cube)
move_above(green_cube, yellow_cube, 0.02)
place(green_cube)
```

Judgment: Correct and sufficient. Properly specifies both cubes and uses smaller offset for stacking.

4. **Move the Soda Can**

```
soda_can = find("soda can")
pick(soda_can)
move_above(soda_can, "right table", 0.1)
place(soda_can)
```

Judgment: Correct and sufficient. Appropriate for table-to-table transfer.

5. **Fill the Tray**

```
aa_battery = find("AA battery")
empty_slot = find("empty slot in tray")
pick(aa_battery)
move_above(aa_battery, empty_slot, 0.02)
place(aa_battery)
```

Judgment: Correct and sufficient. Specifically finds the empty slot for precise placement.

6. **Insert the USB Drive**

```
usb_drive = find("USB drive")
usb_port = find("laptop USB port")
pick(usb_drive)
insert(usb_drive, usb_port)
```

Judgment: Correct and sufficient. Properly uses the insert function for this insertion task.

7. **Assemble the LEGO**

```
lego_brick = find("2x4 Lego brick")
baseplate = find("red baseplate")
pick(lego_brick)
move_above(lego_brick, baseplate, 0.01)
insert(lego_brick, baseplate)
```

Judgment: Correct and sufficient. Uses insert for Lego assembly with appropriate small offset.

8. **Place the Ring**

```
wooden_ring = find("wooden ring")
vertical_post = find("vertical post")
pick(wooden_ring)
move_above(wooden_ring, vertical_post, 0.02)
place(wooden_ring)
```

Judgment: Correct and sufficient. Standard pick-and-place for ring on post.

9. **Put the Lid on the Jar**

```
jar_lid = find("plastic jar lid")
jar = find("jar")
pick(jar_lid)
move_above(jar_lid, jar, 0.01)
place(jar_lid)
```

Judgment: Correct and sufficient. Appropriate for lid placement.

10. **Hang the Key**

```
key = find("key")
keyhook = find("keyhook")
pick(key)
move_perpendicular(key, keyhook)
place(key)
```

Judgment: Correct and sufficient. Properly uses move_perpendicular for hooking orientation.

11. **Push the Dice**

```
white_dice = find("white dice")
black_line = find("black line")
push(white_dice)
```

Judgment: Insufficient. Missing target specification for push. Should be: push(white_dice, black_line)

12. **Flip the Pancake**

```
spatula = find("spatula")
pancake = find("pancake")
flip(pancake)
```

Judgment: Correct and sufficient. Properly uses flip function with pancake as target.

13. **Close the Drawer**

```
kitchen_drawer = find("kitchen drawer")
push(kitchen_drawer)
```

Judgment: Insufficient. Missing target/direction for push. Should specify closed position.

14. **Press the Doorbell**

```
doorbell_button = find("doorbell button")
press(doorbell_button)
```

Judgment: Correct and sufficient. Direct use of press function.

15. **Align the Block**

```
wooden_block = find("wooden block")
table_corner = find("table corner")
push(wooden_block)
```

Judgment: Insufficient. Missing target for push. Should be: push(wooden_block, table_corner)

16. **Scoop the Rice**

```
metal_spoon = find("metal spoon")
rice = find("rice")
bowl = find("bowl")
scoop(metal_spoon, rice)
move_above(rice, bowl, 0.05)
place(rice)
```

Judgment: Incorrect. Cannot place ”rice” after scooping - rice is not a graspable object. Should use scoop directly to bowl.

17. **Stir the Soup**

```
spoon = find("spoon")
soup = find("soup")
stir(spoon, soup)
```

Judgment: Correct and sufficient. Direct use of stir function.

18. **Hammer the Nail**

```
toy_hammer = find("toy hammer")
nail = find("nail")
tap(nail)
```

Judgment: Insufficient. Missing tool specification. Should be: tap(nail) with hammer grasped, or use hammer in tap function.

19. **Screw in the Lightbulb**

```
lightbulb = find("lightbulb")
lamp_socket = find("lamp socket")
pick(lightbulb)
screw_rotation(lightbulb, lamp_socket)
```

Judgment: Correct and sufficient. Proper use of screw_rotation after picking.

20. **Pour the Water**

```
pitcher = find("pitcher")
empty_glass = find("empty glass")
controlled_pour(pitcher, empty_glass)
```

Judgment: Correct and sufficient. Direct use of controlled_pour.

21. **Uncoil the Rope**

```
coiled_rope = find("coiled rope")
straighten(coiled_rope, "rope")
```

Judgment: Correct and sufficient. Proper use of straighten function.

22. **Fold the Washcloth**

```
washcloth = find("washcloth")
fold(washcloth, "washcloth")
```

Judgment: Correct and sufficient. Direct use of fold function.

23. **Open the Bag**

```
bag_handle_left = find("left bag handle")
bag_handle_right = find("right bag handle")
pull_apart(bag_handle_left, bag_handle_right)
```

Judgment: Correct and sufficient. Proper use of pull-l_apart with both handles.

24. **Drape the Towel**

```
hand_towel = find("hand towel")
horizontal_bar = find("horizontal bar")
move_above(hand_towel, horizontal_bar, 0.05)
place(hand_towel)
```

Judgment: Correct and sufficient. Standard placement above the bar.

25. **Route the Cable**

```
usb_cable = find("USB cable")
post1 = find("first post")
post2 = find("second post")
route_around(usb_cable, post1)
route_around(usb_cable, post2)
```

Judgment: Insufficient. route_around typically takes both obstacles or uses different syntax. Should be single call with both posts.

26. **Grasp the Marble**

```
glass_marble = find("glass marble")
pick(glass_marble)
```

Judgment: Correct and sufficient. Simple pick operation.

27. **Grasp the Coin**

```
coin = find("coin")
pick(coin)
```

Judgment: Correct and sufficient. Simple pick operation.

28. **Re-grip the Screwdriver**

```
screwdriver_handle = find("screwdriver handle")
screwdriver_shaft = find("screwdriver shaft")
pick(screwdriver_handle)
place(screwdriver_handle)
pick(screwdriver_shaft)
```

Judgment: Correct and sufficient. Proper sequence for re-gripping.

29. **Pick the Book**

```
book_spine = find("book spine")
pick(book_spine)
```

Judgment: Correct and sufficient. Specific grasp from spine.

30. **Hook the Mug**

```
mug_handle = find("mug handle")
pick(mug_handle)
```

Judgment: Correct and sufficient. Direct grasp of handle for hooking.

31. **Place the T-Block**

```
t_block = find("T-shaped block")
t_slot = find("T-shaped slot")
pick(t_block)
insert(t_block, t_slot)
```

Judgment: Correct and sufficient. Proper use of insert

32. **Assemble the Stack**

