Learning Dynamic Manipulation Skills under Unknown Dynamics with Guided Policy Search

Sergey Levine and Pieter Abbeel
Department of Electrical Engineering and Computer Sciences, UC Berkeley
E-mail: svlevine@eecs.berkeley.edu, pabbeel@cs.berkeley.edu

I. Introduction

Planning and trajectory optimization can readily be used for kinematic control of robotic manipulation. However, planning dynamic motor skills requires a detailed physical simulation, and some aspects of the task, such as contacts, are very difficult to simulate with enough accuracy for dynamic manipulation. Alternatively, manipulation skills can be learned from experience, allowing them to deftly exploit the dynamics of the real world. This is the approach taken in reinforcement learning [12, 14, 6], where a control policy is optimized using experience gathered directly with the robot. However, applying reinforcement learning to realistic robotics tasks typically requires a carefully engineered policy class with a modest number of parameters to make the learning task tractable [3].

Recently developed guided policy search methods can be used to learn general-purpose controllers represented by neural networks, without task-specific engineering, by using trajectory optimization to discover successful task executions [7, 8, 9]. These methods previously required a simulator in order to perform trajectory optimization, making them difficult to apply to robotic motor skill learning. We present a trajectory optimization algorithm suitable for use with guided policy search that does not require a known dynamics model or simulator. Our experimental evaluation shows that this approach can optimize manipulation trajectories that are extremely challenging for previous reinforcement learning methods. We also show that, combined with guided policy search, our method can learn complex policies for a simulated peg insertion task in a partially observed environment.

II. Trajectory Optimization under Unknown Dynamics

Our trajectory optimization algorithm is similar in form to DDP or iLQG [4, 15]. First, an LQR backward pass is used to compute a quadratic value function and a linear feedback controller based on a linearization of the dynamics around a nominal trajectory. The nominal trajectory is then updated by performing rollouts with this linear controller, and the LQR pass is repeated around this new trajectory until convergence.

Our method differs from iLQG and DDP in several ways. Instead of using a simulator, we estimate the linearized dynamics from data sampled from the real system during the rollout phase. We also construct a stochastic time-varying linear Gaussian controller, rather than a deterministic linear controller, which induces a probability distribution $q(\tau)$ over trajectories. This allows us to construct stochastic policies that change gradually on each iteration, avoiding discontinuous jumps, and provides a way to sample multiple trajectories during the rollout phase in order to estimate the dynamics. To ensure that the trajectory distribution changes gradually, we enforce a constraint that the new distribution should not differ too much from the old one. This constraint prevents large steps that can produce unstable trajectories when the true dynamics are nonlinear.

To construct these time-varying linear Gaussian controllers, we optimize a maximum entropy objective given by

$$\mathcal{L}(q(\tau)) = \sum_t E_{q(\tau)}[\ell(\mathbf{x}_t, \mathbf{u}_t)] - \mathcal{H}(q(\tau)), \qquad (1)$$

where $\mathcal{H}(q(\tau))$ is the differential entropy and $\ell(\mathbf{x}_t, \mathbf{u}_t)$ is the cost function. It can be shown that, under linearized dynamics, the conditional distributions $q(\mathbf{u}_t|\mathbf{x}_t)$ that minimize this objective are linear Gaussians, with the mean given by the feedback controller produced by LQR, and the covariance given by the inverse of the quadratic component of the Q-function (the curvature of the state-action cost-to-go function). Details of this derivation can be found in prior work [7].

Optimizing the objective in Equation 1 yields an improved trajectory distribution under the linearized dynamics. However, the new trajectory can be very far from the previous one, and the linearized dynamics may no longer be valid for it. This can lead to divergence. We limit the change in the trajectory distribution between iterations by imposing a constraint on the KL-divergence between the new distribution and the old one, resulting in the following optimization problem:

$$\min_{q(\tau)} \sum_t E_{q(\mathbf{x}_t, \mathbf{u}_t)}[\ell(\mathbf{x}_t, \mathbf{u}_t)] \text{ s.t. } D_{\mathrm{KL}}(q(\tau) \| \hat{q}(\tau)) \leq \epsilon,$$

where $\hat{q}(\tau)$ is the previous trajectory distribution. Using a dual variable η, we can write the Lagrangian of this problem as

