LEVERAGING AFFORDANCE REPRESENTATIONS FOR ROBOT LEARNING

AN UNDERGRADUATE HONORS THESIS
SUBMITTED TO THE DEPARTMENT OF SYMBOLIC SYSTEMS
OF STANFORD UNIVERSITY
IN FULFILLMENT OF THE REQUIREMENTS
FOR HONORS IN THE DEGREE OF
BACHELOR OF SCIENCE

Olivia Y. Lee
June 2024

To the Directors of the Program on Symbolic Systems: I certify that I have read the thesis of Olivia Y. Lee in its final form for submission and have found it to be satisfactory for the degree of Bachelor of Science with Honors.

June 5th, 2024
Prof. Chelsea Finn
Department of Computer Science

To the Directors of the Program on Symbolic Systems: I certify that I have read the thesis of Olivia Y. Lee in its final form for submission and have found it to be satisfactory for the degree of Bachelor of Science with Honors.

June 5th, 2024
Prof. Nick Haber
Graduate School of Education

Abstract

Humans are capable of adapting to novel environments quickly using prior knowledge from past experiences. We can identify new instantiations of previously encountered object classes and easily apply previously learned skills to these new objects, both of which current embodied AI agents struggle with. Online reinforcement learning, where a robotic agent learns a mapping from states to actions to maximize a reward signal, provides a potential solution by enabling robots to learn from trial-and-error. However, current methods are sample-inefficient, lack shaping rewards, and require frequent resets. We propose a method to address the lack of shaping rewards using affordances, the action potential of objects, to create a dense shaping reward for online reinforcement learning. We leverage state-of-the-art vision-language models (VLMs) to predict keypoint-based affordance representations, which we use as intermediate dense rewards for online reinforcement learning, in addition to sparse task completion rewards. We demonstrate that dense shaping rewards speed up online reinforcement learning for robotic manipulation, and enables robots to succeed on a variety of object manipulation tasks, informed by human interaction priors encoded in VLMs.

Acknowledgments

First, I would like express my gratitude to my research advisor, Chelsea Finn. Chelsea's feedback on my research directions and advice for my undergraduate career have been incredibly valuable. I am especially grateful for the mentorship of several members of her lab over the last three years, namely Annie Xie, Suraj Nair, Karl Pertsch, Kuan Fang of UC Berkeley's RAIL Lab, and for the strong community of researchers in IRIS Lab members who continually inspire me with their impactful work and ideas. It is an honor to have been part of this lab since my freshman year and introduced to robotics and reinforcement learning research through this group.

Next, I would like to thank my major advisor, Nick Haber, for his insights and encouragement throughout the project. Nick provided great alternative perspectives on the project, and inspired my interests in related areas such as curiosity, model-based reinforcement learning, and intrinsic motivation. Our discussions have influenced my research interests in bridging human and artificial intelligence, and his excellent teaching has been fundamental to my experience in Symbolic Systems.

I am fortunate to have learned a great deal from several other professors, including Joshua O'Rourke, Thomas Icard, Jeannette Bohg, Dan Jurafsky, Michael Genesereth, and Mykel Kochenderfer. The Symbolic Systems faculty, and these faculty in particular, do a fantastic job embodying the program's interdisciplinary nature in the way they teach, lead discussions, and connect ideas. Credit assignment is a difficult task, but I believe many of my ideas are inspired one way or another by their teachings. Special thanks also to Todd Davies and Michael Frank for leading the department and building the Symbolic Systems community, a truly special part of my Stanford experience.

I have made many amazing friendships through the department: friends who either declared SymSys or contemplated it, the Advising Fellows, the SymSys Society, my Big Sibs and Little Sibs throughout the years, and all the great minds I've met and exchanged ideas with through shared classes. I would especially like to thank my close friends who have seen me through it all - from my first time tinkering with a Franka robot to the writing of this honors thesis - and have been integral to my cherished memories of Stanford. I hope there will be many more to come.

Last but not least, this thesis is dedicated to my family. Even from home far away in Singapore, I have felt their unwavering love and support every step of the way as I navigate life at Stanford and beyond. It is to them that I owe all I have learned and accomplished.

Contents

List of Figures

Chapter 1

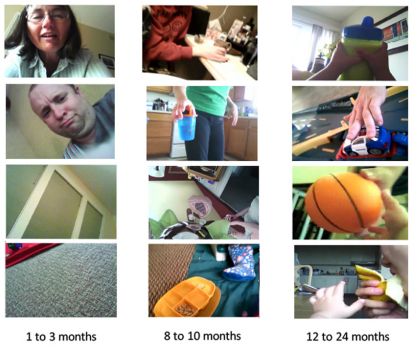
Introduction

An enduring goal that robotics researchers have united around is a general-purpose robot: one that can quickly explore the dynamics of a new environment and then perform a range of useful downstream tasks. There remain many technological developments across the robotics stack, both in hardware and software, for this goal to be achieved. In particular, one key challenge for robotics is the ability to generalize to new objects, tasks, and environments. Humans demonstrate this ability: consider walking into a kitchen with new objects and a different layout from one's familiar home kitchen. After a short exploration phase, one should be able to make a cup of tea by leveraging information about the kitchen learned during exploration, prior understanding of how to manipulate kettles and teabags, as well as previously learned skills like pouring and stirring. Present day embodied agents still struggle with all of the above: intelligent exploration, semantic reasoning, and skill transfer. The central question this work revolves around is how we can develop embodied agents that generalize and adapt quickly to novel environments.

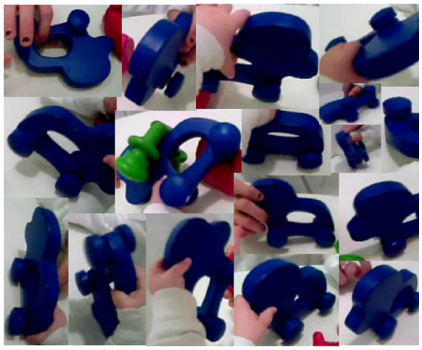
Turning to our understanding of human learning and adaptation is a natural attempt to grapple with this question, as with the kitchen example above. Humans are capable of exploring novel environments using prior knowledge, acquired both as a passive observer and as an active participant in the real world. As a passive observer, a great deal knowledge about the world we have acquired from observing the actions and interactions of others, whether in the real world or now through the virtual world; consider the act of watching how-to videos on YouTube, for instance. In addition, taking actions in the real world, as an active participant in it, is crucial for understanding action dynamics as an agent instantiated in the world, with a particular embodiment and set of skills.

Consider the visual experience of a child: for the first 3 months of their life, a child sees a few dominant faces and limited views of their environment. From 8 to 10 months, they see more cluttered scenes of objects and how people around them interact with objects as their head movements become more diverse. Finally, from 12 to 24 months, grasping and interaction with objects occurs (Figure 1.1a). Notably, 12 to 18 month-old children's category judgments are characterized

by numerous over- and under-generalizations, and sometimes failure to recognize known objects in visually crowded scenes. After 24 months, however, given just one instance of a novel category and its name, most children generalize that name in an adult-like manner [77].



(a) Sample head-camera captured images for three different age ranges of infants.



(b) Sample images of a single object captured by a 15-month-old infant's head-camera during play.

Figure 1.1: Sample images describing a child's visual experience in the first 24 months. [77]

As demonstrated by the developmental process of a child's visual experience, there is a significant period of largely passive observation, where a baby learns key skills like facial and object recognition, as well as the ability to generalize to new faces and objects, before the infant takes any significant actions. This period of passive observation is crucial for their development: infants deprived of early visual input due to congenital cataracts that were removed at just 2 to 6 months of age faced a permanent deficit in configural face processing [51]. From a computational standpoint, this can be viewed as a form of pretraining on a diverse set of scenes with varying objects, environments, and viewpoints. Through this process, besides acquiring the important skills mentioned above, infants also learn "intuitive physics" in the real world such as gravity [34], the ability to generalize to new objects and scenes, and important interaction priors by visually observing people around them (e.g., their parents) interact with objects.

At 12 to 24 months, infants begin taking their first actions such as grasping, picking, placing, pushing, and pulling. While such skills seem primitive, these first attempts at interacting with the world are more structured and informed than they seem; the abundance of priors acquired through passive observation are finally being put into action. While it takes several trials over time to refine these skills, infants require fewer trials per task than artificial agents to learn the same skills to a similar degree of robustness. Infants also display the remarkable generalization ability to different object types, object positions, backgrounds, and other changes to visual input, much faster and more robustly than artificial agents. Furthermore, infants learn these skills messily without a specific order in highly variable environments, yet they retain the fundamental skills allowing them to perform the same tasks in new situations; today's artificial agents still struggle with multi-task learning, continual learning without catastrophic forgetting, and robustness to environment changes.

Notably, there is a significant perspective shift between third-personal observations of object interactions and first-personal interactions with said objects. Infants' early attempts to refine physical skills also seek to bridge that gap by understanding physical interaction from egocentric viewpoints (Figure 1.1b). Besides the viewpoint shift, babies ultimately need to learn how to apply learned priors about physics and interaction by taking actions in the real world, as opposed to just watching how others interact with objects. Even as adults, we are constantly learning new skills and tasks. It is a common experience to watch others perform novel tasks and interact with new objects in the real world or via the Internet, which provides significant information about a task we try to do ourselves. That said, no matter how many live demonstrations or YouTube videos we watch, figuring out how to complete the same task in *our own* embodiment using *our own* skillsets requires some degree of trial-and-error.

This two-phase approach to learning new skills - a phase of passively observing others followed by a phase of learning through trial-and-error - inspires the offline-online framework for robot learning that is presented in this thesis. The phase of passive, offline observation is equivalent to the *pretraining* phase of robot learning algorithms. Pretraining commonly leverages the large amount of human and robot data that has been collected via the Internet, physical human demonstrations, and large-scale data collection efforts like the Bridge dataset [24] [85] and the Open-X Embodiment dataset [7], to learn rewards, representations, and skills in an offline manner, without the agent taking actions in the real world. These informative priors serve to speed up the second phase of learning, where learning entirely from scratch would be time-intensive and computationally inefficient. The second phase of online learning through trial-and-error is equivalent to the *online finetuning* phase of robot learning algorithms. Specifically, we turn to *reinforcement learning*, a machine learning paradigm that learns policies, or mappings from states to actions, to maximize a numerical reward signal [80], as an implementation of online finetuning for robotic agents. The egocentric notion of learning physical skills specific to the agent's embodiment motivates this phase, and this work delves into approaches to online finetuning of pretrained policies.

This thesis presents a method to address the lack of shaped rewards in online reinforcement learning, by leveraging pretrained visual language models (VLMs). In the rest of Section 1, to lay the foundation for this research, we explore several areas of active research, including affordance learning, foundation models for robotics, and autonomous reinforcement learning. In Section 2, we explore an alternative method of learning shaping rewards from diverse egocentric human videos, and discuss the benefits and limitations of this approach. Finally, in Section 3, we present our method of extracting dense rewards for online reinforcement learning from VLMs. We demonstrate the benefit of using affordance-based waypoints and keypoints as dense rewards for faster learning on a variety of robot manipulation tasks.

1.1 Learning & Predicting Affordances

Humans are capable of adapting to novel environments quickly using prior knowledge from past experiences or other forms of information prior to exploration time. Consider the case of entering a new hotel room: even though this specific room and the objects within it are new to us, we can successfully identify light switches, water taps, and remote controls despite not encountering those exact instantiations of the objects before. We also don't need to re-learn how to toggle light switches or turn knobs and can easily apply previously learned skills to these new objects. What we don't know, and must learn via exploration, are the dynamics of this new environment - which switches control which lights in this room, or which direction to turn the tap for hot water - that we can later use for downstream tasks, like making a cup of tea. In such scenarios two main tasks are involved:

1. identifying specific instantiations of objects in this particular environment, and interaction points with high likelihoods of success

2. applying the appropriate skills (learned beforehand) to those interaction points

The concept of *affordances*, introduced by Gibson [1979] can be helpful for both of these tasks. First introduced by Gibson to mean "what it offers the animal, what it provides or furnishes ... It implies the complementarity of the animal and the environment" [29], an affordance can be understood as the potential for action, ranging from actions applied to an object to how the object might be used. Visual affordances (i.e., where one should interact with this object) can help with learning the interactive regions with high likelihoods of success. Skill affordances (i.e., what one can do with this object) can help with selecting a subset of the agent's learned skills that would be appropriate to apply to the given object at the identified interaction points. Pretraining can be helpful for learning both types of affordances.

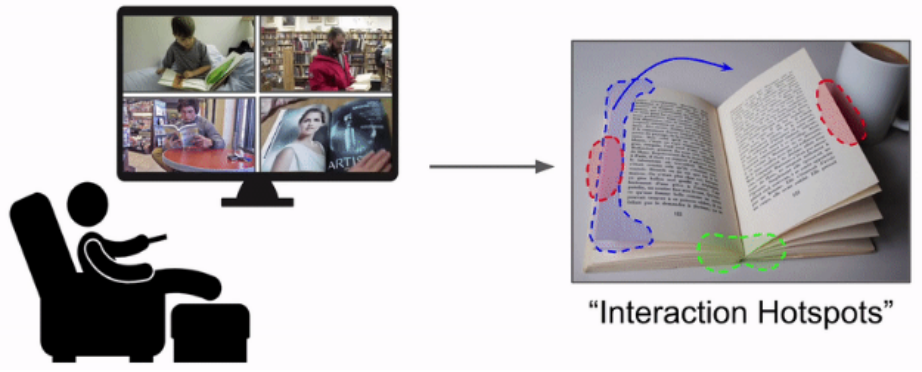


Figure 1.2: Pretraining on diverse prior experiences and inferring interaction regions with high likelihoods of success on a novel object at test time. [58]

Prior work has demonstrated successful learning of visual affordances in the form of "interaction hotspot" representations from first- and third-person videos [58] as well as the usefulness of affordance landscapes of new unmapped 3D environments for intelligent interaction exploration [57]. These works jointly demonstrate the value of learning useful interaction priors, specifically contact points, by leveraging structural assumptions in diverse human behavior data from video. Both methods involve training affordance models that predict action potential for specific image segments - given a novel scene, the model jointly predicts contact regions that are useful as well as specific actions that are sensible to apply to those regions. The outputs of such models greatly streamline processes of exploration and task completion by guiding embodied agents towards useful interaction points (visual affordances) and narrowing down the agent's action space (skill affordances). Affordance models can thus leverage the large amount of human data at our disposal, such as Ego4D [25], HowTo100M [55], and EPIC-KITCHENS [19].

When dealing with highly variable and unstructured data like human videos, it is necessary to have useful heuristics to extract meaning from this data. Several works have demonstrated the usefulness of explicitly tracking human hand poses in videos, premised on human hand interactions being an essential source of affordance information. Hand-tracking models trained extensively on Internet data can be used off-the-shelf to study human hand interactions with objects at scale [75]. They can be applied to a variety of settings, such as learning from diverse human videos [12] or in-domain human play data [86]. In Wang et al. [2023] specifically, MimicPlay uses a hand tracker on human video data, and trains a visual encoder model that takes the current frame and goal frame and predicts a trajectory of human hand poses, using a KL loss between human and robot images to encourage embodiment invariance. This demonstrates both a strength and limitation of learning strictly from human videos: while human interaction data provides strong priors to bias robotic manipulation towards intelligent actions, the embodiment shift still remains a problem; the most optimal way for a dexterous, five-fingered human hand to open a drawer may not be analogous to the optimal way for a two-fingered robotic gripper to do the same task. Nonetheless, in most works studying affordance learning, hand tracking has been the dominant heuristic for extracting structured interaction information from highly unstructured human video datasets. While some works demonstrate zero-shot success on coarse manipulation tasks [12], most complex tasks require some level of online finetuning [38], and the latter is the avenue this work aims to explore.

Besides being able to leverage large human video datasets and thus reducing the quantity of robot data that needs to be collected, trained affordance models are also highly versatile in application. The predicted outputs of affordance models, most commonly contact points and interaction trajectories, can be used for a variety of robot learning paradigms, including offline data collection, goal-conditioned learning, task-agnostic exploration, and action space parameterization [8]. Other works in the goal-conditioned learning space study the application of affordances for zero-shot manipulation [12] [44]. This work primarily studies a subset of goal-conditioned learning, where the

agent leverages affordance information to learn tasks using its own experience. Another important application, not studied in-depth in this thesis but remains highly relevant, is the use of affordances for autonomous, intelligent exploration. Exploration is a unique problem space, almost opposite from goal-conditioned learning as it is task-agnostic. The challenge for exploration, therefore, is designing efficient exploration objectives, given the lack of explicitly specified tasks or goals to condition policies on. In the absence of an extrinsic reward via goals, it is necessary to have intrinsic rewards generated by the agent for intelligent exploration [64]. A variety of objectives have been proposed in the literature: state novelty [11], prediction error [65] [74], model disagreement [66], and uncertainty maximization [35] [52].

We turn our focus to Mendonca et al. [2023] as a case study in particular, as ALAN is one of the more recent examples of implementing curiosity-based models on a physical robot. The approach encourages efficient autonomous exploration using the intrinsic objective of maximizing the uncertainty of predicted environment change, attempting to capture changes in the object space while ignoring changes in robot position. They train their policy to choose actions that maximize the changes in visual features of the observation space, masking out the robot so as to focus on environmental changes. However, it is worth considering whether removing the agent almost entirely from the intrinsic reward function is the best exploration objective. Grounding the intrinsic reward solely in observation space is a sparse reward; if the robot gripper does not make any changes to the environment in the duration of an episode, it will get stuck and struggle to even start the learning process as the model is consistently getting zero rewards. The empirical result of this is that in ALAN, the robot gripper must be positioned very close to objects of interest so that meaningful interactions can occur, thereby enabling the learning process to start.

