Special Issue: Foundation Models and Neural-Symbolic AI for Robotics

The International Journal of Robotics Research
2025, Vol. 0(0) 1–19

DOI: 10.1177/02783649251378196
journals.sagepub.com/home/ijr

LLM-GROP: Visually grounded robot task and motion planning with large language models

Xiaohan Zhang[1,*], Yan Ding[1,2,3,*], Yohei Hayamizu[1,*], Zainab Altaweel[1,*], Yifeng Zhu[4], Yuke Zhu[4], Peter Stone[4,5], Chris Paxton[6] and Shiqi Zhang[1]

Abstract

Task planning and motion planning are two of the most important problems in robotics, where task planning methods help robots achieve high-level goals and motion planning methods maintain low-level feasibility. Task and motion planning (TAMP) methods interleave the two processes of task planning and motion planning to ensure goal achievement and motion feasibility. Within the TAMP context, we are concerned with the mobile manipulation (MoMa) of multiple objects, where it is necessary to interleave actions for navigation and manipulation. In particular, we aim to compute where and how each object should be placed given underspecified goals, such as "set up dinner table with a fork, knife and plate." We leverage the rich common sense knowledge from large language models (LLMs), for example, about how tableware is organized, to facilitate both task-level and motion-level planning. In addition, we use computer vision methods to learn a strategy for selecting base positions to facilitate MoMa behaviors, where the base position corresponds to the robot's "footprint" and orientation in its operating space. Altogether, this article provides a principled TAMP framework for MoMa tasks that accounts for common sense about object rearrangement and is adaptive to novel situations that include many objects that need to be moved. We performed quantitative experiments in both real-world settings and simulated environments. We evaluated the success rate and efficiency in completing long-horizon object rearrangement tasks. While the robot completed 84.4% real-world object rearrangement trials, subjective human evaluations indicated that the robot's performance is still lower than experienced human waiters.

Keywords

task and motion planning, large language models, mobile manipulation

Introduction

Robots require task planning methods to sequence symbolic actions for accomplishing complex tasks. They also need motion planning methods to compute trajectories that realize these symbolic actions while ensuring motion-level feasibility. Task and motion planning (TAMP) refers to a family of algorithms that integrate task and motion planning processes to compute motion trajectories that can be directly executed on robot hardware to achieve task-level goals (Garrett et al., 2021; Zhao et al., 2024). While most existing TAMP algorithms are designed for purely manipulation (i.e., requiring no navigation) domains, robots may need to handle objects located far apart, requiring a combination of

[1]The State University of New York at Binghamton, Binghamton, NY, USA
[2]Shanghai AI Laboratory, Shanghai, China
[3]OneStar Robotics, Suzhou, China
[4]The University of Texas at Austin, Austin, TX, USA
[5]Sony AI
[6]Hello Robot, Martinez, CA, USA

*Equal contribution

Corresponding author:
Shiqi Zhang, The State University of New York at Binghamton, 4400 Vestal Pkwy E, Binghamton , NY 13902-4600, USA.
Email: zhangs@binghamton.edu

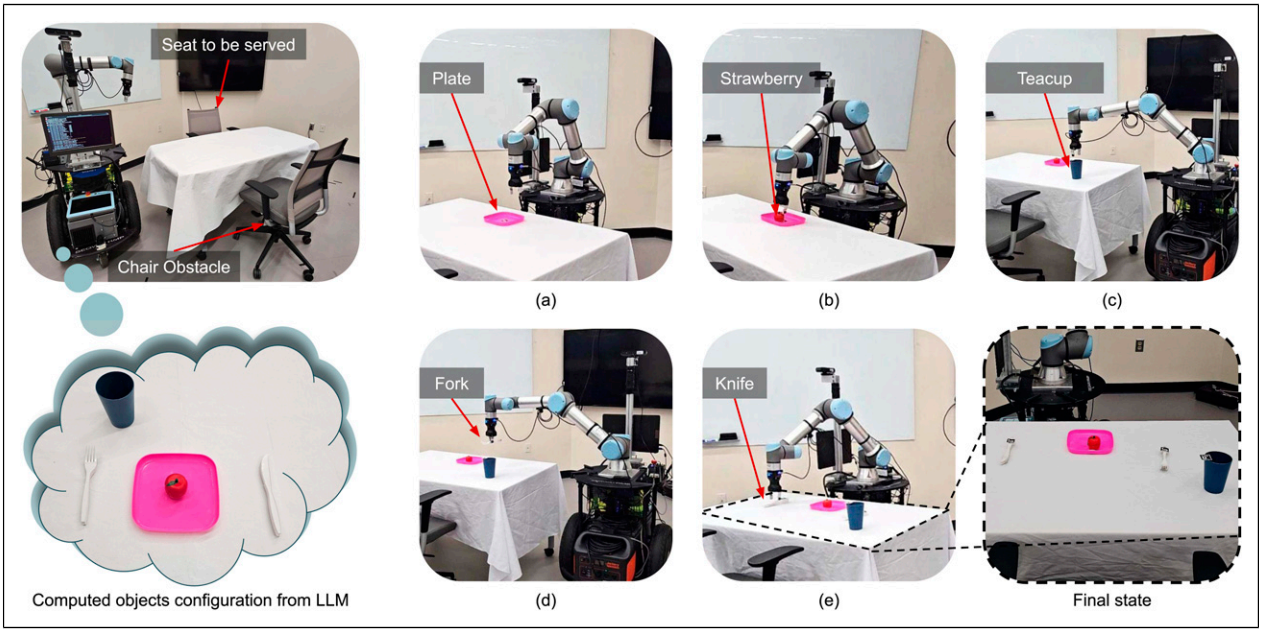

Figure 1. An illustration of our mobile manipulation (MoMa) domain, where a mobile manipulator is tasked with setting a dining table. The robot must arrange several tableware items, including a knife, a fork, a plate, a cup mat, and a mug. These objects are located on other tables, and the environment also includes randomly generated obstacles (e.g., chairs), which are not accounted for in the pre-built map. The robot must compute semantically specified goal configurations of the objects and (task and motion) plans for rearranging the objects on the target table. The computed plan includes both navigation and manipulation behaviors.

navigation and manipulation actions. In this article, we study *mobile manipulation* (MoMa) domains in which robots perform both navigation and manipulation tasks. This article focuses on addressing MoMa challenges by developing TAMP methods that are visually grounded and capable of leveraging common sense knowledge to achieve underspecified goals.

Multi-object rearrangement is an essential skill for service robots to perform everyday tasks such as setting tables, organizing bookshelves, and loading dishwashers (Szot et al., 2022; Weihs et al., 2021). These tasks require robots to demonstrate both manipulation and navigation capabilities. For instance, a robot tasked with setting a dinner table may need to retrieve tableware items like forks and knives from different locations and place them onto a table surrounded by chairs, as illustrated in Figure 1. To complete this task, the robot must accurately position the tableware in semantically specified configurations (e.g., placing the fork to the left of the knife) and efficiently navigate indoor spaces while avoiding obstacles such as chairs or humans, whose locations are not known in advance.

A variety of mobile manipulation systems have been developed for object rearrangement tasks (Goodwin et al., 2022; Liu et al., 2022b; Wei et al., 2023; Huang et al., 2019; Gu et al., 2022; King et al., 2016; Cheong et al., 2020; Vasilopoulos et al., 2021). Most of these systems require explicit instructions, such as arranging similarly colored items in a line or placing them in a specific shape on a table (Cheong et al., 2020; Goodwin et al., 2022; Gu et al., 2022; Huang et al., 2019; Liang et al., 2022; Vasilopoulos et al., 2021). However, real-world user requests are often underspecified; for example, there are many ways to set a table, but some are preferred more than others. How does a robot determine that a fork should be placed to the left of a plate and a knife to the right? Reasoning about such societal conventions requires substantial common sense knowledge. Existing research has shown that large language models (LLMs), such as ChatGPT (OpenAI, 2023), possess a significant amount of such common sense knowledge (Liu et al., 2021a). In the past, researchers have equipped mobile manipulators with semantic information using machine learning methods (Liu et al., 2022b, 2022a; Wei et al., 2023; Zhang and Chai, 2021). However, these methods rely on training data, limiting their applicability for robots performing diverse service tasks in open worlds, where data collection can be difficult.

To compute semantically specified goal configurations and enable mobile manipulation for object rearrangement, we introduce LLM-GROP, which stands for Large Language Model for Grounded RObot Task and Motion Planning, our approach leverages common sense knowledge for planning object rearrangement tasks. LLM-GROP first uses an LLM to generate *symbolic* spatial relationships among objects, for example, placing a fork and a knife to the left and right of a plate, respectively. These symbolic relationships are then mapped to *geometric* spatial relationships, whose feasibility is evaluated by a motion planning system. For example, some areas of a table may be more

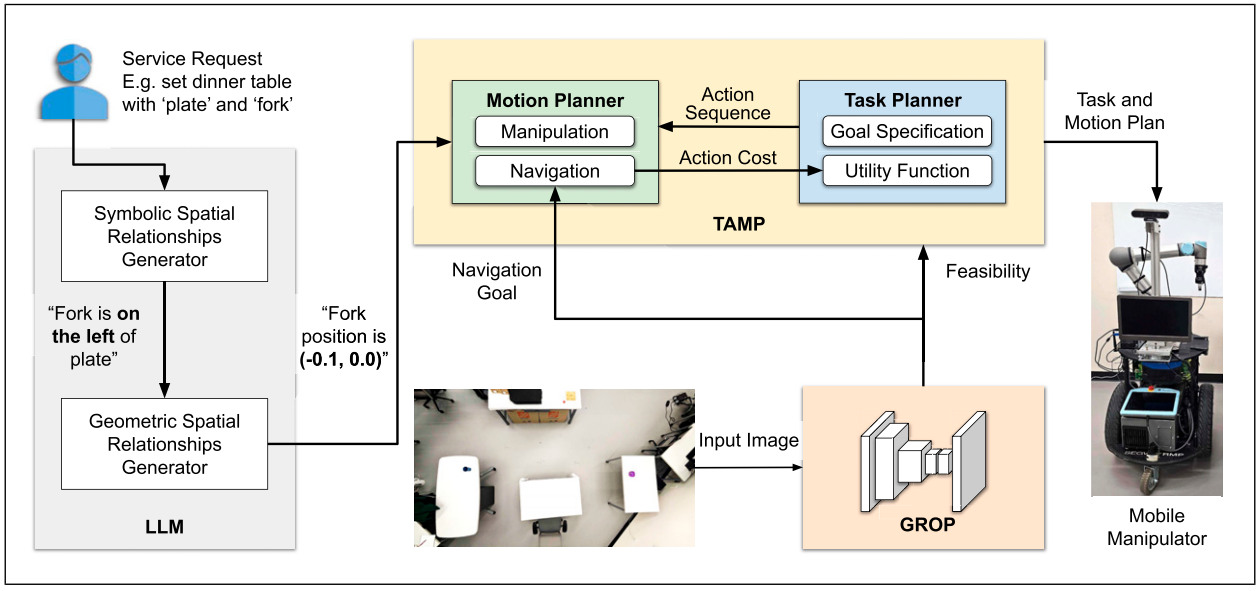

Figure 2. An overview of the LLM-GROP approach. LLM-GROP takes service requests from humans for setting tables and produces a task-motion plan that the robot can execute. LLM-GROP is comprised of two key components: the LLM and the Task and Motion Planner. The LLM is responsible for creating both symbolic and geometric spatial relationships between the tableware objects. This provides the necessary context for the robot to understand how the objects should be arranged on the table. The Task and Motion Planner generates the plan for the robot to execute based on the information provided by the LLM. An important component of LLM-GROP is GROP that takes a top-down view image as the input and suggests standing positions to facilitate MoMa behaviors. GROP is trained exclusively using simulation data. In the real world, the robot estimates poses of objects and builds a digital twin for task and motion planning. Details of GROP are shown in Figure 3.

feasible for object placement than others. Finally, computer vision methods are employed to optimize the feasibility and efficiency of task-motion plans, maximizing long-term utility by balancing motion feasibility and task-completion efficiency.

