# Vision-Language-Conditioned Learning Policy for Robotic Manipulation

Sichao Liu[1,2,3,*], Zhihao Liu[1], Lihui Wang[1], and Xi Vincent Wang[1]

[1]KTH Royal Institute of Technology [2]University of Cambridge [3]École Polytechnique Fédérale de Lausanne

**Abstract:** Humans often use natural language instructions to control and interact with robots for task execution. This presents a significant challenge for robots, as they have to not only comprehend and link human commands with robot actions but also have a semantic understanding of operating scenes/environments. To address this challenge, we present a vision-language-conditioned learning policy for robotic manipulation. Given a scene image, we utilise a vision-language model (GPT-4o) to realise a semantic understanding of the scene and its constituent object detection and grounding. Our approach takes a limited set of images to reveal the spatial-temporal relationship of the objects. Then, we develop a GPT-o1-driven approach to performing logic reasoning behind language tasks and high-level control code generation. With the establishment of 6D pose estimates, a language-perception-action method is proposed to link language instructions with robot behaviours for robotic manipulation. The performance of the developed approach is experimentally validated through industrial object manipulation.

**Keywords:** Vision language models, GPT-o1, robot learning, robotic manipulation

## 1 Introduction

Robots capable of understanding human instructions can enhance the performance of robotic interaction and manipulation, particularly in scenarios requiring robotic support for handling unknown objects [1, 2]. However, leveraging natural language commands for robotic manipulation and interactions within uninstructed/open-world environments remains a challenge [3]. For this purpose, robots need to have the capability of not only parsing and comprehending natural language instructions but also realising semantic understanding of unknown interaction environments [4, 5, 6]. Understanding the scene the robot is operating is the first step toward language-guided robotic manipulation [7, 8], but is still an open challenge. The existing methods of scene understanding such as 3D scene graph [9], neural radiance fields (NeRF) [10], Voxel [11], and volumetric scene [12] have been reported in the literature [13]. However, fine-grained description and detection of the robotic scene for downstream tasks remain a challenge, which we aim to address in this study.

In parallel, advances in foundation models have enabled open-world object manipulation given natural language instructions [14, 15, 16, 17]. Specifically, language-conditioned/guided robotic manipulation has been widely explored to bridge natural-language task instructions with robot action [18, 19, 20, 21]. The existing studies mainly focus on conventional table-top object manipulation and open-vocabulary robotic manipulation [22, 23, 24]. However, complex tasks (e.g., assembly) often need multi-stage robotic manipulation and long-horizon context language understanding. Also, robots cannot operate alone within certain environments, and they have to rely on visual and spatial data with language to understand complex commands [25, 26, 27]. To close these limitations,

*Correspondence to Sichao Liu <sicliu@kth.se; Sichao.Liu@mrc-cbu.cam.ac.uk>

8th Conference on Robot Learning (CoRL 2024), Munich, Germany.

vision-language models (VLMs) are utilised to serve as a bridge between visual and linguistic information, facilitating the handling of a wide variety of vision-language tasks [17, 28, 29, 30]. However, these models demonstrate low or limited reasoning capabilities for language-based tasks and often need extensive guidance or additional information for long-horizon complex language tasks [31, 32]. More recently, OpenAI released a preview of GPT-o1—a new series of AI models designed to spend more time thinking before they respond, and it excels in advanced reasoning for complex tasks [33]. Therefore, leveraging the GPT-o1 to solve harder language tasks seems to be a promising solution.

To close the gaps, we present a vision-language-conditioned learning policy for fine-grained scene understanding and robotic manipulation. For a robotic scene, a vision-language model built on GPT-4o is adopted for fine-grained scene segmentation and language-conditioned object detection and grounding. Given language task commands, a GPT-o1-preview-driven learning and reasoning policy is developed to perform control logic reasoning behind language tasks and generate high-level control commands associated with control steps. Upon establishment of the 6D pose, our method links language instructions with robot actions for fine object manipulation and task execution. The effectiveness of the developed method is experimentally validated by a set of tests for industrial object manipulation. The remainder of the paper is organised as follows. Section 2 presents related work. Section 3 presents methods and experimental results. Finally, Section 4 draws conclusions and highlights future work.

## 2 Related work

**Robotic scene representation and understanding** Robotic manipulation in open-world environments demands spatial and semantic understanding of the scene in which a robot is functioning [25, 22]. Specifically, a robot not only needs to detect and localise the objects within the scene for manipulation but also has to know how to grasp the object based on 3D neural representation [34, 35, 36]. We refer readers to [37, 38] for an overview and applications of 3D neural representation. The concept of neural feature field was widely investigated to distil 2D information into a view-consistent 3D field representation [39]. As an example, NeRF has been utilised for neural representation of 3D scenes and also applied for a wide range of robotic tasks (e.g., manipulation) [10, 40], but NeRF-based approaches struggle with high computational cost and long training and rendering time [41, 42, 43]. Gaussian splatting, on the other hand, represents the environment by distributing 3D Gaussian across the scene to model geometry and appearance, combined with a fast rasterisation-based renderer [42, 44]. It can provide fast training and rendering, but sacrifices some accuracy and detail, especially for complex scenes [45, 46]. Unfortunately, the adoption of these rich representations for scenes alone is often inadequate in robotic manipulation. Specifically, to achieve robot manipulation, object pose will have to be integrated with 3D rendering and reconstruction techniques, which enables the robot to detect and localise grasp affordances on the object [47, 48]. Recently, leveraging VLMs coupled with open-set object detection methods (e.g., CLIP [49] and grounding DINO [50]) for zero-shot performance provides fine-grained pixel-level features used for detailed object detection and visual grounding [51, 52]. Furthermore, FoundationPose presents a unified foundation model for 6D object pose estimation and tracking [53], especially for robust pose estimates for novel objects using a small number of reference images. In this study, we leverage their complementary strengths of using VLM+grounding DINO for fine-grained scene representation and object detection and of using FoundationPose for fast and second-level pose estimate, and the unique combination can handle scene understanding with its constituent object detection and 6D pose estimate for object manipulation in a dynamic environment.

