Achieving Gentle Manipulation with Deep Reinforcement Learning

Sandy H. Huang
University of California, Berkeley
shhuang@cs.berkeley.edu

Martina Zambelli, Yuval Tassa, Jackie Kay,
Murilo F. Martins, Patrick M. Pilarski, Raia Hadsell
DeepMind
{zambellim, tassa, kayj, murilomartins, ppilarski, raia}@google.com

Abstract

Robots must know how to be *gentle* when they need to interact with fragile objects, or when the robot itself is prone to wear-and-tear. We propose an approach that enables deep reinforcement learning to train policies that are gentle, both during exploration and task execution. Our approach involves augmenting the (task) reward with a penalty for non-gentleness. However, augmenting with only this penalty impairs learning: policies get stuck in a local optimum of avoiding all contact with the environment. Introducing surprise-based intrinsic rewards solves this problem, as long as the right kind of surprise is chosen—penalty-based surprise is more effective than the typical dynamics-based surprise. Videos are available at http://sites.google.com/view/gentlemanipulation.

1 Introduction

Deep reinforcement learning (RL) can be used to train policies that achieve superhuman performance on Atari games [20] and Go [29], learn locomotion tasks [18, 27], and perform complex robotic manipulation skills [17]. However, deploying deep RL on real-world robots often leads to a considerable amount of wear-and-tear over time, on both the robot itself and the environment, because existing approaches require many trials to learn. If robots were able to explore and learn safely, minimizing excessive forces and impacts, they would last longer before needing repairs, and the objects they interact with would not need to be replaced as often.

We might also care about gentleness in terms of task execution itself, for instance if the robot needs to pick-and-place an object, but either the object and/or goal location is fragile. In particular, when robot manipulation involves humans (e.g., feeding a paralyzed patient), being gentle is important [15]. In these situations, robots should strive to accomplish the given task in a reasonable amount of time, while minimizing applied force and impact as much as possible.

Thus, in order to broadly deploy deep RL on real robots, we need an approach for training policies that are gentle, both during exploration and task execution. A naive approach is to constrain the maximum torques that a robot's motors can exert. However, many manipulation tasks require occasional, momentary, or variable high force (e.g., hammering a nail or turning a lever); the torque limit cannot be any lower than this, otherwise the robot will not be able to complete the task. But we do not want the robot to freely exert this much force along its entire trajectory. Alternatively, one could constrain the total amount of force or impact allowed, but this requires knowing *a priori* the minimum total amount necessary for accomplishing the task [2, 4, 32].

Instead, our approach is to give the robot negative rewards for actions that are not gentle, for instance those that result in high impact forces. Incorporating this in the reward function is a natural approach for encoding preferences about *how* robots should perform a task (e.g., driving style [1]), and can be seen as an intrinsic "pain" signal which encourages learning safer policies. However, perhaps unsurprisingly, we show adding only this penalty makes it much harder for an agent to learn an optimal policy; instead, they get stuck in a local optimum of avoiding contact altogether, because they encounter the penalties before ever obtaining the task reward, and thus learn a fear of pain.

To motivate agents to interact with the environment *and* do it gently, we propose balancing this "pain" signal by adding another intrinsic signal, this one positive, for curiosity. In particular, we reward the agent for *surprising* experiences—those that contradict the agent's current understanding of the world. A concrete example of this is giving intrinsic rewards for transitions that have low probability under a learned dynamics model [3, 12, 23, 30]. However, we find that using this kind of *dynamics-based surprise* is not as effective as using a *penalty-based surprise*, that leads robots to be explicitly curious about the non-gentleness penalty itself.

Our work takes a step toward using deep RL to train policies for gentle, object- and contact-centric manipulation. We define being gentle as minimizing impact forces. We demonstrate that our proposed approach, which introduces both a penalty for excessive impact forces and a curiosity reward focused on this penalty, enables efficient and safe exploration, precise task execution, and successful manipulation of fragile objects.

2 Related Work

Our goal of gentle manipulation is closely related to impact minimization in classical control, but existing approaches typically rely on having accurate dynamical contact models [13, 14, 34]. Another related domain is safe reinforcement learning [11, 24]: safety may refer to either physical safety or handling environment stochasticity. Typically, the former is concerned with avoiding catastrophic situations (e.g., crashing into another car or falling off a cliff), whereas in our work, we take a broader view of physical safety, in terms of reducing wear-and-tear in order to delay *eventual* damage.

