Adapting Image-based RL Policies via Predicted Rewards

Weiyao Wang   WWANG121@JHU.EDU
Xinayun Fang   XFANG22@JHU.EDU
Gregory D. Hager   HAGER@CS.JHU.EDU
Computer Science Department, Johns Hopkins University

Abstract

Image-based reinforcement learning (RL) faces significant challenges in generalization when the visual environment undergoes substantial changes between training and deployment. Under such circumstances, learned policies may not perform well leading to degraded results. Previous approaches to this problem have largely focused on broadening the training observation distribution, employing techniques like data augmentation and domain randomization. However, given the sequential nature of the RL decision-making problem, it is often the case that residual errors are propagated by the learned policy model and accumulate throughout the trajectory, resulting in highly degraded performance.

In this paper, we leverage the observation that predicted rewards under domain shift, even though imperfect, can still be a useful signal to guide fine-tuning. We exploit this property to fine-tune a policy using reward prediction in the target domain. We have found that, even under significant domain shift, the predicted reward can still provide meaningful signal and fine-tuning substantially improves the original policy. Our approach, termed Predicted Reward Fine-tuning (PRFT), improves performance across diverse tasks in both simulated benchmarks and real-world experiments. More information is available at project web page: https://sites.google.com/view/prft.

Keywords: Reinforcement learning, vision-based policy, domain adaptation

1. Introduction

Image-based reinforcement learning (RL) has gained substantial attention with success in gaming (Mnih et al., 2013), robotic manipulation (Amarjyoti, 2017), and autonomous driving (Chen et al., 2021a), among other areas. However, the visual environment often undergoes significant appearance changes, such as lighting, textures, and camera poses, between the training and testing phases. This leads to performance degradation due to the well recognized challenge of input domain gap (Zhao et al., 2020). This generalization challenge is particularly acute in sequential decision-making processes such as RL where deviations at each step can accumulate, significantly exacerbating performance degradation during rollout (Wang et al., 2019).

To improve generalization across different domains, a common approach involves domain randomization (Ganin et al., 2016; Chen et al., 2021b) or data augmentation (Ma et al., 2022; Kirk et al., 2021). These methods broaden the training observation distribution with the intention of covering the target environment encountered during testing. However, anticipating domain shift can be challenging, rendering this method less effective, particularly when the domain shift is significant. Another method to correct errors under visual domain shift is to fine-tune under the target deployment domain (Wang and Deng, 2018; Guo et al., 2019). This, however, is often impractical for robot learning problems as, in many cases, rewards are specified using internal states only available in the training domain (e.g. in simulation).
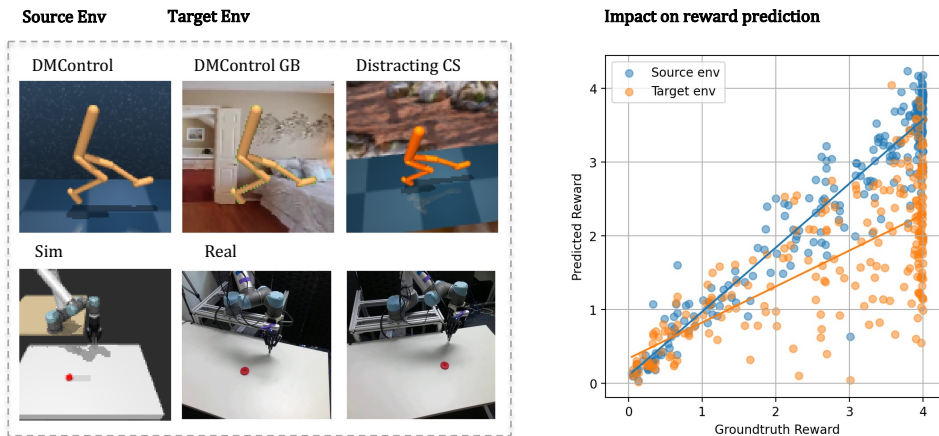
Figure 1: **Left:** Example of source environment observations and target environment observations. **Right:** Illustration of domain shift effect on reward prediction. Samples are collected using a trained policy on the source walker walk environment, and the domain shift effect is tested by evaluating predicted rewards under both the source and the target environment (`video_hard` in DMControl GB) with the same underlying states. Fitted linear regression for the predicted rewards against the groundtruth rewards for both source and target environment are plotted for visualization.

To address this issue, we investigate an alternative approach: fine-tuning the policy in the reward-free target environment using predicted rewards from observations. The proposed method stems from the observation that domain shifts impact the policy model and the reward prediction model in distinct ways.

- RL is able to effectively refine a policy in the presence of random reward space noise (Sun et al., 2021; Wang et al., 2020; Eysenbach and Levine, 2021), which makes improvement with imperfect rewards feasible. Meanwhile, error in action will accumulate throughout the rollout, leading to significantly deteriorated episodic performance with (Ross and Bagnell, 2010). The regret can grow quadratically with the number of timestep and it is difficult to address.

