Adapting control policies from simulation to reality using a pairwise loss

Ulrich Viereck, Kate Saenko, and Robert Platt

Abstract This paper proposes an approach to domain transfer based on a pairwise loss function that helps transfer control policies learned in simulation onto a real robot. We explore the idea in the context of a "category level" manipulation task where a control policy is learned that enables a robot to perform a mating task involving novel objects. We explore the case where depth images are used as the main form of sensor input. Our experimental results demonstrate that proposed method consistently outperforms baseline methods that train only in simulation or that combine real and simulated data in a naive way.

1 Motivation, Problem Statement, Related work

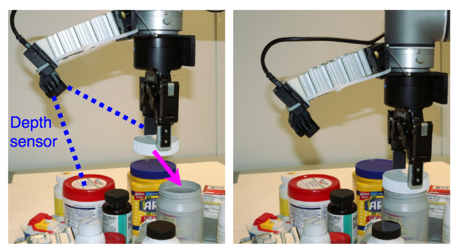

Fig. 1 Our goal is to learn a controller that uses depth image feedback to mate the cap to the bottle in the presence of clutter. Experimental setup on UR5 robot with a Intel RealSense depth sensor mounted as shown.

Recently, there has been a lot of interest in using deep neural networks to learn "pixels-to-torques" visuomotor controllers: robotic controllers that take sequential image data as input and produce low level motor commands as output. Ideally, we would learn these controllers with training data collected using real robotic hardware [3]. However, this approach is rarely feasible because of the large amounts of training experience typically required to train deep neural networks. Instead, it is convenient to learn pixels-to-torques control policies using simulated data. Unfortunately, this exposes us to the *dataset shift problem* [4]. When the simulation is not sufficiently accurate, then the control policy learned in simulation may not work well in reality. There are two types of simulation errors that typically can cause domain shift problems: 1) errors simulating images that the robot would observe, and 2) errors simulating real world contact

Ulrich Viereck, Robert Platt
College of Computer and Information Science, Northeastern University, e-mail: {uliv,rplatt}@ccs.neu.edu

Kate Saenko
Department of Computer Science, Boston University, e-mail: saenko@bu.edu

and frictional dynamics. This paper limits consideration to non-contact tasks and therefore the focus is on domain shift errors caused by image simulation and depth image simulation in particular.

In this paper, we propose an approach to transferring visuomotor control policies learned on simulated depth data to real world observations. The key idea is to reduce the gap between simulation and reality by augmenting the simulated data used to train the system with a small amount of real robot data. Each piece of real robot data is paired with a piece of simulated data that corresponds to the same robot state. We train the neural network using a loss function that has two terms: a *task loss* that encodes the desired robotic behavior and a *pairwise loss* that penalizes networks that do not represent real and simulated data the same way. This paper makes two contributions relative to prior work: 1) we propose a neural network architecture that combines the pairwise loss approach to domain transfer with a pixels-to-torques controller; 2) we characterize the method for depth image data rather than for RGB data. Unlike [6] which uses the pairwise loss function as part of the state estimator and only explores single task instances, our approach learns controllers that can solve “category level” manipulation tasks where object shape and size varies from one instance of the task to the next. We find that the approach can work well even when the real data is produced in a simplified version of the actual robotic scenario that is experienced at test time. This paper complements a variety of recent literature on the subject including work that uses “domain randomization” of simulated images to affect better transfer to reality [5] and work using GANs to affect the transfer [1]. In contrast to that work, the approach followed here is simpler than GAN-based approaches and more relevant to depth data than domain randomization methods.

2 Technical Approach

We learn a pixels-to-torques controller that takes depth images as input and outputs manipulator displacements. The controller is based on a method proposed in our prior work [7] where we estimate a distance function with respect to a goal state using a neural network. Given an image and a candidate manipulator displacement, the distance function predicts expected distance-to-goal on the following time step. We select a manipulator displacement by sampling a set of candidate displacements and selecting the one that is predicted to move closest to a goal. Essentially, this method learns a value function over the cross product of observation and action (depth image and manipulator displacement). However, instead of using reinforcement learning, we train the neural network directly by using supervised learning with distance targets produced by our simulator. Specifically, we create a dataset by sampling from a space of possible task scenarios and initial conditions. For each sample, we simulate the depth image that would be observed and calculate distance-to-goal after performing the associated displacement.

