KitchenShift: Evaluating Zero-Shot Generalization of Imitation-Based Policy Learning Under Domain Shifts

Eliot Xing [1], Abhinav Gupta [2], Sam Powers*[2], Victoria Dean*[2]
[1] Georgia Institute of Technology, [2] Carnegie Mellon University
`exing@gatech.edu`, `{gabhinav, snpowers, vdean}@andrew.cmu.edu`

Abstract

Humans are remarkably capable of zero-shot generalizing while performing tasks in new settings, even when the task is learned entirely from observing others. In this work, we show that current imitation-based policy learning methods do not share this capability, lacking robustness to minor shifts in the training environment. To demonstrate these limitations of current methods, we propose a testing protocol that new methods may use as a benchmark. We implement and evaluate KitchenShift, an instance of our testing protocol that applies domain shifts to a realistic kitchen environment. We train policies from RGB image observations using a set of demonstrations for a multi-stage robotic manipulation task in the kitchen environment. Using KitchenShift, we evaluate imitation and representation learning methods used in current policy learning approaches and find that they are not robust to visual changes in the scene (e.g., lighting, camera view) or changes in the environment state (e.g., orientation of an object). With our benchmark, we hope to encourage the development of algorithms that can generalize under such domain shifts and overcome the challenges preventing robots from completing tasks in diverse everyday settings.

1 Introduction

The real world is ever-changing, with each moment bringing new, visually rich scenes for an agent to see and interact with. Consider a person going to a neighbor's kitchen for the first time to help cook dinner. Humans have a profound ability to maneuver such never-before-seen environments, transition effortlessly between similar domains, and generalize within narrow task specifications. These abilities lie in stark contrast to the current abilities of robots and policy learning algorithms. How can we narrow this gap? We argue that the development of better algorithms that could be deployed in real-world robotics settings has been bottlenecked by the simulation settings and evaluation schemes that are currently used. Established benchmarks, such as video game tasks from Atari [9] and simple control tasks from OpenAI Gym [10] or DeepMind Control Suite [93], evaluate policies under the same environment settings in which they are trained.

Our evaluation scheme is designed to test robustness to visual and environmental changes using realistic out-of-distribution domain shifts. Specifically, we provide a robotic agent with a set of expert demonstrations and train it to perform a multi-stage manipulation task in a realistic kitchen environment. The policy is then evaluated in an unseen scene, which is created by applying one type of domain shift to the training environment. This situation is designed to mirror a scenario that humans excel at but current algorithms do not. In this work, we make the following contributions:

*Equal contribution

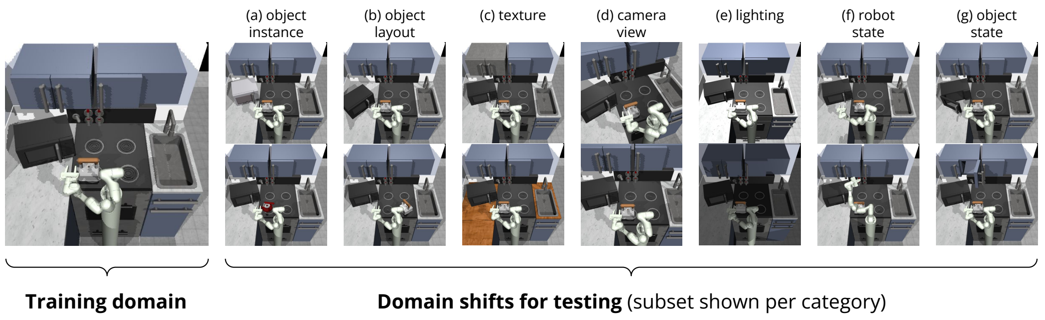

Figure 1: **Visualizing initial observations across KitchenShift domains.** (Left) The training domain for the demonstrations. (Right) Examples of each domain shift type: (a) object instance; (b) object layout; (c) texture; (d) camera view; (e) lighting; (f) robot state; (g) object state.

- We define an evaluation protocol to determine the robustness of policies to minor, out-of-distribution domain shifts to the environment, which a human could adapt to.
- To use this evaluation protocol, we introduce KitchenShift, which adds functionality for applying 7 types of domain shifts to the realistic kitchen simulation from Gupta et al. [33].
- We demonstrate KitchenShift by benchmarking different imitation-based policy learning and representation learning methods, and find that simple behavioral cloning outperforms other approaches evaluated at zero-shot generalizing under domain shifts.

Overall, we observe that current approaches are not robust to realistic out-of-distribution domain shifts. In order to eventually deploy robotic agents into everyday life, we will need approaches that are robust and generalize to out-of-distribution settings. We release KitchenShift to encourage the development of future algorithms that can overcome such domain shifts.

2 Evaluating Policies under Domain Shifts with KitchenShift

In this section, we describe KitchenShift, the implementation of our evaluation paradigm. We divide the section into high-level protocol (Section 2.1), domain shifts (Section 2.2), and policy evaluation (Section 2.3). In Appendix B, we discuss the choice of the kitchen environment we use.

2.1 Evaluation protocol

"Domain" and "distribution" shifts are frequently used interchangeably. In this paper, we use "domain shift" to refer to a change in the environment, world, or domain $\mathcal{D}$. We do not take "domain shift" to include changes to the distribution or set of tasks that the policy learns.

For our evaluation protocol, we train the policy on a task $\mathcal{T}_{train}$ in a new training domain $\mathcal{D}_{train}$, with access to a limited set of demonstrations. These demonstrations reduce the need for exploration, improving sample efficiency and lowering compute requirements. We then evaluate how the policy performs the training task $\mathcal{T}_{train}$ in an unseen set of domains $\{\mathcal{D}_{shifted}\}$ by modifying the evaluation environment. If the policy has learned general behavior, then it should be able to adapt to the new testing domain and perform the task under such domain shifts. We describe and provide a set of domain shifts that can be applied individually to independent parts of the environment, which an average human would be able to overcome and still perform the training task successfully.

Though we focus on a single environment in this work, we note that this setup can also be used to evaluate policy adaptation and transfer to new domains. For instance, a policy could be pretrained on a set of tasks in RoboSuite [116] then finetuned and tested on KitchenShift, our implementation of the evaluation protocol we propose, which we proceed to describe.

2.2 Domain shifts in KitchenShift

Building off the kitchen environment released by Gupta et al. [33], we make changes to the simulation settings (see Section E) and add functionality to modify the environment, which we refer to as KitchenShift. The domain shifts that we evaluate are not only limited to visual changes such as background context [36, 88], colors, lighting, textures, or camera pose [41, 113, 37, 25]. In Figure 1

(Right), we visualize the types of domain shifts available in KitchenShift that can be applied to create different testing scenes. We categorize seven types of domain shifts, based on changing (a) object instance; (b) object layout; (c) texture; (d) camera view; (e) lighting; (f) robot state; (g) object state. For (a) object instance, we change the microwave and kettle assets [2]. For (b) object layout, we change the position and orientation of different objects. For (f) robot state, we change the initial joint positions of the robot manipulator. For (g) object state, one object in the scene is changed to its final goal state. The training scene, which is also the domain used by the demonstrations, is shown in Figure 1 (Left).

