Deep Optimal Control: Using the Euler-Lagrange Equation to learn an Optimal Feedback Control Law

Michael Lutter
Intelligent Autonomous Systems
TU Darmstadt
64289 Darmstadt, Germany
`lutter@ias.tu-darmstadt.de`

Jan Peters [*]
Intelligent Autonomous Systems
TU Darmstadt
64289 Darmstadt, Germany
`mail@jan-peters.net`

Abstract

Learning optimal policies describing a feedback control law capable of executing optimal trajectories is essential for many robotic applications. Such policies can either be learned using reinforcement learning or planned using optimal control. While reinforcement learning is sample inefficient, optimal control can only plan a single optimal trajectory from a specific starting configuration. To overcome these shortcomings, the planned trajectories are frequently augmented with a tracking controller, that has no optimality guarantee, and are constantly replanned to adapt the optimal trajectory to tracking errors

In this paper we propose a new algorithm that is based on optimal control principles but learns an optimal policy rather than a single trajectory. Using the special case of the Pontryagin's principle stating optimality as the Euler-Lagrange equation, we can learn an optimal policy by embedding a policy network inside the Euler-Lagrange equation and minimizing the error of this equality. This approach is inspired by our previous work on Deep Lagrangian Networks that showed that minimizing the error of the Euler-Lagrange equation can be used to learn the system dynamics unsupervised. Our proposed approach enables us to learn an optimal policy describing a feedback control law in continuous time given a differentiable cost function. In contrast to existing optimal control approaches, the policy can generate an optimal trajectory from any point in state-space without the need of replanning. The resulting approach is currently evaluated on a robotic planning task requiring significant adaption to dynamic changes.

Keywords: Optimal Control, Learning for Control, Deep Learning

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No #640554 (SKILLS4ROBOTS). Furthermore, this research was also supported by grants from ABB AG, NVIDIA and the NVIDIA DGX Station.

[*]Max Planck Institute for Intelligent Systems, Spemannstr. 41, 72076 Tübingen, Germany.

1 Extended Abstract

1.1 Introduction

Reinforcement Learning (RL) provides an methodology to learn a policy, i.e., a mapping from system state to action, maximizing a scalar reward by solely interacting with a black-box environment through specified actions and the system state. Using this approach many control tasks such as the robot locomotion [1, 2], robot manipulation [3, 4], control of under-actuated system [5, 6, 7] and games [8, 9] have been successfully addressed. However, these methods usually require millions of sample rendering the learning on the physical systems unrealistic [4], require reward shaping to achieve fast learning [10] and the achieved performance is not only dependent on hyperparameters but also the random seed [11]. In contrast to reinforcement learning, optimal control (OC) plans a trajectory of the system states or actions to maximize a scalar reward function given the true environment model. The resulting solution is an open-loop control sequence and must be augmented with a tracking controller and re-planning to be applied to a noisy physical systems. While the tracking controller tracks the currently known control sequence, the re-planning updates the optimal sequence w.r.t. the current tracking error. Similar to reinforcement learning, these approaches have also been applied to robot locomotion [12, 13], robot manipulation [14] and control of under-actuated systems [15]. In addition, most RL and OC algorithms use discrete time rather than continuous time. For RL the most notable exception for continuous time is the work of Kenji Doya [16]. For OC the discretization of time is especially important as the resulting artifacts might yield solutions violating hard constraints.

In this paper we propose Deep Optimal Control, a continuous time optimal control algorithm which similar to reinforcement learning learns an optimal policy rather than an single optimal trajectory. Deep optimal control is off-policy and the resulting policies describe a feedback control law and do not require any replanning. These properties are achieved by using the special case of Pontryagin's principle stating optimality of any trajectory using the Euler-Lagrange differential equation and learning a policy-network that fulfills these equations on the state-space. More concretely, the policy network and the corresponding derivatives are embedded in the differential equation and the error on the differential equation is minimized using samples. Such learning is possible due to the fully differentiability of deep networks and has been demonstrated in our previous research on Deep Lagrangian Networks [17]. There we showed that the Euler-Lagrange differential equation can be used as training objective to achieve unsupervised learning of the system dynamics and the system energies.