We use curiosity-based intrinsic rewards to encourage agents to (gently) explore their environment, in the presence of non-gentleness penalties. Curiosity-based intrinsic rewards can be viewed as providing reward shaping [22]: they make tasks easier to learn, by making it less likely for agents to get stuck in undesirable local optima. In the context of deep RL, curiosity-based intrinsic rewards typically reward agents for either encountering novel states [7, 10, 31] or encountering surprising experiences [3, 12, 23, 30]. Our work uses the latter, surprise-based intrinsic rewards, but we find that the usual method of computing this with respect to a learned dynamics model is not effective in our setting. Instead, we formulate surprise with respect to a penalty-prediction model.

In our work, the reward comes from a combination of several signals: extrinsic rewards from the task, and intrinsic rewards from non-gentleness and curiosity. This falls under multi-objective reinforcement learning (MORL). Typically MORL approaches either train a single policy by finding the right balance of rewards, or learn a set of policies that approximate the Pareto optimal frontier [19, 25]. Some authors have used the Constrained MDP framework [4] to develop policy optimization methods which ensure that constraints are satisfied at all points throughout learning [2, 8]. We take a separate standpoint, in which we trade off explicitly between the non-gentleness penalty, or intrinsic "pain" signal, and the agent's curiosity about it. This reduces the problem to balancing between the task reward and the this penalty, which can be set with domain knowledge: if it is very important to achieve the task, then a higher task reward implies that being less gentle (e.g., experiencing higher impact forces) is okay. Thus, we can directly add these rewards together, rather than searching for a weighting that leads to the correct balance.

3 Preliminaries

3.1 Markov Decision Process

A Markov Decision Process (MDP) is defined as a tuple (S, A, P, R, γ), where S is the state space, A is the action space, $P : S \quad A \quad S \to \mathbb{R}$ specifies the transition probabilities, $R : S \quad A \quad S \to \mathbb{R}$ specifies the reward function, and $\gamma \in [0, 1]$ is the discount factor.

A policy π is a function that maps each state to a distribution over actions ($\pi : S \; ! \; \Delta_A$, where Δ_A is the probability simplex on A). Reinforcement learning optimizes policies to maximize expected returns (i.e., cumulative discounted future rewards):

$$\mathsf{E}_{a_t \;\; \pi(s_t), s_{t+1} \;\; P(s_t, a_t)}[\sum_t \gamma^t R(s_t, a_t, s_{t+1})]. \tag{1}$$

A policy π's action-value function, when taking action a in state s, is

$$Q^\pi(s, a) = \int_{s^0} P(s, a, s^0) \left(R(s, a, s^0) + \gamma \mathsf{E}_{a^0 \;\; \pi(s^0)}[Q^\pi(s^0, a^0)] \right). \tag{2}$$

Typically R specifies how well the policy is doing in terms of accomplishing a task. In this work, we augment R with several types of reward bonuses, in order to train policies for contract-centric, low-impact manipulation.

3.2 Deep Reinforcement Learning

The policy π can be represented by a function parameterized by θ; for instance, θ may be a weighting on predefined features of the state [1]. In deep RL, θ is the parameterization of a neural network. We use Distributed Distributional Deterministic Policy Gradients (D4PG) [6] to train our policies, but in principle the proposed approach is algorithm-agnostic. D4PG is an actor-critic algorithm used to train policies for continuous control; both the policy and action-value function are parameterized by a neural network.

The critic is a distributional action-value function: it takes the current state s_t and action a_t as input, and outputs a categorical distribution over the predicted $Q(s_t, a_t)$. It is trained with off-policy policy evaluation, on batches of transitions (s_t, a_t, s_{t+1}) sampled from a replay buffer. The actor is a deterministic policy: it takes in the current state s_t as input, and outputs an action a_t. During training, its gradients are computed only with respect to the critic, such that actions are adjusted in the direction of increased Q-values.

3.3 Formalizing gentleness

In this work, we define being gentle as minimizing impact. This is closely related to the notion of impact force in physics, which is the maximum amount of force experienced during a collision. However, we consider a more general definition of "impact", that does not only apply to cases when the initial applied force is zero. Instead, assuming a discrete time step, we define impact m_t as

$$m_t = \max(0, f_{t+1} \quad f_t), \tag{3}$$

where f_t is the sensed force at time step t. In other words, for a robot to be gentle, it should minimize *increases in sensed force*.[1]

