# LEARNING FOR REAL ROBOTS FROM LARGE-SCALE DATASETS

Andrew Kondrich

May 2020

I approve and sign off on this undergraduate thesis.

(Fei-Fei Li) Primary Adviser

I approve and sign off on this undergraduate thesis.

(Silvio Savarese)

# LEARNING FOR REAL ROBOTS FROM LARGE-SCALE DATASETS

**Andrew Kondrich** *
Department of Computer Science
Stanford University
andrewk1@cs.stanford.edu

## ABSTRACT

Training conventional reinforcement learning algorithms notoriously requires large amounts of exploration data (Fan et al., 2018). In robotics, where tasks often have long horizons and require complex interactions with the environment, these data requirements can make it impossible to learn performant policies. Supervised learning methods have recently emerged as promising solutions to these drawbacks: by learning a policy from successful task demonstrations, agents can avoid having to find these optimal trajectories through random exploration (Fu et al., 2020). However, it is challenging to collect these datasets at scale, and research into supervised methods for robotics has largely been restricted to toy scenarios and small datasets (Fujimoto et al., 2018; Kumar et al., 2019). Crowdsourcing is a potential solution to scale this data collection to complex environments and larger datasets. Using the RoboTurk platform (Mandlekar et al., 2018; 2019a), a system for remote teleoperation of robots, we now have the ability to collect massive datasets of human demonstrations on dexterous robot manipulation tasks. However, using this data to then train autonomous agents in the real world has seen limited success. In this work, we propose a novel method for training agents directly in the real world from this data. The common paradigm for instantiating real world agents is to transfer a simulation-trained policy to the real world. However, state-of-the-art methods for sim2real transfer rely on augmenting on-policy behavior in either the simulator or the real world, which is data and resource intensive. We evaluate some zero-shot sim2real methods and demonstrate that they fail to robustly transfer simulation policies to the real world. In contrast, we see that learning directly in the real domain using Batch Reinforcement Learning (Batch RL) can be a viable way to train real world agents. While existing work shows learning on RoboTurk data is challenging, we present a novel data relabeling method that greatly improves results on existing algorithms. By inferring the intended human actions from the data and using this new representation to train our agents, we are able to reach an unprecedented 75% task success when training from the crowdsourced data using state-of-the-art Batch RL methods. Going forward, this data reformulation technique can be a promising tool for learning agents from crowdsourced data in the real world.

## 1 INTRODUCTION

### 1.1 CROWDSOURCING FOR ARTIFICIAL INTELLIGENCE

While neural networks have been well-known in the field of machine learning for decades, they have been relatively unsuccessful for most of their history (Rumelhart et al., 1988; Wang & Raj, 2017). Without using some pretraining scheme or carefully tuned parameter initializations, it used to be difficult to train neural networks end-to-end using backpropagation from scratch. It was only following the release of the ImageNet challenge in 2010 that these methods began to show some forms of success (Krizhevsky et al., 2012; Deng et al., 2009). When it was realized that these models

*I'd like to acknowledge my close collaborators Jonathan Booher and Ajay Mandlekar, and mentors Roberto Martin-Martin and Yuke Zhu.

could learn to accurately and robustly solve tasks in computer vision using strong supervision and massive datasets like ImageNet, deep learning truly took off and has since revolutionized the field of artificial intelligence and technology as a whole to this day.

The innovation that initially paved the way for the deep learning revolution was neither ImageNet nor novel deep learning architectures, but the web services by which we could collect labeled datasets at such large scales. The proliferation of crowdsourcing platforms such as Amazon Mechanical Turk has made it possible to harvest human intelligence and generate large and rich datasets necessary for training deep learning models. Following ImageNet, large-scale crowdsourced datasets have altered the way researchers approach disciplines like cognitive science and linguistics and tasks like natural language understanding, video processing, robot perception, and more (Xu et al., 2016; Rajpurkar et al., 2016; Geiger et al., 2013; Manning, 2015). Large human-labeled datasets are so useful in artificial intelligence today that there are even companies offering advanced data labeling services via crowdsourcing specifically for such use cases (Ziegler et al., 2019).

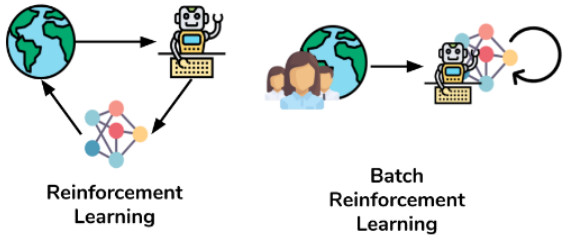


Figure 1: Agents learn through interaction in RL, and from an existing dataset in Batch RL

### 1.2 Learning Autonomous Agents

While large-scale crowdsourced datasets sparked the deep learning revolution, deep learning has since found success in the absence of human-labeled data. Deep learning has shown incredible results in training autonomous agents via reinforcement learning (RL) (OpenAI et al., 2019), where agents collect their own data by directly interacting with an environment, and are judged by a human-specified reward function. We can also learn generalizable skills for robotics without a reward function by letting agents explore an environment via self-play (Lynch et al., 2019) and figure out what kinds of actions correspond to meaningful behaviors, like opening a door or pressing a button. By relying on the robot to first randomly explore an environment in search of useful actions, both of these methods rely on large amounts of training interactions to learn (Fan et al., 2018). As such, while these methods have shown incredible promise in learning performant policies, their reliance on online learning schemes and random exploration constrains their utility for real world applications such as dexterous manipulation tasks on real robots. Furthermore, when considering complex long-horizon, human-robot interactive or multi-agent tasks, it is not clear that searching for optimal behaviors via random exploration could succeed given computational and task constraints. Lastly, in many cases it is hard to properly specify a reward function for the task you wish to solve.