Our goal is to isolate and analyze differences in the performance of existing algorithms, to inform where these methods may be improved. To create testing environments that are explicitly outside of the training distribution, we apply a single type of domain shift to the training environment. If a human was asked to perform the task in the training domain, they would subsequently be able to adapt and succeed in such shifted environments as well. Humans are able to learn from a narrow task specification and training domain, while easily adapting to shifts in the scene. We do not apply domain randomization by training the policy on the set of domain shifts encountered during evaluation. Our purpose is to evaluate policy robustness when encountering unseen, out-of-distribution scenes.

2.3 Policy evaluation

The training task that the policy is asked to learn is multi-stage and involves interacting with a set of at most four objects in the environment. In Appendix C, Figure 4, we visualize frames selected from an expert demonstration for the training task that involves manipulating the [`microwave,kettle,switch,slide`] objects in the scene to the final goal state.

We evaluate policy performance in the training domain and average performance across testing domains within each category of domain shift. We also average across categories of domain shifts to report the overall performance across all testing domains, as a measure of the robustness and generalization of the learned policy. We exclude the (g) object state category from this measure because the object changed may be involved in the task and would improve the average step completion reported, inflating policy performance. For instance, one of these domain shifts opens the microwave door to the desired goal state (see Figure 1 (Left)), allowing a robust policy to skip interacting with the microwave. The (g) object state category is still evaluated separately in Section 3, Figure 2.

3 Experimental Results

KitchenShift is available at `github.com/etaoxing/kitchen-shift`. We provide details on the methods we evaluated and experimental details in Appendix C and Appendix D, respectively. In Appendix E, we discuss and validate the random perturbations we apply when initializing the simulation environment, in contrast to the original environment code.

We compare representation learning and imitation learning methods based on behavioral cloning, in the training domain and in unseen testing environments with specific types of domain shifts applied. Performance of policies trained with different methods is reported in Table 1, and results show that learning-based methods are less successful when evaluated under domain shifts. Evaluating policies with KitchenShift underscores the pitfalls of learning-based policies when testing generalization to out-of-distribution settings, including even minor deviations from the training domain.

To further interpret these results, we show policy performance separated by each category of domain shift in Figure 2. Intuitively, Demo playback is not affected at all by (c) texture, (d) camera view, or (e) lighting, as these are visual-only changes. On the other hand, Demo playback cannot account for changes to the initial robot state of the trajectory or environment, while learning-based methods show some robustness to this. We find that BC (MSE) and BC (β-VAE) drastically overfit to the training distribution and fail to generalize out-of-distribution under minor domain shifts. In comparison, BC which parameterizes a policy as a logistic mixture distribution, generalizes better to unseen scenarios. We also find that current representation learning approaches are not helpful in this imitation-based policy learning setting, which corresponds with recent work by Chen et al. [14].

Note that the (g) object state category cannot be directly compared to the other types of domain shifts, as it sets one object in the scene, that may be involved in the task, to the desired goal state. The

[2]We use assets released by `github.com/vikashplus/furniture_sim`.

Table 1: **Evaluating policy robustness to domain shifts.** Results show policy performance in the training domain and averaged across all testing domains. We report mean±stddev of the average step completion (out of four total). Results are averaged over 20 evaluation episodes and 4 training seeds.

Method	(*training* domain) Avg. steps completed	(across *testing* domains) Avg. steps completed
BC	2.03 ± 1.12	1.23 ± 1.02
BC (MSE)	1.29 ± 1.33	0.36 ± 0.60
BC w/ inv. dynamics	1.62 ± 1.27	1.15 ± 1.04
GCBC (HER relabeling)	1.97 ± 1.12	1.20 ± 1.03
GCBC (HER relabeling) w/ multi-task demos	1.50 ± 1.00	0.78 ± 0.76
BC (β-VAE)	1.23 ± 1.21	0.50 ± 0.70
BC (β-VAE, no stop grad)	0.45 ± 0.70	0.25 ± 0.41
BC (σ-VAE)	1.17 ± 1.17	0.63 ± 0.79
BC (RAE)	1.61 ± 1.16	1.09 ± 0.92
BC (SimSiam, full aug.)	1.27 ± 1.09	0.95 ± 0.87
BC (ATC, random shift)	1.44 ± 1.07	1.14 ± 0.97
BC (IMPALA network)	1.95 ± 1.35	1.15 ± 0.97
BC (fixed random embedding)	1.20 ± 1.03	0.97 ± 0.90
Random actions	0.07 ± 0.27	0.08 ± 0.26
Demo playback	1.26 ± 1.07	0.80 ± 0.96

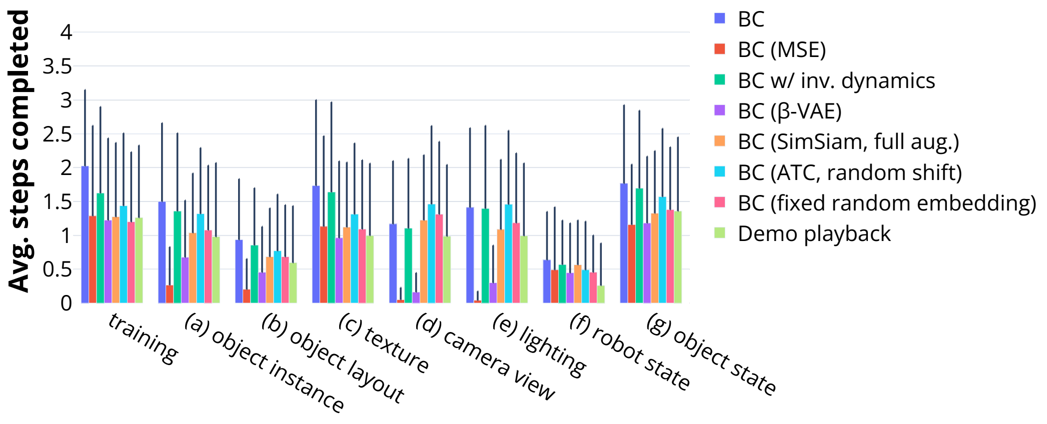

Figure 2: **Evaluating on domain shifts separated by category.** Results show performance in the training domain and testing domains, categorized by the type of domain shifts. We report on the same experimental results as in Table 1, with the bars+lines showing mean+std.

categories that shift (a) object instance; (b) object layout; (f) robot state; and (g) object state, which are not just visual changes but modify the underlying scene, pose the most interesting challenges. One approach to overcome these challenges for future work may be to replace implicitly learned representations with learned object-centric representations [21, 57].