- Not all of the domain shift impact on reward prediction will alter the induced optimal policy. As depicted in Figure 1, aside from exhibiting larger errors, reward prediction in the target environment is become more conservative for out-of-distribution samples and can be regarded as undergoing a linear transformation in the form of $\hat{r} = kr + b$ with $k > 0, b \in \mathcal{R}$. Such linear transformation is known to maintain the optimal policy. This suggests that part of the reward prediction error can be considered as benign. In contrast, errors in action do not possess similar property, rendering them less forgiving.

Building on these insights, we propose to jointly learn (at training time) a policy and a reward prediction model. The policy is then fine-tuned using the predicted rewards in the target testing environment prior to testing and deployment. Through extensive experiments in both simulation and real-world experiments, we show that the reward prediction model generalizes well across visual effect shifts and significantly enhances policy performance through fine-tuning. We benchmark our approach against others addressing the domain adaptation problem in RL, including methods that incorporate data augmentation (Hansen et al., 2021) and self-supervised test-time training (Hansen et al.,

2020) for domain adaptation. Our experimental results demonstrate that our approach outperforms the baseline methods by substantial margins across various benchmark environments.

In summary, we propose a novel approach for image-based RL that employs a reward prediction function to adapt to visual domain shifts in the testing environment. The efficacy of this method is built upon the structural advantage of the reward model in generalizing RL policies. Our approach presents a promising direction to the challenge of domain adaptation in image-based RL.

2. Related Work

Reinforcement learning that leverages visual input has recently gathered substantial interest due to its broad range of real-world applications. Techniques that uses learned image encoders (e.g., convolutional neural networks) to reinforcement learning algorithms have demonstrated impressive achievements by learning policies directly from pixel images (Finn et al., 2016; Lin et al., 2019; Lyle et al., 2021; He et al., 2022).

Various approaches have been proposed to address the generalization problem in visual RL. One common category involves domain randomization (Ganin et al., 2016; Chen et al., 2021b) or data augmentation during training (Kostrikov et al., 2020; Laskin et al., 2020; Ma et al., 2022). One challenge with these methods is that an expanded training observation distribution complicates the optimization of RL policies. To mitigate this issue, a line of research (Hansen and Wang, 2021; Fan et al., 2021) proposed to decouple augmentation from policy learning by latent space regularization or policy distillation. Hansen et al. (2021) further identified that direct application of regularization introduces non-deterministic Q-targets and over-regularization, leading to inefficiency in policy optimization. Their method, SVEA, proposes to jointly optimize the Q-function with both augmented and non-augmented data to improve stability and sample efficiency.

Instead of trying to be invariant to all domains, some works also aim to adapt policy to some specific target environment without reward access, which shares the same setting as our method. One approach uses generative adversarial networks (GANs) (Goodfellow et al., 2020) to translate images (Zhang et al., 2019; Rao et al., 2020; Ho et al., 2021) or latent features (Yoneda et al., 2021) from target domains to source domain and then feed into policies. Despite being effective in some cases, this line of research requires access to a large amount of target domain observations to capture the observational space distribution. Moreover, the training can be challenging with generative adversarial training. Another approach under this setting is to perform adaptation through self-supervised auxillary task under the target environment. PAD (Hansen et al., 2020) jointly learns an inverse dynamics model (IDM) alongside RL learning during training. In testing, it fine-tunes the image encoder by optimizing the IDM objective to adapt to the target environment. Fine-tuning on this self-supervised task circumvents the problem of reward signal unavailability in the test deployment environment. However, its benefits are limited since this signal adapts to the new transition dynamics but is relatively indirect from the task to be performed.

In a similar vein to our method, other approaches explore the use of an evaluation model to provide feedback to fine-tune policies. PAFF (Ge et al., 2022) leverages a robust vision language foundation model to label executed instruction-conditioned policies under testing domain with descriptions and then fine-tunes policies with generated descriptions using imitation learning (IL). More broadly in the space of Natural Language Processing (NLP), the recent popular method Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022) first learns a reward model based on human preferences and then fine-tunes the language generative model based on the
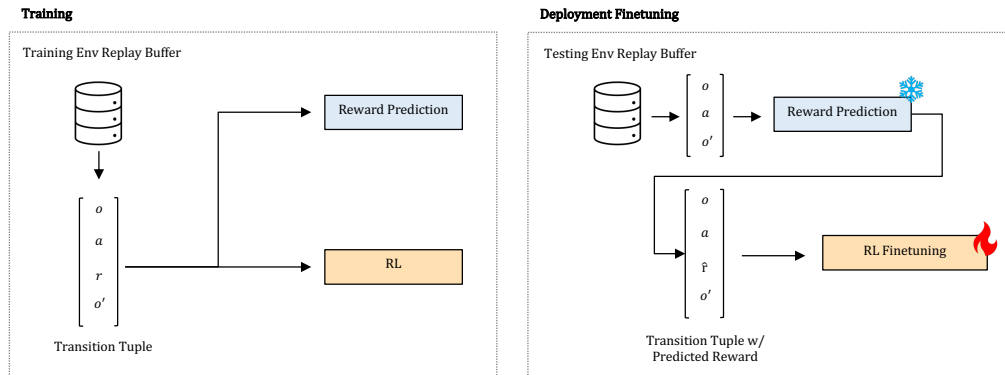
Figure 2: **Left:** During training, we optimize the reward prediction module along with reinforcement learning using sampled transition tuples from replay buffer. **Right:** During deployment finetuning, we use the transition tuples with predicted reward to finetune the reinforcement learning policy. The reward prediction module is frozen in this stage.