Methods for learning robot agents should be sample efficient and leverage off-policy data so that they can learn from others and avoid collecting large amounts of online exploration data. Batch Reinforcement Learning (Batch RL) algorithms are promising because they can learn policies directly from existing demonstrations without any on-policy exploration or rollouts (Lange et al., 2012; Fujimoto et al., 2018), as illustrated in Figure 1. This is useful in the real robot setting, since training rollouts can be prohibitively slow on real hardware. Many tasks we are interested in learning also require some cognitive reasoning or creativity to solve. For example, in the RoboTurk tower creation task (Mandlekar et al., 2019a), the agent is required to devise a method of stacking bowls to make the highest possible tower. The optimal solution involves creative uses of the bowls, such as placing the smaller bowls closer to the ground or flipping them. The only agents readily capable of solving these kinds of tasks are humans, but we can potentially imbue our policies with such reasoning abilities by learning from human demonstrated data. By learning how humans naturally interact with their environments, we can make smarter and more human-friendly robots. Given these

desired qualities, we propose learning from human demonstrations using Batch RL as an exciting and relatively underexplored line of work for real robot skill learning.

### 1.3 THE ROBOTURK SYSTEM FOR CROWDSOURCED TELEOPERATION

Devising a human data collection system for robot learning that is both intuitive for demonstrators and scalable to crowdsourced demonstrators is a challenging task. There have been many attempts at devising such methods. Historically, collecting robot trajectories has been achieved either via kinesthetic guidance or teleoperation (Argall et al., 2009; Goldberg et al., 1994). Although kinesthetic guidance is an intuitive method, it is often limited to collecting limited sets of demonstrations that are too small for learning generalizable policies on complicated tasks. Kinesthetic guidance is also not usable in the simulation setting. Most teleoperation schemes rely on game interfaces such as joysticks or VR headsets for data collection (Laskey et al., 2017; Dragan & Srinivasa, 2012). Such controllers tend to limit the range of motion of the arm, creating a bias in the resulting trajectories that may not be indicative of the demonstrator's intuitions in solving the task. Furthermore, it is unfeasible to distribute the necessary hardware in the crowdsourced setting. As a member of the RoboTurk project in the Stanford Vision and Learning Lab, I have been working on a system that enables us to quickly collect large sets of useful robotics data from non-expert demonstrators. The system uses an iPhone as the controller, where the pose of the phone corresponds to the pose of the end effector on a Rethink Robotics Sawyer robot arm. As such, the system is as intuitive as kinesthetic guidance while being scalable to the crowdsourced setting due to the commonplace ownership of smartphones. The collected trajectories preserve every piece of information present to the teleoperator while they were performing the task. As a result, the data is information rich and can be processed in many ways for downstream tasks. The RoboTurk platform enables us to economically collect rich datasets of human demonstrations for robotic tasks via crowdsourcing. In this work, I rely on the RoboTurk platform as a solution for collecting data in crowdsourced settings. As described in prior RoboTurk publications (Mandlekar et al., 2019a; 2018), we have been able to collect datasets consisting of roughly 1000 human demonstrations per task on either simulated or real robots using this platform. This dataset is unique in its scale and crowdsourced origin, and demonstrating robot learning success on it would assert the viability of crowdsourcing data for robotics tasks.

Using these pilot RoboTurk datasets to train robust agents that can be deployed in the real world remains an unsolved problem. Current techniques for Batch RL have seen limited success when learning on this diverse and noisy data. Furthermore, deploying deep learning models on real robots requires dealing with the safety and time constraints of using real hardware, a challenge for any robot learning model today. In this work, I explore two paradigms for learning real robot policies from crowdsourced datasets. For either of these methods, the end goal is to instantiate a robot policy in the real world that solves the task. There are many decisions involved in designing such a system, and they impact the type of data being used as well as the models considered to achieve the task. To our knowledge, this is the first analysis into the different approaches to learning agents capable of interacting in a real environment from crowdsourced data.

### 1.4 EVALUATING THE SIM2REAL AND BATCH RL PARADIGMS

I present a study on the following methods: 1. **Zero-Shot Policy Agnostic Sim2Real** and 2. **Offline learning**. In the first method, we use data collected in a simulation environment to train a model that then must be transferred to a real robot. While collecting simulation datasets scales easily and allows us to take advantage of privileged information and dense rewards in learning a policy, it is an immense challenge to bridge the perceptual and physical dynamics gap to then achieve successful policy performance in the real world. There are also unique constraints to achieving sim2real transfer when training offline and from the RoboTurk crowdsourced datasets we are interested in, particularly when employing the common sim2real paradigm of domain randomization (OpenAI et al., 2019; Tobin et al., 2017), a technique for randomizing the visual and physical elements of the simulated environment with the goal of achieving robustness in real world settings. In contrast, we can attempt to learn a policy in the goal domain, such as the real world, using Batch RL on crowdsourced datasets. However, it is not clear if a model could learn robust policies given sparsely labeled, diverse and noisy real robot data. An evaluation of these methods reveals insights into how learning systems from crowdsourced data should be designed, since the better option is not initially apparent.

