Project website:`https://sites.google.com/ucsc.edu/vlmbench/home`

## A Task Details

Table 5: All task variations except shape used in VLMbench. The shape variation of each task can be found in the detail descriptions of each task category.

| Variations | Totals | Values |
|---|---|---|
| Color | 25 | seen:red, maroon, lime, green, blue,navy, yellow, cyan, magenta, silver, gray, olive, purple, teal, azure, violet, rose, black, white unseen: brown, gold, pink, chocolate, coral |
| Size | 5 | larger, smaller, large, medium, small |
| Relative Position | 5 | top, front, rear, left, right |
| Level | 3 | top, middle, bottom |
| Amount | 2 | fully, slightly |
| Action Type | 2 | open, close |

Table 6: All object models used in VLMbench. The number behind the object class indicate the instance number of that class.

| Object type | Number of classes | Classes |
|---|---|---|
| Basic model | 3 | cube (1), triangular prism (1), cylinder (1) |
| Special model | 9 | star (1), moon (1), cross (1), flower (1), letter ‘t’ (1), pencil (1), basket (1), box container(1), shape sorter (1) |
| Planar model | 6 | rectangle (1), circle (1), triangle (1), star (1), cross (1), flower (1) |
| Functional model | 2 | mug (6), sponge (1) |
| Articulated model | 2 | door with one rotatble handle (2), cabinet with three vertical drawers (3) |

In the VLMbench, we show eight task categories:“Pick & Place objects", “Stack objects", “Drop pencil", “Put into shape sorter", “Pour water", “Wiper table", “Use drawer", and “Use door". Here, we list variations used for these tasks in Table. 5. For each demonstration, all things in the scene will change the pose at the beginning. When building an instance-level task with one variation, the other variations will also randomly change. For example, in the demonstrations of “Pick & Place objects" with “size" variation, all objects’ color and relative positions, including targets and distractors, will randomly change. In the dataset, we have five types of objects, shown in Table 6. We will explain each task in detail as follows. Visualizations can be found on the project website.

### A.1 Pick & Place Objects

**Task Definition:** The agent needs to distinguish the specific object to grasp and then place it into a particular container. The object can be placed anywhere with any orientation inside the container.

**Task Templates:** Unit task sequence: *(Control, target object)+(M1[a1,b1], target object)*; Goal Conditions: A 3D bounding box without orientation constraints inside the empty space of target container, for *(M1[a1,b1], target object)*.

**Success Conditions:** The object detector inside the target container only can be triggered by the specific object. When the detector is triggered, the task considers a success.

**Instruction Templates:** High-level instructions: “Pick up [target object description] and place it into [target container description]."; Low-level instructions: (“Move to the top of [target object description]; Grasp [target object description].", “Move the object into [target container description]; Release the gripper.").

**Object models:** One container model is used in both seen and unseen. In the seen settings, five object models: star, triangular, cylinder, cube, and moon. In the unseen settings, four object models: cube, the letter 't,' cross, and flower.

**Variations and scene settings:**
All objects are randomly changing colors, size, and positions in each demonstration.
*Color*: There are two same-shape objects and two same-shape containers in the scene initialization. All colors are randomly sampled from the color library. The object description is "[color] object"; The container description is "[color] container."
*Size*: There are two same-shape objects and two same-shape containers in the scene initialization. One object and one container are randomly magnified while others are randomly shrunk. The object description is "[larger/smaller] object"; The container description is "[larger/smaller] container."
*Relative Position*: There are two same-shape objects and two same-shape containers in the scene initialization. All objects are randomly sampled in the workspace until they obey the predefined relative positions. The object description is "[front/rear/left/right] object"; The container description is "[front/rear/left/right] container."
*Shape*: There are two same-shape containers and more than two objects with different shapes in the scene initialization. The number of objects varies from two to the length of the object library. The object description is "[shape]"; The container descriptions is "[color] container."

### A.2 Stack Objects

**Task Definition:** The agent needs to distinguish the specific two objects and then stack them in a particular sequence. Since the objects have different shapes and some surfaces are hard to maintain for stacking, we use plane surfaces. Therefore, the goal pose of the above object should be inside the top plane of the below object and only can rotate along the axis which is perpendicular to the plane.

**Task Templates.** Unit task sequence: *(Control, target object)+(M1[a2,b2], above object)*; Goal Conditions: A plane on the surface of the below object with orientation constraints along the perpendicular axis, for *(M1[a2,b2], above object)*.

**Success Conditions.** The object detector attached to the bottom object will be triggered when the specific thing is above.

**Instruction Templates.** High-level instructions: "Stack [below object description] and [above object description] in sequence."; Low-level instructions: ("Move to the top of [above object description]; Grasp [above object description].", "Move the object on [below object description]; Release the gripper.").

**Object models:** In the seen settings, five object models: star, triangular, cylinder, cube, moon. In the unseen settings, four object models: cube, the letter 't', cross, flower.

**Variations and scene settings:**
All objects are randomly changing colors, size, and positions in each demonstration.
*Color*: There are four same-shape objects in the scene initialization. All colors are randomly sampled from the color library. The object descriptions are all "[color] [shape]."
*Size*: There are three same-shape objects in the scene initialization. One model is randomly magnified while another is randomly shrunk. The object description is "[large/medium/small] [shape]."
*Relative Position*: There are four objects with two different shapes in the scene initialization. All objects are randomly sampled in the workspace until the relative position of the same-shape objects obey the predefined relative positions. The below object description is "[front/rear/left/right] [shape 1]"; The above object description is "[front/rear/left/right] [shape 2]."
*Shape*: There are more than two objects with different shapes in the scene initialization. The number of objects varies from two to the length of the object library. The below object description is "[shape 1]"; The above object description is "[shape 2]."

