Deep Visual Foresight for Planning Robot Motion

Chelsea Finn
UC Berkeley
cbfinn@eecs.berkeley.edu

Sergey Levine
Google Brain
slevine@google.com

Abstract

A key challenge in scaling up robot learning to many skills and environments is removing the need for human supervision, so that robots can collect their own data and improve their own performance without being limited by the cost of requesting human feedback. Model-based reinforcement learning holds the promise of enabling an agent to learn to predict the effects of its actions, which could provide flexible predictive models for a wide range of tasks and environments, without detailed human supervision. We develop a method for combining deep action-conditioned video prediction models with model-predictive control that uses entirely unlabeled training data. Our approach does not require a calibrated camera, an instrumented training set-up, nor precise sensing and actuation. Our results show that our method enables a real robot to perform nonprehensile manipulation – pushing objects – and can handle novel objects not seen during training.

1 Introduction

Most standard robotic manipulation systems consist of a series of modular components for perception and prediction that can be used to plan actions for handling objects. Imagine that a robot needs to push a cup of coffee across the table to give it to a human. This task might involve segmenting an observed point cloud into objects, fitting a 3D model to each object segment, executing a physics simulator using the estimated physical properties of the cup and, finally, choosing the actions which move the cup to the desired location. However, when robots encounter previously unseen objects in complex, unstructured environments, this pipelined model-based approach can break down when any one stage has a sufficiently large modeling error. Motivated by such learning approaches and large-scale, unsupervised robotic learning, we consider the question of whether it is possible to replace the hand-engineered robotic manipulation pipeline with a single general-purpose, learned model that connects low-level perception with physical prediction.

The primary contribution of our paper is to demonstrate that deep predictive models of video can be used by real physical robotic systems to manipulate previously unseen objects from high-level goals. We apply a deep predictive model demonstrated on video prediction in prior work [3], and show how this model's ability to learn implicit stochastic pixel flow can be leveraged for planning. To evaluate the feasibility of robotic manipulation with learned video prediction models, we restrict the problem setting to nonprehensile pushing tasks. By planning through the model in real time, we show that it is possible to learn nonprehensile manipulation skills with fast visual feedback control in an entirely self-supervised setting, using only unlabeled training data and without 3D models, depth observations, or a physics simulator. Although the pushing tasks are relatively simple, the performance of our method suggests that the underlying predictive model learns a sufficiently detailed physical model of the world for basic manipulation. To the best of our knowledge, our work is the first instance of robotic manipulation using learned predictive video models with generalization to new, previously unseen objects.

2 Planning with Learned Video Prediction Models

As we discuss next, our system receives a high level goal from the user and then uses model-predictive control (MPC) with our deep video prediction model to choose controls that will realize the user's commanded goal. Because video prediction models do not explicitly model objects, we cannot represent the task goal using standard notions of object poses.

2.1 Background

To perform visual model-predictive control, we use a deep predictive model of images trained on a large dataset of robotic pushing experience. Our dataset includes 59,000 pushing attempts collected using 10 7-DoF arms, involving hundreds of objects.[1] The inputs to the predictive model $\mathcal{M}$ at each timestep consist of the current and previous images, which we will denote $I_{0:1}$, the current and previous end-effector poses $\mathbf{x}_{0:1}$, and a sequence of future commands $\mathbf{a}_{1:H_p}$, where H_p is the horizon used during training. The model is trained to predict a distribution over the sequence of future image frames $I_{2:H_p+1}$ that result from executing the actions $\mathbf{a}_{1:H_p}$. We train the model with maximum likelihood, resulting in a mean squared error objective.

Following our prior work [3], the model uses a convolutional LSTM to predict future image sequences. As an intermediate step, the model outputs a probabilistic flow map at each time step t. Since our flow operator is linear, the distribution over a subsequent image can be obtained by transforming the mean of the distribution over the current image. The predicted image mean $\hat{I}_{t+1}$ is fed back into the network recursively to generate the next flow and image in the sequence. Note that no explicit flow supervision is provided to the network: training uses only raw images for supervision. The stochastic flow maps are an implicit, emergent property of the model, but one that will prove useful for planning.

2.2 Specifying Goals with Pixel Motion

To plan with a predictive model of images, we need a way to specify task objectives that can be automatically evaluated for each of the model's predicted visual futures. In this work, the goal is specified by the user in terms of pixel motion. The user selects one or more desired source pixels in the initial image, which we will denote $d_0^{(1)}, \ldots, d_0^{(P)}$, with each pixel's position given by $(x_d^{(i)}, y_d^{(i)})$, and then specifies a corresponding goal position for each of those pixels, which we will denote $g^{(1)}, \ldots, g^{(P)}$, with each goal position given by $(x_g^{(i)}, y_g^{(i)})$. With this goal specification, the robot can plan to move the objects for which the selected pixels belong. This goal representation is easy to specify, does not require instrumentation of the environment e.g. via motion capture or AR markers, and can represent a variety of object manipulation tasks. Unlike most approaches to robotic manipulation, it also does not use or need an explicit representation of objects.

