Robust Object Pose Tracking for Robotics

Chelsea Finn Justin Fu Nopphon Siranart

***Abstract*—Policy search methods based on reinforcement learning and optimal control can allow robots to automatically learn a wide range of tasks. However, practical applications of policy search tend to require the policy to be supported by hand-engineered components for perception, state estimation, and low-level control. We propose a method for learning policies that map raw, low-level observations, consisting of joint angles and camera images, directly to the torques at the robot's joints. The policies are represented as deep convolutional neural networks (CNNs) with 92,000 parameters. The high dimensionality of such policies poses a tremendous challenge for policy search. To address this challenge, we develop a sensorimotor guided policy search method that can handle high-dimensional policies and partially observed tasks. We use BADMM to decompose policy search into an optimal control phase and supervised learning phase, allowing CNN policies to be trained with standard supervised learning techniques. This method can learn a number of manipulation tasks that require close coordination between vision and control, including inserting a block into a shape sorting cube, screwing on a bottle cap, fitting the claw of a toy hammer under a nail with various grasps, and placing a coat hanger on a clothes rack.**

I. Introduction

Reinforcement learning and policy search methods hold the promise of allowing robots to acquire new behaviors through experience. They have been applied to a range of robotic tasks, including manipulation [1, 9] and locomotion [3, 5, 11, 20]. However, policies learned using such methods often rely on a number of hand-engineered components for perception and low-level control. The policy might specify a trajectory in task-space, relying on hand-designed PD controllers to execute the desired motion, and a policy for manipulating objects might rely on an existing vision system to localize these objects [17]. The vision system in particular can be complex and prone to errors, and its performance is typically not improved during policy training, nor adapted to the goal of the task.

We propose a method for learning policies that directly map raw observations, including joint angles and camera images, to motor torques. The policies are trained end-to-end using real-world experience, optimizing both the control and perception components on the same measure of task performance. This allows the policy to learn goal-driven perception, which avoids the mistakes that are most costly for task performance. Learning perception and control in a general and flexible way requires a large, expressive model. Our policies are represented with convolutional neural networks (CNNs), which have 92,000 parameters and 7 layers. Deep CNN models have been shown to achieve state of the art results on a number of supervised vision tasks [6, 12, 21], but sensorimotor deep learning remains a challenging prospect. The policies are extremely high dimensional, and the control task is partially observed, since part of the state must be inferred from images.

To address these challenges, we extend the framework of guided policy search to sensorimotor deep learning. Guided

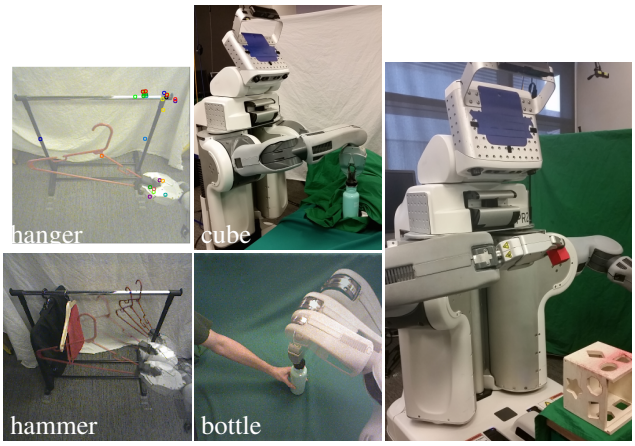


Fig. 1: Our method learns visuomotor policies that directly use camera image observations (left) to set motor torques on a PR2 robot (right).

policy search decomposes the policy learning problem into two phases: a trajectory optimization phase that determines how to solve the task in a few specific conditions, and a supervised learning phase that trains the policy from these successful executions with supervised learning [13]. Since the CNN policy is trained with supervised learning, we can use the tools developed in the deep learning community to make this phase simple and efficient. We handle the partial observability of visuomotor control by optimizing the trajectories with full state information, while providing only partial observations (consisting of images and robot configurations) to the policy. The trajectories are optimized under unknown dynamics, using real-world experience and minimal prior knowledge.

