On Pre-Training for Visuo-Motor Control: Revisiting a Learning-from-Scratch Baseline

Nicklas Hansen [1 *] **Zhecheng Yuan** [2 *] **Yanjie Ze** [1 3 *] **Tongzhou Mu** [1 *]
Aravind Rajeswaran [4 †] **Hao Su** [1 †] **Huazhe Xu** [2 5 †] **Xiaolong Wang** [1 †]

Abstract

In this paper, we examine the effectiveness of pre-training for visuo-motor control tasks. We revisit a simple Learning-from-Scratch (LfS) baseline that incorporates data augmentation and a shallow ConvNet, and find that this baseline is surprisingly competitive with recent approaches (PVR, MVP, R3M) that leverage frozen visual representations trained on large-scale vision datasets – across a variety of algorithms, task domains, and metrics in simulation and on a real robot. Our results demonstrate that these methods are hindered by a significant domain gap between the pre-training datasets and current benchmarks for visuo-motor control, which is alleviated by finetuning. Based on our findings, we provide recommendations for future research in pre-training for control and hope that our simple yet strong baseline will aid in accurately benchmarking progress in this area.[1]

1. Introduction

Large-scale pre-training has delivered promising results in computer vision (Doersch et al., 2015; He et al., 2020; van den Oord et al., 2018; Alayrac et al., 2022) and natural language processing (Devlin et al., 2019; Brown et al., 2020; Radford et al., 2021; Chowdhery et al., 2022). Recent works have extended the pre-training paradigm to visuo-motor control by leveraging pre-trained visual representations for policy learning (Parisi et al., 2022; Nair et al., 2022; Xiao et al., 2022; Ze et al., 2022; Yuan et al., 2022). These works train a visual representation using large out-of-domain vision datasets like ImageNet (Russakovsky et al., 2015) and Ego4D (Grauman et al., 2022), and freeze the vision model weights for downstream policy learning. When compared to simple Learning-from-Scratch (LfS) methods for visuo-motor control, these works find that frozen pre-trained representations help achieve higher sample efficiency and/or asymptotic performance across various task domains.

However, there exists a rich body of work on strategies to improve performance of LfS methods, such as auxiliary self-supervised representation learning (Srinivas et al., 2020; Schwarzer et al., 2021) or using carefully curated data augmentations (Laskin et al., 2020; Kostrikov et al., 2021; Yarats et al., 2021; Raileanu et al., 2020; Hansen et al., 2021). To gain a sharp understanding of the advantages of visual pre-training for visuo-motor control, it is necessary to establish strong LfS baselines.

Towards this end, we adopt the experimental setups of prior works without modification, and implement strong LfS baselines that leverage shallow ConvNet encoders and random shift data augmentation (Kostrikov et al., 2021; Yarats et al., 2021). Surprisingly, we find that this modified LfS baseline can achieve results competitive with prior works that leverage frozen pre-trained visual representations. While our contributions are incremental in nature, we believe that our work contains must-know insights for anyone working on pre-trained representations for visuo-motor control.

We evaluate our approach across a variety of task domains, algorithm classes, and evaluation metrics. Specifically, we examine **4** task domains (Adroit (Rajeswaran et al., 2018), DMControl (Tassa et al., 2018), PixMC (Xiao et al., 2022), and a real robot setup), **3** algorithm classes: imitation learning (behavior cloning), on-policy RL (PPO (Schulman et al., 2017)), and off-policy RL (DrQ-v2 (Yarats et al., 2021)), and multiple evaluation metrics including sample-efficiency, asymptotic performance, visual robustness, and computational cost. To our surprise, our carefully designed LfS baseline is found to be competitive with frozen pre-trained representations across most settings and metrics, and in some cases even outperforms them. At present, frozen pre-trained representations are found to mostly be advantageous in terms of computational cost.

[*] Equal contribution [†] Equal advising [1] University of California San Diego [2] Tsinghua University [3] Shanghai Jiao Tong University [4] Meta AI [5] Shanghai Qi Zhi Institute. Correspondence to: Nicklas Hansen <nihansen@ucsd.edu>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

[1] Code: https://github.com/gemcollector/learning-from-scratch.

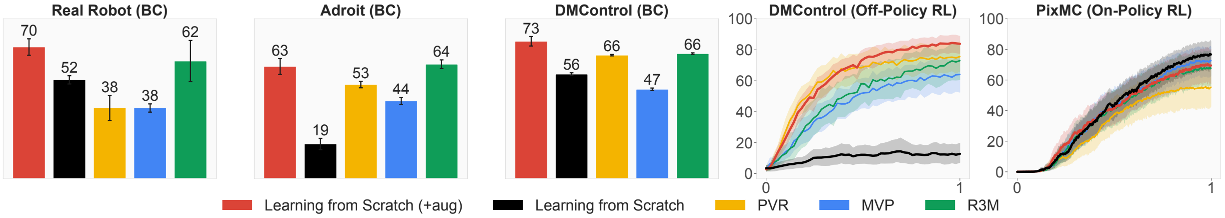
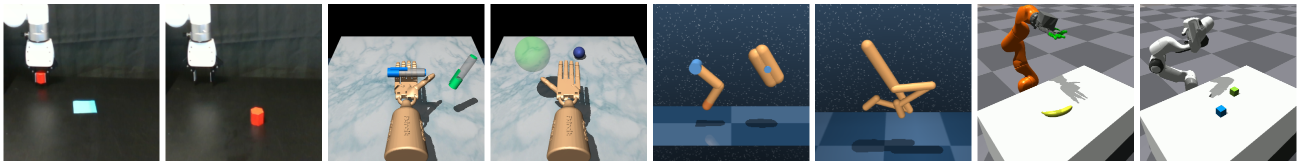


Figure 1. **Pre-training vs. Learning-from-Scratch (LfS).** Success rate (real robot, Adroit, PixMC) and normalized return (DMControl) in each of the four task domains that we consider (aggregated across tasks). BC simulation results are averages of top-3 evaluations over 100 epochs (Parisi et al., 2022), and RL results are reported as a function of environment steps (Yarats et al., 2021; Xiao et al., 2022), normalized to the interval $(0, 1)$ as total steps differ between tasks. We evaluate strong yet simple LfS baselines (Yarats et al., 2021; Hansen et al., 2021) and find them to be competitive with recent frozen pre-trained representations. Mean and 95% CIs over 5 seeds.



Real robot **Adroit** **DMControl** **PixMC**

Figure 2. **Tasks.** We consider challenging and diverse visuo-motor control tasks spanning **4** domains, from left to right: a real robot setup (manipulation), Adroit (dexterous manipulation), DMControl (locomotion, manipulation), and PixMC (manipulation). Our experimental setups in simulation are adopted from PVR, MVP, and R3M, and our real setup is similar to that of R3M. We consider a total of 17 tasks.

We remain optimistic that *pre-trained representations will play an important and increasingly larger role* in visuo-motor control as the paradigm matures. At the same time, we believe that setting a simple yet strong baseline will help accurately benchmark progress in this area. Based on our empirical findings, we provide recommendations for future research in pre-training for control. In particular, we conjecture that current benchmark tasks are not well suited to reap the benefits of pre-trained representations, since they do not require any visual generalization. As the community builds better benchmarks and harder tasks that require both visual and policy generalization, we conjecture that pre-trained representations will play an increasingly important role. Additionally, our results indicate that current pre-trained representations suffer from a substantial domain gap by pre-training on large-scale real-world data and benchmarking on predominantly simulated environments, which we find can be alleviated with careful in-domain finetuning based on our LfS insights. In the following sections, we detail each method, experimental setup, and results, and conclude with a broader discussion on the implications of our findings.

2. Methods

Comparing two *paradigms* fairly is difficult, and comparing LfS with pre-trained representations is no exception. To help narrow our scope, we focus on *representative methods* from each paradigm: a simple Learning-from-Scratch (LfS) method that uses a shallow ConvNet and data augmentation, as well as three **frozen** visual representations trained on large-scale out-of-domain vision datasets (PVR (Parisi et al., 2022), MVP (Xiao et al., 2022), R3M (Nair et al., 2022)). These three visual representations were proposed concurrently and represent the present state-of-the-art in pre-training for visuo-motor control. *We choose to freeze the pre-trained representations to be consistent with their original formulations*. The three pre-trained representations that we consider have been shown to outperform common representations such as supervised learning and MoCo-v2 (He et al., 2020) pre-training on ImageNet (Russakovsky et al., 2015). In the following, we provide a more detailed description of each pre-trained representation, as well as our proposed LfS baseline. See Table 1 for an overview of the three pre-trained representations that we consider.

