Visual Goal-Conditioned Reinforcement Learning by Representation Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

For an autonomous agent to fulfill a wide range of user-specified goals at test time, it must be able to learn broadly applicable and general-purpose skill repertoires. Furthermore, to provide the requisite level of generality, these skills must handle raw sensory input such as images. In this paper, we propose an algorithm that acquires such general-purpose skills by combining unsupervised representation learning and reinforcement learning of goal-conditioned policies. Since the particular goals that might be required at test-time are not known in advance, the agent performs a self-supervised "practice" phase where it imagines goals and attempts to achieve them. We learn a visual representation with three distinct purposes: sampling goals for self-supervised practice, providing a structured transformation of raw sensory inputs, and computing a reward signal for goal reaching. We also propose a retroactive goal relabeling scheme to further improve the sample-efficiency of our method. Our off-policy algorithm is efficient enough to learn policies that operate on raw image observations and goals in a real-world physical system, and substantially outperforms prior techniques.

1 Introduction

Reinforcement learning (RL) algorithms hold the promise of allowing autonomous agents, such as robots, to learn to complete arbitrary tasks. However, the standard RL framework involves learning policies that are specific to individual tasks, which are defined by hand-specified reward functions. Agents that exist persistently in the world can prepare to solve diverse tasks by setting their own goals, practicing complex behaviors, and learning about the world around them. In fact, humans are very proficient at setting abstract goals for themselves, and evidence shows that this behavior is already present from early infancy [36], albeit with very simple goals such as reaching. The behavior and representation of goals grows more complex over time as they learn how to manipulate objects and locomote. How can we begin to devise a reinforcement learning system that sets its own goals and learns from experience with minimal outside intervention and manual engineering?

In this paper, we take a step toward this goal by designing an RL framework that jointly learns representations of raw sensory inputs and policies that achieve arbitrary goals under this representation by practicing to reach self-specified random goals during training. To provide for automated and flexible goal-setting, we must first choose how a general goal can be specified for an agent interacting with a complex and highly variable environment. Even inputting such an environment into a policy is a challenge: for instance, a robot that might need to manipulate various objects would have difficulty representing the world by enumerating the objects directly, since the resulting representation would vary in length and composition depending on the current scene. Directly using raw sensory signals, such as images, avoids this issue, but learning from raw images is substantially harder. In particular, pixel-wise Euclidean distance is not an effective reward function for visual tasks since

Submitted to 32nd Conference on Neural Information Processing Systems (NIPS 2018). Do not distribute.

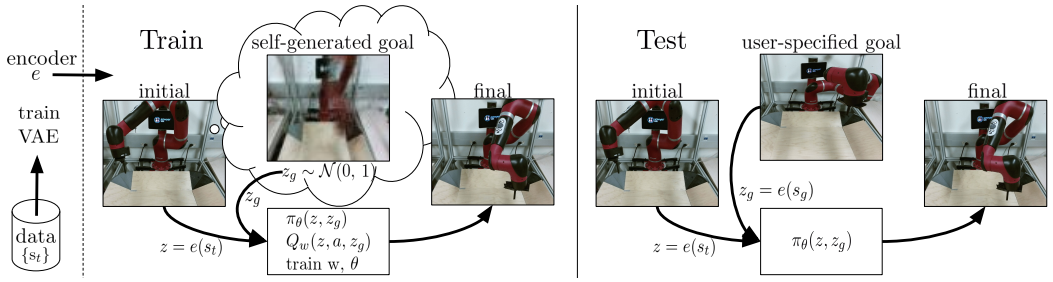

Figure 1: We train a VAE using data generated by our exploration policy (left). We use the VAE for multiple purposes during training time (middle): to sample goals to train the policy, to embed the observations into a latent space, and to compute distances in the latent space. During test time (right), we embed a specified goal observation o_g into a goal latent z_g as input to the policy.

distances between images do not correspond to meaningful distances between world states [30, 41]. Furthermore, although end-to-end model-free reinforcement learning can handle image observations, this comes at a huge cost in sample complexity, making it difficult to use in the real world.

We propose to address both challenges by incorporating unsupervised representation learning into goal-conditioned policies. In our method, which is illustrated in Figure 1, a representation of raw sensory inputs is learned by means of a latent variable model, which in our case is based on the variational autoencoder (VAE) [16]. This model serves three complementary purposes. First, it provides a more structured representation of sensory inputs for RL, making it feasible to learn from images even in the real world. Second, it allows for sampling of new states, which can be used to set synthetic goals during training to allow the goal-conditioned policy to practice diverse behaviors. We can also more efficiently utilize samples from the environment by relabeling synthetic goals in an off-policy RL algorithm, which makes our algorithm substantially more efficient. Third, the learned representation provides a space where distances are more meaningful than the original space of observations, and can therefore provide well-shaped reward functions for RL. By learning to reach random goals sampled from the latent model, the goal-conditioned policy learns about the world and can be used to achieve new, user-specified goals at test-time.