### A.3 Drop Pencil

**Task Definition:** The agent must distinguish the specific pencil and then drop it into an upright container. Only when the pencil is vertical down above the container can it fall appropriately.

**Task Templates.** Unit task sequence: *(Control, target pencil)+(M1[a3,b2], target pencil)*; Goal Conditions: A line, which is perpendicular to the opening of target container, with orientation constraints along the line, for *(M1[a3,b2], target pencil)*.

**Success Conditions.** The object detector attached to the target container will be triggered when the specific pencil is inside.

**Instruction Templates.** High-level instructions: “Drop [target pencil description] into [target container description]."; Low-level instructions: (“Move to the top of [target pencil description]; Grasp [target pencil description].", “Move the object above [target container description]; Release the gripper.").

**Object models:** One pencil and one basket for both seen and unseen.

**Variations and scene settings:**
All objects are randomly changing colors and positions in each demonstration.
*Color*: There are two same-shape pencils and two same-shape baskets in the scene initialization. All colors are randomly sampled from the color library. The object description is “[color] pencil"; The container description is “[color] container."
*Size*: There are two same-shape pencils and two same-shape baskets in the scene initialization. One pencil and one container are randomly magnified while others are randomly shrunk. The object description is “[larger/smaller] pencil"; The basket description is “[larger/smaller] container."
*Relative Position*: There are two same-shape pencils and two same-shape baskets in the scene initialization. All objects are randomly sampled in the workspace until they obey the predefined relative positions. The object description is “[front/rear/left/right] object"; The basket description is “[front/rear/left/right] container."

### A.4 Put Into Shape Sorter

**Task Definition:** There is a shape sorter in the environment, and the agent needs to distinguish the specific object and then put it through the hole of the sorter. Only when the object’s shape and pose fit the hole can it fall.

**Variations:** Color, Shape, and Relative Position. These variations work on all objects except the sorter.

**Task Templates.** Unit task sequence: *(Control, target pencil)+(M1[a4,b3], target object)*; Goal Conditions: A fixed pose related to the sorter for each object shape, for *(M1[a4,b3], target object)*.

**Success Conditions.** The object detector attached to the sorter will be triggered when the specific object is inside.

**Instruction Templates.** High-level instructions: “Put [target object description] through the hole of [target hole description]."; Low-level instructions: (“Move to the top of [target object description]; Grasp [target object description].", “Move the object to the hole of [target hole description]; Release the gripper.").

**Object models:** One shape sorter container for both seen and unseen. In the seen settings, four object models: star, triangular, cylinder, and cube. In the unseen settings, five object models: star, triangular, cylinder, cube, and moon.

**Variations and scene settings:** All objects are randomly changing colors and positions in each demonstration.
*Color*: There are three same-shape objects and one shape sorter in the scene initialization. All colors are randomly sampled from the color library. The object description is “[color] [shape]."
*Relative Position*: There are two same-shape objects and one shape sorter in the scene initialization. All objects are randomly sampled in the workspace until they obey the predefined relative positions. The object description is “[front/rear/left/right] [shape]."
*Shape*: There are one shape sorter and more than two objects with different shapes in the scene initialization. The number of objects varies from two to the length of the object library. The object description is "[shape]."

## A.5 Pour Water

**Task Definition:** The agent needs to pour the water from the specific source mug into the particular container mug. Here we use 50 small particles to simulate the water.

**Variations:** Color, Size, and Relative Position. These variations work on all mugs.

**Task Templates.** Unit task sequence: *(Control, source mug)+(M2[a1,b2], source mug)+(M2[a4, b4], source mug)*; Goal Conditions: For *(M2[a1,b2], source mug)*, we need to keep the opening of the source mug is upward with the rotations along the axis of the opening directions. Further, the goal position of this step needs to make the horizontal distance between the geometry centers of two mugs less than the half-height of the source mug, and the vertical distance should be larger than the height of the container mug. For *(M2[a4, b4], source mug)*, the source mug needs to make the opening point to the container mug. Therefore, the end-effector should follow a particular path, keeping the source mug in the same position but rotating it to the required orientation. Therefore, the goal condition is a set of poses, calculated by a function using the geometry and poses of two mugs.

**Success Conditions.** The object detector attached to the container mug will be triggered when more than half particles are inside the mug.

**Instruction Templates.** High-level instructions: “Pour the water from [source mug description] to [container mug description]."; Low-level instructions: (“Move to the [relative position] of [source mug description]; Grasp [source mug description].", “Move the object to the top of [container mug description] with the opening upwards.", “Rotate [source mug description] toward [container mug description]").

**Object models:** Four different mugs in seen scenes and two different mugs in unseen scenes.

**Variations and scene settings:** All objects are randomly changing colors and positions in each demonstration.
*Color*: There are three same-shape mugs in the scene initialization. All colors are randomly sampled from the color library. The object description is “[color] mug."
*Relative Position*: There are two same-shape mugs in the scene initialization. All objects are randomly sampled in the workspace until they obey the predefined relative positions. The object description is “[front/rear/left/right] mug."
*Size*: There are two same-shape mugs in the scene initialization. One mug is randomly magnified while another is randomly shrunk. The object description is “[larger/smaller] mug."

## A.6 Wipe Table

**Task Definition:** The agent needs to wipe the particular area with a sponge, which means moving the sponge from one side of the area to another with keeping it contacting with the area.