**Foundation models for robotics** Foundation models such as CLIP [49] and SAM [54] have been actively explored and utilised in robotics such as robotic manipulation, robot learning, and zero-shot task performance [24, 55, 56, 57]. For example, large language models (LLMs) can provide common-sense reasoning, code generation, decision making and robot control [4], while VLMs enable open-vocabulary visual recognition [58]. However, despite showing promises for high-level task planning, language-guided object manipulation of LLMs/VLMs often limits complicated tasks that often need multi-stage and long-horizon language-condition task execution [59, 60, 61]. A task

is often composed of multi-stage sub-task execution. Each stage task requires planning and decision-making based on the previous step and can be completed by a set of robotic manipulations such as picking and placing [62]. On the other hand, open-world robotic manipulation (e.g., Splat-MOVER ) enables language-guided and multi-stage execution for a simple task [47], but there lacks an effective control logic reasoning behind language tasks. For ambiguous or implicit language tasks, the success of the task execution heavily relies on context-aware reasoning and situational memory of previous stages [63, 64]. The control logic behind the task reflects the thought of how to complete the task step by step, and it is similar to the chain of thought (CoT) that enables complex reasoning capabilities through intermediate reasoning steps [62]. The success of logical reasoning reveals a coherent task plan and decision-making process during each step but requires contextual understanding and comprehending previous actions. More recently, GPT-o1 models, trained via reinforcement learning to consistently employ CoT reasoning, excel in advanced reasoning for complex task solving [33]. In this study, we utilise its advanced reasoning capability to control logic reasoning behind language tasks. Further, language-action models establish a pipeline to link control steps (as intermediate variables of language instructions) with robot actions in a supported output modality (high-level codes).

## 3 Methods

We discuss: 1) How to realise fine-grained scene understanding and its constituent object detection through a VLM? 2) How to use the GPT-o1 model to perform control logic reasoning behind language tasks? 3) How to generate robot actions from visual observation and language instructions?

### 3.1 Scene understanding and fine-grained object detection

VLM-based scene representation processes the image of the scanned scene along language prompt of users to realise semantic and spatial understanding of the scene and its constituent object detection. The language task is parsed from semantics to extract the predicate (actions) and object (components to be acted on). Then, with the support of visual grounding [50] and SAM 2 [65], a detector built on the text-based object is adopted to perform fine-grained object detection.

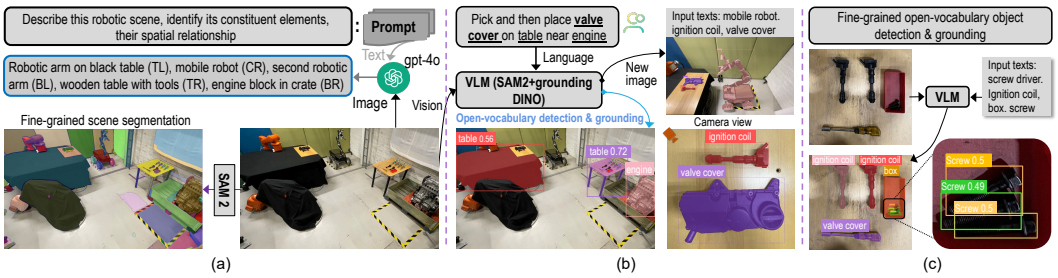


Figure 1: (a) Vision-language-guided robotic scene understanding; (b) Scene segmentation and its constituent object detection; (c) Language-conditioned fine-grained object detection and grounding.

Here, we use GPT-4o [66] which accepts both texts and images as the VLM. The scene image of a customised robotic work cell, as a zero-shot sample, is scanned by an onboard camera and fed into the VLM. A user prompt of '*describe this robotic scene, identify its constituent elements and their spatial relation*' is given to output the text description. Given the long-text output, a semantic-based crop method is developed to simplify the text with a focus on objects. As shown in Fig. 1 (a), the result is 'robotic arm on the black table (TL), mobile robot (CR),

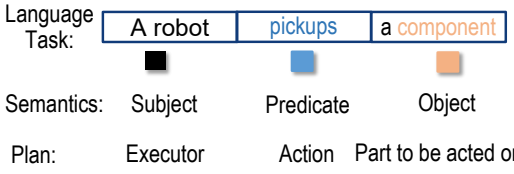


Figure 2: Semantics-based language instruction parsing

second robotic arm (BL), wooden table with tools (TR), engine block in crate (BR)' in a simplified format, in which 'T, B, L, R, C' represent 'top, bottom, left, right, centre', respectively. In addition, we performed scene and its constituent element segmentation using SAM 2 with the input of a raw image (Fig. 1 (a) (right)), and the results (Fig. 1 (a) (left)) show masks of all of the objects with different colours. The masks of the objects will be used for downstream tasks (e.g., pose estimate and object manipulation).