We applied LLM-GROP in a dining domain where a mobile manipulator is tasked with setting a table based on user instructions. The robot is provided with a set of tableware objects and must compute a tabletop configuration that adheres to common sense rules while also generating a task-motion plan to execute the arrangement. To evaluate the performance of our approach, we collected user ratings of different table settings for subjective evaluation. Our results showed that LLM-GROP improved user satisfaction compared to existing object rearrangement methods while maintaining similar or lower cumulative action costs. Additionally, LLM-GROP was demonstrated and evaluated on a real robot.

This article builds on our previous research, which introduced the initial version of LLM-GROP (Ding et al., 2023b) and its vision component, GROP (Zhang et al., 2022b). GROP used visual perception to identify optimal standing positions, maximizing the feasibility and efficiency of both manipulation and navigation actions. LLM-GROP extends this by incorporating LLMs to generate tabletop configurations that adhere to common sense principles and are feasible for TAMP systems. Compared to the two conference papers, the primary contribution of this article is the unification of these two algorithms and systems, providing a cohesive presentation of all LLM-GROP components. We have updated illustrative examples and overview figures for improved clarity (Figures 1, 2, and 5). Additional experiments were conducted to evaluate LLM-GROP using different LLMs (Table 5), on robot hardware (Figure 6), and through subjective evaluations of real-robot performance with human participants (Table 4). Challenges and opportunities are discussed toward the end of the article.

Related work

In this section, we first summarize task and motion planning literature, then introduce the mobile manipulation problem in object rearrangement domains, and finally discuss foundation models for robot planning.

Task and motion planning (TAMP)

TAMP methods aim to compute plans that fulfill task-level goals while maintaining motion-level feasibility, as reviewed in recent articles (Garrett et al., 2021; Lagriffoul et al., 2018; Zhao et al., 2024). Several TAMP algorithms have been introduced in recent years (e.g., (Chitnis et al., 2016; Chitnis et al., 2019; Dantam et al., 2018; Ding et al., 2022; Erdem et al., 2011; Garrett et al., 2018; Gravot et al.,

2005; Kim et al., 2019; Kim and Shimanuki, 2020; Lagriffoul et al., 2014; Plaku et al., 2007; Srivastava et al., 2014; Wang et al., 2018; Zhu et al., 2020). We distinguish a few subareas of TAMP that are closest to our research on learning to visually ground symbolic spatial relationships towards planning efficient and feasible task-motion behaviors under uncertainty.

When high-level actions only take a few seconds, TAMP algorithms can focus mostly on action feasibility constraints without fully optimizing high-level plan efficiency. However, when there are actions that take significant time to execute (e.g., long-distance navigation), task-completion efficiency cannot be overlooked. Some recent methods have considered efficiency in different aspects of TAMP, such as planning task-level optimal behaviors in navigation domains (Lo et al., 2020), integrating reinforcement learning with symbolic planning in dynamic environments (Jiang et al., 2019a), computing safe and efficient plans for urban driving (Ding et al., 2020), and optimizing robot navigation actions under the uncertainty from motion and sensing (Thomas et al., 2021). In contrast to those methods that do not have a perception component, the main difference is that LLM-GROP visually grounds symbols (about spatial relationships) to probabilistically evaluate action feasibility for task-motion planning. Another difference is that LLM-GROP leverage LLMs for computing semantically meaning goal configurations.

While most TAMP methods assume a fully observable and deterministic world (Garrett et al., 2021), some have been developed to account for the uncertainty from perception and action outcomes (Akbari et al., 2020; Garrett et al., 2020; Hadfield-Menell et al., 2015; Kaelbling and Lozano-Pérez, 2013; Nouman et al., 2021; Phiquepal and Toussaint, 2019). For instance, the work of Kaelbling and Lozano-Pérez extended the "hierarchical planning in the now" approach to address both current-state uncertainty and future-state uncertainty (Kaelbling and Lozano-Pérez, 2013). Going beyond those methods that aim to maintain plan feasibility to complete tasks under high-level uncertainty, we consider uncertainty in the robot motion and also incorporate task-completion efficiency into the optimization of robot behaviors. As a result, our LLM-GROP algorithm is particularly suitable for TAMP domains that require robot operations over extended periods of time, such as long-distance navigation.

Existing research has shown that visual information can be used to help robots predict plan feasibility, including task-level feasibility (Driess et al., 2020a; Zhu et al., 2020), and motion-level feasibility (Driess et al., 2020b; Wells et al., 2019). Those methods were developed to maximize task completion rate in manipulation domains, and actions that take relatively long time (such as long-distance navigation) were not included in their evaluations. LLM-GROP incorporates efficiency into plan optimization, while leveraging common sense from LLMs for computing goal configurations. For instance, when highly feasible plans have very high costs, LLM-GROP supports the flexibility of executing slightly less feasible plans with much lower costs. LLM-GROP achieves this desirable trade-off between feasibility and efficiency by probabilistically evaluating plan feasibility, which is not supported by the above-mentioned methods.

MoMa for object rearrangement

Rearranging objects is a critical task for service robots, and much research has focused on moving objects from one location to another and placing them in new positions. Examples include the Habitat Rearrangement Challenge (Szot et al., 2022) and the AI2-THOR Rearrangement Challenge (Weihs et al., 2021). There is rich literature on object rearrangement in robotics (Cheong et al., 2020; Goodwin et al., 2022; Gu et al., 2022; Huang et al., 2019; Liang et al., 2022; Vasilopoulos et al., 2021; Zhang et al., 2022b). A common assumption in those methods is that a goal arrangement is part of the input, and the robot knows the exact desired positions of objects. ALFRED (Shridhar et al., 2020) proposed a language-based multi-step object rearrangement task, for which a number of solutions have been proposed that combine high-level skills (Blukis et al., 2022; Min et al., 2021), and which have recently been extended to use LLMs as input (Inoue and Ohashi, 2022). However, these operate at a very coarse, discrete level, instead of making motion-level and placement decisions, and thus can't make granular decisions about common-sense object arrangements. By contrast, our work accepts underspecified instructions from humans, such as setting a dinner table with a few provided tableware objects. LLM-GROP has the capability to do common sense object rearrangement by extracting knowledge from LLMs, and operates both on a high level and on making motion-level placement decisions.

Object arrangement is a task that involves arranging items on a tabletop to achieve a specific functional, semantically valid goal configuration. This task requires not only the calculation of object positions but also adherence to common sense, such as placing forks to the left and knives to the right when setting a table. Previous studies in this area, such as Kapelyukh et al. (2022), Liu et al. (2022a, 2022b), and Wei et al. (2023), focused on predicting complex object arrangements based on vague instructions. For instance, StructFormer (Liu et al., 2021b) is a transformer-based neural network for arranging objects into semantically specified structures based on natural-language instructions. By comparison, our approach LLM-GROP utilizes an LLM for common sense acquisition to avoid the need of demonstration data for computing object positions. Additionally,

we optimize the feasibility and efficiency of plans for placing tableware objects.

There exist methods for predicting complex object arrangement using web-scale diffusion models (Kapelyukh et al., 2022). Their approach, called DALL-E-Bot, enables a robot to generate goal images using DALL-E (Ramesh et al., 2022) based on a text description derived from an initial scene. The robot aligns objects between the initial and generated images and accordingly arranges objects in a tabletop scenario based on the inferred poses. Similar to DALL-E-Bot, LLM-GROP achieves zero-shot performance using pre-trained models, but it is not restricted to a single top-down view of a table. Additionally, LLM-GROP leverages LLMs to generate symbolic and geometric goal specifications from underspecified instructions, operates in full-room mobile manipulation settings, and integrates task and motion planning (TAMP) to reason about feasibility and uncertainty in both navigation and manipulation, producing efficient and physically executable plans.

Robot planning with foundation models

Many LLMs have been developed in recent years, such as BERT (Devlin et al., 2018), GPT-3 (Brown et al., 2020), ChatGPT (OpenAI, 2023), CodeX (Chen et al., 2021), and OPT (Zhang et al., 2022a). These LLMs can encode a large amount of common sense (Liu et al., 2021a) and have been applied to robot task planning (Ahn et al., 2022; Ding et al., 2023a; Huang et al., 2022a, 2022b; Kant et al., 2022; Liu et al., 2022a, 2023; Rana et al., 2023; Singh et al., 2022; Wu et al., 2023; Zhao et al., 2023). For instance, the work of Huang et al. showed that LLMs can be used for task planning in household domains by iteratively augmenting prompts (Huang et al., 2022a). SayCan is another approach that enabled robot planning with affordance functions to account for action feasibility, where the service requests are specified in natural language (e.g., "make breakfast") (Ahn et al., 2022). Compared with those methods, LLM-GROP optimizes both feasibility and efficiency while computing semantically valid geometric configurations.

More recently, vision-language models (VLMs) have been incorporated into robot planning (Huang et al., 2023, 2024; Yang et al., 2024; Zhang et al., 2024). Among those, the most relevant is VLM-TAMP (Yang et al., 2024), which is a hierarchical planning approach that leverages a VLM to generate semantically-specified and intermediate subgoals that guide a task and motion planner. VLM-TAMP used a VLM to unify the processing of both text and image prompts, and is strong in tabletop visual scene analysis compared with our work. Compared with their approach, ours is capable of visual understanding for MoMa behaviors, such as probabilistically evaluating their feasibility.

Next, we present a statement of the MoMa problem (namely object rearrangement) in the Problem Statement section, including assumptions, formats of its input and output, and success criteria, before we discuss our approach in the Method section.

Problem statement

While we are generally concerned with a mobile manipulation domain, algorithms and systems developed in this article are demonstrated and evaluated using a specific task of object rearrangement. Specifically, the objective is to rearrange multiple tableware objects, which are initially scattered at different locations, into a tabletop configuration that is semantically valid and aligns with common sense. The domain includes N objects Obj. There are obstacles (tables and chairs in our case) that prevent the robot from navigating to some positions in the domain. The robot is provided with prior knowledge about table shapes and locations. Chairs, on the other hand, can only be sensed at planning time. Location l is a symbolic concept that corresponds to a set of obstacle-free 2D poses (X), where each pose ($x \in X$) specifies a 2D position and an orientation. The robot needs to move each object $o \in Obj$ from its initial location to a goal position.

