VoxPoser: Composable 3D Value Maps for Robotic Manipulation with Language Models

Anonymous Author(s)
Affiliation
Address
email

Abstract: Large language models (LLMs) are shown to possess a wealth of actionable knowledge that can be extracted for robot manipulation in the form of reasoning and planning. Despite the progress, most still rely on pre-defined motion primitives to carry out the physical interactions with the environment, which remains a major bottleneck. In this work, we aim to synthesize robot trajectories, i.e., a dense sequence of 6-DoF end-effector waypoints, for a large variety of manipulation tasks given an *open-set of instructions* and an *open-set of objects*. We achieve this by first observing that LLMs excel at inferring affordances and constraints given a free-form language instruction. More importantly, by leveraging their code-writing capabilities, they can interact with a visual-language model (VLM) to compose 3D value maps to ground the knowledge into the observation space of the agent. The composed value maps are then used in a model-based planning framework to *zero-shot* synthesize closed-loop robot trajectories with robustness to dynamic perturbations. We further demonstrate how the proposed framework can benefit from online experiences by efficiently learning a dynamics model for scenes that involve contact-rich interactions. We present a large-scale study of the proposed method in both simulated and real-robot environments, showcasing the ability to perform over 30 everyday manipulation tasks specified in free-form natural language. Project website: voxposer-anon.github.io.

1 Introduction

Language is a compressed medium through which humans distill and communicate their knowledge and experience of the world. Large language models (LLMs) have emerged as a promising approach to capture this abstraction, learning to represent the world through projection into language space [1–4]. While these models are believed to internalize generalizable knowledge in the text form, it remains a question about how to use such generalizable knowledge to enable embodied agents to physically *act* in the real world.

We look at the problem of grounding abstract language instructions (e.g., "set up the table") in the robot actions [5]. Prior works have leveraged lexical analysis to parse the instructions [6–8], while more recently language models have been used to decompose the instructions into a textual sequence of steps [9–11]. However, to enable physical interactions with the environment, existing approaches typically rely on a repertoire of manually-designed or pre-trained motion primitives (i.e., skills) that may be invoked by an LLM or a planner, and this reliance on individual skill acquisition is often considered a major bottleneck of the system due to the lack of large-scale robotic data. The question then arises: how can we leverage the wealth of internalized knowledge of LLMs at the even fine-grained action level for robots, without requiring laborious data collection or manual designs for each individual primitive?

In addressing this challenge, we first note that it is infeasible for LLMs to directly output control actions in text, which are typically driven by high-frequency control signals in high-dimensional space. However, we find that LLMs excel at inferring language-conditioned *affordances* and *constraints*, and by leveraging their code-writing capabilities, they can compose dense 3D voxel maps that ground them in the visual space by orchestrating perception calls (e.g., via CLIP [12] or open-vocabulary detectors [13–15]) and array operations (e.g., via NumPy [16]). For example, given an instruction "open the top drawer and watch out for the vase", LLMs can

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

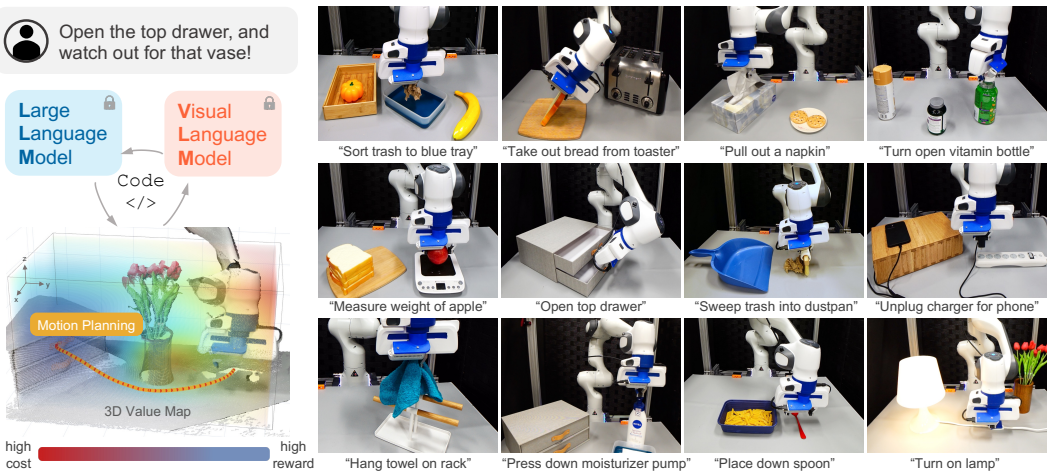

Figure 1: VoxPoser extracts language-conditioned **affordances** and **constraints** from LLMs and grounds them to 3D value maps using VLMs, without additional training to either component. The value maps can **zero-shot** synthesize trajectories for large varieties of everyday manipulation tasks with an **open-set of instructions** and an **open-set of objects**.

be prompted to infer: 1) the top drawer handle should be grasped, 2) the handle needs to be translated outwards, and 3) the robot should stay away from the vase. While these are expressed in the text form, LLMs can generate Python code to invoke perception APIs to obtain spatial-geometric information of relevant objects or parts (e.g., "handle") and then manipulate the 3D voxels to prescribe reward or cost at relevant locations in observation space (e.g., the target location of the handle is assigned a high value while the surrounding of the vase is assigned low values). Finally, the composed value maps can serve as objective functions for motion planners to directly synthesize robot trajectories that achieve the given instruction, without requiring additional training data for each task or for the LLM. An illustration diagram and a subset of tasks we considered are shown in Fig. 1.

We term this approach **VoxPoser**, a formulation that extracts affordances and constraints from LLMs to compose voxel value maps in 3D observation space for guiding robots to interact with the environment. In particular, the method leverages LLMs to *compose* the key aspects for generating robot trajectories rather than attempting to train policies on robotic data that are often of limited amount or variability, effectively achieving generalization for *open-set* instructions in a zero-shot manner. By integrating it into a model-based planning framework, we demonstrate closed-loop executions via model predictive control (MPC) that is robust to external disturbances. We further showcase how VoxPoser can also benefit from limited online interactions to efficiently learn a dynamics model that involves contact-rich interactions.

Our contributions are summarized as follows:

- We present VoxPoser, a method for extracting affordances and constraints for robotic manipulation from pre-trained language models, which requires no additional training and generalizes to open-set instructions.
- Using VoxPoser to represent task objectives, we demonstrate that synthesized trajectories can be robustly executed in closed-loop via MPC for over 30 manipulation tasks in both simulated and real-world environments.
- We show the applicability of VoxPoser to benefit from only a limited amount of online interactions by efficiently learning a dynamics model, e.g., learning to open a door with a lever handle in under 3 minutes.