However, detection and description for small-sized objects is insufficient, given the detection results only for major objects. For robotic manipulation (e.g., object grasping), fine-grained and detailed object detection is needed. Given a language task (e.g., '*pickup and then place valve cover on table near engine*) as shown in Fig. 1 (b), a semantics-based command parsing method extracts the predicates (actions) and objects (component to be acted on) of the text as shown in Fig. 2, and they are '*pick-placing*' actions and '*valve cover, table, and engine* as objects'. These words of the objects are defined as input to the open-vocabulary object detection that combines grounding DINO (1.5) and SAM 2 [67]. The objects of two tables and the engine are pinpointed with their 2D spatial locations with bounding boxes and text labels, but the valve cover was not accurately detected given its tiny size such that the robot cannot know where it is and how to pick it up. To address this problem, a fine-grained object detection scheme is employed for small-sized objects through comprehending multiple images from different view fields. As shown in Fig. 1 (b), a new scene image from a different view field within the same workspace is captured to be fused for fine-grained scene description. Here, the texts of '*valve cover*', '*ignition coil*', and '*mobile robot*' are defined as inputs to open-set object detection and grounding. The result shows they are accurately detected with bounding boxes and coloured masks. Through the query to the GPT-4o, the mobile robot can understand spatial relations with the valve cover and localise by comprehending the content of these two scene images. Specifically, the GPT-4o combines information from these two scene images and reasons the spatial relationship among the identified objects within the scene image of Fig. 1 (a). Finally, the GPT-4o pinpoints that the valve cover is placed on the table to the left of the mobile robot. The spatial relation and location of the object are critical to downstream tasks/actions. For example, it can guide the robot to move to the table when executing actions. After the robot docks in front of the table, the camera can detect and identify the valve cover and ignition coil with masks, bounding boxes, and text labels, which are used for robot grasping. This is realised by running algorithms of visual-language-guided object detection (presented in the following section).

We also tested how small the size of the object can be detected as shown in Fig. 1 (c), and the text of the object is '*ignition coil, screwdriver, box, and screws*' works as open vocabularies for object detection. The results show that the physical objects grounded with open vocabularies are correctly identified and masked. Especially, the success of three small screws in the box demonstrates the performance of the model on small-sized and fine-grained object detection.

However, 2D information of the object alone is often inadequate in robotic manipulation, and its 3D representation and pose information is required. For this purpose, we adopted a unified foundation model for 3D representation and 6D pose estimation [53]. It starts with the collection of an RGB-D video stream of novel objects by a depth camera and segments the video into frames and masks by an online video segmentation network. Each frame consists of RGB and depth images. Together with camera parameters, RGB-D images and masks are input to object

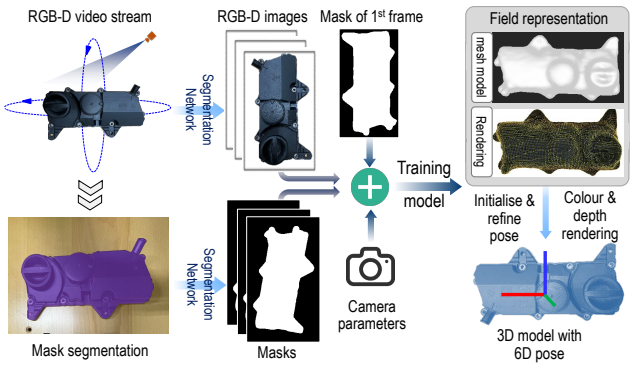


Figure 3: Neural representation and pose estimate of an unknown object (valve cover).

field modelling that learns geometry and appearance information. Once trained, the neural field with colour and depth rendering and marching cubes is used to build and render 3D neural representation of the object built on texture meshes. In parallel, the coarse pose of the current frame is initialised by translation and rotation operations. Then, new frames are iteratively added to an online pose graph refiner to get a fine pose estimate of the object. We use an industrial object as an example to demonstrate this process, and Fig. 3 shows the results of neural representation and pose estimate of the valve cover. The adopted method can be employed for 3D representation and pose estimates for other objects.

### 3.2 Language-conditioned control logic reasoning

Leveraging LLMs (e.g., GPT-4 and its variants) for high-level robot task planning have been reported in [55]. However, the adopted models heavily rely on the prompt design for the desired results, and also different models exhibit different behaviours when given the same prompt. Additionally, they struggle with coherent and consistent comprehension of action-level logic hidden within language tasks [25]. Therefore, given the advanced reasoning capability of GPT-o1, we explore utilising it for control logic reasoning behind language tasks for robotic manipulation, and it is expected to provide a solution to multi-stage and long-horizon language tasks. We start with creating a scenario content that describes robotic working scenarios and robot skill primitives as shown in Fig. 4 (a). It is to allow the model to understand robotic tasks and scenarios, and in-context examples are given to align outputs with expected results. The details of the scenario content are listed in Appendix.A2. Here, we designed a closed-loop reasoning pipeline to evaluate and analyse the feasibility and consistency of control logic reasoning. It adopts a reasoning-validation-generation policy for consistent and robust outputs. In the reasoning stage, a text-based task is given as an input through "*QUERY*", and the command of '*model = 'o1-preview'*' is run for output of the control logic of the given language task. Specifically, how many control steps needed for this task will be analysed and generated step by step. Before direct output, they are defined as inputs into the reverse reasoning module for validation and evaluation. All of these control logic steps are analysed by step-by-step action simulation, and it is to test whether a robot performing these control steps can finish the targeted object manipulation and then give an evaluation. Finally, the control logic steps with validation will be generated. Subsequently, high-level control codes are generated by in-context example learning, in which the object of text-based control steps is segmented and given to variables of high-level control codes. Compared with existing methods (e.g., code as policy [68]), the proposed approach only needs brief and clear instructions without the need for extensive guidance (e.g., CoT prompts).