Actions

The robot is equipped with skills of performing a set of symbolic (task-level) actions denoted as A: $A^n \cup A^m$, where A^n and A^m are *navigation* actions and *manipulation* actions respectively. A navigation action $a^n_{l,l'} \in A^n$ is specified by its initial and goal locations, l, $l' \in L$, where L includes a set of symbolic locations. A manipulation action, $a^m_{o,l} \in A^m$, is specified by an object to be manipulated, $o \in Obj$, and a symbolic location, $l \in L$, to which the robot navigates and performs the manipulation action. We consider two types of manipulation actions of loading and unloading, represented by a^{m+} and a^{m-} respectively. Actions are defined via preconditions and effects. For instance, the action `load`(o_1) has preconditions of `at (robot,`l_1`)` and `at`(o_1, l_1), meaning that to load the object o_1, the object must be co-located with the robot at the location l_1. The effects of `load`(o_1) include o_1 being moved into the robot's hand, that is, `inhand`(o_1).

Perception

The robot visually perceives the environment through top-down views over the areas where manipulation and navigation actions are performed. We use IM to represent a 2D image that captures the current obstacle configuration, such as chairs around tables, as shown in the "Image Input" of Figure 2 (bottom right). To facilitate robot learning, we provide a dataset (as

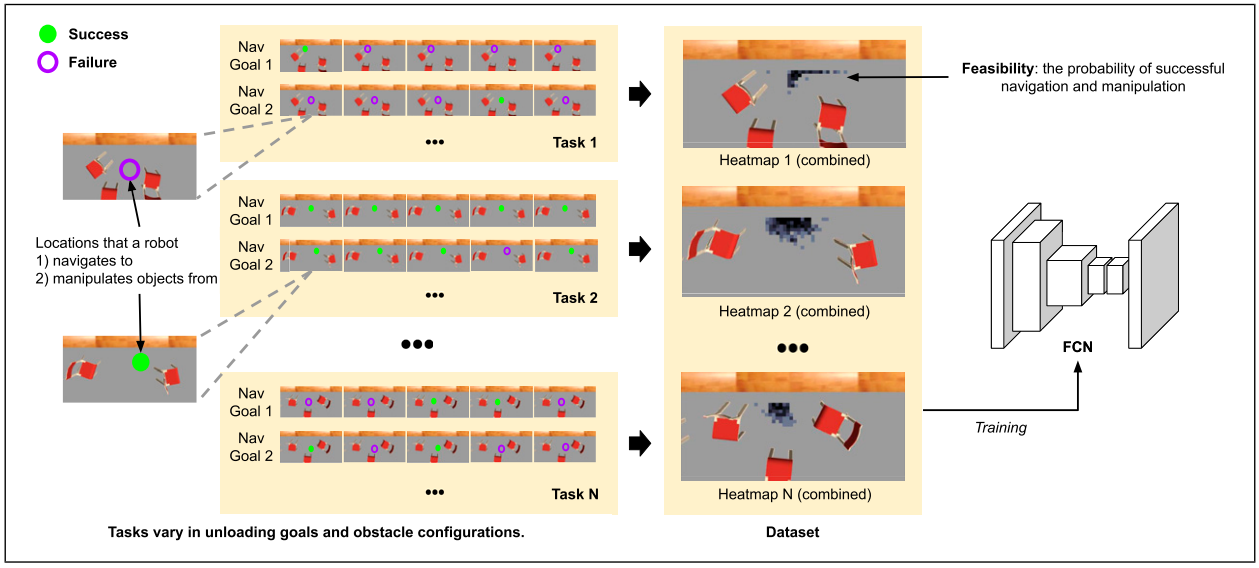

Figure 3. An overview of the data collection and training process in GROP. A *task* corresponds to one "unloading goal" on the table, as well as a configuration of obstacles (chairs in our case). Given a task, every pixel is considered a navigation goal—the robot attempts to navigate there, and unload an object from there. This navigation-manipulation process is referred to as a *trial*. The robot performs multiple trials for each navigation goal, which yields a *feasibility* value for that particular location. The feasibility values together form one *heatmap* for each task. In our *dataset*, each instance is a top-down view image, whose label is the corresponding heatmap. The "Dataset" box shows a few "combined heatmaps" where heatmaps are overlaid onto the corresponding images. Training with the dataset generates an FCN that is used for two purposes: (1) evaluating the feasibility of task-level actions and (2) selecting motion-level navigation goals. Finally, GROP incorporates both efficiency (measured by action costs) and feasibility to compute task-motion plans for a mobile manipulator.

illustrated in the "Dataset" box of Figure 2). Each instance includes a top-down view image, and a target object with a predefined position, while each label is in the form of a heatmap. Each pixel of a heatmap is associated with a 2D position, and has a "feasibility" value that represents the success rate of the robot navigating to the 2D position, and manipulating the target object from there.

A map is generated in a pre-processing step, and provided to the robot as prior information for navigation purposes using rangefinder sensors.

Uncertainty

We consider uncertainty in navigation and manipulation behaviors. For instance, the robot can fail in navigation (at planning or execution time) when its goal is too close to tables or chairs, and it can fail in manipulation when it is not close enough to the target position. Note that uncertainties are treated as black boxes in this work.

Specifically, the outcome of performing navigation action $a^n_{l,l'}$ to goal pose x is deterministic at the task level, but is non-deterministic at the motion level. In other words, the robot will end up in position x', which is not necessarily the same as x. This setting captures the fact that a mobile robot never achieves its exact 2D navigation goal (due to its imperfect localization and actuation capabilities), though successfully navigating to an area (l) is generally possible.

We focus on the interdependency between navigation and manipulation actions. For instance, the execution-time uncertainty from navigation actions results in different standing positions of the robot, which makes the outcomes of manipulation actions non-deterministic. This challenge generally exists in mobile manipulators. We assume no noise in the execution of manipulation actions (loading and unloading) to objects within a reachable area.

Large language models

It is assumed that an off-the-shelf large language model (LLM) is available and can be used by a robot to extract common sense knowledge. An LLM is a type of computational model designed for natural language processing tasks such as language generation. The input of LLMs in this object rearrangement domain includes a description of the problem as well as the format of the output. It is evident that an LLM's performance can be boosted by providing a few inference examples (Kaplan et al., 2020), so MoMa practitioners might want to extend the input by further including such examples.

Format of solution

A solution is in the form of a task-motion plan $p = \langle p^t, p^m \rangle$, where task plan p^t is of the form $\langle a_0^n, a_0^m, a_1^n, a_1^m, \ldots \rangle$, indicating that navigation and manipulation actions are interleaved. Motion plan p^m is of the form $\langle \xi_0^n, \xi_0^m, \xi_1^n, \xi_1^m, \ldots \rangle$, and ξ_i^n (or ξ_i^m) is a trajectory in continuous space for implementing symbolic action a_i^n (or a_i^m).

Quality of solution

The quality of task-motion plan p is evaluated using a utility function $\mathcal{U}(p)$, which considers both feasibility and efficiency of plan p:

$$\mathcal{U}(p) = \mathcal{R} \cdot \mathcal{F}(p) - \mathcal{C}(p), \tag{1}$$

where $\mathcal{F}(p) \in [0, 1]$ is the plan feasibility (i.e., the probability that p can be successfully executed), $\mathcal{C}(p)$ is the overall plan cost of executing p, and $\mathcal{R} \to \mathbb{R}$ is a success bonus reflecting the reward from a successful execution. An optimal algorithm reports a task-motion plan of the highest utility:

$$p^* = \arg\max_p \mathcal{U}(p)$$

Next, we present LLM-GROP that computes goal configurations of objects, and task-motion plans for realizing the goal, through visually grounding spatial relationships while considering both efficiency and feasibility.

Method

In this article, we develop LLM-GROP, a task and motion planning (TAMP) approach that is semantically specified and visually grounded, as applied to object rearrangement tasks. At the high level, LLM-GROP generates object goal configurations (i.e., 2D positions) of relevant objects using common sense knowledge extracted from the LLM (LLM-Guided Goal Generation subsection). At the low level, LLM-GROP computes TAMP solutions for grounding the generated object locations into the physical world (Task and Motion Planning for Grounded Object Rearrangement subsection).

LLM-guided goal generation

This subsection presents our approach that leverages LLMs to compute both symbolic spatial relationships (e.g., a fork should be placed to the left of a plate and a knife to the right) and geometric spatial relationships (e.g., 2D coordinates of the objects on a table).

Generating symbolic spatial relationships. LLMs are first used to extract common sense knowledge regarding symbolic spatial relationships among objects placed on a table. This is accomplished through the utilization of a template-based prompt:

> Template 1: *The goal is to set a dining table with objects. The symbolic spatial relationship between objects includes* [*spatial relationships*]. [*examples*]. *What is a typical way of positioning* [*objects*] *on a table*? [*notes*].

where [*spatial relationships*] includes a few spatial relationships, such as *to the left of* and *on top of*. In presence of [*examples*], the prompting becomes few-shot; when no examples are provided, it is simplified to zero-shot prompting. In practice, few-shot prompts can ensure that the LLM's response follows a predefined format, though more prompt engineering efforts are needed. [*objects*] refers to the objects to be placed on the table, such as *a plate*, *a fork*, and *knife*. To control the LLM's output, [*notes*] can be added, such as the example "*Each action should be on a separate line starting with 'Place'. The answer cannot include other objects*."

LLMs are generally reliable in demonstrating common sense, but there may be times when they produce contradictory results. To prevent logical errors, a logical reasoning-based approach has been developed to evaluate the consistency of generated candidates with explicit symbolic constraints. This approach is implemented on answer set programming (ASP), which is a declarative programming language that expresses a problem as a set of logical rules and constraints (Gebser et al., 2008). In the event of a logical inconsistency, the same template is repeatedly fed to the LLM in an attempt to elicit a different, logically consistent output. ASP enables recursive reasoning, where rules and constraints can be defined in terms of other rules and constraints, providing a modular approach to problem-solving (Jiang et al., 2019b). ASP is particularly useful for determining whether sets of rules and constraints are true or false in a given context.

The approach involves defining spatial relationships, their transitions, and rules for detecting conflicts. These rules are created by human experts and serve to ensure that the generated context is logical and feasible. One such rule is: below (X,Y), right (X,Y), which states that object X cannot be both "below" and "to the right of" object Y at the same time. This rule ensures that the resulting arrangement of objects is physically possible. An instance of identifying a logical error is provided. For example, an LLM may generate instructions for arranging objects as follows:

1. Place fruit bowl in the center of table.
2. *Place butter knife above and to the right of fruit bowl.*
3. *Place dinner fork to the left of butter knife.*
4. Place dinner knife to the right of butter knife.
5. *Place fruit bowl to the right of dinner fork.*
6. Place water cup below and to the left of dinner knife.

Objects and spatial relationships are explicitly listed in the prompts used for querying LLMs, as shown in Template 1. We then use standard search methods to extract the objects and spatial relationships from the LLM outputs. In most cases, the LLMs are able to output instructions in the desired format; otherwise, we re-prompt the LLMs until we are able to extract the objects and spatial relationships.

