CoPa: General Robotic Manipulation through Spatial Constraints of Parts with Foundation Models

Haoxu Huang[2,3,4*], Fanqi Lin[1,2,4*], Yingdong Hu[1,2,4], Shengjie Wang[1,2,4], Yang Gao[1,2,4]

***Abstract*— Foundation models pre-trained on web-scale data are shown to encapsulate extensive world knowledge beneficial for robotic manipulation in the form of task planning. However, the actual physical implementation of these plans often relies on task-specific learning methods, which require significant data collection and struggle with generalizability. In this work, we introduce Robotic Manipulation through Spatial Constraints of Parts (CoPa), a novel framework that leverages the common sense knowledge embedded within foundation models to generate a sequence of 6-DoF end-effector poses for open-world robotic manipulation. Specifically, we decompose the manipulation process into two phases: task-oriented grasping and task-aware motion planning. In the task-oriented grasping phase, we employ foundation vision-language models (VLMs) to select the object's grasping part through a novel coarse-to-fine grounding mechanism. During the task-aware motion planning phase, VLMs are utilized again to identify the spatial geometry constraints of task-relevant object parts, which are then used to derive post-grasp poses. We also demonstrate how CoPa can be seamlessly integrated with existing robotic planning algorithms to accomplish complex, long-horizon tasks. Our comprehensive real-world experiments show that CoPa possesses a fine-grained physical understanding of scenes, capable of handling open-set instructions and objects with minimal prompt engineering and without additional training. Project page: copa-2024.github.io**

I. INTRODUCTION

Developing a general-purpose robot necessitates effective approaches in two critical areas: *(i)* high-level task planning, which determines what to do next, and *(ii)* low-level robotic control, focusing on the precise actuation of joints [1], [2]. The emergence of high-capacity foundation models [3], [4], pre-trained on extensive web-scale datasets, has inspired a surge of recent research efforts aimed at integrating these models into robotics [5], [6]. Nonetheless, these methods generally address only the "higher level" aspects of task planning [7]–[10]. In contrast, the prevailing approach for low-level control continues to revolve around crafting task-specific policies via diverse learning methods [11], [12]. Such policies, however, are brittle and prone to failure when encountering unseen scenarios [13]. Even the largest robotics models struggle outside environments they have previously encountered [14], [15].

The question then arises: what makes generalizable low-level robotic control so hard? We attempt to answer this question through the lens of human object manipulation. For instance, when an individual is tasked with hammering a nail, regardless of their familiarity with the specific hammer, they intuitively grasp it by the handle (instead of the head), adjust its orientation so the striking surface aligns with the nail, and then execute the strike. This process underscores the importance of a fine-grained understanding of the physical properties of task-related objects, or more broadly, the extensive common sense knowledge of the world that facilitates generalizable object manipulation. Some pioneering works [16], [17] have sought to leverage the rich semantic knowledge of Internet-scale foundation models to enhance low-level robotic control. Yet, these approaches are heavily dependent on intricate prompt engineering and suffer from a fundamental limitation: a *coarse* understanding of the scene, leading to failures in tasks requiring *fine-grained* physical understanding. Such a detailed understanding is essential for nearly all real-world robotic tasks of interest.

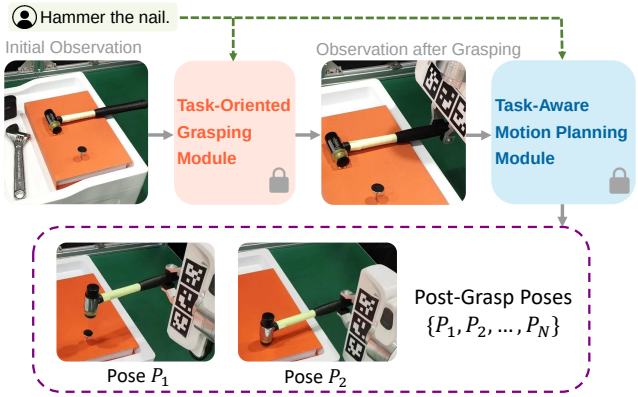

Fig. 1. **Overview.** We present CoPa, a novel framework that utilizes common sense knowledge embedded within VLMs for robotic low-level control. Given an instruction and scene observation, CoPa first generates a grasp pose through Task-Oriented Grasping Module. Subsequently, a Task-Aware Motion Planning Module (detailed in Fig. 3) is utilized to obtain post-grasp poses.

To endow robots with fine-grained physical understanding, we propose Robotic Manipulation through Spatial **Co**nstraints of **Pa**rts (**CoPa**), a novel framework that incorporates common sense knowledge embedded within foundation vision-language models (VLMs), such as GPT-4V, into the robotic manipulation tasks. We have observed that most manipulation tasks require a part-level, fine-grained physical understanding of objects within the scene. Hence, we design a coarse-to-fine grounding module to identify task-relevant parts. Then, to leverage VLMs for aiding the robotic low-level control, it is necessary to design an interface that not

* The first two authors contributed equally.
[1] Institute of Interdisciplinary Information Sciences, Tsinghua University.
[2] Shanghai Qi Zhi Institute.
[3] Shanghai Jiao Tong University.
[4] Shanghai Artificial Intelligence Laboratory.

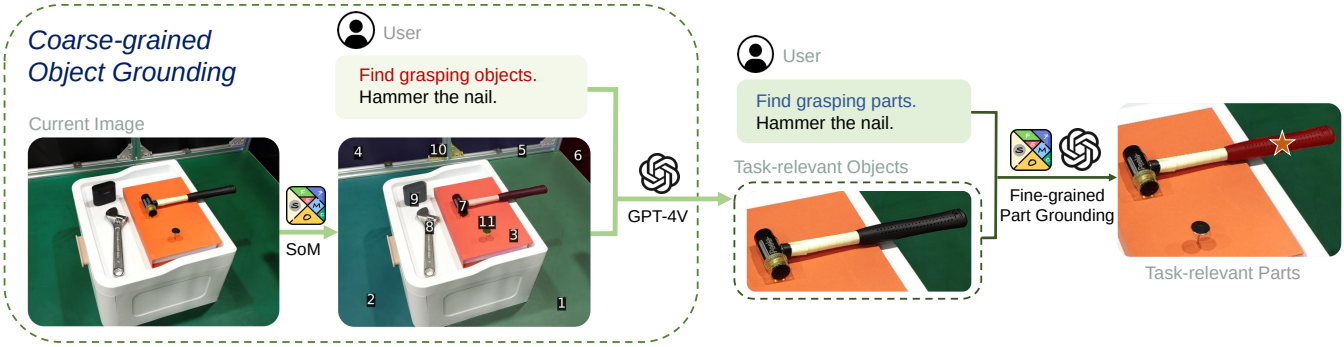

Fig. 2. Grounding Module. The grounding process is divided into two stages: coarse-grained object grounding and fine-grained part grounding. Specifically, we first segment and label objects within the scene using SoM. Then, in conjunction with the instruction, we employ GPT-4V to select the grasping/task-relevant objects. Finally, similar fine-grained part grounding is applied to locate the specific grasping/task-relevant parts.

