Decomposing the Generalization Gap in Imitation Learning for Visual Robotic Manipulation

Annie Xie[1]*, **Lisa Lee**[2]*, **Ted Xiao**[2], **Chelsea Finn**[1]
[1]Stanford University, [2]Google DeepMind

Abstract

What makes generalization hard for imitation learning in visual robotic manipulation? This question is difficult to approach at face value, but the environment from the perspective of a robot can often be decomposed into enumerable *factors of variation*, such as the lighting conditions or the placement of the camera. Empirically, generalization to some of these factors have presented a greater obstacle than others, but existing work sheds little light on precisely how much each factor contributes to the generalization gap. Towards an answer to this question, we study imitation learning policies in simulation and on a real robot language-conditioned manipulation task to quantify the difficulty of generalization to different (sets of) factors. We also design a new simulated benchmark of 19 tasks with 11 factors of variation to facilitate more controlled evaluations of generalization. From our study, we determine an ordering of factors based on generalization difficulty, that is consistent across simulation and our real robot setup.[1]

1 Introduction

Robotic policies often fail to generalize to new environments, even after training on similar contexts and conditions. In robotic manipulation, data augmentation techniques [21, 38, 12, 40, 11] and representations pretrained on large datasets [39, 27, 20, 32, 30, 25, 24] improve performance but a gap still remains. Simultaneously, there has also been a focus on the collection and curation of reusable robotic datasets [31, 23, 22, 6, 9], but there lacks a consensus on how much more data, and what *kind* of data, is needed for good generalization. These efforts could be made significantly more productive with a better understanding of which dimensions existing models struggle with. Hence, this work seeks to answer the question: *What are the factors that contribute most to the difficulty of generalization to new environments in vision-based robotic manipulation?*

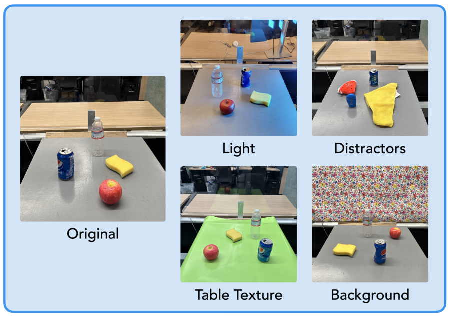


Figure 1: Examples of our real robot environment. We systematically vary factors, including the lighting condition, distractor objects, table texture, background, and camera pose.

To approach this question, we characterize environmental variations as a combination of independent factors, namely the background, lighting condition, distractor objects, table texture, object texture, table position, and camera position. This decomposition allows us to quantify how much each factor contributes to the generalization gap, which we analyze in the imitation learning setting (see Fig. 3 for a summary of our real robot evaluations). While vision models are robust to many of these factors already [15, 14, 10], robotic policies are considerably less mature, due to the smaller and

[1] Videos are available at: `https://sites.google.com/stanford.edu/gengap-icra`.

NeurIPS 2023 Workshop on Robot Learning: Pretraining, Fine-Tuning, and Generalization with Large Scale Models, New Orleans, United States of America

less varied datasets they train on. In robot learning, data collection is largely an *active* process, in which robotics researchers design and control the environment the robot interacts with. As a result, naturally occurring variations, such as different backgrounds, are missing in many robotics datasets. Finally, robotics tasks require dynamic, multi-step decisions, unlike computer vision tasks such as image classification. These differences motivate our formal study of these environment factors in the context of robotic manipulation. In our study, we evaluate a real robot manipulator on over 20 test scenarios featuring new lighting conditions, distractor objects, backgrounds, table textures, and camera positions. We also design a suite of 19 simulated tasks, equipped with 11 customizable environment factors, which we call *Factor World*, to supplement our study. With over 100 configurations for each factor, *Factor World* is a rich benchmark for evaluating generalization, which we hope will facilitate more fine-grained evaluations of new models, reveal potential areas of improvement, and inform future model design.

2 Related Work

Datasets and benchmarks. Existing robotics datasets exhibit rich diversity along multiple dimensions, including objects [18, 6, 9, 2], domains [6, 40, 9], and tasks [31, 23, 22]. However, collecting high-quality and diverse data *at scale* is still an unsolved challenge, which motivates the question of how new data should be collected given its current cost. The goal of this study is to systematically understand the challenges of generalization to new objects and domains and, through our findings, inform future data collection strategies. Simulation can also be a useful tool for understanding the scaling relationship between data diversity and policy performance, as diversity in simulation comes at a much lower cost [34, 26, 4, 16]. Many existing benchmarks aim to study exactly this [41, 5, 33, 37]; these benchmarks evaluate the generalization performance of control policies to new tasks [41, 5] and environments [33, 37]. *KitchenShift* [37] is the most related to our contribution *Factor World*, benchmarking robustness to shifts like lighting, camera view, and texture. However, *Factor World* contains a more complete set of factors (11 versus 7 in *KitchenShift*) with many more configurations of each factor (over 100 versus fewer than 10 in *KitchenShift*).

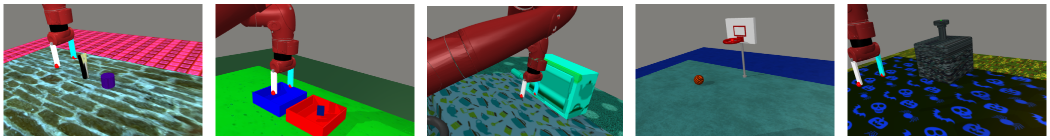
Generalization studies. Several prior works have studied the robustness of robotic policies to different environmental shifts, such as harsher lighting, new backgrounds, and new distractor objects [17, 37, 2, 43]. Many interesting observations have emerged from them, such as how mild lighting changes have little impact on performance [17] and how new backgrounds (in their case, new kitchen countertops) have a bigger impact than new distractor objects [2]. However, these findings are often qualitative or lack specificity. For example, the performance on a new kitchen countertop could be attributed to either the appearance or the height of the new counter. A goal of our study is to formalize these prior observations through systematic evaluations and to extend them with a more comprehensive and fine-grained set of environmental shifts.

