2024 IEEE 20th International Conference on Automation
Science and Engineering (CASE)
August 28 - September 1, 2024. Bari, Italy

Vision-language model-driven scene understanding and robotic object manipulation

Sichao Liu[1,3,4,*] Jianjing Zhang[2] Robert X. Gao[2] Xi Vincent Wang[1] and Lihui Wang[1]

[1]Department of Production Engineering, KTH, Sweden. [2]Department of Mechanical and Aerospace Engineering, Case Western Reserve University, USA. [3]Cognition and Brain Sciences Unit, University of Cambridge, UK. [4]Institute of Bioengineering, Swiss Federal School of Technology in Lausanne, Switzerland. sicliu@kth.se

Abstract—Humans often use natural language instructions to control and interact with robots for task execution. This poses a big challenge to robots that need to not only parse and understand human instructions but also realise semantic understanding of an unknown environment and its constituent elements. To address this challenge, this study presents a visionlanguage model (VLM)-driven approach to scene understanding of an unknown environment to enable robotic object manipulation. Given language instructions, a pre-tained visionlanguage model built on open-sourced Llama2-chat (7B) as the language model backbone is adopted for image description and scene understanding, which translates visual information into text descriptions of the scene. Next, a zero-shot-based approach to fine-grained visual grounding and object detection is developed to extract and localise objects of interest from the scene task. Upon 3D reconstruction and pose estimate establishment of the object, a code-writing large language model (LLM) is adopted to generate high-level control codes and link language instructions with robot actions for downstream tasks. The performance of the developed approach is experimentally validated through table-top object manipulation by a robot.

I. INTRODUCTION

Humans often instruct robots to assist in collaborative tasks, where language instructions are a promising manner to realise natural interactions with robots [1]. However, the use of natural language instructions in robot control and interactions with unknown environments remains a challenge. For this purpose, robots need to have the capability of not only parsing natural language instructions but also semantic understanding of unknown interaction environments [2]. A simple natural language instruction issued by humans is built on the understanding of working environments and cognitive reasoning of operation tasks [3]. However, the robot does not initially have such capabilities such as natural language processing and semantic understanding [4]. Thanks to the advancement of vision techniques, the combination of visual systems with artificial intelligence (AI) algorithms enables environmental perception, object recognition and manipulation, and fusing the visual perception and natural language description enables robots with enhanced capabilities in task execution. As an example, applications of neural radiance fields (NeRFs) in visual-based robotic manipulation have been investigated to realise 3D reconstruction of physical environments [5]. Upon visual representation establishment of objects, NLP algorithms can comprehend language-based instructions and facilitate the downstream tasks with the support of visual information [6].

In recent years, the emergence of large language models (LLMs) such as BERT [7], Llama [8], GPT-4 [9], and Gemma [10] has demonstrated notable performance and achievements in the field of generative AI and robotic applications. Built on transformer architectures, LLMs are trained on massive amounts of datasets, which allows them to generate high-quality and comprehensive language text [11]. More recently, various applications of leveraging LLMs in NLP tasks, cognitive reasoning, decision making and robot control have been reported [12]. For example, leveraging LLMs to facilitate human-robot interactions (HRIs), robot task planning, code generation, and text parsing has been reported in the literature [13][14][15]. Robotists demand natural HRIs and seamless collaborative task execution, given the broad deployment of language models. The fusion of LLMs and robotics can unlock new opportunities to enable robots to have human-like capabilities of NLP and text generation [16]. In addition, massive visual data are included in the training dataset of foundation models, and the emergence of vision-language models (VLMs) can interpret a mixture of visual and language inputs [17][18], and these pre-trained VLMs act as the bridge between visual and textual information, enabling handling a wide range of visionlanguage tasks. Additionally, the use of pre-trained language models for scene understanding of household objects was investigated [19]. However, semantic scene understanding is a problem of paramount importance for robotic manipulation, and robots still lack common-sense knowledge of objects among manipulation tasks.