only allows VLMs to reason in the form of language but also facilitates robot's object manipulation. Therefore, we propose utilizing spatial constraints as a bridge between VLMs and robots. Specifically, we utilize VLMs to generate the spatial constraints that the task-relevant parts must meet to accomplish the task, and then employ a solver to determine the robot's poses based on these constraints. Finally, to ensure the precise execution of the robot's actions, transitions between adjacent poses are achieved through traditional motion planning methods.

We demonstrate that CoPa is capable of completing everyday manipulation tasks with a high success rate through extensive real-world experiments. Attributed to the innovative design of coarse-to-fine grounding and constraint generation module, CoPa possesses a profound physical understanding of the environment and can generate precise 6-Dof poses to complete complex manipulation tasks, significantly surpassing a strong baseline VoxPoser [16].

Our contributions are summarized as follows:

- We propose CoPa, a novel framework that utilizes the common sense knowledge of VLMs for low-level robotic control, which can handle open-set instructions and objects with minimal prompt engineering and without additional training.
- Through extensive real-world experiments, CoPa is demonstrated to possess the capability to complete manipulation tasks that require a fine-grained understanding of physical properties of task-relevant objects, significantly surpassing baselines.
- We show that CoPa can be seamlessly integrated with high-level planning methods to accomplish complex, long-horizon tasks (e.g. make pour-over coffee and set up romantic table).

II. METHOD

A. Promblem Formulation

Most manipulation tasks can be decomposed into two phases: initial grasp of the object and subsequent motion required to complete the task. Motivated by this observation, we structure our approach into two modules: **task-oriented grasping** and **task-aware motion planning** (as shown in Fig. 1). Additionally, we posit that the execution of robotic tasks essentially entails generating a series of target poses for the robot's end-effector. The transition between adjacent target poses can be achieved through motion planning.

Given a language instruction l and the initial scene observation O_0 (RGB-D images), our objective in the task-oriented grasping module is to generate the appropriate grasp pose for the specified objects of interest. This process is represented as $P_0 = f(l, O_0)$. We denote the observation after the robot reaches P_0 as O_1. For the task-aware motion planning module, our goal is to derive a sequence of post-grasp poses, expressed as $g(l, O_1) \longrightarrow \{P_1, P_2, ..., P_N\}$, where N is the total number of poses required to complete the task. After acquiring the target poses, the robot's end-effector can reach these poses utilizing motion planning algorithms such as RRT* [18] and PRM* [19].

B. Task-Oriented Grasping

To generate the task-oriented grasp pose, our approach initially employs a grasping model to produce grasp pose proposals, and filter out the most feasible one through our novel grasping part grounding module.

Grasp Pose Proposals. We leverage a pre-trained grasping model for generating grasp pose proposals. To achieve this, we first convert RGB-D images into point clouds by back-projecting them into 3D space. These point clouds are then input into GraspNet [20], which outputs 6-DOF grasp candidates. However, given that GraspNet yields all potential grasps within a scene, it is necessary for us to employ a filtering mechanism that selects the optimal grasp based on the specific task outlined by the language instruction.

Grasping Part Grounding.

We employ a two-stage process to ground language instructions to the specific parts of objects intended for grasping: *coarse-grained object grounding* and *fine-grained part grounding*. The entire grounding process is shown in Fig. 2. At both stages, we incorporate a recent visual prompting mechanism known as Set-of-Mark (SoM) [21]. SoM leverages segmentation models to partition an image into distinct regions, assigning a numeric marker to each, significantly boosting the visual grounding capabilities of VLMs. During the *coarse-grained object grounding* phase, SoM is utilized at the object level to detect and label all objects within the scene. Following this, VLMs are tasked with pinpointing the

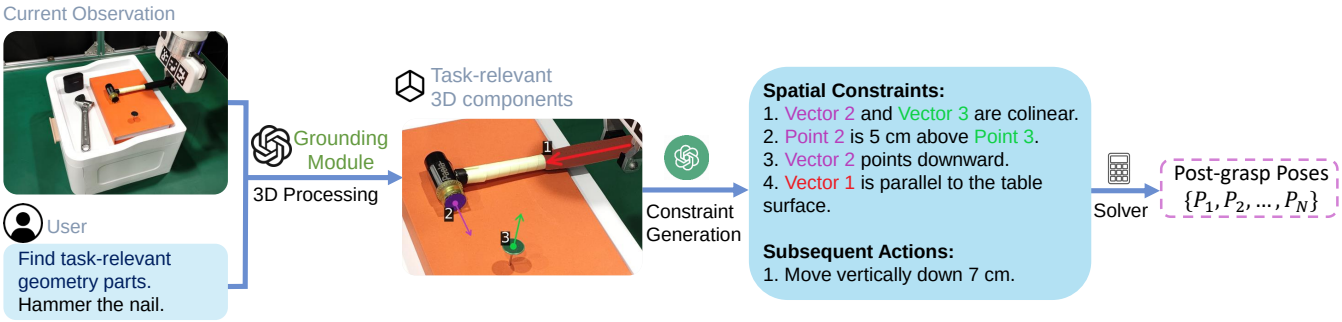

Fig. 3. Task-Aware Motion Planning Module. Given the instruction and the current observation, we first employ a grounding module (detailed in Fig. 2) to identify task-relevant parts within the scene. Subsequently, these parts are modeled in 3D, and are then projected and annotated onto the scene image. Following this, VLMs are utilized to generate spatial constraints for these parts. Finally, a solver is applied to calculate the post-grasp poses based on these constraints.

target object for grasping (e.g., a hammer), guided by the user's instructions. The selected object is then cropped from the image, upon which *fine-grained part grounding* is applied to determine the specific part of the object to be grasped (e.g., the handle of the hammer). This coarse-to-fine design endows our method with fine-grained physical understanding ability, enabling generalization across complex scenarios. Finally, we filter the grasp pose candidates, projecting all the grasp points onto the image and retaining only those within the grasping part mask. From these, the pose with the highest confidence scored by GraspNet is selected as the ultimate grasp pose P_0 for execution.