Table 1. **Overview of frozen pre-trained representations.** We summarize key design choices for each of the three pre-trained representations proposed in prior work, as well as which algorithm they considered in downstream tasks. *Jitter* denotes whether color jitter augmentation was applied during pre-training; this detail pertains to our visual robustness experiments in Section 4.

		Pre-training			Policy
Method	Repr.	Encoder	Dataset	Jitter	Algo.
● PVR	MoCo-v2	ResNet50	ImageNet	✓	BC
● MVP	MAE	ViT-S	HOI	✗	PPO
● R3M	Multi-loss	ResNet50	Ego4D	✗	BC

● **PVR** investigates the efficacy of frozen pre-trained representations for Behavior Cloning (BC) in a variety of control tasks, and proposes a variant of Momentum Contrast (MoCo-v2; (He et al., 2020)) that fuses features from multiple layers. Specifically, the PVR model is a MoCo-v2 representation with a ResNet50 (He et al., 2015) backbone trained on ImageNet (Russakovsky et al., 2015), with intermediate layers fused together with final output features via a

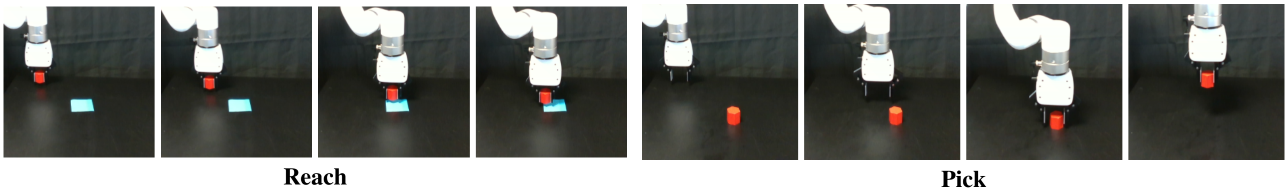


Figure 3. **Real robot tasks.** Sample trajectories for each of the two real robot tasks that we consider, *reach* and *pick*. Visualizations correspond to raw RGB observations at key frames. Actual episode length is 50. Trajectories are generated using ● LfS (+aug) with 10 and 20 demonstrations, respectively, collected by a human teleoperator. We evaluate methods on 20 trials per task and across 2 random seeds.

second finetuning stage. Combining features from early and later layers of the network encourages the PVR model to retain spatial granularity as well as scene-level semantic information. The PVR representation is trained on individual frames without leveraging temporal-sequential information. During pre-training, PVR applies random crop, horizontal flip, gray-scale, and color jitter augmentations. We use the publicly available PVR model in our experiments.

● **MVP** concurrently studies the efficacy of frozen pre-trained representations for on-policy RL using Proximal Policy Optimization (PPO; (Schulman et al., 2017)), and propose to train a Masked Autoencoder (He et al., 2022) visual representation on individual frames from a large (700K frames) human interaction dataset (referred to as *HOI*) sourced from multiple existing datasets. Concretely, the MVP model uses a Vision Transformer (ViT; (Dosovitskiy et al., 2021)) backbone that partitions frames into 16×16 patches. The MVP representation is trained on individual frames without leveraging temporal-sequential information. The parameter count of the MVP encoder (ViT-S; 22M) is comparable to that of PVR and R3M. During pre-training, MVP applies random crop and horizontal flip augmentations. We use the publicly available MVP model in our experiments[2].

● **R3M** proposes to pre-train a ResNet50 backbone using a combination of time-contrastive learning (Sermanet et al., 2016), video-language alignment, and L1 regularization that encourages sparse and compact representations, on 3,500 hours of human interaction video data from the Ego4D dataset (Grauman et al., 2022). In contrast to PVR and MVP, R3M *does* leverage the temporal-sequential nature of video data. During pre-training, R3M only applies random crop augmentations. We use the publicly available R3M model in our experiments.

We evaluate these three pre-trained representations – PVR, MVP, and R3M – against our simple yet strong LfS baseline that uses data augmentation and a shallow ConvNet encoder. To demonstrate the importance of data augmentation in representation learning for control, we include two LfS baselines – with and without use of data augmentation – which we describe in the following.

● **LfS** *(no aug)* uses a shallow ConvNet encoder that consists of 4-6 layers (depending on the experimental setup in which it is applied) of 2D-convolutions with ReLU activations. Each of our three encoder implementations are adopted from prior work and are widely accepted by the RL community. Specifically, the LfS encoder in our BC experiments is identical to that of PVR (Parisi et al., 2022), the LfS encoder that we use for off-policy RL is equivalent to that of Yarats et al. (2019); Srinivas et al. (2020); Laskin et al. (2020); Kostrikov et al. (2021); Yarats et al. (2021), and our on-policy RL LfS baseline is identical to that of Hansen et al. (2022a).

● **LfS** *(+aug)* uses an architecture identical to that of ●LfS. However, it is well documented in literature on visual RL that use of data augmentation is critical to the performance and visual robustness of LfS (Laskin et al., 2020; Kostrikov et al., 2021; Yarats et al., 2021; Hansen & Wang, 2021; Raileanu et al., 2020; Hansen et al., 2021; Yuan et al., 2022). To accurately reflect progress in LfS approaches, our main point of comparison is a LfS method that uses random shift augmentation (Kostrikov et al., 2021) in addition to its shallow ConvNet encoder, which has demonstrated strong empirical performance on a variety of task domains. As our experiments reveal, use of data augmentation is also surprisingly effective for learning behavior cloning policies, although it is not commonly used in this setting.

The reader is referred to our respective experimental setups in Section 3 for a per-algorithm description of our proposed LfS baselines.

3. Experimental setup

We propose a set of strong LfS baselines that span **3** classes of algorithms: imitation learning (behavior cloning), on-policy RL (PPO (Schulman et al., 2017)), and off-policy RL (DrQ-v2 (Yarats et al., 2021)), and consider a total of **17** tasks across **4** domains: Adroit (Rajeswaran et al., 2018) (dexterous manipulation; 2 tasks × 2 views), DMControl (Tassa et al., 2018) (locomotion and control; 5 tasks),

[2] We acknowledge that a newer set of MVP models have been released concurrently with our work. While improved downstream task performance can be anticipated for the new models, we expect our main conclusions to remain unchanged.

PixMC (Xiao et al., 2022) (robotic manipulation; 8 tasks), and a real robot setup (robotic manipulation; 2 tasks). Figure 2 shows sample tasks from each domain; see Appendix A for a detailed description of all tasks. Sample trajectories for each of the two real robot tasks are shown in Figure 3. Importantly, we do *not* propose a new benchmark for pre-trained representations, but rather base our experiments on the public implementations of PVR, MVP, and DrQ-v2, and *meticulously follow their respective experimental setups*. We make no changes to hyperparameters. This strict experimental setup ensures that pre-trained representations are evaluated in favorable conditions (for which they were originally proposed). Our code is available at `https://github.com/gemcollector/learning-from-scratch`. We provide the full details of our experimental setup in the appendices, and summarize it as follows:

— **Behavior Cloning (BC).** We consider two simulation domains – Adroit and DMControl – used in PVR, in addition to our real robot setup. Observations are 256×256 RGB images (center-cropped to 224×224) with no access to proprioceptive information. In simulation, policies are trained with BC on 100 demonstrations per task; we use the exact demonstration dataset that PVR used[3], *i.e.*, Adroit demonstrations are generated by oracle (state-based) DAPG (Rajeswaran et al., 2018) policies, and DMControl demonstrations are generated by oracle DDPG (Lillicrap et al., 2016) policies. We use 10-20 demonstrations in the real world depending on the task, but otherwise follow the same experimental setup as in simulation. The original LfS baseline in PVR uses a shallow ConvNet encoder; we refer to this baseline simply as ●*LfS*. Our improved LfS baseline additionally uses random shift augmentation (Kostrikov et al., 2021; Yarats et al., 2021) during learning, and we refer to this baseline as ●*LfS (+aug)*. Data augmentation is relatively underexplored in BC literature, but we find that it works surprisingly well. In addition to PVR, we also compare with frozen MVP and R3M representations. Consistent with the experimental setup in PVR, we measure the policy performance with success rate in the case of Adroit (and our real setup), and episode return in DMControl. Policies are evaluated every two epochs for a total of 100 epochs in simulation, and we report the average performance over the 3 best epochs over the course of learning. We find that 1 epoch is sufficient for our real robot experiments, where we evaluate for 20 trials per method per task, across 2 random seeds.

