# A Framework for Efficient Robotic Manipulation

Albert Zhan*, Ruihan (Philip) Zhao*, Lerrel Pinto, Pieter Abbeel, Michael Laskin
University of California, Berkeley
{albertzhan,philipzhao}@berkeley.edu

***Abstract*— Data-efficient learning of manipulation policies from visual observations is an outstanding challenge for real-robot learning. While deep reinforcement learning (RL) algorithms have shown success learning policies from visual observations, they still require an impractical number of real-world data samples to learn effective policies. However, recent advances in unsupervised representation learning and data augmentation significantly improved the sample efficiency of training RL policies on common simulated benchmarks. Building on these advances, we present a Framework for Efficient Robotic Manipulation (FERM) that utilizes data augmentation and unsupervised learning to achieve extremely sample-efficient training of robotic manipulation policies with sparse rewards. We show that, given only 10 demonstrations, a single robotic arm can learn sparse-reward manipulation policies from pixels, such as reaching, picking, moving, pulling a large object, flipping a switch, and opening a drawer in just 15-50 minutes of real-world training time. We include videos, code, and additional information on the project website – https://sites.google.com/view/efficient-robotic-manipulation.**

## I. Introduction

Recent advances in deep reinforcement learning (RL) have given rise to unprecedented capabilities in autonomous decision making. Notable successes include learning to solve a diverse set of challenging video games [1]–[4], mastering complex classical games like Go, Chess, Shogi, and Hanabi [5]–[7], and learning autonomous robotic control policies in both simulated [8]–[11] and real-world settings [12], [13]. In particular, deep RL has been an effective method for learning diverse robotic manipulation policies such as grasping [14]–[17] and dexterous in-hand manipulation of objects [18].

However, to date, general purpose RL algorithms have been extremely sample inefficient, which has limited their widespread adoption in the field of robotics. State-of-the-art RL algorithms for discrete [19] and continuous [20] control often require approximately tens of millions of environment interactions to learn effective policies from pixel input [21], while training the Dota5 agent [2] to perform competitively to human experts required an estimated 180 years of game play. Even when the underlying proprioceptive state is accessible, sparse reward robotic manipulation still requires millions of training samples [22], an estimated 2 weeks of training in real time, to achieve reliable success rates on fundamental tasks such as reaching, picking, pushing, and placing objects.

A number of strategies have been proposed to overcome the data-efficiency challenge in deep RL for manipulation.

*Equal contribution

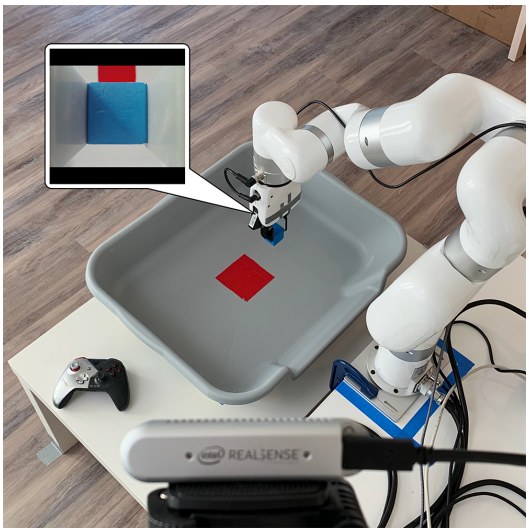
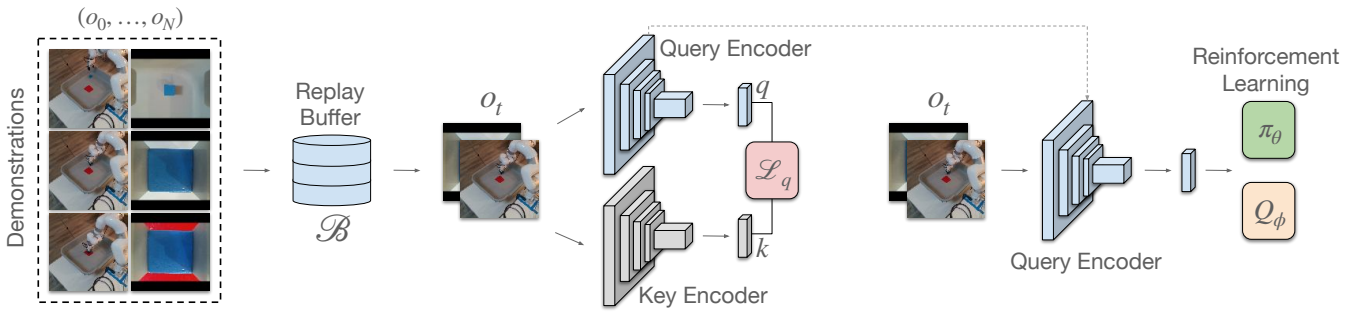


Fig. 1: The **F**ramework for **E**fficient **R**obotic **M**anipulation (FERM) enables robotic agents to learn skills directly from pixels in less than one hour of training. Our setup requires a robotic arm, two cameras, and a joystick to provide 10 demonstrations.

One approach is Sim2Real, where an RL policy is first trained in simulation and then transferred to the real world [18], [23]–[25]. In this framework, RL policies are trained in simulation where both visual and physical attributes of the environment and agent are randomized to expand the support of the training data. The resulting policy is then transferred to a real world system. While Sim2Real can be effective, its drawbacks are high-variance in the resulting policies and significant computational resources required to train the policy with domain randomization [23].

Another common approach to learned control is through imitation learning [26]–[30], where a large number of expert demonstrations are collected and the policy is extracted through supervised learning by regressing onto the expert trajectories. Imitation learning usually requires hundreds or thousands of expert demonstrations, which are laborious to collect, and the resulting policies are bounded by the quality of expert demonstrations. It would be more desirable to learn the optimal policy required to solve a particular task autonomously.