C. Task-Aware Motion Planning

After successfully executing task-oriented grasping, now we aim to obtain a series of post-grasp poses. We divide this step into three modules: task-relevant part grounding, manipulation constrains generation and target pose planning. The entire process is shown in Fig. 3.

Task-Relevant Part Grounding. Similar to the previous grasp part grounding module, we use *coarse-grained object grounding* and *fine-grained part grounding* to locate task-relevant parts. Here we need to identify multiple task-relevant parts (e.g. the hammer's striking surface, handle and the nail's surface). Additionally, we observe that numeric marks on the robotic arm may affect VLMs' selection, so we filter out the masks on the robotic arm (detailed in the Appendix).

Manipulation Constraints Generation. During the execution of tasks, task-relevant objects are often subject to various spatial geometric constraints. similarly, when capping a bottle, the lid must be positioned directly above the mouth of the bottle. These constraints inherently necessitate common sense knowledge, which includes a profound comprehension of the physical properties of objects. We aim to leverage VLMs to generate spatial geometric constraints for the object manipulated by the robot.

We first model identified task-relevant parts as simple geometric elements. Specifically, we represent slender parts (e.g. hammer handle) as vectors, while other parts are modeled as surfaces. For the parts modeled as vectors, we directly draw them on the scene image; for those modeled as surfaces, we ascertain their center points and normal vectors, which are then projected and marked on the 2D scene image. The annotated image is used as input for VLMs, which are prompted to generate spatial constraints for these geometric elements. We craft a set of descriptions for spatial constraints, such as collinearity between two vectors, perpendicularity between a vector and a surface, and so force. We instruct the VLMs to first generate the constraints necessary for the first target pose, followed by the subsequent actions required after reaching that pose. Fig. 3 provides an illustrative example of this process. Implementation details of this process are provided in the Appendix.

Target Pose Planning. Upon obtaining manipulation constraints, we proceed to derive the sequence of post-grasp poses. This is equivalent to computing a sequence of SE(3) matrices such that, when applied to the parts of the object manipulated by the robotic arm, these parts satisfy the spatial geometric constraints. We operate under the assumption that the object part under manipulation and the robotic end-effector together constitute a rigid body. Consequently, these calculated SE(3) transformations can be directly applied to the robotic end-effector. We formalize the computation of the SE(3) matrix as a constrained optimization problem. Specifically, we compute a loss for each constraint, and then a nonlinear constraint solver is used to find the SE(3) matrix that minimizes the sum of these losses. After obtaining the first target pose, we solve for subsequent poses in alignment with the actions specified by VLMs. Concretely, we sequentially compute a new pose corresponding to each subsequent action. This process results in a complete set of post-grasp poses $\{P_1, P_2, ..., P_N\}$, with the transitions between adjacent poses facilitated by motion planning algorithms. The detailed process for solving SE(3) matrix and a comprehensive description of the subsequent actions can be found in the Appendix.

III. Experiments

A. CoPa for Real-World Manipulation

We design 10 real-world manipulation tasks to study whether CoPa can generate robot trajectories to perform real-world manipulation tasks. The quantitative results are detailed in Table I. We find that CoPa achieves a remarkable success rate of 63% across ten different tasks, significantly outperforming the VoxPoser baseline and various ablation variants (detailed in the following sections). A key factor

Tasks	CoPa (Ours)	Voxposer	A	B	C
Hammer nail	30%	0%	0%	0%	10%
Find scissors	70%	50%	10%	70%	70%
Press button	80%	10%	10%	60%	20%
Open drawer	80%	40%	10%	70%	30%
Pour water	30%	0%	0%	10%	0%
Put eraser	80%	30%	30%	60%	80%
Insert flower into vase	70%	0%	0%	60%	0%
Put glasses onto shelf	60%	20%	30%	50%	60%
Put spoon into cup	60%	10%	0%	30%	30%
Sweep nuts	70%	20%	20%	50%	70%
Total	**63%**	18%	11%	46%	37%

TABLE I. Quantitative results in real-world experiments.

in CoPa's superior performance is its leverage of common sense knowledge embedded in VLMs, which enables a fine-grained understanding of objects' physical properties during both part grounding and constraint generation phases.

B. Understanding Properties of CoPa

In this section, we delve deeper into CoPa, shedding light on its intriguing properties through a comparative analysis with Voxposer, another method that utilizes the common sense knowledge embedded in foundation models to synthesize robot trajectories. CoPa exhibits significant advantages in the following three aspects:

Fine-Grained Physical Understanding. Many manipulation tasks require a nuanced physical understanding of the scene, which necessitates not only identifying object parts with fine granularity but also comprehending their intricate attributes. CoPa excels in this aspect, employing a coarse-to-fine part grounding module to select grasping/task-relevant object parts, and then utilizing VLMs to provide their spatial geometry constraints. In contrast, Voxposer only perceives objects in the scene as a whole. This coarse-grained level of comprehension often leads to failure in tasks that require precise operations.

Simple Prompt Engineering. CoPa demonstrates remarkable generalizability across a wide range of scenarios with minimal prompt engineering. In our CoPa experiments, we employ just three examples to aid the VLMs in comprehending their roles. In contrast, Voxposer relies on highly complex prompts containing 85 hand-crafted examples. Its capability for reasoning predominantly stems from the provided prompts, thereby limiting its generalizability to new scenarios. When we attempt to simplify Voxposer's prompts, reducing the example count to three for each module, the system's performance drastically declines, resulting in almost complete failure across all evaluated tasks.

Handling Rotation DoF. Robotic manipulation requires not just the movement of the end-effector to a specified location but also the precise control of its rotation. CoPa calculates the end-effector's 6-DoF pose by considering the spatial geometric constraints of key object parts within the scene, allowing for accurate and continuous control over rotation DoF. Conversely, Voxposer attempts to have LLMs directly specify the end-effector's rotation DoF based on simple examples in prompts, causing the output rotation values to be selected from a limited set of discrete options. This approach often overlooks the dynamic interactions and constraints between objects.

C. Ablation Study

We next conduct a series of ablation studies to demonstrate the significance of the foundation model within our framework, as well as the design of coarse-to-fine grounding and constraint generation. The results are shown in Table I. The three variants are represented in the table as **A**, **B**, **C**.