3 Environment Factors

To draw more robust conclusions, our study is conducted across three different domains: on a real robot and in the *Factor World* and *KitchenShift* [37] simulators. On the real robot and in *KitchenShift*, we use pre-existing datasets to understand how models trained on them are impacted by factored variations. Importantly, the environmental factors and distributions over them are *not* designed by us, and thus are representative of experimental setups studied in robotics research. To augment these domains, we also design *Factor World* which allows easier control over individual factors and generation of datasets with specific factor distributions. See App. A for more details on the implementation of the environment factors in simulation.

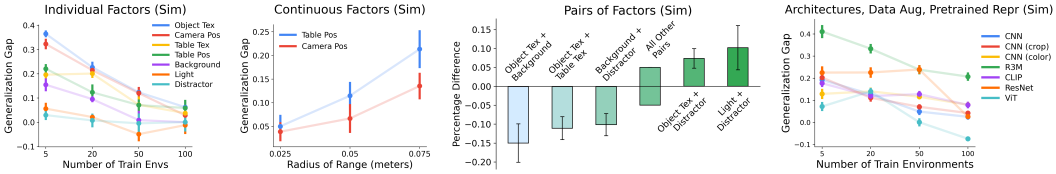
3.1 Real Robot Manipulation

In our real robot evaluations, we study: lighting condition, distractor objects, background, table texture, and camera pose. In addition to selecting factors that are specific and controllable, we also take inspiration from prior work, which has studied robustness to many of these shifts [17, 37, 2], thus signifying their relevance in real-world scenarios. Our experiments are conducted with mobile manipulators. The robot has a right-side arm with seven DoFs, gripper with two fingers, mobile base, and head with integrated cameras. The environment, visualized in Fig. 1, consists of a cabinet top that serves as the robot workspace and an acrylic wall that separates the workspace and office background. To control the lighting condition in our evaluations, we use several bright LED light sources with different colored filters to create colored hues and new shadows. We introduce new table textures



(a) Pick Place (b) Bin Picking (c) Door (Open, Lock) (d) Basketball (e) Button (Top, Side, Wall)

Figure 2: *Factor World*, a suite of 19 visually diverse robotic manipulation tasks. Each task can be configured with multiple factors of variation such as lighting; texture, size, shape, and initial position of objects; texture of background (table, floor); position of the camera and table relative to the robot; and distractor objects.



(a) (b) (c) (d)

Figure 4: (a) Generalization gap under shifts to individual factor in *Factor World*. (b) Generalization gap versus the radius of the range that camera and table positions are sampled from, in *Factor World*. (c) Performance on pairs of factors, reported as the percentage difference relative to the harder factor of the pair, in *Factor World*. All results are averaged across the 3 simulated tasks with 5 seeds for each task. Error bars represent standard error of the mean. (d) Generalization gap with data augmentations, pretrained representations, and different architectures in *Factor World*. Lower is better. Results are averaged across the 7 factors, 3 tasks, and 5 seeds for each task.

and backgrounds by covering the cabinet top and acrylic wall, respectively, with patterned paper. We shift the camera pose by changing the robot's head orientation. Due to the impracticality of studying factors like the table position and height, we reserve them for our simulated experiments.

4 Experimental Results

Our experiments aim to answer the following questions. How much does each environment factor contribute to the generalization gap? (Sec. 4.1) What effects do data augmentation, pretrained representations, and model architecture have on the generalization performance? (Sec. E.2) How do different data collection strategies, such as prioritizing visual diversity in the data, impact downstream generalization? (Sec. E.3) We also study different image resolutions and control frequencies. The results of these ablations are on the website.

4.1 Impact of Environment Factors on Generalization

Individual factors. We begin our real robot evaluation by benchmarking the model's performance on the set of six training tasks, with and without shifts. Without shifts, the policy achieves an average success rate of 91.7%. Our results with shifts are presented in Fig. 7, as the set of green bars. We find that the new backgrounds have little impact on the performance (88.9%), while new distractor objects and new lighting conditions have a slight effect, decreasing success rate to 80.6% and 83.3% respectively. Finally, changing the table texture and camera orientation causes the biggest drop, to 52.8% and 45.8%, as the entire dataset uses a fixed head pose. Since we use the same patterned paper to introduce variations in backgrounds and table textures, we can directly compare these two factors, and conclude that new textures are harder to generalize to than new backgrounds.

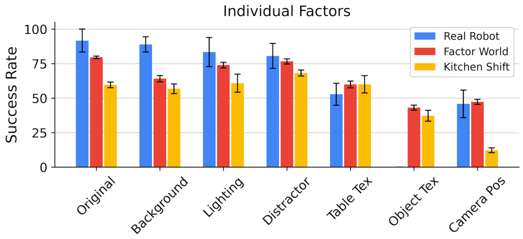


Figure 3: Success rates on different shifts across 3 domains. Object texture is not evaluated on the robot.

Fig. 4a compares the generalization gap due to each individual factor on *Factor World*. We plot this as a function of the number of training environments represented in the dataset, where an environment is parameterized by the sampled value for each factor of variation. The success rates under individual factor shifts in *KitchenShift* are visualized in Fig. 3. **Consistent across simulated and real-world results, new backgrounds, distractors, and lighting are easier factors to generalize to, while new table textures and camera positions are harder.** In *Factor World*, new backgrounds are harder than distractors and lighting, in contrast to the real robot results, where they were the easiest. This may be because the real robot dataset contains a significant amount of background diversity, relative to the lighting and distractor factors, as described in Sec. B.1. In *Factor World*, we additionally study object textures and table positions, including the height of the table. New object textures are about as hard to overcome as camera positions, and new table positions are as hard as table textures. Fortunately, the generalization gap closes significantly for *all* factors, from a maximum gap of 0.4 to less than 0.1, when increasing the number of training environments from 5 to 100. Notably, table textures are easier in *KitchenShift* compared to *Factor World* and the real robot. This is likely because while the texture of the counter changes, the texture of the stovetop, on which the kettle lies, does not.

Pairs of factors. Next, we evaluate with respect to pairs of factors to understand how they interact, i.e., whether generalization to new pairs is harder (or easier) than generalizing to one of them. On the real robot, we study the factors with the most diversity in the training dataset: table texture + distractors and table texture + background. Introducing new background textures or new distractors on top of a new table texture does not make it any harder than the new table texture alone (green bars in Fig. 7). The success rate with new table texture + background is 55.6% and with new table texture + distractors is 50.0%, comparable to the evaluation with only new table textures, which is 52.8%.