4 Proposed Approach

In order to train policies that exhibit gentle manipulation, our approach is to augment the original reward (r_t) with an impact force penalty (r_t^f) and an intrinsic reward based on surprise (r_t^s). Agents are trained to maximize the total reward,

$$r_t^0 = r_t + r_t^f + r_t^s. \tag{4}$$

4.1 Impact penalty

The impact force penalty acts as an intrinsic pain signal to encourage agents to accomplish manipulation tasks in a more gentle way. Of course, in order to accomplish any manipulation task, small impacts are necessary—at some point the robot needs to go from zero to non-zero applied force on

[1] As motivation, consider a robot that needs to apply a force of 20N to push a heavy object. If the robot increases the amount of force it applies from zero to 20N in a fraction of a second, the impact results in more potential wear-and-tear, compared to increasing gradually to 20N over several seconds.

an object, in order to manipulate it. So, the impact penalty should scale non-linearly with the level of impact, by taking into account the *acceptability* of a particular amount of impact.

Let $a_\lambda(m)$ 2 $[0, 1]$, parametrized by λ, be the acceptability of a particular amount of impact m. This is a monotonically increasing function, that should be designed according to how resilient the robot and environment are to impacts; for instance, if the robot is interacting with particularly fragile objects, then the range of acceptable impacts should be smaller. In our experiments, we use a sigmoid function for acceptability:

$$a_\lambda(m) = \text{sigmoid}(\lambda_1(\quad x + \lambda_2)) = \frac{1}{1 + e^{\lambda_1(x \quad \lambda_2)}} \tag{5}$$

The impact penalty at time step t is:

$$r_t^f = \quad \sum_i a_\lambda(m_t^i) \quad m_t^i, \tag{6}$$

where the summation is across force sensors at different locations on the robot (e.g., the fingers of a robot hand). In our experiments, we set $\lambda = [2, 2]^>$.

4.2 Dynamics-based surprise

If the environment reward merely combines the task reward and the impact penalty, that is, $r_t^0 = r_t + r_t^f$, we find that policies get reliably stuck in a local optimum of not making contact with anything in the environment—the agent learns to be afraid of contact, since it encounters the impact penalty before the sparse task reward, hindering exploration. The purpose of adding surprise-based intrinsic rewards is to encourage policies to make *contact* with objects in the environment but still in a *gentle* way.

For an agent to be "surprised," it must have some (learned) understanding of its environment, i.e., a model. In the case of dynamics-based surprise, this model is a dynamics model that takes in the current state and action, and predicts the next state. We train an ensemble of neural networks for the dynamics model, in order to have predictive uncertainty [16]. Predictive uncertainty is useful for capturing novelty: in the case of environments with deterministic dynamics, if the networks in the ensemble either individually have high variance in their predictions, or have high variance across the ensemble, then this indicates a novel area that should be explored further.

Each of the M networks in the ensemble outputs the mean and variance of a Gaussian for each dimension d of the prediction. The ensemble's combined output is a mixture of Gaussians for each output dimension d:

$$\frac{1}{M} \sum_i \mathsf{N}\left(\mu_{\theta_i}(\mathbf{x})_d, \sigma^2_{\theta_i}(\mathbf{x})_d\right),$$

where $\mathbf{x}$ denotes the input and θ_i are the parameters of the ith network in the ensemble. During training, each network is randomly initialized, and they are trained on different batches of transitions. We choose $M = 5$, as recommended by related research [16].

To compute dynamics-based surprise intrinsic reward r_t^s, we approximate the dynamics model's predicted distribution over next states with a single Gaussian per output dimension d, to measure how much variance there is *across* networks in the ensemble. This has mean $\mu \ (\mathbf{x})_d = \frac{1}{M} \sum_i \mu_{\theta_i}(\mathbf{x})_d$ and variance $\sigma^2(\mathbf{x})_d = \frac{1}{M} \sum_i (\sigma^2_{\theta_i}(\mathbf{x})_d + \mu^2_{\theta_i}(\mathbf{x})_d) \quad \mu^2(\mathbf{x})_d$. Then the intrinsic reward is the negative log-likelihood of the true next state under this predicted distribution over next states:

$$r_t^s = \quad \sum_d \log p(s_{t+1,d} \mathsf{j} \mathsf{N}\left(\mu \ (s_t, a_t)_d, \sigma^2(s_t, a_t)_d\right) \tag{7}$$

This intrinsic reward is computed with respect to a target dynamics model, which is updated every 5000 iterations; this makes training more stable, so that the agent is not trying to surprise a model that is constantly changing. In addition, we do not provide intrinsic rewards to the agent until after 20,000 training iterations, once the dynamics model starts becoming more accurate.

