Robotics: Science and Systems 2024
Delft, Netherlands, July 15-July 19, 2024

Efficient Data Collection for Robotic Manipulation via Compositional Generalization

Jensen Gao[1], Annie Xie[1], Ted Xiao[2], Chelsea Finn[1,2], Dorsa Sadigh[1,2]
[1]Stanford University, [2]Google DeepMind

***Abstract*—Data collection has become an increasingly important problem in robotic manipulation, yet there still lacks much understanding of how to effectively collect data to facilitate broad generalization. Recent works on large-scale robotic data collection typically vary many environmental factors of variation (e.g., object types, table textures) during data collection, to cover a diverse range of scenarios. However, they do not explicitly account for the possible compositional abilities of policies trained on the data. If robot policies can compose environmental factors from their data to succeed when encountering unseen factor combinations, we can exploit this to avoid collecting data for situations that composition would address. To investigate this possibility, we conduct thorough empirical studies both in simulation and on a real robot that compare data collection strategies and assess whether visual imitation learning policies can compose environmental factors. We find that policies do exhibit composition, although leveraging prior robotic datasets is critical for this on a real robot. We use these insights to propose better in-domain data collection strategies that exploit composition, which can induce better generalization than naïve approaches for the same amount of effort during data collection. We further demonstrate that a real robot policy trained on data from such a strategy achieves a success rate of 77.5% when transferred to entirely new environments that encompass unseen combinations of environmental factors, whereas policies trained using data collected without accounting for environmental variation fail to transfer effectively, with a success rate of only 2.5%. We provide videos at our project website.[1]**

I. Introduction

For robots to be practical and useful, they must be robust to the wide variety of conditions they will encounter in the world. Recent advances in machine learning have shown that leveraging diverse, internet-scale data can be very effective in facilitating this kind of broad generalization [7, 40], as such data captures much of the complexity and variation in the domain that models will need to reason about. It therefore has been of great interest in the robotics community to apply a similar recipe to robotics, specifically using end-to-end imitation learning, which holds promise to effectively scale with large, diverse datasets to achieve broad generalization. However, we lack access to existing internet-scale robotics data, and furthermore, robotics data – especially human-collected demonstrations – is often limited and challenging to collect. While recent works have made significant progress in scaling real-world robotic manipulation datasets [47, 17, 36, 2], their scale is still much less than the datasets typically used for pre-training in computer vision and natural language processing. As a result, policies trained on such datasets often still exhibit unsatisfactory zero-shot performance when deployed in new environments, necessitating additional in-domain data collection for generalization. Given the lack of large-scale robotics data and the high cost of data collection, it is essential to focus on *how to best collect this in-domain data.*

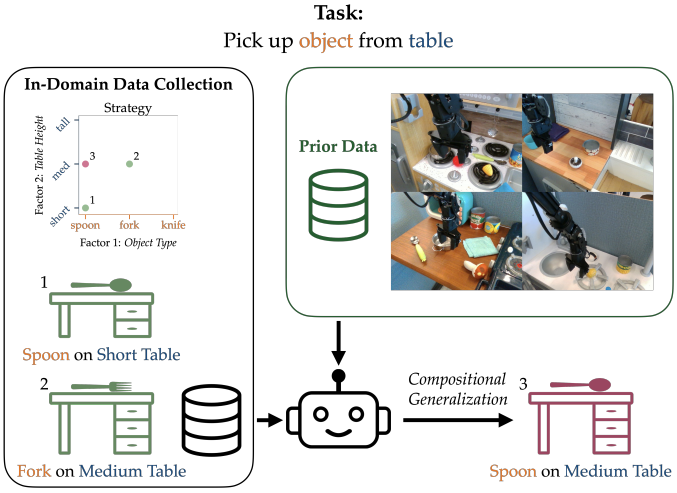


Fig. 1: We investigate if robot policies can compose environmental factors of variation (e.g., object types, table heights) in their in-domain training data (1, 2), and if using prior data can be helpful for enforcing such generalization. We propose efficient in-domain data collection strategies guided by the ability of policies to reason about unseen combinations of factor values (3).

When collecting robotics data to account for a given range of scenarios, it is often infeasible to cover all possible settings. Instead, it would be desirable to exploit the *compositional generalization* capabilities of the policies being trained, i.e., their ability to reason effectively about unseen combinations of environmental factors of variation. For example, for the task of picking up different objects from tables of different heights, as shown in Fig. 1, it would simply not be scalable for humans to collect data for *all* combinations of desired objects and table heights. Instead, it would be much more feasible if one could prioritize collecting data to cover all individual examples of objects and table heights (1 and 2 in Fig. 1), but not necessarily all of their combinations, and then have a policy trained on this data generalize to unseen combinations (3 in Fig. 1). By doing so, we could drastically reduce the amount of data needed.

While prior work has studied generalization to unseen environmental factors in robotics [50, 49, 39], there has not been as much focus on understanding compositional generalization in robotics, particularly for end-to-end imitation learning. Compositional generalization has been previously studied in

[1]http://iliad.stanford.edu/robot-data-comp/

other machine learning domains [20, 1, 26, 23, 24, 44, 4, 52], with results often suggesting that end-to-end neural models can struggle to achieve this. However, recent work has shown that large language models can exhibit strong compositional abilities, likely due to the diversity and scale of their training data [54, 13]. Ideally, prior datasets in robotics could also facilitate composition for robotics, but we currently lack much understanding as to where these datasets benefit generalization.

Our key idea is that understanding composition can provide more guidance for efficient data collection in robotics. While there has been a recent trend in scaling up robotic data collection, most prior works attempt to simply collect *diverse* data covering all possible combinations of different factors of variation, without explicitly accounting for composition [47, 17, 36, 2]. However, we hypothesize that imitation learning policies – similar to their large language model counterparts – can potentially exhibit compositional generalization, in particular when leveraging prior robotics datasets. In this work, we investigate this hypothesis, with the goal of prescribing better practices for robotic data collection.

Through extensive simulated and real experiments, we study the proficiency of visual imitation learning policies when evaluated on unseen combinations of environmental factors. We consider a wide variety of factors that are broadly relevant in robotic manipulation and have been previously studied, including *object position*, *object type*, *table texture*, *table height*, *camera position*, and *distractor objects*. We find significant evidence of compositional generalization with imitation policies when varying these factors. Given this evidence, we propose in-domain data collection strategies that exploit this generalization – providing more systematic guidelines that can help reduce data collection burden for roboticists. On a real robot, we find that a policy using data from such a strategy succeeds in **59/90** settings that assess composition, compared to **22/90** when using the same amount of in-domain data, but without explicit variation. We also find that using prior robot data – in our case, BridgeData V2 [47] – is critical for this composition, with performance dropping to **28/90** without it. Furthermore, we evaluate policies when transferred to entirely new environments that encompass unseen combinations of environmental factors, and find that our best policy achieves a success rate of **77.5%**, compared to **27.5%** when there is no prior data, and **2.5%** when there is no variation in the in-domain data from our training environments.

II. Related Work

In this section, we review prior work on robotic data collection, generalization in robot learning, and compositional generalization more broadly in machine learning.

Data Collection in Robot Learning. Prior works have studied data collection methods in robot learning to improve generalization. Much of this work focuses on scaling up robot data collection [38, 15, 28, 33, 45, 10, 16, 5, 47, 17, 2]. Results from these works have demonstrated that data with greater object and task diversity can improve policy robustness, suggesting that data should be collected with this diversity in mind. In our work, we study how using such datasets as prior data affects compositional generalization. However, results from these works also show that policies trained on these datasets often exhibit poor zero-shot performance in new scenarios, suggesting that in-domain data collection is still often needed in practice. Unlike these prior works, our work focuses on how to collect this in-domain data, by studying how in-domain data composition affects generalization at a more nuanced level, and systematically considering a wide variety of specific environmental factors of variation.

Other works on data collection in imitation learning focus on expanding state coverage via interactive imitation learning [43, 27, 22], or more recently improving other notions of data quality [3]. These works mainly consider the impact of data on the distribution shift problem [43], to improve policy robustness to the state distribution the agent will encounter online. However, distribution shift in robotics can also occur due to environmental variations. We instead focus on addressing this form of distribution shift, through offline data collection with active variation of environmental factors.

Generalization in Robot Learning. Much prior work has studied improving generalization in robot learning, such as by using pre-trained visual representations [37, 34, 31, 41, 11, 32, 21], or leveraging diverse robot data [36, 35]. In our work, we focus on assessing compositional generalization of existing imitation learning methods. Some prior works have similarly studied the robustness of visual imitation learning to environmental factor shifts [50, 49, 8, 39]. However, these works consider generalization to unseen factor values, and primarily focus on how factors individually affect generalization, without extensive focus on how factors interact. Our work instead assesses compositional generalization, where policies must reason about *unseen combinations* of factor values.

Compositional Generalization. Compositional generalization has been studied extensively in many machine learning areas, including visual reasoning [44, 52], image generation [14, 29], natural language processing [26, 23, 24], and visual question answering [20, 1, 4]. Much of these works demonstrate that end-to-end neural models often struggle with composition, although sometimes they can exhibit better composition than methods specifically intended to facilitate it, such as unsupervised representation learning [44]. However, more recent work has shown that large language models can possess strong compositional abilities, likely owing to their internet-scale pretraining data [54, 13]. In our work, we study if leveraging prior data can also be beneficial for composition in robotics.

Compositional generalization has also been investigated in robotics. This includes work on modular approaches for promoting composition [12, 51, 25, 48], and end-to-end architectures with inductive biases that benefit composition [53]. However, these works primarily study composition at the semantic level, for understanding concepts such as text instructions, high-level skills, or object properties and relationships. Unlike these works, we study composition of *environmental factors*, which may significantly change visual observations, or the

low-level physical motions needed for a task. In addition, we study composition for existing end-to-end imitation learning methods. Such approaches have become popular in robotics and hold promise to scale effectively with data.

There has been some evidence of compositional generalization in end-to-end robotic imitation learning. This includes composition of tasks in prior data with the conditions of a target domain [16], objects with manipulation skills [5, 46], semantic concepts from internet data with manipulation skills [6], and behaviors and semantic concepts across robot embodiments [5, 36]. However, our work provides more systematic and fine-grained analysis on when composition is possible, and considers a greater variety of environmental factors. Furthermore, unlike these works, we focus on using this analysis to guide better practices for data collection that take composition into account. We believe these key differences allow our insights to be leveraged for more effective data collection in real-world settings.

III. Exploiting Compositional Generalization for Efficient Data Collection

A. Problem Statement

We consider the goal-conditioned imitation learning setting, where the objective is for a robot to reach a goal. We formalize this by defining the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, H, \mathcal{F}^N, p(f), \rho(s_0|f), \mu(\cdot|f))$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, $\mathcal{T}(s'|s,a)$ is the transition dynamics function, and H is the horizon length. We refer to $\mathcal{F}^N \subset \mathbb{Z}^N$ as the *factor space*, which captures changes in the environment along N different axes that the robot could encounter, which we refer to as *factors*. For example, the first component of $\mathcal{F}^N$ could capture variation in *object type*, and the second could capture variation in *table height*, etc. For simplicity, we assume each component of $\mathcal{F}^N$ consists of k possible discrete values, e.g., there are k possible object types the robot may encounter. We refer to these as *factor values*. Therefore, each element $f \in \mathcal{F}^N$ defines a combination of factor values for the environment, e.g., a specific *object type* and *table height* the robot may encounter. $p(f)$ is the distribution of factor value combinations, and $\rho(s_0|f)$ is the initial state distribution, conditioned on a combination of factor values. We adopt this formalism to express that factor values in the environment can affect tasks by inducing different initial states. Similarly, $\mu(f) : \mathcal{F}^N \to S$ is a deterministic function that maps a combination of factor values to the desired goal state $g \in S$, as the factor values can determine what the goal state is for a task. We aim to learn a goal-conditioned policy $\pi(\cdot|s,g) : S \times S \to \Delta(\mathcal{A})$, where $\Delta(\mathcal{A})$ is the probability simplex over the action space. We aim to maximize the probability of reaching the desired goal, under the distribution of possible factor values:

$$J(\pi) = \mathbb{E}_{f \sim p(f)}[P_\pi(s_H = \mu(f))], \tag{1}$$

where $P_\pi(s_H = \mu(f))$ is the probability that the state visited by policy π at the final timestep H is the goal state $g = \mu(f)$.