The notion of affordance learning inherently accounts for the agent being incorporated into this notion of exploration reward by tracking hand interactions. We hypothesize that leveraging affordances for intelligent exploration can be an effective way to densify the exploration reward and make it more agent-centric rather than solely object-centric, by defining a reward function in terms of what the agent should do with objects around it like in Nagarajan and Grauman [2020]. Affordance models can also be used to sample potential tasks to collect coherent exploration data, which can then be used for goal-conditioned learning, framing affordances as goals rather than actions [40]. By using object localization, contact point segmentation, and trajectory prediction, affordance models can localize objects and human hands, and narrow down the space of exploration to determine good robot end-effector trajectories for intelligent interaction, creating a denser exploration reward signal. For the broader goal of developing scalable robot learning frameworks, it is extremely useful to drop an agent into a novel environment with novel objects and have it explore intelligently with little test-time supervision, as such systems can autonomously collect data in new environments via intelligent exploration, which can be used for adaptation to later complete downstream tasks.

Given an overview of the research landscape in affordance learning, there remain several open questions. Much of the literature revolves around learning and leveraging visual affordances in the form of contact points, interaction heatmaps, hand trajectories, and so forth. There is substantially less work in understanding skill affordances, focusing on the question of how to select useful actions. A key problem in robot learning is that it is difficult to learn solely by trial-and-error in such a large action space: in the reinforcement learning paradigm, should a robotic agent learn entirely from scratch, one would observe a long period of the robot trying highly nonsensical actions before converging on anything remotely sensible, if it even converges at all. The reason why babies and infants don't do this is because of the abundance of structural priors that bias them towards useful actions they have seen being performed, which narrows the action space. Ideally, embodied agents can learn to select a only a subset of skills that are relevant for the task and narrow down the action space, which expedites task completion, and priors from human interaction data can be useful here. It seems plausible that affordances can be used to reduce action space sizes for robots via action parameterization, but the exact implementation of this on a physical robot remains an open question.

Another open question is what data to use for training affordance models. Most works use large, diverse, Internet-scale datasets [8] [58] [12], though some works focus on imitation learning using datasets collected in similar environments [86]. Using large pretrained models such as large language models (LLMs) or vision-language models (VLMs) that are trained on Internet-scale data are also another way to indirectly leverage diverse data, as we will explore in Section 1.2. The question remains of whether large, diverse datasets are sufficient for learning robust, environment-agnostic affordance representations that can be used zero-shot for robotic manipulation tasks, or pure imitation learning from data collected in the same domain is more efficient for robot learning. The two-phase approach of pretraining and online finetuning we propose combines both dataset types for learning to complete robotic manipulation tasks, leveraging useful but general priors from diverse datasets as well as data collected in the same domain but limited by the quantity and diversity of in-domain data in a complementary fashion. We find that using pretraining on a large dataset, namely the Bridge dataset [85], and leveraging VLMs for affordance priors can speed up online finetuning where the agent collects and learns from data in its specific environment and domain.

1.2 Foundation Models for Robotics

The rapid development of foundation models [26] in recent years has drawn significant attention both in the academic community and beyond. The most rapidly developing foundation models recently have been LLMs, such as the BERT [21] [47], Claude [4] [5] [6], LLaMA [82] [28] [2], and GPT model families [67] [68] [15] [63]. More recently, we are seeing similarly exciting developments in text-to-image CLIP-based models [69] such as the DALL-E model family [70] [71], vision-language models [93] [56] [89] [46] [72] [45], and multimodal foundation models [62]. This surge of interest

in foundation models has arisen because they demonstrate that models trained on broad, Internet-scale data are highly adaptable to a wide range of downstream tasks. This is evidenced in the huge breadth of applications that have been developed such as ChatGPT [61], the fastest growing application of all time, and numerous applications wrapped around foundation models, from code generation [88] [42] to image generation [54] from text prompts.

Robotics is a specific downstream task of foundation models that has garnered a lot of interest in the academic community. Ahn et al. [2022] is a widely known pioneering work in this space, with the aim of extracting the "common-sense" knowledge about everyday tasks encoded into LLMs for physically-grounded tasks, to broaden the set of tasks robots can plan and execute abstract, temporally-extended textual instructions. The system consists of two main algorithmic components (Figure 1.3): the Pathways Language Model (PaLM) determines useful actions to accomplish a goal by calculating probability that each skill aids in completing the high-level task instruction, and a set of affordance-based value functions calculates the probability that each skill will succeed given the current state. The overall algorithm jointly determines the probability that each skill will perform the instruction successfully given the current state of the environment, and the most appropriate skill is selected and the corresponding policy trained via behavior cloning or reinforcement learning is executed. The experiments validate that SayCan can complete temporally-extended, complex tasks from abstract instructions, and adapts selected behaviors to each environment and embodiment with successful implementation on several robots with different action spaces.

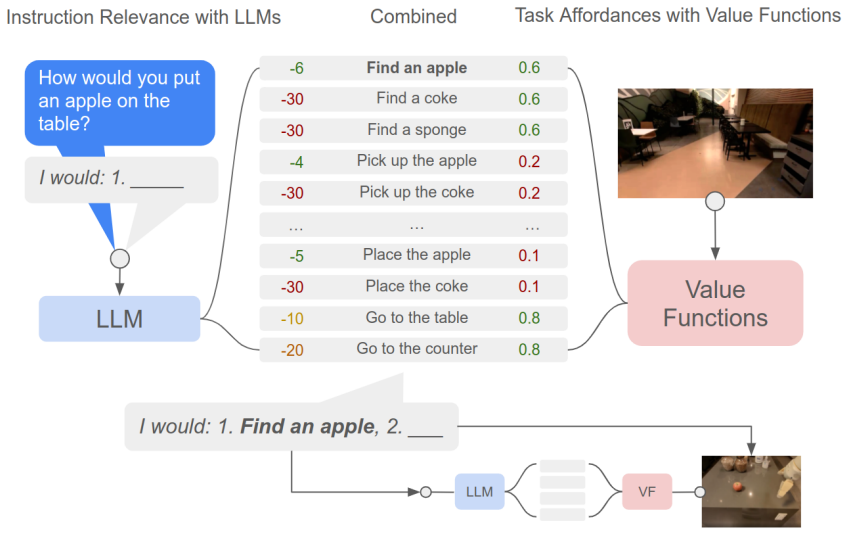


Figure 1.3: Algorithmic components of SayCan: an LLM calculates the probability that a skill is useful for the instruction and the value function module calculates the probability of successfully executing eac skill to select the skill to perform. [1]

Ahn et al. [2022] demonstrates encouraging results in training language-conditioned robotic control policies. Such approaches allow us to leverage and ground the rich knowledge and reasoning capabilities of LLMs to enable embodied agents to complete long-horizon tasks. It also opens up a new perspective on the pretraining process for robot learning: could we use LLMs as a pretraining mechanism for robot policies? Previous approaches covered in Section 1 that involve learning human videos require training models from scratch on diverse Internet-scale datasets, or from human teleoperated robot data. While the priors derived from training such affordance models are rich, the training process is costly and also requires specific heuristics to extract the most meaningful information relevant to robotics tasks. LLMs, on the other hand, encode highly flexible and general knowledge, and querying an LLM could be less costly than training affordance models from scratch, providing an alternative form of capturing information from diverse, offline datasets.

While the results from Ahn et al. [2022] seem promising, there is a critical engineering risk that the reasoning abilities and representations captured by LLMs are overly general for embodied tasks, and more work is needed to properly ground the high-level plans generated by LLMs in low-level actions for embodied tasks. In the case of SayCan, grounding was achieved through the affordance-based value function module, which connected LLM-generated high-level plans with clearly defined low-level action primitives, each with a pretrained policy to be executed on the robot. However, assuming access to robust pretrained skill policies for a specific embodiment is not necessarily scalable, and policy transfer between different robot embodiments is still an open area of research [17].

Converting visual observations into language descriptions and planning in solely in the language space loses a lot of rich information critical to scene understanding, which is a major limitation of using LLMs for spatial planning, reasoning, and task completion. Notably, in Du et al. [2023], the ELLM system generated inaccurate responses to whether objects matched the goal positions when tasked with rearranging objects in a household environment to match the goal arrangement. It is important to pay attention to the pitfalls in the household environment despite the successes in the open-world Crafter environment. The goals for survival in an open-world environment (such as `build house`, or `acquire food`) are fairly general and transferable, and LLMs have likely encountered such scenarios during training and can suggest reasonable goals. However, the general knowledge encoded in LLMs may not necessarily as beneficial for robot learning: LLMs can provide general priors for planning and reasoning, but this generality also results in reduced specificity to the environment that the robot is operating in, thus the general priors may not be as helpful without additional grounding. Planning solely with language thus loses a lot of information associated with the richness of the visual modality that is critical to most robotics applications.

The clear advantage of language is the natural interface for providing task instructions and describing goals. That said, much of robotics research relies heavily on computer vision techniques to accurately perceive and interact with the environment. Thus, grounding LLMs with visual input seems to be the natural course of action for robotics applications. Several state-of-the-art VLMs, such

as OWL-ViT [56], ODISE [89], GroundingDINO [46], and GroundingSAM [72], demonstrate highly generalizable open-vocabulary object localization. Stone et al. [2023] demonstrates a successful application of VLMs for robotic manipulation by interfacing policy learning with OWL-ViT for open-vocabulary object manipulation (Figure 1.4). The VLM grounds natural language to objects as the language instruction and current image observation is passed to OWL-ViT to localize the relevant objects with bounding boxes. The extracted object position, along with task and skill embeddings, are then passed to a model based on RT-1 [14], which trains a language-conditioned policy is trained on a set of demonstrations involving diverse skills and objects, allowing generalization to new object categories and descriptors. Therefore, using VLMs for grounding seems promising as the resultant policy is able take the current observation as input while also being conditioned on language.

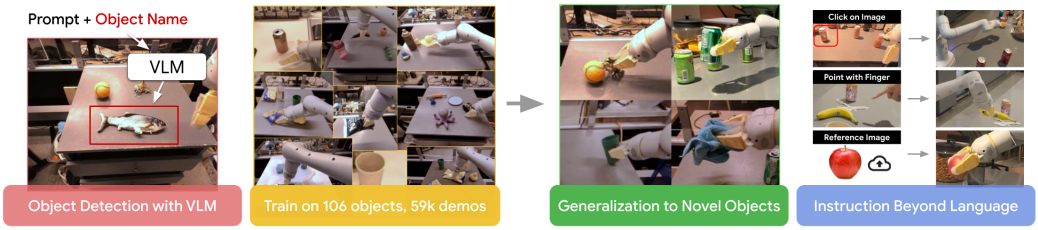


Figure 1.4: MOO trains a language-conditioned policy conditioned on object localizations from a frozen VLM. The VLM's object-centric representations enables the policy to generalize to novel objects and object localizations. [78]

While preliminary VLMs succeed in open-vocabulary object idenfication and segmentation, they still lack the extensive reasoning capabilities of LLMs, since reasoning over image inputs is significantly more complicated. There are also other limitations of using such VLMs in robot learning pipelines, for instance generalization to different objects is still dependent on diversity of robot data, and VLMs struggle with complex object descriptions involving spatial relations which is an important part of reasoning for robotic tasks [78]. Ultimately, preliminary VLMs require text queries where the objects involved are known *a priori* to generate bounding boxes around the requested objects, which are provided to language-conditioned policies or LLMs for downstream reasoning. This leaves open the question of whether it is possible to combine object localization and semantic reasoning into a single end-to-end model, rather than passing the annotated outputs from VLMs into a language-conditioned model. That is, can VLMs be prompted with an image and a task instruction, rather than a set of objects, and reason about the task from the image observation?

Modern VLMs, such as GPT-4V [93] and Gemini [30], have demonstrated promising capabilities in this direction. These VLMs can take extensive language prompts and images as inputs and reason over images beyond object localization and segmentation, combining the reasoning capabilities of LLMs with perception of the environment via visual inputs. A key advantage of using modern VLMs is that it simplifies the process of translating high-level plans into low-level robot actions. While previous works leveraging LLMs like SayCan [1] require pretrained skill policies for each action

primitive, using modern VLMs can circumvent the issue of selecting from a suite of pretrained action policies, by deriving rewards from image space that can be used for learning state-action mappings via reinforcement learning. This is because VLMs, unlike LLMs, can determine success or failure based on image observations, and this reward signal can be used to enable robots to learn via trial-and-error, without training skill policies via imitation learning which are costly and difficult to scale. VLMs can also guide the learning process by generating shaping rewards, in the form of intermediate waypoints the agent needs to move to on the way to completing the task.

Defining rewards in image space by using VLMs to determine task completion and specify intermediate waypoints as goals is a key contribution of our work. A major engineering challenge is tuning the inputs to these VLMs, which are highly expressive but also opaque, to derive useful reward signals for learning. Both language and image prompts require careful tuning to generate accurate and meaningful outputs, as we have found that modern VLMs still struggle to some degree with spatial reasoning. Our work explores combining preliminary VLMs and modern VLMs, as the outputs (bounding boxes or segmentations) of preliminary VLMs can serve as more helpful inputs to modern VLMs for semantic reasoning than raw image observations, thereby leveraging pretrained representations in VLMs as reward predictors.

1.3 Autonomous Reinforcement Learning

Online reinforcement learning (RL) is the paradigm by which agents gathers data through interaction with the environment, then stores this experience in a replay buffer and updates its policy. This contrasts with offline RL, where the agent updates its policy using previously collected data or human demonstrations, without itself interacting with the environment. A longstanding goal is autonomous RL: the potential of placing a robot in a real-world environment and it improves on its own by autonomously gathering in-domain experience, which holds great promise for scalable robot learning. Autonomous RL is the setting where the agent not only learns through its own experience, but does not require human supervision to reset the environment between trials.

Algorithms for autonomous RL have been difficult to implement in the real world, with the primary challenge being sample complexity, the number of calls to the model required to achieve acceptably good performance. In addition, there is the challenge of providing well-shaped rewards for online exploration, as well as the difficulty of continual reset-free training, which requires significant human effort. Several works have developed systems for reset-free training to reduce or eliminate human interventions in the online RL process [10] [92] [32] [79], but reward engineering is an open problem as manually specified reward signals are seen as difficult to engineer and easy to exploit.

Autonomous RL suffers when the reward signal is not informative enough, whether it be too sparse, time-varying or even absent. While it may be true that hand-designing reward functions is challenging, what if we could learn reward functions from previously collected data, or extract

rewards from large pretrained models? Some works have attempted to learn rewards from human feedback [81] [10], while acknowledging that these rewards are noisy and still require human intervention. The large bank of offline image and video datasets, as well as the high inference speed and accessibility of large pretrained models, could potentially offer solutions to further reduce or eliminate human intervention, while providing more precise and informative shaping rewards.

There has been attempts to leverage VLMs to generate rewards for online RL. Yang et al. [2023a] finetunes MiniGPT-4 [96] to generate sparse task completion rewards on a moderate number of in-domain demonstrations. However, these rewards are still sparse and pretraining the model requires a substantial number of in-domain demonstrations per task, which is not only costly but also makes the system more brittle and less robust to generalization. We also draw upon recent work that has extracted rewards from LLMs [48] [94] and VLMs [44] [84] [50] to guide zero-shot robotic manipulation. Instead of finetuning VLMs to define sparse task reward signals or extracting rewards from VLMs for the zero-shot manipulation setting, we look into leveraging affordance-based representations from VLMs to tackle the dense reward shaping problem.

1.4 Key Research Questions

In light of the extensive literature around affordance learning, foundation models for robot learning, and autonomous RL, the research presented in Chapters 2 and 3 explore the following questions:

1. How can we efficiently extract dense shaping rewards for online reinforcement learning?
2. Can pretrained object-centric representations from VLMs facilitate intelligent robotic interaction and object manipulation?
3. What are relative advantages of learning representations from diverse human datasets versus leveraging representations encoded in large pretrained models?
4. Which modality or combination of modalities is most effective for representing affordances for robot manipulation tasks?

Chapter 2

Learning from Diverse Human Videos

Teaching robots novel skills with human-in-the-loop expert demonstrations is costly. A much easier alternative to provide action-free visual data in the form of human videos, which we already have large amounts of from the web. Human interaction data is rich with priors about meaningful object interactions, sensible hand poses, and useful object-specific tasks and goals. Ideally, this data can guide embodied AI agents to learn to perform new tasks in novel environments, informing both what to do and how to do it. This research direction draws inspiration from works that try to learn different forms of useful information for robotic manipulation from diverse human videos: reward functions [16] [91] [49], interaction points [58], and policies [8]. In particular, the methodology detailed in this chapter is motivated by several concurrent trends:

1. prior work learning useful contact and interaction priors from human hand data [86] [12] [8]
2. hand-object detectors trained on Internet-scale data [75]
3. the EPIC-KITCHENS dataset [19], a large annotated egocentric human video dataset tracking human hand interactions with kitchen objects, accompanied by EPIC-FIELDS 3D geometry information [83] and VISOR pixel annotations [20]

This chapter aims to examine opportunities and challenges in learning from diverse human videos, specifically the approach of explicitly tracking hand and object segmentations and trajectories. We look at Bahl et al. [2023] in particular, and leverage the Vision-Robotics Bridge (VRB) approach of learning contact points and action vectors from human videos. In addition, we propose adding additional features to VRB: tracking hand and object trajectories, estimating 3D trajectories instead of 2D, and conditioning learned policies on language, to facilitate online RL.