In this work, rather than relying on transferring policies from simulation or labor intensive human input through imitation learning or environment engineering, we investigate how pixel-based RL can itself be made data-efficient. Recent




**(a)** Collect 10 human demonstrations

**(b)** Initialize CNN encoder with contrastive pre-training

**(c)** Continue training with data-augmented RL

Fig. 2: The FERM architecture. First, demonstrations are collected, and stored in a replay buffer. These observations are used to pre-train the encoder with a contrastive loss. The encoder and replay buffer are then used to train an RL agent using an offline data-augmented RL algorithm.

progress in unsupervised representation learning [31], [32] and data augmentation [10], [33] has significantly improved the efficiency of learning with RL in simulated robotic [21] and video game [34] environments. The primary strength of these methods is learning high quality representations from image input either explicitly through unsupervised learning or implicitly by augmenting the input data.

Building on these advances, we propose a **F**ramework for **E**fficient **R**obotic **M**anipulation (FERM). FERM utilizes off-policy RL with data augmentation along with unsupervised pre-training to learn efficiently with a simple three-staged procedure. First, a small number of demonstrations (10) are collected and stored in a replay buffer. Second, the convolutional encoder weights are initialized with unsupervised contrastive pre-training on the demonstration data. Third, an off-policy RL algorithm is trained with augmented images on both data collected online during training and the initial demonstrations.

We summarize the key benefits of our method: *(1) Data-efficiency:* FERM enables learning optimal policies on 6 diverse manipulation tasks such as reaching, pushing, moving, pulling a large object, flipping a switch, drawer opening **in 15-50 minutes of total training time** for each task. *(2) A simple unified framework:* Our framework combines existing components such as contrastive unsupervised pre-training and online RL with data augmentation into a single framework for efficient learning that is simple and easy to reproduce[1]. *(3) General & lightweight setup:* Our setup requires a robot, one GPU, two cameras, a handful of demonstrations, and a sparse reward function. These requirements are quite lightweight relative to setups that rely on Sim2Real, motion capture, multiple robots, or engineering dense rewards.

## II. Related Work

### A. Imitation Learning

Imitation learning is a framework for learning autonomous skills from demonstrations. One of the simplest and perhaps most widely used forms of imitation learning is behavior cloning (BC) where an agent learns a skill by regressing onto demonstration data. BC has been successfully applied across diverse modalities including video games [35], autonomous navigation [36], [37], autonomous aviation [38], locomotion [39], [40], and manipulation [26], [28], [30], [41]. Other imitation learning approaches include Dataset Aggregation [42], Inverse Reinforcement Learning [43], [44], and Generative Adversarial Imitation Learning [27]. A general limitation of imitation learning approaches is the requirement for a large number of demonstrations for each task [45].

### B. Reinforcement Learning

Reinforcement Learning (RL) has been a promising approach for robotic manipulation due to its ability to learn skills autonomously, but has not achieved widespread adoption in real-world robotics. Recently, deep RL methods excelled at playing video games from pixels [1], [2] as well as learning robotic manipulation policies from visual input [12], [46]–[48]. However, widespread adoption of RL in real-world robotics has been bottle-necked due to the data-inefficiency of the method, among other factors such as safety. Though there exist prior frameworks for efficient position controlled robotic manipulation [49], they still require hours of training per task and provide additional information such as a dense reward function. FERM is most closely related to other methods that use RL with demonstrations. Prior methods [50]–[52] solve robotic manipulation tasks from coordinate state input by initializing the replay buffer of an RL algorithm with demonstrations to overcome the exploration problem in the sparse reward setting.

### C. Data Augmentation

Image augmentation refers to stochastically altering images through transformations such as cropping, rotating, or color-jittering. It is widely used in computer vision architectures including seminal works such as LeNet [53] and AlexNet [54]. Data augmentation has played a crucial role in unsupervised representation learning in computer vision [55]–[57], while other works investigated automatic generation of data augmentation strategies [58]. Data augmentation

[1]Link to website and code: https://sites.google.com/view/efficient-robotic-manipulation

has also been utilized in prior real robot RL methods [13]; however, the extent of its significance for efficient training was not fully understood until recent works [10], [31], [33], which showed that carefully implemented data augmentation makes RL policies from pixels as efficient as those from coordinate state. Finally, data augmentation has also been shown to improve performance in imitation learning [30]. In this work, data augmentation comprises one of three components of a general framework for efficient learning.

### D. Unsupervised Representation Learning

The goal of unsupervised representation learning is to extract representations of high-dimensional unlabeled data that can then be used to learn downstream tasks efficiently. Most relevant to our work is contrastive learning, which is a framework for learning effective representations that satisfy similarity constraints between a pair of points in dataset. In contrastive learning, latent embeddings are learned by minimizing the latent distance between similar data points and maximizing them between dissimilar ones. Recently, a number of contrastive learning methods [55], [57], [59] have achieved state-of-the-art label-efficient training in computer vision. A number of recent investigations in robotics have leveraged contrastive losses to learn viewpoint invariant representations from videos [60], manipulate deformable objects [61], and learn object representations [62]. In this work, we focus on instance-based contrastive learning [63] similar to how it is used in vision [56], [57] and RL on simulated benchmarks [31], [32].

## III. Background

### A. Soft Actor Critic

The Soft Actor Critic (SAC) [47] is an off-policy RL algorithm that jointly learns an action-conditioned state value function through Q learning and a stochastic policy by maximizing expected returns. SAC is a state-of-the-art model-free RL algorithm for continuous control from state [47] and, in the presence of data augmentations, from pixels as well [10], [33]. In simulated benchmarks, such as DeepMind control [21], SAC is as data-efficient from pixels as it is from state. For this reason, we utilize it as our base RL algorithm for sparse-reward manipulation in this work. As an actor-critic method, SAC learns an actor policy $\pi_\theta$ and an ensemble of critics $Q_{\phi_1}$ and $Q_{\phi_2}$.