This paper characterizes a domain transfer technique based on the following loss function (similar to what was originally proposed in [6]):

$$
\begin{aligned}
\mathscr{L} = \alpha & \sum_{(I,a)\in\mathbb{X}_S} \|g\left(f(I,a;\theta_f),\theta_g\right) - y(I,a)\|_1 \\
&+ \beta \sum_{(I,a)\in\mathbb{X}_T} \|g\left(f(I,a;\theta_f),\theta_g\right) - y(I,a)\|_1 \\
&+ \gamma \sum_{(I_S,I_T,a)\in\mathbb{X}_{ST}} \|f(I_S,a;\theta_f) - f(I_T,a;\theta_f)\|_2. \qquad (1)
\end{aligned}
$$

Here, we write the neural network as a composition of two functions, f and g. f denotes the "early" part of the network comprised of convolutional layers that learn feature representations for the images and actions. g denotes the "late" fully connected layers that encode the regression task. The output of the end-to-end network for an image I and action (i.e. displacement) a is written $g(f(I,a;\theta_f);\theta_g)$, where θ_f denotes the parameters of f and θ_g denotes parameters of g. The loss is evaluated over a training dataset comprised of two parts: the data from source domain and the data from the target domain. The source domain consists of simulated images and actions $(I,a) \in \mathbb{X}_S$ and the associated labels $y(I,a)$ (these are the distance-to-goal that would result from taking action a from the state that produced image I). The target domain $\mathbb{X}_T$ consists of real images and actions paired with the associated labels. The set $\mathbb{X}_{ST}$ consists of triples (I_S,I_T,a) where I_S and I_T are the simulated-real image pair and a is the action that was taken. We assume that the cardinality of $\mathbb{X}_S$ is much larger than either $\mathbb{X}_T$ or $\mathbb{X}_{ST}$, i.e. that we have many more simulated images than real images. The first two terms of Equation 1 are called *task losses*. The third is the *pairwise loss*. The first and second task losses are minimized when the neural network fits the simulated and real data well, respectively. The third is minimized when the network assigns both simulated and real depth images the same high-level encoding. The pairwise loss is critical: by "preferring" networks that encode real and simulated data similarly, this term facilitates good transfer from simulation to reality. The approach is implemented by the neural network architecture illustrated in Figure 2. It takes as input a pair of depth images, I_1 and I_2 (matching paired images from simulation and reality), and an action $a = (x,y) \in \mathbb{R}^2$. It learns a function, $d(I,a) \in \mathbb{R}_{>0}$, that describes the distance between the object in the hand and the target after displacing the manipulator by a.

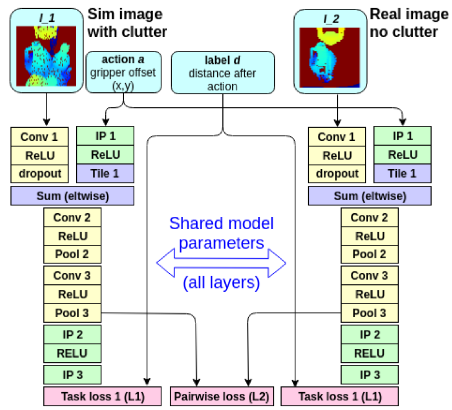

Fig. 2 Neural network architecture. The pairwise loss at pool3 favors networks that give real and simulated images similar representations.

We train using a combination of real and simulated data. The simulated portion of the dataset is generated using OpenRAVE [2] to generate 100k 64x64 pixel depth images. Each depth image is taken for a random robot configuration with clutter objects selected randomly from a set of more than 250 objects and placed randomly in the vicinity of the target bottle. For each of the 100k scenes, we sample 100 actions (i.e. hand displacements) and estimate the distance-to-goal that would result if that action were executed. Finally, we simulate missing pixel noise by setting the value of each pixel to 0 with a 10% probability. We also collect 2904 labeled real training images on the robot (UR5, see Figure 1) and measure the corresponding ground truth distance-to-goal. Importantly, the real images used for training do *not* have clutter. As a result, it is easier to obtain training data semi-automatically because it is unnecessary to reproduce simulated clutter on the real system. These real images were paired with 2904 simulated images that portray the same robotic state.