979-8-3503-5851-3/24/$31.00 ©2024 IEEE

Scene understanding is of critical importance in robotics, especially autonomous robotic systems and interaction control, and it refers to context extraction from visual data [20]. Given language instructions, it facilitates robots have semantic understanding of the scene and its elements (or objects) and then provides a base for downstream tasks such as object localisation and manipulation. Various approaches to facilitate scene understanding in HRIs, autonomous navigation and component recognition have been reported in the literature [21]. For example, a simple task of a robot is to correctly enumerate how many objects are in the scene and segment them from the background without prior knowledge [21]. However, scene understanding of more complex activities is still a challenging task that requires retrieving contextual information from the scene, e.g., objects, events, or concepts. For this purpose, research efforts on a complete semanticlevel description of the scene [22], 3D scene understanding [23], and the spatial relationship of objects in a scene were explored [24]. Most of the existing approaches rely on semantic segmentation from 2D/3D visual information and also require high computational efforts. Few studies have investigated the textual description of the scenes by highlighting critical objects but with the need for fewer computational resources.

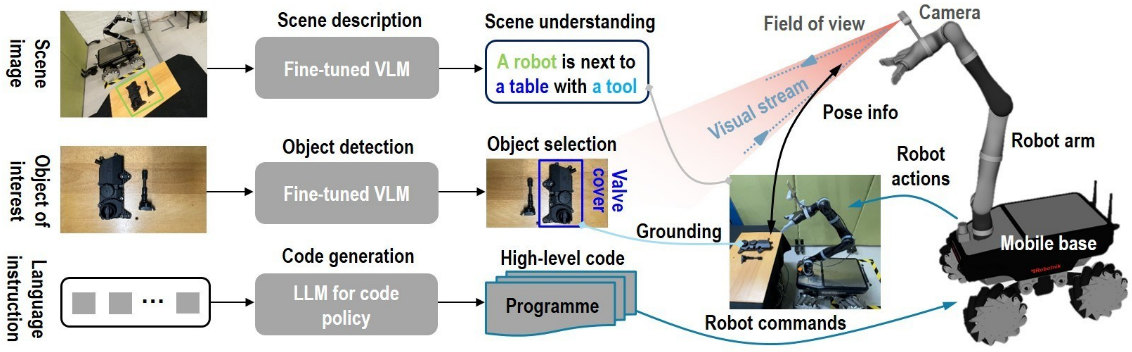


Fig. 1. Overview of VLM-driven scene understanding and robotic object manipulation.

To close the gaps, this study presents a pre-trained visionlanguage model-driven approach to scene understanding and robotic object manipulation. As shown in Fig. 1, visual information of the target scene is fed into a pre-trained VLM built on Llama2 (7B) that is trained on publicly available data, to build a semantic understanding of the scene, and it includes text representation of the scene and its coarse detection of the objects. In parallel, a zero-shot-based approach to fine-grained visual grounding from complex scene tasks is developed for object detection with its location representation by a bounding box with text labels. Upon the detected 2D object, a 3D reconstruction of the object with pose estimates where the details can be found in our previous work [2] is overlaid on the 2D object and defined as control input of downstream tasks. Then, language instructions of tasks are fed into a code-writing LLM to generate highlevel control code for object manipulation. Finally, the visual results of the objects are assigned to the variables of these codes for control action execution.

The remainder of the paper is organised as follows. Section 2 presents the problem statement and methods. Section 3 introduces pre-trained VLM-driven scene understanding and object grounding, and Section 4 links natural language with robot actions via a code-writing LLM, followed by experimental validation. Finally, Section 5 draws conclusions and highlights future work.

II. Problem Statement and Methods

A. Problem definition

As shown in Fig. 2, an NLP task (*NLPT*) to robots is formulated as *NLPT* ={*T*,*S*,*P*,*O*}, where *T* is the textual content that can be a sentence or a phrase. *S*, *P*, and *O* represent a subject (executor), a predicate (action) and an object (component to be acted on) of *T*, respectively. By adding visual information, textual information of *O* is processed and grounded into a real-world object by a pretrained VLM for object detection. However, 2D representation of the object detection cannot support robotic object manipulation (e.g., grasping) that needs object's 3D model and pose information. In parallel, having robots to parse and understanding language instruction for downstream task (e.g., robot control) remains a challenge.