**Task Templates.** Unit task sequence: *(Control, sponge)+(M1[a4,b2], sponge)*
*+(M2[a2, b2], sponge)*; Goal Conditions: For *(M1[a4,b2], sponge)*, the sponge needs to move to one side of the target area with the surface face contacting the area. For *(M2[a2, b2], sponge)*, the sponge needs to move from the current side to another side with keeping contact. The goal conditions are both a set of poses with the fixed position and rotations along the axis perpendicular to the table.

**Success Conditions.** We create 50 small invisible particles in the target area, and these particles will be removed if the sponge surface touches them. The successor attached to the target area will be triggered when more than half particles have been removed.

**Instruction Templates.** High-level instructions: “Wipe [target area description] with a sponge."; Low-level instructions: (“Move to the top of the sponge; Grasp the sponge.", “Move the object to the side of [target area description].", “Move the object along the main direction of [target area description]").

**Object models:** One sponge for both seen and unseen. The four planes in the seen scenes: rectangle, round, star, and triangle; The two planes in the unseen scenes: cross, flower.

**Variations and scene settings:** All objects are randomly changing colors and positions in each demonstration.
*Color*: There are two same-shape planes and one sponge in the scene initialization. All colors are

randomly sampled from the color library. The plane description is “[color] area."
*Size*: There are two same-shape planes and one sponge in the scene initialization. One plane is randomly magnified while another is randomly shrunk. The plane description is “[larger/smaller] area."
*Relative Position*: There are two same-shape planes and one sponge in the scene initialization. All planes are randomly sampled in the workspace until they obey the predefined relative positions. The object description is “[front/rear/left/right] area."
*Direction*: There are two same-shape directional planes and one sponge in the scene initialization. One plane is horizontal to the width of table while another is vertical. The plane description is “[horizontal/vertical] area."
*Shape*: There are one sponge and more than two planes with different shapes in the scene initialization. The number of planes varies from two to the length of the object library. The plane description is “[shape] area."

### A.7 Use Drawer

**Task Definition:** The agent needs to figure out the exact level of the drawers to use and execute the correct action with appropriate degrees. The initial states of the drawer will change according to the action types. For example, if the task is to open the top drawer, the top drawer will be closed at first.

**Task Templates.** Unit task sequence: *(Control, target drawer handle)+(M2[a3,b3], target drawer joint)*; Goal Conditions: For *(M2[a3,b3], target drawer)*, the agent needs to move the target drawer to a fixed position with the linear physic constraints on the joints of the drawer. Therefore, the goal conditions are a set of the end-effector poses whose positions are along an axis, and the orientations are fixed.

**Success Conditions.** The successor will be triggered when the joint of the target drawer have moved a particular length.

**Instruction Templates.** High-level instructions: “[Amount] [Action Type] the [Level] drawer."; Low-level instructions: (“Move to the front of the handle of [target drawer description]; Grasp the handle of [target drawer description].", “Move along the axis of [target drawer description] for [Amount] [Action Type].").

**Object models:** Two cabinets in seen scenes and one cabinet in unseen scenes. The cabinets have different appearance but all have three vertical drawers.

**Variations and scene settings:** The cabinet is randomly changing the pose in the workspace for each demonstration.

*Level, Action Type, and Amount*: One cabinet is sampled from the object list. These variations are all working on the drawer at the same time. For example, we regard “Fully open the top drawer." as one situation.

### A.8 Use Door

**Task Definition:** The agent needs to execute the correct action with appropriate degrees for the door. If the door is closed, the agent needs to rotate the door handle to unlock it for opening it. Like the use drawer tasks, the initial states will change according to the action type.

**Task Templates.** Unit task sequence: For closing the door, *(Control, door handle)+(M2[a4,b4],door plank joint)*. For opening the door, *(Control, door handle)+(M2[a4,b4],door handle joint)+(M2[a4,b4],door plank joint)*; Goal Conditions: For *(M2[a4,b4],door handle joint)*, the agent needs to rotate the joint, which connects the handle and plank, to a degree for unlock. Therefore, the goal conditions are a set of the end-effector poses whose positions are along an arc, and the orientations are depended on the positions. For *((M2[a4,b4], door plank joint)*, the goal conditions are the same as the previous one, but the joint connects the plank and base.

**Success Conditions.** The successor will be triggered when the plank and base joint has reached a particular angle.

**Instruction Templates.** High-level instructions: “[Amount] [Action Type] the door."; Low-level instructions: (“Move to the front of the handle of the door; Grasp the handle of the door.", “Rotate around the axis of the handle joint.", "Rotate around the axis of the door joint for [Amount] [Action Type].").

**Object models:** One door in seen scenes and one door in unseen scenes. The doors have different appearance but all have the rotatable handle.

**Variations and scene settings:** The door is randomly changing the pose in the workspace for each demonstration.

*Action Type and Amount*: One door is sampled from the object list. These variations are all working on the door at the same time. For example, we regard "Fully open the door." as one situation.