There are logical inconsistencies in the italic lines: Steps 2 and 3 suggest placing the *fruit bowl* below the *dinner fork*, while Step 5 suggests placing the *fruit bowl* to the right of the *dinner fork*. This contradicts the established rule and results in no feasible solutions.

Generating geometric spatial relationships. After determining the symbolic spatial relationships between objects, we move on to generate their geometric configurations, where we use the following LLM template.

Template 2: [*object A*] *is placed* [*spatial relationship*] [*object B*]. *How many centimeters* [*spatial relationship*] [*object B*] *should* [*object A*] *be placed*?

For instance, when we use Template 2 to generate prompt "*A dinner plate is placed to the left of a knife. How many centimeters to the left of the water cup should the bread plate be placed*?", GPT-3 produces the output "*Generally, the dinner knife should be placed about 5-7 centimeters to the right of the dinner plate.*" In practice, the exact distance is extracted by searching for keyword "centimeter" to identify the number preceding it.

To determine the positions of objects, we first choose a coordinate origin. This origin could be an object that has a clear spatial relationship to the tabletop and is located centrally. A dinner plate is a good example of such an object. We then use the recommended distances and the spatial relationships between the objects to determine the coordinates of the other objects. Specifically, we can calculate the coordinates of an object by adding or subtracting the recommended distances in the horizontal and vertical directions, respectively, from the coordinates of the coordinate origin. The LLM-guided position for the ith object is denoted as (x^i, y^i), where $i \in N$.

However, relying solely on the response of the LLMs is not practical as they do not account for object attributes such as shape and size, including tables constraints. To address this limitation, we have designed an adaptive sampling-based method that incorporates object attributes after obtaining the recommended object positions. Specifically, our approach involves sequencing the sampling of each object's position using a 2D Gaussian sampling technique (Boor et al., 1999), with (x^i, y^i) as the mean vector, and the covariance matrix describing the probability density function's shape.

The resulting distribution is an ellipse centered at (x^i, y^i) with the major and minor axes determined by the covariance matrix. However, we do not blindly accept all of the sampling results; instead, we apply multiple rules to determine their acceptability, inspired by rejection sampling (Gilks and Wild, 1992). These rules include verifying that the sampled geometric positions adhere to symbolic relationships at a high level, avoiding object overlap, and ensuring that objects remain within the table boundary. For example, if the bounding box of an object position falls outside the detected table bounds, we reject that sample. The bounding box of objects and the table are computed based on their respective properties, such as size or shape. After multiple rounds of sampling, we can obtain M object configuration sequences.

The output of our LLM-guided goal generation approach is a 2D tabletop configuration of relevant objects. Next, we describe our visually grounded TAMP approach for realizing the computed goal configurations.

Task and motion planning for grounded object rearrangement

After identifying feasible object configurations on the tabletop, the next step is to place the objects on the tabletop based on one of object configuration sequences. At the task level, the robot must decide the sequence of object placement and how to approach the table. For example, if a bread is on top of a plate, the robot must first place the plate and then the bread. The robot must also determine how to approach the table, such as from which side of the table. Once the task plan is determined, the robot must compute 2D navigation goals (denoted as *loc*) at the motion level that connect the task and motion levels. Subsequently, the robot plans motion trajectories for navigation and manipulation behaviors.

In the presence of dynamic obstacles, not all navigation goals (*loc*) are equally preferred. For instance, it might be preferable for the robot to position itself close to an object for placement rather than standing at a distance and extending its reach. As a result, we developed a novel TAMP component (namely GROP) in our system for computing the optimal navigation goal *loc*, which enabled the task-motion plan with the maximal utility for placing each object in terms of feasibility and efficiency given an object configuration (x_j^i, y_j^i), where $0 \leq j \leq M$.

The GROP algorithm. Algorithm 1 presents the GROP algorithm. Implementing GROP requires a task planner $Plnr^t$, a motion planner $Plnr^m$ for planning base motions, a success bonus $\mathcal{R} \rightarrow \mathbb{R}$, and a cost function Cst that evaluates the cost of any motion trajectory generated by $Plnr^m$. Inputs of GROP include a rule-based task description T, a robot initial 2D position x^{init}, and a provided dataset D. GROP outputs a task-motion plan p in the form of $\langle p^t, p^m \rangle$.

GROP starts with training an FCN-based feasibility evaluator Ψ using provided dataset D in Line 1. Then it initializes an empty set of task-motion plans $\mathbf{P}$ in Line 2. $Plnr^t$ takes T as input and outputs a set of task-level satisficing plans, denoted as $\mathbf{P}^t$ in Line 3. The outer for-loop (Lines 4–21) iterates over each task-level satisficing plan. In each iteration, GROP evaluates the utility value of one task plan $\mathcal{U}(p)$, which incorporates both plan feasibility $\mathcal{F}(p)$ and plan efficiency $\mathcal{C}(p)$. Aiming to evaluate $\mathcal{F}(p)$ and $\mathcal{C}(p)$, each iteration in the first inner for-loop (Lines 7–13) considers a pair of navigation and manipulation actions in the task plan, and evaluates its feasibility and cost. In the second inner for-loop of Lines 14-17, GROP calls $Plnr^m$ to compute one motion plan for each task-level action. While the generated motion plan p^m only includes base motion, the feasibility of arm motion (equation (4) and Line 10) is considered in our motion plan generation. Line 18 puts together task plan p^t and motion plan p^m to form a task-motion plan p. In the same line, p is added to task-motion plan set $\mathbf{P}$. Lines 22–23 are the final steps to select and return the optimal task-motion plan from $\mathbf{P}$ given utility function $\mathcal{U}(p)$.

For different groups of object configurations, we use GROP to compute the maximal utility value of task-motion plans and select the best one for execution. Figure 4 shows one task-motion plan generated using LLM-GROP for a four-object rearrangement task.

Algorithm 1 GROP

Require: Task planner $Plnr^t$, motion planner $Plnr^m$, success bonus $\mathcal{R}$, and cost function Cst
Input: Task description T, robot initial position x^{init}, dataset D
1: Train a motion-level feasibility evaluator Ψ using dataset D (detailed in Figure 3)
2: Initialize a set of task-motion plans $\mathbf{P} \leftarrow \emptyset$
3: Compute a set of task-level satisficing plans: $\mathbf{P}^t \leftarrow Plnr^t(T)$
4: **for** each plan $p^t \in \mathbf{P}^t$ **do**
5: Initialize a motion-level position sequence: $X^{seq} \leftarrow [x^{init}]$
6: Initialize $tmp^f \leftarrow 0$ and $tmp^c \leftarrow 0$
7: **for** each action pair $\langle a^n_{l,l'}, a^m_{o,l'} \rangle$ in p^t **do**
8: Capture IM of location l'
9: Predict heatmap $h = \Psi(IM)$, using Eqn. 3
10: $tmp^f \leftarrow tmp^f + Fea^t(a^n_{l,l'}, a^m_{o,l'})$, using Eqn. 4
11: $x' \leftarrow Smp(l', h)$, and append x' to X^{seq}
12: $tmp^c \leftarrow tmp^c + Cst\big(Plnr^m(a^n_{l,l'})\big) + Cst\big(Plnr^m(a^m_{o,l'})\big)$
13: **end for**
14: **for** each $(x_i, x_{i+1}) \in X^{seq}$ **do**
15: Compute motion-level trajectory $\xi \leftarrow Plnr^m(x_i, x_{i+1})$
16: Append ξ to motion plan p^m
17: **end for**
18: Generate task-motion plan $p \leftarrow \langle p^t, p^m \rangle$, and append p to the task-motion plan set $\mathbf{P}$
19: Update $\mathcal{F}(p) \leftarrow \frac{tmp^f}{|p^t|}$ and $\mathcal{C}(p) \leftarrow tmp^c$
20: $\mathcal{U}(p) \leftarrow \mathcal{R} \cdot \mathcal{F}(p) - \mathcal{C}(p)$ (Eqn. 1)
21: **end for**
22: Compute optimal task-motion plan: $p^* = \arg\max_{p \in \mathbf{P}} \mathcal{U}(p)$
23: **return** p^*

Motion-level feasibility evaluation in GROP. In our mobile manipulation domain, motion-level feasibility $Fea^m(x, y)$ is a function of 2D positions x and y, and is the probability of a robot successfully navigating to x and manipulating an object that is in position y. $Fea^m(x, y)$ can be extracted

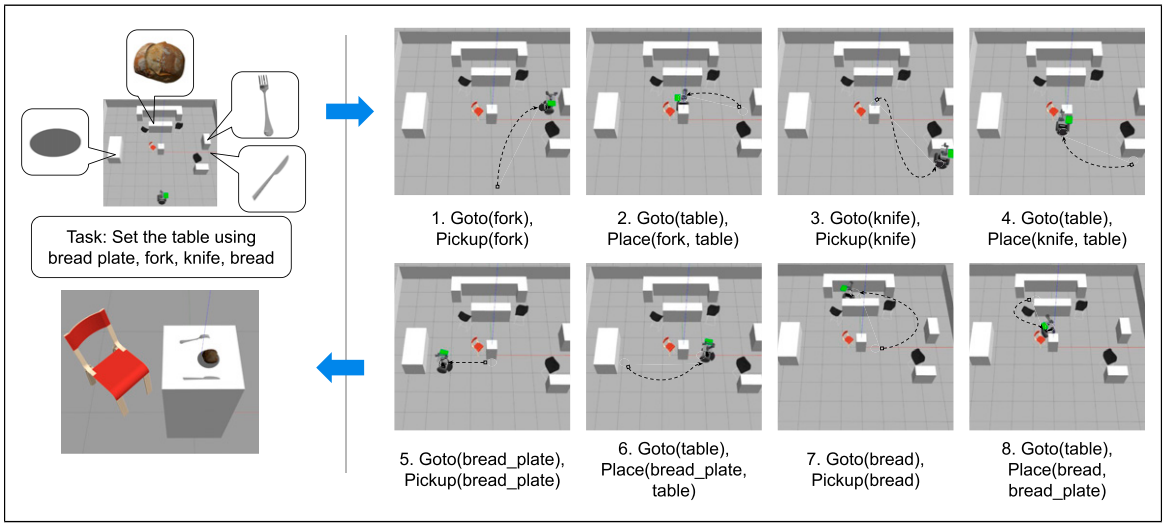

Figure 4. An illustrative example of LLM-GROP showing the robot navigation trajectories (dashed lines) as applied to the task of "set the table with a bread plate, a fork, a knife, and a bread." LLM-GROP is able to adapt to complex environments, using common sense extracted from an LLM to generate efficient (i.e., minimize the overall navigation cost) and feasible (i.e., select an available side of the table to unload) pick-and-place motion plans for the robot.

from gray-scale heatmap image h^y that is centered around y:

$$Fea^m(x,y) = h^y[x] \tag{2}$$

We use a FCN-based feasibility evaluator Ψ to generate heatmap h^y, given a top-down view image IM^y captured right above unloading position y ("Image Input" in Figure 2):

$$h^y = \Psi(IM^y) \tag{3}$$

Data collection and learning Ψ with FCN. Here, we discuss how to learn Ψ in equation (3). A *task* specifies an obstacle configuration and a position y that a robot wants to unload objects to. In each *trial* of our data collection process, a robot attempts to navigate to position x, and then unload an object to position y. Such a trial produces a data point in the following format:

$$(IM^y, x) : r$$

where IM^y is a top-down view image captured right above y, and r is either *true* or *false* depending on if the robot succeeds in both navigation and manipulation actions. The robot repeated the same process for N times ($N = 5$ in our case), and we used the results (r_0, r_1, …, r_{N-1}) to compute a success rate for positions x and y, which determines a gray-scale color for one pixel of a heatmap: $h\ [x]$.