1) CoPa w/o foundation: We eliminate the use of foundation vision-language models (GPT-4V). Specifically, we substitute grasping/task-relevant parts grounding module with an open-vocabulary detector, Owl-ViT. Additionally, we remove the constraint generation phase and instead compute post-grasp poses in a predefined rule-based manner (detailed in the Appendix). The results, as presented in Table I, reveal that this approach encounters significant challenges, with a success rate of merely 11% across all the tasks. This underscores the crucial role of the common sense knowledge embedded within VLMs.

2) CoPa w/o coarse-to-fine: We eliminate the coarse-to-fine design in the grounding module, opting instead for direct utilization of fine-grained SoM and GPT-4V to select object parts within scenes. Experimental results indicate that removing coarse-to-fine design leads to a performance decline, especially in tasks where identifying important parts accurately is challenging.

3) CoPa w/o constraint: In this ablation study, we have VLMs directly output numerical values for the post-grasp poses of the end-effector, instead of the constraints that need to be satisfied by the object being manipulated. Experiments demonstrate that, for most manipulation tasks, directly deriving precise pose values from scene images is extremely challenging. In contrast, utilizing constraints given by VLMs to solve for post-grasp poses presents a more viable option.

D. Integration with High-Level Planning

High-level planning and low-level control are two critical and decoupled aspects of robotic task execution. Our low-level control framework can be seamlessly integrated with high-level planning methods to accomplish complex long-horizon tasks. We design two long-horizon tasks, `Make pour-over coffee` and `Set up romantic table`, to validate the effectiveness of this combination. Not only do these two tasks need to be accurately decomposed into reasonable and actionable steps, but the execution of each step requires a profound understanding of the physical properties of the task-relevant objects. Specifically, we employ VILA [10] as the high-level planning method to decompose the high-level instruction into a sequence of low-level control tasks. Subsequently, these low-level control tasks are executed sequentially using CoPa. Experiments demonstrate that CoPa, combined with high-level planning methods, can effectively complete long-horizon tasks, showcasing the potential of this combination for real-world applications.

REFERENCES

[1] S. Cambon, R. Alami, and F. Gravot, "A hybrid approach to intricate motion, manipulation and task planning," *The International Journal of Robotics Research*, vol. 28, no. 1, pp. 104–126, 2009.

[2] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 1470–1477.

[3] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, *et al.*, "On the opportunities and risks of foundation models," *arXiv preprint arXiv:2108.07258*, 2021.

[4] 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.

[5] Y. Hu, Q. Xie, V. Jain, J. Francis, J. Patrikar, N. Keetha, S. Kim, Y. Xie, T. Zhang, Z. Zhao, *et al.*, "Toward general-purpose robots via foundation models: A survey and meta-analysis," *arXiv preprint arXiv:2312.08782*, 2023.

[6] 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," *arXiv preprint arXiv:2312.07843*, 2023.

[7] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[8] W. Huang, F. Xia, D. Shah, D. Driess, A. Zeng, Y. Lu, P. Florence, I. Mordatch, S. Levine, K. Hausman, *et al.*, "Grounded decoding: Guiding text generation with grounded models for robot control," *arXiv preprint arXiv:2303.00855*, 2023.

[9] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, *et al.*, "Inner monologue: Embodied reasoning through planning with language models," *arXiv preprint arXiv:2207.05608*, 2022.

[10] Y. Hu, F. Lin, T. Zhang, L. Yi, and Y. Gao, "Look before you leap: Unveiling the power of gpt-4v in robotic vision-language planning," *arXiv preprint arXiv:2311.17842*, 2023.

[11] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, pp. 1–35, 2017.

[12] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[13] A. Xie, L. Lee, T. Xiao, and C. Finn, "Decomposing the generalization gap in imitation learning for visual robotic manipulation," *arXiv preprint arXiv:2307.03659*, 2023.

[14] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, *et al.*, "Rt-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022.

[15] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," *arXiv preprint arXiv:2307.15818*, 2023.

[16] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei, "Voxposer: Composable 3d value maps for robotic manipulation with language models," *arXiv preprint arXiv:2307.05973*, 2023.

[17] 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)*. IEEE, 2023, pp. 9493–9500.

[18] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, vol. 30, no. 7, pp. 846–894, 2011.

[19] L. Gang and J. Wang, "Prm path planning optimization algorithm research," *Wseas Transactions on Systems and control*, vol. 11, pp. 81–86, 2016.

[20] H.-S. Fang, C. Wang, M. Gou, and C. Lu, "Graspnet-1billion: A large-scale benchmark for general object grasping," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 444–11 453.

[21] J. Yang, H. Zhang, F. Li, X. Zou, C. Li, and J. Gao, "Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v," *arXiv preprint arXiv:2310.11441*, 2023.

[22] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 824–24 837, 2022.

[23] M. Minderer, A. Gritsenko, A. Stone, M. Neumann, D. Weissenborn, A. Dosovitskiy, A. Mahendran, A. Arnab, M. Dehghani, Z. Shen, *et al.*, "Simple open-vocabulary object detection with vision transformers. arxiv 2022," *arXiv preprint arXiv:2205.06230*.

[24] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, *et al.*, "Segment anything," *arXiv preprint arXiv:2304.02643*, 2023.

APPENDIX

A. Hardware Setup

We set up a real-world tabletop environment. We use a Franka Emika Panda robot (a 7-DoF arm) and a 1-DoF parallel jaw gripper. We use Franka ROS and MoveIt[1] to control the robot, which by default uses an RRT-Connect planner for motion planning. For perception, we mount two RGB-D cameras (Intel RealSense D435) at two opposite ends (left and right from the top-down view) of the table and calibrate them.

B. Tasks and Evaluations.

We design 10 real-world manipulation tasks (as show in Fig. 4), each demanding a comprehensive understanding of the physical properties of objects. We provide a detailed description of these tasks in Table II. For each task, we evaluate all methods across 10 different variations of the environment, which encompass alterations in object types and their arrangements.