4.3 Penalty-based surprise

Perhaps counter-intuitively, we propose to reward the agent for being curious about the impact penalty itself, by adding a reward to focus the learning and exploration of the agent on the intrinsic pain signal, thus enabling better prediction of pain through gentle interaction. To compute this penalty-based surprise reward $r_t^{s_p}$, we train an impact penalty predictor in parallel with the agent, with the same implementation as the general dynamics model (an ensemble of five neural networks).

We compute the intrinsic reward differently though—the problem with directly using negative log-likelihood is that then areas of high penalty are acceptable, as long as the prediction likelihood in those areas is low. However, this leads to excessive non-gentle behavior, which is not what we want: instead, we would like agents to focus on learning about areas with low penalty (i.e., areas where impact is low), so that they learn how to be gentle. For areas of high penalty, it is enough for the agent to just know that the penalty is high, not necessarily *exactly* how high it is. To enforce this preference for exploring areas with low penalty while avoiding ones with high penalty, a natural approach is to augment the task reward with a convex combination between the negative log-likelihood and the impact penalty:

$$a_{\lambda^{\theta}}(r_t^f) \qquad \log p(r_t^f \mathsf{jN}\,(\mu\ \ (s_t, a_t), \sigma^{\mathsf{2}}(s_t, a_t))) + (1 \quad a_{\lambda^{\theta}}(r_t^f)) \quad r_t^f, \tag{8}$$

where $a_{\lambda^{\theta}}(r_t^f)\ \ \mathsf{2}\ \ [0, 1]$ is the acceptability of a particular penalty r_t^f. This is a monotonically increasing function, that should be chosen based on how much penalty the robot may experience for the sake of exploration or task completion. We use a sigmoid function for this acceptability, as we did for impact $a_\lambda(m)$ (Sec. 4.1).

In order to augment the reward with this convex combination, we need to choose $r_t^{s_p}$ such that $r_t^{s_p} + r_t^f$ is equal to (8). In addition, we only provide this intrinsic reward if the penalty is non-zero, because the purpose is to encourage the agent to (cautiously) learn more about the penalty. Based on this, we set the penalty-based surprise intrinsic reward to be:

$$r_t^{s_p} = \begin{cases} a_{\lambda^{\theta}}(r_t^f) \left[\quad \log p(r_t^f \mathsf{jN}\,(\mu\ \ (s_t, a_t), \sigma^{\mathsf{2}}(s_t, a_t))) \quad r_t^f \right] & \text{if } r_t^f < 0 \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

In the same way as dynamics-based surprise, this penalty-based surprise intrinsic reward is computed with respect to a target impact penalty predictor model, which is updated every 1000 iterations, and we do not provide intrinsic rewards to the agent until after 20,000 training iterations.

4.4 Agent architecture

As mentioned, we use D4PG to train an actor (e.g., policy) and critic (e.g., action-value function). We use a separate critic for each of the task reward, surprise-based intrinsic reward, and impact penalty for more stable learning [26]. More details are provided in the Supplementary material.

5 Experiments

Our goal is to learn policies that are safer, with less forceful impacts, while also improving sample efficiency and overall task performance. The following experiments compare three approaches for achieving this; these approaches differ in terms of what the task reward is augmented with:

- an impact penalty (r_t^f)
- an impact penalty and a dynamics-based surprise intrinsic reward ($r_t^f + r_t^s$)
- an impact penalty and a penalty-based surprise intrinsic reward ($r_t^f + r_t^{s_p}$).

5.1 Experimental domain

Our experiments are on a simulated Shadow Dexterous Hand [28] in MuJoCo [33]. This robot hand has five fingers with a total of 24 degrees of freedom, actuated by 20 motors. Each fingertip has a spatial touch sensor, with three channels and a spatial resolution of $4 \quad 4$: one for normal force and two for tangential forces.[2] We simplify this by taking the absolute value and then summing across

[2] On the real-world Shadow Hand, BioTac® sensors [9] provide a more complex set of tactile signals.

the spatial dimensions, to obtain a 3D force vector for each fingertip. The impact force m_t^i is then the sum over the increase in force per channel for fingertip i.

The state consists of proprioception (joint position and joint velocity) and touch. The action space is 20-dimensional. We use position control and a control rate of 20 Hz.

The simulation environment consists of the Shadow Hand and a single block (Fig. 2); the task reward r_t depends on the experiment. Focusing on this simple environment enables us to clearly characterize the effectiveness of our three approaches for training low-impact policies. We find that even in this simple environment, learning policies for gentle manipulation is challenging for most approaches.