Iterating over all possible positions of x in an area of $Width \times Height$ (24 pixels by 8 pixels in our case) in image IM, we were able to generate one full heatmap h for the current task. Here we assume this area is large enough to cover all positions, from which the robot can unload objects to y. To diversify the instances, we randomly placed obstacles (chairs in our case) to generate 10 different "environments," and then randomly sampled unloading positions to generate a total of 100 tasks. As a result, our dataset contains 100 instances, each in the form of a top-down view image (64 × 32). Each instance has a label that is in the form of a heatmap. The size of our dataset is 96,000, that is, $100 \times N \times Width \times Height$. The data collection and learning process is illustrated in Figure 3.

Our feasibility evaluator, Ψ, is trained exclusively on simulated data. We train the FCN as a N-class pixel-level classifier using softmax and a standard cross-entropy loss on the discrete success-level labels. At inference time, rather than selecting the top class, we convert the logits into class probabilities and take a weighted average over the N levels, and this produces smooth, continuous heatmaps as shown in Figure 3.

To deploy in the real world, we first estimate the poses of obstacles (i.e., tables and chairs) and reconstruct a digital twin of the workspace but using the same simulated assets from our training phase. We then render top-down views as the RGB observations and feed them directly into Ψ, sidestepping the sim-to-real gap. We train the FCN as a N-class pixel-level classifier using softmax and a standard cross-entropy loss on the discrete success-level labels. At inference time, rather than selecting the top class, we convert the logits into class probabilities and take a weighted average over the N levels, and this produces smooth, continuous heatmaps as shown in Figure 3.

For real-world experiments (e.g., the top-down view image in Figure 2), the visual input can capture a large area with multiple tables. We only consider feasibility evaluation on the table for placing the objects. There are other tables from which the robot needs to retrieve objects, where we assume there is sufficient free space for picking up the objects and we do not analyze motion feasibility for those tables.

The current training data was collected when the robot moves objects between rectangle-shaped tables, and the objects are distant from each other on the tables. If the table is of irregular shape or the tabletop objects are close to each other, the rearrangement tasks become more difficult and it is likely LLM-GROP will not work well. Otherwise, we expect that LLM-GROP's performance will be robust to object positions on the table and shapes of obstacles. Manipulation behaviors other than grasping and ungrasping, for example, pouring water and folding cloth, can be equipped onto the robot as long as additional training data is provided to guide the robot in selecting standing positions.

Table 1. Objects that are involved in our object rearrangement tasks for evaluation, where tasks 1–5 include three objects, tasks 6 and 7 include four objects, and tasks 8 and 9 include five objects.

Task #ID	Objects
1	Dinner Plate, Dinner Fork, Dinner Knife
2	Bread Plate, Water Cup, Bread
3	Mug, Bread Plate, Mug Mat
4	Fruit Bowl, Mug, Strawberry
5	Mug, Dinner plate, Mug Lid
6	Dinner Plate, Dinner Fork, Mug, Mug Lid
7	Dinner Plate, Dinner Fork, Dinner Knife, Strawberry
8 (Sim. Only)	Dinner Plate, Dinner Fork, Dinner Knife, Mug, Mug Lid
9 (Real Only)	Dinner Plate, Dinner Fork, Dinner Knife, Water Cup, Strawberry

Task-level feasibility evaluation in GROP. $Fea^t(a^n_{l,l'}, a^m_{o,l'})$ evaluates the feasibility (in the form of a probability) of a robot successfully performing both task-level navigation action $a^n_{l,l'}$ and task-level manipulation action $a^m_{o,l'}$.

$$Fea^t(a^n_{l,l'}, a^m_{o,l'}) = \frac{\sum_{i=0\cdots N-1} Fea^m(Smp_i(l',h),y)}{N} \tag{4}$$

Table 2. Rating guidelines for human raters in the experiments. 1 point indicates the poorest tableware object arrangement as it suggests that some objects are missing. Conversely, 5 points represent the best arrangement.

Points	Rating guidelines
1 (Poor)	Missing critical items compared with the objects listed at the top of the interface (e.g., dinner plate, dinner fork, dinner knife), making it hardly possible to complete a meal
2	All items are present, but the arrangement is poor and major adjustments are needed to improve the quality to a satisfactory level
3	All items are present and arranged fairly well, but still there is significant room to improve its quality
4	All items are present and arranged neatly, though an experienced human waiter might want to make minor adjustments to improve
5 (Strong)	All items are present and arranged very neatly, meeting the aesthetic standards of an experienced human waiter

where function $Smp_i(l', h)$ samples the ith 2D position from location l'. The positions are weighted by heatmap h that is centered around object o. Intuitively, positions of higher motion-level feasibility are more likely to be sampled.

This section presents the two key components of LLM-GROP for (1) computing semantically specified goal configurations of objects using common sense extracted from LLMs and (2) visually grounded planning at both task and motion levels for realizing the goals.

Experiments

To evaluate the effectiveness of LLM-GROP, we conducted a series of experiments focused on tableware object rearrangement tasks. In these tasks, a mobile manipulator is assigned the job of setting a dinner table using a specific set of objects. The experiments include nine distinct tasks involving various objects, as detailed in Table 1. The robot must retrieve multiple objects from different locations and place them on a central table, with obstacles (e.g., a chair) randomly positioned around the table. These tasks require the robot to compute semantically valid tabletop arrangements, plan efficient object rearrangement strategies, and execute the plans through navigation and manipulation behaviors.

Real robot experiment

We begin with a real-world demonstration to validate the proposed approach. The mobile manipulator used in this demonstration features a wheeled base for navigation and a 6-DOF robotic arm for performing manipulation tasks.

Experimental setup. We designed our real-world experiment to demonstrate that LLM-GROP can effectively handle a variety of scenarios. Tasks 4, 7, and 9 from Table 1 were selected for these demonstrations. The environment consisted of three tables, with objects initially placed on the left and right tables, and the robot positioned randomly. The robot repeated each task 15 times: during the first five repetitions, no obstacles were present; in the next five, a chair was positioned on the upper side of the table; and in the final five, a chair was positioned on the lower side of the table.

In the tableware object rearrangement tasks, task success was determined by whether the final positions of the objects appeared natural. To evaluate this, we captured an overhead image of the table at the end of each trial and assessed whether the object arrangement looked logically appropriate or exhibited any unnatural placements. To assess the effectiveness of LLM-GROP, we recruited 10 graduate students to evaluate the robot's performance by reviewing the captured images. We implemented a five-point rating system, detailed in Table 2, and asked the volunteers to score the tableware arrangements in the images provided.

We use the MoveIt software (Coleman et al., 2014) for planning grasping and ungrasping behaviors on the real robot, and there was no machine learning involved. To further facilitate robot grasping, we used QR codes to help the robot locate the objects of interest. For those objects that are hard to pick up, such as forks and knives, we use a utensil holder to improve the success rate of object grasping.

For our real-robot experiments, we utilized OpenAI's GPT-3 engines as the LLM component. The specific

Table 3. Hypermeters of OpenAI's GPT-3 engines in our experiment.

Parameter	Value
Model	text-davinci-003
Top p	1.0
Frequency penalty	0.0
Temperature	0.1
Maximum length	512
Presence penalty	0.0

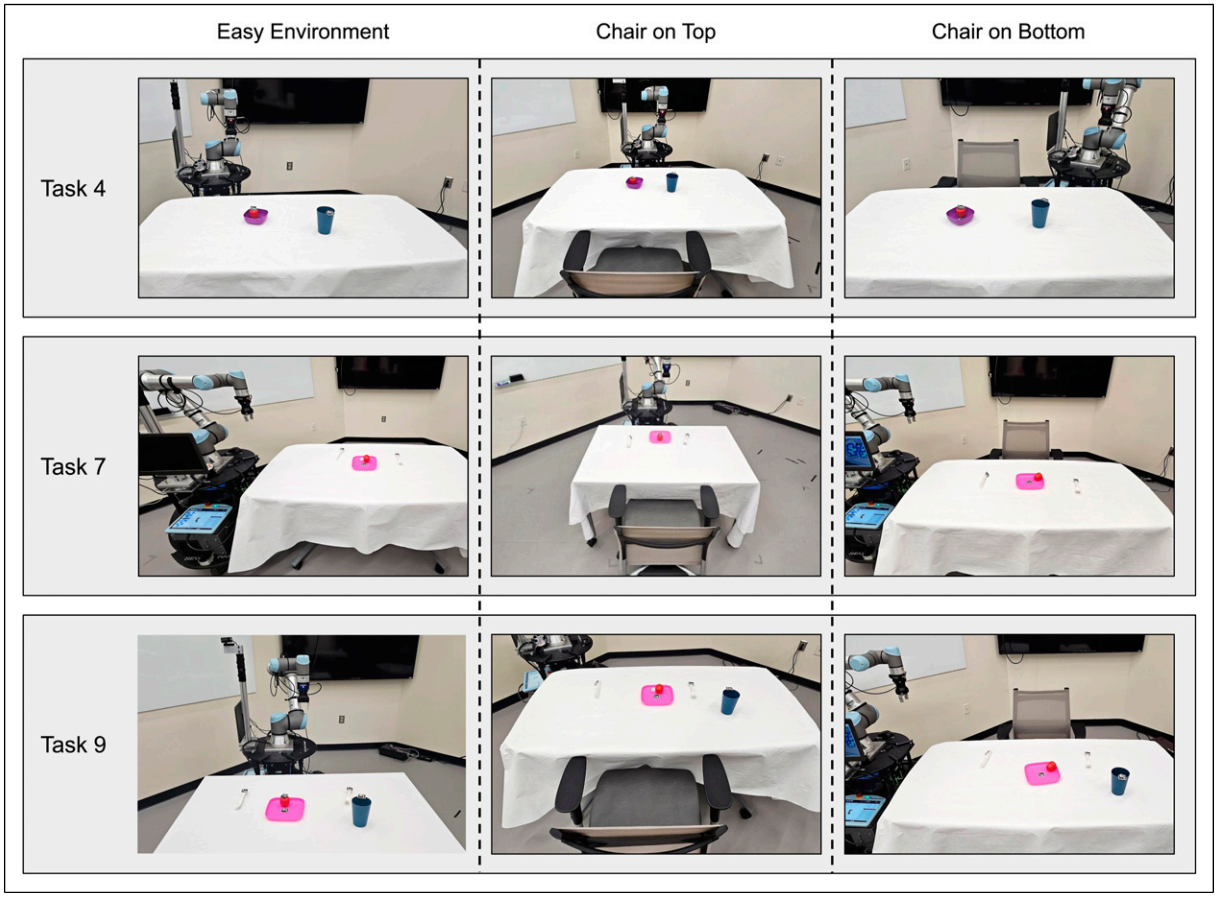

Figure 5. Example outcomes of the robot completing object rearrangement tasks. The "easy" environment did not include any obstacles, while the other environments included a chair on one side of the table. Note that the "top" and "bottom" labels shown in the columns were with respect to the robot's view. There were three tasks (IDs 4, 7, and 9—see Table 1) used for the real-robot experiments covering different numbers of objects being rearranged. The robot dynamically computed the goal configurations of those objects and (task and motion) plans for realizing those configurations.

hyperparameters used in the study are provided in Table 3. While there exist newer LLMs, we decided to use GPT-3 as the default LLM to be consistent to the results reported in the two preceding conference papers (Ding et al., 2023b; Zhang et al., 2022b). When we made comparisons between LLM-GROP and baselines, it was strictly enforced that the same LLM was used in all methods to avoid the potential bias introduced by the LLMs. In addition, we report results comparing multiple versions of LLM-GROP implemented using different LLMs in Section 5.3, to demonstrate that LLM-GROP is capable of adapting to newer LLMs.