learned reward model. Meanwhile in this paper, we systematically examine the benefit of fine-tuning with a reward model in the context of visual domain adaptation for RL policies.

3. Method

Formally, we frame our problem as adaptation of a visual policy. Consider a Markov decision process (MDP) (Sutton and Barto, 2018) $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma\}$ where S and A are the state and action spaces respectively. $P : S \times A \to S$ is the transition function, $R : S \times A \to \mathrm{R}$ represents the scalar reward, and γ represents the discount factor. We assume that the agent cannot directly observe state space S but only receives higher-dimensional input O (e.g., pixel images) as observations. Let $\pi(o, a) \in \Pi : O, A \to [0, 1]$ be the policy that maps observation $o \in O$ and action a to probability. The source environment and target environment differ only in the mapping of the same state S to different observations O due to visual appearance shifts, while all other elements remain the same. Moreover, we presume that the agent can access the target environment to gather interactions but without groundtruth reward provided.

We first consider the standard RL framework, where an agent interacts with an environment over a sequence of discrete time steps. At each time step t, the agent receives an observation o_t from observation space O, takes an action a_t from a set of possible actions $\mathcal{A}$, and receives a scalar reward r_t. The goal is to learn a policy π that maximizes the expected cumulative reward, defined as:

$$J(\pi; r) \triangleq \mathbb{E}_\pi \left[\sum_{t=1}^{T} r(\mathbf{s_t}, \mathbf{a_t}) \gamma^t \right], \tag{1}$$

where T represents the termination timestep.

We denote the reward prediction function as $\phi(o, a)$. This function takes in an observation o and an action a, and then it outputs a scalar reward prediction $\hat{r}$. A neural network is utilized to model this function, which we denote as ϕ with parameters θ_r. We train this neural network to minimize the discrepancy between the predicted reward $\hat{r}$ and the actual reward r. This discrepancy is measured by the mean squared error, defined as follows:

$$\mathcal{L}_{\phi(\theta_r)} = \mathbb{E}_{(o,a,r)\sim\mathcal{B}_{\text{train}}}[(\hat{r} - r)^2] = \mathbb{E}_{(o,a,r)\sim\mathcal{B}_{\text{train}}}[(\phi(o, a; \theta_r) - r)^2]. \tag{2}$$

In this equation, $\mathcal{B}_{\text{train}}$ represents the training replay dataset of observed states and rewards.

Algorithm 1 Training and Fine-tuning PRFT

1: **Train:**
2: *Inputs:* Reward prediction function ϕ, its parameters θ_r, policy π, training replay buffer $\mathcal{B}_{\text{train}}$
3: **for** each epoch **do**
4:     Execute π in the environment to get (o, a, r, o'), store in $\mathcal{B}_{\text{train}}$
5:     Sample (o, a, r, o') from $\mathcal{B}_{\text{train}}$
6:     Update π using MaxEnt RL algorithm with reward (o, a, r, o')
7:     Update θ_r to minimize $\mathcal{L}_{\phi(\theta_r)} = (\phi(o, a; \theta_r) - r)^2$
8: **end for**
9:
10: **Fine-tune:**
11: *Inputs:* Reward prediction function ϕ, its parameters θ_r, policy π, replay buffer $\mathcal{B}_{ft}$
12:     Freeze ϕ
13: **for** each epoch **do**
14:     Execute π in the environment to get (o, a, o')
15:     Compute $\hat{r}$ using $\phi(o, a; \theta_r)$, store $(o, a, \hat{r}, o')$ in $\mathcal{B}_{ft}$
16:     Update π using MaxEnt RL algorithm with samples from $\mathcal{B}_{ft}$
17: **end for**

PRFT is outlined in Algorithm 1 with the process of training and fine-tuning. During the training phase, a policy and reward prediction function are jointly learned. After training, the reward prediction function is frozen and the policy is fine-tuned. During the fine-tuning stage, the agent interacts with the environment and the experiences are stored in a replay buffer. The stored experiences are then used to compute reward predictions which in turn guide the fine-tuning of the policy. This ensures a better alignment of the policy to perform well in the target environment.

