Watch, Predict, Act: Robot Learning meets Web Videos

Homanga Bharadhwaj

CMU-RI-TR-25-42

May, 2025

School of Computer Science
The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania

Thesis Committee:
Abhinav Gupta (Co-Chair)
Shubham Tulsiani (Co-Chair)
Oliver Kroemer
Sergey Levine, UC Berkeley

Submitted in partial fulfillment of the requirements for the Degree of Doctor of Philosophy

Abstract

To enable robots to assist in everyday tasks in diverse natural environments such as homes, offices, and kitchens, it is critical to develop policies that generalize to novel tasks in unseen scenarios. Practical utility demands that these policies do not require task-specific adaptation at test time but can instead execute directly given a natural task specification, such as a language instruction. Moreover, such policies should be able to handle a broad spectrum of tasks—such as manipulating articulated objects, pouring, reorienting objects, and wiping tables — without the need for explicit robot data collection for every possible task, as required by the predominant paradigm of end-to-end imitation learning. The difficulty in collecting large-scale, diverse robot interaction datasets in natural scenarios makes this requirement impractical.

While typical approaches rely on a large amount of demonstration data for such generalization, in this thesis we present approaches for effectively leveraging web data to scalably augment robot interaction datasets. This thesis pioneers the paradigm of conditioning robotic policies explicitly on motion cues from predictive models trained on large-scale video datasets, enabling the policy to perform new tasks with novel objects and novel motions unseen in the robot-specific data. We formalize the notion of factorizing a robotic policy into an embodiment-agnostic interaction plan that can now use general internet data and embodiment-specific action execution conditioned on the plan, which is substantially easier of a problem. Throughout the thesis we develop common goal/language-conditioned policies that can perform multiple tasks without relying on task-specific or scene-specific heuristics.

Contents

Acknowledgement

This thesis would not have been possible without the selfless guidance, mentorship, collaboration, support, and friendship of several individuals over more than half a decade.

I am grateful to my advisors Abhinav Gupta and Shubham Tulsiani for taking a bet on me as their advisee, and bringing me to CMU. I have learned a lot from them over the years about picking exciting research problems, designing innovative solutions, and presenting research in a way that engages the broader community. I am grateful to Abhinav for providing me abundant freedom across the entire research stack: from identifying my own research problems to designing innovative solutions. His big picture insights, infectious enthusiasm, and candid feedback has helped me navigate the intricacies and uncertainties of research over the years. Shubham's mentorship has been invaluable to my projects throughout my PhD. His infinite patience and generosity with his time have been key driving forces behind my growth as a researcher. I have learned a lot about 3D vision and diffusion models from Shubham, and for that I am grateful. Beyond technical mentorship, both Shubham and Abhinav have set examples of how to lead research groups, mentor students, and foster community—qualities I aspire to carry forward in my own academic journey.

My collaboration with Sergey Levine predates my PhD, and his mentorship has had a lasting impact. Working with Sergey has been both intellectually enriching and motivating. His clarity of thought, breadth of knowledge, and deep intuition for decision-making problems have significantly influenced my research trajectory. I have had the privilege of visiting Sergey's group at Berkeley for research at different stages of my graduate studies, and am also lucky to have collaborated with him at Google and remotely across multiple projects. I thank Oliver Kroemer for serving on my thesis and research qualifier committees. Although I haven't directly collaborated on a project yet, his feedback at different stages pushed me to critically assess the limitations of my approaches and think more broadly about next steps.

During my PhD I have been fortunate to spend time in industry doing exciting research at Meta and Google. I want to thank Roozbeh Mottaghi and Vikash Kumar for hosting me at Meta-FAIR, and mentoring me across multiple projects. Their support and mentorship have been key in enabling

me to execute my research vision. I am fortunate to be mentored by Dorsa Sadigh during my Google DeepMind internship, and I thank Sean Kirmani for hosting me at Google. Dorsa's insights on my research, and also feedback regarding academic job search have positively influenced the last year of my PhD.

Prior to Carnegie Mellon, I completed a research master's at the University of Toronto. I am indebted to Florian Shkurti for recruiting me right out of undergrad despite my limited robotics background, and for his unwavering support ever since. I am grateful to Animesh Garg for co-advising me at Toronto, and I am glad we got to work on exciting research during the brief time we were both in Toronto. My time in Toronto was intellectually fulfilling and formative. During this period, I (virtually) visited Sergey Levine's lab at Berkeley during COVID, and interned at NVIDIA and Google Brain. I thank Anima Anandkumar for hosting me at NVIDIA, and Sergey Levine and Dumitru Erhan for hosting me at Google Brain. These experiences laid the foundation for much of my later PhD work.

Early in my academic journey, I had the opportunity to intern at Mila under the mentorship of Yoshua Bengio and Liam Paull, where I worked on my first robotics project. That experience inspired me to have embodied AI as the focus of my graduate studies. I'm also thankful to Brian Lim, who hosted me for a research visit at the National University of Singapore, when I was an inexperienced sophomore at IIT Kanpur. His early belief in me helped kickstart my research career. At IIT Kanpur, I learned a lot from working with Nisheeth Srivastava and Piyush Rai, and their mentorship played an important role in shaping my interests.

Throughout my graduate studies, I had the privilege of collaborating with several outstanding researchers — many of whom became long-time collaborators and friends: Samarth Sinha, Kevin Xie, Philip Huang, Haoyu Xiong, Yun-Chun Chen, Karsten Roth, De-An Huang, Mohammad Babaeizadeh, Chaowei Xiao, Aravind Srinivas, Aviral Kumar, Nicholas Rhinehart, and Danijar Hafner during my time in Toronto; Mandi Zhao, Jay Vakil, Zoey Chen, Shuran Song, Aravind Rajeswaran, Roozbeh Mottaghi, Vikash Kumar, Vincent Moens, Chris Paxton, and Mohit Sharma during my time at Meta; Sergey Levine, Dumitru Erhan, Debidatta Dwibedi, Carl Doersch, Ted Xiao, Dhruv Shah, Fei Xia, Sean Kirmani, Jie Tan, and Dorsa Sadigh during my time

at Google. At Carnegie Mellon, I was fortunate to mentor Chen Bao, Sungjae Park, Ryan Aponte, Raj Ghugare, and Sriram Krishna, and to collaborate with Ben Eysenbach and Hongyi Chen.

My PhD experience would be incomplete without the friendships and support I received in Pittsburgh. Sally's support over the years has been invaluable and hard to put in to words. I am grateful to folks in Abhinav's lab - Shikhar, Sudeep, Jason, Helen, Raunaq, and Judy for welcoming me to Pittsburgh. Over the years, I developed close friendships with Gaurav, Nupur, Ruihan, Dorian, Sheng-Yu, Yilin, Abitha, Philip, Akash, Tarasha, Ananya, Swami, Unnat, Anshika, Jay, Jianren, Justin, Kathy, Judy, Mihir, Gokul, Lili, Rajdeep, Rong, Jesper, Sungjae, Shun, Neehar, Kenny, Bart, Arjun, Chen, Guying, Priyam, Himanshu, Mononito, Murtaza, Russell, Ben N, Gautam, Zhiqiu, Zen, Georgia, Ben E, Jeff, Himangi, Yehonathan, and many others. I am grateful to my roommate in Toronto, Vanessa for the friendship and helping preserve my sanity during covid. Sam, Kevin, Dhruv, Philip, Vanessa, and Sally's support and friendship in Toronto were invaluable in tough times. During grad school, I also had the pleasure of forming friendships through conferences, workshops, and visits outside Pittsburgh — with Mandi, Zoey, Jay, Mahi, Jason Ma, Sehee, Dhruv, Sam, Krishna, Soroush, Laura, Karl, Joey, Or, Krishna, Towaki, and many others along the way.

Finally, I am deeply thankful to my parents, Usha and Pradip, for their unconditional, unwavering, and unbounded support in encouraging me to pursue my dreams.

List of Figures

Chapter 1

Introduction

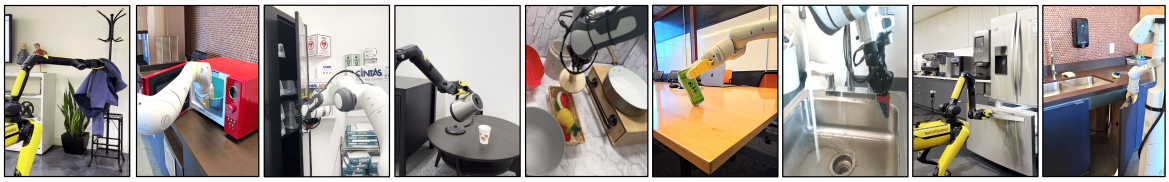

Figure 1.1: *Common goal/language-conditioned multi-task policy* executions from this thesis work with **multiple robotic embodiments** (Spot dog, Franka Panda, Everyday Robot) for different tasks in **diverse (unseen) real-world** kitchens and offices.

Developing robots that can assist in daily activities has been a long-standing goal in AI. However, achieving general-purpose robots that function seamlessly out of the box, perform tasks without manual intervention, and ensure safe human interaction remains an elusive challenge. The predominant approach in robot learning has been *end-to-end imitation learning*, where robots learn from interaction data across various tasks that humans effortlessly perform in everyday life. However, collecting large-scale robot interaction data for every task is highly challenging, requiring significant manual effort and being constrained by the physical accessibility of diverse environments. For instance, large-scale efforts such as RT-1 [22] required 17 months to collect 130k demonstrations, yet the dataset remained largely limited to tabletop pick-and-place tasks.

A fundamental challenge in robotics is enabling robots to perform new tasks in novel environments **without requiring extensive data collection for each new scenario**. This thesis proposes a scalable approach to address this data scarcity by *combining* robot-specific data with *predictive planning from*

diverse web videos—such as YouTube clips of humans performing everyday tasks. While humans and robots differ in their embodiments, many tasks share common motion patterns. For example, when pouring water from a cup, both a human and a robot must grasp the object and tilt it in a similar manner. Hence, we can extract information from web videos at multiple granularities, capturing both high-level semantic and contextual cues, such as how to approach the cup, and low-level motion cues, such as how to grasp the handle and tilt the cup properly. **Learning to predict** such cues from diverse web data can be *directly* useful for informing robotic policy learning, providing a more scalable alternative to the traditional approach of scaling robot data collection and vision-language pretraining. This paradigm allows robots to explicitly infer *how* to perform an unseen task in a new scene rather than simply imitating previously observed behaviors.

This thesis *pioneers the paradigm of leveraging motion and contextual cues from diverse web data for generalizable robotic manipulation*, as opposed to purely end-to-end imitation learning. Since web video data is large-scale, messy, and unstructured—and lacks explicit action labels—it presents unique challenges in predicting task-relevant cues useful for robotic control. We address these challenges by training predictive models for structured visual interaction plans that capture actions in an embodiment-agnostic space and by leveraging off-the-shelf visual generative models trained on web data. To this end, we have developed systems for predicting interaction plans from web videos, including hand-object trajectories [6, 13, 14] and point-tracks [16], and have trained **common** unified robot policies capable of diverse manipulation tasks in-the-wild [14, 16]. Additionally, we have demonstrated the effectiveness of visual generative AI models in robotic policy learning. Our work [9] represents the first demonstration of off-the-shelf video generation models being useful for *robot policy generalization to novel tasks with unseen objects and motions*. Furthermore, across a series of works we demonstrated how

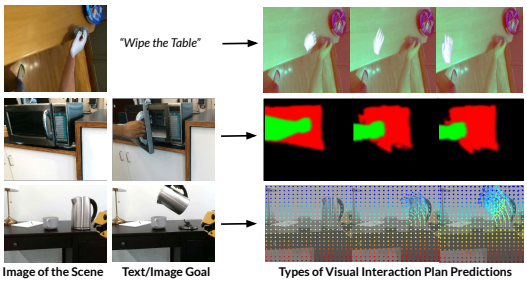

Figure 1.2: Types of motion prediction learned from web videos (hand trajectory, hand-object trajectory, point tracks) that are used in conjunction with a robot's own observations for closed-loop policy.

robotic policies can generalize to new scenes through semantic augmentations via generative inpainting [17, 28, 101]. Such semantic augmentations can be enabled at zero additional human/robot cost once a demonstration is collected, by making use of pre-trained segmentation and in-painting approaches.

A key theme throughout the thesis is how **factorizing** a robotic policy into two key components: an **embodiment-agnostic interaction plan** that leverages general internet-scale data [13, 16] and **embodiment-specific action execution** conditioned on the plan, enables scalable robotic generalization to unseen scenarios. Through this approach, we develop **common goal- and language-conditioned policies capable of performing multiple tasks**, demonstrating a scalable and practical approach to robotic learning beyond traditional imitation-based paradigms. In subsequent chapters, we will describe different ways of instantiating the visual interaction plans, how we can learn to predict these interaction plans from large-scale web video datasets, and enable close-loop robotic policy learning conditioned on these plans. We will also describe how we can repurpose existing visual generative models like video prediction models, for designing visual interaction plans for manipulation. Then we will describe how predictive planning from web data can directly make robotic policy execution more efficient through semantic augmentations based on pre-trained inpainting models. Finally, we will conclude with a vision of this paradigm of predictive planning from web data to applications like dexterous manipulation

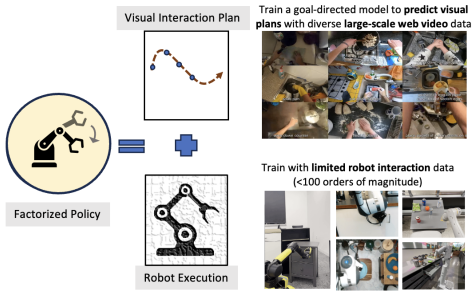

Figure 1.4: The interaction plan prediction model *with its diverse training generalizes well*, and the robot execution is tasked with a *simpler job* of converting these plans to the robot's embodiment.

Figure 1.3: Video Generation for Manipulation. We showed how casting language-conditioned manipulation as *zero-shot (human) video generation* with a pre-trained Gen AI model and *closed-loop policy execution conditioned on the generated video* and the robot's own observations enables a single policy to perform multiple tasks including those unseen in the robot dataset involving novel objects and novel motions.

in the wild, long-horizon mobile manipulation, and application to embodied prediction tasks beyond robotics in the context of wearable devices.

Chapter 2

Background

2.1 Imitation Learning for Robotics

Imitation learning (IL) is a powerful paradigm in robotics that seeks to enable robots to learn behaviors by observing expert demonstrations. Unlike reinforcement learning (RL), where an agent learns through trial and error, imitation learning directly leverages demonstrations to infer optimal policies, reducing the need for extensive online interaction.

Behavior Cloning Behavior cloning (BC) is a direct form of imitation learning where a policy is learned via supervised learning. Given a dataset of state-action pairs (s_i, a_i) collected from an expert, the goal is to learn a policy π_θ parameterized by θ that maps states to actions by minimizing the objective:

$$\mathcal{L}(\theta) = \mathbb{E}(s_i, a_i) \sim D \left[\|\pi\theta(s_i) - a_i\|^2 \right] \tag{2.1}$$

where D is the dataset of expert demonstrations.

Although behavior cloning is effective for tasks where sufficient expert demonstrations are available, it suffers from the compounding error problem. This occurs because the model is trained on states encountered by the expert but, during deployment, it encounters states induced by its own imperfect policy, potentially leading to cascading errors.

Visual Policy Learning. The choice of data modality is crucial for achieving generalization in robot learning. Vision data, which captures intricate details necessary for complex tasks such as spatial reasoning and object manipulation, has emerged as a powerful modality for robotic control policies. Recent works

have demonstrated the effectiveness of visual policies by training models using visual inputs [45, 57, 59, 103, 111, 112, 122, 142, 168].

A critical step in training a generalizable visual policy is collecting and training on diverse data. Most prior works collect data directly generated by robots [22, 179], often limited to specific environments, posing challenges for generalization across broader scenes. More recent studies explore learning image representations from large-scale videos and images beyond robot demonstration data [98, 112], and leveraging language to learn representations from videos [109, 175]. Our approach extends this line of research by leveraging pre-trained generative models to synthetically generate varied visual scenes, enhancing the semantic richness and diversity of training data. Unlike the constrained settings of direct robot-generated data or specific external datasets, we argue that visual diversity is not only about volume but also semantic richness.

Conditional Behavior Cloning. While behavior cloning is traditionally applied to single-task settings, conditional behavior cloning aims to generalize across multiple tasks by conditioning the policy on additional context information. Some prior works train robotic policies conditioned on human videos but require paired in-domain human-robot data [54, 68, 143, 157, 158, 161] and are not capable of leveraging web data. Others use curated datasets of human hand motions for learning task-specific policies [124, 140].

Efforts to leverage web data often involve predicting visual affordances, such as interaction points in an image or local information on how to interact [5, 51, 95, 108, 172]. While useful for initialization, these methods are typically combined with online learning to achieve high performance, requiring extensive deployment-time training [4, 5, 18]. Our work demonstrates how predicting motion from web data, using tools such as hand-object masks [10] and point tracks [15], can be used for conditional behavior cloning without reliance on online learning. Additionally, we explore how pre-trained generative models like video prediction can aid in generating interaction plans for conditioned behavior cloning, thereby generalizing beyond robot-specific interaction data.

Manipulation without deployment-time training. Recent advancements in robotic manipulation have aimed at achieving effective performance without requiring deployment-time training. Works such as RT-1 [22] and RT-2 [179]

have demonstrated impressive results in training general-purpose robotic policies that can handle a wide variety of manipulation tasks purely from offline data. These approaches collect large datasets of robot demonstrations, often encompassing diverse tasks and environments, to learn robust policies capable of generalizing to unseen scenarios without additional fine-tuning or online adaptation. By contrast, other approaches [4, 5, 18] rely on deployment-time training or adaptation, where the model continues to learn or refine itself during deployment, typically requiring extensive robot interaction with the environment to achieve satisfactory performance. While online adaptation offers the potential for continuous improvement, it also imposes significant practical challenges related to data efficiency, safety, and computational resources.

Our thesis focuses on achieving robust performance across a wide range of tasks without necessitating deployment-time training, thus enhancing practicality and scalability. With a goal similar to ours of using human videos to learn models that can be directly deployed, some approaches leverage curated data of human videos [124, 140] for learning task-specific policies (instead of a single model across generic tasks). Others that train a single policy across tasks require large in-domain perfectly aligned human-robot data [143, 157, 161] and are not capable of leveraging passive web data for conditional behavior cloning. Compared to these, our framework utilizes *diverse large-scale* passive human video data on the web, combined with a *small amount* of in-domain robot data, with a single model capable of tackling different manipulation tasks zero-shot.

2.2 Alternate Data Sources in Robotics

Recent successes of large-scale self-supervised approaches within both language and vision communities have showcased the advantage of large-scale data. Many recent works propose using pre-trained visual representations trained primarily on non-robot datasets [36, 53], for learning control policies [100, 112, 118, 136, 142]. Most of these works focus on single-task settings [62, 112, 118, 139], or simulated robot environments [62, 100]. Given challenges with collecting *large* real-world robotics datasets, some works focus on alternate data sources such as language [23, 97, 147, 151], hu-

man videos [4, 5, 11, 13, 113, 138, 141, 176], and generative augmentations [27, 76, 127, 171]. Our work on semantic augmentations is similar to the latter set of works, using diffusion models to generate augmentations for data collected in the real world. However, unlike [27] our approach is fully automatic. We do not need segmentation masks or object meshes [27] for generating augmentation data. In addition, our approaches do not require any further fine-tuning of a separate module for open-vocabulary segmentation and language grounding. In addition to augmenting robot-interaction data with rich augmentations, we also learn to *predict* motion from web videos, for conditional behavior cloning. While augmentations provide *robustness* to scene variations for tasks within the robot interaction data, learning to predict motion plans from web videos enables *generalization* to new tasks outside the robot-specific data.

Learning Visual Representations for Manipulation. Visual imitation is a promising technique for generalizable robot manipulation [12, 45, 103, 168]. Recent works that have scaled this approach for learning large-scale models for manipulation require extremely high number of expert robot trajectories, often demanding years for collection [17, 22, 179], and still suffer from limited generalization to unseen scenarios for novel objects. Going beyond image observations, prior works have also investigated structured representations like point-clouds [24, 116, 134] and keypoints [125] for manipulation, but are restricted to tasks in structured table-top scenarios. Some of these that predict action in the form of flow-based representations [50, 134] require 3D datasets of robot interactions (often from simulation) which constrain them from generalizable real-world deployments. More recently, Vecerik et al. [154] use point tracking for visual servoing, and the setup requires structured multi-stage definitions of the task and is limited to only minor test-time variations compared to training data. Concurrent work [158] that improves upon [154] by predicting future tracks of points in the current image can learn a policy by combining in-domain human videos with in-domain robot videos. However, the framework is not directly amenable for leveraging web videos because the policy relies on per-step image observations for track prediction. Compared to this, and developed independently, we learn to predict trajectories of arbitrary points from web videos given just an initial image and a goal. We show how we can use these predicted tracks to infer rigid transforms of objects for

open-loop execution, and further improve the open-loop plan by predicting residuals over the actions, for closed-loop deployment. This enables much diverse robot manipulation behaviors with a single model, that generalizes to unseen novel objects and scenes in-the-wild.

Leveraging Non-Robot Datasets for Manipulation. One common way of using data beyond robot interactions for efficient learning is to pre-train the visual representations which serve as backbones for the policy models [98, 100, 112, 119, 160] with passive human videos [53, 80] and image data [36]. However, these methods still crucially rely on a lot of in-domain robot data or deployment-time training, and are restricted to learning task-specific policies. Some works that do not require deployment-time training, go beyond visual representations and use curated data of human videos to leverage human hand motion information [124, 140] for learning task-specific policies (instead of a single model across generic tasks). Others that train a single policy across tasks require large in-domain perfectly aligned human-robot data [143, 157, 161] and are not capable of leveraging web data. Towards learning structure more directly related to manipulation from web videos, some works try to predict visual affordances in the form of where to interact in an image, and local information of how to interact [5, 51, 95, 108]. While these could serve as good initializations for a robotic policy, they are not sufficient on their own for accomplishing tasks, and so are typically used in conjunction with online learning, requiring several hours of deployment-time training and robot data [4, 5]. In an early work, we learn to predict masks of hand and objects in the scene [10] for conditional behavior cloning. More recently, we learn to predict an approximate motion of how objects in the scene move in the future through point tracks for the entire trajectory and combined with limited robot interaction data develop a *zero-shot* manipulation system in terms of not requiring any deployment-time training.

Frameworks for Scaling Robot Learning. Given the cost of supervision in robot learning, self-supervised learning [8, 96, 123] methods leveraging large unlabeled datasets have been a dominant paradigm towards building general-purpose agents. Large-scale simulations [69, 107, 170, 177] have also been leveraged with the hope of first learning a general multi-task policy [43, 72, 73, 128, 133, 144] and then transferring it to real world via sim2real[20, 61, 142, 152]. However, most multi-task RL works focus on narrow domains[43, 145],

and those in the real-world show limited generalization and task diversity[55, 101]. While other works [72, 128, 169] focus on diverse multi-task scenarios, they restrict to evaluating trained policies mostly in simulation. By contrast, our work focuses on a large, diverse set of real-world manipulation tasks. Many recent works use imitation learning with large-scale real-world robot tele-operation datasets[33, 34, 42, 70, 102, 104]. While early works collect limited real-world data [70, 104], more recent approaches [22, 42, 74] collect much larger datasets. In fact, [22] gathers, possibly, the largest dataset ($\approx 130K$ demonstrations) outside bin and place settings and shows impressive generalization with skills learned using this data. Our work is similar in spirit, *i.e.*, we focus on real-world manipulation tasks and aim to learn a multi-task policy using *limited* real-world demonstrations. However, unlike [42], we avoid toy environment setups and focus on realistic real-world kitchen setups with clutter and multiple feasible tasks in a scene. Additionally, our robot manipulation systems exhibit a much greater diversity of skills than [20, 22, 74] while being trained only on orders of magnitude less robot interaction data.

2.3 Understanding and Predicting Interactions from Web Videos

Several recent approaches in computer vision have focused on understanding human activities, captured in large-scale datasets of human-object interactions in diverse everyday settings [30, 32, 35, 52, 53, 91, 137]. Specifically, prior work has investigated learning self-supervised visual representations [2, 64, 106], human pose estimation [3, 19, 48, 63, 67, 88, 94, 131, 146, 178], object pose estimation [65, 66, 81, 126, 159], interaction hotspot prediction [51, 95, 110], prediction of plausible hand grasps [21, 108], and activity understanding [26, 149]. Our future hand-object mask prediction module is inspired by these developments in visual understanding, where we focus on learning motions of hands and objects from passive human videos that are directly relevant for manipulation, and abstract out task-irrelevant visual details through semantic masks.

Reconstructing Hand-Object Interactions. Reconstructing 3D represen-

tations of hand-object interactions from images or videos is a complex task due to challenges like occlusions and the intricate articulations of the human hand. Recent approaches have sought to address these challenges by leveraging advancements in computer vision and machine learning. For instance, some methods employ compositional articulated implicit models to jointly reconstruct hands and objects from monocular videos without relying on extensive 3D annotations. Others integrate large language and vision models to retrieve and align 3D object models with observed interactions, enhancing reconstruction accuracy [25]. Additionally, diffusion-guided frameworks have been proposed to infer 3D shapes of hands and objects from short video clips by optimizing neural representations per video [166].

Predicting Hand-Object Interactions. Predicting future hand-object interactions involves forecasting hand movements and identifying potential contact points with objects. This predictive capability is essential for applications such as assistive robotics and augmented reality. Some methods focus on forecasting hand motion trajectories and future contact points, known as interaction hotspots, from egocentric video inputs [90]. Others utilize vision-language models to generate textual responses and predict future hand trajectories through natural language conversations, integrating high-level reasoning with low-level motion prediction [7]. These approaches aim to equip systems with the ability to anticipate and respond to human actions in real-time.

Learning Affordances. Towards learning structure more directly related to manipulation, some works try to predict visual affordances in the form of where to interact in an image, and local information of how to interact [5, 51, 95, 108]. While these could serve as good initializations for a robotic policy, they are not sufficient on their own for accomplishing tasks, and so are typically used in conjunction with online learning, requiring several hours of deployment-time training and robot data [4, 5]. Our work differs from this in terms of predicting an approximate motion of how a human hand and the object is likely to move for the entire trajectory (not just at/near contacts unlike affordances) and is *zero-shot* in terms of not requiring any deployment-time training.

Understanding Generic Object Motion in Videos Understanding generic object-centric motion in videos has gained significant traction with the development of more advanced techniques for point track prediction. These

approaches aim to understand and anticipate generic motion patterns across diverse scenes, rather than being restricted to specific tasks or environments. Point tracking systems, such as TAPIR [38] and CoTracker [78], have made substantial progress by enabling robust tracking of arbitrary points throughout video sequences. These systems work by identifying points of interest in an initial frame and predicting their motion across subsequent frames, allowing for the generation of continuous motion trajectories. Such approaches are particularly effective for understanding broad categories of motion interactions that do not necessarily conform to predefined task templates.

Recent approaches have investigated generating videos given a description of the task, and often conditioned on a scene [40, 56, 163, 165]. Other works have attempted understanding generic videos by identifying visual correspondences between frames [50, 93]. Building on these advancements, our track prediction model leverages video tracking methodologies to generate ground-truth tracks from web videos. We then train models to predict future point tracks of arbitrary points given an initial image and a goal, paving the way for improved understanding and manipulation in real-world scenarios.

Chapter 3

Hand-Object Interaction Plan prediction from Web Videos for Robotic Manipulation

A central goal in the rapidly growing area of robot learning is to develop generalist robots capable of performing a plethora of everyday manipulation tasks in diverse unseen real-world scenarios. In addition, to be practically useful, they should be able to accomplish these tasks out of the box when deployed in unseen scenarios. Towards this goal, our work pursues learning diverse core skills like manipulating articulated objects, picking, placing, scooping, pouring, twisting, stacking, and swiping, among others that humans can effortlessly perform during everyday interactions. Moreover, we want these skills to be generalizable to unseen scenes with new objects, and be executable in a "zero-shot manner" i.e. without deployment-time training.

An unsophisticated way to attempt this goal is to collect a gigantic robot interaction dataset for imitation learning. Albeit simple, this is not scalable for diverse real-world generalization because it would require collecting data not just for different tasks but for interaction across different objects with different skills, and is bottle-necked by physical access constraints. Indeed, recent approaches that attempt at developing diverse manipulation capabilities

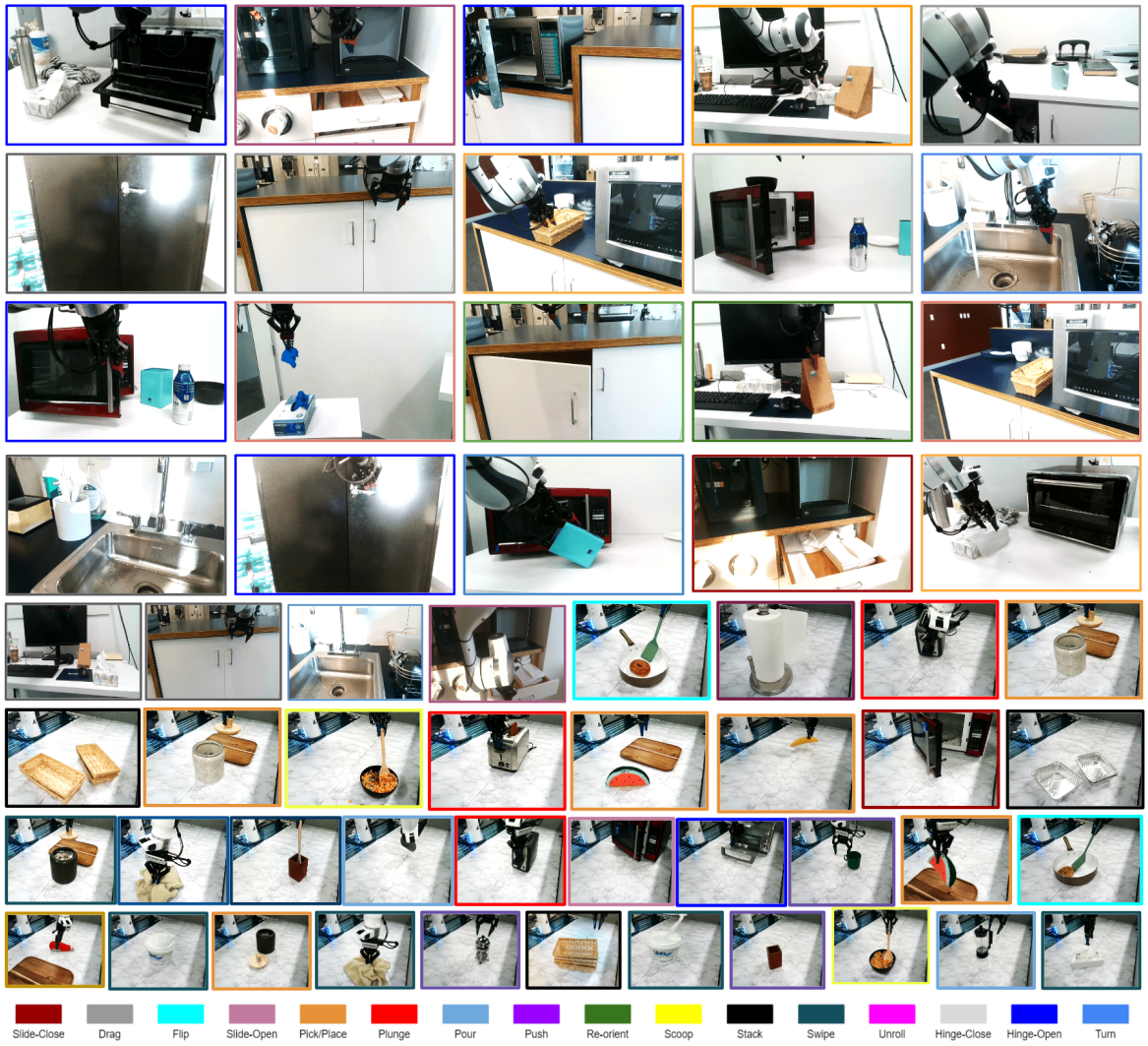

Figure 3.1: A subset of different manipulation behaviors generated by our framework `HOPMan`. By learning task-agnostic *human-plan* prediction and *robot-action* translation models, our system can interact with generic objects and execute diverse skills e.g. unrolling, scooping, pouring, re-orientation, articulated object manipulation, etc.

require years of on-robot data collection [22], and are still largely limited to picking, placing, and pushing skills. Our solution is to factorize the task of learning a generalizable policy into 1) learning an interaction plan that captures changes that the object and the manipulator can undergo, 2) translate the plan into actions that can be executed on a robot. Our key insight is that the first module can leverage non-robot data, and in particular large passive datasets of human videos on the web. Given this *human-interaction-plan*, acting in the

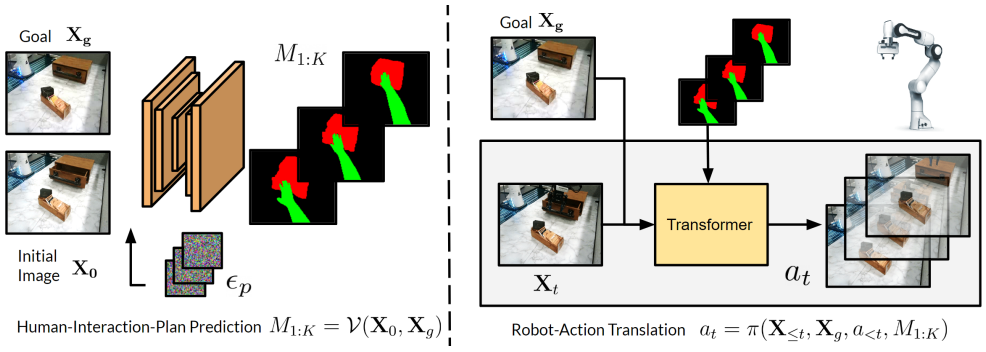

Figure 3.2: `HOPMan` consists of a *human-interaction-plan prediction model* (left), and a *robot-action translation model* (right). Given an initial image of a scene $\mathbf{X_0}$ and a goal image $\mathbf{X_g}$, a diffusion model hallucinates plausible future hand and object masks $M_{1:K}$. These predictions along with current RGB observations of the scene $\mathbf{X}_t$ go as input to a translation model (instantiated as a closed-loop policy $\pi(\cdot)$) that outputs robot actions a_t for executing the motions on a robot. Additional details on the approach are in section 3.1.

real world reduces significantly in complexity as we only need to instantiate the human plan in a robot's context as *robot-actions*. This translation model can be trained with limited paired human-robot data and generalizes to objects and scenarios that are unseen in the robot data since the human-interaction plan generalizes by virtue of diverse training.