2.1 Related Works

2.1.1 Leveraging Affordances for Manipulation Tasks

As explored in Section 1.1, prior work has demonstrated successful learning of interaction hotspots [58] and other visual affordances [57] from first- and third-person video datasets. These works demonstrate that object-centric representations learned from large, diverse datasets of human interaction data facilitated success on downstream robotic manipulation tasks, by transferring informative structural assumptions about object interactions. Bahl et al. [2023] also showed the structure of visual and behavioral affordances enables robots to perform many complex manipulation tasks, and are compatible with a range of robot learning paradigms including goal-conditioned learning and offline data collection. However, instead of leveraging affordances for zero-shot task transfer, this work will use them for online RL, where the agent must leverage affordance information to learn from experience. Affordance information will serve as useful structural priors to guide the agent towards useful behaviors that it can learn from.

2.1.2 Pretraining with Large, Diverse Human Datasets

Bahl et al. [2023] showed that learning directly from diverse, in-the-wild human videos, specifically the EPIC-KITCHENS egocentric video dataset, is successful in enabling policy transfer across several robot learning paradigms. In addition, Ma et al. [2023b] learned rewards implicitly via value functions via pretraining on unlabeled human videos, while Nair et al. [2022] used visual representations learned from egocentric videos showing humans performing manipulation tasks aligned with language annotations. Zakka et al. [2021] learns vision-based reward functions from offline videos of expert demonstrations to bridge the embodiment gap between humans and robots, leveraging temporal cycle-consistency constraints to learn deep visual embeddings that capture task progression.

These works demonstrate that pretraining with large, diverse datasets, particularly human or cross-embodiment datasets, facilitate learning useful skill and object representations and enable knowledge transfer to guide robot behaviors. Our work utilizes this understanding, while also considering the useful and natural interface of language for task specification, leveraging both video data from EPIC-KITCHENS as well as its accompanying VISOR language annotations. Instead of exploring zero-shot manipulation, we instead present an *explicit reward learning* method, which trains policies by optimizing learned rewards and using learned rewards to facilitate learning through experience via online RL.

2.1.3 Explicit Reward Learning

Prior works have addressed the problem of lack of shaping rewards using human data have learned implicit value functions [49] or visual representations or vision-based reward functions [59] [95] for

reward specification on downstream tasks, which can be broadly categorized as an implicit reward learning approach. Other works learn policies either directly from videos [8] or indirectly through latent plans [86], which we refer to as explicit and implicit policy transfer respectively. However, there have not been any proposed methods for *explicit reward learning*: transferring rewards learned directly from human datasets. To better contextualize where this work is situated relative to prior works, below is a table visualizing various previous investigations using the explicit vs. implicit paradigms and reward vs. policy learning paradigms:

	policy transfer	reward learning
explicit	VRB (Bahl, 2023)	Our Work
implicit	SPRINT (Zhang 2023), MimicPlay (Wang, 2023)	VIP (Ma, 2023), R3M (Nair, 2022), XIRL (Zakka, 2021)

To briefly summarize these previous investigations:

- Bahl et al. [2023] (VRB) is an *explicit policy transfer* method that learns actionable, agent-agnostic representations from egocentric videos by predicting contact points (as pixel heatmaps) and post-contact trajectories (as direction vectors) on objects for zero-shot policy execution.
- [Wang et al., 2023] (MimicPlay) is an *implicit policy transfer* method that learns latent plans from human play data for low-level visuomotor control trained on a small number of teleoperated demonstrations.
- [Ma et al., 2023b] (VIP) is the most closely related among the *implicit reward learning* methods. VIP learns implicit, goal-conditioned value functions self-supervised from egocentric human videos to learn effective visual representations, and performs zero-shot reward-specification on unseen downstream robot tasks.

Our work builds on the explicit hand tracking method from VRB and uses it to derive rewards, rather than policies or latent plans, to guide online RL, an approach that has not yet been rigorously explored in the current literature. In particular, we use outputs similar to VRB (contact points and interaction trajectories) as a *reward* instead of policy, and learn a robot policy for more challenging tasks by optimizing this reward instead of zero-shot policy transfer.

To motivate the focus on explicit reward learning, our hypothesis is that directly tracking information from human behavior in-the-wild can facilitate transfer of structural assumptions from human interaction data. Ma et al. [2023b] demonstrated that it is indeed possible to learn reward specification entirely from out-of-domain data. Examples of such assumptions would include, for instance, the most intuitive, safe, or efficient ways to interact with objects, or the most sensible contact points on objects, for instance the handle, rather than the blade, of a knife. Incorporating such

structural assumptions to guide robotic actions not only improves the sample efficiency of online RL, but also safety, by learning from unstructured human interaction data.

We hypothesize that incorporating such structural assumptions from large, diverse human datasets will enable more intelligent, sample-efficient, and safe exploration in novel environments. The broader goal of this work is to facilitate online RL accelerated by pretraining on human video data, as priors specific to human interaction data are used for reward computation. By learning shaping rewards for robot behavior from information directly tracked from human data, policies learned by optimizing these rewards could then perform more complex tasks more intelligently than zero-shot policy execution methods like Bahl et al. [2023]. Therefore, the key question of this research project is: can we perform explicit reward learning from unstructured human video data, and use these rewards to facilitate online RL?

2.2 Methodology

2.2.1 Approach

To extract visual affordances from large-scale human videos to learn policies, our proposed method is to train an affordance model to predict *contact points* on objects and *multi-step, post-contact hand and object trajectories*, and use them as reward guidance for online RL. This approach is inspired by Wang et al. [2023] and Bharadhwaj et al. [2023], which explicitly track hand pose and generate a 3D pose trajectory to learn from human videos. Contact point and post-contact hand trajectory prediction is also used in Bahl et al. [2023] for the purposes of policy transfer. However, we use affordance information for reward shaping rather than imitation learning or policy transfer, and we additionally track the motion of the object after contact.

The primary goal of learning from egocentric human videos to learn (a) relevant skills for a task, represented by hand trajectories and (b) image segmentation to highlight the relevant contact points for interaction. Then, we will implement an online RL policy using the generated outputs to learn to complete novel tasks given an initial image of the environment. An example of model outputs are visualized in Figure 2.1. We focus on training an affordance model using diverse in-the-wild human videos from the EPIC-KITCHENS dataset to predict the desired outputs, and then use these outputs as dense rewards for learning.

2.2.2 Data

For model pretraining, we use the EPIC-KITCHENS egocentric video dataset, the largest egocentric human video dataset to date that consists of egocentric audio-video recordings in native environments (i.e., the wearers' homes) capturing all daily activities in the kitchen over multiple days, using a head-mounted camera. The dataset spans 45 kitchens, 100 hours of full HD recording, 20M frames,

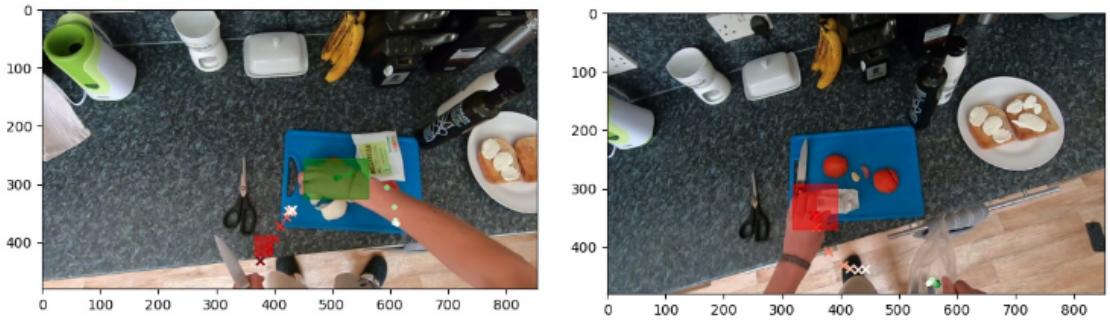
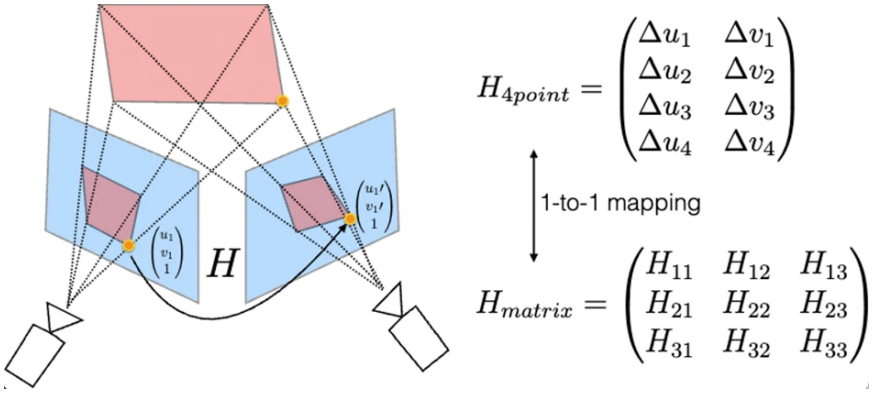


Figure 2.1: Example of model outputs (object contact points and post-contact trajectories) given an image and language annotation of a novel kitchen scene. Red points and bounding box correspond to left hand trajectory and contact points respectively, green points and box correspond to right hand trajectory and contact points respectively.

90K action segments, and 20K unique annotations. The large dataset must be preprocessed to extract the relevant information for affordance learning; specifically, the goal is to generate a dataset of post-contact hand and object trajectories as well as contact points. Left and right hand interactions are processed separately, and data samples are curated by extracting (a) hand trajectories of meaningful interactions with objects and (b) initial frames where neither of the hands are contacting any objects to backproject future frames onto. As the egocentric camera or head position of the user moves throughout the interaction, for each interaction trajectory, hand centroids in each frame must be projected onto nearest preceding initial (contactless) frame using stepwise homographies from initial frame to the corresponding frame in the trajectory. Homography matrices are estimated using sampled points from sequences of dense frames.



$$H_{4point} = \begin{pmatrix} \Delta u_1 & \Delta v_1 \\ \Delta u_2 & \Delta v_2 \\ \Delta u_3 & \Delta v_3 \\ \Delta u_4 & \Delta v_4 \end{pmatrix}$$

$$H_{matrix} = \begin{pmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{pmatrix}$$

Figure 2.2: Using homography matrices for projecting coordinates from a new viewpoint to an initial frame. [90]

Formally, for a trajectory of length T, $H_{T,1} = H_{T,(T-1)} \times H_{(T-1),(T-2)} \times ... \times H_{3,2} \times H_{2,1}$, where $H_{i,j}$ is the homography matrix projecting points in frame i to points in frame j. Therefore, $H_{i,(i-1)}$ is estimated since the frames in the egocentric videos are dense and there are many corresponding points in two subsequent images for an accurate estimation of the homography matrix. We can iteratively calculate $H_{i,1}$ for all i where $2 \leq i \leq T$ and use each $H_{i,1}$ to project the hand centroid in frame i to frame 1, the initial contactless frame, to generate trajectories that look similar to Figure 2.1. These preprocessing steps yield a dataset of hand and object trajectories for different interactions with kitchen objects. The hand and object trajectories are sequences of hand and object centroids, and the object contact points are represented by a segmentation mask. We determine the relevant contact points on objects using off-the-shelf hand detector models [75] and ground-truth segmentation data from EPIC-VISOR [20]. Contact points and post-contact trajectories are jointly used as supervision labels for training the affordance model for prediction on novel kitchen scenes.

The preliminary experiments in Section 2.3.2 are performed in simulation to verify the approach of dense shaping rewards does indeed contribute to more intelligent interactions with kitchen objects, before moving to real robot settings. These experiments use expert play data from D5RL, collected in the standard Franka Kitchen environment on a Franka Emika arm. Contact points and post-contact trajectories are extracted for reward shaping from exploratory interactions with different objects in the expert play data. The agent's trajectories are then optimized using Model Predictive Path Integral Control (MPPI), as done in Ma et al. [2023b], but using explicit reward formulations rather than implicit value functions.

2.2.3 Model Training & Evaluation

The affordance model is trained in a supervised learning fashion, where the inputs are an RGB image of the kitchen scene and a language annotation of the behavior(s) being conducted in the scene. The labels comprise of object contact points and the post-contact trajectory describing how to interact with the object for the trajectory beginning the input image, generated using the preprocessing steps described previously. The RGB image and language annotations are passed through separate image (ResNet-18 [33]) and language (RoBERTa [47]) encoders, and the corresponding encodings are concatenated. The concatenated embeddings are passed into FiLM-conditioned UNet [53] to generate an intermediate language-conditioned image encoding and a segmentation mask representing the object contact points. The FiLM-conditioned UNet has a similar architecture to the standard UNet [73], with additional FiLM layers after the batch normalization layer in each encoder. The intermediate language-conditioned encoding is then passed into a Transformer model to predict sequential trajectory outputs, using labels as ground truth for supervised learning.

At test time, given a novel static image from egocentric EPIC-KITCHENS videos, with the same parameters as the initial backprojection frames (i.e., neither hand is in contact with any objects), the model predicts meaningful contact points on the object as a segmentation mask and post-contact

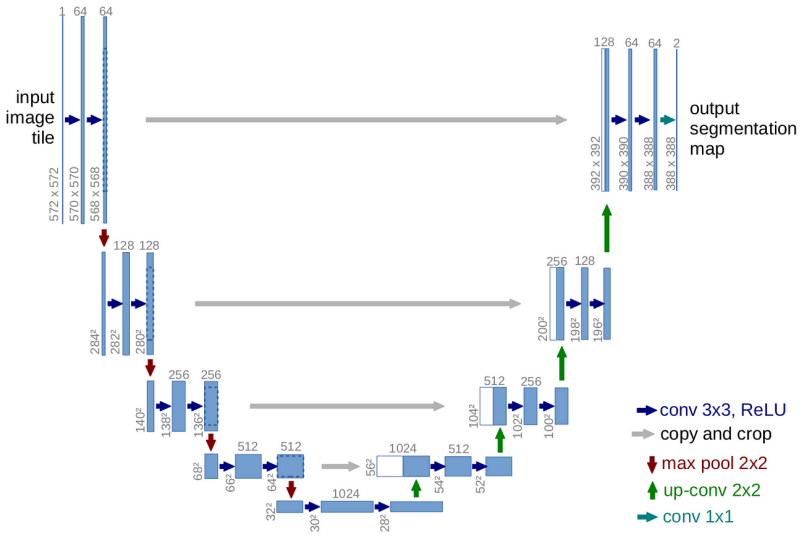


Figure 2.3: Standard UNet architecture. [73]

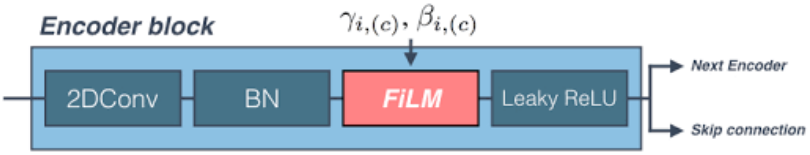


Figure 2.4: FiLM layers are placed after the batch normalization in eacdh encoder block. The output of a encoding block is connected to both, the next encoding block and the equivalent layer in the decoder via skip connections. [53]

trajectories for the hand and object as sequences of centroids for both the hand and the object. When used on images of the real robot setup, the agent is then rewarded based on how closely it follows the predicted trajectories.

The labels extracted from the EPIC-KITCHENS dataset consist of hand and object trajectories are sequences of hand and object centroids, and the object contact points are represented by a segmentation mask. Therefore, the trained affordance model is evaluated using the following metrics:

- Pixelwise accuracy between predicted segmentation mask and ground-truth contact point segmentation mask
- L2 losses between ground-truth and predicted trajectory for hands and objects, padding sequences to the same length where necessary

Since there are no other models producing model outputs in the same format (object contact points and post-contact hand/object trajectories), the performance of the affordance model will be benchmarked against the primary baseline of VRB [8], where we hypothesize that more fine-grained

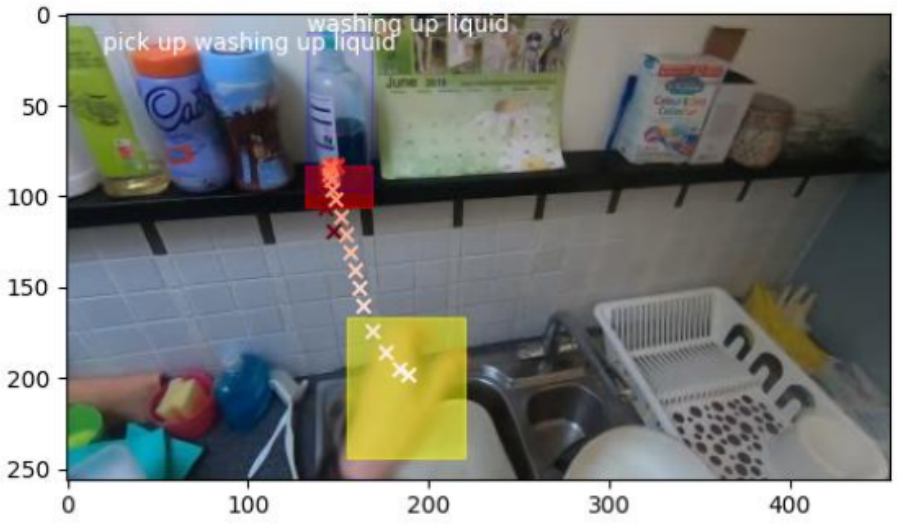


Figure 2.5: Example prediction from the model overlayed on the input image (language annotations for visualization purposes only: task instruction is used as input to language-conditioned model).

outputs from our affordance model would yield better exploratory behaviors.

