VLMbench: A Benchmark for Vision-and-Language Manipulation

Kaizhi Zheng
University of California, Santa Cruz
kzheng31@ucsc.edu

Xiaotong Chen
University of Michigan, Ann Arbor
cxt@umich.edu

Odest Chadwicke Jenkins
University of Michigan, Ann Arbor
ocj@umich.edu

Xin Wang
University of California, Santa Cruz
xwang366@ucsc.edu

1. Introduction

Recent progress in embodied AI [2, 5–7] pushes intelligent robotic systems to reality closer than any other time before. To achieve the final goal of interacting with unstructured environments to accomplish various daily tasks, the agent needs to learn how to manipulate objects through visual observations and natural languages appropriately. Compared with vision-only systems, natural language possesses two essential properties that enhance robot manipulation task learning: compact and flexible specification of various tasks and natural interactive communication interface with humans. We name such an agent that learns from the combined knowledge of language and vision to accomplish robot manipulation tasks a **Vision-and-Language Manipulation (VLM)** embodied agent. Given the rapid growth in VLM research and their limitations on generalization across different tasks, we expect an inclusive, modular, and scalable benchmark to evaluate embodied agents for various manipulation tasks.

Compared to recent benchmarks developed to evaluate robotics manipulation tasks with language guidance and visual input [1, 3, 8], which lack (1) adaptation to novel objects automatically and (2) categorization for modular and flexible composition to complex tasks, we present **VLMbench**, a highly categorical robotic manipulation benchmark that generates numerous task demonstrations, as shown in Fig. 1. To generalize across the different object and task settings, we propose **AMSolver**, an automatic unit task builder that can generate various robot trajectories and grasping poses to accomplish the desired action for a unit task. Compared with previous works [3] that require the user to design a new task from scratch, we propose unit task templates that can generate a new task simply by specifying object properties. For instance, the pick-place task can be automatically generated for different objects provided their geometric models, with free variation in the objects' colors, shapes, and sizes and 6 degrees of freedom (DoF) poses in the scene. The unit tasks can be composed together to create complex multi-step tasks. Besides, VLMbench provides both high-level language descriptions of the entire task and low-level action descriptions that correspond to unit robot moves.

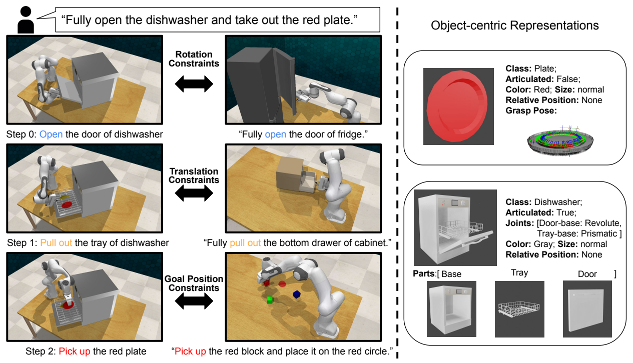
"Fully open the dishwasher and take out the red plate."
Rotation Constraints
Step 0: Open the door of dishwasher
"Fully open the door of fridge."
Translation Constraints
Step 1: Pull out the tray of dishwasher
"Fully pull out the bottom drawer of cabinet."
Goal Position Constraints
Step 2: Pick up the red plate
"Pick up the red block and place it on the red circle."
Object-centric Representations
Class: Plate;
Articulated: False;
Color: Red; Size: normal
Relative Position: None
Grasp Pose:
Class: Dishwasher;
Articulated: True;
Joints: [Door-base: Revolute, Tray-base: Prismatic]
Color: Gray; Size: normal
Relative Position: None
Parts:[Base
Tray
Door]

Figure 1. Given the language instructions and observations, the VLMbench requires the agent to generate an executable manipulation trajectory for specific task goals. On the left, we show that the complex tasks can be divided into the unit tasks according to the constraints of the end-effector, like "Open the doof of the dishwasher" and "Open the door of the fridge" should both follow the rotation constraints of the revolute joint. We show examples of object-centric representations on the right, where all graspable objects or parts will generate local grasping poses as their attributes. Depending on the modular design, we can generate reasonable VLM data automatically.

2. AMSolver: Automatic Manipulation Solver

We consider three main categories of household manipulation tasks: rearrangement (move an object from place to place, e.g., collect trash into the bin), affordance usage (exploit particular usage of some object, e.g., pour water from a mug), and multi-step tasks (can be decomposed into several individual steps, e.g., connect every two pieces to assemble a table from parts). To formulate a unified task representation, we assume that every task within these categories can be considered a combination of objects and unit actions. To describe the essential elements of tasks, we proposed Automatic Manipulation Solver (AMSolver), a rule-based automatic task builder consisting of object-centric representations and unit task templates, shown in Fig. 2.

Rule-based Unit Tasks We define a *unit task* as the semantic step of completing a sub-goal of the entire task. Specifically, a unit task is defined in a formula of 'take action on an object under certain constraints' where a unit task can be parameterized by two constraints: (a) **position constraints** and (b) **orientation constraints**, which describe the valid spatial space or orientation range, respectively, of the end-effector for a specific task. We propose three unit task templates detailed below that can compose the aforementioned complex tasks: **Control** is a preparation or ending step of a task, which models the transition of the object state, where the state indicates whether the robot can move the object or not; **M1** denotes moving the target object with goal pose constraints (position and orientation constraints), which can be modeled as a 6 DoF transform in the robot's workspace; **M2** denotes moving the target object along a trajectory while satisfying the motion constraints during the entire path, which implies a more strict condition than **M1**.