4 Related Work

We cover prior work on benchmarking policy learning in Appendix A.

Addressing domain shift Current approaches for dealing with domain shifts, such as the gap between simulation and the real world, involve system identification to tune simulator parameters [22] or by applying large amounts of domain randomization [95, 2, 5]. While such approaches may work for constrained lab settings or in the data-rich area of computer vision, they require large engineering effort to deploy for robot learning. Collecting large amounts of real-world data with robots is also time and capital intensive [56, 19]. To deploy robots that can learn and adapt directly in the real world, we need drastic improvements in the robustness of learning approaches to out-of-distribution changes. Distribution shift [90, 48, 32, 101] has been studied before in the context of supervised learning such as image classification, but domain shift for policy learning is challenging and under-explored. To the best of our knowledge, single domain generalization [98, 114, 74, 100] has not been studied before in the context of policy learning.

Several works have evaluated visual representation learning [52, 36, 89] and data augmentation [53, 75, 109, 108, 37] alongside policy learning to improve generalization, however these methods train with visual inputs on simple control tasks [93] or Procgen [17], a video game environment. In this work, we show that policies trained with these representation learning approaches still suffer from domain shifts in a more challenging and realistic kitchen environment.

5 Discussion

We hope KitchenShift, with the evaluation protocols we developed and tested, will cast light on the fragility of imitation-based policy learning approaches. Our results show that current methods are not robust under minor visual and environmental domain shifts to which humans could effectively adapt. We also evaluate several representation learning approaches and find that these methods are still unable to overcome the challenge of generalizing with realistic visual scenes in the limited data domain. To move toward robots that can learn directly in the real world and adapt to new, unseen situations, the robotics community will need methods that tackle out-of-distribution domain shifts head-on.

Future directions Future work may evaluate transfer and generalization of policies by pretraining with different simulators, followed by training and evaluation using KitchenShift. This is an exciting direction, as transfer learning has had significant success in computer vision, but how to transfer policies between environments or different robot morphologies remains an open question. On a related note, test-time training [91, 7], which adapts models online rather than fixing parameters for evaluation, presents an interesting path forward. Ideally, learning-based methods deployed in the real world are able to continuously learn and adapt to new settings.

Relaxing some of the available data assumptions, such as imitation directly from a demonstration video without access to the trajectory actions, also poses an interesting line of work. This could involve imitation learning by watching YouTube videos of humans, for which there will be more challenging domain shifts between the real-world videos and the kitchen simulation.

Limitations We evaluate representation learning methods primarily alongside imitation learning, due to time and compute constraints. Such methods may perform differently when trained online with reinforcement learning. Still, we do not expect changes to the final results, as these methods should have an easier time learning in the simpler, stationary imitation setup. We also only test our evaluation protocol using one kitchen environment, but we expect our results would transfer to other simulations, as the underlying policy learning methods would still suffer from similar challenges.

Broader impact In this paper, we evaluate policy robustness to minor, out-of-distribution domain shifts to which a human could easily adapt. Understanding these limitations will aid in the development of new algorithms that can deploy on robots in the real world. Understanding where methods fail is also important in determining the expected behavior of policies for robot safety [3, 30]. On the other hand, improving policy learning algorithms towards human-like generalization capabilities may lead to large-scale automation, disrupting the workforce across many sectors [12]. For the moment, our work is far from these impacts, and we hope that the positive implications of this line of research will come to fruition in parallel with regulation and caution, to mitigate negative consequences.

References

[1] O. Ahmed, F. Träuble, A. Goyal, A. Neitz, M. Wuthrich, Y. Bengio, B. Schölkopf, and S. Bauer, "Causalworld: A robotic manipulation benchmark for causal structure and transfer learning," in *International Conference on Learning Representations*, 2020.

[2] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas *et al.*, "Solving rubik's cube with a robot hand," *arXiv preprint arXiv:1910.07113*, 2019.

[3] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané, "Concrete problems in ai safety," *arXiv preprint arXiv:1606.06565*, 2016.

[4] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba, "Hindsight experience replay," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 5055–5065.

[5] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray *et al.*, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020.

[6] A. S. Azad, E. Kim, Q. Wu, K. Lee, I. Stoica, P. Abbeel, and S. A. Seshia, "Scenic4rl: Programmatic modeling and generation of reinforcement learning environments," *arXiv preprint arXiv:2106.10365*, 2021.

[7] A. Bartler, A. Bühler, F. Wiewel, M. Döbler, and B. Yang, "Mt3: Meta test-time training for self-supervised test-time adaption," *arXiv preprint arXiv:2103.16201*, 2021.

[8] D. Batra, A. X. Chang, S. Chernova, A. J. Davison, J. Deng, V. Koltun, S. Levine, J. Malik, I. Mordatch, R. Mottaghi *et al.*, "Rearrangement: A challenge for embodied ai," *arXiv preprint arXiv:2011.01975*, 2020.

[9] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *Journal of Artificial Intelligence Research*, vol. 47, pp. 253–279, 2013.

[10] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[11] S. Brodeur, E. Perez, A. Anand, F. Golemo, L. Celotti, F. Strub, J. Rouat, H. Larochelle, and A. Courville, "Home: A household multimodal environment," *arXiv preprint arXiv:1711.11017*, 2017.

[12] E. Brynjolfsson and T. Mitchell, "What can machine learning do? workforce implications," *Science*, vol. 358, no. 6370, 2017.

[13] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607.

[14] X. Chen, S. Toyer, C. Wild, S. Emmons, I. Fischer, K.-H. Lee, N. Alex, S. H. Wang, P. Luo, S. Russell, P. Abbeel, and R. Shah, "An empirical investigation of representation learning for imitation," *preprint*, 2021.

[15] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 750–15 758.

[16] M. Chevalier-Boisvert, L. Willems, and S. Pal, "Minimalistic gridworld environment for openai gym," https://github.com/maximecb/gym-minigrid, 2018.

[17] K. Cobbe, C. Hesse, J. Hilton, and J. Schulman, "Leveraging procedural generation to benchmark reinforcement learning," in *International conference on machine learning*. PMLR, 2020, pp. 2048–2056.

[18] S. Dasari and A. Gupta, "Transformers for one-shot imitation learning," in *CoRL 2020*, 2020.

[19] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn, "Robonet: Large-scale multi-robot learning," in *Conference on Robot Learning*. PMLR, 2020, pp. 885–897.