`Hammer nail`	**Instruction:** "Hammer the nail." **Description:** This task requires the robot to first grasp the handle of the hammer, then rotate it until its striking surface aligns with the surface of the nail, and finally hammer downwards. To accomplish this task, it is essential to accurately identify and model the hammer's striking surface, handle and the nail's surface.
`Find scissors`	**Instruction:** "Find scissors for me." **Description:** In this task, the scissors may be partially obscured by other objects, such as books. The robot is required to first locate the scissors and then grasp its handle.
`Press button`	**Instruction:** "Press the button with the stick." **Description:** This task necessitates initially grasping the stick, then rotating it until its axis is directly aligned with the button, and finally pressing it. To accomplish this task, it is imperative to accurately identify and model the stick and the button.
`Open drawer`	**Instruction:** "Open the drawer." **Description:** This task requires the initial grasping of the drawer handle, followed by a linear pull along the handle's normal vector.
`Pour water`	**Instruction:** "Pour water from kettle to funnel/cup." **Description:** This task requires that the spout needs to be moved directly above the funnel, and the kettle needs to be rotated at a certain angle so that the water can flow out. This task imposes stringent demands on the robot's control over its rotation DoF.
`Put eraser into drawer`	**Instruction:** "Put eraser into the drawer." **Description:** In this task, a portion of the eraser is encapsulated by a protective cover, necessitating that the robot exclusively grasps this protective cover.
`Insert flower into vase`	**Instruction:** "Put flowers into the vase." **Description:** This task requires first grasping the flower by its stem (not the petals), then moving the flower directly above the vase while rotating the flower to an upright position, and finally inserting it straight down into the vase.
`Put glasses onto shelf`	**Instruction:** "Put glasses onto the shelf." **Description:** In this task, We need to utilize common sense knowledge to determine that, when picking up glasses, one should grasp the frame rather than the lenses.
`Put spoon into cup`	**Instruction:** "Put spoon into the cup." **Description:** This task requires first grasping the spoon's handle, then rotating it to the vertical direction, moving it directly above the cup, and finally inserting it vertically down into the cup.
`Sweep nuts`	**Instruction:** "Select a tool to sweep nuts aside." **Description:** This task requires the robot to first identify a tool (e.g. rasp) suitable for sweeping nuts through common sense knowledge, and then to grasp the handle of the selected tool.

TABLE II. **A List of 10 Real-World Manipulation Tasks.** These tasks require a profound physical understanding of the scene. We provide the instructions used in our experiments and detailed descriptions for each task.

C. VLMs and Prompting.

We employ GPT-4V from OpenAI API as the VLM. CoPa involves minimal few-shot prompts to aid VLMs in comprehending their roles. Additionally, the chain-of-thought technique [22] is utilized to facilitate a deeper understanding of the scene by VLMs. Prompts used in Section II-B and Section II-C can be found as follows:

[1]http://docs.ros.org/en/kinetic/api/moveit_tutorials/html/

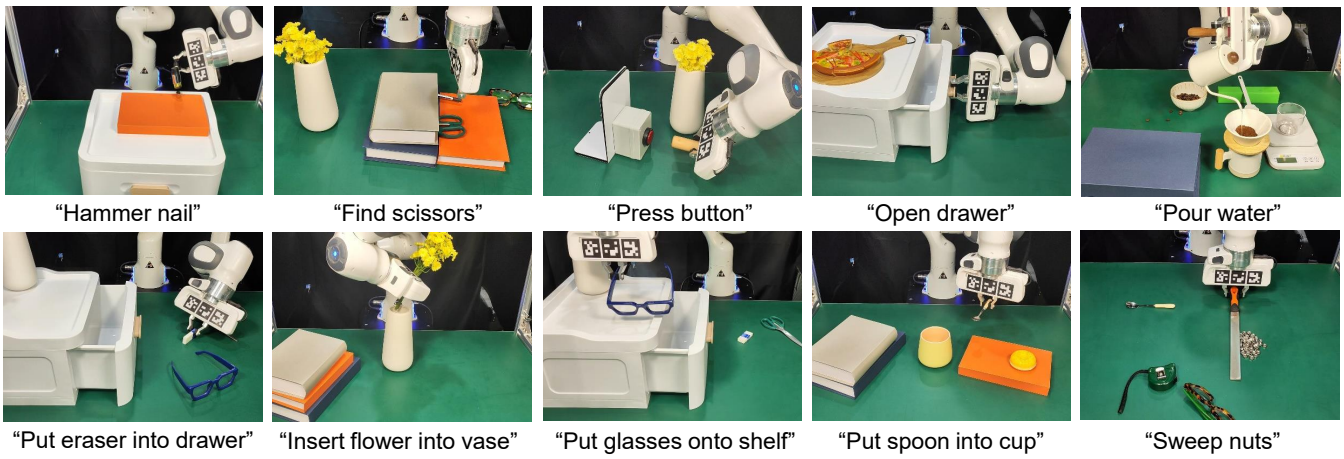

Fig. 4. **10 real-world experiments.** Boasting a fine-grained physical understanding of scenes, CoPa can generalize to open-world scenarios, handling open-set instructions and objects with minimal prompt engineering and without the need for additional training.

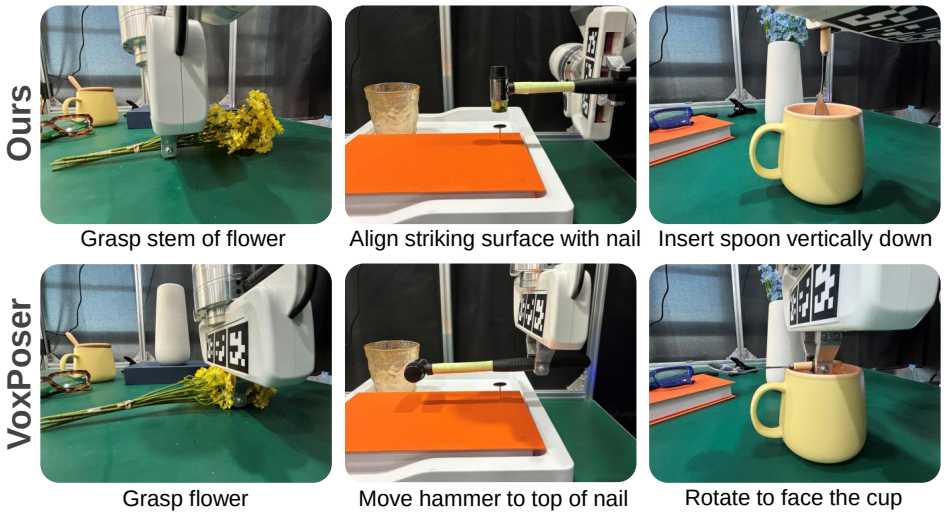

Fig. 5. **Comparison with VoxPoser.** We illustrate the execution of CoPa (top) and VoxPoser (bottom), demonstrating that CoPa possesses a fine-grained physical understanding of scenes and can effectively handle rotation DoF. The tasks from left to right are sequentially `Insert flower into vase`, `Hammer nail`, `Put spoon into cup`.

