Interactive Language
Instructable Robot Learning

Oier Mees

Technische Fakultät
Albert-Ludwigs-Universität Freiburg

UNI FREIBURG

Dissertation zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften

Betreuer: Prof. Dr. Wolfram Burgard

Interactive Language Instructable Robot Learning

Oier Mees

Dissertation zur Erlangung des akademischen Grades Doktor der Naturwissenschaften
Technische Fakultät, Albert-Ludwigs-Universität Freiburg

Dekan	Prof. Dr. Roland Zengerle
Erstgutachter	Prof. Dr. Wolfram Burgard University of Technology Nuremberg
Zweitgutachter	Prof. Dr. Dieter Fox University of Washington
Drittgutachter	Prof. Dr. Joschka Boedecker University of Freiburg
Tag der Disputation	26. Mai 2023

The beliefs which we have the most warrant for have no safeguard, but a standing invitation to the whole world to prove them unfounded.

— John Stuart Mill, 1871

Abstract

One of the grand challenges in robotics is to create robots capable of performing a wide range of tasks in unstructured environments based on arbitrary user command. The key challenges to develop "general-purpose" robots are acquiring a diverse repertoire of general-purpose skills and non-expert users to be able to effectively specify tasks for the robot to solve. Moreover, such robots need strong generalization capabilities to adapt to new environments, identify and handle unfamiliar objects and to understand instructions it has never been given before. Despite the significant strides achieved in robotics over the last decades, the reality is that the majority of robots currently in use in real-world settings, such as warehouses and factories, are still limited to performing only a narrow range of pre-programmed behaviors for specific tasks in controlled industrial environments.

In this thesis, we introduce techniques that enable a robot to acquire general-purpose knowledge from purely offline, unstructured data that allows to compose long-horizon, multi-tier tasks by following unconstrained language instructions.

To achieve this, we first introduce a method to estimate pixelwise distributions for pairwise spatial relations between objects that enables a robot to place objects in accordance with the spatial relations expressed by the user. Our method lifts the requirement for pixelwise ground-truth data by classifying hallucinated high-level scene representations as an auxiliary task. We extend our approach to follow unconstrained language instructions to pick and place arbitrary objects and effectively resolve ambiguities through dialogues by grounding objects and their spatial relations. Secondly, as a step towards instructable robots that can learn many useful behaviors, we present CALVIN, the first public benchmark of instruction following that combines: natural language conditioning, multimodal high-dimensional inputs, 7-DoF continuous control, and long-horizon robotic object manipulation. We additionally contribute an extensive study of the most critical challenges in learning language conditioned policies from offline free-form imitation datasets. Thirdly, taking inspiration from how intelligent beings have the ability to discover, learn and transfer skills without supervision, we present a method that enables a robot to learn skills from unlabeled videos. Our proposed models enables training of continuous control policies to solve novel tasks that require the interpolation of previously seen skills. Fourthly, we address the challenge of producing previously unseen combinations of skills, learned from an offline, unstructured dataset, to reach temporally extended goals by "stitching" together skills. We contribute a self-supervised hierarchical approach that combines the strengths of the imitation learning and reinforcement learning paradigms to learn task-agnostic long-horizon policies from high-dimensional camera observations. Finally, we present a method that learns language-conditioned robotic visuomotor skills from unstructured data in the real world in a data-efficient manner by discovering object affordances to enable efficient policy learning and motion planning. Our contribution goes beyond existing paradigms by allowing the robot to follow unseen abstract natural language instructions, such as "tidy up the workspace and turnoff the lights" with no additional training.

We implemented and tested all methods presented in this thesis on real-world datasets and in extensive experiments with real robots. We show that our proposed approaches are key enablers to (i) learn diverse skills to perform tasks in house-like environments from uncurated data, (ii) relate human language to a robots perceptions and actions, and (iii) efficiently complete long-horizon, multi-tier manipulation tasks in the real world. We hope that these techniques will open the door for the future development of agents that can relate human language to their perception and actions and generalize abstract concepts to unseen entities in the same way humans do.

Zusammenfassung

Eine der großen Herausforderungen in der Robotik besteht darin, Roboter zu entwickeln, die in unstrukturierten, haushaltsähnlichen Umgebungen auf der Grundlage von Benutzerbefehlen eine Vielzahl von alltäglichen Aufgaben ausführen können. Die wichtigsten Herausforderungen bei der Entwicklung von sogenannten "Allzweck"-Robotern sind die Erlangung eines vielfältigen Repertoires an Fähigkeiten und die Möglichkeit für Benutzer, Aufgaben effektiv für den Roboter zu spezifizieren. Darüber hinaus benötigen solche Roboter starke Generalisierungsfähigkeiten, um sich an neue Umgebungen anzupassen, unbekannte Objekte zu identifizieren und zu manipulieren sowie neuartige Anweisungen zu verstehen. Trotz der erheblichen Fortschritte, die in den letzten Jahrzehnten in der Robotik erzielt werden konnten, werden in der Praxis immer noch mehrheitlich Roboter eingesetzt, welche auf eine begrenzte Anzahl vorprogrammierter Verhaltensweisen für spezifische Aufgaben beschränkt sind.

In dieser Arbeit werden Techniken vorgestellt, die es einem Roboter ermöglichen, Fähigkeiten und Fertigkeiten aus rein unstrukturierten Daten zu erlernen. Dies bietet solchen Systemen die Möglichkeit, langfristige, mehrschichtige Aufgaben durch die Befolgung von Sprachanweisungen auszuführen.

Um dies zu erreichen, stellen wir zunächst eine Methode vor, um pixelgenaue Verteilungen für paarweise räumliche Beziehungen zwischen Objekten zu schätzen, welche es es einem Roboter ermöglichen Objekte entsprechend den von dem Benutzer ausgedrückten räumlichen Beziehungen zu platzieren. Die vorgestellte Methode hebt die Anforderung an pixelgenaue Annotationen auf, indem sie halluzinierte Szenendarstellungen als Nebenaufgabe klassifiziert. Wir erweitern unseren Ansatz um Sprachanweisungen zu folgen, um beliebige Objekte zu greifen, zu platzieren und Ambiguitäten durch Dialoge zu lösen. Zweitens, als Schritt in Richtung auf programmierbarer Roboter, die viele nützliche Aufgaben lernen können, stellen wir CALVIN vor, den ersten öffentlichen Testumgebung, welche Folgendes kombiniert: das Folgen von Sprachanweisungen, multimodale hochdimensionale Eingaben, das Lernen von kontinuierlichen Regelungsstrategien mit sieben Freiheitsgraden und zeitlich langen Objektmanipulationsaufgaben. Wir tragen zusätzlich eine umfangreiche Studie der kritischsten Herausforderungen beim Lernen von Sprachbedingten Agenten aus Freiform-Imitationsdatensätzen bei. Drittens, inspiriert von der Fähigkeit intelligenter Wesen, zielgerichtete Handlungsstrategien ohne Aufsicht zu entdecken, zu lernen und zu übertragen, stellen wir eine Methode vor, die es einem Roboter ermöglicht, Fähigkeiten aus Videos zu lernen. Unser Ansatz ermöglicht das Training von kontinuierlichen Steuerungsmodellen, um neue Aufgaben zu lösen, die die Interpolation von zuvor gesehenen Fähigkeiten erfordern. Viertens befassen wir uns mit der Herausforderung, bisher unbekannte Kombinationen von Fähigkeiten aus einem offline, unstrukturierten Datensatz zu erzeugen, um zeitlich erweiterte Ziele zu erreichen, indem man Fähigkeiten zusammenbindet. Wir stellen einen selbst-überwachten hierarchischen Ansatz vor, der die Stärken des Imitationslernens und des berstärkenden Lernens kombiniert, um aufgabenunabhängige langfristige Regelungsstrategi-

en aus hochdimensionalen Kamera-Beobachtungen zu lernen. Schließlich stellen wir eine Methode vor, die sprachbedingte robotische Fähigkeiten aus unstrukturierten Daten in der realen Welt auf dateneffiziente Weise durch die Entdeckung von Objek Affordanzen lernt, um effiziente Manipulationsstrategien und Bewegungsplanung zu ermöglichen. Unser Beitrag geht über bestehende Paradigmen hinaus, indem er dem Roboter ermöglicht, unbekannte, abstrakte natürliche Sprachanweisungen wie "Räaume den Arbeitsbereich auf und schalte das Licht aus" ohne zusätzliches Training zu erforndern.

Sämtliche in dieser Arbeit vorgestellten Methoden wurden auf Datensätzen aus der realen Welt und in umfangreichen Experimenten mit echten Robotern implementiert und getestet. Wir zeigen, dass unsere vorgeschlagenen Ansätze Schlüsselfaktoren sind, um (i) vielfältige Fähigkeiten zur Durchführung von Aufgaben in haushaltsähnlichen Umgebungen aus unstrukturierten Daten zu lernen, (ii) menschlicher Sprache mit den Wahrnehmungen und Aktionen eines Roboters zu verbinden und (iii) langfristigen, mehrschichtigen Manipulationsaufgaben in der realen Welt effizient zu durchführen. Wir hoffen, dass die in dieser Arbeit vorgestellten Ansätze die Tür für die zukünftige Entwicklung von Agenten öffnen werden, die in der Lage sind, menschliche Sprache mit Ihrer Wahrnehmung und Handlungen zu verbinden und abstrakte Konzepte ähnlich zu Menschen zu generalisieren.

Acknowledgments

This thesis would have not been possible without the constant support and encouragement from many people. At the risk of unfairness, I would like to single out some of the many people to whom I feel especially indebted.

First and foremost, I thank my advisor *Prof. Wolfram Burgard* for the opportunity of pursuing a PhD at his AIS lab. He provided an excellent research environment to me, was open to new research ideas, encouraged me to work with real robots and gave me the opportunity to attend multiple conferences around the world. Moreover, I am immensely grateful to Wolfram for giving me the freedom and the resources to pursue my own research ideas. I feel privileged to have had the opportunity to work on open robotics challenges I feel passionate about.

I would like to thank *Prof. Dieter Fox* for all the insightful discussions at the conferences over these years and for agreeing to be the second examiner of this thesis. I am also grateful to have had the opportunity to do an internship at NVIDIA under Prof. Dieter Fox, collaborating with *Valts Blukis* and *Claudia D'Arpino*. I thank *Prof. Joschka Boedecker* for agreeing to be the third examiner of this thesis, and *Prof. Matthias Teschner* and *Prof. Armin Biere* for taking the time to be on my thesis committee. I am also grateful for my undergraduate advisers *Prof. Elena Lazkano* and *Prof. Basilio Sierra*, who sparked my interest in autonomous systems.

Throughout my time at the Autonomous Intelligent Systems (AIS) group, I am very happy to have had the opportunity to collaborate with an amazing set of students, faculty, and researchers. During the my master studies in Freiburg, I was lucky to be mentored by *Gian Diego Tipaldi*, *Nichola Abdo*, *Mladen Mazuran*, *Andreas Eitel* and *Luciano Spinello*. Together with Wolfram, they inspired me to pursue a PhD. I would like to thank *Maxim Tatarchenko* for our interdisciplinary collaboration at the beginning of my PhD and his friendship over the years. I thank *Lukas Hermann* for our successful collaborations and his help setting up the Panda robot. Special thanks goes to *Gabriel Kalweit* for our rewarding robot learning collaborations throughout the years. I would also like to thank *Chenguang Huan* for our great collaboration on zero-shot language based robot navigation. Furthermore, I would like to thank *Andy Zeng* from Google Research, for our multiple collaborations, such as co-organizing the Scaling Robot Learning workshop series, several language-guided robot learning projects and friendship throughout the years.

A big shout out goes to my office mates *Henrich Kolkhorst, Andreas Wachaja, Iman Nematollahi, Gabriel Kalweit* and *Andreas Eitel* for their advice and friendship over the years. I also spent fun time outside the lab, with great barbecues, dinners and activities. Special thanks go to *Andreas Wachaja* for our joint biking tours and to Andreas Eitel for co-founding our Atole Loco music band with me, so that we might have an alternative career choice in case robotics does not pan out.

I thank all the students who I have had the opportunity to supervise and collaborate with. Thank you for your hard work, dedication and your patience as I learned how to best advise you. A big thank to *Erick Rosete* and *Jessica Borja* for always going the extra mile. I would

also like to thank *Alp Emek* and *Markus Merklinger* for their efforts and hard work. I would like to thank all of my colleagues for making work more enjoyable and setting the ideal atmosphere for both productivity and fun: *Tim Caselitz, Tayyab Naseer, Tim Welschehold, Johan Vertens, Abhinav Valada, Alexander Schaefer, Lukas Luft, Johannes Meyer, Chau Do, Marina Kollmitz, Nicolai Dorka, Micheal Krawez, Niklas Wetzel, Noha Radwan, Gabriel Oliveira, Jingwei Zhang, Daniel Büscher, Wera Winterhalter, Freya Fleckenstein, Ayush Dewan, Jannik Zürn, Kshitij Sirohi, Shengchao Yan* and *Kürsat Petek*.

Special thanks to *Michael Keser, Evelyn Rusdea* and *Susanne Bourjaillat* for the tremendous technical and administrative support and being very supportive during this time. Thanks to Gabriel Kalweit and Nicolai Dorka for proofreading this thesis and your thoughtful comments.

I thank my friends in the Basque Country for their support and for all the joint fun activities whenever I came home for recharging my batteries.

My deepest gratitude goes to my parents, *Ludger* and *Begoña*, and my brother *Jan* for their unconditional love, their unwavering support at all times and for believing in me. Finally, I thank my late grandmother *Ascensión*, who did not have a chance to pursue higher education, for lovingly reminding me that education is a privilege.

This work has been supported by the Freiburg Graduate School of Robotics and the German Federal Ministry of Education and Research under contract number 01IS18040B-OML. Their support is gratefully acknowledged.

Contents

Chapter 1

Introduction

Robots are rapidly changing the way we live and work. In manufacturing, robots are being used for tasks such as welding, assembling, and mass-producing products, leading to an increase in productivity. In transportation, the development of autonomous vehicles is promising to bring about a new era of safe transportation, while in healthcare, robots are being used to assist doctors in various procedures such as surgery, resulting in improved outcomes for patients.

Robots are also making an impact in the home, with vacuum-cleaning robots being one of the most prevalent examples. A growing number of consumer home robots address more mundane tasks, such as lawn mowing and the cleaning of windows or pools. While their applications are vast, one field in which robotics has not lived up to our aspirations so far is personal, assistive robotics. For several decades, the idea of a "general-purpose" service robot that can assist for example the elderly by undertaking a variety of household chores has captured our imagination. The first such robots were envisioned by science fiction books (Asimov in the 1950s), movies (C-3PO from Star Wars in the 1970s) and cartoons (Rosey from The Jetsons in the 1980s), and were endowed with human like intelligence. Besides the ability to perform a wide range of everyday tasks, a common feature of these fictional robots is their ability to understand natural language and communicate with humans. In practice, despite considerable progress since the first industrial automation robot was introduced in the 1960s by Unimation, the vast majority of robots deployed out in the real world today continue to remain restricted to a narrow set of preprogrammed behaviors for specific tasks in controlled industrial settings. Moreover, existing domestic robots are currently restricted to performing simple and straightforward tasks that do not require a comprehensive understanding of the environment nor the ability to engage in collaborative dialogue with a human partner.

However, the recent advent of deep learning has revolutionized the fields of machine learning and computer vision. Advances made in training deep models and the availability of Internet-scale data have tremendously accelerated the capacity of AI systems to perform complex tasks – from decision-making [1] and logical reasoning [2], to protein folding [3] and generative content generation [4]. These developments are a constant inspiration towards developing "generalist robots" that can perform not just one, but a broader range of useful tasks in unstructured environments with minimal human interventions. Just as foundation models [5] from natural language processing (*e.g.,* GPT-3 [6]) and computer vision (*e.g.,* CLIP [7]) are capable of generalizing to new tasks with only handful of examples, a driving expectation is that data-driven models may likewise enable generalist robots to assist humans by learning to perform physical tasks with only a few or no examples.

Borrowing from these successes, recent work suggests that it's possible to train end-to-end

artificial neural networks to learn complex robotic manipulation skills [8], *e.g.,* mapping from raw pixels to control torques. Although end-to-end learning enables robots to evolve from performing only structured actions to act re-actively based on perceptual sensing, in practice, most current end-to-end models typically learn individual tasks from scratch one at a time and show poor generalization performance. Traditionally, obtaining multiple skills for robots involves defining a set of tasks, collecting expert demonstrations for each task, and training a separate policy for each task with supervised learning [9]. Another approach is to use reinforcement learning, but this involves manually designing a reward function for each task. The design of reward functions that elicit the desired agent behavior is especially challenging for real-world tasks, particularly when the state of the environment might not be accessible. Besides, designing a reward often requires the installation of specific sensors to measure as to whether the task has been executed successfully [10, 11]. In many scenarios, the need for task-specific engineering of reward functions undermines the benefits of end-to-end learning from pixels, if the reward function itself requires a dedicated perception pipeline. Additionally, using reinforcement learning in complex settings like robotics requires overcoming exploration challenges, which are often addressed by adding manual scripting primitives to guide the robot to areas with a non-zero reward. In general, both approaches require a significant amount of human effort for each new skill a robot is required to perform, and this effort is typically not transferable to other skills. Moreover, the costs of collecting data with real robot hardware can be expensive, time-consuming and difficult to automate. Another formulation is to train multiple tasks simultaneously with either of both approaches by conditioning the policy on a discrete task id, that can represent a skill such as opening a door [12]. Multi-task learning is a promising approach that aims to discover shared structure across tasks in a way that achieves greater efficiency and performance than solving tasks individually [13]. In practice, multi-task settings present a number of optimization challenges, making it difficult to realize large efficiency gains and a positive transfer compared to learning tasks independently [14, 15].

Furthermore, in open world settings, agents will be expected to perform not just a small discrete set of tasks, but rather a wide continuum of behaviors. For example, we might want the robot to open a drawer from the side instead of doing a top-down grasp, for which collecting a new set of demonstrations and retraining might be necessary. On a similar note, defining discrete tasks suffers from ambiguity. Imagine we want to teach a robot a skill for opening a sliding door of a cabinet. Assuming there exists a perception pipeline that can measure the percentage of how far the robot has moved the sliding door, which percentage should be considered as a threshold for a successful trial, 90%, 80%, 60%? People might have different answers for rating such trials as being successful or not. These problems stem from the fact that the world is continuous and can not be fully described by a set of discrete behaviors or symbols [16, 17]. Thus, robots that move to unstructured, real world settings will need to contend with this fact.

If we want systems that can perform a wide range of everyday tasks and exhibit the generality of human intelligence, they should learn a full continuum of behaviors rather than a set of discrete tasks from expert, curated data. In this thesis, we argue that in order to acquire a diverse repertoire of general-purpose skills, the *diversity* of the data is a key factor. Instead of repeating costly data collection and training from scratch for every new task and environment, robots need the ability to learn from diverse and unlabeled unstructured data. Although current embodied agents are typically trained tabula rasa on

large amounts of curated datapoints due to the flexibility of not requiring domain-specific knowledge to solve the tasks, the way humans learn suggests that diversity is paramount to learn multiple transferable and generalizable skills. Humans do not require experience manipulating thousands of instances of few object classes (*e.g.,* screwdrivers, mugs, cutlery) to learn generalizable grasping strategies. On to the contrary, we interact with thousands of object classes for which very few instances might be seen. Thus, a humans remarkable ability to manipulate unfamiliar objects originates from the ability to learn from diverse data and to transfer knowledge.

As robots become more capable to assist us with a broad range of everyday tasks and become ubiquitous across human-centered environments, they will be surrounded by people who do not have the knowledge to program and direct robots. Since most users will not be experts, the need for natural and effective human-robot communication grows. Recent works that have studied scaling up the number of tasks a robot learns, assume tasks being specified via goal images [18] or one-hot skill selectors [19], which are not practical for untrained users to instruct robots. Endowing a robot with the ability to learn how to perform tasks from videos of human demonstrations is a desirable and scalable alternative task specification [20]. In reality, learning to imitate tasks from human videos comes involves solving a difficult correspondence problem and understanding the semantics of the tasks being shown [21]. By contrast, language plays a crucial role in human communication, and its daily use encompasses various functions such as directing actions, asking and answering questions, sharing information, and requesting assistance. Thus, natural language presents a promising alternative form of task specification, providing an intuitive and flexible way for humans to communicate tasks and refer to abstract concepts. Similarly, it is difficult to conceive a truly generalist robot without also possessing the ability to understand and execute instructions conveyed in natural language. To effectively utilize human language, robots must establish a correspondence between words and the physical environment, mediated by the robot's sensorimotor system.

Developing systems that can demonstrate their visual understanding by generating or responding to natural language in the context of perceptual inputs is a long-standing goal. Marvin Minksy, one of the founding father of artificial intelligence, was quoted saying the following on the goal of a 1966 undergraduate summer research project [22]

> ... spend the summer linking a camera to a computer and getting the computer to describe what it saw.
>
> — *Marvin Minsky, 1966*

Despite the tremendous progress made in visual and language understanding since this now famously ambitious summer project, we are far away from achieving robots that can learn to relate human language to their world model. Understanding and following unconstrained language instructions is a notoriously challenging problem, subsuming many long term problems in AI [23, 24]. Learning to follow language instructions involves addressing a difficult symbol grounding problem [25], relating a language instruction to a robot's onboard noisy perception and physical actions. For example, a robot presented with the command "fetch the banana and place it left of the bottom object" must be able to relate language to its low-level perception (what does a banana look like?). It must perform visual and spatial reasoning about where to place the "banana" relative to the "bottom object" in order to reproduce the spatial relation "to the left of", which is inherently ambiguous as natural

language placement instructions do not uniquely identify a location in a scene. Moreover, the robot needs the ability to resolve ambiguous instructions or ask for help through dialogue with the user. Additionally, it must solve a complex sequential decision problem (what commands do I send to fetch an object, or to do a relative placement?). These notoriously challenging problems only get exacerbated if we consider the aforementioned need for learning a full continuum of skills and behaviors rather than a single task. As a result, prior work on mapping language and vision to actions has been studied mostly in restricted environments [26, 27], simplified actuators with discrete motion primitives [28, 29, 30] or restricted structured language inputs [31, 32]. This motivates the age-old question [25, 33], how might intelligent agents ground unconstrained, free-form natural language understanding in their own embodied perception?

From the perspective of human language acquisition, research in fields of psychology and linguistics suggest it to be a highly socially-mediated process [34, 35]. Studies have demonstrated that infants engage in language-learning through interactions with caregivers, who provide input in the form of words [36]. Conversely, there also exist theories in the linguistics fields, such as the theory of universal grammar by Chomsky [37] that postulate that all human languages share some fundamental similarities, and that these are attributable to innate principles unique to human language. While the underlying cognitive mechanisms of language acquisition in humans remain an area of ongoing investigation, robots need to rely on humans for language acquisition. In order to reduce the need for expensive human supervision, one of the issues we will investigate in this work will be bootstrapping instruction following by learning skills from mostly uncurated data, while annotating as little as 1% of it with natural language annotations. This setting mirrors the aforementioned human analogy for learning transferable and generalizable skills from diverse data.

Grounding language into the physical world is tightly coupled to the representation of concepts, skills and their generalization. Consider a robot that has learned to “pick up a green block" and “pick up a red mug", ideally it would understand what “pick up a green mug" means even though it has never encountered that concept in the training data. However, the inherent compositionality of language [38] and its fuzziness make it challenging to represent concepts as disentangled, standalone representations. Moving beyond toy combinations of orthogonal attributes, current deep generative models show remarkable understanding of visuo-lingual concepts [4]. Trained on Internet-scale data consisting of paired images and captions, they are able to generate realistic images for complex queries such as “a photo of an astronaut riding a horse underwater". Translating similar capabilities to an embodied agent that might understand open-ended instructions, such as “pick up slowly the rightmost object that is neither a fruit nor a toy”, is an open robotics problem that highlights the challenges of grounding artificial intelligence and concepts in shared sensory and experience. Moreover, we do not fully understand the mechanisms of human cognition for such complex instructions.

Cognitive psychology suggests that humans possess two distinct visual systems, a semantic stream (*what*) and a spatial stream (*where*) [39]. These vaguely reflect the building blocks needed for instruction following i) a set of low-level skills that coordinate the control of a robots body with its perception and that can be seamlessly combined together to intelligently act in the world, and ii) a higher, semantic policy that reasons on temporally extended horizons about the best way to complete abstract language instructions, such as “tidy up the workspace and turn off the lights”. Developing robots with these capabilities requires solving the many challenges described so far. Contrary to the expectations of

the general public, learning sensorimotor skills alone typically requires a massive large-scale data collection effort [40, 41, 42] with frequent human interventions and solving the aforementioned challenges of multi-task learning and reward definitions. The phenomenon that the apparently easy tasks for humans, such as pouring water into a cup, are difficult to teach a robot to do, is also known as Moravec's paradox [43].

Considering the aforementioned challenges, we pose the following research questions that we address in this thesis:

- How can we scale robot skill learning so that instead of relying on structured, isolated expert demonstrations for each new skill, we can learn many skills simultaneously from large amounts of unlabeled, unstructured data?
- How can we lessen the requirements for specifying tasks to robots, to move from task-specific engineering of reward functions to more intuitive and natural choices, such as natural language or videos?
- How can we improve the long-horizon reasoning capabilities of robots, to break down abstract, long-horizon tasks into sequences of subgoals?
- How can we integrate semantic and structural priors to improve sample-efficiency and generalization of learned skills?
- How can we scale robot learning systems to autonomously acquire general-purpose knowledge from from purely offline, unstructured data in the real world that allows them to compose long-horizon, multi-tier tasks by following unconstrained language instructions?

In the scope of this thesis, we tackle the aforementioned questions and provide solutions which outperform current state-of-the-art methods. Although the application scenarios that we described are in the context of robot object manipulation in human-centered environments, the principles behind these methods are generally applicable to robots in any environment that need scalable ways of learning multiple skills from uncurated data and/or ground them in language.

1.1 Scientific Contributions

In this thesis, we make several contributions to the challenging problems of skill learning and fusing perception, language and control. In the following we provide an overview of the key contributions presented in this thesis.

Learning Object Placements For Relational Instructions
In Chapter 3, we address the challenge of learning object placements for relational instructions, to enable human-robot interactions such as "place the mug on the right of the box". We contribute an approach coined Spatial-RelNet that estimates pixelwise object placement probabilities for a set of spatial relations from a single input image with a convolutional neural network, without requiring ground truth data for pixelwise relational probabilities or 3D models of the objects. We address the problem of the unavailability of ground-truth pixelwise annotations of spatial relations from the perspective of auxiliary learning. During

training, our network receives the learning signal by classifying hallucinated high-level scene representations as an auxiliary task. To this end, our approach receives the learning signal by classifying hallucinated scene representations as an auxiliary task. Concretely, deep features of objects are implanted into a pretrained auxiliary classifier to compute a posterior class probability over spatial relations. By rearranging deep features, we can reason over what relation would most likely be formed if we placed an object at the given location without modifying the input image. Our results obtained using real-world data and human-robot experiments demonstrate the effectiveness of our method in reasoning about the best way to place objects to reproduce a spatial relation.

Composing Pick-and-Place Tasks By Grounding Language

In Chapter 4, we address the challenge of learning to follow unconstrained language instructions to pick and place arbitrary objects and effectively resolves ambiguities through dialogues. Concretely, we present an approach that infers objects and their relationships from input images and language expressions and can place objects in accordance with the spatial relations expressed by the user. Specifically, by grounding objects and their spatial relations, we allow specification of complex placement instructions, e.g. "place it behind the middle red bowl". We contribute a grounding network that processes language input and visual object candidates detected with an off-the-shelf detector and performs referential expression comprehension. Additionally, it generates referential expressions for each object candidate to disambiguate unclear instructions. Once the reference object of a relative placement instruction has been identified, a second network, the Spatial-RelNet presented in Chapter 3 predicts object placing locations for a set of spatial relations. Ours was the first comprehensive system for controlling robots that allowed tackling temporally more extended tasks by sequentially composing pick-and-place language instructions and grounding object semantics and spatial relations. Our results obtained using a real-world PR2 robot demonstrate the effectiveness of our method in understanding pick-and-place language instructions and sequentially composing them to solve tabletop manipulation tasks.

A Benchmark for Language-Conditioned Policy Learning

Thus far, we have introduced a method for picking-and-placing objects based on language instructions that can solve ambiguities through dialog. However, if we want to command the robot to solve more complex tasks, such as opening a drawer, extending our approach presented in Chapter 4 is not trivial. Towards developing generalist robots, it is not only imperative to ground object semantics and spatial relations, but also to be able to ground a diverse repertoire of robot skills. To this end, we advocate defining skills as being continuous instead of discrete, endowing the agent of task-agnostic control: the ability to reach any reachable goal state from any current state.

To address this issues, in Chapter 5 we present a new open-source simulated benchmark, coined CALVIN, that links human language to robot motor skills, behaviors, and objects in interactive visual environments. In this setting, a single agent must solve complex manipulation tasks by understanding a series of language expressions in a row, e.g., "open the drawer ... pick up the blue block ... push the block into the drawer ... open the sliding door". Furthermore, to evaluate the agents' ability for long-horizon planning, agents in this scenario are expected to be able to perform any combination of subtasks in any order. Our framework has been developed from the ground up to support training, prototyping, and validation of language conditioned policies over a range of four indoor environments. To

establish baseline performance levels, we evaluate an approach that uses relabeled imitation learning to distill reusable behaviors into a language-based goal-directed policy. This is the first public benchmark of instruction following that combines: natural language conditioning, multimodal high-dimensional inputs, 7-DoF continuous control, and long-horizon robotic object manipulation.

Studying What Matters in Language-Conditioned Imitation Learning
While recently substantial advances have been achieved in language-driven robotics by leveraging end-to-end learning from pixels, there is no clear and well-understood process for making various design choices due to the underlying variation in setups. Having now a standardized platform in the form of the CALVIN benchmark presented in Chapter 5, in Chapter 6 we conduct an extensive study of the most critical challenges in learning language conditioned policies from offline free-form imitation datasets. We systematically compare key components of language conditioned imitation learning over unstructured data, such as observation and action spaces, losses for aligning visuo-lingual representations, language models and latent plan representations, and we analyze the effect of other choices, such as data augmentation and optimization. We contribute four improvements to these key components: a multimodal transformer encoder to learn to recognize and organize behaviors during robotic interaction into a global categorical latent plan, a hierarchical division of the robot control learning that learns local policies in the gripper camera frame conditioned on the global plan, balancing terms within the KL loss and a self-supervised contrastive visual-language alignment loss. We integrate the best performing improved components in a unified framework, Hierarchical Universal Language Conditioned Policies (HULC). Our model achieved state-of-the-art results on the challenging CALVIN benchmark at the time of publication.

Learning Robot Skills from Unlabeled Videos
In Chapter 7, we address the key challenges of discovery, representation and reuse of skills from unlabeled videos. Learning robot skills from unlabeled videos is a challenging task, with the potential of unlocking greater robot capabilities by tapping into internet-scale unlabeled data sources, such as YouTube videos. To this end, we propose a novel approach, Adversarial Skill Networks (ASN), to learn a task-agnostic skill embedding space from unlabeled multi-view videos. We combine a metric learning loss, which utilizes temporal video coherence to learn a state representation, with an entropy-regularized adversarial skill-transfer loss. The metric learning loss learns a disentangled representation by attracting simultaneous viewpoints of the same observations and repelling visually similar frames from temporal neighbors. The adversarial skill transfer loss enhances re-usability of learned skill embeddings over multiple task domains. Extensive evaluations demonstrate that given a single video of a previously unseen task, the learned embedding enables training of continuous control policies to solve novel tasks that require the interpolation of previously seen skills.

Task-Agnostic Offline Reinforcement Learning
In Chapter 8 we address the challenge of producing previously unseen combinations of skills, learned from an offline, unstructured dataset, to reach temporally extended goals by "stitching" together skills. To this end, we contribute a novel hierarchical approach, Task-AgnostiC Offline Reinforcement Learning (TACO-RL), that combines the strengths of the imitation learning and reinforcement learning paradigms to learn task-agnostic long-horizon policies from high-dimensional camera observations. Specifically, we combine a low-level

policy that learns latent skills via imitation learning and a high-level policy learned from offline reinforcement learning for skill-chaining the latent behavior priors. By stitching latent plans extracted from unstructured data, our formulation offers the simplicity of imitation from collected play data while offering long-term optimality for sequential multi-tier tasks. This hierarchical approach constitutes a practical solution by decomposing a whole task into smaller chunks of sub-tasks. The high-level policy can learn long-horizon tasks as the effective episode horizon is reduced and it does not need to capture in detail the physics of the world, simplifying the underlying dynamics of the RL agent. Extensive evaluations show an order-of-magnitude improvement in performance upon state-of-the-art baselines on various long-horizon tasks. We even learn one multi-task visuomotor policy for 25 distinct manipulation tasks in the real world which outperforms both imitation learning and offline reinforcement learning techniques.

Learning Affordances from Play Data for Sample-Efficient Policy Learning

In Chapter 9 we address the challenge of discovering object affordances from unstructured data and leveraging them to enable efficient policy learning and motion planning. Knowledge of object affordances helps a robot understand how function: what can be done with each object, where this interaction may occur, and how the object is used to achieve a goal. Affordance learning methods typically require manually segmented annotations to learn visual affordances and are limited in the complexity of the actions they model by relying often on predefined action templates. We contribute a novel approach, Visual Affordance-guided Policy Optimization (VAPO), that learns affordances that are grounded in real human behavior from teleoperated play data by leveraging the gripper's opening and closing signal as a heuristic. Our approach decomposes object manipulation into a sample-efficient combination of model-based planning and model-free reinforcement learning. Aside from accelerating learning, a critical advantage of imbuing robots with an object-centric visual affordance prior is generalization: the learned policy generalizes to unseen, functionally similar, objects because our visual affordance model can anticipate their affordance regions. As play data is not random, but rather structured by human knowledge of object affordances, we find that the affordances discovered from it are functional affordances, priming a robot to approach an object the same way a human would.

Grounding Language with Visual Affordances over Unstructured Data

In Chapter 10 we address the challenge of learning language-conditioned robotic visuomotor skills from unstructured data in the real world in a data-efficient manner and composing them to follow abstract natural language instructions, such as "tidy up the workspace and turn off the lights" with no additional training. To this end, we contribute a novel approach to efficiently learn general-purpose language-conditioned robot skills from unstructured, offline and reset-free data in the real world by exploiting a self-supervised visuo-lingual affordance model, which requires annotating as little as 1% of the total data with language. Specifically, we present Hierarchical Universal Language Conditioned Policies 2.0 (HULC++), a hierarchical language-conditioned agent that integrates the task-agnostic control of HULC presented in Chapter 6 with the object-centric semantic understanding of VAPO presented in Chapter 9 to decompose robot manipulation into semantic and spatial pathways. We show that by extending VAPO to learn language-conditioned affordances and combining it with a 7-DoF low-level policy that builds upon HULC, our method is capable of following multiple long-horizon manipulation tasks in a row, directly from images, while requiring an order of

magnitude less data than previous approaches. We evaluate our method in extensive experiments both in simulated and real-world robotic tasks, achieving state-of-the-art performance on the challenging CALVIN benchmark at the time of publication and learning over 25 distinct visuomotor manipulation tasks with a single policy in the real world. We show that when paired with Large Language Models (LLMs) to break down abstract natural language instructions into subgoals via few-shot prompting, our method is capable of completing long-horizon, multi-tier tasks in the real world.

1.2 Dataset Contributions

Generally, research on various robot learning tasks is facilitated by the availability of standardized datasets, which help reproduce results and serve as a foundation for further research. In the context of this thesis, we published the following publicly available datasets that have thereafter been adopted for standardized benchmarking in many works.

- Relational Placing Dataset: Manually annotated bounding boxes and pairwise spatial relations for hundreds of tabletop scenes with several camera viewpoints.
- Multi-view Block Task Dataset: Unlabeled, synchronized multi-view videos of humans manipulating colored blocks.
- Freiburg Poking Dataset: 40K of random robot pushing data with over 34 distinct objects.
- CALVIN Dataset: 24 hours of unstructured, human teleoperated play data distributed across 4 simulated tabletop environments, with 1% of annotated language instructions. This dataset is the foundation of the CALVIN Benchmark.
- Visual Affordances Dataset: Multimodal robot interaction data in both simulated and real world environments for detecting and learning visual affordances.
- Task-Agnostic Robot Play Dataset: 9 hours of undirected, unstructured, multimodal play data in a complex real world tabletop environment, with less than 1% of annotated language instructions.

1.3 Software Contributions

Together with the availability of datasets, acess to software is equally important to accelerate research. In the context of this thesis, we have open-sourced our implementation of several proposed models and developed software stacks to facilitate future research.

- Spatial Relations: `http://spatialrelations.cs.uni-freiburg.de`
- Self-supervised 3D Shape: `https://github.com/mees/self-supervised-3D`
- Adversarial Skill Networks: `http://robotskills.cs.uni-freiburg.de`
- CALVIN Benchmark: `http://calvin.cs.uni-freiburg.de`

- HULC Model: `http://hulc.cs.uni-freiburg.de`
- Visual Affordances: `http://vapo.cs.uni-freiburg.de`
- TACO-RL: `http://tacorl.cs.uni-freiburg.de`
- Visuo-lingual Affordances: `http://hulc2.cs.uni-freiburg.de`
- Visual Language Maps: `https://vlmaps.github.io`

1.4 Experimental Robot Platforms

In this thesis, various experiments are conducted using different robotic platforms to demonstrate the applicability of our models in real-world scenarios. This section gives an overview of the robots and setups used, with some being developed by the author during the course of this PhD.

PR2

In our research, we utilize the PR2 (Personal Robot 2), a service robot developed by Willow Garage and released in 2010 for navigation and manipulation in human-centered environments, see Fig. 1.1. The robot is a highly advanced and capable robot designed for research and development that can be used for mobile manipulation tasks, such as tidying up tabletops or bringing a beer from a fridge.

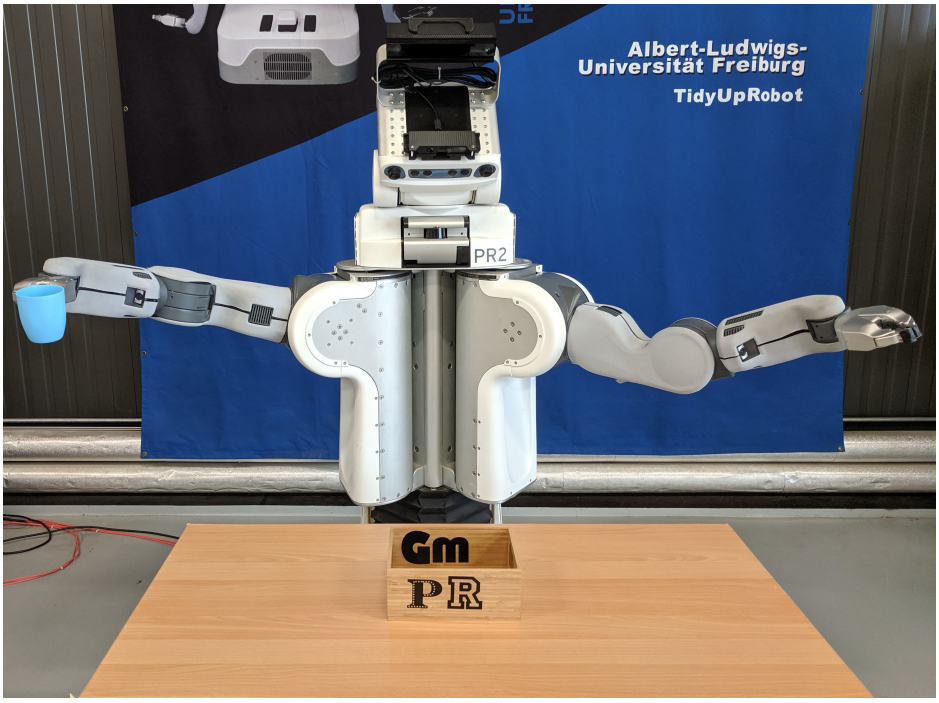


Figure 1.1: The PR2 mobile manipulation robot that we use in our experiments. The robot has two arms and a mounted Kinect 2 RGB-D camera on its head.

The PR2 is equipped with two 7-DoF arms, each with a gripper that can switch between opened and closed states and is compliant due to built-in mechanical springs. The robot

also has a telescoping spine for adjusting the height of the upper body, a pan tilt head with multiple onboard RGB cameras, a tilting laser rangefinder and a mounted Kinect 2 camera that provides RGB and depth images at a frame rate of 30 Hz and a resolution of 1920x1080 pixels. These sensors provide the robot with a detailed understanding of its environment. The PR2 runs the open-source Robot Operating System (ROS) middleware to handle communications between programs.

We primarily used the PR2 for all experiments presented in Chapter 3 and Chapter 4 for the evaluation of the robot's ability to follow language based pick-and-place instructions, understand spatial relations between objects and to identify and resolve ambiguities.

Franka Panda

The Franka Panda is a highly versatile and adaptable robotic arm designed by Franka Emika for use in research settings. The robot, which was released in 2017, has a payload capacity of 3 kg, a 850 mm reach and a repeatability of 0.1 mm. It features a lightweight design, with a total weight of only 18 kg, making it easy to move and deploy in various applications. The Franka Panda can be easily integrated into different systems and environments using a variety of interfaces, including the Robot Operating System (ROS) and Python programming language. This makes the robot highly adaptable and easy to program for a wide range of tasks such as pick and place, assembly, testing, and research applications.

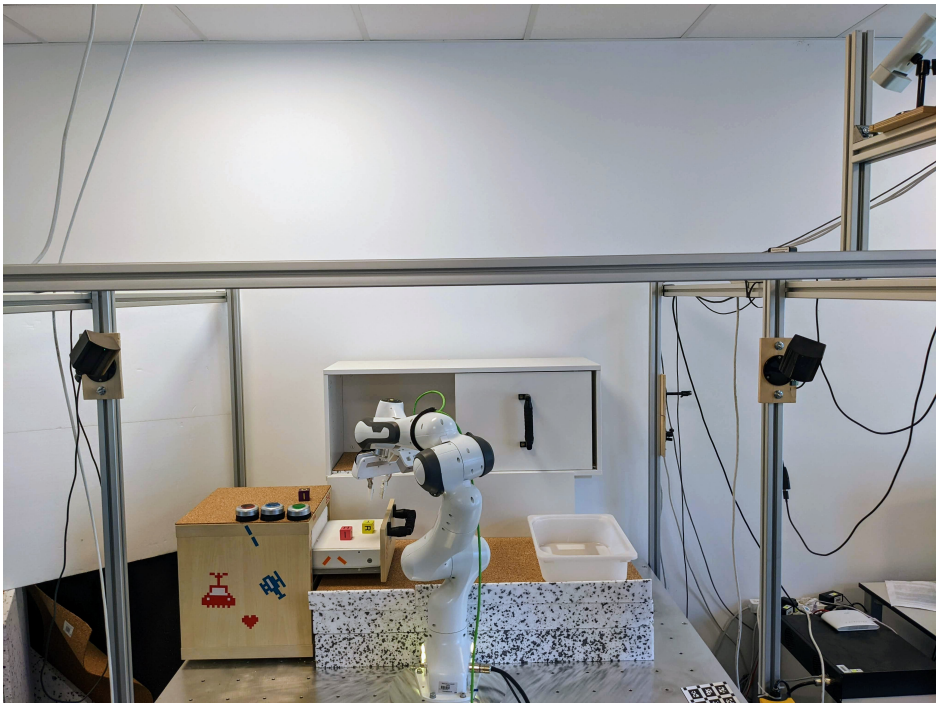


Figure 1.2: The Franka Panda manipulation robot that we use in our experiments for learning to pick-and-place objects, open drawers or moving sliding doors. We developed a custom steel and aluminium made table for it and an alumium profile based construction that allows for a flexible placement of sensors.

In order to allow flexible setups to cater to the needs of different users, we developed a custom made table made of steel and aluminium, with holes every 10 cm that allows screwing the robot at different positions on the table, see Fig. 1.2. The table is 70 cm high, 160 cm wide and 120 cm long. Due to the robustness and weight of the table, the table

can not be damaged by applications that might exert high vibrations upon it, such as online reinforcement learning. We attach an aluminium profile based construction on top of the table that allows for a flexible placement of different sensors. In our case, we attach a Microsoft Kinect Azure RGB-D camera to perceive the scene as a static camera and VR towers to track a VR motion controller that remote controls the robot. Additionally, we integrate a FRAMOS Industrial Depth Camera D435e as a RGB-D gripper camera.

We primarily used the PR2 for all experiments presented in Chapter 8, Chapter 9 and Chapter 10 for the evaluation of the robot's ability to learn a wide range of visuomotor control policies from unstructured data and reason about temporally extended tasks.

1.5 Publications

Major parts of the work presented in this thesis have undergone international peer review. In the following, we list the corresponding publications in chronological order.

- O. Mees*, A. Emek*, J. Vertens, and W. Burgard, "Learning Object Placements For Relational Instructions by Hallucinating Scene Representations". In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

- O. Mees*, M. Merklinger*, G. Kalweit, and W. Burgard, "Adversarial Skill Networks: Unsupervised Robot Skill Learning from Video". In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2020. **Finalist for the Best Paper Award in Cognitive Robotics**.

- O. Mees, and W. Burgard, "Composing Pick-and-Place Tasks By Grounding Language". In *Proc. of the International Symposium on Experimental Robotics (ISER)*, Springer Proceedings in Advanced Robotics book series (SPAR, volume 19), ISBN: 978-3-030-71151-1, 2020.

- O. Mees*, L. Hermann*, Erick Rosete-Beas, and W. Burgard, "CALVIN: A Benchmark for Language-Conditioned Policy Learning for Long-Horizon Robot Manipulation Tasks". *IEEE Robotics and Automation Letters (RAL)*, doi: 10.1109/LRA.2022.3180108, 2022.

- O. Mees*, L. Hermann*, and W. Burgard, "What Matters in Language Conditioned Robotic Imitation Learning over Unstructured Data". *IEEE Robotics and Automation Letters (RAL)*, doi: 10.1109/LRA.2022.3196123, 2022.

- J. Borja-Diaz*, O. Mees*, G. Kalweit, L. Hermann, J. Boedecker, and W. Burgard, "Affordance Learning from Play for Sample-Efficient Policy Learning". In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

- O. Mees, and W. Burgard, "Language-Conditioned Policy Learning for Long-Horizon Robot Manipulation Tasks". In *Proc. of the RSS Pioneers Workshop at Robotics: Science and Systems (RSS)*, 2022.

*Denotes equal contribution

- E. Rosete-Beas*, O. Mees*, G. Kalweit, J. Boedecker and W. Burgard,"Latent Plans for Task-Agnostic Offline Reinforcement Learning". In *Proc. of the 6th Conference on Robot Learning (CoRL)*, 2022.

- O. Mees*, J. Borja-Diaz* and W. Burgard,"Grounding Language with Visual Affordances over Unstructured Data". In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

The following publications of the author of this thesis are related to the work in this thesis, but are outside of its scope.

- O. Mees, A. Eitel, and W. Burgard, "Choosing smartly: Adaptive multimodal fusion for object detection in changing environments. In *IIEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

- O. Mees, N. Abdo, M. Mazuran, and W. Burgard, "Metric Learning for Generalizing Spatial Relations to New Objects". In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

- W. Burgard, A. Valada, N. Radwan, T. Naseer, J. Zhang, J. Vertens, O. Mees, A. Eitel and G. Oliveira. Perspectives on Deep Multimodel Robot Learning. In *Proc. of the International Symposium on Robotics Research (ISRR)*, 2017.

- O. Mees, M. Tatarchenko, T. Brox, and W. Burgard, "Self-supervised 3D Shape and Viewpoint Estimation from Single Images for Robotics". In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

- I. Nematollahi, O. Mees, L. Hermann, and W. Burgard, "Hindsight for Foresight: Unsupervised Structured Dynamics Models from Physical Interaction". In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

- C. Huang, O. Mees, A. Zeng, and W. Burgard, "Visual Language Maps for Robot Navigation". In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

1.6 Collaborations

This thesis work was done in collaboration with other researchers from the University of Freiburg. Prof. Wolfram Burgard was the supervisor of this thesis and therefore, contributed through scientific discussions. The collaborations beyond this supervision are outlined below.

- Chapter 3: The initial work on the Spatial-RelNet architecture for estimating pixelwise object placement heatmaps was formulated in collaboration with Johan Vertens for Alp Emeks Master's thesis, which the author of this thesis supervised together with Johan Vertens. Alp Emek helped with collecting and labeling the dataset and implemented the initial experimental framework. All the results for Spatial-RelNet reported in this thesis were entirely carried out by the author of this thesis. The paper was mostly written by the author of this thesis.

- Chapter 5: I developed the main idea of the paper, implemented the initial simulated environments and baselines, training code and wrote most of the paper. Lukas Hermann implemented most of the robot control code and the evaluation of the policies and contributed speed improvements for loading the dataset. We carried out the simulated experiments jointly. Erick contributed the final 3D models for the simulated environments.

- Chapter 6: I developed the main idea of the paper, contributed the idea of using a multimodal transformer encoder, the KL divergence balancing, the self-supervised contrastive visual-language alignment loss and using discrete categorical latent plans. Lukas Hermann contributed the idea of using relative actions in the gripper frame and improvements to the model's training time. All the results for HULC reported in this thesis were entirely carried out by the author of this thesis. The paper was mostly written by the author of this thesis.

- Chapter 7: The initial work on the Adversarial Skill Networks (ASN) for learning robot skills from unlabeled videos was formulated in collaboration with Gabriel Kalweit for Markus Merklingers Master's thesis, which the author of this thesis supervised together with Gabriel Kalweit. Markus Merklinger implemented the initial experimental framework and we jointly collected the dataset. The insights gained during the aforementioned thesis supervision influenced the subsequent improved implementations of ASN that the author of this thesis carried out. All the results for ASN reported in this thesis were entirely carried out by the author of this thesis. The paper was mostly written by the author of this thesis. Markus Merklinger and Gabriel Kalweit also contributed to the paper writing.

- Chapter 8: The main idea to use offline reinforcement learning with latent skills was developed in collaboration with Gabriel Kalweit. Erick Rosete-Beas implemented the initial experimental framework and we jointly collected the dataset. I contributed the implementations for learning the low-level level policies with latent skills, the PLayLMP baseline, the framework to collect data via teleoperation and perform experiments in both simulation and in the real world. Erick additionally helped design and implemented the high-level policy, baseline implementations and performed the experiments in the CALVIN environment. All the real world experiments reported in this thesis were entirely carried out by the author of this thesis. Gabriel additionally provided consultation on the design of the experiments. All authors contributed to the paper writing. Joschka Boedecker and Wolfram Burgard provided general consultation.

- Chapter 9: I developed the idea for the paper, devised the theoretical framework and wrote most of the paper. Jessica Borja-Diaz implemented the initial experimental framework and we jointly collected the dataset and carried out the experiments for the paper. Gabriel Kalweit provided consultation with initial experiments, and contributed to the conceptualization of the idea and writing of the paper. Lukas Hermann provided consultation for the real world robot experiments and implemented the low-level control of the robot. Joschka Boedecker and Wolfram Burgard provided general consultation.

- Chapter 10: I developed the main idea for the paper, devised the theoretical framework and wrote most of the paper. Jessica Borja-Diaz implemented the initial experimental

framework for learning visuo-lingual affordances and we jointly collected the dataset. I contributed the implementations for translating abstract language input into sequences of subgoals with Large Language Models, learning the language-conditioned low-level level policies with latent skills, the framework to collect data via teleoperation and perform experiments in both simulation and in the real world. Jessica additionally contributed the experiments in the CALVIN environment. All the real world experiments reported in this thesis were entirely carried out by the author of this thesis.

Chapter 2

Background

In this chapter, we briefly describe some of the basic concepts and theoretical foundations for the methods presented later in this thesis. First, we give an overview on different paradigms for learning robot skills with machine learning. This overview serves as an introduction to the field and defines the general setting of the following research results. Furthermore, we describe challenges and methods for learning policies from fixed static datasets that serve as the foundation for many of the approaches described in this thesis. Details of the specific implementation are given in the respective chapters. Finally, we discuss different ways to encode natural language with deep neural networks.

2.1 Learning Sensorimotor Behavior

Generally, the problem of learning a behavior for an agent is considered the task of learning a policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$, mapping from the environment's state to actions of a dynamical system that elicit some desired behavior [9]. Thus, the policy reacts with an appropriate action to every new observation to control an agent, such as a robot. We will denote $\pi_\theta(a_t \mid s_t)$ as the distribution over actions under the policy conditioned on the state. The policy is represented by a set of parameters θ, which may take the form of weights of a neural network. We consider a system with $\mathcal{S}$ being the state, $\mathcal{A}$ the actions an agent can take and $\mathcal{O}$ the observations from the environment. For example, $\mathcal{S}$ might contain the position and orientation of objects in the environment or the joint angles and positions and velocities of the robot's links. Regarding the dynamics, given that applying an action a_t causes a transition into some markovian state $s_{t+1} \in \mathcal{S}$, the state evolves through time according to $p(s_{t+1} \mid s_t, a_t)$. Neither the system dynamics $p(s_{t+1} \mid s_t, a_t)$ nor the observation distribution $p(o_t \mid s_t)$ are assumed to be known in general. We denote $\tau = \{(s_0, a_0), \ldots, s_t, a_t)\}$ as a sequence of state-action pairs from a task demonstration with a finite horizon $t \in [1, \ldots, H]$. The policy and the system dynamics induce the distribution over the trajectories:

$$\rho_\pi(\tau) = p(s_0) \prod_{t=0}^{H} \pi_\theta(a_t \mid s_t) p(s_{t+1} \mid s_t, a_t) \tag{2.1}$$

For the task we are considering, we denote $\mathcal{L}(s_t, a_t)$ the expected cost of performing action a_t in s_t. Thus, the overall objective is to minimize the expectation given by

$$\mathbb{E}_{\pi_\theta(\tau)} \left[\sum_{t=0}^{H} \mathcal{L}(s_t, a_t) \right]. \tag{2.2}$$

Conventionally, a simple way to learn sensorimotor behaviors is by learning from demonstrations [44]. This involves the following steps. First, one has to define a task, such as opening a drawer. Second, collecting a large number of labeled and segmented expert demonstrations, either by manually moving the robot joints (*i.e.* kinesthetic teaching) or teleoperating the robot to execute the desired task. Finally, the policy is trained with supervised learning. Assuming a unimodal Gaussian distribution for $\pi_\theta(a_t \mid s_t)$, the parameters θ can be found for instance by minimizing a mean squared error loss:

$$\theta^* = \arg\min_\theta \sum_{t=0}^{H} || \ a_t - \pi(s_t; \theta) \ ||^2 \tag{2.3}$$

Despite its simplicity, this paradigm will struggle to achieve good long-horizon performance, because a small mistake on the part of the policy or system noise will place the agent into states that are outside of the distribution of the training data, leading to compounding errors [45].

As aforementioned, the state of the system is usually unknown and the robot's sensor observations are a stochastic process produced from the state. Thus, the representation of the state has a large influence over the sample complexity and the robustness of the policies. Traditionally, hand-engineered low-dimensional representations of observations have been used as state representations, which are more tractable and sample-efficient [46]. However, computer vision systems are often unable to estimate the state variables of the model accurately and it has proven non-trivial for downstream control systems to be robust to such errors [47, 48]. One way to alleviate this problem is end-to-end learning from pixels, which in recent years has seen tremendous success by *jointly* learning a suitable state representation and control from the environments raw perceptual observations [8]. Such model-free methods enjoy a high flexibility due to the minimal assumptions required about the state representation of the world. The caveat is the poor sample complexity, as important state information needs to be inferred from raw observations, such as the location of objects, in addition to the control. We refrain from a detailed explanation of deep neural networks and refer to Goodfellow *et al.* [49].

2.1.1 Learning Multiple Behaviors via Imitation

Developing robots that can perform multiple tasks are a long-standing goal. In theory, learning multiple tasks jointly promises to achieve greater efficiency and performance than solving tasks individually by discovering and exploiting shared structure across tasks [13]. In practice, multi-task settings present a number of optimization challenges, making it difficult to realize large efficiency gains and a positive transfer compared to learning tasks independently [14, 15]. These challenges of achieving greater efficiency and performance have been observed in both the computer vision community [50], as well as in the robot learning community [42, 51].

A natural extension of the sensorimotor policy to multiple tasks is to introduce a secondary variable g from a distribution $p(g)$ that conditions the policy $\pi : \mathcal{S} \times \mathcal{G} \rightarrow \mathcal{A}$. Thus, instead of communicating how to perform a task via observation-action pairs, this allows a more flexible specification of what needs to be done, by considering the conditional variable to be a *goal* state [52]. In its most simple form, this variable can take the form of an one-hot vector that selects the corresponding task to execute from a set of pre-defined discrete tasks [12, 19].

Alternative formulations obtain the latent variable from the observations of the desired world states, such as goal images [53] or from encodings of natural language instructions [54]. In this thesis, we use both goal images and language instructions extensively. While conditioning the policies on a goal offers a greater flexibility, achieving good long-horizon performance can be challenging due to the highly multimodal action distribution that can reach a temporally distant goal. As a consequence, goal-conditioned policies $\pi_\theta(a_t \mid s_t, s_g)$ trained via imitation have been studied for reaching short-horizon behaviors, such as pushing [53] or short-distance navigation [55].

Learning Behaviors from Play Data

In order to generalize to a wide range of tasks at test time, the agent should intuitevely be exposed to a diverse set of (s_t, s_g) pairs during training, alongside the actions that connect both states. Randomly exploring the state space is challenging to automate for complex skills in a safe way. Moreover, the likelihood of performing complex tasks and demonstrating interesting behaviors is low. For example, skills that emerged from random exploration strategies in cluttered bins, were shown to be limited to pushing behaviors, not covering complex skills like grasping [56]. Therefore, undirected "play data" has recently emerged as a solution to obtain datasets that are extensive in scope and dense in its coverage of the environment's interaction space [18]. The key idea is to collect data by asking humans to teleoperate a robot without any specific tasks in mind. Thus, unlike expert demonstrations, in this setting an operator engages in a behavior that satisfies their own curiosity. As users naturally interact with the environment, the need for resetting the scene to initial states is eliminated, making it a scalable and inexpensive form to collect diverse behaviors. Due to its scalability and diversity, unstructured play data serves as an ideal foundation for developing robot systems that can perform a wide range of useful tasks, as demonstrated in the approaches outlined in Chapter 5, Chapter 6, Chapter 8, Chapter 9 and Chapter 10.