— **On-policy RL.** We reproduce the results of MVP on their proposed PixMC robotic manipulation benchmark. Observations are 224×224 RGB images and also include proprioceptive information. The original LfS baseline uses a small ViT (Dosovitskiy et al., 2021) encoder. We propose *two* improved LfS baselines for this setting: *(1)* an LfS baseline that uses a shallow ConvNet encoder and *no* data augmentation, referred to as ●*LfS*, and *(2)* an LfS baseline that additionally applies random shift augmentation in critic learning, referred to as ●*LfS (+aug)*. Following prior work (Hansen et al., 2021; Raileanu et al., 2020), we do not augment value targets. In addition to (frozen) MVP, we also compare with frozen PVR and R3M representations. Following the setup in MVP, we use the success rate of the policy as the metric for comparison.

Table 2. **Behavior Cloning: LfS vs. frozen pre-trained visual representations.** Success rate (Adroit, real robot) and unnormalized return (DMControl) of LfS and the **best** result obtained with a pre-trained representation, *i.e.*, for each task we report max{PVR, MVP, R3M}. A well-designed LfS method is competitive with frozen pre-trained representations across all tasks.

Method **Task**	● LfS (*no aug*)	● LfS (*+aug*)	**Best** pre-training
Adroit			
Pen	22.0±4.0	74.8±5.0	**81.3±4.0**
Relocate	16.9±3.5	**51.4±7.7**	47.5±2.6
DMControl			
Finger Spin	647.6±6.9	661.4±22.6	**698.5±8.4**
Reacher Easy	261.3±27.6	**657.4±44.3**	615±27.0
Cheetah Run	469.8±30.0	448.9±56.4	**557.6±18.4**
Walker Stand	699.0±65.0	**875.5±20.4**	818.2±19.4
Walker Walk	699.4±15.2	**791.6±17.8**	788.0±10.2
Real robot			
Reach	80.0±0.0	85.0±5.0	**90.0±10.0**
Pick	25.0±5.0	**55.0±5.0**	35.0±15.0

— **Off-policy RL.** We reproduce the results of the state-of-the-art LfS method DrQ-v2 on the same DMControl tasks as used in PVR. Observations are 84×84 RGB images with no access to proprioceptive information; we upsample observations to 224×224 when using pre-trained representations. DrQ-v2 uses a shallow ConvNet encoder and random shift augmentation by default, and we refer to this baseline as ●*LfS (+aug)*. We compare DrQ-v2 to two alternatives: *(1)* not using data augmentation (simply denoted ●*LfS*), and *(2)* removing data augmentation **and** additionally replacing the LfS encoder with a frozen pre-trained representation, denoted by their representation names (PVR, R3M, MVP) respectively. Following prior work on DMControl, we use (normalized) return as the metric for comparison.

4. Results

In this section, we present a clear summary of our key experimental results, and defer deeper discussion on the implications of these findings (along with practical guidance for practitioners) to Section 5. Our results are as follows:

[3] While the demonstration dataset used in PVR is not publicly available, the authors kindly provided us with the demonstrations in response to our private inquiry. We thank the authors for that.

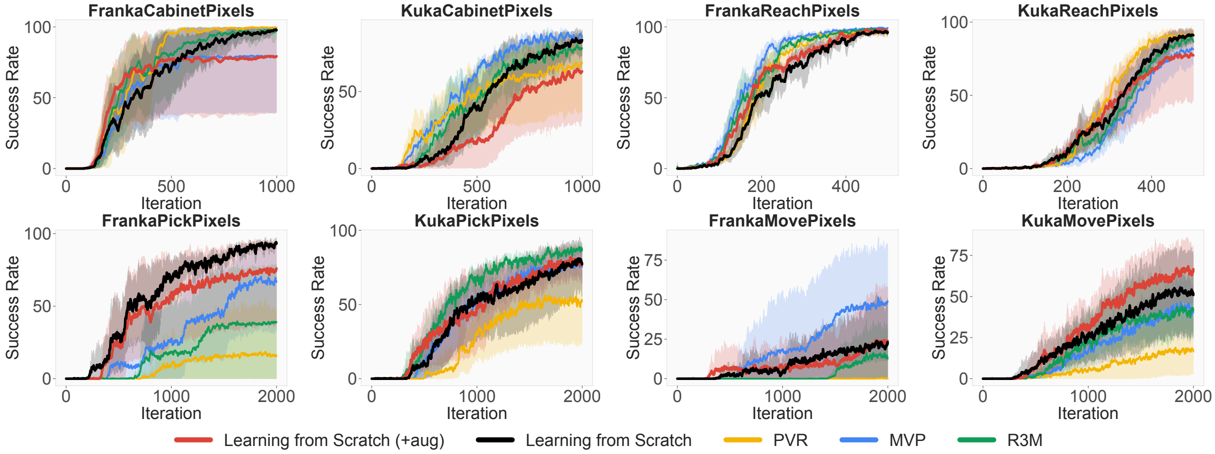
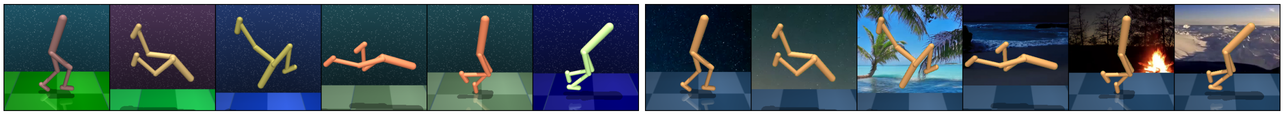


Figure 4. **PixMC benchmark.** Success rate of PPO (Schulman et al., 2017) agents on the 8 robotic manipulation tasks from PixMC (Xiao et al., 2022). Our proposed LfS baseline performs comparably to the frozen pre-trained visual representations on most tasks. Notably, we also observe that no single pre-trained representation is consistently better across all tasks. Results are averaged across 5 seeds.



(a) Randomized colors.

(b) Video backgrounds.

Figure 5. **Evaluation of robustness**. We quantify robustness to visual changes on two test domains from DMControl Generalization Benchmark (Hansen & Wang, 2021): *randomized colors* of agent, floor, and background, and dynamic *video backgrounds* sourced from out-of-domain data, corresponding to the `color_hard` and `video_easy` test domains from the proposed benchmark. Sample environments are visualized. Note that a domain gap remains between augmented observations and test environments.

— **Performance comparison.** Our proposed Learning-from-Scratch (LfS) baselines are competitive with (and in some cases *outperform*) recent frozen pre-trained representations for visuo-motor control across a variety of algorithms and domains in both simulation and the real world; see Figure 1 and Table 2. This indicates that, while pre-trained representations have the potential to replace the LfS paradigm in the future, under the set of most widely used metrics, they have yet to exceed the representational power of a *well-designed* LfS method on standard benchmarks for visuo-motor control. This conclusion appears to generalize to real robot tasks with simple visuals.

— **No free lunch – yet.** Our results indicate that the efficacy of a frozen pre-trained representation is both *task-dependent* (see Figure 4) and *algorithm-dependent* (see Figure 1): on average, ●MVP outperforms other pre-trained representations on PixMC for which it was originally proposed, but performs comparably worse on the two other domains, Adroit and DMControl. However, even within a visually consistent benchmark (PixMC), no single representation convincingly comes out on top across tasks, as evidenced by Figure 4. In contrast, our proposed ●*LfS (+aug)* method produces consistently strong results across all settings, presumably due to learning from task-specific data; this hypothesis is supported by our finetuning results, which we return to later.