Coarse-Grained Grasping Object Grounding: copa-2024.github.io/prompts/coarse_grained_grasping_object_grounding.pdf
Fine-Grained Grasping Part Grounding: copa-2024.github.io/prompts/fine_grained_grasping_part_grounding.pdf
Coarse-Grained Task-Relevant Object Grounding: copa-2024.github.io/prompts/coarse_grained_relevant_object_grounding.pdf
Fine-Grained Task-Relevant Part Grounding: copa-2024.github.io/prompts/fine_grained_relevant_part_grounding.pdf
Constraint Generation: copa-2024.github.io/prompts/constraint_generation.pdf

D. Baselines.

We compare with Voxposer [16], a method capable of synthesizing closed-loop robot trajectories without necessitating additional training through the utilization of a series of foundational models. Following Huang et al [16], we employ GPT-4 from OpenAI API as the LLM, and utilize the open-vocabulary detector Owl-ViT [23] and Segment Anything [24] for perception. Additionally, we adopt their real-world prompt as the prompt for Voxposer in our experiments. We show the comparison between our method and VoxPoser in Fig. 5

E. Task-Oriented Grasping Pipeline.

We show the entire process of task-oriented grasping in Fig. 6.

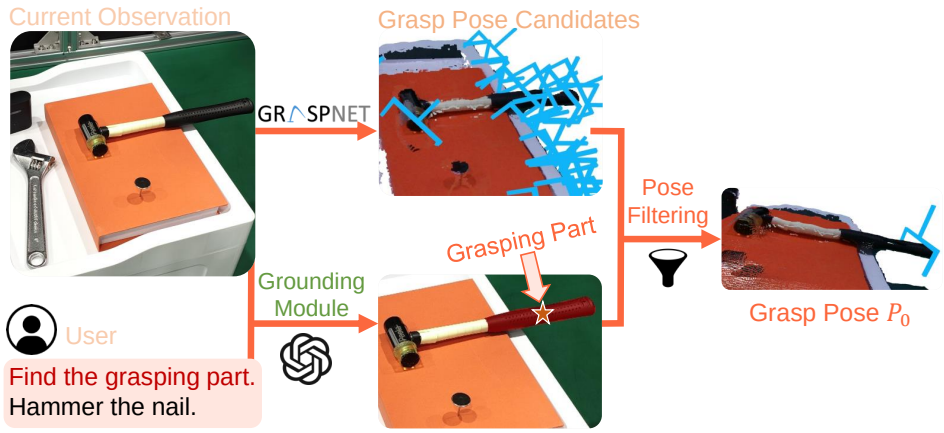

Fig. 6. Task-Oriented Grasping Module. This module is employed to generate grasp poses. Grasp pose candidates are generated from the scene point cloud using GraspNet. Concurrently, given the instruction and the scene image, the grasping part is identified by a grounding module (detailed in Fig. 2). Ultimately, the final grasp pose is selected by filtering candidates based on the grasping part mask and GraspNet scores.

F. Robotic Arm Filtering.

Based on the camera's extrinsic parameters, we render the robot's URDF model onto the camera plane, thereby obtaining the robot's mask.

G. Part Modeling and Annotation.

In the task-aware motion planning phase (Section II-C), we need to model task-relevant parts selected by the grounding module and annotate them in the scene image. We complete this through the following stages:

Part Modeling. We model identified task-relevant parts as vectors or surfaces. Specifically, we commence by obtaining the minimum bounding rectangle for each part. Parts with an aspect ratio exceeding a predetermined threshold are considered slender and are modeled as vectors. The remaining parts are modeled as surfaces.

Part Formulation. We need to obtain the mathematical representation of each part on the 2D image in this stage. For parts modeled as vectors, we first perform linear regression to fit a line that best corresponds to their 2D masks, then identify the intersection points of the line with the boundaries of the parts, which serve as the endpoints of the vector. For parts modeled as surfaces, we first employ the RANSAC algorithm to determine their 3D center points and normal vectors, which are then projected onto the 2D image.

Part Annotation. Now we need to annotate the parts according to their formulation on the scene image. First, each part is masked with color and translucency on the image. Then for the parts modeled as vectors, we connect the two endpoints and put a numerical label adjacent to the endpoint farther from the robot arm. For parts modeled as faces, we mark the center point and normal vector, and label adjacent to the center point.

H. Details of Constraints.

For each task, we obtain a set of constraints through vision-language models. We then utilize optimization algorithms (e.g., the BFGS algorithm or Trust-Region Constrained Optimization) to solve for an SE(3) matrix that minimizes the cumulative loss associated with these constraints. In Table III, we provide a detailed description of the constraints used in our experiments, along with their corresponding loss calculation methods. In the descriptions of these constraints, `Point A` and `Vector A` are located on the object being manipulated, and thus require an SE(3) transformation when calculating loss. Other points and vectors are considered static and do not require SE(3) transformation.

We define the SE(3) matrix we need to solve as follows:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^{\mathbf{T}} & 1 \end{bmatrix} \in \mathbb{SE}(3), \tag{1}$$

where Euclidean group $\mathbb{SE}(3) := \{\mathbf{R}, \mathbf{t} \mid \mathbf{R} \in \mathbb{SO}(3), \mathbf{t} \in \mathbb{R}^3\}$. We denote the points as $p \in \mathbb{R}^3$, and the normalized vectors as $V \in \mathbb{R}^3$. Furthermore, we denote $\mathcal{T}$ as the SE(3) transformation, which can be applied to both points and vectors:

$$\mathcal{T}(p) = \mathbf{R}p + \mathbf{t}, \quad \mathcal{T}(V) = \mathbf{R}V, \tag{2}$$

I. Details of Subsequent Actions.

In Table IV, we provide a detailed description of the subsequent actions utilized in our experiments, along with their corresponding methodologies for calculating new poses.