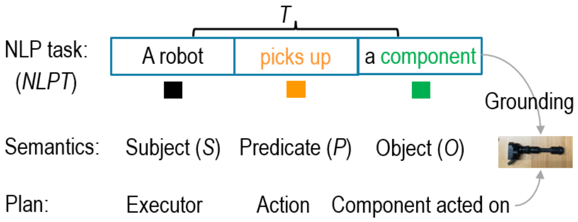


Fig. 2. Semantic segmentation of a natural language task.

Within such a context, the research questions explored in this study are summarised as follows:

- How to use a VLM for zero-shot sample-based text description and semantic understanding of an unknown scene?
- How to link visual grounding and fine-grained object detection with additional 3D reconstruction for downstream tasks (e.g., object manipulation)?
- How to parse language instructions and generate highlevel control codes for robot actions and manipulation?

B. Architecture of LLM built on Llama2

Fig. 3 presents an architecture of a fine-tuned LLM built on open-sourced Llama2 (7B) for scene understanding and object description. It adopts the same architecture of MiniGPT-v2 [25], and it demonstrated better performance on handling various vision-language tasks, compared with LLMs with similar-level parameters including Flamingo9B, MiniGPT-4 (13B), BLIP-2 (13B), InstructBLIP (13B), LLaVA (13B) and Shikra (13B). The model takes a vision transformer (ViT) [26] and a querying transformer (Q-Former) [27] visual backbone, which remains frozen during all training phases. Adjacent visual output tokens from a ViT backbone are concatenated and projected into the Llama2 language model space via a linear projection layer. Finally, Llama2 language tokens are directly utilised to handle vision-language tasks such as image recognition and object grounding. The details of the adopted architecture can be found in [25].

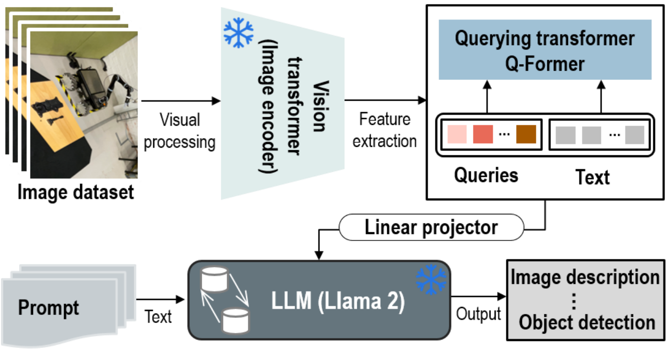


Fig. 3. Architecture of VLM for scene description and object detection (adapted and modified from [25]).

C. Method introduction

Two-system approach: A two-system approach is designed for scene description with a full-view image and object detection with an object-of-interest image. A collection of image datasets is fed into a ViT-based visual encoder to generate a two-dimensional grid of token vectors, and subsequently flatten it to create a one-dimensional sequence [26]. As image resolution increases, the number of visual tokens also grows significantly. By using MiniGPT-v2 for object detection, large objects can be accurately detected and identified, while small objects are often resorted to the whole description of the environment or the image. This means that it cannot accurately recognise small objects within a multi-object complex scene. Given these characteristics, a two-system approach is therefore developed. For scene understanding, it uses a full-view scene image with all of the elements to generate a scene description by text. For fine-grained object manipulation, a scene image is segmented into a set of grid images, and the image with objects of interest is used to accurately detect and ground the object and then provide visual and location information to robots for manipulation. Most of object manipulation is for table-top tasks, and their images token from a top-view or eye-in-hand camera in a certain distance regarding the objects mainly contain only objects, which can be used as well-segmented scenes for fine-grained object detection.

