# Execution-Aware Task Planning for Language-Conditioned Bimanual Manipulation

Emanuele Cuzzocrea*,1,2, Giuseppina Iannotti*,1,2, Luca Rossini[1], Arturo Laurenzi[1], Luca Muratore[1], Nikolaos Tsagarakis[1]

***Abstract*—Robotic manipulation requires connecting high-level task objectives with the physical constraints of action execution. In this work, a zero-shot language-conditioned framework for bimanual manipulation is presented, where task-level reasoning is not decoupled from the way actions are executed, but addressed jointly within a unified decision process. The objective is to move beyond high-level manipulation primitives and enable decisions that account for how an action should be physically realised in context. The proposed framework grounds this process in structured scene understanding, planning over an action space that combines manipulation primitives with context-aware execution choices, and closed-loop validation of action feasibility and outcomes. Preliminary results, obtained using a bimanual quadrupedal system, indicate the potential of this perspective for supporting more diverse manipulation behaviors. In this regard, they suggest a promising direction for future Vision-Language-Action models, where structured action representations of this kind could serve as supervision targets for learning manipulation policies grounded in both task semantics and execution constraints.**

***Index Terms*—Vision-Language Models, Zero-Shot Planning, Bimanual Manipulation, Scene Understanding**

## I. Introduction

Robotic manipulation remains an open problem as effective behavior requires not only accurate perception and control, but also the ability to interpret task context, reason about object relations, and adapt execution to high-level human instructions. These become more challenging in bi-manual scenarios, where the robot must coordinate multiple manipulators, decide how to distribute sub-tasks, and exploit parallel execution when possible. Recent advances in large-scale multi-modal models have opened new opportunities towards this direction. Vision-Language Models (VLMs) enable the integration of visual observations, natural language instructions, and semantic reasoning, allowing robots to infer task goals and generate context-aware action plans [1]–[4]. Prior work has demonstrated the benefits of multi-modal scene understanding and task specification [1], [5], while more recent approaches incorporate structured scene representations and closed-loop re-planning to improve robustness in dynamic settings [6]. In parallel, language-model-based methods for bimanual planning emphasize the need for explicit temporal and spatial coordination between manipulators [7], [8]. However, many existing approaches operate at the level of abstract action representations, without explicitly capturing how actions are executed in the scene.

In this work, a zero-shot language-conditioned framework for bimanual manipulation is explored in which task-level reasoning is addressed jointly with execution-level decision making. Preliminary observations with monolithic end-to-end approaches revealed limitations in capturing structured relationships between objects and execution constraints, motivating the need for more explicit intermediate representations. Instead of relying on a single end-to-end module, the proposed approach introduces a structured pipeline with intermediate representations for scene perception, planning, and execution. This enables reasoning over object relations, accessibility constraints, and high-level instructions. Preliminary results on tabletop manipulation scenarios highlight both the potential of the approach and the current limitations that motivate ongoing efforts towards more robust and scalable language-driven manipulation.

## II. Method

A pipeline for language-conditioned bimanual manipulation is proposed (see Fig. 1). The framework decomposes the decision process into a sequence of vision-language modules, rather than relying on a single monolithic model. Given a visual observation of the scene and a natural-language task instruction, the system first constructs a structured representation of the environment, capturing object relations and interaction constraints. Based on this representation, a sequence of manipulation actions is generated in a structured form that explicitly encodes execution choices. The plan is then reorganized into parallel execution stages to enable coordinated bimanual behaviors, and finally monitored through a validation module that tracks task progress and ensures consistency between expected and observed outcomes.

### A. Scene Perception and Structured Scene Representation

Given an RGB image $I_t$ acquired from the robot viewpoint at time $t$, together with 6D object pose estimates provided by FoundationPose [9] and prior knowledge of object dimensions, semantic and geometric properties of the workspace are extracted. The scene is represented as a graph

$$G_t = (V_t, R_t), \tag{1}$$

where $V_t = O_t \cup E_t$ is the set of scene entities, composed of object nodes $O_t$ and end-effector nodes $E_t$, and $R_t$ is the set of

* Equal contribution.

[1]The author is with the Humanoids and Human Centered Mechatronics (HHCM), Istituto Italiano di Tecnologia, Genova, Italy (e-mail: {name.surname}@iit.it)

[2]The author is with the University of Genova, DIBRIS, Genova, Italy.