In *Factor World*, we evaluate all 21 pairs, and report the success rate gap, normalized by the harder of the two factors. Concretely, this metric is defined as $(P_{A+B} - \min(P_A, P_B)) \,/\, \min(P_A, P_B)$, where P_A is the success rate under shifts to factor A, P_B is the success rate under shifts to factor B, and P_{A+B} is the success rate under shifts to both. **Most pairs of factors do not have a compounding effect on generalization performance.** For 16 of 21 pairs, the relative percentage difference in the success rate lies between -6% and 6%. In other words, generalizing to the combination of two factors is not significantly harder or easier than individual factors. In Fig. 4c, we visualize the performance difference for the remaining 5 factor pairs that lie outside of this $(-6\%, 6\%)$ range (see website for results with all factor pairs). Interestingly, the following factors combine synergistically, making it easier to generalize to compared to the (harder of the) individual factors: object texture + distractor and light + distractor. This result suggests these factors can be studied independently of one another, and improvements with respect to one factor may carry over to multiple factor shifts.

5 Discussion

In this work, we studied the impact of different environmental variations on generalization performance. We determined an ordering of the environment factors in terms of generalization difficulty, that is consistent across simulation and our real robot setup, and quantified the impact of different solutions like data augmentation. Notably, many of the solutions studied were developed for computer vision tasks like image classification. While some of them transferred well to the robotic imitation learning setting, it may be fruitful to develop algorithms that prioritize this setting and its unique considerations, including the sequential nature of predictions and the often continuous, multi-dimensional action space in robotic setups. We hope this work encourages researchers to develop solutions that target the specific challenges in robotic generalization identified by our work.

References

[1] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[2] 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.

[3] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

[4] Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8973–8979. IEEE, 2019.

[5] Karl Cobbe, Chris Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. In *International conference on machine learning*, pages 2048–2056. PMLR, 2020.

[6] 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. *arXiv preprint arXiv:1910.11215*, 2019.

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[8] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B. McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items, 2022.

[9] 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. *arXiv preprint arXiv:2109.13396*, 2021.

[10] Robert Geirhos, Kantharaju Narayanappa, Benjamin Mitzkus, Tizian Thieringer, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Partial success in closing the gap between human and machine vision. *Advances in Neural Information Processing Systems*, 34:23885–23899, 2021.

[11] Christian Graf, David B Adrian, Joshua Weil, Miroslav Gabriel, Philipp Schillinger, Markus Spies, Heiko Neumann, and Andras Kupcsik. Learning dense visual descriptors using image augmentations for robot manipulation tasks. *arXiv preprint arXiv:2209.05213*, 2022.

[12] Nicklas Hansen and Xiaolong Wang. Generalization in reinforcement learning by soft data augmentation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13611–13617. IEEE, 2021.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[14] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8349, 2021.

[15] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[16] Stephen James, Paul Wohlhart, Mrinal Kalakrishnan, Dmitry Kalashnikov, Alex Irpan, Julian Ibarz, Sergey Levine, Raia Hadsell, and Konstantinos Bousmalis. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12627–12637, 2019.

[17] Ryan Julian, Benjamin Swanson, Gaurav S Sukhatme, Sergey Levine, Chelsea Finn, and Karol Hausman. Never stop learning: The effectiveness of fine-tuning in robotic reinforcement learning. *arXiv preprint arXiv:2004.10190*, 2020.

[18] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673. PMLR, 2018.

[19] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673. PMLR, 2018.

[20] Apoorv Khandelwal, Luca Weihs, Roozbeh Mottaghi, and Aniruddha Kembhavi. Simple but effective: Clip embeddings for embodied ai. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14829–14838, 2022.

[21] Misha Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. *Advances in neural information processing systems*, 33:19884–19895, 2020.

[22] Ajay Mandlekar, Jonathan Booher, Max Spero, Albert Tung, Anchit Gupta, Yuke Zhu, Animesh Garg, Silvio Savarese, and Li Fei-Fei. Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1048–1055. IEEE, 2019.

[23] 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.

[24] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[25] Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. The unsurprising effectiveness of pre-trained vision models for control. *arXiv preprint arXiv:2203.03580*, 2022.

[26] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3803–3810. IEEE, 2018.

[27] 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.

[28] Michael Ryoo, AJ Piergiovanni, Anurag Arnab, Mostafa Dehghani, and Anelia Angelova. Tokenlearner: Adaptive space-time tokenization for videos. *Advances in Neural Information Processing Systems*, 34:12786–12797, 2021.

[29] Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International Conference on Machine Learning*, pages 8583–8592. PMLR, 2020.

[30] Rutav Shah and Vikash Kumar. Rrl: Resnet as representation for reinforcement learning. *arXiv preprint arXiv:2107.03380*, 2021.

[31] 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.

[32] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pages 894–906. PMLR, 2022.

[33] Austin Stone, Oscar Ramirez, Kurt Konolige, and Rico Jonschkowski. The distracting control suite–a challenging benchmark for reinforcement learning from pixels. *arXiv preprint arXiv:2101.02722*, 2021.

[34] Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *arXiv preprint arXiv:1804.10332*, 2018.

[35] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 09–15 Jun 2019.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[37] 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.

[38] Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021.

[39] Lin Yen-Chen, Andy Zeng, Shuran Song, Phillip Isola, and Tsung-Yi Lin. Learning to see before learning to act: Visual pre-training for manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7286–7293. IEEE, 2020.

[40] Sarah Young, Dhiraj Gandhi, Shubham Tulsiani, Abhinav Gupta, Pieter Abbeel, and Lerrel Pinto. Visual imitation made easy. In *Conference on Robot Learning*, pages 1992–2005. PMLR, 2021.

[41] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020.

[42] Kevin Zakka. Scanned Objects MuJoCo Models, 7 2022.

[43] Gaoyue Zhou, Victoria Dean, Mohan Kumar Srirama, Aravind Rajeswaran, Jyothish Pari, Kyle Beltran Hatch, Aryan Jain, Tianhe Yu, Pieter Abbeel, Lerrel Pinto, et al. Train offline, test online: A real robot learning benchmark. In *Deep Reinforcement Learning Workshop NeurIPS 2022*, 2022.