5.2 Exploration with impact penalty

First, we are interested in whether these approaches enable training policies that are gentle during exploration. We investigate this in a no-reward setting, where the policy receives intrinsic rewards (either from dynamics-based surprise or penalty-based surprise) and the intrinsic pain penalty, but no task reward. The goal is for policies to be gentle (i.e., experience low impact) while still exploring effectively, in terms of interacting with objects in the environment.

As a baseline, we trained policies with only dynamics-based surprise intrinsic rewards, without an impact penalty. As expected, these policies experience a large amount of impact while exploring: the maximum amount of impact experienced per rollout is in the 5 to 15N range (Fig. 1, left). This suggests that this form of curiosity is not feasible for running on real-world robots, if either the robot or the objects it interacts with are susceptible to wear-and-tear.

When we add the impact penalty, we do observe more gentle exploration: there is a significant decrease in the maximum amount of impact experienced per rollout (now in the 0 to 5N range), for both kinds of intrinsic reward. However, having penalty-based surprise intrinsic rewards leads to more gentle touching, whereas dynamics-based surprise leads to the policy exploring interesting configurations of the hand, but with limited touching (Fig. 1, center and right).

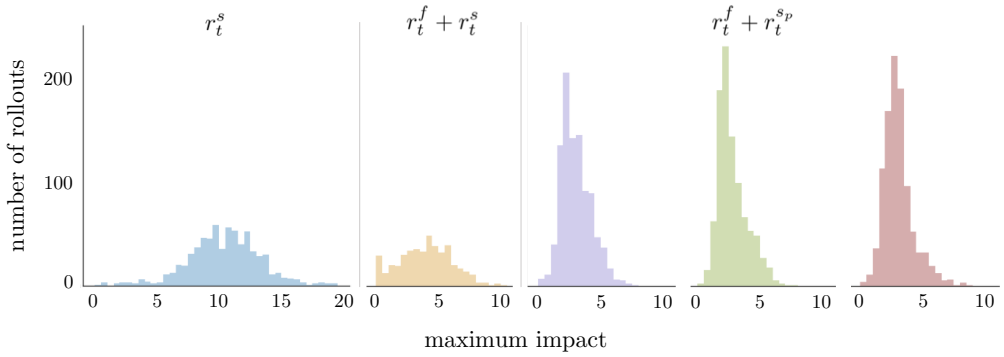

Figure 1: We train policies in a no-reward setting, with either dynamics-based (r_t^s) or penalty-based ($r_t^{s_p}$) surprise intrinsic rewards. These histograms show the maximum amount of impact (in Newtons) per rollout; rollouts are collected regularly throughout 500k training steps, and rollouts with no impact at all are ignored. $\lambda_2^0 = 2$ ■, 3 ■, or 4 ■. $\lambda_1^0 = 2$ for all.

5.3 Manipulation with impact penalty

Next, we are interested in whether these approaches enable training policies that learn how to perform a task gently, while still being relatively sample-efficient compared to the baseline of learning the task without caring about gentleness. In the task, the episode terminates with a reward of +1 if the hand presses the block with any fingerpad (thus activating the touch sensor) with a force greater than 5N. A non-gentle way of achieving this is to go from no contact to 5N of applied force in a single timestep; in contrast, policies trained to be gentle should more gradually increase to 5N of applied force.

This task is simple: without an impact penalty, agents learn this task quickly (Fig. 3, left), although with a significant amount of impact (Fig. 2, top center). However, once impact penalties are added, if there is no form of surprise-based intrinsic rewards to counteract them, then agents fail to learn