Some prior robot learning approaches have also investigated leveraging out-of-domain (human) data, primarily for learning visual representations [98, 100, 112] and robotic affordances [4, 5, 51, 110]. However, these approaches require *a lot of* further robot demonstrations for policy learning and typically also require a lot of deployment-time training. Other approaches learn task-specific action priors [124, 140] for a few categories of manipulation tasks, with separate policies for each task. Compared to these, our approach of factorizing the overall policy can enable zero-shot manipulation over a range of diverse tasks, with a single policy that can be appropriately goal-conditioned and doesn't require any deployment-time training.

We consider semantic masks of hands and objects as a structured space for defining the *human-plan*, since it abstracts out task-irrelevant details of the environment. Given an image of a scene and a goal image, we train the prediction model to predict the *human-plan* as plausible future hand and object masks. We train this model across clips in diverse passive videos on the web and show that it generalizes to new scenes in our real-robot experiments. In order to transform the predictions to a physical embodiment's *robot-actions*

, we train a translation module on a small amount of paired data (∼600 trajectories). We abbreviate our framework as `HOPMan` (**H**and **O**bject **P**lan for robotic **M**anipulation).

Through experiments on a set of 100 tasks, involving 16 skills and 40 objects, we show `HOPMan` can help distill information about manipulation from passive human videos on the web to physical scenes in a robot's workspace, as evaluated through generalization across five different axes. In summary, we make the following contributions:

- Present an approach for learning goal-conditioned prediction of hand-object interaction plans using everyday interaction videos.
- Develop a framework that casts robot manipulation as translation of (predicted) hand-object plans, thus allowing the use of easily available human videos for learning diverse manipulation.
- Demonstrate the overall framework across 100 manipulation tasks involving 40 objects with 16 skills, while evaluating generalization in a structured manner for table-top manipulation and in-the-wild manipulation in unseen scenes.

3.1 Approach

We aim to develop a robot manipulation system that can accomplish diverse skills zero-shot with a plethora of different unseen objects in the real world. Our key insight is to leverage a factorized policy model (see Fig. 3.2) that consists of two stages: a) a goal-conditioned human plan prediction model that predicts future masks for plausible hand and object motions, and b) a translation model that learns to transform the corresponding predicted plans into actions that can be executed with a robot for real-world manipulation. We show how we can train the human-plan prediction model on diverse passive human videos from existing large scale datasets, and use it for predicting plausible plans in a robot's environment. In contrast, the translation model can be trained with a small amount of paired human-robot data. This factorization allows us to generalize to scenarios that are unseen in the robot data, because

the human-interaction-plan model with its diverse training generalizes well, and the translation model is tasked with a simpler job of converting these plans to the robot's embodiment.

3.1.1 The Human-Plan Prediction Model

Instead of predicting the future in the image space, we focus on predicting only the motion of the human hand and the object being interacted with, in terms of respective semantic masks. We enable this prediction through a diffusion model trained on diverse human videos on the web. For each video in the training data, we extract hand-object masks for each frame. Let $\mathbf{M_{1:K}}$ denote the respective mask frames from time steps 1 to K. For simplicity we consider each mask frame to be an image, where all the hand pixels are green, all the object pixels are red, and the rest of the pixels are black. Let $\mathbf{X_0}$ denote the first frame (RGB) of the video, $\mathbf{X_g}$ denote the last frame (RGB) of the video, which will act as a goal frame, and $\mathcal{V}(\mathbf{X_0}, \mathbf{X_g})$ denote the prediction model. In the forward diffusion process, all the mask frames $\mathbf{M_{1:K}}$ are corrupted by

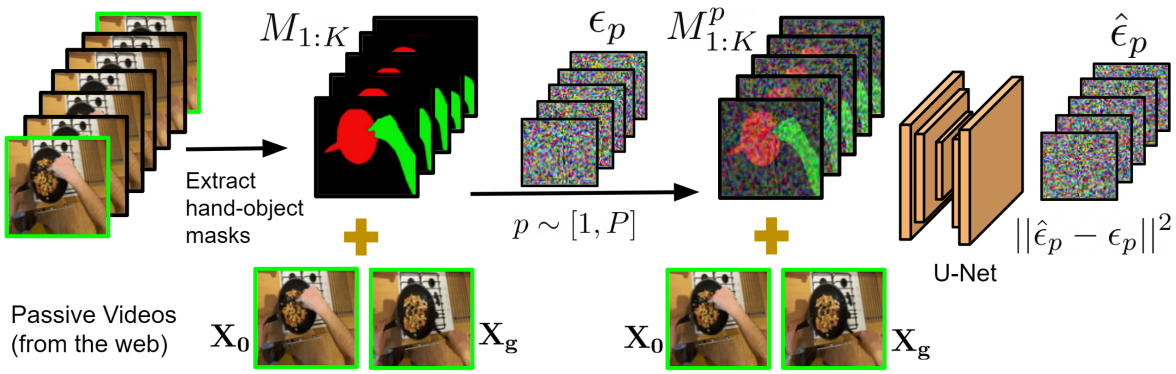

Figure 3.3: Detailed illustration of a training pass through the future prediction model. This is a diffusion model, with a U-net that predicts per-frame noise at each step p of the diffusion process. Additional details on the model and training are in Section 3.1.1.

incrementally adding noise, and converging to a unit Gaussian distribution $N(\mathbf{0}, \mathbf{I})$. New samples can be generated by reversing the forward diffusion process, by going from Gaussian noise back to the space of mask frames. To solve the reverse diffusion process, we need to train a noise predictor $\epsilon_\theta(\cdot|t)$ which is a time-conditioned U-net [129, 155] trained to predict the noise at each step of the diffusion process. The input to the network at step t of the

diffusion process is a channel-wise concatenation of the conditioning frames and noisy mask frames $[\mathbf{X_0}, \mathbf{X_g}, \mathbf{M^t_{1:K}}]$, the output is the predicted noise of same dimensionality as the input. Fig. 3.3 illustrates this visually, and equation 1 shows the training objective $\mathcal{L}(\theta)$.

$$\mathbb{E}_{t,[\mathbf{X_0},\mathbf{X_g},\mathbf{M_{1:K}}]\sim p_{\text{train}},\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})} \left[||\epsilon - \epsilon_\theta(\sqrt{\overline{\alpha}_t}\mathbf{M_{1:K}} + \sqrt{1-\overline{\alpha}_t}\epsilon|\mathbf{X_0}, \mathbf{X_g}, t)||^2\right]$$

Here $\overline{\alpha}_t$ is a hyper-parameter that depends on the noise schedule of the diffusion process. During inference, given $\mathbf{X_0}$, $\mathbf{X_g}$ we obtain $M_{1:K} = \mathcal{V}(\mathbf{X_0}, \mathbf{X_g})$ through reverse diffusion.

3.1.2 The Robot-Action Translation Model

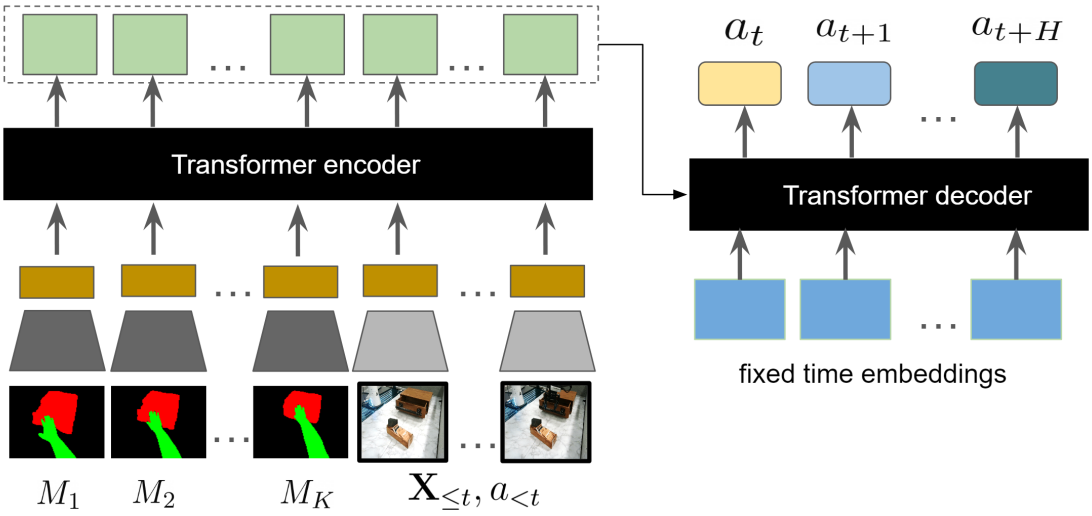

Figure 3.4: Architecture of the translation model that transforms predicted future hand-object masks to a robot trajectory, described in section 3.1.2

We use the human-plan predictor discussed in Section 3.1.1 to hallucinate plausible future hand and object masks for interaction in a robot's physical scene. However, this human-plan doesn't directly inform what actions the robot should execute to be able to perform the desired interaction. To enable robot manipulation in the context of the predicted plans, we learn a translation model. The translation model is a transformer that is conditioned on the outputs of the future prediction model $M_{1:K}$ and for each observation $\mathbf{X}_t$, and

predicts actions for H steps in the future. The model behaves as a closed-loop policy $\pi(\mathbf{X}_{\leq t}, \mathbf{X}_g, a_{<t}, M_{1:K})$ that is queried at each time-step t during deployment. Predicting multiple time-steps H in the future and averaging actions during deployment, helps in executing smooth robot motions, with less compounding errors [174]. We describe the architecture of the translation model in Fig. 3.4 and additional details in Appendix 3.3.2.

For training the translation model, we need some paired human-robot data, where we have pairs of trajectories that involve a robot manipulating an object, and a human manipulating a similar object. To obtain such paired trajectories, we develop two approaches:

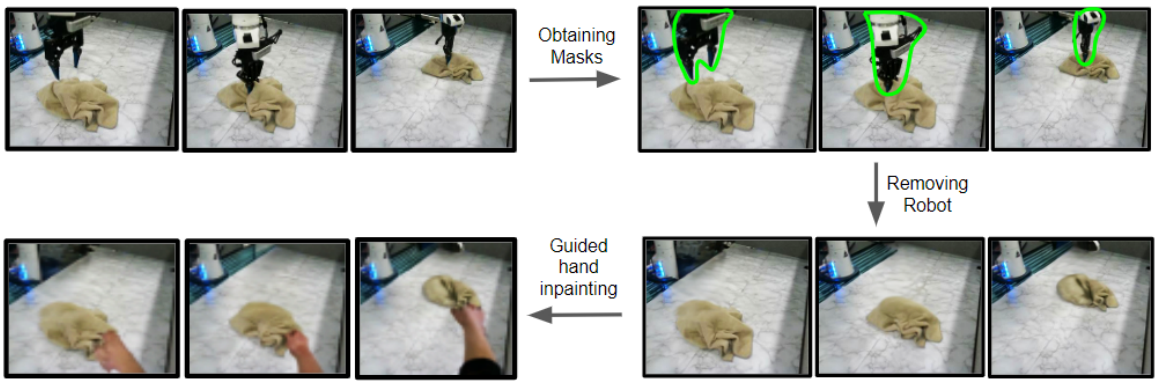

Figure 3.5: Illustration of the different steps in generating hallucinated human hand trajectories from robot trajectories. This is an alternate data source for the translation model in addition to collecting paired human-robot data.

Collecting paired demonstrations: A human operator tele-operates a robot in scene, and after reset, or in a parallel identical setup, a human manipulates a similar object with an approximately similar motion as the robot arm. Collecting this paired data is not very expensive, and we spent around 3 days to collect 600 trajectories.

Hallucinating paired data: To augment the paired demonstrations, we also propose to leverage (more easily collectable) robot-only data. To obtain hallucinated pairs, we can convert videos of a robot trajectory into a videos of a human trajectory through recent advances in hand in-painting techniques [129, 167] . Specifically, we obtain robot masks per frame through simulation, and perform inpainting to remove the robot from the scene. We then perform guided in-painting of a plausible human hand [167] around the location of the robot end-effector in the scene. Fig. 3.5 visually illustrates

this process of hallucinated data generation. In the experiments, we show how hallucinated paired data generated through this approach can be used to boost the performance of the translation model. Additional details on the hallucinated data generation are in the Appendix 3.3.3.

3.2 Experiments

Through experiments with diverse real-world objects in unseen scenarios, we demonstrate generalization of our framework for several robot manipulation tasks.

3.2.1 Experiment Settings

We consider two different types of manipulation settings for experiments - table top scenarios with a fixed robot and camera, and in-the-wild manipulation with the same robot and camera on a mobile base.

Table-Top Manipulation. We consider several everyday objects with different plausible manipulations for our experiments. We demonstrate results on a total of 16 skills: pouring, plunging, pushing, picking/placing, slide-opening, slide-closing, hinge-opening, hinge-closing, swiping, dragging, flipping, scooping, in-place re-orientation, unrolling, and stacking, and 40 object types, with 2-3 instantiations per object type, comprising around 100 tasks. Detailed list of objects and tasks are in the Appendix section 3.3.1

In-the-Wild Manipulation. We drag a Franka Panda arm on a mobile base across natural kitchen and office scenes. The camera is also attached to the base, and moves along with it. For these experiments we fine-tune the translation model used for the table-top experiments, on $\sim$200 additional paired trajectories collected with the mobile robot. For evaluation, we consider the same generalization levels described above. This setting is much more challenging because in addition to object and skill variations, we also have scene variations, including completely new scenes never seen in the paired data. Details of variations are in the supplementary website.

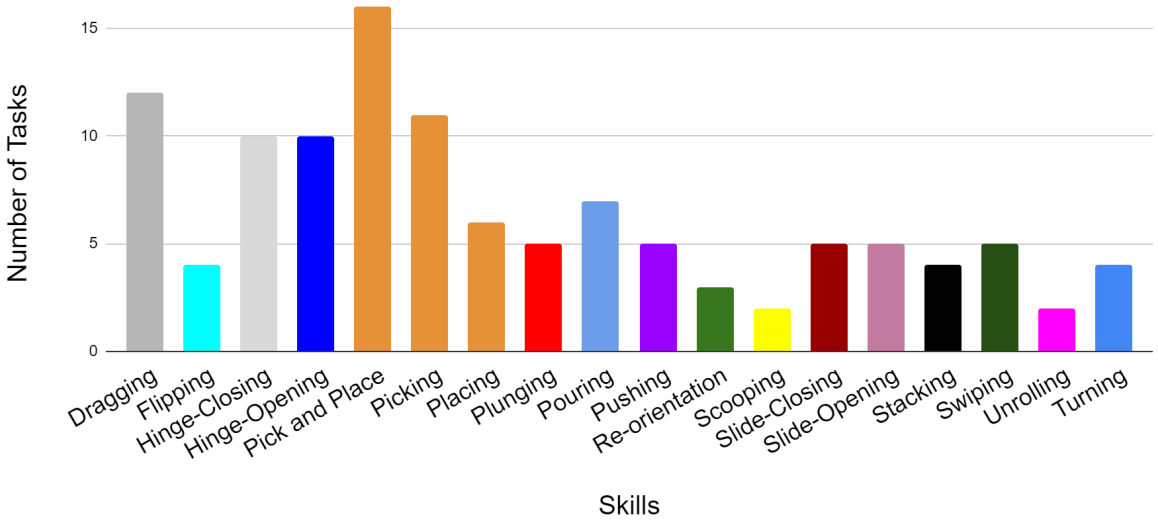

Figure 3.6: Distribution of skills across tasks in our experiments. The diversity of skills is more representative of real-world distributions, compared to pushing/pick and place that is predominant in robot learning papers.

3.2.2 Training data

The training data for our framework consists of a large set of passive web videos, a small amount of paired human-robot in-domain data, and some unpaired robot-only data.

Passive Human Data. For the future prediction model, we use existing passive human videos [31, 52, 53] and obtain ground-truth semantic masks for the right hand and the object being interacted with the right hand in each frame [31, 173]. We sample short video clips, each lasting a few seconds and do not curate the videos in any way with tasks or language labels.

Paired Data. For the translation model, we use a small amount of paired collected by us (∼400 trajectories in-lab and ∼200 trajectories in-the-wild) and a larger robot-only data (∼1000 trajectories) combined with hallucinated hand masks through the approach described in section 3.1.2. All the robot data are collected through an adaptation of the tele-operation stack proposed in [89].

3.2.3 Defining Tasks and Evaluating Generalization

Prior works in robot learning adopt widely different and oftentimes inconsistent definitions of generalization criteria. Some prior works [22, 101, 124, 148]

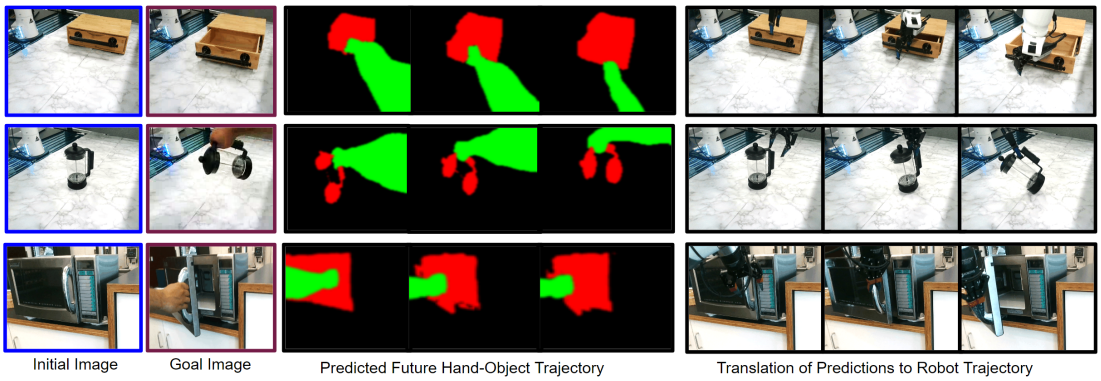

Figure 3.7: Qualitative results for the entire framework. We show qualitative results for the predicted hand-object trajectory given an initial image of a scene and a goal image, followed by translation of the predictions to a robot trajectory for execution in the real world.

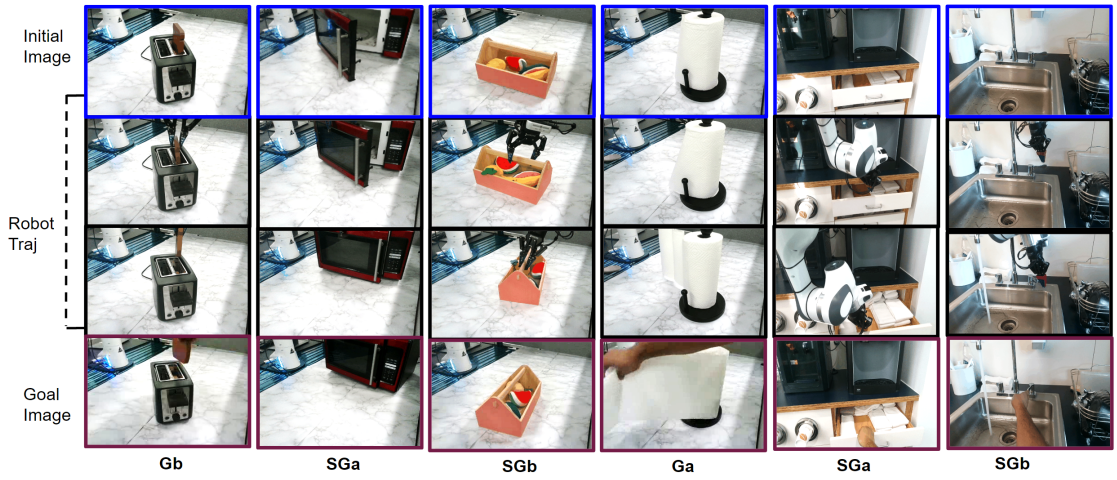

Figure 3.8: Examples of robot evaluations. We show qualitative results for robot evaluations, with an intermediate image and the image corresponding to the final state reached by the robot, for a given initial scene and a goal image. Subscripts show the type of generalization for each evaluation, as described in sec 3.2.3. More robot videos of evaluations are in the linked website.

consider seen vs. unseen objects, where the unseen objects often involve different instantiations of the seen objects, with shape, color, and texture variations, with skills (e.g. pushing, picking etc.) that are always seen in the training data. Others [29, 60] only consider generalization in terms of position and configuration variations of seen objects. In light of this, in this paper, we develop a structured criteria for evaluating generalization in terms of object categories, object instantiations, object configurations, and skills. We adopt

the following definitions

- **Task definition:** Each task is a tuple consisting of (object category, object instance, skill). Here, object category denotes the type of the object e.g. 'drawer', 'mug', 'toaster' etc. While, object instance defines a particular object within a category, with a specific instantiation of color, shape, size, and texture. Finally, skill defines the particular behavior e.g. 'open', 'flip', 'push' etc. that can be done with an object.

- **Mild generalization (`MG`):** This involves generalizing among unseen configurations (i.e. position and orientation variations) for seen object instance and seen skills, along with mild variations in the scene like lighting changes.

- **Standard generalization (`G`):** We have the following types of generalization in this category

  - **instance generalization (`Ga`):** In addition to variations in `MG`, in `Ga` we evaluate unseen object instance for seen skills. For example, only a red mug is seen with the push skill in training, and we generalize to pushing motions for green, and purple mugs of different shapes and textures.
  - **unseen combinations (`Gb`):** This includes scenarios with unseen (object category, skill) pairs but each seen independently in training. So atleast one instance of an object category is seen during training, and the skillis also seen during training but not in relation to this object. For example, 'open' is seen, and 'close door' is seen but 'open door' is not seen in training.

- **Strong Generalization (`SG`):** We categorize the following types of generalization that involve either a completely unseen object category or an unseen skill into this category. These are very challenging tests of generalization.

  - **object category completely unseen (`SGa`):** This includes scenarios where a particular object category e.g. microwave is never seen in training

– **skill completely unseen (`SGb`):** This includes scenarios where a particular skill e.g. re-orientation is never seen in any context during training.

Note that our formalization of generalization is centered around objects being interacted with and the skills that are possible for interaction, and we do not consider scene variations of the background in the definitions, unlike some prior work [17, 22, 27, 101, 171]. However, for experiments, we consider diverse scenes, both for table-top manipulation and manipulation of objects in-the-wild in unseen kitchens and offices.

3.2.4 Baselines and Ablations:

We consider a goal-conditioned behavior cloning baseline (BC) trained on all the robot data (∼1600 trajectories). The architecture of the policy is a transformer similar to our translation model without the conditioning on human-interaction-plans. The next baseline (MP) uses paired human-robot data, and is an adaptation of [157]. We compare with VRB [5] by using the affordance model from the paper to do affordance conditioned imitation learning. We also consider a baseline that is trained entirely with passive human videos, for coarse manipulation (H2R) [13]. In addition to these, we consider variations of our translation model trained on only in-lab paired human-robot data (∼400 trajectories), only hallucinated data (∼1000 trajectories), and combined paired and hallucinated data (∼1400 trajectories).

3.2.5 Evaluating Goal-conditioned Manipulation

In this section, we evaluate `HOPMan` for robot manipulation. Given an image of a scene in the robot workspace and a goal image, we use the human-interaction-plan predictor to output a sequence of plausible hand-object masks, which are input to the translation model that performs closed-loop control for executing a sequence of actions on the robot. We evaluate across diverse unseen objects exhibiting several plausible skills, and unseen scenes in-the-wild, and tabulate success rates by aggregating over objects for each skill. We define success in terms of whether the object is brought to the desired configuration in the goal image.

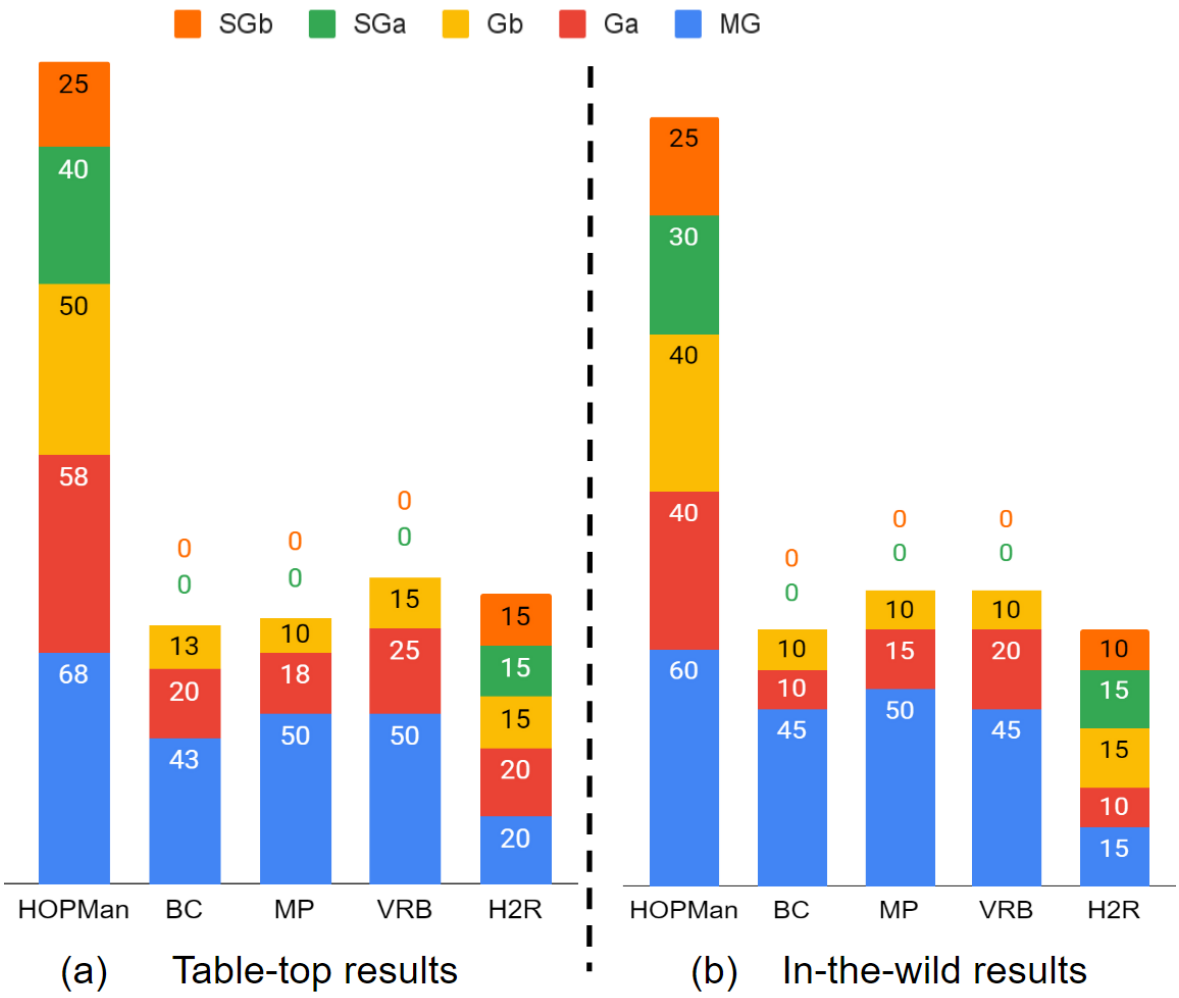

Figure 3.9: Summary of results. The numbers represent success rates for goal-conditioned evaluations, in terms of % of trials that correspond to manipulating objects in the scene to bring them to the desired goal configurations. We perform evaluations separately for the table-top manipulation and in-the-wild manipulation experiments.

Fig. 3.7 shows qualitative results for `HOPMan` where we see that the generated human-interaction-plans are plausible and correspond to manipulating the object to obtain the specified goal configuration. In Fig. 3.8 we show more robot evaluations in terms of an intermediate frame in the trajectory and the final frame reached at the end of robot evaluation, for different initial and goal images.

In Fig. 3.9 we summarize quantitative evaluations across the different generalization axes. For standard generalization `G` and strong generalization `SG`, we see that `HOPMan` achieves significantly high success rate. This demonstrates the effectiveness of learning plausible manipulation trajectories

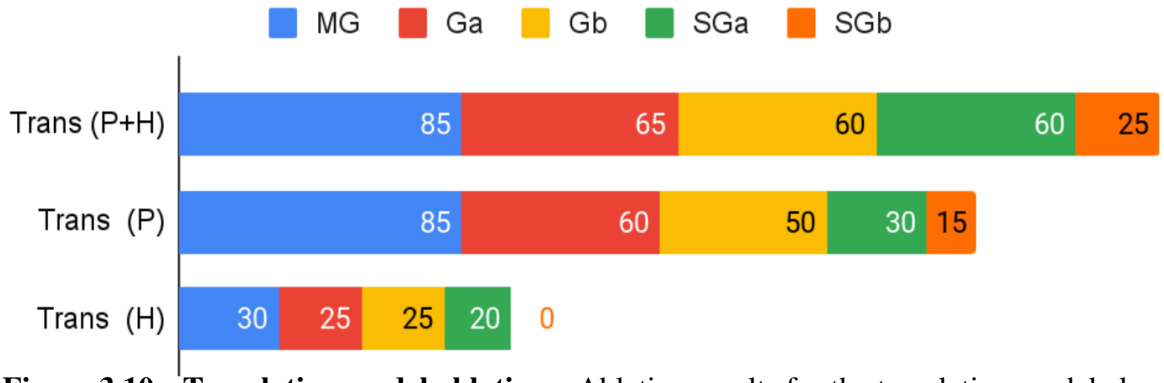

Figure 3.10: Translation model ablations. Ablation results for the translation model alone with specified masked hand-object trajectories instead of future predictions. Here, P denotes paired data, and H denotes hallucinated data, described in section 3.1.2. and the numbers represent success rates.

of hands and objects from internet videos combined with small paired data, for generalization to diverse settings, in comparison to relying on only in-domain data (BC, MP baselines), on predicting visual affordances combined with robot data (VRB) or on only passive data (H2R).

3.2.6 Ablations of the Translation Model

In this section, we evaluate the translation model in isolation independent from the prediction model. Specifically we evaluate how good is the translation model in translating the motion of a ground-truth hand-object trajectory into robot trajectories. Here, we introduce different objects in the scene and manually execute a motion with a human hand to reach the goal, and then pass the video through the hand-object segmentation model. We ablate over three variations of the the translation model, trained with paired data and hallucinated data, trained with only paired data, and trained with only hallucinated data, in table-top settings. From Fig. 3.10, we observe that training the model with combined paired and hallucinated data (P+H) leads to better performance than training with just paired data (P) indicating that the translation model is able to effectively utilize imperfect hallucinated trajectories for improving generalization.