We employ MaxEnt maximum entropy (MaxEnt) RL algorithm to optimize policy π during both the training and fine-tuning phases. Recall that the original reward function is defined as $J_{(\pi;r)} \triangleq \mathbb{E}_\pi \left[\sum_{t=1}^T r(\mathbf{s_t}, \mathbf{a_t})\gamma^t \right]$. The MaxEnt RL objective is to maximize the sum of the expected reward and the conditional action entropy, represented by $J_{\text{MaxEnt}}(\pi; r) \triangleq \mathbb{E}_\pi \left[\sum_{t=1}^T r(\mathbf{s_t}, \mathbf{a_t})\gamma^t + \alpha \mathcal{H}_\pi[\mathbf{a_t} \mid \mathbf{s_t}] \right]$, where $\mathcal{H}_\pi[\mathbf{a_t} \mid \mathbf{s_t}] = \int_{\mathcal{A}} \pi(\mathbf{a_t} \mid \mathbf{s_t}) \log \frac{1}{\pi(\mathbf{a_t}|\mathbf{s_t})} d\mathbf{a_t}$ denotes the entropy of the action distribution. The *entropy coefficient* α balances the reward and entropy terms.

We choose to use MaxEnt RL in particular because it is robust to some degree of misspecification in the reward function. Eysenbach and Levine (2021) shows that assuming $\alpha = 1$, the reward function is finite and the policy has support everywhere (i.e., $\pi(\mathbf{a_t} \mid \mathbf{s_t}) > 0$ for all states and actions), there exists a positive constant $\epsilon > 0$ such that optimizing the MaxEnt RL objective $J_{\text{MaxEnt}}(\pi, \hat{r})$ is equivalent to optimizing a lower bound of the objective function $J(\pi, \tilde{r})$:

$$\min_{\tilde{r}\in\tilde{\mathcal{R}}(\pi)} \mathbb{E}\Big[\sum_t \tilde{r}(\mathbf{s_t}, \mathbf{a_t})\Big] = J_{\mathrm{MaxEnt}}(\pi; p, r) \quad \forall \pi,$$

where the adversary chooses a reward function from the robust set

$$\tilde{\mathcal{R}}(\pi) \triangleq \left\{ \tilde{r}(\mathbf{s_t}, \mathbf{a_t}) \ \middle| \ \mathbb{E}_\pi\Big[\sum_t \log \int_{\mathcal{A}} \exp(r(\mathbf{s_t}, \mathbf{a_t}') - \tilde{r}(\mathbf{s_t}, \mathbf{a_t}'))d\mathbf{a_t}'\Big] \le \epsilon \right\}. \tag{3}$$

Therefore applying MaxEnt RL to one reward function, $r(\mathbf{s_t}, \mathbf{a_t})$, results in a policy that is guaranteed to also achieve high return on a range of other reward functions, $\tilde{r}(\mathbf{s_t}, \mathbf{a_t}) \in \tilde{\mathcal{R}}$. This suggests that we can employ MaxEnt RL algorithm such as SAC (Haarnoja et al., 2018) to improve a policy's performance on the unknown true reward objective even if the provided reward is imperfect (i.e., from a reward predictor under domain shift). With the help of the reward prediction and MaxEnt RL algorithm, we are able to fine-tune the policy on test domain without groundtruth reward signal provided. In the following section, we are going to show that such fine-tuning stage significantly improves the policy performance on various simulated and real-robot tasks.

4. Experiments

We aim to understand the impact of domain adaptation on an agent's ability to generalize out-of-distribution. This section compares PRFT with a state-of-the-art baseline in image-based RL: DrQ (Laskin et al., 2020) and one that uses data-augmentation: SVEA (Hansen et al., 2021). As discussed in the related work, this method enhances generalization capabilities by expanding the support of the training distribution. We use SVEA as a data augmentation baseline to supplement our method, denoting as SVEA + PRFT. In our experiments, we also compare against SVEA+IDM, the combination of SVEA and inverse dynamics model, and SVEA+PAD, the combination of SVEA and policy adaptation during deployment (Hansen et al., 2020) which is a baseline method that adapts the policy through self-supervised auxiliary tasks. All methods in comparison employs MaxEnt RL algorithm SAC (Haarnoja et al., 2018) as base learning algorithm.

Implementation: For all methods, we adopt the same network architecture as Hansen et al. (2021): a 11-layer ConvNet followed by 3-layer MLP with 1024 hidden units. We use the *random overlay* data augmentation (Hansen and Wang, 2021) as a part of SVEA training. *random overlay* interpolates observation image with a random chosen image from Places dataset (Zhou et al., 2017).

4.1. PRFT on Simulated Environments

We conduct experiments on 6 domains from the DeepMind Control Suite (DMControl (Tassa et al., 2018)) and treat it as the source domain for training the RL agents. For the target domain, we use 1) the DMControl with video background introduced in DMControl Generalization Benchmark (DMControl GB) (Hansen et al., 2021) and 2) Distracting Control Suite (Distracting CS)Stone et al. (2021). Distracting CS adds background, color, and camera pose distractions, making it an ideal benchmark for testing the generalization and robustness of reinforcement learning algorithms. All simulated experiments are performed across 4 random seeds with 20 episodes during evaluation.