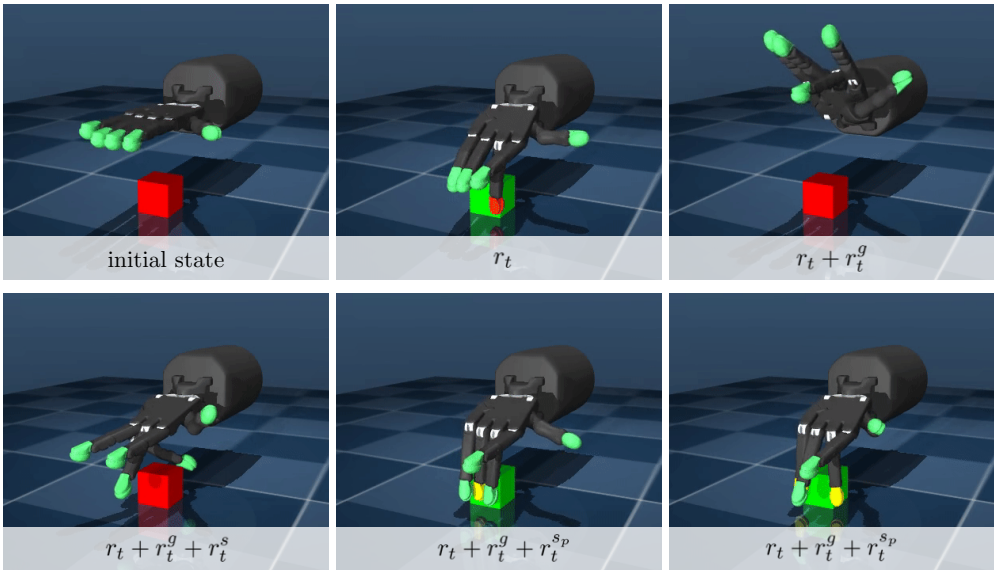

Figure 2: When the robot touches the block with greater than 5N of force, it receives a task reward of +1 and the block turns green. Fingertip color for non-zero impact forces ranges between yellow and red: it is purely yellow for an impact of near-zero, and purely red for a high impact of 10N. Policies trained on only task reward, r_t, learn how to do the task, but do it a high-impact way. In contrast, policies trained on the combination of task reward, an impact penalty, and penalty-based surprise intrinsic reward ($r_t + r_t^f + r_t^{s_p}$) learn to achieve the task in a *gentle* (i.e., low-impact) way, by gradually increasing force applied to the block. Without this particular kind of intrinsic reward, policies get trapped in a local optimum of avoiding contact with the environment ($r_t + r_t^f$ and $r_t + r_t^f + r_t^s$). (Each snapshot is obtained by executing the policy after 500k training steps, and taking the last state in the rollout. The bottom center and right snapshots are from two policies trained with the same reward augmentation, from different random initializations.) Corresponding videos are available at `http://sites.google.com/view/gentlemanipulation`.

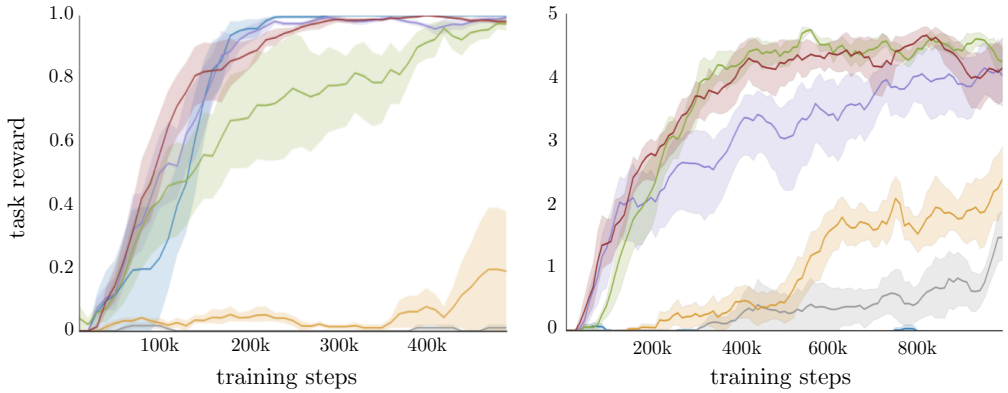

Figure 3: Learning curves for training policies on different reward augmentations (with five random seeds each), for two tasks: pressing a block (**left**) or a *fragile* block (**right**) with greater than 5N of force. When the block is fragile, the episode terminates with a negative reward if the impact is greater than 3N. Our approach of training policies with a combination of task reward, impact penalty, and penalty-based surprise intrinsic reward is the only one that learns effectively for both tasks. Policies are trained on: task reward only ■; task reward with impact penalty and no intrinsic rewards ■, dynamics-based surprise ■, or penalty-based surprise. The parameterization λ^0 for acceptability of penalties varies: $\lambda_2^0 = 1.5$ ■, 2 ■, or 3 ■ (left) and $\lambda_2^0 = 1$ ■, 1.5 ■, or 2 ■ (right). $\lambda_1^0 = 2$ for all. Policies trained on only task reward are unable to learn the fragile-block task at all (right).