2 Related Works

Grounding Language Instructions. Language grounding has been studied extensively both in terms of intelligent agents [17–20] and of robotics [21, 6, 22, 23, 5, 7, 24], where language can be used as a tool for compositional goal specification [5, 25–31], semantic anchor for training multi-modal representation [12, 32, 33], or as an intermediate substrate for planning and reasoning [34–36, 9, 10, 37, 38]. Prior works have looked at using classical tools such as lexical analysis, formal logic, and graphical models to interpret language instructions [25, 7, 6, 24]. More recently, end-to-end approaches, popularized by successful applications to offline domains [39–41, 1], have been applied to directly ground language instructions in robot interactions by learning from data with language annotations, spanning from model learning [42], imitation learning [43, 44, 28, 45–52], to rein-

forcement learning [53–55]. Most closely related to our work is Sharma et al. [48], where an end-to-end model is optimized via supervised learning to map language instructions to 2D costmaps, which are used to steer a motion planner to generate preferred trajectories in a collision-free manner. In contrast, we rely on pre-trained language models for their open-world knowledge and tackle the more challenging robotic manipulation in 3D.

Language Models for Robotics. Leveraging pre-trained language models for embodied applications is an active area of research, where a large body of works focus on planning with language models [9–11, 56, 29, 37, 57–70, 34]. To allow language models to perceive the physical environments, textual descriptions of the scene [37, 11, 57] or perception APIs [71] can be given, vision can be used during decoding [65] or can be directly taken as input by multi-modal language models [66, 2]. In addition to perception, to truly bridge the perception-action loop, an embodied language model must also know how to *act*, which typically is achieved by a library of pre-defined primitives. Liang et al. [71] showed that LLMs exhibit behavioral commonsense that can be useful for low-level control. Despite the promising signs, hand-designed motion primitives are still required, and while LLMs are shown to be capable of composing *sequential* policy logic, it remains unclear whether composition can happen at *spatial* level. A related line of works has also explored using LLMs for reward specification in the context of reward design [72] and exploration in reinforcement learning [73–76], and human preference learning [77]. In contrast, we focus exclusively on grounding the reward generated by LLMs in the *3D observation space* of the robot, which we identify as most useful for manipulation tasks.

Learning-based Trajectory Optimization. Many works have explored leveraging learning-based approaches for trajectory optimization. While the literature is vast, they can be broadly categorized into those that learn the models [78–86] and those that learn the cost/reward or constraints [87–90, 48, 91], where data are typically collected from in-domain interactions. To enable generalization in the wild, a parallel line of works has explored learning task specification from large-scale offline data [92, 93, 33, 32, 42, 94, 95, 52], particularly egocentric videos [96, 97], or leveraging pre-trained foundation models [98–100, 31, 101]. The learned cost functions are then used by reinforcement learning [98, 95], imitation learning [93], or trajectory optimization [92, 33] to generate robot actions. In this work, we leverage large language models for *in the wild* cost specification without requiring in-domain interaction data and with better generalization benefits. Compared to prior works that leverage foundation models, we ground the cost directly in *3D observation space* in *real-time*, which makes VoxPoser amenable to closed-loop MPC that's robust in execution.

3 Method

We first provide the formulation of VoxPoser as an optimization problem (Sec. 3.1). Then we describe how VoxPoser can be used as a general zero-shot framework to map language instructions to 3D value maps (Sec. 3.2). We subsequently demonstrate how trajectories can be synthesized in closed-loop for robotic manipulation (Sec. 3.3). While zero-shot in nature, we demonstrate how VoxPoser can learn from online interactions to efficiently solve contact-rich tasks (Sec. 3.4).

3.1 Problem Formulation

Consider a manipulation problem given as a *free-form* language instruction $\mathcal{L}$ (e.g., "open the top drawer"). However, generating robot trajectories according to $\mathcal{L}$ can be difficult because $\mathcal{L}$ may be arbitrarily long-horizon or under-specified (i.e., requires contextual understanding). Instead, we focus on individual phases (sub-tasks) of the problem ℓ_i that distinctively specify a manipulation task (e.g., "grasp the drawer handle", "pull open the drawer"), where the decomposition $\mathcal{T} \rightarrow (\ell_1, \ell_2, ..., \ell_n)$ is given by a high-level planner (e.g., an LLM or a search-based planner) [1]. The central problem investigated in this work is to generate a motion trajectory $\tau_i^{\mathbf{r}}$ for robot $\mathbf{r}$ and each manipulation phase described by instruction ℓ_i. We represent $\tau_i^{\mathbf{r}}$ as a sequence of dense end-effector waypoints to be executed by an Operational Space Controller [102], where each waypoint consists of a desired 6-DoF end-effector pose, end-effector velocity, and gripper action. However, it is worth noting that other representations of trajectories, such as joint space trajectories, can also be used. Given each sub-task ℓ_i, we formulate this as an optimization problem defined as follows:

[1] Note that the decomposition and sequencing of these sub-tasks are also done by LLMs in this work, though we do not investigate this aspect extensively as it is not the focus of our contributions.

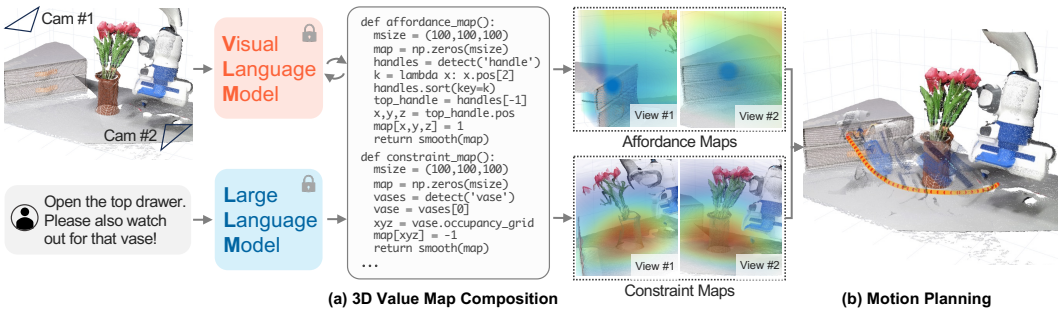

Figure 2: Overview of VoxPoser. Given RGB-D observation of the environment and language instruction, LLMs generate code, which interacts with VLMs, to produce a sequence of 3D affordance maps and constraint maps (collectively referred to as value maps) grounded in the observation space of the robot **(a)**. The composed value maps serve as objective functions for motion planners to synthesize zero-shot trajectories for robot manipulation **(b)**.

$$\min_{\tau_i^{\mathbf{r}}}\{\mathcal{F}_{task}(\mathbf{T}_i,\ell_i)+\mathcal{F}_{control}(\tau_i^{\mathbf{r}})\} \quad \text{subject to} \quad \mathcal{C}(\mathbf{T}_i) \tag{1}$$

where $\mathbf{T}_i$ is the evolution of environment state, $\tau_i^{\mathbf{r}} \subseteq \mathbf{T}_i$ is the robot trajectory, and $\mathcal{C}(\mathbf{T}_i)$ denotes the relevant dynamics and kinematics constraints. $\mathcal{F}_{task}$ scores the extent of $\mathbf{T}_i$ completes the instruction ℓ_i while $\mathcal{F}_{control}$ specifies the control costs, e.g., to encourage $\tau_i^{\mathbf{r}}$ to minimize total control effort or total time. By solving this optimization problem for each sub-task ℓ_i, we obtain a sequence of robot trajectories that collectively achieve the overall task specified by the language instruction $\mathcal{L}$.