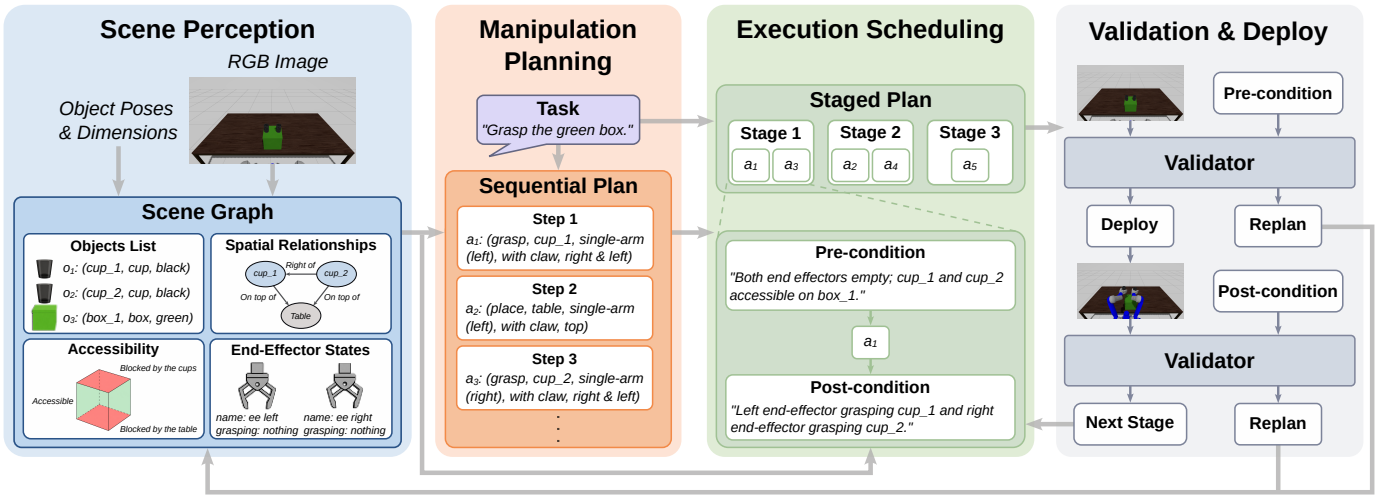



Fig. 1: Overview of the proposed language-conditioned bimanual manipulation pipeline. Starting from an RGB scene observation and object poses and dimensions, the framework builds a structured scene representation, generates a sequential manipulation plan, reorganizes compatible actions into parallel execution stages, and uses a validator to perform checks before and after execution.

pairwise spatial relations. Objects are represented as discrete instances with semantic attributes (e.g., category and color), while end-effectors are modeled through their interaction state (i.e., empty or holding an object). Unlike objects, end-effectors are not associated with persistent scene properties, since their pose depends on the current robot configuration. The scene structure is captured through pairwise relations

$$R_t \subseteq V_t \times T \times V_t, \tag{2}$$

where $T$ defines the set of admissible relation types (i.e., *on_top_of*, *right_of*, *in_front_of*, *grasped_by*). While entity identities and relations provide a structural description of the scene, they do not fully capture how objects can be interacted with. Specifically, for each object, the six principal sides (left, right, front, back, top, bottom) are annotated with their interaction status. Each side is described in terms of *accessibility*, indicating whether it is free or blocked by other objects or by the support surface, and *reachability*, indicating whether it can be approached by the robot given its kinematic constraints.

### B. Manipulation Planning

Based on the scene representation $G_t$, the next step is to generate a manipulation plan from a task instruction $H_t$ provided in natural language. This module produces a sequence of manipulation actions that can be executed step by step. The plan is modeled as an ordered set

$$\Pi_t = (a_1, a_2, \ldots, a_n), \tag{3}$$

where $a_k$ denotes a manipulation action. Each action in the sequence is predicted by a vision-language model in a structured form

$$a_k = (\text{action}_k,\ \text{target}_k,\ \text{mode}_k,\ \text{contact}_k,\ \text{side}_k), \tag{4}$$

where $\text{action}_k$ denotes the manipulation primitive (e.g., grasp, place, push, release), $\text{target}_k$ identifies the object involved, $\text{mode}_k$ specifies whether the action is performed with one or both arms, $\text{contact}_k$ indicates the type of gripper interaction, and $\text{side}_k$ defines the object side from which the interaction should be executed. This structured prediction makes explicit not only which, but also how each planned action should be performed. In particular, we explore how the VLM can align the structured scene representation with the generation of a manipulation plan that satisfies the task while respecting the geometric and interaction constraints encoded during perception. By selecting a specific side, the model effectively restricts the set of feasible interaction regions, reasoning about how the object can be approached and grasped.