The main contribution of our work is a method for end-to-end training of deep visuomotor policies for robotic manipulation. We propose a partially observed guided policy search algorithm that can train high-dimensional policies for tasks where part of the state must be determined from camera images. We also introduce a novel CNN architecture designed for robotic control, shown in Figure 2. The vision layers of this CNN are designed for localizing points of interest in an image, unlike standard vision architectures that discard locational information to induce translational invariance [12]. We evaluate our method by learning policies for inserting a block into a shape sorting cube, screwing a cap onto a bottle, fitting the claw of a toy hammer under a nail with various grasps, and placing a coat hanger on a rack (see Figure 1). Our results demonstrate clear improvements in consistency and generalization from training visuomotor policies end-to-end, when compared to using the poses or features produced by a CNN trained for 3D object localization.

II. Related Work

Reinforcement learning and policy search have been applied in robotics for playing games such as table tennis [9], object

manipulation [1, 17], and locomotion [3, 5, 11, 20]. Several recent papers provide surveys of policy search in robotics [2, 10]. Such methods are typically applied to one component of the robot control pipeline, which often sits on top of a hand-designed controller, such as a PD controller, and accepts processed input, for example from an existing vision pipeline [17]. Our method trains policies that map visual input and joint encoder signals directly to the torques at the robot's joints. By learning the entire mapping from perception to control, the perception layers can be adapted to optimize task performance.

The goal of our approach is also similar to visual servoing, which performs feedback control on feature points in a camera image [4, 16, 23]. However, our visuomotor policies are entirely learned from real-world data, and do not require feature points or feedback controllers to be specified by hand. This gives our method considerable flexibility in choosing how to use the visual signal. Furthermore, our approach does not require any sort of camera calibration, in contrast to many visual servoing methods (though not all – see e.g. [7, 24]).

III. Overview

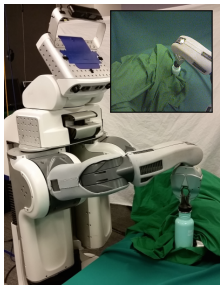
Naïve supervised learning will often fail to produce a good policy, since a small mistake on the part of the policy will put it in states that are not part of the training, causing compounding errors. To avoid this problem, the training data must come from the policy's own state distribution [18]. We use BADMM [22] to adapt the trajectories to the policy, alternating between optimizing the policy to match the trajectories, and optimizing the trajectories to minimize cost and match the policy, such that at convergence, they have the same state distribution.

A. Network Architecture

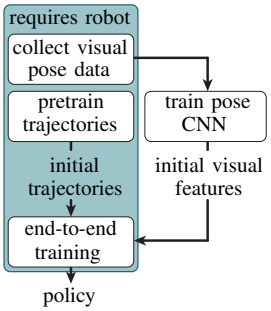
Our network architecture is shown in Figure 2. The visual processing layers of the network consist of three convolutional layers, each of which learns a bank of filters that are applied to patches centered on every pixel of its input. These filters form a hierarchy of local image features. Each convolutional layer is followed by a rectifying nonlinearity of the form $a_{cij} = max(0, z_{cij})$ for each channel c and each pixel coordinate (i, j). The third convolutional layer contains 32 response maps with resolution 109×109. These response maps are passed through a spatial softmax function of the form $s_{cij} = e^{a_{cij}} / \sum_{i'j'} e^{a_{ci'j'}}$. Each output channel of the softmax is a probability distribution over the location of a feature in the image. To convert from this distribution to a spatial representation, the network calculates the expected image position of each feature, yielding a 2D coordinate for each channel. These feature points are concatenated with the robot's configuration and fed through two fully connected layers, each with 40 rectified units, followed by linear connections to the torques. The full visuomotor policy contains about 92,000 parameters, of which 86,000 are in the convolutional layers.

The spatial softmax and the expected position computation serve to convert pixel-wise representations in the convolutional layers to spatial coordinate representations, which can be manipulated by the fully connected layers into 3D positions or motor torques. The softmax also provides lateral inhibition, which suppresses low, erroneous activations. This makes our policy more robust to distractors, providing generalization to novel visual variation. We compare our architecture with more standard alternatives in Section IV-C.

B. Training Data



We train the policy using the full state during the trajectory optimization phase, though the final policy acts under partial observations. This type of instrumented training is a natural choice for many robotics tasks, where the robot is trained under controlled conditions, but must then act intelligently in uncontrolled, real-world situations. In our tasks, the unobserved variables are the pose of a target object (e.g. the bottle on which a cap must be placed). During training, this target object is held in the robot's left gripper, while the robot's right arm performs the task, as shown above. This allows the robot to move the target through a range of known positions. The final visuomotor policy does not receive this position as input, but must instead use the camera images. The left arm is covered with cloth to prevent the policy from associating its appearance with the object's position.