The main contribution of our work is a framework for learning general-purpose goal-conditioned policies that can achieve goals specified with target observations. We call our method goal-conditioned reinforcement learning by representation learning (GRiLL). GRiLL combines sample-efficient off-policy goal-conditioned reinforcement learning with unsupervised representation learning. We use representation learning to acquire a latent distribution that can be used to sample goals for unsupervised practice and data augmentation, to provide a well-shaped distance function for reinforcement learning, and to provide a more structured representation for the value function and policy. While several prior methods, discussed in the following section, have sought to learn goal-conditioned policies, we can do so with image goals and observations without a manually specified reward signal. Our experimental evaluation illustrates that our method substantially improves the performance of image-based reinforcement learning, can effectively learn policies for complex image-based tasks, and can be used to learn real-world robotic manipulation skills with raw image inputs. Videos of our method in simulated and real-world environments can be found at `https://sites.google.com/site/visualrlbyrl/`.

2 Related Work

While prior works on vision-based deep reinforcement learning for robotics can efficiently learn a variety of behaviors such as grasping [28, 27, 20], pushing [1, 7, 9], navigation [26, 18], and other manipulation tasks [22, 19], they each make assumptions that limit their applicability to training general-purpose robots. Levine et al. [19] uses time-varying models, which requires an episodic setup that makes them difficult to extend to non-episodic and continual learning scenarios. Pinto et al. [28] proposed a similar approach that uses goal images, but requires instrumented training in simulation. Lillicrap et al. [22] uses fully model-free training, but does not learn goal-conditioned skills. As we show in our experiments, this approach is very difficult to extend to the goal-conditioned setting with

image inputs. Model-based methods that predict images [40, 9, 7, 24] or learn inverse models [1] can also accommodate various goals, but tend to limit the horizon length due to model drift. To our knowledge, no prior method uses model-free RL to learn policies conditioned on goal images with sufficient efficiency to train directly on real-world robotic systems, without access to ground-truth state or reward information during training.

Our method uses a goal-conditioned value function [37, 34]. To increase the sample-efficiency of our method during off-policy training, we retroactively relabel samples in the replay buffer with goals sampled from the latent representation. Goal relabeling has been explored in prior work [2, 31, 21, 29]. Andrychowicz et al. [2] and Levy et al. [21] use sparse rewards, restricting the resampled goals to states actually encountered along that trajectory, since almost any other goal will have no reward signal. We instead use distances in a learned representation space as the reward, which enables substantially more efficient learning. Furthermore, we can sample random goals from this latent space, rather than restricting ourselves to states actually seen along the sampled trajectory.

A number of prior works have used unsupervised learning to acquire better representations for RL [15, 13, 40, 10, 18]. These methods use the learned representation as a substitute for the state for the policy, but still require access to a ground truth reward function based on the true state during training time. In contrast, we learn to generate goals and use the learned representation to obtain a reward function for those goals. Finn et al. [10] also proposed to combine unsupervised representation learning with reinforcement learning based on guided policy search, but in a framework that trains a policy to reach a single goal. Many prior works have also focused on learning controllable and disentangled representations [35, 4, 5, 32, 6, 38]. We use a method based on variational autoencoders, but these prior techniques are complementary to ours and could be incorporated into our method.

3 Background

Our method combines reinforcement learning with goal-conditioned value functions and unsupervised representation learning. Here, we briefly review the techniques that we build on in our method.

Goal-conditioned reinforcement learning. In reinforcement learning, the goal is to learn a policy $\pi(s_t) = a_t$ that maximizes expected return, which we denote as $R_t = \mathbb{E}[\sum_{i=t}^{T} \gamma^{(i-t)} r_i$, where $r_i = r(s_i, a_i, s_{i+1})$ and the expectation is under the current policy and environment dynamics. Here, $s \in \mathcal{S}$ is a state observation, $a \in \mathcal{A}$ is an action, and γ is a discount factor. Standard model-free RL learns policies that achieve a single task. If our aim is instead to obtain a policy that can accomplish a variety of tasks, we can construct a goal-conditioned policy and reward, and optimize the expected return with respect to a goal distribution: $\mathbb{E}_{g\sim G}[\mathbb{E}_{r_i,s_i\sim E,a_i\sim\pi}[R_0]]$, where $\mathcal{G}$ is the set of goals and the reward is also a function of g. A variety of algorithms can learn goal-conditioned policies, but to enable sample-efficient learning, we focus on algorithms that acquire goal-conditioned Q-functions, which can be trained off-policy. A goal-conditioned Q-function $Q(s, a, g)$ learns the expected return for the goal g starting from state s and taking action a. Given a state s, action a, next state s', goal g, and correspond reward r, one can train an approximate Q-function parameterized by w by minimizing the following Bellman error

$$\mathcal{E}(w) = \frac{1}{2}||Q_w(s, a, g) - (r + \gamma \max_{a'} Q_{\bar{w}}(s', a', g))||^2 \tag{1}$$

where $\bar{w}$ indicates that $\bar{w}$ is treated as a constant. Crucially, one can optimize this loss using off-policy data (s, a, s', g, r) with a standard actor-critic algorithm [22, 12, 23].