3.2 Grounding Language Instruction via VoxPoser

Calculating $\mathcal{F}_{task}$ with respect to free-form language instructions is extremely challenging, not only because of the rich space of semantics language can convey but also because of the lack of robot data labeled with $\mathbf{T}$ and ℓ. However, we provide a critical observation that a large number of tasks can be characterized by a voxel value map $\mathbf{V} \in \mathbb{R}^{w \times h \times d}$ in the observation space of the robot, which guides the motion of an "entity of interest" in the scene, such as the robot end-effector, an object, or an object part. For example, consider the task "open the top drawer" and its first sub-task "grasp the top drawer handle" (inferred by LLMs) in Fig. 2. The "entity of interest" is the robot end-effector, and the voxel value map should reflect the attraction toward the drawer handle. By further commanding "watch out for the vase", the map can also be updated to reflect the repulsion from the vase. We denote the "entity of interest" as $\mathbf{e}$ and its trajectory as $\tau^{\mathbf{e}}$. Using this voxel value map for a given instruction ℓ_i, $\mathcal{F}_{task}$ can be approximated by accumulating the values of $\mathbf{e}$ traversing through $\mathbf{V}_i$, formally calculated as $\mathcal{F}_{task} = -\sum_{j=1}^{|\tau_i^{\mathbf{e}}|} \mathbf{V}(p_j^{\mathbf{e}})$, where $p_j^{\mathbf{e}} \in \mathbb{N}^3$ is the discretized (x,y,z) position of $\mathbf{e}$ at step j.

Notably, we observe large language models, by being pre-trained on Internet-scale data, exhibit capabilities not only to identify the "entity of interest" but also to compose value maps that accurately reflect the task instruction by writing Python programs. Specifically, when an instruction is given as a comment in the code, LLMs can be prompted to 1) call perception APIs (which invoke visual-language models (VLM) such as an open-vocabulary detector [13–15]) to obtain spatial-geometrical information of relevant objects, 2) generate NumPy operations to manipulate 3D arrays, and 3) prescribe precise values at relevant locations. We term this approach as **VoxPoser**. Concretely, we aim to obtain a voxel value map $\mathbf{V}_i^t = \text{VoxPoser}(\mathbf{o}^t, \ell_i)$ by prompting an LLM and executing the code via a Python interpreter, where $\mathbf{o}^t$ is the RGB-D observation at time t and ℓ_i is the current instruction. Additionally, because $\mathbf{V}$ is often sparse, we densify the voxel maps via smoothing operations, as they encourage smoother trajectories optimized by motion planners.

Additional Trajectory Parametrization. The above formulation of VoxPoser uses LLMs to compose $\mathbf{V}: \mathbb{N}^3 \to \mathbb{R}$ to map from discretized coordinates in voxel space to a real-valued "cost", which we can use to optimize a path consisting only of the positional terms. To extend to $\text{SE}(3)$ poses, we can also use LLMs to compose rotation maps $\mathbf{V}_r: \mathbb{N}^3 \to \text{SO}(3)$ at coordinates relevant to the task objectives (e.g., "end-effector should face the support normal of the handle"). Similarly, we further compose gripper maps $\mathbf{V}_g: \mathbb{N}^3 \to \{0,1\}$ to control gripper open/close and velocity maps $\mathbf{V}_v: \mathbb{N}^3 \to \mathbb{R}$ to specify target velocities. Note that while these additional trajectory parametrizations are not mapped to a real-valued "cost", they can also be factored in the optimization procedure (Equation 1) to parametrize the trajectories.

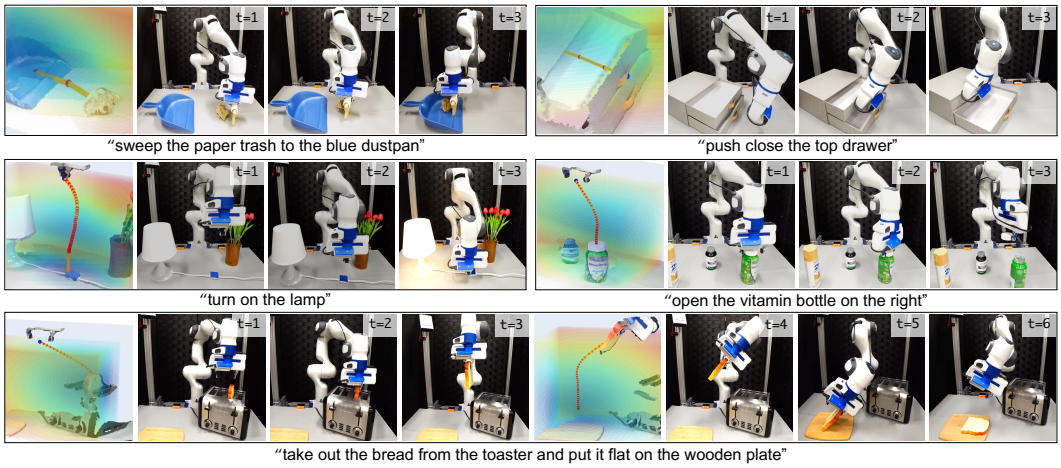

Figure 3: Visualization of composed 3D value maps and rollouts in real-world environments. The top row demonstrates object-centric tasks where value maps guide objects/parts toward target positions. The bottom two rows showcase robot-centric tasks. The bottom-most task involves two phases, which are also orchestrated by LLMs.