After evaluating both methods, we find it more promising to learn using Batch RL in a goal domain from RoboTurk data than effectively bridging the sim2real gap. We first propose a novel method for visual sim2real based on CycleGAN (Zhu et al., 2017) and compare its performance against Random-To-Canonical Networks (RCAN) (James et al., 2019), a standard method from the literature. Using these methods, we attempt to transfer a policy trained to lift a cube from simulation to real. In either case, we see poor transfer performance. Neural networks are particularly sensitive to mismatch between training and test data distributions (Recht et al., 2019). Despite our attempts to close this gap using visual transfer, it is difficult to remove all artifacts from generated images. While RCAN originally uses real world on-policy tuning to address these artifacts, this process is incredibly data intensive and we have no means to supervise such lengthy training procedures on real robots. The minor visual artifacts in sim2real transfer, paired with dynamics mismatch between sim and real, made sim2real transfer of even simple tasks hard to achieve.

We can avoid the reality gap problem by learning off-policy and using offline data in the goal domain. However, prior work has shown that learning off-policy when only given the RoboTurk datasets is extremely challenging, so it is not clear whether such an approach is viable today. Our main insight is that the existing RoboTurk data is very noisy due to its high frequency joint space actions, and discriminating noise from the intended movements of the teleoperator is difficult for Batch RL. We see that a reformulation of the data to better capture the intended actions of the human, rather than the executed actions of the robot, can significantly increase model performance across Batch RL algorithms while providing for safer and more accurate control. The success observed in learning off-policy from this reformulated data suggests that the fundamental challenge of Batch RL on crowdsourced data, namely the presence of suboptimal and multimodal trajectories, is readily addressed by existing algorithms provided that the data is not noisy. Thus, privileged simulator information may not be needed to get performant policies, and learning directly on real robots can be a viable approach going forward.

## 2 Zero-Shot Policy Agnostic Sim2Real

The task of sim2real transfer involves adapting a policy trained in a simulation environment to perform in the real world. For the problems we are interested in, such as learning dexterous manipulation in complex scenes, this kind of transfer is challenging due to the difference in visuals and dynamics, called the reality gap. Our goal is to understand what transfer methods can work best for RoboTurk environments and data.

Many sim2real methods use on-policy augmentations to the learning agent while training in simulation or when tuning on a real robot that are unfeasible to use in the RoboTurk setting (Ramos et al., 2019; James et al., 2019; Tobin et al., 2017; OpenAI et al., 2019; Andrychowicz et al., 2019; Jeong et al., 2019). A common sim2real method is domain randomization (Tobin et al., 2017). While training a deep RL policy, we can randomize the visual features and physical attributes of objects in the scene, with the hope that the policy then generalizes to a distribution of domains that includes the real world. However, on-policy RL algorithms are already very data hungry without such augmentations, and can take days to learn optimal behaviors on RoboTurk tasks, even when using a distributed learning framework such as Surreal (Fan et al., 2018) which pools training experiences from multiple agents in parallel to collect lots of data quickly. Making the learning task more difficult with domain randomizations would be incredibly time and compute intensive. There are sim2real methods that are policy-agnostic such that they do not alter the training behavior of the RL algorithm. Random-To-Canonical Networks use domain randomization to train a generative model that recreates visual scenes of the same task in a "canonical" style. Since RCAN is trained separately from the agent, we avoid the data inefficiencies of training an agent using domain randomization. However, the original paper shows that on-policy tuning of the agent in the real world greatly improves results. This tuning required thousands of real world training iterations, which is still a costly and time-consuming procedure.

Besides transferring across the visual gap, a policy has to also adapt to a dynamics mismatch between the simulator and real world. Transferring a policy from simulation to real dynamics is just as costly as the visual transfer and also requires either randomization in the simulator Peng et al. (2018) or some amount of real world updates (Ramos et al., 2019). These methods for dynamics transfer are all dependent on on-policy data collection, but learning on RoboTurk tasks requires significant

exploration and benefits greatly when existing data can be leveraged using Batch RL. However, there is no way to retroactively apply dynamics randomization to the existing RoboTurk data, while collecting new data with randomizations would require significantly more demonstrations than we are capable of crowdsourcing even for simple tasks. For example, one trajectory alone could be collected under a combinatorial number of dynamics parameter combinations, so the number of trajectories needed to collect even a small number of demos is exponentially greater. As a result, we want a sim2real method that is both policy-agnostic and capable of zero-shot transfer to a real world agent. Since it is not clear how to devise such a method for dynamics transfer, we exclusively evaluate visual sim2real transfer in this work. annot readily apply any form of dynamics transfer.