To learn control, this long temporal state-action stream $\mathcal{D} = \{(s_t, a_t)\}_{t=0}^{\infty}$ is relabeled [57], treating each visited state in the dataset as a "reached goal state", with the preceding states and actions treated as optimal behavior for reaching that goal. Relabeling yields a dataset of $D_{\text{play}} = \{(\tau, s_g)_i\}_{i=0}^{D_{\text{play}}}$, where each goal state s_g has a trajectory demonstration $\tau = \{(s_0, a_0), \ldots, (s_t, a_t)\}$ solving for the goal. These short horizon goal image conditioned demonstrations can be used in a maximum likelihood imitation objective:

$$\mathbb{E}_{(\tau, s_g) \sim D_{\text{play}}} \left[\sum_{t=0}^{|\tau|} \log \pi_\theta(a_t \mid s_t, s_g) \right] \tag{2.4}$$

In order to model the multimodality, Lynch *et al.* [18] introduced a hierarchical latent variable model, by auto-encoding contextual demonstrations through a latent "plan" space with a sequence-to-sequence conditional variational auto-encoder (seq2seq CVAE), which we visualize in Figure 2.1. Conditioning the policy on the latent plan frees up the policy to use the entirety of its capacity for learning uni-modal behavior. The latent plans z can be generated by making use of the variational inference framework [58]. The objective of the latent plan sampler is to model the full distribution over all high-level behaviors that might connect the current and goal state, to provide multi-modal plans at inference time. This distribution is learned with a CVAE by maximizing the marginal log likelihood of the

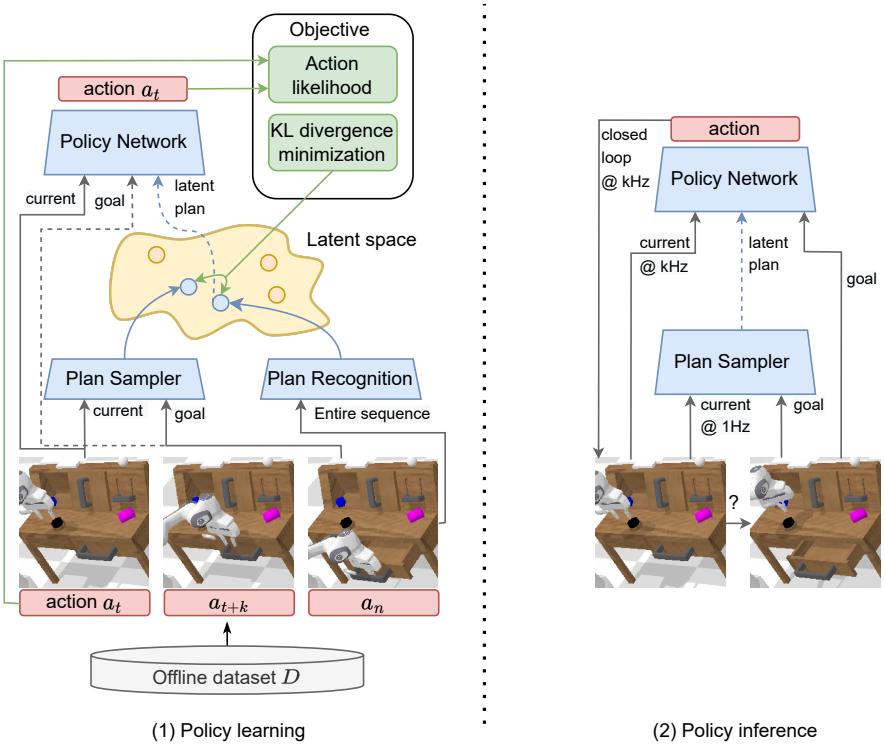


Figure 2.1: Overview of an approach to learn control from play data via imitation [18]. During training a stochastic plan sampler that receives the first and the last frame of a randomly sampled window learns to mimic the latent plans generated by a plan recognition encoder that receives the full sequence as input with a Kullback–Leibler (KL) divergence loss. The policy is trained on reconstructing the actions observed in the window and is conditioned on the current state, the goal state and the sampled latent plan from the plan recognition encoder. At test time only the plan sampler is used to generate latent plans given the actual state and a goal state that is specified by the user.

observed behaviors in the dataset $\log p(x \mid s)$, where x are sampled state-action trajectories from τ. The Evidence Lower Bound (ELBO) [58] for the CVAE can be written as:

$$\log p(x|s) \geq -\text{KL}(q(z|x,s) \mid\mid p(z|s)) + \mathbb{E}_{q(z|x,s)}\left[\log p(x|z,s)\right] \tag{2.5}$$

The decoder is a policy trained to reconstruct input actions, conditioned on state s_t, goal s_g, and an inferred plan z for how to get from s_t to s_g. At test time, it takes a goal as input, and infers and follows plan z in closed-loop. Variants of this formulation are used in the approaches described in Chapter 5, Chapter 6, Chapter 8 and Chapter 10.

2.1.2 Reinforcement Learning

Reinforcement learning (RL) constitutes an alternative paradigm for solving Sequential Decision Making (SDM) problems, which are tasks that require several consecutive decision steps. In its most general form, it learns from an agents trial and error in an environment and tries to maximize the sum of future rewards [59]. Concretely, at each time step the agent receives a scalar value, referred to as the reward r_t, from the environment. The design of the rewards is such that maximization of them leads to achieving the desired goal. In the drawer

opening task, suppose the agent receives a reward of 1 if it successfully opens the drawer and 0 otherwise. In the initial stages of training, the agent's parameters, θ, are typically set to random values, leading to the agent performing random actions. This can result in a lack of rewards being received by the agent until it happens to stumble upon the desired sequence of actions by chance. This phenomenon is referred to as the exploration problem. While in simulated environments agents trained with RL can accumulate large numbers of interactions to achieve success, the cost and safety issues of exploration with real robot platforms makes this a key challenge for the deployment of reinforcement learning agents in the real world. In this thesis, we propose a range of solutions to overcome this issue, such as learning rewards from videos in Chapter 7, leveraging human knowledge of object affordances in Chapter 9 or leveraging offline RL in Chapter 8.

As a sequential decision problem, reinforcement learning is commonly framed as a Markov Decision Process (MDP) $M = (\mathcal{S}, \mathcal{A}, \mathcal{T}, r, \mu_0, \gamma)$. Like before, $\mathcal{S}$ and $\mathcal{A}$ denote the state space and action space respectively. Similarly, as the system dynamics are unknown, $\mathcal{T}(s_{t+1}|s_t, a_t)$ represents the probability of transitioning from state s_t to state s_t when applying action a_t. The reward $r(s_t, a_t)$ is received by an agent for executing action a_t in state s_t and μ_0 is the initial state distribution. Concretely, the sum of rewards for a given fixed trajectory is called the return, denoted by G_t is given by

$$G_t = \sum_{k=0}^{H} \gamma^k r_{t+k+1}, \tag{2.6}$$

where the discount factor $\gamma \in (0, 1)$ weights the importance of long- versus short-term rewards. Concretely, a value close to 0 places higher importance on immediate rewards, wheres a value of 1 prioritizes future rewards. Besides, a discount factor under 1 prevents the return from becoming infinite in case of $H \to \infty$ horizon.

The trajectory distribution is the same as in Equation 2.1. The goal in RL is therefore to optimize a policy $\pi(a_t|s_t)$ that maximizes the expected discounted return:

$$\mathbb{E}_{\pi,\mu_0,\mathcal{T}} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right]. \tag{2.7}$$

We can recover a near-optimal policy by accurately estimating the state or state-action value function. The state-value function $V_\pi(s_t)$ represents the expected return when starting in state s_t and following the policy π thereafter for a sequence of length H[1]:

$$V_\pi(s_t) = \mathbb{E}_{\tau\sim\rho_\pi(\tau|s_t)} \left[\sum_{t'=t}^{H} \gamma^{t'-t} r(s'_t, a'_t) \right]. \tag{2.8}$$

The action-value function $Q_\pi(s_t, a_t)$ denotes the expected return when starting in state s_t, selecting action a_t, and then following π thereafter

$$Q_\pi(s_t, a_t) = \mathbb{E}_{\tau\sim\rho_\pi(\tau|s_t,a_t)} \left[\sum_{t'=t}^{H} \gamma^{t'-t} r(s'_t, a'_t). \right] \tag{2.9}$$

[1]We note that we use prime notation for infinite horizon settings and time subscripts for finite horizon settings.

Therefore, we can express the action-value function in terms of the state-value function as follows:

$$Q_\pi(s_t, a_t) = r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim T(s_{t+1}|s_t, a_t)} \left[V_\pi(s_{t+1}) \right]. \tag{2.10}$$

Likewise, we can also express the state-value function in terms of the action-value function:

$$V_\pi(s_t) = \mathbb{E}_{a_t \sim \pi(a_t|s_t)} \left[Q_\pi(s_t, a_t) \right]. \tag{2.11}$$

These definitions allow a recursive formulation of the state-action value function by using the Bellman expectation equation:

$$Q_\pi(s_t, a_t) = r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim T(s_{t+1}|s_t, a_t), a_{t+1} \sim \pi(a_{t+1}|s_{t+1})} \left[Q_\pi(s_{t+1}, a_{t+1}) \right] \tag{2.12}$$

However, the goal in RL is to find an optimal policy π^*, which for a finite MDP is $\pi(a_t|s_t) = \arg\max_{a_t} Q(s_t, a_t)$. In order to find the optimal policy, many methods use the Bellman optimality equation [60], which for the state-action value function leads to a popular algorithm named Q-learning [61]:

$$Q^\star(s_t, a_t) = r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim T(s_{t+1}|s_t, a_t)} \left[\max_{a_{t+1}} Q^\star(s_{t+1}, a_{t+1}) \right]. \tag{2.13}$$

Deep Reinforcement Learning

Deep Reinforcement Learning is a subfield of Reinforcement Learning (RL) that combines the power of deep learning with the framework of RL to scale to higher dimensional domains. Concretely, a neural network is used as a function approximator to represent the Q-function or the policy. Deep Q-networks (DQN) [62] is considered one of the pioneering works in the field, by learning to play Atari games from raw image observations with a variant of Q-learning.

The vanilla Q-learning formulation performs updates in an online fashion, immediately after an experience tuple (s_t, a_t, s_{t+1}, r_t) gets available. However, gradient based optimization methods typically assume the training data to be independent and identically distributed (i.i.d.) and when the agent interacts with the environment the collected sequence of experience tuples can be highly correlated. Intuitively, the collected experience at a timestep influences the experiences that the agent will encounter in the future. Thus, even small updates can have cascading effects in the resulting data distribution and can cause the policy to diverge catastrophically. Imitation learning does not suffer from this issue, as the underlying training data is unaffected by the learning of the policy due to its inherent offline nature. To mitigate this issue Mnih *et al.* [62] introduced experience replay, which stores a large collections of past experience in a buffer and samples experiences randomly from it. Thus, besides breaking harmful correlations, this allows to learn more effectively from individual experience tuples by replaying them multiple times and additionally helps recalling rare experiences.

A second challenge when training Deep Q-Networks comes from the correlation between the estimated value and the target value for the estimation. Concretely, the Bellman equation states that we can compute the current $Q(s_t, a_t)$ value dependent on the Q value of the next state, $r(s_t, a_t) + \gamma Q(s_{t+1}, a_{t+1})$ and the current parametrization of the network. Thus, an update for the current $Q(s_t, a_t)$ value can indirectly alter the values for $Q(s_{t+1}, a_{t+1})$ causing instability in the learning and an increased possibility of oscillation or divergence. In order

to account for moving targets and their inherent non-stationarity, modern implementations use a *delayed target network*. Typically, this target network is either updated slowly via Polyak-Ruppert averaging [63, 64] or periodically, after a fixed number of time steps.

The application of experience replay and target networks within the DQN framework yielded significant advancements in the field of reinforcement learning. Specifically, it demonstrated that utilizing raw pixel inputs and game scores as a reward signal, the DQN algorithm was capable of achieving human-level performance on a diverse range of 2D Atari games, without any prior domain knowledge about the games.

Algorithms for Continuous Action Spaces

Despite the success of DQN, a limitation for its applicability to learn robot skills is that it is designed to operate with discrete action spaces. This stems from the fact that the maximization of the Q-function for target calculation with continuous actions leads to an optimization problem infeasible to solve for every update. As we consider skills to be continuous rather than discrete, in this thesis we have worked with extensions that are suitable for continuous action spaces.

Policy Gradient Methods: In Chapter 7, we use Proximal Policy Optimization (PPO) [65], which tackles the challenge of continuous action spaces on the basis of the Policy-Gradient theorem [66]. Unlike implicit policy learning via Q-learning, policy gradient methods directly optimize $\pi(a_t|s_t)$ in Equation 2.1.2, by parametrizing the policy with the weights θ of a neural network and taking gradient steps. As policy gradient methods sample from a probability distribution, the policy network outputs logits over the actions $\mathcal{A}$. A general form of a policy gradient can be expressed as:

$$\nabla_\theta J(\pi_\theta) = \mathbb{E}_{\tau\sim\rho_{\pi_\theta}(\tau)} \sum_{t=0}^{H} \gamma^t \nabla_\theta \log \pi_\theta(a_t|s_t) \sum_{t'=t+1}^{H} \gamma^{t'-t-1} r(s_{t'}, a_{t'}) - b(s_t), \tag{2.14}$$

where $b(s_t)$ is a baseline function. As the return is commonly estimated with Monte Carlo samples [67], the main purpose of the baseline function is to reduce the variance of the gradient estimates used to update the policy, making the training process more stable. A common example of a baseline function is the value function $V(s_t)$, which represents the expected return when starting in state s_t and following the policy π thereafter. By subtracting the value function from the cumulative reward, the agent can focus improving over the average policy return estimates, which also improves sample-efficiency. A practical challenge when training policy gradient methods is, that a small change in the parameter space of the policies can have significant differences in performance. This makes it risky to use large step sizes when updating the policy, which can negatively impact the sample efficiency of the algorithm. To prevent his, modern methods, such as TRPO [68] and PPO [65], implement therefore constraints on how much the policy can change from the previous iteration, via a KL divergence $D_{KL}\left(\pi(a_t|s_t)||\pi_{old}(a_t|s_t)\right) < \epsilon$ on the policy distribution for instance. However, the need for new samples to be collected for each gradient step, makes on-policy methods, such as PPO, less sample-efficient compared to off-policy methods that are able to reuse prior experience.

Actor-Critic Methods: Actor-critic methods combine the ideas of Q-learning and Policy Gradient methods by parametrizing both a policy $\pi_\theta(a_t|s_t)$ and a value function $V_\phi(s_t)$ or $Q_\phi(s_t, a_t)$. Instead of using the value function as a baseline, actor-critic methods bootstrap

the target estimates with it. Unlike Q-learning, which directly attempts to learn the optimal Q-function, actor-critic methods aim to learn the Q-function corresponding to the current parameterized policy $\pi_\theta(a_t|s_t)$ as follows:

$$Q_\pi(s_t, a_t) = r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1}\sim T(s_{t+1}|s_t,a_t), a_{t+1}\sim\pi_\theta(a_{t+1}|s_{t+1})} \left[Q_\pi(s_{t+1}, a_{t+1}) \right]. \tag{2.15}$$

Thus, alternating between *policy evaluation*, computing the value function for a policy, and *policy improvement*, using the value function to obtain a better policy, the agent ideally converges to a near-optimal policy. In Chapter 9, we use Soft Actor-Critic (SAC) [69] to teach a real Franka Panda robot manipulation skills. SAC is an actor-critic variant that optimizes a stochastic policy in an off-policy fashion, and augments the expected reward in the objective with the entropy of the policy.

$$\pi^\star = \arg\max_\pi \sum_t \mathbb{E}_{(s_t,a_t)\sim\rho_\pi} \left[r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot|s_t)) \right] \tag{2.16}$$

Thus, the actor has the objective of maximizing both the entropy and the expected return of the policy. This allows the policy to explore more extensively, and thus, increases the chances of finding better solutions for the chosen task by balancing the exploration-exploitation trade-off. Further, the entropy regularization helps to prevent the policy from prematurely converging to sub-optimal solutions. Modern implementations automatically tune the α parameter, which weights the entropy term in relation to the return. At the beginning of the training it is set to a high value, which allows for increased exploration. As the training progresses, the temperature is gradually adapted via gradient descent, which shifts the policy's focus towards exploitation. These changes improve the stability, sample-efficiency and overall robustness of the agent, making SAC a popular choice for a wide ranges of problems, such as games and robotics control problems.

2.1.3 Offline Reinforcement Learning

A drawback of having no prior knowledge about the domain or the task, is that model-free online RL methods require a high number of samples for training. While feasible to scale in simulated environments, robot interaction data in real world settings is scarce and costly. Moreover, evaluating sub-optimal policies at early stages of a training or collecting data with random exploration are notoriously difficult to automate in ways that ensure the safety of both the robot and the physical environment. Consequently, there exists a large interest in applying reinforcement learning in a completely *offline* fashion, with a fixed dataset of transitions $\mathcal{D}$ collected by some potentially unknown behavior policy π_β. We visualize the different reinforcement learning settings in Figure 2.2. Unlike imitation learning, which is limited by the quality of π_β, an appealing property of offline RL is its ability to effectively use all the behaviors in $\mathcal{D}$ to produce unseen combinations of skills. Hence, it effectively leverages possibly sub-optimal data and generalizes over naive imitation learning. Furthermore, offline RL has the potential of acquiring new skills without the need of explicit demonstrations, if the new skills can be assembled from parts of previously learned behaviors.

Unfortunately, the main benefit of offline RL, the lack of environment interaction, is also what raises several algorithmic challenges [70]. An obvious challenge is that $\mathcal{D}$ needs to have a *good state coverage*. As exploration is not possible, if $\mathcal{D}$ does not contain transitions

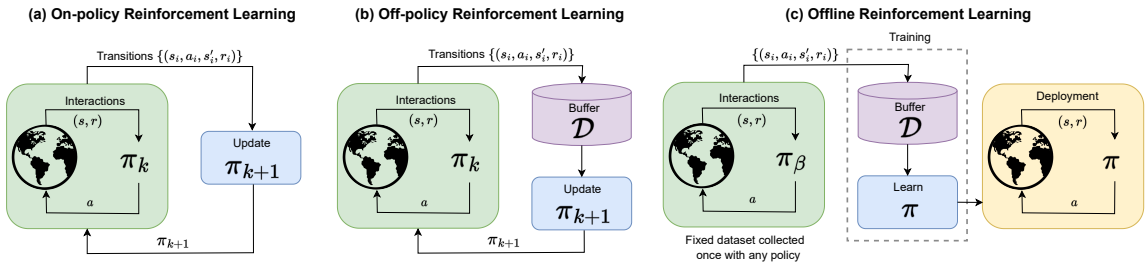


Figure 2.2: Illustration of the various reinforcement learning settings, including (a) online RL, (b) off-policy RL, and (c) offline RL. In the classic online setting (a) the policy π_k is updated using data collected by π_k itself. In the off-policy setting (b), the agent's experience is stored in a replay buffer $\mathcal{D}$. Offline reinforcement learning (c) uses a fixed dataset $\mathcal{D}$ that is collected once by a (often unknown) behavior policy π_β and the policy is deployed without further environment interactions. Adapted from: Levine *et al.* [70].

that lead to high-reward regions of the state space, it might be impossible to discover those regions. Thus, we build upon diverse play data, as described in Section 2.1.1, to train offline RL policies in Chapter 8.

Algorithmically, it may be intuitive to apply commonly used reinforcement learning methods [69, 71] directly in an offline reinforcement learning setting by utilizing the dataset $\mathcal{D}$ as the initial replay buffer and forgoing any additional environmental interactions. Algorithms designed with this idea in mind, such as QT-OPT [72], have been shown to perform poorly without any additional on-policy interaction. The use of high-dimensional, expressive function approximators in offline reinforcement learning exacerbates the issue of *distributional shift*, which is one of the key challenges in offline RL. Intuitively, in offline RL we want to learn a policy that is better than the unknown behavior π_β that produced the dataset $\mathcal{D}$. Therefore, the agent needs to execute and compose action sequences in different ways to the behaviors seen in $\mathcal{D}$, which leads to the agent visiting different states than the ones seen in the training set. In other words, the agent is trained under one distribution, but gets evaluated on a different distribution.

Distributional shifts can harm any policies performance in MDPs. In their seminal paper, Ross *et al.* [45] discovered that for a policy $\pi(a|s)$ trained with supervised learning, given optimal action labels a^* at each state $s \in \mathcal{D}$, the error bound is linear in the time horizon H for the online case and quadratic for the offline case. Intuitively, once the offline policy enters a states outside of its training distribution, it will continue to make mistakes and remain out-of-distribution for the rest of the rollout, accumulating $O(H)$ error. As there is a chance of entering an out-of-distribution state at each of the H time steps, the total error becomes $O(H^2)$. Offline RL algorithms can in principle mitigate this error by constraining the learned policy to the behavior policy induced by the dataset with a KL divergence, similar to the one discussed for policy gradient methods in Section 2.1.2. However, this approach may result in a significant decrease in overall performance, as the behavior policy and any policy that is similar to it, may not be as effective as the best policy that can be inferred from offline data.

Besides the state distribution shift at test time, during training a more subtle *action distribution shift* takes place. Concretely, the computation of the targets for the Bellman backups in Equation 2.12 depends on $a_{t+1} \sim \pi(a_{t+1}|s_{t+1})$. Without the possibility of online interaction, evaluating Q_π at a_{t+1} is challenging due to the fact that the Q-function regression

targets are dependent on the estimated Q value for actions that are not present in the training distribution used to learn the Q-function. Consequently, when $\pi(a|s)$ differs considerably from π_β, the target Q-values can be highly inaccurate. This issue is further compounded by the explicit maximization of the target values $\mathbb{E}_{a\sim\pi(a|s)}Q_\pi(s,a)$, leading the policy to take out-of-distribution actions, because the Q-function mistakenly assigns high values to them. In online reinforcement learning, the issue of over-optimistic Q-values is mitigated through the process of the policy taking actions it mistakenly believes to be good and receiving feedback from the environment to correct these errors. However, in offline reinforcement learning the policy lacks this mechanism for correction. Consequently, errors accumulate over the course of the training, resulting in sub-optimal performance.

There are several approaches to address these issues. In Chapter 8, we build upon Conservative Q-learning (CQL) [73]. The main idea of CQL is to ensure that the Q-values learned from offline data are always conservative, meaning that they are always an underestimation of the true Q-values. CQL achieves these conservative predictions by combining the Bellman objective $J(\mathcal{D},\phi)$ with a Q-value regularizer $C(\mathcal{D},\phi)$:

$$\hat{J}(\mathcal{D},\phi) = \alpha C(\mathcal{D},\phi) + J(\mathcal{D},\phi) \tag{2.17}$$

where $C(\mathcal{D},\phi)$ is the conservative penalty term. CQL implements this penalty with an adversarial action distribution $\mu(a|s)$ which samples actions that can produce high Q-values. The addition of this penalty term acts to constrain the Q-values of these state-action pairs, resulting in a lower bound for the estimated Q-function.

$$C_{CQL_0}(\mathcal{D},\phi) = \mathbb{E}_{s\sim\mathcal{D},a\sim\mu(a|s)}[Q(s,a)] \tag{2.18}$$

For actions that are within the training distribution, the standard Bellman objective term $J(\mathcal{D},\phi)$ still enforces that the Q-values adhere to the Bellman backup. By selecting an appropriate value for the penalty weight α, the conservative penalty should primarily decrease Q-values for actions that fall outside the distribution and have potentially incorrect high values. Kumar *et al.* [73] show that a Q-function trained with this conservative penalty will represent a lower bound of the true Q-function $Q(s,a)$ for the current policy. In general, CQL is a state-of-the-art offline RL approach that can be seamlessly integrated into existing Q-learning and actor-critic frameworks. Specifically, in the context of continuous control tasks it can be integrated on top of SAC, as demonstrated in Chapter 8.

2.2 Language Representation Models

NLP (Natural Language Processing) is an interdisciplinary field of study concerned with making sense of human language from a computational perspective. Making a computer "understand" natural language has a wide range of applications, such as text classification and generation, language translation or speech recognition. One of the challenges we address in this thesis is grounding natural language into a robot's sensorimotor experience. Thus, as a highly interdisciplinary problem, we draw inspiration from NLP advancements to model language in the methods described in Chapter 4, Chapter 5, Chapter 6 and Chapter 10.

The first step towards understanding language computationally is to translate words to a representation on the basis of numbers. The most simple and naive way to do this is building a vocabulary V of the words our program should be able to understand, and represent every

word with a one-hot encoding. For example, we might have the word "lemon" in position 128, "apple" in position 3000 and "robot" in position 130. This representation has two major issues. On the one hand, it is highly inefficient as only a single item from a large vector will have a value different to zero. On the other hand, it treats all words equally, as the distance between every pair is the same. However, intuitively the word "lemon" is more similar to the word "apple" than to "robot", as they are both fruits. Language models need to understand not only single words, but the meaning of sentences. This can be expressed as learning the joint probability of a sequence words. Concretely, the probability of the next word w_t given the sequence of previous words is

$$P(w_i^t) = \prod_{t=1}^{T} P(w_t \mid w_1^{t-1}), \tag{2.19}$$

where $w_i^j = (w_i, w_{i+1} \dots w_{j-1}, w_j)$. Thus, having seen in the training the sentence "The king ate an apple in the kitchen" might help generalize to the unseen sentence "The queen ate a pineapple in the dining room", because there are similar semantical and grammatical roles between the pairs ("queen", "king"), ("apple", "pineappple") and ("kitchen", "dining room"). The underlying linguistic assumption is coined the distributional hypothesis [74, 75], which postulates that words appearing in similar contexts are related to each other semantically.

Word embeddings solve these problems by representing each word in the vocabulary with a fixed size vector, which is learned during the training [76]. These dense vectors position words with semantically or contextually similar meanings in close proximity, as opposed to the sparse representation resulting from one-hot embeddings, which leads to a greater efficiency. Word embeddings can be generally divided into context-free and contextual word embeddings. Context-free representations such as Glove [77] and Word2Vec [78] are based only on the characters the word consists of, while contextual word embeddings take into account the neighboring words, *i.e.* the context.

2.2.1 Sequence-to-Sequence Models

In order to give robots instructions with language, the meaning of a sentence needs to be captured. Thus, in this thesis we leverage contextualized language representations. As sentences consists of a sequence of words, techniques that can effectively process sequential data are used in NLP. Imagine the task of interest is translating a sentence to another language. Vanilla neural networks are not well suited for such tasks, as they map fixed length inputs (*e.g.,* an image) to fixed length outputs (*e.g.,* probabilities over classes). However, in translation tasks it might not be possible to know a priory the length of a sentence in a different language. Thus, one of the most common architectures found to encode sentences are Recurrent Neural Networks (RNNs) [79].

In its most simple form, the RNN applies the following function to each of the sequence elements sequentially:

$$h_t = \tanh(W_{hi}x_t + W_{hh}h_{t-1}) \tag{2.20}$$

where h_t is the hidden state at time t, x_t is the input at time t, and $h_{(t-1)}$ is the hidden state of the previous layer at time $t-1$ or the initial hidden state at time 0. The hidden state encodes the context of the sentence.

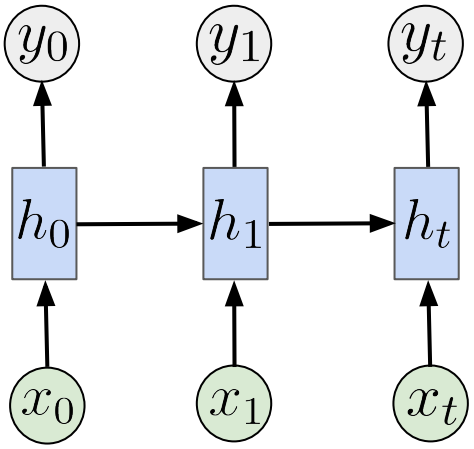


Figure 2.3: Visualization of a Recurrent Neural Network, also known as an Elman RNN [80].

Thus, RNNs can process sequences of arbitrary length and store a persistent representation of it in the form of the hidden state. An attractive property of RNNs is the potential for connecting prior information to current tasks. Imagine the task of predicting the last word in the sentence "I grew up in Germany... I speak fluent *German*" in a sentence completion task [81]. In order to predict correctly the language, the model needs the context of Germany, which is at the beginning of the sentence. However, basic RNNs struggle to capture such long horizon dependencies due to exploding and vanishing gradients [82]. This happens because during backpropagation, the calculation of (partial) derivatives/gradients in the weight update formula follows the chain rule, where gradients in earlier layers are the multiplication of gradients of later layers. Thus, in practice Long Short Term Memory networks (LSTMs) [83] are often used, as their formulation helps mitigate the aforementioned issues. The architecture is similar to basic RNNs, but the update rule for each element in the sequence is given by:

$$
\begin{aligned}
i_t &= \sigma(W_{ii}x_t + W_{hi}h_{t-1}) \\
f_t &= \sigma(W_{if}x_t + W_{hf}h_{t-1}) \\
g_t &= \tanh(W_{ig}x_t + +W_{hg}h_{t-1}) \\
o_t &= \sigma(W_{io}x_t + W_{ho}h_{t-1}) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot \tanh(c_t)
\end{aligned}
\tag{2.21}
$$

where σ represents the sigmoid activation function, $\odot$ the Hadamard product, and i_t, f_t, g_t, o_t are the input, forget, cell, and output gates, respectively. Intuitively, these gates control how much information gets added to the persistent cell state and when it should be updated. In order to make predictions, RNNs or LSTMs are combined in an encoder-decoder architecture, coined Seq2seq model [84], where the decoder uses the resulting context vector for downstream predictions. In Chapter 4, we use LSTM networks and train word embeddings end-to-end to i) encode sentences for referring to objects in the scene the robot should pick up, and ii) to describe image regions with natural language descriptions in order to interact with a user to resolve ambiguities.

2.2.2 Transformer Networks

One practical issue of training RNNs or LSTMs is that they are difficult to parallelize due to the sequential unrolling for each timestep. Besides, the fact that all the information is stored in a context vector can limit the ability to understand long dependencies and complex inputs. The introduction of self-attention and its integration into the transformer network architecture [85] has efficiently addressed both challenges, playing a pivotal role in fostering numerous ML innovations in the recent demi-decade.

An attention mechanism is a method to selectively focus on parts of the input when making a prediction. It helps weigh the importance of different input elements and can improve the performance of tasks such as machine translation and image captioning. In essence, the attention mechanism models correlations. For example, the occurrence of the word "eating" activates an expectation for a subsequent word related to food, as a result of semantic associations. Thus, in the sentence "He is eating a yellow banana", the words "eating" and "banana" have a high attention score, while the pair ("eating", "yellow") has a lower attention score.

Self-attention is an auto-regressive attention mechanism, *i.e.* it has the ability to examine prior inputs in the sequence during the encoding of the current input. In order to compute the attention scores, first each element of the input sequence is linearly projected to a query q_i, key k_i, and value v_i to form the respective matrices $Q \in \mathcal{R}^{d_k}$, $K \in \mathcal{R}^{d_k}$, $V \in \mathcal{R}^{d_v}$. Then, the elements are scored against each other to determine how much attention to place on other parts of the sequence. This is achieved via a scale dot-product attention by computing the dot

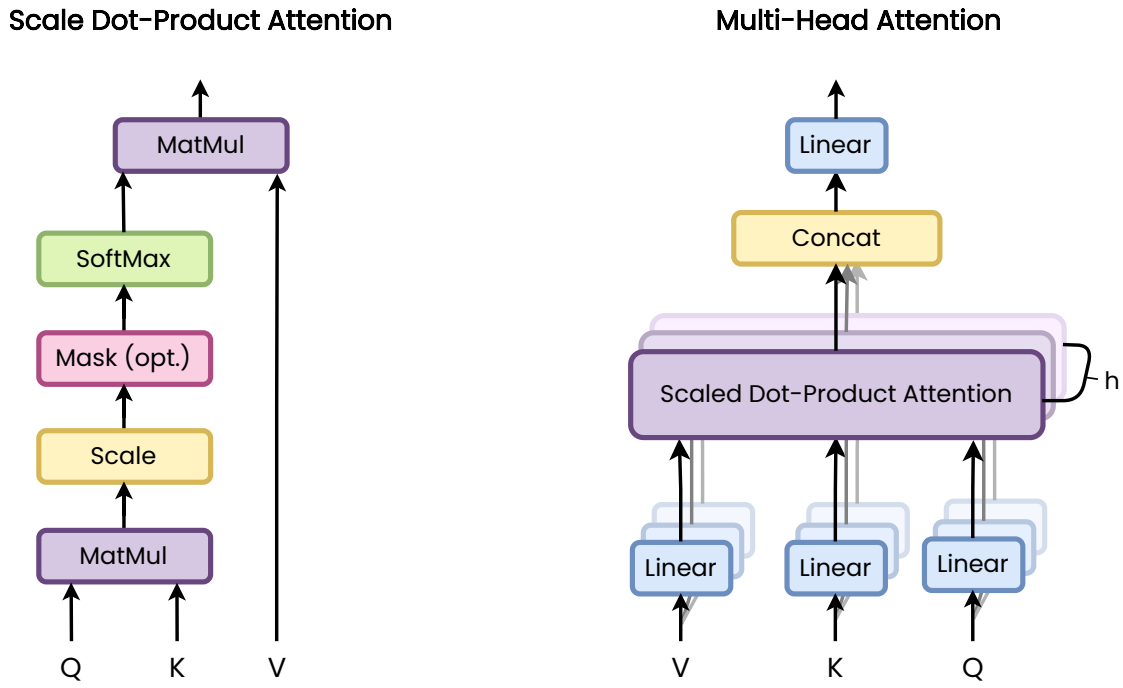


Figure 2.4: Overview of multi-head attention. Each of the h heads in multi-head attention (right) computes the scale dot-product attention (left), enabling the model to learn contextual representations from different representation subspaces at different positions. Source: Vaswani *et al.* [85].

product of the query vector with all the keys. To stabilize the gradients, an additional scaling factor $\frac{1}{\sqrt{d_k}}$ is used. Finally, a softmax function is applied to normalize the scores and obtain the weights of the values:

$$\textit{Attention}(Q, K, V) = \textit{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.22}$$

Instead of only computing the self-attention once, the multi-head mechanism runs the scaled dot-product attention multiple times in parallel, which allows the model to learn contextual representations from multiple subspaces as visualized in Figure 2.4. The full transformer architecture, visualized in Figure 2.5, stacks several multi-head attention layers followed by fully connected networks in the encoder to learn to focus on important parts of the input sequence. The decoder also stacks several identical layers, but additionally computes attention with respect to the encoder to learn the relationship between the input and the target. One of the main advantages of this architecture is that there is no recurrence, as it consists only of standard feed-forward networks. However, the network needs to incorporate knowledge about the relative or absolute position of the elements in the sequence. The

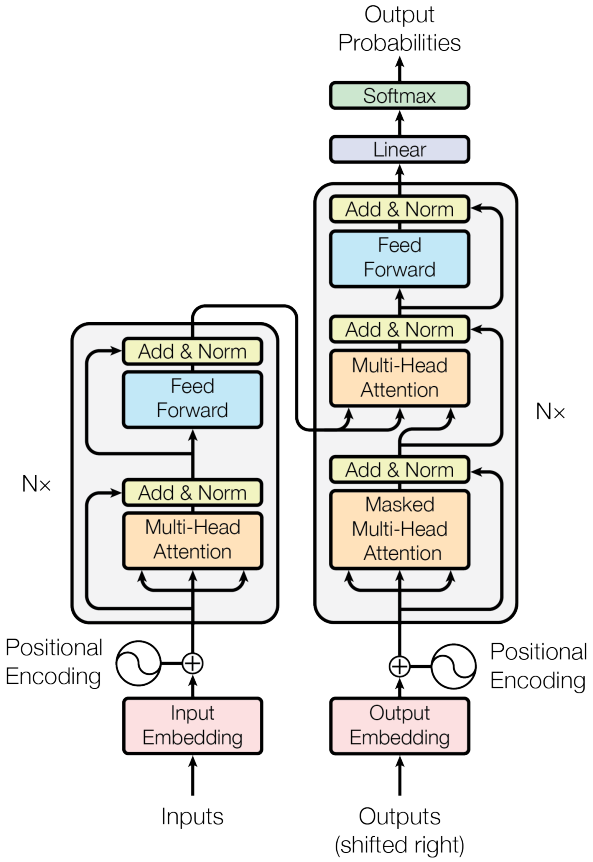


Figure 2.5: Overview of a transformer network. The encoder (left) makes use of self-attention to learn the relationships within the input sequence. The decoder (right), additionally computes attention with respect to the encoder to learn the relationship between the input and the target. Source: Vaswani *et al.* [85].

transformer architecture solves this by injecting positional embeddings in the form of a sinusoid-wave for example.

We use a transformer encoder architecture in Chapter 6 and Chapter 10 to learn temporally contextualized global video representations that are used to recognize and organize high-level behaviors. It is also worth mentioning that the transformer encoder is significantly more efficient both memory and model size wise in comparison to RNNs, *e.g.,* 5.9 M vs 106 M parameters for the video encoder network in Chapter 6.

2.2.3 Pre-trained Language Models

The advent of the transformer network was followed by a proliferation of language models trained on large corpora of text. Among the pioneers was "Bidirectional Encoder Representations from Transformers" (BERT), a natural language processing model proposed by Devlin *et al.* [86]. We will briefly describe it, as most of the modern language models are similar in spirit.

BERT's architecture is a multi-layer bidirectional transformer encoder. Bidirectionality means that it learns to predict both context on the left and right, *i.e.* both the next and previous words determine the representation of a word. For example, the sentences "I very much like rock bands from the 80s" and "I very much like rock formations in the desert" can only be understood by allowing the succeeding words after the word "rock" to influence the representation. Unidirectional models only consider the left context previous to the word, in both cases "I very much like", making it difficult to infer the meaning of the word rock. By considering both directions, a better representation of the word "rock" is obtained. Bidirectionality was found to improve the quality of the language embedding representations.

In order to train a deep bidirectional representation, BERT masks 15% of tokens in each sequence at random, and then predicts those masked tokens. Besides, in order to capture longer relationships at the sentence level, BERT incorporates a next sentence prediction loss (NSP) by forming tuples of sentences (A, B) and predicting if B follows A, which is true for 50% of the tuples formed. These simple auxiliary losses, which can be trivially generated from any monolingual corpus, effectively allows to scale the training to Internet-scale *diverse* datasets. The contextual representations of words in a sentence learned by BERT were shown to be effective for a broad range of NLP tasks, such as question answering or sentiment analysis.

Although BERT demonstrated strong performance in various NLP tasks, its effectiveness is limited in some situations. For instance, when trying to predict the two most similar questions asked in a forum, BERT could form a pairwise comparison. However, the $n \cdot (n-1)/2$ forward passes needed for all possible sentence combinations makes this computationally infeasible. Similarly, when directing a robot with language instructions, a user might use very similar sentences that have different meanings, such as "grasp the pink block and turn it left" and "grasp the pink block and turn it right". Embeddings from BERT for these sentences will be very close in latent space, therefore confusing the robot on the desired behavior. SBERT (Sentence-BERT) [87] alleviates these issues by reasoning over the similarity of sentence-level representations. SBERT is a modification of the BERT architecture using siamese networks in order to derive semantically meaningful sentence embeddings.

SBERT adds a pooling layer on top of BERT to extract a fixed size representation for a sentence. After passing two sentence through a siamese network, a cosine similarity

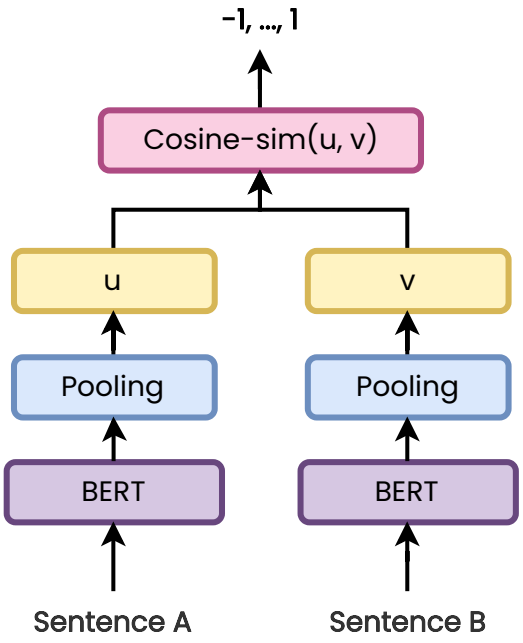


Figure 2.6: SBERT computes the cosine similarity between the sentence embeddings with siamese networks. Source: Reimers *et al.* [87].

is computed to reason over the sentence-level similarity, which is particularly useful for tasks that involve semantic comprehension like paraphrasing or clustering. We visualize the architecture of SBERT in Figure 2.6.

In this thesis, we found that language models finetuned on sentence similarity to work best to understand free-form language instructions for directing robots. We perform a study over the choice of architectures and losses to encode sentence semantics in Chapter 6. Throughout this thesis, we have leveraged pre-trained language models to encode raw text into a semantic pre-trained vector space in Chapter 5, Chapter 6 and Chapter 10.

Chapter 3

Learning Object Placements For Relational Instructions by Hallucinating Scene Representations

The content of this chapter has been published in [88]:

> O. Mees, A. Emek, J. Vertens and W. Burgard
> **Learning Object Placements For Relational Instructions by Hallucinating Scene Representations**
> *IEEE International Conference on Robotics and Automation (ICRA)*, 2020

Me and Alp Emek share the main authorship. The initial work on the Spatial-RelNet architecture for estimating pixelwise object placement heatmaps was formulated in collaboration with Johan Vertens for Alp Emeks Master's thesis, which the author of this thesis supervised together with Johan. Alp helped with collecting and labeling the dataset and implemented the initial experimental framework. All the results for Spatial-RelNet reported in this thesis were entirely carried out by the author of this thesis. The paper was mostly written by the author of this thesis.

Abstract

Robots coexisting with humans in their environment and performing services for them need the ability to interact with them. One particular requirement for such robots is that they are able to understand spatial relations and can place objects in accordance with the spatial relations expressed by their user. In this work, we present a convolutional neural network for estimating pixel-wise object placement probabilities for a set of spatial relations from a single input image. During training, our network receives the learning signal by classifying hallucinated high-level scene representations as an auxiliary task. Unlike previous approaches, our method does not require ground truth data for the pixelwise relational probabilities or 3D models of the objects, which significantly expands the applicability in practical applications. Our results obtained using real-world data and human-robot experiments demonstrate the effectiveness of our method in reasoning about the best way to place objects to reproduce a spatial relation. Videos of our experiments can be found at `https://youtu.be/zaZkHTWFMKM`

3.1 Introduction

Understanding and leveraging spatial relations is a key capability of autonomous service robots operating in human-centered environments. In this work, we aim to develop an approach that enables a robot to understand spatial relations in natural language instructions to place arbitrary objects. The spatial relations include common ones such as “left”, “right” or “inside”. In Figure 3.1, the robot is asked to “place the mug on the right of the box”. To do so, the robot needs to reason about where to place the mug relative to the box in order to reproduce said spatial relation. Moreover, as natural language placement instructions do not uniquely identify a location in a scene, it is desirable to model this using distributions to capture the inherent ambiguity.

Object-object spatial relations can be learned in a fully-supervised manner [89, 90, 91, 92, 93, 94, 95] from 3D vision. The main limiting factor for exploiting this setup in practical robotics applications is the need for collections of corresponding 3D object shapes and relational data, which are difficult to obtain and require additional instrumentation for object tracking. This limits prior methods to training on synthetic datasets or simulators, leading to difficulties in their application to real-world scenarios. A possible solution to this problem is to model relations directly from RGB images [96], which allows direct training on real image data without the need of modeling the scene in 3D. Reasoning about object placements for relational instructions in this context requires estimating pixelwise spatial distributions of placement locations, as shown in Figure 3.1. One of the key challenges to estimate such pixelwise spatial distributions is the lack of ground-truth data. This originates from the inherent ambiguity on modeling such ground-truth distributions without using heuristics. If one wants to model the relation “left”, how far left of the reference object would form a valid relation? Should the distribution have a single or multiple modes? And where is the boundary between “left” and “in front of” for instance?

In this paper, we push the limits of relational learning further and present a method which leverages a weaker form of supervision to model object placement locations conditioned on a

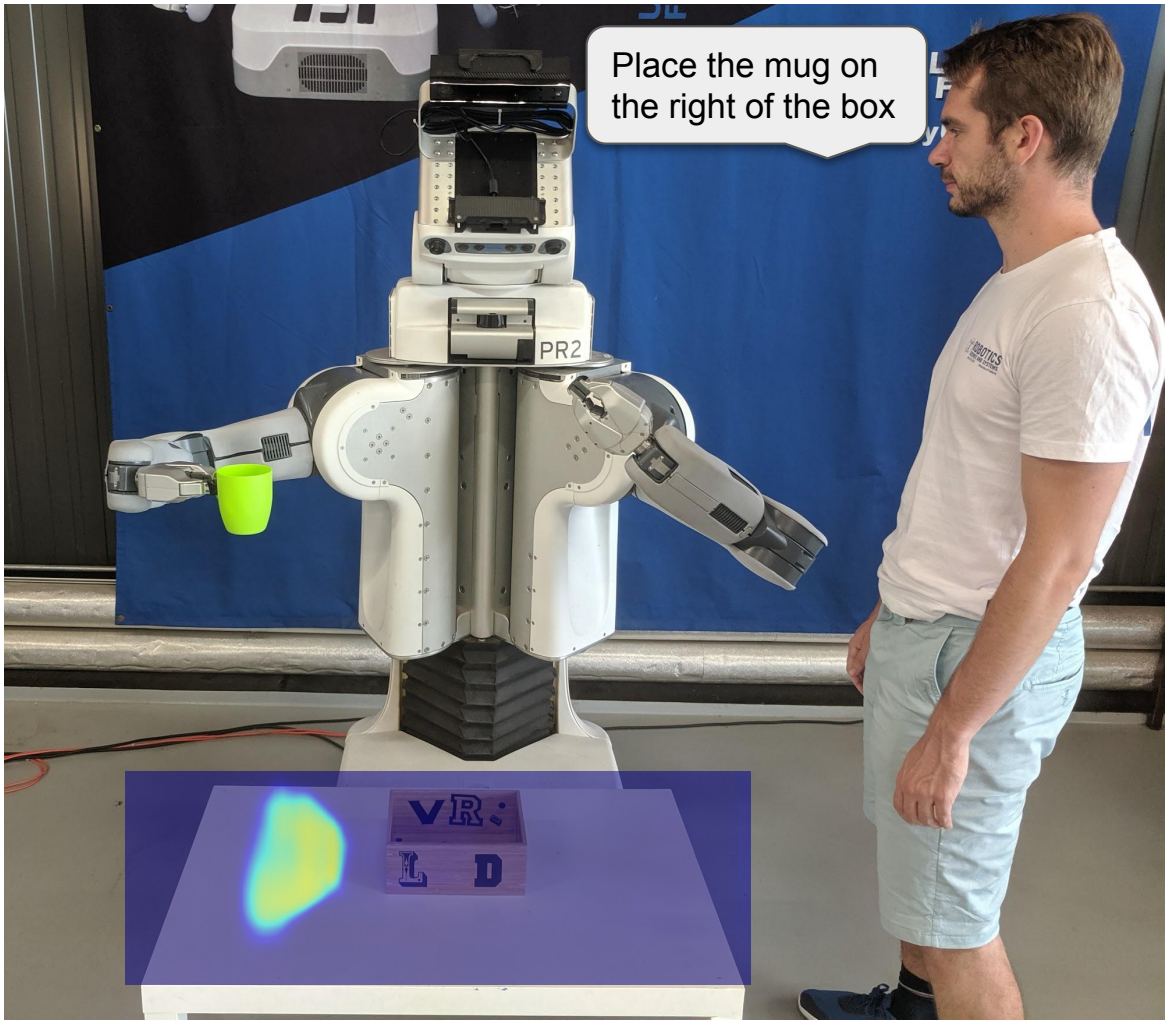


Figure 3.1: The goal of our work is to follow natural language instructions based on spatial relations to place arbitrary objects. Our network learns to predict pixelwise placing probability distributions (heatmap on the table) solely from classifying hallucinated high-level scene representations into a set of spatial relations.

set of spatial relations. We address the problem of the unavailability of ground-truth pixelwise annotations of spatial relations from the perspective of auxiliary learning. Our approach relies solely on relational bounding box annotations and the image context to learn pixelwise distributions of object placement locations over spatial relations, without any additional form of supervision or instrumentation. Though classifying two objects into a spatial relation does not carry any information on the best placement location to reproduce a relation, inserting objects at different locations in the image would allow to infer a distribution over relations. Most commonly, "pasting" objects realistically in an image requires either access to 3D models and silhouettes [97, 98, 99, 100] or carefully designing the optimization procedure of generative adversarial networks [101, 102]. Moreover, naively "pasting" object masks in images creates subtle pixel artifacts that lead to noticeably different features and to the training erroneously focusing on these discrepancies. Our results show that such models lead to reduced performance. Instead, we take a different approach and implant high-level features of objects into feature maps of the scene generated by a network to hallucinate

scene representations, which are then classified as an auxiliary task to get the learning signal. Training a network in this setup solely requires being able to classify relations between pairs of objects from an image.

We demonstrate both qualitatively and quantitatively that our network trained on real-world images successfully predicts pixelwise placement probability distributions for each spatial relation. Our approach can be trained on images with relational bounding box annotations and does not require 3D information or any additional instrumentation to predict the spatial distribution of arbitrary objects, thus making it readily applicable in a variety of practical robotics scenarios. We exemplify this by using the probability distributions produced by our method in a robot experiment to place objects on a tabletop scene by following natural language instructions from humans.

3.2 Related work

Learning spatial relations by relying on the geometries of objects provides a robot with the necessary capability to carry out tasks that require understanding object interactions, such as object placing [89, 90], human robot interaction [32, 103, 104, 105], object manipulation [93] or generalizing spatial relations to new objects [91, 92, 106]. Commonly, spatial relations are modeled based on the geometries of objects given their point cloud models [91, 92, 93]. However, learning object relations from 3D data [91, 92, 93, 94, 95] typically requires additional instrumentation to track objects, with the consequent difficulties due to occlusions for example. One way to overcome this limitation could be learning to predict 3D shapes from single images in a self-supervised manner [107]. In contrast to these works, we learn spatial distributions directly from real-world images.

Spatial relations also play a crucial role in understanding natural language instructions [108, 109, 110], as objects are often described in relation to others. Several studies on human–robot interactions have been conducted, mainly for picking objects. These works focus on analyzing the expressive space of abstract spatial concepts as well as notions of ordinality and cardinality [104, 108]. Complementary to these works, we propose to learn distributions of object relations, for commonly used prepositions in natural language, to enable a service robot to place arbitrary objects given natural language instructions.

There has been a large body of research targeting relations in the vision community. Multiple works attempt to ground object relationships from images for classification [96], referring expression comprehension [111, 112, 113], human-object interactions [114] or relational learning in visual question answering [115, 116]. While these approaches reason about an existing scene, our method learns which future state might follow best a spatial relation grounded in natural language instructions. Generating a future state in a object placement scenario would mean to insert the object to be placed into different locations in the robot's view image. There exists a plethora of work for learning how to synthesize objects realistically into images [97, 98, 99, 100]. Most commonly, such methods requires either access to 3D models and silhouettes or carefully designing the optimization procedure of generative adversarial networks [101, 102]. Instead, our approach implants high-level features of objects to hallucinate scene representations, which are then classified by an auxiliary network to learn spatial distributions.

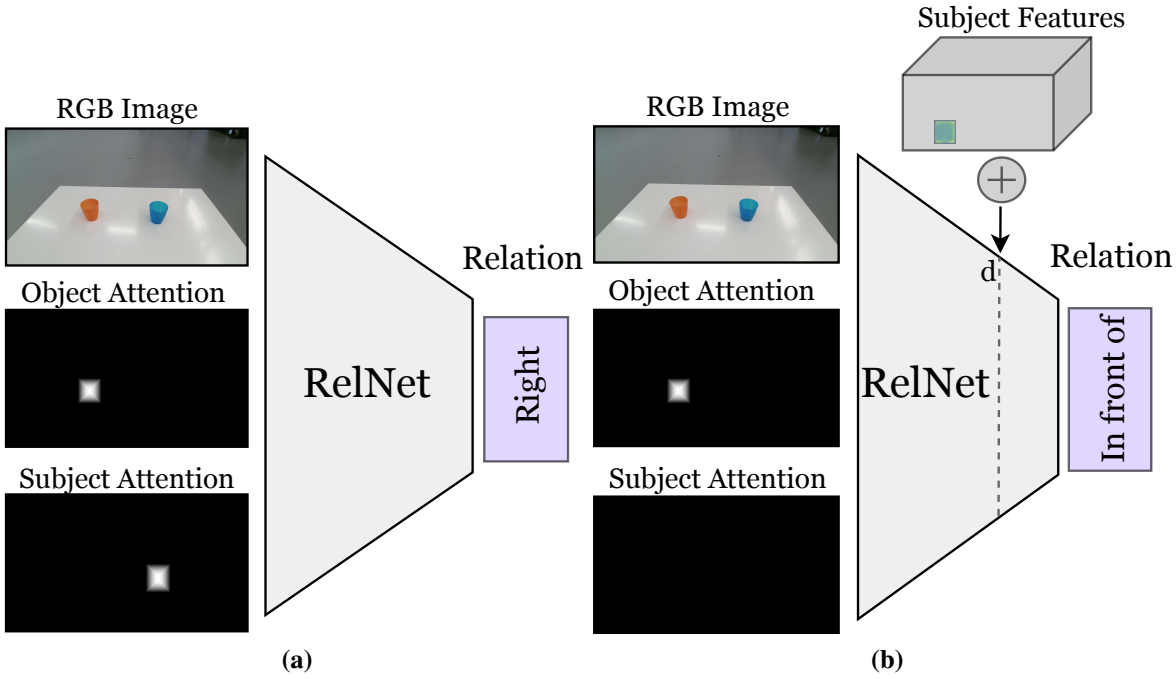


Figure 3.2: In the first stage of our approach, we train an auxiliary convolutional neural network, called RelNet, to predict spatial relations given the input image and the two attention masks referring to the two objects forming a relation (a). After training, we can "trick" the network to classify hallucinated scenes by implanting high-level features of items at different spatial locations (b).

3.3 Method description

In this section we describe the technical details of our method for estimating pixelwise object placement probabilities for a set of spatial relations from a single input image. We consider pairwise relations and express the subject item as being *in relation to* the reference item. We extract subject, object and relation from natural language instructions.

3.3.1 Auxiliary Network

In the first stage of our approach, we encode the input RGB image together with an object and a subject attention mask to classify them into a set of spatial relations with an auxiliary convolutional neural network (CNN). We denote the RGB image, the object and subject attention masks as $\mathbf{x}^i, \mathbf{a}_o^i, \mathbf{a}_s^i$ respectively and $\mathbf{y}^i$ corresponds to the relation label in one-hot encoding – i.e., $\mathbf{y}^i \in \{0,1\}^{|C|}$ is a vector of dimensionality C (the number of relations). We model relations for a set of commonly used natural language spatial prepositions $C = \{\texttt{inside}, \texttt{left}, \texttt{right}, \texttt{in front}, \texttt{behind}, \texttt{on top}\}$. Let $\mathcal{D} = \{(\mathbf{x}^1, \mathbf{a}_o^1, \mathbf{a}_s^1, \mathbf{y}^1), \ldots, (\mathbf{x}^N, \mathbf{a}_o^N, \mathbf{a}_s^N, \mathbf{y}^N)\}$ be the labeled data available for training our auxiliary classification network, which we name RelNet, see Figure 3.2a. Let θ_{RelNet} be the parameters of the network. We denote the mapping of RelNet as $f(\mathbf{x}^i, \mathbf{a}_o^i, \mathbf{a}_s^i; \theta_{RelNet}) \in \mathbb{R}^{|C|}$. The attention masks are calculated as a Gaussian distance transform $a(u,v) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}((1-d_{uv})/\sigma)^2}$ with d_{uv} being the distance transform between (u,v) and the bounding box center, based on the L2 norm and with $\sigma = 2$.

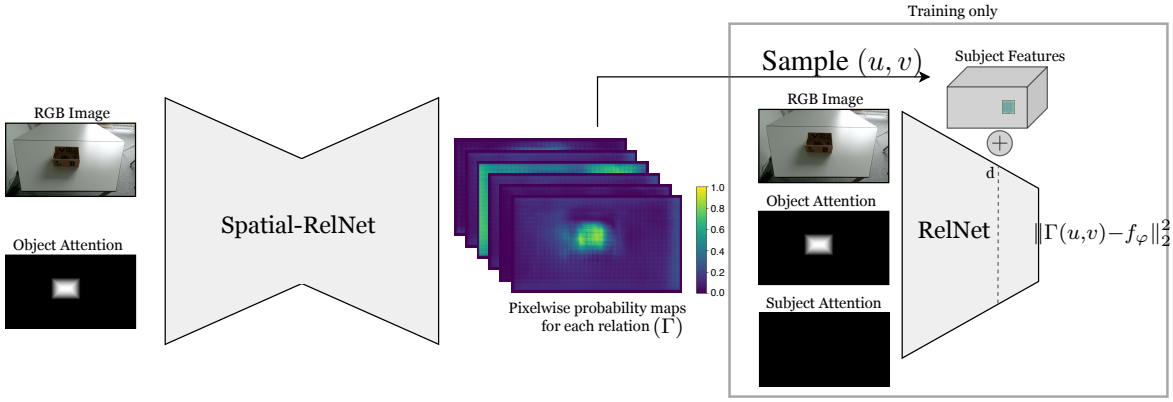


Figure 3.3: Our encoding-decoding Spatial-RelNet network processes the input RGB image and the object attention mask to produce pixelwise probability maps Γ over a set of spatial relations. During training, we sample locations (u, v) according to Γ, implant inside the auxiliary network RelNet at the sampled locations high level features of objects and classify the hallucinated scene representation to get a learning signal for Spatial-RelNet. At test time the auxiliary network is not used.

The goal of RelNet is to learn classifying scenes of pairwise object relations by minimizing the cross-entropy ($softmax$) loss. The $softmax$ function converts a score z_c for class C into a posterior class probability that can be computed as $\mathcal{L}(z_c) = \exp(z_c)/\sum_{j=1}^{|C|} \exp(z_j)$. Using stochastic gradient descent (SGD) we then optimize:

$$\theta^*_{RelNet} \in \arg\min_{\theta_{RelNet}} \sum_{i=1}^{N} \mathcal{L}(f(\mathbf{x}^i, \mathbf{a}^i_o, \mathbf{a}^i_s; \theta_{RelNet}), \mathbf{y}^i). \tag{3.1}$$

RelNet is only utilized during training time and discarded at inference time.

3.3.2 Hallucinating Scene Representations

Clearly, classifying the spatial relation formed by two items is not suitable to identify the best placing location to reproduce a relation. However, inserting objects at different locations in the image would allow to infer a distribution over relations. As mentioned before, "pasting" objects realistically in an image requires commonly either access to 3D models and silhouettes [97, 98, 99, 100] or carefully designing the optimization procedure of generative adversarial networks [101, 102]. To tackle this challenge, we take a different approach and implant high-level features of objects into a high-level feature representation of RelNet to hallucinate scene representations, which are then classified to get the learning signal, as shown in Figure 3.2b. Given an input image $\mathbf{x}^i$ of size $W \times H$, we use the RelNet network on the image to obtain a spatial feature map M_o of size $W_f \times H_f \times N_f$ (width, height, number of filters) at depth d. Given an input image, we extract a slice of the feature map $s \in \mathcal{R}^{W_s \times H_s \times N_f}$ corresponding to a bounding box containing an item in the image. Thus, hallucinating a scene representation requires no more than making a forward pass with RelNet and implanting the high level features of a subject object s into the feature map M_o at a sampled location (u, v) by summation and continuing the forward pass with the modified feature map. We define the implanting operation as:

$$\varphi(M_o, s, u, v) = M_o + M_s(u, v), \tag{3.2}$$

where

$$(M_s(u,v))_{jk} = \begin{cases} s(j-u, k-v), & \text{if } u \leq j \leq u+W_s \text{ and } v \leq k \leq v+H_s \\ 0, & \text{otherwise.} \end{cases} \tag{3.3}$$

This way we can reason over what pairwise spatial relations are most likely to be formed given an existing item in the image and a subject item which can be hallucinated at different locations. In other words, what relation would the two items form, if the subject item was placed at the specified location. Formally, we define the mapping of a RelNet with implanted features s at location u, v as $f_\varphi(\mathbf{x}^i, \mathbf{a}_o^i, \mathbf{a}_s^i, s, u, v) \in \mathbb{R}^{|C|}$.

3.3.3 Learning pixelwise item placement distributions

In the final stage of our approach, we model the primary task of inferring pixelwise spatial distributions to find the best placing locations by following a natural language instruction containing a spatial relation. We define a second network, named Spatial-RelNet, with an encoding-decoding architecture. Given an image $\mathbf{x}^i$ of size $W \times H$ and the object attention mask $\mathbf{a}_o^i$, the network predicts for each pixel in the input image the probabilities of belonging to one of the C classes with respect to the reference object attention, see Figure 3.3. Thus, we denote the mapping of Spatial-RelNet as $g(\mathbf{x}^i, \mathbf{a}_o^i) = \Gamma \in \mathbb{R}^{W \times H \times |C|}$ and $\sum_{j=1}^{|C|} \Gamma_j(u,v) = 1$ for all $u = 1...W, v = 1...H$. Due to the unavailability of ground-truth pixelwise annotation of spatial relations we propose a novel formulation which leverages auxiliary learning. During training, we sample pixel locations $(u,v) \in \xi \subseteq \{0,...,W\} \times \{0,...,H\}$ according to the probability maps Γ produced by Spatial-RelNet and then implant at the specified locations the subject object features to compute with RelNet a posterior class probability over relations. This way, we can reason over what relation would most likely be formed if we placed an object at the given location. Our formulation allows predicting non-parametric probability distributions. Thus, by sampling multiple locations in the scene from Γ and leveraging the auxiliary task of classifying the spatial relation formed by two objects in an image we can train our primary network Spatial-RelNet with the following mean squared error loss:

$$\sum_{u,v \in \xi} ||g(\mathbf{x}^i, \mathbf{a}_o^i)_{uv} - f_\varphi(\mathbf{x}^i, \mathbf{a}_o^i, \mathbf{a}_s^i, s, u, v)||_2^2. \tag{3.4}$$

We note that at inference time the auxiliary RelNet network is discarded.

3.3.4 Implementation Details

In the first stage of our approach, we train the auxiliary RelNet network on the task of classifying spatial relations between two objects in images. The auxiliary RelNet network is based on a ResNet-18 [117] architecture and is initialized with ImageNet pre-trained weights. We use the SGD optimizer with a learning rate of 10^{-3}.