3.3 Zero-Shot Trajectory Synthesis with VoxPoser

After obtaining the task cost $\mathcal{F}_{task}$, we can now approach the full problem defined in Equation 1 to plan a motion trajectory. We use simple zeroth-order optimization by randomly sampling trajectories and scoring them with the proposed objective. The optimization is further implemented in a model predictive control framework that iteratively replans the trajectory at every step using the current observation to robustly execute the trajectories even under dynamic disturbances [2] , where either a learned or physics-based model can be used. However, because VoxPoser effectively provides "dense rewards" in the observation space and we are able to replan at every step, we surprisingly find that the overall system can already achieve a large variety of manipulation tasks considered in this work even with simple heuristics-based models. Since some value maps are defined over "entity of interest", which may not necessarily be the robot, we also use the dynamics model to find the needed robot trajectory to minimize the task cost (i.e., what interactions between the robot and the environment achieve the desired object motions). Because our method is agnostic to specific instantiations of the motion planners or MPC, we leave the more extended discussion of our implementation in Sec. 4.

3.4 Efficient Dynamics Learning with Online Experiences

While Sec. 3.3 presents a zero-shot framework for synthesizing trajectories for robot manipulation, herein we demonstrate how VoxPoser can also benefit from online experiences by efficiently learning a dynamics model. Specifically, consider the standard setup where a robot interleaves between 1) collecting environment transition data $(\mathbf{o}_t, \mathbf{a}_t, \mathbf{o}_{t+1})$, where $\mathbf{o}_t$ is the environment observation at time t and $\mathbf{a}_t = \text{MPC}(\mathbf{o}_t)$, and 2) training a dynamics model $\mathbf{g}_\theta$ parametrized by θ by minimizing the L2 loss between predicted next observation $\hat{\mathbf{o}}_{t+1}$ and $\mathbf{o}_{t+1}$. A critical component that determines the learning efficiency of the dynamics model is the action sampling distribution $P(\mathbf{a}_t \,|\, \mathbf{o}_t)$ in MPC, which typically is a random distribution over the full action space $\mathbf{A}$. This is often inefficient when the goal is to solve a particular task, such as opening a door, because most actions do not interact with the relevant objects in the scene (i.e., the door handle) nor do they necessarily interact with the objects in a meaningful way (i.e., pressing down the door handle). Since VoxPoser synthesizes robot trajectories with LLMs, which have a wealth of commonsense knowledge, the zero-shot synthesized trajectory $\tau_0^{\mathbf{r}}$ can serve as a useful prior to bias the action sampling distribution $P(\mathbf{a}_t \,|\, \mathbf{o}_t, \tau_0^{\mathbf{r}})$, which can significantly speed up the learning process. In practice, this can be implemented by only sampling actions in the vicinity of $\tau_0^{\mathbf{r}}$ by adding small noise ε to encourage local exploration instead of exploring in the full action space $\mathbf{A}$.

[2] Although involving an LLM in the loop, closed-loop execution is possible because the generated code remains the same throughout task ℓ_i, which allows us to cache its output for the current task.

4 Experiments and Analysis

We first discuss our design choices for implementing VoxPoser in Sec. 4.1. Then we validate directly in real-world systems whether VoxPoser can perform everyday manipulation tasks in Sec. 4.2. We also present a detailed quantitative study of the generalization performance of VoxPoser compared to learned and LLM-based baselines in simulation in Sec. 4.3. We further demonstrate how VoxPoser can benefit from only limited online experience to learn a dynamics model for contact-rich tasks in Sec. 4.4. Finally, we study the source of errors in the overall systems and discuss how improvement can be made in Sec. 4.5.

4.1 Implementation of VoxPoser

Herein we discuss our instantiation of VoxPoser. While we focus our discussion on design choices shared between the simulated and real-world domains, more details about environment setup can be found in Appendix.

LLMs and Prompting. We follow prompting structure by Liang et al. [71], which recursively calls LLMs using their own generated code, where each language model program (LMP) is responsible for a unique functionality (e.g., processing perception calls). We use GPT-4 [2] from OpenAI API. Prompts are in Appendix.

VLMs and Perception. Given an object/part query from LLMs, we first invoke open-vocab detector OWL-ViT [15] to obtain a bounding box, feed it into Segment Anything [103] to obtain a mask, and track the mask using video tracker XMEM [104]. The tracked mask is used with RGB-D observation to reconstruct the object/part point cloud.

Value Map Composition. We define the following types of value maps: affordance, avoidance, end-effector velocity, end-effector rotation, and gripper action. Each type uses a different LMP, which takes in an instruction and outputs a voxel map of shape $(100,100,100,k)$, where k differs for each value map (e.g., $k=1$ for affordance and avoidance as it specifies cost, and $k=4$ for rotation as it specifies $\mathrm{SO}(3)$). We apply Euclidean distance transform to affordance maps and Gaussian filters for avoidance maps. On top of value map LMPs, we define two high-level LMPs to orchestrate their behaviors: PLANNER takes user instruction $\mathcal{L}$ as input (e.g., "open drawer") and outputs a sequence of sub-tasks $\ell_{1:N}$, and COMPOSER takes in sub-task ℓ_i and invokes relevant value map LMPs with detailed language parameterization.

Motion Planner. We consider only affordance and avoidance maps in the planner optimization, which finds a sequence of collision-free end-effector positions $p_{1:N} \in \mathbb{R}^3$ using greedy search. Then we enforce other parametrization at each p by the remaining value maps (e.g., rotation map, velocity map). The cost map used by the motion planner is computed as the negative of the weighted sum of normalized affordance and avoidance maps with weights 2 and 1. After a 6-DoF trajectory is synthesized, the first waypoint is executed, and then a new trajectory is re-planned at a frequency of 5 Hz.

Environment Dynamics Model. For tasks in which the specified "entity of interest" is the robot, we assume identity environment dynamics while replanning at every step to account for the latest observation. For tasks in which the "entity of interest" is an object, we study only a planar pushing model parametrized by contact point, push direction, and push distance. The heuristic-based dynamics model translates an input point cloud along the push direction by the push distance. We use MPC with random shooting to optimize for the action parameters. Then a pre-defined pushing primitive is executed based on the action parameters. However, we note that a primitive is not necessary when action parameters are defined over the end-effector or joint space of the robot, which would likely yield smoother trajectories but takes more time for optimization.

4.2 VoxPoser for Everyday Manipulation Tasks

We study whether VoxPoser can zero-shot synthesize robot trajectories to perform everyday manipulation tasks. We set up a real-world tabletop environment with a Franka Emika Panda robot. More details can be found in Appendix A.2. While the proposed method can generalize to an open-set of instructions and an open-set of objects as shown in Fig. 1, we pick 5 representative tasks to provide quantitative evaluations. We also show additional qualitative results in Fig. 3 that include environment rollouts and value map visualizations. As shown in Table 1, we find that VoxPoser can effectively synthesize robot trajectories for everyday manipulation tasks with a high average success rate. Particularly, by leveraging rich world knowledge in LLMs, we can extract

Task	LLM + Prim. [71]		VoxPoser	
	Static	Dist.	Static	Dist.
Move & Avoid	0/10	0/10	9/10	8/10
Set Up Table	7/10	0/10	9/10	7/10
Close Drawer	0/10	0/10	10/10	7/10
Open Bottle	5/10	0/10	7/10	5/10
Sweep Trash	0/10	0/10	9/10	8/10
Total	24.0%	14.0%	88.0%	70.0%

Table 1: Average success rate in real-world domain. VoxPoser is capable of performing everyday manipulation tasks with high success, while more flexible and robust than the baseline with action primitives, especially under external disturbances.

