Combining Self-Supervised Learning and Imitation for Vision-Based Rope Manipulation

Ashvin Nair **Pulkit Agrawal** **Dian Chen** **Phillip Isola**

Pieter Abbeel **Jitendra Malik** **Sergey Levine** *

1 Introduction

Manipulation of deformable objects, such as ropes and cloth, is an important but challenging problem in robotics. Open-loop strategies for deformable object manipulation are often ineffective, since the material can shift in unpredictable ways [4]. Perception of cloth and rope also poses a major challenge, since standard methods for estimating the pose of rigid objects cannot be readily applied to deformable objects for which it is difficult to concretely define the degrees of freedom or provide suitable training data [7]. Despite the numerous industrial and commercial applications that an effective system for deformable object manipulation would have, effective and reliable methods for such tasks remain exceptionally difficult to construct. Previous work on deformable object manipulation has sought to use sophisticated finite element models [4, 2], hand-engineered representations [5, 11, 8], and direct imitation of human-provided demonstrations [6, 10]. Direct model identification for ropes and cloth is challenging and brittle, while imitation of human demonstrations without an internal model of the object's dynamics is liable to fail in conditions that deviate from those in the demonstrations.

In this work, we instead propose a learning-based approach to associate the behavior of a deformable object with a robot's actions, using self-supervision from large amounts of data gathered autonomously by the robot. In particular, the robot learns a goal-directed inverse dynamics model: given a current state and a goal state (both in image space), it predicts the action that will achieve the goal. To handle high-dimensional visual observations, we employ deep convolutional neural networks for learning the inverse dynamics model. Once this model is learned, our method can use human-provided demonstrations as higher level guidance. In effect, the demonstrations tell the robot *what* to do, while the learned model tells it *how* to do it, combining high-level human direction with a learned model of low-level dynamics. Figure 1 shows an overview of our system.

2 Method

We use a Baxter robot for all experiments described in the paper. The robot interacts with a rope placed on a table in front of it using only one arm. The arm has a gripper attached with two degrees of freedom (one rotational and one for closing/opening the two fingers). One end of the rope is tied to a clamp attached to the table. The robot receives visual inputs from the RGB channel of a Kinect camera. The interaction of the robot with the rope is constrained to a single action primitive consisting of two sub-actions - *pick* the rope at location (x_1, y_1) and *drop* the the rope at location (x_2, y_2), where (x_1, y_1, x_2, y_2) are pixel coordinates in the input RGB image. It is possible to manipulate the rope into many complex configurations using just this action primitive.

The robot collects data in a self-supervised manner by randomly choosing pairs of *pick* and *drop* points in the image. If we randomly choose a point on the image, then most points will not be on the rope and the data collection will be inefficient. Instead, we use the point cloud from the Kinect camera to segment the rope and then choose a *pick* point uniformly at random from this segment.

*Department of Electrical Engineering and Computer Science, University of California, Berkeley