```
large_block = find("large square block")
medium_block = find("medium block")
small_block = find("small block")
pick(large_block)
move_above(large_block, "table", 0.02)
place(large_block)
pick(medium_block)
move_above(medium_block, large_block, 0.02)
place(medium_block)
pick(small_block)
move_above(small_block, medium_block, 0.02)
place(small_block)
```

Judgment: Correct and sufficient. Clear sequential stacking with proper offsets.

33. **Plug in the Lamp**

```
power_plug = find("power plug")
wall_outlet = find("wall outlet")
pick(power_plug)
insert(power_plug, wall_outlet)
```

Judgment: Correct and sufficient. Proper use of insert for plug.

J.3. ReKep

Prompt:

```
For each of the task, decompose it into stages and write me the Python intermediate code for each stage. Each code consists of Python basic arithmetic and choose extra function ONLY from the vocabulary: [gripper_close, gripper_open, get_gripper_pos, get_gripper_pose, find, move_perpendicular, move_parallel,
move_relative, move_linear, screw_rotation, cloth_ fold, controlled_pour]. get_gripper_pose() returns the XYZ Euler angle of the gripper, get_gripper_pos() return the xyz coordinate of the gripper. find(obj_part:str) return points of the object part.
move_parallel(obj_part1:str, obj_part2:str) move the object part1 to a pose parallel to the object part2. move_perpendicular(obj_part1:str, obj_part2:str) move the object part1 to a pose perpendicular to the object part2. move_relative(obj_part1:str, obj_part2:str, offset:list) moves object part 1 to object part2 with relative offset "offset". move_linear(obj_part:str, offset:list) moves object part 1 by offset. screw_rotation(obj_part:str, angle:int) rotates object part by angle. cloth_fold(obj_part1:str, obj:str) fold object part on the object to object itself. controlled_pour(object1:str, object2:str) rotates object1 to pour contents in object1 to object2. Your output should be in Latex with enumerate and lstlisting.
```

Generated Intermediate Language by Qwen3-VL

1. **Sort the Red Cube:** Pick the red cube from a group and place it inside the red circle

```
# Stage 1: Locate red cube
red_cube_pos = find("red_cube")

# Stage 2: Move gripper to red cube
move_linear("gripper", [red_cube_pos[0], red_cube_pos[1], red_cube_pos[2] + 0.1])
gripper_close("red_cube")

```

```
# Stage 3: Lift red cube
move_linear("gripper", [0, 0, 0.1])

# Stage 4: Move to red circle
red_circle_pos = find("red_circle")
move_relative("gripper", "red_circle", [0, 0, 0.1])

# Stage 5: Place red cube in circle
move_linear("gripper", [0, 0, -0.1])
gripper_open()
```

2. **Bin the Blue Cylinder:** Pick the blue cylinder and drop it into the bin marked with a blue square

```
# Stage 1: Locate blue cylinder
blue_cyl_pos = find("blue_cylinder")

# Stage 2: Approach and grasp
move_linear("gripper", [blue_cyl_pos[0], blue_cyl_pos[1], blue_cyl_pos[2] + 0.1])
gripper_close("blue_cylinder")

# Stage 3: Lift
move_linear("gripper", [0, 0, 0.1])

# Stage 4: Move to bin
bin_pos = find("blue_square_bin")
move_relative("gripper", "blue_square_bin", [0, 0, 0.1])

# Stage 5: Drop
move_linear("gripper", [0, 0, -0.1])
gripper_open()
```

3. **Stack Cube on Cube:** Pick the green cube and stack it on top of the yellow cube

```
# Stage 1: Locate green cube
green_cube_pos = find("green_cube")

# Stage 2: Grasp green cube
move_linear("gripper", [green_cube_pos[0], green_cube_pos[1], green_cube_pos[2] + 0.1])
gripper_close("green_cube")

# Stage 3: Lift
move_linear("gripper", [0, 0, 0.1])

# Stage 4: Move over yellow cube
yellow_cube_pos = find("yellow_cube")
move_relative("gripper", "yellow_cube", [0, 0, 0.1])

# Stage 5: Stack
move_linear("gripper", [0, 0, -0.1])
gripper_open()
```

4. **Move the Soda Can:** Pick the soda can from the left table and place it on the right table

```
# Stage 1: Locate soda can
soda_can_pos = find("soda_can_left_table")

# Stage 2: Grasp
move_linear("gripper", [soda_can_pos[0], soda_can_pos[1], soda_can_pos[2] + 0.1])
gripper_close("soda_can")

# Stage 3: Lift
move_linear("gripper", [0, 0, 0.1])

# Stage 4: Move to right table
right_table_pos = find("right_table")
move_relative("gripper", "right_table", [0, 0, 0.1])
```

```
# Stage 5: Place
move_linear("gripper", [0, 0, -0.1])
gripper_open()
```

5. **Fill the Tray:** Pick the AA battery and place it into the empty slot in the plastic tray

```
# Stage 1: Locate battery
battery_pos = find("aa_battery")

# Stage 2: Grasp battery
move_linear("gripper", [battery_pos[0], battery_pos[1], battery_pos[2] + 0.1])
gripper_close("aa_battery")

# Stage 3: Lift
move_linear("gripper", [0, 0, 0.1])

# Stage 4: Move to tray slot
tray_slot_pos = find("empty_tray_slot")
move_relative("gripper", "tray_slot", [0, 0, 0.1])

# Stage 5: Insert
move_linear("gripper", [0, 0, -0.1])
gripper_open()
```

6. **Insert the USB Drive:** Pick the USB drive from the table and insert it into the laptop's USB port

```
# Stage 1: Locate USB drive
usb_pos = find("usb_drive")

# Stage 2: Grasp
move_linear("gripper", [usb_pos[0], usb_pos[1], usb_pos[2] + 0.1])
gripper_close("usb_drive")

# Stage 3: Lift
move_linear("gripper", [0, 0, 0.1])

# Stage 4: Move to USB port
usb_port_pos = find("laptop_usb_port")
move_relative("gripper", "laptop_usb_port", [0, 0, 0.1])

# Stage 5: Insert
move_linear("gripper", [0, 0, -0.05])
gripper_open()
```

7. **Assemble the Lego:** Pick the 2x4 Lego brick and attach it to the red baseplate, connecting it to two other bricks

```
# Stage 1: Locate 2x4 brick
lego_brick_pos = find("2x4_lego_brick")

# Stage 2: Grasp
move_linear("gripper", [lego_brick_pos[0], lego_brick_pos[1], lego_brick_pos[2] + 0.1])
gripper_close("2x4_lego_brick")

# Stage 3: Lift
move_linear("gripper", [0, 0, 0.1])

# Stage 4: Move to baseplate
baseplate_pos = find("red_baseplate")
move_relative("gripper", "red_baseplate", [0, 0, 0.1])

# Stage 5: Attach
move_linear("gripper", [0, 0, -0.05])
gripper_open()
```

8. **Place the Ring:** Pick the wooden ring and place it onto the vertical post

```
# Stage 1: Locate ring
ring_pos = find("wooden_ring")