To learn the policy π, we assume a dataset of M expert demonstrations $\mathcal{D}_M = (\tau_1, \ldots, \tau_M)$. Each demonstration $\tau_i = \{(s_1, a_1), \ldots, (s_{T_i}, a_{T_i})\}$ is a sequence of state-action pairs of length T_i, produced by sampling actions from an expert policy $\pi_E(\cdot|s,g)$ through dynamics $\mathcal{T}(s'|s,a)$. Each demonstration also has an associated combination of factor values f_i, which defines both the initial state distribution the demonstration was collected from, and the goal state $g = \mu(f)$ the expert policy aimed to reach. We can then use this dataset to learn a policy π using goal-conditioned behavior cloning [30].

B. Objective

We aim to train robotic manipulation policies using imitation learning that will robustly handle combinations of factor values from some desired distribution $p(f)$, by collecting demonstrations with associated factor values f according to some strategy. We assume without loss of generality that $p(f) = \text{Uniform}(\mathcal{F}^N)$, such that we desire to handle all factor value combinations in $\mathcal{F}^N$. The most direct way to achieve this is sufficiently covering all of $\mathcal{F}^N$, such as by sampling new values $f \sim \text{Uniform}(\mathcal{F}^N)$ enough times, or even by collecting demonstrations for all $f \in \mathcal{F}^N$. However, as the cardinality of $\mathcal{F}^N$ increases exponentially with the number of factors N, collecting demonstrations for all possible factor value combinations is often impractical. Furthermore, constantly resampling new values $f \sim \text{Uniform}(\mathcal{F}^N)$ can often be expensive or challenging in practice, as each factor change often requires some degree of manual effort.

Rather than attempting to collect data for all factor value combinations in $\mathcal{F}^N$, it would be desirable to instead collect data with a focus on capturing *individual* factor values, and then exploit *compositional generalization* of the learned policy to perform well on unseen combinations. With N factors and k possible values each, collecting data for all combinations would require $\mathcal{O}(k^N)$ factor changes in the environment, while covering all individual factor values would only require $\mathcal{O}(kN)$ changes, a dramatic reduction. However, this is a sensible option only if composition is actually achieved by the learned policy. As prior work has shown mixed results on the compositional abilities of end-to-end neural models, it remains important to investigate *when* compositional generalization happens in imitation learning for robotic manipulation. Therefore, to exploit compositional generalization for more efficient data collection, we must answer the following questions:

1) When do robotic imitation learning policies exhibit compositional generalization?
2) What are effective data collection strategies to exploit composition, such that we can achieve broad generalization while reducing data collection effort?

We answer these questions by conducting extensive experiments in both simulation and on a real robot, investigating the compositional abilities of imitation learning policies for a variety of factors. We compare data collection strategies intended to exploit composition by optimizing for coverage of individual factor values, as opposed to naïvely optimizing for coverage of all factor combinations.

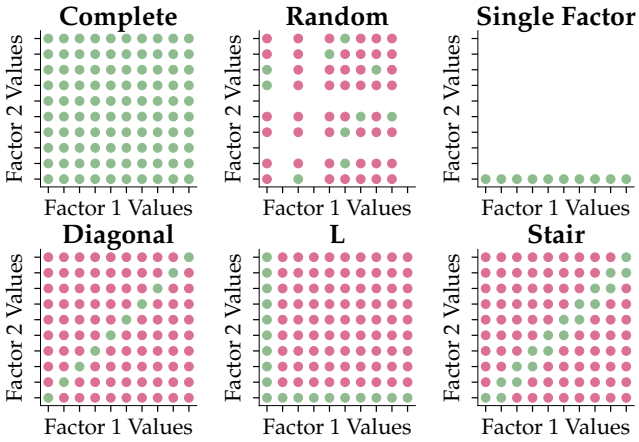


Fig. 2: Visualization of our data collection strategies with $N = 2$ factors. Each axis consists of possible values for a factor. Each green dot indicates that the strategy captures a specific combination of factor values represented by it, and each pink dot represents a combination that compositional generalization may address. We name our strategies based on the patterns in this visualization.

C. Data Collection Strategies

We describe the data collection strategies that we consider. We visualize them for when the number of factors $N = 2$ in Fig. 2, but these can easily be extended for $N > 2$. We also provide pseudocode for **Diagonal**, **L**, and **Stair** in Appendix A-A. The strategies involve periodically setting factor values in the environment during data collection. To quantify effort, we consider the total number of factor changes made in the environment. Note that changing multiple factors at a time involves multiple changes, e.g., setting the environment to have different values for all factors involves N changes.

1) **Complete**: This strategy covers all combinations of factor values in $\mathcal{F}^N$. As this requires $\mathcal{O}(k^N)$ factor changes, this is often deemed infeasible in practice.
2) **Random**: This strategy periodically resamples an entire random combination of factor values f from all possible $f \in \mathcal{F}^N$ without replacement. This strategy will eventually cover all of $\mathcal{F}^N$, but may achieve generalization inefficiently by not actively leveraging composition.
3) **Single Factor**: This strategy only varies one factor during data collection, while keeping the values of all other factors the same as some base factor values f^*.
4) **Diagonal**: This strategy resamples entire combinations of factor values f, but only samples f where each factor value has never been seen. This requires $\mathcal{O}(kN)$ factor changes to cover all values, the fewest possible.
5) **L**: This strategy varies only one factor at a time. Starting from base factor values f^*, the possible combinations of factor values f only deviate from f^* by one factor. This also covers all factor values with $\mathcal{O}(kN)$ factor changes.
6) **Stair**: This strategy starts at f^*, and then cyclically varies one factor at a time, while preserving the values for all other factors. This also covers all factor values with $\mathcal{O}(kN)$ changes. However, compared to **Diagonal**, it covers more combinations of factor values, and compared to **L**, it captures more diverse combinations.
7) **No Variation**: This baseline involves only collecting data for a single base combination of factor values f^*.

Stair, **L**, and **Diagonal** are intended to exploit compositional generalization by prioritizing covering individual factor values with the fewest amount of factor changes. We note that using factor changes as a measure of effort is a rough approximation. For example, this may not be accurate when it is more challenging to vary two factors together than separately. This could be the case if one had to collect data in multiple separate environments to vary a factor like *table texture*, but could collect data in a single environment to vary *object type*. In such situations, it may be easier to use the **L** strategy to collect data varying *object type* in only one environment, and collect data varying *table texture* in separate environments without varying *object type*, instead of varying both factors together like **Stair** and **Diagonal** would require.

D. Hypotheses

We develop the following hypotheses, which we seek to investigate in our experiments:

1) **Stair**, **L**, and **Diagonal** may outperform **Random** by exploiting compositional generalization when possible.
2) **Stair**, **L**, and **Diagonal** may approach the performance of **Complete** when composition is strong.
3) **Stair** may outperform **L** and **Diagonal**, by capturing a greater quantity and diversity of factor combinations.
4) Incorporating prior robot data can promote stronger composition of factor values in the in-domain data.

IV. Simulation Experiments

To evaluate composition and data collection strategies at a large scale, we first conduct extensive experiments in simulation. We use *Factor World*, a robotics simulation platform that supports varying different environmental factors [49].

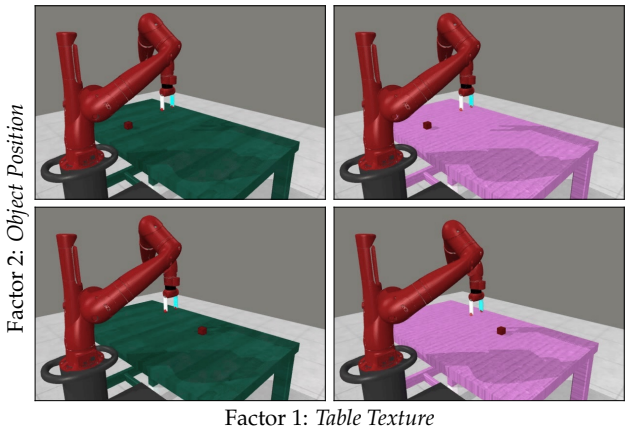


Fig. 3: Visualization of the *Pick Place* task. We show combinations of 2 values each for the factors *table texture* and *object position*.

A. Evaluation Protocol

We consider *Pick Place* and *Door Open*, two commonly studied tasks in robotics. We consider 5 factors, including 4 that generalization was challenging for in the original *Factor World* experiments (*object texture*, *table texture*, *camera position*, and *distractor objects*), as well as *object position*, as an

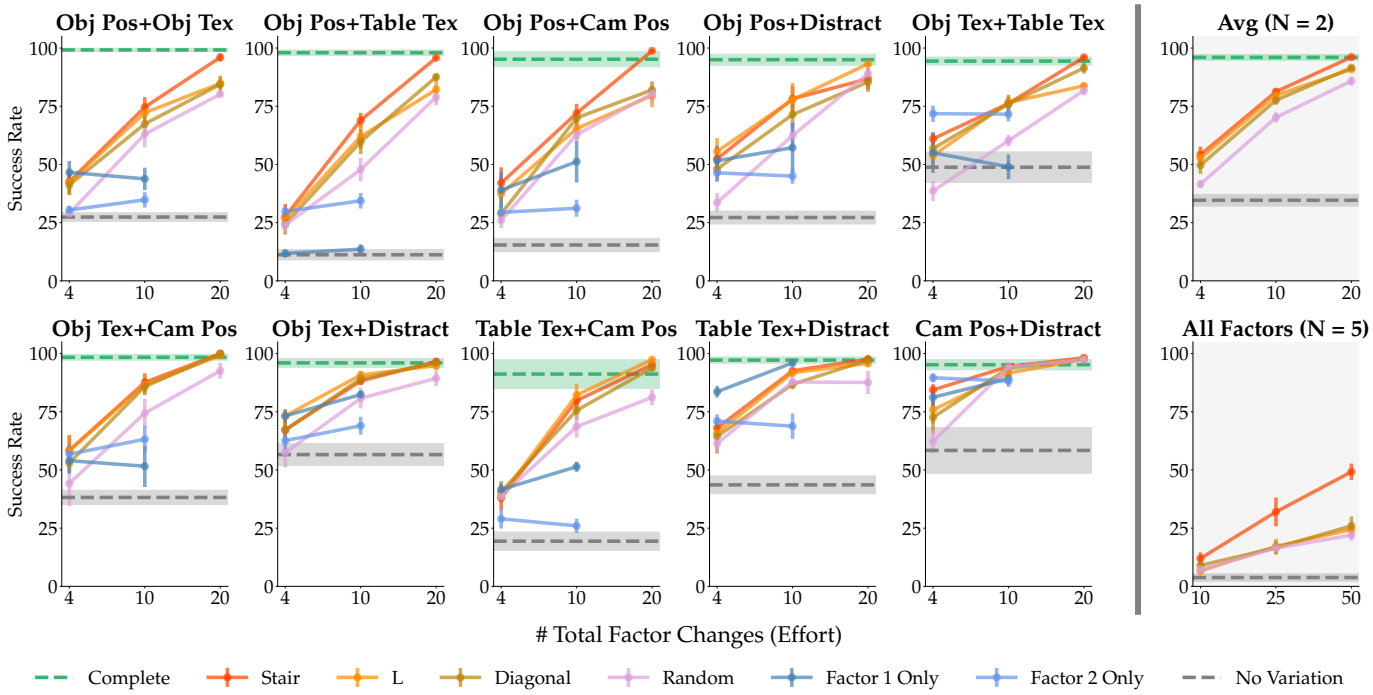


Fig. 4: Simulation results of data collection strategies for *Pick Place*. We report results where $\mathcal{F}^N$ consists of each possible factor pair ($N = 2$), average results across all pairs, and results where $\mathcal{F}^N$ consists of all factors ($N = 5$). All points within the same subplot use the same amount of demonstrations. The strategies that exploit composition (**Stair**, **L**, **Diagonal**) generally outperform **Random**, and often approach **Complete**. **Stair** generally performs the best, especially in the $N = 5$ setting. Error bars represent standard error across 5 seeds.

example of a factor that more strongly affects required physical motion. We show example combinations of *table texture* and *object position* for *Pick Place* in Fig. 3, and examples of individual factor values for both tasks in Appendix B. To study which specific factors can be composed by policies effectively, we consider the settings where $\mathcal{F}^N$ consists of one of the 10 possible factor pairs ($N = 2$). To study composition in a more realistic setting where robustness to more factors is desired, we also consider when $\mathcal{F}^N$ consists of all factors ($N = 5$). For each factor, we sample $k = 10$ values that the factor can take in $\mathcal{F}^N$. For each data strategy, we use a scripted expert policy to collect multiple datasets with different total amounts of factor changes, by setting new factor values according to the strategy at different rates. For example, 20 factor changes would involve setting new values roughly twice as often as with 10 changes. When setting new values, we choose which values to set at random from what is permissible by the current strategy. We compare how the strategies scale with the total number of factor changes, which we use to quantify *effort*.