29th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

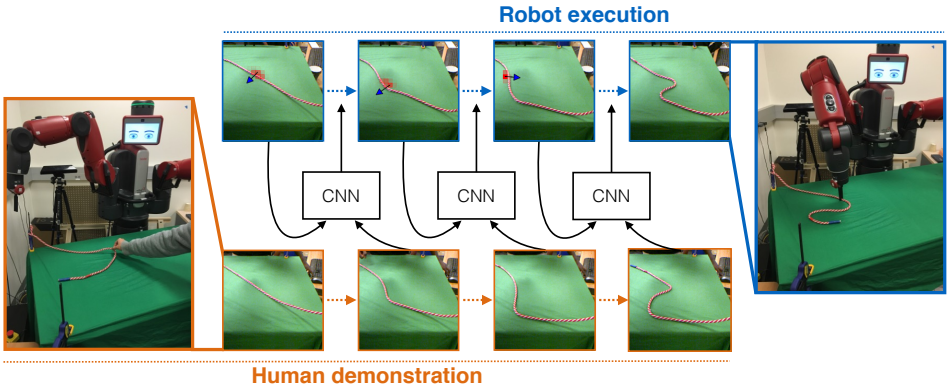

Figure 1: We present a system where the robot is capable of manipulating a rope into target configurations by combining a high-level plan provided by a human with a learned low-level model of rope manipulation. A human provides the robot with a sequence of images recorded while he manipulates the rope from an initial to goal configuration. The robot uses a learnt inverse dynamics model to execute actions to follow the demonstrated trajectory. The robot uses a convolutional neural network (CNN) for learning the inverse model in a self-supervised manner using 30K interactions with the rope with no human supervision. The red heatmap on each image of the robot's execution trace shows the predicted location of the *pick* action and the blue arrow shows the direction of the action. This image is best seen in color.

Once the *pick* point is chosen, the *drop* point can be obtained as a displacement vector from the *pick* point. We represent this displacement vector by the angle $\theta \in [0, 2\pi)$ and length $l \in [1, 15]$ cm. Values of θ and l are uniformly and randomly sampled from their respective ranges to obtain the *drop* point. After choosing the points, the robot executes the following steps: (1) grasp the rope at the *pick* point, (2) move the arm 5 cm vertically above the pick point, (3) move the arm to a point 5 cm vertically above the *drop* point, (4) release the rope by opening the gripper.

Our goal is to have the robot watch a human manipulate a rope and then reproduce this manipulation on its own. The human provides a demonstration in the form of a sequence of images of the rope in intermediate states toward a final goal state. Let V = $\{I_t | t = 1..T\}$ represent this sequence. The task of the robot is to execute a series of actions for transforming I_1 into I_2, then I_2 into I_3 and so on until the end of the sequence. A model that predicts the action that relates a pair of input states is called an inverse dynamics model and is mathematically described in equation 1 below

$$I_{t+1} = F(I_t, u_t), \tag{1}$$

where I_t, I_{t+1} are visual observations of the current and next states and u_t is the action. Following the works of [1, 9], we use deep convolutional neural networks to learn the inverse model.

Our neural network architecture shown in Figure 2 consists of two streams that transform each of the two input images into a latent feature space, x. The architecture of these streams is similar to AlexNet, and the weights of the streams are tied. The latent representations of the two images, (x_t, x_{t+1}), are concatenated with each other and fed into a networks of two fully connected layers of 200 units to predict the action. For the purpose of training, we turn action prediction into a classification problem by discretizing the action space. The action is parameterized as a tuple (p_t, θ_t, l_t), where p_t is the action location, θ_t is the action direction and l_t is the action length. Each dimension of this tuple is independently discretized. The action location is discretized onto a 20×20 spatial grid, and the direction and length are discretized into 36 and 10 bins respectively.

For transforming the rope from a starting configuration into a desired configuration, the robot receives as input the sequence of images depicting each stage of the manipulation performed by a human demonstrator to achieve the same desired rope configuration. Let the sequence of images received by the robot as inputs be, $V = \{I_t | t \in (1, T)\}$. The initial configuration I_1, I_T are images of the rope in the initial and goal configurations. The robot first inputs the pair of images (I_1, I_2) into the learnt inverse model and executes the predicted action. Let $\hat{I}_2$ be the visual state of the world after the action is executed. The robot, then inputs $(\hat{I}_2, I_3)$ in the inverse model and executes the output action. This process is repeated iteratively for T time steps. In some cases the robot does not predict