In the final stage of our approach, we train Spatial-RelNet to predict pixelwise spatial distributions by using the auxiliary RelNet network for supervision. The Spatial-RelNet is inspired by the FastSCN [118] semantic segmentation architecture and initialized randomly. We apply a per-pixel sigmoid activation function for the last layer instead of $softmax$. We use the ADAM optimizer with a learning rate of 10^{-3}.We sample 20 locations per distributions

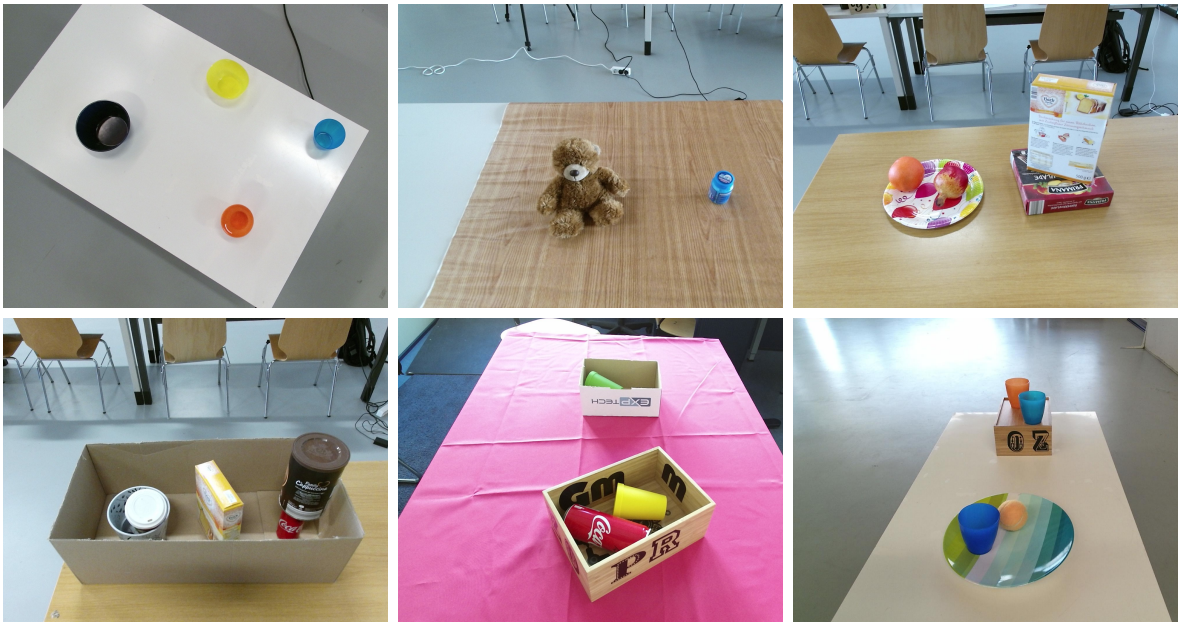


Figure 3.4: Examples of the scenes recorded for training the auxiliary RelNet classifier. We recorded a total of 1237 images of 165 tabletop scenes and manually annotated the bounding boxes of the objects and their spatial relations.

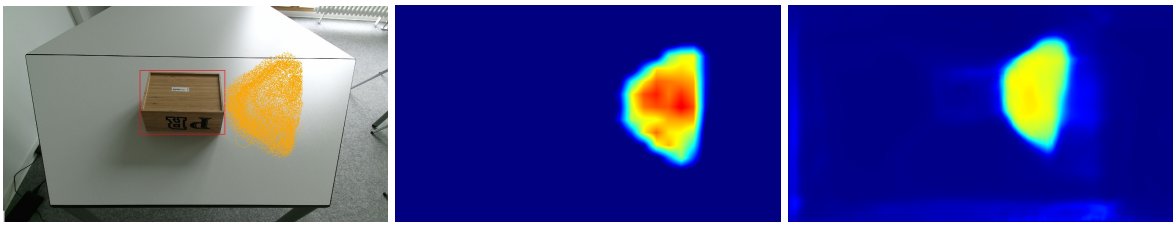
and use a feature map of the size $128, 10, 10$ pertaining to an object to hallucinate the scene representations. For all experiments, we implant the subject features after the third convolutional block "conv3_x" ($d = 3$) of the ResNet-18 architecture that characterizes RelNet. In order to speed up the training we apply a Sobel filter on the output probability maps $\mathbf{H}$ to propagate the gradient to local neighborhoods. We define the Sobel kernels as $\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$ for the x direction and $\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$ for the y direction.

3.4 Experiments

In this section we showcase our approach both qualitatively and quantitatively, and demonstrate its applicability in a human-robot experiment, where participants ask a PR2 robot to place objects with natural language instructions based on spatial relations.

3.4.1 Dataset

We record and annotate a total of 1237 images of 165 tabletop scenes, see Figure 3.4. The images depict tabletop scenes from three different viewpoints (object-centric to top-down) containing spatial relations formed by using combinations of 40 different household objects. Learning from multiple camera viewpoints helps the approach become less sensitive to viewpoint changes and generalize better. We annotate 5304 pairwise bounding boxes with the commonly used natural language spatial prepositions $C = \{\texttt{inside}, \texttt{left}, \texttt{right}, \texttt{in front}, \texttt{behind}, \texttt{on top}\}$. For all recorded scenes we use different tablecloths and tables in different rooms to avoid overfitting. To evaluate the pixelwise probability distributions predicted by Spatial-RelNet, we record 105 scenes containing unseen objects and tables. Due to



(a) (b) (c)

Figure 3.5: Example user annotation of ground-truth points for the relation "right" with a "spray" paint tool (a). We convolve the user annotated points to generate a dense distribution (b). The shown network output distribution and the ground-truth distribution have a $\mathrm{IoU}_{0.5}$ of 0.39 (c).

the inherent ambiguity of defining pixelwise spatial distributions, we ask 3 participants to annotate them. To do so, the user use a "spray" tool to draw points in which placing an item would reproduce a given spatial relation, as shown in Figure 3.5a. The points are then convolved with a fixed kernel to generate a dense pixelwise ground-truth distribution, as seen in Figure 3.5b.

3.4.2 Evaluation protocol

To compare the pixelwise distributions predicted by our method with the ground-truth pixelwise annotations provided by the participants, we report several metrics. Inspired by metrics from object detection and segmentation, we threshold the distributions at different ranges and compute their mean intersection over union (IoU). Additionally, we are interested in comparing the modes between the distributions. First, we compute the maximum mode from each distribution and report the euclidean pixel distance between them (Mode). As the distance between the modes does not model the tails of the distributions, we also calculate the distance between the centroid pixels of the predicted and ground-truth distributions (Centroid). Due to the non-parametric nature of the distributions, we perform a Kruskal-Wallis (KW) test by analyzing if 100 points sampled from the ground-truth distribution and another 100 points sampled from the predicted distribution originated from the same distribution, with a significance of $p < 0.05$. Finally, we measure the similarity of the probability distributions with the Kullback–Leibler (KL) and Jensen–Shannon (JS) divergences.

3.4.3 Quantitative Results

First, we analyze the performance of the auxiliary RelNet network to model spatial relations, as we rely on it to get the learning signal for Spatial-RelNet. We evaluate the performance of RelNet on a test split containing 975 pairwise relations and report an average accuracy of 97% over all relations, as shown in Table 3.1. We compare its performance against a model that was trained only on binary masks of the objects to analyze the importance of using the image context to model the relations. This model achieves an accuracy of 84.4% and we find that the image context is specially important to disambiguate the relations `on top` and `inside`. We also train an intermediate model, which takes as input the image and binary object masks to model the relations, and achieves an accuracy of 94.3%. Our final model

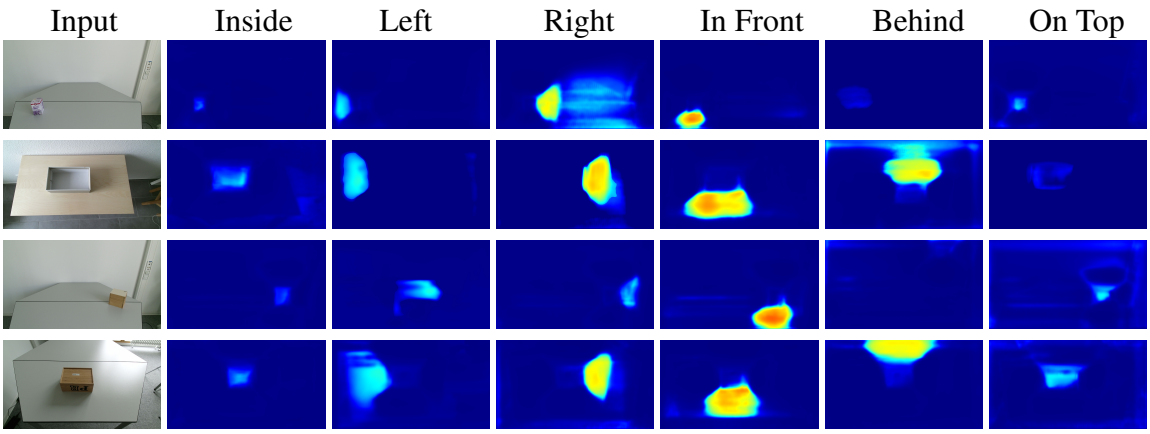


Figure 3.6: Qualitative results for predicting pixelwise distributions for every spatial relation. Placing an object at a location sampled from these distributions maximizes the probability of reproducing the selected spatial relation. Our network produces meaningful distributions, despite relying solely on an auxiliary task of classifying hallucinated high-level scene representations into a set of spatial relations for supervision.

shows the best performance by incorporating the use of the Gaussian distance transforms for the attention masks.

Model	Mean	Inside	Left	Right	In Front	Behind	On Top
Masks only	84.4	60.8	99.3	93.2	99.3	98.1	56.6
Image + Masks	94.3	81.3	99.3	100	98.7	97.5	88.5
Full model	97	93.1	98.7	100	100	98.7	91.5

Table 3.1: Quantitative comparison of RelNet with its variants. Adding the image context helps disambiguating the relations `on top` and `inside`.

Next, we quantitatively evaluate the capability of a baseline model in which we naively "paste" objects masks in the RGB images to predict pixelwise distributions for spatial relations. We report mean Ious of 0.44, 0.4, 0.3 for the thresholds of 0.25, 0.5 and 0.75 respectively. This shows that the artifacts created by "pasting" object masks in RGB images lead to noticeably different features and to the training erroneously focusing on these discrepancies. In comparison, our Spatial-RelNet achieves mean IoUs of 0.63, 0.6 and 0.44, as shown in Table 3.2, by classifying hallucinated scene representations, alleviating this problem. Moreover, for Spatial-RelNet the mean distance between the modes of the distributions lies at 67.2 pixels, corresponding approximately to 5.74cm. As this metric depends on the image resolution and the distance of the camera to the objects, for each image in the test set, we sample uniformly 100 pixels and compute their average distance to the mode of the ground-truth distribution. Thus, the mean distance between a random pixel and the ground-truth mode is 504.5 pixels. To model the tails of the distributions, we also calculate the distance between the centroid pixels of the predicted and ground-truth distributions and we report a mean distance of 113.5 pixels, which corresponds approximately to 9.88 cm. We measure the similarity of the distributions with the Kullback–Leibler and Jensen–Shannon divergences, were we report mean values of 3.78 and 0.46 respectively. Finally, we found in 0.55% of the cases the samples drawn from the predicted and ground-truth distribution to

Metric	Mean	Inside	Left	Right	In Front	Behind	On Top
$\mathrm{IoU}_{0.25}$	0.63	0.66	0.69	0.65	0.64	0.51	0.65
$\mathrm{IoU}_{0.5}$	0.6	0.62	0.6	0.62	0.57	0.57	0.62
$\mathrm{IoU}_{0.75}$	0.44	0.47	0.41	0.46	0.39	0.48	0.5
Mode	67.2	43.5	71	59.2	86.8	115.3	90.5
Centroid	113.5	116.7	241.1	85.1	70.9	51.4	163.4
KL	3.78	5.35	4.47	3.5	1.62	3.9	5.7
JS	0.46	0.54	0.48	0.45	0.34	0.5	0.57
KW	0.55	0.63	0.5	0.51	0.51	0.43	0.73

Table 3.2: Quantitative comparison of the predicted pixelwise distributions with ground-truth annotations for a range of metrics. Our methods yields good results though relying on a weaker form of supervision.

originate from the same distribution according to the Kruskal-Wallis test, with a significance of $p < 0.05$.

We show qualitative results in Figure 3.6. In this challenging setting, the network learns to produce meaningful distributions, from which one can sample object placement locations to reproduce a spatial relation.

3.4.4 Human-Robot Object Placement Experiment

We also evaluate the performance of our approach in a realistic human-robot collaboration context. We exemplify the ability of our approach to reason about the best way to place objects by asking a group of participants to provide relational natural language instructions to a PR2 service robot in a tabletop scene.

Procedure

Our study involved 11 participants recruited from a university community. Each participant was asked to give 20 natural language instructions to the PR2 robot, which were parsed with an Amazon Echo Dot device. For each placement trial, the participants were asked to choose a reference item from a range of 30 household objects and to place it on the table at a random location. Next, the participants were instructed to choose a different item to put on the gripper of the robot. Afterwards, the participants were asked to provide a natural language instruction that contained one of the six spatial relations. We relied on keyword spotting to select the corresponding predicted distribution. The instruction was repeated in case of failures synthesizing the voice input. After sampling a (u, v) location from the predicted distribution, we used the robot's Asus Xtion RGB-D camera to localize the pixel coordinate in 3D space. Our system then planned a top-down grasp pose to the calculated 3D point. The reachability of the proposed plan was checked using MoveIt! [119]. The end-effector was moved above the desired location and then the gripper was opened to complete the placement. After each trial the participants rated the placement on a 10-point Likert scale as well as a binary success rate. The Likert scale helps us rate ambiguous placements such as a top left or diagonal placements for a instruction containing the relation "left".

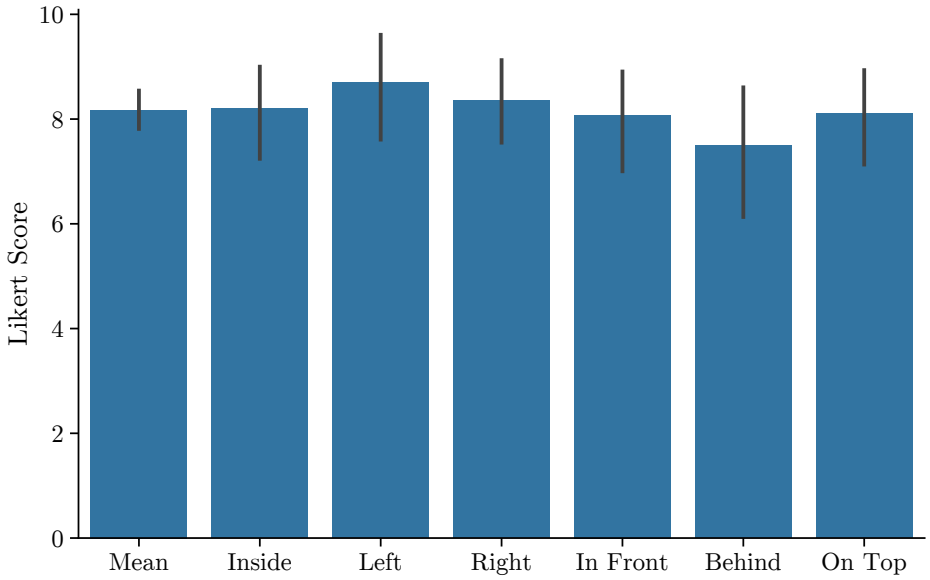


Figure 3.7: Performance of a PR2 robot following natural language instructions of 11 participants for object placement. Error bars indicate 95% confidence intervals.

Results

Figure 3.7 shows the performance of our approach on a PR2 robot for a total of 220 natural language instructions of 11 participants. We report a mean rating of 8.1 over all relations and trials. We observe a lower score for "behind" as many points sampled were outside the reach of the robot's gripper and therefore no valid motion plan was found. We observe a similar behaviour for the success rates reported on Table 3.3. Many participants chose reference items with a small area to place the object for the relations "on top" and "inside", requiring a precision placement. For such challenging scenarios, we observed some failure cases when the sampled location lied at the border of the reference item or the depth information from the RGB-D camera contained noise. Overall, our results demonstrate the ability of our approach to effectively learn object placements for relational instructions.

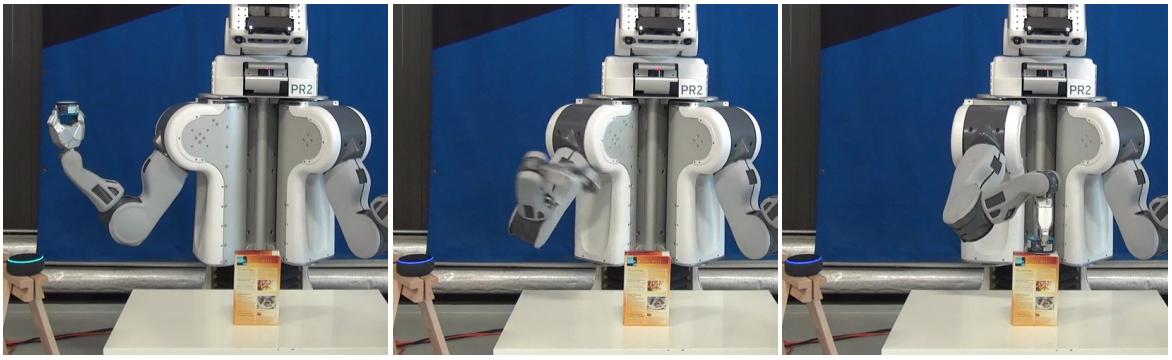


Figure 3.8: Example object placing execution with the PR2 robot for the natural language instruction "place the can on top of the box", which is synthesized with the Amazon Echo Dot.

Metric	Mean	Inside	Left	Right	In Front	Behind	On Top
Success Rate	0.84	0.84	0.87	0.95	0.84	0.80	0.79

Table 3.3: Performance of our approach on a real robot platform following natural language instructions of 11 participants to place objects in a tabletop scenario.

3.5 Conclusion

In this paper, we presented a novel approach to the problem of learning learning object placements for relational instructions from a single image. We exemplified how the distributions produced by our method enables a real-world robot to place objects by following relational natural language instructions. Our method is based on leveraging three key ideas: *i*) modeling object-object spatial relations on natural images instead of 3D, which helps avoiding additional instrumentation for object tracking and the need for large collection of corresponding 3D shapes and relational data *ii*) reasoning about the best way to place objects to reproduce a spatial relation by estimating pixelwise, non-parametric distributions, without the use of priors *iii*) leveraging auxiliary learning to overcome the problem of unavailability of ground-truth pixelwise annotation of spatial relations and thus receive the training signal by classifying hallucinating scene representations.

We feel that this is a promising first step towards enabling a shared understanding between humans and robots. In the future, we plan to extend our approach to incorporate understanding of referring expressions to develop a pick-and-place system that follows natural language instructions.

Chapter 4

Composing Pick-and-Place Tasks By Grounding Language

The content of this chapter has been published in [120]:

O. Mees and W. Burgard
Composing Pick-and-Place Tasks By Grounding Language
International Symposium on Experimental Robotics (ISER), 2020
Springer Proceedings in Advanced Robotics book series (SPAR, volume 19)
ISBN: 978-3-030-71151-1

I am the main author and developed and implemented the method. Further, I carried out the experiments and wrote the publication. Wolfram Burgard provided general consultation and helped revising the publication.

Abstract

Controlling robots to perform tasks via natural language is one of the most challenging topics in human-robot interaction. In this work, we present a robot system that follows unconstrained language instructions to pick and place arbitrary objects and effectively resolves ambiguities through dialogues. Our approach infers objects and their relationships from input images and language expressions and can place objects in accordance with the spatial relations expressed by the user. Unlike previous approaches, we consider grounding not only for the picking but also for the placement of everyday objects from language. Specifically, by grounding objects and their spatial relations, we allow specification of complex placement instructions, e.g. "place it behind the middle red bowl". Our results obtained using a real-world PR2 robot demonstrate the effectiveness of our method in understanding pick-and-place language instructions and sequentially composing them to solve tabletop manipulation tasks. Videos are available at `http://speechrobot.cs.uni-freiburg.de`

4.1 Introduction

As robots become ubiquitous across human-centered environments the need for natural and effective human-robot communication grows. Natural language provides a rich and intuitive way for humans and robots to interact due to the possibility of referring to abstract concepts. Moreover, many real-world tasks can be effectively described by a series of language instructions. In this work, we aim to develop an approach that enables a robot to solve complex manipulation tasks by understanding a series of unconstrained language expressions characterizing pick-and-place commands. To do so, the robot has to locate unconstrained object categories based on arbitrary natural language expressions, known as referring expression comprehension, and understand spatial relations to generate object placing locations. In other words, the robot needs to "ground" the referred objects and their spatial relations from language in its world model.

However, understanding unconstrained language instructions is challenging due to the complexity and wide variety of abstract concepts expressed via human language, e.g. "fetch the yellow thing" and "place it left of the bottom object". Moreover, the expression might contain ambiguities because there are several "yellow things" in which case the robot should be able to resolve the ambiguity through dialogue, as shown in Figure 4.1. Finally, the robot needs to reason about where to place the "yellow thing" relative to the "leftmost container" in order to reproduce the spatial relation "right", which is inherently ambiguous as natural language placement instructions do not uniquely identify a location in a scene.

In this paper, we propose the first comprehensive system for controlling robots to perform tabletop manipulation tasks by sequentially composing unconstrained pick-and-place language instructions. Our approach consists of two neural networks. The first network learns to segment objects in a scene and to comprehend and generate referring expressions. The second network estimates pixelwise object placement probabilities for a set of spatial relations given an input image and a reference object. The interplay between both networks allows for an effective grounding of object semantics and their spatial relationships, without assuming a predefined set of object categories. We demonstrate the effectiveness of our

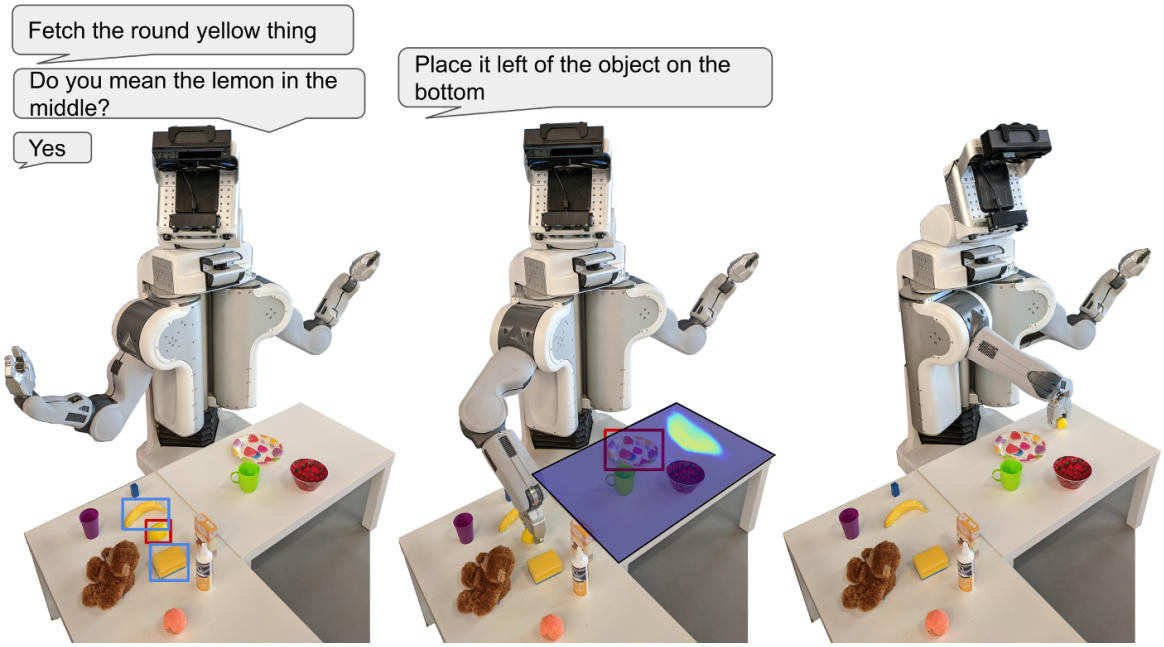


Figure 4.1: The goal of our work is to control a robot to perform tabletop manipulation tasks via natural language instructions. Our approach is able to segment objects in the scene, locate the objects referred to in language expressions, solve ambiguities through dialog and place objects in accordance with the spatial relations expressed by the user.

approach by enabling non-expert users to instruct tabletop manipulation tasks to a robot, based on sequences of pick-and-place speech commands.

4.2 Related Work

Our work is primarily concerned with the task of grounding natural language instructions and spatial relations in the context of the robot's world model [36]. Locating entities in images based on language is closely related to object recognition. Previous works in robotics [32, 121] have addressed semantic object retrieval by training classifiers to recognize predefined object categories. These approaches are limited in real-world scenarios as they are not capable of handling variation in the users natural language descriptions and are restricted to a small number of objects.

Spatial relations also play a crucial role in understanding natural language instructions [108, 109], as objects are often described in relation to others in tasks such as object placing [88, 89, 91] or human robot interaction [32, 104, 122]. Concretely, spatial relations help the robot disambiguate multiple instances of the same object and to define target areas for placing the picked objects. In our previous work, we introduced a novel method to predict pixelwise object placement probability distributions for a set of commonly used prepositions in natural language [88]. In contrast, we relax the assumption of having a single reference object on the tabletop and add a grounding model to effectively place arbitrary objects in a scene that contains multiple objects.

Recently, there has been significant progress made towards systems that can demonstrate their visual understanding by generating or responding to natural language in the context

of images [28, 123, 124, 125, 126]. To learn joint visual-linguistic representations, state-of-the-art approaches use convolutional neural networks to encode visual features and recurrent neural networks to process language, replacing traditional handcrafted visual features and language parsers. We leverage advances in modular networks [112, 127, 128] for referential expression comprehension. This allows decomposing language into modular components related to subject appearance, location, and relationship to other objects, flexibly adapting to expressions containing different types of information in an end-to-end fashion.

Most related to our approach are the works by Shridhar *et al.* [104] and Hatori *et al.* [109], as both use an interactive fetching system to localize objects mentioned in referring expressions with bounding boxes. We tackle temporally more extended tasks, using our model which enables complex object placement commands such as "place the cup on top of the leftmost box". Notably, sequentially composing pick-and-place language instructions can lead to desirable high-level behaviours, such as tidying up a tabletop or table setting for example. Finally, in contrast to the template-based picking approaches of prior interactive fetching systems [104, 109] we leverage state-of-the-art methods for grasping novel objects with 6-DOF grasps [129].

4.3 Method Description

In this section we describe the technical details of our method to control a robot to perform tabletop manipulation tasks via natural language instructions. Our approach relies on two models: a grounding model that identifies the most likely object referred by a language instruction and a neural network that predicts object placing locations conditioned on a set spatial relation. An overview of the system is given in Figure 4.2.

4.3.1 Target Object Selection

We start off by detecting and segmenting all objects in the scene. We train a semantic segmentation network based on Mask-RCNN [130] with a Resnet-101 backbone, which extracts a set of region proposals or object candidates o_i from an image. After all objects on the scene are recognized, we need to identify which object the user is referring to in its language instruction. Given an input image I and expression r, the target object selection is formulated as a task to find the best bounding box from the set of predicted candidate boxes $O = \{o_i\}_{i=1}^{N}$. Our grounding model is based on MAttNet [112], a modular referring expression comprehension network. To enable human-robot communication in cases of ambiguous instructions, we have extended it to support the generation of self-referential expressions, described in Section 4.3.2.

The candidate regions are encoded by a neural network consisting of three modular grounding components related to subject appearance, location and relationship to other objects. These modules combine image features encoded by a Resnet-101 network with relational and geometric features pertaining to the neighborhood or context of each candidate region. The language expression r is encoded in a word embedding layer, which encodes each word in the input sentence to a vector representation, followed by a bi-directional Long Short-Term Memory (LSTM) and a fully-connected (FC) layer. Additionally, the language network learns two types of attention: attention weights that are computed on each word for each module and are summarized as phrase embedding $q^m \mid m \in \{\text{subj, loc, rel}\}$, and

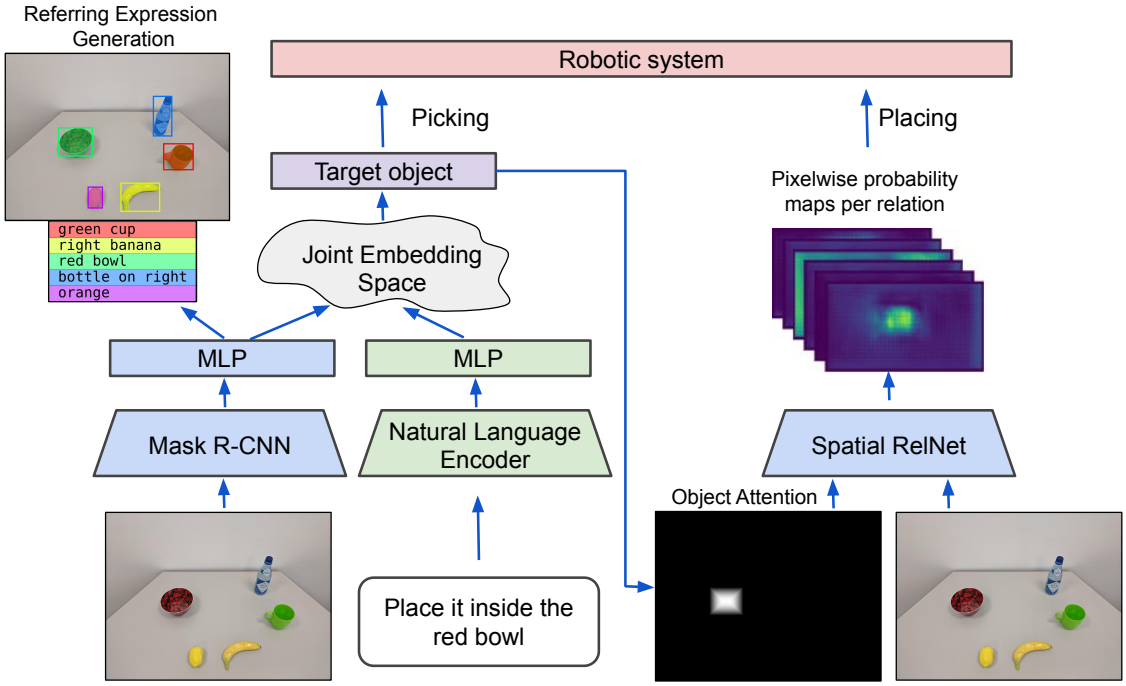


Figure 4.2: Overview of the system architecture. Our grounding network processes the input sentence and visual object candidates detected with Mask-RCNN [130] and performs referential expression comprehension. Additionally, it generates referential expressions for each object candidate to disambiguate unclear instructions. Once the reference object of a relative placement instruction has been identified, a second network predicts object placing locations for a set of spatial relations.

module weights $[w_{subj}, w_{loc}, w_{rel}]$ that estimate how much each module contributes to the overall expression score. Each visual module computes scores for each object candidate by calculating the cosine similarity between the vector representation of the instruction, and that of the candidate image region. Finally, the output module takes a weighted average of these scores to get an overall matching score $S(o_i \mid r) = w_{subj} S(o_i \mid q^{subj}) + w_{loc} S(o_i \mid q^{loc}) + w_{rel} S(o_i \mid q^{rel})$. During training, we sample triplets consisting of a positive match (o_i, r_i) and two random negative samples (o_i, r_j) and (o_k, r_i), where o_k is some other object and r_j is an expression describing some other object in the same image to apply a hinge loss:

$$\begin{aligned} \mathcal{L}_1 = \sum_i [&\lambda_1 \max(0, m_1 + S(o_i \mid r_j) - S(o_i \mid r_i)) \\ &+ \lambda_2 \max(0, m_1 + S(o_k \mid r_i) - S(o_i \mid r_i))]. \end{aligned} \tag{4.1}$$

4.3.2 Resolving Ambiguities

If the referred object cannot be uniquely identified by the grounding model, the system needs to ask for clarification from the human operator. Inspired by recent advances in image caption generation and understanding [125, 131, 132], we incorporate a LSTM based captioning module to our grounding network that allows the robot to describe each detected object with a natural language description. Our referring expression generation module is jointly trained with our grounding network and shares the features used in the three modules related

to subject appearance, location and relationship to other objects. Specifically, the visual target object representation v_i^{vis} is modeled by a concatenation of the ResNet-101 C3 and C4 features, followed by one FC layer which is shared with the comprehension network and one exclusive FC layer. To facilitate the generation of referential expressions that contain location information, such as "the cup in the middle", we leverage the representation learned by the location module v_i^{loc}. This module combines a 5-d vector representing the top-left position, bottom-right position and relative area to the image for the candidate object, together with a relative location encoding of up to five surrounding objects of the same category. Finally, we integrate the output of the relationship module v_i^{rel}, which encodes the appearance and localization offsets of up to five category-agnostic objects in the targets surroundings to enable modeling sentences such as "the teddy bear on top of the box". The final visual representation for the target object is then a concatenation of the above features $v_i = [v_i^{vis}, v_i^{loc}, v_i^{rel}]$. The model is trained to generate sentences r_i by minimizing the negative log-likelihood:

$$\mathcal{L}_2 = -\sum_i \log P(r_i \mid v_i). \tag{4.2}$$

To generate discriminative sentences, we use a Maximal Mutual Information constraint proposed by Mao *et al.* [131] that encourages the generated expression to describe the target object better than the other objects within the image. Concretely, given a positive match (o_i, r_i) we sample a negative (o_k, r_i), where o_k is some other object, and optimize the following max-margin loss:

$$\mathcal{L}_3 = \sum_i [\lambda_3 \max(0, m_2 + \log P(r_i \mid v_k) - \log P(r_i \mid v_i)). \tag{4.3}$$

In order to detect if an instruction is ambiguous, we leverage the max-margin loss the comprehension model is trained with. Concretely, during training the max-margin loss aims to guarantee that every correct pair of a sentence and an object has scores by a margin m_1 than any other pair with a wrong object or sentence. Therefore, if at test time there are more than one objects within that threshold, we consider them potential targets. For each candidate we generate multiple self-referential expressions via beam search and use the comprehension module to rerank these expressions and select the least ambiguous expression, similar to Yu *et al.* [132]. We then let the system ask the human "Do you mean ...?". After asking the question, the user can respond "yes" to choose the referred object or "no" to continue iterating through other possible objects. Alternatively, the user can provide a specific correcting response to the question, e.g., "no, the banana on the right", in which case we re-run our grounding module.

4.3.3 Relational Object Placement

Once an object has been picked, our system needs to be able to place it in accordance with the instructions from the human operator. We combine referring expression comprehension with the grounding of spatial relations to enable complex object placement commands such as "place the ball inside the left box". Given an input image I of the scene and the location of the reference item, identified with our aforementioned grounding module, we generate pixelwise object placement probabilities for a set of spatial relations by leveraging the Spatial-RelNet architecture we introduced in our previous work [88]. We consider pairwise relations and express the subject item as being *in relation to* the reference

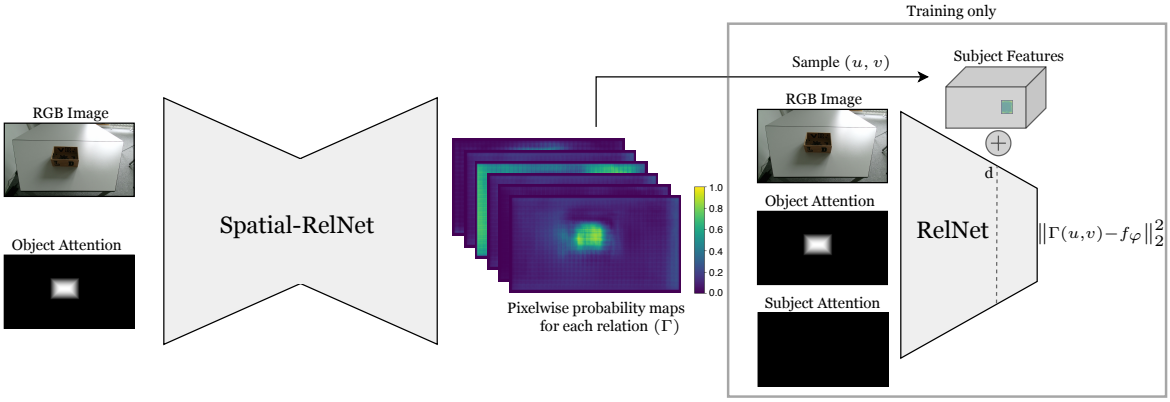


Figure 4.3: Our Spatial-RelNet [88] network processes the input RGB image and an object attention mask to produce pixelwise probability maps Γ over a set of spatial relations. During training, we sample locations (u, v) according to Γ, implant inside an auxiliary classifier network at the sampled locations high level features of objects and classify the hallucinated scene representation to get a learning signal for Spatial-RelNet. At test time the auxiliary network is not used.

item. We model relations for a set of commonly used natural language spatial prepositions $C = \{\texttt{inside}, \texttt{left}, \texttt{right}, \texttt{in front}, \texttt{behind}, \texttt{on top}\}$. As natural language placement instructions do not uniquely identify a location in a scene, Spatial-RelNet predicts non-parametric distributions to capture the inherent ambiguity. A key challenge to learning such pixelwise spatial distributions is the lack of ground-truth data. Spatial-RelNet overcomes this problem by leveraging a novel auxiliary learning formulation, as shown in Figure 4.3. During training, pixel locations (u, v) are sampled according to the probability maps Γ produced by Spatial-RelNet. To get the learning signal, high level features of objects are implanted into a pretrained auxiliary classifier f_φ to compute a posterior class probability over relations. This way, we can reason over what relation would most likely be formed if we placed an object at the given location.

4.4 System Implementation

4.4.1 Machine Learning Setup

During training, we sample the same triplets for both the object comprehension module and the expression generation module. We set the margin $m_1 = 0.1$ for the comprehension ranking and $m_2 = 1.0$ for the generation loss. We additionally use MAttNet's auxiliary visual attribute classification loss. We use the Adam optimizer to train the joint model with an initial learning rate of 0.0004. For the contrastive pairs, we set $\lambda_1 = 1$, $\lambda_2 = 1$ and $\lambda_3 = 0.1$. We make the word embedding of the comprehension and generation modules shared to reduce the number of parameters. For implementation details of Spatial-RelNet, we refer to the original paper [88].

4.4.2 Robot Setup

To pick an object from language, we first identify the object with our grounding model and extract the corresponding segmentation mask of the selected object. We use an Amazon Echo Dot device to synthesize the voice instructions. We localize the object in 3D space and generate grasp poses with Grasp Pose Detection (GPD) [129], which predicts a series of 6-DOF candidate grasp poses given a 3D point cloud for a 2-finger grasp. The reachability of the proposed candidate grasps are checked using MoveIt!, and the highest quality reachable grasp is executed with the PR2 robot. For placing the object, we first sample a location from the spatial distribution predicted by our Spatial-RelNet model. We rely on keyword spotting to select the corresponding predicted distribution. Next, we localize the pixel coordinate in 3D space and plan a top-down grasp pose to the calculated 3D point. Finally, the end-effector is moved above the desired location and then the gripper is opened to complete the placement.

4.5 Experiments

We evaluate our approach under two settings. First, we evaluate the capability of our approach to comprehend and generate referring expressions for a wide variety of objects on the RefCOCO dataset [123]. Next, we evaluate the ability of our robotics system to follow pick-and-place language instructions in human-robot experiments.

4.5.1 RefCOCO Benchmark

The RefCOCO dataset contains images and corresponding referring expressions that uniquely identify a wide variety of objects in the images. We compare our grounding networks ability to comprehend and generate referring expressions against several strong baselines on Table 4.1. For evaluating the comprehension, we compute the intersection-over-union (IoU) of the selected region with the ground-truth bounding box, considering IoU > 0.5 a correct comprehension. To evaluate the generation module, we leverage standard machine translation metrics commonly used in image captioning, such as METEOR and CIDEr. We observe that by jointly training the comprehension and language generation modules, they regularize each other and improve their respective performances, demonstrating the effectiveness of multitask learning [13, 132, 133].

	RefCOCO comprehension			RefCOCO generation			
	val	TestA	TestB	TestA		TestB	
				Meteor	CIDEr	Meteor	CIDEr
Mao [131]	-	63.15	64.21	-	-	-	-
INGRESS [104]	77	76.7	77.7	-	-	-	-
SLR [109, 132]	79.56	78.95	80.22	0.268	0.697	0.329	1.323
MAttNet [112]	85.65	85.26	84.57	-	-	-	-
Ours	**86.15**	**87.18**	**85.36**	**0.29**	**0.753**	**0.33**	**1.33**

Table 4.1: Referring expression comprehension and generation on the RefCOCO dataset, with human-annotated ground-truth object regions.

4.5.2 Robot Experiments

We evaluate our approach on two real-world scenarios: picking and placing objects according to user defined object arrangements and a tidy-up task. We will first describe the setup of the object arrangement experiment. Our study involved 4 participants recruited from a university community[1]. The robots workspace contained two tables, as shown in Figure 4.1. One table contained previously unseen objects in clutter. The second table contained a single reference object. The average number of objects on the cluttered table was 5.6. The participants were asked to instruct a PR2 robot to arrange a desired target scene by picking objects from the cluttered table and using relational expressions to place them on the second table. In addition to the robots RGB-D camera we placed a second camera in front of the cluttered table and performed online registration to compute a global point cloud. The tidy-up task consisted of iteratively picking 4 colored objects from the cluttered table and placing the same colored objects on the left container and the remaining objects on the right container. In this experiment we were interested in evaluating the number of actions the robot has to take to complete the task, given unambiguous instructions.

Table 4.2 shows the performance of our approach on a PR2 robot for the first experiment. Our approach achieves a 78.3% target object selection accuracy and a 85.7% accuracy on selecting the reference object the placing will be relative to. The higher accuracy of the latter

	Target Object Selection	Target Object Grasping	Placing Base Grounding	Placing Success	Avg. Number of Feedback	Pick and Place
Ours	78.3% (47/60)	74.4% (35/47)	85.7% (30/35)	83.3% (25/30)	0.63 (60/95)	63% (60/95)

Table 4.2: Performance of our approach on a real robot platform following natural language instructions to pick and place objects in a tabletop scenario.

is due to fewer candidate objects being on the placing table and the participants preferring to use ambiguous expressions for the picking instructions. The robot took ~ 20 seconds to complete an action from the moment the human started to speak. We report a grasping performance of 74.4% with GPD. We find that some objects such as mugs are particularly difficult for GPD as it often fails to find feasible grasps due to either occluded object parts or noisy measurements on thin structures such as rims. Our object placement approach achieves a success rate of 85.7%. We observe some failure cases for large object placements, because of missing 3D priors of the objects to be placed. Thus, when placing a big box left of a small box, it is possible that the chosen placement results in the big box partially ending up on top of the small box. For the tidy up task, we report a mean task length of 14.4 actions, due to several re-grasp attempts. Overall, our results demonstrate the ability of our approach to allow non-expert users to instruct tabletop manipulation tasks based on sequences of pick-and-place speech commands.

4.6 Conclusions and Discussion

In this paper, we proposed the first robotic system that allows non-expert users to instruct tabletop manipulation tasks by sequentially composing unconstrained pick-and-place language instructions and can clarify a human operator's intention through dialogue. We

[1]Further quantitative experiments were infeasible at time of submission due to COVID-19.

demonstrate the effectiveness of our approach to encode high-level behaviours in a highly challenging, realistic environment. Even though we are far from achieving robots that can learn to relate human language to their world model, we hope our work is a step in this direction.

While the experimental results are promising, our approach has several limitations. First, relative object placement instructions do not allow for fine-grained target specification due to its inherent ambiguity. Addressing this issue would require learning user preferences from feedback [134]. Second, we observe some failure cases for large object placements, because of missing 3D priors of the objects to be placed. Integrating 3D priors is a natural extension to enable optimizing placement poses [135] and to reason over the effects of actions on the scene [136]. Third, we find that GPD often fails to find feasible grasps due to either occluded object parts or noisy measurements. Integrating methods that can complete occluded scene regions [107, 137] or generate more diverse grasps [138] might help alleviating these problems. Finally, our approach is limited to tabletop tasks that can be characterized by pick-and-place actions. An exciting area for future work may be one that not only grounds object semantics and spatial relations, but also grounds actions in order to learn complex behaviours with language conditioned continuous control policies [139, 140].

Chapter 5

CALVIN: A Benchmark for Language-Conditioned Policy Learning for Long-Horizon Robot Manipulation Tasks

The content of this chapter has been published in [141]:

O. Mees, L. Hermann, Erick Rosete-Beas and W. Burgard
CALVIN: A Benchmark for Language-Conditioned Policy Learning for Long-Horizon Robot Manipulation Tasks
IEEE Robotics and Automation Letters (RAL), 2022
DOI: 10.1109/LRA.2022.3180108

Me and Lukas Hermann share the main authorship as we jointly developed and implemented the CALVIN benchmark. I developed the main idea of the paper, implemented the initial simulated environments and baselines, training code and wrote most of the paper. Lukas implemented most of the robot control code and the evaluation of the policies and contributed speed improvements for loading the dataset. We carried out the simulated experiments jointly. Erick contributed the final 3D models for the simulated environments.

Abstract

General-purpose robots coexisting with humans in their environment must learn to relate human language to their perceptions and actions to be useful in a range of daily tasks. Moreover, they need to acquire a diverse repertoire of general-purpose skills that allow composing long-horizon tasks by following unconstrained language instructions. In this paper, we present Composing Actions from Language and Vision (CALVIN) (Composing Actions from Language and Vision), an open-source simulated benchmark to learn long-horizon language-conditioned tasks. Our aim is to make it possible to develop agents that can solve many robotic manipulation tasks over a long horizon, from onboard sensors, and specified only via human language. CALVIN tasks are more complex in terms of sequence length, action space, and language than existing vision-and-language task datasets and supports flexible specification of sensor suites. We evaluate the agents in zero-shot to novel language instructions and to novel environments. We show that a baseline model based on multi-context imitation learning performs poorly on CALVIN, suggesting that there is significant room for developing innovative agents that learn to relate human language to their world models with this benchmark. Code, dataset and trained models available at `http://calvin.cs.uni-freiburg.de`.

5.1 Introduction

A long-standing goal for robotics and embodied agents is to build systems that can perform tasks specified in natural language. Concepts expressed in natural language provide humans with an intuitive way to represent, summarize, and abstract diverse knowledge skills. By means of abstraction, concepts such as "open the drawer and push the middle object into the drawer" can be extended to a potentially infinite set of new and unseen entities. Additionally, humans leverage concepts to describe complex tasks as sequences of natural language instructions. This stands in contrast to current robots, which typically lack this generalization ability and learn individual tasks one at a time. Moreover, multi-task learning approaches traditionally assume that tasks are specified to the agent at test time via mechanisms such as goal images [18] and one-hot skill selectors [19, 142] that are not practical for non-expert users to instruct robots in everyday real-world settings. As robots become ubiquitous across human-centered environments the need for intuitive task specification grows: how can we scale robot learning systems to autonomously acquire general-purpose knowledge that allows them to compose long-horizon tasks by following unconstrained language instructions?

To address this problem we present CALVIN, a new open-source simulated benchmark that links human language to robot motor skills, behaviors, and objects in interactive visual environments. In this setting, a single agent must solve complex manipulation tasks by understanding a series of unconstrained language expressions in a row, e.g., "open the drawer . . . pick up the blue block . . . push the block into the drawer . . . open the sliding door". Furthermore, to evaluate the agents' ability for long-horizon planning, agents in this scenario are expected to be able to perform any combination of subtasks in any order. CALVIN has been developed from the ground up to support training, prototyping, and validation of language-conditioned continuous control policies over a range of four indoor manipulation

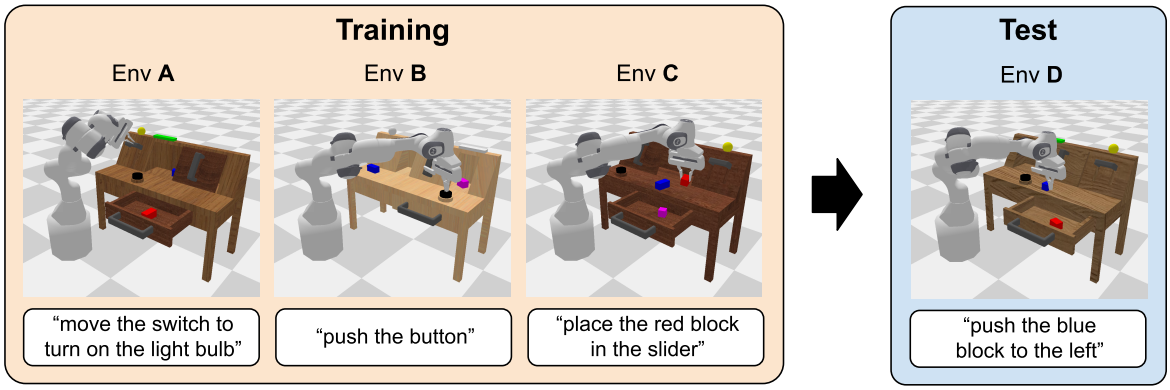


Figure 5.1: CALVIN is a benchmark to learn many long-horizon language-conditioned tasks over a range of four manipulation environments, designed to be diverse yet carry shared structure, from multimodal onboard sensor observations. In the most difficult evaluation, the methods must generalize to unseen entities by training on a large interaction corpora covering three environments and testing on an unseen scene.

environments, visualized in Figure 1. CALVIN includes $\sim$24 hours teleoperated unstructured *play* data together with 20K language directives. Unscripted playful interactions have the advantage of being task-agnostic, diverse, and relatively cheap to obtain [18, 143]. The simulation platform supports a range of sensors commonly utilized for visuomotor control: RGB-D images from both a static and a gripper camera, proprioceptive information, and vision-based tactile sensing [144]. We believe that this flexible sensor suite will allow researchers to develop improved multimodal agents that can solve many tasks in real-world settings. This is the first public benchmark of instruction following, to our knowledge, that combines: natural language conditioning, multimodal high-dimensional inputs, 7-DOF continuous control, and long-horizon robotic object manipulation. We provide an evaluation protocol with evaluation modes of varying difficulty by choosing different combinations of sensor suites and amounts of training environments. This effort joins the recent efforts to standardize robotics research for better benchmarks and more reproducible results. To open the door for future development of agents that can generalize abstract concepts to unseen entities the same way humans do, we include a challenging zero-shot evaluation by training on large play corpora covering three environments and testing on an unseen scene. The language instructions used for testing are not included in the training set and represent novel ways of describing the manipulation tasks seen during training.

To establish baseline performance levels, we evaluate the multi-context imitation learning (MCIL) approach that uses relabeled imitation learning to distill many reusable behaviors into a goal-directed policy [145]. This model is not effective on the complex long horizon robot manipulation tasks in CALVIN. While it achieves up to 53.9% success rate in short horizon tasks, it performs poorly in the long-horizon setting. We note that there is no constraint to use imitation learning approaches to solve CALVIN tasks, as approaches that use reinforcement learning to learn language-conditioned policies can also be applied [146].

In summary, CALVIN facilitates learning models that translate from language to sequences of motor skills in a realistic simulation environment. This benchmark captures many challenges present in real-world settings for relating human language to robot actions and perception for accomplishing long-horizon manipulation tasks. Models that can overcome these

challenges will begin to close the gap towards scalable, general-purpose, language-driven robotics.

5.2 Related Work

Natural language processing has recently received much attention in the field of robotics [24], following the advances made towards learning groundings between vision and language [123, 147]. Recent successes in human-robot interaction include an interactive fetching system to localize objects mentioned in referring expressions [104, 108, 109, 148, 149] or grounding not only objects, but also spatial relations to follow language expressions characterizing pick-and-place commands [120, 150, 151]. By contrast, CALVIN tasks require grounding language to a wide variety of general-purpose robot skills. Prior work on mapping language and vision to actions has been studied mostly in restricted environments [26, 27] and simplified actuators with discrete motion primitives [28, 29, 30]. A growing body of work also looks at learning language-conditioned policies for continuous visuomotor-control in 3D environments via imitation learning [21, 54, 145] or reinforcement learning [140, 146, 152]. These approaches typically require offline data sources of robotic interaction, such as teleoperation or autonomous exploration data, together with post-hoc crowd-sourced language labels. However, the lack of standardized benchmarks and algorithm implementations, makes it difficult to compare approaches and to facilitate future research.

The most closely related benchmark to ours is ALFRED [29], which contains language instructions for combined navigation and manipulation tasks with seven predefined action primitives. In CALVIN, rather than classifying predefined actions, the agent must learn to acquire a diverse repertoire of general-purpose skills that allows composing long-horizon tasks by following unconstrained language instructions in closed loop control. Our tabletop environments are inspired by the one shown in Lynch *et al.* [145] in order to have a fair comparison to their MCIL approach, which we implement to establish baseline performance levels. We note that although considered a state-of-the-art approach, no public implementation of MCIL is available. In contrast to their work, CALVIN contains more subtasks (34 vs 18), longer long-horizon evaluation sequences (5 vs 4), provides a range of sensors commonly utilized for visuomotor control and allows testing zero-shot generalization by leveraging a range of four manipulation environments and unseen language instructions. Finally, CALVIN goes beyond the original MCIL setup by adding a challenging visual grounding problem, where similar language instructions for differently colored blocks are given and the agent needs to identify which block is meant.

5.3 CALVIN

The aim of the CALVIN benchmark is to evaluate the learning of long-horizon language-conditioned continuous control policies. In this setting, a single agent must solve complex manipulation tasks by understanding a series of unconstrained language expressions in a row, e.g., "open the drawer... pick up the blue block... now push the block into the drawer... now open the sliding door". We note that in the benchmark we only allow feasible sequences that can be achieved from a predefined initial environment state. The CALVIN benchmark consists of three key components, which are:

Observation Space	
RGB static camera	$200 \times 200 \times 3$
Depth static camera	200×200
RGB gripper camera	$84 \times 84 \times 3$
Depth gripper camera	84×84
Tactile image	$120 \times 160 \times 2$
Proprioceptive state	EE position (3) EE orientation (3) Gripper width (1) Joint positions (7) Gripper action (1)
Action Space	
Absolute cartesian pose (w.r.t. world frame)	EE position (3) EE orientation (3) Gripper action (1)
Relative cartesian displacement (w.r.t. gripper frame)	EE position (3) EE orientation (3) Gripper action (1)
Joint action	Joint positions (7) Gripper action (1)

Figure 5.2: Observation and action spaces supported by CALVIN.

1. CALVIN Environment

2. CALVIN Dataset

3. CALVIN Challenge

5.3.1 The CALVIN Environment

CALVIN features four different, yet structurally related environments (A, B, C, D) so that it can be used for general playing as well as evaluating specific tasks. The environments contain a 7-DOF Franka Emika Panda robot arm with a parallel gripper and a desk with a sliding door and a drawer that can be opened and closed. On the desk, there is a button that toggles a green light and a switch to control a light bulb. Besides, there are three different colored and shaped rectangular blocks. To better evaluate the generalization capabilities of the learned language groundings, all environments have different textures and all static elements such as the sliding door, the drawer, the light button, and switch are positioned differently. The position of the desk, robot, and the static camera is the same in all environments. Due to the general difficulty of language-conditioned multi-task closed-loop control, we reduced the complexity of the objects to unicolored primitive shapes. If future advances in this field require new challenges we will reflect this by extending CALVIN to environments with more realistic and diverse objects. Physics are simulated using the PyBullet physics engine [153], which supports fast GPU rendering for large-scale parallel data collection.

Observation and Action Space

Unlike prior work which relies on RGB images from an egocentric camera to perceive its surroundings [18, 145], CALVIN offers a range of sensors that can be used to develop and prototype agents that learn task-agnostic control in the real world. Concretely, the agent perceives its surroundings from RGB-D images from both a fixed and a gripper camera. It additionally has access to a vision-based tactile sensor [144] and to continuous internal proprioceptive sensors. A visualization of the supported sensor modalities is shown in Figure 5.3. The agent must perform closed-loop continuous control to follow unconstrained language instructions characterizing complex robot manipulation tasks, sending continuous actions to the robot at 30hz. In order to give researchers and practitioners the freedom to experiment with different action spaces, CALVIN supports absolute and relative cartesian actions, as well as actions in joint space. We encourage the community to study flexible combinations of observation and action spaces since the tasks require a varying degree of precise control vs. coarse locomotion. While the static camera and absolute cartesian actions are the natural choices for tasks that call for a complete traversal of the environment from one side to another, the gripper camera and relative actions (w.r.t to the gripper frame) allow more fine-grained control for tasks like stacking or grasping. Tactile information can become important when the task requires the robot to maintain a stable grasp on the handle while moving the sliding door to the side. See Fig. 5.2 for a description of the observation and action dimensionalities.

Tasks

We define 34 specific tasks (see Fig. 5.4) that can be achieved in each one of the environments The environment has the functionality to automatically detect which one of the tasks has been completed in a sequence of steps, which can serve as a sparse reward for reinforcement learning agents. The criterion for task completion is defined in terms of a change in the environment state between the initial and final step of a sequence. This also enables the automatic task detection in any variable-length sequence of offline data, since the environment can be reset to the state of each one of the recorded frames.

5.3.2 The CALVIN Dataset

Unstructured Demonstrations

Learning generally requires exposure to diverse training data. To effectively cover state space, we collect twenty-four hours of teleoperated "play" data in four environments with a HTC Vive VR headset, spending an approximately equal time of six hours in each environment. This corresponds to $\sim$2.4M interaction steps and $\sim$40M short-horizon windows for relabeled goal conditioned imitation learning [52, 57], each spanning 1-2 seconds. In this setting, an operator is not constrained to a set of predefined tasks, but rather engages in behavior that satisfies their own curiosity or some other intrinsic motivation. Unscripted playful interactions have the advantage of being task-agnostic, diverse, and relatively cheap to obtain [18, 143]. We asked three people to collect data, and these users were untrained and given no information about the downstream tasks. The only guideline we gave data collectors was to "explore the environment without dropping objects from the table". This includes

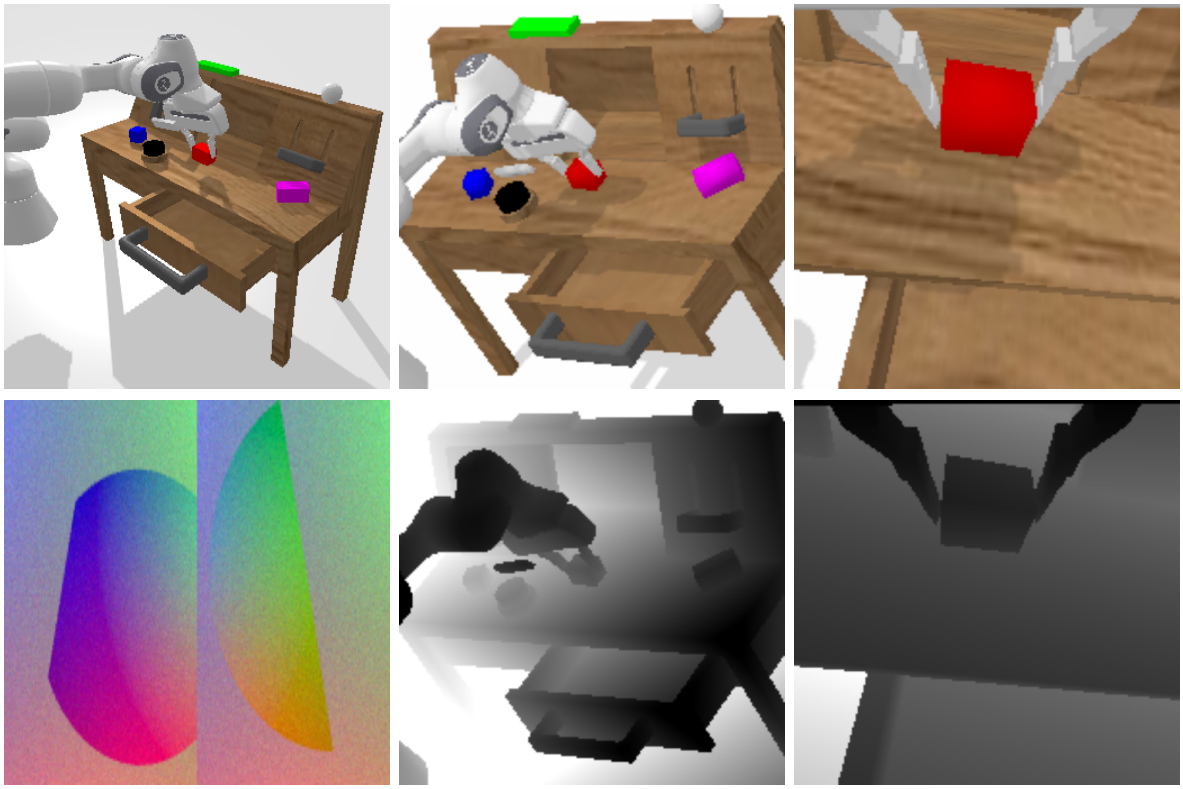


Figure 5.3: CALVIN supports a range of sensors commonly utilized for visuomotor control: RGB-D images from both a static and a gripper camera, proprioceptive information, and vision-based tactile sensing (bottom-left).

picking up and placing objects, opening, and closing drawers, sliding doors, pushing buttons, operating switches and undirected actions. This style of data is very different from commonly used task-specific data, which only consists of expert trajectories. Playful interaction data by design is free-form, so there are no categories associated with the data. This kind of unstructured data is useful because it contains exploratory and sub-optimal behaviors that are critical to learning generalizable and robust representations, e.g., enabling retrying behavior. While expert demonstrations often only show one of the many possible ways to solve a task, play data is richer in the sense that it covers the multimodal space of possible solutions. However, as opposed to expert demonstrations, in play data some task instances naturally occur less frequently than others, especially those that have the completion of another task as a prerequisite.