— **Visual robustness.** To probe representations for visual robustness, we evaluate trained agents on the DMControl Generalization Benchmark (Hansen & Wang, 2021). In this evaluation, agents are trained on the original training environments with no visual variation, and transferred zero-shot to test environments with visual changes. We consider two types of visual changes: *(i) random colors* where the colors of agent, background, and floor are randomized, and *(ii) video backgrounds* where the background is replaced with a dynamically changing texture from out-of-domain videos; see Figure 5 (Appendix) for a visualization of these test environments. Our robustness results are shown in Figure 6. We find that *use of data augmentation is critical to the robustness of learned visual representations* – both when LfS and when using frozen pre-trained representations. Notably – in their original formulations – PVR uses color jitter during pre-training whereas MVP and R3M do not. We compare robustness of these pre-trained representations with LfS using both ●*random shift* augmentation and additionally ●*color jitter*. We find that *(1)* pre-trained representations that do *not* use color jitter augmentation during pre-training (●MVP, ●R3M) are not more robust than their LfS counterpart to visual changes, but *(2)* strong augmentations such as ●*color jitter* improves robustness of *both* LfS and pre-trained representations (●PVR; applied during pre-training) significantly. For completeness, we also evaluate LfS with a different

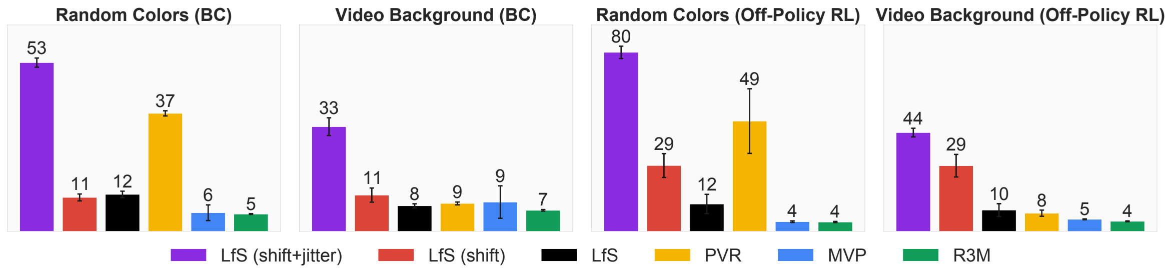


Figure 6. **Robustness to visual changes.** Normalized return of methods when transferred to environments with visual changes from the DMControl Generalization Benchmark (Hansen & Wang, 2021). We consider two visual changes: randomized colors of agent and scene, as well as dynamic video backgrounds. Following our previous setup, BC results are averages of top-3 evaluations over 100 epochs, and final evaluations are reported for RL results. Mean and 95% confidence intervals over 5 seeds and 4 tasks; we omit *Reacher Easy* since it does not support video backgrounds. LfS with strong augmentation is surprisingly robust compared to frozen pre-trained representations.

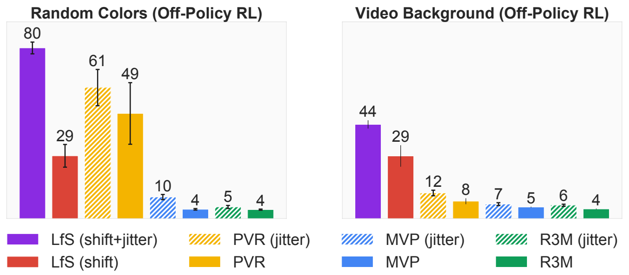


Figure 7. **Improving robustness of frozen pre-trained representations with strong augmentation.** Normalized return of methods when transferred to environments with visual changes from the DMControl Generalization Benchmark (Hansen & Wang, 2021). We report results both with and without additional ●*color jitter* (strong) augmentation during policy learning, and find that applying strong augmentation with a *frozen* representation is ineffective. Mean and 95% confidence intervals over 5 seeds and 4 tasks; we omit *Reacher Easy* since it does not support video backgrounds.

Table 3. **Choice of augmentation matters.** Mean normalized return of BC policies when transferred to environments with visual changes from the DMControl Generalization Benchmark (Hansen & Wang, 2021). We here consider LfS with two distinct choices of strong data augmentation: *color jitter* as in Figure 6, and *random overlay* originally proposed by Hansen & Wang (2021); these augmentations are in addition to random image shifts. For completeness, we also include our **best** result obtained with a pre-trained representation, *i.e.*, we report $\max\{\text{PVR}, \text{MVP}, \text{R3M}\}$.

Method	● LfS	● LfS	**Best**
Test set	(*jitter*)	(*overlay*)	pre-training
Random colors	**53.1**±1.6	39.3±0.2	37.2±0.9
Video background	33.0±3.2	**46.6**±1.3	9.2±5.8

choice of strong augmentation: ●*random overlay* that interpolates between observations and randomly sampled images from an out-of-domain dataset, popularized by Hansen & Wang (2021); results are shown in Table 3. Consistent with prior work, we find that choice of augmentation influences robustness to different visual changes, but that *either choice beats the best pre-trained representation* that we consider.

— **Finetuning a pretrained representation.** To help narrow the scope of our comparison, our study primarily considers frozen visual representations following their original proposals, *i.e.*, neither PVR, MVP, or R3M finetune their representations on in-domain data. However, there is *some* existing evidence that in-domain finetuning of pre-trained representations can be beneficial (Wang et al., 2022; Ze et al., 2022; Xu et al., 2022). For completeness, we also conduct a set of finetuning experiments, where pre-trained representations (●PVR, ●MVP, ●R3M) are finetuned on demonstration data from Adroit using the task-centric behavior cloning objective. Results for this experiment are shown in Table 5. Interestingly, we find that finetuned representations can improve over both their frozen counterparts and our ●*LfS (+aug)* approach, but *only when also using data augmentation* (random shift) during finetuning. This observation indicates that data augmentation is critical to performance when learning on a small (by comparison) in-domain dataset, *regardless* of whether the representation is learned from scratch or finetuned. We are – to the best of our knowledge – the first to make this observation, and conjecture that this discrepancy in performance is due to a domain gap between out-of-domain training data and in-domain data. Given that our finetuning experiments are in simulation whereas the pre-training data consists of real-world images, we dub this phenomenon the *real-to-sim* gap. However, our real robot results also indicate that this gap persists to some extent even when evaluating in the real world. Lastly, we also find that ResNet-based representations (●PVR, ●R3M) are easier to finetune than ViT (●MVP), presumably due to known optimization challenges in ViTs (Dosovitskiy et al., 2021; Chen et al., 2021; Hansen et al., 2021). We consider frozen visual representations in the remainder of our experiments, but provide further discussion on the potential implications of this observation in Section 5.

— **Data efficiency.** A common argument in favor of (frozen) pre-trained representations is that they might require less task-specific data to learn a good policy (Parisi et al., 2022; Xiao et al., 2022; Nair et al., 2022). To test this hypothesis, we train BC policies with a variable num-

Table 4. **Wall-time** of methods learning from scratch vs. using a pre-trained visual representation. For the latter, we report $\min\{\text{PVR}, \text{MVP}, \text{R3M}\}$ for a fair comparison. While LfS generally leads to better downstream task performance, using a **frozen** pre-trained representation can reduce computational cost substantially, especially during the training process. ↓ Lower is better.

	Behavior Cloning				**Reinforcement Learning**	
	Training (s/iteration)		Inference (s/episode)		s/1k frames	s/iteration
Method\Setting	Adroit	DMControl	Adroit	DMControl	DrQ-v2	PPO
● LfS (+aug)	0.263	0.270	**1.61**	**3.81**	**10.20**	19.40
Fastest pre-training	**0.003**	**0.006**	2.66	11.00	13.00	**11.90**

Table 5. **Finetuning pre-trained representations with BC.** Success rate for each method across 5 seeds and all Adroit tasks. *Finetuned* denotes whether a representation has been finetuned on task data, and *data aug* denotes whether random image shift augmentation is applied during finetuning. We find that finetuning ResNet-based representations (PVR, R3M) on task data improves over frozen representations and even outperforms LfS (+aug), but *only when using data augmentation during finetuning*. We are – to the best of our knowledge – the first to make this observation.