We first present the evaluation results on DMControl GB with `video_easy` and `video_hard` domains in Table 1. The table compares different methods, including DrQ (Kostrikov et al., 2020), SVEA (Hansen et al., 2021), PAD (Hansen et al., 2020), and our method, PRFT, based on episodic
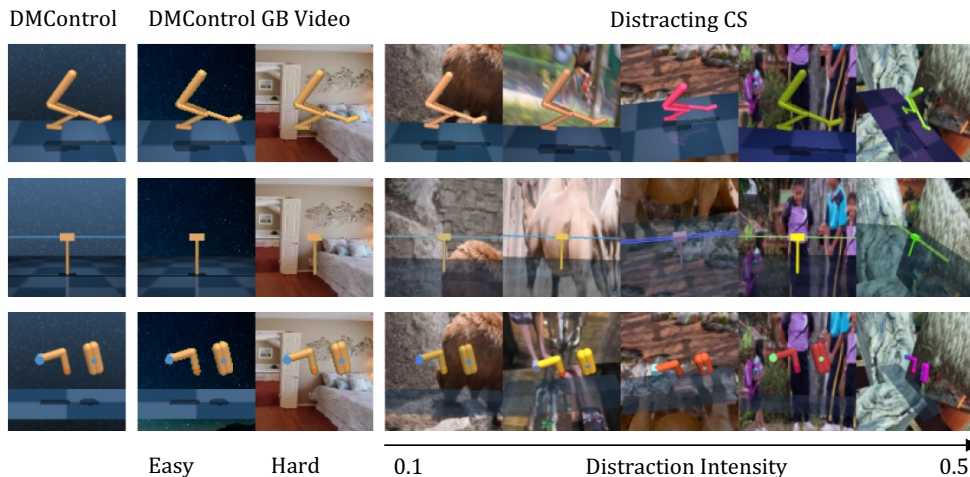
Figure 3: Samples from deepmind control suite (DMControl), deepmind control generalization benchmark (DMControl GB) with video background (easy and hard), and distracting control suite (Distracting CS) with intensity from 0.1 to 0.5.

Task (`video_easy`)	DrQ	SVEA	+IDM	+PAD	+PRFT (ours)
`finger_spin`	477 ± 435	770 ± 306	661 ± 450	685 ± 463	$\mathbf{944 \pm 37}$
`cartpole_balance`	717 ± 162	862 ± 128	640 ± 269	646 ± 296	$\mathbf{897 \pm 114}$
`cartpole_swingup`	496 ± 409	825 ± 28	644 ± 298	644 ± 295	$\mathbf{826 \pm 11}$
`walker_stand`	890 ± 109	$\mathbf{940 \pm 17}$	904 ± 102	899 ± 110	936 ± 15
`walker_walk`	630 ± 258	843 ± 141	870 ± 82	867 ± 42	$\mathbf{878 \pm 43}$
`walker_run`	183 ± 102	240 ± 34	243 ± 21	244 ± 21	$\mathbf{258 \pm 17}$
Task (`video_hard`)					
`finger_spin`	16 ± 27	151 ± 118	107 ± 75	97 ± 67	$\mathbf{193 \pm 115}$
`cartpole_balance`	214 ± 30	303 ± 64	244 ± 23	254 ± 28	$\mathbf{860 \pm 198}$
`cartpole_swingup`	169 ± 33	376 ± 70	189 ± 45	181 ± 54	$\mathbf{510 \pm 289}$
`walker_stand`	287 ± 103	849 ± 64	764 ± 183	769 ± 176	$\mathbf{904 \pm 47}$
`walker_walk`	105 ± 75	377 ± 172	556 ± 199	541 ± 212	$\mathbf{560 \pm 114}$
`walker_run`	36 ± 9	171 ± 40	173 ± 33	$\mathbf{174 \pm 36}$	154 ± 45

Table 1: Evaluation by episodic cumulative rewards (mean ± standard deviation) for 6 DMControl GB environments. Comparison of methods includes DrQ, SVEA, SVEA+IDM, SVEA+PAD, and our method SVEA+PRFT. SVEA is omitted from the method name of the last three columns in the chart for better readability.

reward. Notably, in 10 out of 12 tasks, our proposed PRFT performs the best, surpassing competing approaches. By fine-tuning with predicted reward, we observe a clear improvement in reward across different tasks. This highlights the effectiveness of PRFT in adapting the policy to the target domain and enhancing performance. We notice that in `video_hard` version of `walker_run` environment, our method brought negative change to the performance. This suggests that the error in reward prediction could be too large and make it a harmful fine-tuning signal in this case.

