A Step Closer to Model Based RL

TINGWU WANG,

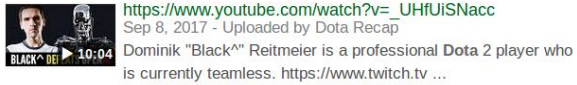
MACHINE LEARNING GROUP,

UNIVERSITY OF TORONTO

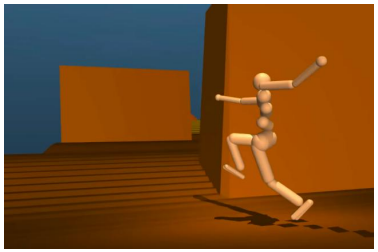
Contents

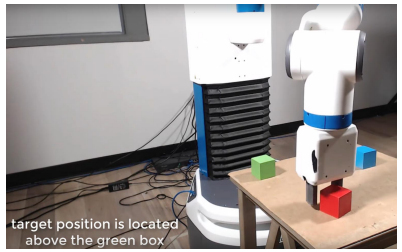
Introduction

1. Recently, Reinforcement Learning is making numerous successes.
   1. Complex Humanoid Control (PPO[2, 4] & TRPO[1] & DDPG [3]).
   2. AlphaGo (Deepmind) [5].
   3. DOTA (OpenAI).



2. "Model-Free methods almost have no place in the future"
   -- said by an anonymous Prof. Joshua
   1. Data inefficiency
   2. Transferability
   3. Safety
   4. Interprebility
   5. Uncertainty
   6. Hierarchical RL?
   7. ...





Keywords in Model-Based RL and Control

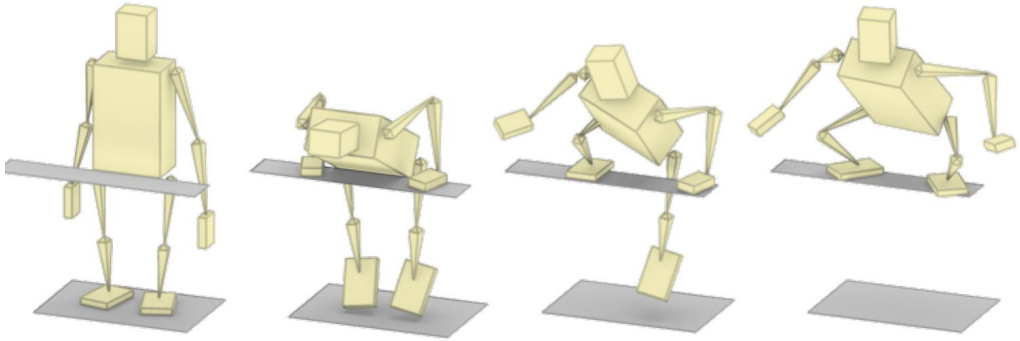
1. Closed-Loop (MPC) vs. Open-Loop
    1. MPC uses feedback loops, namely observes resulting state and update the control signal.
        1. Open-loop: Control signal as a function of **time**.
        2. Closed-loop: Control signal as a function of **state**.
            1. MPC (Model Predictive Control): closed-loop algorithm with online state update
    2. Some of the algorithms are Open-loop.
        1. Off-line calculated trajectories.

Keywords in Model-Based RL and Control

1. Model-Based RL vs. Optimal Control vs. Trajectory Optimization vs. Motion Planning
    1. Model-Based RL:
        1. The term "RL" describes the data / environment.
        2. Supervised Learning & Unsupervised Learning & RL.
    2. Optimal Control:
        1. The problem of finding a control law for a given system.
        2. Essentially RL is one type of optimal control problem.
    3. Trajectory Optimization:
        1. One type of optimal control problem, where a sequence of states (trajectory) are optimized.
        2. Most of the robotics tasks.
    4. Motion Planning:
        1. More related to **navigation**.
        2. Sometimes interchangeably used with trajectory optimization.