A Environment Factors

Factor World. We implement the environmental shifts on top of *Meta World* [41]. While *Meta World* is rich in diversity of control behaviors, it lacks diversity in the environment, placing the same table at the same position against the same background. Hence, we implement 11 different factors of variation, visualized in Fig. 2 and fully enumerated on the supplementary website. These include lighting; texture, size, shape, and initial position of objects; texture of the table and background; the camera pose and table position relative to the robot; the initial arm pose; and distractor objects. In our study, we exclude the object size and shape, as an expert policy that can handle any object is more difficult to design, and the initial arm pose, as this can usually be fixed whereas the same control cannot be exercised over the other factors, which are inherent to the environment.

Textures (table, floor, objects) are sampled from 162 texture images (81 for train, 81 for eval) and continuous RGB values in $[0, 1]^3$, which modifies the texture image. Distractor objects are sampled from 170 object meshes (100 for train, 70 for eval) in Google's Scanned Objects Dataset [42, 8]. For lighting, we sample continuous ambient and diffuse values in $[0.2, 0.8]$. Changes in camera and table positions are sampled from $[-0.025, 0.025]$ meters. While fixing the initial position of an object across trials is feasible with a simulator, it is generally difficult to precisely replace an object to its original position in physical setups. Thus, we randomize the initial position of the object (between $[-0.1, 0.1]$ meters) in each episode in the experiments.

KitchenShift [37]. In addition to *Factor World*, we examine a second simulated environment, *KitchenShift*. *KitchenShift* modifies Franka Kitchen with variations to the lighting, camera view, and textures (object, counter, and floor). There are 4 lighting settings, 10 camera positions, 4 counter textures, 7 floor textures, 4 microwave models, 6 cabinet textures, and 8 kettle models. The microwave and cabinets represent distractors, while the kettle models are different object textures. The table position is fixed in *KitchenShift*.

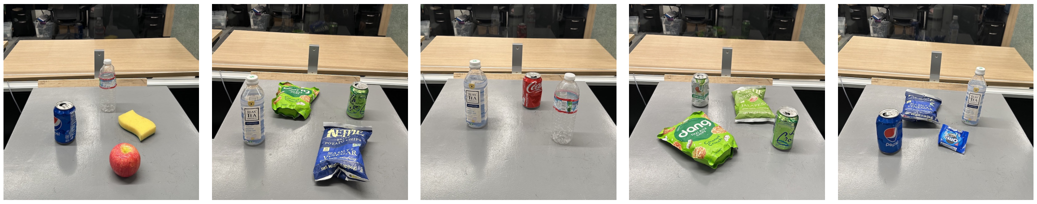
B Study Design

We seek to understand how each factor in Sec. 3 contributes to the difficulty of generalization. In our pursuit of an answer, we aim to replicate, to the best of our ability, the scenarios that robotics practitioners are likely to encounter in the real world. We therefore start by selecting a set of tasks commonly studied in the robotics literature and the data collection procedure (Sec. B.1). Then, we describe the algorithms studied and our evaluation protocol (Sec. B.2).

B.1 Control Tasks and Datasets

Real robot. We study the language-conditioned manipulation problem from [2], specifically, focusing on the "pick" skill for which the most data is available. The goal is to pick up the object specified in the language instruction. For example, when given the instruction "pick pepsi can", the robot should pick up the pepsi can among the distractor objects from the countertop (Fig. 1). We select six objects for our evaluation; all "pick" tasks can be found on the website. The observation consists of 300×300 RGB image observations from the last six time-steps and the language instruction, while the action controls movements of the arm (xyz-position, roll, pitch, yaw, opening of the gripper) and movements of the base (xy-position, yaw). The actions are discretized along each of the 10 dimensions into 256 uniform bins. The real robot manipulation dataset consists of over 115K human-collected demonstrations, collected across 13 skills, with over 100 objects, three tables, and three locations. The dataset is collected with a fixed camera orientation but randomized initial base position in each episode.

Simulation. While *Factor World* supports 19 manipulation tasks, our study focuses on 3 tasks commonly studied in robotics: pick-place (Fig. 2a), bin-picking (Fig. 2b), and door-open (Fig. 2c). In pick-place, the agent must move a block to the goal among a distractor object placed in the scene. In bin-picking, the agent must move a block from the right-side bin to the left-side bin. In door-open, the agent must pull on the door handle. We use scripted expert policies from the *Meta World* benchmark, which compute expert actions given the object poses, to collect demonstrations in each simulated task. The agent is given 84×84 RBG image observations, the robot's end-effector position from the last two time-steps, and the distance between the robot's fingers from the last two time-steps. The actions are the desired change in the 3D-position of the end-effector and whether to open or close the gripper. In *KitchenShift*, we study the kettle task, which requires moving the kettle from the bottom to the top burner. See [37] for environment details.



(a) Pepsi, water bottle (b) Blue chip bag (c) Blue plastic bottle (d) Green jalapeno chip bag (e) Oreo

Figure 5: The six pick tasks in our real robot evaluations.

B.2 Algorithms and Evaluation Protocol

The real robot manipulation policy uses the RT-1 architecture [2], which tokenizes the images, text, and actions, attends over these tokens with a Transformer [36], and trains with a language-conditioned imitation learning objective. In simulation, we equip vanilla behavior cloning with several different methods for improving generalization. Specifically, we evaluate techniques for image data augmentation (random crops and random photometric distortions) and evaluate pretrained representations (CLIP [27] and R3M [24]) for encoding image observations. More details on the implementation and training procedure can be found on the website.

Evaluation protocol. On the real robot task, we evaluate the policies on 2 new lighting conditions, 3 sets of new distractor objects, 3 new table textures, 3 new backgrounds, and 2 new camera poses. For each factor of interest, we conduct 2 evaluation trials in each of the 6 tasks, and randomly shuffle the object and distractor positions between trials. We report the success rate averaged across the 12 trials. To evaluate the generalization behavior of the trained policies in *Factor World*, we shift the train environments by randomly sampling 100 new values for the factor of interest, creating 100 test environments. In *KitchenShift*, we evaluate on 1 lighting setting, 7 camera positions, 1 counter texture, 4 floor textures, 1 microwave model, 3 cabinet textures, and 5 kettle models. We report the average **generalization gap**, which is defined as $P_T - P_F$, where P_T is the success rate on the train environments and P_F is the new success rate under shifts to factor F.