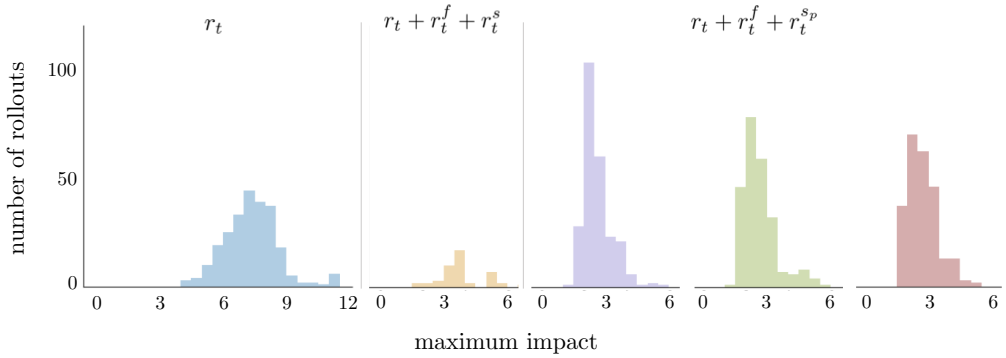

Figure 4: We train policies to press a block with greater than 5N of force. These histograms show the maximum impact (in Newtons) experienced per rollout, when the agent performs the task successfully. Rollouts are collected after 500k training steps. $\lambda_2^0 = 1.5$ ■, 2 ■, or 3 ■. $\lambda_1^0 = 2$ for all. (Note: with only an impact penalty, agents never succeed in performing the task, so that histogram is not shown.)

the task—they get trapped in a local optimum of avoiding contact with the environment, since they experience penalties from contact before discovering how to perform the task (Fig. 2, top right).

In line with the results from our previous experiment, in which we observed that dynamics-based surprise leads to only limited gentle touching in the presence of impact penalties, we saw that penalty-based surprise was much more effective in terms of agents learning how to perform the task gently, with low impacts (Fig. 4). Even more, these agents learned as quickly as ones trained without the impact penalty (Fig. 3, left). This may be because this task is particularly contact-focused (in general manipulation tasks are contact-focused, but to varying degrees), so it is a setting in which contact-focused exploration is especially helpful.

5.4 Manipulation of fragile objects

Finally, we made the task more difficult by introducing a fragile block. This is analogous to if the robot needs to deal with fragile objects, such as picking or sorting ripe fruit. This fragile block breaks if the impact force at any point is greater than 3N, and the episode terminates with a negative reward of -0.5. The reward for completing the task is +5. Now, policies trained with only the task reward are unable to learn the task at all, because they accidentally break the block a few times, and learn that any contact with the block is undesirable. There is no reward shaping that incentivizes these policies to try interacting with the block in a gentle way.

In contrast, policies trained with the impact penalty are better able to learn the task. As before, penalty-based surprise intrinsic rewards are more effective than dynamics-based ones in terms of how quickly policies are able to learn the task (Fig. 3, right).

6 Discussion and Future Work

Our work takes a step toward using deep RL to train policies for gentle, contact-rich manipulation. We found that choosing the appropriate focus of curiosity is important for incentivizing agents to interact gently with the environment, in the presence of impact penalties. This enables efficient and safe exploration, precise task execution, and successful manipulation of fragile objects.

A main direction of future work is to apply this approach to more complex tasks, in particular tasks in real-world and/or dynamic environments. In addition, this work only considers one aspect of being gentle—impact. Our approach could be used to train policies while minimizing other sources of wear and tear, for instance total force (rather than the increase in force), or the torques exerted by a robot's motors (which would reduce energy consumption as well [21]).

References

[1] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-first International Conference on Machine Learning (ICML)*, 2004.

[2] J. Achiam, D. Held, A. Tamar, and P. Abbeel. Constrained policy optimization. In *Proceedings of the Thirty-Fourth International Conference on Machine Learning (ICML)*, 2017.

[3] J. Achiam and S. Sastry. Surprise-based intrinsic motivation for deep reinforcement learning. *arXiv preprint arXiv:1703.01732*, 2017.

[4] E. Altman. *Constrained Markov Decision Processes*. CRC Press, 1999.

[5] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[6] G. Barth-Maron, M. W. Hoffman, D. Budden, W. Dabney, D. Horgan, D. TB, A. Muldal, N. Heess, and T. P. Lillicrap. Distributed distributional deterministic policy gradients. In *Proceedings of the Sixth International Conference on Learning Representations (ICLR)*, 2018.

[7] M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos. Unifying count-based exploration and intrinsic motivation. In *Proceedings of the Twenty-Ninth Advances in Neural Information Processing Systems (NIPS)*, 2016.

[8] G. Dalal, K. Dvijotham, M. Vecerik, T. Hester, C. Paduraru, and Y. Tassa. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*, 2018.

[9] J. A. Fishel and G. E. Loeb. Sensing tactile microvibrations with the biotac—comparison with human sensitivity. In *Proceedings of the Fourth IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, 2012.