We attempt a zero-shot policy-agnostic visual transfer of an existing policy. We try RCAN without the on-policy tuning well as a novel method based on CycleGAN, a model that can learn bijections between two domains by ensuring cyclic-consistency. Using RoboTurk, we can collect large demonstrations of the same task in both the real world and the simulator. We can then learn the bijection between these datasets using CycleGAN, and subsequently perform a sim2real transfer. Using CycleGAN, we can avoid relying on domain randomization to generalize to the real world since we have direct access to this data, and can perform visual sim2real transfer much more efficiently. However, for both RCAN and CycleGAN, we observe fundamental problems with the sim2real transfer that are not clearly addressable using existing techniques.

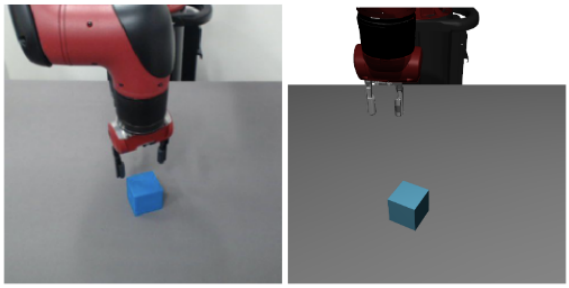

Figure 2: The environment and camera pose are aligned between the real world and simulator

### 2.1 Evaluation

In each case, we train a policy in Robosuite to generate joint velocities given an image of the scene. Note that while we choose a model-free RL algorithm here, the results would be largely applicable to any choice of reinforcement learning algorithm since our sim2real methods are policy-agnostic. The policy is trained using Surreal distributed Proximal Policy Optimization (dPPO) with a shaped reward function on the `SawyerLift` environment in Robosuite (Fan et al., 2018). We train with 16 agents running in parallel for 12 hours to get a robust policy. For each action, the model observes the three most recent frames such that the velocity and direction of the arm can be inferred. We see that this temporal encoding is crucial to the success of the policy. At test time, we transfer the real world images to the style of the simulator, query the policy for an action, and execute it directly on the real robot.

### 2.2 Random-To-Canonical Networks

RCAN relies on visual domain randomization in simulation to train a conditional GAN model that transforms images of a robot manipulation scene painted in arbitrary colors and textures to a single "canonical" visual style. Once trained, this model is then used to transform a real world scene into the accompanying canonical version, which can be used by a reinforcement learning model trained in simulation to perform the task from the same observation space. We identify the capabilities of RCAN models on dexterous manipulation tasks performed on the Sawyer arm platform. We train RCAN on a total of 289250 synthetic training pairs generated with domain randomization for 11 epochs.

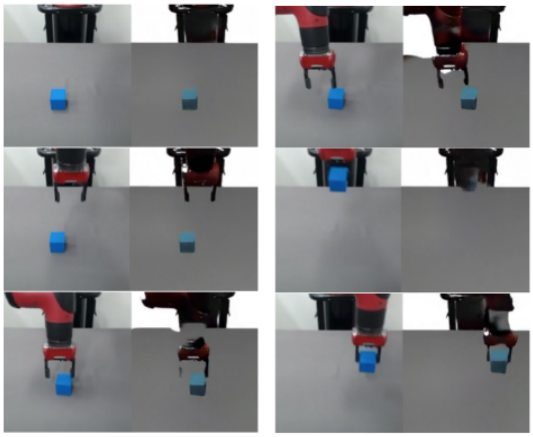

Figure 3: **RCAN transfer** We include some qualitative examples of the RCAN visual transfer of a real robot trajectory to the simulation domain. From left to right, each pair of images shows the real world scene and the RCAN generated equivalent. While cube and arm semantics are generally well preserved, there are notable visual artifacts, such as the blurring of the background or the dissapearing wrist.

We observe that a dPPO policy trained in simulation is capable of reaching 100% success on 100 evaluation rollouts, while exhibiting complete failure in performing the same task in real robot rollouts and only capable of reaching the cube in 2 out of 10 validation rollouts. Qualitatively, RCAN can generate good visual transfers of real world scenes, as seen in Figure 3. However, the high dimensionality of the generated state space means that even with careful tuning and extensive training, the generated image translations contain a number of visual artifacts. It may be that these artifacts can only be remedied with further training on the real robot.

Another major contributor to the poor transfer is the dynamics mismatch. While actions taken by the simulation-trained policy were approximately correct in that they moved closer to the arm, the mismatch was significant enough that the model is incapable of performing precise manipulation tasks needed to accomplish the task. We also note that there existed a mismatch in control frequencies. Our dPPO agent was trained using stacked frames across time. Including the forward pass through the RCAN model slows the control frequency that the model is capable of performing at, which changes the time passed between sequential frames. This introduces another dimension of mismatch between the simulation data and the real world. The visual artifacts combined with the dynamics mismatch in the robot arms and control frequencies explains the limited success we saw using RCAN to do sim2real transfer.