C Experimental Details

In this section, we provide additional details on the experimental setup and evaluation metrics.

C.1 Experimental Setup

Real robot tasks. We define six real-world picking tasks: pepsi can, water bottle, blue chip bag, green jalapeno chip bag, and oreo, which are visualized in Fig. 5.

Factor World. The factors of variation implemented into *Factor World* are enumerated in Fig. 6. In Table 1, we specify the ranges of the continuous-valued factors.

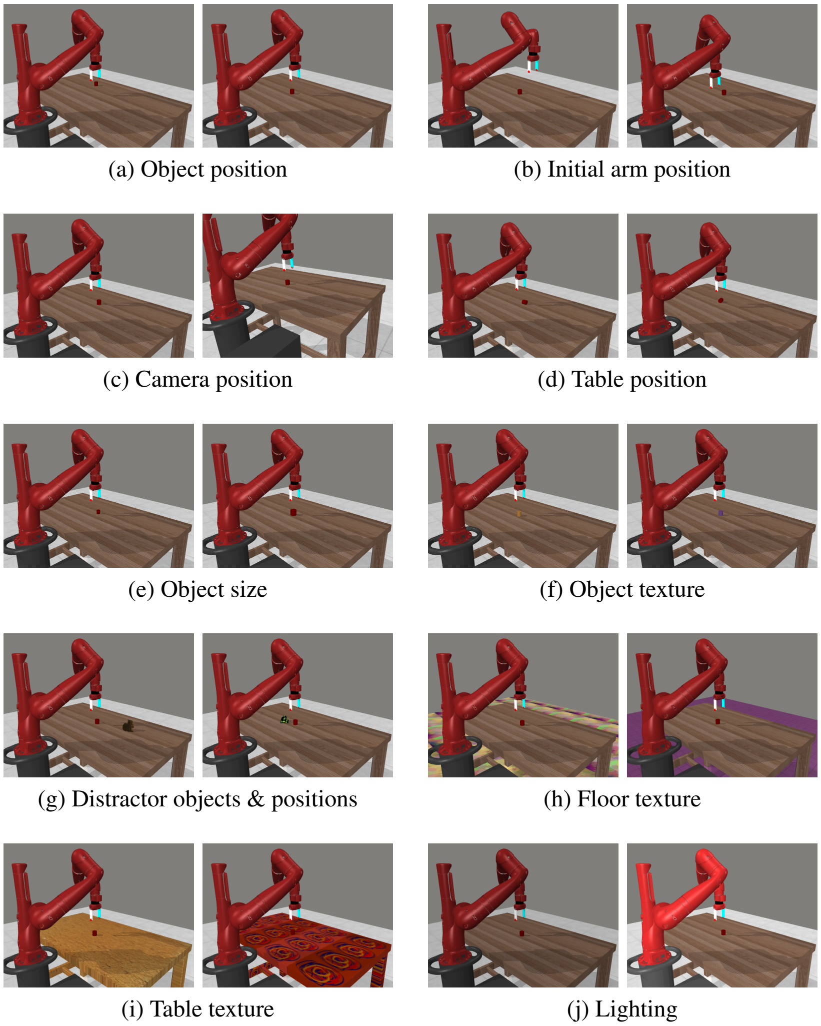
C.2 Dataset Details

Factor World datasets. In the `pick-place` task, we collect datasets of 2000 demonstrations, across $N = 5, 20, 50, 100$ training environments. A training environment is parameterized by a collection of factor values, one for each environment factor. We collect datasets of 1000 demonstrations for `bin-picking` and `door-open`, which we empirically found to be easier than the `pick-place` task.

C.3 Evaluation Metrics

Generalization gap. Besides the success rate, we also measure the generalization gap which is defined as the difference between the performance on the train environments and the performance on the test environments. The test environments have the same setup as the train environments, except 1 (or 2 in the factor pair experiments) of the factors is assigned a new value. For example, in Fig. 3, 'Background' represents the change in success rate when introducing new backgrounds to the train environments.

Percentage difference. When evaluating a pair of factors, we report the percentage difference with respect to the harder of the two factors. Concretely, this metric is computed as


(a) Object position

(b) Initial arm position

(c) Camera position

(d) Table position

(e) Object size

(f) Object texture

(g) Distractor objects & positions

(h) Floor texture

(i) Table texture

(j) Lighting

Figure 6: The 11 factors of variation implemented into *Factor World*, depicted for the `pick-place` environment. Videos are available at: `https://sites.google.com/view/factor-envs`

Factor	Parameters	Narrow	Medium	Wide
Object position	X-position	$[-0.05, 0.05]$	$[-0.1, 0.1]$	-
	Y-position	$[-0.05, 0.05]$	$[-0.075, 0.075]$	-
Camera position	X-position	$[-0.025, 0.025]$	$[-0.05, 0.05]$	$[-0.075, 0.075]$
	Y-position	$[-0.025, 0.025]$	$[-0.05, 0.05]$	$[-0.075, 0.075]$
	Z-position	$[-0.025, 0.025]$	$[-0.05, 0.05]$	$[-0.075, 0.075]$
	q_1	$[-0.025, 0.025]$	$[-0.05, 0.05]$	$[-0.075, 0.075]$
	q_2	$[-0.025, 0.025]$	$[-0.05, 0.05]$	$[-0.075, 0.075]$
	q_3	$[-0.025, 0.025]$	$[-0.05, 0.05]$	$[-0.075, 0.075]$
	q_4	$[-0.025, 0.025]$	$[-0.05, 0.05]$	$[-0.075, 0.075]$
Table position	X-position	$[-0.025, 0.025]$	$[-0.05, 0.05]$	$[-0.075, 0.075]$
	Y-position	$[-0.025, 0.025]$	$[-0.05, 0.05]$	$[-0.075, 0.075]$
	Z-position	$[-0.025, 0.025]$	$[-0.05, 0.025]$	$[-0.05, 0.025]$

Table 1: Range for each continuous factor in meters. As a point of comparison for the position-based factors, the table in the environment measures at $0.7m \times 0.4m$.