[10] J. Fu, J. D. Co-Reyes, and S. Levine. EX2: exploration with exemplar models for deep reinforcement learning. In *Proceedings of the Thirtieth Advances in Neural Information Processing Systems (NIPS)*, 2017.

[11] J. García and F. Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16:1437–1480, 2015.

[12] R. Houthooft, X. Chen, X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel. Vime: Variational information maximizing exploration. In *Proceedings of the Twenty-Ninth Advances In Neural Information Processing Systems (NIPS)*, 2016.

[13] J. Hu and T. Wang. Pre-impact configuration designing of a robot manipulator for impact minimization. *Journal of Mechanisms and Robotics*, 9(3), 2017.

[14] P. Huang, W. Xu, B. Liang, and Y. Xu. Configuration control of space robots for impact minimization. In *Proceedings of the 2006 IEEE International Conference on Robotics and Biomimetics*, 2006.

[15] K. Ikuta, H. Ishii, and M. Nokata. Safety evaluation method of design and control for human-care robots. *The International Journal of Robotics Research*, 22(5):281–297, 2003.

[16] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Proceedings of the Thirtieth Advances in Neural Information Processing Systems (NIPS)*, 2017.

[17] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016.

[18] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. In *Proceedings of the Fourth International Conference on Learning Representations (ICLR)*, 2016.

[19] C. Liu, X. Xu, and D. Hu. Multiobjective reinforcement learning: A comprehensive overview. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(3), 2015.

[20] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing Atari with deep reinforcement learning. In *Neural Information Processing Systems (NIPS) Workshop on Deep Learning*, 2013.

[21] A. Mohammed, B. Schmidt, L. Wang, and L. Gao. Minimizing energy consumption for robot arm movement. *Procedia CIRP*, 25:400 – 405, 2014.

[22] A. Y. Ng, D. Harada, and S. J. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning (ICML)*, 1999.

[23] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the Thirty-Fourth International Conference on Machine Learning (ICML)*, 2017.

[24] M. Pecka and T. Svoboda. Safe exploration techniques for reinforcement learning – an overview. In *Modelling and Simulation for Autonomous Systems*, 2014.

[25] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48(1):67–113, 2013.

[26] S. Russell and A. L. Zimdars. Q-decomposition for reinforcement learning agents. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, 2003.

[27] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel. Trust region policy optimization. In *Proceedings of the Thirty-Second International Conference on Machine Learning (ICML)*, 2015.

[28] ShadowRobot. ShadowRobot Dexterous Hand. `https://www.shadowrobot.com/products/dexterous-hand/`.

[29] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–503, 2016.

[30] B. C. Stadie, S. Levine, and P. Abbeel. Incentivizing exploration in reinforcement learning with deep predictive models. *arXiv preprint arXiv:1507.00814*, 2015.

[31] H. Tang, R. Houthooft, D. Foote, A. Stooke, X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel. #Exploration: A study of count-based exploration for deep reinforcement learning. In *Proceedings of the Thirtieth Advances in Neural Information Processing Systems (NIPS)*, 2017.

[32] C. Tessler, D. J. Mankowitz, and S. Mannor. Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074*, 2018.

[33] E. Todorov, T. Erez, and Y. Tassa. MuJoCo: A physics engine for model-based control. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[34] L.-B. Wee and M. W. Walker. On the dynamics of contact between space robots and configuration control for impact minimization. *IEEE Transactions on Robotics and Automation*, 9(5):581–591, 1993.

Supplementary Material

A Implementation Details

The output of the actor is passed through a tanh, so that it is between -1 and +1. This output specifies delta position: it is added to the current position and then clipped based on the minimum and maximum joint angle per action dimension, to obtain the action. The actor network consists of two fully-connected layers of 300 and 200 hidden units each. Each of the critic networks consists of two fully-connected layers of 400 and 300 hidden units each. The distributional output of the critic has support $(\ \ 100, 100)$ and 101 bins. For both the actor and critic networks, the first hidden layer is followed by layer normalization [5] and a tanh, and all other hidden layers are followed by exponential linear unit (ELU) activations. For D4PG, we used a batch size of 256 and a replay buffer of 1 million transitions.

The dynamics model consists of three ensembles, one each for predicting the three types of state features: joint position, joint velocity, and touch. The non-gentleness predictor model consists of a single ensemble. Each of these ensembles consists of five neural networks, with three fully-connected layers of 128 hidden units each. All hidden layers are followed by rectified linear unit (ReLU) activations.