To initialize the trajectories, we take 15 iterations of guided policy search without optimizing the visuomotor policy. This allows for much faster training in the early iterations, when the trajectories are not yet successful, and optimizing the full visuomotor policy is unnecessarily time consuming. Since we still want the trajectories to arrive at compatible strategies for each target position, we replace the visuomotor policy during these iterations with a small network that receives the full state. This network serves only to constrain the trajectories and avoid divergent behaviors from emerging for similar initial states, which would make subsequent policy learning difficult. As shown in the diagram above, the trajectories can be pre-trained in parallel with the vision layer pre-training, which does not require the robot.

IV. Experimental Results

We evaluated our method by training policies for hanging a coat hanger on a clothes rack, inserting a block into a shape sorting cube, fitting the claw of a toy hammer under a nail with various grasps, and screwing on a bottle cap. The cost function for these tasks encourages low distance between three points on the end-effector and corresponding target points, low torques, and, for the bottle task, spinning the wrist. The equations for these cost functions follow prior work [14]. The tasks are illustrated in Figure 3. Each task involved variation of about 10-20 cm in each direction in the position of the target object (the rack, shape sorting cube, nail, and bottle). In addition, the coat hanger and hammer tasks were trained with two and three grasps, respectively. All tasks used the same policy architecture and model parameters.

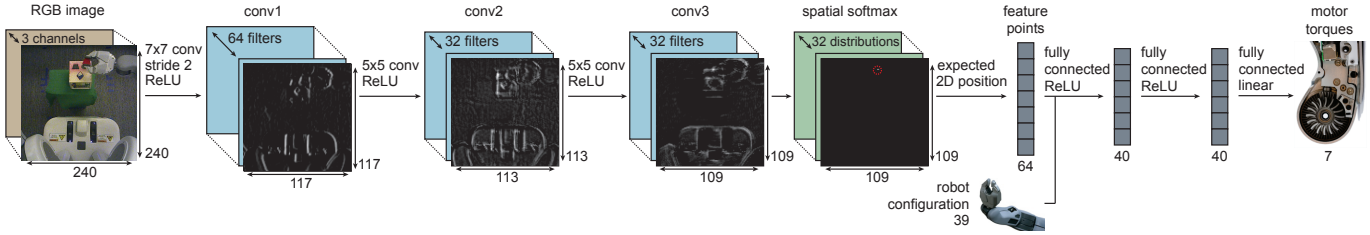
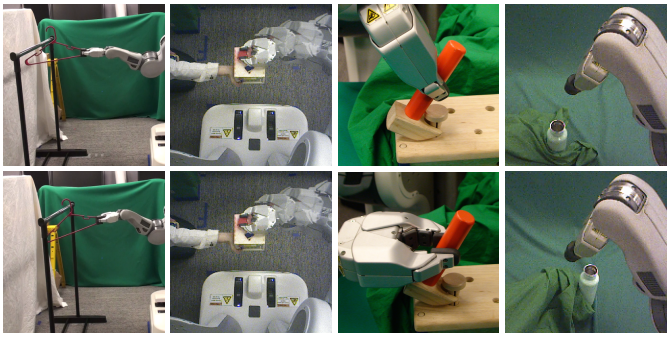
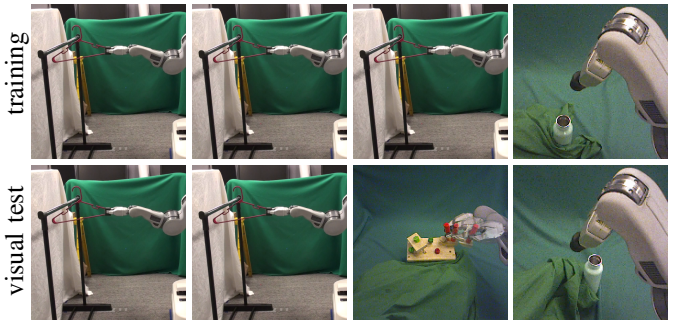


Fig. 2: Visuomotor policy architecture. The network contains three convolutional layers, followed by a spatial softmax and an expected position layer that converts pixel-wise features to feature points, which are better suited for spatial computations. The points are concatenated with the robot configuration, then passed through three fully connected layers to produce the torques.