To learn the actor policy, samples are collected stochastically from $\pi_\theta$ such that $a_\theta(o, \xi) \sim \tanh\left(\mu_\theta(o) + \sigma_\theta(o) \odot \xi\right)$, where $\xi \sim \mathcal{N}(0, I)$ is a sample from a normalized Gaussian noise vector, and then trained to maximize the expected return as measured by the critics $Q_{\phi_i}$, as shown in Equation 1.

$$\mathcal{L}(\theta) = \mathbb{E}_{a\sim\pi}\left[Q^\pi(o, a) - \alpha \log \pi_\theta(a|o)\right] \tag{1}$$

Simultaneously to learning the policy, SAC also trains the critics $Q_{\phi_1}$ and $Q_{\phi_2}$ to minimize the Bellman equation in Equation 2. Here, a transition $t = (o, a, o', r, d)$ is sampled from the replay buffer $\mathcal{B}$, where $(o, o')$ are consecutive timestep observations, $a$ is the action, $r$ is the reward, and $d$ is the terminal flag.

$$\mathcal{L}(\phi_i, \mathcal{B}) = \mathbb{E}_{t\sim\mathcal{B}}\left[\left(Q_{\phi_i}(o, a) - (r + \gamma(1 - d)Q_{\text{targ}})\right)^2\right] \tag{2}$$

The function $Q_{\text{targ}}$ is the target value that the critics are trained to match, defined in Equation 3. The target is the entropy regularized exponential moving average (EMA) of the critic ensemble parameters, which we denote as $\bar{Q}_\phi$.

$$Q_{\text{targ}} = \left(\min_{i=1,2} \bar{Q}_{\phi_i}(o', a') - \alpha \log \pi_\theta(a'|o')\right) \tag{3}$$

where $(a', o')$ are the consecutive timestep action and observation, and $\alpha$ is a positive action-entropy coefficient. A non-zero action-entropy term improves exploration – the higher the value of $\alpha$ to more entropy maximization is prioritized over optimizing the value function.

### B. Unsupervised Contrastive Pretraining

Contrastive learning [59], [63]–[66] is a paradigm for unsupervised representation learning that aims to maximize agreement between similar pairs of data while minimizing it between dissimilar ones. This type of representation learning has seen a recent resurgence in the field of computer vision where it was shown [55]–[57] that representations pre-trained with a contrastive loss on a corpus of unlabeled ImageNet data, are effective for downstream classification tasks, matching and sometimes outperforming fully supervised learning and significantly outperforming it when the percentage of available labels per data point is small.

Contrastive methods require the specification of *query-key* pairs, also known as *anchors* and *positives*, which are similar data pairs whose agreement needs to be maximized. Given a query $q \in \mathbb{Q} = \{q_0, q_1, \dots\}$ and a key $k \in \mathbb{K} = \{k_0, k_1, \dots\}$, we seek to maximize the score $f_{\text{score}}(q, k)$ between them while minimizing them between the query $q$ and negative examples in the dataset $k_-$. The score function is most often represented as an inner product, such as a dot product $f_{\text{score}}(q, k) = q^T k$ [59], [63] or a bilinear product $f_{\text{score}}(q, k) = q^T W k$ [55], [66], while other Euclidean metrics are also available [67], [68].

Since the specification of positive query-key pairs is a design choice, it is usually straightforward to extract such pairs from the unlabeled dataset of interest. However, the exact extraction of negatives can be challenging without prior knowledge due to the lack of labels. For this reason, contrastive methods usually approximate negative sampling with Noise Contrastive Estimation (NCE) [69], which effectively generates negatives by sampling noisily from the dataset. In particular, modern contrastive approaches [31], [55]–[57] employ the InfoNCE loss [66], which is described in Equation 4 and can also be interpreted as a multi-class cross entropy classification loss with $K$ classes.

$$\mathcal{L}_q = \log \frac{\exp(q^T W k)}{\exp\left(\sum_{i=0}^{K} \exp(q^T W k_i)\right)} \tag{4}$$

In the computer vision setting, a simple and natural choice of query-key specification is to define queries and keys as two data augmentations of the same image. This approach, called instance discrimination, is used in most of the state-of-the-art representation learning methods for static images [56], [57] as well as RL from pixels [31]. In the minibatch setting, which we also employ in this work, the InfoNCE loss is computed by sampling $K = \{x_1, \ldots, x_K\}$ images from the dataset, generating queries $Q = \{q_1, \ldots, q_K\}$ and keys $K = \{k_1, \ldots, k_K\}$ with stochastic data augmentations $q_i, k_i = \text{aug}(x_i)$, and for each datapoint $x_i$ treating the rest of the images in the minibatch as negatives.

## IV. Method

Our proposed framework, shown in Figure 2, combines demonstrations, unsupervised pre-training, and off-policy model-free RL with data augmentation into one holistic Framework. FERM has three distinct steps – (i) minimal collection of demonstrations (ii) encoder initialization with unsupervised pre-training and (iii) online policy learning through RL with augmented data – which we describe in detail below.

### A. Minimal Collection of Demonstrations

We initialize the replay buffer with a small number of expert demonstrations (we found 10 to be sufficient) for each task. Demonstrations are collected with a joystick controller, shown in Figure 1. Our goal is to minimize the total time required to acquire a skill for an RL agent, including both policy training as well as time required to collect demonstrations. While collecting a larger number of demonstrations certainly improves training speed, which we discuss in Section V-C.1, we find 10 demonstrations is already sufficient to learn skills quickly. For real world experiments, collecting 10 expert demonstrations can be done within 10 minutes (see Table I), which includes the time needed to reset the environment after every demonstration.