$(p_{A+B} - \min(p_A, p_B)) \,/\, \min(p_A, p_B)$, where p_A is the success rate under shifts to factor A, p_A is the success rate under shifts to factor B, and p_{A+B} is the success rate under shifts to both.

D Implementation and Training Details

In this section, we provide additional details on the implementation and training of all models.

D.1 RT-1

Behavior cloning. We follow the RT-1 architecture that uses tokenized image and language inputs with a categorical cross-entropy objective for tokenized action outputs. The model takes as input a natural language instruction along with the 6 most recent RGB robot observations, and then feeds these through pre-trained language and image encoders (Universal Sentence Encoder [3] and EfficientNet-B3 [35], respectively). These two input modalities are fused with FiLM conditioning, and then passed to a TokenLearner [28] spatial attention module to reduce the number of tokens needed for fast on-robot inference. Then, the network contains 8 decoder only self-attention Transformer layers, followed by a dense action decoding MLP layer. Full details of the RT-1 architecture that we follow can be found in [2].

Data augmentations. Following the image augmentations introduced in Qt-Opt [19], we perform two main types of visual data augmentation during training only: visual disparity augmentations and random cropping. For visual disparity augmentations, we adjust the brightness, contrast, and saturation by sampling uniformly from [-0.125, 0.125], [0.5, 1.5], and [0.5, 1.5] respectively. For random cropping, we subsample the full-resolution camera image to obtain a 300 × 300 random crop. Since RT-1 uses a history length of 6, each timestep is randomly cropped independently.

Pretrained representations. Following the implementation in RT-1, we utilize an EfficientNet-B3 model pretrained on ImageNet [35] for image tokenization, and the Universal Sentence Encoder [3] language encoder for embedding natural language instructions. The rest of the RT-1 model is initialized from scratch.

D.2 Factor World

Behavior cloning. Our behavior cloning policy is parameterized by a convolutional neural network with the same architecture as in [29] and in [37]: there are four convolutional layers with 32, 64, 128, and 128 4×4 filters, respectively. The features are then flattened and passed through a linear layer with output dimension of 128, LayerNorm, and Tanh activation function. The policy head is parameterized as a three-layer feedforward neural network with 256 units per layer. All policies are trained for 100 epochs.

Data augmentations. In our simulated experiments, we experiment with shift augmentations (analogous to the crop augmentations the real robot policy trains with) from [38]: we first pad each side of the 84×84 image by 4 pixels, and then select a random 84×84 crop. We also experiment with color jitter augmentations (analogous to the photometric distortions studied for the real robot policy), which is implemented in torchvision. The brightness, contrast, saturation, and hue factors are set to 0.2. The probability that an image in the batch is augmented is 0.3. All policies are trained for 100 epochs.

Pretrained representations. We use the ResNet50 versions of the publicly available R3M and CLIP representations. We follow the embedding with a BatchNorm, and the same policy head parameterization: three feedforward layers with 256 units per layer. All policies are trained for 100 epochs.

E Additional Experimental Results

In this section, we provide additional results from our simulated and real experiments.

E.1 Continuous Factors

The camera position and table position factors are continuous, unlike the other factors which are discrete, hence the generalization gap with respect to these factors will depend on the range that we train and evaluate on. We aim to understand how much more difficult training and generalizing to a wider range of values is, by studying the gap with the following range radii: 0.025, 0.050, and

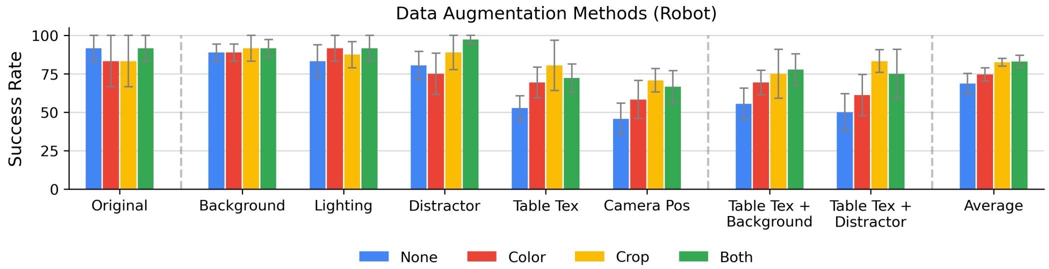


Figure 7: Performance of real-robot policies trained without data augmentation (blue), with random photometric distortions (red), with random crops (yellow), and with both (green). The results discussed in Sec. 4.1 are with "Both". "Original" is the success rate on train environments, "Background" is the success rate when we perturb the background, "Distractors" is where we replace the distractors with new ones, etc. Error bars represent standard error of the mean. We also provide the average over all 7 (sets of) factors on the far right.

0.075 meters. For both camera-position and table-position factors, as we linearly increase the radius, the generalization gap roughly doubles (see Fig. 4b). This pattern suggests: (1) performance can be dramatically improved by keeping the camera and table position as constant as possible, and (2) generalizing to wider ranges may require significantly more diversity, i.e., examples of camera and table positions in the training dataset. However, in Sec. E.2, we see that existing methods can address the latter issue to some degree.

E.2 Augmentations, Pretrained Representations, Architectures

Data augmentation. We study 2 forms of augmentation: (1) random crops and (2) random photometric distortions. The photometric distortion randomly adjusts the brightness, saturation, hue, and contrast of the image, and applies random cutout and random Gaussian noise. Fig. 7 and Fig. 4d show the results for the real robot and *Factor World* respectively. On the robot, **crop augmentation improves generalization along multiple environment factors, most significantly to new camera positions and new table textures**. While the improvement on a spatial factor like camera position is intuitive, we find the improvement on a non-spatial factor like table texture surprising. More in line with our expectations, the photometric distortion augmentation improves the performance on texture-based factors like table texture in the real robot environment and object, table and background in the simulated environment (see the website for *Factor World* results by factor).