3.3 Additional Details

3.3.1 List of Tasks

TASK		
Object Category	**Num of Instances**	**Skill**
Toaster	2	Plunging
Toaster,Toast	4	Picking
Drawer	3	Slide-Opening
Drawer	3	Slide-Closing
Toaster Oven	3	Hinge-Opening
Toaster Oven	3	Hinge-Closing
Towel	3	Swiping
Bowl	3	Pushing
French Press	2	Pouring
Bagel	3	Flipping
Tool Container	2	Pick and Place
Paper Towel	2	Unrolling
Tissue Box	2	Picking
Tea Bags	2	Picking
Spice Container	2	Pick and Place
Tea Cup	3	Pick and Place
Mug	2	Pick and Place
Ketchup Bottle	1	Pick and Place
Tea Cup	3	Dragging
Mug	2	Dragging
French Press	2	Pushing
Tool Container	2	Dragging
French Press	3	Plunging
Toaster,Toast	4	Placing
Ketchup Bottle, Wooden Board	2	Pick and Place
Spatula, Spatula Holder	2	Picking
Spatula, Spatula Holder	2	Placing
Tea Cup	3	Pouring
Glass	2	Pouring
Watermelon	2	Pick and Place
Banana	2	Pick and Place
Strainer	1	Dragging
Microwave	1	Hinge-Opening
Microwave	1	Hinge-Closing
Spatula Holder	2	Dragging
Spatula Holder	2	Pick and Place
Bun	1	Flipping
Watermelon, Wooden Board	2	Pick and Place
Box of wipes	1	Picking
Box	2	Dragging
Basket	1	Dragging
Door (vertical hinge)	1	Hinge-Opening
Door (vertical hinge)	1	Hinge-Closing
Tool Container	2	Re-orientation
Cereal, Bowl, Spoon	2	Scooping
Bowls	2	Stacking
Boxes	2	Stacking
TOTAL TASKS =	**100**	

Figure 3.11: Summary of the different tasks for table-top manipulation experiments in terms of object types, number of instantiations per object type (variations in shape, size, color ,texture) and verbs denoting the type of possible skill with each object type

3.3.2 Additional details on the models

Human-Plan Prediction model:

Instead of predicting the future in the image space, we focus on predicting only the motion of the human hand and the object being interacted with, in terms of respective semantic masks. We enable this prediction through a diffusion model trained on diverse human videos on the web. For each video $\mathcal{V}$ in the training data, we extract hand-object masks for each frame . Let $M_{1:K}$ denote the respective mask frames from time steps 1 to K. We set the value of $K = 7$ for our experiments, which amounts to choosing 7 uniformly space frames in a 2 second window of a video clip. For simplicity we consider each mask frame to be an image, where all the hand pixels are green, all the object pixels are red, and the rest of the pixels are black. the Let X_0 denote the first frame (RGB) of the video, and X_g denote the last frame (RGB) of the video, which will act as an optional goal frame. The diffusion model operates at a resolution of 64x64 for the predicted masked frames.

In the forward diffusion process, all the mask frames $M_{1:K}$ are corrupted by incrementally adding noise, and converging to a unit Gaussian distribution $N(\mathbf{0}, \mathbf{I})$. New samples can be generated by reversing the forward diffusion process, by going from Gaussian noise back to the space of mask frames. To solve the reverse diffusion process, we need to train a noise predictor $\epsilon_\theta(\cdot|t)$ which is a time-conditioned U-net trained to predict the noise at each step of the diffusion process. The input to the network at step t of the diffusion process is a channel-wise concatenation of the conditioning frames and noisy mask frames $[\mathbf{X_0}, \mathbf{X_g}, \mathbf{M^t_{1:K}}]$, and the output is the predicted noise of same dimensionality as the input. The training objective is as follows:

$$\mathbb{E}_{t,[\mathbf{X_0},\mathbf{X_g},\mathbf{M_{1:K}}]\sim p_{\text{train}},\epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[||\epsilon - \epsilon_\theta(\sqrt{\overline{\alpha}_t}\mathbf{M_{1:K}} + \sqrt{1-\overline{\alpha}_t}\epsilon|\mathbf{X_0}, \mathbf{X_g}, t)||^2\right]$$

Here α is a constant hyper-parameter that depends on the noise schedule of the diffusion process. The architecture of the U-Net for the Diffusion model is based on prior works [130, 155], and it uses a combination of 2D convolutions, multi-head self-attention layers, and adaptive group-norm. The noise levels ($p \in [0, 1]$) use positional encodings that are adapted to the correct dimensionality for each residual block through fully connected layers. The

individual residual blocks in the U-Net consist of GroupNorm, conv layers, fully connected layers, and dropout, and follow the architecture in [155].

For training the prediction model we obtain 2 second video clips from EpicKitchens [30] and Ego4D [53]. To obtain ground-truth hand-object masks, we use Visor annotations [31] for EpicKitchens and an off-the-shelf predictor [173] for obtaining the masks from Ego4D videos. In total, we curate around 150,000 video clips for training. The prediction model takes about 70 hours to train for 250,000 iterations on 8 2080Ti GPUs with a batch size of 64, and learning rate 1e-5.

The Translation model

The translation model is a transformer that is conditioned on the outputs of the future prediction model $M_{1:K}$ and for each observation O_t, predicts actions for H steps in the future. The model behaves as a closed-loop policy that is queried at each time-step t during deployment. The horizon lengths for each trajectory is 40, and we predict for $H = 10$ horizon at each time-step. The observations are of resolution 224x224, and we process them with ResNet18 backbones to obtain features. We upsample the predicted masks from 64x64 to to 224x224 dimension images and process them also with ResNet18 CNNs. At each time-step we feed in a history of 3 steps, i.e. the past two observations and actions, and the current observation. The actions are of dimension 8 (7 for joint positions, and the 8th dimension for end-effector open/close). We directly predict target joint positions instead of delta positions, as shown to be helpful by recent work [174]. The transformer encoder has 4 self-attention blocks, and the decoder has 7 cross-attention blocks, and the hidden dimensions are of size 512. We use a learning rate of 1e-5, batch size of 32, and dropout 0.1.

3.3.3 Hallucinated Data Generation Details

To augment the paired demonstrations, we develop an approach of using (more easily collectable) robot-only data. Given a robot trajectory video, we want to obtain a corresponding video where the robot in the scene is replaced by a human hand. To obtain such hallucinated pairs, we convert videos of robot trajectories into videos of human trajectories through recent advances in hand

in-painting techniques [129, 167]. Given a robot trajectory, we first obtain robot masks per frame by bringing the robot to the specific joint position per-timestep in MuJoCo simulation [153]. Based on the per-frame robot masks, we inpaint each image in the trajectory to remove the robot from the scene, using an off-the-shelf inpainting model [129]. After we have removed the robot from each frame of the trajectory, we want to place a human hand where the robot end-effector used to be in each image. To do this, we perform guided in-painting of a plausible human hand around the location of the robot end-effector in the scene, using the approach in [167]. So finally, a video of a robot trajectory is thus converted to a video of an approximate human trajectory, such that the robot arm is replaced with a human hand at approximately the location where the end-effector used to be.

3.3.4 Baselines and Ablations

We consider a goal-conditioned behavior cloning baseline that is not conditioned on the predicted masks, and is directly trained on all the robot data collected in-lab ($\sim$1400 trajectories). For the in-the-wild experiments, we additionally fine-tune the model with the 200 paired trajectories collected for these experiments. The architecture of the policy is a transformer similar to our translation model without the conditioning on hand-object masks, and keeping everything else the same.

We consider another baseline that uses paired in-lab human-robot data, to be an adaptation of MimicPlay [157] . We train the latent planner model of MimicPlay (MP) with the human-data in the paired data of 400 trajectories we have collected for the experiments. For the in-the-wild experiments, we additionally fine-tune the model with the 200 paired trajectories collected for these experiments. Note that in the original paper [157], there are a limited number of tasks (14) and human hand data is collected for 10 minutes per scene. In comparison, our paired data of 400 trajectories is much smaller and encompass around 40 tasks, since we focus mostly on learning from out-of-domain passive human videos from the web. We cannot use this large passive data for MimicPlay baseline as their framework relies on having the human videos in the exact same setup as the robot teleop data.

We compare with two baselines that use passive human videos in different

ways. The first comparison is with VRB [5] by using the affordance model from the paper to do affordance conditioned imitation learning. The second comparison is a baseline that is trained entirely with passive human videos, for coarse manipulation (H2R) [13].

In addition to these, for the table-top experiments we consider variations of our translation model trained on only paired human-robot data ($\sim$400 trajectories), only hallucinated data ($\sim$1000 trajectories), and combined paired and hallucinated data ($\sim$1400 trajectories). These ablations are on the same translation model architecture, and use manually specified hand trajectories transformed to hand-object masks through [173]. We manually provide masks instead of the predictions from the human plan prediction model, in order to evaluate the translation model in isolation independent from the prediction model.

3.3.5 Table-Top Robot Experiment Setup Details

For the robot experiments, we use several everyday objects like doors, microwaves, bowls, spatulas, boxes, french presses etc. (Fig. 3.11 has the overall list of objects), a fixed Intel Realsense camera in the scene, and a Franka Emika Panda arm operated through joint position control. We do not impose any artificial constraints on the robot's motions beyond what is possible without reaching joint limits. The action space of the translation model is 8 dimensional (7 for joint controls, and the 8th dimension for open/close of the gripper) We attach a Robotiq gripper to the arm with two festo finger grippers (for flexible grasps), so the overall end-effector is a two-finger gripper. As is the convention with image goals in real-robot experiments, we evaluate success by manually inspecting proximity of the final object configuration after robot execution, with that in the corresponding goal image.

3.3.6 In-The-Wild Robot Experiment Setup Details

We use the same Franka Emika Panda arm with flexible two finger grippers as the previous table-top experiments. The only difference is that the robot is now mounted on a mobile base with four wheels that can be moved around. The same Intel Realsense camera is mounted next to the robot on the mo-

bile base. We drag the robot across different kitchen and office scenes and perform experiments with the same setup described previously. Importantly, we do not modify the scenes and directly test on existing office and kitchen scenes. Please refer to the evaluation videos on the website for the diversity of manipulation skills and behaviors we are able to demonstrate with our framework.

3.4 Discussion and Limitations

In this work, we developed a framework for learning generalizable robot manipulation by combining internet-scale human videos of everyday interactions with limited in-domain robot demonstrations. Leveraging these, our framework can accomplish diverse tasks by predicting plausible hand-object plans and translating these to the robot's embodiment. Broadly, our work is indicative of how rich out-of-domain datasets like human videos can alleviate the data paucity that greatly bottlenecks robot learning by helping learn hand-object interaction plans, and enable wide generalization of manipulation skills to unseen scenarios. While our framework does allow strong generalization to unseen tasks, these are still limited in their complexity and it would be an interesting future direction to extend our approach for tacking long-horizon tasks that requiring composing multiple skills. Moreover, our framework may struggle with dexterous manipulation tasks as recovering precise hand and finger articulations from web videos remains a challenge in computer vision.

Chapter 4

Predicting Point Track Plan from Web Videos for Robotic Manipulation

Figure 4.1: Glimpse of some of the diverse robot manipulation capabilities across physical office and kitchen scenes enabled by our framework. We learn to predict point tracks from web videos for learning interaction plans that can be used for inferring robot actions in unseen scenarios. This enables a *common* goal-conditioned policy to perform everyday tasks like closing microwaves, pulling out drawers, flipping open toasters, pouring from jars etc. Columns show first and last images of rollouts from our policy.

Going beyond hand-object interaction plans from web videos, this chapter

explores point track predictions that serve as an embodiment-agnostic space for interaction plan prediction from diverse web videos, beyond just human videos. The overall framework still enables the previously articulated *zero-shot* execution capabilities i.e. being able to execute a task out-of-the-box without requiring any test-time training through demonstrations or self-practice before solving a specified task. This is an important desiderata for the system to be repeatedly usable without any downtime, and safe to work alongside humans without performing any exploratory actions. We pursue the goal of developing such *zero-shot* robot manipulation systems that can perform a broad set of everyday tasks. In addition to being deployable *zero-shot*, to be widely accessible, we aim to make the robot manipulators generalizable to diverse offices, and kitchens in the real world.

Developing zero-shot manipulation capabilities has been attempted by prior works, through behavior cloning on robot interaction datasets [17, 22, 71, 179]. While this approach is in-principle scalable with data, collecting diverse real-world robot interaction data is challenging due to operational constraints. Indeed, recent works that have attempted to scale robot datasets, including cross-robot and cross-domain datasets [17, 44, 114] still suffer from task diversity issues and are mostly restricted to lab-like structured scenarios. Instead of learning a single-policy that can be zero-shot deployed, some recent works aimed at in-the-wild deployment have adopted the method of test-time training [4, 99]. They require either a video of a human performing the task [4] followed by online exploration, or a demo through a robot end-effector held by a human [99]. These approaches are not very convenient for diverse deployments because they require a human to solve the task first, and several hours after that for the robot to learn how to solve that exact task in the exact scene. Thus, such approaches are not *zero-shot* deployable for new tasks in new scenes.

Our insight to develop an in-the-wild manipulation strategy that is also zero-shot deployable is to factorize a manipulation policy into an *interaction-plan* that can leverage diverse large-scale video sources on the web and a residual policy that requires a small amount of embodiment-specific robot interaction data. Such a factorized structure is inspired by prior works (e.g. [10]), however, unlike hand-object masks in [10], we instantiate this *interaction-plan* in an embodiment agnostic-manner by predicting how points in an image of

the initial scene move in future frames. This choice of an *interaction-plan* is more expressive compared to hand-object masks adopted by Bharadhwaj et al. [10], as it directly captures point correspondences across time, while at the same time being easier to compute than full RGB frames [40]. Given an initial image of the scene, a goal image defining the task to be performed, and a random set of points in the initial image, we define the interaction plan to be a 2D trajectory of the locations of the points in future frames, such that the goal is achieved. Importantly, we can train this model purely from the abundantly available RGB videos on the web without any data specific to the robot embodiment, by using off-the-shelf point-tracking approaches [78] for generating the ground-truth point trajectories. For deployment in a robot's environment, we can convert the 2D interaction-plan to a sequence of 3D end-effector poses, by having a depth image of the initial scene as an additional input and solving an optimization problem to obtain rigid transforms of the object being manipulated. Finally, with a small amount of embodiment-specific robot interaction data for different tasks (~ 400 trajectories overall), we can learn a goal-conditioned residual policy that corrects for errors in the predicted plan at each time-step and allows for closed-loop deployment.

In summary, this chapeter's contributions are three-fold:

- We develop a framework for predicting embodiment-agnostic *interaction-plans* in the form of point tracks from diverse web videos.
- We show how the interaction-plan prediction model can be used for obtaining 3D rigid transforms in a robot's environment for zero-shot manipulation without using any robot data or online exploration.
- Given a few (~ 400) embodiment-specific task demonstrations, we show how to learn a goal-conditioned residual policy that can correct for errors in the predicted plan at each time-step. The interaction-plan prediction model combined with the residual policy correction can then be used for zero-shot closed-loop deployment for new tasks in new scenes.

Our real-world robot manipulation results with a Spot robot (highlighted in Fig. 5.1) show broad generalization across diverse tasks involving unseen objects in unseen scenes, and demonstrate the potential for leveraging easily

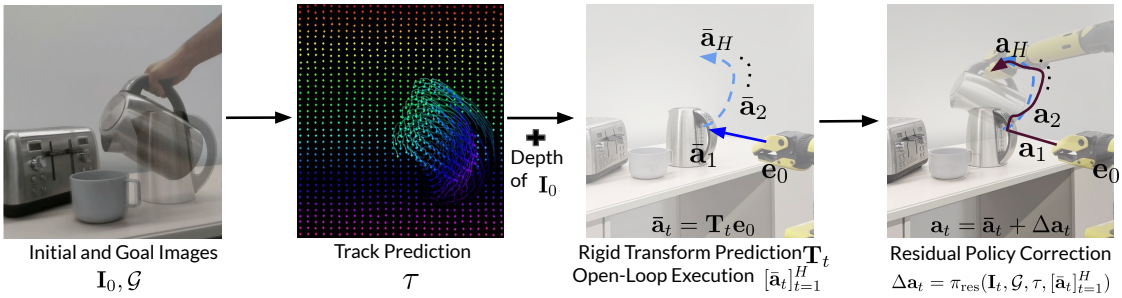

Figure 4.2: Illustration of the pipeline for learning track prediction from web video datasets, inferring rigid transforms of objects based on the predicted tracks in a robot's environment, and fine-tuning with a residual policy learned with limited robot data. This approach allows us to learn a single goal-conditioned policy for diverse (unseen) tasks.

available passive videos on the web for learning embodiment-agnostic interaction plans. This is significant as it enables zero-shot robot manipulation with a *common* goal-conditioned policy, that generalizes to unseen tasks without requiring collection of large scale in-domain manipulation datasets.

4.1 Approach

We aim to develop a zero-shot robot manipulation system that can scalably leverage diverse video data for generalizable real-world manipulation. Our key insight (Fig. 4.2) is to have a factorized policy for 1) learning embodiment-agnostic *interactions plans* of how points in an image of a scene should move in subsequent time-steps to realize a specified goal, followed by 2) inferring robot actions based on the interaction plan through a residual policy. We show how this approach allows us to generalize to diverse scenarios involving unseen tasks and objects, since the prediction model by virtue of being trained on web data generalizes well to new scenes, and the residual policy has a much simpler task of correcting the robot actions derived from the interaction plan.

4.1.1 Overview and Setup

Given a scene specified by an RGB image $\mathbf{I_0}$ and a goal image $\mathcal{G}$ denoting what task should be performed, we want to have a robot manipulator execute

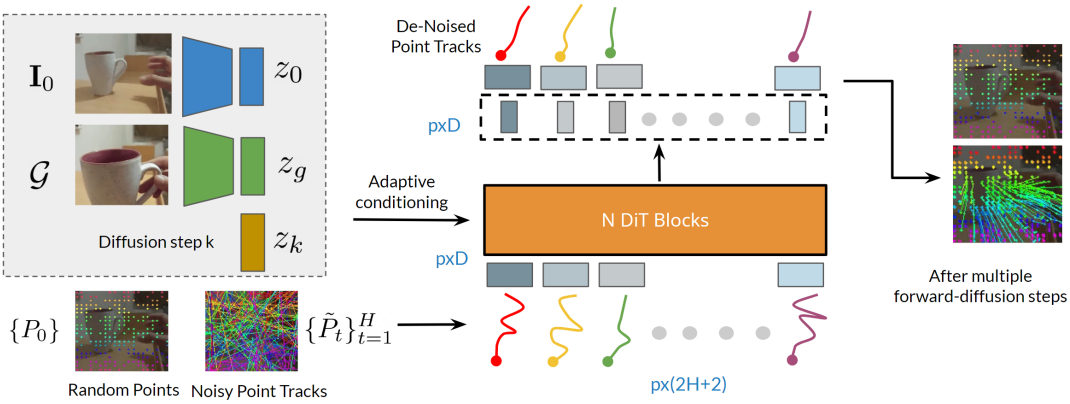

Figure 4.3: Architecture of the Diffusion Transformer $\mathcal{V}_\theta$ for denoising track predictions given initial image $\mathbf{I_0}$, goal $\mathcal{G}$, and an initial set of p points P_0.

actions $\mathbf{a}_{1:H}$ in the scene to achieve the desired goal. To achieve this in unseen scenarios, we leverage web video data by learning a model $\tau = \mathcal{V}_\theta(\mathbf{I_0}, \mathcal{G}, P_0)$ to predict future locations (tracks) of p random points P_0 in the initial image. Given a depth image for the initial frame, we leverage a subset of the predicted tracks τ_{obj} (corresponding to moving points) to infer rigid-transforms of the object being manipulated $[\mathbf{T}_t]_{t=1}^H$ and show that these allow obtaining an open-loop plan in the form of robot end-effector poses $[\bar{\mathbf{a}}_t]_{t=1}^H$. Finally, we consider training a closed-loop residual policy $\pi_{\text{res}}(\mathbf{I}_t, \mathcal{G}, \tau, [\bar{\mathbf{a}}_t]_{t=1}^H)$ that corrects the open-loop action sequence $[\bar{\mathbf{a}}_t]_{t=1}^H$ by predicting residual actions at each timestep $\Delta\mathbf{a}_t$, such that the executed action sequence is $[\mathbf{a}_t = \bar{\mathbf{a}}_t + \Delta\mathbf{a}_t]_{t=1}^H$. In the subsequent sections, we explain the architecture and algorithm design for each of the three stages in our approach.

4.1.2 Point Track Prediction from Web Videos

We instantiate track prediction as a denoising process through a DiT based diffusion model [120]. Let $\mathbf{I_0}$ denote the first frame of a video, and $\mathcal{G}$ denote the goal, which we consider to be the last frame of the video. For longer videos, we obtain multiple video clips of 4-5 seconds each for training. Let there be p points in the initial frame to be tracked, such that P_0 denotes the set of those points and let H be the prediction horizon. $[P_t]_{t=1}^H$ denotes the future locations of those points in the subsequent time-steps that we want to predict. In the forward diffusion process, all the points P_t are corrupted by

Algorithm 1 Predicting Rigid Transforms from Point Tracks

procedure RIGID TRANSFORMS($\tau, \mathbf{I_0}, \mathcal{G}, P_0^{\text{3D}}, \mathbf{K}, H$)
  $\{\{(x_t^i, y_t^i)\}_{i=1}^p\}_{t=1}^H = \tau_{obj} = \texttt{filter}(\tau)$ ▷ Filter moving point tracks
  Unknown rigid transforms $[\mathbf{T}_t]_{t=1}^H$ ▷ $\mathbf{T}_t$ has dimension 3x4
  Run RANSAC on τ_{obj} to filter outliers ▷ optional
  for $t \leftarrow 1$ to H **do**
    $\mathbf{T}_t =_{\mathbf{T}_t} \sum_i^N (||x_t^i - u_t^i|| + ||y_t^i - v_t^i||)$
    where $(u_t^i, v_t^i, 1) \simeq \mathbf{K}\mathbf{T}_t P_t$ ▷ projections in homogeneous coordinates
  return $\{\mathbf{T}_t = (\mathbf{R}_t, \mathbf{t}_t)\}_{t=1}^T$

incrementally adding noise ϵ_k (k denotes the diffusion time-step), to obtain $\tilde{P}_t$, and converging to a unit Gaussian distribution $N(\mathbf{0}, \mathbf{I})$. New samples can be generated by reversing the forward diffusion process, by going from Gaussian noise back to the space of point locations. To solve the reverse diffusion process, we need to train a noise predictor $\mathcal{V}_\theta(\mathbf{I_0}, \mathcal{G}, P_0, k)$. We design a DiT Transformer based architecture [120] for $\mathcal{V}_\theta$ illustrated visually in Fig. 5.2. Different from the original DiT model, we condition on embeddings of initial (z_0) and goal (z_g) images in addition to that of the diffusion step (z_k). The input to the Transformer in each batch is a sequence of p tokens corresponding the number of points specified for tracking. The initial P_0 points are not noisy, as is the convention in training conditional diffusion models on time-series data.

We train the prediction model with web videos by considering variable number of initial points p that need to be tracked. For flexible modeling, the locations of the p points are also randomized, such that at test-time any set of points in the initial image can be specified. We do not make any assumptions on objects to be tracked or camera motions in the videos, and do not curate the training videos in any way apart from ensuring they are of 4-5 second duration. If the goal image is such that multiple objects have moved from the initial scene, or the camera has moved, the track prediction model will predict different groups of motions for different objects and also predict motions of background points to account for camera motion. However, for robot experiments, we consider only a single object to be manipulated at a time,

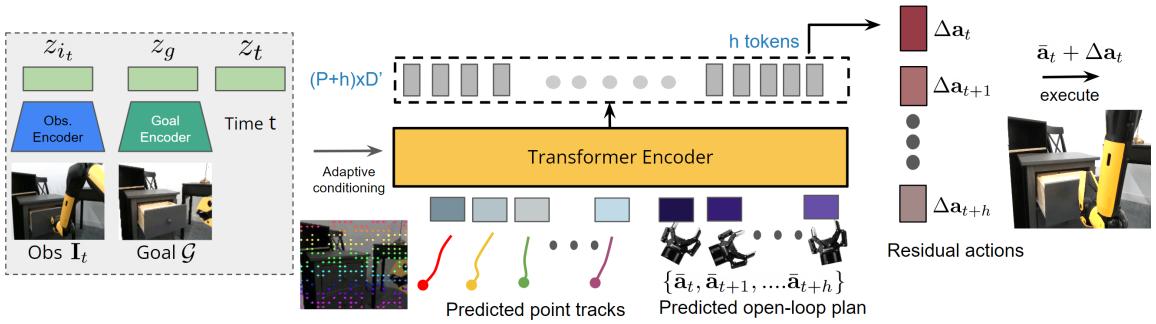

Figure 4.4: Architecture of the residual policy that predicts corrections at each time-step over the predicted open-loop plan, and enables closed-loop deployment.

which is indeed the case with several diverse real-world tasks.

4.1.3 Inferring Coarse Manipulator Trajectory from Interaction Plan

Given an image of a scene in a robot's environment $\mathbf{I}_0$, a goal $\mathcal{G}$, and a random set of points P_0 in the initial image, we can use the trained track prediction model to obtain future 2D locations of these points $\hat{P}_t$. As we consider scenarios with only a single object being manipulated under a fixed camera, only a subset of the points have a large predicted motion. We identify these p points and denote their predicted trajectories as $\tau_{\text{obj}} = [\{(x_t^i, y_t^i)\}_{i=1}^p]_{t=1}^H$. We consider the robot to be equipped with an RGBD camera, so we also have depth for the points P_0 in the first frame. Let us denote these 3D points as $P_0^{\text{3D}} = \{(x_0^i, y_0^i, z_0^i)\}_{i=1}^p$.

We seek to infer a (smooth trajectory of) per-time rigid transforms $\mathbf{T}_t$ of the object to be manipulated at time t relative to first frame, given 3D points in the first frame P_0^{3D}, predicted 2D trajectory of points on the object τ_{obj}, and the camera intrinsic matrix $\mathbf{K}$. As described in Algorithm 1, these can be obtained by ensuring that the projection of the transformed 3D points i.e. $\mathbf{K}\mathbf{T}_t P_0$ matches the predicted 2D tracks $\{(x_t^i, y_t^i)\}_{i=1}^p$ as closely as possible at each time-step t. Let $\mathbf{K}\mathbf{T}_t P_0 \simeq \{(u_t^i, v_t^i, 1)\}_{i=1}^p$. So the 2D projection of the i^{th} point at time t is (u_t^i, v_t^i). Alternatively, we have the same coordinate for the point from the predicted track i.e. (x_t^i, y_t^i). In order to determine the rigid transforms $\mathbf{T}_t$, we can solve the optimization problem in line 6 of Algorithm 1

Algorithm 2 Closed-Loop Deployment with Residual Policy Correction

procedure RESIDUAL CORRECTION($\mathcal{V}_\theta(\cdot), \mathbf{I_0}, \mathcal{G}, P_0, P_0^{\text{3D}}, \mathbf{K}, h, \mathbf{e}_0, \pi_{\text{res}}$)
  $\tau = \mathcal{V}_\theta(\mathbf{I_0}, \mathcal{G}, P_0)$ ▷ Predict future point tracks
  $[\mathbf{T}_t]_{t=1}^{H}$ = RIGID TRANSFORMS($\tau, \mathbf{I_0}, \mathcal{G}, P_0^{\text{3D}}, \mathbf{K}, h$)
  for $t \leftarrow 0$ to H **do**
    $\bar{\mathbf{a}}_t = \mathbf{T}_t \mathbf{e}_0$ ▷ Infer Open-Loop Plan
  for $t \leftarrow 1$ to H **do**
    $\Delta \mathbf{a}_{t:t+h} = \pi_{\text{res}}(\mathbf{I}_t, \mathcal{G}, \tau, [\bar{\mathbf{a}}_t]_{t=t}^{t+h})$ ▷ Predict residual correction
    $\hat{\mathbf{a}}_t = \bar{\mathbf{a}}_t + \Delta \mathbf{a}_t$ ▷ Corrected action at time t
    Execute action $\hat{\mathbf{a}}_t$ with the robot ▷ with IK and gripper action
    if t == H **then**
      Gripper Open; Reset end-effector to home

(e.g. with PnP solvers). This optimization is not under-constrained because the same 3D rigid transform $\mathbf{T}_i$ must explain the 2D motions of several points P_0 in the initial scene. Note that the obtained rigid transforms are *embodiment-agnostic* and describe how the object should move in the scene.

Now, to actually manipulate the object in the scene, we need to bring the robot end-effector[1] near the object, and optionally execute a grasp to hold on to the object, followed by transforming the end-effector based on the predicted rigid transforms $[\mathbf{T}_t]_{t=1}^{H}$. For the first step, we use a heuristic such that given initial end-effector pose $\mathbf{e}_0$ we define the first transform $\mathbf{T}_0$ to be such that the end-effector moves to the center of the 3D points $\{(x_t^i, y_t^i)\}_{i=0}^{p}$ with the same orientation as $\mathbf{e}_0$. After moving the end-effector to this pose $\mathbf{e}_1$ we execute a grasp to hold the object. We obtain subsequent end-effector poses (open-loop action trajectory) by applying the rigid transforms $\bar{\mathbf{a}}_t = \mathbf{T}_t \mathbf{e}_1$.

4.1.4 Closed-loop Manipulation with Residual Policy Correction

The open-loop execution of the predicted 3D end-effector transforms described in the last section $[\bar{\mathbf{a}}_t]_{t=1}^{H}$ might fail due to small errors in the prediction. In

[1] by end-effector we mean the part of the robot that interacts with an object

addition, since the approach does not use any embodiment-specific data, it does not have accurate information for reasoning about contact with objects and might suffer from failures like being unable to grasp the object, in spite of executing the rest of the predicted trajectory correctly. To remedy this, we propose learning a residual policy $\pi_{\text{res}}(\mathbf{I}_t, \mathcal{G}, \tau, [\bar{\mathbf{a}}_t]_{t=1}^{H})$ shown in Fig. 4.4 to correct the predicted end-effector poses in each time-step. So the end-effector pose at time t is

$$\hat{\mathbf{a}}_t = \bar{\mathbf{a}}_t + \Delta\mathbf{a}_t \;\; ; \quad \text{where} \;\; \Delta\mathbf{a}_t = \pi_{\text{res}}(\mathbf{I}_t, \mathcal{G}, \tau, [\bar{\mathbf{a}}_t]_{t=1}^{H}) \tag{4.1}$$

Instead of predicting just a single residual action $\Delta\mathbf{a}_t$ we predict residuals h steps in the future $\Delta\mathbf{a}_{t:t+h}$ and during deployment execute just the first action. This multi-step prediction has been shown to mitigate compounding errors in behavior-cloning based training [17, 174]. We can learn the residual policy with a small amount of robot demonstrations (~ 400 trajectories overall) of representative tasks through behavior cloning. The data for each trajectory consists of observation-action pairs of the form $[(\mathbf{I}_t, \mathbf{a}_t)]_{t=1}^{H}$. Here, $\mathbf{I}_t$ denotes images observed from the robot's camera and $\mathbf{a}_t$ denotes actions in the form of end-effector poses.

Crucially, since the aim of this policy is to learn only small corrections to the predicted waypoints $[\bar{\mathbf{a}}_t]_{t=1}^{H}$, we do not need to learn this policy with data from the exact scenarios that the system will be deployed in and the prediction model is expected to generalize to unseen scenarios by virtue of diverse training. The rationale is that having some embodiment-specific demonstration data in a few scenarios will help correct for the open-loop predictions from web-only data. For evaluation, we consider different levels of generalization with unseen object instances and completely unseen objects in unseen scenes.

4.2 Experiment Setup

We focus our experiments on in-the-wild manipulation scenarios where a mobile manipulator needs to manipulate objects in different living rooms, offices, and kitchens based on specified goals. For all the robot experiments, we use a Boston Dynamics Spot robot equipped with a manipulator (hand) and

a front facing Intel RealSense camera [82]. We manipulate the arm through end-effector control based on the outputs of our policy.

4.2.1 Evaluation Details

Track Prediction. For quantitative evaluation of the track prediction model, we adopt a modification of the metric developed by prior works [38, 78], δ_t^x. For evaluation videos, we consider the output of Co-Tracker [78] to be the ground-truth and compare the difference with respect to the predictions, based on the δ_t^x metric. We define δ_t^x to be the fraction of points that are within a threshold pixel distance of x of their ground truth in a time-step t. We report the *area under the curve* Δ with δ_t^x by varying x from 1 to $N = 10$ and taking the average across the prediction horizon H i.e. $\Delta = (\sum_{t=1}^{H} \sum_{x=1}^{N} \delta_t^x)/H$. Hence, Δ can vary from 0 to 1 with higher being better.