Llama2 based user prompt: A prompt user interface is directly adopted from the Llama2-chat 7B interface to perform vision language tasks. To adopt the pre-trained LLM for robotic tasks, a set of prompts for specific functions are used, and they are 1) 'describe this image as detailed' for image description by texts and associating the objects of the texts with their correspondence in the image; 2) 'detect an object' for object detection and 2D spatial location grounding. Here, 'object' can be instanced by a specific component such as a tool; 3) 'refer an object' for the object referring with a bounding box and a text label on it. The prompt template is adopted from a multi-task instruction template with the task-specific tokens. It consists of a general input format including image features, a task identifier token, and an instruction input. The task-specific tokens provided by MiniGPT-v2 can facilitate precise and accurate task execution such as visual grounding and object detection.

III. Scene understanding and object grounding

A. Zero-shot sample-based scene understanding

Scene understanding in this study is focused on a customised robotic work cell, and the scene images are not included in the datasets of training LLMs. These images can be defined as zero-shot samples for the pre-trained LLM and utilised to explore its transfer and generalisation capabilities. Fig. 4 illustrates the performance of scene understanding by a full-view image. An scene image is fed into the pre-trained LLM together with a language instruction of
'describe this image as detailed'. The result of its description is 'A robot is next to a table with a tool' in a simplified format, and the pre-trained model can recognise most of the objects in the image and reveal the spatial relationship of these objects. Therefore, the result reveals a brief understanding of the given scene and its constituent elements. In parallel, the visual grounding of objects pinpoints their 2D spatial locations with accurate bounding boxes and text labels. Here, associating the object of the text description with its

counterpart of the scene image is implemented by language reasoning. However, the components on the table are ambiguously depicted as 'a tool' without detailed identification. This is limited by the nature of the adopted model that melts small objects into the environment description. To address this problem, a fine-grained description and detection of small objects is necessary, which will be presented in Subsection III-B.

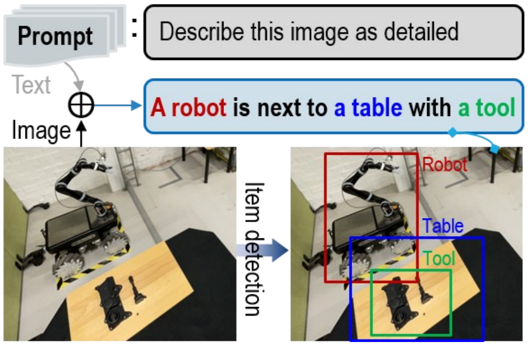

Prompt
Describe this image as detailed
Text
Image
A robot is next to a table with a tool
Item detection
Robot
Table
Tool

Fig. 4. Zero shot sample-based scene understanding and visual grounding.

B. Fine-grained object detection and grounding

As shown in the left-side sub-figure of Fig. 5, an image with only objects of interest (two parts) is segmented from the whole scene image and uploaded to the pre-trained model, and the instruction of 'detect valve cover' is prompted to detect the object of the valve cover, followed by the output of accurate detection and grounding with a bounding box and text label. It reveals that the developed fine-grained object detection approach can realise precise detection and localisation of the small object. Also, a test of how small the object can be detected is performed given available experimental resources. With a prompt of 'detect screw', the small-size screws can be precisely detected and grounded as show in the sub-figure. For the table-top object manipulation, an eye-in-hand camera mounted on a robot's wrist can have fine-grained images of the tasks within a certain distance, which are highly similar to the segmented image of the scene image.

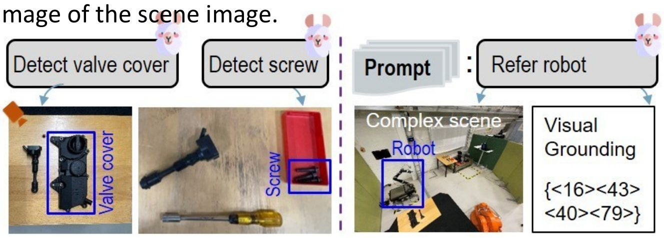


Fig. 5. Fine-grained object detection (left) and object referring within a complex scenario (right).