Language Instructions

Approaches that learn language-conditioned continuous control policies typically require post-hoc crowd-sourced natural language labels aligned with its corresponding robot interaction data [145, 146]. Instead of relying entirely on crowd-sourced annotations, we collect over 400 crowd-sourced natural language instructions corresponding to over 34 tasks and label episodes procedurally using the recorded environment state of the CALVIN dataset. We note that using this labeling scheme, only sequences that display meaningful skills are labeled with language annotations. We visualize example language annotations in Fig. 5.4. In order

Task	Natural language instructions
rotate red block right	"rotate the red block 90 degrees to the right" "turn the red block right"
push blue block left	"go slide the blue block to the left" "push left the blue block"
move slider left	"grasp the door handle, then slide the door to the left" "slide the door to the left"
open drawer	"grasp the handle of the drawer and open it" "go open the drawer"
lift red block	"lift the red block from the table" "pick up the red block"
pick pink block from drawer	"pick up the pink block lying in the drawer"
place in slider	"put the grasped object in the slider"
stack blocks	"stack blocks on top of each other"
unstack blocks	"collapse the stacked blocks" "go to the tower of blocks and take off the top one"
turn on light bulb	"toggle the light switch to turn on the light bulb"
turn off green light	"push the button to turn off the green light"

Figure 5.4: Example crowd-sourced natural language instructions to specify manipulation tasks in CALVIN.

to simulate a real-world scenario where it might not be possible to pair all the collected robot experience with crowd-sourced language annotations, we annotate only 1% of the recorded robot interaction data with language instructions. Besides language instructions, we provide precomputed language embeddings extracted from MiniLM [154]. MiniLM distills a large Transformer based language model and is trained on generic language corpora (e.g., Wikipedia). It has a vocabulary size of 30,522 words and maps a sentence of any length into a vector of size 384. We note that there exist many choices for encoding raw text in a semantic pre-trained vector space and encourage the community to experiment with different choices to solve for CALVIN tasks.

Long-horizon language instructions
"turn on the led" → "open drawer" → "push the blue blue block → "pick up the blue block " → "place in slider"
"move slider left" → "lift red block from slider" → "stack blocks" → "toggle light" → " collapse stacked blocks"
"open drawer" → "push block in drawer" → "pick object from drawer" → "stack blocks" → "close drawer"

Figure 5.5: Example long-horizon language tasks sequences evaluated in CALVIN. We show the abbreviated subtask names instead of the full language annotations due to space constraint.

5.3.3 The CALVIN Challenge

CALVIN combines the challenging settings of open-ended robotic manipulation with open-ended human language conditioning. For example, a robot that is instructed to "place the blue block inside the drawer" must be able to relate language to its world model. Concretely, it needs to learn to identify how a blue block and a drawer look like in its multimodal perceptual observations[1], and then it needs to reason over the best sequence of actions to "place inside the drawer". Ideally, a general-purpose robot should be able to perform any combination of tasks instructed with natural language in any order. Thus, to accelerate progress in language-driven robotics, we present a set of evaluation protocols of varying difficulty by choosing different combinations of sensor suites and amounts of training environments.

Training and Test Environments

CALVIN offers three combinations of training and test environments with varying difficulty:

Single Environment: Training in a single environment and evaluating the policy in the same environment. This corresponds to the setting of Lynch *et al.* [145].

Multi Environment: Training in all four environments and evaluating the policy in one of them. This poses an additional challenge since the policy has to generalize to multiple textures and different locations of the sliding door, button, and switch. On the other hand, the agents can benefit from increased data.

Zero-Shot Multi Environment: To open the door for future development of agents that can generalize abstract concepts to unseen entities the same way humans do, we include a challenging zero-shot evaluation by training in three environments and evaluating the policy in the fourth unseen one. This is the hardest combination since the policy has never seen the test environment during training. However, all elements of the scene were present in different locations in the training environments. While highly challenging, we believe it aligns well with test-time expectations for service robots to be useful in a range of daily tasks in everyday environments. Concretely, in CALVIN agents need to generalize to a room where the environment has different textures and all static elements such as the sliding door, the drawer and the light turning button and switch are positioned differently. Thus, a language-conditioned policy should ideally be able to open a sliding door even if it is differently positioned or looks visually a bit different.

Evaluation Metrics

All three environment combinations are evaluated with the following metrics:

[1] Simulator states consisting of object positions and orientations are also provided, but not used to better capture challenges of real-world settings.

Task	Condition
Rotate red/blue/pink block right	The object has to be rotated clockwise more than 60° around the z-axis while not being rotated more than 30° around the x or y-axis.
Rotate red/blue/pink block left	The object has to be rotated counterclockwise more than 60° around z while not being rotated more than 30° around the x or y-axis.
Push red/blue/pink block right	The object has to move more than 10 cm to the right while having surface contact in both frames.
Push red/blue/pink block left	The object has to move more than 10 cm to the left while having surface contact in both frames.
Move slider left/right	The sliding door has to be pushed at least 12 cm to the left/right.
Open/close drawer	The drawer has to be pushed in/pulled out at least 10 cm.
Lift red/blue/pink block table	The object has to be grasped from the table surface and lifted at least 5 cm high. In the first frame the gripper may not touch the object.
Lift red/blue/pink block slider	The object has to be grasped from the sliding cabinet's surface and lifted at least 3 cm. In the first frame the gripper may not touch the object.
Lift red/blue/pink block drawer	The object has to be grasped from the drawer's surface and lifted at least 5 cm high. In the first frame the gripper may not touch the object.
Place in slider/drawer	The object has to be placed in the sliding cabinet/drawer. It must be lifted by the gripper in the first frame.
Push into drawer	The object has to be pushed into the drawer. It has to touch the table surface in the first frame.
Stack blocks	A block has to be placed on top of another block. It may not be in contact with the gripper in the final frame.
Unstack blocks	A block has to be removed from the top of another block. It may not be in contact with the gripper in the first frame.
Turn on/off light bulb	The switch has to be pushed up/down to turn on/off the yellow light bulb.
Turn on/off LED	The button has to be pressed to turn on/turn off the green LED light.

Figure 5.6: List of all 34 tasks with their respective success criteria.

Multi-Task Language Control (MTLC): The simplest evaluation aims to verify how well the learned multi-task language-conditioned policy generalizes to 34 manipulation tasks, which we visualize in Fig. 5.6. The evaluation begins by resetting the simulator to the first state of a valid unseen demonstration, to ensure that the commanded instruction is valid. For each manipulation task 10 rollouts are performed with their corresponding different starting states. The language instructions used for testing are not included in the training set and represent novel ways of describing the manipulation tasks seen during training.

Long-Horizon MTLC (LH-MTLC): This evaluation aims to verify how well the learned multi-task language-conditioned policy can accomplish several language instructions in a row. This setting is very challenging as it requires agents to be able to transition between different subgoals. We treat the 34 tasks of the previous evaluation as subgoals and compute valid sequences consisting of five sequential tasks. We only allow feasible sequences that can be achieved from a predefined initial environment state. We filter the evaluation sequences for cycles, redundancies and similarities to arrive at 1000 unique instruction chains. Examples for excluded sequences are "close the drawer"…"place in drawer" (unfeasible), "move slider

right"... "move slider left"... "move slider right" (cyclic) or "push blue block left"... "push red block left"(similar). We reset the robot to a neutral position after every sequence to avoid biasing the policies through the robot's initial pose. We note that this neutral initialization breaks correlation between initial state and task, forcing the agent to rely entirely on language to infer and solve the task. We include different initial scene configurations in the evaluation to better evaluate generalization capabilities. We visualize the evaluated subtask distribution in Figure 5.7. For each subtask we condition the policy on the current language instruction and transition to the next subgoal only if the agent successfully completes the current task according to the environments state indicator.

Sensor Combinations

The aim of CALVIN is to develop innovative agents that learn to relate human language from onboard sensors by capturing many challenges present in real-world settings. Most autonomous robots operating in complex environments are equipped with different sensors to perceive their surroundings. To foster development and experimentation of language-conditioned policies that perform manipulation tasks in the real-world, CALVIN supports a range of sensors commonly utilized for visuomotor control: RGB-D images from both a static and a gripper camera, proprioceptive information, and vision-based tactile sensing [144]. We therefore evaluate baseline agents for different sensors combinations.

5.4 Baseline Models

An agent trained for CALVIN needs to jointly reason over perceptual and language input and produce a sequence of low-level motor commands to interact with the environment.

5.4.1 Multicontext Imitation Learning

We model the interactive agent with a general-purpose goal-reaching policy based on multi-context imitation learning (MCIL) from play data [145]. To learn from unstructured "play" we assume access to an unsegmented teleoperated play dataset $\mathcal{D}$ of semantically meaningful behaviors provided by users, without a set of predefined tasks in mind. To learn control, this long temporal state-action stream $\mathcal{D} = \{(x_t, a_t)\}_{t=0}^{\infty}$ is relabeled [57], treating each visited state in the dataset as a "reached goal state", with the preceding states and actions treated as optimal behavior for reaching that goal. Relabeling yields a dataset of $D_{\text{play}} = \{(\tau, x_g)_i\}_{i=0}^{D_{\text{play}}}$ where each goal state x_g has a trajectory demonstration $\tau = \{(x_0, a_0), \ldots\}$ solving for the goal. These short horizon goal image conditioned demonstrations can be fed to a simple maximum likelihood goal conditioned imitation objective:

$$\mathcal{L}_{LfP} = \mathbb{E}_{(\tau, x_g) \sim D_{\text{play}}} \left[\sum_{t=0}^{|\tau|} \log \pi_\theta(a_t \mid x_t, x_g) \right] \tag{5.1}$$

to learn a goal-reaching policy $\pi_\theta\left(a_t \mid x_t, x_g\right)$. Multi-context imitation learning addresses the inherent multi-modality in free-form imitation datasets by auto-encoding contextual demonstrations through a latent "plan" space with an sequence-to-sequence conditional

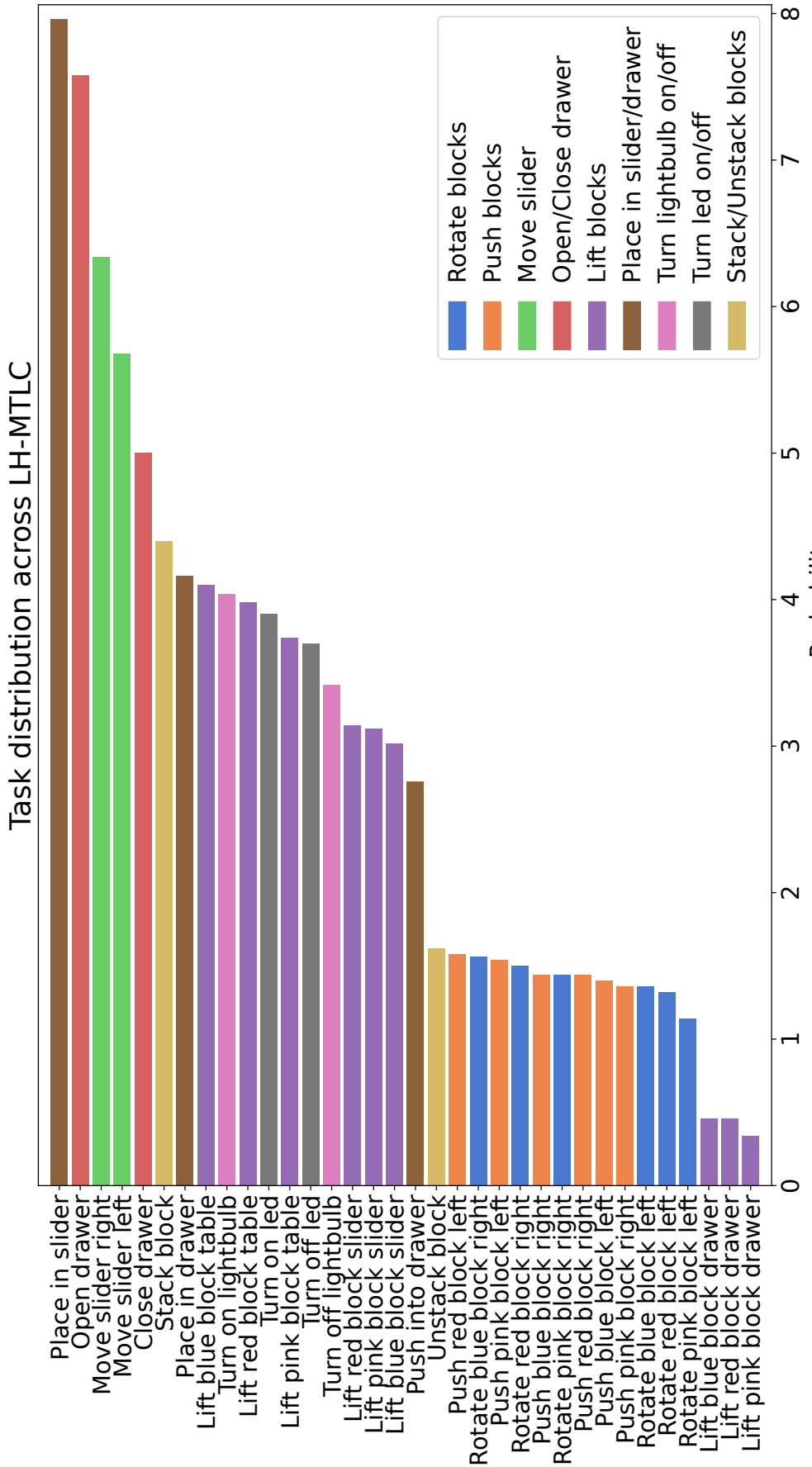


Figure 5.7: Visualization of the subtask distribution across the 1000 instruction chains used for the Long Horizon MTLC evaluation. We show the percentage in which each subtask appears in the distribution.

variational auto-encoder (seq2seq CVAE). The decoder is a policy trained to reconstruct input actions, conditioned on state x_t, goal x_g, and an inferred plan z for how to get from x_t to x_g. At test time, it takes a goal as input, and infers and follows plan z in closed-loop.

However, when learning language-conditioned policies $\pi_\theta\left(a_t \mid x_t, l\right)$ it is not possible to relabel any visited state x to a natural language goal as the goal space is no longer equivalent to the observation space. Lynch *et al.* [145] showed that pairing a small number of random windows with language after-the-fact instructions enables learning a single language-conditioned visuomotor policy that can perform a wide variety of robotic manipulation tasks. The key insight here is that solving a single imitation learning policy for either goal image or language goals, allows for learning control mostly from unlabeled play data and reduces the burden of language annotation to less than 1% of the total data. Concretely, given multiple contextual imitation datasets $\mathcal{D} = \{D^0, D^1, \ldots, D^K\}$, with a different way of describing tasks, MCIL trains a single latent goal conditioned policy $\pi_\theta\left(a_t \mid x_t, z\right)$ over all datasets simultaneously, as well as one parameterized encoder per dataset.

5.4.2 Implementation Details

We follow the baseline architecture implementation reported by Lynch *et al.* [145] unless stated otherwise. We train the agent with the Adam optimizer and a learning rate of 10^{-4}. We set the weight controlling the influence of the KL divergence to the total loss to $\beta = 0.001$. During training, we randomly sample windows between length 16 and 32 and pad them until the max length of 32. As in the original implementation, no image data augmentations are applied and absolute cartesian actions w.r.t the world frame are used. The encoder for the gripper camera takes an image of 84×84 as input and consists of 3 convolutional layers with 32, 64, and 64 channels followed by a 128 unit ReLU MLP. The encoder for the visual-tactile sensor is based on a pre-trained ResNet-18 model. The feature vectors produced by the different modality encoders are concatenated. Depth images are concatenated channel-wise with the RGB images in an early-fusion fashion. In contrast to [145], the gripper fingers of the robot in the CALVIN environment cannot be controlled independently, reducing the action output of the network by one dimension. We note that the same training hyperparameters are used for all splits.

5.5 Experimental Results

The results comparing language-conditioned policies based on multicontext imitation learning for the different evaluation modes in CALVIN are shown in Figure 5.8. We note that there is no constraint to use imitation learning approaches to solve CALVIN tasks, as approaches that use reinforcement learning to learn language-conditioned policies can also be applied [146]. We observe that the baseline with images of the static camera achieves a success rate of 53.9% for the MTLC evaluation setting, when training and testing the 34 manipulation tasks on the same environment. The success rate stays comparable when including a gripper camera, depth channels or tactile sensing. We hypothesize that the reason for not seeing larger improvements when adding the gripper camera is that the policy might benefit from using relative actions instead of global actions. A qualitative analysis indicates that the performance depends significantly on the initial position of the robot, suggesting the agent relies on context rather than learning to disentangle initial states and tasks. It is possible this is

Input					Train → Test	MTLC	LH-MTLC				
Static Camera		Gripper Camera		Tactile		(34 tasks)	No. Instructions in a Row (1000 chains)				
RGB	D	RGB	D	RGB			1	2	3	4	5
✓	✗	✗	✗	✗	D → D	53.9%	48.9%	12.9%	2.6%	0.5%	0.08%
✓	✗	✗	✗	✗	A,B,C,D → D	35.6%	28.2%	2.5%	0.3%	0%	0%
✓	✗	✗	✗	✗	A,B,C → D	38.6%	20.2%	0.2%	0%	0%	0%
✓	✗	✓	✗	✗	D → D	51.8%	34.4%	5.8%	1.1%	0.2%	0.08%
✓	✗	✓	✗	✗	A,B,C,D → D	49.7%	37.3%	2.7%	0.17%	0%	0%
✓	✗	✓	✗	✗	A,B,C → D	38.0%	30.4%	1.3%	0.17%	0%	0%
✓	✗	✗	✗	✓	D → D	54.2%	28.5%	3.2%	0%	0%	0%
✓	✗	✗	✗	✓	A,B,C,D → D	47.9%	22.7%	2.3%	0.3%	0%	0%
✓	✗	✗	✗	✓	A,B,C → D	43.7%	17.3%	0.8%	0.08%	0%	0%
✓	✓	✓	✓	✗	D → D	46.1%	28.2%	4.6%	0.3%	0.08%	0%
✓	✓	✓	✓	✗	A,B,C,D → D	40.7%	14.4%	1.8%	0.08%	0.08%	0%
✓	✓	✓	✓	✗	A,B,C → D	30.8%	21.1%	1.3%	0%	0%	0%

Figure 5.8: Baseline performance of MCIL [145] on the CALVIN Challenge for different combinations of training and test environments and sensor suites.

due to causal confusion between the proprioceptive information and the target actions [155]. Besides, we did not use image data augmentations in the baselines to stay close to the original implementation, but we hypothesize this might be beneficial. Additionally, more elaborate sensor fusion approaches such as mixture of experts [156, 157] or view-invariant contrastive learning [133, 158] might be necessary to learn better multimodal state representations.

For the Long-Horizon MTLC evaluation we observe that the agents perform poorly on CALVIN's long-horizon tasks with high-dimensional state spaces. The best MCIL model achieves a success rate of 0.08% when following chains of five language instructions in a row when training and testing on the same environment. Additionally, it solves the first subtask of the chain, starting from a neutral position, in 48.9% of the cases. We observe that the policy sometimes correctly executes block manipulation tasks, but confuses the red and blue block colors in the instruction. As the language models embed sentences containing the words red and blue similarly, backpropagating through the entire language model and leveraging auxiliary losses that try to align visual and language representations [7] might be beneficial to tackle the complicated perceptual grounding problem.

Finally, the general performance drops significantly when evaluating on the multi environment and zero-shot multi environment settings, which do not follow the standard assumption of imitation learning that training and test tasks are drawn independently from the same distribution. In order to achieve better zero-shot generalization capabilities, additional techniques from the domain adaptation literature [133], better data augmentation and a stronger focus on depth inputs, since they are invariant to texture changes, might be helpful. As MCIL is an offline learning method, we hypothesize that naïve data sharing between multiple domains can be brittle because it can exacerbate the distribution shift between the policy represented in the data and the policy being learned [15]. This motivates further research into agents that can perform the complex long-horizon language-conditioned manipulation tasks introduced by CALVIN.

5.6 Conclusion

In this paper, we presented CALVIN, the first public benchmark of instruction following that combines natural language conditioning, multimodal high-dimensional inputs, 7-DOF continuous control, and long-horizon robotic object manipulation in both seen and unseen environments. As the field of language-driven robotics evolves, a need arises to standardize research for better benchmarks and more reproducible results. CALVIN has the goal of providing researchers with a modular framework that has been developed from the ground up to support training, prototyping, and validation of language-conditioned continuous control policies. Further to that, we hope, along with the help of the community, to continuously expand the tasks available for both training and evaluation.

We use CALVIN to evaluate a conditional sequence-to-sequence variational autoencoder, shown to be effective in other long horizon language-conditioned manipulation tasks [145]. While this model is relatively competent at accomplishing some subgoals, the overall success rates are poor. The long horizon of CALVIN tasks poses a significant challenge with sub-problems including the acquisition of a diverse repertoire of general-purpose skills, object detection, referring expression and action grounding, and task-agnostic continuous control. We hope CALVIN will open the door for the future development of agents that can relate human language to their perception and actions and generalize abstract concepts to unseen

entities in the same way humans do.

Appendix

A Tasks

All tasks are defined in terms of change in the environment state between the first and the final frame of a sequence. In order to see if a task was solved in an arbitrary sequence of frames of the CALVIN dataset, the environment is reset to the state of the first and the last frame of that sequence. The tasks detector compares the two simulator states and checks which task conditions are fulfilled. A key advantage of this strategy is that it enables efficient evaluation of sequences for task completion independent of their length. Figure 5.9 shows a list of all task definitions.

B Language Annotation Generation

The language annotations are extracted automatically from the recorded data with the following procedure: we randomly sample sequences with a window size of 64 frames. For each sequence the task detector checks if a task has been solved between the first and the last frame. Additionally, we check that neither that task nor any other task is solved in the first half of the sequence. The intuition behind this is that we want to include the locomotion behavior prior to the actual task. For example, before opening the drawer, the arm must navigate in the direction of the handle. This is important for learning to solve tasks with language goals from arbitrary starting positions. If a sequence qualifies for labeling, we sample a natural language instruction from a set of predefined sentences with approximately 11 synonymous instructions per task. In total, this gives 389 unique language instructions for 34 tasks. The sequence in which the task "stack blocks" is solved could for example get instructions such as "place the grasped block on top of another block" or "stack blocks on top of each other". In order to simulate a real-world scenario where it might not be possible to pair all the collected robot experience with crowd-sourced language annotations, we annotate only 1% of the recorded robot interaction data with language instructions. The CALVIN dataset conveniently includes precomputed MiniLM language embeddings for all instructions, but researchers are free to use their own language model of choice on the raw input data.

Task	**Condition**
Rotate red block right Rotate blue block right Rotate pink block right	The object has to be rotated clockwise more than 60° around the z-axis while not being rotated for more than 30° around the x or y-axis.
Rotate red block left Rotate blue block left Rotate pink block left	The object has to be rotated counterclockwise more than 60° around the z-axis while not being rotated for more than 30° around the x or y-axis.
Push red block right Push blue block right Push pink block right	The object has to move more than 10 cm to the right while having surface contact in both frames
Push red block left Push blue block left Push pink block left	The object has to move more than 10 cm to the left while having surface contact in both frames
Move slider left	The sliding door has to be pushed at least 12 cm to the left.
Move slider right	The sliding door has to be pushed at least 12 cm to the right.
Open drawer	The drawer has to pulled out at least 10 cm.
Close drawer	The drawer has to be pushed in at least 10 cm.
Lift red block table Lift blue block table Lift pink block table	The object has to be grasped from the table surface and lifted at least 5 cm high. In the first frame the gripper may not touch the object.
Lift red block slider Lift blue block slider Lift pink block slider	The object has to be grasped from the surface of the sliding cabinet and lifted at least 3 cm high. In the first frame the gripper may not touch the object.
Lift red block drawer Lift blue block drawer Lift pink block drawer	The object has to be grasped from the drawer surface and lifted at least 5 cm high. In the first frame the gripper may not touch the object.
Place in slider	The object has to be placed in the sliding cabinet. It must be lifted by the gripper in the first frame.
Place in drawer	The object has to be placed in the drawer. It must be lifted by the gripper in the first frame.
Push into drawer	The object has to be pushed into the drawer. It has to touch the table surface in the first frame.
Stack blocks	A block has to be placed on top of another block. It may not be in contact with the gripper in the final frame.
Unstack blocks	A block that is stacked on another block has to be removed from the top, either by grasping it or by pushing it down. It may not be in contact with the gripper in the first frame.
Turn on light bulb	The switch has to be pushed down to turn on the yellow light bulb.
Turn off light bulb	The switch has to be pushed up to turn off the yellow light bulb.
Turn on LED	The button has to be pressed to turn on the green LED light.
Turn off LED	The button has to be pressed to turn off the green LED light.

Figure 5.9: List of all 34 tasks with their respective success criteria.

Chapter 6

What Matters in Language Conditioned Robotic Imitation Learning over Unstructured Data

The content of this chapter has been published in [159]:

O. Mees, L. Hermann and W. Burgard
What Matters in Language Conditioned Robotic Imitation Learning over Unstructured Data
IEEE Robotics and Automation Letters (RAL), 2022
DOI: 10.1109/LRA.2022.3196123

Me and Lukas Hermann share the main authorship as we jointly developed the HULC method. I developed the main idea of the paper, contributed the idea of using a multimodal transformer encoder, the KL divergence balancing, the self-supervised contrastive visual-language alignment loss and using discrete categorical latent plans. Lukas Hermann contributed the idea of using relative actions in the gripper frame and improvements to the model's training time. All the results for HULC reported in this thesis were entirely carried out by the author of this thesis. The paper was mostly written by the author of this thesis.

Abstract

A long-standing goal in robotics is to build robots that can perform a wide range of daily tasks from perceptions obtained with their onboard sensors and specified only via natural language. While recently substantial advances have been achieved in language-driven robotics by leveraging end-to-end learning from pixels, there is no clear and well-understood process for making various design choices due to the underlying variation in setups. In this paper, we conduct an extensive study of the most critical challenges in learning language conditioned policies from offline free-form imitation datasets. We further identify architectural and algorithmic techniques that improve performance, such as a hierarchical decomposition of the robot control learning, a multimodal transformer encoder, discrete latent plans and a self-supervised contrastive loss that aligns video and language representations. By combining the results of our investigation with our improved model components, we are able to present a novel approach that significantly outperforms the state of the art on the challenging language conditioned long-horizon robot manipulation CALVIN benchmark. We have open-sourced our implementation to facilitate future research in learning to perform many complex manipulation skills in a row specified with natural language. Codebase and trained models available at `http://hulc.cs.uni-freiburg.de`

6.1 Introduction

One of the grand challenges in robotics is to create a generalist robot: a single agent capable of performing a wide variety of tasks in everyday settings based on arbitrary user commands. Doing so requires the robot to acquire a diverse repertoire of general-purpose skills and non-expert users to be able to effectively specify tasks for the robot to solve. This stands in contrast to most current end-to-end models, which typically learn individual tasks one at a time from manually-specified rewards and assume tasks being specified via goal images [18] or one-hot skill selectors [19], which are not practical for untrained users to instruct robots. Not only is this inefficient, but also limits the versatility and adaptivity of the systems that can be built. How can we design learning systems that can efficiently acquire a diverse repertoire of useful skills that allows them to solve many different tasks based on arbitrary user commands?

To address this problem, we must resolve two questions. (1) How can untrained users direct the robot to perform specific tasks? Natural language presents a promising alternative form of specification, providing an intuitive and flexible way for humans to communicate tasks and refer to abstract concepts. However, learning to follow language instructions involves addressing a difficult symbol grounding problem [25], relating a language instruction to a robot's onboard perception and actions. (2) How can the robot efficiently learn general-purpose skills from offline data, without hand-specified rewards? A simple and versatile choice is to define skills as being continuous instead of discrete, endowing the agent of task-agnostic control: the ability to reach any reachable goal state from any current state [52]. These forms of task specification can in principle enable a robot to solve multi-stage tasks by following several language instructions in a row.

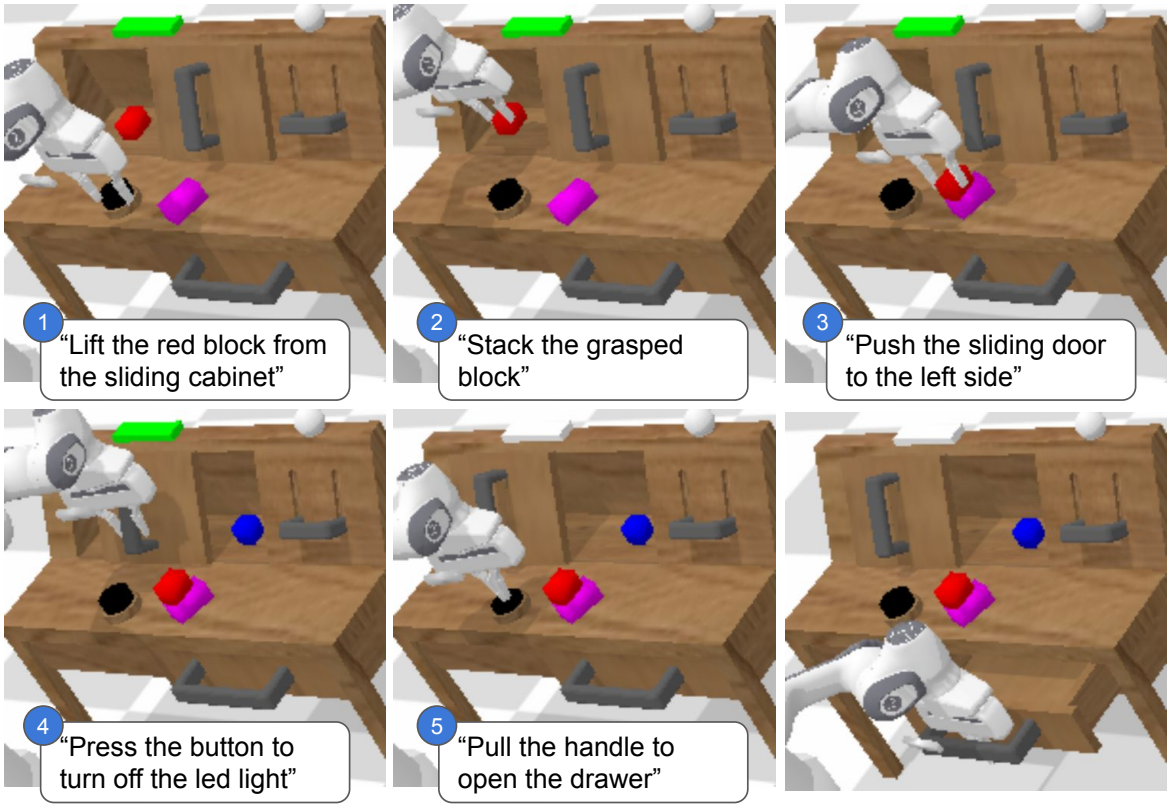


Figure 6.1: CALVIN learns a single 7-DoF language conditioned visuomotor policy from offline, unstructured data that can solve multi-stage, long-horizon robot manipulation tasks. We divide instruction following into learning global plans representing high-level behavior and a local policy conditioned on the plan and the instruction.

Recent advances have been made at learning language conditioned policies for continuous visuomotor-control in 3D environments via imitation learning [21, 54, 145, 160] or reinforcement learning [140, 146]. These approaches typically require offline data sources of robotic interaction together with post-hoc crowd-sourced natural language labels. Although all methods share the basic idea of leveraging instructions that are grounded in the agent's high-dimensional observation space, their details vary greatly. Moreover, evaluating published methods and their components in language conditioned policy learning is difficult due to incomparable setups or subjective task definitions. In this work we systematically compare, improve, and integrate key components by leveraging the recently proposed CALVIN benchmark [161] to further our understanding and provide a unified framework for long-horizon language conditioned policy learning. We build upon relabeled imitation learning [57] to distill many reusable behaviors into a goal-directed policy, as seen in Fig. 6.1. Our approach consists of only standard supervised learning subroutines, and learns perceptual and linguistic understanding, together with task-agnostic control end-to-end as a single neural network. Our contributions are:

- We systematically compare key components of language conditioned imitation learning over unstructured data, such as observation and action spaces, losses for aligning visuolingual representations, language models and latent plan representations, and we analyze the effect of other choices, such as data augmentation and optimization.

- We propose four improvements to these key components: a multimodal transformer encoder to learn to recognize and organize behaviors during robotic interaction into a global categorical latent plan, a hierarchical division of the robot control learning that learns local policies in the gripper camera frame conditioned on the global plan, balancing terms within the KL loss and a self-supervised contrastive visual-language alignment loss.

- We integrate the best performing improved components in a unified framework, Composing Actions from Language and Vision (CALVIN). Our model sets a new state of the art on the challenging CALVIN benchmark [161], on learning a single 7-DoF policy that can perform long-horizon manipulation tasks in a 3D environment, directly from images, and only specified with natural language.

6.2 Related Work

Natural language processing has recently received much attention in the field of robotics [24], following the advances made towards learning groundings between vision and language [7, 147] and grounding behaviors in language [162]. Early works have approached instruction following by designing interactive fetching systems to localize objects mentioned in referring expressions [104, 109] or by grounding not only objects, but also spatial relations to follow language expressions characterizing pick-and-place commands [30, 120, 151]. Unlike these approaches, we directly learn robotic control from images and natural language instructions, and do not assume any predefined motion primitives.

More recently, end-to-end deep learning has been used to condition agents on natural language instructions [21, 54, 140, 145, 146, 160], which are then trained under an imitation or reinforcement learning objective. These works have pushed the state of the art and generated a range of ideas for language conditioned policy learning, such as losses for aligning visual observations and language instructions. However, each work evaluates a different combination of ideas and uses different setups or task definitions, making it unclear how individual ideas compare to each other and which ideas combine well together. For example, the methods BC-Z and MIA [21, 160] use both behavior cloning, but different actions spaces and multi-modal alignment losses, such as regressing the language embedding from visual observations [21] or cross-modality matching [160]. Moreover, BC-Z leverages expert trajectories and task labels, and MIA includes mobile navigation, making them difficult to implement directly in CALVIN, which contains unlabeled play data on different tabletop environments. Nair *et. al.* [146] learn a reward classifier which predicts if a change in state completes a language instruction and leverage it for offline multi-task RL given four camera views. Similar to BC-Z they rely on discrete task labels and do not focus on solving long-horizon language-specified tasks. Most related to our approach is multi-context imitiation learning (MCIL) [145], which also uses relabeled imitation learning to distill reusable behaviors into a goal-reaching policy. Besides different action and observation spaces, these works leverage different language models to encode the raw text instructions into a semantic pre-trained vector space, making it difficult to analyze which language models are best suited for language conditioned policy learning. The ablation studies presented in these papers show that each novel contribution of each work does indeed improve the performance of their model, but due to incomparable setups and evaluation protocols, it is difficult to asses

what matters for language conditioned policy learning. Our work addresses this problem by systematically comparing and combining different observation and action spaces, auxiliary losses and latent representations and integrating the best performing components in a unified framework.

6.3 Problem Formulation and Method Overview

We consider the problem of learning a goal-conditioned policy $\pi_\theta\left(a_t \mid s_t, l\right)$ that outputs action $a_t \in \mathcal{A}$, conditioned on the current state $s_t \in \mathcal{S}$ and free-form language instruction $l \in \mathcal{L}$, under environment dynamics $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$. We note that the agent does not have access to the true state of the environment, but to visual observations. In CALVIN [161] the action space $\mathcal{A}$ consists of the 7-DoF control of a Franka Emika Panda robot arm with a parallel gripper.

We model the interactive agent with a general-purpose goal-reaching policy based on multi-context imitation learning (MCIL) from play data [145]. To learn from unstructured "play" we assume access to an unsegmented teleoperated play dataset $\mathcal{D}$ of semantically meaningful behaviors provided by users, without a set of predefined tasks in mind. To learn control, this long temporal state-action stream $\mathcal{D} = \{(s_t, a_t)\}_{t=0}^{\infty}$ is relabeled [57], treating each visited state in the dataset as a "reached goal state", with the preceding states and actions treated as optimal behavior for reaching that goal. Relabeling yields a dataset of $D_{\text{play}} = \{(\tau, s_g)_i\}_{i=0}^{D_{\text{play}}}$ where each goal state s_g has a trajectory demonstration $\tau = \{(s_0, a_0), \ldots\}$ solving for the goal. These short horizon goal image conditioned demonstrations can be fed to a simple maximum likelihood goal conditioned imitation objective:

$$\mathcal{L}_{LfP} = \mathbb{E}_{(\tau, s_g) \sim D_{\text{play}}} \left[\sum_{t=0}^{|\tau|} \log \pi_\theta(a_t \mid s_t, s_g) \right] \tag{6.1}$$

to learn a goal-reaching policy $\pi_\theta\left(a_t \mid s_t, s_g\right)$. We address the inherent multi-modality in free-form imitation datasets by auto-encoding contextual demonstrations through a latent "plan" space with an sequence-to-sequence conditional variational auto-encoder (seq2seq CVAE) [18]. Conditioning the policy on the latent plan frees up the policy to use the entirety of its capacity for learning uni-modal behavior. To generate latent plans z we make use of the variational inference framework [58]. The objective of the latent plan sampler is to model the full distribution over all high-level behaviors that might connect the current and goal state, to provide multi-modal plans at inference time. This distribution is learned with a CVAE by maximizing the marginal log likelihood of the observed behaviors in the dataset $\log p(x \mid s)$, where x are sampled state-action trajectories from τ. The Evidence Lower Bound (ELBO) [58] for the CVAE can be written as:

$$\log p(x|s) \geq -\text{KL}(q(z|x, s) \mid\mid p(z|s)) + \mathbb{E}_{q(z|x,s)} \left[\log p(x|z, s)\right] \tag{6.2}$$

The decoder is a policy trained to reconstruct input actions, conditioned on state s_t, goal s_g, and an inferred plan z for how to get from s_t to s_g. At test time, it takes a goal as input, and infers and follows plan z in closed-loop.

However, when learning language conditioned policies $\pi_\theta\left(a_t \mid s_t, l\right)$ it is not possible to relabel any visited state s to a natural language goal as the goal space is no longer

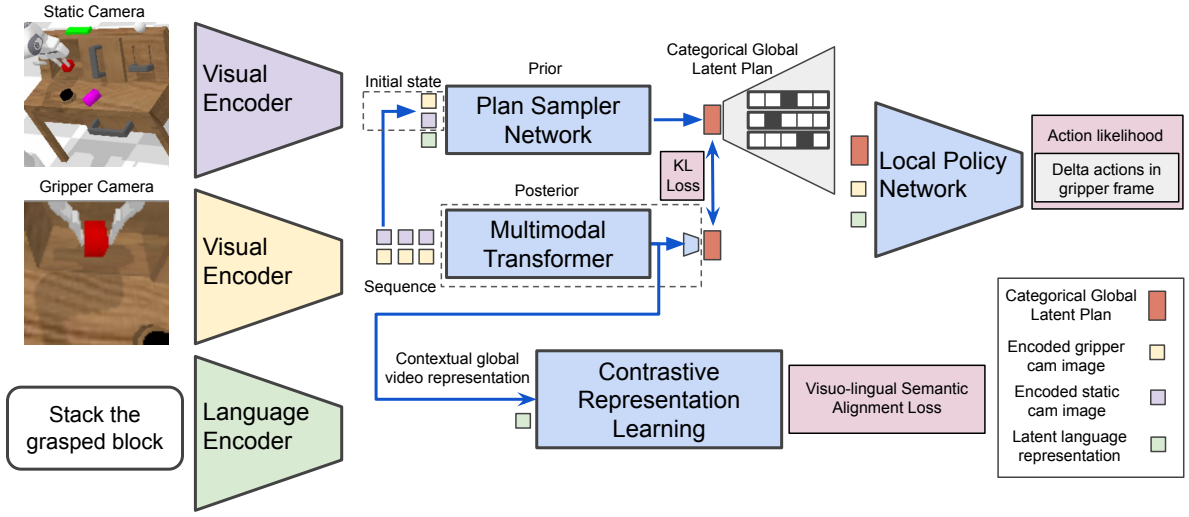


Figure 6.2: Overview of our architecture to learn language conditioned policies from unstructured data. First the language instructions and the visual observations are encoded. During training a multimodal transformer encodes sequences of observations to learn to recognize and organize high-level behaviors through a posterior. Its temporally contextualized features are provided as input to a contrastive visuo-lingual alignment loss. The plan sampler network receives the initial state and the latent language goal and predicts the distribution over plans for achieving the goal. Both prior and posterior distributions are predicted as a vector of multiple categorical variables and are trained by minimizing their KL divergence. The local policy network receives the latent language instruction, the gripper camera observation and the global latent plan to generate a sequence of relative actions in the gripper camera frame to achieve the goal.

equivalent to the observation space. Lynch *et al.* [145] showed that pairing a small number of random windows with language after-the-fact instructions enables learning a single language conditioned visuomotor policy that can perform a wide variety of robotic manipulation tasks. The key insight here is that solving a single imitation learning policy for either goal image or language goals, allows for learning control mostly from unlabeled play data and reduces the burden of language annotation to less than 1% of the total data. Concretely, given multiple contextual imitation datasets $\mathcal{D} = \{D^0, D^1, \ldots, D^K\}$, with a different way of describing tasks, MCIL trains a single latent goal conditioned policy $\pi_\theta\left(a_t \mid s_t, z\right)$ over all datasets simultaneously, as well as one parameterized encoder per dataset.

6.4 Key Components of Language Conditioned Imitation Learning over Unstructured Data

This section compares and improves key components of language conditioned imitation learning over unstructured data. We base our model on MCIL [145] and improve it by decomposing control into a hierarchical approach of generating global plans with a static camera and learning local policies with a gripper camera conditioned on the plan. Then we go through different components that have a large impact on performance: architectures to encode sequences in relabeled imitation learning, the representation of the latent distributions,

how to best align language and visual representations, data augmentation and optimization. We visualize the full architecture in Fig. 6.2.

A Observation and Actions Spaces

How to best represent motion skills is an age-old question in robotics. From a learning perspective, generating the action sequences to solve diverse manipulation tasks with a single network from high-dimensional observations is challenging, because the distribution is multi-modal, discontinuous and imbalanced. For these reasons, finding an efficient representation is crucial to perform this non-trivial reasoning using learning-based methods. MCIL [145] uses global actions learned from a single static RGB camera. We observe that predicting 7-DoF global actions leads to the network primarily solving static element tasks, such as pushing a button, but failing to generalize to dynamic tasks, such as manipulating colored blocks. To alleviate this problem, we propose generating global plans that correspond to reusable common behavior b seen in the play data, but learning local policies conditioned on the plan. This results in a hierarchical approach that frees up the network from having to memorize all locations in the scene were the behaviors were performed. Concretely, we encode RGB images from both the static and a gripper camera to learn a compact representation of all the different high-level plans that take an agent from a current state to a goal state, learning $p\left(b \mid s_t, s_g\right)$. Inspired by a recent line of work that aims to learn hierarchies of controllers based on static and gripper cameras [163], we use the encoded gripper camera representations in the policy network, the global contextualized latent plan, and perform control in the gripper frame with relative actions for an efficient robot control learning. The action space consists of delta XYZ position, delta Euler angles and the gripper action. Our proposed formulation has several advantages: a) local policies based on the gripper camera generalize better to different locations of the objects to be manipulated b) the policy has a prior in the form of a global contextualized latent plan, but is free to discover the exact strategy on how to interact with the objects.

B Latent Plan Encoding

A challenge in self-supervising control on top of free-form imitation data is that in general, there are many valid high-level behaviors that might connect the same (s_t, s_g) pairs. By auto-encoding contextual demonstrations through a latent "plan" space with an sequence-to-sequence conditional variational auto-encoder (seq2seq CVAE) [18], we can learn to recognize which region of the latent plan space an observation-action sequence belongs to. Critically, conditioning the policy on the latent plan frees up the policy to use the entirety of its capacity for learning uni-modal behavior. Thus, learning to generate and represent high-quality latent plans is a key component in the seq2seq CVAE framework. MCIL [145] uses bidirectional recurrent neural networks (RNN) to encode a randomly sampled play sequence and map it into a latent Gaussian distribution. In contrast, we leverage a multimodal transformer encoder [85] to build a contextualized representation of abstract behavior expressed in language instructions and map into a vector of several latent categorical variables [164]. The foundation of the Transformer architecture is the scaled dot-product attention function, which enables elements in a sequence to attend to other elements. The attention function receives as input a sequence $\{x_1, ..., x_n\}$ and outputs a sequence $\{y_1, ..., y_n\}$. Each input x_i is

projected linearly to a query q_i, key k_i, and value v_i. To compute the output y_i the values are summed with weights that take into account the similarity of the query with its corresponding key. The attention function is defined as $\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$, where d_k is the dimension of the keys and queries. The queries, keys, and values are stacked together into matrix $Q \in \mathbb{R}^{n \times d_{\text{model}}}$, $K \in \mathbb{R}^{n \times d_{\text{model}}}$, and $V \in \mathbb{R}^{n \times d_{\text{model}}}$. We encode the sequence of visual observations of both modalities $X_{\{static,gripper\}} \in \mathbb{R}^{T \times H \times W \times 3}$ with separate perceptual encoders, and concatenate them to form the fused perceptual representation $V \in \mathbb{R}^{T \times d}$ of the sampled demonstration, where T represents the sequence length and d the feature dimension. To enable the sequences to carry temporal information, we add positional embeddings [85] and feed the result into the Multimodal Transformer to learn temporally contextualized global video representations. Finally, inspired by the recent line of work that looks into learning discrete instead of continuous latent codes [164, 165], we represent the latent plans as a vector of multiple categorical latent variables and and optimize them using straight-through gradients [166]. Learning discrete representations in the context of language conditioned policies is a natural fit, as language is inherently discrete and images can often be described concisely by language [120]. Furthermore, discrete representations are a natural fit for complex reasoning, planning and predictive learning (e.g., if it is sunny, I will go to the beach).

C Semantic Alignment of Video and Language

Learning to follow language instructions involves addressing a difficult symbol grounding problem [25], relating a language instruction to a robots onboard perception and actions. Although instructions and visual observations are aligned in CALVIN, learning to manipulate the colored blocks is a challenging problem. This is due to the fact that the robot needs to learn a wide variety of diverse behaviors to manipulate the blocks, but also needs to understand which colored block the user is referring to. Thus, the block related instructions are very similar, for the exception of a word that might disambiguate the instruction by indicating a color. Therefore, most pre-trained language models struggle to learn such semantics from text only and the policy needs to learn referring expression comprehension via the imitation loss. There have been a number of multi-modal alignment losses proposed, such as regressing the language embedding from the visual observation [21] or cross-modality matching [160]. We maximize the cosine similarity between the visual features of the sequence i and the corresponding language features while, at the same time, minimizing the cosine similarity between the current visual features and other language instructions in the same batch. We define our $\mathcal{L}_{contrast}$ loss the same way as the contrastive loss for pairing images and captions in CLIP [7]. However, ideally our model should use the time-dependent representation of the sequence visual observations in order to capture the meaning of a language instruction. This can be appreciated only after the sequence of actions have been executed for several timesteps. The usage of in-batch negatives enables re-use of computation both in the forward and the backward pass making training highly efficient. The logits for one batch is a $M \times M$ matrix, where each entry is given by $\text{logit}(x_i, y_j) = \text{cos\_sim}(x_i, y_j) \cdot \exp(\tau), \forall (i, j), i, j \in \{1, 2, \ldots, M\}$ where τ is a trainable temperature parameter. Only entries on the diagonal of the matrix are considered positive examples. The final loss is the sum of the cross entropy losses on the row and the column direction.

D Action Decoder

A challenge in learning control from free-form imitation data, in which different ways of executing the same skill are shown, is that a standard unimodal predictor, such as a Gaussian distribution, will average out dissimilar motions. To address this multimodality, we follow the solution proposed by Lynch *et. al.* [18] of discretizing the action space and then parameterizing the policy as a discretized logistic mixture distribution [167, 168]. Each of the predicted k logistic distributions have a separate mean and scale, and are weighed with α to form the mixture distribution. The imitation loss is the negative log-likelihood for this distribution:

$$\mathcal{L}_{act}(\mathcal{D}_{play}, V) = -\ln(\Sigma_{i=0}^{k}\, \alpha_k(V_t)\; P(a_t, \mu_i(V_t), \sigma_i(V_t))$$

Where, $P(a_t, \mu_i(V_t), \sigma_i(V_t)) = F(\frac{a_t+0.5-\mu_i(V_t)}{\sigma_i(V_t)}) - F(\frac{a_t-0.5-\mu_i(V_t)}{\sigma_i(V_t)})$ and $F(\cdot)$ is the logistic CDF. Additionally, we use a cross-entropy loss to model the binary gripper open/close action.

E Optimization and Implementation Details

Our full training objective for the 1% of the total data that is annotated with after-the-fact language instructions is given by $\mathcal{L} = \mathcal{L}_{act} + \beta\mathcal{L}_{KL} + \lambda\mathcal{L}_{contrast}$. The windows without annotations are trained with the same imitation learning objective, but the language goals are replaced by the last visual frame of the sampled window to learn control in a fully self-supervised manner. A common problem in training VAEs is finding the right balance in the weight of the KL loss. A high β value can result in an over-regularized model in which the decoder ignores the latent plans from the prior, also known as a "posterior collapse" [169]. On the other hand, setting β too low results in the plan sampler network being unable to catch up to plan over the latent space created by the posterior, and as a result at test time the plans generated by the plan sampler network will be unfamiliar inputs for the decoder. Orthogonal to this, as the KL loss is bidirectional, we want to avoid regularizing the plans generated by the posterior toward a poorly trained prior. To solve this problem, we minimize the KL loss faster with respect to the prior than the posterior by using different learning rates, $\alpha = 0.8$ for the prior and $1 - \alpha$ for the posterior, similar to Hafner *et. al.* [164]. We set $\beta = 0.01$ and $\lambda = 3$ for all experiments and train with the Adam optimizer with a learning rate of 2^{-4}. During training, we randomly sample windows between length 20 and 32 and pad them until the max length of 32. For the latent plan representation we use 32 categoricals with 32 classes each. To better compare the differences between approaches, we use the same convolutional encoders as the MCIL baseline available in CALVIN for processing the images of the static and gripper camera. Our multimodal transformer encoder has 2 blocks, 8 self-attention heads, and a hidden size of 2048. In order to encode raw text into a semantic pre-trained vector space, we leverage the paraphrase-MiniLM-L3-v2 model [87], which distills a large Transformer based language model and is trained on paraphrase language corpora that is mainly derived from Wikipedia. It has a vocabulary size of 30,522 words and maps a sentence of any length into a vector of size 384.

F Data Augmentation

To aid learning we apply data augmentation to image observations, both in our method and across all baselines. During training, we apply stochastic image shifts of 0-4 pixels to the

gripper camera images and of 0-10 pixels to the static camera images as in Yarats *et. al.* [170]. Additionally, a bilinear interpolation is applied on top of the shifted image by replacing each pixel with the average of the nearest pixels.

6.5 Experiments

We evaluate our model in an extensive comparison and ablation study, to determine which components matter for language conditioned imitation learning over unstructured data. We ablate single components of our full approach to study the influence of each component. We then compare our resulting model to the best published methods on the CALVIN benchmark, and show that it outperforms all previous methods.

A Evaluation Protocol

The goal of the agent in CALVIN is to solve sequences of up to 5 language instructions in a row using only onboard sensors. This setting is very challenging as it requires agents to be able to transition between different subgoals. CALVIN has a total of 34 different subtasks and evaluates 1000 unique sequence instruction chains. The robot is set to a neutral position after every sequence to avoid biasing the policies through the robot's initial pose. This neutral initialization breaks correlation between initial state and task, forcing the agent to rely entirely on language to infer and solve the task. For each subtask in a row the policy is conditioned on the current subgoal instruction and transitions to the next subgoal only if the agent successfully completes the current task. We perform the ablation studies on the environment D of CALVIN and additionally report numbers of our approach for the other two CALVIN splits, the multi environment and zero-shot multi environment splits. We emphasize that the CALVIN dataset for each of the four environment consists of 6 hours of teleoperated undirected play data that might contain suboptimal behavior. To simulate a real-world scenario, only 1% of that data contains crow-sourced language annotations.

B Results and Ablations of Key Components

Observation and Actions Spaces: We compare our approach of dividing the robot control learning into generating global contextualized plans and conditioning a local policy that receives only the observations of the the gripper camera on the global plan against a "No Local Policy" baseline. Unlike our approach, which performs control in the gripper camera frame, the baseline's policy receives both cameras images and performs control in the robot's base frame, as is usual in most published approaches. We observe in Fig. 6.3, that despite the baseline's decoder having more perceptual information, the performance for completing 5 chains of language instructions sequentially drops from 28.3% to 20.1%. In order to analyze the big performance difference with respect to the original MCIL baseline, we train a MCIL baseline with relative actions and observe that its performance improves significantly from the original MCIL baseline with absolute actions, but performs worse than our models. We speculate that using relative actions with a local policy is easier for the agent to learn instead of memorizing all the locations where interactions have been performed with global actions and a global observation space. By decoupling the control into a hierarchical structure, we show that performance increases significantly. Additionally, we analyze the influence of using

Method	LH-MTLC					
	No. Instructions in a Row (1000 chains)					
	1	2	3	4	5	Avg. Len.
MCIL [145]	34.4%	5.8%	1.1%	0.2%	0.08%	0.41
GCBC [18] + delta actions	64.7% (4.0)	28.4% (6.2)	12.2% (4.1)	4.9% (2.0)	1.3% (0.9)	1.11 (0.3)
MCIL [145] + delta actions	76.4% (1.5)	48.8% (4.1)	30.1% (4.5)	18.1% (3.0)	9.3% (3.5)	1.82 (0.2)
Ours	**82.7**% (0.3)	**64.9**% (1.7)	**50.4**% (1.5)	**38.5**% (1.9)	**28.3**% (1.8)	**2.64** (0.05)
No Transformer Encoder	79.5% (0.6)	61.5% (2.0)	46.7% (1.7)	32.6% (1.2)	24.7% (1.7)	2.45 (0.06)
No Discrete Latents	79.8% (1.6)	60.6% (0.4)	43.3% (0.9)	32.6% (1.1)	23.6% (1.6)	2.39 (0.03)
No Local Policy	78.4% (1.4)	56.2% (2.0)	40.4% (2.4)	29.5% (1.7)	20.1% (1.3)	2.24 (0.06)
No KL Balancing	79.6% (2.3)	59.3% (0.2)	42.3% (0.7)	30.7% (0.8)	21.9% (1.1)	2.33 (0)
No Gripper Log Loss	79.5% (1.3)	61.3% (1.7)	46.5% (2.1)	33.9% (1.9)	24.0% (1.8)	2.45 (0.07)
KL $\beta = 0.1$	78.4% (1.5)	55.8% (1.4)	36.3% (1.7)	23.9% (1.1)	16.2% (1.2)	2.10 (0.09)
No Lang Align. Loss	80.1% (0.9)	55.4% (2.0)	42.2% (1.3)	30.2% (1.3)	21.8% (0.9)	2.29 (0.07)
MIA Lang Align. Loss [160]	79.5% (1.9)	57.8% (0.8)	41.7% (1.4)	29.7% (2.1)	20.9% (1.6)	2.29 (0.04)
Lang Align. Regression [21]	82.5% (0.8)	61.0% (1.0)	45.5% (0.4)	32.9% (1.1)	23.5% (0.7)	2.45 (0.03)
No image augmentation	75.6% (1.4)	51.2% (1.7)	34.0% (2.6)	23.6% (1.9)	15.4% (0.6)	1.99 (0.07)
Proprioceptive input	81.2% (0.3)	56.1% (1.4)	39.2% (1.6)	26.2% (1.3)	18.1% (0.8)	2.20 (0.05)
MPNet SBERT [87]	77.4% (2.0)	57.2% (1.9)	40.7% (1.4)	28.5% (1.7)	20.2% (1.0)	2.24 (0.06)
MPNet [171]	71.5% (2.6)	48.5% (1.4)	33.6% (1.6)	23.1% (1.7)	15.5% (0.7)	1.92 (0.08)
Distilroberta SBERT [87]	78.1% (1.2)	60.9% (1.5)	47.7% (1.9)	36.4% (1.9)	27.5% (0.7)	2.50 (0.07)
Distilroberta [172]	77.8% (1.4)	56.3% (1.7)	40.6% (2.4)	28.2% (1.7)	17.8% (1.9)	2.21 (0.07)
BERT [86]	77.5% (1.8)	52.6% (3.2)	34.7% (2.4)	24.1% (1.5)	14.9% (2.4)	2.03 (0.1)
CLIP Encoders [7]	81.4% (0.5)	60.4% (1.1)	44.7% (2.3)	32.3% (1.2)	23.2% (1.6)	2.42 (0.06)

Figure 6.3: Performance of our model on the D environment of the CALVIN Challenge and ablation of the key components, across 3 seeded runs. All models receive RGB images from both a static and a gripper camera as a input.

the 7-DoF proprioceptive information as input for both the plan encodings and conditioning the policy, as many works report improved performance from it [18, 19, 145]. We observe that the performance drops significantly and the agent relies too much on the robot's initial position, rather than learning to disentangle initial states and tasks. We hypothesize this might be due to a causal confusion between the proprioceptive information and the target actions [155]. We also analyze the effect of modeling the full action space, including the binary gripper action dimension, with the mixture of logistics distribution instead of using the log loss for the open/close gripper action and observe that the average sequence length drops from 2.64 to 2.45. Finally, we note that applying stochastic image shifts to the input images increases the performance significantly.

Method	Train → Test	LH-MTLC					
		No. Instructions in a Row (1000 chains)					
		1	2	3	4	5	Avg. Len.
MCIL [145]	A,B,C,D → D	37.3%	2.7%	0.17%	0%	0%	0.40
Ours	A,B,C,D → D	**88.9**% (0.6)	**73.3**% (1.7)	**58.7**% (1.8)	**47.5**% (1.6)	**38.3**% (1.9)	**3.06** (0.07)
MCIL [145]	A,B,C → D	30.4%	1.3%	0.17%	0%	0%	0.31
Ours	A,B,C → D	**41.8**% (2.3)	**16.5**% (2.5)	**5.7**% (1.3)	**1.9**% (0.9)	**1.1**% (0.5)	**0.67** (0.1)

Figure 6.4: Performance of our model on the multi environment splits of the CALVIN Challenge across 3 seeded runs.

Latent Plan Encoding: In our CVAE framework the latent plan represents valid ways of connecting the actual state and the goal state and thus, frees up the policy to use the entirety of its capacity for learning uni-modal behavior. As language is inherently discrete and discrete representations are a natural fit for complex reasoning and planning, we represent latent plans as a vector of multiple categorical latent variables and and optimize them using straight-through gradients [166]. We observe that the performance for 5-chain evaluation drops from 28.3% to 23.6% when we train our model with a diagonal Gaussian distribution as in MCIL. While it is difficult to judge why categorical latents work better than continuous latent variables, we hypothesize that categorical latents could be a better inductive bias for non-smooth aspects of the CALVIN benchmark, such as when a block is hidden behind the sliding door. Besides, the sparsity level enforced by a categorical distribution could be beneficial for generalization. Additionally, we compare against a goal-conditioned Behavior Cloning (GCBC) baseline [18] which does not condition the policy on a latent plan, and observe that it performs worse than MCIL with relative actions, highlighting the importance of modeling latent behaviors in free-form imitation datasets. We also observe that balancing the KL loss is beneficial in the CVAE training. By scaling up the prior cross entropy relative to the posterior entropy, the agent is encouraged to minimize the KL loss by improving its prior toward the more informed posterior, as opposed to reducing the KL by increasing the posterior entropy. We visualize a t-SNE plot of our learned discrete latent space in Figure 6.5 and that see that even for unseen language instructions it appears to organize the latent space functionally. Additionally, we report degraded performance for an over-regularized model which learns to ignore the latent plans, in which we weight the KL divergence with $\beta = 0.1$. Finally, we evaluate replacing the transformer encoder in the posterior with a GRU bidirectional recurrent network of the same hidden dimension of 2048, similar to MCIL. The results suggest that besides an improved performance, the multimodal transformer encoder is significantly more efficient both memory and model size wise (5.9 M vs 106 M parameters

for the posterior network) and overall training wall clock time. For comparison, with the transformer encoder, our full approach contains 47.1 M trainable parameters.