In the following we briefly formulate the specific OC problem (Section 1.2) and propose our Deep Optimal Control approach in section 1.3. Subsequently, we introduce our current evaluation approach in section 1.4.

1.2 Problem Formulation

Let J be the reward function and π be the policy. Then the optimal policy π^* maximizing the reward for the time horizon T and fixed starting point is described by

$$\pi^* = \arg\max_{\pi} \; J(\mathbf{q}) = \arg\min_{\pi} \int_0^T r(\mathbf{q}, \dot{\mathbf{q}}, t)\,\mathrm{d}t \qquad \text{with} \quad \pi(\mathbf{q}, t) = \dot{\mathbf{q}}, \;\; \mathbf{q}(0) = \mathbf{q}_0, \;\; \mathbf{q} \in \mathcal{S} \subseteq \mathbb{R}^n \;\; \mathbf{q} \in \mathcal{U} \subseteq \mathbb{R}^n \tag{1}$$

where $\mathbf{q}$, $\dot{\mathbf{q}}$ describe the system configuration and r is the scalar reward at time t. This definition simultaneously implies that the trajectory $\mathbf{q}(t)$ and the corresponding acceleration $\ddot{\mathbf{q}}(t)$ is described by

$$\mathbf{q}(t) = \mathbf{q}_0 + \int_0^t \pi(\mathbf{q}(u), u)\,\mathrm{d}u, \qquad \ddot{\mathbf{q}}(t) = \frac{d\pi}{dt} = \frac{\partial \pi}{\partial \mathbf{q}}^T \frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \pi}{\partial t} = \frac{\partial \pi}{\partial \mathbf{q}}^T \pi + \frac{\partial \pi}{\partial t}. \tag{2}$$

Furthermore, $\ddot{\mathbf{q}} := \mathbf{u}$ defines the desired change of the dynamics and hence, a feedback-control law. The optimization problem of Equation 1 is a special case of the Pontryagin's Principle and one can derive a necessary optimality criteria for all trajectories, that are at least twice differentiable, i.e., $\mathbf{q} \in C^2$ [18]. The necessary condition can be derived using calculus of variations by perturbing the locally optimal trajectory, i.e., $\mathbf{q} = \mathbf{q}^* + \delta\mathbf{q}$, and setting the change in cost δJ to zero. Following this derivation shows that all optimal trajectories must fulfill the Euler-Lagrange equation and the boundary conditions described by

$$\frac{\partial L^*(t)}{\partial \mathbf{q}} - \frac{d}{dt}\frac{\partial L^*(t)}{\partial \dot{\mathbf{q}}} = 0 \quad \forall \;\; 0 < t < T \tag{3}$$

$$\frac{\partial L^*(t)}{\partial \dot{\mathbf{q}}}^T \delta\mathbf{q}(t) = 0 \quad \text{for} \;\; t \in \{0, T\} \tag{4}$$

$$\left(L^*(T) - \frac{\partial L^*(T)}{\partial \dot{\mathbf{q}}}^T \dot{\mathbf{q}}^*(t)\right)\delta t = 0 \quad \text{for} \;\; t = T \tag{5}$$

where $L^*(t) = r(\mathbf{q}^*(t), \dot{\mathbf{q}}^*(t), t)$ is the Lagrangian of the optimal trajectory $\mathbf{q}^*(t)$, $\dot{\mathbf{q}}^*(t)$. For the stated problem only the boundary condition at $t = T$ is relevant, because the initial value $\mathbf{q}_0$ guarantees that $\delta\mathbf{q}(0) = 0$. Furthermore, we will only consider the infinite time horizon, i.e. $T = \infty$, and discounted rewards, i.e., all reward functions r must be of form

$$r(\mathbf{q}, \dot{\mathbf{q}}, t) = \exp(-\lambda t)\,\tilde{r}(\mathbf{q}, \dot{\mathbf{q}}) \tag{6}$$

where λ is a positive discounting constant. The discounted reward formulation guarantees that the integral of Equation 1 is defined for the infinite time horizon. Substituting Equation 6 into the boundary conditions (Equation 4 & Equation 5) and taking the limit yields