Beyond domain shifts in the background, we further tested our method in the Distracting Control Suite, which additionally introduces distractions in color and camera pose with controlled intensities ranging from 0.1 to 0.5. Figure 5 visualizes the performance of the different methods, as a function of the intensity of the distractions. Since the baseline method SVEA is trained with image augmentation,

it does exhibit some robustness to distraction. However, we see this robustness rapidly diminishes as the distraction intensity increases. In particular, large changes to camera pose or the image background proved challenging for standard augmentation procedures. Comparatively, PRFT makes it much smoother and slower degradation of performance for most of the environments. The improvement is particularly prominent under high distraction intensity, highlighting the robustness of our reward prediction in generating useful signals despite heavy domain shifts. This supports our hypothesis that adaptation powered by predicted rewards can significantly improve the target environment performance abilities of the policy.

To better visualize the improvements in rewards brought about by fine-tuning, we plotted the average reward improvements for each distraction intensity in Figure 4. As depicted in the graphs, our method, PRFT, enhances the average rewards across all distraction intensities swiftly, using as few as 10K steps. The performances then continue to increase and converge at 50K steps. On comparing the improvements across different distraction intensities, we found that the largest improvement was made at a moderately high distraction intensity of 0.4. Here, fine-tuning with 10K steps brings about a 50% improvement, and fine-tuning with 50K steps brings about a 70% improvement. The improvements at lower and higher distraction intensities were smaller but still substantial. For small distraction values of 0.1 and 0.2, this is because the zero-shot transfer already performs well, thereby limiting the scope for improvement. For large distraction values of 0.5, the accuracy of the predicted reward also drops, which results in diminishing benefits from using it as a fine-tuning signal.
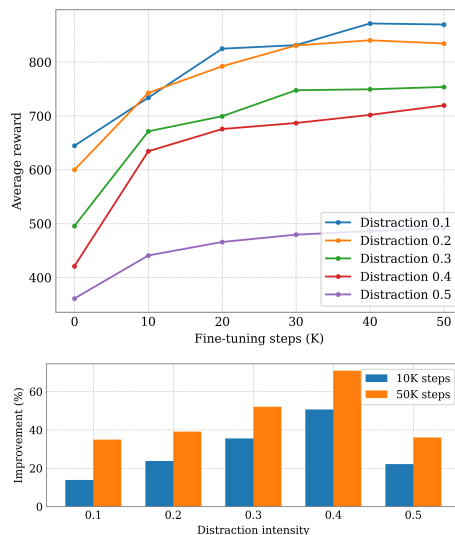

Figure 4: **Top:** Policies improve during fine-tuning using predicted rewards. Average episodic rewards over four tasks and four random seeds are plotted. **Bottom:** Relative improvement of average rewards across different distraction intensities at 10K and 50K fine-tuning steps.

4.2. Sim-to-real Transfer

We are also interested in the ability of PRFT to bridge the gap between simulated and real world robotics environments. To do so we define a reaching task where a target position is given by a red disc placed on a table. The agent's objective is to controls the arm so that the end-effector reaches this target location. Our goal is to train a policy in simulation, and then transfer the policy to a real UR-5 robot at test time. The same as with previous experiments, the test time agent receives no rewards. In both simulation and the real world, the policy's only input

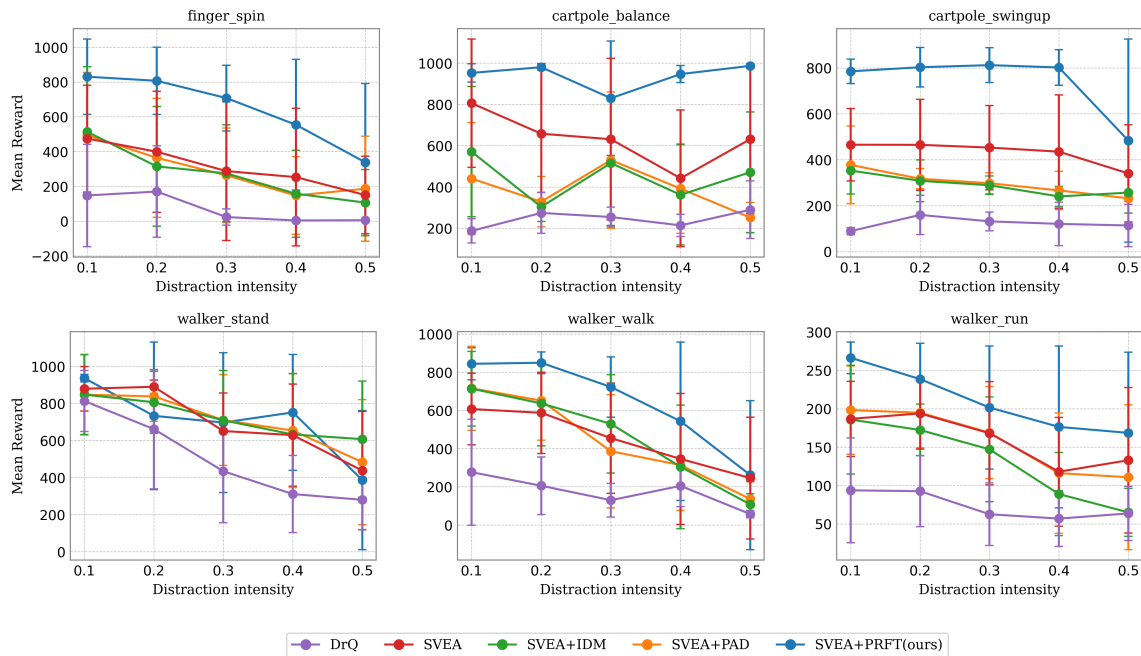| Method	Success Rate
DrQ	0.32
SVEA	0.52
+PAD	0.48
+PRFT (ours)	**0.68**