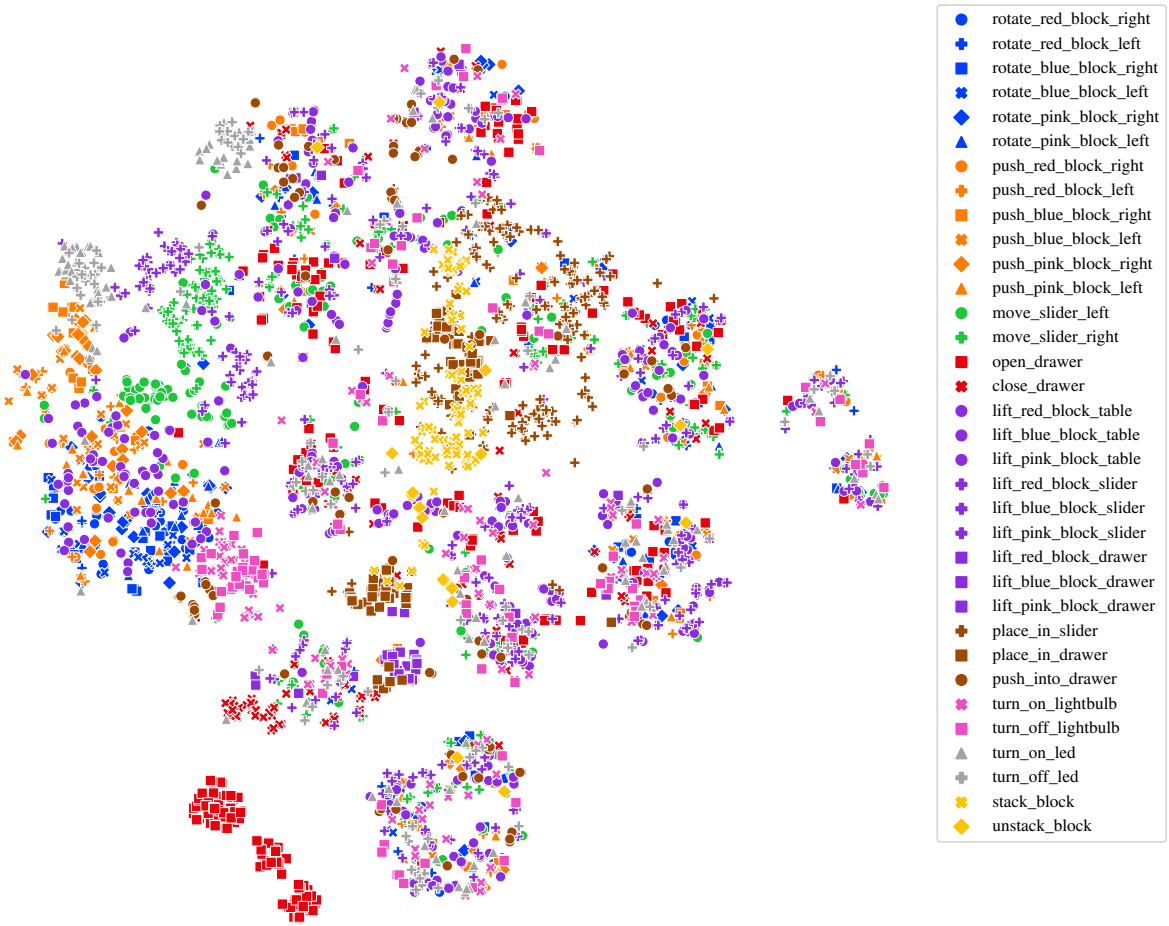


Figure 6.5: t-SNE visualization of the discrete latent plans generated by embedding randomly selected unseen language annotations. Surprisingly, we find that despite not being trained explicitly with task labels, CALVIN appears to organize its latent plan space functionally. We visualize with the same color functionally similar skills, but use different shapes to distinguish sub-skills.

Semantic Alignment of Video and Language: One of the main challenges for language conditioned continuous visuomotor-control is solving a difficult symbol grounding problem [25], relating a language instruction to a robots onboard perception and actions. An agent in CALVIN needs to learn a wide variety of diverse behaviors to manipulate blocks with different shapes, but also needs to understand which colored block the user is instructing it to manipulate. We compare commonly used auxiliary losses for aligning visual and language representations. Concretely, we compare our contrastive loss against predicting the language embedding from the sequence's visual observations with a cosine loss [21], cross-modality matching [160] and not having an auxiliary visuo-lingual alignment loss. We observe that using an auxiliary loss to semantically align the sampled video sequences and the language instructions helps, but both baselines perform similarly. We hypothesize that our contrastive loss works best because it leverages a larger number of in-batch negatives than the cross-modality matching loss. Concretely, we maximize the cosine similarity for N real

pairs in the batch while minimizing the cosine similarity of the multimodal embeddings of the $N^2 - N$ incorrect pairings. The cross-modality matching loss implements a discriminator that produces a binary predictor of whether the embeddings match or not. The batch is shuffled only once to produce the negative samples, contrasting only N negative samples.

Language Models: Despite steady progress in language conditioned policy learning, a fundamental, but less considered aspect is the choice of the pre-trained language model to encode raw text into a semantic pre-trained vector space. We compare the lightweight paraphrase-MiniLM-L3-v2 language embeddings from our full model against several popular alternatives, such as the larger BERT [86], Distilroberta [172] and MPNet [171], which double the embedding size from 384 to 768. Besides the architecture of the language model, we analyze the impact of the loss functions the language models are trained on, by comparing the original embeddings of MPNet and Distilroberta against versions that have been finetuned with contrastive losses at the sentence level to map semantically similar sentences into the same latent space [87]. We observe that the SBERT models that have been finetuned on sentence semantic similarity achieve significantly better results than the original language models trained on masked language modeling. Concretely, the original Distilroberta model achieves an average sequence length of 2.21, while the SBERT Distilroberta model achieves an average sequence length of 2.50. Finally, we also compare against a model conditioned on visual (ResNet-50) and language-goal features from a pre-trained CLIP model [7], which has been trained to align visual and language features from millions of image-caption pairs from the internet. Surprisingly, we find that performance is slightly worse than our best performing model. We hypothesize that this might be due to a domain gap between the natural images that CLIP has been trained on and the simulated images from CALVIN. The results suggest that for complex semantics, the choice of the pre-trained language model has a large impact and models finetuned on sentence level semantic similarity should be preferred. While in this paper, we do not finetune the language models with the action loss, we anticipate this might lead to better performance, specially in order to ground instructions referring to the colored blocks.

Multi Environment and Zero-Shot Generalization: Finally, we investigate the performance of our approach on the larger multi environment splits of CALVIN on Fig. 6.4. On the zero-shot split, which consists on training on three environments and testing on an unseen environment with unseen instructions, we observe that despite modest improvements over the MCIL baseline, the policy achieves just an average sequence length of 0.67. We hypothesize that in order to achieve better zero-shot performance, additional techniques from the domain adaptation literature, such as adversarial skill-transfer losses might be helpful[133]. On the split that trains on all four environments and evaluates on one of them, we observe that CALVIN benefits from the larger dataset size and sets a new state of the art with an average sequence length of 3.06, which is higher than our best performing model trained and tested on environment D (2.64). The results suggest that increasing the number of collected language pairs aids addressing the complicated perceptual grounding problem.

6.6 Conclusion

We have presented a study into what matters in language conditioned robotic imitation learning over unstructured data that systematically analyzes, compares, and improves a set of key components. This study results in a range of novel observations about these components

and their interactions, from which we integrate the best components and improvements into a state-of-the-art approach. Our resulting hierarchical CALVIN model learns a single policy from unstructured imitation data that substantially surpasses the state of the art on the challenging language conditioned long-horizon robot manipulation CALVIN benchmark. We hope it will be useful as a starting point for further research and will bring us closer towards general-purpose robots that can relate human language to their perception and actions.

Chapter 7

Adversarial Skill Networks: Unsupervised Robot Skill Learning from Video

The content of this chapter has been published in [133]:

> O. Mees, M. Merklinger, G. Kalweit and W. Burgard
> **Adversarial Skill Networks: Unsupervised Robot Skill Learning from Video**
> *IEEE International Conference on Robotics and Automation (ICRA)*, 2020
> Finalist for the Best Paper Award in Cognitive Robotics

Me and Markus Merklingler share the main authorship. The initial work on the Adversarial Skill Networks (ASN) for learning robot skills from unlabeled videos was formulated in collaboration with Gabriel Kalweit for Markus Merklingers Master's thesis, which the author of this thesis supervised together with Gabriel Kalweit. Markus Merklinger implemented the initial experimental framework and we jointly collected the dataset. The insights gained during the aforementioned thesis supervision influenced the subsequent improved implementations of ASN that the author of this thesis carried out. All the results for ASN reported in this thesis were entirely carried out by the author of this thesis. The paper was mostly written by the author of this thesis. Markus Merklinger and Gabriel Kalweit also contributed to the paper writing.

Abstract

Key challenges for the deployment of reinforcement learning (RL) agents in the real world are the discovery, representation and reuse of skills in the absence of a reward function. To this end, we propose a novel approach to learn a task-agnostic skill embedding space from unlabeled multi-view videos. Our method learns a general skill embedding independently from the task context by using an adversarial loss. We combine a metric learning loss, which utilizes temporal video coherence to learn a state representation, with an entropy-regularized adversarial skill-transfer loss. The metric learning loss learns a disentangled representation by attracting simultaneous viewpoints of the same observations and repelling visually similar frames from temporal neighbors. The adversarial skill-transfer loss enhances re-usability of learned skill embeddings over multiple task domains. We show that the learned embedding enables training of continuous control policies to solve novel tasks that require the interpolation of previously seen skills. Our extensive evaluation with both simulation and real world data demonstrates the effectiveness of our method in learning transferable skills from unlabeled interaction videos and composing them for new tasks. Code, pretrained models and dataset are available at `http://robotskills.cs.uni-freiburg.de`

7.1 Introduction

Intelligent beings have the ability to discover, learn and transfer skills without supervision. Moreover, they can combine previously learned skills to solve new tasks. This stands in contrast to most current "deep reinforcement learning" (RL) methods, which, despite recent progress [173, 174, 175], typically learn solutions from scratch for every task and often rely on manual, per-task engineering of reward functions. Furthermore, the obtained policies and representations tend to be task-specific and generally do not transfer to new tasks.

The design of reward functions that elicit the desired agent behavior is especially challenging for real-world tasks, particularly when the state of the environment might not be accessible. Additionally, designing a reward often requires the installation of specific sensors to measure as to whether the task has been executed successfully [10, 11]. In many scenarios, the need for task-specific engineering of reward functions prevents us from end-to-end learning from pixels, if the reward function itself requires a dedicated perception pipeline. To address these problems, we propose an unsupervised skill learning method that aims to discover and learn transferable skills by watching videos. The learned embedding is then used to guide an RL-agent in order to solve a wide range of tasks by composing previously seen skills.

Prior work in visual representation learning for deriving reward functions relied on self-supervised objectives [176, 177, 178, 179, 180] and focused on single tasks. Not only is this inefficient, but also limits the versatility and adaptivity of the systems that can be built. Thus, we consider the problem of learning a multi-skill embedding without human supervision.

In this paper, we present a novel approach called Adversarial Skill Networks (ASN). In order to learn a task-agnostic skill embedding space, our method solely relies on unlabeled multi-view observations. Hence, it does not require correspondences between frames and task IDs nor any additional form of supervision or instrumentation. We combine a metric learning loss, which utilizes temporal video coherence, with an entropy-regularized adversarial skill-

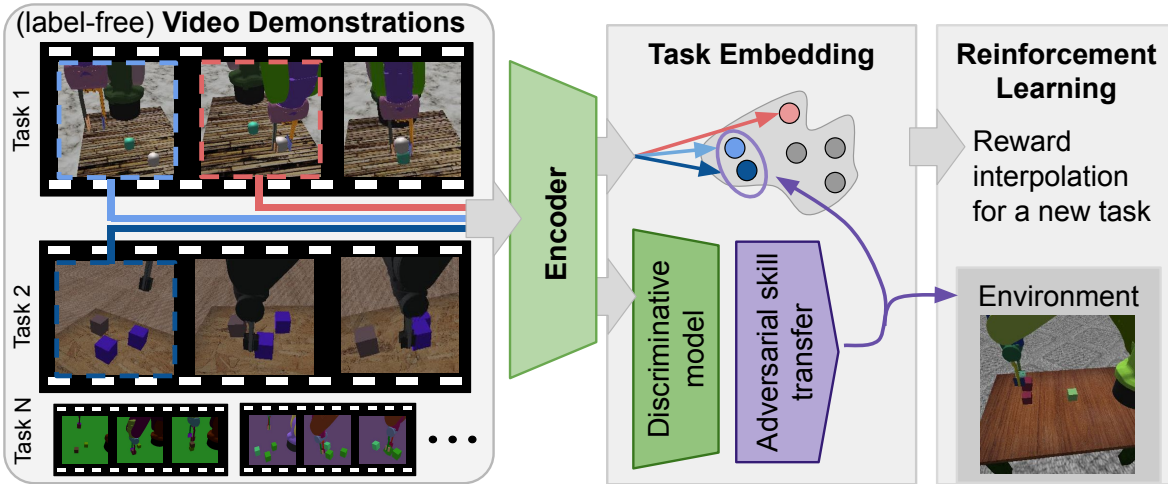


Figure 7.1: Given the demonstration of a new task as input, Adversarial Skill Networks yield a distance measure in skill-embedding space which can be used as the reward signal for a reinforcement learning agent for multiple tasks.

transfer loss. Our results indicate that the learned embedding can be used not only to train RL agents for tasks seen during the training of the embedding, but also for novel tasks that require a composition of previously seen skills.

In extensive experiments, we demonstrate both qualitatively and quantitatively that our method learns transferable skill embeddings for simulated and real demonstrations without the requirement of labels. We represent the skill embedding as a latent variable and apply an adversarial entropy regularization technique to ensure that the learned skills are task independent and versatile and that the embedding space is well formed. We show that the learned embedding enables training of continuous control policies with PPO [65] to solve novel tasks that require the interpolation of previously seen skills. Training an RL-agent to re-use skills in an unseen task, by using the learned embedding space as a reward function, solely requires a single video demonstrating the novel task. This makes our method readily applicable in a variety of robotics scenarios.

7.2 Related Work

Our work is primarily concerned with learning representations that enable a robot to solve multiple tasks by re-using skills without human supervision, thus falling under the category of self-supervised robot learning [107, 176, 178, 180]. There exists a large body of work for learning representations through autoencoders [181, 182], pre-trained supervised features [180], spatial structure [181, 183] and state estimation from vision [184]. Compared to these approaches, we take multiple tasks into account to learn a skill embedding before training a reinforcement learning agent with a self-supervised vision-based training signal.

Further approaches attempt to derive data-driven reward functions [178, 179, 180, 185, 186] by providing a label-free training signal from video or images to minimize human supervision. Related to our work, Sermanet *et al.* [180] provide reward functions by identifying key intermediate steps for one task from multiple video examples. Other methods [177, 179, 181, 186] use images of goal examples to construct a task objective for goal

reaching tasks such as pushing. Atari video games are solved in [185, 187] by constructing an objective from human demonstration videos. However, it is unclear whether the reward signal reflects a good performance when transferring it to a real-world robotic task. Our model is able to find task specific features and generalizes to unseen objects, viewpoints and backgrounds.

Existing methods for learning reusable skill embeddings make use of entropy-maximization of the policy [188, 189, 190] and therefore allow for policy interpolation. Haarnoja *et al.* [189] use a composition of soft Q-functions to create a policy that reaches a new goal. Hausmann *et al.* [188] propose a hierarchical reinforcement learning approach that utilizes two embedding networks and an entropy regularization on the policy to cover a latent space with different skill clusters. Orthogonal to our work, these methods rely on previously designed reward functions.

Most related to our approach is the work by Sermanet *et al.* [176] that introduces *Time-Constrative Networks* (TCN) and a triplet loss combined with multi-view metric learning to increase the distance of embeddings for transitions far apart in time. The learned metric can then be used as a reward signal within a RL-setup by minimizing the distance to a visual demonstration. However, TCN focuses only on the single-task setting and does not leverage information from previously learned skills. Dwibedi *et al.* [191] extend TCN using multiple frames (mfTCN). In contrast to our approach, the embedding is not used as a (label-free) reward signal.

In addition to the metric loss, we use an adversarial loss term [192, 193, 194, 195, 196] as a regularization technique. The adversarial loss was introduced for Generative Adversarial Networks (GAN) [192] and domain adaptation [194, 195, 196]. Similar to the problem of domain adaptation we have multiple videos for different task domains. For domain adaptation, multiple approaches [194, 195] use a gradient reversal layer and Tzeng *et al.* [196] exploit a GAN-based loss. Springenberg *et al.* [193] introduce an objective function for label free classification by extending GANs to categorical distributions. Our approach uses a similar adversarial loss to learn a reusable skill embedding for different task domains.

In contrast to these previously described approaches, we propose a method to learn skills from video by a composition of metric learning and an entropy-regularized adversarial skill-transfer loss. Our method not only allows for the representation of multiple task-specific reward functions, but also builds upon this information in order to interpolate between learned skills, see Figure 7.1.

7.3 Learning a Transferable Skill Embedding

The main incentive of our method is a more general representation of skills that can be re-used and applied to novel tasks. In this work, we define tasks to be composed of a collection of skills. Since we approach this problem in an unsupervised fashion, we do not need any labels describing relations between different task videos or even for different examples of the same task. In our approach, we are interested in the following properties for a learned embedding space:

i) versatility: Multiple tasks can be represented in the same embedding space.

ii) skill representation: A skill can be described by the embedding of two sequential frames, with a time delay in-between (stride).

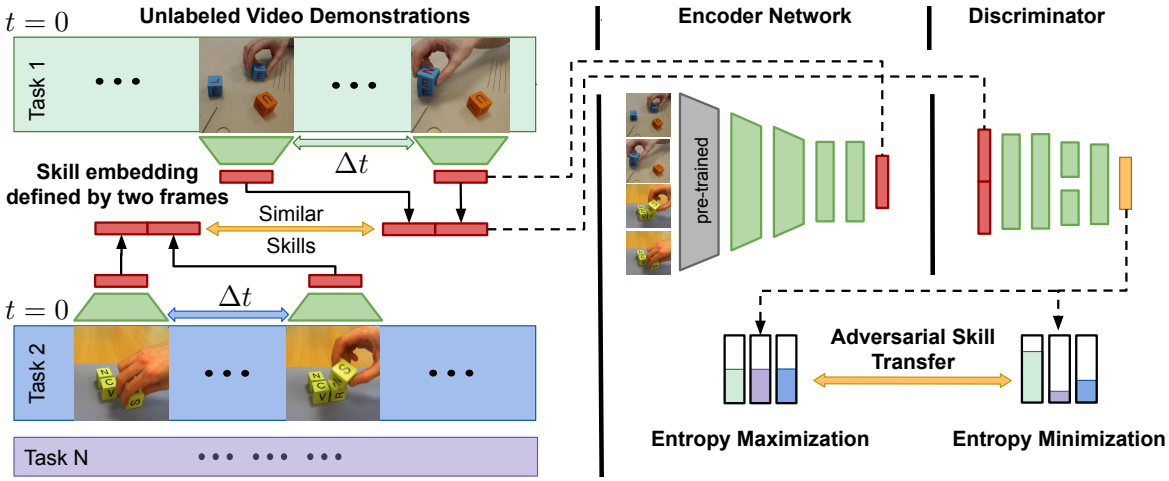


Figure 7.2: Structure of Adversarial Skill Networks: We learn a skill metric space in an adversarial framework. The encoding part of the network tries to maximize the entropy to enforce generality. The discriminator, which is not used at test time, tries to minimize the entropy of its prediction to improve recognition of the skills. Finally, maximizing the marginal class entropy over all skills leads to uniform usage of all task classes. Please note that no information about the relation between frames and the tasks they originated from is needed.

iii) generality: The learned embedding space should generalize to unseen objects, backgrounds and viewpoints.
iv) task independent skills: It should not be possible to distinguish similar skills from different task domains, i.e., the same skill executed in different environments should ideally have an identical embedding.

A Adversarial Skill Networks

We propose Adversarial Skill Networks (ASN) to achieve a novel skill representation, which takes these properties into account. We combine a metric learning loss, which utilizes temporal video coherence to learn a state representation, with an entropy-regularized adversarial skill-transfer loss. An overview can be seen in Figure 7.2.

To transfer similar skills without any label information to unseen tasks, one needs to learn generalized skills that should neither be task- nor domain-specific, following property **iv)**. Since the true class distribution over skills is not known, this problem can naturally be considered as a "soft" probabilistic cluster assignment task.

To solve this, our method introduces a novel entropy regularization by jointly training two networks in an adversarial manner: an encoder network E and a discriminator D. Given two sequential frames (v, w), which are separated by a temporal stride Δt, we define an unlabeled skill embedding $\mathbf{x} = (E(v), E(w))$ and collection of skills $\mathcal{X} = \{\mathbf{x}^1, ..., \mathbf{x}^N\}$ representing the different tasks. The encoder network embeds single frames of the dimension $d_1 \times d_2$ into a lower-dimensional representation of size n, i.e. E: $\mathbb{R}^{d_1 \times d_2} \rightarrow \mathbb{R}^n$. We compute Euclidean distances in the embedding space to compare the similarity of frames. The discriminator network takes two concatenated embedded frames that define an unlabeled skill $\mathbf{x}$ as input and outputs y_c, the probability of the skill being originated from task c. Formally, we require

$D(\mathbf{x}) \in \mathbb{R}^C$ to give rise to a conditional distribution over tasks $\sum_{c=1}^{C} p(y_c = c \mid \mathbf{x}, D) = 1$. Although we define this hyper-parameter a priori as the number of tasks contained in a training set, we observed minor performance drops setting it to a value with small deviation from the true number of tasks. Most importantly, ASN does not need a task label for the demonstration videos.

The encoder parameters are updated using a metric learning loss and maximization of the entropy of the discriminator output. In order to capture the temporal task information, we use a modified version of the lifted structure loss [197]. Given two view-pairs (v_1, v_2), synchronized videos from different perspectives, we attract frames that represent the same temporal task state and repulse temporal neighbors, given a constant margin λ, i.e.

$$\mathcal{L}_{\text{lifted asn}} = \sum_{i=1}^{M} \left(\log \sum_{y_k = y_i} \left(\mathrm{e}^{\lambda - S_{ik}} + \mathbb{1}_{S_{ik} > \xi} \cdot S_{ik} \right) + \log \sum_{y_k \neq y_i} \mathrm{e}^{S_{ik}} \right), \tag{7.1}$$

for M frames $(x_1, x_2 ... x_M)$ and $S_{ij} = E(x_i) \cdot E(x_j)$, as a dense squared pairwise similarity distance matrix of the batch and ξ a similarity threshold. Additionally, we introduce a constraint that bounds the distance between two positive view-pairs. This constraint is tailored to account for high variance in the learned distance metric. By penalizing large distances of positive view-pairs, we aim at smoother transitions between similar states in a RL setting.

The discriminator network minimizes the entropy given an unlabeled skill embedding $\mathbf{x}$ to be certain about which task C the skill originated from. Note that the discriminator D is utilized only during training. Without any additional label information about the C classes, we cannot directly specify which class probability $p(y_c = c \mid \mathbf{x}^i, D)$ should be maximized for any given skill $\mathbf{x}$. We make use of information theoretic measures on the predicted class distribution to group the unlabeled skills into well separated categories in the skill embedding space without explicitly modeling $p(\mathbf{x})$. Specifically, if we want the discriminator to be certain for the class distribution $p(y_c = c \mid \mathbf{x}^i, D)$, this corresponds to minimizing the Shannon information entropy $H[p(y_c \mid \mathbf{x}, D)]$, as any draw from this distribution should most of the times result in the same class. On the other hand, if we want the encoder to learn generalized skill representations to meet requirement **iv)**, it should be uncertain of how to classify the unlabeled skills. Thus, the encoder tries to maximize the entropy $H[p(y_c \mid \mathbf{x}, D)]$, which at the optimum will result in a uniform conditional distribution over task classes. Concretely, we define the empirical estimate of the conditional entropy over embedded skill examples $\mathcal{X}$ as:

$$\begin{aligned} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}} \Big[H[p(y_c \mid \mathbf{x}, D)] \Big] &= \frac{1}{N} \sum_{i=1}^{N} H \left[p(y_c \mid \mathbf{x}^i, D) \right] \\ = \frac{1}{N} \sum_{i=1}^{N} - \sum_{c=1}^{C} p(y_c = c \mid \mathbf{x}^i, D) \log p(y_c = c \mid \mathbf{x}^i, D). \end{aligned} \tag{7.2}$$

With an additional regularizer we enhance the equal usage of all task classes, corresponding to maximizing a uniform marginal distribution:

$$H_{\mathbf{x}}[p(y_c \mid D)] = H \left[\frac{1}{M} \sum_{i=1}^{M} p(y_c \mid \mathbf{x}^i, D) \right], \tag{7.3}$$

where M is set to the number of independently drawn samples [193].

In order to disentangle the learned metric and the mapping to task IDs, we add a sampled latent variable $z = \mu + \sigma \odot \mathcal{E}$ and $\mathcal{E} \sim N(0, 1)$, where D estimates μ, σ of a Gaussian distribution. We use the re-parameterization trick to back-propagate through the random node [58]. With the Kullback-Leibler divergence regularization for z we force D to find similar properties describing the skills. Without this objective, similar skills could end up represented far away from each other in the skill embedding space.

This leads to the following objectives for the encoder E and discriminator D:

$$\begin{aligned} \mathcal{L}_{KL} &= D_{KL}[p(z \mid \mathbf{x})||p(z)], \\ \mathcal{L}_D &= -H_{\mathbf{x}}\Big[p(y_c \mid D)\Big] + \mathbb{E}_{\mathbf{x}\sim\mathcal{X}}\Big[H[p(y_c \mid \mathbf{x}, D)]\Big] \\ &\quad + \beta\mathcal{L}_{KL} \text{ and} \\ \mathcal{L}_E &= H_{\mathbf{x}}\Big[p(y_c \mid D)\Big] + \mathbb{E}_{\mathbf{x}\sim\mathcal{X}}\Big[H[p(y_c \mid \mathbf{x}, D)]\Big] \\ &\quad - \alpha\mathcal{L}_{\text{lifted asn}}. \end{aligned} \tag{7.4}$$

We therefore optimize the discriminator and the encoder according to:

$$\min_D \mathcal{L}_D \tag{7.5}$$

and

$$\max_E \mathcal{L}_E. \tag{7.6}$$

B Implementation Details

The encoder network is inspired by Time-Contrastive Networks (TCN) [176]. We use an Inception network as a feature extractor [198], which is initialized with ImageNet pre-trained weights. The feature extractor is followed by two convolutional layers and a spatial softmax layer for dimension reduction. Finally, after a Fully Connected (FC) layer, the model outputs the embedding vector for a frame. For all experiments, we use $\alpha = 0.1$, $\beta = 1.0$, $\lambda = 1.0$ and an embedding size of 32. The discriminator consists of two FC layers to estimate μ and σ of a Gaussian distribution, followed by two layers to output the task ID. We use dropout for regularization.

We train the encoder and discriminator networks with the Adam optimizer and a learning rate of 0.001. A training batch contains 32 frames from $n = 4$ different view pairs. We load real-world data from video files and sample the simulated data from uncompressed image files. Training directly on images, ensures that our model is not learning any bias introduced by video compression techniques. For frames from the training set, we randomly change brightness, contrast and saturation and randomly mirror frames horizontally. For real-world data, additional training frames are cropped randomly. We train on images of the size $299 \times 299 \times 3$ pixels. For simulated data the discriminator network is only updated with successful task demonstrations, since only they contain the skills we want to transfer. After data augmentation, the frames of a batch are normalized on each RGB channel using the μ and σ of the ImageNet dataset.

7.4 Experimental Results

We evaluate the performance of our ASN model on two data sets, see Figure 7.3 and Figure 7.4.

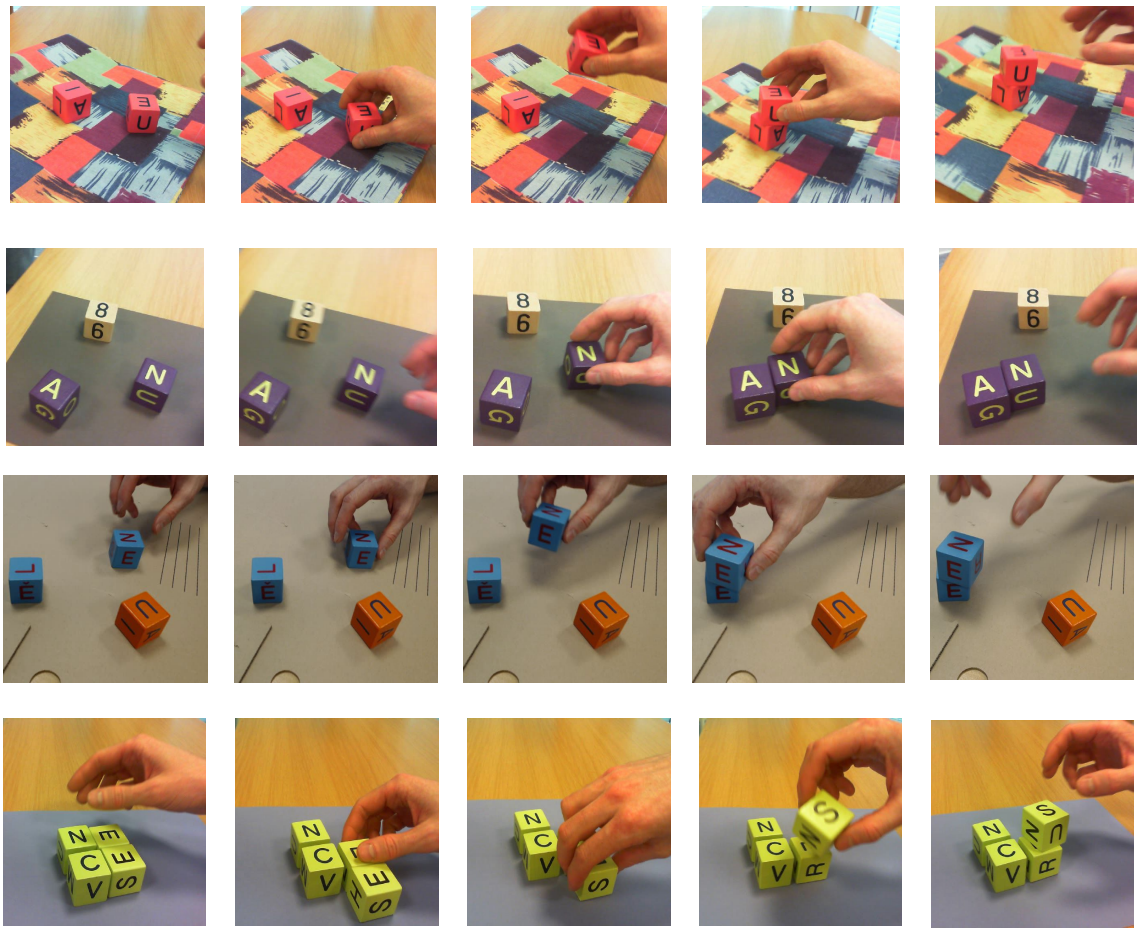


Figure 7.3: Visualization of the real-world multi-task datasets used in this work.

The first data set, visualized in Figure 7.4, consists of three simulated robot tasks: stacking (A), color pushing (B) and color stacking (C). The data set contains 300 multi-view demonstration videos per task. The tasks are simulated with PyBullet. Of these 300 demonstrations, 150 represent unsuccessful executions of the different tasks. We found it helpful to add unsuccessful demonstrations in the training of the embedding to enable training RL agents on it. Without fake examples, the distances in the embedding space for states not seen during training might be noisy. In the initial phase of training, however, the policy to be learned mostly visits areas of the state-action space which are not covered by the (successful) demonstration. Hence, it is important to have unsuccessful examples in the training set. The test set contains the manipulation of blocks. Within the validation set, the blocks are replaced by cylinders of different colors.

The second data set, visualized in Figure 7.3, includes real-world human executions of the simulated robot tasks (A, B and C), as well as demonstrations for a task where one has to first separate blocks in order to stack them (D). Each task contains 60 multi-view demonstration

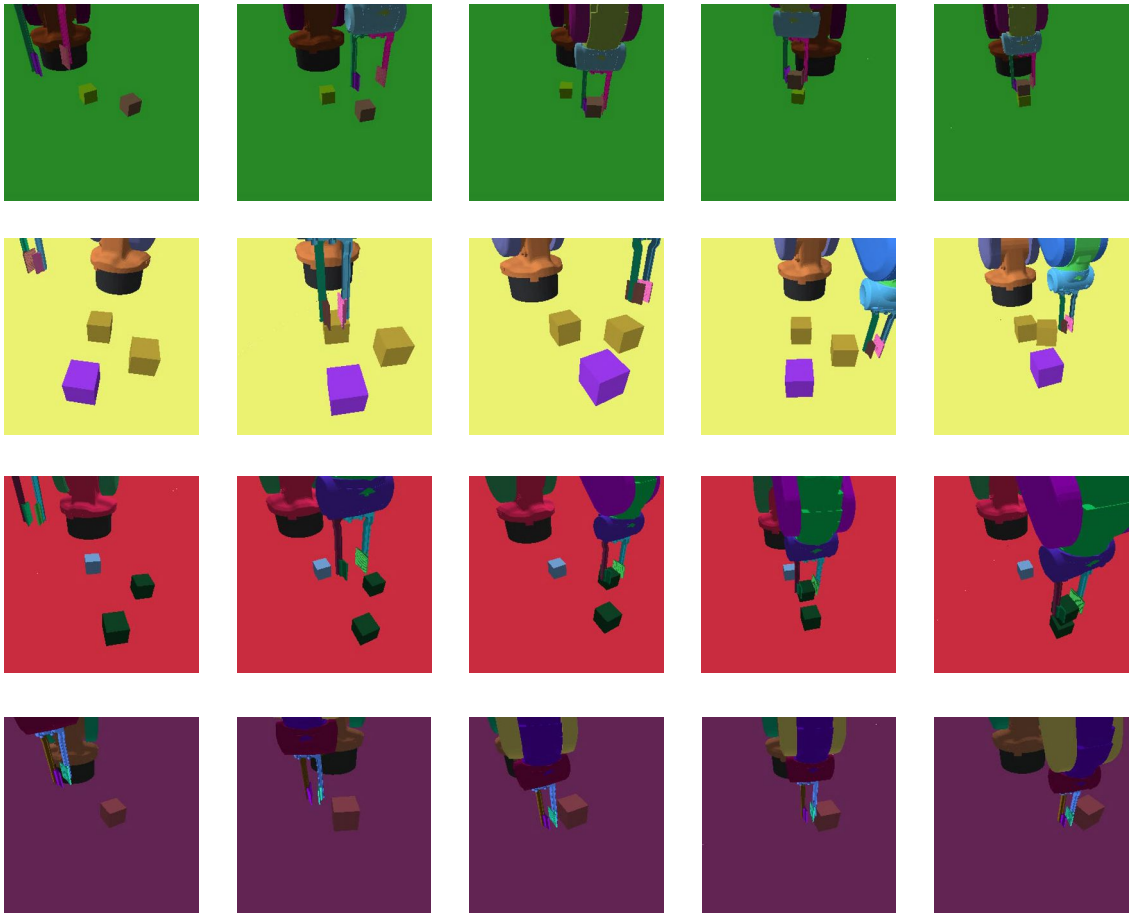


Figure 7.4: Visualization of the simulated multi-task datasets used in this work.

videos, corresponding to 24 minutes of interaction. The test set contains blocks of unseen sizes and textures, as well as unknown backgrounds, in order to evaluate the generality of our approach.

A Quantitative Evaluation

We measure the performance of our models based on the alignment loss, following Sermanet *et al.* [176], to determine how well two views of a video are aligned in time. We take advantage of the fact that frames in both videos are synchronized with each other to get alignment labels for free. We try to sequentially align two view pairs by finding the nearest neighbor in the embedding space normalizing the distances by the demonstration length. After embedding the two videos in our skill embedding space, we search for each frame t_j^i in the first video i the nearest neighbour in the second view and retrieve its time index t_j^{nn}. Thus, for video i the aligment loss is defined by:

$$\text{align}_i = \frac{\sum_{j=1}^{F} |t_j^i - t_j^{nn}|}{F}, \tag{7.7}$$

for a video of length F. As Sermanet *et al.* [176] have shown, the alignment loss reflects the quality of the reward signal within a RL setup. Instead of evaluating the alignment loss

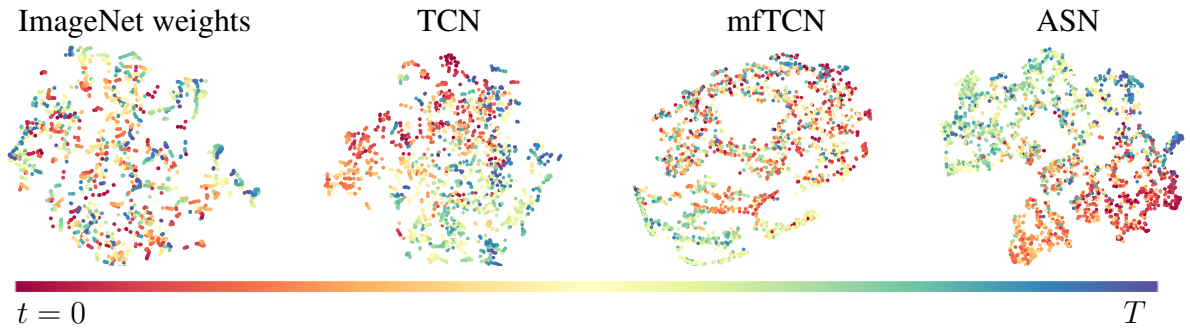


Figure 7.5: t-SNE of an **unseen** color stacking video for models trained on block stacking and color pushing. Our ASN model maintains the temporal coherence of the task better than the baselines. The colorbar indicates the temporal task progress.

on the same task as the embedding was trained on, we measure how well view pairs of novel, unseen tasks are aligned. This form of zero-shot evaluation is very challenging, as it requires the combination of previously seen skills. Adversarial Skill Networks yield the best performance for transfer in the simulated robot setting, which can be seen in Table 7.1. Please note the lower bound (0.081) of alignment loss for single-task TCN. In contrast to TCN trained on multiple tasks, our model gets very close (0.099) despite not being trained on the task. Furthermore, our approach outperforms TCN in both the real robot multi-task setup, as well as in the transfer task, which is depicted in Table 7.2. We also compare against the different metric learning losses and show that the lifted loss in combination with the bound for positive view-pairs outperforms other methods.

Model	**Task Combination, Train → Test**		
	C → C	A,B,C → A,B,C	A,B → C
Inception-ImageNet [198]	0.29	0.31	0.29
TCN - lifted [176]	0.081	0.058	0.112
ASN	-	0.056	**0.099**

Table 7.1: Test alignment loss for the simulated robot multi-task dataset, which includes fake examples, Tasks: A: 2 block stack, B: 3 block color push sort, C: 3 block color stack sort.

A visualization of the learned embedding space is depicted in Figure 7.5. ASN can represent the temporal relations of an unseen task better than TCN or multi-frame TCN, leading to more meaningful distance measures in embedding space. Our proposed multi-task setup is able to reflect skills needed to solve the unseen task and thus can generalize better, while maintaining the temporal coherence of the unseen task.

B Ablation studies

To analyze the influence of our different building blocks on the learned embedding, we conducted several experiments, see Table 7.3. Our results indicate that it is of benefit to describe a skill with a growing stride, so as to cover macro-actions describing events longer in time. In order to keep the embeddings of these skill frames of higher stride aligned, the

Model	Task Combination, Train → Test		
	A,B,C → C	A,B → C	A,B,D → C
TCN - triplet [176]	0.186	0.21	0.218
TCN - lifted [176]	0.171	0.20	0.187
TCN - npair [176]	0.221	0.209	0.221
mfTCN - lifted [191]	0.174	0.23	0.22
ASN - normal lifted	0.168	0.183	0.181
ASN	**0.150**	**0.180**	**0.165**

Table 7.2: Test alignment loss real-world block tasks, Tasks: A: 2 block stack, B: 3 block color push sort, C: 3 block color stack sort, D: 4 block separate to stack.

KL-divergence is shown to be an effective regularization technique, yielding the lowest alignment loss. Furthermore, it seems to be enough to describe a skill by only the start and end frames. A single frame seems to provide too little information whereas using four frames proves to make the state space too high dimensional.

Regularization	#Domain frames	Stride	Real block tasks A,B,D → C
KL	1	-	0.187
KL	4	5	0.185
KL	2	5	0.168
KL	2	15	**0.165**
FC	2	15	0.1987
KL w/o encoder entropy	2	15	0.186
KL w/o entropy	2	15	0.177

Table 7.3: Ablation studies: transfer loss for different regularization techniques and skill definitions.

C Learning Control Policies

Lastly, we integrate the learned metric within a RL-agent to imitate an unseen task given a single video demonstration. Concretely, for learning a continuous control policy on the color stacking (C) task we train the embedding on the tasks of two block stacking (A) and color pushing (B). Thus, successfully imitating the previously never seen color stacking task requires the interpolation of previously seen skills. Additionally, we also learn a continuous control policy for an unseen color pushing task, given an embedding trained on stacking and color stacking. To train the agents, we use the distance measure in embedding space of the agent view v_a^t and the demonstration frame v_d^t for timestep t as the reward signal for the on-policy optimization algorithm PPO [65]:

$$r^{(t)} = \begin{cases} 10 - d\Big(E(v_a^t), E(v_d^t)\Big) & \text{if } d\Big(E(v_a^t), E(v_d^t)\Big) < \xi \\ 0 & \text{otherwise,} \end{cases} \tag{7.8}$$

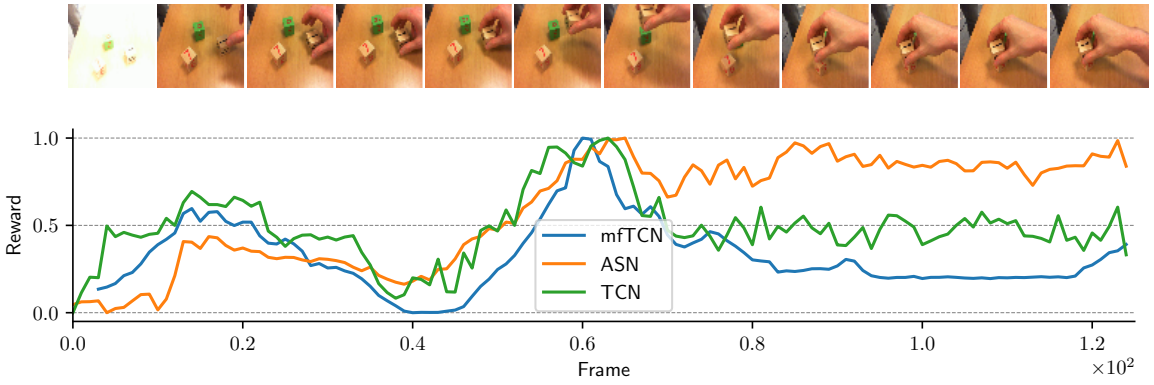


Figure 7.6: Reward plot for a novel color stacking task C for models trained on tasks A, B and D. The reward is based on the distance to a single goal frame from a different perspective.

where d is the euclidean distance and ξ a constant threshold. The agent state consists of the embedding $E(v_a^t)$ and the joint angle of the robot. We train the policy with Adam and a learning rate of 10^{-5} and a batch size of 32. To alleviate the problem of exploration and to focus on the quality of reward signal, we take random samples along the given demonstration as initial states and reset the environment if the end effector vastly differs from the demonstration, following Peng *et al.* [199]. The results are depicted in Figure 7.7.

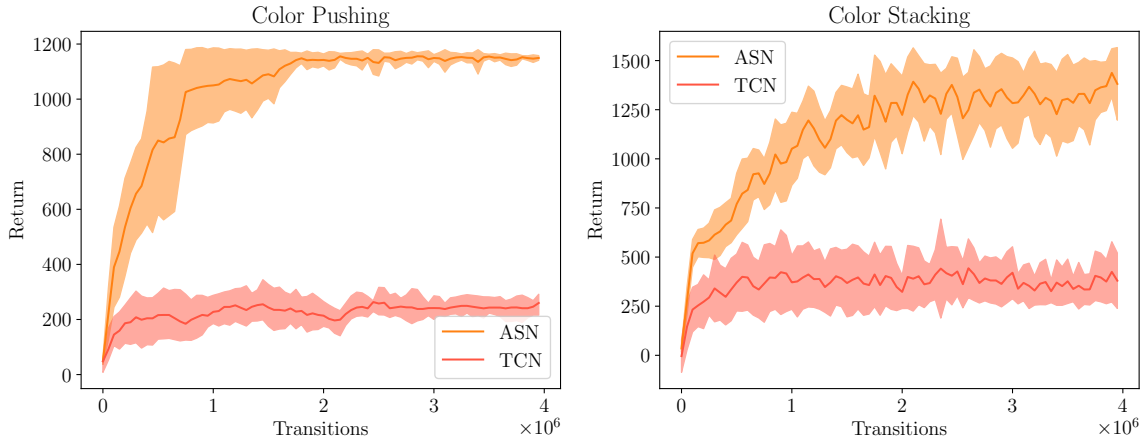


Figure 7.7: Results for training a continuous control policy with PPO on the unseen Color Pushing and Color Stacking tasks with the learned reward function. The plot shows mean and standard deviation over five training runs.

Our method succeeds in solving the tasks, whereas the baseline TCN approach converges to a local minima. This demonstrates the effectiveness of our approach in reusing skills for a novel task given a single video demonstration. Please note that training RL agents on tasks which have never been shown during the training of the embedding is very challenging, as it requires the discovery and reuse of task-independent skills.

Additionally, we evaluate the reward signal on an unseen color stacking task (C) for the real-world dataset. We plot a reward signal, which is based on the distance measurement of the task state for each timestep and a single goal frame from a different perspective, see Figure 7.6. The embedding of all models are trained on tasks A, B and D. To compare the

different models we normalize the negative distance outputs for timestep between zero and one. The baseline model already give a similar reward for many initials states and goal states, despite the states being visually different. Our model shows a continuous and incremental reward as the task progresses and saturates as it is completed.

7.5 Conclusion

We proposed Adversarial Skill Networks, a model to leverage information from multiple label-free demonstrations in order to yield a meaningful embedding for unseen tasks. We showed that our approach is able to reuse learned skills for compositions of tasks and achieves state-of-the-art performance. We demonstrate that the learned embedding enables training of continuous control policies to solve novel tasks that require the interpolation of previously seen skills. Our results show that our model can find a good embedding for vastly different task domains. This is a first step towards discovery, representation and reuse of skills in the absence of a reward function.

Going forward, a natural extension of this work is the application of the learned distance metric in a real-world reinforcement learning setting and in environments that require a higher degree of interpolation for successful completion. Another promising direction for future work is the evaluation of the proposed approach in a sim-to-real setup [200].

Chapter 8

Latent Plans for Task-Agnostic Offline Reinforcement Learning

The content of this chapter has been published in [201]:

> E. Rosete-Beas, O. Mees, G. Kalweit, J. Boedecker and W. Burgard
> **Latent Plans for Task-Agnostic Offline Reinforcement Learning**
> *6th Conference on Robot Learning (CoRL)*, 2022

Erick Rosete-Beas and I share the main authorship. The main idea to use offline reinforcement learning with latent skills was developed in collaboration with Gabriel Kalweit. Erick Rosete-Beas implemented the initial experimental framework and we jointly collected the dataset. I contributed the implementations for learning the low-level level policies with latent skills, the PLayLMP baseline, the framework to collect data via teleoperation and perform experiments in both simulation and in the real world. Erick additionally helped design and implemented the high-level policy, baseline implementations and performed the experiments in the CALVIN environment. All the real world experiments reported in this thesis were entirely carried out by the author of this thesis. Gabriel additionally provided consultation on the design of the experiments. All authors contributed to the paper writing. Joschka Boedecker and Wolfram Burgard provided general consultation.

Abstract

Everyday tasks of long-horizon and comprising a sequence of multiple implicit subtasks still impose a major challenge in offline robot control. While a number of prior methods aimed to address this setting with variants of imitation and offline reinforcement learning, the learned behavior is typically narrow and often struggles to reach configurable long-horizon goals. As both paradigms have complementary strengths and weaknesses, we propose a novel hierarchical approach that combines the strengths of both methods to learn task-agnostic long-horizon policies from high-dimensional camera observations. Concretely, we combine a low-level policy that learns latent skills via imitation learning and a high-level policy learned from offline reinforcement learning for skill-chaining the latent behavior priors. Experiments in various simulated and real robot control tasks show that our formulation enables producing previously unseen combinations of skills to reach temporally extended goals by "stitching" together latent skills through goal chaining with an order-of-magnitude improvement in performance upon state-of-the-art baselines. We even learn one multi-task visuomotor policy for 25 distinct manipulation tasks in the real world which outperforms both imitation learning and offline reinforcement learning techniques. Code and trained models available at `http://tacorl.cs.uni-freiburg.de`.

8.1 Introduction

In recent years, reinforcement learning (RL) has achieved tremendous successes in a variety of domains [175, 202, 203, 204]. Especially offline RL [70, 205, 206, 207, 208, 209] with its appealing property to estimate (close-to) optimal policies from previously collected and fixed datasets yielded a strong current in robot control research. However, despite the exceptional progress in this fast-moving field, current offline RL methods are often evaluated on highly specific and artificial benchmarks lacking the complexity and long-term dependencies of everyday tasks, which inherently entail a sequential relationship of multiple implicit subtasks. It lies in the nature of such composite tasks that this translates to estimating optimal actions for a significant amount of consecutive decision steps, making learning of such optimal policies very difficult. In fact, Kidambi *et al.* [210] discovered a quadratic relationship between the horizon of a task and the worst-case accumulated error of any offline RL method. This poses a major challenge especially in case of raw and unstructured sensory inputs, as robots must be capable of learning a large repertoire of skills and combine them to perform everyday tasks acting on long time scales.

One way to alleviate the problem of long horizons is the hierarchical subdivision of a task into high- and low-level policies, where a high-level policy is chaining executions of multiple low-level policies over primitives. Most commonly, such hierarchical structures are rather rigid and act upon a *fixed* number of low-level policies, thus lacking the flexibility and extendability required for most real-life settings. In addition, the *a priori* definition of useful low-level *skills* or their discovery from data is a highly non-trivial task. Related prior work attempted to solve this via a goal-conditioned reformulation [211] of *Conservative Q-learning* [205]. However, it was shown that on short and distinct robot manipulation

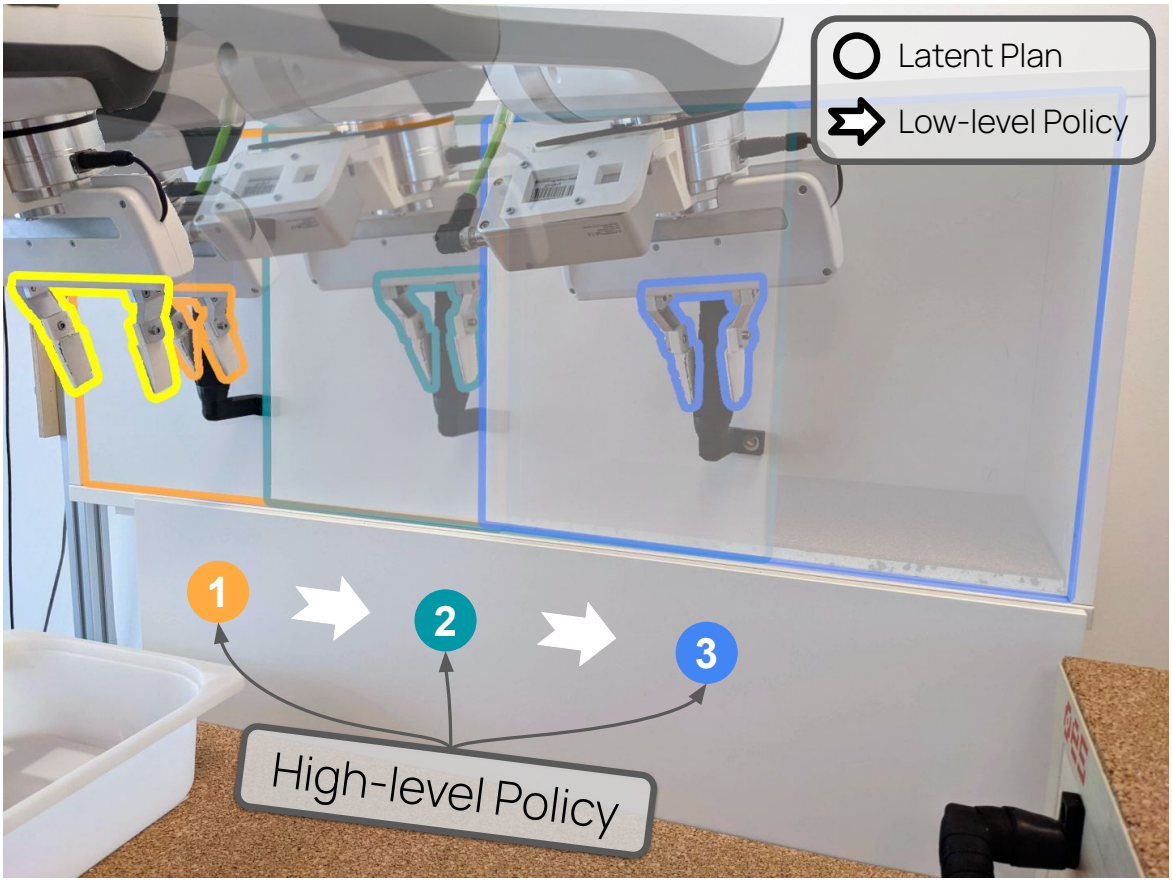


Figure 8.1: TACO-RL learns a single 7-DoF hierarchical visuomotor policy from offline data. It can solve long-horizon robot manipulation tasks by using a high-level policy that divides a task into a sequence of latent behaviors that are executed by a low-level policy that interacts with the environment. It reduces the effective horizon of the high-level policy and learns to chain skills through dynamic programming.

tasks, self-supervised learning on unlabeled *play* can significantly surpass the performance of individual expert-trained behavioral-cloning policies [18] – which, on the flip-side, can be on par with computationally expensive offline RL methods in such settings [212, 213]. In this work, we thus propose to leverage *play data*, i.e., non-goal-directed collections of trajectories grounded in human action execution, to estimate *short-horizon* expert policies via *imitation learning* chained via a coarse-grained high-level policy optimized by *offline RL* to account for optimal solutions over *long horizons*. By stitching latent plans extracted from unstructured data, our formulation offers the simplicity of imitation from collected play data while offering long-term optimality for sequential multi-tier tasks. Specifically, we use the collected data to learn our hierarchical policy as acquiring data with good state coverage and visual variety is important for successful applications of offline RL. Play data assumes access to an unsegmented teleoperated dataset of semantically meaningful behaviors provided by users, without a set of predefined tasks in mind. Unlike previous hierarchical approaches, that learn long-horizon tasks by performing a discrete set of tasks with a low-level policy, we are motivated by the idea of an agent capable of task-agnostic control. In this setting, we are endowing the agent with the ability to reach any possible target state from a given initial state. Specifically, we learn a low-level policy that decodes motor control actions from latent skills

learned through imitation learning in a self-supervised manner. This low-level policy can perform various behaviors when present in a state by conditioning the policy with a latent plan. Consequently, we also learn a high-level policy that will order these behaviors to achieve long-horizon tasks. This high-level policy is modeled as a goal-conditioned policy that outputs latent plans to be decoded by the low-level policy. The skill-chaining of the behavior priors is learned through offline RL by augmenting plan transitions with hindsight relabeling. This hierarchical approach constitutes a practical solution by decomposing a whole task into smaller chunks of sub-tasks. The high-level policy can learn long-horizon tasks as the effective episode horizon is reduced and it does not need to capture in detail the physics of the world, simplifying the underlying dynamics of the RL agent. The model formulation allows to design a general-purpose training objective by considering every possible state reached in the data as a potential task. Additionally, our approach is effective for learning policies from large and diverse datasets which do not necessarily contain optimal behaviors.

The primary contribution of this work is an hierarchical self-supervised approach to learning task-agnostic control policies from high-dimensional observations by combining model-free RL methods with imitation learning. To our knowledge, our method is the first learning system explicitly aiming to solve long-horizon multi-tier tasks from purely offline and unstructured play data without access to a model. We integrate our components in a unified framework, called Task-AgnostiC Offline Reinforcement Learning (TACO-RL). See Figure (8.1) for an overview. We show that our model obtains the highest success rate when tested against other state-of-the-art baselines on various long-horizon tasks of the challenging CALVIN environment [214] and that it is able to learn a single visuomotor 7-DoF policy that can perform a wide range of long-horizon manipulation tasks in both a simulated and a real-world tabletop environment. At test time, the real world system is capable of solving a challenging suit of 25 manipulation tasks at 10 Hz that involve more than 300 decisions per task.

8.2 Related Work

Offline Reinforcement Learning. Offline RL [70], i.e., RL from fixed and possibly mixed transition sets, constitutes a recent trend in RL and robot control research. Generally, at least in model-free settings, these techniques put a regularization on out-of-distribution actions, so as to enforce the learned policy to remain in the coverage of the dataset, since offline RL methods tend to suffer strongly from the problem of value overestimation. The simplest, yet competitive, attempt to solve this issue is a *behavioral cloning* addendum to a classical actor-critic framework [207]. *Conservative Q-learning* [205], on the other hand, imposes a penalty for actions not covered in the dataset making out-of-distribution actions non-optimal. *Fisher-BRC* parameterizes the critic as the log-behavior-policy [215] extended by a weighted offset term. *Implicit Q-learning* [209] modifies the Bellman optimality update towards a *SARSA*-like update, maximizing only over actions in the data-set. However, the rationale behind most current offline RL methods remains rather similar. While we make use of *Conservative Q-learning* in our experiments – which recently emerged as one of the most widely used benchmarks in offline RL – we want to point out that any other sophisticated improvement upon the classical offline RL objective is orthogonal to our work and could in principle be incorporated in our framework.

Hierarchical policy learning. Hierarchical policy learning involves learning a hierarchy

of policies where a low-level policy performs motor control actions and a high-level policy directs the low-level policy to solve a task. While some works [216, 217, 218] learn a discrete set of lower-level policies, each behaving as a primitive skill, this is not appropriate for a general-purpose robot that accomplishes a continuum of behaviors. A large body of hierarchical policy optimization approaches following a similar rationale to our method use planning in latent state spaces [219, 220, 221, 222, 223, 224, 225, 226] and hence require a model covering the complex dynamics of multistep skills, which is an active field of research on its own. We alleviate the necessity of a model by estimating the high-level policy via model-free RL as opposed to model-based planning and thus keep the optimization over the continuous set of skills completely at training time. In contrast to a plethora of prior work [163, 227, 228, 229, 230, 231, 232], our approach acts in the offline paradigm as it yields a lot of appealing properties for learning robot control policies. En route to a skill-chaining policy able to solve long-horizon problems, our approach exploits unstructured play data to estimate the latent skill embeddings as opposed to other work that relies on expert data or predefined skills [233, 234, 235]. Whilst in principle also other latent skill representations could be considered [133, 236, 237, 238, 239], we build upon Play-LMP [18] as it has already shown great performance and robustness in this very setting. Our design choices are directed towards a general-purpose visuomotor agent that has a high zero-shot generalization even for complex long-horizon tasks, a setting more complex than in previous offline methods [18, 240]. In summary, our approach is aiming to solve temporally extended tasks without the necessity of a model or planning from entirely offline, unstructured, unlabeled and suboptimal data. This unique combination of properties thus adds a scalable and extendable optimization method to the toolbox of robot learning.