Method	Finetuned	Data aug	Success (%)	Change
● PVR	✗	✗	52.9±2.1	–
	✓	✗	50.5±7.7	**−2.4**
	✓	✓	65.0±0.0	**+12.1**
● MVP	✗	✗	44.0±2.2	–
	✓	✗	18.7±1.9	**−25.3**
	✓	✓	31.1±2.7	**−12.9**
● R3M	✗	✗	64.0±2.8	–
	✓	✗	55.5±8.9	**−8.5**
	✓	✓	**80.5±2.1**	**+16.5**
● LfS	✓	✗	19.4±3.8	–
● LfS *(+aug)*	✓	✓	63.1±6.4	**+43.7**

ber of demonstrations (10, 25, 100) for both our improved LfS baseline and the three frozen pre-trained representations; following the experimental setup of PVR, we use 100 demonstrations in the remainder of our BC experiments. We report the results of this experiment in Figure 8. Our results indicate that a larger number of demonstrations (100) generally favors LfS methods, whereas frozen pre-trained representations fare marginally better in the very low-data regime (10). However, policy performance degrades quickly with a decrease in available demonstrations, which suggests that the primary performance bottleneck is in policy learning rather than visual representation learning. As discussed in Section 5, this may in part be due to current benchmarks being visually simple. We predict that this observation might not continue to hold true as new simulation benchmarks with more complex visuals are developed.

— **Computational cost.** Our results so far have focused on downstream task performance, *i.e.*, success rate or return in various settings. However, frozen pre-trained representations already demonstrate significant gains along an often-neglected axis: *wall-time*. Training and inference speeds are shown in Table 4. We find that BC policy updates are at least an order of magnitude faster using frozen pre-trained representations compared to LfS, as we can embed and cache features for the entire dataset in a few forward passes. However, inference speed generally favors LfS due to their smaller visual backbones, which is particularly important for real robot applications. Since RL training interleaves learning and inference (data collection), wall-times are more balanced in this setting. We do not factor in the cost of learning a pre-trained representation, since it is a one-time cost, and the representations can be reused across tasks.

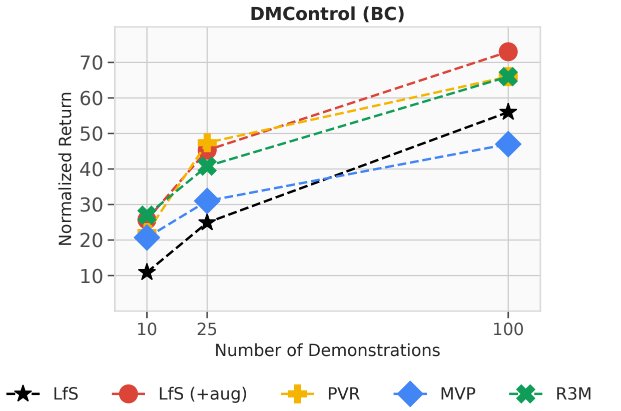


Figure 8. **Data efficiency.** Normalized return of behavior cloning policies as a function of the number of demonstrations. Results are averaged across all of our DMControl tasks and 3 seeds. We find that a larger amount of demonstrations (100) favors LfS, whereas pre-trained representations fare better in the very low-data regime.

5. Discussion

We have shown that a carefully designed LfS baseline is competitive with frozen pre-trained representations across a variety of algorithm classes, domains, and metrics. While this is the current conclusion, we remain optimistic that results will be skewed in favor of pre-trained representations as the paradigm matures. At present, we find that the main benefit of a frozen pre-trained representation is the reduced training cost that comes with its *universality* – a single representation can be reused across tasks. Achieving a performance edge while maintaining universality will thus be critical to the adoption of this new paradigm. Our experiments indicate that pre-trained representations benefit from finetuning on task-specific data (when coupled with use of data augmentation), combining elements of pre-training and LfS. However, finetuning large visual backbones presents optimization challenges (*e.g.*, catastrophical forgetting and instability), and can be costly. In the following, we share our vision for the future of pre-training research for control, which we hope can inspire further research in the area. However, we remark that this vision – while being informed by findings in this work – is ultimately a conjecture.

— A benchmark perspective. Our first conjecture is that current benchmark tasks are not well suited to reap the benefits of pre-trained representations. Historically, the majority of visual RL benchmarks have been repurposed from existing environments that were originally proposed for RL from ground-truth (state) information, with little emphasis on visual complexity, variation, and realism. Furthermore, there have historically been strong emphasis on single-task learning, where limited semantic information is required. In such (visually) simple settings, it is perhaps not surprising that learning a representation from scratch on in-domain data is sufficient and oftentimes better than a general-purpose representation trained solely on out-of-domain ImageNet or human interaction data. We predict that pre-trained representations (frozen and finetuned alike) will fare better as new benchmarks with these properties – visual complexity, variation, realism, and multi-task learning – are developed. To this end, we view evaluation of policies in the real world – such as those shown in Figure 3 – as a step in the right direction. However, most contemporary real robot setups (for which ours is no exception) still leave much to be desired in terms of visual complexity and variation compared to the diversity of the pre-training data leveraged in research on pre-trained representations for control.

— A domain gap perspective. Our second conjecture is directly informed by our experiments. We observe that, while LfS consistently outperforms current frozen pre-trained representations, finetuning on in-domain data (with the addition of data augmentation) results in representations that achieve better downstream task performance and are more robust to visual variations compared to their frozen counterparts and in some cases even LfS. This important result suggests that this discrepancy is due to a large domain gap between pre-training data and in-domain data. Given that current pre-trained representations are learned from out-of-domain ImageNet or human interaction data (*i.e.*, real-world data) and predominantly tested in simulated robot environments (distinctly different domains), it is perhaps not surprising that finetuning representations on a small amount of in-domain data can lead to markedly better downstream task performance. We dub this domain gap the *real-to-sim* gap, although we empirically find that the problem persists to some extent even in real robot experiments. We recommend future work to either *(1)* pre-train on data that better reflect the data distribution of downstream tasks (*e.g.*, by training on simulation data or real-world robot data, or by evaluating policies in real world scenes that more closely match those present in existing pre-training datasets), or *(2)* finetune on a small in-domain dataset using, *e.g.*, a task-centric objective such as BC (if demonstrations are available) or RL (online interaction). While addressing the domain gap by collecting a new dataset for pre-training can be costly, it is relatively easy to finetune current models on a small in-domain dataset; this is reminiscent to the current trend of aligning large language models by finetuning on small curated datasets (Taori et al., 2023). Lastly, our experiments demonstrate that data augmentation is absolutely critical to learning strong, robust representations in all stages of training. To the best of our knowledge, we are the first work on pre-trained representations for control to make this discovery. In comparison to the refined training recipes in computer vision literature, training recipes for visuo-motor control are still relatively underexplored. We predict that – as pre-processing and data augmentation training recipes for visuo-motor control mature – we will see a series of increasingly robust pre-trained representations emerge. We encourage further research in all of these directions, and hope that our LfS baselines will help accurately benchmark progress in this area.

6. Related Work

Pre-training. Representation learning via supervised/self-supervised/unsupervised pre-training on large-scale datasets has emerged as a powerful paradigm in areas such as computer vision (Doersch et al., 2015; He et al., 2020; van den Oord et al., 2018; Alayrac et al., 2022) and natural language processing (Devlin et al., 2019; Brown et al., 2020; Radford et al., 2021; Chowdhery et al., 2022), where large datasets are available. While pre-trained representations can be finetuned to solve various downstream tasks, it may be prohibitively expensive to do so, and representations are therefore commonly used as-is, *i.e.*, with *frozen* weights. We reflect on recent progress and challenges when leveraging pre-trained visual representations for control, which is an emerging and comparably underexplored application area of such representations.

Pre-trained representations for control. Multiple works have explored learning control policies with visual representations pre-trained on large external datasets (Shah & Kumar, 2021; Parisi et al., 2022; Nair et al., 2022; Xiao et al., 2022; Wang et al., 2022; Ze et al., 2022; Yuan et al., 2022; Xu et al., 2022; Brohan et al., 2022). In particular, PVR (Parisi et al., 2022) and R3M (Nair et al., 2022) propose to learn policies by behavior cloning using pre-trained representations; PVR fuses features from several layers of a ResNet50 learned by MoCo-v2 (He et al., 2020), and R3M (Nair et al.,

2022) learns a representation using a time-contrastive objective on ego-centric human videos. MVP (Xiao et al., 2022) learns a policy with PPO (Schulman et al., 2017) and uses a pre-trained visual encoder for feature extraction in addition to proprioceptive state information; the pre-trained representation is an MAE (He et al., 2022) trained on frames from diverse human videos. We show that our improved LfS baseline remains competitive with (frozen) pre-trained representations, but also find that an equally carefully designed *finetuning* procedure of pre-trained representations can outperform LfS in some cases.