Variational Autoencoders. Variational Autoencoders (VAEs) have been demonstrated to learn structured latent representations of high dimensional data [16]. The VAE consists of an encoder p_ϕ, which maps states to latent distributions, and a decoder p_ψ, which maps latents to distributions over states. The encoder and decoder parameters, ψ and ϕ respectively, are jointly trained to maximize

$$\mathcal{L}(\psi, \phi; s^{(i)}) = -\beta D_{KL}(q_\phi(z|s^{(i)})||p(z)) + \mathbb{E}_{q_\phi(z|s^{(i)})}[\log p_\psi(s^{(i)} \mid z)], \tag{2}$$

where $p(z)$ is some prior, which we take to be the unit Gaussian, D_{KL} is the Kullback-Leibler divergence, and β is a hyperparameter that balances the two terms. The use of β values other than one is sometimes referred to as a β-VAE [14]. The encoder q_ϕ parameterizes the mean and log-variance diagonal of a Gaussian distribution, $q_\phi(s) = \mathcal{N}(\mu_\phi(s), \sigma^2_\phi(s))$. The decoder p_ψ parameterizes a Bernoulli distribution for each pixel value. This parameterization corresponds to training the decoder with cross-entropy loss on normalized pixel values.

4 Goal-Conditioned Policies with Unsupervised Representation Learning

To devise a practical algorithm based on goal-conditioned value functions, we must choose a suitable goal representation. In the absence of domain knowledge and instrumentation, a general-purpose choice is to set the goal space $\mathcal{G}$ to be the same as the state observations space $\mathcal{S}$. This choice is fully general as it can be applied to any task, and still permits considerable user control since the user can choose a "goal state" to set a desired goal for a trained goal-conditioned policy. But when the state space $\mathcal{S}$ corresponds to high-dimensional sensory inputs such as images, [1] learning a goal-conditioned Q-function and policy becomes exceedingly difficult as we illustrate empirically in Section 5.

Our method jointly addresses a number of problems that arise when working with high-dimensional inputs such as images: sample efficient learning, reward specification, and automated goal-setting.We address these problems by learning a latent embedding using a β-VAE. We use this latent space to represent the goal and state and retroactively relabel data with latent goals sampled from the VAE prior to improve sample efficiency. We also show that distances in the latent space give us a well-shaped reward function for images. Lastly, we sample from the prior to allow an agent to set and "practice" reaching its own goal, removing the need for humans to specify new goals during training time. We next describe the specific components of our method, and summarize our complete algorithm in Section 4.5.

4.1 Sample-Efficient RL with Learned Representations

One challenging problem with end-to-end approaches for visual RL tasks is that the resulting policy needs to learn both perception and control. Rather than operating directly on observations, we embed the state s_t and goals g into a latent space $\mathcal{Z}$ using an encoder e to obtain a latent state $z_t = e(s_t)$ and latent goal $z_g = e(g)$. To learn a representation of the state and goal space, we train a β-VAE by executing a random policy and collecting state observations, $\{s^{(i)}\}$, and optimize Equation (2). We then use the mean of the encoder as the state encoding, i.e. $z = e(s) \triangleq \mu_\phi(s)$.

After training the VAE, we train a goal-conditioned Q-function $Q(z, a, z_g)$ and corresponding policy $\pi_\theta(z, z_g)$ in this latent space. The policy is trained to reach a goal z_g using the reward function discussed in Section 4.2. For the underlying RL algorithm, we use twin delayed deep deterministic policy gradients (TD3) [12], though any value-based RL algorithm could be used. Note that the policy (and Q-function) operates completely in the latent space. During test time, to reach a specific goal state g, we encode the goal $z_g = e(g)$ and input this latent goal to the policy.

As the policy improves, it may visit parts of the state space that the VAE was never trained on, resulting in arbitrary encodings that may not make learning easier. Therefore, in addition to procedure described above, we fine-tine the VAE using both the randomly generated state observations $\{s^{(i)}\}$ and the state observations collected during exploration. We show in Section 5 that this additional training helps the performance of the algorithm.

4.2 Reward Specification

Training the goal-conditioned value function requires defining a goal-conditioned reward $r(s, g)$. Using Euclidean distances in the space of image pixels provides a poor metric, since similar configurations in the world can be extremely different in image space. In addition to compactly representing high-dimensional observations, we can utilize our representation to obtain a reward function based on a metric that better reflects the similarity between the state and the goal. One choice for such a reward is to use the negative Mahalanobis distance in the latent space:

$$r(s, g) = -||e(s) - e(g)||_A = -||z - z_g||_A,$$

where the matrix A weights different dimensions in the latent space. This approach has an appealing interpretation when we set A to be the precision matrix of the VAE encoder, q_ϕ. Since we use a Gaussian encoder, we have that

$$r(s, g) = -||z - z_g||_A \propto \sqrt{\log e_\phi(z_g \mid s)} \tag{3}$$

[1] We make the simplifying assumption that the system is Markovian with respect to the sensory input, and one could incorporate memory into the state for partially observed tasks.