$$\lim_{T\to\infty} \frac{\partial L(T)}{\partial \dot{\mathbf{q}}}^T \delta\mathbf{q}(t) = \lim_{T\to\infty} \exp(-\lambda T) \left(\frac{\partial \tilde{r}}{\partial \dot{\mathbf{q}}}^T \delta\mathbf{q}(t)\right) \qquad = 0 \tag{7}$$

$$\lim_{T\to\infty} \left(L(T) - \frac{\partial L(T)}{\partial \dot{\mathbf{q}}}^T \dot{\mathbf{q}}(t)\right)\delta t = \lim_{T\to\infty} \exp(-\lambda T) \left(\tilde{r} - \frac{\partial \tilde{r}}{\partial \dot{\mathbf{q}}}^T \dot{\mathbf{q}}(t)\right)\delta t = 0. \tag{8}$$

These equations show that the boundary constraints are always fulfilled for the discounted reward and infinite time horizon problem. Therefore, the optimal policy must only adhere to the Euler-Lagrange differential equation described by Equation 3 and is time-invariant due to the infinite horizon, i.e., $\pi^* = \pi(\mathbf{q})$.

1.3 Deep Optimal Control

Optimal control would use a differential equation solver to solve for the optimal trajectory using numerical optimization leading to a single trajectory. Instead of solving this differential equation, we propose Deep Optimal Control to learn a policy that fulfills the necessary optimality condition in continuous time. Therefore, we embed a policy network $\pi(\mathbf{q};\ \theta)$ inside the differential equation and minimize the error of the equality. More concretely, let r be a known and differentiable reward function, than the Euler-Lagrange equation is expressed by

$$\frac{\partial r}{\partial \mathbf{q}} - \left(\frac{\partial}{\partial \mathbf{q}}\frac{\partial r}{\partial \dot{\mathbf{q}}}\right)^T \dot{\mathbf{q}} - \left(\frac{\partial}{\partial \dot{\mathbf{q}}}\frac{\partial r}{\partial \dot{\mathbf{q}}}\right)^T \ddot{\mathbf{q}} - \frac{\partial}{\partial t}\frac{\partial r}{\partial \dot{\mathbf{q}}} = \mathbf{0}, \tag{9}$$

$$\exp(-\lambda t)\left(\frac{\partial \tilde{r}}{\partial \mathbf{q}} - \left(\frac{\partial}{\partial \mathbf{q}}\frac{\partial \tilde{r}}{\partial \dot{\mathbf{q}}}\right)^T \dot{\mathbf{q}} - \left(\frac{\partial}{\partial \dot{\mathbf{q}}}\frac{\partial \tilde{r}}{\partial \dot{\mathbf{q}}}\right)^T \ddot{\mathbf{q}} + \lambda \frac{\partial \tilde{r}}{\partial \dot{\mathbf{q}}}\right) = \mathbf{0}. \tag{10}$$

It is important to note that $\tilde{r}$ is independent of time, and hence, this equality depends only on system configuration $\mathbf{q}$, $\dot{\mathbf{q}}$ and the discounting factor λ but not on time for $t \ll \infty$. The dependence on λ is important as λ tunes the short- or farsightedness of the optimal policy. Substituting $\dot{\mathbf{q}} = \pi$ and $\ddot{\mathbf{q}} = (\partial\pi/\partial\mathbf{q})^T \pi$ and reformulating this equality as cost function yields the optimization problem for learning the optimal parameters θ^* of the policy. The resulting optimization problem is described by