Data augmentation in RL. Numerous recent studies demonstrate the effectiveness of data augmentation in visual RL (Lee et al., 2019; Laskin et al., 2020; Raileanu et al., 2020; Kostrikov et al., 2021; Yarats et al., 2021; Hansen & Wang, 2021; Hansen et al., 2021; Ma et al., 2022; Hansen et al., 2022b). For example, Lee et al. (2019); Hansen & Wang (2021) show that strong data augmentation can greatly improve the visual robustness and generalization of RL policies. Domain randomization (Tobin et al., 2017; Pinto et al., 2017), a closely related idea, has similarly been shown to improve generalization and sim-to-real transfer. Laskin et al. (2020) conducts a comprehensive study on data augmentations for RL, and finds that random crops can lead to significant gains in both sample-efficiency and asymptotic performance. Finally, Kostrikov et al. (2021); Yarats et al. (2021) propose a simple *random shift* augmentation that further improves over random crop augmentation in the context of visual off-policy RL; we apply this augmentation in all of our ●*LfS (+aug)* experiments. Our study confirms the observations of prior work, and shows that the resulting LfS baselines remain competitive with frozen pre-trained representations trained on large-scale out-of-domain datasets.

7. Conclusion

To conclude, we reiterate the main takeaways of our study:

— A *carefully designed* LfS approach remains competitive with frozen pre-trained representations across a variety of algorithms, task domains, and evaluation metrics.

— At this time, no single frozen pre-trained representation is consistently better across all tasks.

— Finetuning pre-trained representations on task-specific data leads to significant improvements in performance (when also using data augmentation during finetuning), even surpassing the performance of ●LfS *(+aug)* in some cases.

— LfS with strong data augmentation (●*color jitter*) outperforms frozen pre-trained representations by a large margin on visual robustness benchmarks. However, adding strong data augmentation to pre-training and policy learning pipelines consistently improves their robustness.

— Pre-trained representations fare slightly better than LfS approaches in the very low-data regime, but our experiments indicate that policy learning might be a bigger bottleneck when data is limited.

— Using *frozen* pre-trained representations can lead to significant improvements in training wall-time, at the expense of slower inference compared to our smaller LfS backbones. Since RL training interleaves learning and inference (data collection), wall-times are more balanced in this setting.

— We remain optimistic about the future of pre-trained representations for visuo-motor control, and hope that our strong LfS baselines will help accurately benchmark progress in the area.

Acknowledgements Experiments were divided equally between NH, ZY, YZ, and TM. AR, HS, HX, and XW advised equally. All authors contributed to writing. The authors would like to thank Ziyan Xiong for their help with conducting real robot experiments for the camera-ready version of the paper. Prof. XW's lab is supported, in part, by an Amazon Research Award and gifts from Qualcomm.

References

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022.

Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Dabis, J., Finn, C., Gopalakrishnan, K., Hausman, K., Herzog, A., Hsu, J., Ibarz, J., Ichter, B., Irpan, A., Jackson, T., Jesmonth, S., Joshi, N., Julian, R., Kalashnikov, D., Kuang, Y., Leal, I., Lee, K.-H., Levine, S., Lu, Y., Malla, U., Manjunath, D., Mordatch, I., Nachum, O., Parada, C., Peralta, J., Perez, E., Pertsch, K., Quiambao, J., Rao, K., Ryoo, M., Salazar, G., Sanketi, P., Sayed, K., Singh, J., Sontakke, S., Stone, A., Tan, C., Tran, H., Vanhoucke, V., Vega, S., Vuong, Q., Xia, F., Xiao, T., Xu, P., Xu, S., Yu, T., and Zitkovich, B. Rt-1: Robotics transformer for real-world control at scale. In *arXiv preprint arXiv:2212.06817*, 2022.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T. J., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. *ArXiv*, abs/2005.14165, 2020.

Chen, X., Hsieh, C.-J., and Gong, B. When vision transformers outperform resnets without pretraining or strong data augmentations. *ArXiv*, abs/2106.01548, 2021.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805, 2019.

Doersch, C., Gupta, A. K., and Efros, A. A. Unsupervised visual representation learning by context prediction. *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1422–1430, 2015.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2021.

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

Hansen, N. and Wang, X. Generalization in reinforcement learning by soft data augmentation. In *International Conference on Robotics and Automation (ICRA)*, 2021.

Hansen, N., Su, H., and Wang, X. Stabilizing deep q-learning with convnets and vision transformers under data augmentation. In *NeurIPS*, 2021.

Hansen, N., Lin, Y., Su, H., Wang, X., Kumar, V., and Rajeswaran, A. Modem: Accelerating visual model-based reinforcement learning with demonstrations. *arXiv preprint*, 2022a.

Hansen, N., Wang, X., and Su, H. Temporal difference learning for model predictive control. In *ICML*, 2022b.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2015.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. B. Momentum contrast for unsupervised visual representation learning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9726–9735, 2020.

He, K., Chen, X., Xie, S., Li, Y., Doll'ar, P., and Girshick, R. B. Masked autoencoders are scalable vision learners. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15979–15988, 2022.

Kostrikov, I., Yarats, D., and Fergus, R. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *ArXiv*, abs/2004.13649, 2021.

Laskin, M., Lee, K., Stooke, A., Pinto, L., Abbeel, P., and Srinivas, A. Reinforcement learning with augmented data. *ArXiv*, abs/2004.14990, 2020.

Lee, K., Lee, K., Shin, J., and Lee, H. A simple randomization technique for generalization in deep reinforcement learning. *ArXiv*, abs/1910.05396, 2019.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N. M. O., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *CoRR*, abs/1509.02971, 2016.

Ma, G., Wang, Z., Yuan, Z., Wang, X., Yuan, B., and Tao, D. A comprehensive survey of data augmentation in visual reinforcement learning. *arXiv preprint arXiv:2210.04561*, 2022.

Nair, S., Rajeswaran, A., Kumar, V., Finn, C., and Gupta, A. R3m: A universal visual representation for robot manipulation. *ArXiv*, abs/2203.12601, 2022.

Parisi, S., Rajeswaran, A., Purushwalkam, S., and Gupta, A. K. The unsurprising effectiveness of pre-trained vision models for control. In *ICML*, 2022.

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

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763, 2021.

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

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

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein,

M. S., Berg, A. C., and Fei-Fei, L. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.

Schwarzer, M., Anand, A., Goel, R., Hjelm, R. D., Courville, A. C., and Bachman, P. Data-efficient reinforcement learning with self-predictive representations. In *ICLR*, 2021.

Sermanet, P., Xu, K., and Levine, S. Unsupervised perceptual rewards for imitation learning. *ArXiv*, abs/1612.06699, 2016.

Shah, R. and Kumar, V. Rrl: Resnet as representation for reinforcement learning. *ArXiv*, abs/2107.03380, 2021.

Shang, W., Wang, X., Srinivas, A., Rajeswaran, A., Gao, Y., Abbeel, P., and Laskin, M. Reinforcement learning with latent flow. In *Neural Information Processing Systems*, 2021.

Srinivas, A., Laskin, M., and Abbeel, P. Curl: Contrastive unsupervised representations for reinforcement learning. In *ICML*, 2020.

Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P., and Hashimoto, T. B. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., de Las Casas, D., Budden, D., Abdolmaleki, A., et al. Deepmind control suite. Technical report, DeepMind, 2018.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. Domain randomization for transferring deep neural networks from simulation to the real world. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep 2017.

van den Oord, A., Li, Y., and Vinyals, O. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748, 2018.

Wang, C., Luo, X., Ross, K. W., and Li, D. Vrl3: A data-driven framework for visual deep reinforcement learning. *ArXiv*, abs/2202.10324, 2022.

Xiao, T., Radosavovic, I., Darrell, T., and Malik, J. Masked visual pre-training for motor control. *ArXiv*, abs/2203.06173, 2022.

Xu, Y., Hansen, N., Wang, Z., Chan, Y.-C., Su, H., and Tu, Z. On the feasibility of cross-task transfer with model-based reinforcement learning. *arXiv preprint arXiv:2210.10763*, 2022.

Yarats, D., Zhang, A., Kostrikov, I., Amos, B., Pineau, J., and Fergus, R. Improving sample efficiency in model-free reinforcement learning from images. 2019.