Train/Test	Category	U-Net	Language Models	
		MP [48]	Prim. [71]	MP (Ours)
SI SA	Object Int.	21.0%	41.0%	64.0%
SI SA	Composition	53.8%	43.8%	77.5%
SI UA	Object Int.	3.0%	46.0%	60.0%
SI UA	Composition	3.8%	25.0%	58.8%
UI UA	Object Int.	0.0%	17.5%	65.0%
UI UA	Composition	0.0%	25.0%	76.7%

Table 2: Average success rate in simulated domain. "SI" and "UI" refer to seen and unseen instructions. "SA" and "UA" refer to seen and unseen attributes. VoxPoser outperforms both baselines across two categories (total of 13 tasks) not only on seen tasks but also on unseen tasks with similar success rates.

language-conditioned affordances for diverse scenes and objects. For example, LLMs can infer that a bottle can be opened by turning counter-clockwise around the z-axis. Subsequently, VoxPoser can ground this knowledge in the observation space that directly guides motion planners to complete the tasks. We further compare to a variant of Code as Policies [71] that uses LLMs to parameterize a pre-defined list of simple primitives (e.g., `move_to_pose`, `open_gripper`). We find that compared to chaining sequential policy logic, the ability to *compose spatially* while considering other constraints under a joint optimization scheme is a more flexible formulation, unlocking the possibility for more manipulation tasks and leading to more robust execution. In particular, by leveraging the composed spatial maps in MPC, VoxPoser can effectively recover external disturbances, such as moving targets/obstacles and pulling the drawer open after it has been closed by the robot.

4.3 Generalization to Unseen Instructions and Attributes

Herein we investigate the generalization capabilities of VoxPoser. To provide rigorous quantitative results, we set up a simulated environment that mirrors our real-world setup, but with a fixed list of objects (consisting of a cabinet along with 10 colored blocks and lines) and a fixed list of 13 templated instructions (e.g., "push [obj] to [pos]"), where [obj] and [pos] are attributes that are randomized over a pre-defined list. The instructions and attributes are divided into seen and unseen groups, where seen instructions/attributes may appear in the prompt (or in the training data for supervised baselines). We further group them into 2 categories, where "Object Interactions" refer to tasks that require interactions with objects (instead of collision-free path planning), and "Spatial Composition" refer to tasks that require robots to account for spatial constraints in the environment in their trajectories (e.g., moving slower near a particular object). For baselines, we ablate the two components of VoxPoser, LLM and motion planner, by comparing to a variant of [71] that combines an LLM with primitives and to a variant of [48] that learns a U-Net [105] to synthesize costmaps for motion planning. Table 2 shows the success rates averaged across 20 episodes per task. While VoxPoser performs on par with the supervised baseline on seen instructions and seen attributes, leveraging an LLM offers significantly better generalization in unseen attributes and more so in unseen instructions. Importantly, by grounding the knowledge of LLMs in the observation space, VoxPoser does not require designing a list of primitives suitable for a particular set of tasks and enjoys the ability to consider constraints in the environment.

4.4 Efficient Dynamics Learning with Online Experiences

Despite having zero-shot generalization to unseen instructions, we investigate how VoxPoser can also benefit from online interactions in tasks that involve contact-rich dynamics, as many behavioral nuances may not be present in the LLMs. To this end, we investigate a suite of simulated tasks involving interactions with common articulated objects, including opening doors, fridges, and windows. We hypothesize that while these are challenging tasks for autonomous agents due to difficult exploration, trajectories zero-shot synthesized by VoxPoser would provide useful hints for exploration (e.g., "handle needs to be pressed down first in order to open a door"). Specifically, we first synthesize k different trajectories using VoxPoser, each represented as a sequence of end-effector waypoints. Then we learn a simple MLP dynamics model that predicts $\mathbf{o}_{t+1}$ from $\mathbf{o}_t$ and $\mathbf{a}_t$ through an iterative procedure where the agent alternates between data collection and model learning. The initial synthesized trajectories are used as prior in the action sampling distribution of MPC,

	Zero-Shot	No Prior		w/ Prior	
Task	Success	Success	Time(s)	Success	Time(s)
Door	6.7%$_{\pm 4.4\%}$	58.3$_{\pm 4.4\%}$	TLE	88.3%$_{\pm 1.67\%}$	142.3$_{\pm 22.4}$
Window	3.3%$_{\pm 3.3\%}$	36.7%$_{\pm 1.7\%}$	TLE	80.0%$_{\pm 2.9\%}$	137.0$_{\pm 7.5}$
Fridge	18.3%$_{\pm 3.3\%}$	70.0%$_{\pm 2.9\%}$	TLE	91.7%$_{\pm 4.4\%}$	71.0$_{\pm 4.4}$

Table 3: Zero-shot synthesized trajectories by VoxPoser, though limiting at contact-rich tasks, can serve as priors for efficient learning from online interactions. Compared to exploring without the priors, VoxPoser can learn to open articulated objects in under 3 minutes. TLE (time limit exceeded) means exceeding 12 hours. Experiments are done with 3 different seeds.

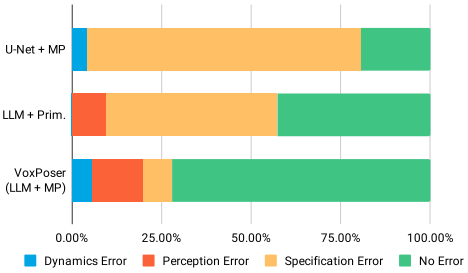

Figure 4: Error breakdown of VoxPoser and baselines. Compared to baselines, although VoxPoser relies on multiple components, it effectively reduces the error in specifying objectives/values.

where we add $\varepsilon \sim \mathcal{N}(0,\sigma^2)$ to each waypoint in τ_0^{r} to encourage local exploration. Results are shown in Table 3. For these tasks involving complex interactions with articulated objects, we find trajectories zero-shot synthesized by VoxPoser are meaningful but insufficient. However, we can learn an effective dynamics model with less than 3 minutes of online interactions by using these trajectories as exploration prior, leading to high eventual success rates. In comparison, without using a prior, it is extremely difficult to learn a dynamics model because most actions do not lead to meaningful changes in the environment. In all cases, we terminate the experiments as their times exceed the limit of 12 hours.