## B Implementation Details

### B.1 AMSolver

**Task Creation** To generate demonstrations for semantic tasks, we need to use the predefined unit task sequence combined with object configurations and goal conditions. The goal conditions are the feasible goal pose sets of the current manipulated object for each unit task. The goal conditions can be formatted as bounding boxes, planes, lines, joint angles, or generated by functions for special pose sets. For example, a rearrangement task *"Put the ball into the basket"* needs to transit a ball into the empty space of the basket without any constraints requirement, which means the goal condition of the ball is a set of poses inside the blank space of basket. Then, we can format this task as $(Control, ball) + (M1[a1, b1], ball)$. Note that *ball* can be replaced with other objects, so $(Control, object) + (M1[a1, b1], object)$ can represent any pick-and-place task without specific constraints. Another example is *"Pour the water from one mug to another"*. This task needs to keep the mug opening upward while moving toward the top of another mug. We can format this task as $(Control, source\ mug) + (M2[a1, b2], source\ mug) + (M2[a4, b4], source\ mug)$. For the first $(M2[a1, b2], source\ mug)$, the goal condition should make the horizontal distance between the geometry centers of two mugs less than the half-height of the source mug, and the vertical distance should be larger than the height of the container mug. Therefore, the goal condition is a set of poses calculated by a function using the geometry and poses of two mugs.

**Implementation** The AMSolver is built inside the CoppeliaSim [23] environment and written in Python and Lua based on RLbench [11] and PyRep [10]. The agent is a Franka Emika Panda robot arm with 7 DoF, standing on a wooden table. To achieve the grasping ability for any object shape, we implement one version of the algorithm of ten Pas et al. [19]. The generator will calculate the normals and principal axis of partial point clouds. Then, the normals, principal axis, and cross-product will establish the grasp poses. Finally, the agent will calculate inverse kinematics to check the validation. More details can be found in [19]. Each unit task will generate one waypoint as the sub-goal of the end-effector or one fixed cartesian path as the fixed sub-trajectory for the end-effector in the environment. We use the OMPL library [29] to find a valid trajectory between two waypoint configurations. In the end, we will get a valid path from the beginning to the task finish, consisting of the critical waypoints information. The object detector used in the simulator is a 3D box, where the poses (positions and orientations) and properties (length, width, height) can be modified. The object detector is placed in the target destinations of the task. When the target object mesh overlaps with the detector, it will return the true value for checking the object.

### B.2 VLMbench

**Dataset Generation** We collect the VLMbench dataset in the environment of RLbench with AMSolver. There are five RGB-D cameras in the environment: front view, left view, right view, overhead view, and wrist view. We use the image resolution of $360 \times 360$ in this dataset. As mentioned in section 3.4, the AMSolver will output the trajectories with waypoints. Therefore, the robot arm will use motion planning to transfer from one waypoint to another, and the simulator will record the observation with the time step 50 ms. Each camera can produce an RGB-D image with point cloud and instance masks for observation. We also record the low-level states of the robot joints, including joint velocity, joint torque, joint position, and end-effector pose. In addition, the observations record the object pose at each time step and the corresponding relationship between the frames and waypoints so that the demonstrations can be automatically divided into the sub-demonstrations with the critical

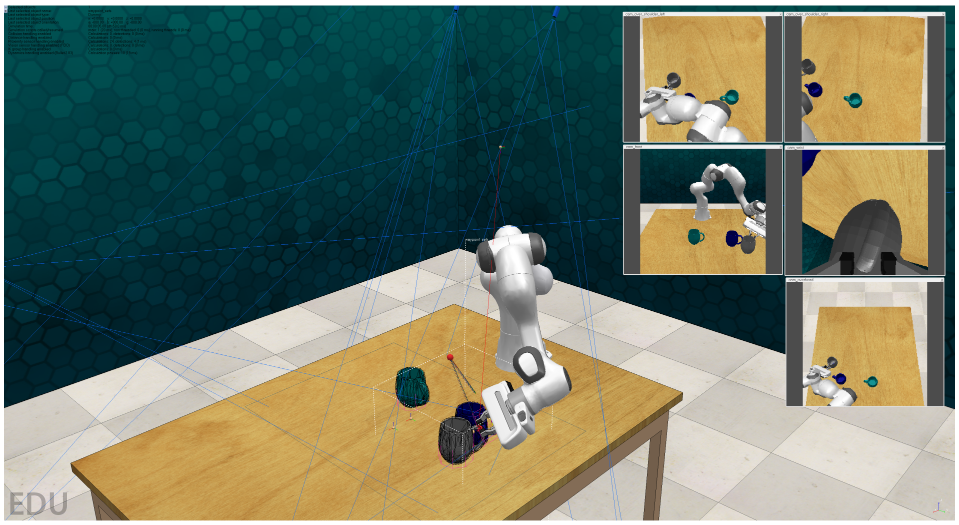


Figure 5: The environment used in VLMbench. A 7 DoF robot arm stands on the table with five cameras from different views. Meanwhile, the images from the camera views are shown aside. We also can see the predefine of the paths in the environment, which are the ground truth waypoints generated by the AMSolver.

frame. The objects will change the poses among different demonstrations, and the distractors will be randomly added to the environment.

For language instructions, we predefined some templates for each task category to quickly generate various language instructions by filling the object properties' descriptions, shown in Fig. 3. For example, the "Pick & Place objects" task has the template "Pick [Object] and place it into [Container]," where [Object] denotes the target object descriptions corresponding to variations mentioned above. Meanwhile, by defining the structured language templates on the unit task, we can simultaneously generate the low-level instructions in the dataset, which structurally describe the basic actions corresponding to the frames, e.g. "Move to [relative position] [manipulated object]; Grasp [manipulated object]" correspond to the pre-grasp action and grasp action in the Control unit task. We hope this information will be helpful for further research in this area.

### B.3 6D-CLIPort

In this section, we provide hyperparameter values in our 6D-CLIPort training. We have trained each agent on eight A5000 GPUs for one hour with a batch size of 16, an epoch of 15, and a learning rate of 1e-3. The pixel-wise feature maps from the attention module have one dimension. The output feature maps of the value and key module have four dimensions in each pixel to estimate 2D positions and the rotation along the perpendicular axis. Meanwhile, The output feature maps have 12 dimensions in each pixel for the other three regressors and are chunked into three parts for separate convolutions, where the convolutions of value and key feature maps are input to each regressor.