# Stage 2: Grasp
move_linear("gripper", [ring_pos[0], ring_pos[1], ring_pos[2] + 0.1])
gripper_close("wooden_ring")

# Stage 3: Lift
move_linear("gripper", [0, 0, 0.1])

# Stage 4: Move over post
post_pos = find("vertical_post")
move_relative("gripper", "vertical_post", [0, 0, 0.1])

# Stage 5: Place
move_linear("gripper", [0, 0, -0.1])
gripper_open()
```

9. **Put the Lid on the Jar:** Pick the plastic jar lid and place it on top of the jar

```
# Stage 1: Locate lid
lid_pos = find("plastic_lid")

# Stage 2: Grasp
move_linear("gripper", [lid_pos[0], lid_pos[1], lid_pos[2] + 0.1])
gripper_close("plastic_lid")

# Stage 3: Lift
move_linear("gripper", [0, 0, 0.1])

# Stage 4: Move over jar
jar_pos = find("jar_top")
move_relative("gripper", "jar_top", [0, 0, 0.1])

# Stage 5: Place
move_linear("gripper", [0, 0, -0.05])
gripper_open()
```

10. **Hang the Key:** Pick the key and hang it on the keyhook by its hole.

```
# Stage 1: Locate key
key_pos = find("key")

# Stage 2: Grasp
move_linear("gripper", [key_pos[0], key_pos[1], key_pos[2] + 0.1])
gripper_close("key")

# Stage 3: Lift
move_linear("gripper", [0, 0, 0.1])

# Stage 4: Move to keyhook
keyhook_pos = find("keyhook")
move_relative("gripper", "keyhook", [0, 0, 0.1])

# Stage 5: Hang
move_linear("gripper", [0, 0, -0.05])
gripper_open()
```

11. **Push the Dice:** Push the white dice across the table until it crosses the black line

```
# Stage 1: Locate dice
dice_pos = find("white_dice")

# Stage 2: Position gripper above dice
```

```
move_linear("gripper", [dice_pos[0], dice_pos[1], dice_pos[2] + 0.05])

# Stage 3: Push forward
black_line_pos = find("black_line")
push_distance = black_line_pos[0] - dice_pos[0] + 0.1
move_linear("gripper", [push_distance, 0, 0])
```

12. **Flip the Pancake:** Use the spatula to flip the pancake in the frying pan

```
# Stage 1: Locate spatula
spatula_pos = find("spatula")

# Stage 2: Grasp spatula
move_linear("gripper", [spatula_pos[0], spatula_pos[1], spatula_pos[2] + 0.1])
gripper_close("spatula")

# Stage 3: Move under pancake
pancake_pos = find("pancake")
move_relative("gripper", "pancake", [0, 0, 0.05])

# Stage 4: Flip motion (simulated as rotation)
screw_rotation("spatula", 180)
```

13. **Close the Drawer:** Push the kitchen drawer closed using the flat of the gripper

```
# Stage 1: Locate drawer front
drawer_front_pos = find("drawer_front")

# Stage 2: Position gripper to push
move_linear("gripper", [drawer_front_pos[0], drawer_front_pos[1], drawer_front_pos[2] + 0.05])

# Stage 3: Push drawer closed
move_linear("gripper", [-0.1, 0, 0])
```

14. **Press the Doorbell:** Press the round, lit doorbell button on the wall

```
# Stage 1: Locate doorbell
doorbell_pos = find("doorbell_button")

# Stage 2: Move gripper to button
move_linear("gripper", [doorbell_pos[0], doorbell_pos[1], doorbell_pos[2] + 0.05])

# Stage 3: Press
move_linear("gripper", [0, 0, -0.02])
move_linear("gripper", [0, 0, 0.02])  # Release
```

15. **Align the Block:** Push the wooden block until it is flush against the corner of the table

```
# Stage 1: Locate block
block_pos = find("wooden_block")

# Stage 2: Position gripper to push
move_linear("gripper", [block_pos[0], block_pos[1], block_pos[2] + 0.05])

# Stage 3: Push toward corner
corner_pos = find("table_corner")
push_x = corner_pos[0] - block_pos[0]
push_y = corner_pos[1] - block_pos[1]
move_linear("gripper", [push_x, push_y, 0])
```

16. **Scoop the Rice:** Use the metal spoon to scoop rice from the pot into the bowl

```
# Stage 1: Locate spoon
spoon_pos = find("metal_spoon")
```

```
# Stage 2: Grasp spoon
move_linear("gripper", [spoon_pos[0], spoon_pos[1], spoon_pos[2] + 0.1])
gripper_close("metal_spoon")

# Stage 3: Scoop from pot
pot_pos = find("rice_pot")
move_relative("gripper", "rice_pot", [0, 0, 0.05])

# Stage 4: Move to bowl
bowl_pos = find("bowl")
move_relative("gripper", "bowl", [0, 0, 0.05])

# Stage 5: Pour
controlled_pour("metal_spoon", "bowl")
```

17. **Stir the Soup:** Use the spoon to stir the liquid in the pot three times clockwise

```
# Stage 1: Locate spoon
spoon_pos = find("soup_spoon")

# Stage 2: Grasp spoon
move_linear("gripper", [spoon_pos[0], spoon_pos[1], spoon_pos[2] + 0.1])
gripper_close("soup_spoon")

# Stage 3: Move to pot
pot_pos = find("soup_pot")
move_relative("gripper", "soup_pot", [0, 0, 0.05])

# Stage 4: Stir 3 times clockwise
for _ in range(3):
    screw_rotation("soup_spoon", 360)
```

18. **Hammer the Nail:** Use the toy hammer to tap the nail until its head is flush with the board

```
# Stage 1: Locate hammer
hammer_pos = find("toy_hammer")

# Stage 2: Grasp hammer
move_linear("gripper", [hammer_pos[0], hammer_pos[1], hammer_pos[2] + 0.1])
gripper_close("toy_hammer")

# Stage 3: Move to nail
nail_pos = find("nail_head")
move_relative("gripper", "nail_head", [0, 0, 0.1])

# Stage 4: Tap (simulate with small downward motions)
for _ in range(5):
    move_linear("gripper", [0, 0, -0.02])
    move_linear("gripper", [0, 0, 0.02])
```

19. **Screw in the Lightbulb:** Pick the lightbulb and screw it into the empty lamp socket

```
# Stage 1: Locate lightbulb
bulb_pos = find("lightbulb")

# Stage 2: Grasp
move_linear("gripper", [bulb_pos[0], bulb_pos[1], bulb_pos[2] + 0.1])
gripper_close("lightbulb")

# Stage 3: Lift
move_linear("gripper", [0, 0, 0.1])

# Stage 4: Move to socket
socket_pos = find("lamp_socket")
move_relative("gripper", "lamp_socket", [0, 0, 0.1])
```

```
# Stage 5: Screw in
screw_rotation("lightbulb", 90)
gripper_open()
```

20. **Pour the Water:** Pick the pitcher and pour water into the empty glass until it is half-full