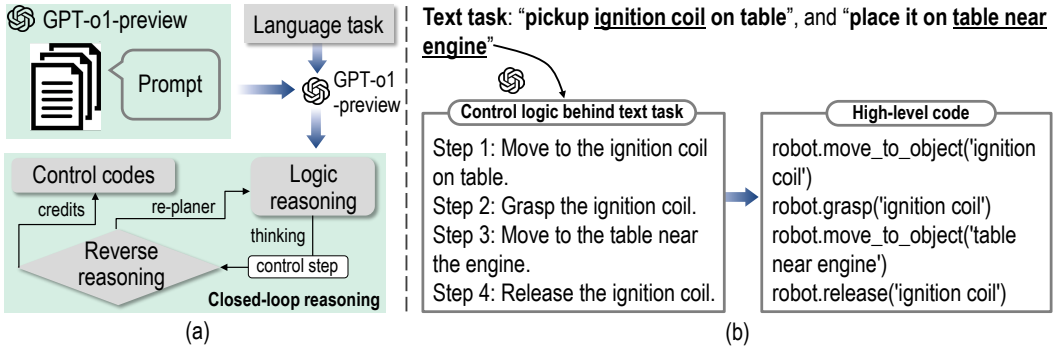


Figure 4: (a) GPT-o1-driven control logic reasoning behind language task; (b) Linking control steps with high-level code generation.

As an example, the language command is given by *pickup ignition coil on table*' and '*place it on the table near engine*' or similar texts as shown in Fig. 4 (b). Firstly, the GPT-o1-preview performs control logic reasoning. The words used for open-vocabulary object indexing and detection are extracted from a subject-predicate-object semantic perspective. Here, we assume that the actions are always performed by a robot. Then, the predicate and object for open vocabularies are '*pickup-*

*place*' actions and '*ignition coil*' (component to be acted on), respectively. Here, the object index of '*engine*' is to distinguish two tables in the robotic scene as shown in Fig. 1. The control logic behind the language task is reasoned and output the step numbers as well as specifications of each step, and then they are verified and validated step by step with an evaluation. Finally, the final output is shown in the block of "Control logic behind text task", and 4 control steps are involved. These control steps are then defined as inputs to generate high-level control codes. Each control step is represented in the form of texts and is composed of predicates and objects. Similarly, the semantics-based instruction parsing method is adopted to extract predicate and object indexing, which is used for determining robot actions from skill primitives and objects to be acted on. Specifically, for control step 1 (*move to the ignition coil on table*), the action that determines the code type is a robot motion, and the object that the robot approaches is the '*ignition coil*'. Finally, the code is represented as '`robot.move_to_object('ignition coil')`. Conversely, the robot action of control step 2 is the '*picking-up*' operation, and the code is '`robot.grasp('ignition coil')`'. All of the high-level control codes are shown in the "High-level code" block. The GPT-o1-preview takes 11 seconds for comprehending the prompt instruction and 12 seconds for result output through the reasoning-validation-generation process. These high-level control codes cannot work alone and need to be integrated with low-level robot actions through a language-perception-action loop, which will be discussed in the next section.

We use a set of language tasks for three objects (ignition coil, valve cover, and screwdriver) to test the consistency and performance of using GPT-o1-preview for task reasoning. These language tasks are defined within the robotic scene and are represented by implicit and explicit instructions. An explicit instruction defines a concise and complete language task (e.g., '*pickup valve cover on the black table*'). While an implicit instruction represents an ambiguous or incomplete language task. As an example, '*pickup and then place ignition coil on the table near engine*' has an ambiguous definition of where the ignition coil is. The control logic reasoning behind such language tasks may rely on either context-aware guidance or additional contexts. For this purpose, we add a query mechanism to allow the LLM to ask for more information if having an implicit representation. For example, the LLM can ask where the ignition coil is for the pick-up operation, which is to make implicit instructions explicit. We start with implicit language tasks without and with query, and then with explicit tasks. "Reasoning success" measures whether the generated logic control steps are feasible for the robot to execute, which tests both implicit and explicit instructions. Task success rate is reported in Table 1. We can observe that the GPT-o1-preview shows a low-level reasoning success on implicit language tasks, due to the absence of context-aware information and key feature information (e.g., object's location). Adding the query can obviously improve the accuracy of the control logic reasoning, where the GPT-o1-review can ask for additional information guidance. For explicit tasks, the GPT-o1 demonstrates good performance on the control logic reasoning. It should be noted that weekly rate limits are 30 messages for o1-preview and 50 for o1-mini [33], so that we test with a limited number of language tasks and more tests can be completed if more messages are allowed. The details of language tasks are listed in Appendix.A3.

Table 1: Success rate of logic reasoning behind language tasks using GPT-o1-preview (w/wo: with/without query)

| Object | Ignition coil | Valve cover | Screwdriver |
|---|---|---|---|
| Implicit task (wo) | 2/5 | 3/5 | 1/5 |
| Implicit task (w) | 5/5 | 4/5 | 4/5 |
| Explicit task | 5/5 | 5/5 | 5/5 |

### 3.3 Link language tasks with robot actions

The high-level control codes in Subsection 3.2 cannot be directly employed on low-level controllers for robotic manipulation. For this purpose, we present a language-perception-action learning method for fine manipulation of objects as shown in Fig. 5. It consists of three core modules, namely language instructions, active perception, and robot actions.

Given the generated control steps, each control step is represented as a text command that is composed of predicates and objects from the semantics perspective. As shown in Fig. 6, the text command is tokenised into prompt tokens, and they are defined text and object tokens. The object tokens define and represent objects to be acted on. For example, given the text command of '*pickup ignition coil on table*', it is tokenised into each token that represents each word of the text instructions. The text token ('*pickup*') is associated with the grasping action selected from robot skill primitives, and the object tokens of ("*ignition coil*') are extracted to be acted on. They are input to the LLM for action token generation.