8.3 Mathematical Foundation

In this section, we introduce notation and define the problem setting. We model the interaction between and environment and a goal-conditioned policy as a goal-augmented Markov decision process $M = (S, A, p, r, G, p_0, \gamma)$ where S represents the state-observation space, A represents the action space, $p(s'|s, a)$ is a state-transition probability function, $r(s, a, s')$ represents the reward function, $G \subseteq S$ specifies the goal space, $p_0(s)$ is an initial state distribution, and $\gamma \in (0, 1)$ represents the discount factor. We note that the agent does not have access to the true state of the environment, but to visual observations. We learn in an offline manner by assuming to use a large, unlabeled, and undirected fixed dataset $\mathcal{D} = \{(s_1, a_1), (s_2, a_2), ..., (s_T, a_T)\}$. We then relabel this long temporal state-action stream to produce a dataset of trajectories $D = \{(\tau_i = (s_t, a_t)_{t=0}^{k}\}_{i=1}^{N}$ that can be used to learn both the low-level and high-level policy without access to a model.

8.4 Offline goal-conditioned RL with TACO-RL

In this section, we elaborate on TACO-RL. Our proposed method considers the bottom-up approach; we start by training a low-level policy and we use it to provide a higher-level action space for a high-level policy that, due to this task division, is ideally facing an easier learning problem. First, we describe our unsupervised objective, which learns a continuous space of latent-conditioned behaviors $\pi_\omega(a|s_c, z)$ from D, where s_c represents the current

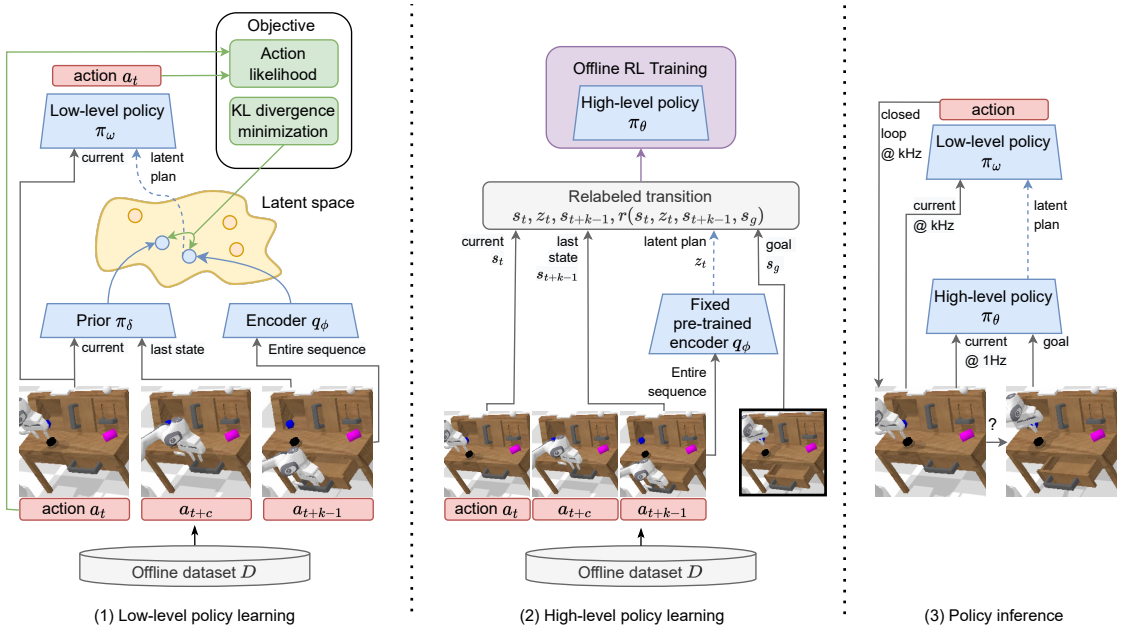


Figure 8.2: TACO-RL Overview. TACO-RL is a self-supervised general-purpose model learned from an offline dataset of robot interactions, it generalizes to a wide variety of long-horizon manipulation tasks. (1) Low-level policy: Recognizes and organizes a repertoire of behaviors from unlabeled, undirected dataset in a latent plan space. (2) High-level policy: Hindsight relabeling of sampled windows of experience into reward-augmented latent plan transitions. Learned with offline RL, this allows the high-level policy to stitch plans together to achieve complex long-horizon tasks. (3) Inference: the hierarchical model is used to perform goal-conditioned rollouts in robot manipulation tasks.

state. Afterward, we detail how to learn the high-level policy with offline RL by hindsight relabeling sub-trajectories with the aid of the previously learned low-level policy. This reduces our effective task horizon, making it easier to learn long-horizon tasks. Additionally, the low-level policy will predict actions close to the offline data distribution, bringing stability to the whole learning pipeline. See Figure (8.2) for an overview.

A Learning the low-level policy

We would like to extract a continuous space of primitives that propose meaningful behaviors for an agent to take within a given state. We learn a low-level policy $\pi_\omega(a|s_c, z)$ from the offline, unstructured dataset D that is able to decode a latent plan z to its respective motor-control actions a. After training, we can use the latent plans as an action space for the high-level policy to learn reaching temporally extended goals by "stitching" together latent skills through goal chaining.

In our fixed static dataset D, it is expected to find different valid behaviors achieving the same outcome in a scene, e.g. closing a drawer quickly or slowly. We address this inherent multi-modality by auto-encoding contextual data through a latent plan space with a sequence-to-sequence conditional variational auto-encoder (seq2seq CVAE) [18, 241]. Conditioning the action decoder on the latent plan allows the policy to use the entirety of its capacity for learning uni-modal behavior. Consequently, we propose the following objective for learning

the low-level policy $\pi_\omega(a|s_c, z)$:

$$\min_{\omega,\phi} \mathbb{E}_{\tau\sim D, z\sim q_\phi(z|\tau)} \left[-\sum_{t=0}^{|\tau|} \log(\pi_\omega(a_t|s_t, z)) \right], \tag{8.1}$$

where $\mathbb{E}$ indicates empirical expectation and $q_\phi(z|\tau)$ may be interpreted as the latent plan encoder. As an additional component of the algorithm, we enforce consistency in the latent variables predicted by encoder $q_\phi(z|\tau)$ and prior $\pi_\delta(z|s_t, s_g)$. Since our goal is to obtain a latent plan z that captures a temporal sequence of actions for a given trajectory $\tau = (s_0, a_0, ..., s_k, a_k)$, we utilize a regularization that enforces the distribution $q_\phi(z|\tau)$ to be close to just predicting the primitive or the latent variable z given the initial and last state of this sub-trajectory, i.e., $\pi_\delta(z|s_c, s_g)$. The Evidence Lower Bound (ELBO) [242] for the CVAE can be written as:

$$\log p(x|s) \geq -\mathrm{KL}\left(q(z|x, s) \parallel p(z|s)\right) + \mathbb{E}_{q(z|x,s)}\left[\log p(x|z, s)\right]. \tag{8.2}$$

The conditioning of the prior $\pi_\delta(z|s_c, s_g)$ on the initial and final state regularizes the distribution $q_\phi(z|\tau)$ to not overfit to the complete sub-trajectory τ. In practice, rather than solving the constrained optimization directly, we implement the KL-constraint as a penalty, weighted by an appropriately chosen coefficient β. Thus, one may interpret our objective as using a sequential β-VAE [243]. Finally, we use balancing terms within the KL loss [159, 244], see Appendix D.

B Offline RL with Hindsight relabeling

After distilling learned behaviors from D in terms of an encoder $q_\phi(z|\tau)$, a latent behavior policy $\pi_\omega(a|s_c, z)$, and a prior $\pi_\delta(z|s_c, s_g)$, TACO-RL then applies these behaviors to learn a general-purpose agent with offline RL. We formulate a goal augmented MDP by augmenting environment trajectories with a reward function. Thereby, we sample a trajectory from the dataset $\tau =\sim D$. Then, we use this trajectory to represent a high-level policy transition by using the pre-trained fixed encoder. For this, we sample a latent plan from the policy encoder $z_t \sim q_\phi(z|\tau)$ and we generate an interaction transition using the sampled latent plan (s_t, z_t, s_{t+k-1}). To augment this transition with a reward, we use hindsight relabeling with a sparse reward as follows $r(s_t, z_t, s_{t+k-1}, s_g) = \mathbb{1}_{s_{t+k-1}=s_g}$. In this formulation, we assume that during inference we can decode the latent plan using the low-level policy and we will reach the final trajectory state s_{t+k-1}. Note that s_t, s_{t+k-1}, and s_g all represent images, and the reward is only given when the high-level transition reached state and goal state exactly match. During training, goals are sampled according to a distribution $s_g \in S$, which we will discuss later. Our Q-learning approach corresponds to the following Bellman error optimization objective:

$$\begin{aligned} &\min_{\lambda} \mathbb{E}_{\tau\sim D, z\sim q_\phi(z|\tau), s_g\sim S} \left[Q_\lambda(s_t, z_t, g) - \hat{Q}(s_t, z_t, g) \right]^2 \\ \text{where: } \hat{Q}(s_t, z_t, g) &= \left(\mathbb{1}_{s_{t+k-1}=s_g} + \gamma \mathbb{1}_{s_{t+k-1}\neq s_g} \max_{z_{t+k-1}} Q_\lambda(s_{t+k-1}, z_{t+k-1}, s_g) \right) \end{aligned} \tag{8.3}$$

We can then learn a high-level policy $\pi_\theta(z|s_c, s_g)$ with an off-the-shelf offline actor-critic method. In TACO-RL, we use Conservative Q-Learning (CQL) [205] where we initialize

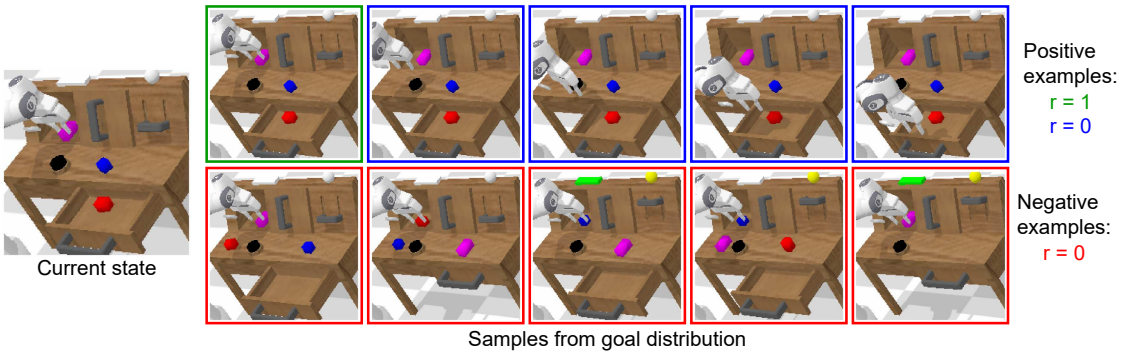


Figure 8.3: We relabel sampled trajectories into reward augmented transitions by sampling goal states that can be reached after executing a sequence of behaviors. With green border, we have the frame found at the end of the sampled trajectory. As this state will be reached after executing the latent behavior, the reward for this transition is 1. With blue border, we find future states that occur after the sampled sequence. These goals are necessary for chaining behaviors. The reward for these transitions is 0. Finally, with red border, we present images with similar proprioceptive information to the final state in the sampled trajectory, but a different scene arrangement. The reward for these transitions is 0.

the actor weights with the previously learned prior policy $\pi_\delta(z|s_c, s_g)$, as the prior policy is already a good starting point that is able to solve short-horizon tasks.

Selecting goals for relabeling transitions. Our high-level policy is learned via offline RL as we want to learn through dynamic programming how to chain skills to reach long-horizon goals. For this we need to create a formulation that sample goal states s_g that can be reached after executing a sequence of plans. Naively choosing s_g, say by sampling random states uniformly from the dataset, will provide an extremely sparse reward signal, as two random state images will rarely be identical. The sparse reward problem can be mitigated by selectively sampling as goals the states that were reached in future time steps along the same trajectory as s_t [245]. As we want to sample states that are reached after executing a plan, we assume an arbitrary window size k. Concretely, to sample goals for a transition at time step t, we sample a discrete-time offset $\Delta \sim Geom(p)$, with $p \in [0, 1]$, and use the state at time $t + \Delta * (k - 1)$ as the goal. Note that if we assume consistent transitions from latent plan decoding, then if $\Delta = 1$, the reward for this transition is 1, as the low-level policy will reach the specified state after executing the plan, avoiding the sparsity issue.

However, relabeling all transitions in this manner introduces a problem: because the distance function is only trained on goals that have been achieved, it will systematically underestimate the distance to unreachable goals. We require a method for selecting "negative" goals that are distant, but still relevant. Randomly selecting states will produce pairs of images that are likely to be distant, but not necessarily relevant (e.g., pairs in which all objects and the robot have been moved). We want a goal sampling procedure that generates less obvious examples of distant states that are more informative. Similar to Tian *et al.* [246], we sample negative goal states s_g which have a similar proprioceptive state. This constraint enables the Q-function to learn to focus on the scene's under-actuated parts (e.g., objects), which are likely to have distinct positions. As a result, these timesteps act as hard negatives, encouraging the model to pay closer attention to the scene. This sampling approach is

computationally inexpensive, as we can query a precomputed k-nearest neighbors structure.

8.5 Experimental Results

We evaluate TACO-RL for learning a general-purpose robot in both simulated and real-world environments. The goals of these experiments are to investigate: (i) whether our hierarchical model is effective in performing complex long-horizon skills, (ii) how TACO-RL compares with alternative goal-conditioned policies, (iii) if our model scales to be used in real-world robotics.

A Experimental Setup

We evaluate our approach in both simulated and real-world environments. We first investigate learning 7-DoF visuomotor robot skills in the CALVIN environment [214]. We train on the environment D of CALVIN, which contains 6 hours of unstructured play data collected via teleoperating a Franka Emika Panda robot arm to manipulate objects in a 3D tabletop environment.

Baselines Methods. As we aim to combine the complementary strengths of both paradigms, imitation and offline RL, we compare TACO-RL to representatives of the two extrema of the spectrum: the offline RL method *Conservative Q-learning* [205] extended by hindsight relabeling (CQL+HER) and the imitation learning method *Play-supervised Latent Motor Plan* (LMP) [18]. CQL+HER is trained on the derived reward as explained in Section B and introduces a penalty for out-of-distribution actions to limit the respective values of unseen actions. To make a fair comparison, this baseline is also trained with the negative mining trick. LMP, on the other hand, trains a goal conditioned imitation agent and resembles the low-level policy in TACO-RL, however, without any long-term optimality guarantees which TACO-RL accounts for by offline RL of a higher-level plan-selective policy. Additionally, we also compare against *Relay Imitation Learning* (RIL) [230]. This algorithm is the most related hierarchical algorithm, as it also learns from offline play data. RIL represents the family of methods that learns to predict latent subgoals for a low level policy.

B Simulation Results

We start by evaluating our approach in the CALVIN environment [214]. This is a challenging environment as the scene changes through time and we act by using only RGB images of a static camera as input. As there is no predefined reward signal in this dataset, we relabel the transitions analogously as we do with TACO-RL. We investigate if our method is capable of performing complex long-horizon tasks in a robot control setting. We first attempt to solve 500 unique chains of 5 image-based goals queried in a row. For each subtask in a row the policy is conditioned on the current sub-goal image instruction and transitions to the next sub-goal only if the agent successfully completes the current task or if 180 timesteps have passed without reaching a success. We call this evaluation of performing multiple tasks on a row, long-horizon multitask with visual observations *LH-MTVis*. This setting is very challenging as it requires agents to be able to transition between different subgoals. Additionally, we ablate our model by increasing the negative goals ratio to 50% and removing the negative mining.

Method	LH-MTVis					
	No. Instructions in a Row (500 chains)					
	1	2	3	4	5	Avg. Len.
Ours	**95.4%**±2	**82%**±7.3	**57.8%**±15	**32.7%**±10	**6.9%**±3.1	**2.7**±0.3
No neg. goals	94.9%±4.5	69.7%±14.3	31.1%±15.7	5.5%±4.5	0.9%±0.8	2.02±0.4
50% neg. goals	71.8%±2.2	27.1%±1.8	6.2%±2.6	0.2%±0.3	0%±0	1.05±0.1
RIL [230]	70.3%±3.5	32.9%±7.2	10.5±4.01	2.4±0.7	0.1±0.1	1.17±0.15
LMP [18]	91.4%±2.3	63.3%±3.5	23.1%±2.2	3.6%±0.9	0.2%±0.08	1.8±0.08
CQL+HER	65.5%±12.7	25.3%±11	5.6%±3.2	0.6%±0.2	0%	0.9±0.2

Table 8.1: Success rates of models running 500 chains per three different random seeds, using intermediate sub-goal images.

In Table 8.1. we can see that TACO-RL is able to outperform all baselines. This experiment demonstrates that our agent is able to transition between different sub-goal images more naturally, enabling chaining more tasks in a row. Through our ablations, we observe that the performance of our model drops significantly when increasing the negative goal ratio to 50%. This result is to be expected as reducing the number of positive examples leads to a less informative reward indicating which are the useful behaviors to reach a transition. If we remove the negative goals from the goal distribution, the agent is still able to perform more sequential tasks than the baselines, but it underestimates the distance to the goals. This results in a decreased performance compared to our full approach.

Method	LH-MTVis		
	No. Instructions in a Row (1000 chains)		
	1	2	Avg. Len.
Ours	**67.9%**±3.9	**27%**±2.5	**0.94**±0.06
No neg. goals	39.5%±2.7	4.2%±1.6	0.44±0.04
50% neg. goals	45.4%±5.2	6.3%±0.6	0.52±0.05
RIL [230]	66.2%±6.5	13.3%±2.3	0.79±0.09
LMP [18]	34.3%±3.2	2.7%±0.1	0.3±0.03
CQL+HER	35.2%±3.4	2.4%±0.8	0.37±0.04

Table 8.2: Success rates of models running 1000 chains per three different random seeds conditioned only on the last goal image.

We then evaluate the capacity of our model to perform 1000 rollouts of two sequential tasks using a single goal image. For this, we allow the agent to perform actions until 300 timesteps has passed. We record the success rate of all models in Table 8.2. TACO-RL successfully performs long-horizon tasks that require reasoning over sequential behaviors with an final success rate of 27% which corresponds to an order of magnitude improvement upon the LMP and CQL+HER baselines. RIL also exploits a hierarchical structure which allows it to reason over longer horizons than the other baselines, but TACO-RL still obtains more than two times its accuracy when performing both sequential tasks using a single image, proving its effectiveness in chaining skills through dynamic programming.

We further test the capacity of the models to perform a single task when the goal image does not contain the robot performing the task, but the end-effector appears in another position after the task was performed (cf. Table 8.3). We run 50 rollouts for each task. These harder tasks require reasoning about changes in the scene and additionally evaluates generalization, as each rollout uses a different goal image not seen during training. With

Figure 8.4: Real-world Manipulation Tasks. Examples shown from left to right are: closing the drawer, opening the drawer, moving the sliding door left, moving the sliding door right, lifting the block, rotating the block, pushing the block, turning the green LED on, placing the block on top of the drawer and placing the block in the container.

our ablations, we can see that including the negative goals in our framework contributes greatly to obtain a good performance in this scenario alleviating to not only imitate the final end-effector position, but to reach the entire scene configuration. We noticed that TACO-RL was able to outperform the baselines, which can be explained through the skill-chaining abilities of our model and the negative goal mining used to train the critic network of the high-level policy. On the other hand, CQL+HER is not able to achieve the desired goal image and LMP has a strong bias towards the end-effector position ignoring the changes in the environment. RIL can imagine intermediate latent sub-goals required to achieve the task, which reduces the bias towards the end-effector position, but it stills obtain a lower accuracy rate than our method.

C Real-Robot Experiments

For the real-world experiments, we investigate learning a single policy capable of performing multiple goal conditioned tasks. Examples of the tasks are shown in Figure (8.4). To generate the training dataset we collected nine hours of play data recorded via teleoperating a Franka Emika Panda robot arm with a VR controller. To avoid self-occlusions in the scene, these models also receive RGB images of a gripper camera as an additional input. After training the models with the offline dataset, we performed 20 rollouts for each task using multiple goal images and start positions. TACO-RL was able to outperform the baselines consistently, especially when evaluated from a start position far away from the goal image or that required extended reasoning. We recorded the success rate of each model in Table 8.4.

We also tested our approach to perform sequential tasks with the real robot to verify that our approach can be scaled for long-horizon tasks. For this experiment we use a goal image for each task that the robot executes. See the supplementary video for qualitative results that showcase the diversity of tasks and the long-horizon capabilities of the different methods. Our agent trained completely from unlabeled play data is able to successfully perform most of these sequential tasks, by inferring how to transition between tasks and reach the state depicted by the goal image. More details in Appendix F.

Finally, we evaluate TACO-RL performing complex tasks with a single goal image, such as lifting the block and moving it to a desired position, and stacking a block on top of another. These tasks requires the agent to reason in a long-horizon manner, as if the robot imitates only the end-effector position, the objects would not be arranged correctly in the scene. By stitching together the learned latent behaviors, our model was able to perform these tasks consistently.

Method \Task	Place block in drawer	Open drawer	Move slider left	Turn on lightbulb
Ours	**94%**±8.7	**87.3%**±5	**79.3%**±11.7	**94%**±4
No neg. goals	77.3%±6.1	37.3%±32.3	13.3%±9.5	9.3%±6.1
50% neg. goals	88%±7.2	58.7%±21.6	39.3%±25.3	92%±4
RIL [230]	74.67%±26.6	**87.3**%±1.15	77%±1.4	92.7%±2.3
LMP [18]	78.6%±6.1	12%±2	12%±5.2	10%±2
CQL+HER	67.6%±20.8	8.6%±5	17.3%±8	4%±4

Table 8.3: The average success rate of goal-conditioned models running 50 rollouts where the goal image does not contain the end-effector performing the task. Three models trained from different random seeds were used to perform the rollouts.

Task \Method	Ours	LMP [18]	CQL+HER
Lift the block on top of the drawer	**60**%	**60**%	20%
Lift the block inside the drawer	**65**%	50%	15%
Lift the block from the slider	**60**%	30%	10%
Lift the block from the container	**65**%	60%	20%
Lift the block from the table	**70**%	**70**%	30%
Place the block on top of the drawer	**60**%	50%	30%
Place the block inside the drawer	**70**%	40%	20%
Place the block in the slider	**30**%	0%	0%
Place the block in the container	**65**%	30%	15%
Stack the blocks	**30**%	0%	0%
Unstack the blocks	**30**%	10%	0%
Rotate block left	**70**%	40%	10%
Rotate block right	**70**%	50%	15%
Push block left	**60**%	50%	20%
Push block right	**60**%	50%	10%
Close drawer	**90**%	70%	20%
Open drawer	**70**%	50%	10%
Move slider left	**75**%	30%	0%
Move slider right	**70**%	10%	0%
Turn red light on	**60**%	30%	0%
Turn red light off	**50**%	20%	0%
Turn green light on	**70**%	60%	10%
Turn green light off	**65**%	50%	10%
Turn blue light on	**50**%	**50**%	5%
Turn blue light off	**50**%	30%	10%
Average over tasks	**61**%	40%	11%

Table 8.4: The average success rate of the multi-task goal-conditioned models running roll-outs in the real world.

8.6 Conclusion and Limitations

In this paper, we introduced *Task-Agnostic Offline Reinforcement Learning* (TACO-RL), which exploits a latent plan representation estimated from unstructured play data to effectively limit the horizon of a high-level offline RL policy acting upon this latent plan space. By dividing sequential multi-tier tasks into chunks of implicit subtasks solved by imitation learning, TACO-RL showed up to an order-of-magnitude improvement in performance compared to state-of-the-art both imitation learning and offline reinforcement learning baselines in both, simulated and real robot control tasks.

While TACO-RL is quite capable, it does have a number of limitations. Specifying a task to the requires providing a suitable goal image at test-time, which should be consistent with the current scene. Besides, tracking task progress might be useful when sequencing skills of different time horizons. We discuss limitations in more detail in Appendix H. But overall, we are excited about the confluence of imitation and offline RL methods towards scaling robot

learning.

Appendix

A Teleoperation Interface

Simulation

To collect data for simulation the human teleoperator uses the HTC VIVE Pro headset to visualize the CALVIN environment and its controller to collect play data. Clicking the application menu button emits a signal to start recording a play data episode, since then the robot arm tracks the controller's position and orientation and maps it to the gripper end-effector position and orientation.

Control	Function
Application menu	Starts or finishes recording a play data episode
Trigger	When pressed it closes the gripper end-effector, otherwise it opens

Real World

For the real world we use the HTC VIVE Pro controller. At the beginning of the teleoperation, we calibrate our X axis position by clicking the grip button and moving towards its positive axis, we then click the application menu to indicate that we finished the calibration movement. This calibration procedure allows us to define the reference frame in which the global coordinates from the robot end-effector is be recorded. In this setting, the teleoperator is facing the table in the same direction as the robot, hence if the controller moves right so does the robot. This way we can map the position of the controller to an absolute action defining the desired robot arm end-effector 3D position. Additionally, we must press continuously the grip button to enable movement of the robot arm, this is to avoid involuntary hand tracking and prevent possible accidents.

Control	Function
Grip	Starts calibration; enables robot movement
Application menu	Finishes calibration; Starts or finishes recording a play data episode
Trigger	When pressed it closes the gripper end-effector, otherwise it opens

B Experimental Setup Details

For both our real and simulated robot environments we use the following 7-dimensional action scheme:

$$[\delta x, \delta y, \delta z, \delta alpha, \delta beta, \delta gamma, gripperAction]$$

The $\delta x, \delta y, \delta z$ action dimensions corresponds to a change in the end-effector's position in 3D space. The $\delta alpha, \delta beta, \delta gamma$ action dimensions specifies a change in the end-effector's orientation in the robot's base frame. All these 6 dimensions accept continuous values between $[-1, 1]$. Finally, the $gripperAction$ command can have two discrete values -1.0 and 1.0. An action of -1.0 in this dimension commands the robot to open the gripper and 1.0 indicates that we desire to close it.

Data Collection Details

For the real world robot, we collected nine hours of play data by teleoperating a Franka Emika Panda robot arm, were we also manipulate objects in a 3D tabletop environment. This environment consists of a table with a drawer that can be opened and closed. The environment also contains a sliding door on top of a wooden base, such that the handle can be reached by the end-effector. On top of the drawer, there are three led buttons with green, blue and orange coatings to be able to identify them, on the recorded play data we only interacted with the led button with green coating. When the led button is clicked, it toggles the state of the light. Additionally, there are three different colored blocks with letters on top. We visualize the data collection and setup in Figure 8.5.

Figure 8.5: Visualization of the real world data collection procedure (left) and the full robot setup (right).

In every time step we record the measurements from the robot proprioceptive sensors, as well as an static RGB-D image of size 150×200 from the Azure Kinect camera and a gripper RGB-D image of size 200×200 emitted by the FRAMOS Industrial Depth Camera D435e and the commanded absolute action. For this project we only use the static RGB image and the gripper camera RGB image as part of the observation. We visualize the input observations in Figure 8.6.
We extract the relative action a_t from the change in the absolute actions from time steps a_t and a_{t-1} as in practice, reproducing the relative actions computed this way showed a better performance than when computed from the noisy measurements of the proprioceptive sensors.
The data is collected at 30 Hz, but given that there is relatively a very small change in the end-effector position between frames, we downsample the processed data to a frequency of 15 Hz.

C Network Architecture

Hyperparameters for TACO-RL

Low-Level Policy: To learn the low-level policy we trained the model using 8 gpus with Distributed Data Parallel. Throughout training, we randomly sample windows between length 8 and 16 and pad them until reaching the max length of 16 by repeating the last observation and an action equivalent to keeping the end-effector in the same state. We visualize the low-level policy architecture in Figure 8.7. Additional hyperparameters are listed below:

Figure 8.6: Visualization of the observations obtained in the real world environment. On the left we show the RGB image captured by the static camera. On the right we show the RGB image captured by the gripper camera.

- Batch size of 64
- Latent plan dim of 16
- Learning rate of $1e-4$
- The KL loss weight β is $1e-3$ and uses KL balancing

These hyperparameters were chosen in order to make the action space of the high-level policy tractable for reinforcement learning, as the latent plan z is used as an action when learning the high-level policy.

High-Level Policy: Similarly as in the low-level policy, we also trained the model using 8 gpus with Distributed Data Parallel. We visualize the architecture in Figure 8.8. We use the following hyperparameters:

- Batch size of 64
- Learning rates $-$ Q-function: $3e-4$, Policy: $1e-4$,
- Target network update rate of 0.005
- Ratio of policy to Q-function updates of $1:1$
- Number of Q-functions: 2 Q-functions, min(Q1, Q2) used for Q-function backup and policy update
- Automatic entropy tuning: $True$, with target entropy set to $-\log|A|$
- CQL version: $CQL(\mathcal{H})$ (Using deterministic backup)
- α in CQL: 1.0 (we used the non-Lagrange version of $CQL(\mathcal{H})$)
- Number of negative samples used for estimating $logsumexp$: 4

- Initial BC warmstart epochs: 5
- Discount factor of 0.95

It is important to mention that the model performance is robust to slight changes in the hyperparameter selection.

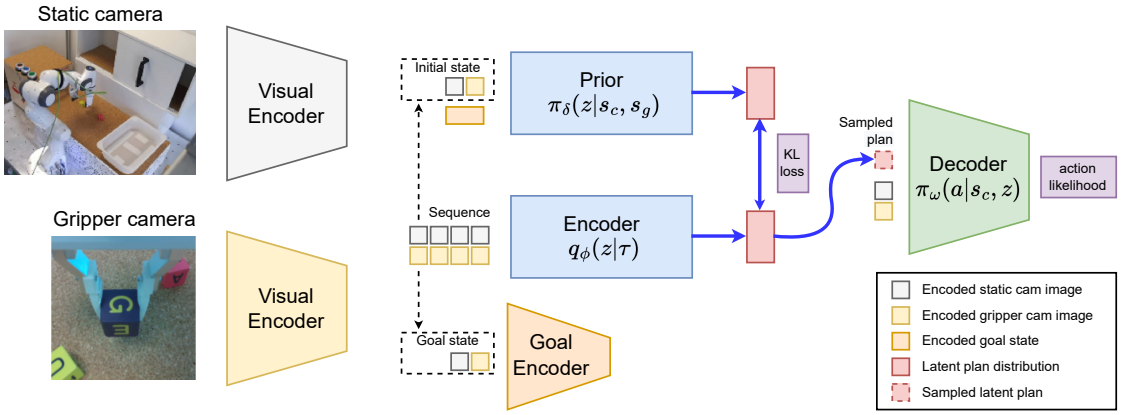


Figure 8.7: Overview of the architecture used in the real world to learn the low-level policy.

Goal Sampling Strategy: We use the same geometric distribution probability of $p = 0.3$ for all experiments. When this hyperparameter is closer to 0 it will be able to stitch plans to achieve longer horizon tasks, but it will encounter less often a reward of 1 which makes the optimization problem harder. For the 3D tabletop environment in both simulation and real world, we sample positive examples (goals from the future) 90% of the time and negative examples (goals with similar proprioceptive state) 10% of the time.

Encoder

The encoder (aka Posterior) $q_\phi(z|\tau)$ encodes the trajectory τ of state-action pairs into a distribution in latent space and gives out parameters of that distribution. In our case, we represent $q_\phi(z|\tau)$ with a transformer network, which takes τ and outputs parameters of a Gaussian distribution $(\mu_z^{enc}, \sigma_z^{enc})$. We encode the sequence of visual observations with our vision encoder. Then, we add positional embeddings to enable the experience window to carry temporal information. Finally, we fed the result into the transformer to learn temporally contextualized global video representations. In particular, our transformer encoder architecture consists of 2 blocks, 8 self-attention heads, and a hidden dimension of 2048.

Decoder

The decoder (aka Low-Level Policy): $\pi_\omega(a|s_c, z)$ is the latent-conditioned policy. It maximizes the conditional log-likelihood of actions in τ given the state and the latent vector. In our implementation, we parameterize it as a recurrent neural network which takes as input the current state and the sampled latent plan and gives out parameters of a discretized logistic mixture distribution [167].
For our experiments, we use 2 RNN layers each containing a hidden dimension of 2048. In the discretized logistic mixture distribution we use 10 distributions, predicting 10 classes

per dimension. We predict an action where the 6 first dimensions are continuous in a range between -1.0 and 1.0 and its last dimension contains the gripper action, which is a discrete value optimized by cross entropy loss.

Prior

The prior $\pi_\delta(z|s_c, s_g)$ tries to predict the encoded distribution of the trajectory τ from its initial state and its goal state. Our implementation uses a feed-forward neural network which takes in the embedded representation of the initial state and goal state and predicts the parameters of a Gaussian distribution $(\mu_z^{pr}, \sigma_z^{pr})$. The prior network consists of 3 fully connected layers with a size of 256.

Visual Encoder

To obtain the current state embedded representation we pass each input modality observation through a convolutional encoder and we perform late fusion by concatenating the embedded representations of all the observation modalities.

Simulation

In simulation we only use the RGB static image as input, this is passed through the same convolutional encoder proposed in the original Play LMP implementation with 3 convolulational layers and a spatial softmax layer, after flattening its representation it is fed through two fully connected layers which output an embedded latent size of 32.

Real World

For the real world we use both RGB static image and the RGB gripper image as input. We send the RGB gripper image to a convolutional encoder as in original Play LMP implementation, with an embedded latent size of 32. For the RGB static image we used the pretrained R3M [247] Resnet18 networks with fixed weights, afterwards we passed it through 2 feed-forward networks with a hidden size of 256 and an embedded latent size of 32.

Goal Encoder

We obtain a compact perceptual representation from the goal observation by passing it through the visual encoder. Then, we use 2 hidden layers with a size of 256 and an output layer that maps its representation to 32 latent features.

Actor

To learn the high level policy we use CQL. In particular, The actor network uses the same network architecture as the prior $\pi_\delta(z|s_c, s_g)$, as we can use the pre-learned weights as a good initialization. It has 3 hidden fully connected layers with a size of 256 and an output layer that predicts the mean and standard deviation of a latent plan Gaussian distribution.

Critic

The critic network takes as input the embedded representation of the current state, the encoded goal state and a sampled latent plan. All the inputs are concatenated and passed through a feed forward network with 3 hidden layers with a size of 256 and an output layer which predicts the expected state-action value of the policy.

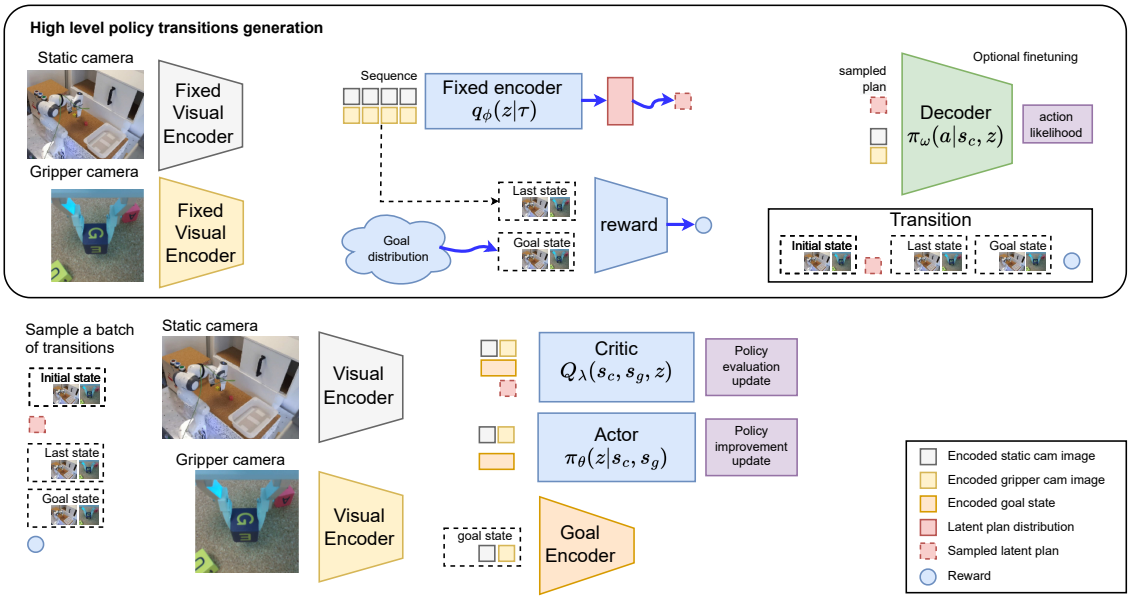


Figure 8.8: Overview of the architecture used in the real world to learn the high-level policy. We first used the learned low level policy to generate transitions using latent plans, afterwards we sample a batch of transitions and use CQL to learn the high level policy.

D Policy Training Details

In this section we specify some implementation details of our baseline models used as comparison against TACO-RL.

Play-supervised Latent Motor Plans (LMP)

For our reimplementation of LMP we used the same network architecture described in C, the main differences with the original implementation is that the latent goal representation is only added to the prior, but not to the decoder. Additionally, we apply a $\tanh$ transformation to the sampled latent plan to ensure that every dimension is between -1.0 and 1.0. Another difference with respect to the original formulation is that we implement a KL balancing. As the KL-loss is bidirectional, we want to avoid regularizing the plans generated by the posterior toward a poorly trained prior. To solve this problem, we minimize the KL-loss faster with respect to the prior than the posterior by using different learning rates, $\alpha = 0.8$ for the prior and $1 - \alpha$ for the posterior, similar to Hafner *et al.* [244]. These architectural changes were made to do a fair comparison against our implementation, and to make sure the improvements of our method were due to the plan stitching capabilities and our self-supervised reward function.
Additionally, the decoder predicts relative actions instead of absolute actions as this leads to an overall better performance. We note that our LMP baseline is implemented in the same way.

Conservative Q-Learning with Hindsight Experience Replay (CQL + HER)

We use the same visual encoder architecture as in TACO-RL. The critic networks are made with 3 fully connected layers using a hidden dimension of 256. The actor network also uses 3

fully connected layers with a size of 256 and predicts in the last dimension a discrete action to open and close the parallel gripper. This last dimension is predicted through a Gumbel Softmax distribution.
The self-supervised reward function is done similarly as in our method with a small difference when sampling goal states from the future. The CQL transitions are (s_t, a, s_{t+1}) instead of (s_t, z_t, s_{t+k-1}), therefore when we sample a goal from discrete-time offset $\Delta \sim Geom(p)$, we use the observations at the time step $t + \Delta$ as a goal instead of the observation at the timestep $t + \Delta * (k - 1)$, where k is the window size and p is the same value for both implementations. This change is required as CQL needs to take isolated decisions every time step. In contrast, our formulation allows TACO-RL to reduce the effective episode horizon by predicting high-level actions (latent plans).

Relay Imitation Learning (RIL)

We use the same visual encoder architecture as in TACO-RL to do a fair comparison. For the policy architecture, we first tried the architecture proposed by the original authors of each policy using two layer neural networks with 256 units each and ReLu nonlinearities. This architecture obtained poor performance for our application, we noticed that the respective loss objectives were not being correctly optimized for which we solved this issue by training a bigger network. For our tested implementation, for both the high-level and low-level policy we used four layer neural networks with 1024 units each.

E Data Preprocessing

Simulation

In simulation we only use the static camera RGB image as input, we first resize the RGB image from 200×200 to 128×128. Then we perform stochastic image shifts of $0 - 6$ pixels to the static camera images and a bilinear interpolation is applied on top of the shifted image by replacing each pixel with the average of the nearest pixels. Then we apply a color jitter transform augmentation with a contrast of 0.1, a brightness of 0.1 and hue of 0.02. Finally, we normalize the input image to have pixels with float values between -1.0 and 1.0.

Real World

In the real world as there are several occlusions only using the static camera RGB image, we also add the gripper camera RGB image to the observation. For the static camera RGB image we use the original size of 150×200, we then apply a color jitter transform with contrast of 0.05, a brightness of 0.05 and a hue of 0.02. Finally, we use the values for the pretrained R3M normalization, i.e., mean $= [0.485, 0.456, 0.406]$ and a standard deviation, std $= [0.229, 0.224, 0.225]$.
For the gripper camera RGB image we resize the image from 200×200 to 84×84, we then apply a color jitter transform with contrast of 0.05, a brightness of 0.05 and a hue of 0.02. Then we perform stochastic image shifts of $0 - 4$ pixels to the and a bilinear interpolation is applied on top of the shifted image by replacing each pixel with the average of the nearest pixels. Finally, we normalize the input image to have pixels with float values between -1.0

and 1.0.

F Additional Results

CALVIN

We add the rest of the results of all the different tasks that could be made in the CALVIN environment where the goal image does not need to contain the robot end-effector performing the task in Figure 8.9. We run 50 rollouts for each task. Each rollout uses a different goal image which was not seen throughout training.

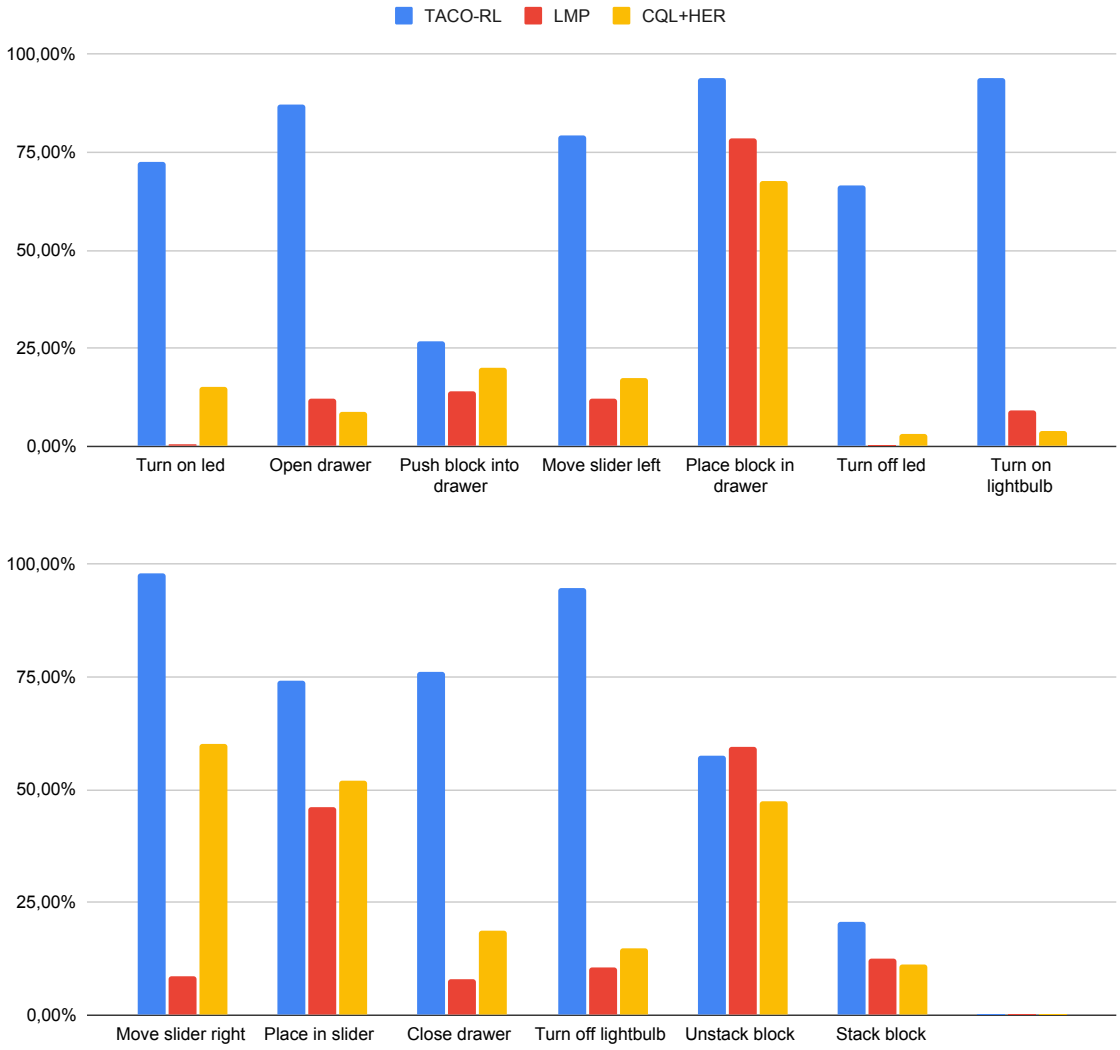


Figure 8.9: The average success rate of goal-conditioned models running 50 rollouts where the goal image does not contain the end-effector performing the task. Results were calculated using 3 seeds.

Maze2D

We also evaluate our algorithm in the Maze2D environments from D4RL [248]. These tasks are designed to provide a simple test that the model is capable of stitching together various subtrajectories such that the agent is capable of reaching the goal in the smallest amount of steps. For TACO-RL, we learn the policy by relabeling the transitions instead of using the rewards given by the dataset. Additionally, given that the scene does not change through time, we only add positive examples in our goal sampling approach. We perform the rollouts of both LMP and TACO-RL by exposing the desired target position. TACO-RL outperforms both LMP and CQL as it is able to stitch plans through dynamic programming as shown in Table 8.5.

Environment	TACO-RL	LMP [18]	CQL [205]
maze2d-umaze	**110**$\pm$2.2	81.9$\pm$14.9	5.7
maze2d-medium	**88.9**$\pm$2	77$\pm$18	5.0
maze2d-large	**76.7**$\pm$9.7	20.1$\pm$29.3	12.5

Table 8.5: The average normalized score for three different random seeds. The CQL results were obtained from the D4RL whitepaper [248].

Real World

To make sure that our method can be scaled for long-term tasks, we used our framework to control a real robot to carry out consecutive tasks. For the purposes of this experiment, we assign a goal image for each of the robot's tasks. Our agent, which was trained entirely from unlabeled play data, can complete all of these sequential tasks by inferring how to transition between them, achieving the state depicted by the goal image. Examples of these sequential tasks are listed below:

- Moving the sliding door to the right and then opening the drawer
- Lifting the block and putting it on top of the drawer
- Lifting the block and placing it on a plastic container
- Turning the blue light on and then opening the drawer
- Turning the green light on and then open the drawer

G Negative Results

We present an incomplete list of experiments tried throughout the course of the research project. These ideas were tried a couple of times, but they in general do not improve performance. It is possible that they could work better with a more thorough investigation. We hope this experience will be helpful to researchers building on top of this work.

- Predicting absolute actions instead of relative actions with the decoder decreases the low-level policy performance. We believe that using relative actions might be easier for the agent to learn, as it does not have to memorize all the locations where an interaction has been performed.
- Decoder designed only with fully connected layers instead of a recurrent neural network lead to a slightly worse performance. We speculate that this could be due to the environment not completely following the Markov assumption.
- Using a gaussian mixture model instead of a mixture of logistics to predict the decoder actions lead to a similar performance.
- Using a pretrained ResNet18 in the imagenet dataset as visual encoder in simulation decreases performance for the low-level policy. More experiments by using the R3M pretrained model in simulation might offer different results.
- In the real world experiments, training the visual encoder from scratch for the static camera network decreases performance for the low-level policy. This can be explained due to the small amount of data present in the real-world dataset.

H Limitations

Despite the promising ability to learn diverse goal-reaching tasks even reasoning over long-horizon tasks, our method has a few aspects that warrant future research. Specifying a task to the goal-conditioned policy, requires providing a suitable goal image at test-time, which should be consistent with the current scene. An exciting direction for future work is to use natural language processing techniques to command the robot policy [120, 159]. If one wishes to sequence various tasks in the real world, an open question we did not address in this work is tracking task progress, in order to know when to move to the next task. In this work we acted with a fixed time-horizon for sequencing tasks in the real world, but this implicitly assumes that all tasks take approximately the same timesteps to complete. In addition, recent results [212] show that, in general, pure offline RL methods tend to offer better data-efficiency compared to behavioral cloning, which could be counted as one general limitation of imitation learning methods and derivatives of it, as the approach introduced in this work. However, since one of the most appealing properties of play lies in the simplicity of data collection, this limitation appears to be rather minor. Our method has a specific limitation in that we must first train the low-level policy before training the high-level policy, which requires more training time than training them together. However, because we train our approach offline, the robot does not need to perform rollouts in the environment during this time, making this a minor issue.

Chapter 9

Affordance Learning from Play for Sample-Efficient Policy Learning

The content of this chapter has been published in [163]:

> J. Borja-Diaz, O. Mees, G. Kalweit, L. Hermann, J. Boedecker, and W. Burgard
> **Affordance Learning from Play for Sample-Efficient Policy Learning**
> *IEEE International Conference on Robotics and Automation (ICRA)*, 2022

Jessica Borja-Diaz and I share the main authorship. I developed the idea for the paper, devised the theoretical framework and wrote most of the paper. Jessica Borja-Diaz implemented the initial experimental framework and we jointly collected the dataset and carried out the experiments for the paper. Gabriel Kalweit provided consultation with initial experiments, and contributed to the conceptualization of the idea and writing of the paper. Lukas Hermann provided consultation for the real world robot experiments and implemented the low-level control of the robot. Joschka Boedecker and Wolfram Burgard provided general consultation.

Abstract

Robots operating in human-centered environments should have the ability to understand how objects function: *what* can be done with each object, *where* this interaction may occur, and *how* the object is used to achieve a goal. To this end, we propose a novel approach that extracts a self-supervised visual affordance model from human teleoperated play data and leverages it to enable efficient policy learning and motion planning. We combine model-based planning with model-free deep reinforcement learning (RL) to learn policies that favor the same object regions favored by people, while requiring minimal robot interactions with the environment. We evaluate our algorithm, Visual Affordance-guided Policy Optimization (VAPO), with both diverse simulation manipulation tasks and real world robot tidy-up experiments to demonstrate the effectiveness of our affordance-guided policies. We find that our policies train $4\times$ faster than the baselines and generalize better to novel objects because our visual affordance model can anticipate their affordance regions. Code and trained models available at `http://vapo.cs.uni-freiburg.de`.

9.1 Introduction

Humans have the ability to effortlessly recognize and infer functionalities of objects despite their large variation in appearance and shape. For example, we understand that we need to pull the handle of a drawer to open it or grasp a knife by the handle to use it. This capacity to focus on the most relevant behaviors in a given situation enables efficient decision making by limiting the choices of action that are even considered. Gibson's theory of affordances [249] provides a way to reason about object function. It suggests that objects have action possibilities, e.g., a mug is "graspable" and a door is "openable" and has been extensively studied in both the robotics and the computer vision communities [250].

However, the abstract notion of "what actions are possible?" addressed by current affordance learning methods is limited. A robot needs to know *where* are actionable regions in an environment, the specific points on the object that need to be manipulated for a successful interaction, *what* it can achieve with it and *how* the object is used to achieve a goal. Current affordance learning methods have two major problems. First, they are limited by requiring heavy supervision in the form of manually annotated segmentation masks [251, 252, 253, 254] or expensive interactive exploration [255, 256], restricting their scalability and applicability in practical robotics scenarios. Second, current affordance-augmented robotic systems are limited in the complexity of the actions they model by relying often on predefined action templates [255, 256, 257, 258]. Together, these limitations naturally restrict the scope of affordance learning systems to a narrow set of objects and robotics applications.

In light of these issues, we propose a method for sample-efficient policy learning of complex manipulation tasks that is guided by a self-supervised visual affordance model. Therefore, we call our algorithm Visual Affordance-guided Policy Optimization (VAPO). Towards overcoming the issues of expensive manual supervision and exploration, we propose to learn affordances that are *grounded* in real human behavior from teleoperated play data [18]. Play data is not random, but rather structured by human knowledge of object affordances (e.g., if people see a drawer in a scene, they tend to open it). Moreover, affordances discovered

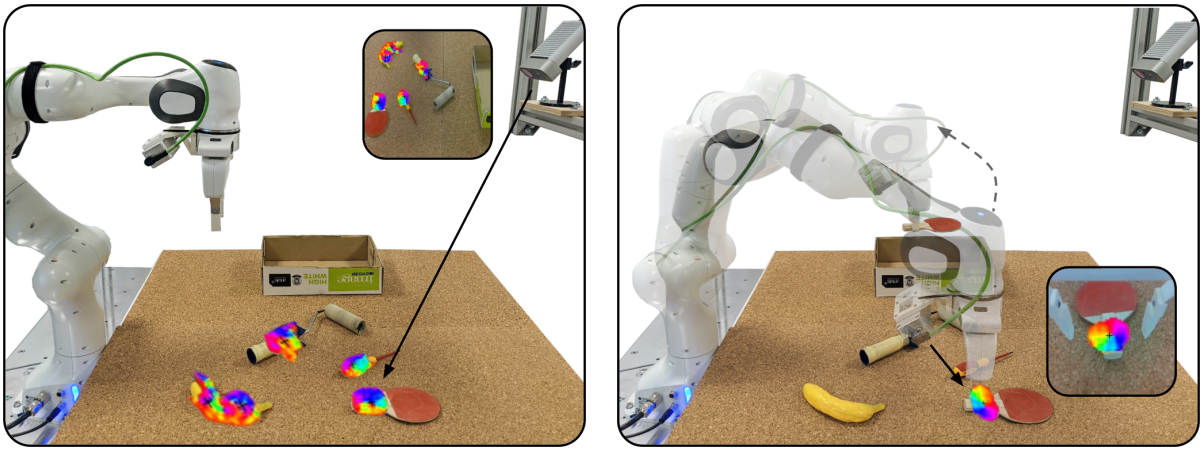


Figure 9.1: Real world setup for a tidy up task: our self-supervised visual affordance model guides the robot to the vicinity of actionable regions in the environment with a model-based policy. Once inside this area, we switch to a local reinforcement learning policy, in which we embed our affordance model to favor the same object regions favored by people and to boost sample-efficiency.

from unlabeled play are *functional affordances*, priming a robot to approach an object the way a human would. On the other hand, teleoperated play data does not bear the risk of the correspondence problem as opposed to recordings directly from human demonstrations. We hence leverage this visual affordance model to guide a robot to perform complex manipulation tasks. Aside from accelerating learning, a critical advantage of imbuing robots with an object-centric visual affordance prior is generalization: the learned policy generalizes to unseen object instances because our visual affordance model can anticipate their affordance regions.

Our approach decomposes object manipulation into a sample-efficient combination of model-based planning and model-free reinforcement learning, inspired by a recent line of work that aims to combine classical motion planning with machine learning [259, 260, 261]. Concretely, we first predict object affordances and drive the end-effector from free-space to the vicinity of the afforded region with a model-based method. Once inside this area, the model cannot be trusted and we switch to a RL policy in which the agent is rewarded for interacting with the afforded regions. This way, the local policy has a "human prior" for how to approach an object, but is free to discover its exact grasping strategy. Our self-supervised visual affordance model is leveraged twice to boost sample-efficiency: 1) driving the model-based planner to the vicinity of afforded regions, 2) guiding a local grasping RL policy to favor the same object regions favored by people. Standard model-free RL faces a number of challenges, since the policy must solve two problems: representation learning and task learning from high-dimensional raw observations in a single end-to-end training procedure. As in practice solving *both* problems together is difficult, embedding our visual affordance model within a reinforcement learning loop alleviates the representation learning challenge. The interplay between model-based and model-free policies allows for a sample-efficient division of the robot control learning, without assuming a predefined set of manipulation primitives, 3D object shapes or a tracking system.

9.2 Related Work

Predicting Semantic Representations To successfully interact with a 3D object, a robot must be able to "understand" it given some perceptual observation. There exists a large body of work in the computer vision community targeting such an understanding in the form of different semantic labels. For example, predicting category labels [262], or more fine-grained output such as semantic keypoints [263], part segmentations [264] or afforded spatial relations [88] can arguably yield more actionable representations e.g. allowing one to infer where "handles" are. However, merely obtaining such semantic labels is clearly not sufficient on its own, a robot must also understand *what* needs to be done and *how* the object is used to achieve a goal.

Acting with Model-based Planning Towards obtaining useful information for how to act, some methods aim for representations that can be leveraged by classical robotics techniques. In particular, traditional analytical approaches use knowledge of the 3D object pose [265, 266], shape [107, 262], gripper configuration, friction coefficients, etc. to determine optimal action trajectories. However, model-based methods rely on an accurate model of the environment and they normally do not handle perception errors and physical interactions naturally [120], limiting their reliability. Our approach uses model-based planning to guide the robot to the vicinity of detected affordance regions and switches then to a local RL policy.

Reinforcement Learning Grasping RL models offer a counterpoint to the planning paradigm. Instead of breaking the task into two steps, static grasp synthesis followed by motion planning, it can operate directly from raw sensory inputs in closed-loop feedback control, which are not subject to estimation errors [8, 72]. Unlike model-based methods, RL methods do not require a detailed description of the environment and the task, but rather require access to interaction with the environment and to a reward function. Such binary rewards are easy to describe, but unfortunately they render RL methods extremely sample-inefficient and brittle. Although there have been promising advances in learning data-driven reward functions [133, 176, 186], for most complex problems of interest, learning RL policies from scratch remains intractable. In contrast, we inject an object-centric visual affordance prior extracted from human teleoperated play data to boost sample efficiency.

Visual Affordances Most closely related to our approach is the line of work where visual affordances are learned for object manipulation [253, 254, 267, 268, 269]. Traditionally, visual affordance learning methods are limited by their requirement of manually drawn segmentation masks or keypoints [251, 252, 253, 254] and some leverage additional sensing, such as force gloves [270]. Recently, there has been a shift to explore other forms of supervision such as videos [271], a robot's gripper grasp success/failure [258, 267] or thermal image contact data [269]. In contrast, we leverage a self-supervised signal of a robot's gripper opening and closing during human teleoperation to learn image-based functional affordances.

9.3 Approach

The main incentive of our method is to learn sample-efficient policies of complex manipulation tasks that are guided by a self-supervised visual affordance model. Our approach consists of three steps. First, we train a network to discover and learn object affordances in unlabeled play data (Sec. A). Second, we divide the space into regions where a model-based policy is reliable and regions where it may have limitations handling perception errors or

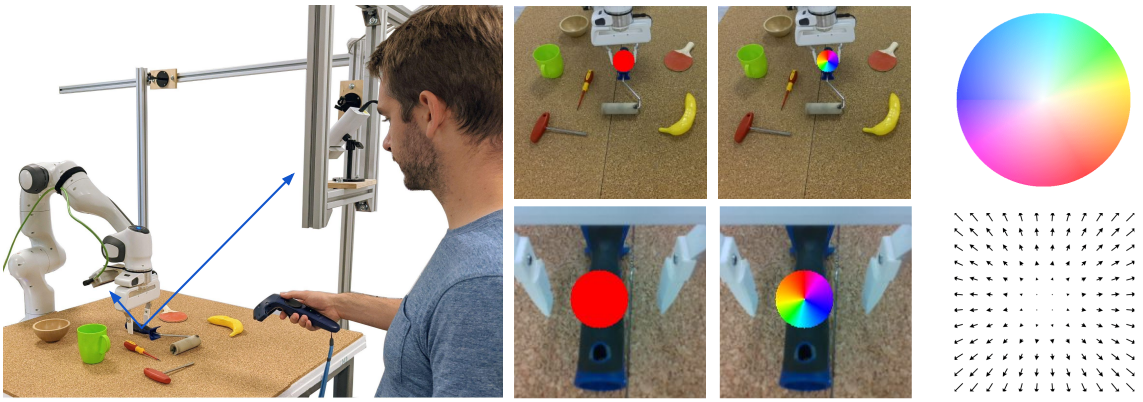


Figure 9.2: Visualization of our self-supervised object affordance labelling. We leverage a self-supervised signal of a robot's gripper opening and closing during human teleoperation to project the 3D tool-center-point into the static and gripper cameras. We label the neighboring pixels within a radius around the afforded region with a binary segmentation mask and direction vectors from each pixel towards the affordance region center. On the right we show the color code used to interpret the direction vectors.

physical interactions. We leverage the learned affordance model to drive the end-effector from free-space to the vicinity of the afforded region with a model-based policy π_{mod} (Sec. B). Third, once inside this area we switch to a local reinforcement learning policy π_{rl}, in which we embed our affordance model to favor the same object regions favored by people and to boost sample-efficiency (Sec. C). Thus, our final policy is defined as a mixture:

$$\pi(a|s) = (1 - \alpha(s)) \cdot \pi_{mod}(a|s) + \alpha(s) \cdot \pi_{rl}(a|s), \tag{9.1}$$

where $\alpha(s) \in [0, 1]$. We use an estimate of the normalized distance between the robot's gripper and the affordance region $\alpha(s)$ to switch between the policies. An overview of the system is given in Figure 9.1.

A Learning Visual Affordances from Play

Our key insight is to learn about object interactions from play data by leveraging a self-supervised signal of a robot's gripper opening and closing during human teleoperation, as shown in Figure 9.2. In this way, without explicit manual segmentation labels, we learn to anticipate not only *where* are regions that afford human-object interactions, but also a powerful prior on *how* humans approach those objects. The only assumption our method makes is an existing robot-camera calibration. We decouple the affordance prediction task into different components.