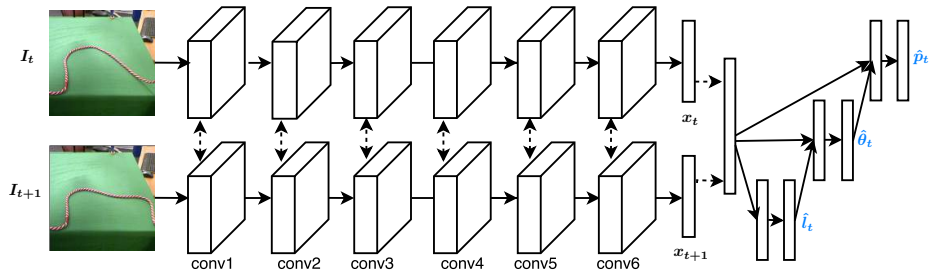

Figure 2: We use a convolutional neural network (CNN) to build the inverse dynamics model. The input to the CNN is a pair of images (I_t, I_{t+1}) and the output is the action that can transform the rope configuration in I_t into the configuration in I_{t+1}. The action is parameterized as a tuple (p_t, θ_t, l_t), where p_t, θ_t, l_t is the action location, direction and length respectively. $\hat{p}_t, \hat{\theta}_t, \hat{l}_t$ denote the predictions.

the poke location on the rope. For these cases we use the rope segmentation information to find the point on the rope that is closest to predicted pick location, to execute the pick primitve.

3 Evaluation

We evaluate the performance of the robot by measuring the distance between the rope configurations in the sequence of images provided as the demonstration and the sequence of images achieved by the robot after executing the series of actions using the inverse dynamics model. The distance metric uses the segmented point cloud of the rope to measure the distance between two rope configurations using the thin plate spline robust point matching (TPS-RPM) method [3].

We compare the performance of our method against a hand-engineered baseline. In a fashion similar to the proposed method, our baseline method takes as input the sequence of images from the human demonstration. In order to predict the action that will transform the rope from the configuration in I_t into the configuration in I_{t+1}, we first segment the rope in both the images using point cloud data. We then use TPS-RPM to register the segmented point clouds. In the absence of a model of rope dynamics, an intuitive way to transform the rope into a target configuration is to *pick* the rope at the point with the largest deformation in the first image relative to the second and then *drop* the rope at the corresponding point in the second image. As the point with largest distance may be an outlier, we use the point at the 90th percentile of the deformation distances for the *pick* action. Figure 3 compares the performance of the proposed method against the baseline method. We show quantitative results for demonstration sequences of three different lengths. For each sequence, we report the mean distance across 10 different repeats (each repeat used the same demonstration sequence). The results in Figure 3 show that our method outperforms the baseline. The baseline method uses a heuristic strategy that assumes no knowledge of the dynamics model of the rope. The superior

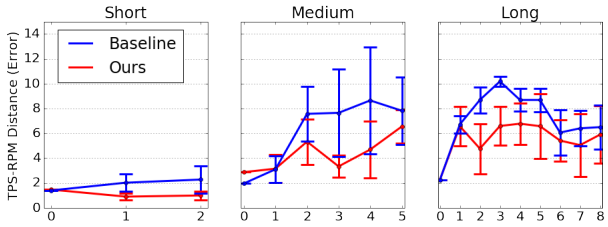

Figure 3: Comparison of the proposed method against a hand engineered baseline. The performance of each method was measured by computing the distance between the rope configurations in the sequence of images provided as the demonstration and that achieved by the robot. The figure shows the performance measured using TPS-RPM, and lower distance indicates better performance. We measured performance using sequences of short, medium and long lengths. Our method outperforms the baseline and has lower performance variance.