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{N} \left\| \frac{\partial \tilde{r}_i}{\partial \mathbf{q}} - \left(\frac{\partial}{\partial \mathbf{q}}\frac{\partial \tilde{r}_i}{\partial \pi}\right)^T \pi - \left(\frac{\partial}{\partial \pi}\frac{\partial r_i}{\partial \pi}\right)^T \frac{\partial \pi}{\partial \mathbf{q}}^T \pi + \lambda \frac{\partial \tilde{r}_i}{\partial \pi} \right\|_2^2 \tag{11}$$

where $\tilde{r}_i = \tilde{r}(\mathbf{q}_i,\ \pi(\mathbf{q}_i;\theta))$ and $\|.\|_2$ is the l_2-norm. Due to the fully differentiability of the neural network $\partial\pi/\partial\mathbf{q}$ can be computed in closed form and machine precision [19]. As no component within the objective is dependent on time, the optimization problem can be solved by uniformly sampling $\mathbf{q}_i$ in the state domain $\mathcal{S}$. The samples must not originate from the current policy as the optimal control formulation is off-policy. Despite that the optimization objective is quite different from standard optimization losses our previous work on Deep Lagrangian Networks showed that such objective can be used for training a deep network.

1.4 Evaluation

Currently we are evaluating this approach on a simple toy task and a robotic planning experiment. For the toy task, Deep Optimal Control is applied to a simple 2d navigation experiment in continuous time. Within this environment the agent can choose between to equally important goals. Depending on the starting position, one goal is better than the other. However, during execution minimal tracking errors due to transition noise can significantly change the optimal trajectory. Therefore, standard optimal control algorithms cannot solve this problem without replanning while the policy learned using Deep Optimal Control should yield optimal performance.

References

[1] J. Z. Kolter and A. Y. Ng, "Policy search via the signed derivative." in *Robotics: science and systems*, 2009, p. 34.

[2] N. Heess, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. Eslami, M. Riedmiller *et al.*, "Emergence of locomotion behaviours in rich environments," *arXiv preprint arXiv:1707.02286*, 2017.

[3] J. Kober and J. R. Peters, "Policy search for motor primitives in robotics," in *Advances in neural information processing systems*, 2009, pp. 849–856.

[4] M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray *et al.*, "Learning dexterous in-hand manipulation," *arXiv preprint arXiv:1808.00177*, 2018.

[5] M. Deisenroth and C. E. Rasmussen, “Pilco: A model-based and data-efficient approach to policy search,” in *Proceedings of the 28th International Conference on machine learning (ICML-11)*, 2011, pp. 465–472.

[6] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning,” *arXiv preprint arXiv:1509.02971*, 2015.

[7] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, “Benchmarking deep reinforcement learning for continuous control,” in *International Conference on Machine Learning*, 2016, pp. 1329–1338.

[8] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, p. 529, 2015.

[9] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, “Mastering the game of go without human knowledge,” *Nature*, vol. 550, no. 7676, p. 354, 2017.

[10] M. J. Mataric, “Reward functions for accelerated learning,” in *Machine Learning Proceedings 1994*. Elsevier, 1994, pp. 181–189.

[11] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, “Deep reinforcement learning that matters,” in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[12] G. Schultz and K. Mombaur, “Modeling and optimal control of human-like running,” *IEEE/ASME Transactions on mechatronics*, vol. 15, no. 5, pp. 783–792, 2010.

[13] K. Mombaur, A. Truong, and J.-P. Laumond, “From human to humanoid locomotion—an inverse optimal control approach,” *Autonomous robots*, vol. 28, no. 3, pp. 369–383, 2010.

[14] M. Diehl, H. G. Bock, H. Diedam, and P.-B. Wieber, “Fast direct multiple shooting algorithms for optimal robot control,” in *Fast motions in biomechanics and robotics*. Springer, 2006, pp. 65–93.

[15] L. Magni, R. Scattolini, and K. J. Åström, “Global stabilization of the inverted pendulum using model predictive control,” *IFAC Proceedings Volumes*, vol. 35, no. 1, pp. 141–146, 2002.

[16] K. Doya, “Reinforcement learning in continuous time and space,” *Neural computation*, vol. 12, no. 1, pp. 219–245, 2000.

[17] M. Lutter, C. Ritter, and J. Peters, “Deep lagrangian networks: Using physics as model prior for deep learning,” in *International Conference on Learning Representations*, 2019.

[18] M. Papageorgiou, M. Leibold, and M. Buss, *Optimierung*. Springer, 1991.

[19] M. Raissi and G. E. Karniadakis, “Hidden physics models: Machine learning of nonlinear partial differential equations,” *Journal of Computational Physics*, vol. 357, pp. 125–141, 2018.