We only compare against **Complete** in the $N = 2$ setting, as $N = 5$ would require a minimum of 10^5 demonstrations, which is impractical. Each dataset consists of 100 demonstrations for the $N = 2$ experiments, and 1000 demonstrations for $N = 5$. We train a policy on each dataset using behavior cloning, and evaluate on 100 episodes, each with a different f randomly sampled from $\mathcal{F}^N$, to assess overall robustness to factor combinations. For the $N = 2$ experiments, this consists of all possible $f \in \mathcal{F}^N$. We evaluate across 5 random seeds, which include different possible factor values in $\mathcal{F}^N$ for each seed. We provide more details in Appendix B.

B. Results

We illustrate our results for *Pick Place* in Fig. 4. We first focus on the $N = 2$ setting. We see that **No Variation** (dashed gray) performs poorly, verifying that a variety of factor values is needed to generalize. The **Single Factor** strategies (shades of blue) also perform poorly for most factor pairs, verifying that most pairs require variation in both factors to generalize. Note that these strategies can vary only a single factor at most 10 times, leading to shorter lines in the plots. We see that the strategies intended to exploit composition – **Stair**, **L**, **Diagonal** (shades of orange) – generally outperform **Random** (light purple) at all levels of factor changes. They also approach the performance of **Complete** (dashed green line) with only 20 factor changes (enough to capture all individual factor values), compared to the 100 changes needed for **Complete**. These results suggest that these policies exhibit strong pairwise compositional abilities for these factors, and that **Stair**, **L**, and **Diagonal** are able to exploit this for more efficient data collection. We also see that **Stair** does slightly better overall than **L** and **Diagonal**, possibly suggesting some benefit from a greater quantity and diversity of factor value combinations.

In the $N = 5$ setting, performance is lower overall, indicating that generalization is harder when more factors are varied. Furthermore, **Stair** is now the only strategy that substantially improves over **Random**. This suggests that a greater quantity and diversity of factor value combinations is more important for composition when there are more factors involved.

We provide similar results for *Door Open*, results with different data augmentation, and results on accounting for factor value similarity during data collection in Appendix B.

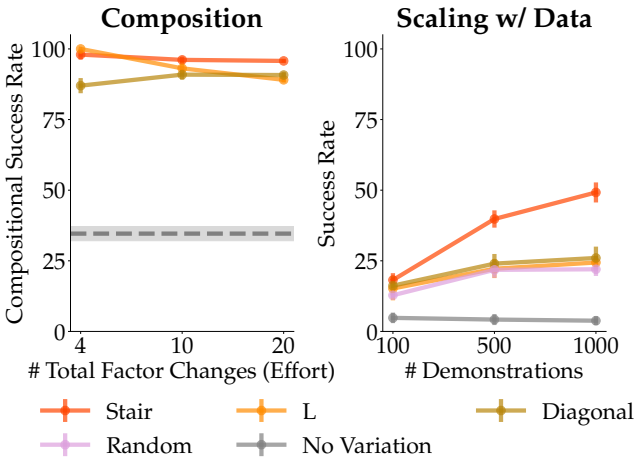


Fig. 5: (left) Compositional success rate of different strategies. (right) Generalization of strategies with increasing dataset sizes.

Assessing Composition. In Fig. 5 (left), we more closely investigate composition for the strategies intended to exploit it (**Stair**, **L**, **Diagonal**), and how this scales with the number of factor values seen. Unlike before, here we report each policy's *compositional success rate*: the success rate **only** on combinations of factor values where each factor value was seen individually, but not in the exact same combination. The previous results considered all possible combinations, to assess overall generalization. We report this metric averaged across all pairs from our $N = 2$ evaluation. We see these strategies exhibit strong composition for all levels of factor changes, suggesting that composition does not require observing a large number of factor values. However, **Stair** does appear to induce composition slightly better than the other strategies.

Scaling with Dataset Size. In Fig. 5 (right), we investigate how generalization scales with dataset size. We focus on the $N = 5$ setting, where generalization is more challenging, and therefore the impact of data scale is more pronounced. We report the success rate of policies trained on datasets from different strategies, which all have 50 factor changes (the amount needed for **Stair**, **L**, and **Diagonal** to capture all factor values), but different amounts of demonstrations. We also compare against **No Variation**, which does not scale well, verifying that simply more data is not enough to generalize. The other strategies do scale positively, but **Stair** does significantly better than the rest, and it is the **only strategy** that is able to significantly benefit when increasing from 500 to 1000 demonstrations. This suggests that having more data, even without greater factor diversity, can help facilitate improved compositional generalization, but having a greater quantity and diversity of factor combinations, which **Stair** provides, may be critical for enabling this scaling.

V. Real Robot Experiments

Our simulation results have been encouraging regarding composition in robotic imitation learning, and the potential to exploit this with strategies for more efficient data collection. However, for such strategies to be useful for real-world data collection, real robot policies must also have compositional abilities. Therefore, we conduct experiments to evaluate if and when compositional generalization happens on a real robot. In addition, we include experiments incorporating prior robotic datasets, to investigate if access to such datasets can benefit composition in robotics, as is the case with natural language.

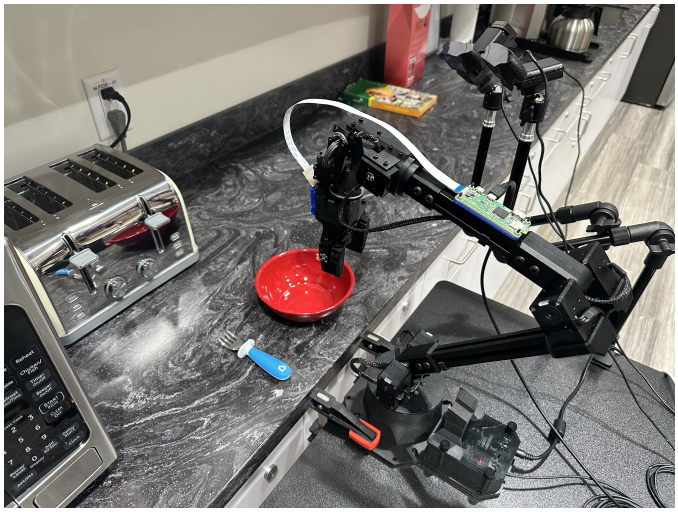


Fig. 6: Our WidowX robot setup. We consider the task of putting a fork inside a container, in a real office kitchen.

A. Evaluation Protocol

Robot Platform. We conduct experiments using a WidowX 250 6DOF robot arm. We adapt the hardware and control setup used in BridgeData V2 [47], except we directly mount the robot and over-the-shoulder RGB camera to a mobile table. While our hardware setup is not identical to the original, we still seek to leverage BridgeData V2 as prior data, which represents a realistic use case of a prior robotic dataset. To promote transfer, we tune our camera setup so that policies trained on only BridgeData V2 can sometimes perform basic pick-place tasks zero-shot, although we were unable to have it perform the exact tasks in our evaluation. We show our robot setup in Fig. 6, and provide more details in Appendix C-A.

Task and Factors. We consider the task of placing a fork inside a container, on the countertop of an office kitchen. We primarily consider the following factors: *object (fork) type*, *object (fork) position*, *container type*, *table height*, and *table texture*. Each factor has $k = 4$ possible values. We visualize our base values f^*, and also every deviation from f^* by one factor, in Fig. 7. We additionally consider the factors *object/container position*, *object (fork) orientation*, *distractor objects*, and *camera position* in secondary experiments.

Compared to our simulation experiments, our real experiments have a greater emphasis on *physical* factors that affect required physical motions, as opposed to *visual* factors that a policy should be mostly invariant to. For instance, we only study *object type*, *container type*, and *table height* in real, which are *physical* factors that can affect required grasp and place motions. This is because it is more difficult to conduct large-scale simulation experiments for such factors, due to the challenge of automatically collecting demonstrations that account for them. We believe that addressing *physical* factors

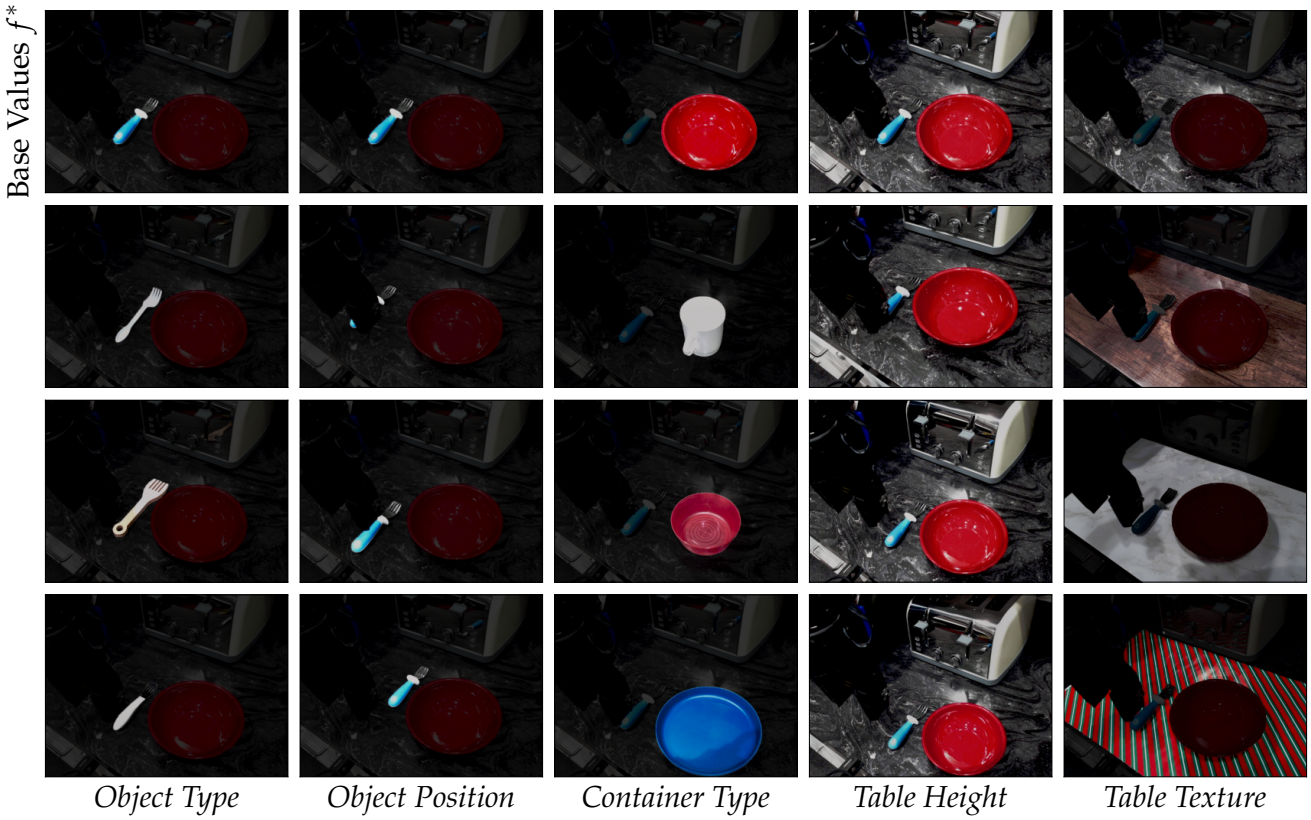


Fig. 7: Visualization of our primary real robot factors in *BaseKitch*. The top row shows our base factor values f^*. The other rows show all deviations from f^* by one factor value. We highlight the portion of each image affected by the specified factor value.

is especially important because they are likely more resistant to visual data augmentation, and thus may require more emphasis during data collection. Also, *physical* factors are less likely to benefit from using internet data for semantic reasoning [19, 6].