[20] M. Dehghani, Y. Tay, A. A. Gritsenko, Z. Zhao, N. Houlsby, F. Diaz, D. Metzler, and O. Vinyals, "The benchmark lottery," *arXiv preprint arXiv:2107.07002*, 2021.

[21] C. Devin, P. Abbeel, T. Darrell, and S. Levine, "Deep object-centric representations for generalizable robot learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7111–7118.

[22] Y. Du, O. Watkins, T. Darrell, P. Abbeel, and D. Pathak, "Auto-tuned sim-to-real transfer," *arXiv preprint arXiv:2104.07662*, 2021.

[23] K. Ehsani, W. Han, A. Herrasti, E. VanderBilt, L. Weihs, E. Kolve, A. Kembhavi, and R. Mottaghi, "Manipulathor: A framework for visual object manipulation," in *CVPR*, 2021.

[24] L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning *et al.*, "Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures," in *International Conference on Machine Learning*. PMLR, 2018, pp. 1407–1416.

[25] L. Fan, G. Wang, D.-A. Huang, Z. Yu, L. Fei-Fei, Y. Zhu, and A. Anandkumar, "Secant: Self-expert cloning for zero-shot generalization of visual policies," *arXiv preprint arXiv:2106.09678*, 2021.

[26] P. Florence, C. Lynch, A. Zeng, O. A. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson, "Implicit behavioral cloning," in *5th Annual Conference on Robot Learning*, 2021.

[27] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4rl: Datasets for deep data-driven reinforcement learning," *arXiv preprint arXiv:2004.07219*, 2020.

[28] C. Gan, J. Schwartz, S. Alter, D. Mrowca, M. Schrimpf, J. Traer, J. D. Freitas, J. Kubilius, A. Bhandwaldar, N. Haber, M. Sano, K. Kim, E. Wang, M. Lingelbach, A. Curtis, K. T. Feigelis, D. Bear, D. Gutfreund, D. D. Cox, A. Torralba, J. J. DiCarlo, J. B. Tenenbaum, J. Mcdermott, and D. L. Yamins, "ThreeDWorld: A platform for interactive multi-modal physical simulation," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[29] X. Gao, R. Gong, T. Shu, X. Xie, S. Wang, and S.-C. Zhu, "Vrkitchen: an interactive 3d virtual environment for task-oriented learning," *arXiv preprint arXiv:1903.05757*, 2019.

[30] J. Garcıa and F. Fernández, "A comprehensive survey on safe reinforcement learning," *Journal of Machine Learning Research*, vol. 16, no. 1, 2015.

[31] P. Ghosh, M. S. M. Sajjadi, A. Vergari, M. Black, and B. Scholkopf, "From variational to deterministic autoencoders," in *International Conference on Learning Representations*, 2020.

[32] I. Gulrajani and D. Lopez-Paz, "In search of lost domain generalization," in *International Conference on Learning Representations*, 2021.

[33] A. Gupta, V. Kumar, C. Lynch, S. Levine, and K. Hausman, "Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning," in *Conference on Robot Learning*. PMLR, 2020, pp. 1025–1037.

[34] W. H. Guss, C. Codel, K. Hofmann, B. Houghton, N. Kuno, S. Milani, S. Mohanty, D. P. Liebana, R. Salakhutdinov, N. Topin *et al.*, "The minerl 2019 competition on sample efficient reinforcement learning using human priors," *arXiv preprint arXiv:1904.10079*, 2019.

[35] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 297–304.

[36] N. Hansen and X. Wang, "Generalization in reinforcement learning by soft data augmentation," *arXiv preprint arXiv:2011.13389*, 2020.

[37] N. Hansen, H. Su, and X. Wang, "Stabilizing deep q-learning with convnets and vision transformers under data augmentation," *arXiv preprint arXiv:2107.00644*, 2021.

[38] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9729–9738.

[39] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[40] I. Higgins, L. Matthey, A. Pal, C. P. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," in *ICLR*, 2017.

[41] I. Higgins, A. Pal, A. Rusu, L. Matthey, C. Burgess, A. Pritzel, M. Botvinick, C. Blundell, and A. Lerchner, "Darla: Improving zero-shot transfer in reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1480–1490.

[42] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison, "Rlbench: The robot learning benchmark & learning environment," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3019–3026, 2020.

[43] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn, "BC-0: Zero-shot task generalization with robotic imitation learning," in *5th Annual Conference on Robot Learning*, 2021.

[44] A. Juliani, A. Khalifa, V.-P. Berges, J. Harper, E. Teng, H. Henry, A. Crespi, J. Togelius, and D. Lange, "Obstacle tower: A generalization challenge in vision, control, and planning," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2019.

[45] H. Kannan, D. Hafner, C. Finn, and D. Erhan, "Robodesk: A multi-task reinforcement learning benchmark," https://github.com/google-research/robodesk, 2021.

[46] M. Kempka, M. Wydmuch, G. Runc, J. Toczek, and W. Jaśkowski, "Vizdoom: A doom-based ai research platform for visual reinforcement learning," in *2016 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, 2016, pp. 1–8.

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

[48] P. W. Koh, S. Sagawa, S. M. Xie, M. Zhang, A. Balsubramani, W. Hu, M. Yasunaga, R. L. Phillips, I. Gao, T. Lee *et al.*, "Wilds: A benchmark of in-the-wild distribution shifts," in *International Conference on Machine Learning*. PMLR, 2021, pp. 5637–5664.

[49] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, D. Gordon, Y. Zhu, A. Gupta, and A. Farhadi, "AI2-THOR: An Interactive 3D Environment for Visual AI," *arXiv*, 2017.

[50] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *arXiv preprint arXiv:2006.04779*, 2020.

[51] H. Küttler, N. Nardelli, A. H. Miller, R. Raileanu, M. Selvatici, E. Grefenstette, and T. Rocktäschel, "The NetHack Learning Environment," in *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[52] M. Laskin, A. Srinivas, and P. Abbeel, "Curl: Contrastive unsupervised representations for reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 5639–5650.

[53] M. Laskin, K. Lee, A. Stooke, L. Pinto, P. Abbeel, and A. Srinivas, "Reinforcement learning with augmented data," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[54] K. Lee, Y. Seo, S. Lee, H. Lee, and J. Shin, "Context-aware dynamics model for generalization in model-based reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 5757–5766.

[55] Y. Lee, E. S. Hu, and J. J. Lim, "IKEA furniture assembly environment for long-horizon complex manipulation tasks," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021. [Online]. Available: https://clvrai.com/furniture

[56] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *The International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 421–436, 2018.

[57] F. Locatello, D. Weissenborn, T. Unterthiner, A. Mahendran, G. Heigold, J. Uszkoreit, A. Dosovitskiy, and T. Kipf, "Object-centric learning with slot attention," *arXiv preprint arXiv:2006.15055*, 2020.