### 2.3 Using CycleGAN with privileged information

Since we can leverage RoboTurk to collect large sets of unpaired simulation and real demonstrations, we also try a sim2real method that uses existing datasets in simulation and the real world. CycleGAN is a model that learns correspondences between unpaired images in different domains using a cycle-consistency loss and adversarial training. Given the same environment in the real world and the simulator, there exists a bijection between the visual domains that can be learned using CycleGAN. Using the RoboTurk platform, we collect a set of 80 real and 200 simulation task demonstrations on the cube lift task in visually similar scenes in the two domains. This model benefits from training directly on the real data distribution, and is able to generate plausible recreations of the real world in the simulator visual domain with minimal stylistic artifacts. However, we found that this model is sensitive to the training data distribution, and can learn semantically wrong mappings for states that never appeared while training. To ensure some level of semantic consistency, we include an

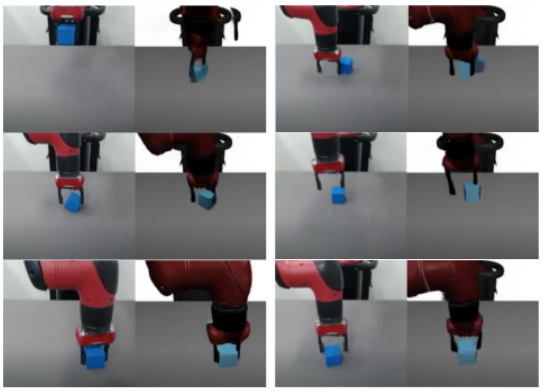

Figure 4: **CycleGAN transfer** We show some qualitative results of the CycleGAN visual transfer on a real trajectory to the simulation domain. Each pair displays the real image on the left and the translated CycleGAN version on the right. Note that the quality of the recreation is generally good, and for the most part preserves semantic alignment. However, generated images tend to contain some visual artifacts or minor semantic mismatch, as notably visible in the top left transfer shown, where the gripper and cube location are mispredicted.

additional model that is trained to predict the proprioception of the real image from the recreated counterpart. While this is able to enforce that the arm remains in the correct position in the generated simulation image, we have no such supervision for the cube. This mismatch occurs in states that are not seen within the simulated dataset, where the enforced bijection does not have to be semantically consistent, as seen in Figure 4. Nevertheless, the quality of CycleGAN transferred images is notably higher than RCAN, and most semantic are retained. Yet we observe similarly poor performance using CycleGAN for visual transfer as with RCAN, resulting in 0% reach or lift success using our trained dPPO policy. Notably, there is still some blurriness and artifacts present in the generated CycleGAN images. Since these are never observed in training, it is likely that policy performance will be impacted. The largest contributor to the poor performance is likely the dynamics mismatch, which is challenging to address in our offline policy-agnostic setting. Notably, previous work has used CycleGAN for some form of semantic transfer for robotics tasks (Smith et al., 2019). However, it addresses the semantic mismatch we observe by severely restricting the observed data distribution, and the dynamics gap by using a simple 2-dimensional controller and task. The issues we observe here remain unsolved when using CycleGAN for sim2real transfer.

### 2.4 The challenging nature of policy agnostic sim2real

Policy-agnostic sim2real methods that work zero-shot could allow us to transfer RL models efficiently to the real world, and therefore be viable in the RoboTurk setting, where the tasks are complex and resources to support real world training are limited. In learning visual sim2real adaptation, we identified that even minor semantic mispredictions and dynamics mismatches at test time can cause major failure modes for policy rollouts, due to the fundamental sensitivity of deep learning models to changes in the observed data distribution.

## 3 Offline learning

Besides learning policies in simulation and attempting a sim2real transfer, we can also learn directly from real robot demonstrations. While existing Batch RL algorithms (Kumar et al., 2019; Fujimoto et al., 2018) allow for offline learning, they have not been extensively evaluated on challenging tasks and datasets like RoboTurk. First, many Batch RL algorithms rely on easy evaluation domains where even random data can cover the space of good behaviors. In contrast, tasks like RoboTurk Bin picking require long-horizon reasoning and lend themselves to multiple optimal policies. Second,

it is not clear that they can learn from crowdsourced datasets, where demonstrations are collected from multiple humans who are diverse and inconsistent in their behaviors. Prior datasets have been collected from either a single demonstrator or from partially trained reinforcement learning agents (Fu et al., 2020). Lastly, real RoboTurk data contains only a sparse reward signal and raw perception state. In contrast, existing Batch RL methods have only been trained using simpler ground truth object states and dense rewards (Kumar et al., 2019; Fujimoto et al., 2018). In summary, learning from RoboTurk data requires learning in difficult evaluation domains from complicated data sources and a sparse reward signal. As such, this task is much more complex than what has been explored in previous works.


Figure 5: RoboTurk tasks require manipulating the environment and multi-stage reasoning, and crowdsourcers can demonstrate diverse and suboptimal ways to solving them. Learning from these demonstrations is challenging with standard Batch RL Methods.

In this section, we evaluate the feasibility of training Batch RL algorithms from the RoboTurk dataset. Batch RL algorithms struggle to learn from RoboTurk data when comparing against their performance on clean data on the same task. We identify a number of key characteristics of the data that result in this poor behavior. First, there exists a complex relationship between the used action space and the task objective. Our dataset represents the actions in the executed robot commands, high frequency joint velocities, which can be noisy and not clearly represent the intended trajectories of the user. Since the original teleoperator controlled the robot by moving their phone to move the position of the robot gripper in cartesian space, we find that deltas in end-effector pose are a more natural representation of the user's actions. Furthermore, because joint velocities are a higher dimensional representation of the actions than end-effector poses in cartesian space, there are many possible joint velocities commands that result in the same change in position. Reducing the action space to cartesian space eliminates this ambiguity, further making the action space easier to learn in. Second, we identify that the noisy high-frequency control and long task horizons of the original dataset make learning difficult. As seen in Figure 6, actions in joint velocities at high frequencies contain a lot of jitter between subsequent actions. Since these trajectories contain over a thousand time-steps, a low level controller must learn actions over a wide distribution of states from noisy and potentially conflicting supervision.