Experimental Setup. We adopt a different experimental setup for our real experiments, due to the much greater cost of collecting data, training policies, and evaluating. We collect datasets for only the $N = 5$ setting with all of our primary factors. Because the objective of these experiments is to determine which factors can be composed by a real robot policy, we focus on data collection strategies intended to exploit composition. In particular, we evaluate **Stair**, because it performed the best in simulation, and **L**, because by collecting data for each factor separately, we can use this to study pairwise composition using a model trained on a single dataset. **L** may also be more practical in real-world situations where it is easier to collect data for factors separately. For these strategies, we collect 10 human demonstrations for each of the 16 combinations of factor values required to cover all values per factor, resulting in 160 total demonstrations for each strategy. We also compare against **No Variation** with the same number of demonstrations. We collect data exclusively in one kitchen, which we refer to as *BaseKitch*. We train policies either from scratch, or with the addition of BridgeData V2, to assess the impact of prior data on composition. We provide more details on our experimental setup in Appendix C-B.

Training. We primarily train policies based on the diffusion goal-conditioned behavior cloning method proposed in BridgeData V2 [47], adapting the implementation provided in their code. For policies that use BridgeData V2 as prior data, unless otherwise stated, we first pre-train a policy on only the prior data, and then co-fine-tune on a mixture of in-domain and prior data. We verify that all policies succeed with base factor values f^* in *BaseKitch*. We provide more details in Appendix C-C.

B. Pairwise Composition

In this evaluation, we study pairwise composition of factors in *BaseKitch*. We compare **L** and **No Variation** to evaluate composition. For each pair of factors, we evaluate on all compositional combinations of factor values for that pair with respect to the **L** dataset, and set the values for other factors to their base values in f^*. This results in 9 evaluation scenarios for each of the 10 pairs.[2] For **L**, we compare both training from scratch, and using BridgeData V2 as prior data. For **No Variation**, we only evaluate training with BridgeData V2.

We report our results in Table I. We find that by training on **L** data and using BridgeData V2, our policy is able to generalize to **59/90** of the compositional factor value combinations. This suggests that composition is possible in real-world robotic imitation learning, similar to our simulation results. However, unlike in simulation, leveraging prior data is critical for strengthening composition. Without prior data, the policy's overall compositional success rate is roughly halved

[2]We have 3 values per factor that are not in the base values f^*, resulting in $3^2 = 9$ combinations for each of the $\binom{5}{2} = 10$ pairs.

Data Strategy	**L**								**No Variation**			
Train Method	Bridge				From Scratch				Bridge			
Factor 1 \ Factor 2	Object Type	Container Type	Table Height	Table Tex	Object Type	Container Type	Table Height	Table Tex	Object Type	Container Type	Table Height	Table Tex
Object Pos	**8/9**	**5/9**	**2/9**	**5/9**	0/9	0/9	0/9	1/9	3/9	2/9	1/9	1/9
Object Type		**8/9**	**8/9**	**8/9**		5/9	5/9	4/9		4/9	3/9	2/9
Container Type			**5/9**	**6/9**			4/9	**6/9**			2/9	3/9
Table Height				**4/9**				3/9				1/9
Overall		**59/90**				28/90				22/90		

TABLE I: Real robot pairwise composition results for our "*put fork in container*" task. When leveraging BridgeData V2 as prior data, a policy is able to compose the factor values present in the **L** data to succeed on **59/90** compositional combinations of factor values. Without prior data, the model is unable to compose nearly as effectively, with compositional success rate dropping by roughly half. Prior data alone is also not enough to generalize to these situations, as a policy trained with prior data on **No Variation** also performs poorly.

to **28/90**, with success rates dropping for **9/10** factor pairs. This resembles results in prior work that suggest end-to-end neural models can struggle with composition, but large pre-trained models can exhibit strong composition [54, 13]. We hypothesize that prior data may be needed in real, but not simulation, due to additional minor factor variations inherent to real experiments, including those for factors that we do not account for. Our greater emphasis on *physical* factors in real may also account for some of this discrepancy.

However, prior data alone is not enough for generalization, as training on **No Variation** with prior data performs much worse than using **L** data, achieving a success rate of only **22/90**. This suggests that with current prior robotic datasets, varied in-domain data is often still critical for generalization, motivating the need for efficient in-domain data collection.

We note that composition is generally strongest for pairs where at least one factor is *visual*. We hypothesize this is because *physical* factors can interact in more complex ways, making it more challenging for policies to compose unseen combinations of them. For example, *object position* and *table height* have the weakest composition together, possibly because both significantly affect the required grasp motion, and thus composition requires executing completely unseen grasps. Similarly, composition for *container type* is the weakest for the value **white cup** (second row in Fig. 7), which is narrower and taller than the other containers, and therefore requires a different place motion. In particular, this composes poorly with *object position* and *table height*, both *physical* factors.

In contrast, composition is generally the strongest for *object type*. Although *object type* may affect grasp and place motions, especially when interacting with other *physical* factors, we hypothesize it composes well because the different forks we consider are similar enough to not require drastically different motions. Overall, these results suggest that even with current prior robotics datasets, composition between *physical* factors can still be challenging, and thus more exhaustive coverage for these factors during data collection may be necessary.

We provide further pairwise composition results for additional factors *object/container position* and *object orientation* in Appendix C-D. In Appendix C-E, we provide additional analysis on how pairwise composition is affected by how similar new factor values are to the base factor values f^*.

Data Strategy	**L**				**No Variation**		
Train Method	Bridge	R3M	VC-1	From Scratch	Bridge	R3M	VC-1
Object Type + Table Tex	**8/9**	1/9	0/9	4/9	2/9	0/9	0/9
Object Pos + Table Tex	**5/9**	0/9	0/9	1/9	1/9	0/9	0/9
Object Pos + Table Height	**2/9**	1/9	0/9	0/9	1/9	0/9	0/9
Overall	**15/27**	2/27	0/27	5/27	4/27	0/27	0/27

TABLE II: Additional real robot pairwise composition results for our "*put fork in container*" task with R3M and VC-1.

Pre-trained Representations. Pre-trained visual representations have become popular for promoting generalization in robot learning. To assess their impact on composition, we additionally evaluate using R3M [34] and VC-1 [32] as frozen visual encoders. We evaluate on a subset of factor pairs from our full pairwise evaluation. Specifically, we consider *object type + table texture* (the pair of factors that affect physical motion the least), *object position + table texture* (a pair where one factor is *physical* and the other is *visual*), and *object position + table height* (a pair where both factors are *physical*).

In our results in Table II, we find that both R3M and VC-1 perform poorly, with VC-1 failing to complete the task at all. Qualitatively, we observe that the trajectories are more jittery than when learning end-to-end, as also noted in prior work that tried R3M with diffusion-based policies [9]. This suggests that modern frozen visual representations are not effective for compositional generalization on real robots, and that end-to-end learning with prior robot data is superior for this.

C. Out-Of-Domain Transfer

To further demonstrate the utility of exploiting composition during data collection, we assess the transfer abilities of policies trained on our datasets to entirely new environments that capture some of the factor variety accounted for during data collection. We evaluate in two new kitchens, which we refer to as *CompKitch* and *TileKitch*.[3] These kitchens inherently have some completely out-of-distribution factor values from those studied in *BaseKitch* (e.g., *table texture*, shown in Fig. 8), including for some factors we do not account for during data collection (e.g., *distractor objects*, *lighting*).

[3]We name these kitchens based on the material of their countertops. See Fig. 15 in Appendix C-B for additional views that make this more apparent.

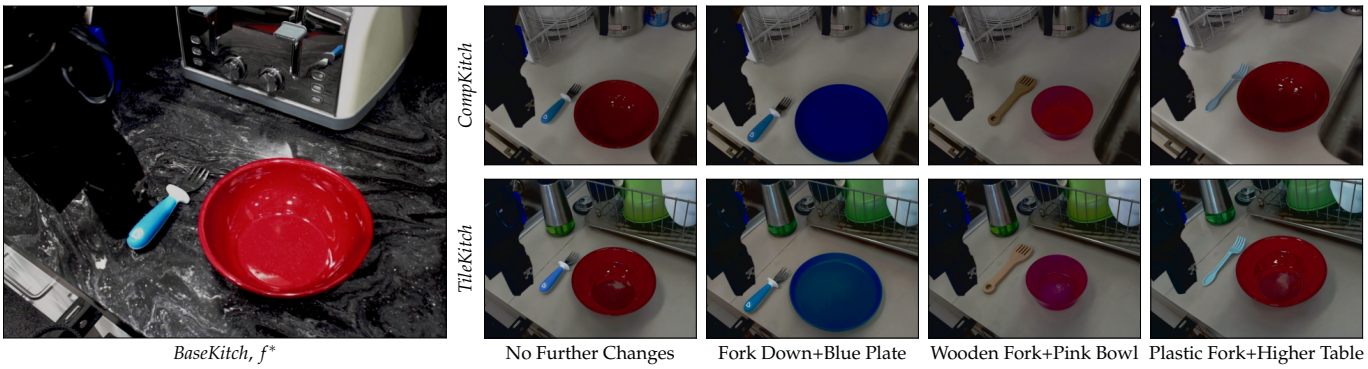


Fig. 8: Visualization of *BaseKitch* with base factor values f^* (left), compared to our transfer conditions in *CompKitch* (top right) and *TileKitch* (bottom right), which inherently have different factor values (e.g., *table texture*) and other characteristics (e.g., *distractor objects*, *lighting*). We consider setups both with no further changes from f^* beyond these inherent differences, as well as with additional changes.

Data Strategy		**Stair**			**L**			**No Variation**	
Train Method		Bridge (Co-FT)	Bridge (FT)	From Scratch	Bridge (Co-FT)	Bridge (FT)	From Scratch	Bridge (Co-FT)	Bridge (FT)
No Further Changes	*CompKitch*	4/5	4/5	0/5	**5/5**	0/5	0/5	0/5	0/5
	TileKitch	**5/5**	2/5	0/5	**5/5**	**5/5**	3/5	0/5	0/5
Fork Down + Blue Plate	*CompKitch*	**4/5**	**4/5**	0/5	3/5	0/5	0/5	1/5	0/5
	TileKitch	4/5	3/5	0/5	3/5	**5/5**	**5/5**	0/5	0/5
Wooden Fork + Pink Bowl	*CompKitch*	3/5	**5/5**	0/5	2/5	0/5	0/5	0/5	0/5
	TileKitch	4/5	3/5	0/5	3/5	**5/5**	2/5	0/5	0/5
Plastic Fork + Higher Table	*CompKitch*	**4/5**	0/5	0/5	1/5	0/5	0/5	0/5	0/5
	TileKitch	**3/5**	0/5	0/5	2/5	0/5	1/5	0/5	0/5
Overall	*CompKitch*	**15/20**	13/20	0/20	11/20	0/20	0/20	1/20	0/20
	TileKitch	**16/20**	8/20	0/20	13/20	15/20	11/20	0/20	0/20
	Combined	**31/40**	21/40	0/40	24/40	15/40	11/40	1/40	0/40

TABLE III: Out-of-domain transfer results to new kitchens *CompKitch* and *TileKitch*. We find that varied in-domain data from *BaseKitch*, and BridgeData V2 as prior data, are both critical for effective transfer to these new kitchens. **Stair** outperforms **L**, although both achieve significant levels of transfer. Co-fine-tuning generally performs better than only fine-tuning.

We first evaluate in these new kitchens with no further changes from the base factor values f^* in *BaseKitch*, aside from their inherent shifts. In addition, to assess beyond pair-wise composition for multiple factors, we also evaluate in these kitchens with additional combinations of factor shifts. We visualize these different conditions in Fig. 8. For each data collection strategy, we again compare using BridgeData V2 as prior data, and training from scratch. In addition to using prior data for both pre-training and co-fine-tuning (as done with the previously evaluated models), we also compare to fine-tuning on only in-domain data from a pre-trained BridgeData V2 policy, to assess the importance of co-fine-tuning for compositional generalization and transfer.