```
# Stage 1: Locate pitcher
pitcher_pos = find("water_pitcher")

# Stage 2: Grasp pitcher
move_linear("gripper", [pitcher_pos[0], pitcher_pos[1], pitcher_pos[2] + 0.1])
gripper_close("water_pitcher")

# Stage 3: Move to glass
glass_pos = find("empty_glass")
move_relative("gripper", "empty_glass", [0, 0, 0.1])

# Stage 4: Pour
controlled_pour("water_pitcher", "empty_glass")
```

21. **Uncoil the Rope:** Manipulate the coiled rope until it forms a straight line from start to end

```
# Stage 1: Locate rope ends
rope_start = find("rope_start")
rope_end = find("rope_end")

# Stage 2: Grasp start
move_linear("gripper", [rope_start[0], rope_start[1], rope_start[2] + 0.1])
gripper_close("rope_start")

# Stage 3: Pull to straighten
move_linear("gripper", [rope_end[0] - rope_start[0], rope_end[1] - rope_start[1], 0])
gripper_open()
```

22. **Fold the Washcloth:** Fold the small, square washcloth in half

```
# Stage 1: Locate washcloth
washcloth_pos = find("washcloth")

# Stage 2: Grasp one edge
move_linear("gripper", [washcloth_pos[0], washcloth_pos[1], washcloth_pos[2] + 0.1])
gripper_close("washcloth_edge")

# Stage 3: Fold
cloth_fold("washcloth_edge", "washcloth")
gripper_open()
```

23. **Open the Bag:** Use two grippers to pull the handles of the plastic bag apart

```
# Stage 1: Locate left handle
left_handle = find("bag_handle_left")

# Stage 2: Grasp left handle
move_linear("gripper_left", [left_handle[0], left_handle[1], left_handle[2] + 0.1])
gripper_close("bag_handle_left")

# Stage 3: Locate right handle
right_handle = find("bag_handle_right")

# Stage 4: Grasp right handle
move_linear("gripper_right", [right_handle[0], right_handle[1], right_handle[2] + 0.1])
gripper_close("bag_handle_right")

# Stage 5: Pull apart
move_linear("gripper_left", [-0.1, 0, 0])
```

```
move_linear("gripper_right", [0.1, 0, 0])
```

24. **Drape the Towel:** Drape the hand towel over the horizontal bar

```
# Stage 1: Locate towel
towel_pos = find("hand_towel")

# Stage 2: Grasp towel center
move_linear("gripper", [towel_pos[0], towel_pos[1], towel_pos[2] + 0.1])
gripper_close("hand_towel_center")

# Stage 3: Move to bar
bar_pos = find("horizontal_bar")
move_relative("gripper", "horizontal_bar", [0, 0, 0.1])

# Stage 4: Drape
move_linear("gripper", [0, 0, -0.05])
gripper_open()
```

25. **Route the Cable:** Route the USB cable around the two posts in an S-shape

```
# Stage 1: Locate cable
cable_pos = find("usb_cable")

# Stage 2: Grasp cable
move_linear("gripper", [cable_pos[0], cable_pos[1], cable_pos[2] + 0.1])
gripper_close("usb_cable")

# Stage 3: Route around first post
post1_pos = find("first_post")
move_relative("gripper", "first_post", [0, 0, 0.1])

# Stage 4: Route around second post
post2_pos = find("second_post")
move_relative("gripper", "second_post", [0, 0, 0.1])

# Stage 5: Release
gripper_open()
```

26. **Grasp the Marble:** Pick the glass marble from a flat surface

```
# Stage 1: Locate marble
marble_pos = find("glass_marble")

# Stage 2: Move gripper above marble
move_linear("gripper", [marble_pos[0], marble_pos[1], marble_pos[2] + 0.05])

# Stage 3: Grasp
gripper_close("glass_marble")

# Stage 4: Lift
move_linear("gripper", [0, 0, 0.1])
```

27. **Grasp the Coin:** Pick the single coin from the table

```
# Stage 1: Locate coin
coin_pos = find("single_coin")

# Stage 2: Move gripper above coin
move_linear("gripper", [coin_pos[0], coin_pos[1], coin_pos[2] + 0.05])

# Stage 3: Grasp
gripper_close("single_coin")

# Stage 4: Lift
```

```
move_linear("gripper", [0, 0, 0.1])

```

28. **Re-grip the Screwdriver:** Pick the screwdriver by its handle, then place it down and re-grip it by its shaft

```
# Stage 1: Locate handle
handle_pos = find("screwdriver_handle")

# Stage 2: Grasp handle
move_linear("gripper", [handle_pos[0], handle_pos[1], handle_pos[2] + 0.1])
gripper_close("screwdriver_handle")

# Stage 3: Place down
table_pos = find("table_surface")
move_relative("gripper", "table_surface", [0, 0, 0.1])
gripper_open()

# Stage 4: Locate shaft
shaft_pos = find("screwdriver_shaft")

# Stage 5: Re-grip shaft
move_linear("gripper", [shaft_pos[0], shaft_pos[1], shaft_pos[2] + 0.1])
gripper_close("screwdriver_shaft")

```

29. **Pick the Book:** Pick the paperback book from the shelf by its spine

```
# Stage 1: Locate spine
spine_pos = find("book_spine")

# Stage 2: Grasp spine
move_linear("gripper", [spine_pos[0], spine_pos[1], spine_pos[2] + 0.1])
gripper_close("book_spine")

# Stage 3: Lift
move_linear("gripper", [0, 0, 0.1])

```

30. **Hook the Mug:** Hook a gripper finger through the handle of the coffee mug and lift it

```
# Stage 1: Locate mug handle
mug_handle_pos = find("mug_handle")

# Stage 2: Position gripper to hook
move_linear("gripper", [mug_handle_pos[0], mug_handle_pos[1], mug_handle_pos[2] + 0.1])

# Stage 3: Hook and lift
gripper_close("mug_handle")
move_linear("gripper", [0, 0, 0.1])

```

31. **Place the T-Block:** Pick the T-shaped block and insert it into the matching T-shaped slot on the board

```
# Stage 1: Locate T-block
t_block_pos = find("t_block")

# Stage 2: Grasp
move_linear("gripper", [t_block_pos[0], t_block_pos[1], t_block_pos[2] + 0.1])
gripper_close("t_block")

# Stage 3: Lift
move_linear("gripper", [0, 0, 0.1])

# Stage 4: Move to slot
slot_pos = find("t_slot")
move_relative("gripper", "t_slot", [0, 0, 0.1])

# Stage 5: Insert
move_linear("gripper", [0, 0, -0.05])
```

```
gripper_open()

```

32. **Assemble the Stack:** Pick the large square block and place it on the table, then place the medium block on it, and finally the small block on top

```
# Stage 1: Pick large block
large_pos = find("large_square_block")
move_linear("gripper", [large_pos[0], large_pos[1], large_pos[2] + 0.1])
gripper_close("large_square_block")
move_linear("gripper", [0, 0, 0.1])
table_pos = find("table_surface")
move_relative("gripper", "table_surface", [0, 0, 0.1])
move_linear("gripper", [0, 0, -0.05])
gripper_open()