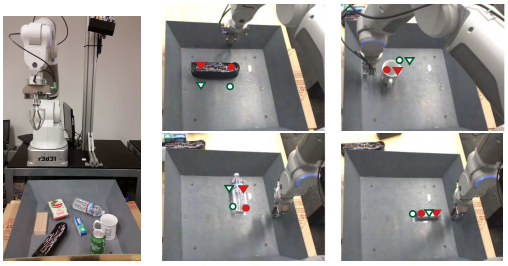
2.3 Evaluating Actions with Implicit Pixel Advection

Here, we describe how our method evaluates a proposed action sequence using probabilistic inference under the learned predictive model $\mathcal{M}$. Consider the goal of moving a single designated pixel $d_0 = (x_d, y_d)$ in the initial image to a goal position $g = (x_g, y_g)$. As described in Section 2.1, our video prediction model predicts stochastic flow operators at each time step, which can be used to transform prior images or independent Gaussian pixel distributions into independent Gaussian pixel distributions for the next image. We can also use these stochastic flow operators to predict how individual pixels will move, conditioned on a sequence of actions. Since the flow operators are stochastic, they provide a distribution over pixel motion conditioned on the actions. This suggests a natural approach to planning using maximum likelihood: determine the sequence of actions that maximizes the probability of the designated pixel moving to the goal position. To do so, we recursively apply the predicted flow to the probability distribution over the pixel's location to obtain a probability distribution over the pixel's location after H timesteps.

2.4 Sampling-Based Model-Predictive Control with Deep Models

The goal of our MPC-based controller is to determine the sequence of actions that maximizes the probability of the designated pixels being moved to their corresponding goal locations. In order to use this evaluation to choose the best actions, we perform an optimization over a short horizon of future actions, using the cross-entropy method (CEM) [4]. At each time step t, we sample M

[1] We have made the dataset available at `http://sites.google.com/site/brainrobotdata`



method	mean pixel distance
initial pixel position	5.10 ± 2.25
1) random actions	4.05 ± 1.75
2) move end-effector to goal	3.79 ± 2.66
3) move end-effector along vector (with replanning)	3.19 ± 1.68
visual MPC (ours)	$\mathbf{2.52 \pm 1.06}$

Figure 1: Left: The experimental setup, including the 7 test objects not in the training set. Middle: 4 of the 10 pushing tasks in the quantitative evaluation. The red and green-outlined markers indicate the start and goal pixel locations. Right: The quantitative evaluation, showing mean distance between final and goal pixel positions.

action sequences of length H, $\{\mathbf{a}_t^{(m)}, \ldots, \mathbf{a}_{t+H}^{(m)}\}$, and compute the probabilities of success for each one, denoted by $\mathbf{P}_{t+H}^{(m)}(g) = P(s_{t+H} = g | I_{t-1:t}, \mathbf{x}_{t-1:t}, \mathbf{a}_{t:t+H}^{(m)}, d_t)$. We then select the K action sequences with the highest values of $\mathbf{P}_{t+H}^{(m)}(g)$, fit a multivariate Gaussian distribution to these K selected action sequences, and resample a new set of M action sequences from this distribution. The new set of action sequences improves on the previous set, and the resampling and refitting process is repeated for J_t iterations. This corresponds to the CEM stochastic optimization algorithm. At the end of the last iteration, we take the sampled action sequence $\mathbf{a}_t^*, \ldots, \mathbf{a}_{t+H}^*$ that is most likely to be successful, and execute $\mathbf{a}_t^*$ on the robot. We use $J_t = 4$ iterations, $M = 40$ samples per iteration, and $K = 10$ samples during the initial planning phase ($t = 0$) and, when replanning in real-time ($t > 0$), we take $J_t = 1$ iteration of $M = 20$ samples. Note that each batch of M samples corresponds to a forward pass through deep recurrent network with a batch size of M, and therefore can be trivially parallelized.

3 Experiments

In our experimental evaluation, we aim to answer the following questions: (1) Can we use action-conditioned video prediction models to manipulate novel objects that were not previously seen during training? (2) Can video prediction models trained entirely on raw image pixels make meaningful and nontrivial inferences about the behavior of physical objects? We aim to answer question (1) by evaluating our method on new objects not seen during training, and we answer question (2) through comparison to baseline methods and through qualitative experiments aimed to construct physically nuanced pushing scenarios that require reasoning about rotations and centers of mass.

3.1 Experimental Setup

The experimental set-up is shown in Figure 1. The pushing task consists of episodes of length $T = 15$ where the goal is to move P pixels from their current locations $\{(x_s^{(i)}, y_s^{(i)})\}$ to corresponding goal locations $\{(x_g^{(i)}, y_g^{(i)})\}$. Unless otherwise specified, all objects in the experiments were not seen previously in the training set. A video of our experiments is available online.[2]

We use 64×64 RGB images and a planning horizon $H = 3$, corresponding to about 800 ms. This allows us to replan controls in real time every 200 ms. Because of the short time horizon, we largely consider pushing tasks that involve fast, reactive control rather than long-term planning.