## C Dataset Details

We sample VLMbench in the CoppeliaSim simulator, shown in Fig. 5. The simulator collects the data at a 20 Hz frequency, and at each step, the state record all visual, robot, and object information. To illuminate the dataset's structure, we have a folder for each instance-level task, e.g., pouring water with color variation. Each variation value creates a folder under the task folder, e.g., the target mug is red in the pouring water with color variation. Inside one variation value folder, we have demonstration folders that contain the state record for demonstrations. Then, we save the RGB-D, segmentation, and point cloud from each camera in one demonstration folder. Another file contains all low-level robot states, object states, and waypoint information for every step. All instance-level tasks are shown in Fig. 6. Demonstration demos can be found in the video on the website.

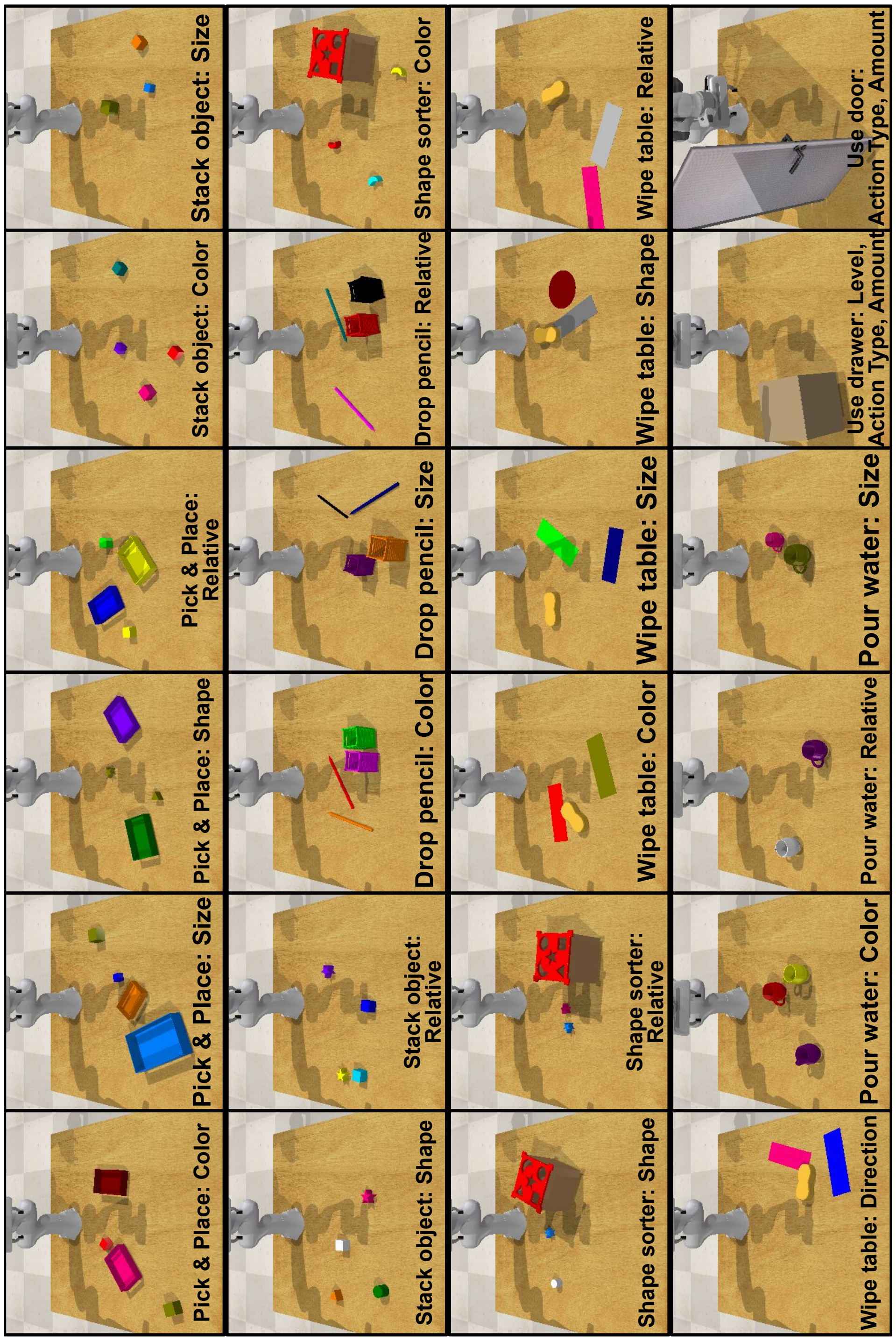


Figure 6: All instance-level tasks in VLMbench.