We report our results in Table III. We find that BridgeData V2 is critical for transfer, as one of the two policies trained from scratch is unable to transfer at all, while the other only achieves a success rate of **11/40**. When co-fine-tuning with BridgeData V2, having varied in-domain data is still needed for robust transfer, as **No Variation** only achieves a success rate of **1/40**. These results indicate that this transfer setting represents a significant and challenging domain shift.

Despite this, **Stair** and **L** both achieve significant levels of transfer, with success rates of **31/40** and **24/40**, respectively. **Stair** generally outperforms **L**, as was the case in simulation. This suggests that policies that use prior data are able to effectively transfer to new settings that require composition. When fine-tuning from a pre-trained model, **Stair** and **L** produce policies that achieve some transfer, but less consistently than when co-fine-tuning, with reduced success rates of **21/40** and **15/40**, respectively. In particular, the **L** policy fails to transfer to *CompKitch*, and both policies fail to transfer to **Plastic Fork + Higher Table** in both kitchens. This suggests that co-fine-tuning is generally superior for facilitating composition than pre-training alone. Overall, we believe these results further suggest that policies can exhibit composition to generalize to unseen settings, our data collection strategies are sufficient to achieve some of this composition, and that prior data is important for this composition to happen effectively.

Data Strategy	**Stair**		**L**		**No Variation**	
Train Method	Bridge	From Scratch	Bridge	From Scratch	Bridge	From Scratch
Tape Measure	5/5	1/5	5/5	5/5	5/5	5/5
Pink Bowl	5/5	3/5	5/5	4/5	5/5	5/5
Spoon	5/5	5/5	5/5	0/5	5/5	4/5
Overall	**15/15**	9/15	**15/15**	9/15	**15/15**	**15/15**

TABLE IV: Evaluation on held-out factor *distractor objects*. Data collection strategies that do not account for this factor can perform worse than **No Variation** when training from scratch, but this issue is alleviated completely by using prior data.

(a) Tape Measure (b) Pink Bowl (c) Spoon

Fig. 9: Visualization of the different values for *distractor objects* we consider in our held-out factor evaluation.

D. Additional Experiments

Unaccounted Factors. Our transfer experiments involve generalization to some factors that were unaccounted for during data collection, such as *distractor objects* and *lighting*. For more focused assessment of the impact of data collection strategies on such unaccounted factors, we additionally evaluate on *distractor objects* as a held-out factor in *BaseKitch*. In early experiments, we noticed a large degree of robustness to this factor from **No Variation** policies, so we did not include it among our primary factors. We evaluate our policies on 3 different values for this factor (consisting of distinct objects and positions for each value), which we visualize in Fig. 9. Our results in Table IV confirm the aforementioned robustness of the **No Variation** policies. However, when training from scratch, policies using **Stair** and **L** data perform worse, indicating that these data collection strategies may worsen robustness for unaccounted factors, possibly due to introducing spurious correlations. However, incorporating prior data completely mitigates this issue and restores robustness to this factor. This suggests another reason for why using prior robot data can be important for generalization: to address factors unaccounted for during in-domain data collection.

Camera Position Composition. We do not consider *camera position* among the primary factors in our real evaluation, due to the difficulty of varying this factor in a controlled manner. However, we have one additional third-person camera on our robot platform that we also collect data for, which allows us to separately study compositional generalization with this secondary camera. We visualize the difference in perspective with this secondary camera from our main camera in Fig. 10.

We focus on the composition of *camera position* with *table texture*. We evaluate in *BaseKitch*, but using the secondary camera for observations. We set *table texture* to the value **white marble** (third row in Fig. 7), as this was the value

(a) Main Camera (b) Secondary Camera

Fig. 10: Visualization of the secondary camera view we use for studying composition of *camera position* (right), compared with our main camera view used in all other experiments (left).

that was the most challenging for policies lacking variation for *table texture* to generalize to. We train a policy on the combination of two sub-datasets: our **L** dataset from the primary camera view, and the same **L** dataset but from the secondary camera view, with the exception of data for *table texture*. We compare against training on only one of these sub-datasets, which either lack data with the value for *camera position* or *table texture* that is seen during evaluation. We again evaluate how BridgeData V2 affects this composition.

In our results in Table V, we find that yet again, policies that do not use prior data struggle to generalize. When using prior data, training on each sub-dataset individually results in a policy that sometimes generalizes, but not consistently. However, training on both sub-datasets together is able to achieve a perfect success rate, suggesting effective composition of *camera position* with *table texture*.

Train Method	Bridge	From Scratch
No Camera Position	1/10	0/10
No Table Texture	6/10	0/10
Both	**10/10**	0/10

TABLE V: Our policy composes datasets missing either the correct *camera position* or *table texture* seen during evaluation, to outperform training on either dataset alone. Prior data is critical for composition.

VI. Discussion

A. Summary

We investigate the compositional abilities of visual imitation learning policies for robotic manipulation, and whether this can be exploited by data collection strategies to more efficiently achieve broad generalization. In summary, our simulated and real-world experiments suggest the following:

- Robot policies do exhibit significant composition, although the degree of composition varies across different factor pairs, with composition between *physical* factors generally being the most challenging.
- Leveraging prior robot data is critical for strengthening compositional abilities with a real robot, and co-fine-tuning is generally better for this than fine-tuning.
- Our proposed data collection strategies are able to exploit composition to reduce data collection effort, while producing policies that can generalize to unseen settings.
- Two of our proposed strategies, **Stair** and **L**, are able to transfer policies to entirely new environments that require

a high degree of compositional generalization, showing they can produce data that is useful beyond the original domain it was collected in.
- Our proposed **Stair** strategy is generally the most effective, as it achieves the best pairwise composition in simulation and real, the best transfer results in real, and is the only strategy in simulation that effectively scales with dataset size for the same amount of factor variation.

Overall, we believe these results provide insights on how roboticists can more efficiently collect in-domain data for achieving generalization in their desired settings.

B. Limitations and Future Work

More Robot Platforms and Tasks. While we conduct thorough real-world experiments for evaluating compositional generalization, it would be interesting to scale our experiments and analysis to more complex robot platforms and tasks. We did some preliminary investigation into treating different tasks as a factor for composition, by training a policy on the combination of the **L** dataset for our main task "*put fork in container*", and **No Variation** data for a new task "*remove fork from container*". We then evaluated this on the new task with factor values found in the main task data, but it did not succeed. This could be because our new task data did not have any factor variation, which might be necessary for composition. Future work can more thoroughly consider this setting.

Large-Scale Data Collection. Although we believe our results can be broadly informative for data collection in robotics, we focus primarily on in-domain data. Future work could scale this analysis to large-scale data collection efforts, to better understand how to most efficiently collect prior data for downstream transfer. While we do show that incorporating such prior data is critical for strengthening composition, we only consider one prior dataset. Future work can investigate how other prior robotic datasets impact composition, and what aspects of prior datasets are the most important for this.

Improving Composition. While our results show that robot policies can possess significant composition and transfer capabilities, they still struggle at times with composition, particularly for factors that interact physically. Future work can better address generalization in these settings, such as by prioritizing data collection for certain factor combinations over others. Also, we only consider straightforward behavior cloning methods for policy learning. It would be interesting to study if other policy learning methods have better compositional abilities, such as different learning algorithms or architectures.

For example, one possible direction for future work could involve conditioning diffusion-based policies on individual environmental factors. Then, the score predictions for separate factors could be combined to produce a policy that more effectively achieves composition, similar to prior work in the context of text-to-image generation [29]. Prior work has studied combining score predictions in this manner via classifier-free guidance [18] in the context of goal-conditioned robotic policy learning [42], but this direction has not yet been studied for composing environmental factors.

ACKNOWLEDGMENTS

This work was supported by DARPA W911NF2210214, ONR N00014-22-1-2293, NSF 1941722, Toyota Research Institute, and Other Transaction award HR00112490375 from the U.S. Defense Advanced Research Projects Agency (DARPA) Friction for Accountability in Conversational Transactions (FACT) program. We thank Suvir Mirchandani, Joey Hejna, and other members of the ILIAD lab for helpful discussions and feedback. We also thank Abraham Lee for advice on setting up our robot platform to work with BridgeData V2.

REFERENCES

[1] Aishwarya Agrawal, Aniruddha Kembhavi, Dhruv Batra, and Devi Parikh. C-vqa: A compositional split of the visual question answering (vqa) v1. 0 dataset. *arXiv preprint arXiv:1704.08243*, 2017.

[2] Michael Ahn, Debidatta Dwibedi, Chelsea Finn, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Karol Hausman, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Sean Kirmani, Isabel Leal, Edward Lee, Sergey Levine, Yao Lu, Sharath Maddineni, Kanishka Rao, Dorsa Sadigh, Pannag Sanketi, Pierre Sermanet, Quan Vuong, Stefan Welker, Fei Xia, Ted Xiao, Peng Xu, Steve Xu, and Zhuo Xu. Autort: Embodied foundation models for large scale orchestration of robotic agents, 2024.

[3] Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. Data quality in imitation learning. conference on neural information processing systems. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

[4] Ben Bogin, Shivanshu Gupta, Matt Gardner, and Jonathan Berant. Covr: A test-bed for visually grounded compositional generalization with real images. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9824–9846, 2021.

[5] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[6] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alex Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna

Zitkovich. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *arXiv preprint arXiv:2307.15818*, 2023.

[7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[8] Kaylee Burns, Zach Witzel, Jubayer Ibn Hamid, Tianhe Yu, Chelsea Finn, and Karol Hausman. What makes pre-trained visual representations successful for robust manipulation? *arXiv preprint arXiv:2312.12444*, 2023.

[9] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[10] Sudeep Dasari, Frederik Ebert, Stephen Tian, Suraj Nair, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, Sergey Levine, and Chelsea Finn. Robonet: Large-scale multi-robot learning. In *Conference on Robot Learning*, pages 885–897. PMLR, 2020.

[11] Sudeep Dasari, Mohan Kumar Srirama, Unnat Jain, and Abhinav Gupta. An unbiased look at datasets for visuomotor pre-training. In *Conference on Robot Learning*, pages 1183–1198. PMLR, 2023.

[12] Coline Devin, Abhishek Gupta, Trevor Darrell, Pieter Abbeel, and Sergey Levine. Learning modular neural network policies for multi-task and multi-robot transfer. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2169–2176. IEEE, 2017.

[13] Andrew Drozdov, Nathanael Schärli, Ekin Akyürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. Compositional semantic parsing with large language models. In *The Eleventh International Conference on Learning Representations*, 2023.

[14] Yilun Du, Shuang Li, and Igor Mordatch. Compositional visual generation with energy based models. *Advances in Neural Information Processing Systems*, 33:6637–6647, 2020.

[15] Frederik Ebert, Chelsea Finn, Alex X Lee, and Sergey Levine. Self-supervised visual planning with temporal skip connections. *CoRL*, 12:16, 2017.

[16] Frederik Ebert, Yanlai Yang, Karl Schmeckpeper, Bernadette Bucher, Georgios Georgakis, Kostas Daniilidis, Chelsea Finn, and Sergey Levine. Bridge Data: Boosting Generalization of Robotic Skills with Cross-Domain Datasets. In *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, June 2022. doi: 10.15607/RSS.2022.XVIII.063.

[17] Hao-Shu Fang, Hongjie Fang, Zhenyu Tang, Jirong Liu, Junbo Wang, Haoyi Zhu, and Cewu Lu. Rh20t: A robotic dataset for learning diverse skills in one-shot. In *RSS 2023 Workshop on Learning for Task and Motion Planning*, 2023.

[18] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

[19] Brian Ichter, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, Dmitry Kalashnikov, Sergey Levine, Yao Lu, Carolina Parada, Kanishka Rao, Pierre Sermanet, Alexander T Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Mengyuan Yan, Noah Brown, Michael Ahn, Omar Cortes, Nicolas Sievers, Clayton Tan, Sichun Xu, Diego Reyes, Jarek Rettinghouse, Jornell Quiambao, Peter Pastor, Linda Luu, Kuang-Huei Lee, Yuheng Kuang, Sally Jesmonth, Kyle Jeffrey, Rosario Jauregui Ruano, Jasmine Hsu, Keerthana Gopalakrishnan, Byron David, Andy Zeng, and Chuyuan Kelly Fu. Do as i can, not as i say: Grounding language in robotic affordances. In *6th Annual Conference on Robot Learning*, 2022.