In other words, minimizing this squared distance in the latent space is equivalent to rewarding reaching states that maximize the probability of the latent goal z_g. In practice, we found that setting $A = \mathbf{I}$, corresponding to Euclidean distance, performed better than Mahalanobis distance, though its overall effect is the same — to bring z close to z_g and maximize the probability of the latent goal z_g given the current observation. This interpretation would not be possible when using normal autoencoders since distances are not trained to have any probabilistic meaning. Indeed, we show in Section 5 that using distances in a normal autoencoder representation does not result in meaningful behavior.

4.3 Improving Sample Efficiency with Latent Goal Relabeling

To further enable sample-efficient learning in the real world, we use the VAE to relabel goals. Note that we can optimize Equation (1) using any valid (s, a, s', g, r) tuple. If we could artificially generate these tuples, then we could train our entire RL algorithm without collecting any data. However, we do not know the system dynamics, and therefore have to sample transitions (s, a, s') by interacting with the world. However, we have the freedom to relabel the goal and reward synthetically. In particular, if we have a mechanism for generating goals and computing rewards, then given (s, a, s'), we can generate a new goal g and new reward $r(s, a, s', g)$ to produce a new tuple (s, a, s', g, r). By artificially generating and recomputing rewards, we can convert a single (s, a, s') transition into potentially infinitely many valid training datums.

For image-based tasks, this procedure would require generating goal images, an onerous task on its own. However, our reinforcement learning algorithm operates directly in the latent space for goals and rewards. So rather than generating goals g, we generate latent goals z_g by sampling from the VAE prior $p(z)$. We then recompute rewards using Equation (3). By retroactively relabeling the goals and rewards, we obtain much more data to train our value function. This sampling procedure is made possible by our use of a VAE, which is explicitly trained so that sampling from the latent distribution is straightforward.

Retroactively generating goals is also explored in hindsight experience replay (HER) [2]. However, HER is limited to sampling goals seen along a trajectory, which greatly limits the number and diversity of goals with which one can relabel a given transition. Our final method uses a mixture of the two strategies: half of the goals are generated from the prior and half from goals seen along the trajectory. We show in Section 5 that relabeling the goal with samples from the VAE prior results in significantly better sample-efficiency.

4.4 Automated Goal-Generation for Exploration

If we do not know which particular goals will be provided at test time, we would like our RL agent to carry out a self-supervised "practice" phase during training, where the algorithm proposes its own goals, and then practices how to reach them. Since the VAE prior represents a distribution over latent goals and state observations, we again sample from this distribution to obtain plausible goals. After sampling a goal latent from the prior $z_g \sim p(z)$, we give this to our policy $\pi(z, z_g)$ to collect data.

Algorithm 1 GRiLL: Goal-conditioned reinforcement learning by representation learning

Require: VAE encoder q_ϕ, VAE decoder p_ψ, policy π_θ, goal-conditioned value function Q_w.
Collect $\mathcal{D} = \{s^{(i)}\}$ using exploration policy.
Train β-VAE on $\mathcal{D}$ by optimizing (2).
for $n = 0, ..., N-1$ episodes **do**
  Sample latent goal from prior $z_g \sim p(z)$.
  Sample initial state $s_0 \sim E$.
  for $t = 0, ..., H-1$ steps **do**
    Get action $a_t = \pi_\theta(e(s_t), z_g) +$ noise.
    Get next state $s_{t+1} \sim p(\cdot \mid s_t, a_t)$.
    Store (s_t, a_t, s_{t+1}, z_g) into replay buffer $\mathcal{R}$.
    Sample transition $(s, a, s', z_g) \sim \mathcal{R}$.
    Encode $z' = e(s')$.
    (Probability 0.5) replace z_g with $z'_g \sim p(z)$.
    Compute new reward $r = -||z' - z_g||$.
    Minimize (1) using (z, a, z', z_g, r).
  end for
  Fine-tune β-VAE every K episodes on mixture of $\mathcal{D}$ and $\mathcal{R}$.
end for

4.5 Algorithm Summary

We call the complete algorithm goal-conditioned reinforcement learning by representation learning (GRiLL) and summarize it in Algorithm 1. We first collect data with a simple exploration policy,