Table 7: The number of episodes for each task in the dataset.

| Task category | Variation | Train | Valid | | Test | |
|---|---|---|---|---|---|---|
| | | | Seen | Unseen | Seen | Unseen |
| Pick | Color | 400 | 100 | 25 | 100 | 100 |
| | Relative | 320 | 80 | 80 | 96 | 96 |
| | Shape | 100 | 25 | 20 | 100 | 100 |
| | Size | 80 | 20 | 20 | 100 | 100 |
| Stack | Color | 400 | 100 | 25 | 100 | 100 |
| | Relative | 320 | 80 | 80 | 96 | 96 |
| | Shape | 100 | 25 | 20 | 100 | 100 |
| | Size | 120 | 30 | 30 | 96 | 96 |
| Drop | Color | 395 | 100 | 25 | 100 | 100 |
| | Relative | 320 | 80 | 80 | 96 | 96 |
| | Size | 80 | 20 | 20 | 100 | 100 |
| Place | Color | 400 | 100 | 25 | 100 | 100 |
| | Relative | 80 | 20 | 20 | 100 | 100 |
| | Shape | 80 | 20 | 25 | 100 | 100 |
| Wipe | Color | 400 | 100 | 25 | 100 | 100 |
| | Relative | 80 | 20 | 20 | 100 | 100 |
| | Shape | 80 | 20 | 20 | 100 | 100 |
| | Size | 40 | 10 | 10 | 100 | 100 |
| | Direction | 40 | 10 | 10 | 100 | 100 |
| Pour | Color | 388 | 100 | 25 | 100 | 100 |
| | Relative | 80 | 20 | 20 | 100 | 100 |
| | Size | 40 | 10 | 10 | 100 | 100 |
| Door | action&amount | 200 | 20 | 20 | 100 | 100 |
| Drawer | action&amount&level | 240 | 60 | 60 | 96 | 96 |
| | Total | 4783 | 1170 | 705 | 2380 | 2380 |

**Task Statistics** In total, we have 24 instance-level tasks with 234 variations, which include 120 color variations, 18 size variations, 60 relative position variations, 18 shape variations, 2 direction variations, and 16 variations of the combinations of level, amount, and action type. The benchmark have 6,658 manipulation demonstrations which contain 4,783 train demonstrations, 1,170 seen validation demonstrations, and 705 unseen validation demonstrations. Since the agent test in the online simulator, we have generated 2,380 seen test settings and 2,380 unseen test settings for all instance-level tasks. The detailed numbers can be found in Table 7. All demonstrations are generated by two servers with 64-Core CPUs and 8 GPUs within 24 hours.

## D Additional Experiments

We have shown task success rates of each baseline agent in both seen and unseen settings. Here, we provide the results of step success for both seen and unseen settings. Then, we want to provide more explanation and analysis of failure cases and ablations study.

### D.1 Goal-Conditioned Success Rates

In Table 8 and 9, we show the success rates of each step for every agent. For tasks in the VLMbench, each task can be considered as a multi-step task, since each task at least consists of one grasping step and one following movement step. Therefore, we used two other metrics for the step's success: Goal-condition Grasp (G-G) success rate and Goal-conditioned Movement (G-M) success rate. The Goal-conditioned Grasp (G-G) success rates indicate whether the agent has grasped the correct object for the first step. The Goal-conditioned Movement (G-M) success rates indicate whether the agent can satisfy the success conditions in the following movement by using the pre-generated grasp poses

| Agent | Pick&Place | | | | | | Stack | | | | | | Drop | | | | | | Shape Sorter | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Seen | | | Unseen | | | Seen | | | Unseen | | | Seen | | | Unseen | | | Seen | | | Unseen | | |
| | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC |
| Language-Only | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| Vision-Only | 29.55 | 29.80 | 6.31 | 36.62 | 26.52 | 9.85 | 34.95 | 32.65 | 6.89 | 32.65 | 22.45 | 1.79 | 10.47 | 0.00 | 0.00 | 16.89 | 0.00 | 0.00 | 8.33 | 8.67 | 0.00 | 4.00 | 6.33 | 0.33 |
| 6D-CLIPort | **58.33** | **43.33** | **28.28** | **63.14** | **41.16** | **27.53** | **53.06** | **44.64** | **22.19** | **48.21** | **28.06** | **18.37** | **70.61** | **9.12** | **6.42** | **65.88** | **8.78** | **6.42** | **73.00** | **23.67** | **17.33** | **66.67** | **18.67** | **12.33** |
| 6D-CLIPort (GT Ori) | 60.61 | 42.93 | 28.03 | 65.40 | 36.87 | 26.26 | 56.38 | 44.64 | 26.53 | 51.02 | 48.98 | 26.02 | 65.88 | 31.08 | 17.91 | 62.16 | 28.72 | 16.22 | 72.67 | 32.67 | 24.00 | 67.33 | 24.00 | 15.67 |
| 6D-CLIPort (GT Pos) | 94.95 | 88.64 | 83.84 | 95.96 | 83.33 | 75.25 | 96.94 | 69.39 | 58.93 | 96.43 | 47.70 | 50.51 | 95.61 | 13.85 | 16.89 | 94.26 | 13.18 | 11.82 | 92.00 | 19.33 | 18.00 | 94.00 | 22.67 | 17.33 |