3 Experiments

We evaluate this approach using the *cap-on-bottle* task, where the robot must align a cap with a bottle opening. This is a challenging task because we are learning a "category level" manipulation skill where the system must learn to perform the task for novel object instances in the presence of randomly placed clutter. We use a Robotiq 85 two-fingered hand mounted on a UR5 arm. The Intel RealSense SR300 is mounted to the UR5 wrist as shown in Figure 1. The RealSense creates depth images that are input to our controller and used to estimate desired hand displacements. A test bottle is placed on a table surrounded by clutter intended to make the task more challenging. The bottle cap is manually placed into the robotic hand. The position of the bottle is unknown to the algorithm but measured for the purposes of experimental evaluation. The gripper is initialized to a random offset within a 10cm box centered on the bottle at a height of 5cm above the table. At each iteration, the controller acquires a depth image, samples 1k manipulator displacements, moves the gripper in the direction predicted to reduce distance-to-goal by the most, and moves toward the table by 1cm. After executing 5 iterations, we measure how close the cap is to the bottle opening.

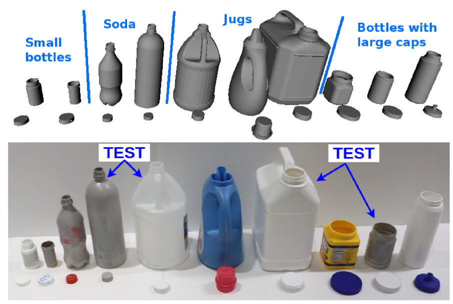

Fig. 3 Simulated and real bottles used in our experiments. From left to right: *vitamin, pills, soda, water, 1-gal, laundry, 2-gal, corn starch, peanut butter, sport bottle*. Test bottles are shown with arrows.

Our proposed approach is to train the neural network using both the task and pairwise loss over both simulated and real images. We compare this approach against four different baselines: 1) training the network using only real images *without* clutter; 2) training using only real images *with* clutter; 3) training using simulated images with clutter; 4) training using simulated with clutter combined with real images

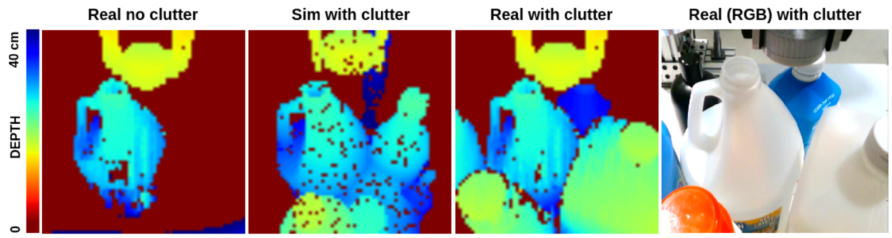

Fig. 4 Depth images generated for training. *Real no clutter* and *Sim no clutter* are paired and augmented by *Sim with clutter* images.

without clutter. We will refer to the combination of the four baselines and the proposed approach as the five "domain transfer methods".

We evaluate the five domain transfer methods over a set of five different problem scenarios. In each problem scenario, we train using data derived from a different object category. The five object categories are: (i) small pill bottles; (ii) soda bottles; (iii) jugs; (iv) bottles with large caps; (v) cornstarch and vitamin bottles. These categories are illustrated in Figure 3. Figure 5 shows the results grouped by object category. A separate network is trained for each object category and is evaluated on as many as four different test objects. (We do not test on objects that happen also to