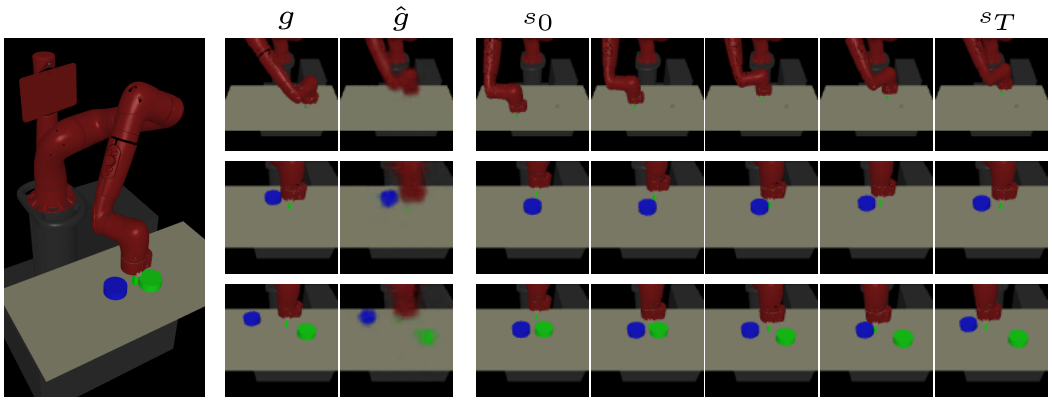

Figure 2: (Left) The simulated environment. (Right) Test rollouts from our learned policy on the three simulated environments. Each row is one rollout. The middle shows the goal image g and its VAE reconstruction $\hat{g}$. The right columns shows frames from the trajectory to reach the given goal.

though any exploration strategy could be used for this stage, including off-the-shelf exploration bonuses [25, 3] or unsupervised reinforcement learning methods [8, 11]. Then, we train a VAE on state data and fine tune the VAE throughout the algorithm. We use this VAE for multiple purposes: We sample a latent goal z_g from the VAE and condition the policy on this goal. We embed all states and goals using the VAE encoder. When we train our (latent) goal-conditioned value function, we resample goals from the VAE prior and compute rewards in the latent space using Equation (3). The overall RL algorithm is trained using TD3 [12].

5 Experiments

Our experiments address the following questions:

1. How does our method compare to prior model-free RL algorithms in terms of sample efficiency and performance, when learning continuous control tasks from images?
2. How critical is each component of our algorithm to sample efficient learning?
3. Does our method work on tasks where the state space cannot be easily specified ahead of time, such as tasks that require interaction with variable numbers of objects?
4. Can our method scale to real world vision-based robotic control tasks?

For the first two questions, we evaluate our method against a number of prior algorithms and ablated versions of our approach on a suite of simulated and real-world tasks: *Visual Reacher*: a MuJoCo [39] environment with a 7-dof Sawyer arm reaching goal positions. The arm is shown the left of Figure 2. The end-effector (EE) is constrained to a 2-dimensional rectangle parallel to a table. The action controls EE velocity within a maximum velocity. *Visual Pusher*: a MuJoCo environment with a 7-dof Sawyer arm and a small puck on a table that the arm must push to a target push. *Visual Multi-Object Pusher*: a copy of the Visual Pusher environment with two pucks. Detailed descriptions of the environments are provided in the Supplementary Material.

Solving these tasks directly from images poses a challenge since the controller must learn both perception and control. The evaluation metric is the distance of objects (including the arm) to their respective goals. To evaluate our policy, we set the environment to a sampled goal position, capture an image, and encode the image to use as the goal. [2] Although we use the ground-truth positions for evaluation, **we do not use the ground-truth positions for training the policies**. The only inputs from the environment that our algorithm receives are the image observations. For Visual Reacher, we pretrained the VAE with 100 images. For other tasks, we used 10000 images.

We compare our method with the following prior works. *L&R*: Lange and Riedmiller [17] trains an autoencoder to handle images. *DSAE*: Deep spatial autoencoders [10] learns a spatial autoencoder and uses guided policy search [19] to achieve a single goal image. *HER*: Hindsight experience replay [2] utilizes a sparse reward signal and relabeling trajectories with achieved goals. *Oracle*: RL with direct access to state information for observations and rewards.

[2] In all our simulation results, each plot shows a 95% confidence interval of the mean across 5 seeds.

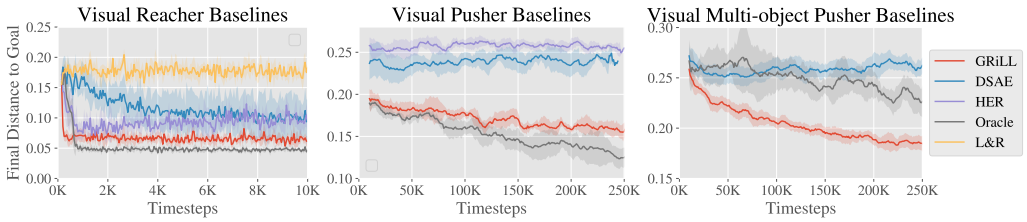

Figure 3: Simulations results, showing final distance to goal vs environment steps[4]. GRiLL (red) consistently outperforms the baselines. Methods that failed to learn on easier tasks were not tested on harder tasks. See text for description of environment and baselines.