Track Prediction. As is the convention in goal-conditioned robot learning, we perform evaluations by quantifying success rate, where a successful trajectory is defined to be one where the final pose of the object in the scene to be manipulated is identical to the pose of the object in the goal image. We categorize results based on different levels of generalization, the definitions of which are inspired by prior works [10, 17, 22, 179]:

- Mild Generalization (**MG**): unseen configurations of seen object instances in seen scenes; organic scene variations like lighting and background changes
- Standard Generalization (**G**): unseen object instances in seen/unseen scenes
- Combinatorial Generalization (**CG**): unseen activity-object type combinations in seen/unseen scenes
- Type Generalization (**TG**): completely unseen object types, or completely unseen activities, in unseen scenes

4.2.2 Baselines and Comparisons

For quantitative evaluations, we first compare our track prediction approach with other related baselines and then perform comparisons with baselines for robot manipulation experiments.

Track Prediction. We perform comparisons with two baselines that have the same inputs as our track prediction model, i.e. an initial image, a goal image and points specified on the initial image, and the same output type i.e. point tracks in between the initial and goal images. We compare with a flow-based baseline that directly predicts flow between the initial and goal images, and then performs a per-timestep interpolation of the flow vectors [162]. The second baseline performs video-infilling given initial and goal images [46], and then uses Co-Tracker [78] to obtain tracks on the generated video.

Robot Experiments. We perform several comparisons with baselines and ablation studies for goal-conditioned robot manipulation. For baselines, we use the same embodiment-specific demonstrations as *Ours*, the goal-conditioned policy that predicts residuals over open-loop actions at each time-step (Algorithm 2).

- *Goal-Conditioned BC* is a baseline for multi-task policy learning, similar to prior works [17, 22, 179].
- *Affordance-Conditioned BC* is the approach from [5] that conditions the policy on predicted affordances in the initial image.
- *Video-Conditioned BC* based on [40, 46, 86] first predicts RGB video and then does tracking on top of it.
- *Hand-Object Mask Conditioned BC* from [10] conditions the policy on a predicted interaction plan consisting of masks of hands and objects.

Ours (Open Loop) is the approach for track prediction followed by open-loop execution as described in Algorithm 1. This does not use any embodiment-specific data for training. To understand the benefit of predicting residuals over actions as opposed to predicting complete actions, we compare with an ablated variant *Ours (actions; not residuals)* that predicts actions $\hat{\mathbf{a}}_t$ directly without predicting residuals $\Delta\mathbf{a}_t$ and not relying on an open-loop plan as input.

4.2.3 Training Data

For training the track prediction model, we leverage diverse passive videos available on the web that are not collected by us. Specifically, we use human video clips from EpicKitchens [30] (clipping videos to ensure they are of 4-5 seconds duration), human videos on YouTube sourced in SmthSmthv2 [52], and large-scale robot videos released in RT1 data [22] and BridgeData [156]. To obtain ground-truth tracks for training the prediction model, we run Co-Tracker [78] on the resulting 400,000 video clips. Note that the robot datasets (RT1 and Bridge) are on completely different robots and scenarios than the robot we use for experiments (Spot). For training the residual policy, the embodiment-specific data we collect consists of ~ 400 trajectories obtained by tele-operating the Spot, for solving 10 tasks of manipulating everyday objects like doors, drawers, bottles, jugs. Note that this embodiment-specific data we collect is 3-4 orders of magnitude less than that what related works [17, 22, 179] require for policy learning.

4.3 Results

We present qualitative results of the predicted tracks, and robot evaluations, followed by quantitative comparisons with the metrics defined in section 4.2.1. Please refer to the supplementary zip for detailed qualitative results and robot evaluation videos.

4.3.1 Point Track Prediction Results

We first look at some qualitative results of the track prediction model in different unseen scenes. In Fig. 4.7 we show visualization of track predictions on unseen initial and goal images across diverse datasets. We choose points on a grid in the initial frame, as shown in the third row. The prediction model is conditioned on the initial image, the goal image, and the set of points in the initial image whose future tracks are to be predicted. We can see that the predictions (shown in the fourth row) are plausible and correspond to manipulating the objects in the scene as described by the respective goal images. We can also see that when multiple entities (e.g. human and object or

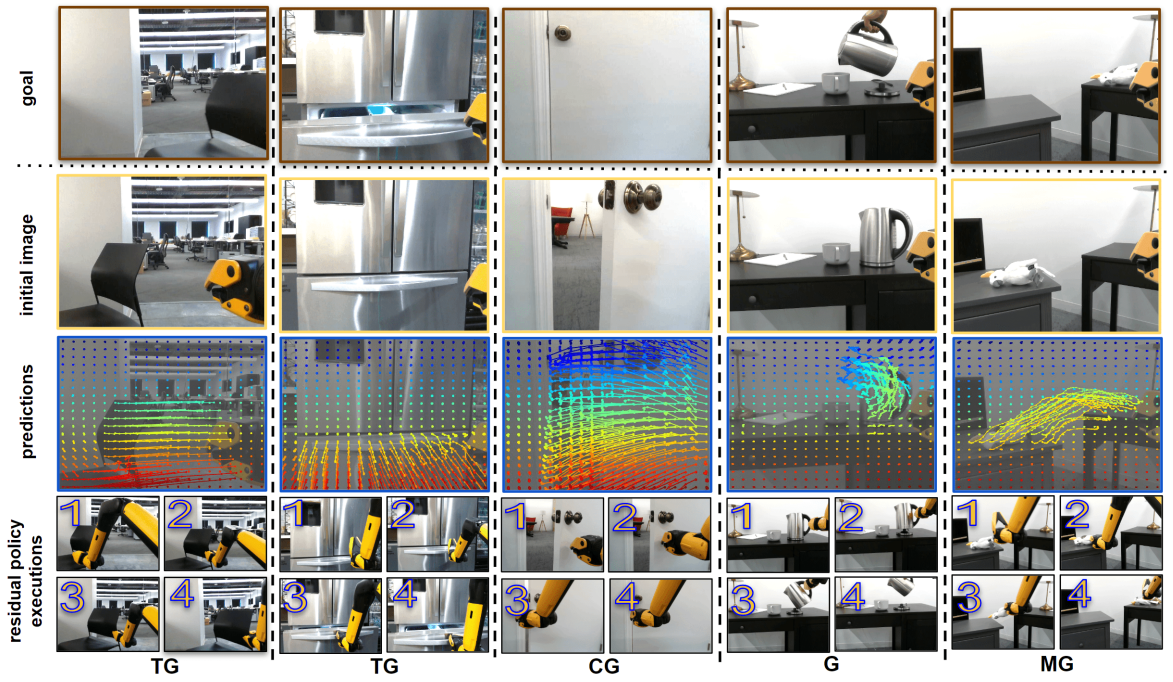

Figure 4.5: We show visualizations of point track predictions for different tasks, followed by closed-loop execution with the residual policy. We can see that the predictions are plausible and the robot execution successfully realizes the predictions to complete the respective tasks specified by the goal images. The bottom row shows the generalization level for each task, defined in section 4.2.1.

Table 4.1: Evaluation of track prediction performance on held-out videos from different datasets on the web. EpicKitchens [30] and SmthSmthv2 [52] are datsets of human videos, and BridgeData [156] and RT1 data [22] are datasets of robot videos. Note that we train a *single* model that we evaluate on these different datasets. The metric Δ is defined in section 4.2.1. Higher is better and the range is from 0 to 1.

	EpicK [30]	**SSV2** [52]	**Bridge Data** [156]	**RT1 Data** [22]
Flow [162]	0.21	0.27	0.42	0.38
Video [46]	0.30	0.17	-	-
Ours	0.67	0.70	0.77	0.75

robot and object) or the camera moves between the initial and goal images, there are different sets of point tracks predicting the respective motions.

In Table 4.1 we perform evaluations for track prediction by comparing with the flow-based [162] and video-based [46] baselines. We can see that both the

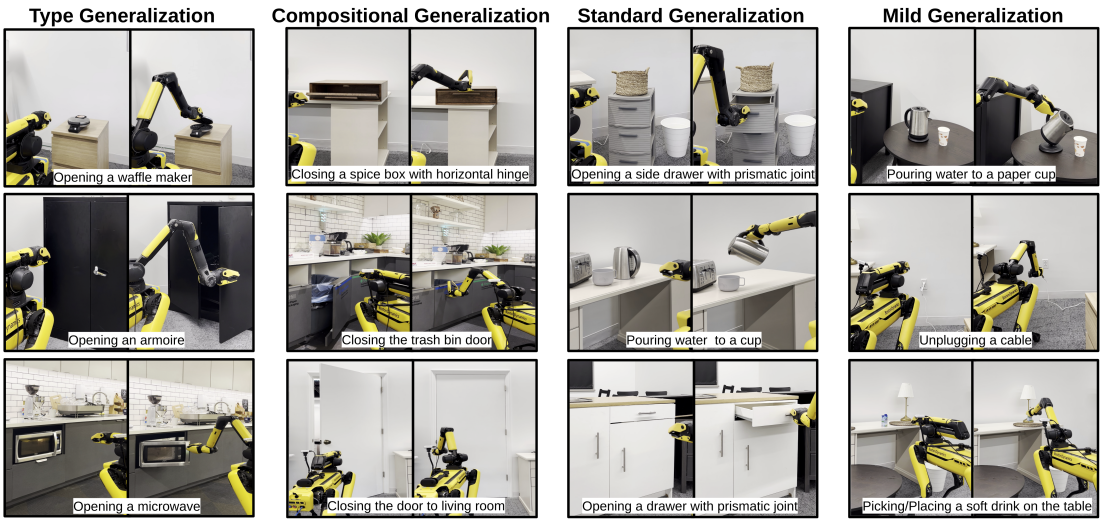

Figure 4.6: Qualitative results showing robot executions (from a third person view) with the residual policy for different tasks with respect to the generalization levels defined in section 4.2.1. We show the first and last images of a rollout. The robot executions are best viewed as videos in the supplementary zip.

baselines have much lower accuracy compared to our approach of predicting point tracks. This is because flow is too coarse to capture large non-linear state changes in between the initial and goal images. Whereas, predicting an RGB video followed by tracking suffers due to issues of implausible generation because video generation is a much more complex task than predicting the tracks of a set of points where the details about appearance, texture etc. are abstracted out. For reference, not predicting any movement for any point at all time-steps scores 0.03, 0.05, 0.36, 0.28. This suggests the benefit of directly predicting future point tracks as done by our approach if the aim is to capture motion of objects in the scene between an initial and a goal image.

4.3.2 Robot Manipulation Results

We visualize results of point track predictions using the trained prediction model in a robot's environment, followed by residual policy executions based on the predictions. In Fig. 4.5 we show the track predictions overlayed on the initial image, corresponding to the goal image shown in the top row. The bottom row shows robot execution with the first frame (1), two intermediate

frames (2,3) and the last frame (4) of a rollout. We can see that the predicted tracks correspond to points on the object moving in a way that satisfies the goal, and the policy is able to manipulate the object to the desired goal configuration. Since the camera doesn't move between the initial and goal images, we can see that background (non object) pixels remain stationary in the predictions, which is useful for accurate prediction of rigid transforms of the object.

In Table 4.2 we show comparisons for robot manipulation experiments, respectively for each level of generalization. We evaluate each approach for 20 rollouts in each level, across a total of 25 tasks in 5 different physical kitchen, office, and living room locations. We first note that our residual policy outperforms our approach for directly executing an open-loop plan based on predicted rigid transforms. This suggests that the residual policy is able to correct for inaccuracies in the open-loop plan by virtue of leveraging some embodiment-specific data that helps in performing accurate grasps on objects and recovering from potential failures during a trajectory.

We observe that for mild generalization (MG), the goal-conditioned BC baseline has slightly lower success rate compared to our residual policy, and significantly lower (or zero) success rates for standard (G), compositional (CG), and type (TG) generalization. This suggests the benefit of leveraging web video data for learning interaction plans that helps our approach generalize effectively. Finally, compared to baselines that also leverage web data like affordance-conditioned BC, video-conditioned BC, and hand-object mask-conditioned BC, we observe significant gains from our approach in the higher levels of generalization (CG and TG). This suggests that predicting static affordances without reasoning about motion trajectories, hallucinating RGB videos that suffer from incorrect generations and produce implausible artifacts in the scene, or predicting 2D masks of hands and objects without reasoning about correspondences are insufficient cues for effectively leveraging web videos. Compared to these, our interaction plan learned through track prediction provides sufficient cues for solving unseen manipulation tasks by virtue of allowing inference of 3D rigid transforms, and the residual policy helps correct for errors in the predictions.

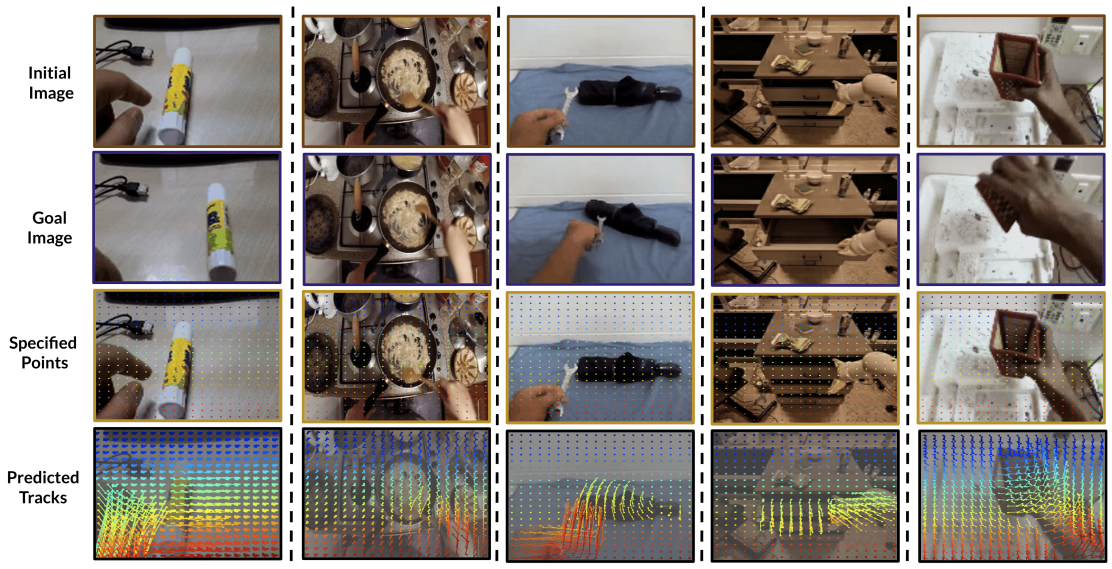

Figure 4.7: We show qualitative results of the track predictions for Track2Act on unseen initial and goal images across diverse datasets. Given specified points on the initial image we predict future tracks of these points, corresponding to the goal image. We can see that the predictions are plausible and correspond to manipulating the object(s) in the scene.

Table 4.2: Evaluation of goal-conditioned robot manipulation experiments, per the protocol described in section 4.2.1. The numbers denote success rate averaged over 20 rollouts for different tasks within each generalization axis (Higher is better). Detailed list of tasks are in the Supplementary pdf. Refer to Fig. 4.6 for visualizations of some task rollouts corresponding to each of the four generalization axes.

	MG	G	CG	TG
Behavior Cloning (BC)	60%	20%	0%	0%
Affordance-Conditioned	65%	30%	10%	5%
Video-Conditioned	60%	25%	0%	0%
Hand-Object Mask-Conditioned	70%	40%	25%	20%
Ours (Open-Loop)	35%	25%	30%	25%
Ours (Ablation; actions not residuals)	70%	45%	30%	30%
Ours	70%	60%	55%	40%

4.3.3 Analysis of Failures

Here we discuss the failures displayed by our framework. For the open-loop plan based on residual transforms, the main failure modes we observe are inability to grasp the object at the right location, and inability to recover from intermediate failures. The residual policy corrects for these behaviors by virtue of leveraging some embodiment-specific data, and thus has higher success rates. We note that the success rate for higher levels of generalization CG and TG) is still not very high since these are very challenging settings and the residual policy sometimes fails by incorrectly grasping the object, getting stuck during the execution by trying to execute a non-feasible motion, or by executing a trajectory that does not conform with the goal image specified.

Additional Details and Results

4.3.4 Additional Qualitative results

Please refer to the supplementary website `website.html` for detailed qualitative results of our framework including robot video evaluations, as well as comparisons to baselines. The video `glimpse.mp4` contains a summary of the diverse robot manipulation capabilities enabled by our framework.

4.3.5 Robot Experiment Details

We perform all robot manipulation experiments with a Boston Dynamics Spot Robot, operated through end-effector control. The robot is a quadruped with an arm attached to its base. We connect a front-facing Intel Realsense camera to the base such that it always moves with the robot, and it static with respect to the base. The end-effector of the arm is a two-fingered gripper. The horizon H of rollouts is 50 steps, and we operate the robot at a frequency of 5 Hz. For the residual policy, at each step we predict actions $h = 4$ time-steps in the future, and execute the first action. We execute the predicted actions on the robot through an Inverse Kinematics (IK) controller. This controller converts the end-effector poses to robot joint actions for appropriately manipulating the arm. We use the IK controller provided by Boston Dynamics for this purpose.

4.3.6 Track Prediction Model details

We instantiate track prediction as a denoising process through a DiT based diffusion model [120]. Let $\mathbf{I_0}$ denote the first frame of a video, and $\mathcal{G}$ denote the goal, which we consider to be the last frame of the video. For longer videos, we obtain multiple video clips of 4-5 seconds each for training. Let there be p points in the initial frame to be tracked, such that P_0 denotes the set of those points and let H be the prediction horizon. $[P_t]_{t=1}^{H}$ denotes the future locations of those points in the subsequent time-steps that we want to predict. In the forward diffusion process, all the points P_t are corrupted by incrementally adding noise ϵ_k (k denotes the diffusion time-step), to obtain $\tilde{P}_t$, and converging to a unit Gaussian distribution $N(\mathbf{0}, \mathbf{I})$. New samples can be generated by reversing the forward diffusion process, by going from Gaussian noise back to the space of point locations. To solve the reverse diffusion process, we need to train a noise predictor $\mathcal{V}_\theta(\mathbf{I_0}, \mathcal{G}, P_0, k)$. We design a DiT Transformer based architecture [120] for $\mathcal{V}_\theta$ illustrated visually in Fig. 5.2. Different from the original DiT model, we condition on embeddings of initial (z_0) and goal (z_g) images in addition to that of the diffusion step (z_k). The input to the Transformer in each batch is a sequence of p tokens corresponding the number of points specified for tracking. The initial P_0 points are not noisy, as is the convention in training conditional diffusion models on time-series data. We train the prediction model with web videos by considering variable number of initial points p that need to be tracked. We vary p from 200 to 400. For flexible modeling, the locations of the p points are also randomized, such that at test-time any set of points in the initial image can be specified. We do not make any assumptions on objects to be tracked or camera motions in the videos, and do not curate the training videos in any way apart from ensuring they are of 4-5 second duration.

The model has 24 DiT blocks, with a hidden size of 1024, and 16 heads. The ResNet18 embeddings of initial image and goal image have dimensions 512. The condition to each DiT block consists of the sum of initial image embedding, goal image embedding, and diffusion time-step embedding through adaptive modulation (adaLN) layers. The adaptive modulation layers and final MLP layers are zero-initialized, and the rest are Xavier uniform initialized. We use Adam optimizer with default Adam betas = (0.9,0.999) and a constant

learning rate of 1e-4 for experiments. The rest of the architecture and training details are similar to DiT [120].

4.3.7 Residual Policy Model details

To correct the predicted open-loop plan, with a small amount of embodiment-specific data, we propose learning a residual policy $\pi_{\text{res}}(\mathbf{I}_t, \mathcal{G}, \tau, [\bar{\mathbf{a}}_t]_{t=1}^{H})$ shown in Fig. 4.4 to correct the predicted end-effector poses in each time-step. So the end-effector pose at time t is $\hat{\mathbf{a}}_t = \bar{\mathbf{a}}_t + \Delta\mathbf{a}_t \quad ; \quad$ where $\Delta\mathbf{a}_t = \pi_{\text{res}}(\mathbf{I}_t, \mathcal{G}, \tau, [\bar{\mathbf{a}}_t]_{t=1}^{H})$ Instead of predicting just a single residual action $\Delta\mathbf{a}_t$ we predict residuals h steps in the future $\Delta\mathbf{a}_{t:t+h}$ and during deployment execute just the first action. This multi-step prediction has been shown to mitigate compounding errors in behavior-cloning based training [17, 174]. We can learn the residual policy with a small amount of robot demonstrations (~ 400 trajectories overall) of representative tasks through behavior cloning. The data for each trajectory consists of observation-action pairs of the form $[(\mathbf{I}_t, \mathbf{a}_t)]_{t=1}^{H}$. Here, $\mathbf{I}_t$ denotes images observed from the robot's camera and $\mathbf{a}_t$ denotes actions in the form of end-effector poses.

The residual policy model is a Transformer based on the DiT architecture. The model has 12 DiT blocks, with a hidden size of 512, and 8 heads. The ResNet18 embeddings of initial image and goal image have dimensions 512. The condition to each DiT block consists of the sum of current image embedding, goal image embedding, and emebedding of the current time-step t through adaptive modulation (adaLN) layers. The adaptive modulation layers and final MLP layers are zero-initialized, and the rest are Xavier uniform initialized. We use Adam optimizer with default Adam betas = (0.9,0.999) and a constant learning rate of 1e-4 for experiments. The input to the model consists of the predicted tracks of p points in the initial image (we keep $p = 400$ to ensure a dense grid in the initial image of dimensions 256x256x3) and the predicted open-loop plan with h steps from $t : t + h$. So there are $p + h$ input tokens. We read off the final h tokens corresponding to the updated open-loop plan for these h steps and after a final MLP layer, output actions for h steps . We will release all code and models upon acceptance.

4.3.8 Training Data for Track Prediction

We use four different web data sources for training the track prediction model - videos from Something-Something-v2 [52], Epic-Kitchens [30], RT1 data [22], and BridgeData [156]. Something-Something-v2 contains short YouTube videos of people doing everyday activities. We consider videos from this dataset as is, and choose the first frame as the initial image and the last frame as goal image. Epic-Kitchens contains ego-centric videos of humans in different locations performing diverse tasks in kitchens. Since these videos are long (¿20 min each), we choose clips of duration 4-5 seconds by cutting the long videos, and choosing clips where a human hand is visible in the scene (so as to have clips where an object is being manipulated, instead of a person just moving around). RT1 Data and Bridge Data are large-scale robot datasets that contains rollouts of two different types of robots being tele-operated for different tasks. For these datasets, we consider the first and last images to be the first and last frames of a rollout, and each rollout to be a separate video.

In total we obtain around 400,000 videos clips from the above sources, choose a dense grid of 400 points on the first frame and we run Co-Tracker [78] on these clips, for obtaining the ground-truth intermediate tracks of points. Our prediction model is conditioned on the first and last frames for each video, and the task of predicting the tracks of random points on the initial frame is supervised by the tracks we obtain from Co-Tracker (ground-truth).

4.3.9 Training Data for Residual Policy

For training the residual policy we collected tele-operated demonstrations with the Spot robot by controlling it with a joystick across 10 tasks in 3 physical locations. These scenarios correspond to only a subset of the diverse tasks, objects, and scenes we consider for evaluation . Concretely, the evaluation scenarios with same tasks as the collected data correspond to the *mild generalization* (MG) category. Rest of the generalization axes corresponding to unseen instances and categories are described in detail in section 4.2.1.

The training data consists of 400 teleoperated trajectories, each consisting of H (observation,action) pairs ($H = 50$). The data for each trajectory consists of observation-action pairs of the form $[(\mathbf{I}_t, \mathbf{a}_t)]_{t=1}^{H}$. Here, $\mathbf{I}_t$ denotes images

observed from the robot's camera and $\mathbf{a}_t$ denotes actions in the form of end-effector poses. This data is collected at the same frequency of 5 Hz that we deploy the policy for eventual evaluations. Note that this embodiment-specific data we collect is 3-4 orders of magnitude less than that what related works [17, 22, 179] require for policy learning. This is a major advantage of our framework as it precludes the need to spend years on real-world data collection, while achieving generalization to more diverse scenarios by virtue of leveraging *passive* web videos for track prediction.

4.3.10 Details on baselines

We perform several comparisons with baselines and ablation studies for goal-conditioned robot manipulation. For baselines, we use the same embodiment-specific demonstrations as *Ours*, the goal-conditioned policy that predicts residuals over open-loop actions at each time-step (Algorithm 2).

- *Goal-Conditioned BC* is a baseline for multi-task policy learning, similar to prior works [17, 22, 179]. This is trained with the same data we use for training our residual policy, and is conditioned on goal image, similar to our residual policy.

- *Affordance-Conditioned BC* is the approach from [5] that conditions the policy on predicted affordances in the initial image. These affordances capture what is *plausible* to be manipulated in the scene, and so are different from our time-series predictions of point tracks. We directly adopt the affordance model from [5] that was trained on web data, and use the same embodiment-specific data as our residual policy for training through conditional behavior cloning.

- *Video-Conditioned BC* based on [40, 46, 86] first predicts RGB video and then does tracking on top of it. We adopt the video prediction model from [46] (without language conditioning) trained on web data, and use the same embodiment-specific data as our residual policy for training through conditional behavior cloning.

- *Hand-Object Mask Conditioned BC* from [10] conditions the policy on a predicted interaction plan consisting of masks of hands and ob-

jects. We use the hand-object plan prediction model from [10], and use the same embodiment-specific data as our residual policy for training through conditional behavior cloning. Note that this baseline is slightly different from the translation model in [10] because we do not collect paired human-robot demonstrations unlike [10] and so the policy is conditioned on predicted hand-object plans as opposed to ground-truth plans unlike [10].

Comparison to *Goal-Conditioned BC* helps understand the potential benefits of leveraging web data for generalizable manipulation, and comparisons to *Affordance-Conditioned BC*, *Video-Conditioned BC*, *Hand-Object Mask Conditioned BC* help understand the potential of predicting point tracks from web videos, compared to other ways of using web data for prediction geared towards manipulation.
Ours (Open Loop) is the approach for track prediction followed by open-loop execution as described in Algorithm 1. This does not use any embodiment-specific data for training. To understand the benefit of predicting residuals over actions as opposed to predicting complete actions, we compare with an ablated variant *Ours (actions; not residuals)* that predicts actions $\hat{\mathbf{a}}_t$ directly without predicting residuals $\Delta\mathbf{a}_t$ and not relying on an open-loop plan as input.

4.3.11 Qualitative Results for baselines

We provide qualitative comparisons of the baselines with our approach, in the figures below. For detailed qualitative video results of our approach, please refer to the attached video and local webpage.

4.4 Discussion and Conclusion

In this paper, we developed a framework for generalizable zero-shot robot manipulation by leveraging large-scale web video data to learn embodiment agnostic plans of how objects should be manipulated in a scene to satisfy a goal. We combined this with a small amount of embodiment-specific data to learn residual corrections over the predicted plans through a closed-loop policy. Our

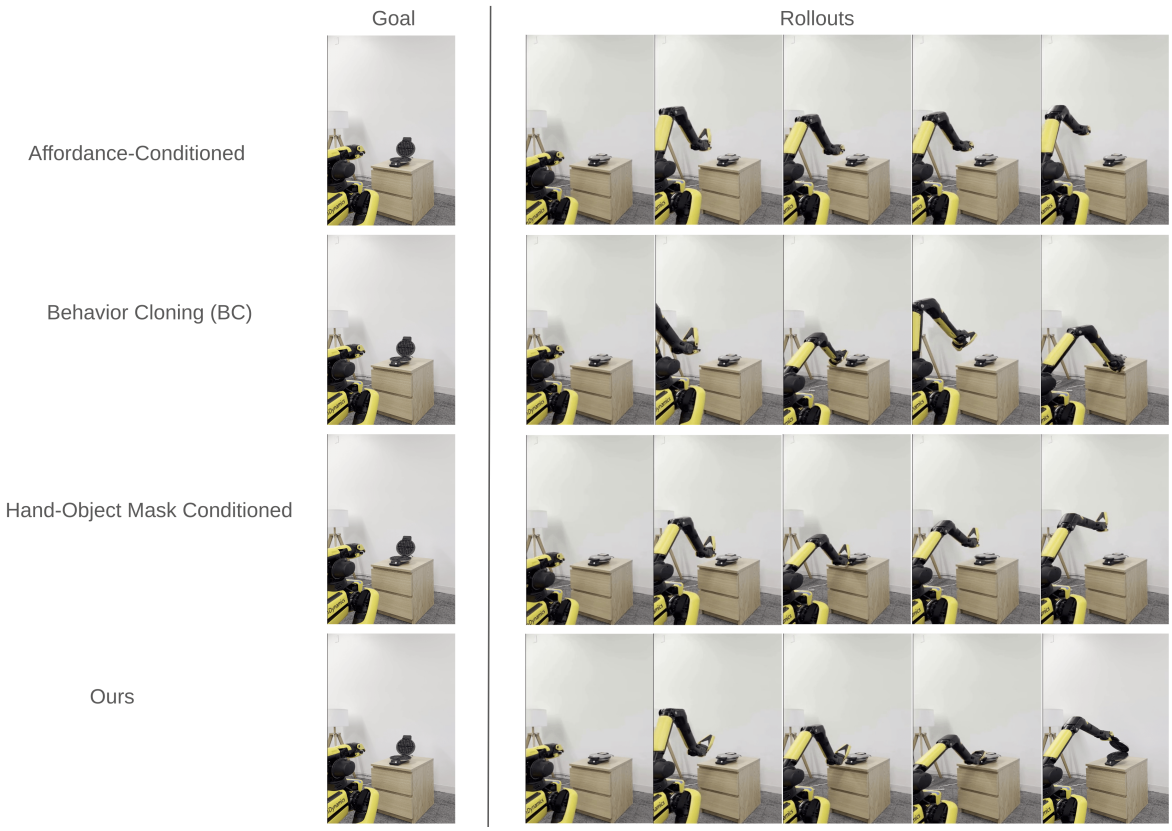

Figure 4.8: Type Generalization (TG). We show rollouts from baselines for the same goal. The views are from a third person camera.

real world manipulation results across a range of diverse tasks with varying levels of generalization demonstrate the potential of scalably leveraging web data to predict plans for object manipulation. While our framework allows for strong generalization to unseen tasks in-the-wild, the tasks are still of short-horizon and involve manipulating a single object in the scene. It would be an interesting direction of future work to extend our framework for tackling long-horizon tasks that involve successive manipulations of multiple objects in the scene.

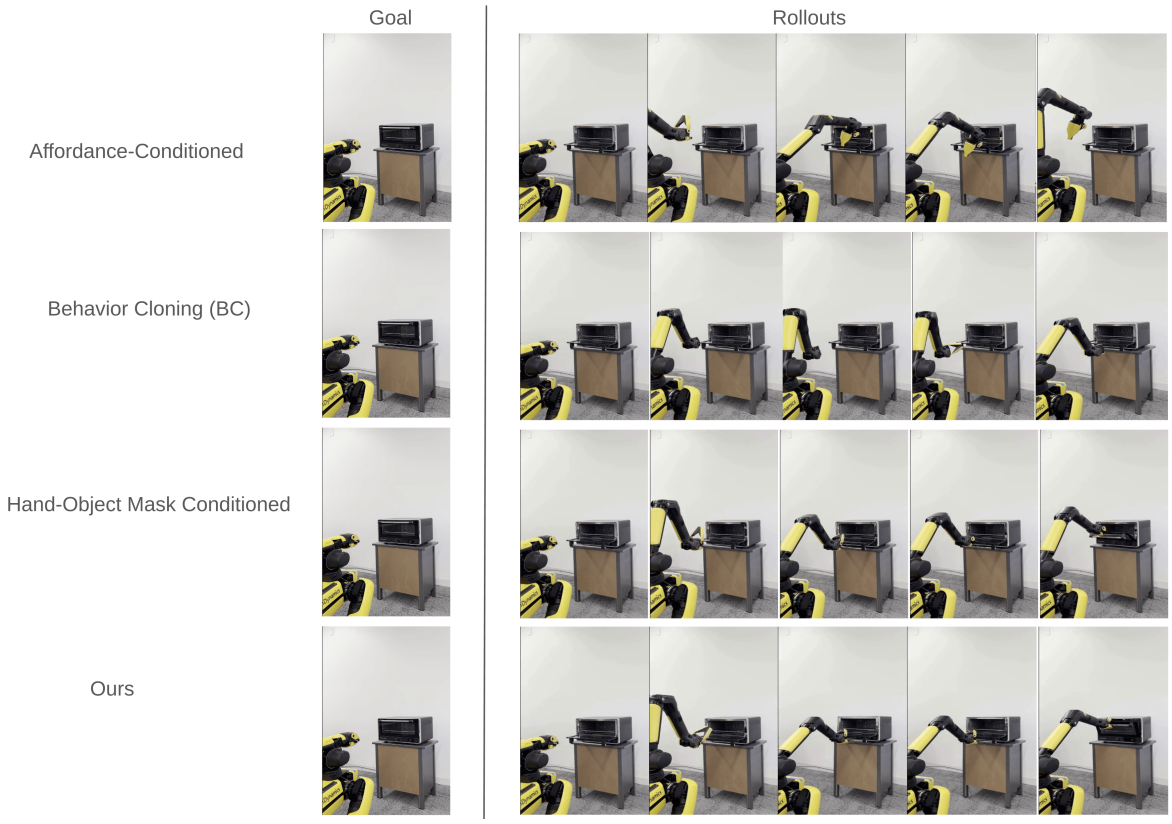

Figure 4.9: Compositional Generalization (CG). We show rollouts from baselines for the same goal. The views are from a third person camera.