[58] Y. Lu, Y. Shen, S. Zhou, A. Courville, J. B. Tenenbaum, and C. Gan, "Learning task decomposition with ordered memory policy network," in *International Conference on Learning Representations*, 2021.

[59] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet, "Learning latent plans from play," in *Conference on Robot Learning*. PMLR, 2020, pp. 1113–1132.

[60] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, "Isaac gym: High performance gpu-based physics simulation for robot learning," 2021.

[61] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," in *5th Annual Conference on Robot Learning*, 2021.

[62] H. Mania, A. Guy, and B. Recht, "Simple random search of static linear policies is competitive for reinforcement learning," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018, pp. 1805–1814.

[63] T. Mu, Z. Ling, F. Xiang, D. Yang, X. Li, S. Tao, Z. Huang, Z. Jia, and H. Su, "Maniskill: Learning-from-demonstrations benchmark for generalizable manipulation skills," *arXiv preprint arXiv:2107.14483*, 2021.

[64] A. Nair, S. Bahl, A. Khazatsky, V. Pong, G. Berseth, and S. Levine, "Contextual imagined goals for self-supervised robotic learning," in *Conference on Robot Learning*. PMLR, 2020, pp. 530–539.

[65] A. V. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine, "Visual reinforcement learning with imagined goals," *Advances in Neural Information Processing Systems*, vol. 31, pp. 9191–9200, 2018.

[66] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[67] O. OpenAI, M. Plappert, R. Sampedro, T. Xu, I. Akkaya, V. Kosaraju, P. Welinder, R. D'Sa, A. Petron, H. P. d. O. Pinto *et al.*, "Asymmetric self-play for automatic goal discovery in robotic manipulation," *arXiv preprint arXiv:2101.04882*, 2021.

[68] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, J. Peters *et al.*, "An algorithmic perspective on imitation learning," *Foundations and Trends in Robotics*, vol. 7, no. 1-2, pp. 1–179, 2018.

[69] K. Pertsch, Y. Lee, and J. J. Lim, "Accelerating reinforcement learning with learned skill priors," *arXiv preprint arXiv:2010.11944*, 2020.

[70] K. Pertsch, Y. Lee, Y. Wu, and J. J. Lim, "Demonstration-guided reinforcement learning with learned skills," in *Self-Supervision for Reinforcement Learning Workshop - ICLR 2021*, 2021.

[71] M. Plappert, M. Andrychowicz, A. Ray, B. McGrew, B. Baker, G. Powell, J. Schneider, J. Tobin, M. Chociej, P. Welinder *et al.*, "Multi-goal reinforcement learning: Challenging robotics environments and request for research," *arXiv preprint arXiv:1802.09464*, 2018.

[72] V. Pong, M. Dalal, S. Lin, A. Nair, S. Bahl, and S. Levine, "Skew-fit: State-covering self-supervised reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 7783–7792.

[73] X. Puig, K. Ra, M. Boben, J. Li, T. Wang, S. Fidler, and A. Torralba, "Virtualhome: Simulating household activities via programs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8494–8502.

[74] F. Qiao, L. Zhao, and X. Peng, "Learning to learn single domain generalization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 556–12 565.

[75] R. Raileanu, M. Goldstein, D. Yarats, I. Kostrikov, and R. Fergus, "Automatic data augmentation for generalization in deep reinforcement learning," *arXiv preprint arXiv:2006.12862*, 2020.

[76] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations," in *Proceedings of Robotics: Science and Systems (RSS)*, 2018.

[77] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," in *International conference on machine learning*. PMLR, 2014, pp. 1278–1286.

[78] O. Rybkin, K. Daniilidis, and S. Levine, "Simple and effective vae training with calibrated decoders," in *International Conference on Machine Learning*. PMLR, 2021, pp. 9179–9189.

[79] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma, "Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications," *arXiv preprint arXiv:1701.05517*, 2017.

[80] M. Samvelyan, R. Kirk, V. Kurin, J. Parker-Holder, M. Jiang, E. Hambro, F. Petroni, H. Kuttler, E. Grefenstette, and T. Rocktäschel, "Minihack the planet: A sandbox for open-ended reinforcement learning research," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[81] M. Savva, A. X. Chang, A. Dosovitskiy, T. Funkhouser, and V. Koltun, “Minos: Multimodal indoor simulator for navigation in complex environments,” *arXiv preprint arXiv:1712.03931*, 2017.

[82] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik *et al.*, “Habitat: A platform for embodied ai research,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9339–9347.

[83] D. Seita, P. Florence, J. Tompson, E. Coumans, V. Sindhwani, K. Goldberg, and A. Zeng, “Learning to Rearrange Deformable Cables, Fabrics, and Bags with Goal-Conditioned Transporter Networks,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[84] R. Sekar, O. Rybkin, K. Daniilidis, P. Abbeel, D. Hafner, and D. Pathak, “Planning to explore via self-supervised world models,” in *International Conference on Machine Learning*. PMLR, 2020, pp. 8583–8592.

[85] Y. Seo, L. Chen, J. Shin, H. Lee, P. Abbeel, and K. Lee, “State entropy maximization with random encoders for efficient exploration,” *arXiv preprint arXiv:2102.09430*, 2021.

[86] B. Shen, F. Xia, C. Li, R. Martín-Martín, L. Fan, G. Wang, S. Buch, C. D’Arpino, S. Srivastava, L. P. Tchapmi *et al.*, “igibson, a simulation environment for interactive tasks in large realisticscenes,” *arXiv preprint arXiv:2012.02924*, 2020.

[87] M. Shridhar, J. Thomason, D. Gordon, Y. Bisk, W. Han, R. Mottaghi, L. Zettlemoyer, and D. Fox, “Alfred: A benchmark for interpreting grounded instructions for everyday tasks,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 10 740–10 749.

[88] A. Stone, O. Ramirez, K. Konolige, and R. Jonschkowski, “The distracting control suite–a challenging benchmark for reinforcement learning from pixels,” *arXiv preprint arXiv:2101.02722*, 2021.

[89] A. Stooke, K. Lee, P. Abbeel, and M. Laskin, “Decoupling representation learning from reinforcement learning,” in *International Conference on Machine Learning*. PMLR, 2021, pp. 9870–9879.

[90] B. Sun, J. Feng, and K. Saenko, “Return of frustratingly easy domain adaptation,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.

[91] Y. Sun, X. Wang, Z. Liu, J. Miller, A. Efros, and M. Hardt, “Test-time training with self-supervision for generalization under distribution shifts,” in *International Conference on Machine Learning*. PMLR, 2020, pp. 9229–9248.