For the preliminary experiments conducted in simulation in the Franka Kitchen environment (Section 2.3.2), the trajectories generated will be evaluated using four metrics:

- Last state similarity of robot end effector
- Last state similarity of target object in the interaction
- Cumulative L2 losses between the robot end effector in the ground-truth and actual trajectory over all timesteps
- Cumulative L2 losses between the target object in the ground-truth and actual trajectory over all timesteps

2.3 Experiments & Results

2.3.1 Training Affordance Model

Following the heuristics outlined in Bahl et al. [2023], we generated a preprocessed dataset of 80K meaningful trajectories after deduplication for model training, including multi-step, post-contact hand trajectories and contact points represented by segmentation masks, as in Figure 2.1. This is a relatively small dataset size for a supervised learning regime, as each trajectory consists of multiple timesteps unlike VRB, which generated a direction vector per timestep, causing the dataset to be

much smaller than anticipated. We trained the model on the dataset to test the suitability of the model architecture, and qualitatively the outputs look sensible. An example predicted interaction can be seen in Figure 2.5.

2.3.2 Simulation Tests for Dense Reward Shaping

The simulation experiments in Franka Kitchen serve to verify the approach of dense shaping rewards indeed contributes to more intelligent interactions with kitchen objects, before moving to real robot settings. Using the expert play data from D5RL, collected in the standard Franka Kitchen environment, we test several reward formulations, optimized using the MPPI trajectory optimizer. For all the dense reward formulations below (exclusing the baseline, which is a sparse reward formulation), reward is calculated at each timestep R^t, and the total trajectory reward $R = \sum_{t=1}^{T} R^t$ where T is the trajectory length.

- *Baseline (Sparse)*: Task completion reward R_{task} that is set to 1 if the object is within a certain threshold of the goal position from the expert trajectory and 0 otherwise.
- *Naive reward formulations*
  - Robot trajectory reward, R_{robot}:

    $$R^t = R_{\text{robot}} = ||o_{\text{robot}} - gt_{\text{robot}}||_2$$

    which is the negative L2 norm between robot end effector position at the current timestep (o_{robot}) and closest end effector position in D5RL expert trajectory (gt_{robot}). Note that finding the closest end effector position in the trajectory is preferred over exact timestep matching due to lags in robot trajectory compared to expert.
  - Combined trajectory reward of robot and object,

    $$R^t = R_{\text{combined}} = (1-p)R_{\text{robot}} + pR_{\text{object}}$$

    where R_{robot} as defined above added to negative L2 norm between current object position and corresponding position at timestep for R_{robot}, cumulative over all timesteps in the trajectory. p is a hyperparameter that is tuned.
  - Combined trajectory with added sparse task reward,

    $$R^t = R_{\text{combined}} + 10 * R_{\text{task}}$$

    which adds a sparse reward bonus to incentivize reaching the goal faster.

- *Conditional reward formulation*: Reward at timestep t, R^t is set depending on if contact made has been made at or before timestep t:

$$R^t = \begin{cases} R_{\text{robot}} & \text{if no contact yet} \\ R_{\text{task}} + (1-p)R_{\text{robot}} + pR_{\text{object}} & \text{otherwise} \end{cases}$$

  The agent is therefore incentivized to closely follow the expert's pre-contact trajectory to reach the object, and then to follow the trajectory of both the robot and the object in the post-contact trajectory with variable weighting on the two terms. Similarly to before, the sparse reward bonus incentivizes reaching the goal faster.

- *Adaptive reward formulation*: Reward at timestep t, R^t dynamically changes the weighting on each of the reward terms as time progresses,

$$R^t = R_{\text{task}} + \gamma^t * R_{\text{robot}} + \gamma^{T-t} * R_{\text{object}}$$

  where γ is a discount factor, t is the current timestep, and T is the total trajectory length. The goal of this dynamic reward formulation is to upweight R_{robot} in the initial phase of the trajectory, before contact with the object is made, and then transitioning to upweight R_{object} once contact has been made to facilitate meaningful object interaction. It can be viewed as a more elegant formulation of the previous conditional reward, but does not explicitly check the condition of whether contact with the object has been made, and attempts to implicitly ensure the agent learns this by upweighting R_{object} later in the trajectory.

The results for the task of `move the kettle to the top burner` are shown in Table 2.1. We see from the quantitative results that using a conditional reward formulation with a 0.75 weighting on the object position yielded the best quantitative results across all four evaluation metrics. Specifically, at each timestep, the reward is set to R_{robot} if no contact has been made before that timestep, and the reward is set to $R_{\text{task}} + 0.25 * R_{\text{robot}} + 0.75 * R_{\text{object}}$ if contact made has been made at or before

Reward Function	Trajectory reward	Last state similarity (robot)	Last state similarity (obj.)	Cumulative L2 losses (robot)	Cumulative L2 losses (object)
R_{robot}	-12.256933	0.398392	0.416386	21.440533	15.459495
$R_{\text{robot}} + R_{\text{object}}$	-12.222351	0.530047	0.416386	31.933604	15.459495
$R_{\text{combined}} + 10R_{\text{task}}$	-12.412558	0.289948	0.330250	27.716251	10.526182
Cond. $p = 0.25$	-17.913206	0.350854	0.330773	20.797026	12.850192
Cond. $p = 0.5$	413.380460	0.215454	0.194178	17.906603	10.637828
Cond. $p = 0.75$	**427.145726**	**0.146718**	**0.171739**	**15.970397**	**9.706456**
Adaptive, $\gamma = 0.9$	418.904287	0.180724	0.175628	17.217086	10.573261

Table 2.1: Results of experiments in simulated D5RL environment.

that timestep. The adaptive reward formulation also does comparably well across all metrics, and its performance could be improved with additional hyperparameter tuning, namely the discount factor.

Qualitative analysis of the visualized robot trajectory also confirms that the robot behavior is qualitatively most similar to the expert demonstrations. The sparse reward formulation resulted in no progress towards the task, because there were no reward terms focused on the trajectory or progress towards the goal. The naïve reward formulation tracking only robot reward follows the robot trajectory decently, but disregards whether actual contact was made with the object and misses the object entirely. An alternative naïve reward formulation combining the robot and object reward struggled to even reach the object, because the negative reward from R_{object} during the reaching phase made it difficult to learn how to approach the object. Thus, reward formulations distinguishing the pre-contact and post-contact trajectory by emphasizing R_{robot} initially then transitioned to upweighting R_{object} closer to task completion were the most successful.

(a) Baseline (object sparse reward only): no meaningful contact made.

(b) Naive reward formulation (robot reward only): contact missed, no progress to goal.

(c) Naive reward formulation (combined robot and object reward): no progress to the goal.

(d) Conditional reward formulation: meaningful post-contact trajectory.

Figure 2.6: Qualitative behavior in D5RL simulation using different reward formulations.

Overall, the trends in quantitative metrics in Table 2.1 and qualitative analyses of the robot trajectories in Figure 2.6 suggest that dense shaping rewards facilitate more intelligent interactions with kitchen objects, which validates the experimental hypothesis.

2.3.3 Limitations of VRB

While training an affordance model from scratch and leveraging its predicted outputs for dense reward shaping, one might wonder why utilizing the Vision-Robotics Bridge (VRB) [8] which inspired much of our method is not sufficient, given that VRB is a flexible approach amenable to several robot learning paradigms. A potential alternative method is to simply leverage the model outputs from VRB and use those as the dense reward shaping objective for online RL. We argue that the current instantiation of VRB does not generate outputs that are conducive for robust and intelligent exploration that could yield high-quality in-domain data for downstream tasks.

VRB outputs object contact points as a pixel-wise heatmap laid over the original image and per-timestep direction vectors indicating the ideal direction of movement of the robot arm. Inspection of the codebase reveals that VRB uses LangSAM to generate bounding boxes around objects in the scene, and from there generates direction vectors and contact points on the cropped image of just the object. Preliminary experiments testing the model outputs of VRB on novel kitchen scenes taken from real-world kitchens are shown in Figure 2.7.

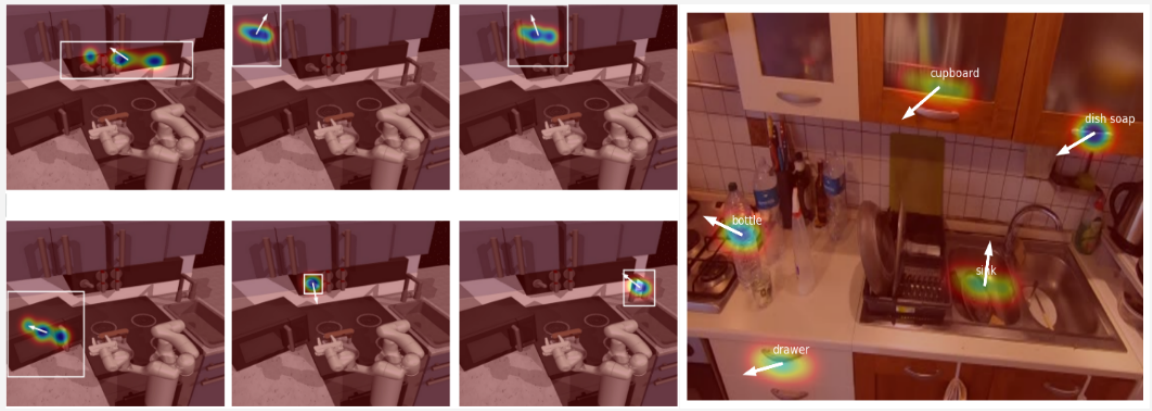


Figure 2.7: VRB model outputs on novel kitchen scenes, simulated (left) and real (right).

As seen in Figure 2.7, contact points are mostly predicted in the middle of the object bounding box generated by LangSAM. This might work sufficiently well for simple actions like pushing and pulling objects, but certainly will not work for interacting with constrained objects, for instance opening cabinet doors where the door handle is not in the center, or lifting pots and pans with off-center handles. The rightmost image in 2.7 predicts contact points that are off-center for hinged cabinets and dishwashers, for instance. In addition, the direction vectors are surprisingly random, and comparing direction vectors of several sequential image frames does not generate a continuous action. For instance, based on qualitative analysis of VRB outputs on continuous sequences of images such as the images on the left in 2.7 for the task `open the cupboard door`, both the contact points and direction vectors do not form a coherent action over sequential timesteps. Given the limitations

described, it is unclear how applicable VRB is off-the-shelf, or even with fine-tuning, for good dense rewards for online RL. One might hypothesize that the LangSAM object detection model is doing a significant amount of the work, and VRB could incorporate more temporal and semantic information to improve its predictions. Our work aims to close this gap by predicting multi-timestep trajectories and learn more semantically-informed contact points.

2.4 Discussion & Analysis

The primary finding of this research is validating the hypothesis of *explicit reward learning* for learning through experience, wherein rewards learned directly from human datasets are transferred to robotic systems for online RL. We hypothesized is that directly tracking information and incorporating structural assumptions learned from human interaction data in the form of shaping rewards, will enable more intelligent, sample-efficient online RL in novel environments.

Our simulated experiments verified that the explicit reward learning approach successfully facilitates intelligent interaction with kitchen objects in novel environments. In particular, reward formulations that prioritize following the pre-contact robot trajectory to reach the object, and then prioritize following the post-contact object trajectory, were successful in guiding embodied agents to interact intelligently with objects in a novel kitchen scene. This can be a conditional reward formulation, which explicitly checks whether contact with a target object has been made and changes the reward computation accordingly, or an adaptive reward formulation, which changes the weighting on reward terms for the robot trajectory and the object trajectory over time. However, in spite of these positive developments in simulated experiments, the limitations of VRB were significant enough for us to reconsider our approach, since much of our approach was premised on its success. Based on the shortcomings of VRB when tested on unseen kitchen environments, on top of the fact that curating the dataset based on our heuristic of multi-step time trajectories yielded a small dataset, it is unclear whether pursuing this direction and engineering new features, such as 3D hand pose tracking or dense object tracking, will serve as sufficiently shaped reward signals for online RL in real world robot setups.

Preliminary tests in the real world demonstrated difficulties in transferring to novel robot learning setups, and required a number of in-domain demonstrations to facilitate transfer. However, there are several works demonstrating the possibility of performing imitation learning on a modest number of demonstrations [3] [27] [22] [86], which undermines the argument for leveraging assumptions from human video data as either rewards or policies could instead be learned from in-domain interaction data. The goal of learning dense rewards is to significantly reduce or eliminate in-domain demos by enabling the agent to learn online from its own experience. Therefore, in light of the challenges in dataset size for the affordance model, the limitations of VRB, and difficulties transferring to the real world, we reconsidered how to extract meaningful trajectories to shape rewards for online RL.

Learning from diverse, unstructured human videos is a challenging but promising field with significant active, ongoing research [91] [13] [9]. However, having verified the hypothesis of the usefulness of explicit dense rewards for online learning, we explored other methods besides diverse human videos which are challenging to extract structured data from. While human video data is widely available, diverse, and encodes rich human priors, extracting these in the form of rewards has several engineering challenges. On the other hand, rather than pretraining on large human video datasets, foundation models similarly encodes general priors from highly diverse, Internet-scale datasets. Using large pretrained models such as LLMs and VLMs could therefore be another way to indirectly leverage diverse data. The rapid rise in accessibility and robustness of foundation models in the course of this research led us to pursue this direction: are the priors and representations encoded in large pretrained models useful for embodied learning?

Chapter 3

Extracting Shaping Rewards from VLMs for Robot Learning

Recent advances in LLMs and VLMs show promising results in using "common-sense" understanding to plan and reason [63] [18] [1] [69] [41]. However, even state-of-the-art large pretrained models still struggle with understanding interactions and physical dynamics in 3D space, which is essential to robotic control. Determining how to ground such models in the specific embodiment and environment dynamics is also a significant engineering challenge. In order to leverage the fast-paced and robust development of vision and language reasoning capabilities for robotics, we must address the open question of how to tune these models to generate outputs that can guide a robotic system to effectively and intelligently interact with the physical world.

Several prior works have utilized large pretrained models for robotic control, either through few-shot prompting or finetuning of large models to generate plans [1] [37] [36], code [42] [43], and rewards [50] [94]. In particular, the works closest to ours in the literature leverage VLMs to generate rewards for robotic control. However, many of these works generate sparse rewards [92] and focus on the zero-shot manipulation setting [84] [50]. No works thus far have explored the idea of leveraging VLM-derived rewards as shaping rewards for online RL, which has great potential for autonomous robot learning and reducing reliance on human teleoperation or demonstration collection. We explore leveraging affordance-based representations from VLMs to tackle the dense reward shaping problem.

This chapter presents a method for open-vocabulary visual prompting to extract rewards from VLMs for online RL. We leverage the insights about effective visual prompting methods from Liu et al. [2024] to develop a pipeline for generating dense waypoint trajectories from which dense shaping rewards can be calculated. We integrate this dense reward computation framework into Yang et al. [2023a], which has an established autonomous RL pipeline but with sparse rewards, to demonstrate improvement in success rates on a variety of complex object manipulation tasks.

3.1 Related Work

3.1.1 Reward Specification for Online RL

Online RL is the paradigm by which agents gathers data through interaction with the environment, then stores this experience in a replay buffer and updates its policy. A specific instantiation is autonomous RL [76], which enables robots to improve on their own by autonomously gathering in-domain experience. There are several key challenges for implementing autonomous RL in the real world, one of which is the challenge of providing well-shaped rewards for online exploration. Reward specification is an open problem for pretraining for robotic control, as manually specified reward signals are seen as difficult to engineer.

While hand-designing reward functions is challenging, the rapid development of large pretrained models opens up new potential methods to circumvent the traditional problems of reward engineering. Learning rewards from diverse human data, or doing so indirectly by extracting rewards from foundation models trained on Internet-scale data, presents new opportunities in this direction. We focus on Yang et al. [2023a] in particular, as a substantial part of our method demonstrating our explicit reward learning hypothesis demonstrated in Chapter 2 on a real robot system builds upon the autonomous RL infrastructure in RoboFuME.