(a) hanger (b) cube (c) hammer (d) bottle

Fig. 3: Illustration of the tasks in our experiments, showing the variation in the position of the target for the hanger, cube, and bottle tasks, as well as two of the three grasps for the hammer, which also included variation in position (not shown).



(a) hanger (b) cube (c) hammer (d) bottle

Fig. 4: Training and visual test scenes as seen by the policy at the ends of successful episodes. The hammer and bottle images were cropped for visualization only.

A. *Visuomotor Policy Generalization*

We evaluated the visuomotor policies in three conditions: (1) the training target positions and grasps, (2) new target positions not seen during training and, for the hammer, new grasps (spatial test), and (3) training positions with visual distractors (visual test). A selection of these experiments is shown in the supplementary video.[1] For the visual test, the shape sorting cube was placed on a table rather than held in the gripper, the coat hanger was placed on a rack with clothes, and the bottle and hammer tasks were done in the presence of clutter. Illustrations of this test are shown in Figure 4.

The success rates for each test are shown in Table I. We compared to two baselines, both of which train the vision layers in advance for pose prediction, instead of training the entire policy end-to-end. The features baseline discards the last layer of the pose predictor and uses the feature points, resulting in the same architecture as our policy, while the prediction baseline feeds the predicted pose into the control layers.

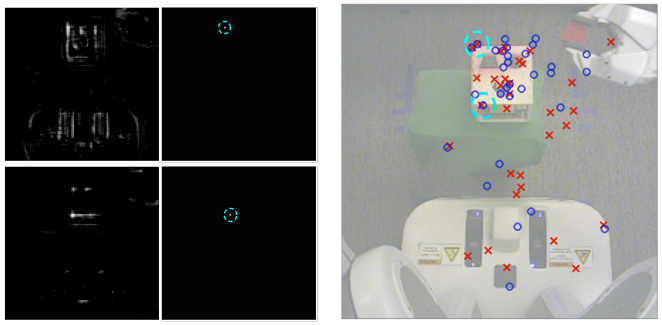
The pose prediction baseline is analogous to a standard modular approach to policy learning, where the vision system is first trained to localize the target, and the policy is trained on top of it. This variant achieves poor performance, because although the pose is accurate to about 1 cm, this is insufficient for such precise tasks. As shown in the video, the shape sorting cube and bottle cap insertions have tolerances of just a few millimeters. Such accuracy is difficult to achieve even with calibrated cameras and checkerboards. Indeed, prior work has reported that the PR2 can maintain a camera to end effector accuracy of about 2 cm during open loop motion [15]. This suggests that the failure of this baseline is not atypical, and that our visuomotor policies are learning visual features and control strategies that improve the robot's accuracy.

When provided with pose estimation features, the policy has more freedom in how it uses the visual information, and achieves somewhat higher success rates. However, full end-to-end training performs significantly better, achieving high accuracy even on the challenging bottle task, and successfully adapting to the variety of grasps on the hammer task. This suggests that, although the vision layer pre-training is clearly beneficial for reducing computation time, it is not sufficient by itself for discovering good features for visuomotor policies.

coat hanger	training (18)	spatial test (24)	visual test (18)
end-to-end training	**100%**	**100%**	**100%**
pose features	88.9%	87.5%	83.3%
pose prediction	55.6%	58.3%	66.7%
shape sorting cube	training (27)	spatial test (36)	visual test (40)
end-to-end training	**96.3%**	**91.7%**	**87.5%**
pose features	70.4%	83.3%	40%
pose prediction	0%	0%	n/a
toy claw hammer	training (45)	spatial test (60)	visual test (60)
end-to-end training	**91.1%**	**86.7%**	**78.3%**
pose features	62.2%	75.0%	53.3%
pose prediction	8.9%	18.3%	n/a
bottle cap	training (27)	spatial test (12)	visual test (40)
end-to-end training	**88.9%**	**83.3%**	**62.5%**
pose features	55.6%	58.3%	27.5%

TABLE I: Success rates on training positions, on novel test positions, and in the presence of visual distractors. The number of trials per test is shown in parentheses.

[1]The video can be viewed at http://sites.google.com/site/visuomotorpolicy



(a) conv3 (b) softmax (c) feature points

Fig. 5: Feature points learned by the shape sorting cube policy. Two of the 32 conv3 response maps are shown in (a), and the corresponding softmax distributions are displayed in (b). In (c), we show the output feature points for this input image in blue, while the feature points of the pose prediction network are shown in red. The end-to-end trained model discovers more feature points on the cube and the gripper.