We propose a novel method for relabeling these demonstrations into low-noise short horizon trajectories in cartesian space. Using a controller playback mechanism built on Operational Space Control (Nakanishi et al., 2008), an analytical framework for controlling directly in cartesian space, we can generate the trajectories using end-effector poses at a reduced control frequency, effectively removing the artifacts from the original robot commands and creating a better representation of the intended trajectory of the user. Using this new control formulation significantly improves performance across all algorithms on the crowdsourced data. It is promising that learning from the RoboTurk data using existing techniques is possible provided the right representation of the data, and suggests that learning on real systems with Batch RL can be feasible today.

### 3.1 Evaluating Batch RL on different datasets

In order to showcase the challenge of learning on the RoboTurk data, we compare results on three datasets, each consisting of 200 demonstrations on the RoboTurk Cans task in simulation. In this task, the robot must pick up a soda can from a container and place it into its corresponding bin. In the "unimodal" dataset, I personally collect trajectories that follow the shortest path from where the arm is initialized to where it performs a precise top down grasp to pick the can, ensuring the trajectories are very similar in solving the task. The "bimodal" dataset consists of 100 trajectories

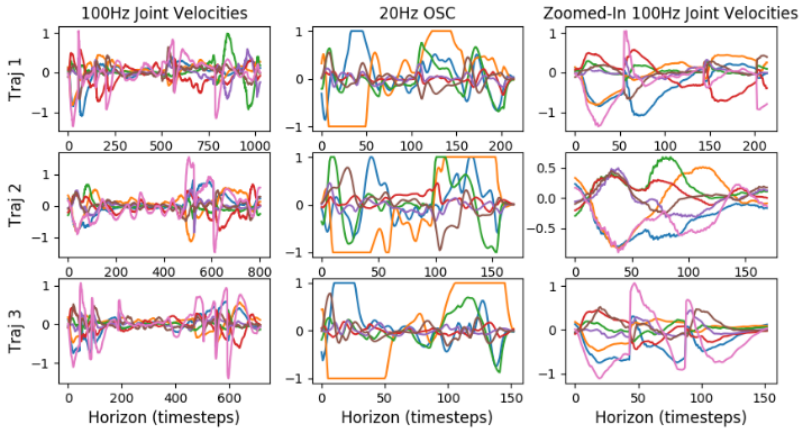


Figure 6: We visualize the actions for three trajectories (rows) in 100Hz Joint Velocity, 20Hz Operational Space Control, as well as a zoomed in section of the 100Hz Joint Velocity at the same time horizon length as the 20Hz OSC. We plot each dimension of the action space across time in a separate color. We see that 20Hz OSC trajectories are much shorter than the original data and have less high frequency jitter when compared to the Zoomed-In 100Hz Joint Velocities. This makes the 20Hz OSC trajectories much easier to learn from.

from the unimodal dataset and 100 using a different path to reach the can position while performing a similar top down grasp. The final dataset samples the fastest 200 demonstrations from the existing RoboTurk Pilot dataset on the Cans pick and place task. We compare the performance of 4 models on these 3 datasets, including two behavioral cloning baselines, a popular Batch RL algorithm called Batch Constrained Q-Learning (BCQ) (Fujimoto et al., 2018), and IRIS (Mandlekar et al., 2019b), a recent Batch RL method designed for learning in the crowdsourced setting. The models observe full proprioception and the pose of the can as state input, and generate joint velocities as actions. BCQ and IRIS also observe a sparse reward signal, where the final state in all the trajectories is labeled with task success. We evaluate model performance on 50 randomly perturbed can initializations.

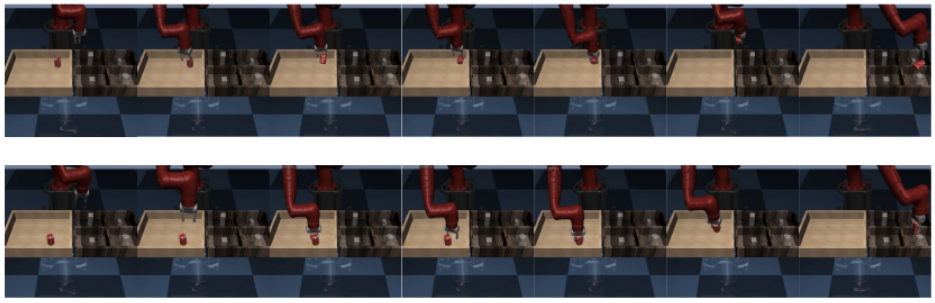

Figure 7: We present two crowdsourced trajectories on the RoboTurk Can pick task. In each case, we see different behaviors: the first teleoperator fumbled when picking up the can, while the second required multiple grasps to succeed. This highlights the diversity and inconsistencies of the dataset.