[20] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910, 2017.

[21] Siddharth Karamcheti, Suraj Nair, Annie S. Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-driven representation learning for robotics. In *Robotics: Science and Systems (RSS)*, 2023.

[22] Michael Kelly, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J Kochenderfer. Hg-dagger: Interactive imitation learning with human experts. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8077–8083. IEEE, 2019.

[23] Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, et al. Measuring compositional generalization: A comprehensive method on realistic data. In *International Conference on Learning Representations*, 2019.

[24] Najoung Kim and Tal Linzen. Cogs: A compositional generalization challenge based on semantic interpretation. In *2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*, pages 9087–9105. Association for Computational Linguistics (ACL), 2020.

[25] Yen-Ling Kuo, Boris Katz, and Andrei Barbu. Deep compositional robotic planners that follow natural language commands. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4906–4912. IEEE, 2020.

[26] Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International conference on machine learning*, pages 2873–2882. PMLR, 2018.

[27] Michael Laskey, Jonathan Lee, Roy Fox, Anca Dragan,

and Ken Goldberg. Dart: Noise injection for robust imitation learning. In *Conference on robot learning*, pages 143–156. PMLR, 2017.

[28] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International journal of robotics research*, 37(4-5):421–436, 2018.

[29] Nan Liu, Shuang Li, Yilun Du, Antonio Torralba, and Joshua B Tenenbaum. Compositional visual generation with composable diffusion models. In *European Conference on Computer Vision*, pages 423–439. Springer, 2022.

[30] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on robot learning*, pages 1113–1132. PMLR, 2020.

[31] Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. In *The Eleventh International Conference on Learning Representations*, 2022.

[32] Arjun Majumdar, Karmesh Yadav, Sergio Arnaud, Yecheng Jason Ma, Claire Chen, Sneha Silwal, Aryan Jain, Vincent-Pierre Berges, Tingfan Wu, Jay Vakil, Pieter Abbeel, Jitendra Malik, Dhruv Batra, Yixin Lin, Oleksandr Maksymets, Aravind Rajeswaran, and Franziska Meier. Where are we in the search for an artificial visual cortex for embodied intelligence? In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[33] Ajay Mandlekar, Yuke Zhu, Animesh Garg, Jonathan Booher, Max Spero, Albert Tung, Julian Gao, John Emmons, Anchit Gupta, Emre Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pages 879–893. PMLR, 2018.

[34] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In *Conference on Robot Learning*, pages 892–909. PMLR, 2022.

[35] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. https://octo-models.github.io, 2023.

[36] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[37] Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. The unsurprising effectiveness of pre-trained vision models for control. In *International Conference on Machine Learning*, pages 17359–17371. PMLR, 2022.

[38] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 3406–3413. IEEE, 2016.

[39] Wilbert Pumacay, Ishika Singh, Jiafei Duan, Ranjay Krishna, Jesse Thomason, and Dieter Fox. The colosseum: A benchmark for evaluating generalization for robotic manipulation. 2024.

[40] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[41] Ilija Radosavovic, Tete Xiao, Stephen James, Pieter Abbeel, Jitendra Malik, and Trevor Darrell. Real-world robot learning with masked visual pre-training. In *Conference on Robot Learning*, pages 416–426. PMLR, 2022.

[42] Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal conditioned imitation learning using score-based diffusion policies. In *Robotics: Science and Systems*, 2023.

[43] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[44] Lukas Schott, Julius Von Kügelgen, Frederik Träuble, Peter Vincent Gehler, Chris Russell, Matthias Bethge, Bernhard Schölkopf, Francesco Locatello, and Wieland Brendel. Visual representation learning does not generalize strongly within the same domain. In *International Conference on Learning Representations*, 2021.

[45] Pratyusha Sharma, Lekha Mohan, Lerrel Pinto, and Abhinav Gupta. Multiple interactions made easy (mime): Large scale demonstrations data for imitation. In *Conference on robot learning*, pages 906–915. PMLR, 2018.

[46] Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Sean Kirmani, Brianna Zitkovich, Fei Xia, Chelsea Finn, and Karol Hausman. Open-world object manipulation using pre-trained vision-language models. In *7th Annual Conference on Robot Learning*, 2023.

[47] Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023.

[48] Renhao Wang, Jiayuan Mao, Joy Hsu, Hang Zhao, Jiajun Wu, and Yang Gao. Programmatically grounded, compositionally generalizable robotic manipulation. In *The*

Eleventh International Conference on Learning Representations, 2023.

[49] Annie Xie, Lisa Lee, Ted Xiao, and Chelsea Finn. Decomposing the generalization gap in imitation learning for visual robotic manipulation. *arXiv preprint arXiv:2307.03659*, 2023.

[50] Eliot Xing, Abhinav Gupta, Sam Powers, and Victoria Dean. Kitchenshift: Evaluating zero-shot generalization of imitation-based policy learning under domain shifts. In *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*, 2021.

[51] Danfei Xu, Suraj Nair, Yuke Zhu, Julian Gao, Animesh Garg, Li Fei-Fei, and Silvio Savarese. Neural task programming: Learning to generalize across hierarchical tasks. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3795–3802. IEEE, 2018.

[52] Zhenlin Xu, Marc Niethammer, and Colin A Raffel. Compositional generalization in unsupervised compositional representation learning: A study on disentanglement and emergent language. *Advances in Neural Information Processing Systems*, 35:25074–25087, 2022.

[53] Allan Zhou, Vikash Kumar, Chelsea Finn, and Aravind Rajeswaran. Policy architectures for compositional generalization in control. *arXiv preprint arXiv:2203.05960*, 2022.

[54] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, et al. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*, 2023.

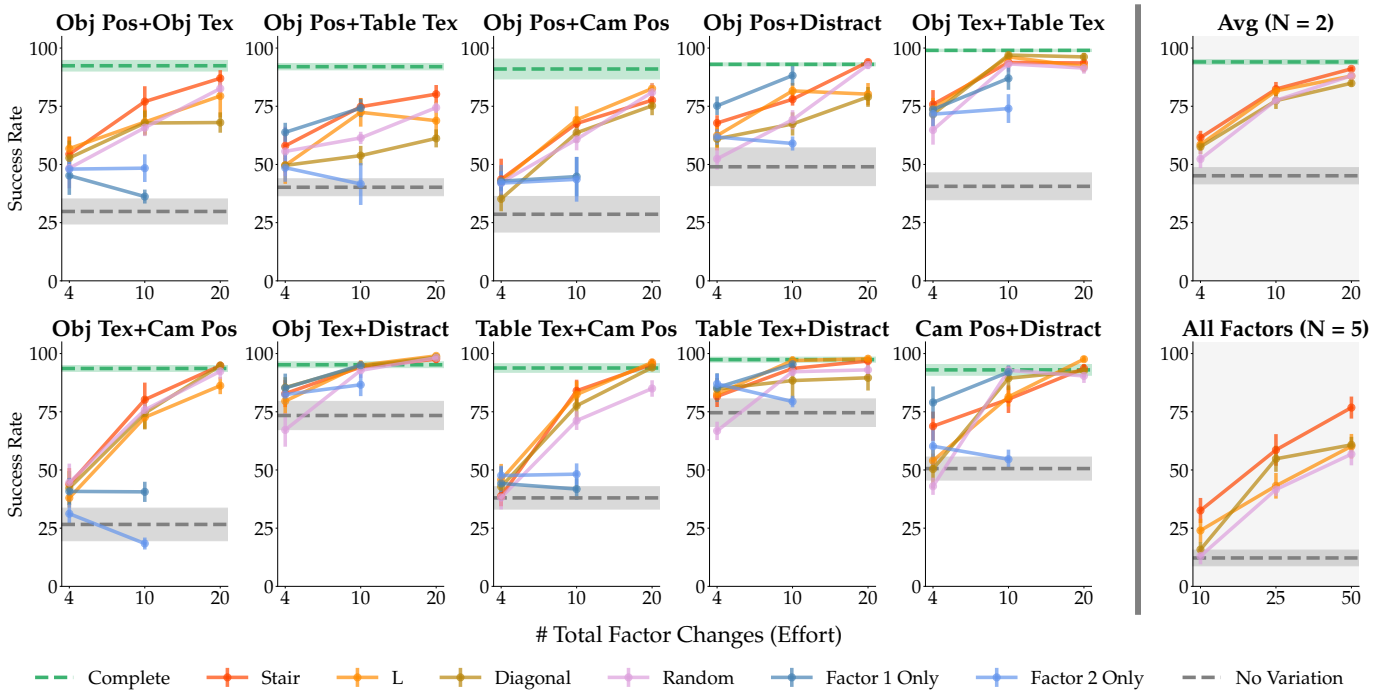


Fig. 11: Simulation results of different data collection strategies for *Door Open*. We report results where $\mathcal{F}^N$ consists of each possible factor pair ($N = 2$), average results across all pairs, and results where $\mathcal{F}^N$ consists of all factors ($N = 5$). Results are similar as with *Pick Place*, with **Stair** generally performing the best, especially in the ($N = 5$) setting. Error bars represent standard error across 5 seeds.

APPENDIX A
DATA COLLECTION STRATEGIES

A. Pseudocode

We provide pseudocode for the data collection strategies proposed in Section III-C that are intended to exploit compositional generalization (**Diagonal**, **L**, **Stair**).

Algorithm 1 Diagonal

Input: scene **S**, factor values **F** (size N factors $\times$ k values)
for $j \leftarrow 1$ to k **do**
  $f \leftarrow 0^N$
  for $i \leftarrow 1$ to N **do**
    $f_i \leftarrow \mathbf{F}_{ij}$
  end for
  SETFACTORS(**S**, f)
  COLLECTDATA(**S**)
end for

Algorithm 2 L

Input: scene **S**, factor values **F** (size N factors $\times$ k values), base factor values f^* (size N factors)
for $i \leftarrow 1$ to N **do**
  $f \leftarrow f^*$
  for $j \leftarrow 1$ to k **do**
    $f_i \leftarrow \mathbf{F}_{ij}$
    SETFACTORS(**S**, f)
    COLLECTDATA(**S**)
  end for
end for

Algorithm 3 Stair

Input: scene **S**, factor values **F** (size N factors $\times$ k values), base factor values f^* (size N factors)
$f \leftarrow f^*$
for $j \leftarrow 1$ to k **do**
  for $i \leftarrow 1$ to N **do**
    $f_i \leftarrow \mathbf{F}_{ij}$
    SETFACTORS(**S**, f)
    COLLECTDATA(**S**)
  end for
end for