Table 2: Evaluation on sim-to-real.

Figure 5: Evaluation in environments under distracting control suite with varying degrees of distraction intensities. Our method significantly outperforms baseline methods in five out of six environments. Error bar shows one standard deviation.

is an image from a camera placed in front of the robot and table. The action space is a 2D position controller that drives a small movement of the robotic gripper.

We carry out the experiment by first training a policy with DrQ and SVEA [14] in simulation. For adaptation, we apply PAD and PRFT to finetune the SVEA policy in the real environment to bridge the sim-to-real gap. We evaluated the success rate of zero-shot and the adapted policies over 25 real episodes. We consider the episode to be success if the gripper's tip is within 5 cm of the center of the target. Due to the challenging domain shift between simulator and real world, both the zero-shot policies and PAD fails to adapt adequately. One of the failure cases is repeating the same action of moving the gripper to an edge of the table, regardless of the given goal location. This results in a final success rate around 50% for SVEA on zero-shot. On the same task, PRFT is able to adapt to this domain shift, achieving a final success rate around 70%.

In summary, our results demonstrate the robustness of our method in handling domain shift introduced in both simulated and real environments. The substantial improvements achieved by our approach, even under high distraction intensity, highlight the effectiveness predicted rewards as fine-tuning signals.

5. Conclusion

In this paper we have presented and evaluated a novel approach, PRFT (Predicted Reward Fine-Tuning), for adapting policy under domain shift. Our method demonstrated superior performance across various robotic environments, outperforming baseline methods in large margins. Notably, our
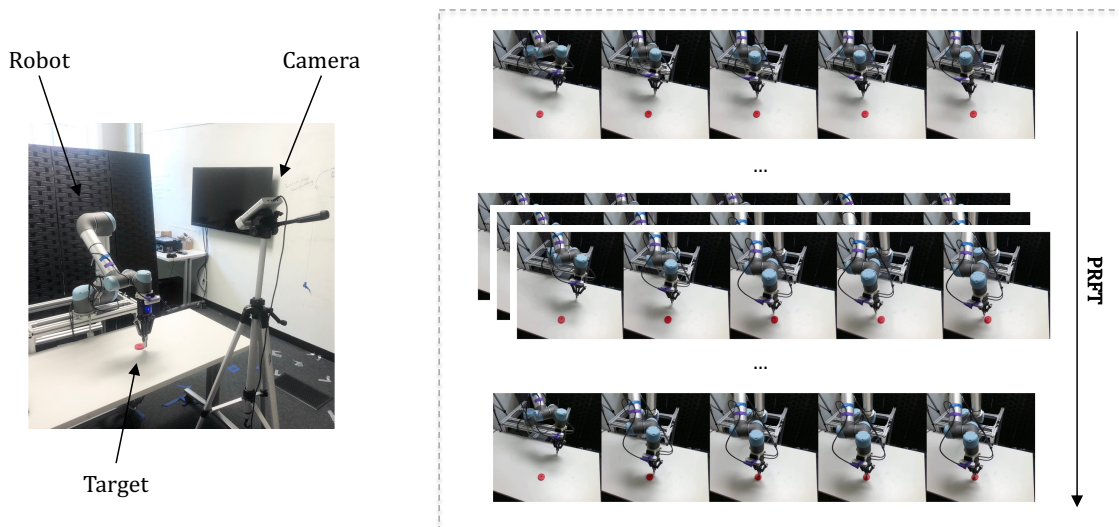
Figure 6: **Left:** Our real-world experiment setup with robot, camera and target. **Right:** Example rollouts of progress made in real-world predicted reward fine-tuning. The robot is able to reach target that is not achievable before fine-tuning.

approach exhibited significant improvement even under high distraction intensity, highlighting the robustness of our method under large domain shifts.

Our results indicate that, imperfect reward prediction is still a useful fine-tuning signal in the test domain. This, however, no longer holds when the error exceeds certain level. When large domain shifts happens, incorrectly predicted reward may misguide the policy fine-tuning, leading to even worse performance compared to zero-shot testing. How to detect such scenarios and mitigate accordingly is one direction for future work.

Acknowledgments

References

Smruti Amarjyoti. Deep reinforcement learning for robotic manipulation-the state of the art. *arXiv preprint arXiv:1701.08878*, 2017.