### C. Parallel Execution Scheduling

The manipulation plan $\Pi_t$ is initially generated as a sequential list of actions. Actions are grouped based on their compatibility, considering object interactions, spatial constraints, and the usage of the robot end-effectors. For this reason, a dedicated scheduling module reorganizes this sequence into parallel execution stages. This is done while preserving the original action content and ensuring that concurrent execution does not introduce conflicts. The resulting stage sequence is

$$S_t = (s_1, \ldots, s_m), \tag{5}$$

where each stage $s_i$ is a set of mutually compatible actions to be executed in parallel, e.g., $s_i = \{a_p, a_q\}$, with $a_p, a_q \in A_k$. For each stage, the module additionally defines a *pre-condition* and a *post-condition*, describing the required scene state before the stage starts and the expected state after its completion. Formally, for each stage $s_i$, these are denoted as $c_i^{pre}$ and $c_i^{post}$. This mechanism improves robustness to unexpected changes in the environment and execution failures.

### D. Validator

To monitor execution and ensure consistency between the expected and the observed evolution of the scene, a validation module is introduced. The validator evaluates whether a condition $c$ is satisfied in the current scene, based on both the RGB observation $I$ and the structured object representation $O$. It can be defined as a function

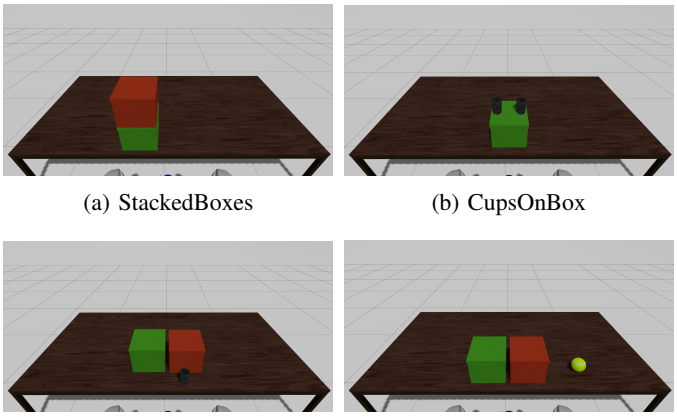
$$\mathcal{V}(I, O, c) \in \{0, 1\}, \tag{6}$$

where the output indicates whether the condition holds in the current state. During execution, validation is performed at the level of execution stages $s_i \in S_t$. Before executing a stage, the corresponding precondition $c_i^{pre}$ is evaluated on the current observation $(I_t, O_t)$ to verify that all actions within the stage can be safely executed. If the condition is satisfied, the actions in the stage are executed concurrently, producing an updated observation $I_{t+1}$. A post-condition $c_i^{post}$ is then evaluated on $(I_{t+1}, O_t)$ to verify that the expected outcome of the stage has been achieved. The object representation $O_t$ provides a consistent mapping between symbolic object identifiers used in the conditions and the corresponding entities in the scene, preventing ambiguities in object references. This representation is kept fixed during execution and is updated only when re-planning is triggered. To account for ambiguity in the visual observations, a conservative policy is adopted: a condition is considered satisfied unless there is clear evidence that it is violated. In this case, the system triggers a re-planning step and the full pipeline is restarted from the current observation. Through this validation mechanism, execution is continuously monitored and corrected, enabling the overall framework to operate in a closed loop.

## III. Experiments

### A. Experiment Setup

**Implementation Details.** Experiments are conducted in simulation using Gazebo Sim [10] within a ROS 2-based framework, featuring *KYON* [], a bimanual hybrid wheel-legged quadruped. Each arm has 5 degrees of freedom, is equipped with a Dagana end-effector [11], and is controlled through joint-space references computed by a QP-based inverse kinematics module [12]. In all experiments, the robot is kept fixed w.r.t. the workspace, and locomotion is not considered. Scene perception is provided by a fixed external RGB camera placed at a height of 1.70 m and oriented to observe the workspace from the robot perspective, providing a top down-view of object arrangements. Object pose estimates and tracking are obtained through the perception pipeline described in Sec. II-A. Throughout the experiments, a perfect manipulation is assumed, restricting the failures to the VLM-based modules and assessing their performance. For the execution of the entire pipeline, OpenAI models are used, in particular GPT-o3 and GPT-5.2. GPT-o3 is used for the manipulation planning module, while GPT-5.2 is used for the remaining modules.