### B. Unsupervised Encoder Pre-training

After intializing the replay buffer with 10 demonstrations, we pre-train the convolutional encoder with instance-based contrastive learning, using stochastic random crop [31] to generate query-key pairs. The key encoder is an exponentially moving average of the query encoder [56], and the similarity measure between query-key pairs is the bi-linear inner product [66] shown in Equation 4. Note that the bi-linear inner product is only used to pre-train the encoder. After pre-training, the weight matrix in the bi-linear measure is discarded.

### C. Reinforcement Learning with Augmented Data

After pre-training the convolutional encoder on offline demonstration data, we train a SAC [47] agent with data augmentation [10] as the robot interacts with the environment. Since the replay buffer was initialized with demonstrations and SAC is an off-policy RL algorithm, during each minibatch update the agent receives a mix of demonstration observations and observations collected during training when performing gradient updates. The image augmentation used during training is random crop – the same augmentation used during contrastive pre-training.

## V. Experimental Evaluation

In this section, we investigate the efficacy of our proposed method – FERM. Our goal is to provide a simple yet effective baseline for robotic manipulation from pixels that is accessible to other researchers. Our hypothesis is that contrastive pre-training combined with data augmentated RL should result in data-efficient training given a handful of demonstrations to reduce the exploration challenge in the presence of sparse rewards.

Since FERM is composed of three independent ingredients, we ablate how each piece contributes to the overall framework. In addition to our hypothesis, we investigate the contribution of each component of the framework by answering the following questions: (1) Are demonstrations required to learn efficiently and, if so, how many? (2) How does contrastive pre-training affect the performance of our agent and how many updates are required for initialization? (3) How important is data augmentation during online training of RL?

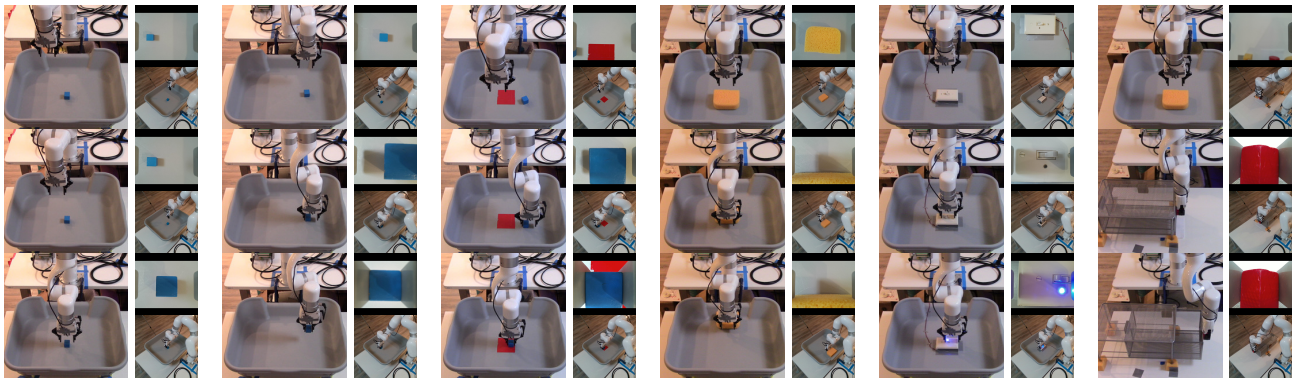
### A. Experimental Setup

**Real Robot**: We use the xArm [70] robot for all real-world experiments. The end effector, a parallel two-jaw gripper, is position controlled with three degrees of freedom. The action input to the robot is the gripper motion and aperture displacement. **Input**: We use two RGB cameras, one positioned over the shoulder for maximal view of the arm, and the other located within the gripper to provide a local object-level view. The inputs images have a resoluition of $1280 \times 720$ and $640 \times 480$ respectively, and are downsized, concatenated, and cropped randomly before being passed into the neural networks. **Demonstrations**: Using a Xbox controller [71], we teleoperate the robot. Collecting demonstrations for each task requires less than 10 minutes, which includes resetting the environment.

**Tasks:** For the main results shown in Table I, we evaluate FERM on six robotic manipulation tasks - reaching an object, picking up a block, moving a block to a target destination, pulling a large deformable object, flipping a switch, and opening a drawer. The block manipulation tasks (reach, pickup, move) are real-world versions of tasks from the OpenAI Gym Fetch suite [72]. Since our method uses demonstrations, we include pull, which has been used in prior work on imitation learning [41], [73]. Flipping a switch is included as it demands high precision, while drawer opening is a common task in existing simulated robotic benchmarks [74] . Details of task setup are provided in Section VIII-A.

### B. Results

The main results of our investigation, including the time required to train an optimal policy as well the first successful task completion, are shown in Table I. We summarize the



(a) Reach (b) Pickup (c) Move (d) Pull (e) Light Switch (f) Drawer Open

Fig. 3: The set of tasks used in this work, along with their pixel observations. Each column shows initial, intermediate, and completion states of a rollout during evaluation of our optimal policy. The right two images comprise the observational input to the RL agent. The sparse reward is only given when the robot completes the task. FERM is able to solve all 6 tasks within an hour, using only 10 demonstrations.

| Tasks | Reach | Pickup | Move | Pull | Light Switch | Drawer Open |
|---|---|---|---|---|---|---|
| Time to record Demonstrations (min) | <10:00 | <10:00 | <10:00 | <10:00 | <10:00 | <10:00 |
| # Expert demonstrations | 10 | 10 | 10 | 10 | 10 | 10 |
| Time to First Success (mm:ss) | 3:05 | 15:00 | 33:00 | 05:12 | 05:01 | 5:56 |
| # Episodes to First Success | 6 | 30 | 40 | 5 | 6 | 7 |
| Time to Optimal Policy (mm:ss) | 15:00 | 26:00 | 46:00 | 29:10 | 16:05 | 20:21 |
| # Episodes to Optimal Policy | 20 | 60 | 80 | 45 | 20 | 25 |
| Number of Successes for Eval ( /30) | 30 | 30 | 26 | 28 | 30 | 30 |
| Success Rate for Eval (%) | 100 | 100 | 86.7 | 93.3 | 100 | 100 |

TABLE I: The speed at which our agents learn to complete the tasks. Listed above are the to demonstration collection times, as well as the time at which the policy first achieves a success, and when an optimal policy is learnt. The optimal policy is then used to evaluate for 30 episodes, and the number of successes and the converted success rates are shown. Our method starts to complete the tasks in around 30 minutes of training, and as little as 3 minutes for simple tasks such as Reach.

key findings below:

(i) On average, FERM enables a single robotic arm to learn optimal policies across all 6 tasks tested within **within 25 minutes of training time** with a range of 15-50 minutes, which corresponds to to 20-80 episodes of training.