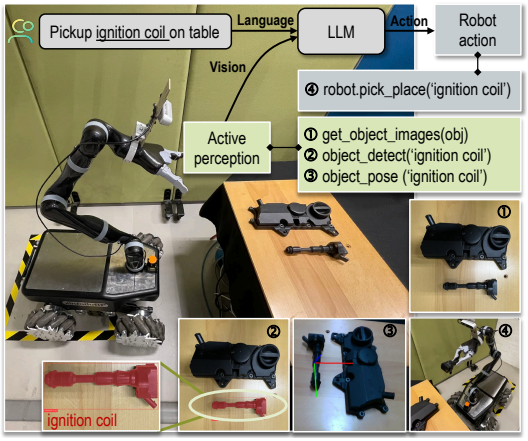


Figure 5: A language-perception-action pipeline for fine manipulation of objects.

In parallel, as shown in Fig. 5, the vision perception module is to capture scene images, and the robot is equipped with an eye-in-hand camera. The camera is initialised to collect RGB-D video streams via a function of `get_scene_images(obj)`, and the online video segmentation network developed in Subsection 3.1 is employed to segment the video into scene image frames. The obtained scene image is shown in the inset ①. Then, open-vocabulary object detection and grounding are employed to identify the object. The segmented object ('`ignition coil`') from the text command is given to the variable of '`object`' to the function of `object_detect('object')`. It is `object_detect('ignition coil')` and used for fine-grained object detection. Specifically, it calls the method developed in Subsection 3.1 to segment and identify the ignition coil within the scene image. It is masked and localised with a bounding box and a text label as shown in the inset ②. To manipulate the object, the 3D representation of the object as shown in Fig. 2 is overlaid on the 2D image, followed by the pose estimate for the downstream task of robot actions. This relies on the established neural representation of the object in Subsection 3.1 and is realised by a function of `object_pose('ignition coil')`. The pose estimate and grasping affordance of the ignition coil are shown in the inset ③. Given the language and visual information, the robot action for fine object manipulation on the low-level controller is generated from the LLM as shown in Fig. 6, which is represented as an action token. Specifically, the control function of `robot_action('object')` instructs the robot to grasp the object of '`ignition coil`' based on its pose estimate (position and orientation) in the robotic coordinate system (as shown in the inset ③), and the low-level motion planner generates the robot paths that perform a top-down object grasping as shown in the inset ④ and Fig. 6. After completing the robot action, the action token is generated to work as a historical token that is input to the next language-conditioned control step (e.g., move to table near the engine).

Given the language task in Fig. 4, the next control step for the robot is to move to the table near the engine. It is realised by the high-level control code of `robot.move_to_object('table near engine')`. After picking up the ignition coil, the motion planner built on the SLAM map generates robot paths to navigate the robot with the object to the table near the engine. The technical details of building the SLAM map can be found in our previous work [2], which is not a focus of this study. The robot arm is subsequently controlled to release the griper for placing the ignition coil on the table near the engine.

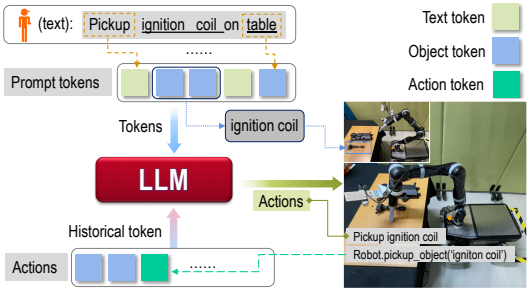


Figure 6: Takenisation from prompt to robot actions.

We use a set of tests to validate the robustness and performance of the proposed approach that links text-based control steps to real robot actions. The tests consist of picking up the ignition coil and valve cover, respectively. Each is measured with ten trails. The difference between the trails is the position of the objects. The results are reported in Table 2. We can observe that the execution of the text-based control steps and high-level control codes shows an acceptable success rate. The failures of picking up the ignition coil lie in 1) out of the workspace of the robot arm; 2) grasping affordance pinpointing the far-end of the ignition coil (not centre of gravity); 3) gripper collision with the table. For the valve cover, the failure is the selection of the grasping point.

Table 2: Results of executing object manipulation by text-based control steps

| Object | Ignition coil | Valve cover |
|---|---|---|
| Results | 7/10 | 9/10 |

## 4 Conclusions and future work

We present a vision-language-conditioned learning policy for robotic manipulation. Given robotic scene images and language tasks, we use GPT-4o as a VLM for robotic scene understanding and its constituent object detection and grounding. It can combine multiple scene images to reason and predict spatial relationships between the objects as well as fine-grained detection of small-sized objects. Then, we employ GPT-o1-preview to perform control logic reasoning behind language tasks and generate high-level control codes, in which a reasoning-validation-generation scheme is developed to provide consistent and feasible control steps. Finally, we present a language-perception-action approach to linking language instructions with robot actions for industrial object manipulation. However, the 'thinking' process of the GPT-o1-preview during logic reasoning behind language tasks often takes time and this can limit wide applications of our methods in robotics, especially for scenarios requiring real-time interactions and decision making. Our future work will focus on how to make GPT-o1 timely respond to language tasks and also include reverse retrieval and action-contextual reasoning and replanning.

**Acknowledgments**

The authors acknowledge financial support from the Swedish Research Council (Vetenskapsrådet) under award 2023-00493, National Academic Infrastructure for Supercomputing in Sweden (NAISS) under award 2024/5-164, National Supercomputer Centre (NSC) Berzelius-2024-124. For the purpose of open access, the author has applied a Creative Commons Attribution (CC BY) licence to any Author Accepted Manuscript version arising from this submission.