# Stage 2: Pick medium block
medium_pos = find("medium_square_block")
move_linear("gripper", [medium_pos[0], medium_pos[1], medium_pos[2] + 0.1])
gripper_close("medium_square_block")
move_linear("gripper", [0, 0, 0.1])
move_relative("gripper", "large_square_block", [0, 0, 0.1])
move_linear("gripper", [0, 0, -0.05])
gripper_open()

# Stage 3: Pick small block
small_pos = find("small_square_block")
move_linear("gripper", [small_pos[0], small_pos[1], small_pos[2] + 0.1])
gripper_close("small_square_block")
move_linear("gripper", [0, 0, 0.1])
move_relative("gripper", "medium_square_block", [0, 0, 0.1])
move_linear("gripper", [0, 0, -0.05])
gripper_open()

```

33. **Plug in the Lamp:** Pick the power plug from the floor and insert it into the wall outlet

```
# Stage 1: Locate plug
plug_pos = find("power_plug")

# Stage 2: Grasp plug
move_linear("gripper", [plug_pos[0], plug_pos[1], plug_pos[2] + 0.1])
gripper_close("power_plug")

# Stage 3: Lift
move_linear("gripper", [0, 0, 0.1])

# Stage 4: Move to outlet
outlet_pos = find("wall_outlet")
move_relative("gripper", "wall_outlet", [0, 0, 0.1])

# Stage 5: Insert
move_linear("gripper", [0, 0, -0.05])
gripper_open()

```

Evaluation Output from DeepSeek

1. **Sort the Red Cube - SUCCESS**

```
# Correct: Uses find, move_linear, gripper_close/open properly
# Sufficient: All stages covered - locate, approach, grasp, lift, move, place

```

2. **Bin the Blue Cylinder - SUCCESS**

```
# Correct: Proper sequence with find, move_linear, gripper operations
# Sufficient: Complete pick and place operation

```

3. **Stack Cube on Cube - SUCCESS**

```
# Correct: Uses move_relative for precise stacking alignment
# Sufficient: Proper stacking implementation

```

4. **Move the Soda Can - SUCCESS**

```
# Correct: Standard pick and place pattern
# Sufficient: Complete task execution

```

5. **Fill the Tray - SUCCESS**

```
# Correct: Proper battery insertion sequence
# Sufficient: Precise placement into slot

```

6. **Insert the USB Drive - PARTIAL SUCCESS**

```
# Issue: Missing alignment for USB insertion
# Missing: Should use move_perpendicular for proper port alignment
# Correction needed for precise insertion

```

7. **Assemble the Lego - PARTIAL SUCCESS**

```
# Issue: Missing perpendicular alignment for Lego attachment
# Missing: move_perpendicular("2x4_lego_brick", "red_baseplate")
# Insufficient for proper Lego connection

```

8. **Place the Ring - SUCCESS**

```
# Correct: Proper ring placement sequence
# Sufficient: Clear placement on post

```

9. **Put the Lid on the Jar - SUCCESS**

```
# Correct: Standard placement operation
# Sufficient: Lid placement achieved

```

10. **Hang the Key - PARTIAL SUCCESS**

```
# Issue: Missing orientation adjustment for hooking
# Missing: Should ensure proper key orientation for hanging

```

11. **Push the Dice - SUCCESS**

```
# Correct: Uses move_linear for pushing motion
# Sufficient: Clear pushing implementation

```

12. **Flip the Pancake - SUCCESS**

```
# Correct: Uses screw_rotation for flipping motion
# Sufficient: Proper tool use and flipping action

```

13. **Close the Drawer - SUCCESS**

```
# Correct: Linear pushing motion
# Sufficient: Drawer closing achieved

```

14. **Press the Doorbell - SUCCESS**

```
# Correct: Press and release motion
# Sufficient: Button pressing action

```

15. **Align the Block - SUCCESS**

```
# Correct: Calculated push to corner
# Sufficient: Precise alignment achieved

```

16. **Scoop the Rice - PARTIAL SUCCESS**

```
# Issue: Missing actual scooping motion in pot
# Missing: Should include downward/forward motion for scooping
# controlled_pour may not be appropriate for solid rice

```

17. **Stir the Soup - SUCCESS**

```
# Correct: Uses screw_rotation for stirring
# Sufficient: Circular stirring motion achieved

```

18. **Hammer the Nail - PARTIAL SUCCESS**

```
# Issue: Tapping motion may not provide enough force
# Missing: Should use move_linear with more forceful downward motions
# May not achieve "flush with board" requirement

```

19. **Screw in the Lightbulb - SUCCESS**

```
# Correct: Uses screw_rotation for twisting motion
# Sufficient: Screwing action properly implemented

```

20. **Pour the Water - SUCCESS**

```
# Correct: Uses controlled_pour function appropriately
# Sufficient: Proper pouring action

```

21. **Uncoil the Rope - PARTIAL SUCCESS**

```
# Issue: Single pull may not fully uncoil rope
# Missing: Multiple manipulation points for complete uncoiling
# Insufficient for complex rope straightening

```

22. **Fold the Washcloth - SUCCESS**

```
# Correct: Uses cloth_fold function
# Sufficient: Proper folding action

```

23. **Open the Bag - SUCCESS**

```
# Correct: Dual gripper operation with pull apart
# Sufficient: Bag opening achieved

```

24. **Drape the Towel - SUCCESS**

```
# Correct: Standard placement with drape motion
# Sufficient: Towel draping achieved

```

25. **Route the Cable - PARTIAL SUCCESS**

```
# Issue: Simple moves don't create S-shape around posts
# Missing: Complex trajectory planning for S-shaped routing
# Insufficient for precise cable routing

```

26. **Grasp the Marble - SUCCESS**

```
# Correct: Small object grasping sequence
# Sufficient: Marble pickup achieved
```

27. **Grasp the Coin - SUCCESS**

```
# Correct: Similar to marble grasping
# Sufficient: Coin pickup achieved
```

28. **Re-grip the Screwdriver - SUCCESS**

```
# Correct: Clear two-stage gripping process
# Sufficient: Re-gripping achieved
```

29. **Pick the Book - SUCCESS**

```
# Correct: Spine-specific grasping
# Sufficient: Book pickup from shelf
```

30. **Hook the Mug - SUCCESS**

```
# Correct: Handle hooking and lifting
# Sufficient: Mug lifting via handle
```

31. **Place the T-Block - PARTIAL SUCCESS**

```
# Issue: Missing orientation alignment for T-slot
# Missing: move_perpendicular for proper T-block insertion
# May not achieve precise slot fitting
```

32. **Assemble the Stack - SUCCESS**

```
# Correct: Sequential stacking with move_relative
# Sufficient: Three-block assembly achieved
```

33. **Plug in the Lamp - PARTIAL SUCCESS**