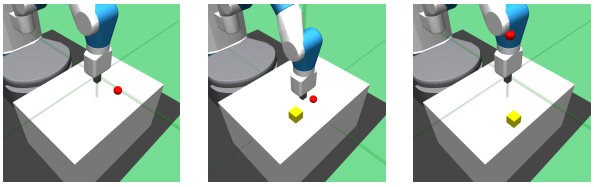
(ii) The time to first successful task completion is **on average 11 minutes** with a range of 3-33 minutes. The final policies achieve an **average success rate of 96.7%** with a range of 86.7-100% across the tasks tested, suggesting that they have converged to near-optimal solutions to the tasks.

(iii) Collecting demonstrations and contrastive pre-training don't introduce significant overhead. Collecting 10 expert demonstrations with a joystick requires 10 minutes of human operation, and contrastive pre-training is fast, completed within 40 seconds on a single GPU.

(iv) FERM solves all 6 tasks using the **same hyperparameters** and without altering the camera setup, which demonstrates the ease of use and generality of the framework.

Altogether, RL trained with FERM is able to learn optimal policies for the 6 tasks extremely efficiently. While prior work was able to solve dexterous manipulation tasks using RL with demonstrations in 2-3 hours of training [49], it also utilized dense rewards and more demonstrations. To the best of our knowledge, FERM is the first method to solve a diverse set of sparse-reward robotic manipulation tasks directly from pixels in less than one hour.

### C. Ablations



(a) Reach (b) Push (c) Pick-and-place

Fig. 4: Simulated environments used in addition to the real robot experiments include the reach, push, and pick-and-place tasks from the OpenAI Gym Fetch environment [72].

In this section, we investigate how the three core components of FERM – demonstrations, contrastive pre-training, and data augmentation – contribute to the overall efficiency of the framework.

*1) Demonstrations:* In real robot settings, assigning dense rewards is often difficult or infeasible. While sparse rewards

are simpler to define, they pose an exploration challenge since the robot is unlikely to randomly stumble on a reward state. We address this issue by providing demonstrations to the RL agent. We ablate the number of demonstrations required to learn efficiently on the simulated pick and place task in Figure 5. We find that while the agent fails entirely with zero demonstrations, it is able to start learning the task with just one demonstration. While more demonstrations improve learning efficiency and reduce the variance of the policy, ten demonstrations suffice to learn quickly.

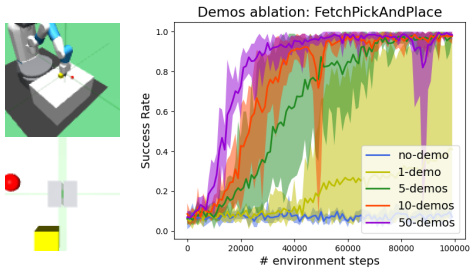



Fig. 5: We ablate the number of demonstrations required by FERM, and find that though the agent fails to learn with zero demonstrations, it can learn the pick-and-place task efficiently using only 10 demonstrations.

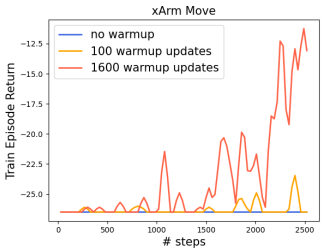



Fig. 6: We compare the performance of the move task with and without the use of pre-training on the real xArm robot. The plotted episode returns during training show that the pick and move task fails to learn without contrastive pre-training.

*2) Unsupervised pretraining:* We next study the role of contrastive pre-training in FERM. We ablate our method with and without contrastive pre-training on the real world move task, shown in Figure 6, where we compare using 0, 100, and 1600 iterations of pre-training to initialize the encoder. With 1600 contrastive iterations, the agent is able to learn an optimal policy while the other runs fail to learn. In the case of no pre-training at all, the agent is only able to succeed once during the entire hour of training.

*3) Data augmentation:* To justify the use of data augmentation during online RL training, we compare the performance of SAC with and without data augmentation for a simple, dense reward reaching task. In the FetchReach environment, we use the dense reward $r = -d$ where $d$ is the Euclidean distance between the gripper and the goal. As shown in Figure 7, without data augmentation, the RL agent is unable to learn the simple task, and asymptotically collapses. This motivates us to use data augmentation for more difficult tasks along with sparse reward functions, which encounter even less signal to learn features.

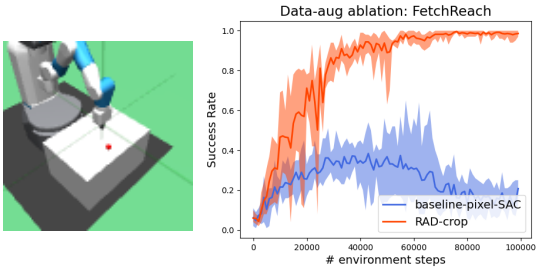



Fig. 7: Policy performance is measured by evaluation success rate. A single camera view is provided as the observation (left). Using data augmentation, the agent achieves optimal performance while using non-augmented observations, the agent fails to learn the task.