$$\mathcal{L}(q(\tau), \eta) = \sum_t E_{q(\mathbf{x}_t, \mathbf{u}_t)}[\ell(\mathbf{x}_t, \mathbf{u}_t)] + \eta[D_{\mathrm{KL}}(q(\tau) \| \hat{q}(\tau)) - \epsilon]$$

$$= \left[\sum_t E_{q(\mathbf{x}_t, \mathbf{u}_t)}[\ell(\mathbf{x}_t, \mathbf{u}_t) - \eta \log \hat{q}(\mathbf{u}_t|\mathbf{x}_t)] - \eta \mathcal{H}(q(\tau))\right] - \eta\epsilon.$$

The constrained problem can then be solved by dual gradient descent [1], where we alternate between optimizing the Lagrangian with respect to $q(\tau)$, and updating the dual variable by incrementing it by a multiple of the current constraint violation. Noting that the Lagrangian has a similar form to

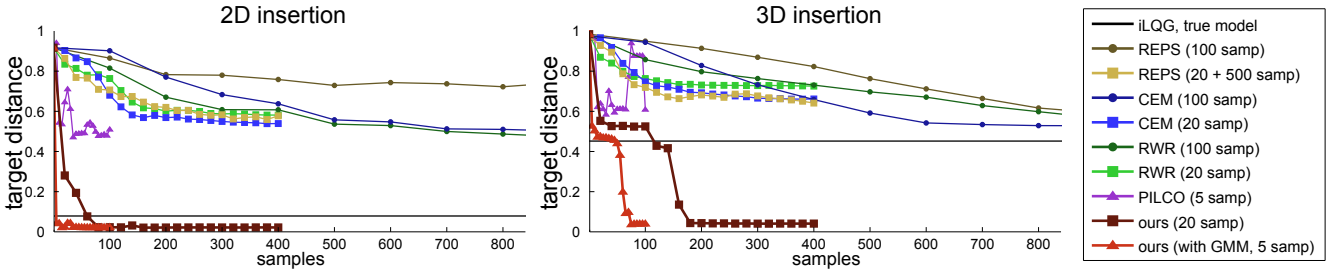

Fig. 1. Results for trajectory optimization on 2D and 3D insertion. Our approach uses fewer samples and finds better final solutions than prior methods, and the GMM further reduces the required sample count.

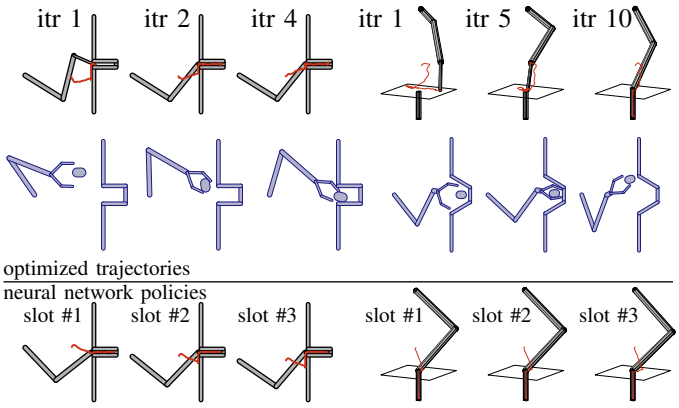

Fig. 2. End-effector traces (in red) at various iterations of trajectory optimization (top), dynamic grasping for moving a ball into and out of a basket (middle), and end-effector traces for a neural network policy that searches for the unknown hole position by sliding along the surface (bottom).

Equation 1, we can show that optimizing it with respect to $q(\tau)$ corresponds to optimizing the trajectory distribution using the previously described LQR algorithm with an augmented cost, given by

$$\tilde{\ell}(\mathbf{x}_t, \mathbf{u}_t) = \frac{1}{\eta}\ell(\mathbf{x}_t, \mathbf{u}_t) - \log \hat{q}(\mathbf{u}_t|\mathbf{x}_t).$$

In our approach, this optimization uses empirically estimated time-varying linear Gaussian dynamics obtained using samples from the previous trajectory distribution $\hat{q}(\tau)$, which can be obtained using rollouts on the real system. We can reduce the number of samples required to accurately fit the dynamics at each iteration by fitting a simple global dynamics model, and using this global model as a prior on the linear regression. Unlike in model-based RL, this global model need not itself be a good forward model, since it only provides a prior for the dynamics fit. We used a Gaussian mixture model (GMM) fitted to samples from all time steps at the current and previous iterations to generate this prior.