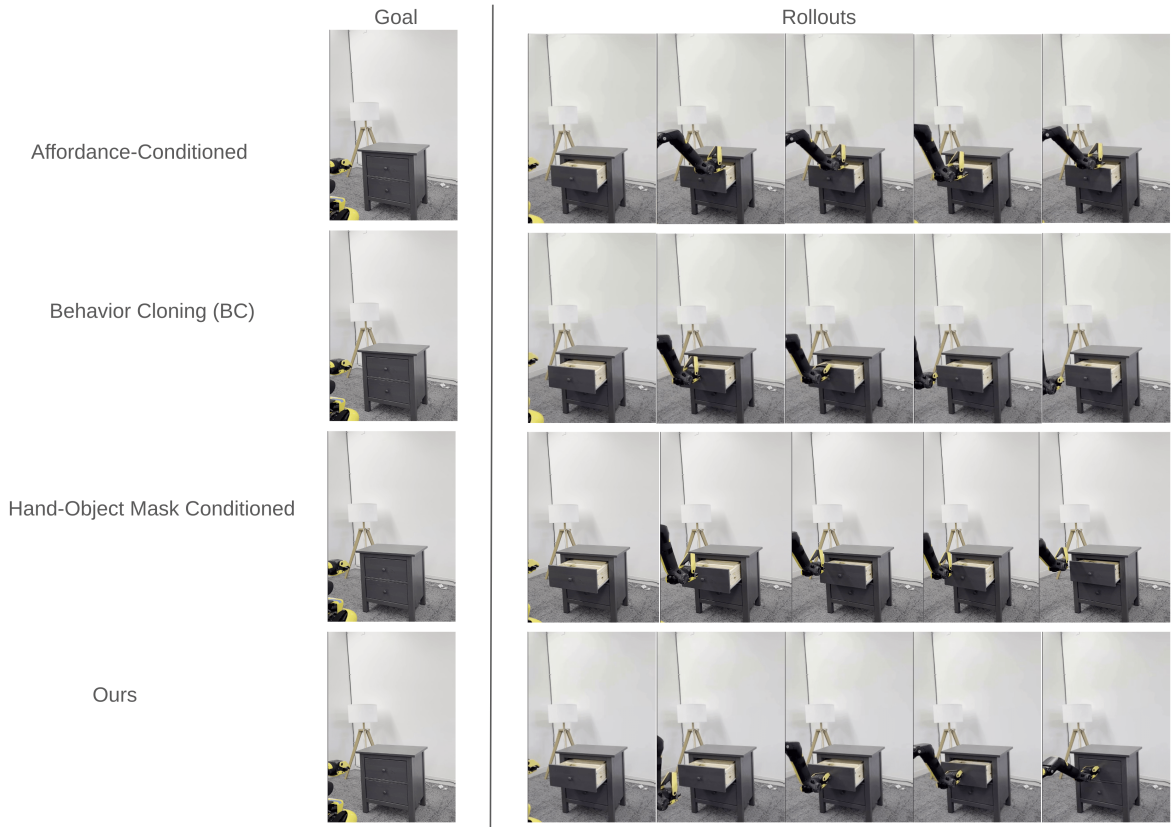

Figure 4.10: Standard Generalization (G). We show rollouts from baselines for the same goal. The views are from a third person camera.

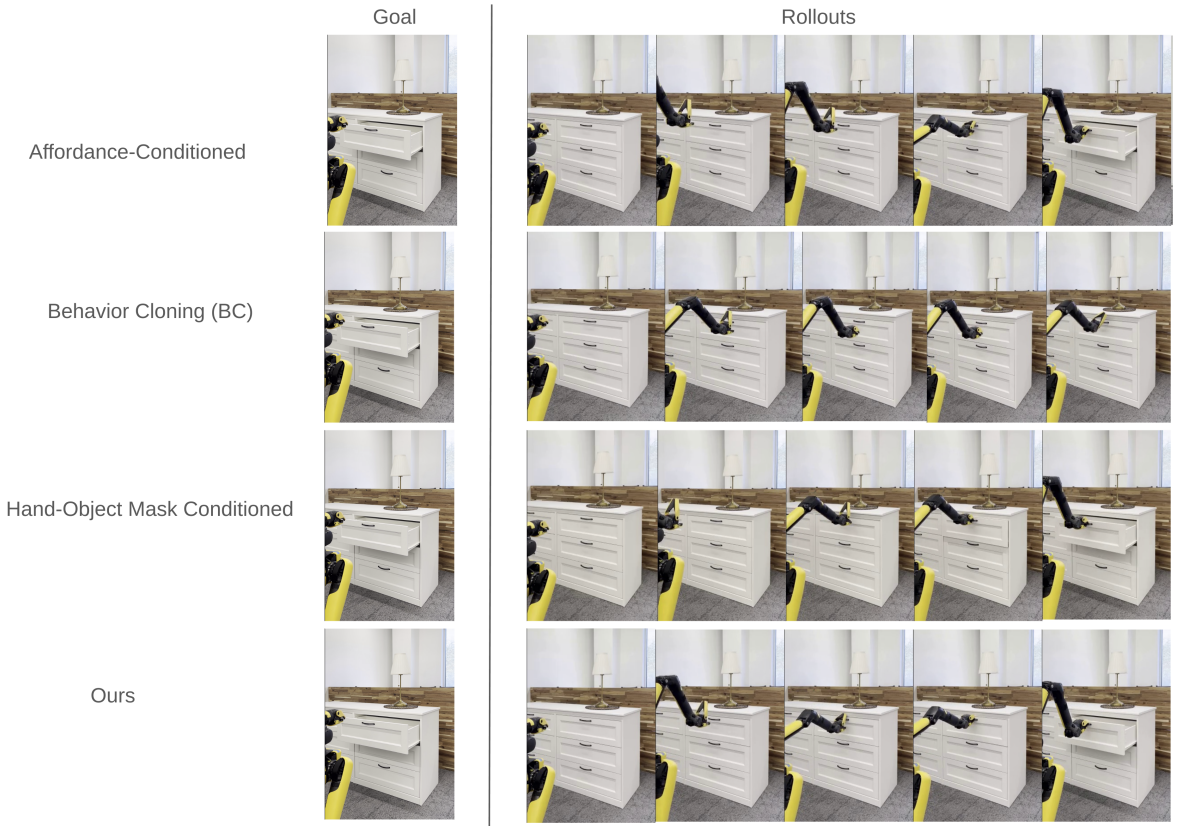

Figure 4.11: Mild Generalization (MG). We show rollouts from baselines for the same goal. The views are from a third person camera.

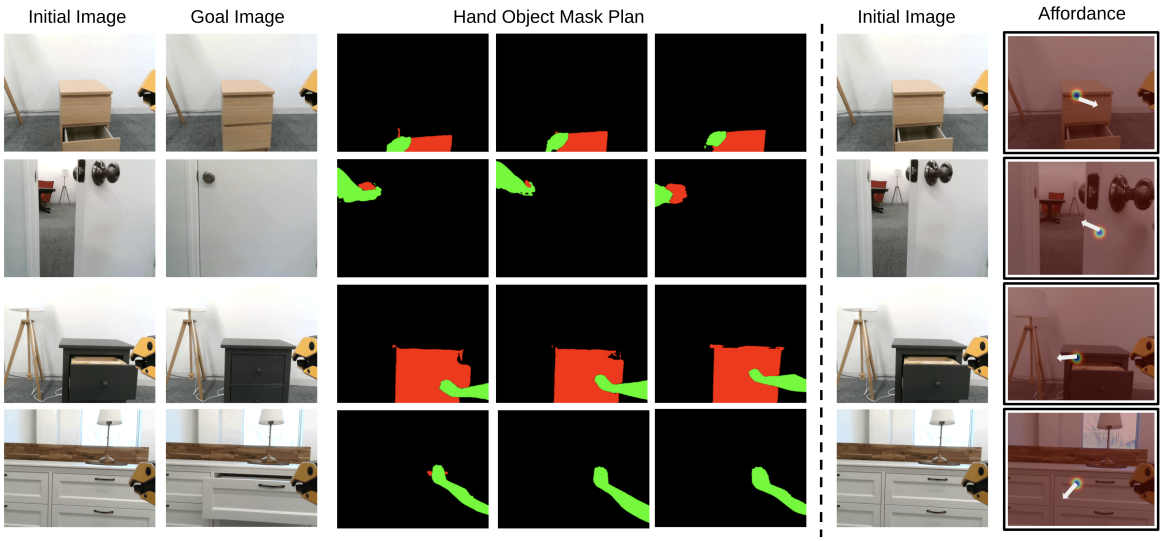

Figure 4.12: We show visualizations of predictions from the Hand-Object Mask Prediction and Affordance Prediction baselines, on different initial and goal images in the robot's environment.

Chapter 5

Zero-Shot Human Video Generation for Robot Manipulation

In order to mitigate issues with purely scaling robotic datasets, a line of recent works have sought to incorporate additional behavioral priors in representation learning by pre-training visual encoders with non-robotic datasets [79, 100, 112, 118, 160] and co-training policies with vision-language models [83, 115, 179]. Going beyond abstract representations, other works have learned attributes from web videos more directly informative of motion in the form of predicting goal images [12, 18, 77], hand-object mask plans [10], and embodiment-agnostic point tracks [15]. These approaches show promising signs of generalization to tasks unseen in the robot interaction datasets, but training such specific predictive models from web video data requires utilizing other intermediate models for providing ground-truths and thus are hard to scale up.

Our key insight for enabling generalization in manipulation is to cast motion prediction from web data in the very generic form of zero-shot video prediction. This lets us directly leverage advances in video generation models, by conditioning a robot policy on the generated video for new tasks that are unseen in the robot datasets. We posit that as video generation models get better due to large interest in generative AI [49, 87, 132] beyond robotics, an

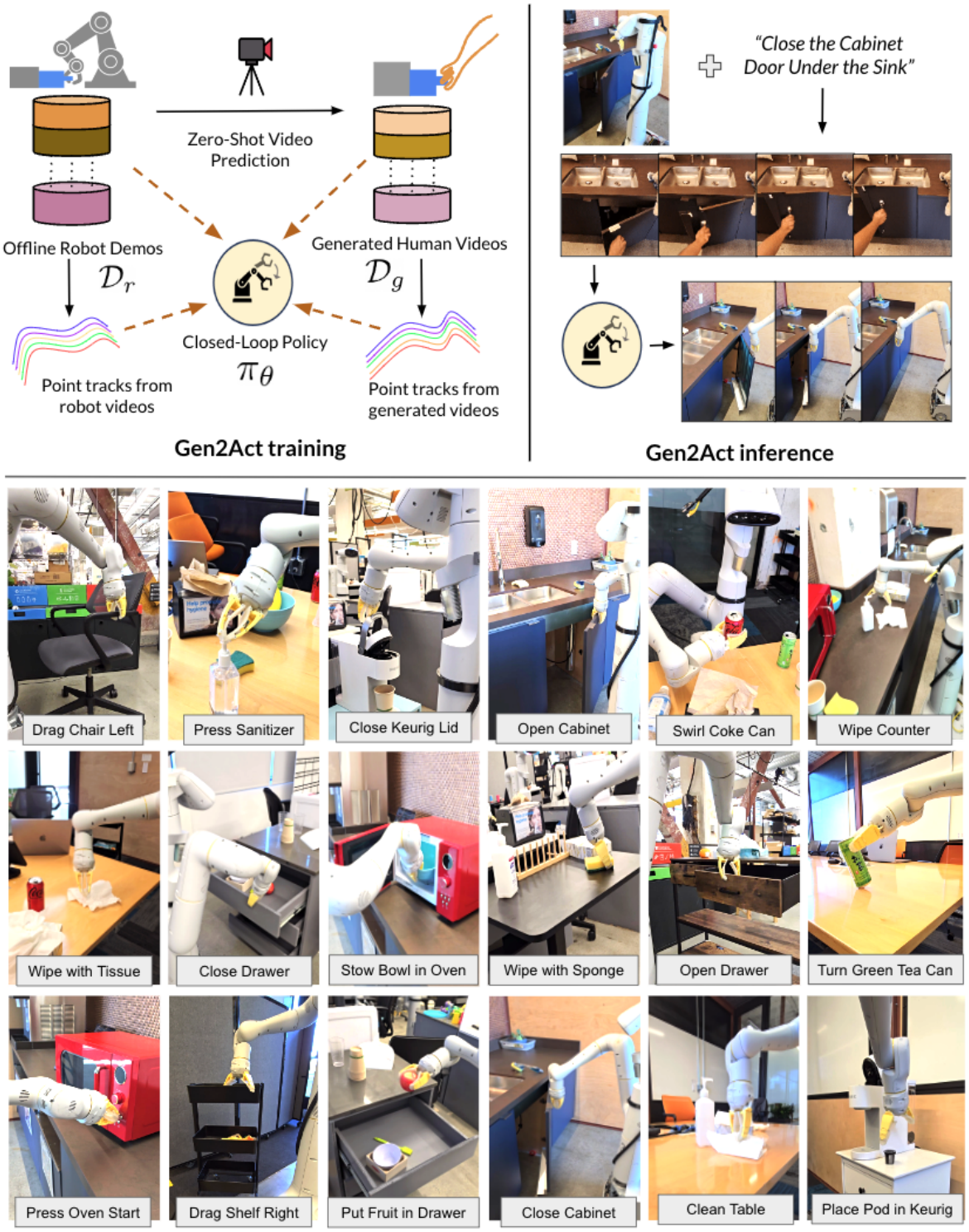

Figure 5.1: *Gen2Act* learns to generate a human video followed by robot policy execution conditioned on the generated video. This enables diverse real-world manipulation in unseen scenarios.

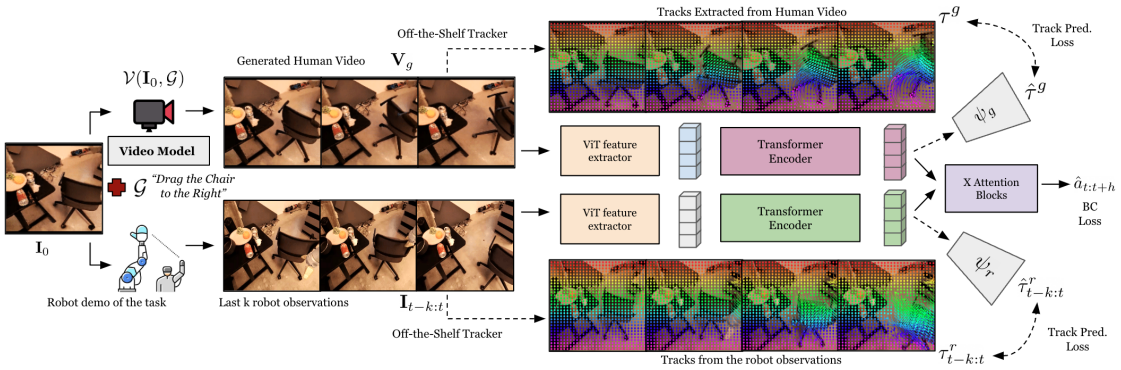

Figure 5.2: Architecture of the translation model of *Gen2Act* (closed-loop policy π_θ). Given an image of a scene $\mathbf{I}_0$ and a language-goal description of the task $\mathcal{G}$, we generate a human video $\mathbf{V}_g$ with a pre-trained video generation model $\mathcal{V}(\mathbf{I}_0, \mathcal{G})$. During training of the policy, we incorporate track prediction from the policy latents as an auxiliary loss in addition to a behavior cloning loss. Dotted pathways show training-specific computations. During inference, we do not require track prediction and only use the video model $\mathcal{V}$ in conjunction with the policy $\pi_\theta(\mathbf{I}_{t-k:t}, \mathbf{V}_g)$.

approach that relies on learning a policy conditioned on zero-shot video prediction can effectively scale and generalize to increasingly diverse real-world scenarios. For performing a manipulation task in a novel scene, a generated video conditioned on the language description of the task is particularly useful for conveying *what* needs to be done and in capturing motion-centric information of *how* to perform the task that can then be converted to robot actions through a learned policy. Compared to a generated video, a language description or a goal image alone only conveys what the task is.

We develop *Gen2Act* by instantiating language-conditioned manipulation as human video generation followed by generated human video to robot translation with a closed-loop policy (Figure 5.1). We opt for generating human videos as opposed to directly generating robot videos since video generation models are often trained with human data on the web, and they are able to generate human videos zero-shot given a new scene. We then train a translation model that needs some offline robot demonstrations and corresponding generated human videos. We generate these corresponding human videos offline with an off-the-shelf model [87] by conditioning on the first frame of each trajectory (the first frame doesn't have the robot in the scene) and the language description of the task. We instantiate this translation model as a closed loop policy that is conditioned on the history of robot observations in addition to the generated human video so that it can take

advantage of the visual cues in the scene and adjust its behavior reactively.

In order to capture motion information beyond that implicitly provided by visual features from the generated video, we extract point tracks from the generated human video and the video of robot observations (through an off-the-shelf tracker [39]) and optimize a track prediction auxiliary loss during training. The aim of this loss function is to ensure that the latent tokens of the closed-loop policy are informative of the motion of points in the scene. We train the policy to optimize the typical behavior cloning loss for action prediction combined with this track prediction loss. For deployment, give a language description of a task to be performed, we generate a human video and run the policy conditioned on this video.

The diverse real-world manipulation results of *Gen2Act* (featured in Figure 5.1) demonstrate the broad generalization capabilities enabled by learning to infer motion cues from web video data through zero-shot video generation combined with motion extraction through point track prediction for solving novel manipulation tasks in unseen scenarios. For generalization to novel object types and novel motion types unseen in the robot interaction training data, we show that *Gen2Act* achieves on average $\sim 30\%$ higher absolute success rate over the most competitive baseline. Further, we demonstrate how *Gen2Act* can be chained in sequence for performing long-horizon activities like "making coffee" consisting of several intermediate tasks.

5.1 Approach

We develop a language-conditioned robot manipulation system, *Gen2Act* that generalizes to novel tasks in unseen scenarios. To achieve this, we adopt a factorized approach: 1) Given a scene and a task description, using an existing video prediction model generate a video of a human solving the task, 2) Conditioned on the generated human video infer robot actions through a learned human-to-robot translation model that can take advantage of the motion cues in the generated video. We show that this factorized strategy is scalable in leveraging web-scale motion understanding inherent in large video models, for synthesizing *how* the manipulation should happen for a novel task, and utilizing orders of magnitude less robot interaction data for the much

simpler task of translation from a generated human video to *what* actions the robot should execute.

5.1.1 Overview and Setup

Given a scene specified by an image $\mathbf{I}_0$ and a goal $\mathcal{G}$ describing in text the task to be performed, we want a robot manipulation system to execute actions $\mathbf{a}_{1:H}$ for solving the task. To achieve this in unseen scenarios, we learn motion predictive information from web video data in the form of a video prediction model $\mathcal{V}(\mathbf{I}_0, \mathcal{G})$ that zero-shot generates a human video of the task, $\mathbf{V}_g$. In order to translate this generated video to robot actions, we train a closed-loop policy $\pi_\theta(\mathbf{I}_{t-k:t}, \mathbf{V}_g)$ conditioned on the video and the last k robot observations, through behavior cloning on a small robot interaction dataset $\mathcal{D}_r$. In order to implicitly encode motion information from $\mathbf{V}_g$ in the policy π_θ, we extract point tracks from both $\mathbf{V}_g$ and $\mathbf{I}_{t-k:t}$, respectively τ_g and τ_r, and incorporate track prediction as an auxiliary loss $\mathcal{L}_\tau$ during training. Figure 5.2 shows an overview of this setup.

5.1.2 Human Video Generation

We use an existing video generation model for the task of text+image conditioned video generation. We find that current video generation models are good at generating human videos zero-shot without requiring any fine-tuning or adaptation (some examples in Fig. 5.3). Instead of trying to generate robot videos as done by some prior works [40, 92], we focus on just human video generation because current video generation models cannot generate robot videos zero-shot and require robot-specific fine-tuning data for achieving this. Such fine-tuning often subtracts the benefits of generalization to novel scenes that is inherent in video generation models trained on web-scale data.

For training, given an offline dataset of robot trajectories $\mathcal{D}_r$ along with language task instructions $\mathcal{G}$, we create a corresponding generated human video dataset $\mathcal{D}_g$ by generating videos conditioned on the first frame of the robot trajectories and the language instruction. This procedure of generating paired datasets $\{\mathcal{D}_r, \mathcal{D}_g\}$ is fully automatic and does not require manually collecting human videos as done by prior works [68, 117]. We do not require

Figure 5.3: Visualization of zero-shot video generation for different tasks. The blue frame and the language description are input to the video generation model of *Gen2Act* and the black frames show sub-sampled frames of the generated video. These results demonstrate the applicability of off-the-shelf video generation models for image+text conditioned video generation that preserves the scene and performs the desired manipulation task.

the generated human videos to have any particular structure apart from looking visually realistic, manipulating the relevant objects plausibly, and having minimal camera motion. As seen in the qualitative results in Figure 5.3, all of this is achieved zero-shot with a pre-trained video model.

During evaluation, we move the robot to a new scene $\mathbf{I}_0$, specify a task to be performed in language $\mathcal{G}$, and then generate a human video $\mathbf{V}_g = \mathcal{V}(\mathbf{I}_0, \mathcal{G})$ that is fed into the human-to-robot translation policy, described in Section 5.1.3. Our approach is not tied to a specific video generative model and as video models become better, this stage of our approach will likely scale upwards. We expect the overall approach to generalize as well since the translation model is tasked with a simpler job of inferring motion cues from the generated human video in novel scenarios, and implicitly converting that to robot actions.

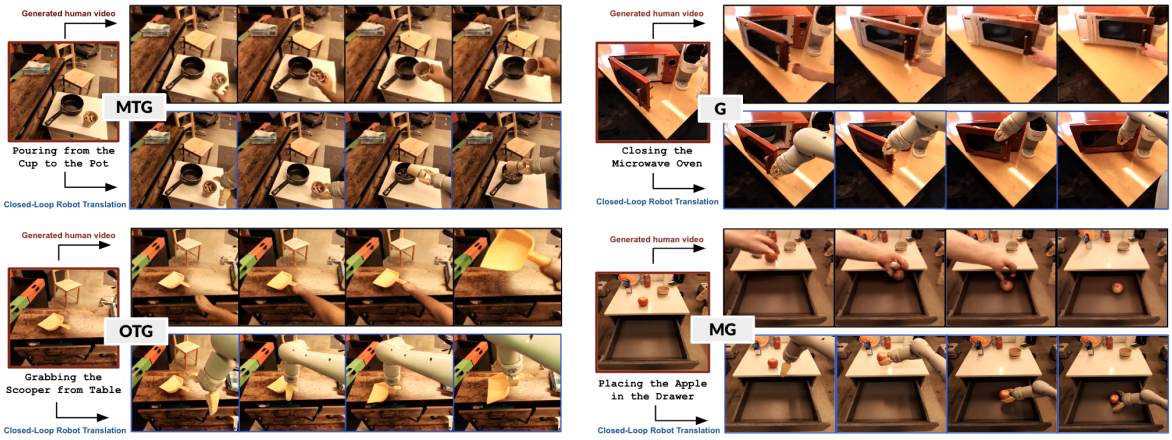

Figure 5.4: Visualization of the closed-loop policy rollouts (bottom row) conditioned on the generated human videos (top row) for four tasks. The red frame and the language description are input to the video generation model of *Gen2Act* . The black frames show sub-sampled frames of the generated video, and the blue frames show robot executions conditioned on the generated video.

As we show through results in Section 5.1.3 only a small amount of diverse robot trajectories (~ 400) combined with existing offline datasets is enough to train a robust translation model.

5.1.3 Generated Human Video to Robot Action Translation

We instantiate generated human video to robot action translation as a closed loop policy π_θ. Given a new scene and a task description, the generated human video provides motion cues for how the manipulation should happen in the scene, and the role of the policy is to leverage relevant information from the generated video, combined with observations in the robot's frame, for interacting in the scene. Instead of attempting to explicitly extract waypoints from the generated video based on heuristics, we adopt a more end-to-end approach that relies on general visual features of the video, and general point tracks extracted from the video. This implicit conditioning on the generated video is helpful in mitigating potential artifacts in the generation and in making the approach more robust to mismatch in the video and the robot's embodiment. Note that we perform human video generation and ground-truth track extraction completely offline for training.

Visual Feature Extraction. For each frame in the generated human video

$\mathbf{V}_g$ and the robot video $\mathbf{I}_{t-k:k}$, we first extract features, i_g and i_r through a ViT encoder χ. The number of video tokens extracted this way is very large and they are temporally uncorrelated, so we have Transformer encoders Φ_g and Φ_r that process the respective video tokens through gated Cross-Attention Layers based on a Perceiver-Resampler architecture [1] and output a fixed number $N = 64$ of tokens. These tokens respectively are $z_g = \Phi_g(i_g)$ and $z_r = \Phi_r(i_r)$.

In addition to visual features from the generated video, we encode explicit motion information in the human-to-robot translation policy through point track prediction.

Point Track Prediction. We run an off-the-shelf tracking model [38, 39] on the generated video $\mathbf{V}_g$ to obtain tracks τ_g of a random set of points in the first frame P^0. In order to ensure that the latent embeddings from the generated video z_g can distill motion information in the video, we set up a track prediction task conditioned on the video tokens. For this, we define a track prediction transformer $\psi_g(P^0, i_g^0, z_g)$ to predict tracks $\hat{\tau_g}$ and define an auxiliary loss $||\tau_g - \hat{\tau_g}||_2$ to update tokens g_e.

Similarly, for the current robot video $\mathbf{I}_{t-k:k}$, we set up a similar track prediction auxiliary loss. We run the ground-truth track prediction once over the entire robot observation sequence (again with random points in the first frame P_0), but during training, the policy is input a chunk of length k in one pass. So here, the track prediction transformer $\psi_r(P^{t-k}, i_{t-k}, r_e^{t-k:t})$ is conditioned on the points in the beginning of the chunk P_{t-k}, the image features at that time-step i^{t-k} and the observation tokens for the chunk z_r.

BC Loss. For ease of prediction, we discretize the action space such that each dimension has 256 bins. We optimize a Behavior Cloning (BC) objective by minimizing error between the predicted actions $\hat{a}_{t:t+h}$ and the ground-truth $a_{t:t+h}$ through a cross-entropy loss.

In *Gen2Act*, we incorporate track prediction as an auxiliary loss during training combined with the BC loss and the track prediction transformer is not used at test-time. This is helpful in reducing test-time computations for efficient deployment.

5.1.4 Deployment

For deploying *Gen2Act* to solve a manipulation task, we first generate a human video conditioned on the language description of the task and the image of the scene. We then roll out the generated video conditioned closed-loop policy. For chaining *Gen2Act* to perform long-horizon activities consisting of several tasks, we first use an off-the-shelf LLM (e.g. Gemini) to obtain language descriptions of the different tasks. We chain *Gen2Act* for the task sequence by using the last image of the previous policy rollout as the first frame for generating a human video of the subsequent task. We do this chaining in sequence as opposed to generating all the videos from the first image because the final state of the objects in the scene might be different after the robot execution of an intermediate task.

5.2 Experiments

We perform experiments in diverse kitchen, office, and lab scenes, across a wide array of manipulation tasks. Through these experiments we aim to answer the following questions:

- Is *Gen2Act* able to generate plausible human videos of manipulation in diverse everyday scenes?
- How does *Gen2Act* perform in terms of varying levels of generalization with new scenes, objects, and motions?
- Can *Gen2Act* enable long-horizon manipulation through chaining of the video generation and video-conditioned policy execution?
- Can the performance of *Gen2Act* for new tasks be improved by co-training with a small amount of additional diverse human tele-operated demonstrations?

5.2.1 Details of the Evaluation Setup

Following prior works in language/goal-conditioned policy learning, we quantify success in terms of whether the executed robot trajectory solves the task

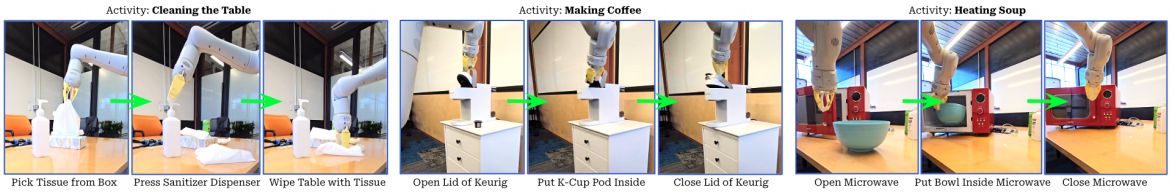

Figure 5.5: Robot executions for a sequence of tasks. The last frame of the previous execution serves as the conditioning frame for next stage video generation.

specified in the instruction, and define success rate over different rollouts for the same task description. We categorize evaluations with respect to different levels of generalization by following the terminology of prior works [15, 22]:

- Mild Generalization (**MG**): unseen configurations of seen object instances in seen scenes; organic scene variations like lighting and background changes
- Standard Generalization (**G**): unseen object instances in seen/unseen scenes
- Object-Type Generalization (**OTG**): completely unseen object types, in unseen scenes
- Motion-Type Generalization (**MTG**): completely unseen motion types, in unseen scenes

Here, seen vs. unseen is defined with respect to the robot interaction data, and the assumption is that the video generation model has seen diverse web data including things that are unseen in the robot data.

5.2.2 Dataset and hardware details

For video generation, we use an existing video model, VideoPoet [87] by adapting it to condition on square images in addition to language description of tasks. We do not do any fine-tuning of this model for our experiments, and find that it directly generalizes to human video generation in all the robot experiment scenes.

Table 5.1: Comparison of success rates for *Gen2Act* with different baselines and an ablated variant for the different levels of generalization as defined in Section 5.2.1

	Mild (MG)	Standard (G)	Obj. Type (OTG)	Motion. Type (MTG)	Avg.
RT1	68	18	0	0	22
RT1-GC	75	24	5	0	26
Vid2Robot	83	38	25	0	37
Gen2Act (w/o track)	83	58	50	5	49
Gen2Act	83	67	58	30	60

For robot experiments, we use a mobile manipulator with compliant two finger-grippers, and operate this robot for policy deployment through end-effector control. The arm is attached to the body of the robot on the right. We manually move the robot around across offices, kitchens, and labs and ask it to manipulate different objects in these scenes. We operate the robot for manipulation at a frequency of 3Hz. Before each task, we reset the robot arm to a fixed pre-defined reset position such that the scene is not occluded through the robot's camera.

For training the video-conditioned policy, we use an existing offline dataset of robot demonstrations collected by a prior work [22] and augment this with some paired demonstrations of human videos collected by another prior work [68]. In addition, we create pairs of the form (`generated_human_vid`, `robot_demo`) using the video generation model conditioned on the first frame of the respective robot demo, to generate a corresponding human video. For obtaining tracks on the generated human video and the robot demo, we use an off-the-shelf tracking approach [38, 39]. Generating human videos, and generating point tracks are done completely offline once and do not induce any additional cost during policy training.

5.2.3 Baselines and Comparisons

We perform comparisons with baselines and ablations with variants of *Gen2Act*. In particular, we compare with a language-conditioned policy baseline (*RT1*) [22] trained on the same robot data as *Gen2Act*. We also compare with a video-

conditioned policy baseline trained on paired real human and robot videos (*Vid2Robot*) [68], a goal-image conditioned policy baseline trained with the same real and generated videos of *Gen2Act* but by conditioning on just the last video frames (i.e. goal image) of the generated human videos (*RT1-GC*). Finally, we consider an ablated variant of *Gen2Act* without the track prediction loss.