4.5 Error Breakdown

Since VoxPoser involves multiple components working jointly to synthesize trajectories for various manipulation tasks, herein we analyze the errors resulting from each component and how the overall system can be further improved. We conduct experiments in simulation where we have access to ground-truth perception and dynamics model (i.e., the simulator). Results are shown in Fig. 4. **U-Net + MP** [48] trains a U-Net [105, 106] to directly map RGB-D observations to value maps, thus not having an independent perception module. "Specification error" refers to the error by U-Net, such as noisy predictions that are difficult to be used by motion planners. **LLM + Primitives** [71] uses LLMs to sequentially compose primitives, thus not having a dynamics module. And its "specification error" refers to errors by LLMs when composing policy logic, where common errors include incorrect specification of primitive parameters by LLMs. In comparison, although VoxPoser uses multiple components, by formulating it as a joint model-based optimization problem, VoxPoser achieves the lowest overall error, and its biggest source of error is the perception module. We also observe that having access to a better dynamics model (rather than a heuristics-based model) can contribute to better overall performance, such as a learned model or a physics-based model.

5 Conclusion, Limitations, & Future Works

In this work, we present **VoxPoser**, a general robot manipulation framework that extracts affordances and constraints from LLMs, offering significant generalization advantages for open-set instructions and objects. In particular, we use code-writing LLMs to compose 3D value maps grounded in observation space, which are used to synthesize trajectories for everyday manipulation tasks. Furthermore, we show VoxPoser can benefit from online interactions by efficiently learning a dynamics model for contact-rich tasks.

VoxPoser has several limitations. First, it relies on external perception modules, which is limiting in tasks that require holistic visual reasoning or understanding of fine-grained object geometries. Second, while applicable to efficient dynamics learning, a general-purpose dynamics model is still required to achieve contact-rich tasks. Third, our motion planner considers only end-effector trajectories while whole-arm planning is also feasible and likely a better design choice [107–109]. Finally, manual prompt engineering is required for LLMs.

We also see several exciting venues for future work. For instance, the recent success of multi-modal LLMs [66, 2, 110] can be directly translated into VoxPoser for direct visual grounding. Methods developed for alignment [111, 112] and prompting [113] can also be used to improve the quality of synthesized value maps and alleviate prompt engineering effort. Finally, while we use greedy search in trajectory optimization, more advanced optimization methods can be developed that best interface with value maps synthesized by LLMs.

References

[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[2] OpenAI. Gpt-4 technical report. *arXiv*, 2023.

[3] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

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

[5] S. Tellex, N. Gopalan, H. Kress-Gazit, and C. Matuszek. Robots that use language. *Annual Review of Control, Robotics, and Autonomous Systems*, 2020.

[6] S. Tellex, T. Kollar, S. Dickerson, M. Walter, A. Banerjee, S. Teller, and N. Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 25, pages 1507–1514, 2011.

[7] T. Kollar, S. Tellex, D. Roy, and N. Roy. Toward understanding natural language directions. In *2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 259–266. IEEE, 2010.

[8] M. Bollini, S. Tellex, T. Thompson, N. Roy, and D. Rus. Interpreting and executing recipes with a cooking robot. In *Experimental Robotics*, pages 481–495. Springer, 2013.

[9] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*. PMLR, 2022.

[10] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, and M. Yan. Do as i can and not as i say: Grounding language in robotic affordances. In *arXiv preprint arXiv:2204.01691*, 2022.

[11] A. Zeng, A. Wong, S. Welker, K. Choromanski, F. Tombari, A. Purohit, M. Ryoo, V. Sindhwani, J. Lee, V. Vanhoucke, et al. Socratic models: Composing zero-shot multimodal reasoning with language. *arXiv preprint arXiv:2204.00598*, 2022.

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

[13] X. Gu, T.-Y. Lin, W. Kuo, and Y. Cui. Open-vocabulary object detection via vision and language knowledge distillation. *arXiv preprint arXiv:2104.13921*, 2021.

[14] A. Kamath, M. Singh, Y. LeCun, G. Synnaeve, I. Misra, and N. Carion. Mdetr-modulated detection for end-to-end multi-modal understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1780–1790, 2021.

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

[16] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020. doi:10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

[17] J. Andreas, D. Klein, and S. Levine. Learning with latent language. *arXiv preprint arXiv:1711.00482*, 2017.

[18] R. Zellers, A. Holtzman, M. Peters, R. Mottaghi, A. Kembhavi, A. Farhadi, and Y. Choi. Piglet: Language grounding through neuro-symbolic interaction in a 3d world. *arXiv preprint arXiv:2106.00188*, 2021.

[19] R. Zellers, X. Lu, J. Hessel, Y. Yu, J. S. Park, J. Cao, A. Farhadi, and Y. Choi. Merlot: Multimodal neural script knowledge models. *Advances in Neural Information Processing Systems*, 2021.

[20] V. Shwartz, P. West, R. L. Bras, C. Bhagavatula, and Y. Choi. Unsupervised commonsense question answering with self-talk. *arXiv preprint arXiv:2004.05483*, 2020.

[21] T. Winograd. Procedures as a representation for data in a computer program for understanding natural language. Technical report, MASSACHUSETTS INST OF TECH CAMBRIDGE PROJECT MAC, 1971.

[22] V. Blukis, R. A. Knepper, and Y. Artzi. Few-shot object grounding and mapping for natural language robot instruction following. *arXiv preprint arXiv:2011.07384*, 2020.

[23] S. Tellex, R. Knepper, A. Li, D. Rus, and N. Roy. Asking for help using inverse semantics. 2014.

[24] T. Kollar, S. Tellex, D. Roy, and N. Roy. Grounding verbs of motion in natural language commands to robots. In *Experimental robotics*, pages 31–47. Springer, 2014.

[25] J. Thomason, S. Zhang, R. J. Mooney, and P. Stone. Learning to interpret natural language commands through human-robot dialog. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[26] J. Thomason, A. Padmakumar, J. Sinapov, N. Walker, Y. Jiang, H. Yedidsion, J. Hart, P. Stone, and R. Mooney. Jointly improving parsing and perception for natural language commands through human-robot dialog. *Journal of Artificial Intelligence Research*, 67:327–374, 2020.

[27] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2021.

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

[29] D. Shah, B. Osinski, B. Ichter, and S. Levine. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. *arXiv preprint arXiv:2207.04429*, 2022.