[92] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. Chaplot, O. Maksymets *et al.*, “Habitat 2.0: Training home assistants to rearrange their habitat,” *arXiv preprint arXiv:2106.14405*, 2021.

[93] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. d. L. Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq *et al.*, “Deepmind control suite,” *arXiv preprint arXiv:1801.00690*, 2018.

[94] Y. Tassa, S. Tunyasuvunakool, A. Muldal, Y. Doron, S. Liu, S. Bohez, J. Merel, T. Erez, T. Lillicrap, and N. Heess, “dm_control: Software and tasks for continuous control,” 2020.

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

[96] S. Toyer, R. Shah, A. Critch, and S. Russell, “The magical benchmark for robust imitation,” *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[97] O. Vinyals, T. Ewalds, S. Bartunov, P. Georgiev, A. S. Vezhnevets, M. Yeo, A. Makhzani, H. Küttler, J. Agapiou, J. Schrittwieser *et al.*, “Starcraft ii: A new challenge for reinforcement learning,” *arXiv preprint arXiv:1708.04782*, 2017.

[98] R. Volpi, H. Namkoong, O. Sener, J. Duchi, V. Murino, and S. Savarese, “Generalizing to unseen domains via adversarial data augmentation,” in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018, pp. 5339–5349.

[99] J. Wang, Y. Lu, and H. Zhao, “Cloud: Contrastive learning of unsupervised dynamics,” *arXiv preprint arXiv:2010.12488*, 2020.

[100] Z. Wang, Y. Luo, R. Qiu, Z. Huang, and M. Baktashmotlagh, “Learning to diversify for single domain generalization,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 834–843.

[101] O. Wiles, S. Gowal, F. Stimberg, S. Alvise-Rebuffi, I. Ktena, T. Cemgil *et al.*, "A fine-grained analysis on distribution shift," *arXiv preprint arXiv:2110.11328*, 2021.

[102] J. Wong, A. Tung, A. Kurenkov, A. Mandlekar, L. Fei-Fei, S. Savarese, and R. Martín-Martín, "Error-aware imitation learning from teleoperation data for mobile manipulation," in *5th Annual Conference on Robot Learning*, 2021.

[103] Y. Wu, Y. Wu, G. Gkioxari, and Y. Tian, "Building generalizable agents with a realistic and rich 3d environment," *arXiv preprint arXiv:1801.02209*, 2018.

[104] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang *et al.*, "Sapien: A simulated part-based interactive environment," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 097–11 107.

[105] C. Yan, D. Misra, A. Bennnett, A. Walsman, Y. Bisk, and Y. Artzi, "Chalet: Cornell house agent learning environment," *arXiv preprint arXiv:1801.07357*, 2018.

[106] W. Yan, A. Vangipuram, P. Abbeel, and L. Pinto, "Learning predictive representations for deformable objects using contrastive estimation," *arXiv preprint arXiv:2003.05436*, 2020.

[107] C. Yang, X. Ma, W. Huang, F. Sun, H. Liu, J. Huang, and C. Gan, "Imitation learning from observations by minimizing inverse dynamics disagreement," in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019, pp. 239–249.

[108] D. Yarats, R. Fergus, A. Lazaric, and L. Pinto, "Mastering visual continuous control: Improved data-augmented reinforcement learning," *arXiv preprint arXiv:2107.09645*, 2021.

[109] D. Yarats, I. Kostrikov, and R. Fergus, "Image augmentation is all you need: Regularizing deep reinforcement learning from pixels," in *International Conference on Learning Representations*, 2021.

[110] D. Yarats, A. Zhang, I. Kostrikov, B. Amos, J. Pineau, and R. Fergus, "Improving sample efficiency in model-free reinforcement learning from images," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 12, 2021, pp. 10 674–10 681.

[111] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *Conference on Robot Learning*. PMLR, 2020, pp. 1094–1100.

[112] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani, and J. Lee, "Transporter networks: Rearranging the visual world for robotic manipulation," *Conference on Robot Learning (CoRL)*, 2020.

[113] A. Zhang, Y. Wu, and J. Pineau, "Natural environment benchmarks for reinforcement learning," *arXiv preprint arXiv:1811.06032*, 2018.

[114] L. Zhao, T. Liu, X. Peng, and D. Metaxas, "Maximum-entropy adversarial data augmentation for improved generalization and robustness," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[115] H. Zhu, J. Yu, A. Gupta, D. Shah, K. Hartikainen, A. Singh, V. Kumar, and S. Levine, "The ingredients of real world robotic reinforcement learning," in *International Conference on Learning Representations*, 2020.

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

Checklist

1. For all authors...
    (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
    (b) Did you describe the limitations of your work? [Yes] Appendix 5, Appendix B.
    (c) Did you discuss any potential negative societal impacts of your work? [Yes] Appendix 5.
    (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
    (a) Did you state the full set of assumptions of all theoretical results? [N/A]
    (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
    (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Appendix C contains a code repository link.
    (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? See Appendix C for an overview and Appendix D for additional hyperparameter details.
    (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] All tables and plots show standard deviation either written numerically or as error bars.
    (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Appendix D.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
    (a) If your work uses existing assets, did you cite the creators? [Yes]
    (b) Did you mention the license of the assets? [Yes] We provide links or references that point to the licenses of the assets used.
    (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
    (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
    (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
    (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
    (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
    (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Extended Related Work

Benchmarking policy learning Random search algorithms have been shown to solve [62] the established benchmarks such as low-dimensional control tasks from the OpenAI Gym [10] or DeepMind Control Suite [93, 94]. Furthermore, RL algorithms which have frequently been developed and evaluated on these tasks are known to be brittle, sample-inefficient, and difficult to apply in the real world. Video game-like environments [46, 97, 16, 34, 80] such as Atari [9] have also been used to evaluate policy learning, but these environments do not have realistic graphical assets and do not simulate real-world physics. While procedurally-generated game environments [44, 51, 17, 6] have been proposed to evaluate policies in unseen testing environments, the test environments are usually in-distribution to the train settings and difficult to apply realistic instances of domain shifts in.

Many environments have also been introduced that involve robots for object manipulation using a manipulator arm [71, 33, 111, 42, 116, 59, 45, 67, 55] or dexterous hand [76, 5], as well as mobile agents in more complex settings such as home environments [11, 105, 81, 73, 103, 29, 49, 87, 23, 86, 104, 28, 82, 92]. We note that planar manipulation tasks of rigid [112] and even deformable [83] objects, where the robot is given a top-down view of a tabletop, can be nearly or completely solved by methods that can take advantage of such spatial structure. As such, we avoid selection of these tasks for this benchmark. Additionally, the latter types of home simulation environments pose an interesting and challenging set of tasks for policy learning, compared to more basic scenes that only contain a robot and objects on a solid-colored surface. However, these simulators are difficult to set up, with many layers of software complexity. State-of-the-art RL algorithms are also very far from learning on such long-horizon manipulation and navigation tasks.