To our knowledge, no prior work demonstrates policies that achieve a variety of goal images without access to a true-state reward function, and so we needed to make modifications to make the comparisons feasible. L&R assumes a reward function from the environment. Since we have no state-based reward function, we specify the reward function as distance in the autoencoder latent space. HER does not embed inputs into a latent space but instead operates directly on the input, so we use pixel-wise mean squared error (MSE) as the metric. DSAE is trained only for a single goal, so we allow the method to generalize to a variety of test goal images by using goal-conditioned TD3. To make the implementations comparable, we also use TD3 to train L&R and HER. Unlike our method, prior methods do not specify how to select goals during training, so we favorably give them real images as goals for rollouts, sampled from the same distribution that we use to test.

We see in Figure 3 that our method can efficiently learn policies from visual inputs to perform simulated reaching and pushing, without access to the state information. Our approach substantially outperforms the prior methods, for which the use of image goals and observations poses a severe challenge. HER struggles because pixel-wise MSE is extremely hard to optimize; our latent-space rewards are much better shaped and allow us to learn even on more complex tasks. Finally, our method is close to the state-based "oracle" method in terms of sample efficiency and performance, without having any access to state information. Notably, in the multi-object environment, our method actually outperforms the oracle, likely because the state-based reward contains local minima. Overall, these result show that our method is capable of handling raw image observations much more effectively than previously proposed goal-conditioned RL methods. Next, we perform ablations on Visual Pusher to evaluate our contributions in isolation. For results on all environments, see the Supplementary Material.

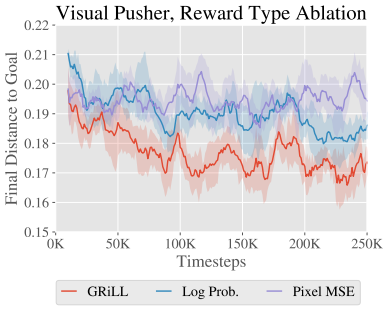

Figure 4: Reward type ablation results. GRiLL (red) uses latent Euclidean distance outperforms the other methods.

Reward Specification Comparison We evaluate how effective distances in the VAE latent space are for the Visual Pusher task. We keep our method the same, and only change the reward function that we use to train the goal-conditioned valued function. We include the following methods for comparison: *Latent Distance*, which is the reward used in GRiLL, i.e. $A = \mathbf{I}$ in Equation (3); *Log Probability*, which uses the Mahalanobis distance in Equation (3), where A is the precision matrix of the encoder; and *Pixel MSE*, which computes mean-squared error (MSE) between state and goal in pixel space.[5] In Figure 4, we see that latent distance significantly outperforms the log probability. We suspect that small variances of the VAE encoder results in drastically large rewards, making the learning more difficult. We also see that latent distances results in faster learning when compared to pixel MSE.

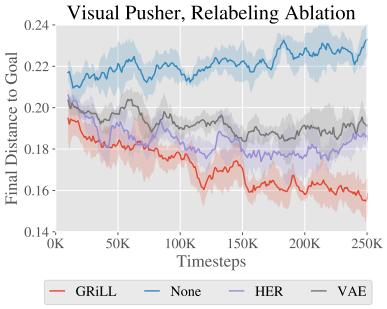

Figure 5: Relabeling ablation results. GRiLL (red), which uses a mixture of VAE and future, outperforms the other methods.

Relabeling Strategy Comparison As described in section 4.3, our method uses a novel goal relabeling method based on sampling from the VAE prior. To isolate how much our new goal

[5]To compute the pixel MSE for a sampled latent goal, we decode the goal latent using the VAE decoder, p_ψ, to generate the corresponding goal image.

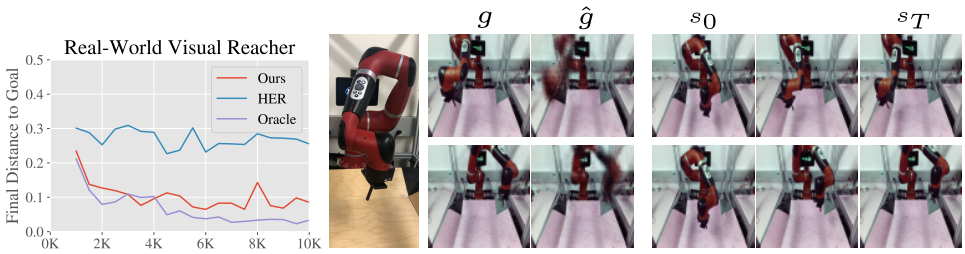

Figure 7: (Left) Our method compared to the HER baseline and oracle on a real-world visual reaching task. (Middle) Our robot setup is pictured. (Right) Test rollouts of our learned policy.

relabeling method contributes to our algorithm, we vary the resampling strategy while fixing other components of our algorithm. The resampling strategies that we consider are: *HER*, relabeling the goal for a transition by sampling uniformly from future states in the trajectory as done in Andrychowicz et al. [2]; *VAE*, sampling goals from the VAE only; *GRiLL*, relabeling goals with probability 0.5 from the VAE and probability 0.5 using the future strategy; and *None*, no relabeling. In Figure 5, we see that sampling from the VAE is significantly better than HER or not relabeling at all. Lastly, a mixture of the the VAE and HER sampling performs the best, which we use in GRiLL.