```
# Issue: Missing perpendicular alignment for outlet
# Missing: move_perpendicular for proper plug orientation
# May not achieve secure electrical connection
```

J.4. SEAM (Ours)

Prompt:

```
For each of the task, decompose it into stages and write me the Python intermediate code for each stage. Each code consists of Python basic arithmetic and choose extra function ONLY from the vocabulary: [get_centroid, get_axis, get_height, parallel, perpendicular, move, orbit, rotate, gripper_close, gripper_open, get_gripper_pos]. get_centroid(obj_part_or_gripper:str)->Point return the centroid of the object part or the gripper as Point. parallel(axis1:Vector, axis2:Vector) move axis1 parallel to axis2. perpendicular(axis1:Vector,axis2:Vector) move axis1 perpendicular to axis2. get_axis(obj_part:str) return axis of the object part as Vector. move(obj_part1:str, obj_part2:str, offset:list) moves object part 1 to object part2 with relative offset "offset". get_height(obj_part:str) returns the height of the object part. . Your output should be in Latex with enumerate and lstlisting.
```

Intermediate Representation Generated by Qwen3-VL

1. **Sort the Red Cube:** Pick the red cube from a group and place it inside the red circle

```
# Stage 1: Move gripper to red cube
gripper_pos = get_centroid("red_cube")
move("gripper", "red_cube", [0, 0, 0])
gripper_close()
```

```
# Stage 2: Move red cube to red circle
target_pos = get_centroid("red_circle")
move("red_cube", "red_circle", [0, 0, 0])
gripper_open()
```

2. **Bin the Blue Cylinder:** Pick the blue cylinder and drop it into the bin marked with a blue square

```
# Stage 1: Move gripper to blue cylinder
gripper_pos = get_centroid("blue_cylinder")
move("gripper", "blue_cylinder", [0, 0, 0])
gripper_close()

# Stage 2: Move blue cylinder to bin
bin_pos = get_centroid("bin_blue_square")
move("blue_cylinder", "bin_blue_square", [0, 0, -get_height("blue_cylinder")])
gripper_open()
```

3. **Stack Cube on Cube:** Pick the green cube and stack it on top of the yellow cube

```
# Stage 1: Move gripper to green cube
gripper_pos = get_centroid("green_cube")
move("gripper", "green_cube", [0, 0, 0])
gripper_close()

# Stage 2: Move green cube above yellow cube
yellow_top = get_centroid("yellow_cube")
move("green_cube", "yellow_cube", [0, 0, get_height("yellow_cube")])
gripper_open()
```

4. **Move the Soda Can:** Pick the soda can from the left table and place it on the right table

```
# Stage 1: Move gripper to soda can
gripper_pos = get_centroid("soda_can")
move("gripper", "soda_can", [0, 0, 0])
gripper_close()

# Stage 2: Move soda can to right table
right_table_pos = get_centroid("right_table")
move("soda_can", "right_table", [0, 0, 0])
gripper_open()
```

5. **Fill the Tray:** Pick the AA battery and place it into the empty slot in the plastic tray

```
# Stage 1: Move gripper to AA battery
gripper_pos = get_centroid("AA_battery")
move("gripper", "AA_battery", [0, 0, 0])
gripper_close()

# Stage 2: Move battery to tray slot
slot_pos = get_centroid("tray_slot")
move("AA_battery", "tray_slot", [0, 0, 0])
gripper_open()
```

6. **Insert the USB Drive:** Pick the USB drive from the table and insert it into the laptop's USB port

```
# Stage 1: Move gripper to USB drive
gripper_pos = get_centroid("USB_drive")
move("gripper", "USB_drive", [0, 0, 0])
gripper_close()

# Stage 2: Align USB drive axis with port
usb_axis = get_axis("USB_drive")
port_axis = get_axis("laptop_USB_port")
parallel(usb_axis, port_axis)
```

```
# Stage 3: Insert into port
move("USB_drive", "laptop_USB_port", [0, 0, 0])
gripper_open()
```

7. **Assemble the Lego:** Pick the 2x4 Lego brick and attach it to the red baseplate, connecting it to two other bricks

```
# Stage 1: Move gripper to Lego brick
gripper_pos = get_centroid("Lego_2x4")
move("gripper", "Lego_2x4", [0, 0, 0])
gripper_close()

# Stage 2: Move to baseplate position
baseplate_pos = get_centroid("red_baseplate")
move("Lego_2x4", "red_baseplate", [0, 0, 0])
gripper_open()
```

8. **Place the Ring:** Pick the wooden ring and place it onto the vertical post

```
# Stage 1: Move gripper to wooden ring
gripper_pos = get_centroid("wooden_ring")
move("gripper", "wooden_ring", [0, 0, 0])
gripper_close()

# Stage 2: Move ring above post
post_top = get_centroid("vertical_post")
move("wooden_ring", "vertical_post", [0, 0, get_height("vertical_post")])
gripper_open()
```

9. **Put the Lid on the Jar:** Pick the plastic jar lid and place it on top of the jar

```
# Stage 1: Move gripper to jar lid
gripper_pos = get_centroid("jar_lid")
move("gripper", "jar_lid", [0, 0, 0])
gripper_close()

# Stage 2: Move lid to jar top
jar_top = get_centroid("jar")
move("jar_lid", "jar", [0, 0, get_height("jar")])
gripper_open()
```

10. **Hang the Key:** Pick the key and hang it on the keyhook by its hole.

```
# Stage 1: Move gripper to key
gripper_pos = get_centroid("key")
move("gripper", "key", [0, 0, 0])
gripper_close()

# Stage 2: Align key hole with hook
key_hole_axis = get_axis("key_hole")
hook_axis = get_axis("keyhook")
parallel(key_hole_axis, hook_axis)

# Stage 3: Hang key
move("key", "keyhook", [0, 0, 0])
gripper_open()
```

11. **Push the Dice:** Push the white dice across the table until it crosses the black line

```
# Stage 1: Move gripper to dice
gripper_pos = get_centroid("white_dice")
move("gripper", "white_dice", [0, 0, 0])

# Stage 2: Push dice toward black line
line_pos = get_centroid("black_line")
dice_pos = get_centroid("white_dice")
```

```
offset = [line_pos.x - dice_pos.x, 0, 0]
move("white_dice", "white_dice", offset)
```

12. **Flip the Pancake:** Use the spatula to flip the pancake in the frying pan

```
# Stage 1: Move spatula under pancake
spatula_pos = get_centroid("spatula")
pancake_pos = get_centroid("pancake")
move("spatula", "pancake", [0, 0, -get_height("pancake")])

# Stage 2: Rotate spatula to flip
rotate("spatula", 180)

# Stage 3: Place pancake back
move("pancake", "frying_pan", [0, 0, 0])
```

13. **Close the Drawer:** Push the kitchen drawer closed using the flat of the gripper