Pretrained representations. On the real robot, we evaluate the RT-2 policy [1], which finetunes PaLI-55B on a robot dataset. RT-2 has been shown to generalize better to new objects, instructions, and, most relevant to our work, *environments*. Importantly, the new "environments" that [1] evaluate include a kitchen and desk, which present new objects and workstation heights, among many other factors. Hence, we are interested in evaluating RT-2 along factored environment variations. As shown in Fig. 8, the generalization performance of RT-2 (green) does not improve upon RT-1 (yellow). Interestingly, the success rate of RT-2 on all factors is similar, except on camera positions.

We also study (1) R3M [24] and (2) CLIP [27] in *Factor World*. While these representations are trained on real, non-robotics datasets, policies trained on top of them have been shown to perform well in (simulated and real) robotics environments from a small amount of data. However, while they achieve good performance on training environments, they struggle to generalize to new but similar environments, leaving a large generalization gap across many factors (see Fig. 4d). Though, we find that CLIP does improve upon a trained-from-scratch CNN with new object textures.

Model architectures. In addition to the CNN, we also evaluate policies trained with a ResNet [13] and a Vision Transformer (ViT) [7] encoder. Both encoders succeed under more training environments (see Fig. 4d). However, with fewer train environments, the ViT encoder tends to outperform the CNN variants, while the ResNet encoder performs the worst of the three. We also find a similar ordering of factors across architectures (see the website for results by factor), with one main exception: ResNets generalize to camera positions better relative to other factors.

E.3 Investigating Different Strategies for Data Collection

Augmenting visual diversity with out-of-domain data. As described in Sec. B.1, our real robot dataset includes demonstrations collected from other domains and tasks like opening a fridge and operating a cereal dispenser. Only 35.2% of the 115K demonstrations are collected in the same

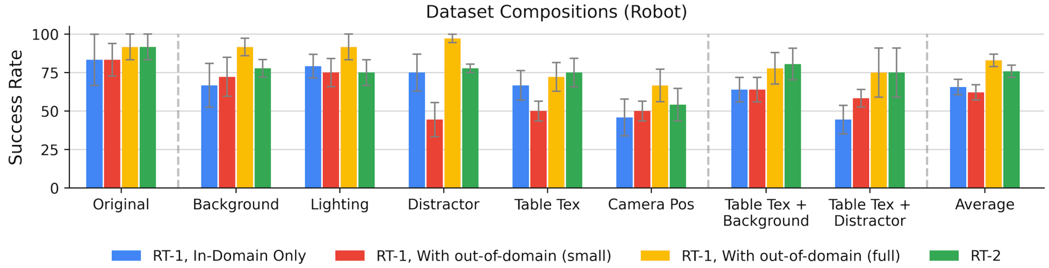


Figure 8: Performance of RT-1 policies trained with in-domain data only (blue), a small version of the in- and out-of-domain dataset (red), and the full version of the in- and out-of-domain dataset (yellow). The RT-2 policy is pretrained and co-finetuned on Internet-scale data (green). Error bars represent standard error of the mean. We also provide the average over all 7 (sets of) factors on the far right.

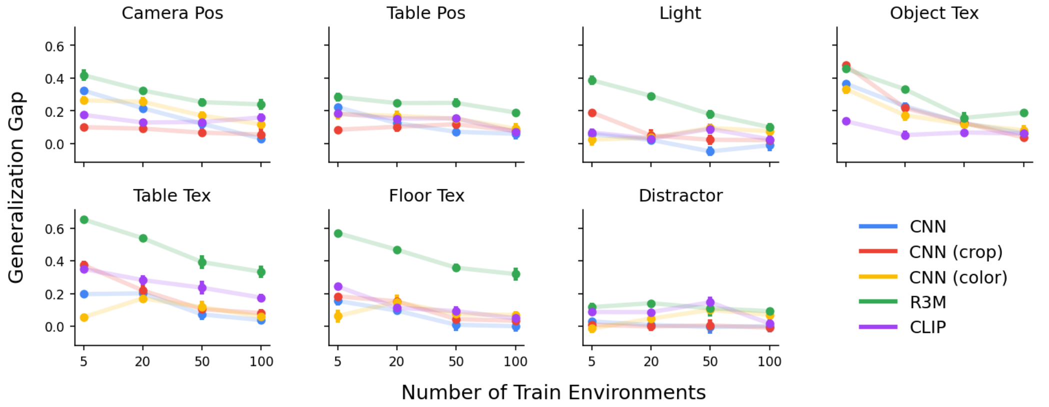


Figure 9: Generalization gap for different data augmentations and pretrained representations in *Factor World*. Subplots share the same x- and y-axes. Results are averaged across the 3 simulated tasks with 5 seeds for each task. Error bars represent standard error of the mean.

domain as our evaluations. While the remaining demonstrations are out of domain and focus on other skills such as drawer manipulation, they add visual diversity, such as new objects and new backgrounds, and demonstrate robotic manipulation behavior, unlike the data that R3M and CLIP pretrain on. We consider the dataset with only in-domain data, which we refer to as In-domain only. In Fig. 8, we compare In-domain only (blue) to the full dataset, which we refer to as With out-of-domain (full) (yellow). While the performance on the original six training tasks is comparable, the success rate of the In-domain only policy drops significantly across the different environment shifts, and the With out-of-domain (full) policy is more successful across the board. **Unlike representations pretrained on non-robotics datasets (Sec. E.2), out-of-domain robotics data can improve in-domain generalization.**

Prioritizing visual diversity with out-of-domain data. We also consider a uniformly subsampled version of the With out-of-domain (full) dataset, which we refer to as With out-of-domain (small). With out-of-domain (small) has the same number of demonstrations as In-domain only, allowing us to directly compare whether the in-domain data or out-of-domain data is more valuable. We emphasize that With out-of-domain (small) has significantly fewer in-domain demonstrations of the "pick" skill than In-domain only. Intuitively, one would expect the in-domain data to be more useful. However, in Fig. 8, we see that the With out-of-domain (small) policy (red) performs comparably with the In-domain only policy (blue) across most of the factors. The main exception is scenarios with new distractors, where the In-domain only policy has a 75.0% success rate while the With out-of-domain (small) policy is successful in 44.4% of the trials. Thus, if a particular application demands good generalization to distractors or table textures over other factors, in-domain data should be prioritized. However, if we only consider the average performance over all factors, collecting out-of-domain data does not harm performance.