Jianyu Chen, Shengbo Eben Li, and Masayoshi Tomizuka. Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 23(6):5068–5078, 2021a.

Xiaoyu Chen, Jiachen Hu, Chi Jin, Lihong Li, and Liwei Wang. Understanding domain randomization for sim-to-real transfer. *arXiv preprint arXiv:2110.03239*, 2021b.

Benjamin Eysenbach and Sergey Levine. Maximum entropy rl (provably) solves some robust rl problems. *arXiv preprint arXiv:2103.06257*, 2021.

Linxi Fan, Guanzhi Wang, De-An Huang, Zhiding Yu, Li Fei-Fei, Yuke Zhu, and Anima Anandkumar. Secant: Self-expert cloning for zero-shot generalization of visual policies. *arXiv preprint arXiv:2106.09678*, 2021.

Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, and Pieter Abbeel. Deep spatial autoencoders for visuomotor learning. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 512–519. IEEE, 2016.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016.

Yuying Ge, Annabella Macaluso, Li Erran Li, Ping Luo, and Xiaolong Wang. Self-play and self-describe: Policy adaptation with vision-language foundation models. *arXiv preprint arXiv:2212.07398*, 2022.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Yunhui Guo, Honghui Shi, Abhishek Kumar, Kristen Grauman, Tajana Rosing, and Rogerio Feris. Spottune: transfer learning through adaptive fine-tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4805–4814, 2019.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

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

Nicklas Hansen, Rishabh Jangir, Yu Sun, Guillem Alenyà, Pieter Abbeel, Alexei A. Efros, Lerrel Pinto, and Xiaolong Wang. Self-supervised policy adaptation during deployment. 2020.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Stabilizing deep q-learning with convnets and vision transformers under data augmentation. *Advances in Neural Information Processing Systems*, 34: 3680–3693, 2021.

Tairan He, Yuge Zhang, Kan Ren, Minghuan Liu, Che Wang, Weinan Zhang, Yuqing Yang, and Dongsheng Li. Reinforcement learning with automated auxiliary loss search. *arXiv preprint arXiv:2210.06041*, 2022.

Daniel Ho, Kanishka Rao, Zhuo Xu, Eric Jang, Mohi Khansari, and Yunfei Bai. Retinagan: An object-aware approach to sim-to-real transfer. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10920–10926. IEEE, 2021.

Robert Kirk, Amy Zhang, Edward Grefenstette, and Tim Rocktäschel. A survey of generalisation in deep reinforcement learning. *arXiv preprint arXiv:2111.09794*, 2021.

Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *arXiv preprint arXiv:2004.13649*, 2020.

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

Xingyu Lin, Harjatin Baweja, George Kantor, and David Held. Adaptive auxiliary task weighting for reinforcement learning. *Advances in neural information processing systems*, 32, 2019.

Clare Lyle, Mark Rowland, Georg Ostrovski, and Will Dabney. On the effect of auxiliary tasks on representation dynamics. In *International Conference on Artificial Intelligence and Statistics*, pages 1–9. PMLR, 2021.

Guozheng Ma, Zhen Wang, Zhecheng Yuan, Xueqian Wang, Bo Yuan, and Dacheng Tao. A comprehensive survey of data augmentation in visual reinforcement learning. *arXiv preprint arXiv:2210.04561*, 2022.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744, 2022.

Kanishka Rao, Chris Harris, Alex Irpan, Sergey Levine, Julian Ibarz, and Mohi Khansari. Rl-cyclegan: Reinforcement learning aware simulation-to-real. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11157–11166, 2020.

Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 661–668. JMLR Workshop and Conference Proceedings, 2010.

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

Chuxiong Sun, Rui Wang, Qian Li, and Xiaohui Hu. Reward space noise for exploration in deep reinforcement learning. *International Journal of Pattern Recognition and Artificial Intelligence*, 35(10):2152013, 2021.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. Deepmind control suite, 2018.

Huan Wang, Stephan Zheng, Caiming Xiong, and Richard Socher. On the generalization gap in reparameterizable reinforcement learning. In *International Conference on Machine Learning*, pages 6648–6658. PMLR, 2019.

Jingkang Wang, Yang Liu, and Bo Li. Reinforcement learning with perturbed rewards. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 6202–6209, 2020.

Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312: 135–153, 2018.

Takuma Yoneda, Ge Yang, Matthew R Walter, and Bradly Stadie. Invariance through inference. *arXiv preprint arXiv:2112.08526*, 2021.

Jingwei Zhang, Lei Tai, Peng Yun, Yufeng Xiong, Ming Liu, Joschka Boedecker, and Wolfram Burgard. Vr-goggles for robots: Real-to-sim domain adaptation for visual control. *IEEE Robotics and Automation Letters*, 4(2):1148–1155, 2019.

Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *2020 IEEE symposium series on computational intelligence (SSCI)*, pages 737–744. IEEE, 2020.

Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017.