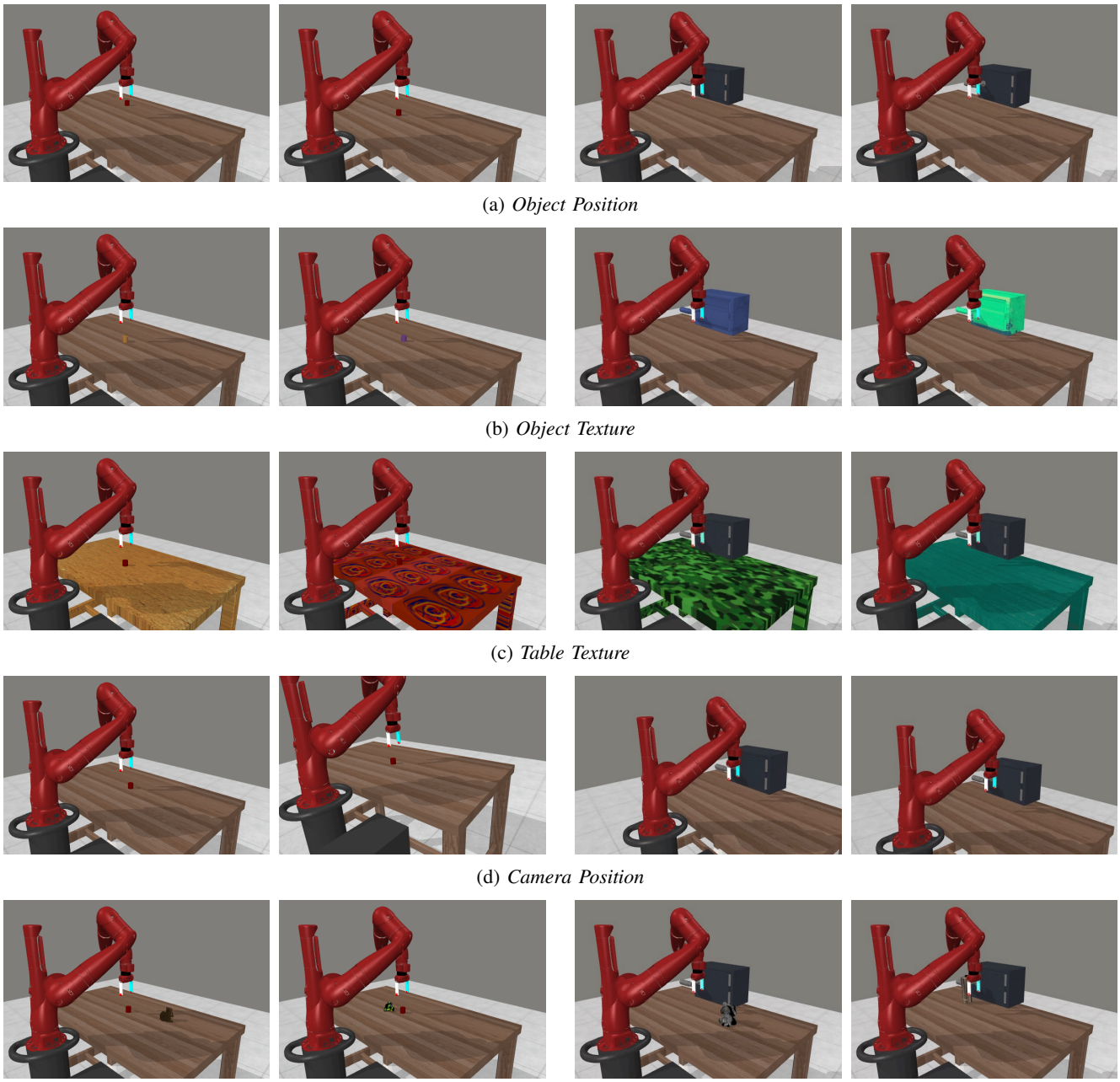
APPENDIX B
SIMULATION EXPERIMENTS

A. Door Open Results

We include additional simulation results for the task *Door Open* in Fig. 11. Results are similar as in *Pick Place*, with generally strong pairwise composition, and **Stair** generally performing the best, especially in the $N = 5$ setting.

B. Factors

For the inherently discrete-valued factors *object texture*, *table texture*, and *distractor objects*, we sample our $k = 10$ values for $\mathcal{F}^N$ from all possible training values specified in *Factor World*. *Distractor objects* also include a size scale (sampled from range $[0.3, 0.8]$), 2D rotation (sampled from range $[0, 2\pi]$), and 2D position (sampled from all possible positions on the table) as part of each value. For *object position*, we sample 2D xy positions from the range $[-0.1, 0.1]$ for both coordinates. We note that the *Pick Place* task includes



(a) *Object Position*

(b) *Object Texture*

(c) *Table Texture*

(d) *Camera Position*

(e) *Distractor Objects*

Fig. 12: Visualization of our factors for the *Pick Place* (left) and *Door Open* (right) tasks from *Factor World*. We show two examples of values for each factor we consider.

a small amount of added noise to *object position* each episode, sampled uniformly from the range $[-0.03, 0.03]$. For *camera position*, we sample 3D xyz positions and 4D rotation quaternions all from the range $[-0.05, 0.05]$. When sampling our k values for each factor, we ensure that the scripted policy is able to solve the task for each value, because some values for *object position* and *distractor objects* can impede the task. In Fig. 12, we visualize two examples of values for each factor, for both the *Pick Place* and *Door Open* tasks. We note that each random seed in our evaluation includes a different set of $k = 10$ values sampled for each factor for $\mathcal{F}^N$.

C. Training

We use the same policy architecture and training hyperparameters from the original *Factor World* experiments [49]. We condition policies on the same observations (2 84x84 RGB images from 2 camera views without history, and proprioception), and use the same action space (3D end-effector position deltas and open/close gripper). Unlike the original *Factor World* experiments, we always use random shift augmentation. Unlike our real robot experiments, we train policies without goal conditioning, as we do not leverage diverse prior data in this setting, so task conditioning is not as essential.

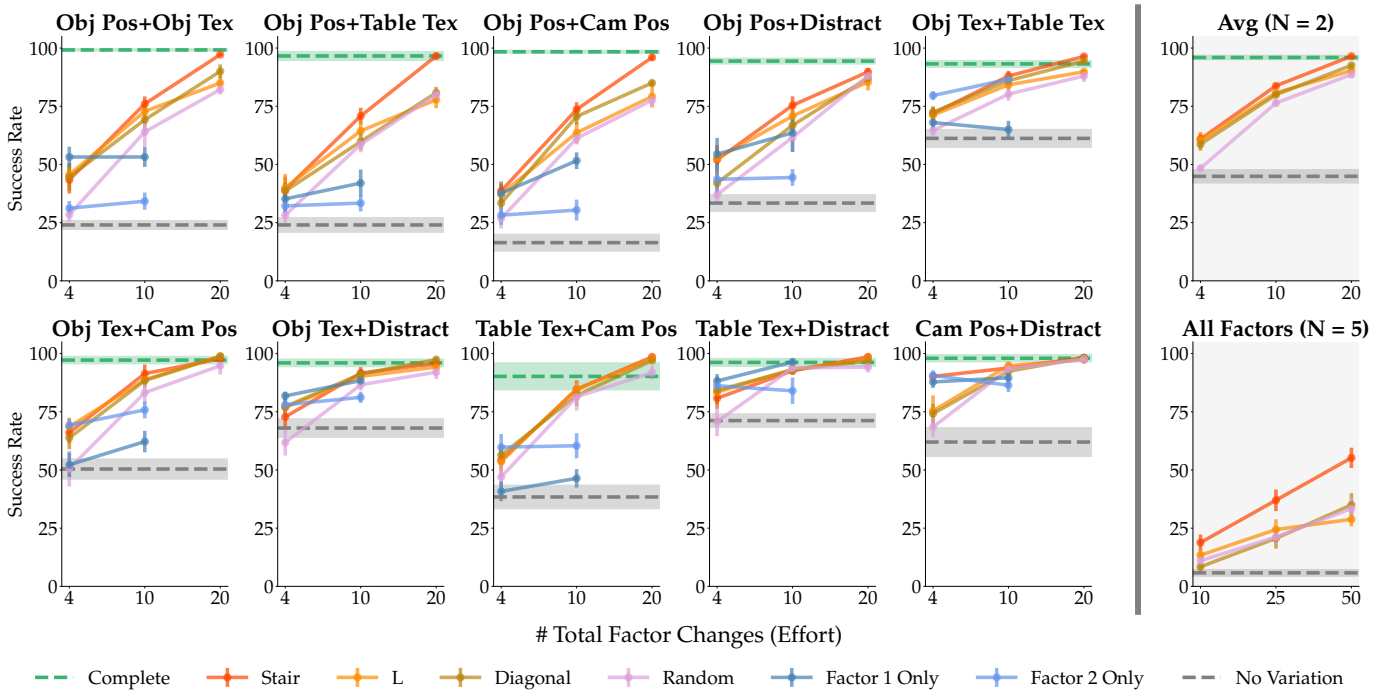


Fig. 13: Simulation results of data collection strategies for *Pick Place*, with additional color jitter augmentation. Overall performance across all strategies slightly improves, but the overall trends are similar to as without color jitter. Error bars represent standard error across 5 seeds.

D. Color Jitter Augmentation

We do not use color jitter augmentation in our main simulation experiments, because the original experiments for *Factor World* found this to reduce overall generalization, except for when training variation was very low [49]. However, here we include additional results with color jitter (in addition to random shift). We find that overall performance across all strategies does slightly improve, but the overall trends across strategies are similar to as without color jitter.

E. Accounting for Factor Value Similarity

In practice, data collection strategies should ideally account for similarity between factor values to improve generalization. For example, when training policies to be robust to the factor *object type*, it would be desirable to prioritize object diversity during data collection to generalize to as many objects as possible, rather than collecting data for overly similar objects. Here, we demonstrate how our proposed data collection framework could incorporate notions of factor similarity when available, through additional *Pick Place* experiments.

We consider the composition of *object position* and *camera position* in the $N = 2$ setting, as these are the only factors we study in simulation where computing similarity/distance is straightforward (we use Euclidean distance for 3D positions, and angular distance for camera rotation quaternions). We modify **Stair** to choose factor values using a similarity-aware strategy, rather than randomly as before. We do this by running a k-medoids algorithm on the set of 10 values for each factor, to determine which k values/medoids in the set minimize the sum of distances from each value to its nearest medoid. We determine k according to what is permissible under different factor change budgets (e.g., for 10 total changes in the $N = 2$ setting, each factor can have $k = \frac{10}{2} = 5$ values).

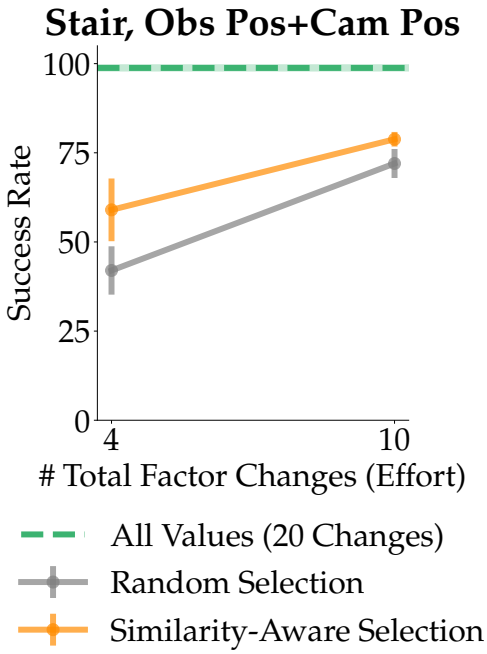


Fig. 14: Performance of the **Stair** strategy for the *Pick Place* task, when composing *object position* and *camera position*. We find that selecting factor values using a similarity-aware strategy outperforms random selection (as done in our main simulation results).

In Fig. 14, we compare using k-medoids for factor value selection (orange), and random selection as done in our main results (gray). We evaluate using the same procedure as in our main simulation results in Section IV-B, where the aim is to generalize to all factor value combinations. We find that

(a) *BaseKitch* (b) *CompKitch* (c) *TileKitch*

Fig. 15: Additional views of each kitchen in our real robot experiments. We name *CompKitch* and *TileKitch* after their countertop materials of composite and tile, respectively. These images were taken after our evaluations, so there may be some slight differences from then.

accounting for similarity does indeed improve upon random selection, although it does not match observing all factor values (green), which requires 20 factor changes.

We note that it often not straightforward to compute similarity metrics for other factors, and that while accounting for similarity can be easily incorporated into our data collection framework, it is mostly orthogonal to our study of composition. We believe it would be interesting for future work to investigate methods of computing similarity metrics for other factors, such as by using text and/or image embeddings, and leveraging this to further improve data collection.

F. Computing Factor Changes

For our results in Section IV-B, we compute the total number of factor changes for each strategy as follows. We assume the initial configuration of factor values requires n changes, one for each factor. For **Stair**, **L**, and **Single Factor**, we assume each new configuration of factor values requires 1 additional change. We note this could be a slight underestimate for **L** in practice, as this strategy could require some additional changes when finishing varying one factor and returning to the base factor values f^*, to begin data collection for varying another factor. For **Diagonal** and **Random**, we assume each new configuration of factor values requires n additional changes, as new values for all factors are resampled. We note that this could be a slight overestimate for **Random** in practice, as some of the resampled factor values may not change.

For each budget of factor value changes, we determine the amount of factor value configurations allowed for each strategy. We then divide the budget of total demonstrations by this to determine how many to collect for each configuration, with any remainder going to the last configuration.