5.2.4 Analysis of Human Video Generations

Fig. 5.3 shows qualitative results for human video generation in diverse scenarios. We can see that the generated videos correspond to plausibly manipulating the scene in the initial image as described by the text instruction. We can see that the respective object in the scene is manipulated while preserving the background and without introducing camera movements and artifacts in the generations. This is exciting because these generations are zero-shot in novel scenarios and can be directly used in a robot's context to imagine how an unseen object in an unseen scene should be manipulated by a human.

5.2.5 Generalization of *Gen2Act* to scenes, objects, motions

In this section we compare performance of *Gen2Act* with baselines and ablated variants for different levels of generalization. Table 5.1 shows success rates for tasks averaged across different levels of generalization. We observe that for higher levels of generalization, *Gen2Act* achieves much higher success rates indicating that human video generation combined with explicitly extracting motion information from track prediction is helpful in unseen tasks.

5.2.6 Chaining *Gen2Act* for long-horizon manipulation

We now analyze the feasibility of *Gen2Act* for solving a sequence of manipulation tasks through chaining. Table 5.2 shows results for long-horizon activities like "Making Coffee" that consist of multiple tasks to be performed in sequence. We obtain this sequence of tasks through Gemini [150], and for each task, condition the video generation on the last image of the scene from the previous execution and execute the policy for the current task conditioned

Table 5.2: Comparison of success rates for long-horizon activities via chaining of different tasks. We first obtain sub-tasks for activities with an off-the-shelf LLM and then rollout *Gen2Act* in sequence for the different intermediate tasks.

Activity	**Stages (from Gemini)**	**Success** % Stage 1, Stage 2, Stage 3
Stowing Apple	1. Open the Drawer 2. Place Apple in Drawer 3. Close the Drawer	80, 60, 60
Making Coffee	1. Open the Lid 2. Place K-Cup Pod inside 3. Close the Lid	40, 20, 20
Cleaning Table	1. Pick Tissues from Box 2. Press the Sanitizer Dispenser 3. Wipe the Table with Tissues	60, 40, 40
Heating Soup	1. Open the Microwave 2. Put Bowl inside Microwave 3. Close the Microwave	40, 20, 20

on the generated human video. We repeat this in sequence for all the stages, and report success rates for successful completion upto each stage over 5 trials. Figure 5.5 visually illustrates single-take rollouts from four such long-horizon activities.

Table 5.3: Analysis of co-training with an additional dataset of diverse tele-operated robot demonstrations ($\sim$ 400 trajectories).

Co-Training	Mild (MG)	Standard (G)	Obj. Type (OTG)	Motion. Type (MTG)	Avg.
Gen2Act (w/o co-train)	83	67	58	30	60
Gen2Act (w/ co-train)	85	75	62	35	64

5.2.7 Co-Training with additional teleop demonstrations

The offline dataset we used for experiments in the previous section had limited coverage over scenes and types of tasks thereby allowing less than 60% success rate of *Gen2Act* for higher levels of generalization (OTG and MTG in Table 5.1). In this section, we perform experiments to understand if adding a small amount of additional *diverse* tele-operated trajectories, for co-training with the existing offline dataset, can help improve generalization. We keep the video generation model fixed as usual. From the results in Table 5.3 we see improved performance of *Gen2Act* with such co-training. This is exciting because it suggests that with only a small amount of diverse demonstrations, the translation model of *Gen2Act* can be improved to better condition on the generated videos for higher levels of generalization where robot data support is limited.

5.2.8 Analysis of Failures

Here we discuss the type of failures exhibited by *Gen2Act*. We observe that for MG and to some extent in G, inaccuracies in video generation are less correlated with failures of the policy. While, for the higher levels of generalization, object type (OTG) and motion type (MTG), if video generation yields implausible videos, then the policy doesn't succeed in performing the tasks. This is also evidence that the policy of *Gen2Act* is using the generated human video for inferring motion cues while completing a task, and as such when video generation is incorrect in scenarios where robot data support is limited (e.g. in OTG and MTG), the policy fails.

Additional Details

Here we provide additional details on the method and experiments of *Gen2Act*.

5.2.9 Human Video Generation

We use a pre-trained VideoPoet model [87] directly without any adaptation or fine-tuning. The input to the model for video generation is a language description of a task (the prompt) and a square-shaped image. By virtue of being trained on diverse large-scale video datasets ($> 270M$ videos) we find that this model generalizes well to everyday tasks we develop *Gen2Act* for. It can generate realistic and plausible videos of humans manipulating objects, without introducing significant camera motions/artifacts in the generated videos. We ensure that the image of the scene input to the model doesn't have the robot in the frame (the initial reset position of the robot is such that the arm is mostly out of camera view). The language prompt to the model is of the form "A person `task-name`, static camera" e.g. for the task 'opening the microwave' the input prompt is "A person opening the microwave, static camera."

5.2.10 Closed-Loop Policy

For each frame in the generated human video $\mathbf{V}_g$ and the robot video $\mathbf{I}_{t-k:k}$, we first extract features, i_g and i_r through a ViT encoder χ. The number of video tokens extracted this way is very large and they are temporally uncorrelated, so we have Transformer encoders Φ_g and Φ_r that process the respective video tokens through gated Cross-Attention Layers based on a Perceiver-Resampler architecture [1] and output a fixed number $N = 64$ of tokens. We use 2 Perceiver-Resampler layers for both the generated video token processing and the robot observation history video processing. These tokens respectively are $z_g = \Phi_g(i_g)$ and $z_r = \Phi_r(i_r)$. During training we sample a fixed sequence of 16 frames from the generated video ensuring that we always sample the first and last frames. For the robot history, we choose the last 8 frames of robot observations. We resize all images to 224x224 dimensions.

We run an off-the-shelf tracking model [38, 39] on the generated video $\mathbf{V}_g$ to obtain tracks τ_g of a random set of points in the first frame P^0. In order to ensure that the latent embeddings from the generated video z_g can distill motion information in the video, we set up a track prediction task conditioned on the video tokens. For this, we define a track prediction transformer $\psi_g(P^0, i_g^0, z_g)$ to predict tracks $\hat{\tau}_g$ and define an auxiliary loss $||\tau_g - \hat{\tau}_g||_2$ to update tokens g_e. Similarly, for the current robot video $\mathbf{I}_{t-k:k}$, we set up a similar track prediction auxiliary loss. We run the ground-truth track prediction once over the entire robot observation sequence (again with random points in the first frame P_0), but during training, the policy is input a chunk of length k in one pass. So here, the track prediction transformer $\psi_r(P^{t-k}, i_{t-k}, r_e^{t-k:t})$ is conditioned on the points in the beginning of the chunk P_{t-k}, the image features at that time-step i^{t-k} and the observation tokens for the chunk z_r. The track prediction transformer has 6 self-attention layers with 8 heads and its role is solely to make the input tokens from generated video / robot observations informative of motion cues. Note that any ground-truth track prediction model can be used for this, and recent advances in point tracking can help improve this step [78]

For ease of prediction, we discretize the action space such that each dimension has 256 bins. So each action dimension can take values in the range $[0, 255]$. The bins are uniformaly distributed within the bounds of each dimension. We predict actions in the end-effector space, and also predict whether to terminate the episode, and whether the gripepr should be open/close. We optimize a Behavior Cloning (BC) objective by minimizing error between the predicted actions $\hat{a}_{t:t+h}$ and the ground-truth $a_{t:t+h}$ through a cross-entropy loss. This discrete action-space for prediction is based on prior works in multi-task imitation learning [22].

5.2.11 Chaining Gen2Act for Long-Horizon Manipulation

For deploying *Gen2Act* to solve a manipulation task, we first generate a human video conditioned on the language description of the task and the image of the scene. We then roll out the generated video conditioned closed-loop policy. For chaining *Gen2Act* to perform long-horizon activities consisting of several tasks, we first use an off-the-shelf LLM (e.g. Gemini) to obtain

language descriptions of the different tasks. We chain *Gen2Act* for the task sequence by using the last image of the previous policy rollout as the first frame for generating a human video of the subsequent task. We do this chaining in sequence as opposed to generating all the videos from the first image because the final state of the objects in the scene might be different after the robot execution of an intermediate task. This iterative sequencing of video generation after the previous robot execution is made possible by the fast generation capabilities of the Video model we use, VideoPoet (it takes less than 10 seconds to generate a new video after generating the very first video).

We provide examples of the prompt to Gemini and the subsequent outputs for task decomposition below:
Cleaning the Table. Q. What are the steps involved in cleaning a table with sanitizer dispenser and tissues? (write three steps)

A: Steps for Cleaning a Table with Sanitizer Dispenser and Tissues:

- Gather materials: Pick tissues from the box.
- Apply sanitizer: Press the sanitizer dispenser onto the tissue.
- Wipe: Gently wipe the table surface with the sanitizer-dampened tissues.

Making Coffee. Q. What are the steps involved in making coffee with a Keurig? (write first three steps)

A: Steps for Making Coffee with a Keurig:

- Open the lid: Open the lid of the Keurig machine.
- Insert K-Cup: Place a K-Cup pod into the machine.
- Close the lid: Close the lid of the Keurig machine.

Videos for these tasks are best viewed in the project website.

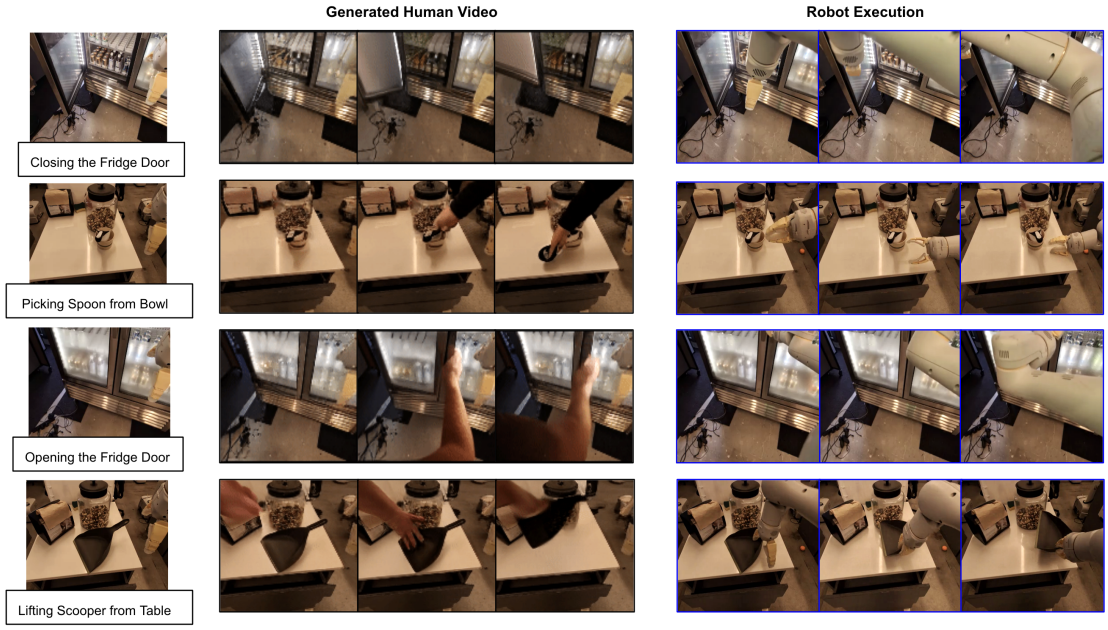

Figure 5.6: Analysis of failures of *Gen2Act*. The tasks here correspond to object type generalization. We can see that most of the failures of robot execution (top 3 rows) are correlated with incorrect video generations. In the last row the video generation is plausible but the execution is incorrect in following the trajectory of the generated video afetr grasping the object.

5.2.12 Analysis of Failures

Here we discuss the type of failures exhibited by *Gen2Act*. We observe that for MG and to some extent in G, inaccuracies in video generation are less correlated with failures of the policy. While, for the higher levels of generalization, object type (OTG) and motion type (MTG), if video generation yields implausible videos, then the policy doesn't succeed in performing the tasks. This is also evidence that the policy of *Gen2Act* is using the generated human video for inferring motion cues while completing a task, and as such when video generation is incorrect in scenarios where robot data support is limited (e.g. in OTG and MTG), the policy fails. Figure 5.6 shows some examples of failures of *Gen2Act* in different tasks. Most of the failures are correlated with video generation (first three rows) but generating a video plausibly (fourth row) is not a guarantee of the policy succeeding because there might be issues with grasping the object correctly and following the trajectory of the object post grasp. This indicates potential for future work to

explore recovering more dense motion information from the generated videos beyond point tracks, like object meshes for mitigating some of the failures.

5.3 Discussion and Conclusion

Summary. In this work, we developed a framework for learning generalizable robot manipulation by combining zero-shot human video generation from web data with limited robot demonstrations. Broadly, our work is indicative of how motion predictive models trained on non-robotic datasets like web videos can be used to used to enable generalization of manipulation policies to unseen scenarios, without requiring collection of robot data for every task.
Limitations. Our work focused on zero-shot human video generation combined with point track prediction on the videos as a way for providing motion cues to a robot manipulation system for interacting with unseen objects and performing novel tasks. As such, the capabilities of our system are limited by the current capabilities of video generation models, like inability to generate realistic hands and thereby limited ability to perform very dexterous tasks.
Future Work. It would be an interesting direction of future work to explore recovering more dense motion information from the generated videos beyond point tracks, like object meshes for addressing some of the limitations. Another important direction would be to enable reliable long-horizon manipulation by augmenting chaining with learning recovery policies for intermediate failures.

Chapter 6

Sample Efficient Robot Manipulation with Semantic Augmentations and Action Chunking

Developing a robot manipulator with multiple skills requires exposure to diverse experiences and the ability to acquire skills from a diverse data corpus. Collecting such multi-skill data corpus in the real world requires substantial effort and suffers from high operational costs and safety challenges. Given the expense, efficiency in robot learning paradigms is necessary for real-world training and deployment. While there are recent efforts in scaling real-world robotic datasets despite these challenges [33, 42, 104], efficiency seems to be overlooked in the attempts to scale [22, 71, 74, 75].

With the acknowledgment that robot learning will generally benefit as the scale of the robotics dataset grows, the focus of this chapter is on investigating generalization in developing capable agents under a *given data budget*. We restrict ourselves to a dataset with 7,500 robot manipulation trajectories (an order of magnitude less than related works [22]) containing a diverse collection of manipulation skills across different tasks. As a robot under deployment in real environments like homes, hospitals, etc., will always find itself in unseen scenarios, we set out to develop the most capable agent with an emphasis on

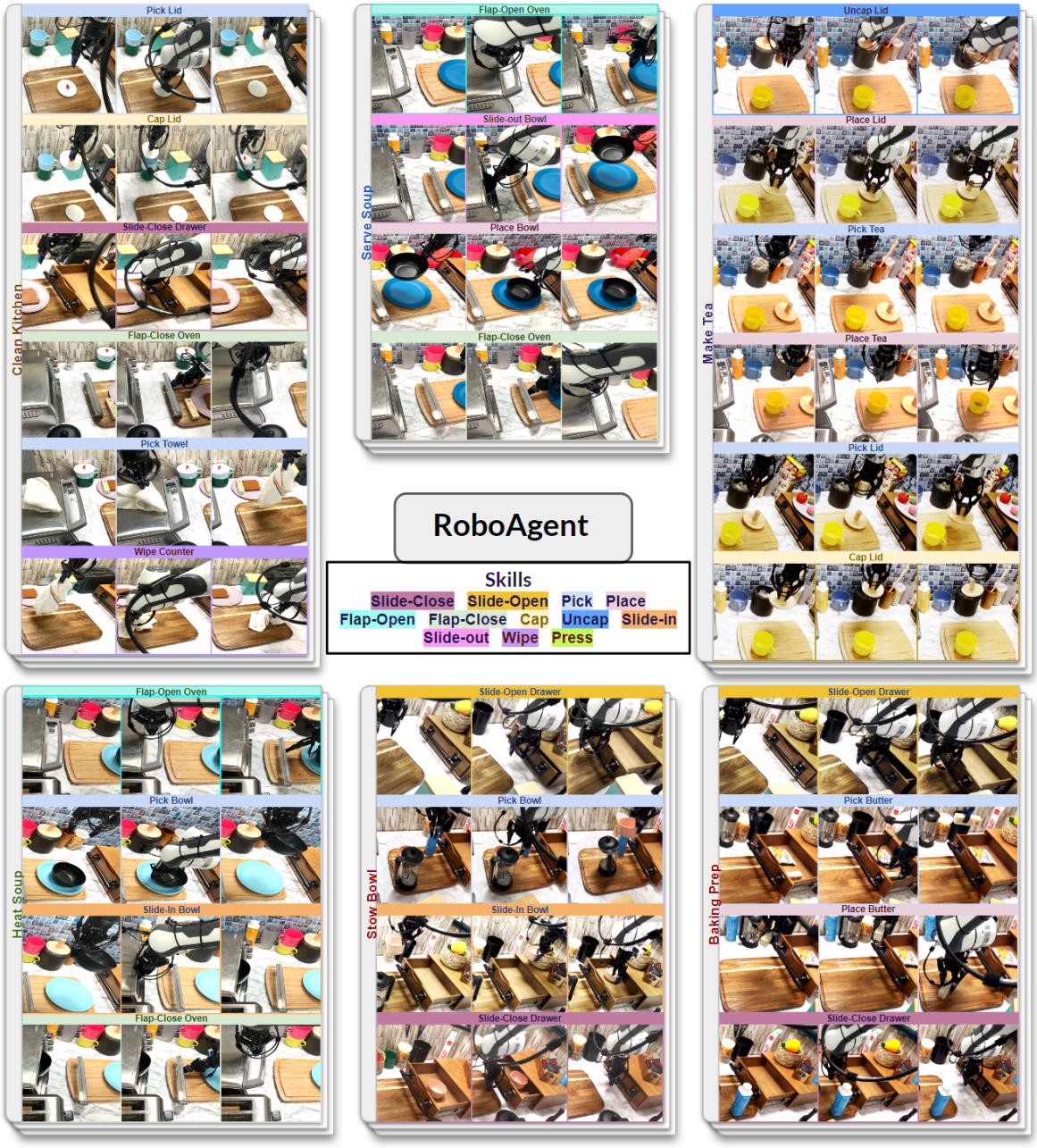

Figure 6.1: A glimpse of the diverse manipulation capabilities of *RoboAgent*– a single agent capable of 12 manipulation skills across 38 tasks encompassing 6 activities. For videos, visit: https://robopen.github.io/

its *ability to generalize to novel situations within this data budget*.

At first sight, wide generalization with a data budget seems like wishful

thinking - while it's possible to provide large representation capabilities to the agent's policy, scaling without data diversity will likely lead to overfitting and no generalization. Our insight is twofold: (1) collect a reasonably sized dataset (7,500 trajectories) with diverse coverage of skills, and devise a semantic augmentation strategy to rapidy multiply the dataset without additional human / robot cost, (2) devise a language-conditioned multi-task multi-scene policy architecture capable of handling the multi-modal data distribution. The architecture leverages the fact that robot movements are temporally correlated, by predicting action chunks [174] instead of per-step actions, leading to smoother behaviors and mitigation of covariate shift commonly observed in the low data imitation learning regime.

Overall, we emphasize that the data efficiency lessons we present are *general* and will help in achieving generalizable agents independent of the available data budget. Building on these insights, we make the following contributions:

- We present an efficient method MT-ACT designed to recover **generalist agents on a data budget**. MT-ACT leverages data multiplication via semantic augmentations and action representations to drive efficiency gains in low-data settings.
- MT-ACT's architecture can effectively ingest multi-modal trajectory data to recover *RoboAgent* – a single policy that can perform a diverse set of tasks through language instructions. Through extensive real-world experiments, we show *RoboAgent* is **capable of exhibiting 12 manipulation skills**.
- We perform extensive generalization studies to demonstrate that MT-ACT is 40 $\%$ more performant than alternatives, exhibits **superior generalization to diverse novel scenarios**, is amenable to **improvements and extensions during deployment through fine-tuning** and is robust for reproduction in completely new geographical setups.
- We meticulously recorded all the data collected during the course of the project which we are open-sourcing as part of `RoboSet` - one of the **largest open-source robotics dataset** on commodity hardware. It contains high-quality human teleOp trajectories spanning a balanced distribution of 12 skills across 38 tasks in diverse kitchen scenes.

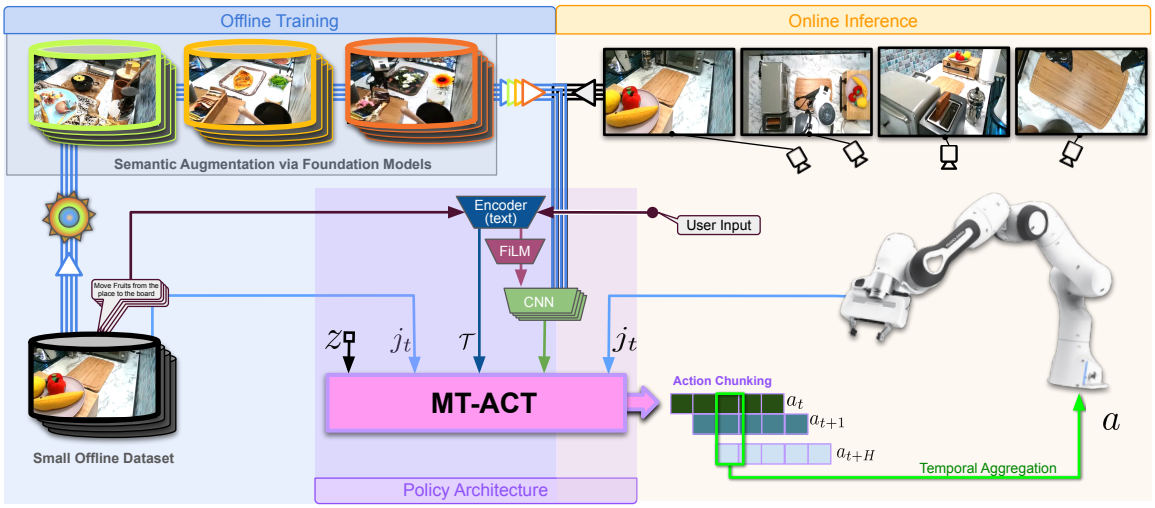

Figure 6.2: Two stage framework: [Left] **Semantic augmentation** stage diversifies the robot data offline using inpainting augmentations at no extra human/robot cost. [Right] **Policy learning** stage trains language-conditioned policy using MT-ACT – multi-task action-chunking transformers – which leverages efficient action representations for ingesting multi-modal multi-task data into a single multi-skill multi-task policy.

6.1 MT-ACT: Multi-Task Action Chunking Transformer

To learn generalizable manipulation policies, robots require rich and diverse experiences, encompassing a wide range of skills and contextual variations. However, operational costs and real-world challenges in collecting such extensive datasets pose a practical limit on their overall size. We address these limitations by developing *a paradigm that can learn effective multi-task agents under a limited data budget*. Our approach consists of two stages (Figure 6.2):

Semantic Augmentation – the first stage multiplies the pre-collected dataset by creating a diverse collection of semantic augmentations over existing robot's experiences. These semantic augmentations recreate a single robot demonstration into several demonstrations, each with a different semantic context (objects, textures, backgrounds, etc), at no extra robot or human cost. Such data diversification incorporates real-world semantic priors to make the multi-task agent robust to test-time out-of-distribution scenarios.

Policy Learning – the second stage learns robust skills from limited skill

	Trajectories	Tasks	Skills	Scenes	Source
RoboSet (MT-ACT)	7,500	38	12	10	TeleOp
RoboSet (kitchen)	30,050	38	12	10	TeleOp
RoboSet (bin)	70,000	10	4	1	Heuristics
RoboSet (full)	98,050	48	12	11	TeleOp+Heuristics
BridgeData [42]	33,200	72	8	10	TeleOp
BC-Z [71]	25,000	100	9	N/A	TeleOp
RoboTurk [104]	2,100	N/A	3	1	TeleOp
Amazon Pick-Place [105]	100,000	N/A	1	1	Heuristics
RoboNet [33]	162,000	N/A	2	7	Heuristics
BAIR Pushing [41]	N/A	N/A	1	1	Heuristics

Table 6.1: Open-source real-world manipulation dataset landscape: RoboSet(ours) https://robopen.github.io/roboset/ is one of the largest open-source robotics datasets. It contains high-quality demonstration, including human tele-operation, trajectories spanning a balanced distribution of 12 skills across 38 tasks in diverse kitchen scenes.

data by adapting design choices from prior works limited to single-task settings for large-scale generalization in multi-task multi-scene manipulation tasks. We develop MT-ACT – a language-conditioned novel policy architecture to train robust agents capable of recovering multiple skills from multi-modal datasets.

6.1.1 Dataset (RoboSet)

Training a general agent capable of robustly exhibiting a diverse repertoire of skills in novel scenes and tasks needs exposure to experiences matching this diversity. To align with our goal of building a data-efficient robot learning paradigm, we restrict ourselves to a frozen pre-collected small but diverse dataset – RoboSet(MT-ACT). To capture behavioral diversity, we ensure sufficient coverage over different core skills, where each skill if defined as a temporally correlated sequence of actions that lead to plausible change in an object's pose. Example skills include *closing/opening* articulated objects, *sliding*, *wiping*. Each skill is instantiated across a set of objects. We refer to such (skill, object) combinations as a **task**. Our tasks are instantiated in different kitchen scenes, visually illustrated in Appendix (see webpage).

Instead of a random collection of tasks, we structure groups of tasks as belonging to be part of a household **activity**, such that they can be executed in sequence to obtain a meaningful outcome, such as cleaning a kitchen.

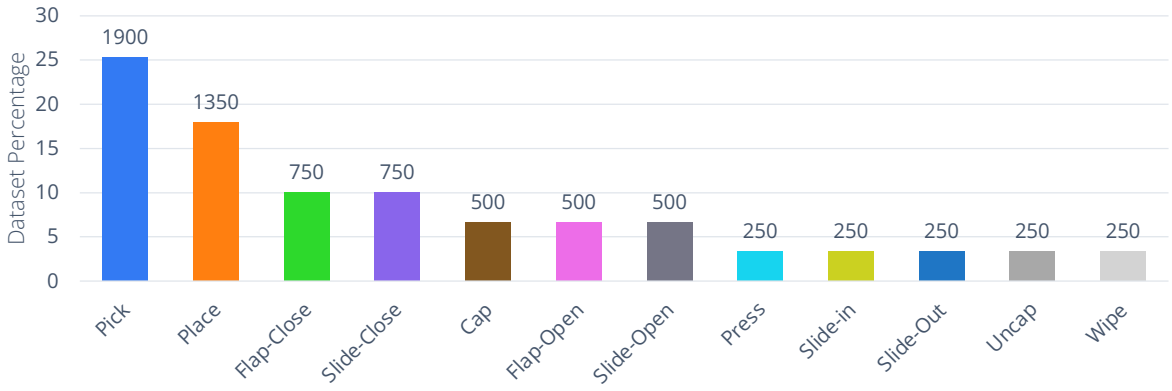

Figure 6.3: Skill distribution in terms of % of trajectories with a certain skill used to *train RoboAgent*. Number on top shows number of trajectories.

`RoboSet(MT-ACT)` – the dataset we used for this project (i.e. to train *RoboAgent*) consists of 7,500 trajectories (Table 6.1)[1] collected using human teleoperation. The dataset involves 12 skills (see Figure 6.3 for skill distribution). While the common pick-place skills cover 40% of the dataset, we also include contact-rich skills (*Wipe*, *Cap*) as well as skills involving articulated objects (*Flap-Open*, *Flap-Close*). We collect the overall dataset across four different physical setups. Each setup is instantiated with various everyday objects to create a kitchen scene. We frequently vary each set up with different variations of objects, thereby exposing each skill to multiple target objects and scene instantiations. Figure 6.4 provides a glimpse of the overall setup and a subset of objects. Overall, unlike previous datasets, `RoboSet` provides a broad coverage of manipulations skills for generalist robots required to operate in kitchen environments.

In Table 6.1, we compare our dataset with existing *open-source* robot manipulation datasets. As noted above, we use `RoboSet`(MT-ACT) (7.5K) trajectories to train *RoboAgent*. However, we release a much larger dataset, `RoboSet` which includes more teleoperated data, data collected during policy evaluation and data for non-kitchen settings. Overall, the entire `RoboSet`

[1]Note that the entire `RoboSet` is much larger and much more diverse. *RoboAgent* is trained on `RoboSet`(MT-ACT) – a subset consisting of 7500 trajectories

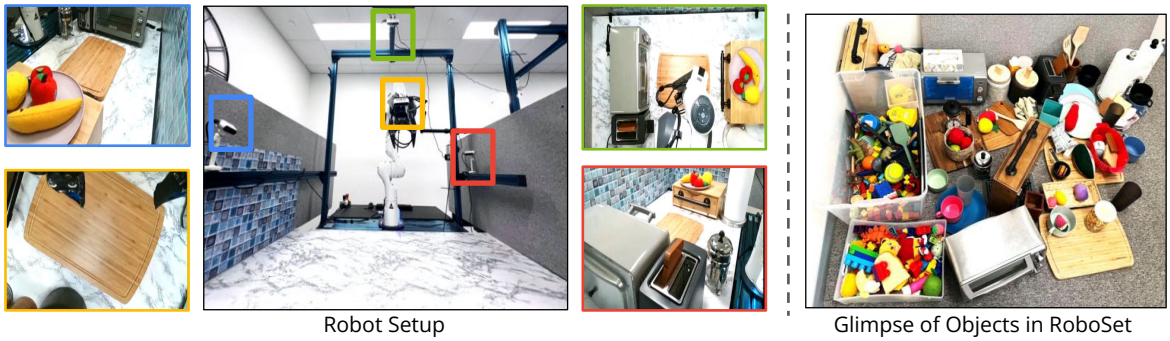

Figure 6.4: A zoomed-out view of the robot environment, showing all four cameras in the scene. *Right:* A glimpse of the diverse objects in `RoboSet`. The objects include articulated objects (drawers, ovens), smaller rigid objects (french press, bowls) and deformable objects (towels, cloth).

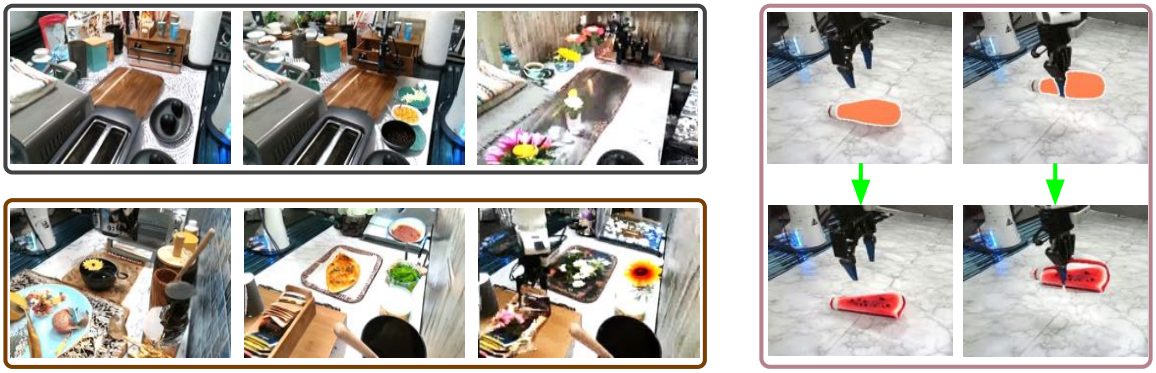

Figure 6.5: Illustration of the data augmentations used to rapidly multiply limited robot datasets with diverse semantic scene variations. (a) shows the scene around the robot and the interaction object changing. (b) shows the interaction object itself changing while preserving the rest of the scene.

is one of the largest publically released datasets with commodity robots and collected in real-world setup. `RoboSet` contains a large number of diverse skills and scene variations.

6.1.2 Semantic Data Augmentation

Generally useful robot manipulation systems will need to be able to deal with out-of-distribution scenarios (e.g. different homes and offices). Since any dataset of a practical size will have a limited diversity of objects and

scenes (due to physical access and operational constraints) compared to what agents will encounter during deployment, we develop a *fully automatic* offline process to multiply our dataset.