[30] Y. Cui, S. Karamcheti, R. Palleti, N. Shivakumar, P. Liang, and D. Sadigh. " no, to the right"–online language corrections for robotic manipulation via shared autonomy. *arXiv preprint arXiv:2301.02555*, 2023.

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

[32] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[33] Y. J. Ma, V. Kumar, A. Zhang, O. Bastani, and D. Jayaraman. Liv: Language-image representations and rewards for robotic control. 2023.

[34] P. A. Jansen. Visually-grounded planning without vision: Language models infer detailed plans from high-level instructions. *arXiv preprint arXiv:2009.14259*, 2020.

[35] V. Micheli and F. Fleuret. Language models are few-shot butlers. *arXiv preprint arXiv:2104.07972*, 2021.

[36] P. Sharma, A. Torralba, and J. Andreas. Skill induction and planning with latent language. *arXiv preprint arXiv:2110.01517*, 2021.

[37] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, P. Sermanet, N. Brown, T. Jackson, L. Luu, S. Levine, K. Hausman, and B. Ichter. Inner monologue: Embodied reasoning through planning with language models. In *arXiv preprint arXiv:2207.05608*, 2022.

[38] B. Z. Li, W. Chen, P. Sharma, and J. Andreas. Lampp: Language models as probabilistic priors for perception and action. *arXiv e-prints*, pages arXiv–2302, 2023.

[39] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[41] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[42] S. Nair, E. Mitchell, K. Chen, S. Savarese, C. Finn, et al. Learning language-conditioned robot behavior from offline data and crowd-sourced annotation. In *Conference on Robot Learning*, pages 1303–1315. PMLR, 2022.

[43] M. Shridhar, L. Manuelli, and D. Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pages 894–906. PMLR, 2022.

[44] M. Shridhar, L. Manuelli, and D. Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Proceedings of the 6th Conference on Robot Learning (CoRL)*, 2022.

[45] S. Li, X. Puig, Y. Du, C. Wang, E. Akyurek, A. Torralba, J. Andreas, and I. Mordatch. Pre-trained language models for interactive decision-making. *arXiv preprint arXiv:2202.01771*, 2022.

[46] O. Mees, L. Hermann, and W. Burgard. What matters in language conditioned robotic imitation learning. *arXiv preprint arXiv:2204.06252*, 2022.

[47] O. Mees, J. Borja-Diaz, and W. Burgard. Grounding language with visual affordances over unstructured data. *arXiv preprint arXiv:2210.01911*, 2022.

[48] P. Sharma, B. Sundaralingam, V. Blukis, C. Paxton, T. Hermans, A. Torralba, J. Andreas, and D. Fox. Correcting robot plans with natural language feedback. *arXiv preprint arXiv:2204.05186*, 2022.

[49] W. Liu, C. Paxton, T. Hermans, and D. Fox. Structformer: Learning spatial structure for language-guided semantic rearrangement of novel objects. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6322–6329. IEEE, 2022.

[50] C. Lynch and P. Sermanet. Language conditioned imitation learning over unstructured data. *Robotics: Science and Systems*, 2021. URL https://arxiv.org/abs/2005.07648.

[51] C. Lynch, A. Wahid, J. Tompson, T. Ding, J. Betker, R. Baruch, T. Armstrong, and P. Florence. Interactive language: Talking to robots in real time. *arXiv preprint arXiv:2210.06407*, 2022.

[52] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. *The International Journal of Robotics Research*, 40(12-14):1419–1434, 2021.

[53] J. Luketina, N. Nardelli, G. Farquhar, J. N. Foerster, J. Andreas, E. Grefenstette, S. Whiteson, and T. Rocktäschel. A survey of reinforcement learning informed by natural language. In *IJCAI*, 2019.

[54] J. Andreas, D. Klein, and S. Levine. Modular multitask reinforcement learning with policy sketches. *ArXiv*, abs/1611.01796, 2017.

[55] Y. Jiang, S. S. Gu, K. P. Murphy, and C. Finn. Language as an abstraction for hierarchical deep reinforcement learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[56] B. Chen, F. Xia, B. Ichter, K. Rao, K. Gopalakrishnan, M. S. Ryoo, A. Stone, and D. Kappler. Open-vocabulary queryable scene representations for real world planning. *arXiv preprint arXiv:2209.09874*, 2022.

[57] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg. Progprompt: Generating situated robot task plans using large language models. *arXiv preprint arXiv:2209.11302*, 2022.

[58] C. Huang, O. Mees, A. Zeng, and W. Burgard. Visual language maps for robot navigation. *arXiv preprint arXiv:2210.05714*, 2022.

[59] S. S. Raman, V. Cohen, E. Rosen, I. Idrees, D. Paulius, and S. Tellex. Planning with large language models via corrective re-prompting. *arXiv preprint arXiv:2211.09935*, 2022.

[60] C. H. Song, J. Wu, C. Washington, B. M. Sadler, W.-L. Chao, and Y. Su. Llm-planner: Few-shot grounded planning for embodied agents with large language models. *arXiv preprint arXiv:2212.04088*, 2022.

[61] B. Liu, Y. Jiang, X. Zhang, Q. Liu, S. Zhang, J. Biswas, and P. Stone. Llm+ p: Empowering large language models with optimal planning proficiency. *arXiv preprint arXiv:2304.11477*, 2023.

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

[63] K. Lin, C. Agia, T. Migimatsu, M. Pavone, and J. Bohg. Text2motion: From natural language instructions to feasible plans. *arXiv preprint arXiv:2303.12153*, 2023.

[64] Y. Ding, X. Zhang, C. Paxton, and S. Zhang. Task and motion planning with large language models for object rearrangement. *arXiv preprint arXiv:2303.06247*, 2023.

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

[66] 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.

[67] H. Yuan, C. Zhang, H. Wang, F. Xie, P. Cai, H. Dong, and Z. Lu. Plan4mc: Skill reinforcement learning and planning for open-world minecraft tasks. *arXiv preprint arXiv:2303.16563*, 2023.

[68] Y. Xie, C. Yu, T. Zhu, J. Bai, Z. Gong, and H. Soh. Translating natural language to planning goals with large-language models. *arXiv preprint arXiv:2302.05128*, 2023.

[69] Y. Lu, P. Lu, Z. Chen, W. Zhu, X. E. Wang, and W. Y. Wang. Multimodal procedural planning via dual text-image prompting. *arXiv preprint arXiv:2305.01795*, 2023.

[70] D. Patel, H. Eghbalzadeh, N. Kamra, M. L. Iuzzolino, U. Jain, and R. Desai. Pretrained language models as visual planners for human assistance. *arXiv preprint arXiv:2304.09179*, 2023.

[71] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng. Code as policies: Language model programs for embodied control. *arXiv preprint arXiv:2209.07753*, 2022.