Demonstrations. Figure 5 presents samples of the images collected during the real-robot demonstrations. The GROP algorithm enables the robot to adapt its position based on variations in the chair's placement, ensuring correct object positioning regardless of the chair's location. Simultaneously, the LLM component generates reasonable configurations for each task, maintaining accurate object placement even under changing conditions. The results of the user ratings are summarized in Table 4. Performance decreases as the number of objects increases, and in more complex environments with obstacles, performance further declines due to the additional challenges of navigating around obstacles to reach the goal positions. These

Table 4. Average of user ratings score for three individual object rearrangement tasks within 3 environments, an easy one without any obstacles, and a hard one with the chair being placed to the top of the table and to the bottom of the table.

Task	Easy environment	Hard environment (chair to the top)	Hard environment (chair to the bottom)
Task 4	**3.80**	3.60	3.73
Task 7	3.53	3.73	**3.93**
Task 9	**3.73**	3.13	3.06

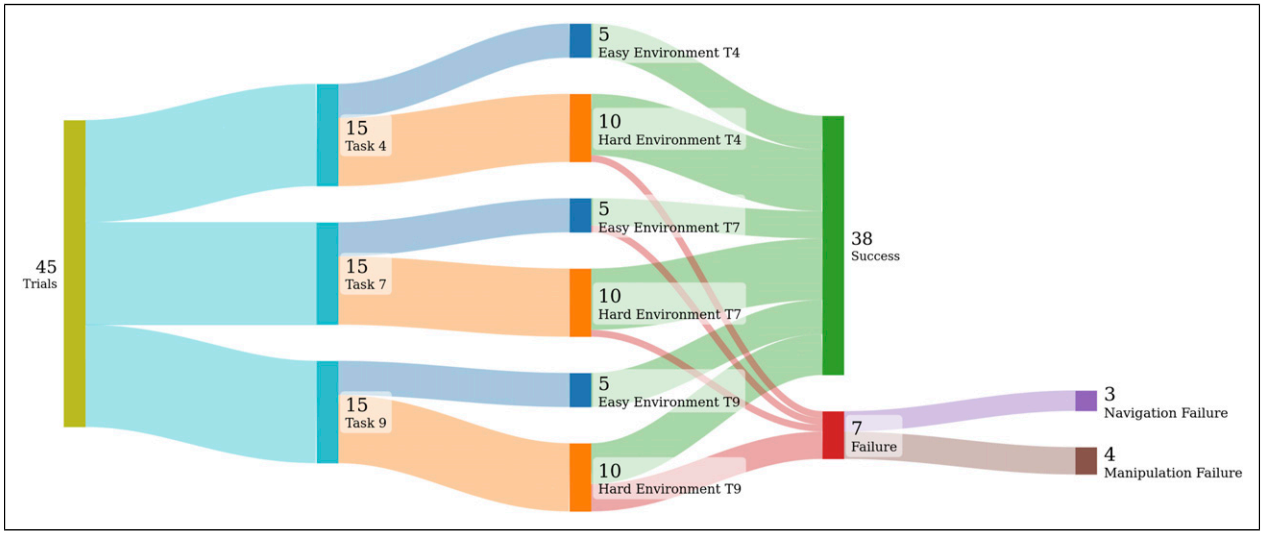

Figure 6. An overview of real robot experiment trials. There are 45 trials categorized in three tasks, and executed in two environments, an easy environment where there are no obstacles, and a hard one with a chair (obstacle) placed on top or bottom of the table. Successes total to 38 trials, while failure accounts to 7 trials, producing an overall 84.4% success rate. The failure trials are further categorized based on the reason of failure, navigation failure (3 trials) and manipulation failure (4 trials). This graph highlights how task and environment complexity impact the success of each trial.

results demonstrate the robustness of our approach in enabling a robotic platform to effectively perform real-world tasks. To provide a detailed view of the outcomes, Figure 6 categorizes the 45 robot execution trials by task, environment complexity, success rate, and failure modes.

Performance assessment

To assess the overall performance of the method and gain insights into the contributions of LLM-GROP, we conducted performance evaluations in simulated scenarios.

Experimental setup. As with the real-robot experiment's configuration, we conducted eight different tasks in a simulated environment[1], as detailed in Table 1. We execute each task 20 times using the LLM-GROP system with the same prompt templates described in LLM-Guided Goal Generation subsection, and after each task is completed, we capture an image of the table, the chair, and the objects on the tabletop for later human evaluation. We used the same LLM as the real-robot demonstration to carry out the simulation experiment.

Baselines. LLM-GROP is evaluated by comparing its performance to three baselines, where the first baseline is the weakest.

- Task Planning with Random Arrangement (TPRA): This baseline uses a task planner to sequence navigation and manipulation behaviors, while it randomly selects standing positions next to the target table and randomly places objects in no-collision positions on the table.
- LLM-based arrangement and task planning (LATP): It can predict object arrangements using LLMs and perform task planning. It uniformly samples standing positions around the table for manipulating objects.
- GROP: It considers plan efficiency and feasibility for task-motion planning, and lacks the capability of computing semantically valid arrangements. Similar to TPRA, GROP also randomly places objects in no-collision positions on the table.

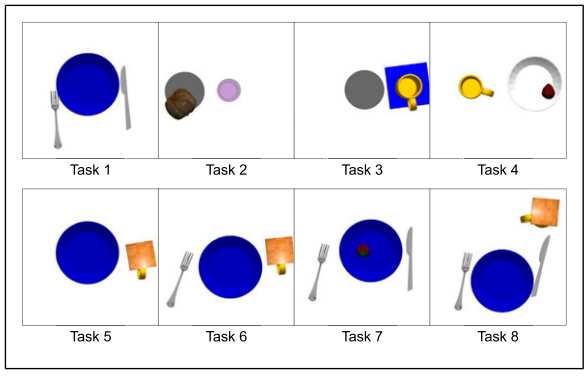

Figure 7. Examples of tableware objects rearranged by our LLM-GROP agent in eight tasks, where the objects used in these tasks can be found in Table 1. Our LLM-GROP enables the arrangement of tableware objects to be both semantically valid.

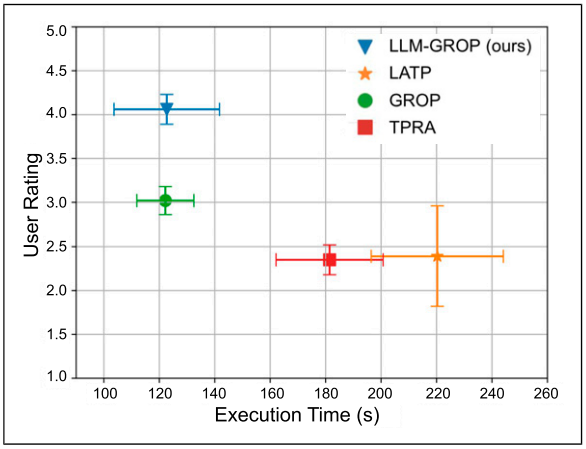

Figure 8. Overall performance of LLM-GROP as compared to three baselines based on mean values and standard errors of user ratings and robot execution time for all tableware object arrangement tasks.

Rating criteria. We recruited five graduate students with engineering backgrounds, three females and two males between the ages of 22 and 30. We generated 640 images from the four methods (three baselines and LLM-GROP) for eight tasks and each image required evaluation from all volunteers, resulting in a total sample size of 3200 images (the examples are shown in Figure 7). The volunteers were shown one image at a time on a website page that we provided, and they scored each image from 1 to 5 based on the rating rules. We ensured that the rating was rigorous by using a website to collect rating results, thereby minimizing any potential biases that could arise from further interaction with the volunteers once they entered the website.

LLM-GROP versus baselines. Figure 8 shows the key findings of our experiments, which compares the performance of LLM-GROP to the three other baseline approaches. The x-axis indicates the time each method takes to complete a single task, while the y-axis indicates the corresponding user rating. The results demonstrate that our LLM-GROP achieves the highest user rating and the shortest execution time compared to the other approaches. While GROP proves to be as efficient as our approach, it receives a significantly lower rating score. By contrast, TPRA and LATP both receive lower user ratings than our LLM-GROP. They also display poor efficiency. This is because they lack the navigation capabilities to efficiently navigate through complex environments. For instance, when their navigation goals are located within an obstacle area, they struggle to adjust their trajectory, leading to longer task completion times.

Figure 9 presents the individual comparison results of each method for individual tasks. The x-axis corresponds to Task #ID in Table 1, while the y-axis represents the average user rating for each method. Our LLM-GROP demonstrates superior performance over the baselines for each task. Specifically, tasks 1 to 5 receive slightly higher scores than tasks 6 and 8. This is reasonable because the latter two tasks require the robot to manipulate more objects, posing additional challenges for the robot.

Implementations with different LLMs

An important component of LLM-GROP is an off-the-shelf LLM. To evaluate the sensitivity of LLM-GROP to the choice of LLMs, we conducted experiments by realizing LLM-GROPs with four different LLMs.

Experimental setup. We evaluated four versions of our system using different LLMs: GPT-3, GPT-4, Gemini, and Claude. These models were selected for their diverse architectures and capabilities, enabling us to assess their respective influences. Another group of

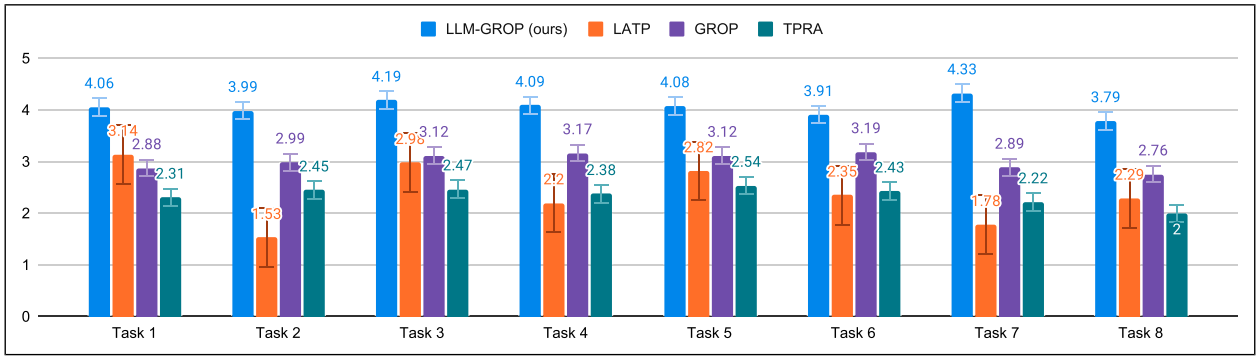

Figure 9. User ratings of individual object rearrangement tasks, with the x-axis representing the task and the y-axis representing the user rating score. It can be observed that LLM-GROP consistently performs the best compared to baselines. Tasks 1-5 involve three objects, tasks 6 and 7 involve four objects, and task 8 involves five objects. The numerical value displayed on each bar indicates the mean rating for the corresponding task.