## VI. Conclusion and Future Work

We present FERM, a framework that combines demonstrations, unsupervised learning, and RL, to efficiently learn complex tasks in the real world. Using purely image input, our method is able to successfully solve a diverse set of tasks, all using the same hyperparameters, and from sparse reward. Due to the limited amount of supervision required, our work presents exciting avenues for applying RL to real robots in a quick and efficient manner.

## VII. Acknowledgements

We gratefully acknowledge support from Open Philanthropy, Darpa LwLL, Berkeley Deep Drive and Amazon Web Services.

## References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[2] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse *et al.*, "Dota 2 with large scale deep reinforcement learning," *arXiv preprint arXiv:1912.06680*, 2019.

[3] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J. P. Agapiou, M. Jaderberg, A. S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T. L. Paine, C. Gulcehre, Z. Wang, T. Pfaff, Y. Wu, R. Ring, D. Yogatama, D. Wünsch, K. McKinney, O. Smith, T. Schaul, T. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, and D. Silver, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019. [Online]. Available: https://doi.org/10.1038/s41586-019-1724-z

[4] A. P. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, D. Guo, and C. Blundell, "Agent57: Outperforming the atari human benchmark," in *International Conference on Machine Learning*, 2020.

[5] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, pp. 484–503, 2016. [Online]. Available: http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html

[6] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. Driessche, T. Graepel, and D. Hassabis, "Mastering the game of go without human knowledge," *Nature*, vol. 550, pp. 354–359, 10 2017.

[7] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel *et al.*, "Mastering atari, go, chess and shogi by planning with a learned model," *arXiv preprint arXiv:1911.08265*, 2019.

[8] J. Schulman, S. Levine, P. Abbeel, M. I. Jordan, and P. Moritz, "Trust region policy optimization." in *ICML*, 2015, pp. 1889–1897.

[9] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[10] M. Laskin, K. Lee, A. Stooke, L. Pinto, P. Abbeel, and A. Srinivas, "Reinforcement learning with augmented data," *arXiv preprint arXiv:2004.14990*, 2020.

[11] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," in *International Conference on Learning Representations*, 2020.

[12] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *CoRR*, vol. abs/1504.00702, 2015. [Online]. Available: http://arxiv.org/abs/1504.00702

[13] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke *et al.*, "Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation," *arXiv preprint arXiv:1806.10293*, 2018.

[14] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," *ICRA*, 2016.

[15] J. Mahler, F. T. Pokorny, B. Hou, M. Roderick, M. Laskey, M. Aubry, K. Kohlhoff, T. Kröger, J. Kuffner, and K. Goldberg, "Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards," in *ICRA*, 2016.

[16] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *ISER*, 2016.

[17] A. Gupta, A. Murali, D. P. Gandhi, and L. Pinto, "Robot learning in homes: Improving generalization and reducing dataset bias," in *Advances in Neural Information Processing Systems*, 2018, pp. 9094–9104.

[18] M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray *et al.*, "Learning dexterous in-hand manipulation," *arXiv preprint arXiv:1808.00177*, 2018.

[19] M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," 2017.

[20] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[21] 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.

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

[23] 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 *IROS*, 2017.

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

[25] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *International Conference on Robotics and Automation*, 2018.

[26] T. Zhang, Z. McCarthy, O. Jow, D. Lee, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–8, 2018.

[27] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 4565–4573. [Online]. Available: http://papers.nips.cc/paper/6391-generative-adversarial-imitation-learning.pdf

[28] Y. Duan, M. Andrychowicz, B. C. Stadie, J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba, "One-shot imitation learning," 2017.

[29] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine, "One-shot visual imitation learning via meta-learning," ser. Proceedings of Machine Learning Research, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., vol. 78. PMLR, 13–15 Nov 2017, pp. 357–368. [Online]. Available: http://proceedings.mlr.press/v78/finn17a.html

[30] S. Young, D. Gandhi, S. Tulsiani, A. Gupta, P. Abbeel, and L. Pinto, "Visual imitation made easy," *CoRR*, vol. abs/2008.04899, 2020. [Online]. Available: https://arxiv.org/abs/2008.04899

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

[32] A. Stooke, K. Lee, P. Abbeel, and M. Laskin, "Decoupling representation learning from reinforcement learning," 2020.

[33] I. Kostrikov, D. Yarats, and R. Fergus, "Image augmentation is all you need: Regularizing deep reinforcement learning from pixels," *arXiv preprint arXiv:2004.13649*, 2020.

[34] 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.

[35] S. Ross, G. J. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, ser. JMLR Proceedings, G. J. Gordon, D. B. Dunson, and M. Dudík, Eds., vol. 15. JMLR.org, 2011, pp. 627–635. [Online]. Available: http://proceedings.mlr.press/v15/ross11a/ross11a.pdf

[36] D. Pomerleau, "ALVINN: an autonomous land vehicle in a neural network," in *Advances in Neural Information Processing Systems 1, [NIPS Conference, Denver, Colorado, USA, 1988]*, D. S. Touretzky, Ed. Morgan Kaufmann, 1988, pp. 305–313. [Online]. Available: http://papers.nips.cc/paper/95-alvinn-an-autonomous-land-vehicle-in-a-neural-network

[37] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," *CoRR*, vol. abs/1604.07316, 2016. [Online]. Available: http://arxiv.org/abs/1604.07316

[38] A. Giusti, J. Guzzi, D. C. Ciresan, F. He, J. P. Rodriguez, F. Fontana, M. Faessler, C. Forster, J. Schmidhuber, G. D. Caro, D. Scaramuzza, and L. M. Gambardella, "A machine learning approach to visual perception of forest trails for mobile robots," *IEEE Robotics Autom. Lett.*, vol. 1, no. 2, pp. 661–667, 2016. [Online]. Available: https://doi.org/10.1109/LRA.2015.2509024