[72] M. Kwon, S. M. Xie, K. Bullard, and D. Sadigh. Reward design with language models. *arXiv preprint arXiv:2303.00001*, 2023.

[73] A. Tam, N. Rabinowitz, A. Lampinen, N. A. Roy, S. Chan, D. Strouse, J. Wang, A. Banino, and F. Hill. Semantic exploration from language abstractions and pretrained representations. *Advances in Neural Information Processing Systems*, 35:25377–25389, 2022.

[74] J. Mu, V. Zhong, R. Raileanu, M. Jiang, N. Goodman, T. Rocktäschel, and E. Grefenstette. Improving intrinsic exploration with language abstractions. *arXiv preprint arXiv:2202.08938*, 2022.

[75] C. Colas, T. Karch, N. Lair, J.-M. Dussoux, C. Moulin-Frier, P. Dominey, and P.-Y. Oudeyer. Language as a cognitive tool to imagine goals in curiosity driven exploration. *Advances in Neural Information Processing Systems*, 33:3761–3774, 2020.

[76] Y. Du, O. Watkins, Z. Wang, C. Colas, T. Darrell, P. Abbeel, A. Gupta, and J. Andreas. Guiding pre-training in reinforcement learning with large language models. *arXiv preprint arXiv:2302.06692*, 2023.

[77] H. Hu and D. Sadigh. Language instructed reinforcement learning for human-ai coordination. *arXiv preprint arXiv:2304.07297*, 2023.

[78] I. Lenz, R. A. Knepper, and A. Saxena. Deepmpc: Learning deep latent features for model predictive control. In *Robotics: Science and Systems*, volume 10. Rome, Italy, 2015.

[79] L. Hewing, K. P. Wabersich, M. Menner, and M. N. Zeilinger. Learning-based model predictive control: Toward safe learning in control. *Annual Review of Control, Robotics, and Autonomous Systems*, 3:269–296, 2020.

[80] M. B. Chang, T. Ullman, A. Torralba, and J. B. Tenenbaum. A compositional object-based approach to learning physical dynamics. *arXiv preprint arXiv:1612.00341*, 2016.

[81] P. Battaglia, R. Pascanu, M. Lai, D. Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. *Advances in neural information processing systems*, 29, 2016.

[82] Z. Xu, J. Wu, A. Zeng, J. B. Tenenbaum, and S. Song. Densephysnet: Learning dense physical object representations via multi-step dynamic interactions. *arXiv preprint arXiv:1906.03853*, 2019.

[83] A. Byravan and D. Fox. Se3-nets: Learning rigid body motion using deep neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 173–180. IEEE, 2017.

[84] A. Nagabandi, K. Konolige, S. Levine, and V. Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning*, pages 1101–1112. PMLR, 2020.

[85] A. Sanchez-Gonzalez, N. Heess, J. T. Springenberg, J. Merel, M. Riedmiller, R. Hadsell, and P. Battaglia. Graph networks as learnable physics engines for inference and control. In *International Conference on Machine Learning*, pages 4470–4479. PMLR, 2018.

[86] Y. Li, J. Wu, R. Tedrake, J. B. Tenenbaum, and A. Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. *arXiv preprint arXiv:1810.01566*, 2018.

[87] C. Finn, S. Levine, and P. Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*, pages 49–58. PMLR, 2016.

[88] J. Fu, K. Luo, and S. Levine. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017.

[89] D. Driess, O. Oguz, J.-S. Ha, and M. Toussaint. Deep visual heuristics: Learning feasibility of mixed-integer programs for manipulation planning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9563–9569. IEEE, 2020.

[90] B. Amos, I. Jimenez, J. Sacks, B. Boots, and J. Z. Kolter. Differentiable mpc for end-to-end planning and control. *Advances in neural information processing systems*, 31, 2018.

[91] M. Mittal, D. Hoeller, F. Farshidian, M. Hutter, and A. Garg. Articulated object interaction in unknown scenes with whole-body mobile manipulation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1647–1654. IEEE, 2022.

[92] S. Bahl, A. Gupta, and D. Pathak. Human-to-robot imitation in the wild. *arXiv preprint arXiv:2207.09450*, 2022.

[93] S. Bahl, R. Mendonca, L. Chen, U. Jain, and D. Pathak. Affordances from human videos as a versatile representation for robotics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13778–13790, 2023.

[94] H. Bharadhwaj, A. Gupta, S. Tulsiani, and V. Kumar. Zero-shot robot manipulation from passive human videos. *arXiv preprint arXiv:2302.02011*, 2023.

[95] Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, and A. Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

[96] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 720–736, 2018.

[97] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

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

[99] T. Yu, T. Xiao, A. Stone, J. Tompson, A. Brohan, S. Wang, J. Singh, C. Tan, J. Peralta, B. Ichter, et al. Scaling robot learning with semantically imagined experience. *arXiv preprint arXiv:2302.11550*, 2023.

[100] Z. Mandi, H. Bharadhwaj, V. Moens, S. Song, A. Rajeswaran, and V. Kumar. Cacti: A framework for scalable multi-task multi-scene visual imitation learning. *arXiv preprint arXiv:2212.05711*, 2022.

[101] T. Xiao, H. Chan, P. Sermanet, A. Wahid, A. Brohan, K. Hausman, S. Levine, and J. Tompson. Robotic skill acquisition via instruction augmentation with vision-language models. *arXiv preprint arXiv:2211.11736*, 2022.

[102] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.

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

[104] H. K. Cheng and A. G. Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVIII*, pages 640–658. Springer, 2022.

[105] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[106] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II 19*, pages 424–432. Springer, 2016.

[107] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE transactions on Robotics and Automation*, 12(4):566–580, 1996.

[108] N. D. Ratliff, J. Issac, D. Kappler, S. Birchfield, and D. Fox. Riemannian motion policies. *arXiv preprint arXiv:1801.02854*, 2018.

[109] T. Marcucci, M. Petersen, D. von Wrangel, and R. Tedrake. Motion planning around obstacles with convex optimization. *arXiv preprint arXiv:2205.04422*, 2022.

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

[111] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022.

[112] Y. Wang, Y. Kordi, S. Mishra, A. Liu, N. A. Smith, D. Khashabi, and H. Hajishirzi. Self-instruct: Aligning language model with self generated instructions. *arXiv preprint arXiv:2212.10560*, 2022.

[113] J. Wei, X. Wang, D. Schuurmans, M. Bosma, E. Chi, Q. Le, and D. Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

[114] Y. Zhu, A. Joshi, P. Stone, and Y. Zhu. Viola: Imitation learning for vision-based manipulation with object proposal priors. *6th Annual Conference on Robot Learning*, 2022.

[115] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11097–11107, 2020.