## References

[1] D. Shah, B. Osiński, S. Levine, et al. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *Conference on robot learning*, pages 492–504. PMLR, 2023.

[2] S. Liu, J. Zhang, L. Wang, and R. X. Gao. Vision ai-based human-robot collaborative assembly driven by autonomous robots. *CIRP Annals*, 2024.

[3] G. Tang, S. Rajkumar, Y. Zhou, H. R. Walke, S. Levine, and K. Fang. Kalie: Fine-tuning vision-language models for open-world manipulation without robot data. *arXiv preprint arXiv:2409.14066*, 2024.

[4] J. Wang, Z. Wu, Y. Li, H. Jiang, P. Shu, E. Shi, H. Hu, C. Ma, Y. Liu, X. Wang, et al. Large language models for robotics: Opportunities, challenges, and perspectives. *arXiv preprint arXiv:2401.04334*, 2024.

[5] L. Nwankwo and E. Rueckert. The conversation is the command: Interacting with real-world autonomous robots through natural language. In *Companion of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, pages 808–812, 2024.

[6] S. Liu, J. Zhang, R. X. Gao, X. V. Wang, and L. Wang. Vision-language model-driven scene understanding and robotic object manipulation. In *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*, pages 1–6. IEEE, 2024.

[7] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[8] W. Chen, S. Hu, R. Talak, and L. Carlone. Leveraging large language models for robot 3d scene understanding. *arXiv preprint arXiv:2209.05629*, 2022.

[9] I. Armeni, Z.-Y. He, J. Gwak, A. R. Zamir, M. Fischer, J. Malik, and S. Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5664–5673, 2019.

[10] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[11] B.-s. Kim, P. Kohli, and S. Savarese. 3d scene understanding by voxel-crf. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1425–1432, 2013.

[12] S. Peng, K. Genova, C. Jiang, A. Tagliasacchi, M. Pollefeys, T. Funkhouser, et al. Openscene: 3d scene understanding with open vocabularies. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 815–824, 2023.

[13] J. Fan, P. Zheng, and S. Li. Vision-based holistic scene understanding towards proactive human–robot collaboration. *Robotics and Computer-Integrated Manufacturing*, 75:102304, 2022.

[14] A. Stone, T. Xiao, Y. Lu, K. Gopalakrishnan, K.-H. Lee, Q. Vuong, P. Wohlhart, S. Kirmani, B. Zitkovich, F. Xia, et al. Open-world object manipulation using pre-trained vision-language models. *arXiv preprint arXiv:2303.00905*, 2023.

[15] G. Tziafas and H. Kasaei. Towards open-world grasping with large vision-language models. *arXiv preprint arXiv:2406.18722*, 2024.

[16] H. Xiong, R. Mendonca, K. Shaw, and D. Pathak. Adaptive mobile manipulation for articulated objects in the open world. *arXiv preprint arXiv:2401.14403*, 2024.

[17] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[18] T. Ma, J. Zhou, Z. Wang, R. Qiu, and J. Liang. Contrastive imitation learning for language-guided multi-task robotic manipulation. *arXiv preprint arXiv:2406.09738*, 2024.

[19] E. Triantafyllidis, F. Christianos, and Z. Li. Intrinsic language-guided exploration for complex long-horizon robotic manipulation tasks. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7493–7500. IEEE, 2024.

[20] H. Ha, P. Florence, and S. Song. Scaling up and distilling down: Language-guided robot skill acquisition. In *Conference on Robot Learning*, pages 3766–3777. PMLR, 2023.

[21] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022.

[22] F. Liu, K. Fang, P. Abbeel, and S. Levine. Moka: Open-vocabulary robotic manipulation through mark-based visual prompting. *arXiv preprint arXiv:2403.03174*, 2024.

[23] K. Ehsani, W. Han, A. Herrasti, E. VanderBilt, L. Weihs, E. Kolve, A. Kembhavi, and R. Mottaghi. Manipulathor: A framework for visual object manipulation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4497–4506, 2021.

[24] K. Kawaharazuka, T. Matsushima, A. Gambardella, J. Guo, C. Paxton, and A. Zeng. Real-world robot applications of foundation models: A review. *arXiv preprint arXiv:2402.05741*, 2024.

[25] W. Huang, C. Wang, Y. Li, R. Zhang, and L. Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*, 2024.

[26] C. Huang, O. Mees, A. Zeng, and W. Burgard. Visual language maps for robot navigation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10608–10615. IEEE, 2023.

[27] J. X. Liu, A. Shah, G. Konidaris, S. Tellex, and D. Paulius. Lang2ltl-2: Grounding spatiotemporal navigation commands using large language and vision-language models. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024.

[28] J. Zhang, J. Huang, S. Jin, and S. Lu. Vision-language models for vision tasks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[29] X. Li, Y. Fang, M. Liu, Z. Ling, Z. Tu, and H. Su. Distilling large vision-language model with out-of-distribution generalizability. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2492–2503, 2023.

[30] J. Bai, S. Bai, S. Yang, S. Wang, S. Tan, P. Wang, J. Lin, C. Zhou, and J. Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. 2023.

[31] F. Zeng, W. Gan, Y. Wang, N. Liu, and P. S. Yu. Large language models for robotics: A survey. *arXiv preprint arXiv:2311.07226*, 2023.

[32] H. Zhou, X. Yao, Y. Meng, S. Sun, Z. BIng, K. Huang, and A. Knoll. Language-conditioned learning for robotic manipulation: A survey. *arXiv preprint arXiv:2312.10807*, 2023.