(a) StackedBoxes (b) CupsOnBox

(c) BoxesAside (d) TennisBall

Fig. 2: Representative views of the four manipulation scenarios considered in the experimental evaluation.

**Task Scenarios** To evaluate the capabilities of the proposed framework, four tabletop manipulation scenarios are considered, each defined by a fixed initial configuration and a natural language instruction (see Fig. 2). Each setup is evaluated over 25 runs from the same initial configuration, with observations updated based on the executed plan and validation feedback. Task details are summarized in Table I. The scenarios include object dependency handling (*StackedBoxes*), coordinated removal of multiple objects (*CupsOnBox*), multi-step workspace reconfiguration (*BoxesAside*), and dynamic scene adaptation (*TennisBall*), where an additional object is introduced during execution, therefore requiring re-planning.

**Evaluation Metrics** The performance of the proposed framework is assessed in terms of computational cost and task execution, using *Inference Latency* (IL) and *Success Rate* (SR). *Inference Latency* measures the average time required by each module to produce its output. *Success Rate* evaluates the percentage of successful executions, reflecting the ability of the system to correctly complete the assigned tasks; it is computed both at the module level, to assess the reliability of individual components, and at the scenario level, to evaluate end-to-end task completion.

### B. Results

**Inference Latency** Fig. 3 reports the average inference time of each module across scenarios. Execution scheduling and validation exhibit low and stable latency, largely independent of the task configuration. In contrast, scene perception and manipulation planning show higher variability, as their cost depends on scene complexity and reasoning requirements. *StackedBoxes* results in the lowest overall latency, while *BoxesAside* shows the highest planning time. This reflects the increased reasoning complexity of this scenario, where the system must infer a multi-step strategy involving both prehensile and non-prehensile actions under constrained accessibility. The higher cost of the planning module is also influenced by the use of GPT-o3, characterized by a stronger reasoning capability in complex decision-making settings.

TABLE I: Task definition and reasoning requirements for each scenario.

| Scenario | Instruction | Scene Setup | Required Reasoning |
|---|---|---|---|
| StackedBoxes | "Grasp the green box. Be delicate. The red box is very fragile." | Red box on top of green box | Remove blocking object while preserving safety constraints |
| CupsOnBox | "Grasp the green box. Be careful with the glasses." | Two cups placed on top of green box | Coordinate dual-arm actions for efficient object removal |
| BoxesAside | "Grasp the green box." | Red box on the side and cup in front of target | Sequence actions to clear multiple obstacles |
| TennisBall | "Place the ball on the red box and keep the green box top free." | Ball near red box; additional tennis ball introduced on green box during execution | Detect scene changes and replan under constraints |

**Success Rate** Table II reports module-level success rates. Overall, all modules achieve success rates above 91%, indicating a generally reliable pipeline. The lowest values are observed for scene perception and validation, which directly operate on RGB observations and are therefore more sensitive to visual ambiguities and errors in spatial interpretation. This aspect is particularly critical for scene perception, as inaccuracies at this stage can propagate to downstream modules and affect the overall execution. In contrast, modules operating on structured representations of the scene exhibit higher reliability, with execution scheduling achieving near-perfect performance. Manipulation planning shows slightly lower performance compared to scheduling, reflecting the higher complexity of inferring consistent action sequences from the scene and the task instruction.

TABLE II: Module-level success rate (%).

| Module | Model | Success Rate (%) |
|---|---|---|
| Scene Perception | GPT-5.2 | 91.63 |
| Manipulation Planning | GPT-o3 | 95.83 |
| Execution Scheduling | GPT-5.2 | 99.15 |
| Validator | GPT-5.2 | 93.60 |

Table III reports scenario-level success rates. Three scenarios achieve perfect task completion, while *CupsOnBox* reaches a lower success rate of 70.0%. This reduced performance is mainly due to perceptual ambiguity during bimanual manipulation, where the proximity of the end-effectors to the box can lead the system to assume that the box has been grasped, causing premature task completion. Despite these difficulties, the re-planning mechanism enables recovery from most intermediate failures, allowing many executions to still converge to task completion.