Multi-Joint Dynamics and Contacts Modelling

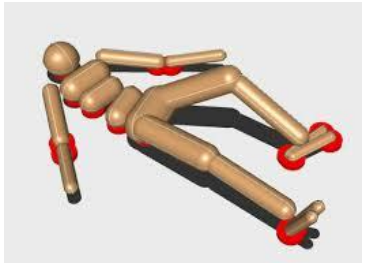
1. For years, the robotics community is more concerned with dynamics than policy.
    1. Given the ground-truth dynamics, you might theoretically solve any trajectory optimization problem.
    2. Example:
        1. MPC control on humanoid [7].
        2. Motion planning [6].
    3. If ground-truth dynamics is not given, **none** of the methods have been proven to work.
        1. Learn the dynamics? Much more difficult than you imagine.
            1. Provide a very good initial dynamics.
        2. GPS [8], PILCO [9] work respectively on graphics enginine and cart-pole (real-life).



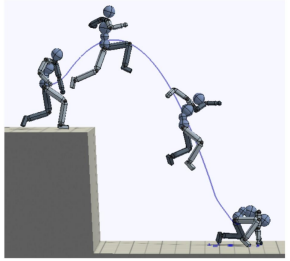
Multi-Joint Dynamics and Contacts Modelling

1. Dynamics is not always easy to model
    1. In the physical world contact happens on very short time-scales.
        1. Trajectory optimizer will fail mathematically (gradient needed).
    2. Contacts could not be modelled perfectly.
        1. Hertz-Hunt-Crossley spring-dampers.
            1. Prohibitively expensive.
        2. Time stepping integrators (relatively low fidelity, used in MuJoCo)
        3. Treat contact as a variable to optimize (used in CIO)
            1. If fidelity is not your first concern (e.g. motion synthesis [10])

Real-life Level



Robotic-simulation Level



Computer Graphics Level

Multi-Joint Dynamics and Contacts Modelling

1. Dynamics is not always easy to model
   1. High-fidelity simulator? Dynamics is never perfectly modelled.
      1. Currently established model: **Mu**lti-**Jo**int Dynamics with **Co**ntacts estimation

$$M(\mathbf{q})\,d\mathbf{v} = (\mathbf{b}(\mathbf{q},\mathbf{v}) + \tau)\,dt + J_E(\mathbf{q})^{\mathsf{T}}\,\mathbf{f}_E(\mathbf{q},\mathbf{v},\tau) + J_C(\mathbf{q})^{\mathsf{T}}\,\mathbf{f}_C(\mathbf{q},\mathbf{v},\tau)$$

$\mathbf{q}$	position in generalized coordinates	$\mathbf{v}_E^*$	desired velocity in equality constraint coordinates
$\mathbf{v}$	velocity in generalized coordinates	$\mathbf{f}_E$	impulse caused by equality constraints
M	inertia matrix in generalized coordinates	J_C	Jacobian of active contacts
$\mathbf{b}$	"bias" forces: Coriolis, centrifugal, gravity, springs	$\mathbf{v}_C$	velocity in contact coordinates
τ	external/applied forces	$\mathbf{f}_C$	impulse caused by contacts
ϕ	equality constraints: $\phi(\mathbf{q}) = 0$	h	time step
J_E	Jacobian of equality constraints		

      2. MuJoCo Could model:
         1. Hinge joint, slide joint, tendon, between-body contact & friction
      3. The gradient? Finite difference.
   2. Both model based and model free methods perform poorly in sim-to-real transfer. (Perhaps the simulator is too bad?)

Contents

Classic Cases

1. LQR and iLQR with MPC [7]
   1. Problem formulation (dynamic function and cost function):

$$\mathbf{x}_{i+1} = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i)$$

$$J_0(\mathbf{x}, \mathbf{U}) = \sum_{i=0}^{N-1} \ell(\mathbf{x}_i, \mathbf{u}_i) + \ell_f(\mathbf{x}_N)$$

Equivalently, at every timestep:

$$J_i(\mathbf{x}, \mathbf{U}_i) = \sum_{j=i}^{N-1} \ell(\mathbf{x}_j, \mathbf{u}_j) + \ell_f(\mathbf{x}_N).$$

$$V(\mathbf{x}, i) \equiv \min_{\mathbf{U}_i} J_i(\mathbf{x}, \mathbf{U}_i).$$

(From i=N to i=0) Given the current state and potentially changes on the current state, what's the best actions u?