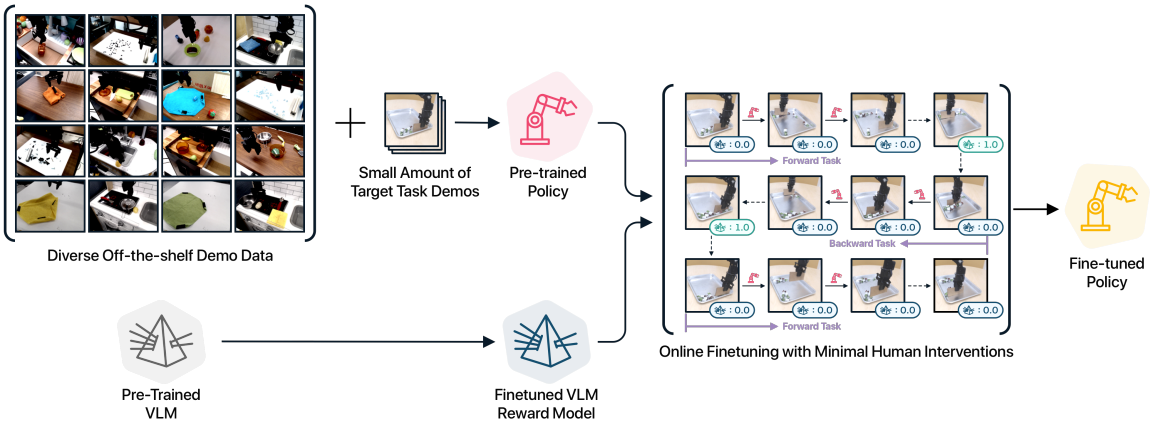


Figure 3.1: RoboFuME Pipeline: Pretraining policies offline on Internet data and a small amount of in-domain data, and finetuning online using sparse rewards from a VLM.

RoboFuME similarly identifies the challenges of environment resets and manual reward specification for online RL. To tackle the former challenge, RoboFuME is a reset-free fine-tuning system that pretrains a multi-task manipulation policy from diverse datasets of prior experiences and self-improves online to learn a target task with minimal human intervention (Figure 3.1). RoboFuME leverages the pretrain-finetune paradigm in robot learning, by pretraining policies on a multi-task manipulation policy from diverse datasets of prior experiences, specifically a subset of the Bridge

dataset [85], as well as 120 in-domain demonstrations per task (50 forward tasks, 50 backward tasks, and 20 failures). Pretraining on bridge data enables faster and easier learning of new tasks, with in-domain demonstrations to facilitate the transfer of skills. It then finetunes online using a sparse reward signal from a MiniGPT-4 task classifier finetuned on in-domain data [96] to learn a target task with resets only every 15-25 episodes. Overall, RoboFuME demonstrates that calibrated offline RL techniques can facilitate efficient online finetuning of pretrained policies in the presence of distribution shifts, and leverages VLMs finetuned on in-domain data to provide sparse reward signals for autonomous online finetuning. More broadly, it demonstrates the success of the pretrain-finetune paradigm for robot learning, allowing a robot to learn a new task with less human effort by leveraging Internet data and pretrained models.

In light of the rapid advances in the capabilities and accessibility of foundation models, we considered two alternative methods to further reduce human effort required in autonomous RL pipelines. First, replacing the sparse rewards generated from MiniGPT-4 finetuned on in-domain demos with zero-shot prompting of more powerful VLMs. The main tradeoff is latency, as larger models have longer inference times than MiniGPT-4; it remains an open question if latency is worth avoiding human data collection to finetune a task classifier. Second, instead of relying on in-domain demonstrations to facilitate task transfer, we can pretrain policies offline on Bridge data with much fewer or no in-domain demonstrations and generate dense shaping rewards from VLMs to guide online RL. Dense shaping rewards are formalized as waypoint trajectories that guide task completion in new environments, therefore leveraging skills learned from offline pretraining on Bridge data.

3.1.2 VLM-Generated Rewards

In Section 1.2, we discussed at length the potential of LLMs and VLMs for advancing robotic manipulation. Despite these encouraging results, however, even state-of-the-art LLMs and VLMs are subject to crucial limitations that prevent robotics from reaping the benefits of advances in vision and language models. Because advances in LLMs preceded VLMs, several of the previous works cited convert visual inputs into language descriptions, so that planning and reasoning can be performed by more advanced LLMs. This loses a lot of crucial visual information about the scene and physical dynamics that are essential for robotics tasks. Failing to properly ground LLMs in physical scenes will result in difficulties completing manipulation tasks in the real world.

Liu et al. [2024] proposes a system to enable reasoning in the visual space by providing language instructions and annotated image inputs to VLMs, thereby enabling robots to solve novel manipulation tasks through mark-based visual prompting. MOKA leverages point-based affordance representations that ground the reasoning capabilities of VLMs in image space in real-world robot interactions. Given a task instruction and visual observation of the scene, the model predicts a grasp keypoint on the object representing where to contact the object, a function keypoint on the object representing its functionality, and a target keypoint representing the goal position at the end

of the task, as well as intermediate waypoints to guide task completion. These points captures the essential information of the motion via a trajectory of waypoints needed for the robot to follow to complete the task, spanning several manipulation skills.

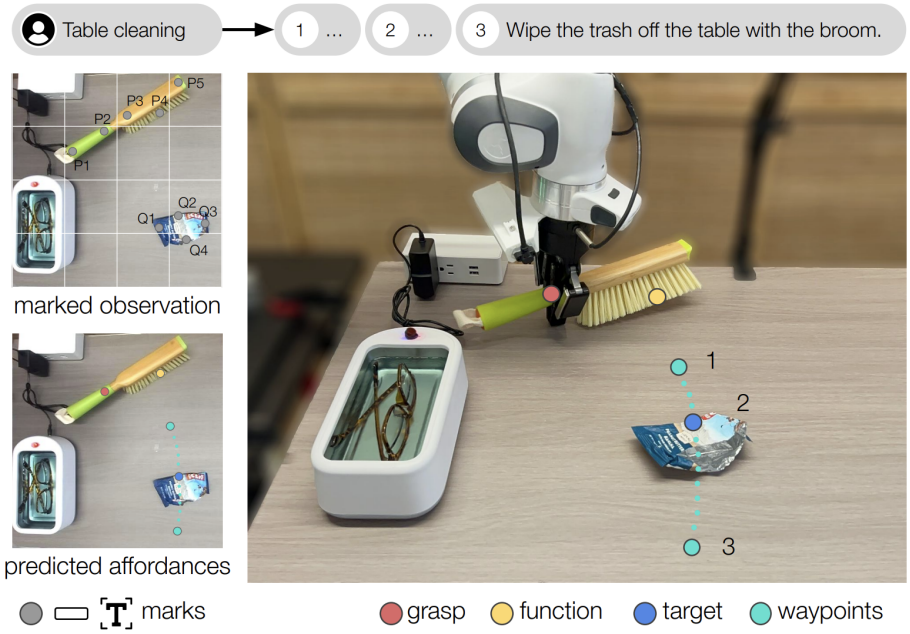


Figure 3.2: MOKA: GPT-4V-predicted point-based affordances to guide robotic manipulation.

MOKA implements a number of important preprocessing steps on the raw visual input to generate more accurate affordance predictions. First, MOKA overlays a grid on the raw image, and the waypoints are selected from grid point coordinates. In addition, MOKA uses GroundingSAM [72] to segment the objects of interest, and samples six keypoints on the object to grasp and six keypoints at the goal location. The grasp and function keypoints are selected from the first six sampled points on the object, and the target keypoint is selected from the latter six sampled points at the goal location. This encourages the VLM to attend only to the important parts of the visual observation, based on empirical experimentation suggesting that VLMs are much better at visual question-answering in a multiple-choice style. That is, VLMs are generally better at choosing a point among sampled points than predicting points via its own generation. Chain of thought prompting [87] and the order of generating predictions also influenced output accuracy. We leverage these insights in our approach.

Liu et al. [2024] demonstrates the strength of leveraging state-of-the-art VLMs for robotic manipulation via mark-based visual prompting. Besides generating shaping rewards by leveraging useful pretrained representations in foundation models, the incredible flexibility and generality of these models also facilitates open-vocabulary generalization, a major challenge for robotic systems. Foundation models encapsulate extensive prior knowledge from training on broad, diverse datasets.

They can therefore assist with solving tasks in novel environments and handling the large diversity and complexity of the physical world, all while using language commands as a natural interface for users. Both LLMs and VLMs have also demonstrated successful decomposition of high-level tasks into intermediate subtasks, which can be useful for solving complex, long horizon tasks. We leverage similar notions of affordance keypoints in our approach, though for generating shaping rewards for online RL rather than zero-shot manipulation.

While this may be said, there remain open questions for the usage of VLMs for robotic manipulation and learning. MOKA demonstrated that tuning the language instructions and meta-prompts are critical for model performance; as state-of-the-art models are generally opaque, this requires engineering effort in tuning these prompts. MOKA also provides on average 4 to 5 waypoints per task, which is likely not sufficiently dense for reward shaping in online RL, and it is an open question whether VLMs can generate dense but still highly accurate trajectories for reward shaping. MOKA uses coarse-grained depth information for its tabletop task, using the VLM to predict pre- and post-contact height as one of "above", "below", or "same". The importance of depth prediction for robotic manipulation [31] suggests that we may need to incorporate more fine-grained depth information, potentially by adding another camera angle besides a top-down camera angle, which loses critical depth information unless a depth camera or depth estimator is used.

3.2 Methodology

3.2.1 Approach

We consider problems that can be formulated as a Markov Decision Process (MDP) , described as a tuple $(\mathcal{S}, \mathcal{A}, \gamma, p, r, d_0)$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space and $\gamma \in (0, 1)$ is the discount factor, $r(s, a)$ is the reward function and $d_0(s)$ is the initial state distribution $d_0(s)$. The dynamics are governed by a transition function $p(s'|s, a)$. The goal of RL is then to maximize the expected sum of discounted rewards $\mathbb{E}_\pi[\sum_{t=1}^{\infty} \gamma^t r(s_t, a_t)]$. However, $r(s, a)$ is unknown to us. In sparse reward approaches such as RoboFuME, $r(s, a)$ is approximated only for a small subset $\mathcal{S}_{\text{sparse}} \subset \mathcal{S}$, making learning difficult because if the agent never reaches states in $\mathcal{S}_{\text{sparse}}$, it fails to get any meaningful reward signal for learning.

Our method approximates $r(s, a)$ by a combination of sparse task completion reward and additionally a dense shaping reward calculated with respect to a sequence of intermediate waypoints marking trajectory points towards the goal. This can be seen as breaking down a trajectory into short sub-trajectories or subtasks that are more easily reachable by the agent. This effectively expands the set of states for which a meaningful reward signal is received by the agent i.e. $\mathcal{S}_{\text{sparse}} \subset \mathcal{S}_{\text{sparse+dense}} \subset \mathcal{S}$ and therefore facilitates learning complex manipulation tasks.

We implement our explicit reward learning method by adding dense shaping rewards into RoboFuME's online RL pipeline. We pretrain a policy but only on a subset of Bridge data relevant to each

task, with a varying number of in-domain demonstrations. During the online finetuning procedure, at every episode of a forward or backward task, RoboFuME saves the entire episode rollout into a replay buffer, and then calculates a sparse reward per frame in the replay buffer. Instead, we process the episode replay buffer in two steps (Figure 3.3):

1. *Dense reward specification via VLM*: We take the first image observation in the episode x_0 and retrieve the top-down image x_0^d and the side view image x_0^s. We preprocess these images by (1) passing x_0^d through GroundingSAM to get segmentations of relevant objects and sample six points on the grasped objects and six points at the target location, and (2) overlaying a grid on x_0^d with the twelve labeled keypoints to get $(x_0^d)'$ and a series of evenly-spaced labeled horizontal lines on x_0^s to get $(x_0^s)'$. We pass $(x_0^d)'$ and $(x_0^s)'$, together with a language instruction and metaprompt (Appendix A.2) to GPT-4V, which generates several outputs but most important among them is `block_sequence`. This sequence is a list of tuples $([xy], [z])$, where $[xy]$ is a grid point chosen from $(x_0^d)'$ representing a position in the x-y plane from top-down and $[z]$ is a line chosen from $(x_0^s)'$ representing a position in the z-axis from the side view. The dense rewards are calculated with respect to `block_sequence` in the next step.

2. *Reward computation per frame in replay buffer*: For each frame in the episode replay buffer, we compute the positions of the robot and object in image space. We use a fitted RANSAC regressor to compute the robot position $(x_{\text{rob}}, y_{\text{rob}})$, and an off-the-shelf pixel tracker [39] to track a specific point on the object $(x_{\text{obj}}, y_{\text{obj}})$. Using these coordinates, we locate them in 3D space with both the top-down grid and side view height lines, and compute the nearest block to each of the robot and object position in `block_sequence`, B^t_{rob} and B^t_{obj} respectively. Leveraging the insights from Section 2.3.2, we compute a conditional reward based on the robot and object positions, where R_{robot} is the negative L2 distance from the robot to the *block after the closest block in* `block_sequence`, B^{t+1}_{rob}, and correspondingly for the object, B^{t+1}_{obj}. We use the next block to encourage progression towards the goal, instead of stagnating at the current position. This is the dense reward for every frame in the replay buffer, which we post-process through a modified sigmoid function to be between 0 and 1 (Section 3.3.3).
   Additionally, we query GPT-4V zero-shot with the current image observation and a metaprompt (Appendix A.1) to obtain the sparse task completion reward. We combine the dense and sparse reward (either 0 or 1) to get the total reward for the current timestep, with scaling such that the overall reward is between 0 and 1. We perform this reward computation for every frame in the episode replay buffer, which we save out and use to learn via online RL.

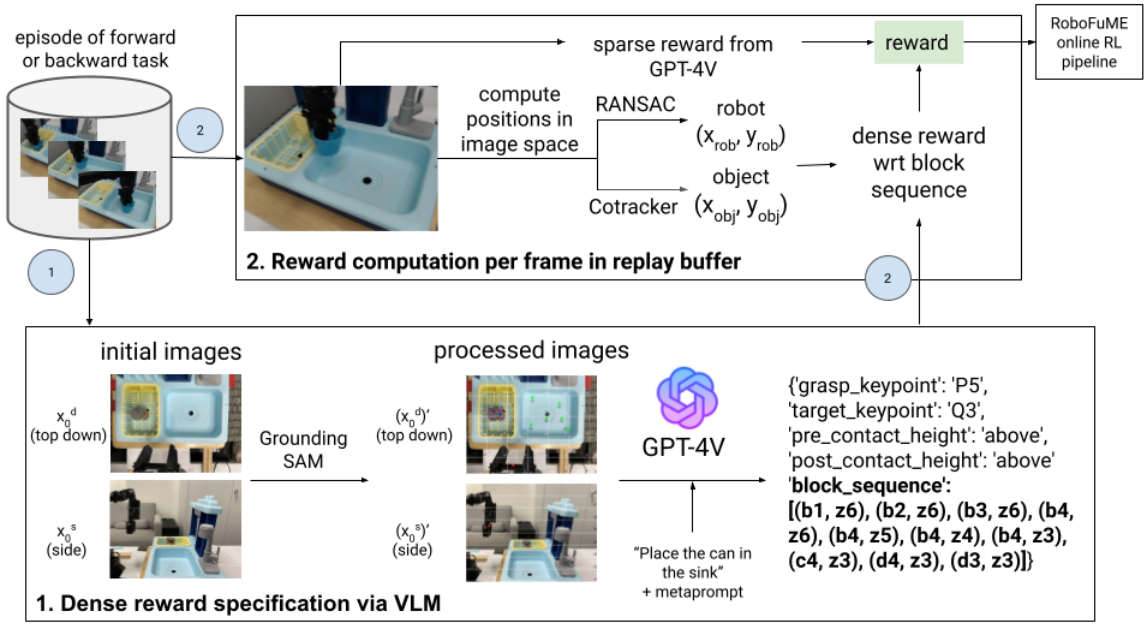


Figure 3.3: Our method consists of two components. The first (represented by the arrow (1) in the image) leverages GPT-4V to generate a sequence of waypoints in 3D space. The second (represented by the arrows (2) in the image) involves per timestep reward computation for each frame in the replay buffer, computing reward with respect to the waypoint sequence and a sparse reward derived from zero-shot VLM inference. The combined reward is used for online RL.

To summarize, the motivation for using dense shaping rewards generated by GPT-4V is to (1) replace the sparse task completion reward with a signal rewarding the agent for reaching intermediate waypoints thereby guiding task completion, and (2) reduce the reliance on in-domain demonstrations in the pipeline by getting fairly generalizable rewards from a large pretrained VLM. This approach is also highly flexible, as the VLM used in the pipeline can easily be replaced with a more advanced iteration of GPT or even other model families like Gemini [30] with little or no modifications to the metaprompt, based on preliminary experiments. Our approach uses GPT-4V as the generated waypoints were sufficiently accurate and robust to different tasks for our experiments.

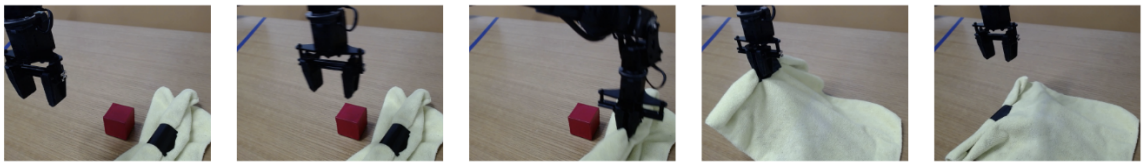
3.2.2 Model Training & Evaluation

There are two main sources of data used in our pipeline: the Bridge dataset [24] [85] as well as in-domain expert demonstrations. We used the following tasks: Cloth Folding, Cube Covering, and Spatula Pick-Place. The former two tasks were used in RoboFuME, and are mainly used to demonstrate successful reproduction of RoboFuME as a benchmark and for ablation tests to demonstrate the necessity of in-domain demonstrations for the pipeline. For Spatula Pick-Place, we demonstrate the challenges of reproducing RoboFuME on novel tasks and the use of dense shaping

(a) Cloth Folding forward trajectory.