Right-side sub-figure of Fig. 5 shows the result of a test of object grounding in a complex scene. Compared with the scene image in Fig. 4, this scene image contains more objects and a complex environmental background. A language instruction of 'refer robot' is prompted to refer the robot in the complex scene and pinpoint its 2D position with an accurate bounding box. Its location information is depicted in the left-top (16,43) and right-bottom (40,79) corner position of the bounding box. This can validate the performance of the pre-trained model in the visual grounding of key objects within a complex scene.

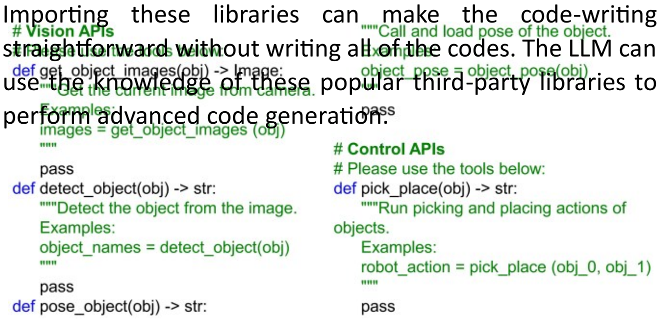
IV. LANGUAGE-ACTION TRANSLATION SUPPORTED BY CODE-WRITING LLM

A. Link language texts with code generation

As shown in Fig. 2, an NLP task for robots can be decomposed into an executor, actions, and components to be acted on [28]. Upon a brief understanding of the target scene and 2D spatial representation of the object in Section III, this section investigates a vision-language-action model-driven approach to downstream tasks. Inspired by Code as Policies (CaP)[29] and Instruct2Act [30], it uses a code-writing LLM to link language instructions of the tasks and high-level codes of robot actions, by integrating vision perception and robot control functions. It allows a robot to execute a sequence of actions based on an instruction from the user and an observation image captured by an eye-in-hand camera.

The code-writing model relies on a well-designed prompt to the LLM for code generation. A complete prompt to generate codes contains third-party libraries, API definitions, and in-context examples, and they are introduced as follows:

Third-party libraries: Python code libraries, such as NumPy, PyTorch, and cv2 can offer essential information about how APIs use the parameter types in these libraries for specific functions such as calculation and image processing. Importing these libraries can make the code-writing straightforward without writing all of the codes. The LLM can use the knowledge of these popular third-party libraries to perform advanced code generation.



```
# Vision APIs                                 """Call and load pose of the object.
# Please use the tools below:                    Examples:
def get_object_images(obj) -> Image:             object_pose = object_pose(obj)
    """Get the current image from camera.        """
    Examples:                                    pass
    images = get_object_images (obj)
    """                                       # Control APIs
    pass                                      # Please use the tools below:
def detect_object(obj) -> str:                def pick_place(obj) -> str:
    """Detect the object from the image.          """Run picking and placing actions of
    Examples:                                 objects.
    object_names = detect_object(obj)             Examples:
    """                                           robot_action = pick_place (obj_0, obj_1)
    pass                                          """
def pose_object(obj) -> str:                      pass
```

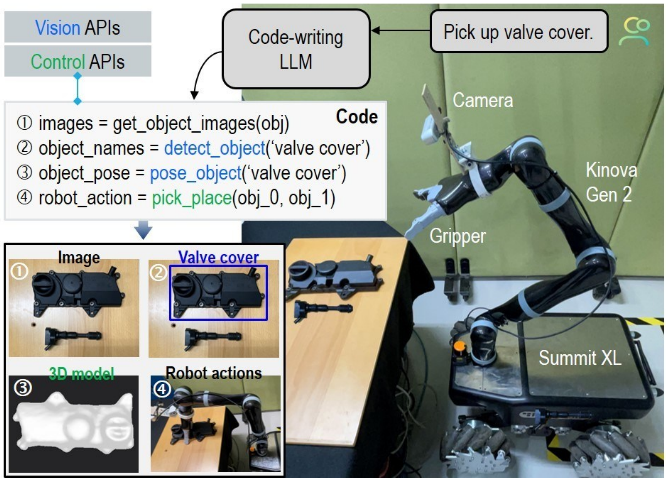
Fig. 6. Example of prompts for vision and control APIs.