In the search for better evaluations, there has been a consistent stream of benchmarks [20] released recently, and in particular environments for the evaluation of policy learning. In this work, we do not introduce a new environment, but instead add new functionality to an existing environment [33] that has already been used by the community [27, 50, 69, 70, 58]. The task we train policies on can be viewed as a kind of rearrangement task [8], where a fixed robot arm is asked to manipulate the objects in the scene to some given goal state. We do not use a mobile manipulator and circumvent the challenge of combining navigation and manipulation in this work.

Commonly, RL and robotics methods either report performance on the training domain or assume that the train and test sets are drawn from the same distributions, following standard assumptions and practices with training neural networks. We take the stance that deploying robotic agents in the real world requires policy learning approaches that can generalize or adapt to out-of-distribution testing scenes, which we evaluate in simulation with KitchenShift in this work.

For causal learning in robotics, CausalWorld [1] introduces task generators for parameter variation of cuboid block manipulation with a three finger robot, though their setup could be modified to evaluate zero-shot generalization. Our work is similar to ManiSkill [63] and MAGICAL [96], which leverage demonstrations and evaluate policy generalization. However, MAGICAL uses a visually simplistic 2D world, and ManiSkill is focused on robust object-level manipulation in an empty background scene. In contrast, we evaluate out-of-distribution policy generalization across 7 categories of domain shifts in a realistic kitchen scene. We find that BC alone performs better than other methods, which corresponds with results from other recent work [63, 43, 61, 102, 26] that also evaluate on more challenging tasks.

B Choice of Environment

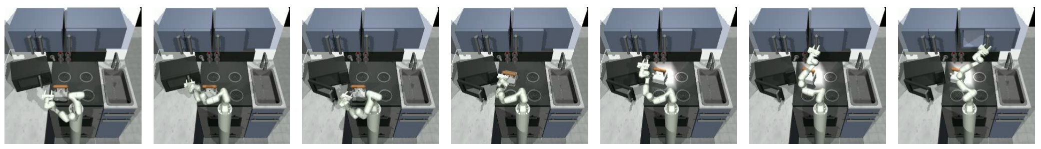

Figure 3: **Example demonstration observations.** In this `[microwave,kettle,switch,slide]` task demonstration, the robot first opens the microwave, then moves the kettle to the back burner, turns on the light switch, and finally opens the sliding cabinet.

To evaluate policy learning, we chose the kitchen environment, task definitions, and demonstration trajectories from Gupta et al. [33]. This environment is visually realistic with challenging multi-stage tasks involving a robot manipulator. Demonstrations given by human VR teleoperation of the robot are also available, compared to environments which use carefully designed policies to generate demonstrations or even ones without any demonstration data. Furthermore, this environment has already been used by the community [27, 50, 69, 70, 58].

MuJoCo has been acquired and open-sourced by DeepMind, which makes KitchenShift accessible for the community. There is also active development on other simulator backends such as NVIDIA Isaac Sim, [60], and in the near future, KitchenShift could be ported over to use other simulations.

Prior work [95, 2, 5, 22] has used the MuJoCo simulator for the purposes of domain randomization, and we use similar functionality to apply domain shifts to the environment. We also note that our domain shift evaluation protocol is not uniquely applicable to this particular kitchen environment. Our evaluation paradigm could be applied to other environments such as recent home environment simulations [49, 86, 104, 92, 28], which also have a wide range of objects, texture assets, and scenes; though that is outside the scope of this work.

C Experimental Setup

Using domain shifts and the KitchenShift benchmark as described in Section 2, we evaluate the generalization of different imitation-based policy learning approaches. As in Gupta et al. [33], we report the average step completion, out of a maximum of four total subgoals. Agents are given a small set of 19 expert demonstration trajectories of the training task. All results reported are averaged across 4 random seeds. Additional details can be found in Appendix D.

We prioritized methods with publicly available code that were easier to implement and tune, as well as ones that could take advantage of learning from demonstrations for improved sample efficiency. The implementations we developed are open-sourced in a shared codebase [3]. The repository also includes the full set of experiment logs and models used to generate the results in this paper. We evaluate combinations of the following imitation learning and representation learning baselines.

C.1 Imitation learning

We use an implementation of behavioral cloning (BC) open-sourced by Dasari and Gupta [18], which parameterizes the policy as a discretized logistic mixture distribution [79, 59]. We include a comparison to standard BC trained with mean-squared error (MSE), which corresponds to maximizing the expected log likelihood under a Gaussian distribution [68]. We also experiment with jointly optimizing an auxiliary dynamics modeling objective to regularize policy learning, which was studied by Dasari and Gupta [18] and in other work as well [107, 54, 99, 106].

To support training with multi-task demonstration data, we also evaluate goal-conditioned BC (GCBC) [59], which conditions the policy on a goal image specifying the task to perform. We use the goal relabeling strategy originally proposed by Hindsight Experience Replay (HER) [4], but we do not call a reward function to relabel rewards as in HER.

C.2 Representation learning

Reconstruction representations have been used extensively for perception in policy learning. We experiment with different autoencoders [47, 77]: β-VAE [40], RAE [31], and σ-VAE [78]. For robotics applications, β-VAE is commonly used [65, 64, 72, 115]. Rybkin et al. [78] recently proposed σ-VAE as an extension to β-VAE that automatically sets the β hyperparameter. Furthermore, Yarats et al. [110] find that a model-free RL policy using a deterministic autoencoder (RAE) outperforms more complex model-based planning approaches that employed β-VAE, when the environment background contained noise distractors.

Building off the recent success of instance discrimination for self-supervised learning in computer vision [13, 38, 15], contrastive representations have been studied for policy learning. We evaluate ATC [89], a version of noise contrastive estimation [35, 66, 13, 38] that associates images within a temporal window and applies the random shift augmentation proposed in Yarats et al. [110]. We also

[3]Our code: `github.com/etaoxing/kitchen-shift`.

experiment with SimSiam [15], which associates two augmentations of the same image without using negative pairs, and use the larger set of augmentations from computer vision tasks.

The same four-layer ConvNet from Sekar et al. [84] is used for the encoder network to compare different methods. As a baseline, we fix the initial random weights of the encoder network layers [89, 85], to validate that the policy leverages image observations to aid control. We also experiment with a deeper encoder network architecture: the 15-layer network used in IMPALA [24].

C.3 Random actions and Demo playback

We also report performance on two basic, non-learning baselines. For Random actions, we sample an action uniformly from the action space at each timestep. For Demo playback, we randomly sample a demonstration and execute the actions given by the demonstration trajectory open-loop.