III. General Policy Search

To learn general parameterized policies, we combine our trajectory optimization algorithm with guided policy search, which trains a parameterized policy by alternating between optimizing a trajectory to minimize cost and match the policy, and optimizing the policy in supervised fashion to match the trajectory [7, 8, 9]. Using our trajectory optimization algorithm together with guided policy search has several advantages over standard model-free policy search methods. Previous work has shown that guided policy search algorithms can train much more complex policies with many more parameters than model-free methods, including complex locomotion controllers represented by large neural networks [7, 8, 9]. Furthermore, because our trajectory optimization algorithm exploits the structure of the conditional linear Gaussian controller to simplify the policy search task, it can optimize trajectories faster and with fewer samples than general-purpose model-free policy search algorithms, leading to an efficient policy search method that can learn complex, high-dimensional policies.

IV. Experimental Evaluation

Figure 1 presents a comparison between our algorithm and prior methods on a challenging simulated peg insertion task. We compare our approach to REPS, including a variant proposed by Lioutikov et al. that fits linear dynamics and generates 500 synthetic samples, denoted "REPS (20 + 500 samps)" [10], reward-weighted regression (RWR) [11, 5], the cross-entropy method (CEM) [13], and the model-based PILCO method [2]. The task resembles inserting a key into a hole, or assembling an object with screws or nails, and is performed either in 2D or 3D. The discontinuous dynamics induced by the contact forces make this a challenging trajectory optimization task. The graph indicates the distance between the end of the peg and the bottom of the 0.5-unit hole. While our method could optimize trajectories for both tasks, prior methods did not achieve distances below 0.5, indicating that they were unable to actually insert the peg into the hole.

In Figure 2, we show each system with end-effector traces at different iterations of our algorithm, along with snapshots from two dynamic grasping tasks, where the goal is to either place an object into a basket, or remove it from the basket. Unlike planned kinematic grasps, the motion is fluid and quick, exploiting the inertia of the arm and object.

To evaluate general parameterized policy learning with guided policy search, we trained neural network policies in 2D and 3D that can insert the peg into holes at various positions. The network was not provided the hole position as input, and had to learn to search for it within a radius of 0.2 units in 2D and 0.1 in 3D. To learn this behavior, the neural network policy was trained with four trajectories optimized for four different hole positions. Once trained, it was able to generalize to any hole position within the training radius. Some images with end-effector traces are shown in Figure 2. Note how the end-effector follows the wall until it finds the opening.

REFERENCES

[1] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, New York, NY, USA, 2004.

[2] M. Deisenroth and C. Rasmussen. PILCO: a model-based and data-efficient approach to policy search. In *International Conference on Machine Learning (ICML)*, 2011.

[3] 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.

[4] D. Jacobson and D. Mayne. *Differential Dynamic Programming*. Elsevier, 1970.

[5] J. Kober and J. Peters. Learning motor primitives for robotics. In *International Conference on Robotics and Automation (ICRA)*, 2009.

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

[7] S. Levine and V. Koltun. Guided policy search. In *International Conference on Machine Learning (ICML)*, 2013.

[8] S. Levine and V. Koltun. Variational policy search via trajectory optimization. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

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

[10] R. Lioutikov, A. Paraschos, G. Neumann, and J. Peters. Sample-based information-theoretic stochastic optimal control. In *International Conference on Robotics and Automation*, 2014.

[11] J. Peters and S. Schaal. Applying the episodic natural actor-critic architecture to motor primitive learning. In *European Symposium on Artificial Neural Networks (ESANN)*, 2007.

[12] J. Peters and S. Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008.

[13] R. Rubinstein and D. Kroese. *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation and Machine Learning*. Springer, 2004.

[14] E. Theodorou, J. Buchli, and S. Schaal. Reinforcement learning of motor skills in high dimensions. In *International Conference on Robotics and Automation (ICRA)*, 2010.

[15] E. Todorov and W. Li. A generalized iterative LQG method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *American Control Conference*, 2005.