[39] J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato, "Learning from demonstration and adaptation of biped locomotion," *Robotics Auton. Syst.*, vol. 47, no. 2-3, pp. 79–91, 2004. [Online]. Available: https://doi.org/10.1016/j.robot.2004.03.003

[40] M. Kalakrishnan, J. Buchli, P. Pastor, and S. Schaal, "Learning locomotion over rough terrain using terrain templates," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, October 11-15, 2009, St. Louis, MO, USA*. IEEE, 2009, pp. 167–172. [Online]. Available: https://doi.org/10.1109/IROS.2009.5354701

[41] R. Rahmatizadeh, P. Abolghasemi, L. Bölöni, and S. Levine, "Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration," *CoRR*, vol. abs/1707.02920, 2017. [Online]. Available: http://arxiv.org/abs/1707.02920

[42] S. Ross, G. Gordan, and A. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *arXiv preprint arXiv:1011.0686*, 2010.

[43] A. Y. Ng and S. J. Russell, "Algorithms for inverse reinforcement learning," in *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2000), Stanford University, Stanford, CA, USA, June 29 - July 2, 2000*, P. Langley, Ed. Morgan Kaufmann, 2000, pp. 663–670.

[44] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Machine Learning, Proceedings of the Twenty-first International Conference (ICML 2004), Banff, Alberta, Canada, July 4-8, 2004*, ser. ACM International Conference Proceeding Series, C. E. Brodley, Ed., vol. 69. ACM, 2004. [Online]. Available: https://doi.org/10.1145/1015330.1015430

[45] P. Sharma, L. Mohan, L. Pinto, and A. Gupta, "Multiple interactions

made easy (mime): Large scale demonstrations data for imitation," *arXiv preprint arXiv:1810.07121*, 2018.
[46] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 2786–2793.
[47] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.
[48] A. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine, "Visual reinforcement learning with imagined goals," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., 2018, pp. 9209–9220. [Online]. Available: http://papers.nips.cc/paper/8132-visual-reinforcement-learning-with-imagined-goals
[49] H. Zhu, A. Gupta, A. Rajeswaran, S. Levine, and V. Kumar, "Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 3651–3657.
[50] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Overcoming exploration in reinforcement learning with demonstrations," in *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*. IEEE, 2018, pp. 6292–6299. [Online]. Available: https://doi.org/10.1109/ICRA.2018.8463162
[51] 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," *arXiv preprint arXiv:1709.10087*, 2017.
[52] M. Vecerík, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. A. Riedmiller, "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards," *CoRR*, vol. abs/1707.08817, 2017. [Online]. Available: http://arxiv.org/abs/1707.08817
[53] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[54] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017. [Online]. Available: http://doi.acm.org/10.1145/3065386
[55] O. J. Hénaff, A. Srinivas, J. De Fauw, A. Razavi, C. Doersch, S. Eslami, and A. v. d. Oord, "Data-efficient image recognition with contrastive predictive coding," *arXiv preprint arXiv:1905.09272*, 2019.
[56] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.
[57] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning*, 2020.
[58] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation strategies from data," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*. Computer Vision Foundation / IEEE, 2019, pp. 113–123. [Online]. Available: http://openaccess.thecvf.com/content_CVPR_2019/html/Cubuk_AutoAugment_Learning_Augmentation_Strategies_From_Data_CVPR_2019_paper.html
[59] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," *arXiv preprint arXiv:1911.05722*, 2019.
[60] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, and S. Levine, "Time-contrastive networks: Self-supervised learning from video," in *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*. IEEE, 2018, pp. 1134–1141. [Online]. Available: https://doi.org/10.1109/ICRA.2018.8462891
[61] W. Yan, A. Vangipuram, P. Abbeel, and L. Pinto, "Learning predictive representations for deformable objects using contrastive estimation," *CoRR*, vol. abs/2003.05436, 2020. [Online]. Available: https://arxiv.org/abs/2003.05436
[62] S. Pirk, M. Khansari, Y. Bai, C. Lynch, and P. Sermanet, "Online object representations with contrastive learning," *CoRR*, vol. abs/1906.04312, 2019. [Online]. Available: http://arxiv.org/abs/1906.04312
[63] Z. Wu, Y. Xiong, S. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance-level discrimination," *arXiv preprint arXiv:1805.01978*, 2018.
[64] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2. IEEE, 2006, pp. 1735–1742.
[65] Y. LeCun, S. Chopra, R. Hadsell, M. Ranzato, and F. Huang, "A tutorial on energy-based learning," 2006.
[66] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.
[67] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.
[68] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *ICCV*, 2015.
[69] M. U. Gutmann and A. Hyvärinen, "Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics," *J. Mach. Learn. Res.*, vol. 13, no. 1, p. 307–361, Feb. 2012.
[70] "xarm 7." [Online]. Available: https://store.ufactory.cc/products/xarm-7-2020
[71] "Xbox wireless controller." [Online]. Available: https://www.xbox.com/en-US/accessories/controllers/xbox-wireless-controller
[72] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.
[73] P. R. Florence, L. Manuelli, and R. Tedrake, "Self-supervised correspondence in visuomotor policy learning," *IEEE Robotics Autom. Lett.*, vol. 5, no. 2, pp. 492–499, 2020. [Online]. Available: https://doi.org/10.1109/LRA.2019.2956365
[74] 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 (CoRL)*, 2019. [Online]. Available: https://arxiv.org/abs/1910.10897

## VIII. Appendix

### A. Task Description

For all of our real robot tasks, the reward function is the same as the discrete reward in Fetch suite, with $0$ when the task is in a completion state, and $-1$ everywhere else. By design, our experiments are easy to reset at completion states, by simple hard-coded procedures. Our assumptions allow FERM to simply run with very little supervision, where the only human supervision is the 10 collected demonstrations.