The policies exhibit moderate tolerance to distractors that are visually separated from the target object. However, as expected, they tend to perform poorly under drastic changes to the backdrop, or when the distractors are adjacent to or occluding the manipulated objects, as shown in the supplementary video. In future work, this could be mitigated by varying the scene at training time, or by artificially augmenting the image samples with synthetic transformations, as discussed in prior work in computer vision [19].

B. Features Learned with End-to-End Training

In Figure 5, we compare the feature points learned through guided policy search to those learned by a CNN trained for pose prediction. After end-to-end training, the policy acquired a distinctly different set of feature points compared to the pose prediction CNN used for initialization. The end-to-end trained model finds more feature points on task-relevant objects and fewer points on background objects. This suggests that the policy improves its performance by acquiring *task-specific* visual features that differ from those learned for object localization. We further analyze the features learned by our policies in the supplementary appendix.

C. CNN Architecture Evaluation

To evaluate the visual processing portion of our architecture, we measured its accuracy on the pose estimation pre-training task discussed in Section III-B. We compare to a network where the fixed transformation from the softmax to the feature points is replaced with a conventional learned fully connected layer, as well as to networks that omit the softmax and use 3×3 max pooling with stride 2 at the first two layers. These alternative architectures have many more parameters, since the new fully connected layer takes as input the entire bank of response maps from the third convolutional layer. The results in Table II indicate that using the softmax and the fixed transformation from the softmax output to the spatial feature representation improves pose estimation accuracy and reduces overfitting. Our network is able to outperform the more standard architectures because it is forced by the softmax and expected position layers to learn feature points, which provide a concise representation suitable for spatial inference. The lower number of parameters also results in an easier optimization and reduces overfitting.

network architecture	training error (cm)	test error (cm)
softmax + feature points (**ours**)	**1.14 ± 1.67**	**1.30 ± 0.73**
softmax + fully connected layer	2.27 ± 1.70	2.59 ± 1.19
fully connected layer	4.65 ± 2.90	4.75 ± 2.29
max-pooling + fully connected	2.89 ± 2.08	3.71 ± 1.73

TABLE II: Average pose estimation accuracy and standard deviation with various architectures, measured as average Euclidean error for the three target points in 3D, with ground truth determined by forward kinematics from the left arm.

D. Implementation and Computational Performance

CNN training was implemented using the Caffe [8] deep learning library. Each visuomotor policy required 3-4 hours of training time: 20-30 minutes for the pose prediction data collection on the robot, 40-60 minutes for the fully observed trajectory pre-training on the robot and offline pose pre-training (which can be done in parallel), and between 1.5 and 2.5 hours for end-to-end training with guided policy search. The coat hanger task required two iterations of guided policy search, the shape sorting cube and the hammer required three, and the bottle task required four. Training time was dominated by computation rather than robot interaction time, and we expect significant speedup from a more efficient implementation.