   2. Solution: Second order optimization recursively (dynamic programming).

Classic Cases

1. LQR and iLQR with MPC
   1. Optimization using Gauss-Newton Hessian Method.

$$V(\mathbf{x}, i) \equiv \min_{\mathbf{U}_i} J_i(\mathbf{x}, \mathbf{U}_i).$$

Dynamic Programming

$$V(\mathbf{x}, i) = \min_{\mathbf{u}}[\ell(\mathbf{x}, \mathbf{u}) + V(\mathbf{f}(\mathbf{x}, \mathbf{u}), i+1)]$$

   2. Define a function of perturbations Q to figure out
      1. What's the optimal control signal u, if we know how much the current state x will change?
      2. Directly optimizing control signal u fails (why? The states is changing)

$$Q(\delta\mathbf{x}, \delta\mathbf{u}) = \ell(\mathbf{x} + \delta\mathbf{x}, \mathbf{u} + \delta\mathbf{u}, i) - \ell(\mathbf{x}, \mathbf{u}, i)$$
$$+ V(\mathbf{f}(\mathbf{x} + \delta\mathbf{x}, \mathbf{u} + \delta\mathbf{u}), i+1) - V(\mathbf{f}(\mathbf{x}, \mathbf{u}), i+1)$$

$$Q_{\mathbf{x}} = \ell_{\mathbf{x}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{x}}$$
$$Q_{\mathbf{u}} = \ell_{\mathbf{u}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{x}}$$
$$Q_{\mathbf{xx}} = \ell_{\mathbf{xx}} + \mathbf{f}_{\mathbf{x}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{xx}}$$
$$Q_{\mathbf{uu}} = \ell_{\mathbf{uu}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{u}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{uu}}$$
$$Q_{\mathbf{ux}} = \ell_{\mathbf{ux}} + \mathbf{f}_{\mathbf{u}}^{\mathsf{T}} V'_{\mathbf{xx}} \mathbf{f}_{\mathbf{x}} + V'_{\mathbf{x}} \cdot \mathbf{f}_{\mathbf{ux}}.$$

Ignored in MPC

Could be computed recursively with

$$V_{\mathbf{x}}(i) = Q_{\mathbf{x}} - Q_{\mathbf{u}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}$$
$$V_{\mathbf{xx}}(i) = Q_{\mathbf{xx}} - Q_{\mathbf{xu}} Q_{\mathbf{uu}}^{-1} Q_{\mathbf{ux}}.$$

Classic Cases

1. LQR and iLQR with MPC
   1. Given the perturbation on x (caused by pervious change of u), the optimal perturbation of u will be

$$\delta \mathbf{u}^* = \underset{\delta \mathbf{u}}{\operatorname{argmin}} Q(\delta \mathbf{x}, \delta \mathbf{u}) = -Q_{\mathbf{uu}}^{-1}(Q_{\mathbf{u}} + Q_{\mathbf{ux}}\delta \mathbf{x}),$$

$$\text{open-loop term } \mathbf{k} = -Q_{\mathbf{uu}}^{-1}Q_{\mathbf{u}}$$
$$\text{feedback gain term } \mathbf{K} = -Q_{\mathbf{uu}}^{-1}Q_{\mathbf{ux}}$$

   2. Update the control signals and states as:

$$\hat{\mathbf{x}}(1) = \mathbf{x}(1)$$
$$\hat{\mathbf{u}}(i) = \mathbf{u}(i) + \mathbf{k}(i) + \mathbf{K}(i)(\hat{\mathbf{x}}(i) - \mathbf{x}(i))$$
$$\hat{\mathbf{x}}(i+1) = \mathbf{f}(\hat{\mathbf{x}}(i), \hat{\mathbf{u}}(i))$$

Classic Cases

1. Contact Invariant Optimization (CIO) and CIO with Policy Network [6, 11].
   1. Direct Collocation method

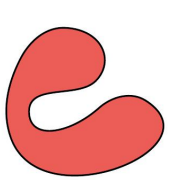
Forward Shooting:

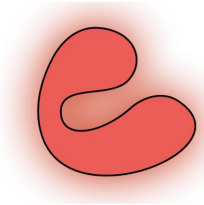
$$\min_{\mathbf{u}^0 \ldots \mathbf{u}^T} \sum_t C^t(\mathbf{x}^t), \quad \mathbf{x}^{t+1} = f(\mathbf{x}^t, \mathbf{u}^t)$$

inverse dynamics function

Direct Collocation:

$$\min_{\mathbf{x}^0 \ldots \mathbf{x}^T} \sum_t C^t(\mathbf{x}^t), \; st \; f^{-1}(\mathbf{x}^t, \mathbf{x}^{t+1}) = \mathbf{u}^t \in \mathcal{U}$$





   2. Instead of optimize a contrained problem, optimize a uncontrained objective

$$L(\mathbf{s}) = L_{\text{CI}}(\mathbf{s}) + L_{\text{Physics}}(\mathbf{s})$$

   3. Shooting method (MPC) seems to be not capable of solving task where reward signal is not consistent and dense.
      1. Hindsight Experience Replay [12]

Classic Cases

1. Contact Invariant Optimization (CIO) and CIO with Policy Network.

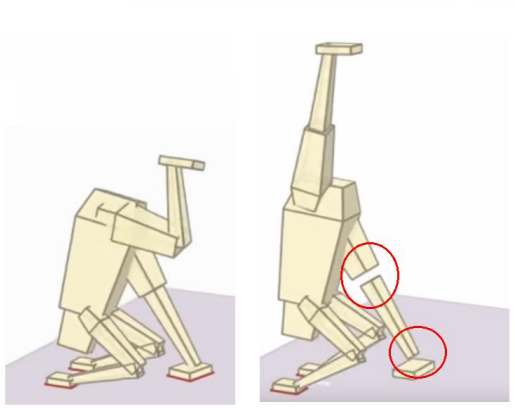
$$C(\mathbf{X}) = \sum_t c(\phi^t(\mathbf{X}))$$

$$C_{\mathbf{X}} = \sum_t c^t_\phi \phi^t_{\mathbf{X}}$$

$$C_{\mathbf{XX}} = \sum_t (\phi^t_{\mathbf{X}})^\top c^t_{\phi\phi} \phi^t_{\mathbf{X}} + c^t_\phi \phi^t_{\mathbf{XX}} \approx \sum_t (\phi^t_{\mathbf{X}})^\top c^t_{\phi\phi} \phi^t_{\mathbf{X}}$$

- Find optimal solution by iterative Gauss-Newton steps

$$\mathbf{X}^* = \mathbf{X}^* - C^{-1}_{\mathbf{XX}} C_{\mathbf{X}}$$

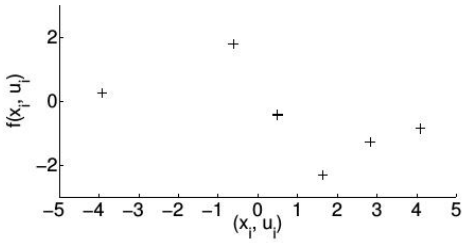


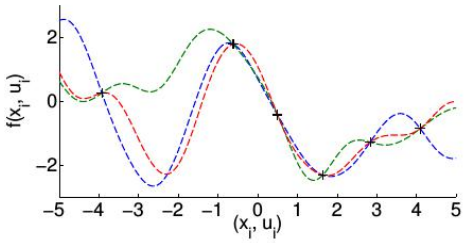
   1. No computational speed-up reported.
   2. Open-loop, unstable and unrobust
      1. Only works with low-fidelity graphics engine
      2. No proof of ability of working on robotics simulator nor complex real-life system.
   3. Could train a neural network to distill the policy [11].