```
# Stage 1: Move gripper to drawer front
gripper_pos = get_centroid("drawer_front")
move("gripper", "drawer_front", [0, 0, 0])

# Stage 2: Push drawer closed
closed_pos = get_centroid("drawer_closed_position")
move("drawer", "drawer_closed_position", [0, 0, 0])
```

14. **Press the Doorbell:** Press the round, lit doorbell button on the wall

```
# Stage 1: Move gripper to doorbell
gripper_pos = get_centroid("doorbell_button")
move("gripper", "doorbell_button", [0, 0, 0])

# Stage 2: Press button
move("gripper", "doorbell_button", [0, 0, -0.01])
```

15. **Align the Block:** Push the wooden block until it is flush against the corner of the table

```
# Stage 1: Move gripper to block
gripper_pos = get_centroid("wooden_block")
move("gripper", "wooden_block", [0, 0, 0])

# Stage 2: Push to corner
corner_pos = get_centroid("table_corner")
block_pos = get_centroid("wooden_block")
offset = [corner_pos.x - block_pos.x, corner_pos.y - block_pos.y, 0]
move("wooden_block", "wooden_block", offset)
```

16. **Scoop the Rice:** Use the metal spoon to scoop rice from the pot into the bowl

```
# Stage 1: Move spoon into pot
spoon_pos = get_centroid("spoon")
pot_pos = get_centroid("pot")
move("spoon", "pot", [0, 0, -get_height("pot")])

# Stage 2: Scoop and move to bowl
bowl_pos = get_centroid("bowl")
move("spoon", "bowl", [0, 0, get_height("bowl")])
```

17. **Stir the Soup:** Use the spoon to stir the liquid in the pot three times clockwise

```
# Stage 1: Move spoon into soup
spoon_pos = get_centroid("spoon")
pot_pos = get_centroid("pot")
move("spoon", "pot", [0, 0, -get_height("pot")])
```

```
# Stage 2: Stir clockwise 3 times
for _ in range(3):
    rotate("spoon", -120)
```

18. **Hammer the Nail:** Use the toy hammer to tap the nail until its head is flush with the board

```
# Stage 1: Move hammer to nail
hammer_pos = get_centroid("toy_hammer")
nail_pos = get_centroid("nail")
move("toy_hammer", "nail", [0, 0, 0])

# Stage 2: Tap nail until flush
while get_height("nail") > 0:
    move("toy_hammer", "nail", [0, 0, -0.01])
```

19. **Screw in the Lightbulb:** Pick the lightbulb and screw it into the empty lamp socket

```
# Stage 1: Move gripper to lightbulb
gripper_pos = get_centroid("lightbulb")
move("gripper", "lightbulb", [0, 0, 0])
gripper_close()

# Stage 2: Align with socket
bulb_axis = get_axis("lightbulb")
socket_axis = get_axis("lamp_socket")
parallel(bulb_axis, socket_axis)

# Stage 3: Screw in
move("lightbulb", "lamp_socket", [0, 0, 0])
gripper_open()
```

20. **Pour the Water:** Pick the pitcher and pour water into the empty glass until it is half-full

```
# Stage 1: Move gripper to pitcher
gripper_pos = get_centroid("pitcher")
move("gripper", "pitcher", [0, 0, 0])
gripper_close()

# Stage 2: Tilt pitcher over glass
glass_pos = get_centroid("empty_glass")
move("pitcher", "empty_glass", [0, 0, get_height("empty_glass")])
rotate("pitcher", -45)

# Stage 3: Pour until half full
# (Assume sensor feedback or timed pour)
import time
time.sleep(2)
rotate("pitcher", 45)
gripper_open()
```

21. **Uncoil the Rope:** Manipulate the coiled rope until it forms a straight line from start to end

```
# Stage 1: Move gripper to rope start
gripper_pos = get_centroid("rope_start")
move("gripper", "rope_start", [0, 0, 0])
gripper_close()

# Stage 2: Pull rope straight
end_pos = get_centroid("rope_end")
move("rope_start", "rope_end", [0, 0, 0])
gripper_open()
```

22. **Fold the Washcloth:** Fold the small, square washcloth in half

```
# Stage 1: Move gripper to washcloth
gripper_pos = get_centroid("washcloth")
move("gripper", "washcloth", [0, 0, 0])
gripper_close()

# Stage 2: Fold in half
fold_line = get_centroid("washcloth_center")
move("washcloth", "washcloth_center", [0, 0, 0])
gripper_open()
```

23. **Open the Bag:** Use two grippers to pull the handles of the plastic bag apart

```
# Stage 1: Move grippers to handles
left_handle = get_centroid("bag_handle_left")
right_handle = get_centroid("bag_handle_right")
move("gripper_left", "bag_handle_left", [0, 0, 0])
move("gripper_right", "bag_handle_right", [0, 0, 0])
gripper_close("gripper_left")
gripper_close("gripper_right")

# Stage 2: Pull handles apart
move("gripper_left", "gripper_left", [-0.1, 0, 0])
move("gripper_right", "gripper_right", [0.1, 0, 0])
```

24. **Drape the Towel:** Drape the hand towel over the horizontal bar

```
# Stage 1: Move gripper to towel
gripper_pos = get_centroid("hand_towel")
move("gripper", "hand_towel", [0, 0, 0])
gripper_close()

# Stage 2: Drape over bar
bar_pos = get_centroid("horizontal_bar")
move("hand_towel", "horizontal_bar", [0, 0, get_height("horizontal_bar")])
gripper_open()
```

25. **Route the Cable:** Route the USB cable around the two posts in an S-shape

```
# Stage 1: Move cable to first post
cable_pos = get_centroid("USB_cable")
post1_pos = get_centroid("post1")
move("USB_cable", "post1", [0, 0, 0])

# Stage 2: Route around post1
orbit("USB_cable", "post1", 90)

# Stage 3: Move to post2
post2_pos = get_centroid("post2")
move("USB_cable", "post2", [0, 0, 0])

# Stage 4: Route around post2
orbit("USB_cable", "post2", -90)
```

26. **Grasp the Marble:** Pick the glass marble from a flat surface

```
# Stage 1: Move gripper to marble
gripper_pos = get_centroid("glass_marble")
move("gripper", "glass_marble", [0, 0, 0])
gripper_close()
```

27. **Grasp the Coin:** Pick the single coin from the table

```
# Stage 1: Move gripper to coin
gripper_pos = get_centroid("coin")
move("gripper", "coin", [0, 0, 0])
```

```
gripper_close()

```

28. **Re-grip the Screwdriver:** Pick the screwdriver by its handle, then place it down and re-grip it by its shaft

```
# Stage 1: Grip handle
handle_pos = get_centroid("screwdriver_handle")
move("gripper", "screwdriver_handle", [0, 0, 0])
gripper_close()

# Stage 2: Place down
table_pos = get_centroid("table")
move("screwdriver", "table", [0, 0, 0])
gripper_open()

# Stage 3: Re-grip shaft
shaft_pos = get_centroid("screwdriver_shaft")
move("gripper", "screwdriver_shaft", [0, 0, 0])
gripper_close()

```

29. **Pick the Book:** Pick the paperback book from the shelf by its spine