(b) Cloth Folding backward trajectory.

Figure 3.4: Cloth Folding trajectories.



(a) Cube Covering forward trajectory.

(b) Cube Covering backward trajectory.

Figure 3.5: Cube Covering trajectories.

rewards to overcome this generalization problem and reduce reliance on in-domain demonstrations.

In RoboFuME, only subsets of the Bridge dataset were used to pretrain policies with language-conditioned BC and offline RL. For the two tasks also used in RoboFuME, we used the same Bridge data subsets, mostly featuring cloth-related tasks. For the novel task, we explore the selection of these subsets of Bridge data in Section 3.3.1. To ensure effective transfer learning from Bridge data, the camera angle and setup for the in-domain demonstrations were made highly similar to the camera angles used in the Bridge data. We evaluated pretrained checkpoints for goal-conditioned behavior cloning from goal images on some basic pick and place tasks from BridgeV2 [85] as a sanity check that the camera angle was sufficiently similar.

Visualizations of the in-domain demonstrations collected for the forward and backward task of each task category are included for Cloth Folding (Figure 3.4), Cube Covering (Figure 3.5), and

(a) Spatula Pick-Place forward trajectory.

(b) Spatula Pick-Place backward trajectory.

Figure 3.6: Spatula Pick-Place trajectories.

Spatula Pick-Place (Figure 3.6). Policies were pretrained using language-conditioned BC and offline RL on 50 forward task trajectories, 50 backward task trajectories, and 20 mixed-mode failures. The forward and backward task trajectories were collected with minimal multimodality and randomization, to facilitate reset-free RL. After pretraining on Bridge and in-domain data using offline RL, policies are finetuned online using CalQL [60] for 20K steps. While RoboFuME claimed to enable minimal resets (every 10-15 episodes), since reset-free learning was not an emphasis of our method, we reset the model every 2 episodes to maximize useful interactions and speed up learning time.

The pretrained models (with language-conditioned BC and offline RL) as well as the offline RL models finetuned online were evaluated on success out of 20 trials each of the forward and backward tasks in each task category. Success was evaluated qualitatively by similarity to the in-domain demonstrations collected: for Cloth Folding, the cloth had to be folded or unfolded to a degree similar to the expert demonstrations; for Cube Covering, the entire cube had to be covered or uncovered from the camera perspective shown in Figure 3.5; for Spatula Pick-Place, the spatula had to be on the yellow plate (forward task) or on the left side of the plate close to where it was picked originally (backward task).

3.3 Experiments & Results

3.3.1 Selecting Pretraining Datasets

Pretraining on the entire Bridge dataset would be computationally and practically infeasible. As such, choosing subsets of prior datasets to train on is crucial for downstream performance. We test this with a novel Spatula Pick-Place task that is not part of the list of RoboFuME tasks. We define the forward task to be `put spatula on plate` and the backward task to be `move spatula to the`

`left of the plate`, demonstrated in the image sequences in Figure 3.6. We consider three different subsets of Bridge data. `tabletop granular` comprises 796 trajectories performing tasks on a tabletop similar to the one used in our experiments. `tabletop granular + toy kitchen` comprises 823 trajectories, with `toy kitchen` trajectories specifically focusing on pick and place tasks with a variety of objects. `tabletop granular + toy kitchen + dark wood` comprises 1764 trajectories, with `dark wood` trajectories including some spatula pick and place tasks similar to ours, among many other tasks. We pretrain three separate policies using offline RL on each of the Bridge data combinations, as well as 120 in-domain demonstrations (50 forward task rollouts, 50 backward task rollouts, and 20 failures), as done in RoboFuME. The results of evaluating the three policies on the forward and backward tasks are shown in Table 3.1.

Bridge data subsets	tabletop granular	tabletop granular + toy kitchen	tabletop granular + toy kitchen + dark wood
Forward success rate	20%	10%	5%
Backward success rate	25%	10%	10%

Table 3.1: Success rates on forward and backward task for Spatula Pick-Place using different subsets of Bridge data, on 20 trials of each task.

From these results, we observe that increasing the size and diversity of the pretraining dataset does not necessarily lead to better results on the task, in fact it is the opposite. Qualitative analysis of the pick and place behavior shows a similar declining trend with more Bridge data. Even when adding the `dark wood` dataset which has the same task in the dataset, the performance is the worst among the three subset combinations. This demonstrates the importance of pretraining data selection for the performance of the eventual policy.

We noted that the success rates for the novel Spatula Pick-Place task was much lower than the reported success rates in Yang et al. [2023a], suggesting issues with generalization capabilities of the pipeline. We also observed that RoboFuME codebase not only pretrains on Bridge data, but also in-domain demonstration data that is upsampled by 8x to be proportional to the size of Bridge data. This could explain the observed results that increasing the size of pretraining Bridge datasets causes worse performance, because it dilutes the upsampled in-domain data. This demonstrates that the RoboFuME pipeline is heavily dependent on in-domain demonstrations to succeed. We test the effect of these demonstrations in the next set of experiments.

3.3.2 In-domain Demonstrations for RoboFuME

While RoboFuME has demonstrated interesting results in reducing human effort in reward engineering and resets, there is a notable cost incurred with collecting in-domain demonstrations for each new task. In-domain demonstrations are crucial for policies pretrained offline to succeed in task transfer

to the new environment, as we will see in the following experiments. In-domain demonstrations are also crucial for finetuning MiniGPT-4 to yield accurate sparse task completion rewards.

To avoid confounding generalization issues and to verify successful reproduction of the pipeline, we picked two tasks in the RoboFuME task suite that performed well, Cloth Folding (Figure 3.4) and Cube Covering (Figure 3.5). On both tasks, language-conditioned policies pretrained with behavior cloning (BC) and offline RL had decent success rates, which improved with online finetuning. To test the necessity of in-domain demonstrations for the success of the pipeline, for each task we pretrained four policies: language-conditioned BC on Bridge data and in-domain demonstrations, offline on Bridge data and in-domain demonstrations, language-conditioned BC on Bridge data only, and offline RL on Bridge data. We used the same Bridge data subsets as RoboFuME for each task, with newly collected in-domain demonstrations using our setup. We evaluated the four policies on the forward and backward tasks for each task category, and the results are shown in Table 3.2.

We successfully reproduced the results of RoboFuME for the BC and RL policies trained on both Bridge and in-domain data (see the first two columns of Table I in Yang et al. [2023a]). However, removing in-domain demonstration data from the pretraining dataset was catastrophic for policy learning, resulting in zero successes for both task categories on forward and backward tasks. This confirms the heavy reliance of the RoboFuME pipeline on in-domain demonstrations.

		BC bridge + indomain	RL bridge + indomain	BC bridge only	RL bridge only
Cloth Folding	forward success rate	65%	70%	0%	0%
	backward success rate	40%	50%	0%	0%
Cube Covering	forward success rate	35%	55%	0%	0%
	backward success rate	50%	65%	0%	0%

Table 3.2: Success rates on forward and backward task for Cloth Folding and Cube Covering tasks, on 20 trials of each task.

In-domain demonstrations are crucial for both aspects of the RoboFuME pipeline: pretraining policies with language-conditioned BC or offline RL as well as finetuning the task classifier that yields sparse rewards. It is not scalable to collect in-domain demonstrations for every new task we care about. Furthermore, the pipeline required in-domain demonstrations to meet several constraints, such as minimizing multimodality in the demonstrations, and if at evaluation or during finetuning there are any differences in the environment (e.g., lighting changes, changes in object position, changes in background), the system is very likely to fail. Overall, these experiments demonstrate reliance on in-domain demonstrations for transfer to new environments makes the system highly

brittle, evidenced by the ablation experiments conducted that pretrained policies only on Bridge data without the in-domain demos resulting in zero success on all tasks. Therefore, while RoboFuME reduces human effort in reward specification and resets, collecting demonstrations is still a bottleneck for this method, both in human effort and in the fragility of the system.

3.3.3 Online Finetuning with Dense Shaping Rewards

Demonstrating success rates on the cloth tasks comparable to the RoboFuME results was a positive indicator for successful reproduction of the online RL pipeline. However, based on the results in Table 3.1, failure to achieve similar results on the Spatula Pick-Place pretrained offline RL policy on Bridge data and high-quality in-domain demonstrations, regardless of the Bridge data subset used, suggests that generalizing the online RL pipeline to new tasks is challenging.

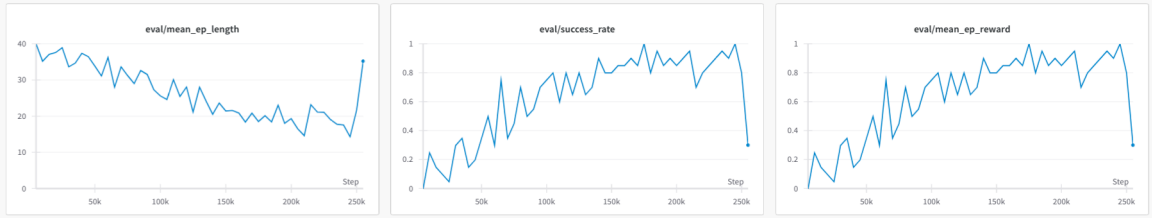
Our ablation experiments in Table 3.2 demonstrated that without in-domain data, all policy variants struggled to get any meaningful learning signal on every task. Furthermore, a preliminary experiment finetuning the offline RL policy pretrained only on Bridge data demonstrated challenges learning with sparse rewards achieves barely any increase in success rates (column 6 of Table 3.3). Therefore, we hypothesized that with dense shaping rewards, policies pretrained on both dense and sparse rewards would struggle to learn during online finetuning, and we were unlikely to achieve meaningful success rates on Spatula Pick-Place without in-domain demonstrations.

Given the difficulty achieving similar success rates to RoboFuME tasks by the language-conditioned BC and offline RL policies pretrained on Bridge and in-domain data for Spatula Pick-Place (Table 3.1), it is possible that even policies pretrained on in-domain data may struggle to learn using RoboFuME's online RL pipeline. The goal of the following experiments, therefore, is to see how much online finetuning with sparse rewards can improve the somewhat lackluster results of the offline RL policy pretrained on Bridge and in-domain data. Subsequently, we test whether dense reward shaping can achieve higher success rates with the same number of steps of online finetuning, thus showing demonstrable acceleration of the online finetuning process. Therefore, to help online RL pipelines like RoboFuME generalize to new tasks, adding dense rewards can help, especially since the dense rewards are obtained using VLMs which demonstrate significant generalization capabilities.

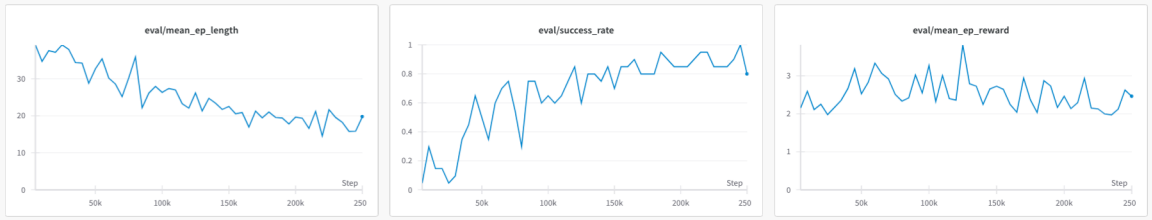
The full approach for dense reward shaping is detailed in 3.2.1, with the implementation that once the waypoint trajectory is generated using GPT-4V, we use the centroid of each grid tile in the trajectory and interpolate it with some randomness to create a denser trajectory of pixel coordinates in image space. We use calculate the negative L2 distance between the current robot position and the target waypoint (initialized as the first waypoint in the VLM-generated sequence), changing the target waypoint to the next one in the sequence to encourage progress along the trajectory. We pass each distance through a modified tanh function, so $r_{\text{dense}} = 0.5(1 - \tanh(\lambda(d_t - \varphi))$, where d_t is the negative L2 distance between the robot position and the target waypoint at timestep t, and scaling factor λ and offset φ are hyperparameters. We set $\lambda = -0.02$ and $\varphi = 120$ for

the results reported in Table 3.3. This ensures the dense reward stays between 0 and 1, with values closer to 1 when the robot trajectory is closer to the interpolated VLM-generated waypoint trajectory. What distinguishes trajectories that stay close to the VLM-generated trajectories and truly good trajectories that complete the task is the sparse task completion reward, which effectively doubles the reward (since sparse reward is either 0 or 1) for trajectories that are close to the VLM-generated trajectory and also complete the task. We set the final reward for each timestep $r = 0.5(r_{\text{dense}} + r_{\text{sparse}})$ to scale the overall reward to between between 0 and 1 also.

We conduct some preliminary experiments in simulation to investigate the effects of finetuning with a dense reward. We further investigate the claim of whether using a dense reward formulation can reduce the reliance of the policy on in-domain demonstrations during policy pretraining. The results of the simulation experiments for policies using the standard number of in-domain demonstrations, including the reproduction of the RoboFuME pipeline, can be seen in Figure 3.7:



(a) Standard quantity of in-domain demonstrations, sparse reward only (RoboFuME).

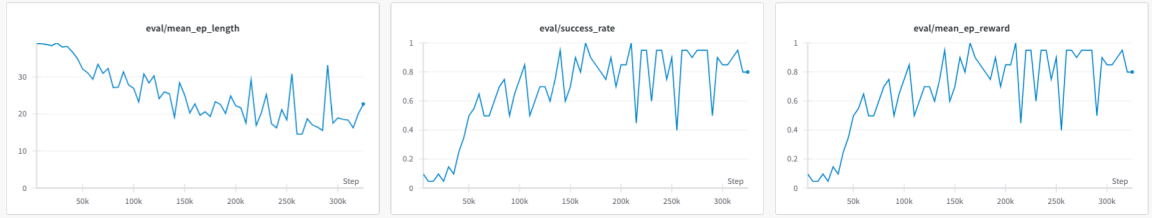


(b) Standard quantity of in-domain demonstrations, dense and sparse rewards.

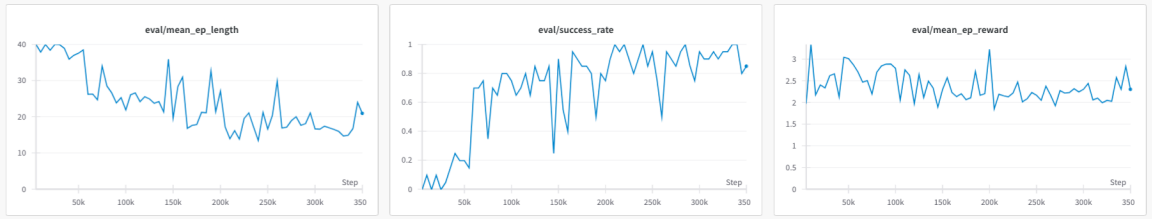
Figure 3.7: Evaluation metrics of policies trained on the standard number of in-domain demonstrations, using sparse only or dense and sparse rewards.

We see here that adding the dense reward generally performs comparably. To determine whether one of these reward formulations, sparse only vs. dense and sparse, is more adversely affected by reducing the number of in-domain demonstrations during pretraining, we conduct two simulation experiments for policies using fewer in-domain demonstrations, for which the results are shown in Figure 3.8. We see that these also perform comparably with each other and with the previous experiment using the standard number of in-domain demonsrtrations. The dense and sparse reward formulation reaches higher success rates slightly faster despite fewer in-domain demonstrations. Due to the dense rewards in simulation only being approximated with a single waypoint at the target location, we believe a denser waypoint trajectory should help with learning on the real robot, with

the simulation experiments verifying that adding dense rewards at least does not hurt or inhibit learning. The fact that RoboFuME also uses 100 in-domain demonstrations also suggests that for the real robot setup, the quantity of in-domain demonstrations is more important, more so than the simulation environment.



(a) Fewer in-domain demonstrations, sparse reward only.



(b) Fewer in-domain demonstrations, dense and sparse rewards.

Figure 3.8: Evaluation metrics of policies trained on fewer in-domain demonstrations, using sparse only or dense and sparse rewards.

For policy learning for the real robot setup, we pretrain policies for Spatula Pick-Place on the best performing subset of Bridge data from the experiments in Section 3.3.1, namely `tabletop_granular` only. The offline RL policy pretrained on Bridge and high-quality in-domain demonstration data achieved 25% on both the forward and backward tasks. We finetune this policy online for 20K steps using only VLM-generated sparse rewards, with resets every 2 episodes to maximize meaningful online interactions, which is equivalent to the the RoboFuME method. Separately, we finetune this policy online for 20K steps using both VLM-generated sparse rewards and dense rewards calculated with respect to a VLM-generated waypoint trajectory, again with resets every 2 episodes. The results are shown in Table 3.3.