Table 5. Average of user ratings score for individual object rearrangement tasks. Tasks 1–5 involve three objects, tasks 6 and 7 involve four objects, and task 8 involves five objects. Results of four different LLMs (including their model names) are reported in four different columns.

	GPT-3 (text-davinci-003)	GPT-4 (gpt-4-0613)	Gemini (gemini-1.5-flash-20240520)	Claude (claude-3.5-sonnet-20240620)
Task 1	4.06 ± 0.75	**4.98 ± 0.04**	4.87 ± 0.18	4.50 ± 0.00
Task 2	3.99 ± 0.30	4.48 ± 0.42	**4.70 ± 0.16**	3.65 ± 0.85
Task 3	4.19 ± 0.30	**4.60 ± 0.40**	2.95 ± 0.05	2.83 ± 0.00
Task 4	4.09 ± 0.22	3.15 ± 1.27	**4.83 ± 0.00**	4.65 ± 0.04
Task 5	4.08 ± 0.31	4.67 ± 0.00	2.27 ± 0.07	**4.73 ± 0.04**
Task 6	3.91 ± 0.60	**4.92 ± 0.00**	2.36 ± 0.07	4.90 ± 0.04
Task 7	4.33 ± 0.23	**4.98 ± 0.04**	4.65 ± 0.14	3.72 ± 1.34
Task 8	**3.79 ± 0.31**	2.22 ± 0.60	1.83 ± 0.34	1.00 ± 0.00

10 participants with engineering backgrounds, comprising four females and six males, was recruited to evaluate the method. We applied the same rating criteria used in the performance assessment.

Results. The results of this experiment, presented in Table 5, revealed that GPT-4 outperformed other models in four out of eight tasks, demonstrating robust performance across varying task complexities. Notably, GPT-4 achieved the highest average user rating scores for Tasks 1, 3, 6, and 7, which involved object counts ranging from three to four. This consistency indicates that GPT-4 is well suited for managing both simpler and moderately complex object rearrangement tasks with reliable performance.

However, the results also highlight the strengths of other models in specific tasks, underscoring their varied capabilities depending on task characteristics. For instance, Gemini achieved the highest scores in Tasks 2 and 4, suggesting a particular aptitude for tasks with unique relational or spatial requirements. Similarly, Claude outperformed the other models in Task 5, indicating potential advantages in handling intricate or nuanced task relationships within a three-object setup.

Interestingly, in Task 8—the most complex task involving five objects—GPT-3 received the highest score. This variation in performance demonstrates that no single model consistently excels across all task types. As such, these findings emphasize the importance of selecting the most appropriate model based on the specific relational and structural demands of each task.

Conclusion and discussion

To summarize, we propose LLM-GROP, which demonstrates how we can extract semantic information from LLMs and use it as a way to make common sense, semantically valid decisions about object placements as a part of a task and motion planner—letting us execute multi-step tasks in complex environments in response to natural-language commands. Further, LLM-GROP leverages computer vision methods to visually ground task and motion planning (TAMP) solutions for mobile manipulation (MoMa) tasks, and enable the robot to select standing positions to simultaneously facilitate both navigation and manipulation actions. LLM-GROP was evaluated through comparisons with existing TAMP methods, both in simulation and on a real robot, and results demonstrated its superiority in MoMa tasks.

End-to-end robot control

One important contribution of this research is the introduction of a novel computer vision approach to help a mobile manipulator select base positions. Such positions are not too close to the table, so the robot's base does not have physical contact with the table or other obstacles; at the same time, the standing positions should not be too far away from the table, so interaction behaviors with the target object(s) on the table are secured with high feasibility. Our discussion assumes fixed motion controllers for the mobile base and the manipulator, whereas in practice, one can customize those controllers for different tasks and different robot platforms. For one example, a biped humanoid robot might be able to squeeze through the crowded to reach a table, which is impossible for a chubby wheeled robot. For another, grasping a heavy beer pitcher would require a robot to stand close to the table compared with lighter objects such as forks and knives. Incorporating such control-level capabilities into the framework for LLM-GROP practitioners would be one step closer to producing globally optimal TAMP solutions.

Open worlds

The real world is generally open and there can be innumerable situations that are unforeseen in the

development of robot hardware and software. In MoMa tasks, a robot needs to move around obstacles and rearrange objects into a goal configuration. The target objects and obstacles can be novel, and the service requests for specifying goal configurations can be novel too. LLM-GROP assumes that complete 3D models exist for every object in the scene. In practice, we pick real-world objects whose geometry and appearance closely match our simulation assets. Although one could employ Real2Sim methods, such as 3D reconstruction or image-to-3D generative models, to build accurate object models, we leave those extensions for future work.

TAMP methods frequently assume known objects and known robot capabilities. As a result, finding TAMP solutions at both task and motion levels in open-world scenarios is still an open question. Foundation models are equipped with rich common sense information and can be potentially useful for addressing those open-world questions. While this work demonstrated that LLMs are useful for computing semantically specified object configurations, future work can further look into the capability of foundation models (LLMs and multimodal models) to better embrace the openness of the real world.

Natural language input

Recent advancements in foundation models including LLMs have made it possible for service robots to accurately interpret open-vocabulary, potentially ambiguous natural language inputs. In doing so, the robots need to process language and non-language inputs at the same time, and reliably build associations between them, for example, associating the tokens of "banana" to one physical object on a specific dining table. Natural language often involves under-specified and incomplete task specifications, as it is unrealistic to expect people to specify every detail of a goal configuration. The robot has at least the two options of asking clarification questions to seek additional details of the service request and using common sense to provide the service that makes the best sense. Using common sense to make decisions, for example, people like coffee in the morning, can lead to efficient user experience, while the downside is the hallucinations of user preferences or even risks to people and environments. The robot needs to make such decisions based on its AI alignment, utility functions and value systems.

We used commercially available LLMs in this research including GPTs, Claude, and Gemini, so the expenses go beyond time and energy, while further extending to monetary costs of querying the LLMs. Such costs are not discussed in the experiments. One future work direction can be in the minimization of costs in the usage of LLMs.

Ego-centric vision

To compute the standing positions for optimally supporting the navigation and manipulation actions, the robot needs an estimation of the world configurations. In this paper, we use top-down view images for perceiving the object locations and the world configurations. There is rich literature in pose estimation of objects in 3D worlds. One future work direction is to incorporate ego-centric vision for estimating world configurations, and active perception would even further enhance the robot's state estimation capability. Recent vision-language models (such as LLaVA and GPT-4o) can potentially be used as navigation goal selectors, and their performance can potentially be improved by multimodal prompting techniques like set-of-mark (SoM) (Yang et al., 2023).

Simultaneous manipulation and navigation. In this research, manipulation and navigation behaviors are temporally interleaved, and they are connected through the selection of standing positions. In general case, mobile manipulators can choose to do both at the same time leveraging whole-body control methods. Simultaneous manipulation and navigation is an open problem to robotics researchers, where the main challenge is the curse of dimensionality in control space. While interleaving manipulation and navigation is sufficient for MoMa robots to complete most object rearrangement tasks, there are scenarios where doing both is a must. For instance, when a robot and a human move an object together, the robot needs to adjust its grasping strategy while co-navigating with the human. Another example is to push a plate from one end of a long banquet table to the other. Those behaviors are still uncommon for current service robots in household environments, but generally can be useful for MoMa tasks.

Movable objects

Many objects in the real worlds are movable and their movability depends on the robot's MoMa skills. Among those movable objects, humans are special because their locations are changed through interaction actions such as language and gestures, instead of contact-rich behaviors such as grasping and pushing. This research assumes objects are stationary except for those target objects that are involved in the rearrangement tasks. For generalization, future work can incorporate interaction behaviors, such as saying "excuse me," for encouraging people movements. In addition, there is existing research on TAMP methods for domains with movable objects, where a robot can do second-order reasoning to move obstacle objects to facilitate the manipulation of target objects.

Acknowledgment

A portion of this work has taken place at the Autonomous Intelligent Robotics (AIR) Group, SUNY Binghamton. A portion of this work has taken place in the Learning Agents Research Group (LARG) at UT Austin. Peter Stone serves as the Chief Scientist of Sony AI and receives financial compensation for that role. The terms of this arrangement have been reviewed and approved by the University of Texas at Austin in accordance with its policy on objectivity in research.

ORCID iD

Shiqi Zhang https://orcid.org/0000-0003-4110-8213

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: AIR research is supported in part by the NSF (NRI-1925044, IIS-2428998), DEEP Robotics, Ford Motor Company, OPPO, Guiding Eyes, and SUNY RF, LARG research is supported in part by NSF (FAIN-2019844, NRT-2125858), ONR (N00014-24-1-2550), ARO (W911NF-17-2-0181, W911NF-23-2-0004, W911NF-25-1-0065), DARPA (Cooperative Agreement HR00112520004 on Ad Hoc Teamwork) Lockheed Martin, and UT Austin's Good Systems grand challenge.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Note

1. Implemented in the Gazebo simulator.

References

Ahn M, Brohan A, Brown N, et al. (2022) Do as i can and not as i say: grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*.

Akbari A, Diab M and Rosell J (2020) Contingent task and motion planning under uncertainty for human–robot interactions. *Applied Sciences* 10(5): 1665.

Blukis V, Paxton C, Fox D, et al. (2022) A persistent spatial semantic representation for high-level natural language instruction execution. In: *Conference on Robot Learning*. PMLR, 706–717.

Boor V, Overmars MH and Van Der Stappen AF (1999) The Gaussian sampling strategy for probabilistic roadmap planners. *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)* 2: 1018–1023.

Brown T, Mann B, Ryder N, et al. (2020) Language models are few-shot learners. *Advances in Neural Information Processing Systems* 33: 1877–1901.

Chen M, Tworek J, Jun H, et al. (2021) Evaluating large language models trained on code. *arXiv preprint arXiv: 2107.03374*.

Cheong SH, Cho BY, Lee J, et al. (2020) Where to relocate? object rearrangement inside cluttered and confined environments for robotic manipulation. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 7791–7797.

Chitnis R, Hadfield-Menell D, Gupta A, et al. (2016) Guided search for task and motion plans using learned heuristics. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 447–454.

Chitnis R, Kaelbling LP and Lozano-Pérez T (2019) Learning quickly to plan quickly using modular meta-learning. In: *International Conference on Robotics and Automation (ICRA)*. IEEE.