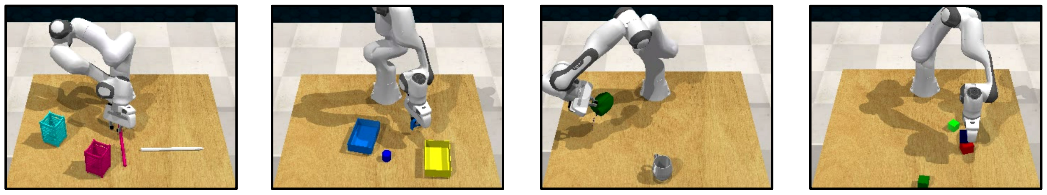
| | Pour | | | | | | Wipe | | | | | | Door | | | | | | Drawer | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Seen | | | Unseen | | | Seen | | | Unseen | | | Seen | | | Unseen | | | Seen | | | Unseen | | |
| | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC |
| Language-Only | 0.00 | 0.76 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 5.21 | 15.63 | 4.17 | 3.13 | 13.54 | 1.04 |
| Vision-Only | 2.67 | 2.00 | 0.00 | 4.00 | **1.33** | 0.00 | 99.20 | 19.40 | 19.80 | **96.00** | 21.20 | 20.80 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | **26.04** | 21.88 | 14.58 | 1.04 | 18.75 | 7.29 |
| 6D-CLIPort | **61.00** | **2.00** | **1.00** | **69.67** | 1.00 | **1.00** | **99.20** | **21.60** | **22.40** | 95.20 | **21.60** | **21.80** | **29.00** | **14.00** | **6.00** | **5.00** | **15.00** | **5.00** | 22.92 | **25.00** | **22.92** | **8.33** | **21.88** | **15.63** |
| 6D-CLIPort (GT Ori) | 70.00 | 4.00 | 3.67 | 76.67 | 3.67 | 3.67 | 99.60 | 25.40 | 25.80 | 97.20 | 25.20 | 25.20 | 49.00 | 14.00 | 6.00 | 5.00 | 15.00 | 5.00 | 31.25 | 30.00 | 23.96 | 8.33 | 23.96 | 17.71 |
| 6D-CLIPort (GT Pos) | 61.67 | 3.00 | 0.33 | 67.33 | 0.67 | 0.67 | 100.00 | 61.00 | 60.20 | 99.40 | 57.40 | 53.40 | 80.00 | 46.00 | 27.00 | 83.00 | 49.00 | 27.00 | 79.17 | 43.75 | 43.75 | 87.50 | 50.00 | 52.08 |

Table 8: Results of all tasks, including both seen and unseen settings. In the table, G-G denotes Goal-conditioned Grasp success rate, which reflect whether the agent can correctly grasp the target. G-M denotes Goal-conditioned Movement success rate, which reflect whether the agent can finish the task by using the pre-generated grasping poses. SC denotes success rates for the whole estimation trajectory. More explanations can be found in section D.1.

| Agent | Color | | | | | | Shape | | | | | | Size | | | | | | Relative Position | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Seen | | | Unseen | | | Seen | | | Unseen | | | Seen | | | Unseen | | | Seen | | | Unseen | | |
| | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC |
| Language-Only | 0.00 | 0.33 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.25 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 |
| Vision-Only | 29.67 | 14.17 | 6.00 | 30.00 | 13.00 | 5.00 | 42.75 | 21.00 | 6.00 | 46.25 | 20.00 | 8.50 | 36.49 | 15.12 | 6.05 | 36.09 | 12.76 | 5.10 | 31.46 | 19.73 | 6.80 | 31.97 | 12.76 | 5.10 |
| 6D-CLIPort | **69.83** | **19.67** | **15.17** | **69.00** | **17.50** | **13.00** | **69.75** | **33.25** | **23.00** | **71.25** | **27.25** | **19.50** | **70.56** | **25.40** | **18.35** | **64.31** | **21.77** | **14.92** | **68.03** | **27.72** | **15.31** | **69.56** | **20.75** | **15.82** |
| 6D-CLIPort (GT Ori) | 72.00 | 24.67 | 18.67 | 70.83 | 26.00 | 17.67 | 70.25 | 39.75 | 28.00 | 72.50 | 33.75 | 27.25 | 71.57 | 29.23 | 20.97 | 66.53 | 28.02 | 17.74 | 70.58 | 33.33 | 21.43 | 71.09 | 29.42 | 18.37 |
| 6D-CLIPort (GT Pos) | 90.67 | 42.33 | 40.17 | 90.33 | 37.50 | 35.00 | 96.00 | 59.75 | 56.25 | 96.75 | 53.25 | 51.25 | 90.32 | 47.58 | 44.96 | 91.53 | 41.33 | 38.51 | 89.12 | 41.50 | 38.61 | 90.99 | 36.39 | 34.86 |

| | Direction | | | | | | Level | | | | | | Action Type | | | | | | Amount | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Seen | | | Unseen | | | Seen | | | Unseen | | | Seen | | | Unseen | | | Seen | | | Unseen | | |
| | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC | G-G | G-M | SC |
| Language-Only | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 5.21 | 15.63 | 4.17 | 3.13 | 13.54 | 1.04 | 2.55 | 7.65 | 2.04 | 1.53 | 6.63 | 0.51 | 2.55 | 7.65 | 2.04 | 1.53 | 6.63 | 0.51 |
| Vision-Only | **99.00** | 15.00 | 21.00 | 95.00 | 22.00 | 24.00 | 26.04 | 21.88 | 14.58 | 1.04 | 18.75 | 7.29 | 12.76 | 10.71 | 7.14 | 0.51 | 9.18 | 3.57 | 12.76 | 10.71 | 7.14 | 0.51 | 9.18 | 3.57 |
| 6D-CLIPort | 98.00 | **19.00** | **21.00** | **100.00** | **22.00** | **26.00** | **22.92** | **25.00** | **22.92** | **8.33** | **21.88** | **15.63** | **26.02** | **19.39** | **14.29** | **6.63** | **18.37** | **10.20** | **26.02** | **19.39** | **14.29** | **6.63** | **18.37** | **10.20** |
| 6D-CLIPort (GT Ori) | 99.00 | 26.00 | 26.00 | 98.00 | 29.00 | 27.00 | 31.25 | 30.00 | 23.96 | 8.33 | 23.96 | 17.71 | 40.31 | 22.45 | 14.80 | 6.63 | 19.39 | 11.22 | 40.31 | 22.45 | 14.80 | 6.63 | 19.39 | 11.22 |
| 6D-CLIPort (GT Pos) | 100.00 | 63.00 | 53.00 | 100.00 | 56.00 | 41.00 | 79.17 | 43.75 | 43.75 | 87.50 | 50.00 | 52.08 | 79.59 | 44.90 | 35.20 | 85.20 | 49.49 | 39.29 | 79.59 | 44.90 | 35.20 | 85.20 | 49.49 | 39.29 |

Table 9: Results of all variations. The notations are used the same as in Table 3.