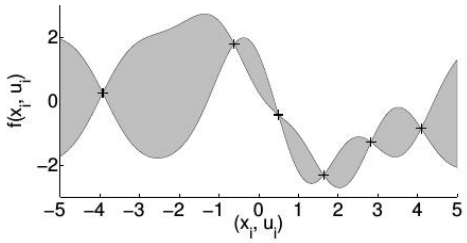
Classic Cases

1. Probabilistic Inference for Learning Control (PILCO) [9]
   1. **Unknow dynamics!**
   2. Consider the uncertainty in dynamics with Gaussian Process
      1. Input: current state x_{t-1} and u_{t-1}
      2. Target: the difference of the state \Delta_t = x_{t} - x_{t-1}

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) = \mathcal{N}\big(\mathbf{x}_t \,|\, \mu_t, \mathbf{\Sigma}_t\big)\,,$$
$$\mu_t = \mathbf{x}_{t-1} + \mathbb{E}_f[\Delta_t]\,,$$
$$\mathbf{\Sigma}_t = \mathrm{var}_f[\Delta_t]\,.$$







Classic Cases

1. Probabilistic Inference for Learning Control (PILCO)
   1. Evaluate the value function under current policy

$$V^{\pi}(\mathbf{x}_0) = \sum_{t=0}^{T} \mathbb{E}_{\mathbf{x}_t}[c(\mathbf{x}_t)]$$

$$\mathbb{E}_{\mathbf{x}_t}[c(\mathbf{x}_t)] = \int c(\mathbf{x}_t) \underbrace{p(\mathbf{x}_t)}_{\text{Gaussian}} \, \mathrm{d}\mathbf{x}_t \,,$$

$$p(\mathbf{x}_t) = \iint p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{u}_{t-1}) p(\mathbf{u}_{t-1}|\mathbf{x}_{t-1}) p(\mathbf{x}_{t-1}) \, \mathrm{d}\mathbf{x}_{t-1} \, \mathrm{d}\mathbf{u}_{t-1} \,,$$

All the intermediant results are analytic (play with gaussians)

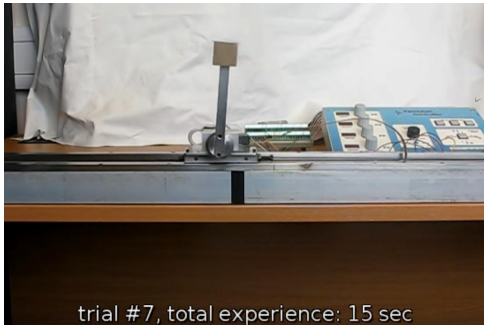
   2. Update policy

$$\frac{\mathrm{d}}{\mathrm{d}\boldsymbol{\psi}} \mathbb{E}_{\mathbf{x}_t}[c(\mathbf{x}_t)] = \left(\frac{\partial}{\partial \boldsymbol{\mu}_t} \mathbb{E}_{\mathbf{x}_t}[c(\mathbf{x}_t)] \right) \frac{\mathrm{d}\boldsymbol{\mu}_t}{\mathrm{d}\boldsymbol{\psi}} + \left(\frac{\partial}{\partial \boldsymbol{\Sigma}_t} \mathbb{E}_{\mathbf{x}_t}[c(\mathbf{x}_t)] \right) \frac{\mathrm{d}\boldsymbol{\Sigma}_t}{\mathrm{d}\boldsymbol{\psi}}$$

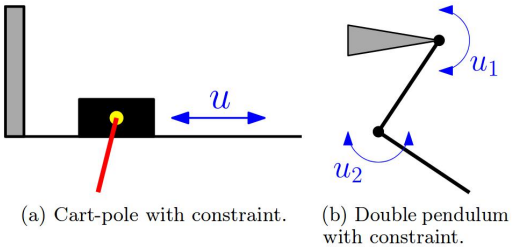
standard gradient based optimization

Classic Cases

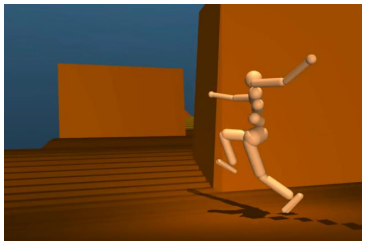
1. Probabilistic Inference for Learning Control (PILCO)
    1. Extremely data-efficient

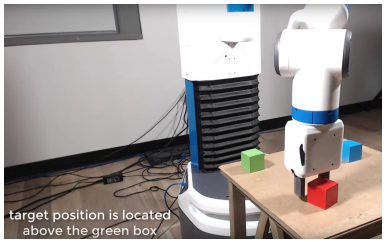


    2. Extremely slow and cannot be scaled to complex problems.
        1. In the newest follow-up paper of PILCO [13] (AISTATS 2018), PILCO is still contrained on cart-pole (compared to other complex benchmarks).