First, the affordance model $\mathcal{F}_a$ learns to transform an image I into a binary segmentation map $A \in \mathbb{R}^{H \times W}$, indicating regions that afford an interaction. Second, it estimates 2D pixel coordinates of the affordance region centers by predicting a vector from each affordance pixel towards the center. Estimating the center points of the afforded regions is a key component in order to disambiguate affordances from multiple objects in a scene. Clearly, play data showing people naturally interacting with objects partially reveals the afforded regions in an environment. Thus, in order to discover affordances in unlabeled data the gripper action is used as a heuristic to detect human-object interactions.

Intuitively, if the gripper closes during play, it is indicative of a possible interaction that will start at that position. Thus, we can project the gripper's 3D point p^t_{grip} to a camera image pixel u^t_{grip} and label the pixels within a radius r for the past n frames as an afforded region. Similarly, if the gripper transitions from being closed to open, it means that an interaction with an object ended at the 3D position p_i. This allows us to discover a set of interaction points throughout time $P^k = (p_1, p_2, ..., p_k)$, which represent the world coordinates of where interactions have occurred until timestep k. To get the full set of interaction locations for a timestep t we consider the 3D positions from where a grasp will occur and where an interaction has been previously occurred until t. Finally, each 3D point is projected to a camera image pixel to create the affordance mask label by marking neighboring pixels. The pixel coordinates of the projected points are used as the affordance region centers.

In order to disambiguate affordances from multiple objects in a scene, we let the network estimate 2D pixel coordinates of the affordance region centers by predicting a vector from each affordance pixel towards the center $V \in \mathbb{R}^{H \times W \times 2}$. We construct these labels by calculating the displacement from each pixel of the affordance mask to the corresponding projected center. The background pixels are pointed towards a fixed position to avoid false positives.

One limitation of the proposed heuristic is that it assumes users interacting with the environment during play will only close the gripper to perform meaningful interactions. To avoid erroneous labeling due to closing/opening the gripper in free-space without object-interaction, we introduce an additional check by requiring the griper-width to stay for Δt timesteps in a range between opened and closed.

To train the full affordance model F_a we apply two different loss functions. For the affordance segementation A loss , we use a weighted sum between a cross entropy ℓ_{ce} and a dice loss ℓ_{dice} to account for class imbalance. Similar to Xie *et al.* [272], for the direction prediction we optimize a weighted cosine similarity loss given by:

$$\ell_{dir} = \sum_{i \in \mathcal{O}} \alpha_i (1 - V_i^{\mathbf{T}} \bar{V}_i) + \frac{\lambda_b}{\mid \mathcal{B} \mid} \sum_{i \in \mathcal{B}} \left(1 - V_i^{\mathbf{T}} \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right)$$

Where V_i, $\bar{V}_i$ are the predicted and ground truth unit directions of pixel i respectively. $\mathcal{B}, \mathcal{O}$ are the sets of pixels belonging to the background and affordance region classes. The total loss for the affordance model is given by $w_{ce}\ell_{ce} + w_{dice}\ell_{dice} + w_{dir}\ell_{dir}$.

B From Model-Based to Reinforcement Learning Workspace

Classical motion planning algorithms have difficulty in the presence of stochastic dynamics and high-dimensional systems. RL methods on the other hand offers a promising solution for its ability to learn general policies that can handle complex interactions and high-dimensional observations. However, for most complex problems of interest, learning RL policies from scratch remains intractable. Inspired by recent works that aim to combine both type of controllers [259, 260, 261], we divide the space into regions where a model-based policy is reliable and regions where it may have limitations handling perception errors or physical interactions.

Concretely, we predict affordances and the corresponding region centers using a static camera image. Given this information of *where* are regions that afford human-object interactions, we localize a chosen pixel region center in 3D and drive the end-effector from free-space to

the vicinity of the afforded region with a model-based policy π_{mod} and hand control over to the model-free policy π_{rl}. We use an estimate of the distance between the robot's gripper and the predicted affordance region center to switch between the policies. Restricting the area where the RL policy is active to the vicinity of regions that afford human-object interactions has the advantage that it makes it more sample-efficient. Besides, this division of labour allows to learn local RL policies by switching to a gripper camera, improving generalization across different locations.

C Affordance-guided Reinforcement Learning Grasping

Once the model-based policy π_{mod} has brought the end-effector to the vicinity of a region that affords human-object interactions, we switch to a local gripper-camera based RL policy which we augment with an object-centric visual affordance prior to boost sample efficiency.

Problem Formulation: We consider the standard Markov decision process (MDP) $M = (\mathcal{S}, \mathcal{A}, \mathcal{T}, r, \mu_0, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state space and action space respectively. $\mathcal{T}(s'|s, a)$ is the probability of transitioning from state s to state s' when applying action a. The actions are drawn from a probability distribution over actions $\pi(a|s)$ referred to as the agent's policy. $r(s, a)$ is the reward received by an agent for executing action a in state s, μ_0 the initial state distribution, and $\gamma \in (0, 1)$ the discount factor prioritizing long- versus short-term reward. The goal in RL is to optimize a policy $\pi(a|s)$ that maximizes the expected discounted return $\mathbb{E}_{\pi,\mu_0,\mathcal{T}}\left[\sum_{t=0}^{\infty} \gamma^t r(s, a)\right]$.

Observation Space: The observation space is composed of two parts: 1) the proprioceptive state including the 3D world coordinates of the end-effector, the orientation Euler angles and the gripper width. 2) The visual inputs consisting of the current RGB-D image observed by the gripper camera and the binary affordance mask predicted from the corresponding affordance model.

Action Space: We use a 7-DOF Franka Emika Panda robot with a parallel gripper both in simulation and in the real world. The action space consists of delta XYZ position, delta Euler angles and the binary gripper action.

Reward: The reward function should not only signal a successful object interaction, but also guide the exploration process to focus on actionable object regions. To realize this, we leverage the visual affordance model to guide the agent to get close to the affordance centers. This way, the local policy has a "human prior" for how to approach an object, but is free to discover its exact grasping strategy. Given the detected affordance center and the fact that the RL policy only acts locally within a neighborhood, we normalize the euclidean distance between the end-effector and the affordance center to create a positive reward R_{aff} which increases as we get closer to the detected center. Additionally if the agent goes outside the neighborhood, it receives a negative reward R_{out} and if it successfully manipulates an object it receives a positive reward of R_{succ}. Our total reward function is:

$$r(s, a) = \lambda_1 R_{succ} + \lambda_2 R_{aff} + \lambda_3 R_{out} \tag{9.2}$$

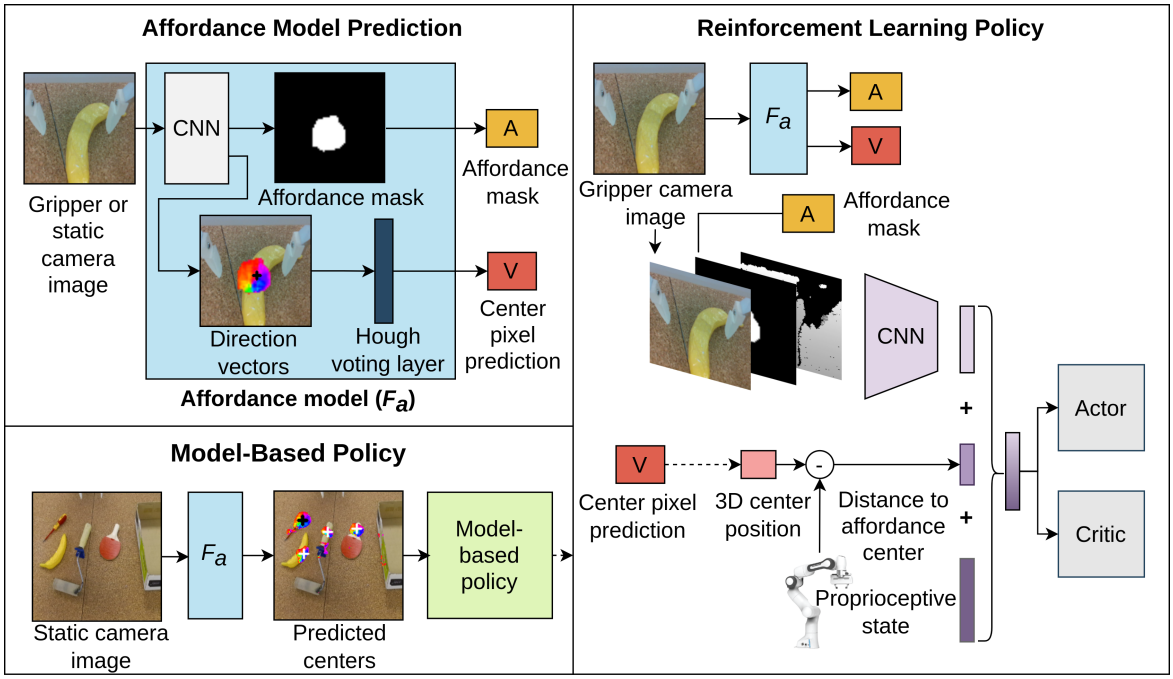


Figure 9.3: Overview of the full approach. The affordance model takes an image from either camera as input to predict object affordance masks and center pixel predictions (top left). The static camera affordances are used to select a position that the model-based policy will move towards (bottom left). We then switch to a RL policy which takes as input the the predictions of the gripper camera affordance, the robot's proprioception, the distance to the predicted center, and the current RGB-D image (right).

9.4 Implementation Details

Teleoperated play data During the unscripted teleoperated interactions we record images from two cameras: a static camera that captures the global scene, and a camera mounted on the robot's gripper. The static camera image has a resolution of 200×200 and the gripper camera uses a resolution of 64×64. We label the images of the static camera with a radius of $r = 10$ pixels around the projected center and the the gripper camera images with $r = 25$.

Affordance model We use a U-net [273] architecture followed by two parallel branches of convolutional layers that produce the affordance mask and center directions. Similar to Xiang *et al.* [274], we use a Hough voting layer to predict the 2D object centers during inference. The Hough voting layer takes the affordance mask and the direction vectors as input to compute a score for each pixel, indicating its likelihood of being an affordance region center. The location with the maximum score is selected as the object center.

We define a two-stage affordance detection by training separate models for the two cameras as shown in Figure 9.3. One model is trained with images from a static camera and predicts a spatial interaction hotspots map, indicating actionable regions. Similarly, we train an affordance model with images from a gripper camera, which gives a finer-grained spatial interaction map about where humans tend to interact with each object.

The affordance model should give insight into which parts of an object are relevant for its use. As this is dependent on the shape of the objects rather than the color, we would like the affordance model to be invariant to different colors. For this reason, the images are

(a) (b)

Figure 9.4: Objects used in simulation grasping experiments. (a) Seen objects during affordance model training. (b) Unseen objects during affordance model training. The objects propose different challenges as some are small or must be grasped in a specific manner, e.g. grasping the frying pan requires to use the handle to be successfully lifted.

converted to grayscale before being fed to the networks. Both affordance models are trained with stochastic gradient descent with a learning rate of 1e-5 and a batch size of 256. The loss weights are set to $w_{ce} = 1$, $w_{dice} = 5$, $w_{dir} = 2.5$.

Affordance-guided Reinforcement Learning

We train the policy using Soft Actor-Critic [69]. We concatenate the RGB-D images with the inferred affordance mask and pass it through a convolutional neural network (CNN) as depicted in Figure 9.3. The CNN is composed by three convolutional layers with kernel size [8,4,3] respectively and one linear layer to obtain a feature representation of size 16. Then we concatenate the obtained representation to the robot state and the distance to the affordance center. Finally this is passed through four fully connected layers. The critic and actor are implemented following the same architecture without weight-sharing. For the simulation experiments, we train a single policy for all the objects with an episode length of 100 steps during 400K episode steps. This amounts to 30hrs of learning experience. We train for 3 seed initializations. In the reward function we set $\lambda_1 = \lambda_2 = \lambda_3 = 1$ and the rewards $R_{succ} = 200$, $R_{out} = -1$.

9.5 Experimental Results

In this section we seek to answer the following questions: how does our method compare to the baseline policies in terms of sample efficiency and task completion? And, is the proposed approach applicable to a real world tidy-up task?

A Experimental Setup

We evaluate our method with both diverse simulation manipulation tasks and real world robot tidy-up experiments.

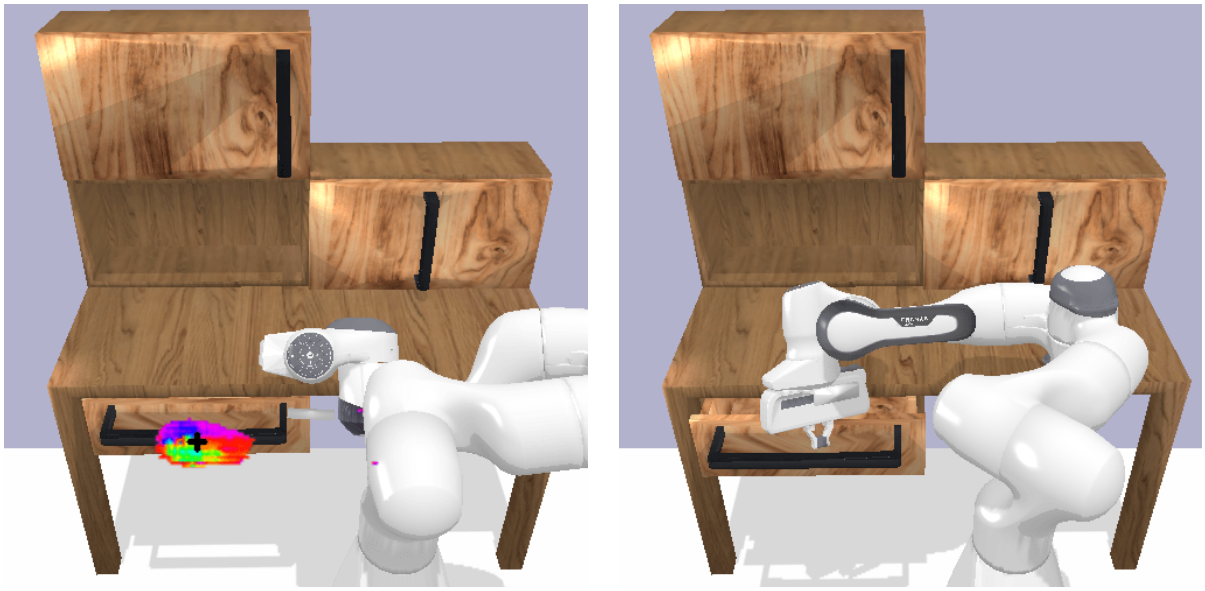


Figure 9.5: Drawer opening task. On the left, the detected affordance and the corresponding center are shown. On the right we show a rollout of the RL agent opening the drawer.

Simulation

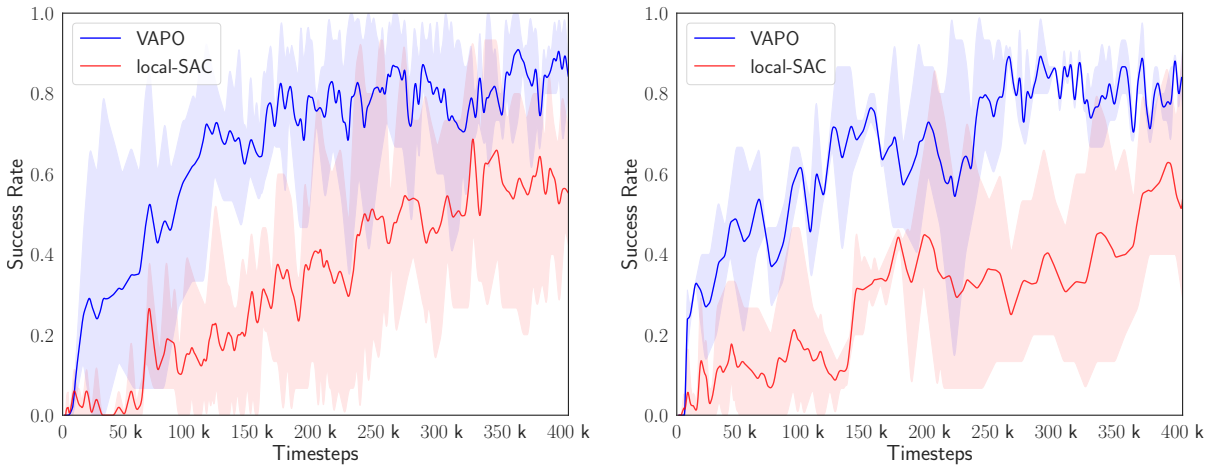
We evaluate two tasks in simulation: a grasping task and a drawer opening task. The grasping task consists on lifting different objects in a PyBullet simulated environment. The policy is trained over 15 different objects with varying degrees of complexity, such as hammers, knifes and power drills, as shown in Figure 9.4. After the policy executes a close-gripper action, the gripper attempts to lift the object and waits in the air for two seconds. If the object is still in the gripper at the end of this time, we define the grasp as being successful.

VAPO is not exclusive to a grasping task. To show this, we train a policy to open a drawer as shown in Figure 9.5. Every episode consists of the drawer on a closed position and the robot in a neutral position. The episode is deemed successful if the robot opens the drawer at least 15cm.

To train the affordance models we teleoperate the robot using a virtual reality (VR) controller to collect unscripted play data. We gather two hours of human interaction which amounts to $\sim$100K images for each environment to train the static camera and gripper camera affordance models.

Real world

For the real world experiment, we setup the environment using a 7-DOF Franka Emika Panda robot. The full setup can be seen in Figure 9.1. Similar as in simulation, we collect play data by teleoperating the robot using a VR controller as shown in Figure 9.2. We accumulate 1.5 hours of human interaction, which results in $\sim$70K images and use this to train both the gripper camera and static camera affordance models. The labels for both simulation and real world experiments are obtained as described in Section A. We only use the data to train the affordance models and do not need human annotation.



(a) (b)

Figure 9.6: VAPO vs. local-SAC for the pick up tasks. (a) Policy trained on seen objects by the affordance model. (b) Policy trained on **unseen** objects by the affordance model. In both experiments, our method learns $4\times$ faster as compared to the baseline and successfully lifts most of the objects.

B Evaluation Protocol

To test the sample efficiency of the affordance-guided RL policy, we compare against a sparse-reward SAC agent, *local-SAC*. For this baseline, we remove R_{aff} from the reward function and we modify the observation by removing the affordance mask and distance to the center. This policy still uses π_{mod} to move through free space, but does not use the affordances for interaction. In essence, it is a sparse-reward SAC agent operating with the RGB-D images of the gripper camera in the vicinity of the objects. For all the experiments we show the success rate as the average success over a given number of attempts to complete a task.

C Simulation Experiments

We start of by training policies to lift a diverse set of 15 objects on which the affordance model was also trained on. We observe that our approach outperforms the baseline and lifts significantly more objects as it has a strong prior on how objects should be interacted with. We observe that VAPO successfully can grasp objects at the anticipated afforded regions (handle of a pan, power-drill, knife), while the baseline fails to grasp objects of complex (frying pan) or ambiguous (bowl) geometries. This shows the effectiveness of the affordance-guided policy in learning stable functional grasps. Not only does our method learn better, but it is critically more sample-efficient. After $\sim$30 hours (400k timesteps) of robot interaction the baseline reaches a success rate of 0.6, while VAPO matches this performance at 100k steps. This indicates that our method learns up to $4\times$ faster than the baseline. After training for 400k timesteps, VAPO remains stable at an overall success rate of 0.90.

Next we push our affordance model to generalize to unseen objects in two sets of experiments. In the first setting, we train and test the policies on 15 objects which were not seen by the affordance model during training. We observe in Figure 9.6 that VAPO outperforms the baseline by a large margin in terms of both number of objects lifted and sample-efficiency. In

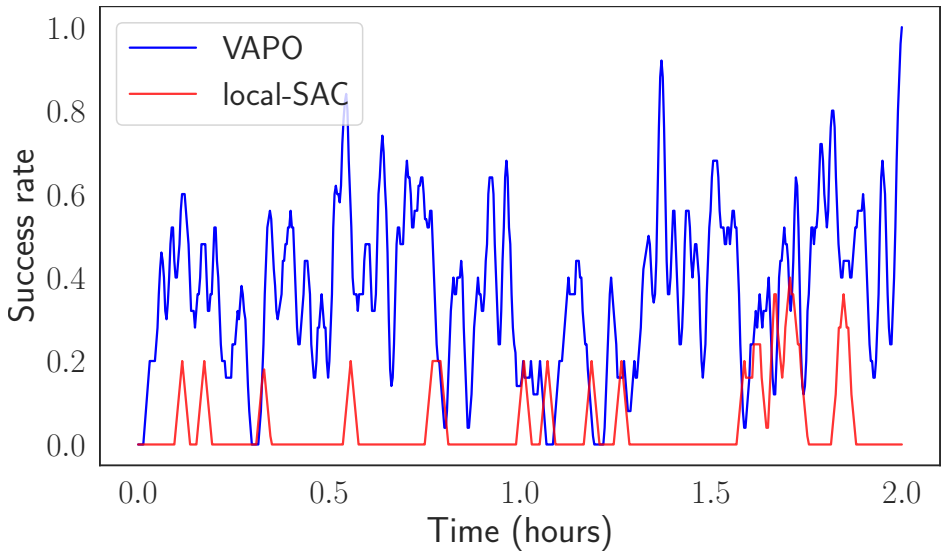


Figure 9.7: VAPO vs. local-SAC in real world tidy-up experiments. The success rate over the last ten episodes is shown. After two hours of real world robot interaction, the baseline rarely lifts any objects, while our approach consistently "functionally" grasps all the objects.

the second setting, we evaluate the trained policies zero-shot on lifting 15 unseen objects. This form of zero-shot evaluation is very challenging, as the objects are unseen for both the affordance model and the RL agent. We report a lifting success of 13/15 for VAPO and 8/15 for the baseline. This demonstrates the effectiveness of imbuing robots with an object-centric visual affordance. Aside from accelerating learning, the visual affordance model generalizes sufficiently to new object shapes and can anticipate their affordance regions, providing a useful object-centric prior.

To analyze if our approach is applicable for more tasks, we conduct experiments on a drawer opening task (Figure 9.5). We report a success rate over 100 episodes of 0.84 for VAPO and of 0.52 for the baseline. The results are consistent with the previous experiments showing that our method outperforms the baselines, while being more sample-efficient.

D Real World Experiments

We finally evaluate our approach on a real world tidy-up experiment. We show the learning curves for this experiment in Figure 9.7. We use a 7-DOF Franka Emika Panda robot and run our policy at 20 Hz. We train all methods to pickup four objects: a plastic banana, a screwdriver, a table tennis racket and a paint roller. After two hours of training VAPO is able to consistently "functionally" grasps all the objects, e.g., grasping the objects by the handles, while the SAC baseline rarely achieves to lift any object, despite the agent starting at the same robot pose as our method. This is due to the low number of samples that sparse-reward SAC is trained on, since most success stories of RL in the real world require several orders of magnitude more data [72]. Overall, our results demonstrate the effectiveness of our approach to learn sample-efficient policies by leveraging self-supervised visual affordances.

9.6 Conclusion

In this paper, we introduced the novel approach VAPO (Visual Affordance-guided Policy Optimization) as a method for sample-efficient policy learning of manipulation tasks that is guided by a self-supervised visual affordance model. The key advantage of our formulation is the extraction of visual affordances from unlabeled human teleoperated play data to learn a strong prior about *where* actionable regions in an environment are. We distill this knowledge into an interplay between model-based and model-free policies that allows for a sample-efficient division of the robot control learning, without assuming a predefined set of manipulation primitives, 3D object shapes or a tracking system. Our results show that aside from accelerating learning, a critical advantage of imbuing robots with an object-centric visual affordance prior is the ability of policies to generalize to unseen, functionally similar, objects. To the best of our knowledge, this work is the first one to demonstrate the effectiveness of visual affordances to guide model-based policies and closed-loop RL policies to learn robot manipulation tasks in the real world.

Chapter 10

Grounding Language with Visual Affordances over Unstructured Data

The content of this chapter has been published in [275]:

> O. Mees, J. Borja-Diaz and W. Burgard
> **Grounding Language with Visual Affordances over Unstructured Data**
> *IIEEE International Conference on Robotics and Automation (ICRA)*, 2023.

Jessica Borja-Diaz and I share the main authorship. I developed the main idea for the paper, devised the theoretical framework and wrote most of the paper. Jessica Borja-Diaz implemented the initial experimental framework for learning visuo-lingual affordances and we jointly collected the dataset. I contributed the implementations for translating abstract language input into sequences of subgoals with Large Language Models, learning the language-conditioned low-level level policies with latent skills, the framework to collect data via teleoperation and perform experiments in both simulation and in the real world. Jessica additionally contributed the experiments in the CALVIN environment. All the real world experiments reported in this thesis were entirely carried out by the author of this thesis.

Abstract

Recent works have shown that Large Language Models (LLMs) can be applied to ground natural language to a wide variety of robot skills. However, in practice, learning multi-task, language-conditioned robotic skills typically requires large-scale data collection and frequent human intervention to reset the environment or help correcting the current policies. In this work, we propose a novel approach to efficiently learn general-purpose language-conditioned robot skills from unstructured, offline and reset-free data in the real world by exploiting a self-supervised visuo-lingual affordance model, which requires annotating as little as 1% of the total data with language. We evaluate our method in extensive experiments both in simulated and real-world robotic tasks, achieving state-of-the-art performance on the challenging CALVIN benchmark and learning over 25 distinct visuomotor manipulation tasks with a single policy in the real world. We find that when paired with LLMs to break down abstract natural language instructions into subgoals via few-shot prompting, our method is capable of completing long-horizon, multi-tier tasks in the real world, while requiring an order of magnitude less data than previous approaches. Code and videos are available at http://hulc2.cs.uni-freiburg.de.

10.1 Introduction

Recent advances in large-scale language modeling have produced promising results in bridging their semantic knowledge of the world to robot instruction following and planning [40, 276, 277]. In reality, planning with Large Language Models (LLMs) requires having a large set of diverse low-level behaviors that can be seamlessly combined together to intelligently act in the world. Learning such sensorimotor skills and grounding them in language typically requires either a massive large-scale data collection effort [40, 41, 42, 276] with frequent human interventions, limiting the skills to templated pick-and-place operations [30, 278] or deploying the policies in simpler simulated environments [141, 145, 159]. The phenomenon that the apparently easy tasks for humans, such as pouring water into a cup, are difficult to teach a robot to do, is also known as Moravec's paradox [43]. This raises the question: how can we learn a diverse repertoire of visuo-motor skills in the real world in a scalable and data-efficient manner for instruction following?

Prior studies show that decomposing robot manipulation into semantic and spatial pathways [30, 163, 257], improves generalization, data-efficiency, and understanding of multimodal information. Inspired by these pathway architectures, we propose a novel, sample-efficient method for learning general-purpose language-conditioned robot skills from unstructured, offline and reset-free data in the real world by exploiting a self-supervised visuo-lingual affordance model. Our key observation is that instead of scaling the data collection to learn how to reach any reachable goal state from any current state [52] with a single end-to-end model, we can decompose the goal-reaching problem hierarchically with a high-level stream that grounds semantic concepts and a low-level stream that grounds 3D spatial interaction knowledge, as seen in Figure 6.1.

Specifically, we present Composing Actions from Language and Vision (CALVIN), a hierarchical language-conditioned agent that integrates the task-agnostic control of HULC [159]

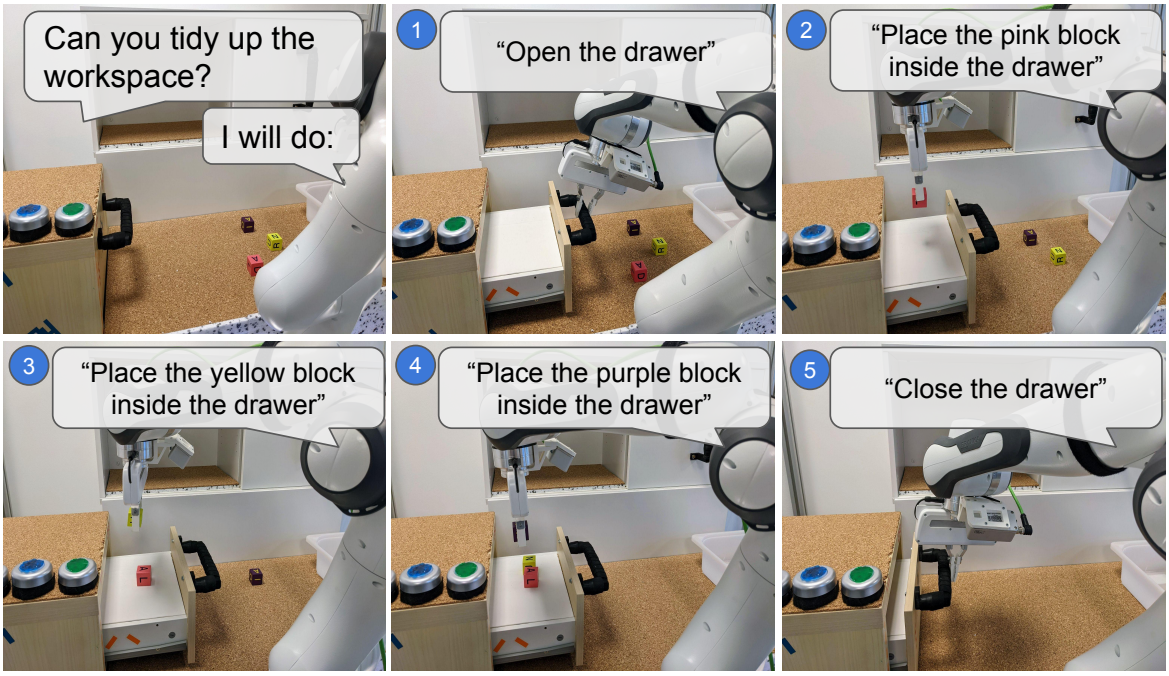


Figure 10.1: When paired with Large Language Models, CALVIN enables completing long-horizon, multi-tier tasks from abstract natural language instructions in the real world, such as “tidy up the workspace” with no additional training. We leverage a visual affordance model to guide the robot to the vicinity of actionable regions referred by language. Once inside this area, we switch to a single 7-DoF language-conditioned visuomotor policy, trained from offline, unstructured data.

with the object-centric semantic understanding of VAPO [163]. HULC is a state-of-the-art language-conditioned imitation learning agent that learns 7-DoF goal-reaching policies end-to-end. However, in order to jointly learn language, vision, and control, it needs a large amount of robot interaction data, similar to other end-to-end agents [41, 145, 160]. VAPO extracts a self-supervised visual affordance model of unstructured data and not only accelerates learning, but was also shown to boost generalization of downstream control policies. We show that by extending VAPO to learn *language-conditioned affordances* and combining it with a 7-DoF low-level policy that builds upon HULC, our method is capable of following multiple long-horizon manipulation tasks in a row, directly from images, while requiring an order of magnitude less data than previous approaches. Unlike prior work, which relies on costly expert demonstrations and fully annotated datasets to learn language-conditioned agents in the real world, our approach leverages a more scalable data collection scheme: unstructured, reset-free and possibly suboptimal, teleoperated *play* data [18]. Moreover, our approach requires annotating as little as 1% of the total data with language. Extensive experiments show that when paired with LLMs that translate abstract natural language instructions into a sequence of subgoals, CALVIN enables completing long-horizon, multi-stage natural language instructions in the real world. Finally, we show that our model sets a new state of the art on the challenging CALVIN benchmark [141], on following multiple long-horizon manipulation tasks in a row with 7-DoF control, from high-dimensional perceptual observations, and specified via natural language. To our knowledge, our method is the first explicitly aiming to solve language-conditioned long-horizon, multi-tier tasks from purely offline,

reset-free and unstructured data in the real world, while requiring as little as 1% of language annotations.

10.2 Related Work

There has been a growing interest in the robotics community to build language-driven robot systems [24], spurred by the advancements in grounding language and vision [4, 7]. Earlier works focused on localizing objects mentioned in referring expressions [104, 108, 109, 148, 149] and following pick-and-place instructions with predefined motion primitives [30, 120, 151]. More recently, end-to-end learning has been used to study the challenging problem of fusing perception, language and control [40, 41, 42, 145, 146, 159, 160, 279]. End-to-end learning from pixels is an attractive choice for modeling general-purpose agents due to its flexibility, as it makes the least assumptions about objects and tasks. However, such pixel-to-action models often have a poor sample efficiency. In the area of robot manipulation, the two extremes of the spectrum are CLIPort [30] on the one hand, and agents like GATO [42] and BC-Z [41] on the other, which range from needing a few hundred expert demonstrations for pick-and-placing objects with motion planning, to several months of data collection of expert demonstrations to learn visuomotor manipulation skills for continuous control. In contrast, we lift the requirement of collecting expert demonstrations and the corresponding need for manually resetting the scene, to learn from unstructured, reset-free, teleoperated *play* data [18]. Another orthogonal line of work tackles data inefficiency by using pre-trained image representations [30, 247, 280] to bootstrap downstream task learning, which we also leverage in this work.

We propose a novel hierarchical approach that combines the strengths of both paradigms to learn language-conditioned, task-agnostic, long-horizon policies from high-dimensional camera observations. Inspired by the line of work that decomposes robot manipulation into semantic and spatial pathways [30, 163, 257], we propose leveraging a self-supervised affordance model from unstructured data that guides the robot to the vicinity of actionable regions referred in language instructions. Once inside this area, we switch to a single multi-task 7-DoF language-conditioned visuomotor policy, trained also from offline, unstructured data.

10.3 Method

We decompose our approach into three main steps. First we train a language-conditioned affordance model from unstructured, teleoperated data to predict 3D locations of an object that affords an input language instruction (Section A). Second, we leverage model-based planning to move towards the predicted location and switch to a local language-conditioned, learning-based policy π_{free} to interact with the scene (Section C). Third, we show how CALVIN can be used together with large language models (LLMs) for decomposing abstract language instructions into a sequence of feasible, executable subtasks (Section D).

Formally, our final robot policy is defined as a mixture:

$$\begin{aligned} \pi(a \mid s, l) &= (1 - \alpha(s, l)) \cdot \pi_{mod}(a \mid s) \\ &\quad + \alpha(s, l) \cdot \pi_{free}(a \mid s, l) \end{aligned} \tag{10.1}$$

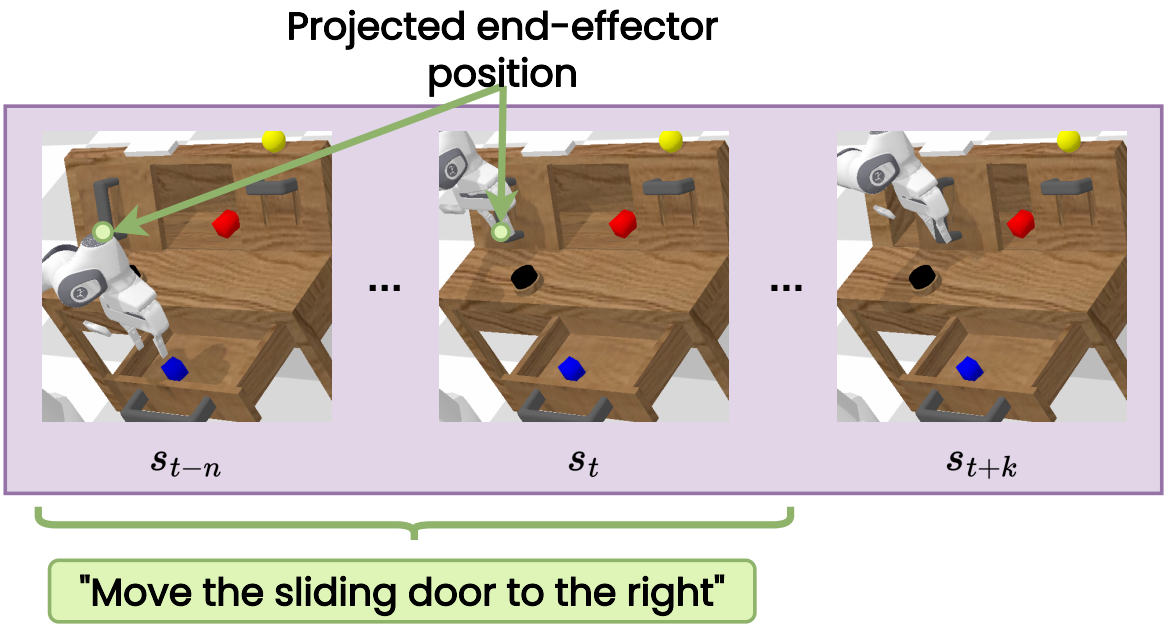


Figure 10.2: Visualization of the procedure to extract language-conditioned visual affordances from human teleoperated unstructured, free-form interaction data. We leverage the gripper open/close signal during teleoperation to project the end-effector into the camera images to detect affordances in undirected data.

Specifically, we use the pixel distance between the projected end-effector position I_{tcp} and the predicted pixel from the affordance model I_{aff} to select which policy to use. If the distance is larger than a threshold ϵ, the predicted region is far from the robots current position and we use the model-based policy π_{mod} to move to the predicted location. Otherwise, the end-effector is already near the predicted position and we keep using the learning-based policy π_{free}. Thus, we define α as:

$$\alpha(s, l) = \begin{cases} 0, & \text{if } |I_{aff} - I_{tcp}| > \epsilon \\ 1, & \text{otherwise} \end{cases} \tag{10.2}$$

As the affordance prediction is conditioned on language, each time the agent receives a new instruction, our agent decides which policy to use based on $\alpha(s, l)$. Restricting the area where the model-free policy is active to the vicinity of regions that afford human-object interactions has the advantage that it makes it more sample efficient, as it only needs to learn local behaviors.

A Extracting Human Affordances from Unstructured Data

We aim to learn an affordance model $\mathcal{F}_a$ that can predict a world location when given a natural language instruction. Unlike prior affordance learning methods that require manually drawn segmentation masks [254], we automatically extract affordances from unstructured, human teleoperated play data [18]. Leveraging play data has several advantages: it is cheap and scalable to collect, contains general behavior, and is not random, but rather structured by human knowledge of affordances. Concretely, play data consists of a long unsegmented

dataset $\mathcal{D}$ of semantically meaningful behaviors provided by users teleoperating the robot without a specific task in mind. The full state-action stream $\mathcal{D} = \{(s_t, a_t)_{t=0}^{\infty}\}$ is relabeled to treat the preceding states and actions as optimal behaviour to reach a visited state [18]. Additionally, we assume that a small number of random sequences, less than 1% of the dataset, are annotated with a language instruction describing the task being completed in the sequence.

In order to extract visual affordances from unstructured data, we use the gripper action as a heuristic to discover elements of the scene that are relevant for task completion. Consider the following scenario: a random sequence $\tau = \{(s_0, a_0), ..., (s_k, a_k)\}$, where k denotes the window size, is annotated with a language instruction $s_g = l$. If for any state s_i in the sequence, the action a_i contains a gripper closing signal, we assume that there is an object that is needed for executing the task l at the position of the end-effector. To learn a visuo-lingual affordance model, we project the end-effector world position to the camera images to obtain a pixel p_t, and we annotate the previous frames with said pixel and the language instruction l, see Figure 10.2. Intuitively, this allows the affordance model to learn to predict a pixel corresponding to an object that is needed for completing the task l.

During test time, given a predicted pixel location, assuming an existing camera calibration, depth information is needed to compute the 3D position where the model-based policy should move to. Instead of relying on the sensory depth observations, our model is trained to produce an estimated depth, by using the position of the end-effector during the gripper closing as supervision. A key advantage of our formulation is that by predicting the depth from visuo-lingual features, our model can better adapt to partial occlusions that might occur in the scene.

B Language-Conditioned Visual Affordances

Our visuo-lingual affordance model, see Figure **??**, consists of an encoder decoder architecture with two decoder heads. The first head predicts a distribution over the image, representing each pixels likelihood to be an afforded point. The second head predicts Gaussian distribution from which the corresponding predicted depth is sampled. Both heads share the same encoder and are conditioned on the input language instruction. Formally, given an input consisting of a visual observation I and a language instruction l, the affordance model $\mathcal{F}_a$ produces an output o of (1) a pixel-wise heatmap $A \in \mathbb{R}^{H \times W}$, indicating regions that afford the commanded task and (2) a corresponding depth estimate d. We denote this mapping as $\mathcal{F}_a(I, l) \mapsto o = (A, d)$.

Visual Module

The visual prediction module produces a heatmap A given an input (I_t, l_t). To train it, we apply a softmax function over all the pixels of A. This results in a distribution V over the image where the sum of all the pixel values equals to one.

$$V = \text{softmax}(A) = \frac{\exp(a_i)}{\sum_{j=1}^{N} \exp(a_j)} \tag{10.3}$$

Similarly, the target T is constructed with the same shape as V, by initializing all its values to zero. Then, we generate a binary one-hot pixel map with the pixel of the projected position

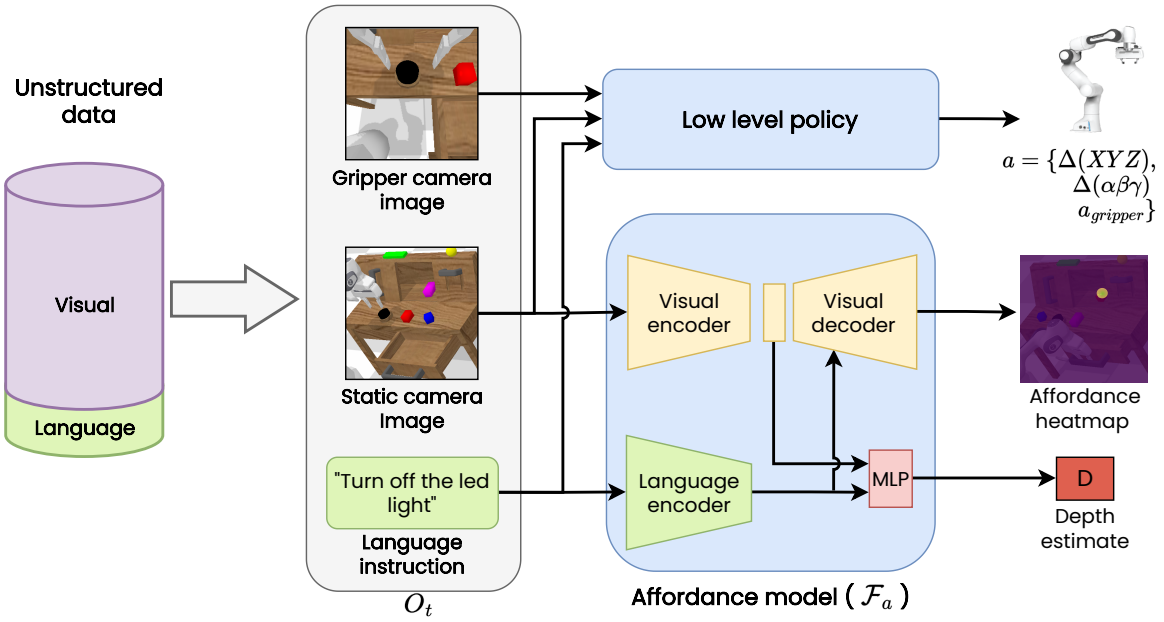


Figure 10.3: Overview of the system architecture. CALVIN first processes a language instruction and an image from a static camera to predict the afforded region and guides the robot to its vicinity. Once inside this area, we switch to a language-conditioned imitation learning agent that receives RGB observations from both a gripper and a static camera, and learns 7-DoF goal-reaching policies end-to-end. Both modules learn from the same free-form, unstructured dataset and require as little as 1% of language annotations.

that corresponds to the current state input. Finally, we optimize the visual prediction module with the cross-entropy loss:

$$\mathcal{L}_{aff} = -\sum_{i=1}^{N} t_i \log v_i, \tag{10.4}$$

where $t_i \in T$ and $v_i \in V$. This optimization scheme [281] allows the visual module to learn a multimodal belief over the image, where the pixel with the highest value denotes the most likely image location given the input. During inference, we use the dense pixelwise output prediction A to select a pixel location I_i:

$$I_i = \arg\max_{(u,v)} V((u,v) \mid (I,l)) \tag{10.5}$$

The affordance prediction follows a U-Net [273] architecture, where we repeatedly apply language-conditioning to three of the decoder layers after the bottleneck, taking inspiration from LingUNet [282].

Depth Module

As aforementioned, we can compute a target for the depth module by transforming p_t to the camera frame to obtain p_t^{cam}, where the z coordinate of this point corresponds to the ground truth depth $p_{t,z}^{cam}$. Although we compute the true value, typical depth sensors present measurement errors. Therefore, in order to design a system that models the depth error, we use

the ground truth depth information to train a Gaussian distribution $\mathcal{N}(\mu, \sigma)$ by maximizing the log likelihood.

$$\mathcal{L}_{depth} = \frac{1}{2}\left(\log \sigma^2 + \frac{(y-\mu)^2}{\sigma^2}\right) \tag{10.6}$$

As shown in Figure 10.3, the depth module consists of a set of linear layers that take as input the encoded visuo-lingual features. Here, the language-conditioning is done by concatenating the natural language encoding to the first two layers of the multilayer perceptron. The output of the network are the parameters of a Gaussian distribution $d \sim N(\mu, \sigma)$, which is sampled during inference to obtain the depth prediction d. The total loss function used to train the full affordance model is defined as a weighted combination of the affordance module and depth prediction module losses:

$$\mathcal{L} = \beta\mathcal{L}_{aff} + (1-\beta)\mathcal{L}_{depth} \tag{10.7}$$

C Low-Level Language-Conditioned Policy

In order to interact with objects, we learn a goal-conditioned policy $\pi_\theta\left(a_t \mid s_t, l\right)$ that outputs action $a_t \in \mathcal{A}$, conditioned on the current state $s_t \in \mathcal{S}$ and free-form language instruction $l \in \mathcal{L}$, under environment dynamics $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$. We note that the agent does not have access to the true state of the environment, but to visual observations. We model the low-level policy with a general-purpose goal-reaching policy based on HULC [159] and trained with multi-context imitation learning [145]. We leverage the same, long unstructured dataset $\mathcal{D}$ of semantically meaningful behaviors provided by users we previously utilized to learn affordances in Section A. In order to learn task-agnostic control, we leverage goal relabeling [57], by feeding these short horizon goal image conditioned demonstrations into a simple maximum likelihood goal conditioned imitation objective:

$$\mathcal{L}_{LfP} = \mathbb{E}_{(\tau, s_g)\sim D_{play}}\left[\sum_{t=0}^{|\tau|} \log \pi_\theta(a_t \mid s_t, s_g)\right] \tag{10.8}$$

However, when learning language-conditioned policies $\pi_\theta\left(a_t \mid s_t, l\right)$ it is not possible to relabel any visited state s to a natural language goal, as the goal space is no longer equivalent to the observation space. Lynch *et al.* [145] showed that pairing a small number of random windows with language after-the-fact instructions, enables learning a single language-conditioned visuomotor policy that can perform a wide variety of robotic manipulation tasks. The key insight here is, that solving a single imitation learning policy for either goal image or language goals, allows for learning control mostly from unlabeled play data and reduces the burden of language annotation to less than 1% of the total data. Concretely, given multiple contextual imitation datasets $\mathcal{D} = \{D^0, D^1, \ldots, D^K\}$, with different ways of describing tasks, multi-context imitation learning trains a single latent goal conditioned policy $\pi_\theta\left(a_t \mid s_t, z\right)$ over all datasets simultaneously.

D Decomposing Instructions with LLMs

Guiding the robot to areas afforded by a language instruction with the affordance model and then leveraging the low-level policy to execute the task, enables in principle to chain several language instructions in a row. Although natural language provides an intuitive and scalable

```
state = ‘drawer_open’: False, ‘blocks_on_table’: [‘red’],
‘buttons_on’: [‘green’]
# put away the red block.
open_drawer()
pick_and_place(‘red’, ‘drawer’)
close_drawer()
...
state = ‘drawer_open’: False, ‘blocks_on_table’: [],
‘buttons_on’: [‘yellow’]
# turn off the lights.
push_button(‘yellow’)
state = ‘drawer_open’: False, ‘blocks_on_table’: [‘red’,
‘green’, ‘blue’], ‘buttons_on’: [‘green’, ‘yellow’]
# tidy up the workspace and turn off all the lights
open_drawer()
pick_and_place(‘red’, ‘drawer’)
pick_and_place(‘green’, ‘drawer’)
pick_and_place(‘blue’, ‘drawer’)
close_drawer()
push_button(‘green’)
push_button(‘yellow’)
```

Figure 10.4: Example prompt to decompose abstract instructions into sequences of subtasks. Prompt context is in gray, input task commands are magenta, and generated outputs are highlighted.

way for task specification, it might not be practical to have to continually input low level language instructions, such as “open the drawer”, “now pick up the pink block and place it inside the drawer”, “now pick up the yellow block and place it inside the drawer” to perform a tidy up task for instance. Ideally, we would like to give the robot an abstract high level instruction, such as “tidy up the workspace and turn off all the lights”. Similar to Zeng *et. al.* [278], we use a standard pre-trained LLM, to decompose abstract language instructions into a sequence of feasible subtasks, by priming them with several input examples of natural language commands (formatted as comments) paired with corresponding robot code (via few-shot prompting). We leverage the code-writing capabilities of LLMs [277, 283] to generate executable Python robot code that can be translated into manipulation skills expressed in language. For example, the skill expressed by the API call push_button(‘green’), is translated into “turn on the green light” and then used to execute an inference of the policy. The only assumption we make is that the scene description fed into the prompt matches the environments state. We show a example prompt in Figure 10.4.

10.4 Experiments

Our experiments aim to answer the following questions: 1) Does integrating the proposed visuo-lingual affordance model improve performance and data-efficiency on following language instructions over using an end-to-end model? 2) Is the proposed method applicable to

the real world? 3) When paired with LLMs, can the agent generalize to new behaviors, by following the subgoals proposed by the LLM?

A Simulation Experiments

Evaluation Protocol. We design our experiments using the environment D of the CALVIN benchmark [141], which consists of 6 hours of teleoperated undirected play data that might contain suboptimal behavior. To simulate a real-world scenario, only 1% of that data contains crowd-sourced language annotations. The goal of the agent in CALVIN is to solve up to 1000 unique sequence chains with 5 distinct subtasks instructed via natural language, using onboard sensing. During inference, the agent receives the next subtask in a chain only if it successfully completes the current one.

Results and Ablations. We compare our approach of dividing the robot control learning into a high-level stream that grounds semantic concepts and a low-level stream that grounds 3D spatial interaction knowledge against HULC [159], a state-of-the-art end-to-end model that learns general skills grounded on language from play data. For a fair comparison, we retrain the original HULC agent to also finetune the language encoder, as this gives a boost in average sequence length from 2.64 to 2.69. We observe in Table 10.5, that when combined with our affordances model, the performance increases to an average sequence length of 2.93. By decoupling the control into a hierarchical structure, we show that performance increases significantly. Moreover, when initializing our affordance model with pretrained weights of R3M [247], a work that aims to learn reusable representations for learning robotic skills, CALVIN sets a new state of the art with an average sequence length of 3.30.

In order to study the data-efficiency of our proposed approach, we additionally compare our model on smaller data splits that contain 50% and 25% of the total play data. Our results indicate that our approach is up to 50% more sample efficient than the baseline. As it might be difficult to judge how much each module contributes to the overall sample-efficiency gains, we investigate the effect of pairing our affordance model trained on 25% of the data with a low-level policy trained on the full dataset. We report little difference, with an average sequence length of 2.92.

B Real-Robot Experiments

System Setup. We validate our results with a Franka Emika Panda robot arm in a 3D tabletop environment that is inspired by the simulated CALVIN environment. This environment consists of a table with a drawer that can be opened and closed and also contains a sliding door on top of a wooden base, such that the handle can be reached by the end-effector. Additionally, the environment also contains three colored light switches and colored blocks. We use an offline dataset from concurrent work [201], consisting of 9 hours of unstructured data and that was collected by asking participants to teleoperate the robot without performing any specific task. Additionally, we annotate less than 1% of the total data with language, 3605 windows concretely, by asking human annotators to describe the behavior of randomly sampled windows of the interaction dataset. The dataset contains over 25 distinct manipulation skills. We note that learning such a large range of diverse skills in the real world, from unstructured, reset-free and possibly suboptimal data, paired with less than 1% of it being annotated with language, is extremely challenging. Additionally, this setting contains an order of magnitude

Training data	Method	Language Finetuned	Tasks completed in a row					
			1	2	3	4	5	Avg. Len.
100 %	Ours + R3M	✔	**93% (0.007)**	**79% (0.002)**	**64% (0.008)**	**52% (0.003)**	**40% (0.001)**	**3.30 (0.006)**
	Ours	✔	89% (0.014)	71% (0.018)	55% (0.025)	43% (0.028)	33% (0.015)	2.93 (0.090)
	HULC	✔	84% (0.009)	66% (0.023)	50% (0.023)	38% (0.030)	29% (0.029)	2.69 (0.113)
	HULC-original	✗	82.7% (0.3)	64.9% (1.7)	50.4% (1.5)	38.5% (1.9)	28.3% (1.8)	2.64 (0.05)
50 %	Ours + R3M	✔	88% (0.030)	69% (0.032)	52% (0.016)	38% (0.013)	27% (0.004)	2.75 (0.2705)
	Ours	✔	84% (0.035)	63% (0.061)	44% (0.062)	32% (0.064)	21% (0.053)	2.45 (0.274)
	HULC	✔	79% (0.031)	54% (0.067)	37% (0.072)	26% (0.066)	17% (0.045)	2.15 (0.278)
25 %	Ours + R3M	✔	78% (0.009)	56% (0.006)	36% (0.011)	23% (0.016)	14% (0.009)	2.068 (0.046)
	Ours	✔	81% (0.007)	56% (0.006)	37% (0.008)	24% (0.017)	15% (0.016)	2.15 (0.049)
	HULC	✔	72% (0.045)	45% (0.026)	27% (0.022)	17% (0.022)	9% (0.026)	1.72 (0.135)

Figure 10.5: Performance of our model on the D environment of the CALVIN Challenge and ablations, across 3 seeded runs.

less data than related approaches [41].

Baselines. To study the effectiveness of our hierarchical architecture, we benchmark against two language-conditioned baselines: HULC [159] and BC-Z [41]. The first baseline serves to evaluate the influence of leveraging the affordance model to enable a hierarchical decomposition of the control loop, as the low-level policy is tailored to learning task-agnostic control from unstructured data. The BC-Z baseline, on the other hand, is trained only on the data that contains language annotation and includes the proposed auxiliary loss that predicts the language embeddings from the visual ones for better aligning the visuo-lingual skill embeddings [41]. For a fair comparison, all models have the same observation and action space, and have their visual encoders for the static camera initialized with pre-trained ResNet-18 R3M features [247]. For CALVIN this entails both, the visual encoder for the affordance model and the visual encoder for the static camera of the low-level policy. The

Task\Method	Ours	HULC [159]	BC-Z [41]
Lift the block on top of the drawer	**70**%	60%	20%
Lift the block inside the drawer	**70**%	50%	10%
Lift the block from the slider	**40**%	20%	10%
Lift the block from the container	**70**%	60%	20%
Lift the block from the table	**80**%	70%	40%
Place the block on top of the drawer	**90**%	50%	30%
Place the block inside the drawer	**70**%	40%	20%
Place the block in the slider	**30**%	20%	0%
Place the block in the container	**60**%	30%	20%
Stack the blocks	**50**%	30%	0%
Unstack the blocks	**50**%	40%	0%
Rotate block left	**70**%	40%	10%
Rotate block right	**70**%	50%	20%
Push block left	**70**%	50%	20%
Push block right	**60**%	50%	10%
Close drawer	**90**%	70%	20%
Open drawer	**80**%	50%	10%
Move slider left	**70**%	10%	0%
Move slider right	**70**%	30%	0%
Turn red light on	**50**%	30%	0%
Turn red light off	**40**%	20%	0%
Turn green light on	**70**%	60%	10%
Turn green light off	**70**%	50%	10%
Turn blue light on	**70**%	50%	10%
Turn blue light off	**70**%	30%	10%
Average over tasks	**65.2**%	42.4%	16.6%
Average no. of sequential tasks	6.4	2.7	1.3

Table 10.1: The average success rate of the multi-task goal-conditioned models running roll-outs in the real world.

encoder for the gripper camera is trained from scratch.

Evaluation. We start off by evaluating the success rate of the individual skills conditioned with language. After training the models with the offline play dataset, we performed 10 rollouts for each task using neutral starting positions to avoid biasing the policies through the robot's initial pose. This neutral initialization breaks correlation between initial state and task, forcing the agent to rely entirely on language to infer and solve the task. We recorded the success rate of each model in Table 10.1. We observe that the BC-Z baseline has near zero performance in most tasks, due to insufficient demonstrations. HULC is more capable, as it leverages the full play dataset with an average of 42.4% over 10 rollouts, but struggles with long-horizon planning, as do most end-to-end agents trained with imitation learning. Overall, CALVIN is more capable with an average of 65.2% success rate over 25 distinct manipulation tasks, demonstrating the effectiveness of incorporating a semantic viso-lingual affordance prior for decoupling the control into a hierarchical structure.

Finally, we evaluate how many tasks in a row each method can follow in the real world, by leveraging GPT-3 to generate sequences of subgoals for abstract language inputs, such as "tidy up the workspace and turn off the lights". We report an average number of 6.4 subgoals being executed for our method, while the baselines tend to fail after completing 2 to 3 subgoals. See the supplementary video for qualitative results that showcase the diversity of tasks and the long-horizon capabilities of the different methods. Overall, our results demonstrate the effectiveness of our approach to learn sample-efficient, language-conditioned policies from unstructured data by leveraging visuo-lingual affordances.

10.5 Conclusion and Limitations

In this paper, we introduced a novel approach to efficiently learn general-purpose, language-conditioned robot skills from unstructured, offline and reset-free data containing as little as 1% of language annotations.The key idea is to extract *language-conditioned affordances* from diverse human teleoperated data to learn a semantic prior on where in the environment the interaction should take place given a natural language instruction. We distill this knowledge into an interplay between model-based and model-free policies that allows for a sample-efficient division of the robot control learning, substantially surpassing the state of the art on the challenging language-conditioned robot manipulation CALVIN benchmark. We show that when paired with LLMs to translate abstract natural language instructions into sequences of subgoals, CALVIN is capable of completing long-horizon, multi-tier tasks the real world, while requiring an order of magnitude less data than previous approaches.

While the experimental results are promising, our approach has several limitations. First, when sequencing skills in the real world, an open question is tracking task progress in order to know when to move to the next task. In this work, we acted with a fixed time-horizon for sequencing tasks in the real world, implicitly assuming that all tasks take approximately the same timesteps to complete. Second, the code-generation module to translate abstract language inputs to sequences of subgoals assumes that the prompted scene description matches the environment's state, which could be automated by integrating a perceptual system [276]. Finally, an exciting area for future work may be one that not only grounds actions with language models, but also explores improving the language models themselves by incorporating real-world robot data [284].

Appendix

A Affordance Model Ablations

In this section we perform more ablation studies of our method on the CALVIN environment. Concretely, to better study the data-efficiency of our method, we perform ablation studies by pairing affordance and policy models trained with 25% and 100% of the training data. We observe in Table 10.2 that the performance does not change much, demonstrating the sample-efficiency of the visuo-lingual affordance model.

Training data		Tasks completed in a row					
Policy	Affordance	1	2	3	4	5	Avg. Len.
25%	25%	81%	56%	37%	24%	15%	2.15
25%	100%	82%	58%	38%	24%	15%	2.18
100%	100%	89%	71%	55%	43%	33%	2.93
100%	25%	89%	72%	55%	42%	31%	2.92

Table 10.2: Ablation of our approach trained with different data quantities for the affordance and low-level policy networks.

Next, we perform similar ablation studies for the depth prediction module trained on 25%, 50% and 100% of the dataset. We report two metrics: mean pixel distance error and the mean depth error. We plot the pixel distance error for the validation split in Figure 10.6, and observe that the error increases only in ~ 3 pixels when training the model with 25% of the data instead of the full dataset.

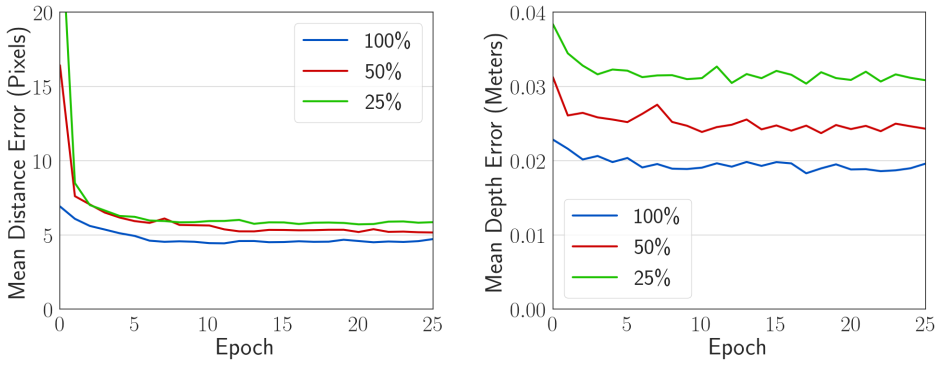


Figure 10.6: Pixel distance and depth validation error for the affordance model's depth prediction module trained with different data quantities.