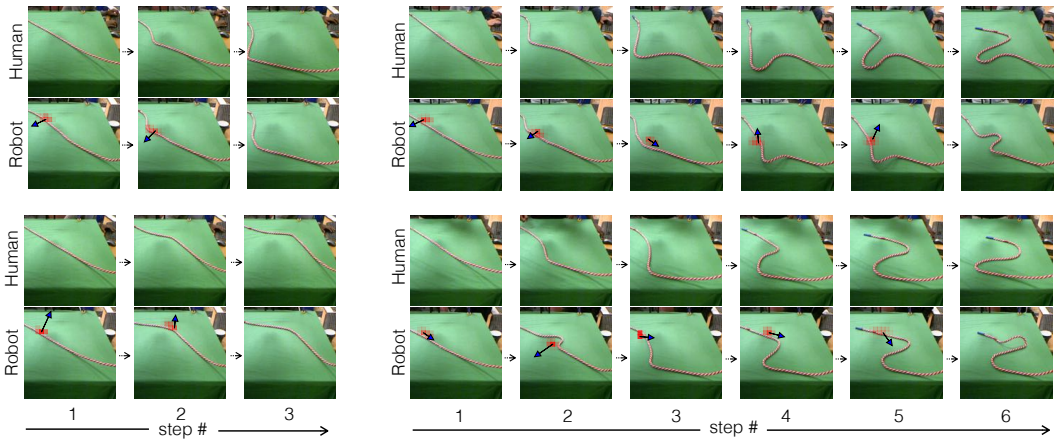

Figure 4: Each pair of rows in the figure show the result of the robot imitating a human demonstration. The first row in each pair, shows the the sequence of images provides as inputs to the robot through a demonstration and the second row in the pair shows the states achieved by the robot as it tries to follow the demonstrated trajectory. On the left are 3 step sequences and on the right are more difficult 6 step sequences. The robot fails to follow the demonstration in the top pair of the 6 step sequence whereas is it is quite sucessful in the bottom pair. The red heatmap on each image of the robot's execution trace shows the probability distribution over the predicted location of the *pick* action and the blue arrow shows the direction of the action.

performance of our method indicates that through the self-supervision phase, the robot has indeed learnt a meaningful model for rope manipulation. Some example sequences are shown in Figure 4. Videos of the self-supervised data collection, the demonstrations, and the autonomous executions are available at `https://ropemanipulation.github.io/`

References

[1] Pulkit Agrawal, Ashvin Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to poke by poking: Experiential learning of intuitive physics. *arXiv preprint arXiv:1606.07419*, 2016.

[2] Matthew Bell. *Flexible object manipulation*. PhD thesis, Dartmouth College, Hanover, New Hampshire, 2010.

[3] Haili Chui and Anand Rangarajan. A new algorithm for non-rigid point matching. In *CVPR*, 2000.

[4] John E Hopcroft, Joseph K Kearney, and Dean B Krafft. A case study of flexible object manipulation. *The International Journal of Robotics Research*, 10(1):41–50, 1991.

[5] Yasuo Kuniyoshi, Masayuki Inaba, and Hirochika Inoue. Learning by watching: Extracting reusable task knowledge from visual observation of human performance. *IEEE transactions on robotics and automation*, 10(6):799–822, 1994.

[6] Hermann Mayer, Faustino Gomez, Daan Wierstra, Istvan Nagy, Alois Knoll, and Jürgen Schmidhuber. A system for robotic heart surgery that learns to tie knots using recurrent neural networks. *Advanced Robotics*, 22(13-14):1521–1537, 2008.

[7] Stephen Miller, Mario Fritz, Trevor Darrell, and Pieter Abbeel. Parametrized shape models for clothing. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4861–4868. IEEE, 2011.

[8] Takuma Morita, Jun Takamatsu, Koichi Ogawara, Hiroshi Kimura, and Katsushi Ikeuchi. Knot planning from observation. In *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 3, pages 3887–3892. IEEE, 2003.

[9] Lerrel Pinto, Dhiraj Gandhi, Yuanfeng Han, Yong-Lae Park, and Abhinav Gupta. The curious robot: Learning visual representations via physical interactions. *arXiv preprint arXiv:1604.01360*, 2016.

[10] John Schulman, Ankush Gupta, Sibi Venkatesan, Mallory Tayson-Frederick, and Pieter Abbeel. A case study of trajectory transfer through non-rigid registration for a simplified suturing scenario. In *IROS*, 2013.

[11] Hidefumi Wakamatsu, Eiji Arai, and Shinichi Hirai. Knotting/unknotting manipulation of deformable linear objects. *The International Journal of Robotics Research*, 25(4):371–395, 2006.