Descriptions of Constraints	Loss Calculation
`Vector A` and `Vector B` are on the same line, with the opposite direction.	$loss = \|\mathcal{T}(V_A) \times V_B\| + \|(\mathcal{T}(p_A) - p_B) \times V_B\| + \|\mathcal{T}(V_A) + V_B\|$
The target position of `Point A` is x cm along `Vector B` from `Point C`'s current position.	$loss = \|(\mathcal{T}(p_A) - p_C) \cdot V_B - x\| + \|(\mathcal{T}(p_A) - p_C) \times V_B\|$
`Vector A` is parallel to the table surface.	$loss = \|\mathcal{T}(V_A) \cdot V_{table}\|$
`Point A` is x cm above the table surface.	$loss = \|(\mathcal{T}(p_A) - p_{table}) \cdot V_{table} - x\|$
`Vector A` is perpendicular to the table surface.	$loss = \|\mathcal{T}(V_A) \times V_{table}\|$

TABLE III. **Desciptions of Constraints and Their Corresponding Loss Calculation Methods.** $\|\cdot\|$ represents l_2-norm. The variables p_{table} and V_{table} respectively denote the coordinate and the normal vector of the table, with their values being $(0.5, 0, 0.07)$ and $(0, 0, 1)$ respectively. Each `Vector` corresponds to a `Point`. The normal vector of the part modeled as the surface corresponds to the center point of the surface, while the point corresponding to the part modeled as the vector is the point farther away from the robotic arm among its two endpoints.

Descriptions of Subsequent Actions	New Pose Calculation
Move vertically down x cm.	Subtract x cm from the current pose on the z-axis.
Move forward x cm.	Move x cm along the current orientation of the end-effector.
Open the gripper.	Open the gripper.
End-effector rotates 180 degrees.	Rotate the joint corresponding to the end-effector (the 7th joint for Franka Emika Panda robot) by 180 degrees.

TABLE IV. **Desciptions of subsequent actions and their corresponding new pose calculation methods.**

J. Predefined Rule-Based Post-Grasp Pose Generation.

We design a rule-based method to replace the constraint generation module within our framework to generate post-grasp poses. This method entails a prescribed pose calculation protocol specific to each format of instruction. The formats of the instructions along with their corresponding post-grasp poses calculation methodologies are detailed in Table V.

Instruction Format	Post-Grasp Pose Calculation
Hammer `A`.	1. Move hammer to 5 cm above `A`. 2. Move vertically down 6 cm.
Press `A` with `B`.	1. Move `B` to 5 cm above `A`. 2. Move vertically down 6 cm.
Open `A`.	1. Move backward 10 cm.
Pour water from `A` to `B`.	1. Move `A` to 5 cm above `B`. 2. End-effector rotates 180 degrees.
Put `A` into `B`.	1. Move `A` to 5 cm above `B`. 2. Open the gripper.

TABLE V. **Predefined rule-based pose calculation methods.** `A` and `B` in the instruction can refer to any object.

K. Long-Horizon Tasks

We show the execution process of the two long-horizon tasks, `Make pour-over coffee` and `Set up romantic table` in Figure 7.

L. More Visualization.

Additional visualization for grounding module are presented in Fig. 8, for task-oriented grasping in Fig. 9, and for task-relevant motion planning in Fig. 10.

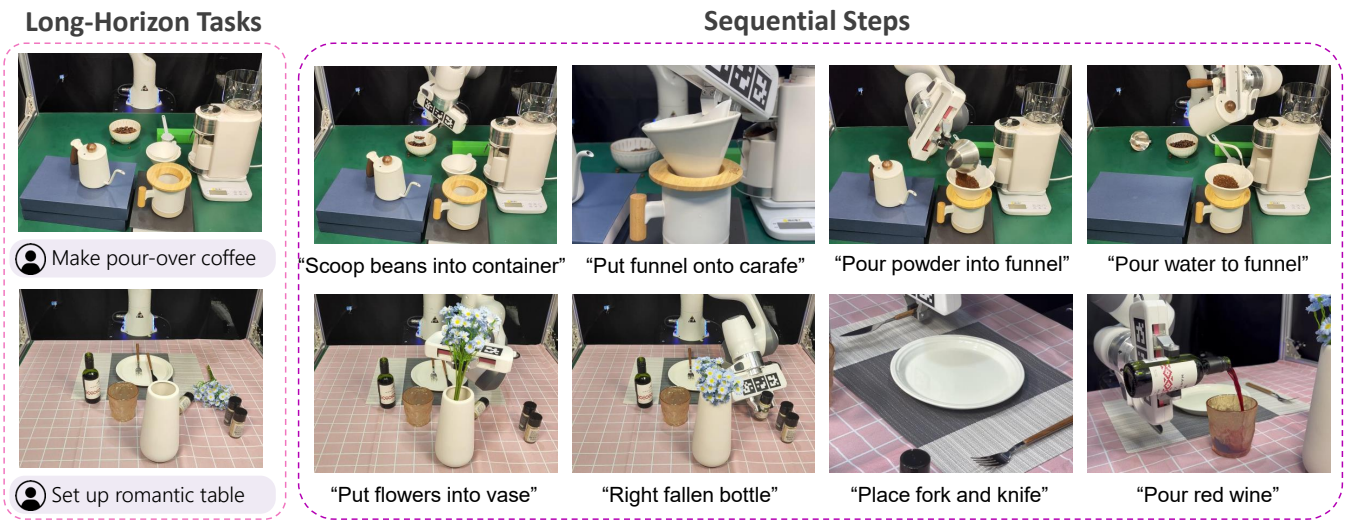

Fig. 7. **Long-Horizon Tasks.** We show the execution process of two long-horizon tasks: `Make pour-over coffee` and `Set up romantic table`. We demonstrate that CoPa can be seamlessly integrated with high-level planning methods to accomplish complex long-horizon tasks.