API definitions: Given natural language instructions, they are decomposed into a set of function tasks. These subfunction tasks are defined and linked with specific APIs, and the control flow of these APIs is organised sequentially for task execution. Specifically, this study mainly relies on two types of APIs and they are for vision perception and robot action control, namely vision and control APIs as shown in Fig. 6. Specifically, their specifications are summarised as follows:

- getobjimages: gets fine-grained images of objects of interest by an eye-in-hand camera. It calls a visual servoing system to transmit and store the images in the cache variables.
- detectobject: detects and grounds the object that is extracted from language instructions, among the images. Specifically, the object of the language instruction (e.g., valve cover) is assigned to this function, and then the visual feature of the object is extracted for recognition and detection, followed by pinpointing the 2D spatial location in the image.
- poseobject: adds a 3D model representation of the object and its pose on the detected object where the technical details can be found in our previous study [2]. It provides position and orientation information of the object with CAD data for manipulation (e.g., grasping). The object pose regarding the robotic coordinate system is calculated by a frame transform and cached in the defined variable objectpose
- pickplace: perform pick and/or placing tasks of objects. The variables obj0 and obj1 are assigned with the object to be manipulated. It receives the object's pose and position in the robotic coordinate system, and generates robot paths of object grasping by using robot operating system (ROS)-based motion planners.

In-context examples: work as a crucial step in in-context learning, and instruction-to-code pairs as examples present the demonstrations of how to learn and generate code from examples [31]. Specifically, instructions are written as comments directly preceding a block of corresponding solution code. These instructions are concatenated with examples to construct a prompt. The prompt is fed into the code-writing model with the output of a corresponding program.

Given a language instruction, its object names are parsed and extracted by using language reasoning, and it can be few-shot prompted using code-writing LLMs to associate object names with language descriptions, categories, or past context. Then, the vision APIs for image processing and object detection is called to provide object's position and pose to the control APIs, where the motion planning of the robot arm is generated by a ROS-based motion planner.



```
① images = get_object_images(obj)
② object_names = detect_object('valve cover')
③ object_pose = pose_object('valve cover')
④ robot_action = pick_place(obj_0, obj_1)
```

Fig. 7. A scheme of linking language instructions into robot actions facilitated by vision perception and a code-writing LLM.

B. Experimental Validation

A case study of table-top object manipulation is performed to test and validate the developed system. As shown in Fig. 7, a user instructs a Kinova robot arm (with Robotnik SummitXL mobile base) to pick up a valve cover. The robot system is controlled by an ROS-based architecture, and a prompt interface to the LLM is connected to a PC with Ubuntu20.04 and a single NVIDIA RTX 4090 GPU.

Upon the scene understanding of the physical environment, a text instruction of 'pick up valve cover' is segmented into an action and a component to be acted on, and they are a predicate of 'pick up' and an object of 'valve cover'. Then, the language instruction is input to the code-writing LLM that outputs the high-level control codes, as shown in the 'Code' module of Figure 7. In parallel, a collection of the scene image is obtained using an eyein-hand camera (RealSense D435) by calling a vision API function of 'getobjectimages (obj)' in the code module, and the collected image with a specific resolution is shown in subset 1 , and is cached into a variable of images. The object of the language instruction (valve cover) is indexed and grounded into a task identifier token, and the indexed object is assigned to the object of the image detection. The function of objectdetect('valve cover') connects with the image processing approaches in Section III to extract the visual feature of the image, and then identify the object of the valve cover. The visual grounding is used to accurately determine the 2D spatial location among the image, indicated by a bounding box and a text label as shown in inset 2 .