(a) Cart-pole with constraint. (b) Double pendulum with constraint.





Classic Cases

1. Guided Policy Search (GPS) [8] and GPS with unknown dynamics [4]
    1. Find the optimal control with MPC (talked about in previous slides)
    2. Train the policy network in a supervised-learning fashion.

policy search (RL)	complex dynamics	complex policy	HARD
supervised learning	~~complex dynamics~~	complex policy	EASY
trajectory optimization	complex dynamics	~~complex policy~~	EASY

    3. It is essentially imitation learning (if we assume that MPC solution is perfect) --> Guided Cost Learning [15]

2. With unknown dynamics
    1. Fitted dynamics are only valid in a local region around the samples
    2. Limiting the change in the trajectory distribution in each dynamic programming
        1. Add KL contraints to the cost.

Some Current Progress

1. Nagabandi, A., Kahn, G., Fearing, R. S., & Levine, S. (2017). Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. arXiv preprint arXiv:1708.02596.
2. Kurutach, T., Clavera, I., Duan, Y., Tamar, A., & Abbeel, P. (2018). Model-ensemble trust-region policy optimization. arXiv preprint arXiv:1802.10592.
3. Pong, V., Gu, S., Dalal, M., & Levine, S. (2018). Temporal difference models: Model-free deep rl for model-based control. arXiv preprint arXiv:1802.09081.

Reference

[1] Schulman, J., Levine, S., Abbeel, P., Jordan, M., & Moritz, P. (2015, June). Trust region policy optimization. In International Conference on Machine Learning (pp. 1889-1897).

[2] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.

[3] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., ... & Wierstra, D. (2015). Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.

[4] Heess, N., Sriram, S., Lemmon, J., Merel, J., Wayne, G., Tassa, Y., ... & Silver, D. (2017). Emergence of locomotion behaviours in rich environments. arXiv preprint arXiv:1707.02286.

[5] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... & Dieleman, S. (2016). Mastering the game of Go with deep neural networks and tree search. nature, 529(7587), 484-489.

[6] Mordatch, I., Todorov, E., & Popović, Z. (2012). Discovery of complex behaviors through contact-invariant optimization. ACM Transactions on Graphics (TOG), 31(4), 43.

[7] Tassa, Y., Erez, T., & Todorov, E. (2012, October). Synthesis and stabilization of complex behaviors through online trajectory optimization. In Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on (pp. 4906-4913). IEEE.

[8] Levine, S., & Koltun, V. (2013, February). Guided policy search. In International Conference on Machine Learning (pp. 1-9).

[9] Deisenroth, M., & Rasmussen, C. E. (2011). PILCO: A model-based and data-efficient approach to policy search. In Proceedings of the 28th International Conference on machine learning (ICML-11) (pp. 465-472).

[10] Peng, X. B., Abbeel, P., Levine, S., & van de Panne, M. (2018). DeepMimic: Example-Guided Deep Reinforcement Learning of Physics-Based Character Skills. arXiv preprint arXiv:1804.02717.

[11] Mordatch, I., Lowrey, K., Andrew, G., Popovic, Z., & Todorov, E. V. (2015). Interactive control of diverse complex characters with neural networks. In Advances in Neural Information Processing Systems (pp. 3132-3140).

[12] Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., ... & Zaremba, W. (2017). Hindsight experience replay. In Advances in Neural Information Processing Systems (pp. 5048-5058).

[13] Kamthe, S., & Deisenroth, M. P. (2017). Data-Efficient Reinforcement Learning with Probabilistic Model Predictive Control. arXiv preprint arXiv:1706.06491.

[14] Levine, S., & Abbeel, P. (2014). Learning neural network policies with guided policy search under unknown dynamics. In Advances in Neural Information Processing Systems (pp. 1071-1079).

[15] Finn, C., Levine, S., & Abbeel, P. (2016, June). Guided cost learning: Deep inverse optimal control via policy optimization. In International Conference on Machine Learning (pp. 49-58).