D Experimental Details

We render (256, 256) RGB image observations of the environment and resize them to (224, 224) as input to the agent, following conventions from computer vision with ConvNets for image classification. Image observations are embedded into a 128-dim latent vector by an encoder network. We also provide the policy with a noisy observation of the robot's proprioceptive state (36 dimensions): joint positions, joint velocities, as well as end effector position, orientation, and contact forces. Policies are parameterized by a three-layer feedforward neural network with 256 units per layer. The continuous action space is 9-dim, controlling the robot gripper and the arm's joint velocities.

For methods that condition on goal context, we provide an image that specifies the final environment state (GCBC) and use the same encoder network as for the image observation. All methods are trained with the Adam optimizer ($\alpha = 0.001$, $\epsilon = 0.01$) and a batch size of 128 episode steps for a total of 1e5 optimization iterations. In early experiments, we trained policies for longer but found 1e5 iterations to be sufficient to evaluate policies.

Beyond the single training task evaluation where the policy must interact with the `[microwave,kettle,switch,slide]` objects, we widen the training distribution to include demonstrations of other tasks. All demos selected involve at least interacting with the `microwave` object, with 235 demos across a total of 12 multi-stage tasks. Agents are trained for twice the number of optimization steps used in the single training task setup. We train GCBC with this larger demonstrations dataset to compare to GCBC trained only with trajectories of the single task.

We used a university cluster to generate results; we ran four seeds at a time on 4-GPU (either Titan X (Pascal) or 1080 Ti) machines with 32-vCPUs each. Config files are available in our repository, containing hyperparameters used for all methods and experiments we ran.

E Impact of simulator initialization

The original kitchen environment code [33] resets and initializes the state of the underlying physics simulator in a deterministic fashion. We found that this determinism inflates the performance of methods, as the policy is evaluated in the exact same simulation conditions, even with uniform noise added to proprioceptive observations given to the policy as per the original settings.

To illustrate this gap, we generate demonstrations in environments with deterministic initialization and with noise perturbations applied during initialization. For the latter, we combine two types of perturbation: (i) small uniform noise added to the initial state when resetting the simulator; (ii) stepping the simulation a random number of times when resetting. Perturbation (ii) further changes the simulator state because we do not compensate for gravity in the robot controller, so the unactuated robot arm can drift. This noise perturbation only affects the initial state of the environment.

In Table 2, we show results of deterministic versus stochastic environment initialization when evaluating policies. We report results with BC (state-vector) as a baseline which achieves similar performance as the original behavioral cloning results from Gupta et al. [33] in environments with deterministic initialization. Notably, evaluating in environments with the stochastic initialization reduces policy performance for all methods, most notably by 51% for BC (β-VAE).

Table 2: **Comparing simulator initializations.** We use 'deterministic' to refer to deterministically resetting the environment to the same initial state, while 'stochastic' refers to adding a small amount of noise to the initial state and stepping the simulation a random number of times after resetting. We report mean$\pm$stddev of the average number of steps completed (out of a total of four). All results are averaged across 20 episodes and 4 random seeds. Policy performance is significantly better when the evaluation environment initialization is deterministic. Evaluating with initialization noise reduces performance by 51% for BC (β-VAE) (compare 2.51 in row three vs. 1.23 in row six).

Method	Environment initialization	(*training* domain) Avg. steps completed
BC (state-vector)	deterministic	1.44 ± 1.01
BC	deterministic	2.79 ± 0.90
BC (β-VAE)	deterministic	2.51 ± 1.12
Demo playback	deterministic	1.62 ± 1.03
BC (state-vector)	stochastic	0.96 ± 0.94
BC	stochastic	2.03 ± 1.12
BC (β-VAE)	stochastic	1.23 ± 1.21
Demo playback	stochastic	1.26 ± 1.07

To ensure that we report on a reasonable metric for realistic conditions (the importance of which is highlighted by Henderson et al. [39]), for all other experiments evaluate policies in environments with stochastic initialization. Our purpose here is to show that a small amount of perturbation to the initial state significantly reduces the reported performance of the policy.

F Domain shifts evaluated in KitchenShift

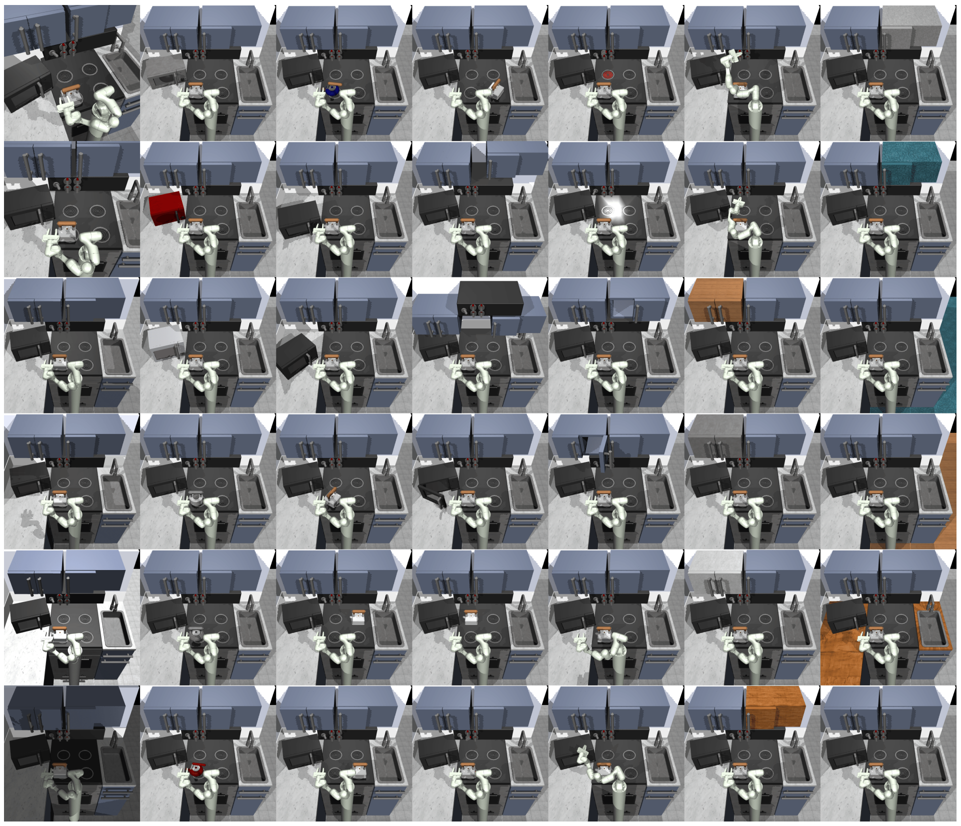

Figure 4: **Visualizing initial observations of domains in KitchenShift, in a grid.**