Upon 3D reconstruction and pose estimate establishment through a neural field object modelling [2], a 3D model of the valve cover with its pose estimate as shown in Figure 8 (left) is overlaid onto its 2D image, and this is implemented by running the function of referobject('valve cover'). Then, the

position and pose information of the valve cover regarding the robotic coordinate system are cached in the variable of objectpose. The grasping point of object is the centre of the object's re-defined coordinate frame (as shown in the pose image of Figure 8 (left)), which is created by taking the centre and oriented box of the mesh model of the object's surface structure model and geometry. In parallel, such information is sent to the robot controller via a built-in visual servoing system (the details can be found in [2]). Once the visual information is detected, the control API is executed to perform object grasping by the robot. Specifically, the position and orientation of the object are loaded into a function of 'pickplace('valve cover')' where the valve cover is assigned to the variable of obj0 and the variable of obj1 is empty. Next, the embedded ROS-based motion planner outputs robot paths that adopt a top-down grasping policy as shown in Figure 8 (middle). Finally, the robot arm follows the motion path to perform the robot action of the pick place, and the final result is shown in Figure 8
(right).

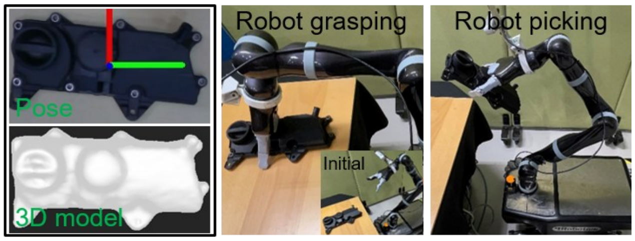


Fig. 8. Experimental results of robot actions: object's pose and 3D model (left), robot grasping (middle), and robot picking action (right).

V. Conclusions and future work

This study presents a vision language model-driven approach to scene understanding of unknown environments and robotic object manipulation supported by vision perception and a code-writing LLM. The method adopts a pre-trained VLM to bridge visual and textual information for scene understanding with outputs of text-based description, and it realises the visual grounding of objects of interest with zeroshot samples, and fine-grained object detection by a visual encoder and a language decoder. Upon 3D reconstruction and pose estimate establishment for the object, a codewriting LLM is used to generate high-level control code for object manipulation, and the results of visual perception are employed to the control codes for specific robot actions. Experimental evaluation using a robotic grasping task confirms the following contributions from the vision LLM method:

- VLM-driven scene understanding of an unknown interaction environment and text description of its constituent elements.
- A zero-shot-based approach to fine-grained visual grounding and object. detection from complex scene tasks.
- Linking language instructions with robot actions for object manipulation facilitated by vision perception and a code-writing LLM.

Future efforts will be directed to improving the performance of VLMs in the fine-grained understanding of complex environments and consistent element detection and generalisation to new tasks. Its utility in realising natural human-robot interactions and autonomous robotic systems driven by natural language will be investigated.

Acknowledgment

The authors acknowledge support from the Swedish Research Council (Vetenskapsradet) under award 2023-00493,˚ National Academic Infrastructure for Super computing in Sweden (NAISS) under award 2024/5-164, National Supercomputer Centre (NSC) Berzelius-2024-124, and the US National Science Foundation under awards CMMI-1830295 and EEC-2133630 (ERC HAMMER). 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] S. Liu, L. Wang, X. V. Wang, Function block-based multimodal control for symbiotic human–robot collaborative assembly, Journal of Manufacturing Science and Engineering 143 (9) (2021) 091001.

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

[3] S. Stepputtis, J. Campbell, M. Phielipp, S. Lee, C. Baral, H. Ben Amor, Language-conditioned imitation learning for robot manipulation tasks, Advances in Neural Information Processing Systems 33 (2020) 13139–13150.

[4] G. Orru, A. Piarulli, C. Conversano, A. Gemignani, Human-like` problem-solving abilities in large language models using chatgpt, Frontiers in Artificial Intelligence 6 (2023) 1199350.

[5] J. Zhang, S. Liu, R. X. Gao, L. Wang, Neural rendering-enabled 3d modeling for rapid digitization of in-service products, CIRP Annals (2023).

[6] K. Chowdhary, K. Chowdhary, Natural language processing, Fundamentals of artificial intelligence (2020) 603–649.