APPENDIX C
REAL ROBOT EXPERIMENTS

A. Robot Platform

We use Logitech C920 webcams for our main and secondary third-person cameras. Our setup also includes the wrist camera used in BridgeData V2, and we experimented with using it in addition to our main third-person camera. While we found that it improved overall robustness when training from scratch, particularly for some factors that wrist cameras provide invariance to (e.g., *object position*, *table height*), policies still benefited from data coverage for these factors. More importantly, we also found that it was incompatible with using BridgeData V2 as prior data, reducing performance when using prior data compared to not using prior data at all. We believe this is because only a small fraction of BridgeData V2 contains wrist camera data, as similar negative results have been observed in other work that use prior robotic datasets with a small proportion of wrist camera data [35]. As we were only able to achieve successful transfer in our experiments in Section V-C by using prior data without a wrist camera, we decided to omit the wrist camera in all our experiments.

B. Evaluation Protocol

For our out-of-domain transfer experiments in Section V-C, we name our transfer kitchens *CompKitch* and *TileKitch* after their countertop materials of composite and tile, respectively. We provide additional views of these kitchens, as well as *BaseKitch*, in Fig. 15 to make the difference between these kitchens more apparent. These images were taken after our evaluations, so there may be some slight differences from then.

Our factor *table height* refers to the height of the object table (where objects are manipulated on) relative to the robot. However, we vary this factor in practice by adjusting the height of the mobile table the robot and third-person cameras are mounted on. This still changes the relative height of the robot with respect to the object table, the same as if the object table changed height. For example, in Section V-C, **Higher Table** refers to the object table being higher relative to the robot, which was achieved by lowering the robot's table.

We collect demonstrations using a Meta Quest 2 VR headset for teleoperation. All demonstrations are collected by a single experienced human teleoperator for consistency. We collect our **Stair** dataset by starting at f^*, and then varying factors cyclically in the order *object position*, *object type*, *container type*, *table height*, *table texture*. The order we vary values for each factor is the same as in Fig. 7, from top to bottom.

C. Training

We use the same ResNet-34 diffusion goal-conditioned policy architecture from the original BridgeData V2 experiments, except we condition on a history of 2 128x128 RGB image

Data Strategy	**L**									
Train Method	Bridge					From Scratch				
Factor 1 \ Factor 2	Object Pos	Object Type	Container Type	Table Height	Table Tex	Object Pos	Object Type	Container Type	Table Height	Table Tex
Object/Container Pos	N/A	**7/9**	**6/9**	**2/9**	**6/9**	N/A	5/9	2/9	0/9	3/9
Object Orientation	**2/9**	**5/9**	**5/9**	**1/9**	**3/9**	0/9	2/9	3/9	0/9	1/9
Overall	**36/81**					16/81				

TABLE VI: Additional real robot pairwise composition results for our "*put fork in container*" task, with additional factors *object/container position* and *object orientation*. Similarly as in our main pairwise composition results in Table I, leveraging BridgeData V2 as prior data significantly improves composition for these factors compared to training from scratch.

observations from a third-person view (in addition to a goal image from the same view), and use action chunking to predict the next 4 actions. However, we noticed better performance during inference by only executing the first predicted action, so we do this for all experiments unless otherwise stated. We share the same visual encoder across image observations in the history. We use the same 7D action space as the original experiments (6D end-effector pose deltas, and open/close gripper). We use the same data augmentation from the original experiments, which consists of random crops, random resizing, and color jitter.

For policies using BridgeData V2 as prior data, we first pre-train a model on BridgeData V2 using the original training hyperparameters for 2M gradient steps. As done in the original BridgeData V2 experiments, 10% of trajectories in this data is reserved for validation. We checkpoint every 50K steps, and choose the checkpoint with the lowest validation action prediction mean-squared error as the initialization for later fine-tuning. During fine-tuning, we use the same hyperparameters as during pre-training, except we train for only 300K gradient steps. Also, we do not use a validation set, and instead simply evaluate the final checkpoint. When co-fine-tuning, we train on a mixture of 75% in-domain data and 25% prior data.

D. Pairwise Composition for Additional Factors

We conduct experiments for two new factors: *object/container position* and *object orientation*. We visualize these factors and their values in Fig. 16. Note that *object/container position* involves new positions of both the fork and container, with more significant changes compared to our previous values for *object position*. We extended the **L** dataset for our original factors with 10 demonstrations for each new factor value, re-trained policies on this extended dataset, and evaluated pairwise composition with these new factors. Unlike our previous results in Section V-B, we do not evaluate the **No Variation** Bridge policy, because we found it was unable to succeed at all with shifts for these factors in isolation, so there was no potential for composition.

While these policies worked for *object/container position*, we found they was unable to perform the task for the *object orientation* shifts we considered in isolation. We hypothesize this could be because different values for this factor have relatively small changes in their visual observations, but require

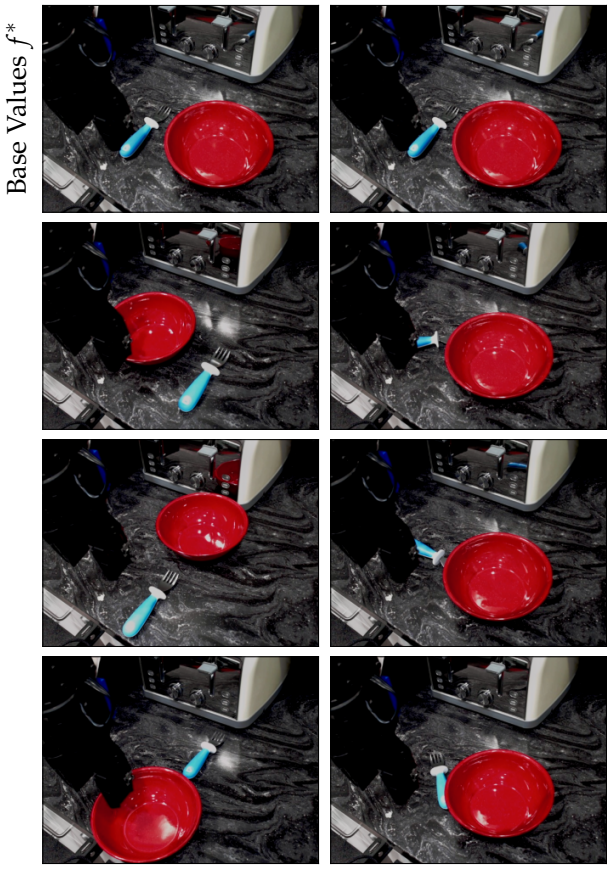


Fig. 16: Visualization of our additional real robot factors in *BaseKitch*. The top row shows our base factor values f^*. The other rows show all deviations from f^* by one factor value.

significantly different behavior, which can make it challenging to learn when to apply the correct behavior. Therefore, for our evaluation on composing *object orientation*, we trained separate policies where we balance training batches such that 50% of in-domain data consists of data for *object orientation*.

We report these results in Table VI. We find that with prior data, *object/container position* achieves similar composition as our original *object position* factor, with a success rate of **20/36** for both. However, when training from scratch, the policy is able to compose this new factor more effectively than the

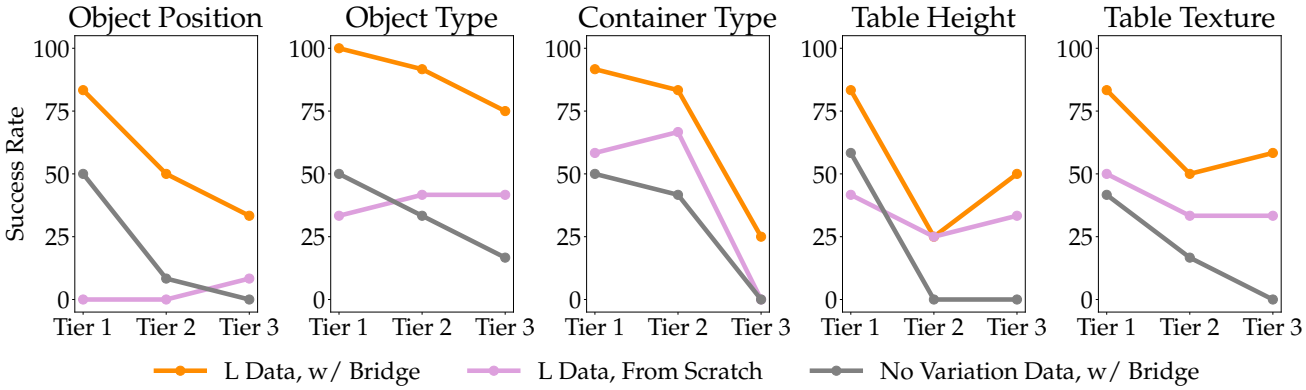


Fig. 17: Per-factor value success rates of policies from our pairwise composition results in Table I. Factor values are placed in tiers, where increasing tiers are more dissimilar from the base factor values f^*. Composition is generally more challenging for factor values that are more dissimilar from the base factor values.

Factor	Tier 1	Tier 2	Tier 3
Object Pos	Down (3)	Up (4)	Left (2)
Object Type	Wooden (3)	Gray (4)	Plastic (2)
Container Type	Blue Plate (4)	Pink Bowl (3)	White Cup (2)
Table Height	Higher 5cm (2)	Lower 5cm (3)	Lower 8cm (4)
Table Tex	Brown Wood (2)	Gift Wrap (4)	White Marble (3)

TABLE VII: Tiers for each factor value used in Fig. 17, determined using our success rate-based similarity metric as described in Appendix C-E. In parentheses next to each factor, we provide the row number where the factor value is visualized in Fig. 7.

original, achieving a success rate of **10/36** compared to **1/36**. This could be because different values for *object/container position* are more visually distinguishable, and their required behavior is also significantly more different, which could make it easier to learn when to apply the correct behavior.

Our policies can sometimes compose *object orientation*, although composition for this is the weakest compared to the other factors we study. This could be due to the aforementioned challenges with learning for this factor, as well as because our data balancing may have insufficiently represented the other factors.

E. Factor Similarity Analysis

Here, we provide additional analysis on when our policies are able to compose factor values from our pairwise evaluation in Section V-B. To do this, for each factor, we consider how similar each of the non-base factor values we consider are to the base factor value, with respect to a policy's ability to generalize across factor values.

To compute a similarity metric, we consider the success rates from our results in Table I, in particular for the policy trained on **No Variation** data and BridgeData V2. We then obtain a success rate for each non-base factor value, by aggregating the results for all factor value pairs that contain that factor value. We use this success rate as our similarity metric, where higher success rates indicate greater similarity, because this captures how well a policy trained on base factor values generalizes to other factor values.

Using this similarity metric, we rank the non-base factor values for each factor as *Tier 1*, *Tier 2*, or *Tier 3*, where a higher tier is more dissimilar from the base factor value. We list the tiers for each factor value in Table VII. We then take the aggregated per-factor value success rates for each policy from our results in Table I (where we aggregate success rates as we do for computing the similarity metric), and plot this against our factor value tiers in Fig. 17.

We find that the policies trained on **L** data (orange and pink lines) generally achieve lower compositional success rates for factor values that are more dissimilar from the base factor values f^*, suggesting that composition is more challenging for these more dissimilar values, although this trend is not strictly monotonic. We note that part of this effect could be because our success rate-based similarity metric may also capture how challenging factor values are in general.

F. R3M

We use the ResNet-50 version of R3M. When training, we pre-compute representations beforehand, and standardize the dataset such that each feature has mean 0 and standard deviation 1 (similar to the batch normalization used in the original R3M experiments). We use the training dataset mean and standard deviation for normalization during inference. We feed normalized representations to the same diffusion policy head architecture used when learning end-to-end, except we do not use goal-conditioning. We increase the amount of training gradient steps from 300K to 500K, to reduce training loss. We also tried 1M gradient steps, which reduces loss even further, but this resulted in worse performance. Instead of executing only the first predicted action as with the end-to-end policies, we execute all 4, which we found to slightly reduce jitteriness. Like when learning end-to-end, we verify our R3M policies are able to succeed with base factor values f^* in *BaseKitch*.

G. VC-1

We train and evaluate VC-1 policies using the same procedure as with R3M, except using the ViT-L version of VC-1 instead of R3M.