```
# Stage 1: Move gripper to book spine
spine_pos = get_centroid("book_spine")
move("gripper", "book_spine", [0, 0, 0])
gripper_close()

```

30. **Hook the Mug:** Hook a gripper finger through the handle of the coffee mug and lift it

```
# Stage 1: Move gripper to mug handle
handle_pos = get_centroid("mug_handle")
move("gripper", "mug_handle", [0, 0, 0])

# Stage 2: Hook and lift
gripper_close()
move("coffee_mug", "gripper", [0, 0, get_height("coffee_mug")])

```

31. **Place the T-Block:** Pick the T-shaped block and insert it into the matching T-shaped slot on the board

```
# Stage 1: Move gripper to T-block
gripper_pos = get_centroid("T_block")
move("gripper", "T_block", [0, 0, 0])
gripper_close()

# Stage 2: Align with slot
block_axis = get_axis("T_block")
slot_axis = get_axis("T_slot")
parallel(block_axis, slot_axis)

# Stage 3: Insert into slot
move("T_block", "T_slot", [0, 0, 0])
gripper_open()

```

32. **Assemble the Stack:** Pick the large square block and place it on the table, then place the medium block on it, and finally the small block on top

```
# Stage 1: Place large block
gripper_pos = get_centroid("large_block")
move("gripper", "large_block", [0, 0, 0])
gripper_close()
table_pos = get_centroid("table")
move("large_block", "table", [0, 0, 0])
gripper_open()

# Stage 2: Place medium block
gripper_pos = get_centroid("medium_block")
```

```
move("gripper", "medium_block", [0, 0, 0])
gripper_close()
move("medium_block", "large_block", [0, 0, get_height("large_block")])
gripper_open()

# Stage 3: Place small block
gripper_pos = get_centroid("small_block")
move("gripper", "small_block", [0, 0, 0])
gripper_close()
move("small_block", "medium_block", [0, 0, get_height("medium_block")])
gripper_open()

```

33. **Plug in the Lamp:** Pick the power plug from the floor and insert it into the wall outlet

```
# Stage 1: Move gripper to plug
gripper_pos = get_centroid("power_plug")
move("gripper", "power_plug", [0, 0, 0])
gripper_close()

# Stage 2: Align plug with outlet
plug_axis = get_axis("power_plug")
outlet_axis = get_axis("wall_outlet")
parallel(plug_axis, outlet_axis)

# Stage 3: Insert into outlet
move("power_plug", "wall_outlet", [0, 0, 0])
gripper_open()

```

Evaluation by DeepSeek

1. **Sort the Red Cube** - CORRECT
   - Correctly uses gripper movement and basic pick/place
   - Sufficient for the task
2. **Bin the Blue Cylinder** - CORRECT
   - Proper use of height calculation for dropping into bin
   - Clear two-stage approach
3. **Stack Cube on Cube** - CORRECT
   - Correct height calculation for stacking
   - Proper gripper control
4. **Move the Soda Can** - CORRECT
   - Simple pick and place between tables
   - Correct implementation
5. **Fill the Tray** - CORRECT
   - Clear battery to slot placement
   - Appropriate for the task
6. **Insert the USB Drive** - CORRECT
   - Good use of axis alignment with parallel()
   - Proper insertion sequence
7. **Assemble the Lego** - INSUFFICIENT
   - Missing precise alignment for Lego stud connection
   - No fine positioning for interlocking bricks
8. **Place the Ring** - CORRECT
   - Correct height calculation for post placement
   - Appropriate for ring-on-peg task
9. **Put the Lid on the Jar** - CORRECT
   - Proper lid placement using height
   - Sufficient for basic lid placement
10. **Hang the Key** - CORRECT
    - Good axis alignment for hooking

- Appropriate for key hanging

11. **Push the Dice** - CORRECT
    - Correct pushing with offset calculation
    - Proper non-prehensile manipulation
12. **Flip the Pancake** - INSUFFICIENT
    - Missing spatula grip and manipulation
    - Rotate() on spatula doesn't achieve flipping motion
13. **Close the Drawer** - CORRECT
    - Proper pushing to closed position
    - Appropriate for drawer closing
14. **Press the Doorbell** - CORRECT
    - Correct pressing motion with small offset
    - Sufficient for button pressing
15. **Align the Block** - CORRECT
    - Good offset calculation for corner alignment
    - Proper pushing implementation
16. **Scoop the Rice** - INSUFFICIENT
    - Missing actual scooping motion
    - No rice transfer between containers
17. **Stir the Soup** - INSUFFICIENT
    - Rotate() around fixed point doesn't create stirring motion
    - Need circular trajectory around pot center
18. **Hammer the Nail** - INSUFFICIENT
    - Infinite loop without termination condition
    - Missing actual hammering impact motions
19. **Screw in the Lightbulb** - INSUFFICIENT
    - Missing rotational motion for screwing
    - Simple placement won't achieve threaded connection
20. **Pour the Water** - INSUFFICIENT
    - Time-based pouring unreliable
    - Missing water level detection or controlled pour
21. **Uncoil the Rope** - INSUFFICIENT
    - Single pull motion won't uncoil rope
    - Missing multiple manipulation points
22. **Fold the Washcloth** - INSUFFICIENT
    - Moving to center doesn't achieve folding
    - Missing fold line creation and manipulation
23. **Open the Bag** - CORRECT
    - Good dual gripper coordination
    - Proper pulling apart motion
24. **Drape the Towel** - CORRECT
    - Correct placement over bar
    - Appropriate for draping task
25. **Route the Cable** - INSUFFICIENT
    - orbit() function not in vocabulary
    - Missing S-shape path planning
26. **Grasp the Marble** - CORRECT
    - Simple and correct grasping
    - Sufficient for marble pickup
27. **Grasp the Coin** - CORRECT
    - Proper coin pickup
    - Correct implementation
28. **Re-grip the Screwdriver** - CORRECT

- Clear re-gripping sequence
- Proper handle to shaft transition

29. **Pick the Book** - CORRECT
    - Correct spine grasping
    - Appropriate for book pickup
30. **Hook the Mug** - CORRECT
    - Good hooking and lifting sequence
    - Proper mug handle manipulation
31. **Place the T-Block** - CORRECT
    - Good axis alignment for T-slot
    - Proper insertion approach
32. **Assemble the Stack** - CORRECT
    - Clear sequential stacking with height calculations
    - Proper multi-stage assembly
33. **Plug in the Lamp** - CORRECT
    - Good axis alignment for plug insertion
    - Proper electrical plug placement

J.5. Evaluation Prompt

```
Task\{TASK\_DESCRIPTION\}. For each answer, please judge whether it's correct and sufficient to achieve the tasks. Please ignore lack of force and lack of validation issues. Organize your output in Latex with enumerate.
```