1) **Reach**: The Robot must move to the block location. We consider a success when the gripper camera view has the block in its center. The gripper is constrained to be unable to reach low enough to interact with the block. The gripper aperture is locked to a set position. During training, we fix the block location, however the demonstrations include random block locations. The arm is reset to a random location after every episode.
2) **Pickup**: Success is declared if the robot picks the block up a specified height (70mm) above the work surface. At the end of each episode, if the gripper is around the block, it will reset the block to a random position, as well, resetting the gripper to a random location.
3) **Move**: An episode is deemed successful when the block is moved to the center, onto the goal. Specifically, reward is given when the goal and the block are close while both visible from the griper camera. This task is especially difficult, as the block can be anywhere relative to the goal, so the system must understand to move the block in many directions,

rather than a generic direction. As with Pickup, the block is reset at the end of each episode if the gripper can close and pick the block to a random location.

4) **Pull**: The gripper aperture is locked at a set position. Without gripping onto the sponge, the robot must pull the sponge to an area around its base. At the end of each successful episode, the sponge is moved to a new random position.
5) **Light switch**: A light switch panel is fixed to the work surface, and a blue LED lights up when the switch is flipped on. The gripper aperture is locked at a set position. Reward is given when blue light is visible from the gripper camera. At the end of each episode, a hard-coded reset procedure is executed to turn off the light.
6) **Drawer open**: The drawer is fixed to the work surface. The robot must grab onto the handle to pull open the drawer. Success is declared when the handle is visible from the gripper camera while the gripper position corresponds to the drawer being open. The drawer is closed by a hard-coded reset procedure at the end of each episode.

For Reach, Light switch, and Drawer open tasks, the goal is fixed, and so the reset is hard-coded. For Pickup and Move, the block is only reset to a random location when the gripper is gripping the block, and for Pull, the sponge is only reset upon successfully pulling the sponge to the base of the robot.

### B. Baselines

We compare against behavior cloning for the real world experiments.

For our real world experiments, we qualitatively examine the policies learnt from the same 10 demonstrations on a random goal (pickup), and a fixed goal task (switch). Videos of the policies are on the project website [2]. For light switch task, we found that behavior cloning was able to complete the task around half of the time (17/30 trials), as the policy learned to memorize the steps necessary to flip the switch at the specified position. Failure modes occurred when the policy did performed the movement to flip the switch, but missed hitting it. For the Pickup task, the policy was unable to locate the block at all. Even with lucky resets near the block, the policy is not robust and fails to pick the block up.

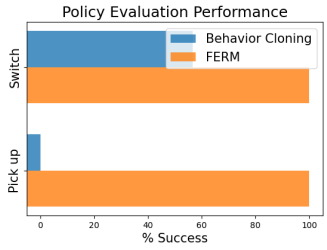



Fig. 8: Behavior cloning behavior on the Light Switch and Pickup task. Using the same demonstrations as our method, behavior cloning has limited capabilities due to low amounts of demonstrations.

[2] https://sites.google.com/efficient-robotic-manipulation

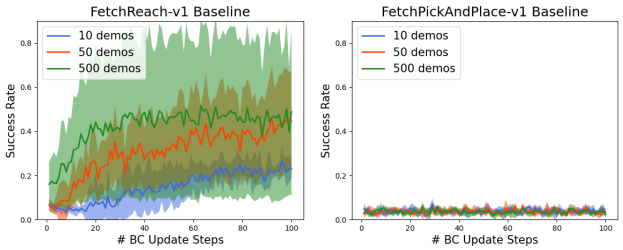



Fig. 9: In simulation, the Behavior cloning baseline is only able to recover a sub-optimal policy for the Reach task.

### C. Further Ablations

*1) Camera setup:* In our experiments, we find that within the two-camera setup, the gripper-mount egocentric camera provides strong signals to the policies. We ablate the effect of camera placement to justify our final camera configuration. Shown in Figure 10, the egocentric view is crucial for the Pick-and-place task, as it alone is able to achieve decent results. However, taking frames from both cameras still proves advantageous, as the over the shoulder camera provides guide in direction when the object or the goal is outside the view of the gripper mount camera. For push, both cameras are needed for the agent to learn a meaningful control policy.

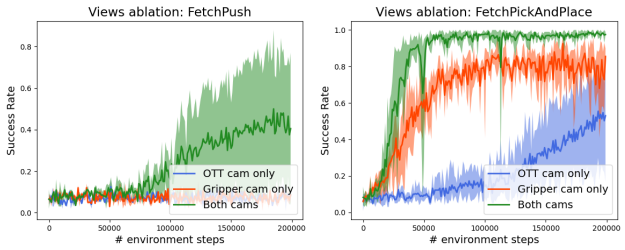



Fig. 10: Camera setup ablation: we compare the policy performance when trained with either one of RGB images or both. The use of both cameras proves essential for both the push and pick-and-place task.

*2) Unsupervised Pre-training:* For simulation and easier tasks in our suite, we noticed that the unsupervised pre-training had no significant benefit in performance. Figure 11 summarizes our results for the Pickup task, and Pick-And-Place task in sim.

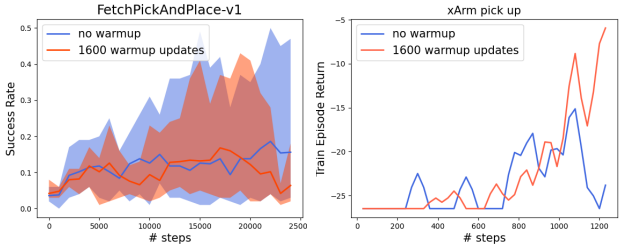



Fig. 11: Unsupervised pretraining ablation on simpler tasks. In both pick up and simulated pick and place, warming up the encoder doesn't introduce significant difference to the RL training performance. The real robot plot (right) is smoothed using a Gaussian kernel for better visibility.