3.2 Quantitative Comparisons

We provide quantitative comparisons to three baselines, with the aim of evaluating whether or not our video prediction model has learned a meaningful and nontrivial notion of objects and physical interaction. Object identity, inertia, and contact dynamics are not provided to nor encoded in the model explicitly, but must be learned entirely from data. Correspondingly, our baselines do not use any knowledge about the objects in the scene, but are reasonably effective for the short horizon pushing tasks that we consider. The baselines are: (1) select actions randomly from a uniform distribution; (2) servo the gripper to the goal pixel position (x_g, y_g), using a known camera calibration; and (3) servo the gripper along the vector from the current pixel position $d_t = (x_t, y_t)$ to the goal (x_g, y_g), with continuous replanning based on the current pixel position estimated using optical flow.

The first baseline of randomly selecting actions serves to calibrate the difficulty of the task in choosing effective actions. The last two baselines serve as a comparison to our method to test whether our

[2] See `https://sites.google.com/site/robotforesight/`

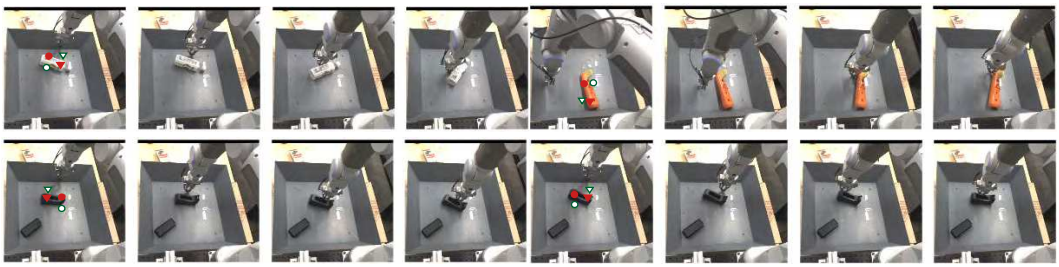


Figure 2: When commanded to rotate objects by moving their end-points in opposing directions, our method plans for actions that touch the object on one side. In the initial image on the left, the red and green-outlined markers indicate the user-specified start and goal pixel locations. The black tape in the bottom row was in the original dataset. The other objects were not.

model is learning something meaningful about physical object interaction, beyond simple motions of the arm. The final baseline and our method use an optical flow solver [1] to track the position of the pixel for replanning. The optical flow is used only during the replanning phase to provide the initial $P(s_t)$ distribution. The flow solver is also used to quantitatively evaluate the distance between the final position of the pixel and the goal position of the pixel.

Our results, shown in Figure 1, indicate that our method is indeed able to leverage the predictive model to improve over the performance of the baselines. The performance of our method compared to the last two baselines suggests that our method is making meaningful inferences about the motion of objects in response to the arm. Although these results leave significant room for improvement, they suggest that predictive models can be used with minimal prior knowledge to perform robotic manipulation tasks. As video prediction models continue to improve, we expect that the performance of our method will improve with them.

3.3 Qualitative Results

In this section, we evaluate the capabilities and limitations of visual MPC with a set of qualitative experiments. The goal of these experiments is to determine whether or not the model can perform more complex manipulations that require reasoning about object rotations and centers of mass. One of the benefits of planning actions with a predictive model of all pixels is that the model can be used to plan actions that affect multiple pixels, causing them to move in different directions in a coordinated fashion. To evaluate this capability, we command the robot to rotate objects by specifying opposing motions for pixels on either extreme of the object, as shown on the left in Figure 2. When commanded to move pixels such that an object rotates, our method is able to produce the desired rotational motion, as seen on the right in Figure 2. These examples indicate that the model can indeed be used for predicting the motion of multiple pixels and planning to affect them in a coordinated way.

4 Discussion & Future Work

We presented a learning-based approach for basic nonprehensile manipulation. Our method uses a deep predictive model to plan pushing motions with minimal prior engineering. We believe that this work represents a relatively early first step toward model-based robotic control using learned predictive models. Neural network-based video prediction is still in its early stages, with most state-of-the-art methods making accurate predictions only a few frames into the future [5]. As video prediction methods improves, model-based methods will become increasingly more powerful. Of particular interest for robotic manipulation, hierarchical models operating at varying time scales [2] may even make it practical to plan highly elaborate skills entirely using learned predictive models.

References

[1] R. Anderson, D. Gallup, J. T. Barron, J. Kontkanen, N. Snavely, C. Hernández, S. Agarwal, and S. M. Seitz. Jump: Virtual reality video. *SIGGRAPH Asia*, 2016.

[2] S. El Hihi and Y. Bengio. Hierarchical recurrent neural networks for long-term dependencies. In *Neural Information Processing Systems (NIPS)*, 1995.

[3] C. Finn, I. Goodfellow, and S. Levine. Unsupervised learning for physical interaction through video prediction. In *Neural Information Processing Systems (NIPS)*, 2016.

[4] R. Y. Rubinstein and D. P. Kroese. *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation and machine learning*. Springer Science & Business Media, 2013.

[5] T. Xue, J. Wu, K. L. Bouman, and W. T. Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. *Neural Information Processing Systems (NIPS)*, 2016.