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

Yuan, Z., Xue, Z., Yuan, B., Wang, X., Wu, Y., Gao, Y., and Xu, H. Pre-trained image encoder for generalizable visual reinforcement learning. *arXiv preprint arXiv:2212.08860*, 2022.

Ze, Y., Hansen, N., Chen, Y., Jain, M., and Wang, X. Visual reinforcement learning with self-supervised 3d representations. *arXiv preprint arXiv:2210.07241*, 2022.

A. Task Descriptions

We conduct experiments on three different task domains in simulation – Adroit (Rajeswaran et al., 2018), DMControl (Tassa et al., 2018), and PixMC (Xiao et al., 2022)– used in prior work on pre-training for visuo-motor control, as well as a real robot setup. PVR (Parisi et al., 2022) experiments with Adroit and DMControl, MVP (Xiao et al., 2022) proposed the PixMC benchmark, and R3M (Nair et al., 2022) experiments with Adroit (among others). To make our study more self-contained, we include a detailed description of each task below.

A.1. Adroit

Following PVR, we consider two tasks from the Adroit domain: *pen* and *relocate*, which represent the two most challenging tasks from this task domain. The two Adroit tasks are goal-conditioned dexterous manipulation tasks with goals rendered visually in a 3D scene, as shown in Figure 2 *(left)*. The robot hand has 24 degrees of freedom (DoF). We describe each task as follows:

- *Pen* ($\mathcal{A} \in \mathbb{R}^{18}$). A blue pen is initialized in the palm of the dexterous robot hand. The task is to reorient the pen in-hand to bring it to a desired orientation, which is visualized as a transparent pen floating next to the hand. The agent controls all joints but its wrist is locked and cannot move in 3D space.
- *Relocate* ($\mathcal{A} \in \mathbb{R}^{21}$). A blue ball is initialized at a random location on a table. The task is to pick up the ball using the dexterous robot hand, and move it to a desired (randomly selected) location in 3D space, which is visualized as a transparent green ball. The agent controls all joints, as well as the wrist which can move freely in 3D space.

We refer to Rajeswaran et al. (2018) for additional task details.

A.2. DMControl

Following PVR, we consider five tasks from the DMControl suite: *Finger Spin*, *Reacher Easy*, *Cheetah Run*, *Walker Stand*, and *Walker Walk*, which represent continuous control tasks of varying difficulty. These tasks vary in embodiment, objective, action space, and reward type. Two of the DMControl tasks (*Finger Spin* and *Walker Walk*) are visualized in Figure 2 *(center left)*. We describe each task as follows:

- *Finger Spin* ($\mathcal{A} \in \mathbb{R}^{2}$). A simple manipulation task with a planar 3 DoF finger. The task is to continuously spin a free-floating body at high velocity. There is a positive reward of $+1$ for each timestep that the body is spinning and 0 otherwise.
- *Reacher Easy* ($\mathcal{A} \in \mathbb{R}^{2}$). A simple manipulation task with a planar 3 DoF finger. The task is to move the fingertip to a randomly selected location in 2D space. There is a positive reward of $+1$ for each timestep that the fingertip is near the target and 0 otherwise.
- *Cheetah Run* ($\mathcal{A} \in \mathbb{R}^{6}$). A locomotion task with a planar cheetah embodiment. The task is to run forward at high velocity until the end of the episode. There is a dense (shaped) reward that varies with forward velocity and positioning of joints.
- *Walker Stand* ($\mathcal{A} \in \mathbb{R}^{6}$). A locomotion task with a planar Walker embodiment. The task is to stand up until the end of the episode. There is a dense (shaped) reward that varies with the positioning of joints.
- *Walker Walk* ($\mathcal{A} \in \mathbb{R}^{6}$). A locomotion task with a planar Walker embodiment. The task is to walk forward at medium velocity until the end of the episode. There is a dense (shaped) reward that varies with forward velocity and positioning of joints.

We refer to Tassa et al. (2018) for additional task details.

A.3. PixMC

Following MVP, we consider all eight tasks from the proposed PixMC benchmark. These tasks consist of four robot manipulation tasks (*Cabinet*, *Pick*, *Move*, and *Reach*) across two 7-DoF robots (*Franka* Emika and *Kuka* LBR iiwa) that use a mounted parallel jaw gripper and multi-finger hand, respectively. Observations are captured by a wrist-mounted camera. Besides embodiment and action space, these tasks also vary in interaction type and difficulty, and there is variability in objects and locations between each episode. Two of the PixMC tasks (*Kuka Pick* and *Franka Move*) are shown in Figure 2 *(center right)*. We describe each task as follows:

- *Reach* (Franka: $\mathcal{A} \in \mathbb{R}^9$, Kuka: $\mathcal{A} \in \mathbb{R}^{23}$). A simple manipulation task. The task is to reach a randomly selected location in 3D space with the end-effector. There is a dense (shaped) reward.
- *Cabinet* (Franka: $\mathcal{A} \in \mathbb{R}^9$, Kuka: $\mathcal{A} \in \mathbb{R}^{23}$). A complex articulated object manipulation task. The task is open the top drawer of a free-standing cabinet. There is a dense (shaped) reward.
- *Pick* (Franka: $\mathcal{A} \in \mathbb{R}^9$, Kuka: $\mathcal{A} \in \mathbb{R}^{23}$). An object manipulation task. The task is to pick up a randomly initialized object from the table, and hold it above a certain height threshold. There is a dense (shaped) reward.
- *Move* (Franka: $\mathcal{A} \in \mathbb{R}^9$, Kuka: $\mathcal{A} \in \mathbb{R}^{23}$). An object manipulation task. The task is to move a randomly initialized object to a different location. There is a dense (shaped) reward.

We refer to Xiao et al. (2022) for additional task details.

A.4. Real robot

In addition to our three simulation domains, we also consider two manipulation tasks on a real robot: *reach* and *pick*, which resemble the two PixMC tasks of the same name. Our experimental setup roughly mimics that of R3M. The agent controls a 7-DoF xArm 7 robot with a jaw gripper using positional control, and visual observations are captured by a static third-person Intel RealSense camera. We randomize object configuration between each episode. To minimize human bias in evaluation, we use a manually designed success detector to determine whether a given trial is successful. The two real robot tasks are visualized in Figure 2 *(right)*. We describe each task as follows:

- *Reach* ($\mathcal{A} \in \mathbb{R}^3$). A blue target is initialized at a random location within the robot workspace. The task is to move the end-effector (grasping a red object) to the target location. This task is therefore goal-conditioned. The agent controls the end-effector using positional control and its gripper is locked. We evaluate success based on the distance between the end-effector and goal at the end of a trial.
- *Pick* ($\mathcal{A} \in \mathbb{R}^4$). A red octagonal prism is initialized at a random location within the robot workspace. The task is to pick up the object using the gripper, and lift it above a predefined height threshold. The agent controls both the end-effector and gripper using positional control. We evaluate success based on a binary threshold on the end-effector height (assuming that the object is successfully grasped) at the end of a trial.

B. Implementation Details

We provide further implementation details on our improved LfS baselines in the following. For simplicity, we separate the implementation details by algorithm class, but remark that all details not pertaining to the changes that we make to the LfS baselines (shallow ConvNet encoder and data augmentation) are kept identical to prior work to ensure a fair comparison, *i.e.*, we do *not* modify the experimental setup nor hyperparameters. Our code is made available at `https://github.com/gemcollector/learning-from-scratch`.