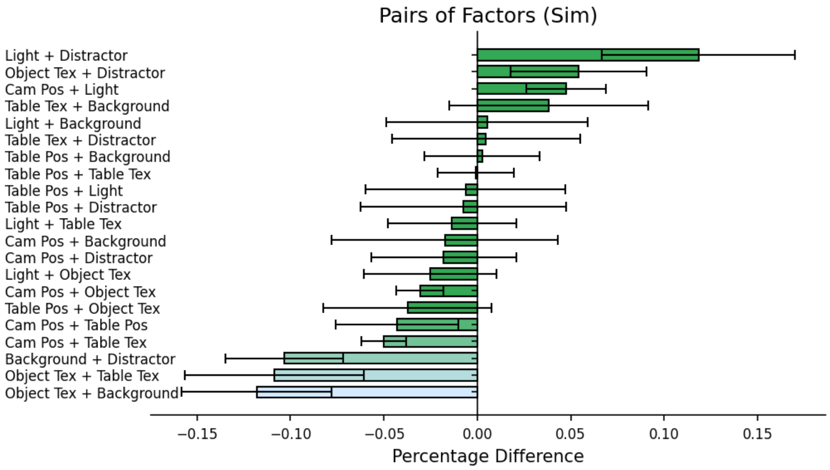


Figure 10: Generalization gap on all pairs of factors, reported as the percentage difference relative to the harder factor of the pair. Results are averaged across the 3 simulated tasks with 5 seeds for each task.

E.4 Simulation: Factor Pairs

In Fig. 10, we report the results for all factor pairs, a partial subset of which was visualized in Fig. 4c. In Fig. 4c, we selected the pairs with the highest magnitude percentage difference, excluding the pairs with error bars that overlap with zero.

E.5 Simulation: Success Rates

In Fig. 11, we report the performance of policies trained with data augmentations and with pretrained representations, in terms of raw success rates. We find that for some policies, the performance on the train environments (see “Original”) degrades as we increase the number of training environments. Nonetheless, as we increase the number of training environments, we see higher success rates on the factor-shifted environments. However, it may be possible to see even more improvements in the success rate with larger-capacity models that fit the training environments better.

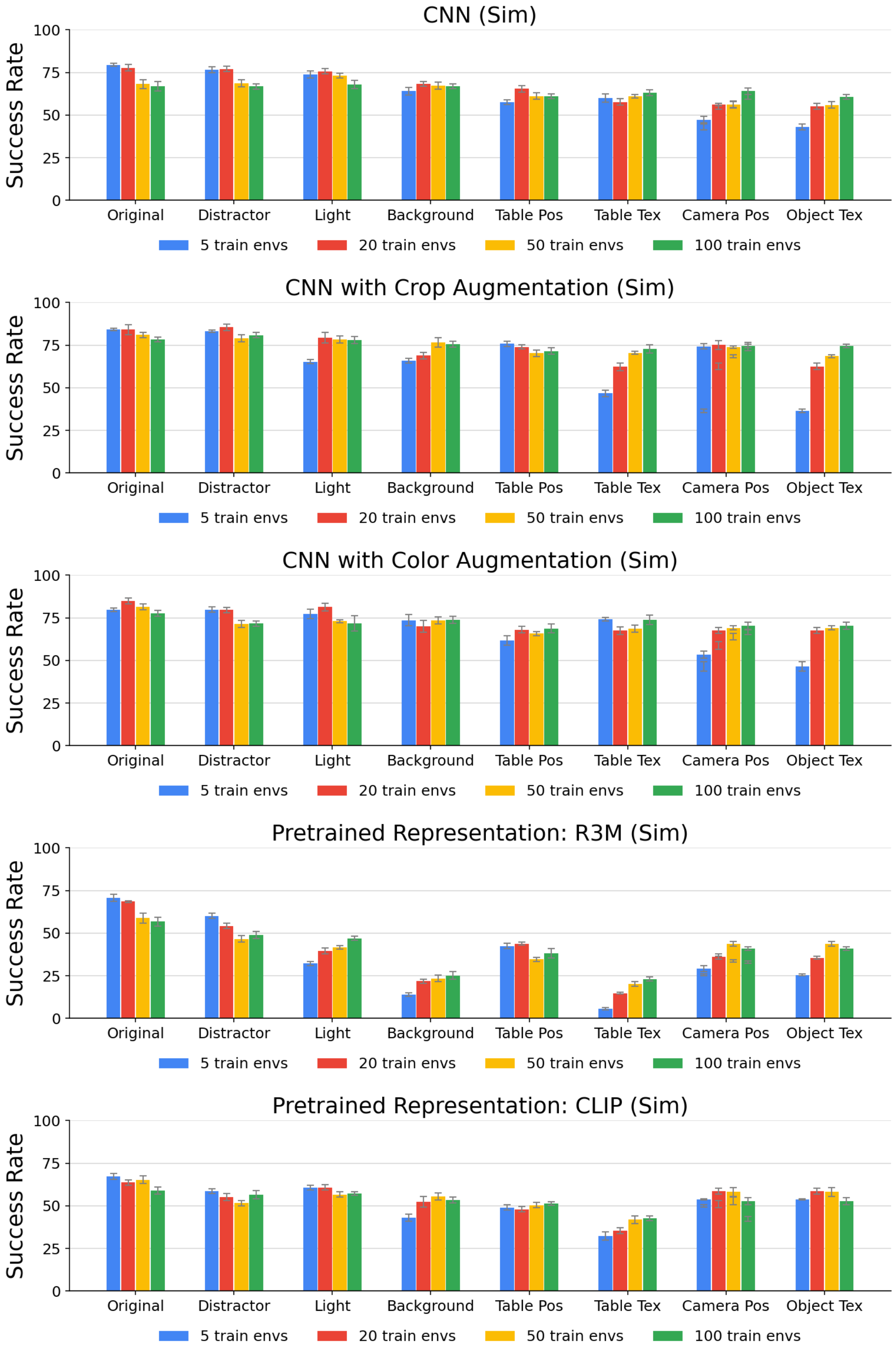


Figure 11: Success rates of simulated policies with data augmentations and with pretrained representations. Results are averaged over the 3 simulated tasks, with 5 seeds run for each task.