Learning Locomotion Controllers with a Policy Iteration Algorithm

John Schulman, Sergey Levine, Pieter Abbeel
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, California
Email: {joschu,svlevine,pabbeel}@eecs.berkeley.edu

***Abstract*—Reinforcement learning and policy search methods can in principle solve a wide range of control tasks automatically. However, practical robotic applications of policy search typically require a carefully designed policy representation that is specific to each task [2]. For high dimensional robotics tasks where value function estimation is impractical, policy gradient methods usually achieve the best results [6]. However, these methods assume that the policy return is a smooth function of the parameters. Since locomotion is inherently a hybrid task that combines both smooth and discontinuous dynamics [3], such methods are difficult to apply to locomotion without a carefully engineered policy parameterization that subsumes the nonsmooth aspects of the problem.**

We present a policy optimization method that can effectively handle discontinuous dynamics and learn locomotion controllers represented by general-purpose function approximators such as neural networks. Our method combines policy iteration with a trust region method that limits the change in the policy at each iteration. The algorithm can be shown to monotonically improve the policy and converge to a local optimum without assumptions about the smoothness of the objective landscape. Preliminary empirical results suggest that it can learn effective planar swimming, hopping, and walking gaits in simulation. Since the algorithm produces a neural network that directly maps system state to joint torques, it is well suited for real-time control and does not require task-specific policy classes or features.

I. Policy Iteration for Locomotion

General-purpose controller optimization has previously been addressed with direct policy search methods [2] and approximate dynamic programming techniques, such as policy iteration [1]. In direct policy search, samples from the current controller are used to estimate the gradient of the return with respect to the controller parameters. While such methods have achieved impressive results on real robotic systems, they assume that the return is a smooth function of the parameters, and typically require a compact, low-dimensional policy representation [2]. Approximate dynamic programming does not assume smoothness, but such methods are difficult to apply to high-dimensional, continuous dynamical systems.

A common limitation of approximate dynamic programming methods is the need to maintain an estimate of the value function. This is problematic, because the value function is often nonsmooth and has a large dynamic range, requiring a complex function approximator. Furthermore, a value function that has low Bellman error is not necessarily accurate [5], and a good RMS fit to the value function does not necessarily result in a good controller [9].

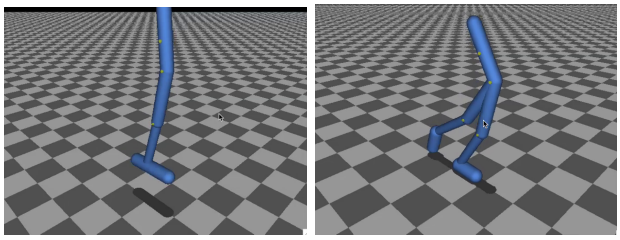

Fig. 1. Neural network controllers for hopping and bipedal walking (in 2D) learned using our method. Simulated using MuJoCo [8].

We avoid fitting a function approximator to the value function by using simulation rollouts to estimate the value of a few randomly chosen actions at each visited state. We then fit a parameteric representation of the policy to these value function estimates. For many high-dimensional problems, representing a policy is much easier than representing the value function.

Another critical component of our approach is an explicit bound on the change in the policy at each iteration, to ensure that the new policy does not visit drastically different states where we did not collect samples. This is similar to several recent policy search methods [6, 7]. The difference between these methods and ours is that the policy is fitted to samples of the value function collected for a number of different actions at each visited state, which effectively mitigates many of the problems caused by discontinuous objective landscapes, since the additional action samples serve to "preview" the outcome of taking a different action in each state.

A brief summary of our method is as follows. We use a stochastic policy, which maps each state to a probability distribution over the action space. At each iteration, our method samples states from the current policy by performing rollouts. At each state along these primary rollouts, we sample additional actions and perform branch rollouts to estimate the action's value. These branch rollouts allow us to form a loss function which locally approximates the expected returns of the policy. This loss function is then minimized subject to a constraint that the change in the action distribution compared to the previous policy is small. It can be shown that for a sufficiently small step, the performance of the policy improves

with high probability, using an argument similar to the one presented by Kakade and Langford [4], without assumptions about the smoothness of the objective landscape.

Images of preliminary locomotion policies for hopping and bipedal walking for simulated 2D robots are shown in Figure 1.

REFERENCES

[1] Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 2. Athena Scientific, 2012.

[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.

[3] J. W. Grizzle, C. Chevallereau, A. D. Ames, and R. W. Sinnet. 3d bipedal robotic walking: Models, feedback control, and open problems. In *Proceedings of the IFAC Symposium on Nonlinear Control Systems*, 2010.

[4] Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *ICML*, volume 2, pages 267–274, 2002.

[5] Remi Munos, Leemon C Baird, and Andrew W Moore. Gradient Descent Approaches to Neural-Net- Based Solutions of the Hamilton-Jacobi-Bellman Equation Gradient Descent Approaches to Neural-Net-Based Solutions of the Hamilton-Jacobi-Bellman Equation. 1999.

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

[7] J. Peters, K. Mülling, and Y. Altün. Relative entropy policy search. In *AAAI Conference on Artificial Intelligence*, 2010.

[8] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 5026–5033. IEEE, 2012.

[9] Mingyuan Zhong, Mikala Johnson, Yuval Tassa, Tom Erez, and Emanuel Todorov. Value function approximation and model predictive control. In *Adaptive Dynamic Programming And Reinforcement Learning (ADPRL), 2013 IEEE Symposium on*, pages 100–107. IEEE, 2013.