Learning with Variable Numbers of Objects A major advantage of working directly from pixels is that the policy input can easily represent combinatorial structure in the environment, which would be difficult to encode into a fixed-length state vector even if a perfect perception system were available. For example, if a robot has to interact with different combinations and numbers of objects, picking a single MDP state representation would be challenging, even with access object poses. By directly processing images for both the state and the goal, no modification is needed to handle the combinatorial structure: the number of pixels always remains the same, regardless of how many objects are in the scene.

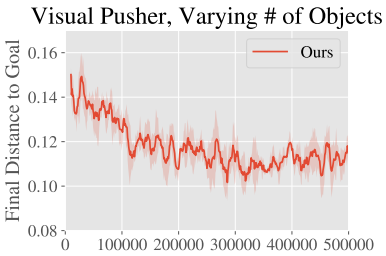

Figure 6: Training curve for learning with varying number of objects.

We demonstrate that our method can handle this difficult scenario by evaluating on a task where the environment, based on the Visual Multi-Object Pusher, randomly contains zero, one, or two objects in each episode during testing. During training, each episode still always starts with both objects in the scene, so the experiments tests whether a trained policy can handle variable numbers of objects at test time. Figure 6 shows that our method can learn to solve this task successfully, without decrease in performance from the base setting where both objects are present (in Figure 3). Developing and demonstrating algorithms that solve tasks with varied underlying structure is an important step toward creating autonomous agents that can handle the diversity of tasks present "in the wild."

5.1 Visual RL with Physical Robots

GRiLL is a practical and straightforward algorithm to apply to real physical systems: the efficiency of off-policy learning with goal relabeling makes training times manageable, while the use of image-based rewards through the learned representation frees us from the burden of manually design reward functions, which itself can require hand-engineered perception systems [33]. We trained policies for visual reaching on a real-world Sawyer robotic arm, shown in Figure 7. The control setup matches Visual Reacher, meaning that **the only input from the environment is images**. We see in Figure 7 that our method is applicable to real-world robotic tasks, almost matching the state-based oracle method and far exceeding the baseline method. Our method is able to solve this visually complex task in just 10,000 samples, which is about an hour of real-world interaction time.

6 Discussion and Future Work

In this paper, we present a new RL algorithm that can efficiently solve goal-conditioned, vision-based tasks without access to any ground truth state or reward functions. Our method trains a VAE which is used for multiple purposes: we embed the state and goals using the encoder; we sample from the prior to generate goals for exploration; we also sample latents to retroactively relabel goals and rewards;

and we use distances in the latent space for rewards to train a goal-conditioned value function. We show that these contributes culminate in a sample efficient algorithm that works directly from vision. As a result, we are able to apply our method to a variety of simulated visual tasks, including a variable-object task that cannot be easily represented with a fixed length vector, as well as a real world robotic task.

References

[1] Pulkit Agrawal, Ashvin Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to Poke by Poking: Experiential Learning of Intuitive Physics. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[2] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob Mcgrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight Experience Replay. In *Advances in Neural Information Processing Systems (NIPS)*, jul 2017.

[3] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1471–1479, 2016.

[4] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2172–2180, 2016.

[5] Brian Cheung, Jesse A Livezey, Arjun K Bansal, and Bruno A Olshausen. Discovering hidden factors of variation in deep networks. *arXiv preprint arXiv:1412.6583*, 2014.

[6] Guillaume Desjardins, Aaron Courville, and Yoshua Bengio. Disentangling factors of variation via generative entangling. *CoRR*, abs/1210.5, 2012.

[7] Frederik Ebert, Chelsea Finn, Alex X Lee, and Sergey Levine. Self-Supervised Visual Planning with Temporal Skip Connections. In *Conference on Robot Learning (CoRL)*, 2017.

[8] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is All You Need: Learning Skills without a Reward Function. *arXiv preprint arXiv:1802.06070*, 2018.

[9] Chelsea Finn and Sergey Levine. Deep Visual Foresight for Planning Robot Motion. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

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

[11] Carlos Florensa, Yan Duan, and Pieter Abbeel. Stochastic neural networks for hierarchical reinforcement learning. *arXiv preprint arXiv:1704.03012*, 2017.

[12] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing Function Approximation Error in Actor-Critic Methods. *arXiv preprint arXiv:1802.09477*, 2018.

[13] David Ha and Jürgen Schmidhuber. World Models. *arXiv preprint arXiv:1803.10122*, 2018.

[14] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-VAE: Learning basic visual concepts with a constrained variational framework. *International Conference on Learning Representations (ICLR)*, 2017.