Results on the three datasets, as shown in Table 1, suggest that learning from teleoperated data is challenging. While recurrent behavioral cloning is able to perform well on my collected datasets, confirming the limited number of modes present, they fail to learn in the crowdsourced setting, due to the assumption that the observed data is optimal (Osa et al., 2018). BCQ, a method for learning

Table 1: Batch RL Algorithm performance across datasets with varying diversity

| Dataset | BC | RNN-BC | BCQ | IRIS |
|---|---|---|---|---|
| Unimodal Cans | 19% | **40%** | 16% | 23% |
| Bimodal Cans | 0% | 28% | 16% | **48%** |
| RoboTurk Cans | 0% | 0.33% | 0% | **28.3%** |

from suboptimal data in the batch setting, is notably underperforming, and is even outperformed by a baseline behavioral cloning model. IRIS, which learns both a goal proposal network and a Q function, has some capacity to identify which trajectories are better than others, and even benefits from the wider exploration of the state space present in the Bimodal and RoboTurk datasets. Nevertheless, we see poor task performance across the board, where no model is able to surpass 50% success rates. We claim that the issue causing this poor performance is the data's current representation as high frequency joint controls executed by the robot. By imitating the teleoperator's intended actions using low frequency cartesian commands instead, we see significant gains in policy performance when learning from this data.

### 3.2 Learning on the relabeled RoboTurk data

Relabeling our existing RoboTurk data to represent intended human actions rather than executed robot actions filters out noise in the trajectories while retaining trajectory information needed to solve the task. We choose 20Hz Operational Space Control as our representation of intended human actions. Operational Space Control (OSC) is an analytical framework for control in end-effector space. Using the robot mass matrix and the jacobian of the end-effector, we can directly solve for the needed motor commands to move the end effector to a desired pose in cartesian space. To relabel our data, we extract the end effector poses from the RoboTurk Cans data at 20Hz, and verify that the trajectory can be followed using OSC in simulation. Intuitively, these new trajectories are a higher level description of the desired teleoperation trajectories. The relabeled actions in this representation are much smoother and the overall trajectories are shorter in horizon. We present a qualitative visualization of some trajectories in both representations to confirm these intuitions.

We perform the same evaluation of the 4 models trained on the relabeled 20Hz OSC RoboTurk Cans data and show results in Table 2. Across the board, we see significant improvement in model performance, demonstrating that it is easier to learn under this better representation of the intended actions of the user. Notably, we see significant performance gains on IRIS, a model designed for learning on RoboTurk data. Once the noise of the joint space actions is filtered using our relabeling process, IRIS is capable of learning robust and performant policies from this existing set of user demonstrations completely off-policy using sparse rewards. An example trajectory is presented in Figure 7. This success suggests that learning directly from crowdsourced data can be a viable approach for learning policies for real world tasks.

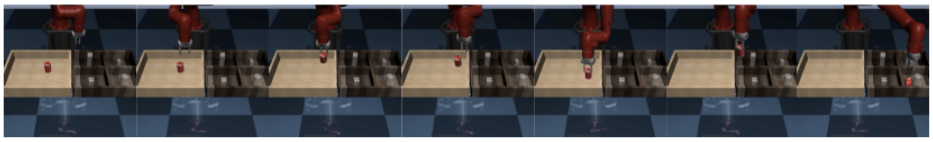

Figure 8: We present a sample action trajectory from a learned IRIS agent using 20Hz OSC, where the agent demonstrates a recovery from a failed grasp. Despite the lower control frequency, the agent can still perform accurate dexterous manipulation. Furthermore, since the reformulated data is a clearer representation of the task semantics, the agent can better learn robust behaviors that solve the task, such as this recovery.

Table 2: Algorithm performance on 20Hz OSC RoboTurk Data

| Dataset | BC | RNN-BC | BCQ | IRIS |
|---|---|---|---|---|
| RoboTurk Cans | 37.5% | 18.8% | 22% | **75%** |

## 4 Conclusion

In this work, I explore how crowdsourced data can be leveraged to train agents that can be used in the real world, particularly in the context of the RoboTurk platform. The goal of this project is to understand how different embodiments of RoboTurk can be utilized to achieve this end goal. Primarily, I explore the two paradigms of learning from simulation and learning directly in the real world. I showcase the different challenges faced in designing a system for learning and deploying policies in either of these cases, and the different methods that are applicable. Modern sim2real methods still require unfeasible amounts of either simulation or real world data to achieve good dynamics or visual transfer. Furthermore, policies are particularly sensitive to dynamics mismatch between sim and real, for which

In the end, we show that learning directly in the real world has more promise moving forward. By filtering out the high frequency noise in crowdsourced data, we can make the data easier to learn from using Batch RL methods. Using IRIS, we are able to reach 75% accuracy when learning just from sparse rewards completely offline, a promising result given that the data and environment used are not much different from real world scenarios. This is a tremendous result that suggests we can learn effectively and efficiently for real robots just by collecting some crowdsourced data on the task. As soon as possible, we would like to confirm these results on a real robot system.

## References

OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Józefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, and et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, Nov 2019. ISSN 1741-3176. doi: 10.1177/0278364919887447. URL http://dx.doi.org/10.1177/0278364919887447.

Brenna D. Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, May 2009. ISSN 0921-8890. doi: 10.1016/j.robot.2008.10.024.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

A. D. Dragan and S. S. Srinivasa. Online customization of teleoperation interfaces. In *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*, pp. 919–924, 2012.