Given an initial dataset of robot behaviors, we multiply the dataset by creating multiple semantic variations of the dataset while preserving the *robot behavior* within each trajectory. These semantic variations are created by applying augmentations per frame within the trajectory. Augmentations are created by inpainting a part of the image frame introducing new objects and scene variations. The inpainting locations are specified by a mask and are informed by a text prompt. As opposed to [27, 101, 171] needing manual masks, object templates, etc., our approach is fully automatic. We use the SegmentAnything model [85] to automatically detect semantic boundaries in the scene to create augmentation masks. See Section 6.1.3 for additional details. We emphasize that our approach toward semantic augmentation is fully automatic and offline. It takes advantage of and is also well poised to continually benefit from rapidly advancing progress in segmentation and in-painting models [85, 164]. Akin to fields of natural language processing and computer vision, by distilling semantic real-world priors present in internet images/videos into robotics datasets, it provides robot learning a scalable mechanism to benefit from internet-scale data at no extra cost to humans/robots.

6.1.3 MT-ACT Policy Learning

Recovery of a generalizable robot manipulation policy under a practical data budget available in robotics demands an efficient policy architecture. In scenarios that have sufficient coverage within the training data, we want the policy to stay close to nominal behaviors (efficient imitation). The policy also needs to be effective to new variations (effective generalization) and contexts (efficient task conditioning) that are unseen during training . In addition, we want the policies to exhibit temporally correlated smooth behaviors accomplishing tasks with minimal errors and safety violations.

Our policy architecture – MT-ACT is designed to be a Transformer model of sufficient capacity that can handle multi-modal multi-task robot datasets. In order to capture multi-modal data, following prior works [174] we incorporate

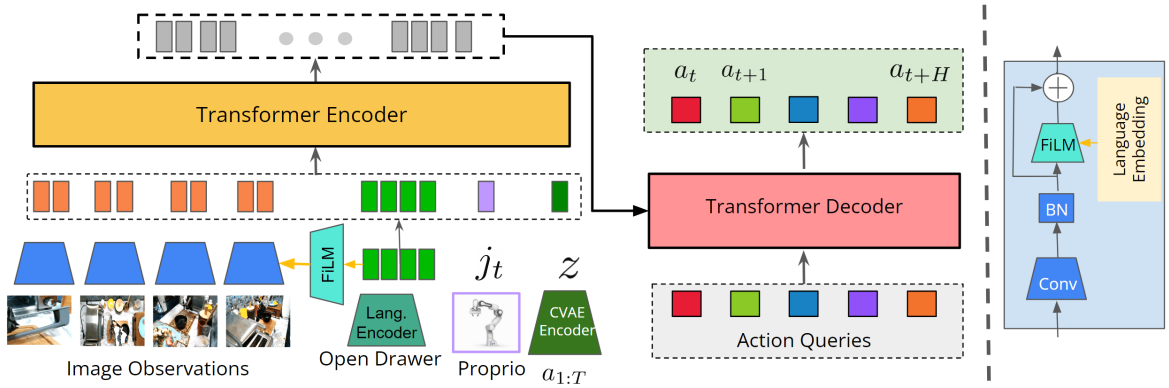

Figure 6.6: Policy architecture for MT-ACT . We use a CVAE that learns latent encodings z for action sequences to implicitly identify different *modes* in the data. A transformer takes as input a latent code, language embedding of the task, and image embeddings from four camera views, to autoregressively output an action sequence $a_{t:t+H}$ for chunk size H. On the right, we shows details for the FiLM layer [121] that we use for language-conditioning.

a CVAE [84] that encodes action sequences into latent *style* embeddings z. The decoder of the CVAE is the Transformer policy that conditions on the latents z. This formulation of expressing the policy as a generative model helps in effectively fitting to the multi-modal teleop data, without ignoring regions of a trajectory crucial for precision, which are also likely to be more stochastic. In order to model multi-task data, we incorporate a pre-trained language encoder [47] that learns an embedding $\mathcal{T}$ of a particular task description. To mitigate issues of compounding error and to achieve smooth temporally correlated robot motions, at each time-step, we predict actions H steps in the future and execute them through temporal-aggregation of overlapping actions predicted for a particular time-step [174]. To improve effectiveness towards scene variations and robustness towards occlusions in clutter, we provide the policy with four different views of the workspace through four cameras.

At time-step t, the transformer encoder takes four camera views , $o_t^{1:4}$, the joint pose of the robot j_t, the style embedding from the CVAE z, and the language embedding $\mathcal{T}$. We use a FiLM-based conditioning [22, 121], in order to ensure that the image tokens are able to reliably focus on the language instruction, such that the policy doesn't get confused about the task when multiple tasks are possible in a scene. The encoded tokens go to the decoder of the Transformer policy with fixed position embeddings, which finally outputs

the next action chunk (H actions) for the current time-step. For execution, we average over all overlapping actions predicted for the current time-step (As $H > 1$, the action chunks overlap), and execute the resulting averaged action. Overall, our proposed architecture extends ACT [174] to multi-task ACT (MT-ACT) using appropriate language conditioning (see Section 6.3.2). Since `RoboSet`(MT-ACT) contains diverse skills we show that the VAE prior can capture such behavior diversity. Finally, we demonstrate for the first time that action-chunking and temporal aggregation are useful for learning diverse multi-task behaviors for quasi-static (low-frequency control) tasks in diverse scenes.

6.2 Experimental Design

Our experiments aim to understand the following questions

- How does MT-ACT perform, quantitatively and qualitatively, on a large set of vision-based robotic manipulation tasks? How does it generalize to new tasks, objects, and environments?
- Does semantic augmentation improve policy generalization (i.e. scenes with new target objects)?
- Does action chunking help with temporally consistent trajectories, achieving higher success?

To answer these research questions we instantiate our framework in the real world using commodity hardware and objects commonly used in everyday kitchens.

Robot hardware. As noted before, Figure 6.4 shows our robot environment, called *RoboPen* that consists of a kitchen setup with everyday objects, a Franka Emika Panda arm with a two-finger gripper with adaptive fingers, three fixed cameras (top, left, right), and a wrist camera. We utilize all cameras for robust policy learning.

Data collection. As noted in (Section 6.1.1) our tele-operated dataset is collected across four different physical setups with periodically changing kitchen-like environments. Additional details regarding the dataset, along with

sample trajectories, and a link to the entire dataset are in the project website. We are publicly releasing this dataset, as part of RoboSet – a large multi-skill robotics dataset. To our knowledge, this is one of the largest open-source robot manipulation datasets with the most commonly used *non-proprietary robot hardware* (Franka Panda [58]) containing diverse real-world behaviors beyond pick and place.

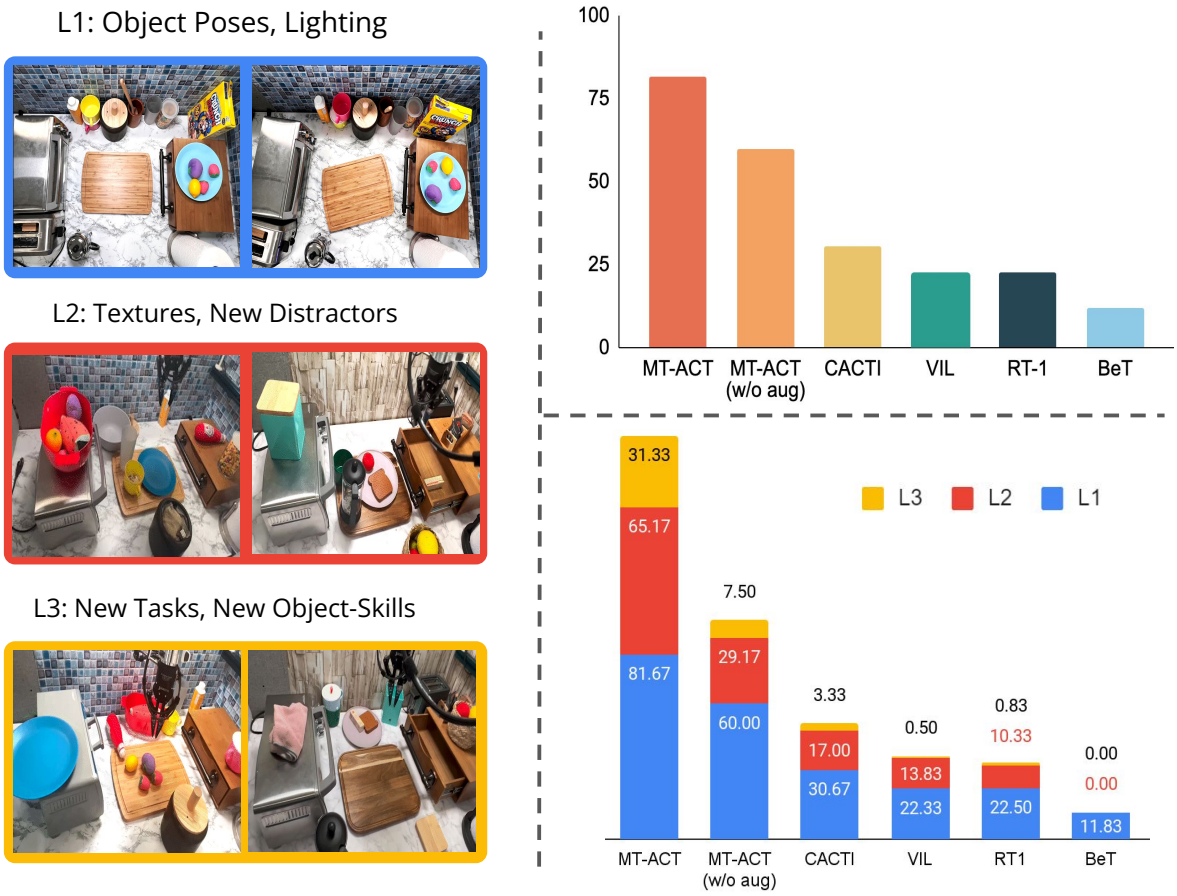

Figure 6.7: Visualization of different generalization axes, evaluating effectiveness with lighting variations and smaller scene changes such as object poses (L1), robustness to significant scene variations (L2), generalization to unseen tasks (L3). *Top-Right:* Success rates for commonly used L1-generalization. *Bottom-Right:* Multi-Task (universal policy) results for different levels of generalization showing success rates. See 6.9 for L4-generalization results.

Generalization Axes. Following prior work [22, 71, 72], we define each *task* to consist of a particular language command like *'pick a cube of butter from the drawer on the left'* that defines an object to be interacted with (butter),

a skill to be executed (pick), and some context (drawer on the left). Each activity consists of a collection of 4-5 close tasks that can be executed in sequence. The policy trained to achieve (all tasks of) an activity is referred to as *activity policy* and the policy trained over all the activities as the *universal policy*. We consider different levels of generalization, illustrated visually for a scene in Figure 6.7: **L1(Effectiveness)**: Generalization of the agent to variations in object positions and orientations, and in lighting conditions. **L2 (Robustness):** New background, different distractor object variations, and unseen distractor objects introduced in the scene. **L3 (Generalization):** New tasks never seen before, including new object-skill combinations. **L4 (Strong Generalization):** New kitchen never seen before (see Figure 6.9 Left).

6.3 Experiments

Through detailed real-world robot manipulation experiments, we evaluate the proposed framework for sample efficiency, and generalization of the agent to diverse scenes. We provide further results (including videos and appendix) on our webpage https://robopen.github.io/.

Baselines. We compare multiple baselines that use visual policy learning for robotics. *Single Task Agents*: We compare ACT-based policies [174] trained for individual tasks, and evaluated on the respective tasks. These policies don't need to generalize across tasks and scene, and represent an approximate *oracle* performance per task. *Visual Imitation Learning (VIL)*: We compare with regular language-conditioned multi-task visual imitation learning. *CACTI* [101]: This baseline is a prior framework for multi-task learning that also uses some scene augmentations for generalization. *RT1* [22]: We re-implement a baseline RT1-like agent. *BeT* [135]: We modify the Behavior Transformer architecture with language conditioning and train it in a multi-task manner.

6.3.1 Multi-Task Real-World Results

Performance. Figure 6.7 (Right) compares our proposed MT-ACT policies against commonly used imitation learning architectures. We show success

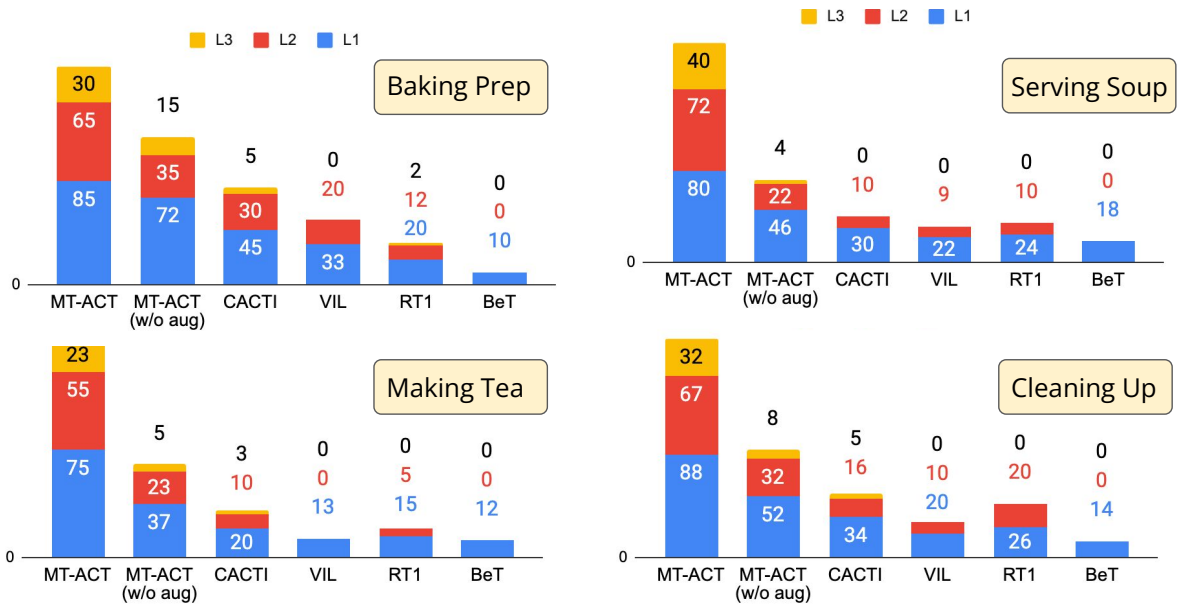

Figure 6.8: Results of success rate for MT-ACT, its ablated variant without semantic augmentations, and baselines, averaged over tasks in activities, with L1, L2, L3 levels of generalization. Each activity consists of 4-5 tasks, and the results average over the tasks in an activity. The results show that semantic augmentations significantly improve performance of MT-ACT over the baselines. In addition, even without augmentations, the MT-ACT policy achieves much higher success rates compared to the baselines. Full results on all activities are in the Appendix.

rates on the y-axis, with 20 evaluation rollouts per task, averaged over all tasks. In this figure (Figure 6.7 Left-Bottom) we only plot results for *L1-generalization* since this is the standard setting most other imitation learning algorithms use. We observe that all approaches that only model next-step actions (instead of sub-trajectories) exhibit weaker performance. Among these approaches, we find that action-clustering-based approaches (BeT [135]) for multi-task settings, perform significantly worse. We believe this happens because naive clustering in very diverse action distributions may not result in clusters that generalize across diverse skills. Additionally, since we are operating under a data budget, we observe that RT1-like approaches that require a lot of data do not perform well in the low data regime. By contrast, our MT-ACT policy which uses action-chunking and CVAE to model multi-modal sub-trajectories significantly outperforms all baselines.

Generalization and Robustness. Figure 6.7 (Bottom-Right) shows the

results for all methods across multiple levels of generalization (**L1**, **L2**, and **L3**). Recall that these levels of generalization include diverse table backgrounds, distractors (**L2**) and novel skill-object combinations (**L3**). From Figure 6.7 (Bottom-Right) we see that by virtue of semantic augmentations and action representations, MT-ACT significantly outperforms all the baselines we consider. More interestingly, we see that semantic augmentations have less effect for L1-generalization ($\approx 30\%$ relative), they provide a *much more* significant improvement for both L2-generalization ($\approx 100\%$ relative) and L3-generalization ($\approx 400\%$ relative). Since semantic augmentations affect both scenes (backgrounds and distractor objects) as well as target objects being manipulated they provide useful support for the policy to achieve increasing levels of generalization.

Additionally, in Figure 6.8 we also report generalization results for each activity separately. From Figure 6.8 we see that our proposed semantic augmentations positively affect each activity's performance. Interestingly, we find that for some of the harder activities (Making-Tea, Stowing-Bowl, Heating Soup) the relative improvement in performance due to semantic augmentations is much larger.

Overall, our results show that traditional visual imitation learning (without any augmentations), i.e., VIL and RT1 trained on our relatively small dataset, completely fail for L2 and L3, indicating a lack of generalization to unseen scenarios, due to data paucity. Finally, we also test our policy on a completely new kitchen with novel objects, arrangements, distractors, i.e., L4 generalization. Figure 6.9 (Left) visualizes this new kitchen environment. We evaluate all methods in this new kitchen for 3 tasks. Figure 6.9 (Right) shows the results for each method on this new environment. Specifically, we obtain an average success rate of 25% for MT-ACT (and 0 for all baselines). Even MT-ACT without semantic augmentations fails completely on this new environment thus showing the strong generalization ability of our approach.

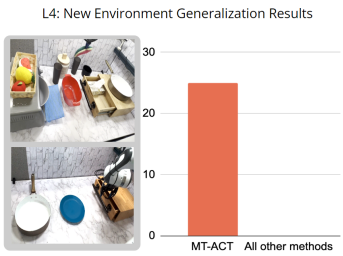

Figure 6.9: Only MT-ACT policies perform tasks in a completely new kitchen environment (L4).

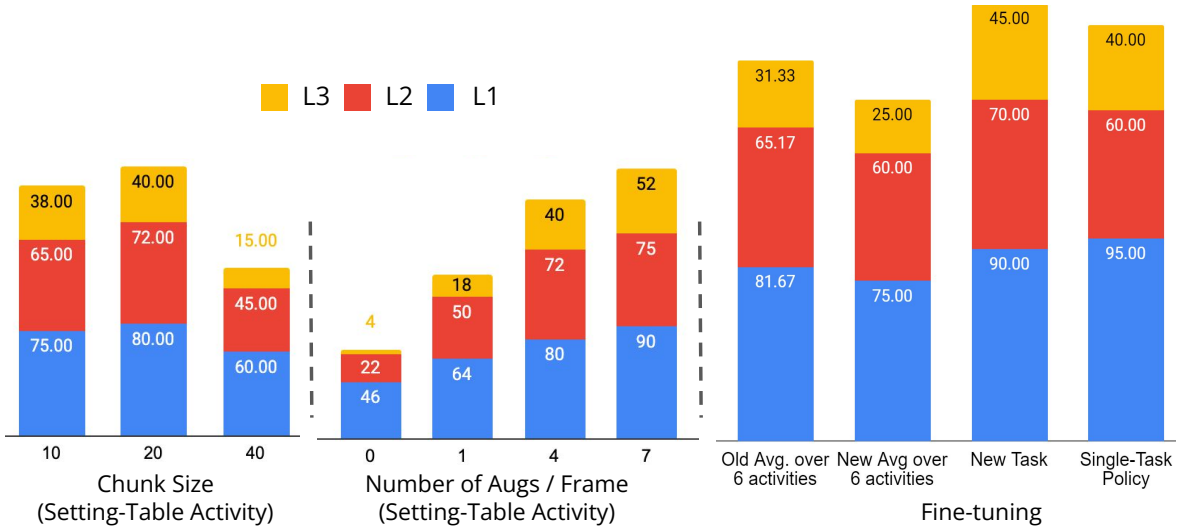

Figure 6.10: Results for different ablations (see section 6.3.2), showing the benefits of FiLM conditioning, the effect of varying chunk sizes in the predictions, the number of augmentations per frame for multiplying the dataset, and the feasibility of fine-tuning MT-ACT for improved deployment.

6.3.2 Ablation studies

Language conditioning using FiLM. For language conditioned multi-task policy, as described in section 6.1.3, we use a FiLM based conditioning [121] for the language embedding of task descriptions [37]. We ablate this choice by comparing with simple concatenation-based conditioning. We observe around 10% drop in performance of the version of MT-ACT without FiLM conditioning, across all 4 generalization levels.

Chunk Size for Action Representations. We ablate our choice of action chunk size. Figure 6.10 (Left), shows that a chunk size of 20 performs the best, with a 0-5% drop in performance with chunk size 10. However, a large chunk size 40 performs significantly worse with more than 20% drop in performance indicating the inability of the policy to correct errors as the chunks grow in size.

Number of augmentations per frame. Figure 6.10 (Middle) ablates the number of augmentations per frame, to see if more augmentations help MT-ACT in learning a more performant policy. We see that number of augmentations per frame is strongly correlated with overall performance gains. Thanks to the real-world semantic priors injected via data augmentation, the

gains are more notable for L2 and L3 levels where out-of-domain generalization is required.

Robustness analysis. We perform robustness analyses of the universal MT-ACT agent, by manually perturbing the scene during evaluation, and also introduce system failures such as blocking camera views. On average, we find that the policy is robust to these strong active variations, and can solve the specified task in about 70% of the 20 evaluations we run for this analysis (videos in the website).

Plasticity. We evaluate the feasibility of adding additional capabilities to the universal MT-ACT agent, without requiring significant re-training. We take the trained agent (on 38 tasks) and fine-tune on $(1/10)^{\text{th}}$ of the original data combined with data for a new held-out task (placing toast in toaster oven). The new task has 50 trajectories, multiplied with 4 augmentations per frame, for a total of 250 trajectories. Fig. 6.10 (Right) shows that the fine-tuned agent is able to learn this new task, without significantly deteriorating in performance on the previous 6 activities. Also, it achieves slightly better L2, L3 performance ($\approx$ 10%) than a single-task policy trained only on augmented data of the new task, indicating efficient data re-use.

6.3.3 Reproducibility Experiments

To better understand the generalization and plasticity capabilities of *RoboAgent* , we perform a challenging experiment by deploying the trained agent in a completely different location 3000 miles (5000km) away from where data was collected, and observe comparable success rates of 30-60% on new tasks in this setup both for zero-shot deployment and fine-tuning. Detailed results are in the Appendix.

6.4 Dataset details

MT-ACT uses 7,500 human teleoperated demonstrations from the *RoboSet* dataset [2]. MT-ACT dataset consisted of RGB and depth frames from four

[2]The full RoboSet is much more diverse and consists of 9,500 teleoperated demonstrations, 20,500 kinesthetic demonstrations in various kitchen and table-top settings. In addition, it contains about 70,000 trajectories in bin

Heat Soup	Serve Soup	Baking Prep	Making Tea	Cleaning Up	Stow Bowl
Flap-Open Oven	Flap-Open Oven	Slide-Open Drawer	Uncap Lid	Pick Lid	Slide-Open Drawer
Pick Bowl	Pick Bowl	Pick Butter	Place Lid	Cap Lid	Pick Bowl
Slide-In Bowl	Slide-Out Bowl	place Butter	Pick Tea	Slide-Close Drawer	Place Bowl
Flap-Close Oven	Flap-Close Oven	Slide-Close Drawer	Place Tea	Flap-Close Oven	Slide-Close Drawer
			Pick Lid	Pick Towel	

Table 6.2: List of activities (Top Row) and the associated tasks for each activity.

camera views (right, left, top, and wrist) as shown in figure 6.4, Franka joint positions and velocities, end-effector/gripper position and velocities, controls applied to the Franka joints and end-effector/gripper, and the time-steps (40 steps).

The data was collected using an Oculus Quest 2 controller on a kitchen table-top setup at 5Hz and saved in HDF5 format. Rollouts from the data are shown in Figure 6.11 as well as in `https://robopen.github.io/roboset/`.

6.4.1 Dataset Terminology

Skill Different works in robotics often assign different meanings when they refer to "skills". In our work, we refer to a skill when the robot performs a similar motion across different object instances (both shape and size). For instance, pick, place, open, close objects are considered as different skills. Since our dataset contains articulated objects if the "open" skill with multiple objects results in different motion we classify them as different skills. For instance, "Open Drawer" requires interacting with a prismatic joint while "Open Oven" interacts with a revolute joint. Hence, we classify these as separate skills. Our definition is broadly similar to some previous works [22]. We use 12 skills in RoboSet – *Slide-Open, Slide-Close, Flap-Open, Flap-Close, Cap, Uncap, Pick, Place, Wipe, Plunge, Slide-in, Slide-out*.

Task: We define each instantiation of our skill with a particular object class as a different task. For instance, "Pick Mug" and "Pick Butter" correspond to the same "Pick" skill but are two different tasks.

Activity: A general robot agent will eventually need to perform a sequence

settings collected through heuristics

of tasks, e.g. make tea. We refer to such sequence of tasks as *activities*. Table 6.2 lists the activities used in our work as well as tasks corresponding to each activity. Our final aim is to train a *single* robot agent to perform all activities.

Policies: We train and compare different policies in our work. We classify these policies into *single-task* policy, *multi-task (single-activity)* and *multi-task (universal)* policies. As each name suggests, single-task policies are trained on specific tasks. Multi-Task (single-activity) policies are trained on all tasks belonging to an activity. Finally, Multi-Task (universal) policies are trained on all tasks and activities. Our final *RoboAgent* is trained as a Multi-Task (universal) policy.

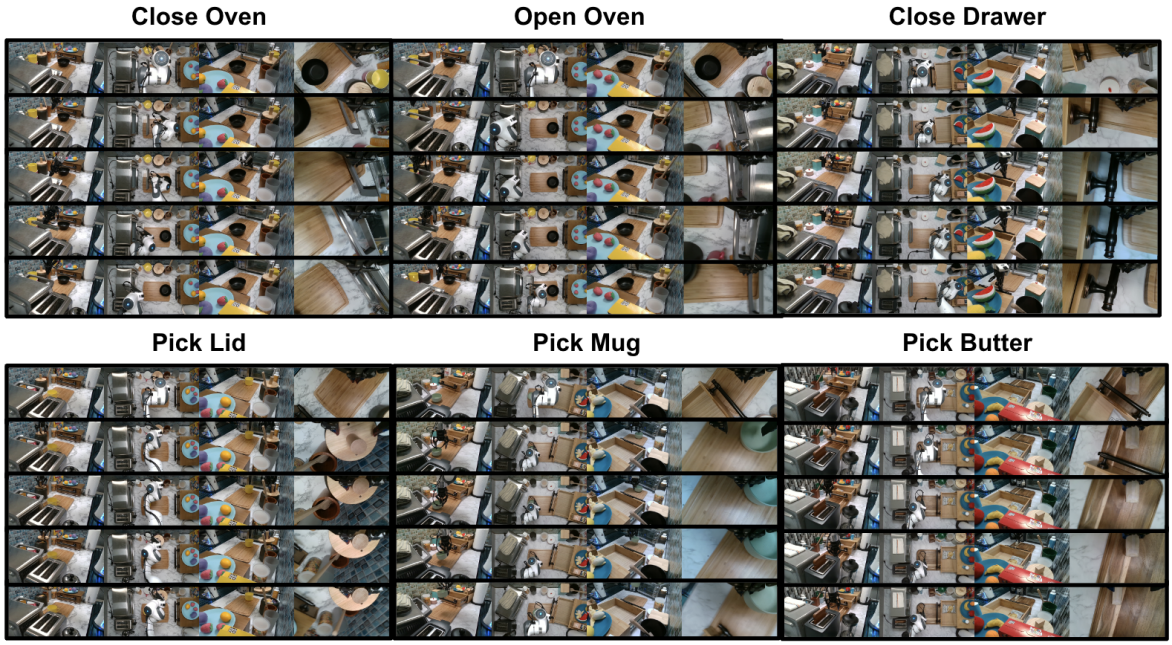

Figure 6.11: Sample task demonstrations in the `RoboSet`(visualizing four views horizontally, and five timesteps vertically), used for training.

6.4.2 Details on Semantic Augmentations

We enable two different types of scene augmentations for multiplying data, for enabling generalization to different scenes with novel distractors, and to scenes with different objects for interaction:

- **Augmenting interaction object:** Given the joint angle of the robot in a frame of a trajectory, we use forward kinematics to recover the robot mask as well as the end-effector position of the robot. We use the end-effector location to prompt SegmentAnything [85] for obtaining a mask of the object being interacted with. We then inpaint the region of the object being interacted with, based on a text prompt, and keep it consistent across time by tracking with TrackAnything [164].

- **Augmenting background:** We use SegmentAnything [85] to randomly choose a set of objects in the background that do not overlap with the robot mask, and the mask of the object being interacted with, and inpaint the scene based on the resulting overall mask over all the objects identified by SegmentAnything.

Note that our augmentation approaches are all automatic and do not require any manual effort in specifying masks or object meshes etc. This is in contrast to prior works that require manual specification of a fixed mask per trajectory [101], and those that require templates of object textures and meshes [27]. In addition, unlike [171], we do not require training any further modules for identifying objects through open-vocabulary detection that relies on language grounding.

6.5 Train and Evaluation Details

In this section we present training and evaluation details both for our methods and the baselines.

6.5.1 Robot Environment and Evaluation Details

The robot environments for evaluation consist of table-top kitchen setups with diverse real objects in the scene. There are 4 cameras providing complementary views of the workspace. The robot is a Franka Emika Panda arm operated with joint position control, with an action space dimension of 8 (7 joint positions, 1 dimension for end-effector open/close). The robot arm has a

Figure 6.12: Qualitative results of rollouts for L2 and L3 levels of generalization, showing tasks *open drawer* and *pick a slab of butter*. For L2 we introduce different distractors in the scene, and change the background tiles. For L3, in addition to changes in L2 we introduce different task objects, for example by replacing a slab of butter with a piece of watermelon, or a banana.

two-finger gripper, and a wrist camera. The robot is operated at a frequency of 5Hz.

6.5.2 Hyper-parameters for MT-ACT and baselines

Here we provide hyper-parameter details of the policy architecture. We train all policies for 2000 epochs. For the overall MT-ACT agent, trained on the augmented dataset, this takes about 48 hours on a single 2080Ti GPU with a batch size of 8.

For our baseline implementations we did a hyperparameter search for relevant parameters. For each baseline implementation we try to adapt them from their officially released code. Specifically, for RT1 [22] we use `https://github.com/google-research/robotics_transformer` for reference. On the other hand, for BET [135] we use `https://github.com/notmahi/bet`. To provide language conditioning for both baselines we use similar FiLM [121] implementation as our approach.

For hyper-parameters we use 3 different discrete action sizes – 64, 256 and 512, we vary the learning rates from $(1e-3, 1e-4)$. We use the AdamW optimizer with a weight decay range in $(1e-2, 1e-3, 1e-4)$. Our RT-1 transformer uses 6 layers with 8 parallel attention heads and each head with size 64. Each transformer uses a feedforward layer with intermediate sie of 1024. On the other hand for [135] we experiment with 3 different action cluster

Table 6.3: Hyper-parameters for MT-ACT

Name	Value
learning rate	1e-5
batch size	8
feedforward size	3200
Attention heads	8
chunk size	20
dropout	0.1
Transformer encoder layers	4
Transformer decoder layers	7
Language Embedding size	384

Table 6.4: Hyper-parameters for RT-1 [22]

Name	Value
learning rate	1e-4
discrete action tokens	256
batch size	64
feedforward size	1024
Attention heads	8
dropout	0.1
Transformer layers	6
Language Embedding size	384

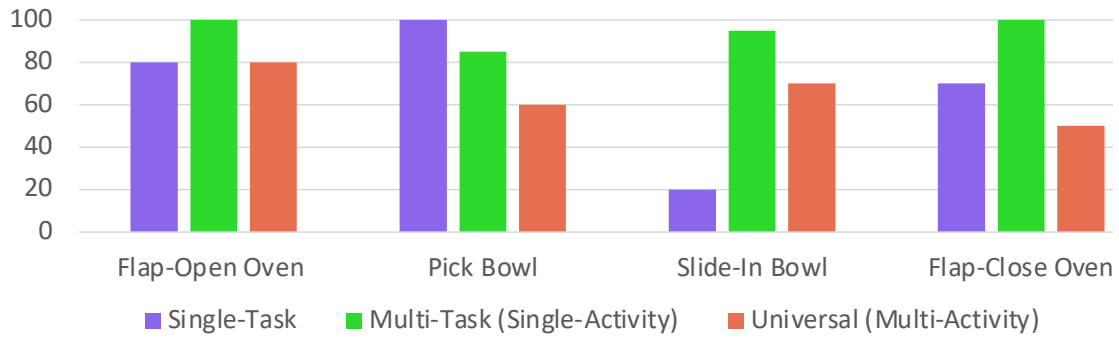

Figure 6.13: Single-Task vs Multi-Task comparison for Heat Soup activity. Multi-Task (Single Activity) represents a multi-task policy trained on only 4 tasks in Heat-Soup activity.

sizes – 64, 256 and 512. We use a similar transformer implementation for BET as RT-1. Finally, for real-world evaluation we use the hyper-parameters with lowest validation loss.