Object-centric Representation Some recent works [4, 9] have used object-centric representations for manipulation. Since the properties are defined on objects, these representations can easily cross the variations of environments, agents, and tasks. Our benchmark assumes that objects used in the tasks are rigid, and their fundamental properties will not change during the tasks. Therefore, we can parameterize the objects as a set of configurations, including class, color, size, and geometry shape. If the object is articulated, its whole configuration will contain the configuration of each part and the physical constraint of each connection.

3. VLMbench: Visual-and-Language Manipulation Benchmark

Given language instructions, the Vision-and-Language Manipulation (VLM) task requires an embodied agent to follow the instructions to complete tabletop manipulation tasks. Formally, at the beginning of the task, the agent receives a set of language instructions $L = \{L_1, L_2, ..., L_n\}$, where L_i denotes one sentence of arbitrary length. The initial state s_0 contains multi-view RGB images, depth images, segmentation information, and robot states, including joint angles, velocities and torques, and end-effector pose. Given the observations and language instructions, the agent needs to estimate an executable action command a_0, directly working on the end-effector or joints. Then, at each step t, the agent receive new observations o_t and generate the action $a_t = f(s_t | s_0, s_1, ..., s_{t-1}, L)$ for the next step. The step loop will repeat until the agent sends a stop action or should be terminated, e.g., achieve the success conditions or the limitation steps. The agent should obey the constraints provided by language instructions during the whole running.

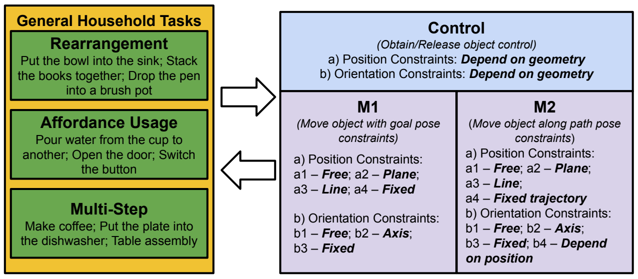
General Household Tasks
Rearrangement
Put the bowl into the sink; Stack the books together; Drop the pen into a brush pot
Affordance Usage
Pour water from the cup to another; Open the door; Switch the button
Multi-Step
Make coffee; Put the plate into the dishwasher; Table assembly
Control
(Obtain/Release object control)
a) Position Constraints: Depend on geometry
b) Orientation Constraints: Depend on geometry
M1
(Move object with goal pose constraints)
a) Position Constraints:
a1 – Free; a2 – Plane;
a3 – Line; a4 – Fixed
b) Orientation Constraints:
b1 – Free; b2 – Axis;
b3 – Fixed
M2
(Move object along path pose constraints)
a) Position Constraints:
a1 – Free; a2 – Plane;
a3 – Line;
a4 – Fixed trajectory
b) Orientation Constraints:
b1 – Free; b2 – Axis;
b3 – Fixed; b4 – Depend on position

Figure 2. The unit task templates in the AMSolver. On the left, we show three main common task types of household tasks and give several example tasks for each. On the right, we show three unit task templates parameterized by position and orientation constraints over the robot end-effector on either goal pose or entire path. By combining these unit task templates, various task examples can be generated.

Tasks Generation The manipulated object's properties can randomly change for each task category, and every combination leads to a task instance. In the VLMbench, we use 8 different variations from object and motion perspectives Object variations include colors, sizes, shapes, relative positions, and directions. From the motion view, the variations are amounts and action types. The amounts mean how far the task needs to be done, consisting of "fully" and "slightly." The action types include "open" and "close", especially for the articulated objects.

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 224×224 in this dataset. The objects will change the poses among different demonstrations.

For language instructions, we predefined some templates for each task category to quickly generate various language instructions by filling the object properties' descriptions. 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. In total, we have generated 3760 trajectories in 24 hours on a 64-core computer with 8 GPUs.

References

[1] Ossama Ahmed, Frederik Träuble, Anirudh Goyal, Alexander Neitz, Yoshua Bengio, Bernhard Schölkopf, Manuel Wüthrich, and Stefan Bauer. Causalworld: A robotic manipulation benchmark for causal structure and transfer learning. *arXiv preprint arXiv:2010.04296*, 2020. 1

[2] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. pages 1–10, 2018. 1

[3] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020. 1

[4] Chengshu Li, Fei Xia, Roberto Martín-Martín, Michael Lingelbach, Sanjana Srivastava, Bokui Shen, Kent Vainio, Cem Gokmen, Gokul Dharan, Tanish Jain, et al. Igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. *arXiv preprint arXiv:2108.03272*, 2021. 2

[5] Weiyu Liu, Chris Paxton, Tucker Hermans, and Dieter Fox. Structformer: Learning spatial structure for language-guided semantic rearrangement of novel objects. *arXiv preprint arXiv:2110.10189*, 2021. 1

[6] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. pages 894–906, 2022. 1

[7] Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6629–6638, 2019. 1

[8] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. pages 1094–1100, 2020. 1

[9] Wentao Yuan, Chris Paxton, Karthik Desingh, and Dieter Fox. Sornet: Spatial object-centric representations for sequential manipulation. pages 148–157, 2022. 2