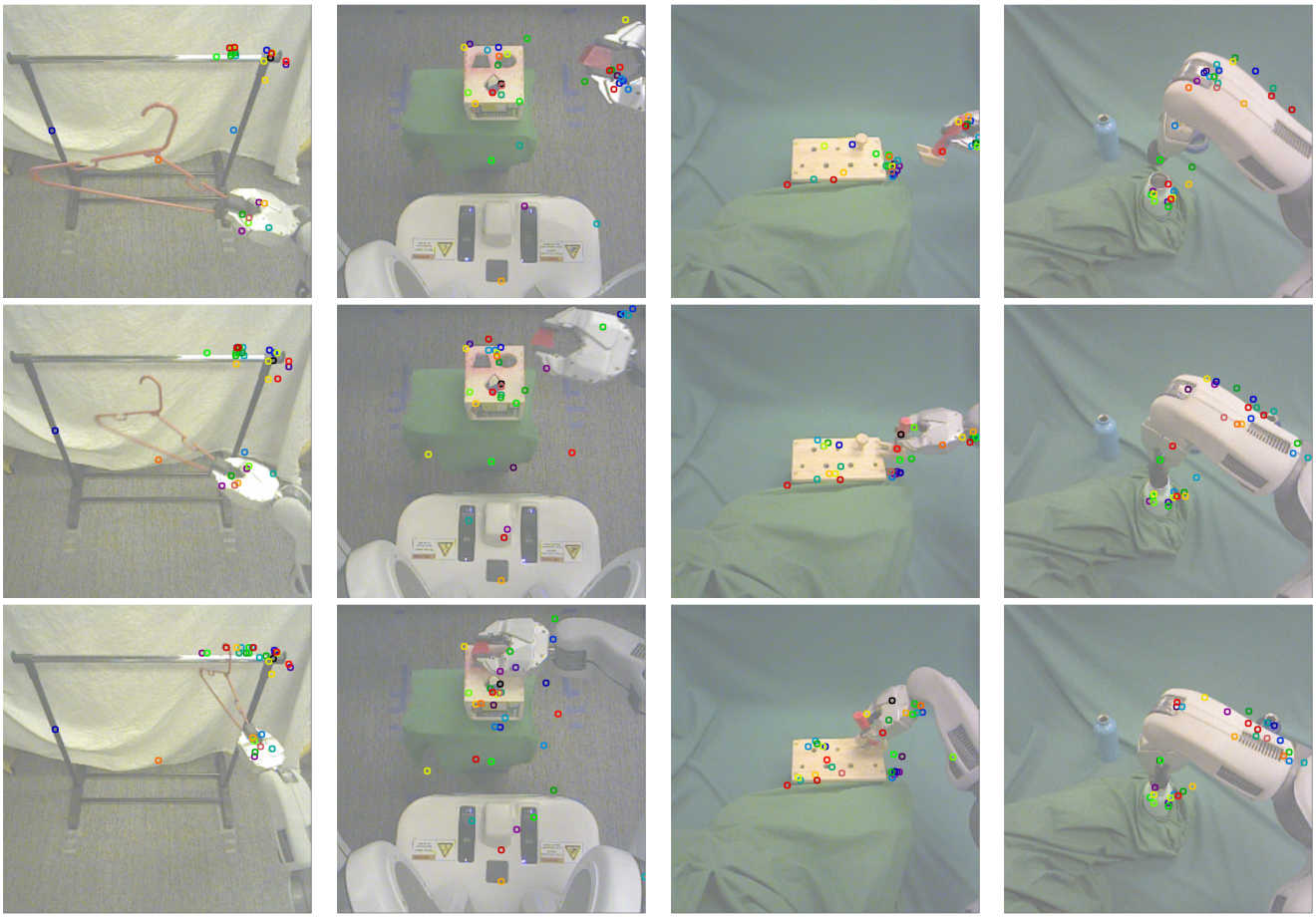
V. Discussion and Future Work

In this paper, we presented a method for learning robotic control policies that use raw input from a monocular camera. These policies are represented by a novel convolutional neural network architecture, and can be trained end-to-end using our partially observed guided policy search algorithm, which decomposes the policy search problem in a trajectory optimization phase that uses full state information and a supervised learning phase that only uses partial observations. This decomposition allows us to leverage state-of-the-art tools from supervised learning, making it straightforward to optimize extremely high-dimensional policies. Our experimental results show that our method can execute complex manipulation skills, and that end-to-end training produces significant improvements in policy performance compared to using fixed vision layers trained for pose prediction.

References

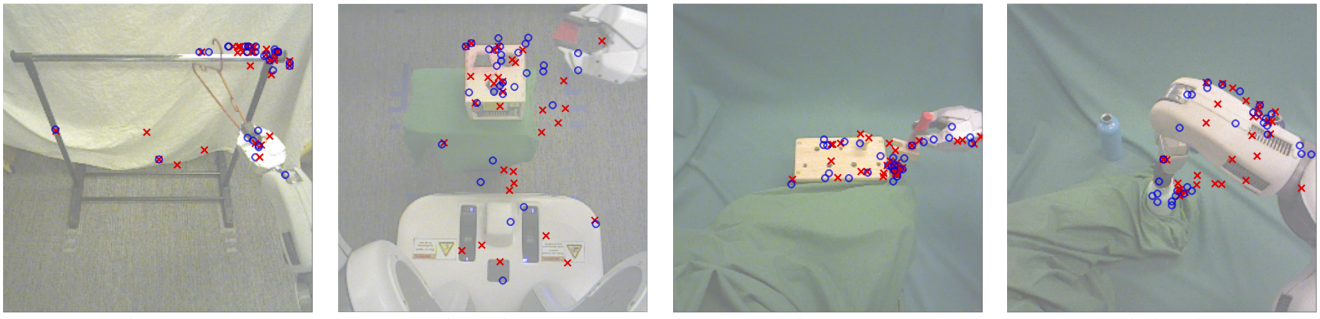
[1] M. Deisenroth, C. Rasmussen, and D. Fox. Learning to control a low-cost manipulator using data-efficient reinforcement learning. In *Robotics: Science and Systems (RSS)*, 2011.