B.1. Behavior Cloning

We closely follow the implementation of PVR (Parisi et al., 2022) for our LfS baseline in both the Adroit, DMControl, and real robot task domains. Specifically, the network consists of an encoder:

```
(0): Conv2d(3, 32, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
(1): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(2): ReLU()
(3): Conv2d(32, 32, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
(4): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(5): ReLU()
```

```
(6): Conv2d(32, 32, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
(7): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(8): ReLU()
(9): Conv2d(32, 32, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
(10): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(11): ReLU()
(12): Conv2d(32, 32, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
(13): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
(14): ReLU()
(15): Flatten(start_dim=1, end_dim=-1)
```

and a policy head:

```
(0): Linear(in_features=Z, out_features=256, bias=True)
(1): ReLU()
(2): Linear(in_features=256, out_features=256, bias=True)
(3): ReLU()
(4): Linear(in_features=256, out_features=256, bias=True)
(5): ReLU()
(6): Linear(in_features=256, out_features=A, bias=True)
```

where `Z` and `A` denote the dimensionality of the encoder output and action space, respectively. As in PVR, the encoder encodes images in a stack individually and fuses features using Flare (Shang et al., 2021) in our simulation experiments. In our real robot experiments, we find it sufficient to use a single frame. The policy has an additional 1D BatchNorm layer at the beginning when using pre-trained representations. We apply random shift augmentation to inputs (a stack of 256×256 RGB images with no access to state information) using a padding of 12 to keep the padding-to-image ratio consistent with its original proposal. All augmentations are applied to the stack consistently across time (if applicable). We use 100 expert demonstrations for each task in simulation, and 10-20 demonstrations in the real world depending on the task (reach: 10, pick: 20). Following PVR, demonstrations are collected using oracle (state-based) DAPG policies in Adroit (Rajeswaran et al., 2018) and oracle DDPG (Lillicrap et al., 2016) policies in DMControl. We consider the same tasks as PVR, but average Adroit results over two camera views to improve robustness of results (PVR only considers two tasks in this domain). Results for individual views are shown in Figure 10. Our real robot demonstrations are collected via human teleoperation. PVR did not conduct real-world experiments. *Our key difference compared to the original LfS baseline in PVR is the use of random shift augmentation*; all other implementation details remain identical to the original paper.

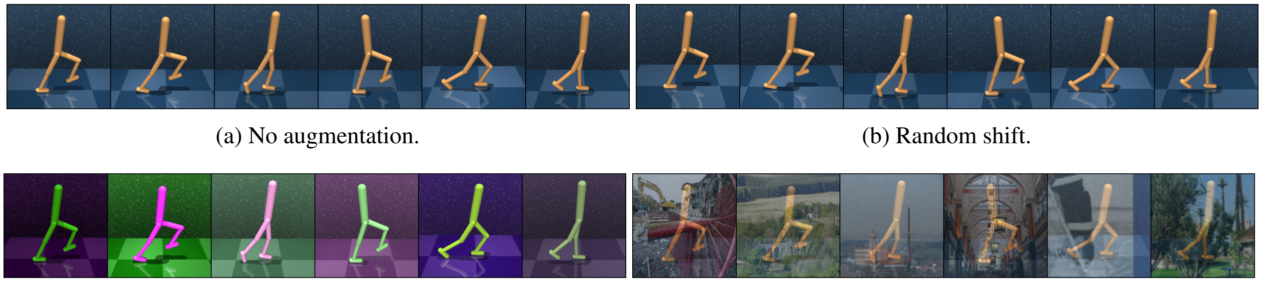
B.2. On-Policy RL

We closely follow the implementation of MVP (Xiao et al., 2022). Observations are single 224×224 RGB images and also include proprioceptive state information. The original LfS baseline proposed in MVP uses a small ViT (Dosovitskiy et al., 2021) encoder. We propose *two* improved LfS baselines for this setting: *(1)* an LfS baseline that uses a shallow ConvNet encoder and *no* data augmentation, referred to as *LfS*, and *(2)* an LfS baseline that additionally applies random shift augmentation to the input images, referred to as *LfS (+aug)*. Our proposed LfS encoder for on-policy RL can be summarized as:

```
(0): Conv2d(3, 32, kernel_size=(7, 7), stride=2)
(1): ReLU()
(2): Conv2d(32, 32, kernel_size=(5, 5), stride=2)
(3): ReLU()
(4): Conv2d(32, 32, kernel_size=(3, 3), stride=2)
(5): ReLU()
(6): Conv2d(32, 32, kernel_size=(3, 3), stride=2)
(7): ReLU()
(8): Conv2d(32, 32, kernel_size=(3, 3), stride=2)
(9): ReLU()
(10): Conv2d(32, 32, kernel_size=(3, 3), stride=2)
(11): ReLU()
```

Following the MVP implementation, output feature maps are flattened and passed through a LayerNorm and linear projection:

```
x = encoder(x)
x = x.view(x.shape[0], -1)
```



(a) No augmentation.

(b) Random shift.

(c) Color jitter.

(d) Random overlay.

Figure 9. **Data augmentation**. Visualization of all choices of data augmentation considered in this work. We adopt augmentation hyperparameters from prior work without modification.

```
x = Linear(LayerNorm(x))
```

Meanwhile, following previous work on visual RL (Hansen et al., 2021; Yarats et al., 2021), we add an additional trunk layer to the policy head:

```
(0): Linear(Z, Z)
(1): nn.LayerNorm(Z)
(2): nn.Tanh()
```

where Z denotes the dimensionality of image features concatenated with the proprioceptive state. We apply random shift augmentation to inputs using a padding of 10 to keep the padding-to-image ratio consistent with its original proposal. Following prior work (Hansen et al., 2021; Raileanu et al., 2020), we do not augment value targets. All other implementation details are kept identical. As our results in Figure 1 reveal, data augmentation is not necessary for on-policy RL algorithms such as PPO, and both of our two LfS baselines thus improve over the original baseline.

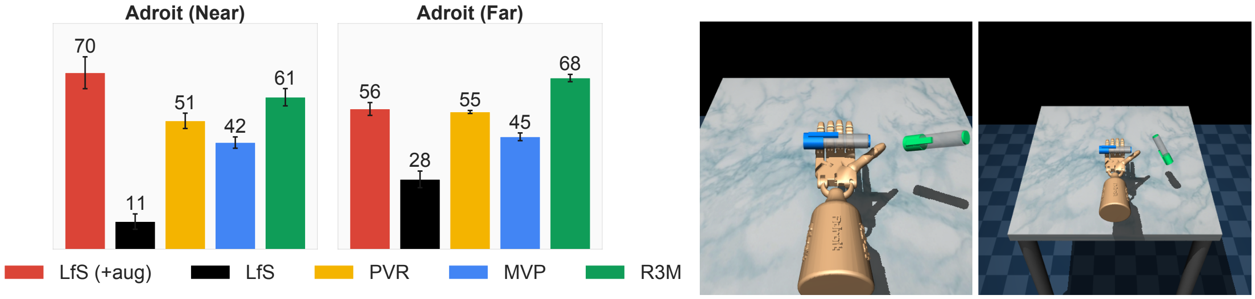
B.3. Off-Policy RL

We closely follow the implementation of DrQ-v2 (Yarats et al., 2021) – a state-of-the-art LfS method that uses random shift augmentation – for our off-policy RL experiments. Observations are stacks of 84×84 RGB images (3 frames) with no access to state information. We denote this baseline as *LfS (+aug)* since it already employs augmentation by default, and construct our *LfS* baseline by simply disabling augmentation in DrQ-v2. We make no changes to the architecture nor hyperparameters, but list the encoder here for completeness:

```
(0): Conv2d(9, 32, kernel_size=(3, 3), stride=2)
(1): ReLU()
(2): Conv2d(32, 32, kernel_size=(3, 3), stride=1)
(3): ReLU()
(4): Conv2d(32, 32, kernel_size=(3, 3), stride=1)
(5): ReLU()
(6): Conv2d(32, 32, kernel_size=(3, 3), stride=1)
(7): ReLU()
```

C. Data Augmentation

We consider three choices of data augmentation in this work: random *image shift* (Kostrikov et al., 2021; Yarats et al., 2021), random *color jitter*, and random *overlay* (Hansen & Wang, 2021). Augmentations are visualized in Figure 9. As in prior work, augmentations are applied consistently across time when using frame stacking. For completeness, we also visualize sample environments from the two test domains from DMControl Generalization Benchmark (Hansen & Wang, 2021) in Figure 5, for which a sizable domain gap remains even after applying data augmentation to observations. In our RL experiments on visual robustness (strong augmentation), we use the objective of Hansen et al. (2021) to stabilize training.



(a) Results for individual views.

(b) Near view.

(c) Far view.

Figure 10. **Results for individual camera views in Adroit.** To improve reliability of our results, we report the average success rate over two camera views for Adroit: *Near* (`fixed`) and *Far* (`vil_camera`). PVR (Parisi et al., 2022) reports results for the *Far* view only. All numbers are means across two tasks and 5 seeds. We find that pre-trained representations benefit more from the farther view, whereas LfS benefits more from the near view.