[15] Irina Higgins, Arka Pal, Andrei A Rusu, Loic Matthey, Christopher P Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, and Alexander Lerchner. Darla: Improving zero-shot transfer in reinforcement learning. *International Conference on Machine Learning (ICML)*, 2017.

[16] Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014.

[17] Sascha Lange and Martin A Riedmiller. Deep learning of visual control policies. In *European Symposium on Artificial Neural Networks (ESANN)*. Citeseer, 2010.

[18] Sascha Lange, Martin Riedmiller, Arne Voigtlander, and Arne Voigtländer. Autonomous reinforcement learning on raw visual input data in a real world application. In *International Joint Conference on Neural Networks (IJCNN)*, number June, pages 1–8. IEEE, 2012.

[19] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-End Training of Deep Visuomotor Policies. *Journal of Machine Learning Research (JMLR)*, 17(1):1334–1373, 2016. ISSN 15337928. doi: 10.1007/s13398-014-0173-7.2.

[20] Sergey Levine, Peter Pastor, Alex Krizhevsky, and Deirdre Quillen. Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection. *International Journal of Robotics Research*, 2017.

[21] Andrew Levy, Robert Platt, and Kate Saenko. Hierarchical Actor-Critic. *arXiv preprint arXiv:1712.00948*, 2017.

[22] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2016.

[23] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Tim Harley, Timothy P Lillicrap, David Silver, Koray Kavukcuoglu, Korayk@google Com, and Google Deepmind. Asynchronous Methods for Deep Reinforcement Learning. In *International Conference on Machine Learning (ICML)*, 2016.

[24] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard Lewis, and Satinder Singh. Action-Conditional Video Prediction using Deep Networks in Atari Games. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[25] Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-Driven Exploration by Self-Supervised Prediction. In *International Conference on Machine Learning (ICML)*, pages 488–489. IEEE, 2017.

[26] Deepak Pathak, Parsa Mahmoudieh, Guanghao Luo, Pulkit Agrawal, Dian Chen, Yide Shentu, Evan Shelhamer, Jitendra Malik, Alexei A Efros, and Trevor Darrell. Zero-Shot Visual Imitation. In *International Conference on Learning Representations (ICLR)*, 2018.

[27] Lerrel Pinto and Abhinav Gupta. Supersizing Self-supervision: Learning to Grasp from 50K Tries and 700 Robot Hours. *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[28] Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. Asymmetric Actor Critic for Image-Based Robot Learning. *arXiv preprint arXiv:1710.06542*, 2017.

[29] Vitchyr Pong, Shixiang Gu, Murtaza Dalal, and Sergey Levine. Temporal Difference Models: Model-Free Deep RL For Model-Based Control. In *International Conference on Learning Representations (ICLR)*, 2018.

[30] Nikolay Ponomarenko, Lina Jin, Oleg Ieremeiev, Vladimir Lukin, Karen Egiazarian, Jaakko Astola, Benoit Vozel, Kacem Chehdi, Marco Carli, Federica Battisti, and Others. Image database TID2013: Peculiarities, results and perspectives. *Signal Processing: Image Communication*, 30:57–77, 2015.

[31] Paulo Rauber, Filipe Mutz, and Juergen Jürgen Schmidhuber. Hindsight policy gradients. In *CoRR*, volume abs/1711.0, 2017.

[32] Scott Reed, Kihyuk Sohn, Yuting Zhang, and Honglak Lee. Learning to disentangle factors of variation with manifold interaction. In *International Conference on Machine Learning*, pages 1431–1439, 2014.

[33] Andrei A Rusu, Matej Vecerik, Thomas Rothörl, Nicolas Heess, Razvan Pascanu, and Raia Hadsell. Sim-to-real robot learning from pixels with progressive nets. *Conference on Robot Learning (CoRL)*, 2017.

[34] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal Value Function Approximators. In *International Conference on Machine Learning (ICML)*, pages 1312–1320, 2015.

[35] Jürgen Schmidhuber. Learning factorial codes by predictability minimization. *Neural Computation*, 4(6): 863–879, 1992.

[36] Linda Smith and Michael Gasser. The development of embodied cognition: Six lessons from babies. *Artificial life*, 11(1-2):13–29, 2005.

[37] Richard S Sutton, Joseph Modayil, Michael Delp, Thomas Degris, Patrick M Pilarski, Adam White, and Doina Precup. Horde: A Scalable Real-time Architecture for Learning Knowledge from Unsupervised Sensorimotor Interaction. *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 10:761–768, 2011.

[38] Valentin Thomas, Jules Pondard, Emmanuel Bengio, Marc Sarfati, Philippe Beaudoin, Marie-Jean Meurs, Joelle Pineau, Doina Precup, and Yoshua Bengio. Independently Controllable Factors. In *NIPS Workshop*, 2017.

[39] Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *The IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012.

[40] Manuel Watter, Jost Tobias Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to Control: A Locally Linear Latent Dynamics Model for Control from Raw Images. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2728–2736, 2015.

[41] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. *arXiv preprint arXiv:1801.03924*, 2018.