Fig. 8. **Visualization for Grounding Module.**

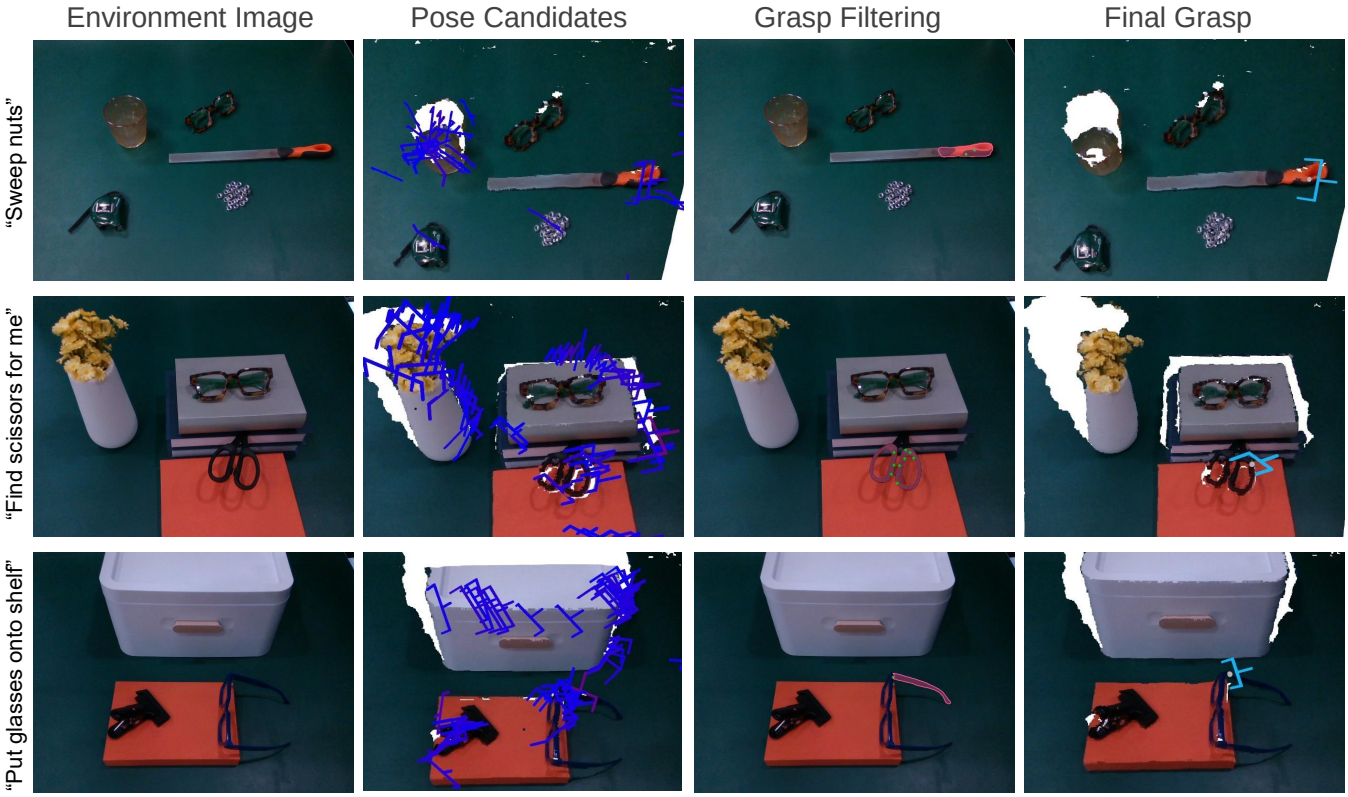

Fig. 9. **Visualization for Task-Oriented Grasping.**

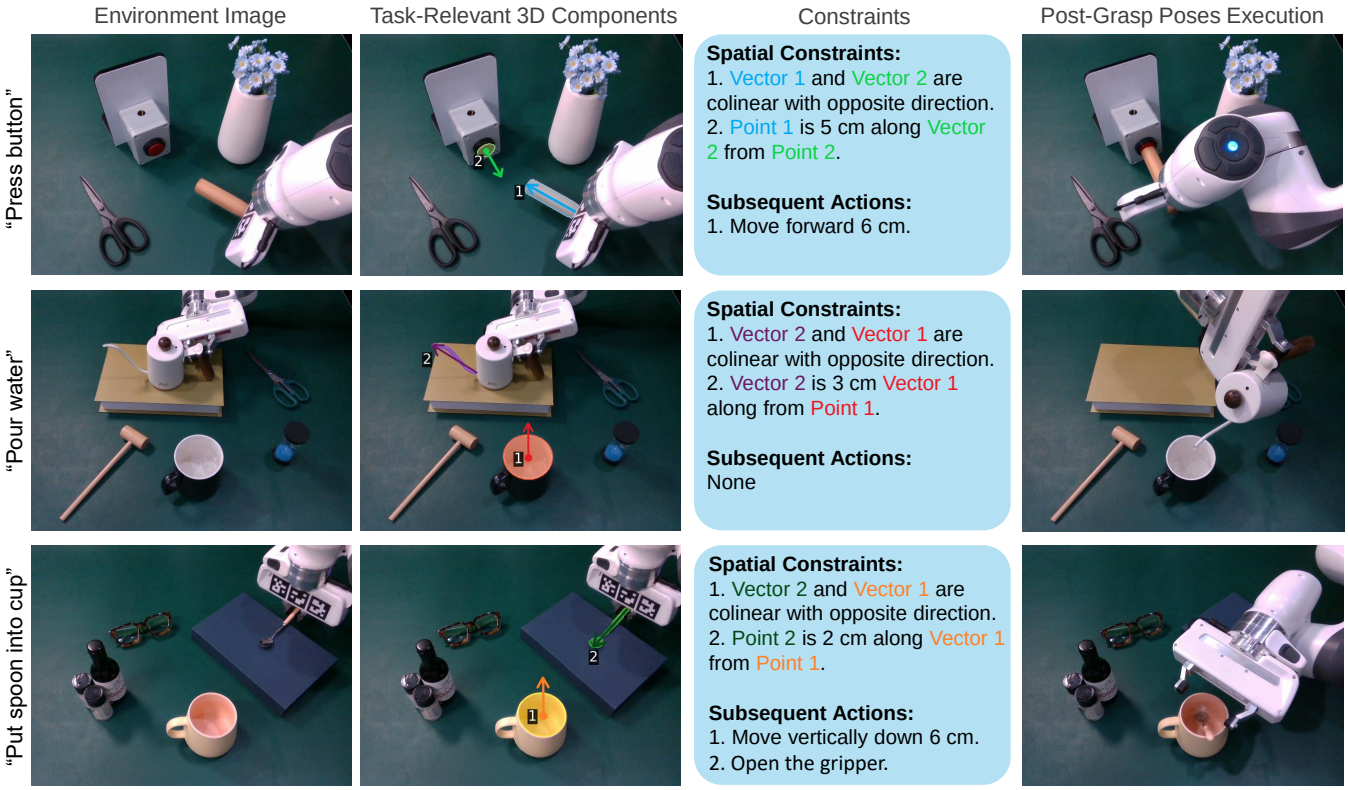

Fig. 10. **Visualization for Task-Aware Motion Planning.**