[33] OpenAI. Openai o1 system card. https://openai.com/index/openai-o1-system-card, 2024. Accessed: 2024-09-12.

[34] W. Yuan, J. Duan, V. Blukis, W. Pumacay, R. Krishna, A. Murali, A. Mousavian, and D. Fox. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv preprint arXiv:2406.10721*, 2024.

[35] Y. Li and D. Pathak. Object-aware gaussian splatting for robotic manipulation. In *ICRA 2024 Workshop on 3D Visual Representations for Robot Manipulation*.

[36] M. Ji, R.-Z. Qiu, X. Zou, and X. Wang. Graspsplats: Efficient manipulation with 3d feature splatting. *arXiv preprint arXiv:2409.02084*, 2024.

[37] V. Sitzmann, M. Zollhöfer, and G. Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *Advances in Neural Information Processing Systems*, 32, 2019.

[38] K. Gao, Y. Gao, H. He, D. Lu, L. Xu, and J. Li. Nerf: Neural radiance field in 3d vision, a comprehensive review. *arXiv preprint arXiv:2210.00379*, 2022.

[39] V. Tschernezki, I. Laina, D. Larlus, and A. Vedaldi. Neural feature fusion fields: 3d distillation of self-supervised 2d image representations. In *2022 International Conference on 3D Vision (3DV)*, pages 443–453. IEEE, 2022.

[40] J. Zhang, S. Liu, R. X. Gao, and L. Wang. Neural rendering-enabled 3d modeling for rapid digitization of in-service products. *CIRP annals*, 72(1):93–96, 2023.

[41] Y. Guo, Y. Bai, L. Hu, Z. Guo, M. Liu, Y. Cai, T. Huang, and L. Ma. Prtgs: Precomputed radiance transfer of gaussian splats for real-time high-quality relighting. *arXiv preprint arXiv:2408.03538*, 2024.

[42] Y. Wu, J. Liu, and S. Ji. 3d gaussian splatting for large-scale 3d surface reconstruction from aerial images. *arXiv preprint arXiv:2409.00381*, 2024.

[43] C. Lindström, G. Hess, A. Lilja, M. Fatemi, L. Hammarstrand, C. Petersson, and L. Svensson. Are nerfs ready for autonomous driving? towards closing the real-to-simulation gap. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4461–4471, 2024.

[44] Y. Weng, Z. Shen, R. Chen, Q. Wang, and J. Wang. Eadeblur-gs: Event assisted 3d deblur reconstruction with gaussian splatting. *arXiv preprint arXiv:2407.13520*, 2024.

[45] G. Chen and W. Wang. A survey on 3d gaussian splatting. *arXiv preprint arXiv:2401.03890*, 2024.

[46] Y. Li, C. Lyu, Y. Di, G. Zhai, G. H. Lee, and F. Tombari. Geogaussian: Geometry-aware gaussian splatting for scene rendering. *arXiv preprint arXiv:2403.11324*, 2024.

[47] O. Shorinwa, J. Tucker, A. Smith, A. Swann, T. Chen, R. Firoozi, M. Kennedy III, and M. Schwager. Splat-mover: Multi-stage, open-vocabulary robotic manipulation via editable gaussian splatting. *arXiv preprint arXiv:2405.04378*, 2024.

[48] Q. Yu, C. Hao, J. Wang, W. Liu, L. Liu, Y. Mu, Y. You, H. Yan, and C. Lu. Manipose: A comprehensive benchmark for pose-aware object manipulation in robotics. *arXiv preprint arXiv:2403.13365*, 2024.

[49] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[50] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

[51] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, R. Howes, P.-Y. Huang, H. Xu, V. Sharma, S.-W. Li, W. Galuba, M. Rabbat, M. Assran, N. Ballas, G. Synnaeve, I. Misra, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.

[52] R. Xu, Z. Huang, T. Wang, Y. Chen, J. Pang, and D. Lin. Vlm-grounder: A vlm agent for zero-shot 3d visual grounding. In *8th Annual Conference on Robot Learning*.

[53] B. Wen, W. Yang, J. Kautz, and S. Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17868–17879, 2024.

[54] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick. Segment anything. *arXiv:2304.02643*, 2023.

[55] R. Firoozi, J. Tucker, S. Tian, A. Majumdar, J. Sun, W. Liu, Y. Zhu, S. Song, A. Kapoor, K. Hausman, et al. Foundation models in robotics: Applications, challenges, and the future. *The International Journal of Robotics Research*, page 02783649241281508, 2023.

[56] X. Xiao, J. Liu, Z. Wang, Y. Zhou, Y. Qi, Q. Cheng, B. He, and S. Jiang. Robot learning in the era of foundation models: A survey. *arXiv preprint arXiv:2311.14379*, 2023.

[57] Y. Cui, S. Niekum, A. Gupta, V. Kumar, and A. Rajeswaran. Can foundation models perform zero-shot task specification for robot manipulation? In *Learning for dynamics and control conference*, pages 893–905. PMLR, 2022.

[58] W. Kuo, Y. Cui, X. Gu, A. Piergiovanni, and A. Angelova. F-vlm: Open-vocabulary object detection upon frozen vision and language models. *arXiv preprint arXiv:2209.15639*, 2022.

[59] M. Dalal, T. Chiruvolu, D. Chaplot, and R. Salakhutdinov. Plan-seq-learn: Language model guided rl for solving long horizon robotics tasks. *arXiv preprint arXiv:2405.01534*, 2024.

[60] Y. Wu, J. Zhang, N. Hu, L. Tang, G. Qi, J. Shao, J. Ren, and W. Song. Mldt: Multi-level decomposition for complex long-horizon robotic task planning with open-source large language model. *arXiv preprint arXiv:2403.18760*, 2024.