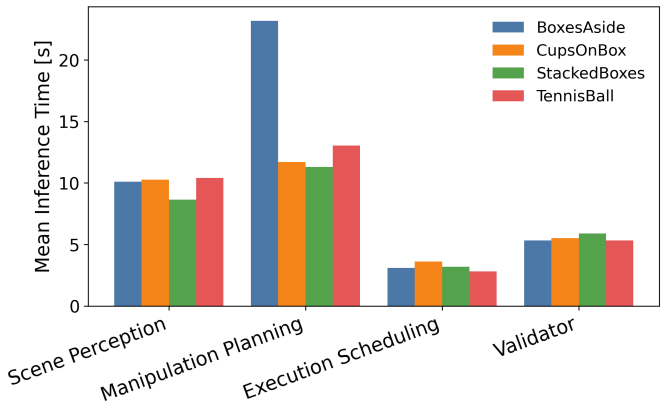



Fig. 3: Mean inference time of the different modules across the considered scenarios.

TABLE III: Scenario-level success rate (%).

| Scenario | Success Rate (%) |
|---|---|
| BoxesAside | 100.0 |
| StackedBoxes | 100.0 |
| CupsOnBox | 70.0 |
| TennisBall | 100.0 |

A qualitative analysis of failures indicates that most errors originate from ambiguities in perception and validation rather than from deficiencies in high-level planning. In particular, conditions involving object accessibility, available placement space, and spatial proximity remain the most challenging to assess reliably from RGB observations, especially when the scene becomes cluttered or when the end-effectors partially occlude the objects.

## IV. Conclusion

The experimental results suggest that this formulation is a promising direction for language-conditioned bimanual manipulation. However, the current evaluation is conducted under a set of simplifying assumptions that motivate its preliminary nature. In particular, manipulation is assumed to be perfect, preventing execution failures from propagating to the observed scene and thus yielding cleaner and more predictable interactions. In addition, the visual setup was tuned on simulated images, which are less noisy and more structured than real-world settings. Navigation is also excluded, and a fixed external camera is used instead of an onboard perception system. Future work will therefore focus on relaxing these assumptions by incorporating manipulation errors into the evaluation, handling more ambiguous and noisy observations, and extending the framework to more realistic settings. Additional efforts will focus on evaluating the approach on longer-horizon and more diverse multi-object scenarios, as well as on enriching the validator with more informative feedback on failure causes. Finally, this framework will also be investigated in the context of Vision-Language-Action systems, as it may help define more informative supervision targets for learning policies that must connect task-level intent with physically grounded execution.

## ACKNOWLEDGMENT

This project has received funding from the European Union's Horizon Europe Framework Programme under grant agreement N°101070596 (euROBIN).

## REFERENCES

[1] M. Shridhar, L. Manuelli, and D. Fox, "Cliport: What and where pathways for robotic manipulation," in *Conference on robot learning*. PMLR, 2022, pp. 894–906.

[2] 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.

[3] 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.

[4] A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain *et al.*, "Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 6892–6903.

[5] 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)*. IEEE, 2024, pp. 21–26.

[6] Z. Lou, K. Xu, Z. Zhou, and R. Xiong, "Explorevlm: Closed-loop robot exploration task planning with vision-language models," *arXiv preprint arXiv:2508.11918*, 2025.

[7] K. Chu, X. Zhao, C. Weber, M. Li, W. Lu, and S. Wermter, "Large language models for orchestrating bimanual robots," in *2024 IEEE-RAS 23rd International Conference on Humanoid Robots (Humanoids)*. IEEE, 2024, pp. 328–334.

[8] K. Chu, X. Zhao, C. Weber, and S. Wermter, "Llm+ map: Bimanual robot task planning using large language models and planning domain definition language," *arXiv preprint arXiv:2503.17309*, 2025.

[9] 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*, 2024, pp. 17 868–17 879.

[10] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2004, pp. 2149–2154.

[11] E. Del Bianco, D. Torielli, F. Rollo, D. Gasperini, A. Laurenzi, L. Baccelliere, L. Muratore, M. Roveri, and N. G. Tsagarakis, "A high-force gripper with embedded multimodal sensing for powerful and perception driven grasping," in *2024 IEEE-RAS 23rd International Conference on Humanoid Robots (Humanoids)*. IEEE, 2024, pp. 149–156.

[12] A. Laurenzi, E. M. Hoffman, L. Muratore, and N. G. Tsagarakis, "Cartesi/o: A ros based real-time capable cartesian control framework," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 591–596.