(a) **Wrong Grasping Pose** "Drop the pink pencil into the pink container."

(b) **Wrong Grasping Object** "Pick up the cylinder and place it into the yellow container."

(c) **Wrong Destination Pose** "Pour water from the green mug to the gray mug."

(d) **Wrong Destination Object** "Stack the green cube and the navy cube in sequence."

Figure 7: Four failure cases in the results of the 6D-CLIPort. Details can be found in D.2.

for the grasping step. We can find that the agent has high grasping successes in both seen and unseen settings for most tasks, including pick, stack, drop, shape sorter, pour, and wiper, which indicates that 6D-CLIPort can successfully reason the correct grasping objects with both seen and unseen settings. For these tasks, the prominent failure cases come from the following movement. The conclusion can also be obtained from the minor difference between the goal-conditioned movement and the total success rate. In the wiping tasks, we can see high grasping success rates in all settings since we only have one sponge shape model in both training and testing. We also can find that the Vision-Only agent has a higher grasping success rate than 6D-CLIPort in the "Drawer" tasks with seen settings. However, it has a much lower grasping success rate in the unseen settings, which indicates that the Vision-Only agent has overfitted on the grasping step in the "Drawer" tasks. For opening/closing drawer and door tasks, we find that goal-conditioned movement success rates are similar for seen and unseen settings, meaning the agent performs similarly after the agent has correct grasping. In opening/closing drawer tasks, we also find that the goal-condition grasp is smaller than total success, which means some tasks reach the success conditions without grasping the handle of doors. It also makes sense that closing the drawer can be finished by pushing instead of grasping the handle first.

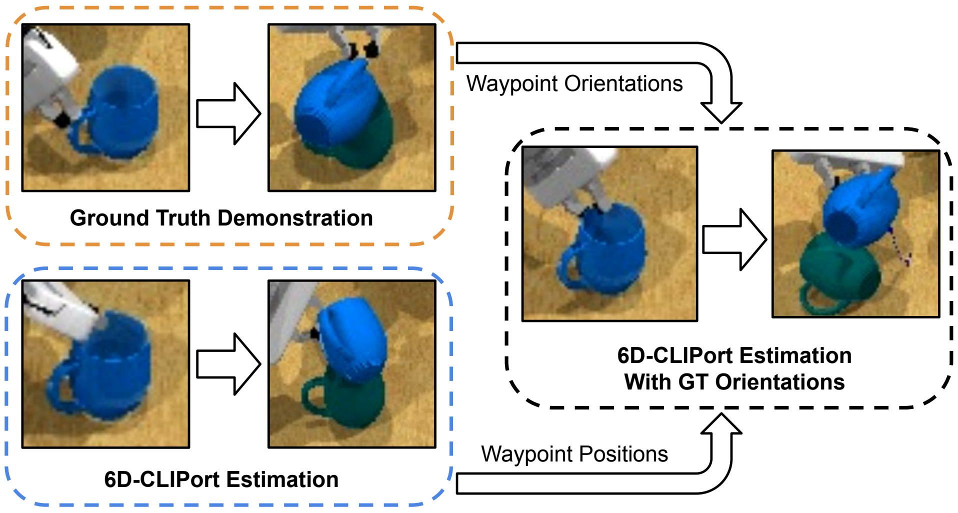


Figure 8: 6D-CLIPort with ground truth waypoint information. The instruction here is “Pour water from the blue mug to the green mug." Since the orientations and positions are unmatched, we can see 6D-CLIPort with ground truth orientation can pour the water but have collisions with the container mug. In this situation, 6D-CLIPort with ground truth positions cannot find a valid path by motion planning.

### D.2 Failure Cases

The results show that the 6D-CLIPort agent has failed in many tasks. Except for the unreachable poses due to the errors in the position and orientation estimations, we summarize the four kinds of failure cases, shown in Fig. 7: (a) Wrong grasping pose, which means the agent cannot grasp any object. (b) Wrong grasping object, which means the agent grasps the incorrect object. (c) Wrong destination pose, which means the destination pose of the agent cannot finish any semantic task. (d) Wrong destination object, which means the estimation destination is inconsistent with the instructions. More visualization can be found in the attached video.

### D.3 Partially Modal Agents

In the experiment section, we have tested the 6D-CLIPort with the ground truth waypoints' positions or orientations. We can see the agent still failed the tasks even with this privileged information. The reason is the unmatched positions and orientations. The ground truth positions and orientations are obtained from the pre-generated waypoints. Therefore, this ground truth information is not the optimal parameter associated with the estimation. The unmatched positions and orientations can lead to a failure or even an unreachable pose where the motion planner cannot find a feasible path. For the tasks that need the cooperation of position and orientation, such as dropping, pouring, and opening the door, we can figure out that giving partially ground truth still cannot finish the task correctly. We shown a example in Fig 8.Since the estimation trajectory and ground truth trajectory can be valid but different solutions for the same task. The combination of the positions and orientations will fail the task. Furthermore, the wrong estimation of positions can lead to task failure, even given the ground truth orientation, since the tasks in the VLMbench require correct compositional reasoning. More visualization can be found in the video on the project web page.