[2] M. Deisenroth, G. Neumann, and J. Peters. A survey on policy search for robotics. *Foundations and Trends in Robotics*, 2(1-2):1–142, 2013.



(a) hanger (b) cube (c) hammer (d) bottle

Fig. 6: Feature points tracked by the policy during task execution for each of the four tasks. Each feature point is displayed in a different random color, with consistent coloring across images. The policy finds features on the target object and the robot gripper and arm. In the bottle cap task, note that the policy correctly ignores the distractor bottle in the background, even though it was not present during training.



(a) hanger (b) cube (c) hammer (d) bottle

Fig. 7: Feature points learned for each task. For each input image, the feature points produced by the policy are shown in blue, while the feature points of the pose prediction network are shown in red. The end-to-end trained policy tends to discover more feature points on the target object and the robot arm than the pose prediction network.

[3] G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng. Learning CPG-based biped locomotion with a policy gradient method: Application to a humanoid robot. *International Journal of Robotic Research*, 27(2):213–228, 2008.

[4] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3), 1992.

[5] T. Geng, B. Porr, and F. Wörgötter. Fast biped walking with a reflexive controller and realtime policy searching. In *Advances in Neural Information Processing Systems (NIPS)*, 2006.

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[7] M. Jägersand, O. Fuentes, and R. C. Nelson. Experimental evaluation of uncalibrated visual servoing for precision manipulation. In *International Conference on Robotics and Automation (ICRA)*, 1997.

[8] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[9] J. Kober, E. Oztop, and J. Peters. Reinforcement learning to adjust robot movements to new situations. In *Robotics: Science and Systems (RSS)*, 2010.

[10] J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *International Journal of Robotic Research*, 32(11):1238–1274, 2013.

[11] N. Kohl and P. Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *International Conference on Robotics and Automation (IROS)*, 2004.

[12] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*. 2012.

[13] S. Levine and V. Koltun. Learning complex neural network policies with trajectory optimization. In *International Conference on Machine Learning (ICML)*, 2014.

[14] S. Levine, N. Wagener, and P. Abbeel. Learning contact-rich manipulation skills with guided policy search. *arXiv preprint arXiv:1501.05611*, 2015.

[15] W. Meeussen, M. Wise, S. Glaser, S. Chitta, C. McGann, P. Mihelich, E. Marder-Eppstein, M. Muja, Victor Eruhimov, T. Foote, J. Hsu, R.B. Rusu, B. Marthi, G. Bradski, K. Konolige, B. Gerkey, and E. Berger. Autonomous door opening and plugging in with a personal robot. In *International Conference on Robotics and Automation (ICRA)*, 2010.

[16] K. Mohta, V. Kumar, and K. Daniilidis. Vision based control of a quadrotor for perching on planes and lines. In *International Conference on Robotics and Automation (ICRA)*, 2014.

[17] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal. Learning and generalization of motor skills by learning from demonstration. In *International Conference on Robotics and Automation (ICRA)*, 2009.

[18] S. Ross, G. Gordon, and A. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. *Journal of Machine Learning Research*, 15:627–635, 2011.

[19] P. Y. Simard, D. Steinkraus, and J. C. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *Seventh International Conference on Document Analysis and Recognition*, 2003.

[20] R. Tedrake, T. Zhang, and H. Seung. Stochastic policy gradient reinforcement learning on a simple 3d biped. In *International Conference on Intelligent Robots and Systems (IROS)*, 2004.

[21] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[22] H. Wang and A. Banerjee. Bregman alternating direction method of multipliers. In *Advances in Neural Information Processing Systems (NIPS)*. 2014.

[23] W. J. Wilson, C. W. Williams Hulls, and G. S. Bell. Relative end-effector control using cartesian position based visual servoing. *IEEE Transactions on Robotics and Automation*, 12(5), 1996.

[24] B. H. Yoshimi and P. K. Allen. Active, uncalibrated visual servoing. In *International Conference on Robotics and Automation (ICRA)*, 1994.