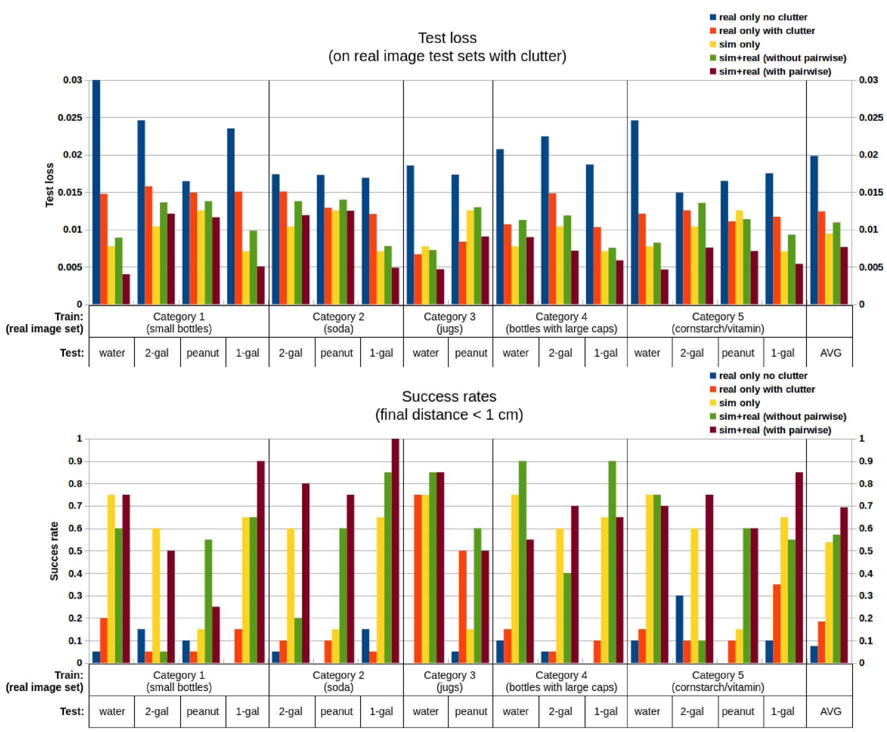

Fig. 5 Test losses (top) and average success rates (bottom) on real image test sets with clutter. See text for details.

be in the object category used for training.) For example, the "Category 1" results shown as the far left of Figure 5 show performance for each of the five domain transfer methods evaluated on four different test objects (water, 2-gal, peanut, and 1-gal). The top of Figure 5 show L1 test loss of the learned network (lower is better). The bottom of Figure 5 shows the success rate of the end-to-end controller (we define "success" to occur when the cap final position is within 1cm of the position of the bottle top). Each bar is an average of 20 trials (higher is better).

4 Main Experimental Insights

The five bars at the far right of Figure 5 labeled "AVG" summarize the comparison. Each bar (i.e. each domain transfer method) is an average of all experiments for that domain transfer type. This illustrates a few key results. First, the domain transfer method using our proposed task-pairwise loss function does best overall (lowest loss, highest task success rate). Second, the method using only real images does worst, probably because we do not train on enough different objects to facilitate generalization to novel objects. Third, training using only simulated images does pretty well (54% success rate): not quite as well as we can do using our proposed method (70% success rate), but not nearly as bad as training on only real data. Overall, we conclude that training visuomotor policies for category-level tasks in simulation is a promising approach, and that by collecting a small amount of labeled real data in simplified scenarios and using the pairwise loss, we can improve performance on real systems.

References

1. Konstantinos Bousmalis, Alex Irpan, Paul Wohlhart, Yunfei Bai, Matthew Kelcey, Mrinal Kalakrishnan, Laura Downs, Julian Ibarz, Peter Pastor, Kurt Konolige, et al. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. *arXiv preprint arXiv:1709.07857*, 2017.
2. R. Diankov. Openrave. *http://openrave.org*.
3. Sergey Levine, Peter Pastor, Alex Krizhevsky, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *arXiv preprint arXiv:1603.02199*, 2016.
4. Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009.
5. 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. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 23–30. IEEE, 2017.
6. Eric Tzeng, Coline Devin, Judy Hoffman, Chelsea Finn, Pieter Abbeel, Sergey Levine, Kate Saenko, and Trevor Darrell. Adapting deep visuomotor representations with weak pairwise constraints. *arXiv preprint arXiv:1511.07111*, 2015.
7. Ulrich Viereck, Andreas Pas, Kate Saenko, and Robert Platt. Learning a visuomotor controller for real world robotic grasping using simulated depth images. In *Conference on Robot Learning (CoRL)*, 2017.