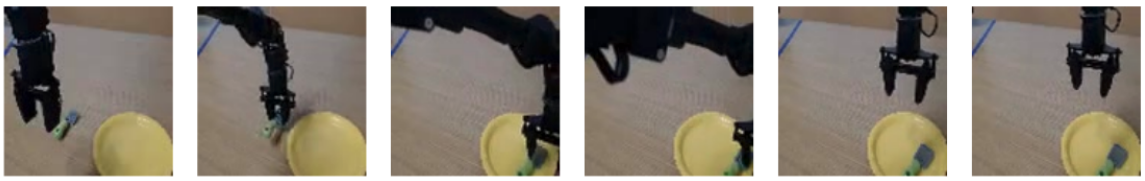
Spatula Pick-Place Task	BC bridge only	RL bridge only	BC bridge + indomain	RL bridge + indomain	RL bridge only + online FT (sparse)	RL bridge + indomain + online FT (sparse)	RL bridge + indomain + online FT (sparse+dense)
forward success rate	0%	0%	30%	25%	5%	40%	45%
backward success rate	0%	0%	20%	25%	0%	35%	40%

Table 3.3: Results of experiments finetuning pretrained policies online for 20K steps with dense rewards for Spatula Pick-Place task.

We see that after finetuning with VLM-generated sparse rewards (the full RoboFuME pipeline), the performance of the offline RL policy pretrained on Bridge and in-domain data increases from 25% to 40% for the forward task and 25% to 35% for the backward task. While this is still not as high as the success rates of the cloth tasks, the increase in success rate with finetuning using sparse rewards is comparable to a Pot Pick-Place task in the original RoboFuME paper, and both Pick-Place tasks are considerably harder than the cloth tasks. After finetuning with VLM-generated sparse rewards and dense shaping rewards, the performance of the offline RL policy pretrained on Bridge and in-domain data further increases to 45% for the forward task and 40% for the backward task. This is comparable to the success rate of RoboFuME in the Pot Pick-Place task after finetuning with 30K steps using only sparse rewards.

While the increase in performance finetuning with both dense and sparse rewards compared to using just sparse rewards is not extremely significant just based on success rate, the qualitative behavior observed was better, with representative image sequences demonstrating policy performance on the real robot shown in Figure 3.9. The policy finetuned with sparse rewards, while fairly successful, commonly demonstrated the behavior shown in Figure 3.9a where the spatula was dropped onto the plate from a high height, rather than lowering and placing the spatula as done in the demonstrations. The robot gripper also continues moving to the right rather than hovering above the placement point, which is indicative of fairly frequent coincidental successes. On the other hand, the policy finetuned with dense rewards, while slightly more performant success rate-wise, demonstrated better qualitative behavior shown in Figure 3.9b, lowering the spatula onto the plate and hovering above the placement point, as done in the demonstrations. This is likely due to dense rewards shaping the behavior to be more similar to the trajectory generated by GPT-4V.

(a) Qualitative behavior of policy finetuned only on sparse rewards: Robot drops spatula on the plate from a height and continues moving rightward.



(b) Qualitative behavior of policy finetuned only both dense and sparse rewards: Robot lowers and place spatula on the plate and subsequently hovers above the placement position.

Figure 3.9: Qualitative behavior of policies finetuned on sparse only vs. dense and sparse rewards.

Therefore, the quantitative results in Table 3.3 and qualitative results in Figure 3.9 collectively suggest that dense shaping rewards extracted from GPT-4V can help facilitate generalization of online RL pipelines like RoboFuME to new tasks where only using sparse rewards would not be able to generalize as well. Policies finetuned with dense rewards may also be less brittle when reducing the number of in-domain demonstrations during pretraining, though more real-robot experiments finetuning policies pretrained with fewer demonstrations are needed to verify this. Further hyperparameter tuning in the reward computation algorithm (namely scaling factor λ and offset φ in the reward formulation above) as well as in the CalQL algorithm could further improve the success rate of policies finetuned with dense and sparse rewards. Future work can also investigate whether it is possible to use fewer in-domain demonstrations when finetuning with dense and sparse rewards without significantly hurting the policy's performance, as done in the simulation experiments.

3.4 Discussion & Analysis

Our experiments have presented several takeaways on the opportunities and challenges of autonomous online RL pipelines such as RoboFuME, and the potential of mark-based visual prompting of VLMs like GPT-4V for improving generalization capabilities in robot learning pipelines. From our experiments selecting different pretraining datasets for the novel Spatula Pick-Place, the selection of prior offline datasets is incredibly important for downstream success of the pretrained policy. It might be unique to the RoboFuME pipeline that smaller prior datasets are better to avoid diluting the in-domain demonstration data, and the optimal selection of prior data could have greatly improved both the pretrained policy and finetuned policy's performance. However, it is still surprising that including demonstrations of the exact task being evaluated on (though in a different environment) does not help with task completion. More research into extracting relevant data from prior datasets can alleviate this issue.

The experiments also demonstrate the challenges of relying on high-quality in-domain demonstrations with low multimodality to successfully pretrain policies for online RL. Such pipelines are made much more brittle and less robust to generalizing to new tasks, objects, and environments. It is possible that a combination of selecting a highly task-relevant subset of the Bridge dataset facilitating good transfer to the current task and improved hyperparameter tuning for CalQL could have enabled the system to learn via online finetuning with dense rewards without needing any in-domain demonstrations at all, but this would require much more extensive further exploration in this direction in both pretraining data selection and hyperparameter search for online RL. A broader goal of subsequent research in this area would be to develop online RL methods that are much less reliant on or completely eliminate the need for in-domain demonstrations, and able to more effectively extract useful priors from offline datasets like Bridge datasets during pretraining, in addition to improving finetuning methods.

That being said, our final experiment demonstrated that dense shaping rewards extracted from VLMs can indeed help to speed up online RL, and facilitate generalization to new tasks where only relying on sparse rewards may not do as well. Though it was necessary to pretrain the policy on both Bridge and in-domain data, using a combination of dense and sparse rewards were able to facilitate faster and better learning than just sparse rewards. Our reward formulation could be a possible modification to existing finetuning methods to make them more robust to changes in tasks, objects, and environments. We have demonstrated the benefits of dense shaping rewards extracted from VLMs, and open up new avenues of exploration for new ways of leveraging the generalization capabilities of VLMs to enhance the robustness of robot learning systems. That said, both RoboFuME and this method are bottlenecked by the speed of inference of VLMs; finetuning for 20K steps took about 3-4 human hours with resets. Reducing the latency of online RL methods will allow finetuning for many more steps in a shorter period of time. Future work in this area can look to using VLMs, more sophisticated pretraining methods, or other approaches to scale efficiently to a greater diversity of tasks, objects, and environments.

Chapter 4

Conclusions & Future Work

In this thesis, we present two research projects that leverage prior data to develop embodied agents that generalize and adapt quickly to novel environments. We leverage the notion of affordances, the action potential of objects, to extract relevant representations and priors from human data to guide robot behaviors, specifically enabling robotic agents to learn through their own experience.

Concretely, in Chapter 2, we explored how to extract affordances from diverse human video datasets to facilitate intelligent robot interactions in novel environments. We did this by training a language-conditioned model from scratch on interaction trajectories and contact points extracted from the EPIC-KITCHENS egocentric human video dataset to predict hand and object trajectories given a novel image and language instruction. However, the limitations of the VRB model demonstrated empirical challenges with training a predictive model on egocentric human video data. That said, we were able to successfully verify the hypothesis that dense rewards were helpful in completing tasks in the D5RL simulated environment, which left open the question of the most effective and generalizable way to generate dense shaping trajectories to guide online RL.

In light of the rapid advances in capabilities of multimodal foundation models, in Chapter 3, we explored leveraging state-of-the-art VLMs to extract dense shaping rewards for online RL. In contrast to prior works that do majority of planning and reasoning in language space, we highlight the benefits of extracting affordance representations in the visual domain, representing affordances for robotic manipulation tasks in image space. Compared to training an affordance model from scratch on diverse human videos, leveraging representations encoded in VLMs is much more efficient and generalizable, allowing us to indirectly leverage the large, Internet-scale datasets that these models are trained on. Building on an existing pipeline for autonomous RL, and taking inspiration from visual prompting methods that enabled zero-shot robotic manipulation, we used mark-based visual prompting to extract affordance representations from GPT-4V in the form of keypoints and waypoints, and used the dense waypoint trajectory as a shaping reward for online RL.

While the goal of eliminating in-domain demonstrations for pretraining policies was ambitious,

we were able to demonstrate that using dense shaping rewards from VLMs facilitated improved generalization to new tasks which existing online RL pipelines were not robust to and finetuning with only sparse rewards were insufficient. This suggests that pretrained object-centric representations from VLMs can facilitate intelligent robotic interaction for learning through experience. Together, these methods contribute insights to tackling the lack of shaping rewards for online RL, and more broadly aim to improve the sample efficiency of online RL methods enabling robots to learn through trial-and-error with minimal human supervision. We also contribute insights to the pretrain-finetune paradigm that is becoming increasingly popular for robot learning, especially the importance of selecting task-relevant subsets of pretraining data and methods to accelerate finetuning procedures.

Several open questions remain in this research direction. First, truly robust learning techniques will scale with more data. We have not seen this trend in robotics thus far, and developing approaches that scale without selecting subsets of previously collected human or robot data will be paramount for robotics to see similarly rapid advancements like with language and vision. Affordances are just one formalization to extract meaningful representations for robot learning; developing pretraining techniques that can effectively leverage a variety of data sources - real robot data, human data, simulator data, and out-of-domain Internet data - will be crucial to advancing the field. Next, in addition to autonomous robot learning through trial-and-error, improving autonomous robot data collection will be hugely important in facilitating effective transfer to new environments. Perhaps pretraining solely on out-of-domain data is an overly ambitious goal, but collecting in-domain demonstrations under strict constraints is also not scalable. Developing pipelines that pretrain policies on out-of-domain data, including leveraging foundation models, but can also scalably and autonomously collect high-quality in-domain data for pretraining could help with downstream task completion in novel environments, which can improve the efficiency of learning to perform new tasks via online RL. Finally, we have explored leveraging dense shaping rewards to effectively finetune policies online, but there are several other ways for robotic agents to effectively extract or infer rewards from the environment or human feedback to guide task completion. Learning rewards for online policy improvement without manually engineering or specifying reward signals will be important for robots to continually and autonomously learn to adapt and generalize.

Researchers in the robotics community commonly debate about whether doubling down on data, algorithms, benchmarks, or simulations is the key to solving robotics. However, unlike language and vision, the unparalleled diversity in embodiments, action spaces, environments, tasks, and objects presents a huge challenge to robotics such that there is no silver bullet solution. This research presents the insight that efficiently collecting and curating the right data from a variety of sources for pretraining, as well as finetuning techniques that facilitate adaptation to a specific embodiment and environment without forgetting information learned during pretraining, are essential for a scalable end-to-end framework for robot learning. A mixture of these crucial ingredients may then unlock unprecedented capabilities in harnessing deep learning techniques for robotics.

Bibliography

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Mengyuan Yan, and Andy Zeng. Do as i can and not as i say: Grounding language in robotic affordances. 2022.

Meta AI. Introducing meta llama 3: The most capable openly available llm to date, 2024. URL https://ai.meta.com/blog/meta-llama-3/.

Lars Ankile, Anthony Simeonov, Idan Shenfeld, and Pulkit Agrawal. Juicer: Data-efficient imitation learning for robotic assembly, 2024.

Anthropic. Introducing claude, 2023a. URL https://www.anthropic.com/news/introducing-claude.

Anthropic. Claude 2, 2023b. URL https://www.anthropic.com/news/claude-2.

Anthropic. Introducing the next generation of claude, 2024. URL https://www.anthropic.com/news/claude-3-family.

Many authors. Open X-Embodiment: Robotic learning datasets and RT-X models. https://arxiv.org/abs/2310.08864, 2023.

Shikhar Bahl, Russell Mendonca, Lili Chen, Unnat Jain, and Deepak Pathak. Affordances from human videos as a versatile representation for robotics. In *Computer Vision and Pattern Recognition (CVPR)*, 2023.

Bowen Baker, Ilge Akkaya, Peter Zhokhov, Joost Huizinga, Jie Tang, Adrien Ecoffet, Brandon Houghton, Raul Sampedro, and Jeff Clune. Video pretraining (vpt): Learning to act by watching unlabeled online videos, 2022.

Max Balsells, Marcel Torne, Zihan Wang, Samedh Desai, Pulkit Agrawal, and Abhishek Gupta. Autonomous robotic reinforcement learning with asynchronous human feedback, 2023.

Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation, 2016.

Homanga Bharadhwaj, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. Zero-shot robot manipulation from passive human videos. *arXiv preprint arXiv:2302.02011*, 2023.

Chethan Bhateja, Derek Guo, Dibya Ghosh, Anikait Singh, Manan Tomar, Quan Vuong, Yevgen Chebotar, Sergey Levine, and Aviral Kumar. Robotic offline rl from internet videos via value-function pre-training, 2023.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-1: Robotics transformer for real-world control at scale, 2023.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

Annie S. Chen, Suraj Nair, and Chelsea Finn. Learning generalizable robotic reward functions from "in-the-wild" human videos, 2021.

Lawrence Yunliang Chen, Kush Hari, Karthik Dharmarajan, Chenfeng Xu, Quan Vuong, and Ken Goldberg. Mirage: Cross-embodiment zero-shot policy transfer with cross-painting, 2024.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin

Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways, 2022.

Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling egocentric vision: The epic-kitchens dataset. In *European Conference on Computer Vision (ECCV)*, 2018.

Ahmad Darkhalil, Dandan Shan, Bin Zhu, Jian Ma, Amlan Kar, Richard Higgins, Sanja Fidler, David Fouhey, and Dima Damen. Epic-kitchens visor benchmark: Video segmentations and object relations, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

Maximilian Du, Olivia Y. Lee, Suraj Nair, and Chelsea Finn. Play it by ear: Learning skills amidst occlusion through audio-visual imitation learning, 2022.

Yuqing Du, Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas. Guiding pretraining in reinforcement learning with large language models, 2023.

Frederik Ebert, Yanlai Yang, Karl Schmeckpeper, Bernadette Bucher, Georgios Georgakis, Kostas Daniilidis, Chelsea Finn, and Sergey Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets, 2021.

Kristen Grauman et al. Ego4d: Around the world in 3, 000 hours of egocentric video. *CoRR*, abs/2110.07058, 2021. URL https://arxiv.org/abs/2110.07058.

Rishi Bommasani et al. On the opportunities and risks of foundation models, 2022.

Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot visual imitation learning via meta-learning, 2017.

Meta GenAI. Llama 2: Open foundation and fine-tuned chat models, 2023.

James J. Gibson. *The Ecological Approach to Visual Perception: Classic Edition*. Houghton Mifflin, 1979.

Gemini Team Google. Gemini: A family of highly capable multimodal models, 2024.

Jiayuan Gu, Sean Kirmani, Paul Wohlhart, Yao Lu, Montserrat Gonzalez Arenas, Kanishka Rao, Wenhao Yu, Chuyuan Fu, Keerthana Gopalakrishnan, Zhuo Xu, Priya Sundaresan, Peng Xu, Hao Su, Karol Hausman, Chelsea Finn, Quan Vuong, and Ted Xiao. Rt-trajectory: Robotic task generalization via hindsight trajectory sketches, 2023.

Abhishek Gupta, Justin Yu, Tony Z. Zhao, Vikash Kumar, Aaron Rovinsky, Kelvin Xu, Thomas Devlin, and Sergey Levine. Reset-free reinforcement learning via multi-task learning: Learning dexterous manipulation behaviors without human intervention, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Susan J Hespos and Kristy vanMarle. Physics for infants: Characterizing the origins of knowledge about objects, substances, and number. *Wiley Interdisciplinary Reviews: Cognitive Science*, 3(1): 19–27, January 2012. ISSN 1939-5078. doi: 10.1002/wcs.157.

Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration, 2017.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents, 2022a.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner monologue: Embodied reasoning through planning with language models, 2022b.

Aditya Kannan, Kenneth Shaw, Shikhar Bahl, Pragna Mannam, and Deepak Pathak. Deft: Dexterous fine-tuning for real-world hand policies. *CoRL*, 2023.

Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together, 2023.

Alexander Khazatsky, Ashvin Nair, Daniel Jing, and Sergey Levine. What can I do here? learning new skills by imagining visual affordances. *CoRR*, abs/2106.00671, 2021. URL `https://arxiv.org/abs/2106.00671`.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation, 2022.

Raymond Li, Loubna Ben Allal, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, Jia Li, Jenny Chim, Qian Liu, Evgenii Zheltonozhskii, Terry Yue

Zhuo, Thomas Wang, Olivier Dehaene, Mishig Davaadorj, Joel Lamy-Poirier, João Monteiro, Oleh Shliazhko, Nicolas Gontier, Nicholas Meade, Armel Zebaze, Ming-Ho Yee, Logesh Kumar Umapathi, Jian Zhu, Benjamin Lipkin, Muhtasham Oblokulov, Zhiruo Wang, Rudra Murthy, Jason Stillerman, Siva Sankalp Patel, Dmitry Abulkhanov, Marco Zocca, Manan Dey, Zhihan Zhang, Nour Fahmy, Urvashi Bhattacharyya, Wenhao Yu, Swayam Singh, Sasha Luccioni, Paulo Villegas, Maxim Kunakov, Fedor Zhdanov, Manuel Romero, Tony Lee, Nadav Timor, Jennifer Ding, Claire Schlesinger, Hailey Schoelkopf, Jan Ebert, Tri Dao, Mayank Mishra, Alex Gu, Jennifer Robinson, Carolyn Jane Anderson, Brendan Dolan-Gavitt, Danish Contractor, Siva Reddy, Daniel Fried, Dzmitry Bahdanau, Yacine Jernite, Carlos Muñoz Ferrandis, Sean Hughes, Thomas Wolf, Arjun Guha, Leandro von Werra, and Harm de Vries. Starcoder: may the source be with you!, 2023.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control, 2023.

Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey Levine. Moka: Open-vocabulary robotic manipulation through mark-based visual prompting, 2024.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023a.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding dino: Marrying dino with grounded pre-training for open-set object detection, 2023b.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. *arXiv preprint arXiv: Arxiv-2310.12931*, 2023a.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. In *International Conference on Learning Representations (ICLR)*, 2023b.

Parsa Mahmoudieh, Deepak Pathak, and Trevor Darrell. Zero-shot reward specification via grounded natural language. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 14743–14752. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/mahmoudieh22a.html.

Daphne Maurer, Catherine Mondloch, and Terri Lewis. Effects of early visual deprivation on perceptual and cognitive development. *Progress in brain research*, 164:87–104, 02 2007. doi: 10.1016/S0079-6123(07)64005-9.

Russell Mendonca, Shikhar Bahl, and Deepak Pathak. Alan: Autonomously exploring robotic agents in the real world. In *International Conference on Robotics and Automation (ICRA)*, 2023.

Gabriel Meseguer-Brocal and Geoffroy Peeters. Conditioned-u-net: Introducing a control mechanism in the u-net for multiple source separations. In *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, 2019.

Inc. Midjourney. Midjourney, 2022. URL https://www.midjourney.com/.

Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. HowTo100M: Learning a Text-Video Embedding by Watching Hundred Million Narrated Video Clips. In *ICCV*, 2019.

Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, Xiao Wang, Xiaohua Zhai, Thomas Kipf, and Neil Houlsby. Simple open-vocabulary object detection with vision transformers, 2022.

Tushar Nagarajan and Kristen Grauman. Learning affordance landscapes for interaction exploration in 3d environments. In *Neural Information Processing Systems (NeurIPS)*, 2020.

Tushar Nagarajan, Christoph Feichtenhofer, and Kristen Grauman. Grounded human-object interaction hotspots from video. In *International Conference on Computer Vision (ICCV)*, 2019.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In *Conference on Robot Learning (CoRL)*, 2022.

Mitsuhiko Nakamoto, Yuexiang Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and year=2023 Sergey Levine, booktitle=Neural Information Processing Systems (NeurIPS). Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning.

OpenAI. Introducing chatgpt, 2022. URL https://openai.com/index/chatgpt/.

OpenAI. Hello gpt-4o, 2024a. URL https://openai.com/index/hello-gpt-4o/.

OpenAI. Gpt-4 technical report, 2024b.

Pierre-Yves Oudeyer and F. Kaplan. What is intrinsic motivation? a typology of computational approaches. *Frontiers in Neurorobotics*, 1, 2007. URL https://api.semanticscholar.org/CorpusID:6933296.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*, 2017.

Deepak Pathak, Dhiraj Gandhi, and Abhinav Gupta. Self-supervised exploration via disagreement. In *ICML*, 2019.

Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. 2018. URL https://api.semanticscholar.org/CorpusID:49313245.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019. URL https://api.semanticscholar.org/CorpusID:160025533.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation, 2021.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022.

Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, Zhaoyang Zeng, Hao Zhang, Feng Li, Jie Yang, Hongyang Li, Qing Jiang, and Lei Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 2024.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

Jürgen Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. 1991. URL https://api.semanticscholar.org/CorpusID:18060048.

Dandan Shan, Jiaqi Geng, Michelle Shu, and David Fouhey. Understanding human hands in contact at internet scale. 2020.

Archit Sharma, Kelvin Xu, Nikhil Sardana, Abhishek Gupta, Karol Hausman, Sergey Levine, and Chelsea Finn. Autonomous reinforcement learning: Formalism and benchmarking, 2022.

Linda Smith and Lauren Slone. A developmental approach to machine learning? *Frontiers in Psychology*, 8:2124, 12 2017. doi: 10.3389/fpsyg.2017.02124.

Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Sean Kirmani, Brianna Zitkovich, Fei Xia, Chelsea Finn, and Karol Hausman. Open-world object manipulation using pre-trained vision-language models, 2023.

Charles Sun, Jędrzej Orbik, Coline Manon Devin, Brian H. Yang, Abhishek Gupta, Glen Berseth, and Sergey Levine. Fully autonomous real-world reinforcement learning with applications to mobile manipulation. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 308–319. PMLR, 08–11 Nov 2022. URL https://proceedings.mlr.press/v164/sun22a.html.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018. ISBN 0262039249.

Marcel Torne, Max Balsells, Zihan Wang, Samedh Desai, Tao Chen, Pulkit Agrawal, and Abhishek Gupta. Breadcrumbs to the goal: Goal-conditioned exploration from human-in-the-loop feedback, 2023.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

Vadim Tschernezki, Ahmad Darkhalil, Zhifan Zhu, David Fouhey, Iro Laina, Diane Larlus, Dima Damen, and Andrea Vedaldi. Epic fields: Marrying 3d geometry and video understanding, 2024.

Andrés Villa, Juan León Alcázar, Motasem Alfarra, Kumail Alhamoud, Julio Hurtado, Fabian Caba Heilbron, Alvaro Soto, and Bernard Ghanem. Pivot: Prompting for video continual learning, 2023.

Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning (CoRL)*, 2023.

Chen Wang, Linxi Fan, Jiankai Sun, Ruohan Zhang, Li Fei-Fei, Danfei Xu, Yuke Zhu, and Anima Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. In *Conference on Robot Learning (CoRL)*, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

Scott Wu. Introducing devin the first ai software engineer, 2024. URL https://www.cognition.ai/blog/introducing-devin.

Jiarui Xu, Sifei Liu, Arash Vahdat, Wonmin Byeon, Xiaolong Wang, and Shalini De Mello. Open-vocabulary panoptic segmentation with text-to-image diffusion models, 2023.

Ni Yan, Yupeng Mei, Ling Xu, Huihui Yu, Boyang Sun, Zimao Wang, and Yingyi Chen. Deep learning on image stitching with multi-viewpoint images: A survey. *Neural Processing Letters*, 55: 1–36, 03 2023. doi: 10.1007/s11063-023-11226-z.

Daniel Yang, Davin Tjia, Jacob Berg, Dima Damen, Pulkit Agrawal, and Abhishek Gupta. Rank2reward: Learning shaped reward functions from passive video. *ICRA*, 2024.

Jingyun Yang, Max Sobol Mark, Brandon Vu, Archit Sharma, Jeannette Bohg, and Chelsea Finn. Robot fine-tuning made easy: Pre-training rewards and policies for autonomous real-world reinforcement learning. *Conference on Robot Learning (CoRL)*, 2023a.

Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. The dawn of lmms: Preliminary explorations with gpt-4v(ision), 2023b.

Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montse Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, Brian Ichter, Ted Xiao, Peng Xu, Andy Zeng, Tingnan Zhang, Nicolas Heess, Dorsa Sadigh, Jie Tan, Yuval Tassa, and Fei Xia. Language to rewards for robotic skill synthesis, 2023.

Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. *Conference on Robot Learning (CoRL)*, 2021.

Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models, 2023.

Appendix A

GPT-4V Metaprompts

A.1 Metaprompt for Sparse Reward Generation

Given the task instruction and the image, has the task been completed in this image from the camera's perspective? Answer 'yes' or 'no', do not explain your reasoning.

A.2 Metaprompt for Waypoint Generation

Note: The metaprompt below was largely inspired by the metaprompt using by MOKA [44], with some modifications to generate the dense waypoint trajectory required for our project. We have tested this metaprompt with a variety of models, including Gemini Pro and GPT-4o, and this metaprompt is suitable for use with other models with comparable performance to GPT-4V, as of June 2024.

Describe the robot gripper's motion to solve the task by selecting pre-defined keypoints and waypoints. The input request contains:

- The task information as dictionaries. The dictionary contains these fields:

  * 'instruction': The task in natural language forms.

  * 'object_grasped': The object that the robot gripper will hold in hand while executing the task.

  * 'object_unattached': The object that the robot gripper will interact with either directly or indirectly via 'object_grasped'.

- An image of the current table-top environment captured from a top-down camera, annotated with a set of visual marks:

  * candidate keypoints on 'object_grasped': Purple dots marked as 'P[i]' on the image, where [i] is an integer.

* candidate keypoints on 'object_unattached': Green dots marked as 'Q[i]' on the image, where [i] is an integer.

* grid for waypoints: Grid lines that uniformly divide the images into tiles. The grid equally divides the image into columns marked as 'a', 'b', 'c', 'd', 'e', 'f', 'g' from left to right and rows marked as 1, 2, 3, 4, 5 from bottom to top. * start point of gripper: Red dot marking the starting position of the gripper.

* start point of 'object_grasped': Blue dot marking the starting position of 'object_grasped'.

- An image of the current table-top environment captured from a side view camera, annotated with a set of visual marks:

* horizontal gridlines: Grid lines that uniformly divide the image from bottom to top, where each line represents height from the tabletop. The grid equally divides the image into segments marked as 'z1', 'z2', 'z3', 'z4', 'z5', 'z6', 'z7', 'z8' from bottom to top.

* start point of gripper: Red dot marking the starting position of the gripper.

* start point of 'object_grasped': Blue dot marking the starting position of 'object_grasped'.

The motion consists of a grasping phase and a manipulation phase, specified by 'grasp_keypoint', 'function_keypoint', 'target_keypoint', 'pre_contact_waypoint', 'post_contact_waypoint'.

Please note: In the grasping phase, the robot gripper sequentially moves to the 'pre_contact_waypoint' and grasps 'object_grasped' at the 'grasp_keypoint'. In the manipulation phase, the robot gripper moves to 'post_contact_waypoint' first, then moves the 'function_keypoint (or 'object_grasped' if 'function_keypoint' is '') to 'target_keypoint', performing a motion trajectory that completes the task instruction by first completing the grasping phase then the manipulation phase.

More specifically, the definitions of these points are:

- 'grasp_keypoint': The point on 'object_grasped' indicates the part where the robot gripper should hold.

- 'function_keypoint': The point on 'object_grasped' indicates the part that will make contact with 'object_unattached.'

- 'target_keypoint': If the task is pick-and-place, this is the location where 'object_grasped' will be moved to. Otherwise, this is the point on 'object_unattached' indicating the part that will be contacted at the end of the motion by 'function_keypoint', or the robot gripper (if 'function_keypoint' is '').

- 'pre_contact_waypoint': The waypoint in the free space that the robot gripper moves to before making contact with the 'grasp_keypoint'.

- 'post_contact_waypoint': The waypoint in the free space that the robot gripper moves to after making contact with the 'grasp_keypoint', before moving 'function_keypoint' (or 'object_grasped' if 'function_keypoint' is '') to 'target_keypoint'.

The response should be a dictionary in JSON form, which contains:

- 'grasp_keypoint': Selected from candidate keypoints marked as 'P[i]' on the image. This will

be '' if and only if 'object_grasped' is ''.

- 'function_keypoint': Selected from candidate keypoints marked as 'P[i]' on the image. This will be '' if and only if 'object_grasped' or 'object_unattached' is '' or the task is pick-and-place.

- 'target_keypoint': Selected from keypoint candidates marked as 'Q[i]' on the image. This will be '' if and only if 'object_unattached' is ''.

- 'start_block': A tuple ([pos], [height]): [pos] is the tile where the robot gripper (marked by a red dot) is located, which is selected from candidate tiles '[x][i]' marked on the first top-down image, where [x] is the column index as a lower letter and [i] is the row index as an integer. [height] is the line representing the height of the robot gripper, which is selected from candidate lines 'z[i]' marked on the second side view image, where [i] is the line index as an integer.

- 'grasp_block': A tuple ([pos], [height]): [pos] is the tile where the robot gripper grasps the object at 'grasp_keypoint', which is selected from candidate tiles '[x][i]' marked on the first top-down image, where [x] is the column index as a lower letter and [i] is the row index as an integer. [height] is the line representing the height of the grasped object, which is selected from candidate lines 'z[i]' marked on the second side view image, where [i] is the line index as an integer.

- 'target_block': A tuple ([pos], [height]): [pos] is the tile where 'target_keypoint' is currently located in, which is selected from candidate tiles '[x][i]' marked on the first top-down image, where [x] is the column index as a lower letter and [i] is the row index as an integer. [height] is the line representing the height of the target location, which is selected from candidate lines 'z[i]' marked on the second side view image, where [i] is the line index as an integer.

- 'block_sequence': A list of tuples of the form ([pos], [height]), where [pos] is selected from candidate tiles '[x][i]' marked on the first top-down image, where [x] is the column index as a lower letter and [i] is the row index as an integer, and [height] is selected from candidate lines 'z[i]' marked on the second side view image, where [i] is the line index as an integer. The list of tuples represents the locations that the robot gripper should move to sequentially to complete the task. Remember that the robot gripper first completes the grasping phase by moving to 'pre_contact_waypoint' and grasping 'object_grasped' at the 'grasp_keypoint'. Next, in the manipulation phase, the robot gripper moves to 'post_contact_waypoint' first, then moves the 'function_keypoint (or 'object_grasped' if 'function_keypoint' is '') to 'target_keypoint', performing a motion trajectory that completes the task instruction by first completing the grasping phase then the manipulation phase. Think about this step by step: the list should begin with 'start_block', navigate to 'grasp_block', and end with 'target_block'. The elements of the list should be tuples of the form ([pos], [height]), where [pos] is selected from candidate tiles '[x][i]' marked on the first top-down image, where [x] is the column index as a lower letter and [i] is the row index as an integer, and [height] is selected from from candidate lines 'z[i]' marked on the second side view image, where [i] is the line index as an integer. Furthermore, as 'block_sequence' is sequentially generated starting with 'start_block', from the last element ([pos1], [height1]) in the sequence so far, the next block ([pos2], [height2]) is generated as

follows: first generate [pos2] by choosing one of the eight tiles surrounding the current tile [pos1] (up, up-left, up-right, left, right, down, down-left, down-right), then generate [height2] by choosing the line above, below, or the same as the current line [height1]. The next tile should be chosen in a way that the resultant motion in 3D can be followed by the robot gripper to first complete the grasping phase, then after a successful grasp, complete the manipulation phase, thereby completing the task instruction with collision avoidance and all proper contacts made. This sequence generation process continues until the last tile in the sequence so far 'target_block', which is when the robot gripper's motion is completed. Double check that for each consecutive block of format ([pos], [height]), [pos] is reachable from the previous [pos] by moving either up, up-left, up-right, left, right, down, down-left, down-right, and [height] is reachable from the previous [height] by moving up, down, or staying the same.

- 'pre_contact_height': The height of 'pre_contact_waypoint' as one of the two options 'same' (same as the height of making contact with 'target_keypoint') or 'above' (higher than the height of making contact with 'target_keypoint').

- 'post_contact_height': The height of 'post_contact_waypoint' as one of the two options 'same' (same as the height of making contact with 'target_keypoint') or 'above' (higher than the height of making contact with 'target_keypoint').

- 'target_angle': Describe how the object should be oriented during this motion in terms of the axis pointing from the grasping point to the function point. Think about this step by step. First analyze whether this axis should be parallel with or perpendicular to the motion direction and the table surface respectively to better perform the task, then choose the axis orientation from one of these strings based on the motion direction: 'forward' (toward the top side of the image), 'backward' (toward the bottom side of the image), 'upward' (perpendicular to and away from the table surface), 'downward' (perpendicular to and towards the table surface), 'left' (towards the left side of the image), 'right' (towards the right side of the image). e.g., if the axis is parallel to the table surface and perpendicular to the motion direction, and the motion direction is backward, then the axis direction should be either 'left' or 'right'; if the axis is perpendicular to the table surface and parallel to the motion direction, and the motion direction is upward, then the axis direction should be either 'upward' or 'downward'.

We will first provide one in-context example, which contains the input (the corresponding task instruction and pair of input images (top-down and side view)), and the correct response. Then we will provide a new task instruction and pair of input images (top-down and side view) and ask for the corresponding response based on the in-context example.

Think about this step by step: First, choose 'grasp_keypoint', 'function_keypoint', and 'target_keypoint' on the correct parts of the objects. Next, determine which block the robot gripper is in ('start_block'), which block the 'grasp_keypoint' is located in ('grasp_block'), and which block the 'target_keypoint'

is located in ('target_block'). Then generate 'block_sequence' (starting with 'start_block', moving through some sequence of blocks to 'grasp_block', then moving through another sequence of blocks and ending with 'target_block') and choose 'pre_contact_height', 'post_contact_height' accordingly such that the robot gripper's resultant motion of 'pre_contact_waypoint'→ 'grasp_keypoint' → 'post_contact_waypoint' then moving 'function_keypoint' (or 'object_grasped' if 'function_keypoint' is '') to the 'target_waypoint' in 3D completes the grasping phase first then the manipulation phase and proper contacts will be made. Remember that in the first top-down image, the columns are marked as 'a', 'b', 'c', 'd', 'e', 'f', 'g' from left to right, and the rows are marked as 1, 2, 3, 4, 5 from bottom to top; in the second side view image, the lines are labeled 'z1', 'z2', 'z3', 'z4', 'z5', 'z6', 'z7', 'z8' from bottom to top. Explain the reasoning steps.