[61] A. Mei, G.-N. Zhu, H. Zhang, and Z. Gan. Replanvlm: Replanning robotic tasks with visual language models. *arXiv preprint arXiv:2407.21762*, 2024.

[62] S. S. Kannan, V. L. Venkatesh, and B.-C. Min. Smart-llm: Smart multi-agent robot task planning using large language models. *arXiv preprint arXiv:2309.10062*, 2023.

[63] B. Kim, J. Kim, Y. Kim, C. Min, and J. Choi. Context-aware planning and environment-aware memory for instruction following embodied agents. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10936–10946, 2023.

[64] O. Barbany, M. Huang, X. Zhu, and A. Dhua. Leveraging large language models for multi-modal search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1201–1210, 2024.

[65] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.

[66] OpenAI. Gpt-4 technical report. https://openai.com/research/gpt-4, 2023. Accessed: 2024-09-12.

[67] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024.

[68] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE, 2023.

# Appendix

### A1. Object detection and localisation using GPT-4o

Instructions of using GPT-4o for fine-grained object detection and localisation from multiple scene images is shown below. Given a language task and a scene image, the GPT-4o performs scene understanding and its constituent object detection. However, the small-sized object cannot be detected, and another image within the same robotic scene from a different view is combined to detect and identify fine-grained object detection.

**Image:** A scene image of Figure 1 (a) (left).
**Prompt:** Please describe this robotic scene, identify its constituent elements, and their spatial relationship.
**GPT-4o:** This scene appears to be in a laboratory or workshop setting focused on robotics and possibly industrial automation.

**Constituent elements:** 1) robotic arm (top left on the table); 2) mobile robot (center-right); 3) robotic arm (bottom left); 4) wooden with tools (top right); 5) engine block in carte (bottom right).
**Spatial relationships:**

- The robotic arm on the left (orange, uncovered) is placed on a table, likely set up for manipulation tasks.
- The mobile robot occupies the centre-right of the image, in its own designated square. It is placed near the wooden table.
- The partially covered robotic arm is positioned away from the other active workspace
- The engine block is placed in its own area, indicating that it may either be an object to be handled by the robots.

**Image:** A raw scene image of Figure 1 (b) (top-right).
**Prompt:** This robotic scene is a part of the workspace from a different view. Please tell me which table you should go if you need to pickup the valve cover.
**GPT-4o:** In this new view of the workspace, the **valve cover** is placed on the **wooden table** to the left of the mobile robot. If you need to pick up the valve cover, you should go to the wooden table on the left side of the image, where the valve cover and another part (possibly a camshaft) are laid out. This is likely the same table seen in the previous image, next to the robotic arm and covered in black fabric.

### A2. GPT-o1-preview for control logic reasoning behind language tasks

We observe that the model exhibits slightly different behaviours when given the same query instruction in the specific text prediction, especially for implicit language tasks. However, the meaning of the text output is with high similarity despite slight differences in the terms of the generated texts. Here, we focus on the demonstration of a pipeline of using GPT-o1-preview for logic reasoning behind language tasks, rather than design the best prompt for the language tasks in this study. The pipeline presents key blocks and can be further improved by future study along arising the latest models. The details of the instructions of using GPT-o1-preview for control logic reasoning behind language tasks are listed below.

```
### Instructions
Suppose you are controlling a robot to perform manipulation tasks by writing control codes in Python,
The manipulation task is given as a text instruction, and you have to reason the control logic behind the given text instruction.
Before outputting the results, you need to verify and validate whether the generated control steps are feasible and sufficient to complete the language task, and then output what you think for verification and evaluation as metrics. Finally, you need to generate high-level control code for each control step in Python.
For each given task, please perform the following steps:
Determine how many steps are involved in the language task. Grasping must be an independent stage.
Some examples:
    - "pickup and place one object on location":
    - 4 steps: "move to object" "grasp object", "move to location", and "release object"
If you do not know where the object is for handling, and you can ask through query.
Generate high-level codes for each control step. Some examples:
    - control step: "move to shelf near wall" or  "move to shelf" or "move to apple on shelf"
    - code: robot.move_to_object('shelf')
    - control step: "pickup apple" or similar
    - code: robot.grasp('apple')

Structure your output in a single python code block as follows for a mobile robot:
```python
# Explain how many steps are involved in the task and what each step is about.
# ...
num_steps=?

### Control step 1
def control_step(control_logic, num)
    """Put your explanation here for reasoning."""
     ...
    control_step_{num}="control_logic"

# repeat for more control steps
...
```
### Reverse reasoning for validation and evaluation
def validation_control_step(control_step)
    """Put your explanation here for validation."""
     ...
    return cost
# repeat for more control steps
...
```
### Output python code block for each control step
def control_code(control_step)
    """Put high-level control code associated with each control step here."""
     ...
    return code

## Query
Query Task: "[INSTRUCTION]"
```
```

### A3. Language tasks for logic reasoning

Here, we list implicit and explicit instructions mentioned in Subsection 3.2 as follows:

**Implicit instructions**: 1) pickup and then place `object` on the table; 2) handover `object`; 3) give me `object`; 4) pickup `object` and move to the table; 5) grasp two `objects` on the table.

**Explicit instructions**: 1) pickup `object` on the table and then place it on the table near engine; 2) handover `object` on the table near engine; 3) give me `object` on the table to left of the mobile robot; 4) pickup `object` on the table with black cover and move to the engine; 5) grasp two `object` on the table near engine. Here, the `object` represents 'ignition coil, valve cover, and screwdriver', respectively.