Coleman D, Sucan I, Chitta S, et al. (2014) Reducing the barrier to entry of complex robotic software: a moveit! case study. *arXiv preprint arXiv:1404.3785*.

Dantam NT, Kingston ZK, Chaudhuri S, et al. (2018) An incremental constraint-based framework for task and motion planning. *The International Journal of Robotics Research* 37(10): 1134–1151.

Devlin J, Chang MW, Lee K, et al. (2018) Bert: pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Ding Y, Zhang X, Zhan X, et al. (2020) *Task-Motion Planning for Safe and Efficient Urban Driving*. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Ding Y, Zhang X, Zhan X, et al. (2022) Learning to ground objects for robot task and motion planning. In: *IEEE Robotics and Automation Letters*. IEEE.

Ding Y, Zhang X, Amiri S, et al. (2023a) Integrating action knowledge and llms for task planning and situation handling in open worlds. *arXiv preprint arXiv:2305.17590*.

Ding Y, Zhang X, Paxton C, et al. (2023b) Task and motion planning with large language models for object rearrangement. In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2086–2092.

Driess D, Ha JS and Toussaint M (2020a) Deep visual reasoning: learning to predict action sequences for task and motion planning from an initial scene image. In: *Proceedings of Robotics: Science and Systems*. MIT Press.

Driess D, Oguz O, Ha JS, et al. (2020b) Deep visual heuristics: learning feasibility of mixed-integer programs for manipulation planning. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 9563–9569.

Erdem E, Haspalamutgil K, Palaz C, et al. (2011) Combining high-level causal reasoning with low-level geometric reasoning and motion planning for robotic manipulation. In: *IEEE International Conference on Robotics and Automation*. IEEE.

Garrett CR, Lozano-Perez T and Kaelbling LP (2018) Ffrob: leveraging symbolic planning for efficient task and motion planning. *The International Journal of Robotics Research* 37(1): 104–136.

Garrett CR, Paxton C, Lozano-Pérez T, et al. (2020) Online replanning in belief space for partially observable task and

motion problems. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 5678–5684.

Garrett CR, Chitnis R, Holladay R, et al. (2021) Integrated task and motion planning. *Annual review of control, robotics, and autonomous systems* 4: 265–293.

Gebser M, Kaminski R, Kaufmann B, et al. (2008) A user's guide to gringo, clasp, clingo, and iclingo.

Gilks WR and Wild P (1992) Adaptive rejection sampling for gibbs sampling. *Applied Statistics* 41(2): 337–348.

Goodwin W, Vaze S, Havoutis I, et al. (2022) Semantically grounded object matching for robust robotic scene rearrangement. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 11138–11144.

Gravot F, Cambon S and Alami R (2005) Asymov: a planner that deals with intricate symbolic and geometric problems. In: *The Eleventh International Symposium of Robotics Research*. Springer.

Gu J, Chaplot DS, Su H, et al. (2022) Multi-skill mobile manipulation for object rearrangement. *arXiv preprint arXiv: 2209.02778*.

Hadfield-Menell D, Groshev E, Chitnis R, et al. (2015) *Modular Task and Motion Planning in Belief Space*. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Huang E, Jia Z and Mason MT (2019) Large-scale multi-object rearrangement. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 211–218.

Huang W, Abbeel P, Pathak D, et al. (2022a) Language models as zero-shot planners: extracting actionable knowledge for embodied agents. In: Thirty-Ninth International Conference on Machine Learning, Maryland, USA, 17-23 July 2022.

Huang W, Xia F, Xiao T, et al. (2022b) Inner monologue: embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*.

Huang W, Wang C, Zhang R, et al. (2023) Voxposer: composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*.

Huang W, Wang C, Li Y, et al. (2024) Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*.

Inoue Y and Ohashi H (2022) Prompter: utilizing large language model prompting for a data efficient embodied instruction following. *arXiv preprint arXiv:2211.03267*.

Jiang Y, Yang F, Zhang S, et al. (2019a) *Task-Motion Planning with Reinforcement Learning for Adaptable Mobile Service Robots*. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Jiang Y, Zhang S, Khandelwal P, et al. (2019b) Task planning in robotics: an empirical comparison of pddl-and asp-based systems. *Frontiers of Information Technology & Electronic Engineering* 20(3): 363–373.

Kaelbling LP and Lozano-Pérez T (2013) Integrated task and motion planning in belief space. *The International Journal of Robotics Research* 32(9-10): 1194–1227.

Kant Y, Ramachandran A, Yenamandra S, et al. (2022) Housekeep: tidying virtual households using commonsense reasoning. *arXiv preprint arXiv:2205.10712*.

Kapelyukh I, Vosylius V and Johns E (2022) Dall-e-bot: introducing web-scale diffusion models to robotics. *arXiv preprint arXiv:2210.02438*.

Kaplan J, McCandlish S, Henighan T, et al. (2020) Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Kim B and Shimanuki L (2020) Learning value functions with relational state representations for guiding task-and-motion planning. In: *Conference on Robot Learning*. PMLR, 955–968.

Kim B, Wang Z, Kaelbling LP, et al. (2019) Learning to guide task and motion planning using score-space representation. *The International Journal of Robotics Research* 38(7): 793–812.

King JE, Cognetti M and Srinivasa SS (2016) *(????) rearrangement planning using object-centric and robot-centric action spaces*. ICRA, 3940–3947.

Lagriffoul F, Dimitrov D, Bidot J, et al. (2014) Efficiently combining task and motion planning using geometric constraints. *The International Journal of Robotics Research* 33(14): 1726–1747.

Lagriffoul F, Dantam NT, Garrett C, et al. (2018) Platform-independent benchmarks for task and motion planning. *IEEE Robotics and Automation Letters* 3(4): 3765–3772.

Liang J, Huang W, Xia F, et al. (2022) Code as policies: language model programs for embodied control. *arXiv preprint arXiv: 2209.07753*.

Liu P, Yuan W, Fu J, et al. (2021a) Pre-train, prompt, and predict: a systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Liu W, Paxton C, Hermans T, et al. (2021b) Structformer: learning spatial structure for language-guided semantic rearrangement of novel objects. *arXiv preprint arXiv:2110.10189*.

Liu W, Hermans T, Chernova S, et al. (2022a) Structdiffusion: Object-centric diffusion for semantic rearrangement of novel objects. *arXiv preprint arXiv:2211.04604*.

Liu W, Paxton C, Hermans T, et al. (2022b) Structformer: learning spatial structure for language-guided semantic rearrangement of novel objects. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 6322–6329.

Liu B, Jiang Y, Zhang X, et al. (2023) Llm+ p: empowering large language models with optimal planning proficiency. *arXiv preprint arXiv:2304.11477*.

Lo SY, Zhang S and Stone P (2020) The petlon algorithm to plan efficiently for task-level-optimal navigation. *Journal of Artificial Intelligence Research* 69: 471–500.

Min SY, Chaplot DS, Ravikumar P, et al. (2021) Film: following instructions in language with modular methods. *arXiv preprint arXiv:2110.07342*.

Nouman A, Patoglu V and Erdem E (2021) Hybrid conditional planning for robotic applications. *The International Journal of Robotics Research* 40(2-3): 594–623.

OpenAI (2023) *Chatgpt*. Accessed: 2023-02-08.https://openai.com/blog/chatgpt/.Cit.on

Phiquepal C and Toussaint M (2019) *Combined Task and Motion Planning Under Partial Observability: An Optimization-based Approach*. IEEE International Conference on Robotics and Automation (ICRA).

Plaku E, Kavraki LE and Vardi MY (2007) Discrete search leading continuous exploration for kinodynamic motion planning. In: Robotics: Science and Systems, Los Angeles, California, June 21 – June 25, 2025, 326–333.

Ramesh A, Dhariwal P, Nichol A, et al. (2022) Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*.

Rana K, Haviland J, Garg S, et al. (2023) Sayplan: grounding large language models using 3d scene graphs for scalable task planning. *arXiv preprint arXiv:2307.06135*.

Shridhar M, Thomason J, Gordon D, et al. (2020) Alfred: a benchmark for interpreting grounded instructions for everyday tasks. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, 10740–10749.

Singh I, Blukis V, Mousavian A, et al. (2022) Progprompt: generating situated robot task plans using large language models. *arXiv preprint arXiv:2209.11302*.

Srivastava S, Fang E, Riano L, et al. (2014) Combined task and motion planning through an extensible planner-independent interface layer. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 639–646.

Szot A, Yadav K, Clegg A, et al. (2022) Habitat rearrangement challenge 2022. https://aihabitat.org/challenge/rearrange_2022

Thomas A, Mastrogiovanni F and Baglietto M (2021) Mptp: Motion-planning-aware task planning for navigation in belief space. *Robotics and Autonomous Systems* 141: 103786.

Vasilopoulos V, Kantaros Y, Pappas GJ, et al. (2021) Reactive planning for mobile manipulation tasks in unexplored semantic environments. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 6385–6392.

Wang Z, Garrett CR, Kaelbling LP, et al. (2018) Active model learning and diverse action sampling for task and motion planning. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 4107–4114.

Wei QA, Ding S, Park JJ, et al. (2023) Lego-net: learning regular rearrangements of objects in rooms. *arXiv preprint arXiv: 2301.09629*.

Weihs L, Deitke M, Kembhavi A, et al. (2021) Visual room rearrangement. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Wells AM, Dantam NT, Shrivastava A, et al. (2019) Learning feasibility for task and motion planning in tabletop environments. In: *IEEE Robotics and Automation Letters*. IEEE.

Wu J, Antonova R, Kan A, et al. (2023) Tidybot: personalized robot assistance with large language models. *arXiv preprint arXiv:2305.05658*.

Yang J, Zhang H, Li F, et al. (2023) Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *arXiv preprint arXiv:2310.11441*.

Yang Z, Garrett C, Fox D, et al. (2024) Guiding long-horizon task and motion planning with vision language models. *arXiv preprint arXiv:2410.02193*.

Zhang Y and Chai J (2021) Hierarchical task learning from language instructions with unified transformers and self-monitoring. *arXiv preprint arXiv:2106.03427*.

Zhang S, Roller S, Goyal N, et al. (2022a) Opt: open pre-trained transformer language models. *arXiv preprint arXiv: 2205.01068*.

Zhang X, Zhu Y, Ding Y, et al. (2022b) Visually grounded task and motion planning for mobile manipulation. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 1925–1931.

Zhang X, Altaweel Z, Hayamizu Y, et al. (2024) Dkprompt: domain knowledge prompting vision-language models for open-world planning. *arXiv preprint arXiv:2406.17659*.

Zhao Z, Lee WS and Hsu D (2023) Large language models as commonsense knowledge for large-scale task planning. *arXiv preprint arXiv:2305.14078*.

Zhao Z, Cheng S, Ding Y, et al. (2024) *A Survey of Optimization-based Task and Motion Planning: From Classical to Learning Approaches*. IEEE/ASME Transactions on Mechatronics.

Zhu Y, Tremblay J, Birchfield S, et al. (2020) *Hierarchical Planning for long-horizon Manipulation with Geometric and Symbolic Scene Graphs. 2021*. IEEE International Conference on Robotics and Automation (ICRA).