Similarly, we observe that the depth error increases in $\sim$2 cm when training the model with 25% of the data instead of the full dataset. These results show that the proposed visuo-lingual affordance model is very sample-efficient, making it attractive for real world robotic applications, where collecting robot interaction data and annotating them with natural language might be costly.

B Hyperparameters

Low-Level Policy

To learn the low-level policy we train the model using 8 gpus with Distributed Data Parallel (DDP). Throughout training, we randomly sample windows between length 16 and 32 and pad them until reaching the max length of 32 by repeating the last observation and an action equivalent to keeping the end-effector in the same state. We use a batch size of 64, which with DDP results in an effective batch size of 512. We train using the Adam optimizer with a learning rate of $2e-4$. The latent plan is a vector of categorical variables, concretely we use 32 categoricals with 32 classes each. The KL loss weight β is $1e-2$ and uses KL balancing. Concretely, we minimize the KL loss faster with respect to the prior than the posterior by using different learning rates, $\alpha = 0.8$ for the prior and $1-\alpha$ for the posterior. In order to encode raw text into a semantic pre-trained vector space, we leverage the paraphrase-MiniLM-L3-v2 model [87], which distills a large Transformer based language model and is trained on paraphrase language corpora that is mainly derived from Wikipedia. It has a vocabulary size of 30,522 words and maps a sentence of any length into a vector of size 384.

For the real world experiments, the static camera RGB images have a size of 150×200, we then apply a color jitter transform with contrast of 0.05, a brightness of 0.05 and a hue of 0.02. Finally, we use the values for the pretrained R3M normalization, i.e., mean $= [0.485, 0.456, 0.406]$ and a standard deviation, std $= [0.229, 0.224, 0.225]$. For the gripper camera RGB image, we resize the image from 200×200 to 84×84, we then apply a color jitter transform with contrast of 0.05, a brightness of 0.05 and a hue of 0.02. Then we perform stochastic image shifts of $0-4$ pixels to the and a bilinear interpolation is applied on top of the shifted image by replacing each pixel with the average of the nearest pixels. Finally, we normalize the input image to have pixels with float values between -1.0 and 1.0.

Affordance Model

For the affordance model we use a Gaussian distribution to model the depth estimate. We normalize the depth values with the dataset statistics. We train the network end-to-end using a learning rate of $1e-4$ with the Adam optimizer and a batch size of 32 in a single GPU. During training, we resize the input images to $224 \times 224 \times 3$, apply stochastic image shifts of 5 pixels and apply a color jitter transform with contrast of 0.05, a brightness of 0.05 and a hue of 0.02 as data augmentation. We use the paraphrase-MiniLM-L3-v2 pretrained model [87] to encode raw text into a semantic vector space. In our experiments, we observed that the affordance model starts learning accurate predictions for the 2d pixel affordance faster than making proper depth estimations. In order to balance both tasks, we define a higher weight for the depth loss $\mathcal{L}_{depth}$ than for the affordance loss $\mathcal{L}_{aff}$ by setting β to 0.1.

C Qualitative Results

In order to better understand how the visuo-lingual affordance model, the model-based policy and the model-free policy interact with each other, we visualize a rollout for one chain of the CALVIN benchmark in Figure 10.7. Given a language instruction and a visual observation,

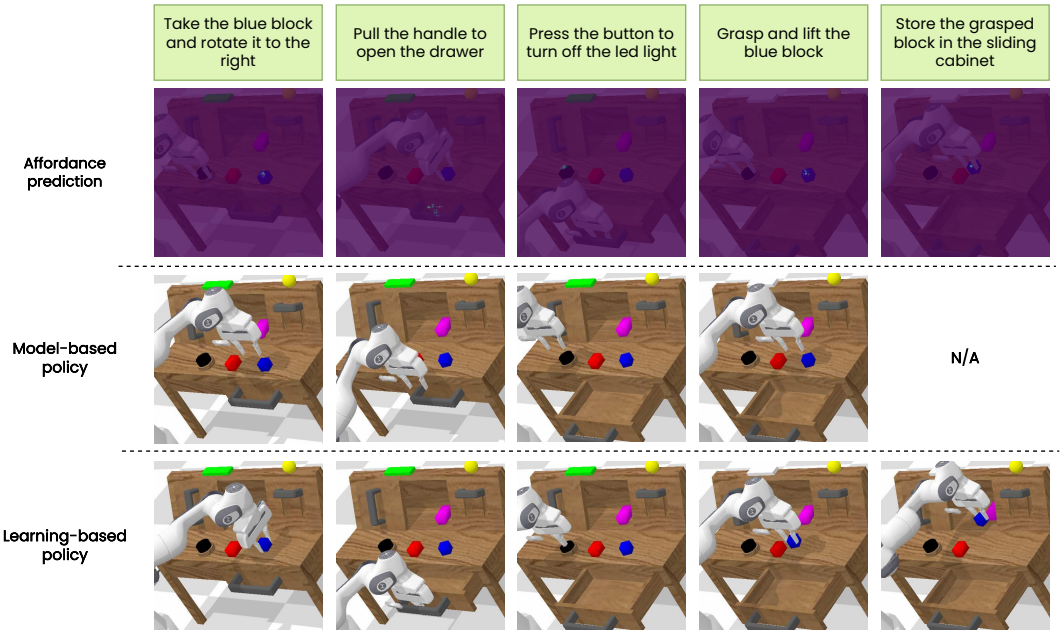


Figure 10.7: Visualization of a sample rollout for our approach in the CALVIN environment. For each column, we show the input language instruction, the predicted affordance, the reached state by the model-based policy after executing the command, and the final reached state by the learning-based policy for completing the requested task.

the visuo-lingual affordance model predicts a location which affords the given instruction. The model-based policy guides the robot to the vicinity of the afforded region. Once inside this area, we switch to the model-free language-conditioned visuomotor policy that interacts with the environment.

Chapter 11

Conclusions and Discussion

In this dissertation, we presented a line of work exploring the development of service robots that can relate human language to their perception and actions to efficiently solve long-horizon manipulation tasks. Our contributions revolved around the challenge of enabling robots to perform a wide range of tasks based on arbitrary user commands, by learning to leverage unstructured, offline data. Specifically, we considered leveraging self-supervision and structural priors to enable sample-efficient learning of language-conditioned manipulation policies. We presented frameworks for (i) following language instructions to pick and place arbitrary objects and effectively resolve ambiguities through dialogues by grounding objects and their spatial relations, (ii) learning diverse skills to perform tasks in house-like environments from uncurated data (iii) discovering, learning and transferring skills from unlabeled videos (iv) relating human language to a robots perceptions and actions, (v) efficiently completing long-horizon, multi-tier manipulation tasks in the real world. We extensively evaluated our proposed frameworks on several standard benchmarks and real-world environments. The results demonstrate that our proposed methods surpass the state of the art, while enabling efficient real world deployment.

We first tackled the challenge of learning spatial relations that enable a robot to place objects in accordance with the spatial relations expressed by the user. Typically, modeling spatial relations is challenging due to their inherent ambiguity. If one wants to model the relation "left", how far left of the reference object would form a valid relation? And where is the boundary between "left" and "in front of" for instance? Moreover, most methods assume access to corresponding 3D object shapes and relational data, which are difficult to obtain and require additional instrumentation for object tracking. We proposed a novel solution to this problem by leveraging the paradigm of auxiliary learning in combination with modeling relations directly on RGB images. In order to learn pixelwise distributions for modeling spatial relations, we proposed a novel method that implants deep features of objects into a pre-trained classifier to compute a posterior class probability over spatial relations. Our novel approach rearranges deep features, crucially enabling to reason over what relation would most likely be formed if we placed an object at the given location without modifying the input image. Our contribution is the first approach to address the problem using an end-to-end learning technique while lifting the requirement of ground-truth pixelwise annotations and the need for tracking objects with 3D models. We extensively evaluate our method using real-world data and real-world human-robot experiments. Our results demonstrate the effectiveness of our method in reasoning about the best way to place objects to reproduce a spatial relation.

The above contribution enables a robot to infer locations on a tabletop that are suitable to place an object following relational instructions from a user. However, this limits the robot to

tasks were it already has successfully grasped an object. Therefore, we additionally tackled the challenge of enabling a robot to understand natural language instructions to first pick an object from a cluttered tabletop and then place it on a second tabletop according to the relational placing instructions given by the same user. With the goal of grounding language in the perception of a robot, we proposed a multitask learning convolutional neural network architecture for jointly comprehending and generating referential language expressions. We achieved state-of-the-art results on the RefCOCO benchmark and showed that the proposed multitask learning scheme exploits synergies to improve the overall performance. This architecture effectively integrates language understanding of visual regions. It enables both to locate the referred objects on a tabletop and to resolve ambiguities via dialogue for unclear instructions, such as "fetch the yellow thing", by generating corresponding questions, such as "do you mean the lemon in the middle?". As a result, the full approach can follow complex placement instructions such as "place it behind the middle red bow". Additionally, we presented real-world experiments using our PR2 robot that demonstrates the effectiveness of our model to complete multi-stage tasks, such as tidying up a workspace or setting the table. Ours was the first comprehensive system for controlling robots that allowed tackling temporally more extended tasks by sequentially composing pick-and-place language instructions and grounding object semantics and spatial relations.

A practical limitation of the aforementioned approach is that it is not easily extendable to more complex tasks a robot might need to master in order to assist with everyday tasks, *e.g.,* opening a drawer or a fridge before retrieving an object. Therefore, a key contribution of this thesis are innovative methods that discover and learn a broad range of skills from uncurated and unlabeled data. First, we proposed a novel benchmark, coined CALVIN, that links human language to robot motor skills, behaviors, and objects in interactive visual environments. In this setting, a single agent must solve complex manipulation tasks by understanding a series of language expressions in a row, e.g., "open the drawer ... pick up the blue block ... push the block into the drawer ... open the sliding door". We contribute $\sim$24 hours of teleoperated unstructured play data together with 20K language directives across four different simulated tabletop environments and offer flexible sensor and training and test setups for evaluating long-horizon robot manipulation performance. Ours is the first public benchmark of instruction following that combines: natural language conditioning, multimodal high-dimensional inputs, 7-DoF continuous control, and long-horizon robotic object manipulation. We established baseline performance levels with an approach shown to be effective in other long horizon language-conditioned manipulation tasks and concluded that there is significant room for improvement due to the poor overall performance. Second, we conducted an extensive study of the most critical challenges in learning language conditioned policies from offline free-form imitation datasets. We systematically compared and improved key components of language conditioned imitation learning over unstructured data, such as observation and action spaces, losses for aligning visuo-lingual representations, language models and latent plan representations, and we analyzed the effect of other choices, such as data augmentation and optimization. We integrated the best performing improved components in a unified framework, which achieves state-of-the-art results on the challenging CALVIN benchmark.

Language is an intuitive and flexible of specifying tasks to a robot, but it might not be well suited for all tasks, as language looses the spatial precision of goal images. Moreover, when a user wants to teach a novel skill to a robot, it might intuitively perform a demonstration. Therefore, endowing robots with the ability to learn how to perform tasks from videos of

demonstrations is a desirable and scalable alternative task specification, with the potential of unlocking greater robot capabilities by tapping into Internet-scale unlabeled data sources, such as YouTube videos. By drawing inspiration from information theoretic measures, we proposed a novel approach that learns a task-agnostic skill embedding space from unlabeled multi-view videos. Our solution combines a metric learning loss, which utilizes temporal video coherence to learn a state representation, with a novel entropy-regularized adversarial skill-transfer loss. This technique substantially enhances the re-usability of the learned skills over multiple, vastly different domains. Extensive evaluations demonstrate that given a single video of a previously unseen task, the learned embedding enables training of continuous control policies to solve novel tasks that require the interpolation of previously seen skills.

In an effort to improve long-horizon performance for sequential multi-tier tasks, we further tackled the challenge of producing previously unseen combinations of skills, learned from an offline, unstructured robot interaction dataset, to reach temporally extended goals by "stitching" together skills. Most existing skill learning methods rely on either imitation learning or reinforcement learning as the underlying paradigm. As these two approaches have complementary strengths and weaknesses, we contribute a novel hierarchical approach that combines the strengths of the imitation learning and reinforcement learning paradigms to learn task-agnostic long-horizon policies from high-dimensional camera observations. Specifically, our proposed framework combines a low-level policy that learns latent skills via imitation learning and a high-level policy learned from offline reinforcement learning for skill-chaining the latent behavior priors. This hierarchical approach constitutes a practical solution by decomposing a whole task into smaller chunks of sub-tasks. Accordingly, the high-level policy does not need to capture in detail the physics of the world, simplifying the underlying dynamics of the RL agent. Using extensive experiments on the CALVIN and D4RL benchmarks, we demonstrated an order-of-magnitude improvement in performance upon state-of-the-art baselines on various long-horizon tasks. Additionally, we presented real-world experiments using our Franka Panda robot that demonstrates the effectiveness of our model in learning one multi-task visuomotor policy for 25 distinct manipulation tasks in the real world.

At the heart of the above approaches are goal conditioned policies learned from unstructured data that aim to endow the agent of task-agnostic control: the ability to reach any reachable goal state from any current state. In practice, this formulation tends to be data intensive. It is important to note, that most related approaches require several months of humans teleoperating robots to collect sufficient data to train visuomotor policies that can solve a wide range of everyday tasks. This drawback motivated our extensions that leveraged structural priors in the form of object affordances to accelerate learning and data-efficiency. The knowledge of object affordances helps a robot understand how objects function: what can be done with each object, where this interaction may occur, and how the object is used to achieve a goal. However, affordance learning methods typically require manually segmented annotations to learn visual affordances, limiting their applicability. We contribute a novel approach that learn affordances that are grounded in real human behavior from teleoperated play data by leveraging the gripper's opening and closing signal as a heuristic. We showed that besides enabling efficient policy learning and motion planning, a critical advantage of imbuing robots with an object-centric visual affordance prior is generalization: the learned policy generalizes to unseen, functionally similar, objects because our visual affordance model can anticipate their affordance regions. Interestingly, we also observed that the affor-

dances discovered from play data are functional affordances, priming a robot to approach an object the same way a human would.

Subsequently, we addressed learning language conditioned visual affordances with the aim of improving language conditioned policies in low data regimes. We proposed a novel approach that decomposes robot manipulation into semantic and spatial pathways and requires annotating as little as 1% of the collected uncurated data with language directives, in addition to an order of magnitude less data than comparable approaches. We evaluate our method in extensive experiments both in simulated and real-world robotic tasks, achieving state-of-the-art performance on the challenging CALVIN benchmark and learning over 25 distinct visuomotor manipulation tasks with a single policy in the real world. More importantly, we show that when paired with Large Language Models to break down abstract natural language instructions into subgoals via few-shot prompting, our method is capable of completing long-horizon, multi-tier tasks in the real world, *i.e.* "tidy up the workspace and turnoff the lights", with no additional training.

In summary, we proposed several contributions in this thesis that enable robots to acquire a wide range of general-purpose skills from unstructured data, relate them to natural language and efficiently compose them to complete long-horizon, multi-tier manipulation tasks in the real world. Our models outperform the state-of-the-art in each of the presented tasks, while often relying on weaker forms of supervision and showing greater sample-efficiency. Our contributions encompass the full robotics stack, presenting a principled and unified approach to learn low-level control, perception and high-level planning over temporally extended tasks while grounding natural language into the robot's world model. We believe that the proposed techniques have brought us closer towards the goal of general-purpose robots that can relate human language to their perception and actions by leveraging complementary forms of unstructured, scalable data.

11.1 Outlook

There are several ways by which future research can extend the scope and capabilities of the approaches we proposed in this thesis. While the main focus of this thesis has been to develop policies to effectively solve long-horizon robot manipulation tasks, general-purpose robots operating in human-centered environments need the ability to navigate in the physical world. Therefore, it would be beneficial to imbue robots with the knowledge of navigating to goals described via language, without the need of collecting data for every new environment the robot is deployed in. Recently, we started to tackle this problem by exploring how to best connect visual-language models trained on Internet-scale data to a spatial representation of the physical world that can be used by robots. In Visual Language Maps (VLMaps) [285] we have developed a map representation that directly fuses pretrained visual-language embeddings into a 3D reconstruction of the environment. By leveraging similar few-shot prompting techniques as the ones proposed in Chapter 10, VLMaps allows robots to (i) navigate to spatial goals such as "go in between the sofa and TV" or "move 3 meters to the right of the chair" without any additional training, and (ii) generate open-vocabulary obstacle maps – allowing multiple robots with different morphologies (*e.g.,* mobile manipulators and drones) to use the same VLMap for path planning.

This effectively demonstrates how natural language can act as a common grounding across otherwise incompatible embodiments and foundation models. Learning multimodal,

multitask foundation models with complementary forms of commonsense has the potential to unlock combinatorial generalization of robots to novel behaviors. In order for a robot to interact in unseen environments and understand free-form instructions, we need to explore novel ways of endowing robots "few-shot" and "zero-shot" capabilities. Unlike NLP models in which few shot text examples can be added to the context prompt, in robotics we need to look at novel ways of task specification that consider a robot's inherent multimodality, e.g. learning from third-person demonstrations, or language conditioning. A first step towards this goal might be building upon VLMaps and the manipulation policies proposed in Chapter 8 and Chapter 10 to spatially anchor more modalities (*e.g.,* sound, affordances, touch or video) to build a multimodal robot memory and solving more complex mobile manipulation tasks (*e.g.,* "bring me Tom's favorite book"). This would also open new exciting avenues, such as multimodal prompting that interleave language with other modalities [286], *i.e.* "go to the left of where the bird was heard singing" or "grasp the object that feels similar to a pineapple, but is brown" (*e.g.,* a pinecone). Another overlooked aspect in the field is that language also plays a crucial role in creating safe and interpretable autonomous robots, by verbalizing and analyzing their decision making process in language. Having a bidirectional language grounding of a robotics experience memory might be an interesting opportunity for improved interpretability and more flexible safety specifications in language, effectively treating natural language as a robotics "middleware".

One specific area of interest towards scaling robot learning is how we can reuse diverse, task-agnostic robot interaction data to move away from the default paradigm of repeating costly data collection and training from scratch for every new task and environment. A crucial ingredient of the success stories in NLP and computer vision was to move towards models pre-trained on diverse, large datasets. Achieving such generalization requires sufficiently broad datasets, which can be prohibitively expensive to collect in a robotics context. Although great efforts were made in this thesis to enable scalable ways of collecting diverse data, collecting a broader offline dataset with different robots and environments would be a promising future research direction. First attempts have been made at pooling together several robot interaction offline datasets [287, 288], but they are limited in either not supporting multiple robots, multiple environments or language annotations. Although the up-front challenge of assembling a dataset suitable for building a pre-trained robotic model is rather high, in the long-run it is cost-effective due to the possibility of bootstrapping autonomous data collection. The approaches in this thesis that learn from play data, implicitly assume the data not to be overly imbalanced. When either combining robot interaction data from different sources, or bootstrapping a lifelong learning process, it might be interesting to investigate techniques that can handle imbalanced data.

Not only is language effective for bridging multi-embodied robots, but first results also indicate that it is also a powerful implicit state representation that can crucially lead to positive transfer in multi-task settings [30] and task generalization in higher data regimes [41]. Therefore, an exciting area for future work may be one that explores leveraging natural language to bridge offline data from vastly different robot embodiments and tasks, paving the way for a robotics foundation model that enables zero or few-shot capabilities for unseen tasks with minimal human interventions. These results suggest that natural language might be a key component for both improving low-level skill learning, as well as helping robots bridge their semantic knowledge of the world. In parallel, there are some intriguing results from Lu *et al.* [289], which show that the internal structure of pre-trained language models

can be frozen and used for a variety of non-language tasks, *e.g.,* protein folding, numerical computation and vision. This suggests that it might be possible to improve generalization simply by scaling up language data, instead of collecting large amounts of task-specific data. Exploring these ideas in a robotics context remains an open research question.

In summary, the aforementioned directions highlight the benefits and opportunities that a tighter confluence of language grounding and skill learning can offer towards scaling robot learning. We hope the insights gained in this thesis will open the door for future agents that can generalize abstract concepts to unseen entities the same way humans do. Since Marvin Minsky's summer project, these are the most exciting times to work towards general-purpose robots that can relate human language to their perception and actions.

Bibliography

[1] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, "Mastering the game of go without human knowledge," *nature*, vol. 550, no. 7676, pp. 354–359, 2017.

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

[3] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko *et al.*, "Highly accurate protein structure prediction with alphafold," *Nature*, vol. 596, no. 7873, pp. 583–589, 2021.

[4] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, "Zero-shot text-to-image generation," in *International Conference on Machine Learning*. PMLR, 2021, pp. 8821–8831.

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

[6] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *NeurIPS*, 2020.

[7] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," *arXiv preprint arXiv:2103.00020*, 2021.

[8] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[9] D. A. Pomerleau, "Efficient training of artificial neural networks for autonomous navigation," *Neural computation*, vol. 3, no. 1, pp. 88–97, 1991.

[10] C. Schenck and D. Fox, "Visual closed-loop control for pouring liquids," in *IEEE International Conference on Robotics and Automation, ICRA*, 2017.

[11] A. A. Rusu, M. Večerík, T. Rothörl, N. Heess, R. Pascanu, and R. Hadsell, "Sim-to-real robot learning from pixels with progressive nets," in *Conference on Robot Learning, CoRL*, 2017.

[12] R. Rahmatizadeh, P. Abolghasemi, L. Bölöni, and S. Levine, "Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3758–3765.

[13] R. Caruana, "Multitask learning," *Machine learning*, 1997.

[14] T. Yu, S. Kumar, A. Gupta, S. Levine, K. Hausman, and C. Finn, "Gradient surgery for multi-task learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 5824–5836, 2020.

[15] T. Yu, A. Kumar, Y. Chebotar, K. Hausman, S. Levine, and C. Finn, "Conservative data sharing for multi-task offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 11 501–11 516, 2021.

[16] R. A. Brooks, "Elephants don't play chess," *Robotics and autonomous systems*, vol. 6, no. 1-2, pp. 3–15, 1990.

[17] ——, "Intelligence without representation," *Artificial intelligence*, vol. 47, no. 1-3, pp. 139–159, 1991.

[18] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet, "Learning latent plans from play," in *Conference on Robot Learning (CoRL)*, 2019.

[19] D. Kalashnikov, J. Varley, Y. Chebotar, B. Swanson, R. Jonschkowski, C. Finn, S. Levine, and K. Hausman, "Scaling up multi-task robotic reinforcement learning," in *Conference on Robot Learning (CoRL)*, 2021.

[20] A. Bonardi, S. James, and A. J. Davison, "Learning one-shot imitation from humans without humans," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3533–3539, 2020.

[21] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn, "BC-0: Zero-Shot Task Generalization with Robotic Imitation Learning," in *CoRL*, 2021.

[22] M. A. Boden, *Mind as machine: A history of cognitive science*. Oxford University Press, 2008.

[23] K. M. Hermann, F. Hill, S. Green, F. Wang, R. Faulkner, H. Soyer, D. Szepesvari, W. M. Czarnecki, M. Jaderberg, D. Teplyashin *et al.*, "Grounded language learning in a simulated 3d world," *arXiv preprint arXiv:1706.06551*, 2017.

[24] S. Tellex, N. Gopalan, H. Kress-Gazit, and C. Matuszek, "Robots that use language," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 25–55, 2020.

[25] S. Harnad, "The symbol grounding problem," *Physica D: Nonlinear Phenomena*, vol. 42, no. 1-3, pp. 335–346, 1990.

[26] D. Misra, J. Langford, and Y. Artzi, "Mapping instructions and visual observations to actions with reinforcement learning," in *EMNLP*, 2017.

[27] H. Yu, H. Zhang, and W. Xu, "Interactive grounded language acquisition and generalization in a 2d world," in *ICLR*, 2018.

[28] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. Van Den Hengel, "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments," in *CVPR*, 2018.

[29] M. Shridhar, J. Thomason, D. Gordon, Y. Bisk, W. Han, R. Mottaghi, L. Zettlemoyer, and D. Fox, "Alfred: A benchmark for interpreting grounded instructions for everyday tasks," in *CVPR*, 2020.

[30] M. Shridhar, L. Manuelli, and D. Fox, "CLIPort: What and Where Pathways for Robotic Manipulation," in *CoRL*, 2021.

[31] M. MacMahon, B. Stankiewicz, and B. Kuipers, "Walk the talk: Connecting language, knowledge, and action in route instructions," *Def*, vol. 2, no. 6, p. 4, 2006.

[32] S. Guadarrama, L. Riano, D. Golland, D. Go, Y. Jia, D. Klein, P. Abbeel, and T. Darrell, "Grounding spatial relations for human-robot interaction," in *IROS*, 2013.

[33] R. J. Mooney, "Learning to connect language and perception." in *AAAI*. Chicago, 2008, pp. 1598–1601.

[34] B. F. Skinner, *Verbal behavior*. New York: Appleton-Century-Crofts, 1957.

[35] D. Baldwin and M. Meyer, "How inherently social is language?" 2007.

[36] H. H. Clark and S. E. Brennan, "Grounding in communication." 1991.

[37] N. Chomsky *et al.*, *Reflections on language*. Temple Smith London, 1976.

[38] D. Hupkes, V. Dankers, M. Mul, and E. Bruni, "Compositionality decomposed: How do neural networks generalise?" *Journal of Artificial Intelligence Research*, vol. 67, pp. 757–795, 2020.

[39] M. A. Goodale and A. D. Milner, "Separate visual pathways for perception and action," *Trends in neurosciences*, vol. 15, no. 1, pp. 20–25, 1992.

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

[41] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn, "Bc-z: Zero-shot task generalization with robotic imitation learning," in *Conference on Robot Learning*. PMLR, 2022, pp. 991–1002.

[42] S. Reed, K. Zolna, E. Parisotto, S. G. Colmenarejo, A. Novikov, G. Barth-Maron, M. Gimenez, Y. Sulsky, J. Kay, J. T. Springenberg *et al.*, "A generalist agent," *arXiv preprint arXiv:2205.06175*, 2022.

[43] H. Moravec, *Mind children: The future of robot and human intelligence*. Harvard University Press, 1988.

[44] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 763–768.

[45] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[46] K. Mülling, J. Kober, O. Kroemer, and J. Peters, "Learning to select and generalize striking movements in robot table tennis," *The International Journal of Robotics Research*, vol. 32, no. 3, pp. 263–279, 2013.

[47] A. Zeng, K.-T. Yu, S. Song, D. Suo, E. Walker, A. Rodriguez, and J. Xiao, "Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 1386–1383.

[48] C. Eppner, S. Höfer, R. Jonschkowski, R. Martín-Martín, A. Sieverling, V. Wall, and O. Brock, "Lessons from the amazon picking challenge: Four aspects of building robotic systems." in *RSS*, 2016.

[49] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[50] A. R. Zamir, A. Sax, W. Shen, L. J. Guibas, J. Malik, and S. Savarese, "Taskonomy: Disentangling task transfer learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3712–3722.

[51] S. Toyer, R. Shah, A. Critch, and S. Russell, "The magical benchmark for robust imitation," *Advances in Neural Information Processing Systems*, vol. 33, pp. 18 284–18 295, 2020.

[52] L. P. Kaelbling, "Learning to achieve goals," in *IJCAI*, 1993, pp. 1094–1099.

[53] P. Agrawal, A. V. Nair, P. Abbeel, J. Malik, and S. Levine, "Learning to poke by poking: Experiential learning of intuitive physics," *Advances in neural information processing systems*, vol. 29, 2016.

[54] S. Stepputtis, J. Campbell, M. Phielipp, S. Lee, C. Baral, and H. B. Amor, "Language-Conditioned Imitation Learning for Robot Manipulation Tasks," in *NeurIPS*, 2020.

[55] D. Pathak, P. Mahmoudieh, G. Luo, P. Agrawal, D. Chen, Y. Shentu, E. Shelhamer, J. Malik, A. A. Efros, and T. Darrell, "Zero-shot visual imitation," in *International Conference on Learning Representations*, 2018.

[56] F. Ebert, C. Finn, S. Dasari, A. Xie, A. Lee, and S. Levine, "Visual foresight: Model-based deep reinforcement learning for vision-based robotic control," *arXiv preprint arXiv:1812.00568*, 2018.

[57] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba, "Hindsight experience replay," in *NeurIPS*, 2017.

[58] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *International Conference on Learning Representations, ICLR*, 2014.

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

[60] R. A. Howard, "Dynamic programming and markov processes." 1960.

[61] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, pp. 279–292, 1992.

[62] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[63] D. Ruppert, "Efficient estimations from a slowly convergent robbins-monro process," Cornell University Operations Research and Industrial Engineering, Tech. Rep., 1988.

[64] B. T. Polyak and A. B. Juditsky, "Acceleration of stochastic approximation by averaging," *SIAM journal on control and optimization*, vol. 30, no. 4, pp. 838–855, 1992.

[65] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[66] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," *Advances in neural information processing systems*, vol. 12, 1999.

[67] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Reinforcement learning*, pp. 5–32, 1992.

[68] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.

[69] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[70] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020.

[71] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[72] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke *et al.*, "Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation," *arXiv preprint arXiv:1806.10293*, 2018.

[73] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1179–1191, 2020.

[74] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[75] J. Firth, "A synopsis of linguistic theory, 1930-1955," *Studies in linguistic analysis*, pp. 10–32, 1957.

[76] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model," *Advances in neural information processing systems*, vol. 13, 2000.

[77] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[78] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in neural information processing systems*, vol. 26, 2013.

[79] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[80] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[81] I. Sutskever, J. Martens, and G. E. Hinton, "Generating text with recurrent neural networks," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 1017–1024.

[82] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[83] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[84] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[85] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[86] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[87] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," in *EMNLP*, 2019.

[88] O. Mees, A. Emek, J. Vertens, and W. Burgard, "Learning object placements for relational instructions by hallucinating scene representations," in *ICRA*, 2020.

[89] Y. Jiang, M. Lim, C. Zheng, and A. Saxena, "Learning to place new objects in a scene," *The International Journal of Robotics Research*, vol. 31, no. 9, pp. 1021–1043, 2012.

[90] Y. Jiang, M. Lim, and A. Saxena, "Learning object arrangements in 3d scenes using human context," *arXiv preprint arXiv:1206.6462*, 2012.

[91] O. Mees, N. Abdo, M. Mazuran, and W. Burgard, "Metric learning for generalizing spatial relations to new objects," in *IROS*, 2017.

[92] P. Jund, A. Eitel, N. Abdo, and W. Burgard, "Optimization beyond the convolution: Generalizing spatial relations with end-to-end metric learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1–7.

[93] K. Zampogiannis, Y. Yang, C. Fermüller, and Y. Aloimonos, "Learning the spatial semantics of manipulation actions through preposition grounding," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 1389–1396.

[94] S. Fichtl, A. McManus, W. Mustafa, D. Kraft, N. Krüger, and F. Guerin, "Learning spatial relationships from 3d vision using histograms," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 501–508.

[95] B. Rosman and S. Ramamoorthy, "Learning spatial relationships between objects," *The International Journal of Robotics Research*, vol. 30, no. 11, pp. 1328–1342, 2011.

[96] B. Dai, Y. Zhang, and D. Lin, "Detecting visual relationships with deep relational networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3076–3086.

[97] D. Dwibedi, I. Misra, and M. Hebert, "Cut, paste and learn: Surprisingly easy synthesis for instance detection," in *ICCV*, 2017.

[98] H. Su, C. R. Qi, Y. Li, and L. J. Guibas, "Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2686–2694.

[99] G. Georgakis, A. Mousavian, A. C. Berg, and J. Kosecka, "Synthesizing training data for object detection in indoor scenes," *arXiv preprint arXiv:1702.07836*, 2017.

[100] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon, and S. Birchfield, "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," in *CVPR*, 2018.

[101] D. Lee, S. Liu, J. Gu, M.-Y. Liu, M.-H. Yang, and J. Kautz, "Context-aware synthesis and placement of object instances," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 393–10 403.

[102] C.-H. Lin, E. Yumer, O. Wang, E. Shechtman, and S. Lucey, "St-gan: Spatial transformer generative adversarial networks for image compositing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9455–9464.

[103] R. Schulz, "Collaborative robots learning spatial language for picking and placing objects on a table," in *Proceedings of the 5th International Conference on Human Agent Interaction*. ACM, 2017, pp. 329–333.

[104] M. Shridhar and D. Hsu, "Interactive Visual Grounding of Referring Expressions for Human-Robot Interaction," in *RSS*, 2018.

[105] A. Aly and T. Taniguchi, "Towards understanding object-directed actions: A generative model for grounding syntactic categories of speech through visual perception," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7143–7150.

[106] J. Li, D. Meger, and G. Dudek, "Learning to generalize 3d spatial relationships," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 5744–5749.

[107] O. Mees, M. Tatarchenko, T. Brox, and W. Burgard, "Self-supervised 3d shape and viewpoint estimation from single images for robotics," in *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, Macao, China, 2019.

[108] R. Paul, J. Arkin, N. Roy, and T. M Howard, "Efficient grounding of abstract spatial concepts for natural language interaction with robot manipulators," in *RSS*, 2016.

[109] J. Hatori, Y. Kikuchi, S. Kobayashi, K. Takahashi, Y. Tsuboi, Y. Unno, W. Ko, and J. Tan, "Interactively picking real-world objects with unconstrained spoken language instructions," in *ICRA*, 2018.

[110] A. Magassouba, K. Sugiura, and H. Kawai, "A multimodal classifier generative adversarial network for carry and place tasks from ambiguous language instructions," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3113–3120, 2018.

[111] V. K. Nagaraja, V. I. Morariu, and L. S. Davis, "Modeling context between objects for referring expression understanding," in *ECCV*, 2016.

[112] L. Yu, Z. Lin, X. Shen, J. Yang, X. Lu, M. Bansal, and T. L. Berg, "Mattnet: Modular attention network for referring expression comprehension," in *CVPR*, 2018.

[113] R. Krishna, I. Chami, M. Bernstein, and L. Fei-Fei, "Referring relationships," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6867–6876.

[114] G. Gkioxari, R. Girshick, P. Dollár, and K. He, "Detecting and recognizing human-object interactions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8359–8367.

[115] P. Battaglia, R. Pascanu, M. Lai, D. J. Rezende *et al.*, "Interaction networks for learning about objects, relations and physics," in *Advances in neural information processing systems*, 2016, pp. 4502–4510.

[116] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap, "A simple neural network module for relational reasoning," in *Advances in neural information processing systems*, 2017, pp. 4967–4976.

[117] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[118] R. P. Poudel, S. Liwicki, and R. Cipolla, "Fast-scnn: fast semantic segmentation network," *arXiv preprint arXiv:1902.04502*, 2019.

[119] S. Chitta, I. Sucan, and S. Cousins, "Moveit!" *IEEE Robotics & Automation Magazine*, vol. 19, 2012.

[120] O. Mees and W. Burgard, "Composing pick-and-place tasks by grounding language," in *ISER*, 2021.

[121] D. Pangercic, B. Pitzer, M. Tenorth, and M. Beetz, "Semantic object maps for robotic housework-representation, acquisition and use," in *IROS*, 2012.

[122] D. K. Misra, J. Sung, K. Lee, and A. Saxena, "Tell me dave: Context-sensitive grounding of natural language to manipulation instructions," *IJRR*, 2016.

[123] S. Kazemzadeh, V. Ordonez, M. Matten, and T. Berg, "Referitgame: Referring to objects in photographs of natural scenes," in *EMNLP*, 2014.

[124] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, and D. Parikh, "Vqa: Visual question answering," in *ICCV*, 2015.

[125] J. Johnson, A. Karpathy, and L. Fei-Fei, "Densecap: Fully convolutional localization networks for dense captioning," in *CVPR*, 2016.

[126] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative Adversarial Text to Image Synthesis," in *ICML*, 2016.

[127] R. Hu, M. Rohrbach, J. Andreas, T. Darrell, and K. Saenko, "Modeling relationships in referential expressions with compositional modular networks," in *CVPR*, 2017.

[128] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein, "Neural module networks," in *CVPR*, 2016.

[129] M. Gualtieri, A. Ten Pas, K. Saenko, and R. Platt, "High precision grasp pose detection in dense clutter," in *IROS*, 2016.

[130] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *CVPR*, 2017.

[131] J. Mao, J. Huang, A. Toshev, O. Camburu, A. L. Yuille, and K. Murphy, "Generation and comprehension of unambiguous object descriptions," in *CVPR*, 2016.

[132] L. Yu, H. Tan, M. Bansal, and T. L. Berg, "A joint speaker-listener-reinforcer model for referring expressions," in *CVPR*, 2017.

[133] O. Mees, M. Merklinger, G. Kalweit, and W. Burgard, "Adversarial skill networks: Unsupervised robot skill learning from videos," in *ICRA*, 2020.

[134] N. Abdo, C. Stachniss, L. Spinello, and W. Burgard, "Organizing objects by predicting user preferences through collaborative filtering," *IJRR*, 2016.

[135] J. A. Haustein, K. Hang, J. A. Stork, and D. Kragic, "Object placement planning and optimization for robot manipulators," in *IROS*, 2019.

[136] I. Nematollahi, O. Mees, L. Hermann, and W. Burgard, "Hindsight for foresight: Unsupervised structured dynamics models from physical interaction," in *IROS*, 2020.

[137] J. Varley, C. DeChant, A. Richardson, J. Ruales, and P. Allen, "Shape completion enabled robotic grasping," in *IROS*, 2017.

[138] A. Mousavian, C. Eppner, and D. Fox, "6-dof graspnet: Variational grasp generation for object manipulation," in *ICCV*, 2019.

[139] C. Lynch and P. Sermanet, "Grounding language in play," *arXiv preprint arXiv:2005.07648*, 2020.

[140] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg, "Concept2Robot: Learning Manipulation Concepts from Instructions and Human Demonstrations," in *RSS*, 2020.

[141] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard, "Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks," *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 3, pp. 7327–7334, 2022.

[142] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *CoRL*, 2019.

[143] S. Young, J. Pari, P. Abbeel, and L. Pinto, "Playful Interactions for Representation Learning," *arXiv preprint arXiv:2107.09046*, 2021.

[144] S. Wang, M. Lambeta, P.-W. Chou, and R. Calandra, "Tacto: A fast, flexible and open-source simulator for high-resolution vision-based tactile sensors," *arXiv preprint arXiv:2012.08456*, 2020.

[145] C. Lynch and P. Sermanet, "Language Conditioned Imitation Learning Over Unstructured Data," in *RSS*, 2021.

[146] S. Nair, E. Mitchell, K. Chen, B. Ichter, S. Savarese, and C. Finn, "Learning Language-Conditioned Robot Behavior from Offline Data and Crowd-Sourced Annotation," in *CoRL*, 2021.

[147] J. Lu, D. Batra, D. Parikh, and S. Lee, "Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," in *NeurIPS*, 2019.

[148] T. Nguyen, N. Gopalan, R. Patel, M. Corsaro, E. Pavlick, and S. Tellex, "Robot Object Retrieval with Contextual Natural Language Queries," in *RSS*, 2020.

[149] H. Zhang, Y. Lu, C. Yu, D. Hsu, X. La, and N. Zheng, "INVIGORATE: Interactive Visual Grounding and Grasping in Clutter," in *RSS*, 2021.

[150] S. G. Venkatesh, A. Biswas, R. Upadrashta, V. Srinivasan, P. Talukdar, and B. Amrutur, "Spatial Reasoning from Natural Language Instructions for Robot Manipulation," in *ICRA*, 2021.

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

[152] V. Blukis, D. Misra, R. A. Knepper, and Y. Artzi, "Mapping navigation instructions to continuous control actions with position-visitation prediction," in *CoRL*, 2018.

[153] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2021.

[154] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou, "Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers," in *NeurIPS*, 2020.

[155] P. de Haan, D. Jayaraman, and S. Levine, "Causal confusion in imitation learning," *NeurIPS*, 2019.

[156] O. Mees, A. Eitel, and W. Burgard, "Choosing smartly: Adaptive multimodal fusion for object detection in changing environments," in *IROS*, 2016.

[157] M. A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg, "Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks," in *ICRA*, 2019.

[158] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," in *ECCV*, 2020.

[159] O. Mees, L. Hermann, and W. Burgard, "What matters in language conditioned robotic imitation learning over unstructured data," *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 4, pp. 11 205–11 212, 2022.

[160] D. I. A. Team, J. Abramson, A. Ahuja, A. Brussee, F. Carnevale, M. Cassin, F. Fischer, P. Georgiev, A. Goldin, T. Harley *et al.*, "Creating Multimodal Interactive Agents with Imitation and Self-Supervised Learning," *arXiv preprint arXiv:2112.03763*, 2021.

[161] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard, "Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks," *arXiv preprint arXiv:2112.03227*, 2021.

[162] T. Winograd, "Understanding natural language," *Cognitive psychology*, vol. 3, no. 1, pp. 1–191, 1972.

[163] J. Borja-Diaz, O. Mees, G. Kalweit, L. Hermann, J. Boedecker, and W. Burgard, "Affordance learning from play for sample-efficient policy learning," in *ICRA*, 2022.

[164] D. Hafner, T. P. Lillicrap, M. Norouzi, and J. Ba, "Mastering atari with discrete world models," in *ICLR*, 2020.

[165] A. Van Den Oord, O. Vinyals *et al.*, "Neural discrete representation learning," *NeurIPS*, 2017.

[166] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *arXiv preprint arXiv:1308.3432*, 2013.

[167] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma, "Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications," *arXiv preprint arXiv:1701.05517*, 2017.

[168] S. Dasari and A. Gupta, "Transformers for one-shot visual imitation," in *Conference on Robot Learning*. PMLR, 2021, pp. 2071–2084.

[169] S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio, "Generating sentences from a continuous space," *arXiv preprint arXiv:1511.06349*, 2015.

[170] D. Yarats, R. Fergus, A. Lazaric, and L. Pinto, "Mastering visual continuous control: Improved data-augmented reinforcement learning," *arXiv preprint arXiv:2107.09645*, 2021.

[171] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, "Mpnet: Masked and permuted pre-training for language understanding," *NeurIPS*, 2020.

[172] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[173] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," in *AAAI Conference on Artificial Intelligence*, 2018.

[174] G. Barth-Maron, M. W. Hoffman, D. Budden, W. Dabney, D. Horgan, A. Muldal, N. Heess, and T. Lillicrap, "Distributed distributional deterministic policy gradients," in *International Conference on Learning Representations, ICLR*, 2018.

[175] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play," *Science*, vol. 362, 2018.

[176] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, and S. Levine, "Time-contrastive networks: Self-supervised learning from video," in *ICRA*, 2018.

[177] M. Watter, J. Springenberg, J. Boedecker, and M. Riedmiller, "Embed to control: A locally linear latent dynamics model for control from raw images," in *Advances in neural information processing systems, NeurIPS*, 2015.

[178] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," in *IEEE International Conference on Robotics and Automation, ICRA*, 2017.

[179] T. Yu, G. Shevchuk, D. Sadigh, and C. Finn, "Unsupervised visuomotor control through distributional planning networks," *Proceedings of Robotics: Science and Systems, RSS*, 2019.

[180] P. Sermanet, K. Xu, and S. Levine, "Unsupervised perceptual rewards for imitation learning," *Proceedings of Robotics: Science and Systems, RSS*, 2017.

[181] C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel, "Deep spatial autoencoders for visuomotor learning," in *IEEE International Conference on Robotics and Automation, ICRA*, 2016.

[182] C. P. Burgess, L. Matthey, N. Watters, R. Kabra, I. Higgins, M. Botvinick, and A. Lerchner, "Monet: Unsupervised scene decomposition and representation," *arXiv preprint arXiv:1901.11390*, 2019.

[183] R. Jonschkowski, R. Hafner, J. Scholz, and M. Riedmiller, "Pves: Position-velocity encoders for unsupervised learning of structured state representations," *arXiv preprint arXiv:1705.09805*, 2017.

[184] M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray *et al.*, "Learning dexterous in-hand manipulation," *arXiv preprint arXiv:1808.00177*, 2018.

[185] Y. Aytar, T. Pfaff, D. Budden, T. Paine, Z. Wang, and N. de Freitas, "Playing hard exploration games by watching youtube," in *Advances in Neural Information Processing Systems, NeurIPS*, 2018.

[186] A. Singh, L. Yang, K. Hartikainen, C. Finn, and S. Levine, "End-to-end robotic reinforcement learning without reward engineering," *Proceedings of Robotics: Science and Systems, RSS*, 2019.

[187] B. Ibarz, J. Leike, T. Pohlen, G. Irving, S. Legg, and D. Amodei, "Reward learning from human preferences and demonstrations in atari," in *Advances in Neural Information Processing Systems, NeurIPS*, 2018.

[188] K. Hausman, J. T. Springenberg, Z. Wang, N. Heess, and M. Riedmiller, "Learning an embedding space for transferable robot skills," in *ICLR*, 2018.

[189] T. Haarnoja, V. Pong, A. Zhou, M. Dalal, P. Abbeel, and S. Levine, "Composable deep reinforcement learning for robotic manipulation," in *IEEE International Conference on Robotics and Automation, ICRA*, 2018.

[190] D. Esteban, L. Rozo, and D. G. Caldwell, "Hierarchical reinforcement learning for concurrent discovery of compound and composable policies," *arXiv preprint arXiv:1905.09668*, 2019.

[191] D. Dwibedi, J. Tompson, C. Lynch, and P. Sermanet, "Learning actionable representations from visual observations," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, 2018.

[192] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014.

[193] J. T. Springenberg, "Unsupervised and semi-supervised learning with categorical generative adversarial networks," in *International Conference on Learning Representations, ICLR*, 2016.

[194] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International Conference on Machine Learning, ICML*, 2015.

[195] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," in *Advances in Neural Information Processing Systems, NeurIPS*, 2018.

[196] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2017.

[197] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4004–4012.

[198] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *CVPR*, 2016.

[199] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, p. 143, 2018.

[200] L. Hermann, M. Argus, A. Eitel, A. Amiranashvili, W. Burgard, and T. Brox, "Adaptive curriculum generation from demonstrations for sim-to-real visuomotor control," *arXiv preprint arXiv:1910.07972*, 2019.

[201] E. Rosete-Beas, O. Mees, G. Kalweit, J. Boedecker, and W. Burgard, "Latent plans for task agnostic offline reinforcement learning," in *Proceedings of the 6th Conference on Robot Learning (CoRL)*, Auckland, New Zealand, 2022.

[202] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[203] M. Riedmiller, R. Hafner, T. Lampe, M. Neunert, J. Degrave, T. van de Wiele, V. Mnih, N. Heess, and J. T. Springenberg, "Learning by playing solving sparse reward tasks from scratch," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, 10–15 Jul 2018, pp. 4344–4353.

[204] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine, "Learning to walk via deep reinforcement learning," in *Proceedings of Robotics: Science and Systems*, 2019.

[205] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1179–1191, 2020.

[206] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2052–2062.

[207] S. Fujimoto and S. Gu, "A minimalist approach to offline reinforcement learning," in *Advances in Neural Information Processing Systems*, 2021.

[208] T. Yu, G. Thomas, L. Yu, S. Ermon, J. Y. Zou, S. Levine, C. Finn, and T. Ma, "Mopo: Model-based offline policy optimization," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 14 129–14 142.

[209] I. Kostrikov, A. Nair, and S. Levine, "Offline reinforcement learning with implicit q-learning," in *International Conference on Learning Representations*, 2022.

[210] R. Kidambi, A. Rajeswaran, P. Netrapalli, and T. Joachims, "Morel: Model-based offline reinforcement learning," in *NeurIPS*, 2020.

[211] Y. Chebotar, K. Hausman, Y. Lu, T. Xiao, D. Kalashnikov, J. Varley, A. Irpan, B. Eysenbach, R. C. Julian, C. Finn *et al.*, "Actionable models: Unsupervised offline reinforcement learning of robotic skills," in *International Conference on Machine Learning*. PMLR, 2021, pp. 1518–1528.

[212] A. Kumar, J. Hong, A. Singh, and S. Levine, "Should i run offline reinforcement learning or behavioral cloning?" in *International Conference on Learning Representations*, 2022.

[213] P. Florence, C. Lynch, A. Zeng, O. A. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson, "Implicit behavioral cloning," in *Conference on Robot Learning*, 2021.

[214] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard, "Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks," *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 3, pp. 7327–7334, 2022.

[215] I. Kostrikov, R. Fergus, J. Tompson, and O. Nachum, "Offline reinforcement learning with fisher divergence critic regularization," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 139. PMLR, 2021, pp. 5774–5783.

[216] A. Gupta, C. Lynch, B. Kinman, G. Peake, S. Levine, and K. Hausman, "Demonstration-bootstrapped autonomous practicing via multi-task reinforcement learning," *arXiv*, 2022.

[217] X. B. Peng, M. Chang, G. Zhang, P. Abbeel, and S. Levine, "Mcp: Learning composable hierarchical control with multiplicative compositional policies," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[218] P.-L. Bacon, J. Harb, and D. Precup, "The option-critic architecture," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.

[219] O. Rybkin, C. Zhu, A. Nagabandi, K. Daniilidis, I. Mordatch, and S. Levine, "Model-based reinforcement learning via latent-space collocation," in *ICML*, ser. Proceedings of Machine Learning Research, vol. 139. PMLR, 2021, pp. 9190–9201.

[220] K. Pertsch, O. Rybkin, F. Ebert, S. Zhou, D. Jayaraman, C. Finn, and S. Levine, "Long-horizon visual planning with goal-conditioned hierarchical predictors," in *NeurIPS*, 2020.

[221] K. Pertsch, O. Rybkin, J. Yang, S. Zhou, K. G. Derpanis, K. Daniilidis, J. J. Lim, and A. Jaegle, "Keyframing the future: Keyframe discovery for visual prediction and planning," in *L4DC*, ser. Proceedings of Machine Learning Research, vol. 120. PMLR, 2020, pp. 969–979.

[222] S. Nair and C. Finn, "Hierarchical foresight: Self-supervised learning of long-horizon tasks via visual subgoal generation," in *International Conference on Learning Representations,*, 2020.

[223] D. Jayaraman, F. Ebert, A. A. Efros, and S. Levine, "Time-agnostic prediction: Predicting predictable video frames," in *International Conference on Learning Representations*, 2019.

[224] K. Fang, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei, "Dynamics learning with cascaded variational inference for multi-step manipulation," in *CoRL*, ser. Proceedings of Machine Learning Research, vol. 100. PMLR, 2019, pp. 42–52.

[225] B. Ichter, P. Sermanet, and C. Lynch, "Broadly-exploring, local-policy trees for long-horizon task planning," in *CoRL*, ser. Proceedings of Machine Learning Research, vol. 164. PMLR, 2021, pp. 59–69.

[226] L. X. Shi, J. J. Lim, and Y. Lee, "Skill-based model-based reinforcement learning," *arXiv preprint arXiv:2207.07560*, 2022.

[227] K. Pertsch, Y. Lee, and J. J. Lim, "Accelerating reinforcement learning with learned skill priors," in *CoRL*, ser. Proceedings of Machine Learning Research, vol. 155. PMLR, 2020, pp. 188–204.

[228] K. Fang, P. Yin, A. Nair, and S. Levine, "Planning to practice: Efficient online fine-tuning by composing goals in latent space," *arXiv preprint arXiv:2205.08129*, 2022.

[229] S. Nasiriany, V. Pong, S. Lin, and S. Levine, "Planning with goal-conditioned policies," in *NeurIPS*, 2019, pp. 14 814–14 825.

[230] A. Gupta, V. Kumar, C. Lynch, S. Levine, and K. Hausman, "Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning," in *CoRL*, ser. Proceedings of Machine Learning Research, vol. 100. PMLR, 2019, pp. 1025–1037.

[231] O. Nachum, S. Gu, H. Lee, and S. Levine, "Data-efficient hierarchical reinforcement learning," in *NeurIPS*, 2018, pp. 3307–3317.

[232] T. Shankar and A. Gupta, "Learning robot skills with temporal variational inference," in *International Conference on Machine Learning*. PMLR, 2020, pp. 8624–8633.

[233] A. Mandlekar, D. Xu, R. Martín-Martín, S. Savarese, and L. Fei-Fei, "GTI: learning to generalize across long-horizon tasks from human demonstrations," in *Robotics: Science and Systems*, 2020.

[234] D. Xu, S. Nair, Y. Zhu, J. Gao, A. Garg, L. Fei-Fei, and S. Savarese, "Neural task programming: Learning to generalize across hierarchical tasks," in *ICRA*, 2018.

[235] D. Shah, P. Xu, Y. Lu, T. Xiao, A. T. Toshev, S. Levine *et al.*, "Value function spaces: Skill-centric state abstractions for long-horizon reasoning," in *International Conference on Learning Representations*, 2021.

[236] A. Singh, H. Liu, G. Zhou, A. Yu, N. Rhinehart, and S. Levine, "Parrot: Data-driven behavioral priors for reinforcement learning," in *International Conference on Learning Representations*, 2021.

[237] D. Ghosh, A. Gupta, J. Fu, A. Reddy, C. Devin, B. Eysenbach, and S. Levine, "Learning to reach goals without reinforcement learning," 2019.

[238] O. Nachum, M. Ahn, H. Ponte, S. S. Gu, and V. Kumar, "Multi-agent manipulation via locomotion using hierarchical sim2real," in *Conference on Robot Learning*. PMLR, 2020, pp. 110–121.

[239] K. Hausman, J. T. Springenberg, Z. Wang, N. Heess, and M. Riedmiller, "Learning an embedding space for transferable robot skills," in *International Conference on Learning Representations*, 2018.

[240] A. Ajay, A. Kumar, P. Agrawal, S. Levine, and O. Nachum, "OPAL: offline primitive discovery for accelerating offline reinforcement learning," in *International Conference on Learning Representations*, 2021.

[241] L. Wang, X. Meng, Y. Xiang, and D. Fox, "Hierarchical policies for cluttered-scene grasping with latent plans," *IEEE Robotics and Automation Letters*, 2022.

[242] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[243] I. Higgins, L. Matthey, A. Pal, C. P. Burgess, X. Glorot, M. M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," in *International Conference on Learning Representations*, 2017.

[244] D. Hafner, T. P. Lillicrap, M. Norouzi, and J. Ba, "Mastering atari with discrete world models," in *International Conference on Learning Representations*, 2020.

[245] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba, "Hindsight experience replay," *Advances in neural information processing systems*, vol. 30, 2017.

[246] S. Tian, S. Nair, F. Ebert, S. Dasari, B. Eysenbach, C. Finn, and S. Levine, "Model-based visual planning with self-supervised functional distances," in *International Conference on Learning Representations*, 2021.

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

[248] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4rl: Datasets for deep data-driven reinforcement learning," *arXiv preprint arXiv:2004.07219*, 2020.

[249] J. J. Gibson, "The ecological approach to visual perception," *Boston: Houghton Mifflin*, 1979.

[250] M. Hassanin, S. Khan, and M. Tahtali, "Visual affordance and function understanding: A survey," *arXiv preprint arXiv:1807.06775*, 2018.

[251] A. Nguyen, D. Kanoulas, D. G. Caldwell, and N. G. Tsagarakis, "Detecting object affordances with convolutional neural networks," in *IROS*, 2016.

[252] A. Myers, C. L. Teo, C. Fermüller, and Y. Aloimonos, "Affordance detection of tool parts from geometric features," in *ICRA*, 2015.

[253] A. Nguyen, D. Kanoulas, D. G. Caldwell, and N. G. Tsagarakis, "Object-based affordances detection with convolutional neural networks and dense conditional random fields," in *IROS*, 2017.

[254] T.-T. Do, A. Nguyen, and I. Reid, "Affordancenet: An end-to-end deep learning approach for object affordance detection," in *ICRA*, 2018.

[255] K. Mo, L. Guibas, M. Mukadam, A. Gupta, and S. Tulsiani, "Where2act: From pixels to actions for articulated 3d objects," in *ICCV*, 2021.

[256] T. Nagarajan and K. Grauman, "Learning affordance landscapes for interaction exploration in 3d environments," in *NeurIPS*, 2020.

[257] A. Zeng, S. Song, K.-T. Yu, E. Donlon, F. R. Hogan, M. Bauza, D. Ma, O. Taylor, M. Liu, E. Romo *et al.*, "Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching," in *ICRA*, 2018.

[258] L. Yen-Chen, A. Zeng, S. Song, P. Isola, and T.-Y. Lin, "Learning to see before learning to act: Visual pre-training for manipulation," in *ICRA*, 2020.

[259] T. Silver, K. Allen, J. Tenenbaum, and L. Kaelbling, "Residual policy learning," *arXiv preprint arXiv:1812.06298*, 2018.

[260] M. A. Lee, C. Florensa, J. Tremblay, N. Ratliff, A. Garg, F. Ramos, and D. Fox, "Guided uncertainty-aware policy optimization: Combining learning and model-based strategies for sample-efficient policy learning," in *ICRA*, 2020.

[261] B. Ichter, P. Sermanet, and C. Lynch, "Broadly-exploring, local-policy trees for long-horizon task planning," in *CoRL*, 2021.

[262] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[263] Y. You, Y. Lou, C. Li, Z. Cheng, L. Li, L. Ma, C. Lu, and W. Wang, "Keypointnet: A large-scale 3d keypoint dataset aggregated from numerous human annotations," in *CVPR*, 2020.

[264] K. Mo, S. Zhu, A. X. Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. Su, "Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding," in *CVPR*, 2019.

[265] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," in *RSS*, 2017.

[266] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," in *CoRL*, 2018.

[267] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *IJRR*, vol. 34, no. 4-5, pp. 705–724, 2015.

[268] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *IJRR*, vol. 37, 2018.

[269] P. Mandikal and K. Grauman, "Dexterous robotic grasping with object-centric visual affordances," in *ICRA*, 2021.

[270] C. Castellini, T. Tommasi, N. Noceti, F. Odone, and B. Caputo, "Using object affordances to improve object recognition," *IEEE transactions on autonomous mental development*, vol. 3, no. 3, pp. 207–215, 2011.

[271] T. Nagarajan, C. Feichtenhofer, and K. Grauman, "Grounded human-object interaction hotspots from video," in *ICCV*, 2019.

[272] C. Xie, Y. Xiang, A. Mousavian, and D. Fox, "Unseen object instance segmentation for robotic environments," *IEEE Transactions on Robotics (T-RO)*, 2021.

[273] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *arXiv preprint arXiv:1505.04597*, 2015.

[274] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," in *RSS*, 2018.

[275] O. Mees, J. Borja-Diaz, and W. Burgard, "Grounding language with visual affordances over unstructured data," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, London, UK, 2023.

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

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

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

[279] V. Blukis, R. Knepper, and Y. Artzi, "Few-shot object grounding and mapping for natural language robot instruction following," in *Conference on Robot Learning*. PMLR, 2021, pp. 1829–1854.

[280] W. Yuan, C. Paxton, K. Desingh, and D. Fox, "Sornet: Spatial object-centric representations for sequential manipulation," in *Conference on Robot Learning*. PMLR, 2022, pp. 148–157.

[281] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani, and J. Lee, "Transporter networks: Rearranging the visual world for robotic manipulation," *CoRL*, 2020.

[282] D. Misra, A. Bennett, V. Blukis, E. Niklasson, M. Shatkhin, and Y. Artzi, "Mapping instructions to actions in 3d environments with visual goal prediction," *arXiv preprint arXiv:1809.00786*, 2018.

[283] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. d. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman *et al.*, "Evaluating large language models trained on code," *arXiv preprint arXiv:2107.03374*, 2021.

[284] Y. Bisk, A. Holtzman, J. Thomason, J. Andreas, Y. Bengio, J. Chai, M. Lapata, A. Lazaridou, J. May, A. Nisnevich *et al.*, "Experience grounds language," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 8718–8735.

[285] C. Huang, O. Mees, A. Zeng, and W. Burgard, "Visual language maps for robot navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, London, UK, 2023.

[286] Y. Jiang, A. Gupta, Z. Zhang, G. Wang, Y. Dou, Y. Chen, L. Fei-Fei, A. Anandkumar, Y. Zhu, and L. Fan, "Vima: General robot manipulation with multimodal prompts," *arXiv preprint arXiv:2210.03094*, 2022.

[287] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn, "Robonet: Large-scale multi-robot learning," *arXiv preprint arXiv:1910.11215*, 2019.

[288] F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine, "Bridge data: Boosting generalization of robotic skills with cross-domain datasets," *arXiv preprint arXiv:2109.13396*, 2021.

[289] K. Lu, A. Grover, P. Abbeel, and I. Mordatch, "Pretrained transformers as universal computation engines," *arXiv preprint arXiv:2103.05247*, vol. 1, 2021.