Linxi Fan, Yuke Zhu, Jiren Zhu, Zihua Liu, Orien Zeng, Anchit Gupta, Joan Creus-Costa, Silvio Savarese, and Li Fei-Fei. Surreal: Open-source reinforcement learning framework and robot manipulation benchmark. In *Conference on Robot Learning*, 2018.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning, 2020.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. *CoRR*, abs/1812.02900, 2018. URL http://arxiv.org/abs/1812.02900.

Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

Ken Goldberg, Michael Mascha, Steven Gentner, Nick Rothenberg, Carl Sutter, and Jeff Wiegley. Beyond the web: excavating the real world via mosaic. In *WWW Spring 1994*, 1994.

Stephen James, Paul Wohlhart, Mrinal Kalakrishnan, Dmitry Kalashnikov, Alex Irpan, Julian Ibarz, Sergey Levine, Raia Hadsell, and Konstantinos Bousmalis. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2019. doi: 10.1109/cvpr.2019.01291. URL http://dx.doi.org/10.1109/CVPR.2019.01291.

Rae Jeong, Yusuf Aytar, David Khosid, Yuxiang Zhou, Jackie Kay, Thomas Lampe, Konstantinos Bousmalis, and Francesco Nori. Self-supervised sim-to-real adaptation for visual robotic manipulation, 2019.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 25*, pp. 1097–1105. Curran Associates, Inc., 2012. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction, 2019.

Sascha Lange, Thomas Gabel, and Martin Riedmiller. *Batch Reinforcement Learning*, pp. 45–73. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-27645-3. doi: 10.1007/978-3-642-27645-3_2. URL https://doi.org/10.1007/978-3-642-27645-3_2.

Michael Laskey, Caleb Chuck, Jonathan Lee, Jeffrey Mahler, Sanjay Krishnan, Kevin Jamieson, Anca Dragan, and Ken Goldberg. Comparing human-centric and robot-centric sampling for robot deep learning from demonstrations. *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017. doi: 10.1109/icra.2017.7989046. URL http://dx.doi.org/10.1109/ICRA.2017.7989046.

Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. *Conference on Robot Learning (CoRL)*, 2019. URL https://arxiv.org/abs/1903.01973.

Ajay Mandlekar, Yuke Zhu, Animesh Garg, Jonathan Booher, Max Spero, Albert Tung, Julian Gao, John Emmons, Anchit Gupta, Emre Orbay, Silvio Savarese, and Li Fei-Fei. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, 2018.

Ajay Mandlekar, Jonathan Booher, Max Spero, Albert Tung, Anchit Gupta, Yuke Zhu, Animesh Garg, Silvio Savarese, and Li Fei-Fei. Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity, 2019a.

Ajay Mandlekar, Fabio Ramos, Byron Boots, Silvio Savarese, Li Fei-Fei, Animesh Garg, and Dieter Fox. Iris: Implicit reinforcement without interaction at scale for learning control from offline robot manipulation data, 2019b.

Christopher D. Manning. Computational linguistics and deep learning. *Comput. Linguist.*, 41(4): 701–707, December 2015. ISSN 0891-2017. doi: 10.1162/COLI_a_00239. URL https://doi.org/10.1162/COLI_a_00239.

Jun Nakanishi, Rick Cory, Michael Mistry, Jan Peters, and Stefan Schaal. Operational space control: A theoretical and empirical comparison. *The International Journal of Robotics Research*, 27(6): 737–757, 2008. doi: 10.1177/0278364908091463. URL https://doi.org/10.1177/0278364908091463.

OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. Solving rubik's cube with a robot hand, 2019.

Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J. Andrew Bagnell, Pieter Abbeel, and Jan Peters. An algorithmic perspective on imitation learning. *Foundations and Trends in Robotics*, 7(1-2): 1–179, 2018. ISSN 1935-8261. doi: 10.1561/2300000053. URL http://dx.doi.org/10.1561/2300000053.

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018. doi: 10.1109/icra.2018.8460528. URL http://dx.doi.org/10.1109/ICRA.2018.8460528.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250, 2016. URL http://arxiv.org/abs/1606.05250.

Fabio Ramos, Rafael Possas, and Dieter Fox. Bayessim: Adaptive domain randomization via probabilistic inference for robotics simulators. *Robotics: Science and Systems XV*, Jun 2019. doi: 10.15607/rss.2019.xv.029. URL http://dx.doi.org/10.15607/RSS.2019.XV.029.

Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet?, 2019.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. *Learning Representations by Back-Propagating Errors*, pp. 696–699. MIT Press, Cambridge, MA, USA, 1988. ISBN 0262010976.

Laura Smith, Nikita Dhawan, Marvin Zhang, Pieter Abbeel, and Sergey Levine. Avid: Learning multi-stage tasks via pixel-level translation of human videos, 2019.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep 2017. doi: 10.1109/iros.2017.8202133. URL http://dx.doi.org/10.1109/IROS.2017.8202133.

Haohan Wang and Bhiksha Raj. On the origin of deep learning, 2017.

Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), June 2016. URL https://www.microsoft.com/en-us/research/publication/msr-vtt-a-large-video-description-dataset-for-bridging-video-and-language/.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. doi: 10.1109/iccv.2017.244. URL http://dx.doi.org/10.1109/ICCV.2017.244.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences, 2019.