6.6 Additional Results

In this section, we present some additional results. First, we present results and discuss how well our multi-task policy performs when compared to single-task policies. Figure 6.13 compares single-task policy performance against two sets of multi-task policies for the *Heat Soup* activity. For the first multi-task Single-Activity policy (MT Single-Activity) we only train it across all tasks within

Heat Soup	Success	**Serve Soup**	Success	**Baking Prep**	Success
Flap-Open Oven	80%	Flap-Open Oven	90%	Slide-Open Drawer	70%
Pick Bowl	60%	Pick Bowl	50%	Pick Butter	70%
Slide-In Bowl	70%	Slide-Out Bowl	80%	place Butter	90%
Flap-Close Oven	50%	Flap-Close Oven	80%	Slide-Close Drawer	90%

Making Tea	Success	**Cleaning Up**	Success	**Stow Bowl**	Success
Uncap Lid	80%	Pick Lid	70%	Slide-Open Drawer	70%
Place Lid	90%	Cap Lid	100%	Pick Bowl	70%
Pick Tea	40%	Slide-Close Drawer	90%	Place Bowl	80%
Place Tea	60%	Flap-Close Oven	80%	Slide-Close Drawer	80%
Pick Lid	50%	Pick Towel	90%		
Cap Lid	70%	Wipe Counter	90%		

Table 6.5: Results for different tasks using the learned **universal policy**.

the same activity. For the latter multi-task universal multi-activity policy (MT-Universal) we train it across all tasks in all activities. From Figure 6.13 we see that for most tasks *MT Single-Activity* is able to outperform single task policies. Additionally, single-task policies are able to perform well on most tasks ($\approx 80\%$) except the more challenging constrained manipulation tasks (slide-in-bowl) ($\approx 20\%$). Finally, we also observe that MT-Single-Activity can outperform MT-Universal for most tasks. This happens because the universal agent is trained to perform a much larger variety of tasks. Given the very large variety of skills (Figure 6.3), such multi-task training can result in some negative transfer compared to training on a narrowly defined skills. We believe these reduced multi-task results present useful avenues for future research. Finally, in Table 6.5 we show results for all tasks in all activities using our single universal policy. From the below table, we see that the universal policy is able to perform well on most tasks except the more challenging tasks such as grasping small deformable objects (Pick Tea: 40%, Pick Lid: 50%).

6.7 Discussion and Limitations

We develop a framework for sample-efficient and generalizable multi-task robot manipulation in the real world. Our framework is based on rapidly

multiplying a small robotics dataset through semantic augmentations, and training a language-conditioned policy that can ingest the diverse multi-modal data. We combine and adapt several design choices like action chunking and temporal aggregation proposed in the context of single-task policies, and show that they yield significant boosts in performance even in multi-task settings. We also release one of the largest manipulation datasets to date involving over 12 skills in kitchen environments which we hope will facilitate further research in developing diverse real-world robot manipulation systems. A limitation of our work is that we do not consider composing skills across similar/different activities. Another limitation is that we do not explore the axes of language generalization, and use language embeddings from pre-trained encoders as is. Future work could investigate better language conditioning that is more flexibly adaptable to changes in task descriptions.

Chapter 7

Conclusion

In this thesis, we developed a scalable framework for learning generalizable robotic manipulation policies by leveraging passive web videos—ubiquitously available at scale—as a primary source of supervisory signal. Traditionally, robotic policy learning has been bottlenecked by the need for large-scale robot interaction data, which is expensive to collect and often limited in diversity. In contrast, human videos on the web capture rich and varied interactions with everyday objects across countless tasks and environments. Our core insight is that, although human and robot embodiments differ, the underlying structure of object interactions—what to manipulate, how it moves, and how it changes state—remains largely consistent and can be abstracted into transferable interaction plans. This insight forms the foundation for a new learning paradigm: predictive planning from passive video.

We introduced several key instantiations of this paradigm. We developed HOPMan (Hand-Object Plan for Manipulation), a framework that predicts future hand-object interaction masks from goal-conditioned human videos. These interaction plans are learned entirely from passive web video datasets and then translated into robot actions using a policy trained with a small amount of paired human-robot data. We also proposed a more expressive, embodiment-agnostic alternative, Track2Act: predicting object-centric point tracks from initial and goal images. This track-based interaction plan captures how arbitrary points on the object move across time and can be used to infer rigid body transformations for planning robot trajectories. A residual policy, trained on limited robot data, corrects any execution errors in a closed-loop

manner. We also show how pre-trained generative models like those for video prediction can enable visual interaction plan prediction in the form of generated human videos, in a zero-shot manner without requiring any adaptation. Finally, in addition to enabling interaction plan prediction, we show how web data can directly enable efficient imitation learning via semantic augmentations of a robot interaction dataset. All of these approaches require no online adaptation and support zero-shot execution in unseen scenes and with novel objects.

Across a suite of more than 100 real-world manipulation tasks—ranging from articulated object manipulation to tool use and non-prehensile actions—we demonstrated strong generalization across object categories, object instances, skills, and environmental scenes. Our systems performed manipulation in both static tabletop setups and dynamic in-the-wild settings such as kitchens and offices using a mobile robot base. Notably, we achieved this by using orders of magnitude less robot-specific data than prior work, highlighting the efficiency and practicality of our method.

Crucially, we show that web videos can serve as more than just a source of representation learning; they can directly guide robot behavior through learned interaction plans. This shifts the conventional view of generalization in robotics—from merely building robustness to variations in seen tasks, to synthesizing plausible trajectories for entirely unseen tasks using large-scale passive supervision. Unlike prior approaches that require manually aligned human-robot demonstrations or extensive fine-tuning during deployment, our approach is fully zero-shot, requiring no task-specific adaptation at test time.

While our results are promising, several challenges remain. Accurately capturing fine-grained contact dynamics and finger articulations remains difficult with passive videos, limiting performance on highly dexterous tasks. Moreover, long-horizon task composition and reasoning remains an open area for future work. Nevertheless, by demonstrating that predictive planning from web data can serve as a viable foundation for robotic skill acquisition, this thesis opens new avenues for building generalist, scalable, and practical robot learning systems that operate beyond the confines of curated lab settings in the open-world.

Bibliography

[1] J.-B. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

[2] K. Ashutosh, R. Girdhar, L. Torresani, and K. Grauman. Hiervl: Learning hierarchical video-language embeddings. *arXiv preprint arXiv:2301.02311*, 2023.

[3] S. Baek, K. I. Kim, and T.-K. Kim. Pushing the envelope for rgb-based dense 3d hand pose estimation via neural rendering. In *CVPR*, 2019.

[4] S. Bahl, A. Gupta, and D. Pathak. Human-to-robot imitation in the wild. *RSS*, 2022.

[5] S. Bahl, R. Mendonca, L. Chen, U. Jain, and D. Pathak. Affordances from human videos as a versatile representation for robotics. In *CVPR*, 2023.

[6] C. Bao, J. Xu, X. Wang*, A. Gupta*, and H. Bharadhwaj*. Handsonvlm: Vision-language models for hand-object interaction prediction. *Under Review*, 2024.

[7] C. Bao, J. Xu, X. Wang, A. Gupta, and H. Bharadhwaj. Handsonvlm: Vision-language models for hand-object interaction prediction. *arXiv preprint arXiv:2412.13187*, 2024.

[8] L. Berscheid, T. Rühr, and T. Kröger. Improving data efficiency of self-supervised learning for robotic grasping. In *2019 International*

Conference on Robotics and Automation (ICRA), pages 2125–2131. IEEE, 2019.

[9] H. Bharadhwaj, D. Dwibedi, A. Gupta, S. Tulsiani, C. Doersch, T. Xiao, D. Shah, F. Xia, D. Sadigh, and S. Kirmani. Gen2act: Human video generation in novel scenarios enables generalizable robot manipulation. *arxiv*, 2024.

[10] H. Bharadhwaj, A. Gupta, V. Kumar, and S. Tulsiani. Towards generalizable zero-shot manipulation via translating human interaction plans. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[11] H. Bharadhwaj, A. Gupta, and S. Tulsiani. Visual affordance prediction for guiding robot exploration. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[12] H. Bharadhwaj, A. Gupta, and S. Tulsiani. Visual affordance prediction for guiding robot exploration. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3029–3036. IEEE, 2023.

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

[14] H. Bharadhwaj, A. Gupta*, S. Tulsiani*, and V. Kumar*. Towards zero-shot diverse manipulation via translating human plans. *International Conference on Robotics and Automation (ICRA)*, 2024.

[15] H. Bharadhwaj, R. Mottaghi, A. Gupta, and S. Tulsiani. Track2act: Predicting point tracks from internet videos enables diverse zero-shot robot manipulation. *arXiv preprint arXiv:2405.01527*, 2024.

[16] H. Bharadhwaj, R. Mottaghi*, A. Gupta*, and S. Tulsiani*. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. *European Conference on Computer Vision (ECCV)*, 2024.

[17] H. Bharadhwaj*, J. Vakil*, M. Sharma*, A. Gupta, S. Tulsiani, and V. Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. *International Conference on Robotics and Automation (ICRA)*, 2024.

[18] K. Black, M. Nakamoto, P. Atreya, H. Walke, C. Finn, A. Kumar, and S. Levine. Zero-shot robotic manipulation with pretrained image-editing diffusion models. *arXiv preprint arXiv:2310.10639*, 2023.

[19] A. Boukhayma, R. d. Bem, and P. H. Torr. 3d hand shape and pose from images in the wild. In *CVPR*, 2019.

[20] K. Bousmalis, G. Vezzani, D. Rao, C. Devin, A. X. Lee, M. Bauza, T. Davchev, Y. Zhou, A. Gupta, A. Raju, et al. Robocat: A self-improving foundation agent for robotic manipulation. *arXiv preprint arXiv:2306.11706*, 2023.

[21] S. Brahmbhatt, A. Handa, J. Hays, and D. Fox. Contactgrasp: Functional multi-finger grasp synthesis from contact. *arXiv*, 2019.

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

[23] A. Brohan, Y. Chebotar, C. Finn, K. Hausman, A. Herzog, D. Ho, J. Ibarz, A. Irpan, E. Jang, R. Julian, et al. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on Robot Learning*, pages 287–318. PMLR, 2023.

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

[25] A. C and A. D. Large language and vision models for 3d interaction understanding. *arXiv preprint arXiv:2404.06507*, 2024.

[26] F. Caba Heilbron, V. Escorcia, B. Ghanem, and J. Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 961–970, 2015.

[27] Z. Chen, S. Kiami, A. Gupta, and V. Kumar. Genaug: Retargeting behaviors to unseen situations via generative augmentation. *arXiv preprint arXiv:2302.06671*, 2023.

[28] Z. Chen*, Z. Mandi*, H. Bharadhwaj*, M. Sharma, S. Song, A. Gupta, and V. Kumar. Semantically controllable augmentations for generalizable robot learning. *International Journal of Robotics Research (IJRR)*, 2024.

[29] Z. J. Cui, Y. Wang, N. Muhammad, L. Pinto, et al. From play to policy: Conditional behavior generation from uncurated robot data. *arXiv preprint arXiv:2210.10047*, 2022.

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

[31] A. Darkhalil, D. Shan, B. Zhu, J. Ma, A. Kar, R. Higgins, S. Fidler, D. Fouhey, and D. Damen. Epic-kitchens visor benchmark: Video segmentations and object relations. *Advances in Neural Information Processing Systems*, 35:13745–13758, 2022.

[32] P. Das, C. Xu, R. F. Doell, and J. J. Corso. A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2634–2641, 2013.

[33] 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. In *Conference on Robot Learning*, pages 885–897. PMLR, 2020.

[34] S. Dasari, J. Wang, J. Hong, S. Bahl, Y. Lin, A. Wang, A. Thankaraj, K. Chahal, B. Calli, S. Gupta, et al. Rb2: Robotic manipulation benchmarking with a twist. *arXiv preprint arXiv:2203.08098*, 2022.

[35] F. De la Torre, J. Hodgins, A. Bargteil, X. Martin, J. Macey, A. Collado, and P. Beltran. Guide to the carnegie mellon university multimodal activity (cmu-mmac) database. 2009.

[36] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[37] 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.

[38] C. Doersch, A. Gupta, L. Markeeva, A. Recasens, L. Smaira, Y. Aytar, J. Carreira, A. Zisserman, and Y. Yang. Tap-vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems*, 35:13610–13626, 2022.

[39] C. Doersch, Y. Yang, D. Gokay, P. Luc, S. Koppula, A. Gupta, J. Heyward, R. Goroshin, J. Carreira, and A. Zisserman. Bootstap: Bootstrapped training for tracking-any-point. *arXiv preprint arXiv:2402.00847*, 2024.

[40] Y. Du, S. Yang, B. Dai, H. Dai, O. Nachum, J. Tenenbaum, D. Schuurmans, and P. Abbeel. Learning universal policies via text-guided video generation. *Advances in Neural Information Processing Systems*, 36, 2024.

[41] F. Ebert, C. Finn, A. X. Lee, and S. Levine. Self-supervised visual planning with temporal skip connections. *CoRL*, 12:16, 2017.

[42] 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.

[43] L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. *arXiv preprint arXiv:1802.01561*, 2018.

[44] H.-S. Fang, H. Fang, Z. Tang, J. Liu, J. Wang, H. Zhu, and C. Lu. Rh20t: A robotic dataset for learning diverse skills in one-shot. *arXiv preprint arXiv:2307.00595*, 2023.

[45] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-shot visual imitation learning via meta-learning. In *Conference on robot learning*, pages 357–368. PMLR, 2017.

[46] T.-J. Fu, L. Yu, N. Zhang, C.-Y. Fu, J.-C. Su, W. Y. Wang, and S. Bell. Tell me what happened: Unifying text-guided video completion via multimodal masked video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10681–10692, 2023.

[47] S. Y. Gadre, M. Wortsman, G. Ilharco, L. Schmidt, and S. Song. Clip on wheels: Zero-shot object navigation as object localization and exploration. *arXiv preprint arXiv:2203.10421*, 2022.

[48] L. Ge, Z. Ren, Y. Li, Z. Xue, Y. Wang, J. Cai, and J. Yuan. 3d hand shape and pose estimation from a single rgb image. In *CVPR*, 2019.

[49] R. Girdhar, M. Singh, A. Brown, Q. Duval, S. Azadi, S. S. Rambhatla, A. Shah, X. Yin, D. Parikh, and I. Misra. Emu video: Factorizing text-to-video generation by explicit image conditioning. *arXiv preprint arXiv:2311.10709*, 2023.

[50] A. Goyal, A. Mousavian, C. Paxton, Y.-W. Chao, B. Okorn, J. Deng, and D. Fox. Ifor: Iterative flow minimization for robotic object rearrangement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14787–14797, 2022.

[51] M. Goyal, S. Modi, R. Goyal, and S. Gupta. Human hands as probes for interactive object understanding. In *Proceedings of the IEEE/CVF*

Conference on Computer Vision and Pattern Recognition, pages 3293–3303, 2022.

[52] R. Goyal, S. Ebrahimi Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017.

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

[54] J. Gu, S. Kirmani, P. Wohlhart, Y. Lu, M. G. Arenas, K. Rao, W. Yu, C. Fu, K. Gopalakrishnan, Z. Xu, et al. Rt-trajectory: Robotic task generalization via hindsight trajectory sketches. *arXiv preprint arXiv:2311.01977*, 2023.

[55] A. Gupta, J. Yu, T. Z. Zhao, V. Kumar, A. Rovinsky, K. Xu, T. Devlin, and S. Levine. Reset-free reinforcement learning via multi-task learning: Learning dexterous manipulation behaviors without human intervention. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6664–6671. IEEE, 2021.

[56] A. Gupta, L. Yu, K. Sohn, X. Gu, M. Hahn, L. Fei-Fei, I. Essa, L. Jiang, and J. Lezama. Photorealistic video generation with diffusion models. *arXiv preprint arXiv:2312.06662*, 2023.

[57] D. Ha and J. Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[58] S. Haddadin, S. Parusel, L. Johannsmeier, S. Golz, S. Gabl, F. Walch, M. Sabaghian, C. Jähne, L. Hausperger, and S. Haddadin. The franka emika robot: A reference platform for robotics research and education. *IEEE Robotics & Automation Magazine*, 29(2):46–64, 2022.

[59] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

[60] S. Haldar, V. Mathur, D. Yarats, and L. Pinto. Watch and match: Supercharging imitation with regularized optimal transport. In *Conference on Robot Learning*, pages 32–43. PMLR, 2023.

[61] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. Van Wyk, A. Zhurkevich, B. Sundaralingam, et al. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5977–5984. IEEE, 2023.

[62] N. Hansen, Z. Yuan, Y. Ze, T. Mu, A. Rajeswaran, H. Su, H. Xu, and X. Wang. On pre-training for visuo-motor control: Revisiting a learning-from-scratch baseline. *arXiv preprint arXiv:2212.05749*, 2022.

[63] Y. Hasson, G. Varol, D. Tzionas, I. Kalevatykh, M. J. Black, I. Laptev, and C. Schmid. Learning joint reconstruction of hands and manipulated objects. In *CVPR*, 2019.

[64] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

[65] Y. He, W. Sun, H. Huang, J. Liu, H. Fan, and J. Sun. Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation. *CVPR*, 2020.

[66] Y. Hu, J. Hugonot, P. Fua, and M. Salzmann. Segmentation-driven 6d object pose estimation. *CVPR*, 2019.

[67] U. Iqbal, P. Molchanov, T. Breuel Juergen Gall, and J. Kautz. Hand pose estimation via latent 2.5 d heatmap regression. In *ECCV*, 2018.

[68] V. Jain, M. Attarian, N. J. Joshi, A. Wahid, D. Driess, Q. Vuong, P. R. Sanketi, P. Sermanet, S. Welker, C. Chan, et al. Vid2robot: End-to-end video-conditioned policy learning with cross-attention transformers. *arXiv preprint arXiv:2403.12943*, 2024.

[69] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

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

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

[72] 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.

[73] L. Kaiser, M. Babaeizadeh, P. Milos, B. Osinski, R. H. Campbell, K. Czechowski, D. Erhan, C. Finn, P. Kozakowski, S. Levine, et al. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*, 2019.

[74] 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.

[75] D. Kalashnikov, J. Varley, Y. Chebotar, B. Swanson, R. Jonschkowski, C. Finn, S. Levine, and K. Hausman. Mt-opt: Continuous multi-task robotic reinforcement learning at scale. *arXiv preprint arXiv:2104.08212*, 2021.

[76] I. Kapelyukh, V. Vosylius, and E. Johns. Dall-e-bot: Introducing web-scale diffusion models to robotics. *arXiv preprint arXiv:2210.02438*, 2022.

[77] I. Kapelyukh, V. Vosylius, and E. Johns. Dall-e-bot: Introducing web-scale diffusion models to robotics. *IEEE Robotics and Automation Letters*, 8(7):3956–3963, 2023.

[78] N. Karaev, I. Rocco, B. Graham, N. Neverova, A. Vedaldi, and C. Rupprecht. Cotracker: It is better to track together. *arXiv preprint arXiv:2307.07635*, 2023.

[79] S. Karamcheti, S. Nair, A. S. Chen, T. Kollar, C. Finn, D. Sadigh, and P. Liang. Language-driven representation learning for robotics. *arXiv preprint arXiv:2302.12766*, 2023.

[80] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[81] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. *ICCV*, 2017.

[82] L. Keselman, J. Iselin Woodfill, A. Grunnet-Jepsen, and A. Bhowmik. Intel realsense stereoscopic depth cameras. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 1–10, 2017.

[83] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[84] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

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

[86] P.-C. Ko, J. Mao, Y. Du, S.-H. Sun, and J. B. Tenenbaum. Learning to act from actionless videos through dense correspondences. *arXiv preprint arXiv:2310.08576*, 2023.

[87] D. Kondratyuk, L. Yu, X. Gu, J. Lezama, J. Huang, R. Hornung, H. Adam, H. Akbari, Y. Alon, V. Birodkar, et al. Videopoet: A large language model for zero-shot video generation. *arXiv preprint arXiv:2312.14125*, 2023.

[88] D. Kulon, R. A. Guler, I. Kokkinos, M. M. Bronstein, and S. Zafeiriou. Weakly-supervised mesh-convolutional hand reconstruction in the wild. In *CVPR*, 2020.

[89] V. Kumar and E. Todorov. Mujoco haptix: A virtual reality system for hand manipulation. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 657–663. IEEE, 2015.

[90] S. L. and Others. Hoi-forecast: Predicting future hand-object interactions. *Proceedings of CVPR*, 2024.

[91] Y. Li, M. Liu, and J. M. Rehg. In the eye of beholder: Joint learning of gaze and actions in first person video. In *Proceedings of the European conference on computer vision (ECCV)*, pages 619–635, 2018.

[92] J. Liang, R. Liu, E. Ozguroglu, S. Sudhakar, A. Dave, P. Tokmakov, S. Song, and C. Vondrick. Dreamitate: Real-world visuomotor policy learning via video generation. *arXiv preprint arXiv:2406.16862*, 2024.

[93] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. T. Freeman. Sift flow: Dense correspondence across different scenes. In *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part III 10*, pages 28–42. Springer, 2008.

[94] S. Liu, H. Jiang, J. Xu, S. Liu, and X. Wang. Semi-supervised 3d hand-object poses estimation with interactions in time. In *CVPR*, 2021.

[95] S. Liu, S. Tripathi, S. Majumdar, and X. Wang. Joint hand motion and interaction hotspots prediction from egocentric videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3282–3292, 2022.

[96] 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*, pages 1113–1132. PMLR, 2020.

[97] C. Lynch and P. Sermanet. Language conditioned imitation learning over unstructured data. *arXiv preprint arXiv:2005.07648*, 2020.

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

[99] N. M. Mahi Shafiullah, A. Rai, H. Etukuru, Y. Liu, I. Misra, S. Chintala, and L. Pinto. On bringing robots home. *arXiv e-prints*, pages arXiv–2311, 2023.

[100] A. Majumdar, K. Yadav, S. Arnaud, Y. J. Ma, C. Chen, S. Silwal, A. Jain, V.-P. Berges, P. Abbeel, J. Malik, et al. Where are we in the search for an artificial visual cortex for embodied intelligence? *arXiv preprint arXiv:2303.18240*, 2023.

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

[102] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning*, pages 1678–1690. PMLR, 2022.

[103] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pages 879–893. PMLR, 2018.

[104] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pages 879–893. PMLR, 2018.

[105] C. Mitash, F. Wang, S. Lu, V. Terhuja, T. Garaas, F. Polido, and M. Nambi. Armbench: An object-centric benchmark dataset for robotic manipulation. *arXiv preprint arXiv:2303.16382*, 2023.

[106] H. Mittal, P. Morgado, U. Jain, and A. Gupta. Learning state-aware visual representations from audible interactions. *arXiv preprint arXiv:2209.13583*, 2022.

[107] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, et al. Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics and Automation Letters*, 2023.

[108] K. Mo, L. J. Guibas, M. Mukadam, A. Gupta, and S. Tulsiani. Where2act: From pixels to actions for articulated 3d objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6813–6823, 2021.

[109] L. Momeni, M. Caron, A. Nagrani, A. Zisserman, and C. Schmid. Verbs in action: Improving verb understanding in video-language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15579–15591, 2023.

[110] T. Nagarajan, C. Feichtenhofer, and K. Grauman. Grounded human-object interaction hotspots from video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8688–8697, 2019.

[111] A. V. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine. Visual reinforcement learning with imagined goals. *Advances in neural information processing systems*, 31, 2018.

[112] 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.

[113] A. Nguyen, D. Kanoulas, L. Muratore, D. G. Caldwell, and N. G. Tsagarakis. Translating videos to commands for robotic manipulation with deep recurrent neural networks. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3782–3788. IEEE, 2018.

[114] A. Padalkar, A. Pooley, A. Jain, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Singh, A. Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[115] N. D. Palo and E. Johns. Dinobot: Robot manipulation via retrieval and alignment with vision foundation models. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[116] C. Pan, B. Okorn, H. Zhang, B. Eisner, and D. Held. Tax-pose: Task-specific cross-pose estimation for robot manipulation. In *Conference on Robot Learning*, pages 1783–1792. PMLR, 2023.

[117] G. Papagiannis, N. Di Palo, P. Vitiello, and E. Johns. R+ x: Retrieval and execution from everyday human videos. *arXiv preprint arXiv:2407.12957*, 2024.

[118] S. Parisi, A. Rajeswaran, S. Purushwalkam, and A. Gupta. The unsurprising effectiveness of pre-trained vision models for control. *arXiv preprint arXiv:2203.03580*, 2022.

[119] S. Parisi, A. Rajeswaran, S. Purushwalkam, and A. Gupta. The unsurprising effectiveness of pre-trained vision models for control. *arXiv preprint arXiv:2203.03580*, 2022.

[120] W. Peebles and S. Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[121] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[122] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel. Asymmetric actor critic for image-based robot learning. *arXiv preprint arXiv:1710.06542*, 2017.

[123] L. Pinto and A. Gupta. Learning to push by grasping: Using multiple tasks for effective learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2161–2168. IEEE, 2017.

[124] Y. Qin, Y.-H. Wu, S. Liu, H. Jiang, R. Yang, Y. Fu, and X. Wang. Dexmv: Imitation learning for dexterous manipulation from human videos. *arXiv preprint arXiv:2108.05877*, 2021.

[125] Z. Qin, K. Fang, Y. Zhu, L. Fei-Fei, and S. Savarese. Keto: Learning keypoint representations for tool manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7278–7285. IEEE, 2020.

[126] M. Rad and V. Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. *ICCV*, 2017.

[127] K. Rao, C. Harris, A. Irpan, S. Levine, J. Ibarz, and M. Khansari. Rl-cyclegan: Reinforcement learning aware simulation-to-real. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11157–11166, 2020.

[128] 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.

[129] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[130] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models, 2021.

[131] Y. Rong, T. Shiratori, and H. Joo. Frankmocap: Fast monocular 3d hand and body motion capture by regression and integration. *arXiv preprint arXiv:2008.08324*, 2020.

[132] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. Denton, S. K. S. Ghasemipour, B. K. Ayan, S. S. Mahdavi, R. G. Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022.

[133] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[134] D. Seita, Y. Wang, S. J. Shetty, E. Y. Li, Z. Erickson, and D. Held. Toolflownet: Robotic manipulation with tools via predicting tool flow from point clouds. In *Conference on Robot Learning*, pages 1038–1049. PMLR, 2023.

[135] N. M. M. Shafiullah, Z. J. Cui, A. Altanzaya, and L. Pinto. Behavior transformers: Cloning k modes with one stone. *arXiv preprint arXiv:2206.11251*, 2022.

[136] R. Shah and V. Kumar. Rrl: Resnet as representation for reinforcement learning. *arXiv preprint arXiv:2107.03380*, 2021.

[137] D. Shan, J. Geng, M. Shu, and D. Fouhey. Understanding human hands in contact at internet scale. In *CVPR*, 2020.

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

[139] M. Sharma, C. Fantacci, Y. Zhou, S. Koppula, N. Heess, J. Scholz, and Y. Aytar. Lossless adaptation of pretrained vision models for robotic manipulation. *arXiv preprint arXiv:2304.06600*, 2023.

[140] K. Shaw, S. Bahl, and D. Pathak. Videodex: Learning dexterity from internet videos. In *6th Annual Conference on Robot Learning*.

[141] K. Shaw, S. Bahl, and D. Pathak. Videodex: Learning dexterity from internet videos. In *Conference on Robot Learning*, pages 654–665. PMLR, 2023.

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

[143] L. Smith, N. Dhawan, M. Zhang, P. Abbeel, and S. Levine. Avid: Learning multi-stage tasks via pixel-level translation of human videos. *arXiv*, 2019.

[144] S. Sodhani, A. Zhang, and J. Pineau. Multi-task reinforcement learning with context-based representations. In *International Conference on Machine Learning*, pages 9767–9779. PMLR, 2021.

[145] H. F. Song, A. Abdolmaleki, J. T. Springenberg, A. Clark, H. Soyer, J. W. Rae, S. Noury, A. Ahuja, S. Liu, D. Tirumala, et al. V-mpo: On-policy maximum a posteriori policy optimization for discrete and continuous control. *arXiv preprint arXiv:1909.12238*, 2019.

[146] A. Spurr, J. Song, S. Park, and O. Hilliges. Cross-modal deep variational hand pose estimation. In *CVPR*, 2018.

[147] S. Stepputtis, J. Campbell, M. Phielipp, S. Lee, C. Baral, and H. Ben Amor. Language-conditioned imitation learning for robot ma-

nipulation tasks. *Advances in Neural Information Processing Systems*, 33:13139–13150, 2020.

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

[149] S. Tan, T. Nagarajan, and K. Grauman. Egodistill: Egocentric head motion distillation for efficient video understanding. *arXiv preprint arXiv:2301.02217*, 2023.

[150] G. Team, R. Anil, S. Borgeaud, Y. Wu, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

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

[152] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[153] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *IROS*, 2012.

[154] M. Vecerik, C. Doersch, Y. Yang, T. Davchev, Y. Aytar, G. Zhou, R. Hadsell, L. Agapito, and J. Scholz. Robotap: Tracking arbitrary points for few-shot visual imitation. *arXiv preprint arXiv:2308.15975*, 2023.

[155] V. Voleti, A. Jolicoeur-Martineau, and C. Pal. Masked conditional video diffusion for prediction, generation, and interpolation. *arXiv preprint arXiv:2205.09853*, 2022.

[156] H. R. Walke, K. Black, T. Z. Zhao, Q. Vuong, C. Zheng, P. Hansen-Estruch, A. W. He, V. Myers, M. J. Kim, M. Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023.

[157] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. *arXiv preprint arXiv:2302.12422*, 2023.

[158] C. Wen, X. Lin, J. So, K. Chen, Q. Dou, Y. Gao, and P. Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023.

[159] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv*, 2018.

[160] T. Xiao, I. Radosavovic, T. Darrell, and J. Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.

[161] H. Xiong, Q. Li, Y.-C. Chen, H. Bharadhwaj, S. Sinha, and A. Garg. Learning by watching: Physical imitation of manipulation skills from human videos. *arXiv*, 2021.

[162] H. Xu, J. Zhang, J. Cai, H. Rezatofighi, F. Yu, D. Tao, and A. Geiger. Unifying flow, stereo and depth estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[163] W. Yan, D. Hafner, S. James, and P. Abbeel. Temporally consistent transformers for video generation. *arXiv preprint arXiv:2210.02396*, 2022.

[164] J. Yang, M. Gao, Z. Li, S. Gao, F. Wang, and F. Zheng. Track anything: Segment anything meets videos. *arXiv preprint arXiv:2304.11968*, 2023.

[165] M. Yang, Y. Du, K. Ghasemipour, J. Tompson, D. Schuurmans, and P. Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 2023.

[166] J. Ye et al. Diffhoi: Diffusion-guided 3d hand-object interaction understanding. *arXiv preprint arXiv:2404.08907*, 2024.

[167] Y. Ye, X. Li, A. Gupta, S. De Mello, S. Birchfield, J. Song, S. Tulsiani, and S. Liu. Affordance diffusion: Synthesizing hand-object interactions. In *CVPR*, pages 22479–22489, 2023.

[168] S. Young, D. Gandhi, S. Tulsiani, A. Gupta, P. Abbeel, and L. Pinto. Visual imitation made easy. In *Conference on Robot Learning (CoRL)*, 2020.

[169] 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*, 33:5824–5836, 2020.

[170] 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 *Conference on robot learning*, pages 1094–1100. PMLR, 2020.

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

[172] C. Yuan, C. Wen, T. Zhang, and Y. Gao. General flow as foundation affordance for scalable robot learning. *arXiv preprint arXiv:2401.11439*, 2024.

[173] L. Zhang, S. Zhou, S. Stent, and J. Shi. Fine-grained egocentric hand-object segmentation: Dataset, model, and applications. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIX*, pages 127–145. Springer, 2022.

[174] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[175] Y. Zhao, I. Misra, P. Krähenbühl, and R. Girdhar. Learning video representations from large language models. In *arXiv preprint arXiv:2212.04501*, 2022.

[176] Y. Zhou, Y. Aytar, and K. Bousmalis. Manipulator-independent representations for visual imitation. *arXiv preprint arXiv:2103.09016*, 2021.

[177] Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, S. Nasiriany, and Y. Zhu. robosuite: A modular simulation framework and benchmark for robot learning. *arXiv preprint arXiv:2009.12293*, 2020.

[178] C. Zimmermann and T. Brox. Learning to estimate 3d hand pose from single rgb images. In *CVPR*, 2017.

[179] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.