[7] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).

[8] K. I. Roumeliotis, N. D. Tselikas, D. K. Nasiopoulos, Llama 2: Early adopters' utilization of meta's new open-source pretrained model (2023).

[9] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al., Gpt-4 technical report, arXiv preprint arXiv:2303.08774 (2023).

[10] J. Banks, T. Warkentin, Gemma: Introducing new state-of-the-art open models (2024).

[11] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, et al., Emergent abilities of large language models, arXiv preprint arXiv:2206.07682 (2022).

[12] B. Min, H. Ross, E. Sulem, A. P. B. Veyseh, T. H. Nguyen, O. Sainz, E. Agirre, I. Heintz, D. Roth, Recent advances in natural language processing via large pre-trained language models: A survey, ACM Computing Surveys 56 (2) (2023) 1–40.

[13] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, A. Garg, Progprompt: Generating situated robot task plans using large language models, in: 2023 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2023, pp. 11523–11530.

[14] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, et al., Palm-e: An embodied multimodal language model, arXiv preprint arXiv:2303.03378 (2023).

[15] S. Yi, S. Liu, Y. Yang, S. Yan, D. Guo, X. V. Wang, L. Wang, Safetyaware human-centric collaborative assembly, Advanced Engineering Informatics 60 (2024) 102371.

[16] A. Brohan, Y. Chebotar, C. Finn, K. Hausman, A. Herzog, D. Ho, J. Ibarz, A. Irpan, E. Jang, R. Julian, et al., Do as i can, not as i say: Grounding language in robotic affordances, in: Conference on Robot Learning, PMLR, 2023, pp. 287–318.

[17] Y. Jiang, A. Gupta, Z. Zhang, G. Wang, Y. Dou, Y. Chen, L. Fei-Fei, A. Anandkumar, Y. Zhu, L. Fan, Vima: General robot manipulation with multimodal prompts, arXiv (2022).

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

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

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

[21] M. Johnson-Roberson, J. Bohg, G. Skantze, J. Gustafson, R. Carlson, B. Rasolzadeh, D. Kragic, Enhanced visual scene understanding through human-robot dialog, in: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2011, pp. 3342–3348.

[22] G. De Magistris, R. Caprari, G. Castro, S. Russo, L. Iocchi, D. Nardi, C. Napoli, Vision-based holistic scene understanding for context-aware human-robot interaction, in: International Conference of the Italian Association for Artificial Intelligence, Springer, 2021, pp. 310–325.

[23] M. Jaritz, J. Gu, H. Su, Multi-view pointnet for 3d scene understanding, in: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, 2019, pp. 0–0.

[24] D. Yang, X. Xu, M. Xiong, E. Babaians, E. Steinbach, Sri-graph: A novel scene-robot interaction graph for robust scene understanding, in: 2023 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2023, pp. 8171–8178.

[25] J. Chen, D. Zhu, X. Shen, X. Li, Z. Liu, P. Zhang, R. Krishnamoorthi, V. Chandra, Y. Xiong, M. Elhoseiny, Minigpt-v2: large language model as a unified interface for vision-language multi-task learning, arXiv preprint arXiv:2310.09478 (2023).

[26] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, arXiv preprint arXiv:2010.11929 (2020).

[27] J. Li, D. Li, S. Savarese, S. Hoi, Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, arXiv preprint arXiv:2301.12597 (2023).

[28] S. Liu, L. Wang, R. X. Gao, Cognitive neuroscience and robotics: Advancements and future research directions, Robotics and ComputerIntegrated Manufacturing 85 (2024) 102610.

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

[30] S. Huang, Z. Jiang, H. Dong, Y. Qiao, P. Gao, H. Li, Instruct2act: Mapping multi-modality instructions to robotic actions with large language model, arXiv preprint arXiv:2305.11176 (2023).

[31] S. Vemprala, R. Bonatti, A. Bucker, A. Kapoor, Chatgpt for robotics: Design principles and model abilities. 2023 (2023).