Decoupled Data-Based Approach for Learning to Control Nonlinear Dynamical Systems

Ran Wang[1], Karthikeya S. Parunandi[1], Dan Yu[2], Dileep Kalathil[3], Suman Chakravorty[1]

***Abstract*— This paper addresses the problem of learning the optimal control policy for a nonlinear stochastic dynamical. This problem is subject to the 'curse of dimensionality' associated with the dynamic programming method. This paper proposes a novel decoupled data-based control (D2C) algorithm that addresses this problem using a decoupled, 'open-loop - closed-loop', approach. First, an open-loop deterministic trajectory optimization problem is solved using a black-box simulation model of the dynamical system. Then, closed-loop control is developed around this open-loop trajectory by linearization of the dynamics about this nominal trajectory. By virtue of linearization, a linear quadratic regulator based algorithm can be used for this closed-loop control. We show that the performance of D2C algorithm is approximately optimal. Moreover, simulation performance suggests a significant reduction in training time compared to other state of the art algorithms.**

I. Introduction

The control of an unknown dynamical system adaptively has a rich history in control literature [1] This classical literature provides a rigorous analysis of the asymptotic performance and stability of the linear closed-loop system. The optimal control of a stochastic nonlinear system with continuous state space and action space is a significantly more challenging problem due to the 'curse of dimensionality', the exponential computational complexity growth associated with dynamic programming. *Learning to control* problems where the model of the system is unknown also suffer from this computational complexity issues, in addition to the usual identifiability problems in adaptive control.

The last several years have seen significant progress in deep neural networks-based reinforcement learning approaches for controlling unknown dynamical systems, with applications in many areas like playing games [2], locomotion [3] and robotic hand manipulation [4]. A number of new algorithms that show promising performance are proposed [5] [6] and various improvements and innovations have been continuously developed. However, despite excellent performance on a number of tasks, reinforcement learning (RL) is still considered very data and time-intensive. The training time for such algorithms is typically really large. Moreover, high variance and reproducibility issues on the performance are reported [7].

Our Contributions: In this work, we propose a novel decoupled data-based control (D2C) algorithm for learning to control an unknown nonlinear dynamical system. Our approach introduces a rigorous decoupling of the open-loop (planning) problem from the closed-loop (feedback control) problem. This decoupling allows us to come up with a highly efficient approach to solve the problem in a completely data-based fashion. Our approach proceeds in two steps: (i) first, we optimize the nominal open-loop trajectory of the system using a blackbox simulation model, (ii) then we identify the linear system governing perturbations from the nominal trajectory using random input-output perturbation data, and design an LQR controller for this linearized system. We show that the performance of D2C algorithm is approximately optimal, in the sense that the decoupled design is near-optimal to second order in a suitably defined noise parameter. Moreover, simulation performance suggests a significant reduction in training time compared to other state of the art algorithms.

Related work: The approaches to the problem of controlling an unknown dynamical system can be divided into two broad classes, model-based methods and model-free methods.

In the model-based methods, many techniques [8] rely on a discretization of the underlying state and action space, and hence, run into the curse of dimensionality, the fact that the computational complexity grows exponentially with the dimension of the state space of the problem. The most computationally efficient among these techniques are trajectory-optimization methods such as differential dynamic programming (DDP) [9] and the iterative linear quadratic Gaussian (ILQG) algorithm [10], which is closely related to DDP, but considers only the first-order expansion of the dynamics (in DDP, a second-order expansion is considered), and is shown to be computationally more efficient. In both approaches, the control policy is executed to compute a new nominal trajectory, and the procedure is repeated until convergence.

Model-free methods, more popularly known as approximate dynamic programming [11] or reinforcement learning (RL) methods [12], seek to improve the control policy by repeated interactions with the environment, while observing the system's responses. Standard RL algorithms are broadly divided into value-based methods, like Q-learning, and policy-based methods, like policy gradient algorithms. Recently, function approximation using deep neural networks has significantly

This work was submitted for review on February 19, 2021. The work of Wang, Parunandi and Chakravorty was supported by NSF grants ECCS-1637889, CDSE 1802867, and AFOSR DDIP grant FA9550-17-1-0068. The work of Yu was supported by NSFC project 61801213. The work of Kalathil was supported in part by NSF grant CRII-CPS-1850206.

[1]R. Wang, K. Parunandi and S. Chakravorty are with the Department of Aerospace Engineering, Texas A&M University, Texas, USA. {rwang0417, s.parunandi, schakrav}@tamu.edu

[2]D.Yu is with the College of Astronautics, Nanjing University of Aeronautics and Astronautics, Nanjing, 210016, China. yudan@nuaa.edu.cn

[3]D. Kalathil is with the Department of Electrical and Computer Engineering, Texas A&M University, Texas, USA. dileep.kalathil@tamu.edu

improved the performance of the reinforcement learning algorithm, leading to a growing class of literature on 'deep reinforcement learning'. Despite the success, the amount of samples and training time required still seem prohibitive. On the other hand, works such as [13] demonstrated that simple policies such as the ones with linear parameterization showed a promising performance comparable to benchmark results obtained by policies represented using deep neural networks. We note that the preliminary ideas of this paper have previously appeared in the conference publication [14]: this paper provides an analysis for the decoupling principle and also gives an extensive empirical validation of the proposed algorithm and comparison with a benchmark deep RL algorithm.

The rest of the paper is organized as follows. In Section II, the basic problem formulation is outlined. In Section III, a decoupling result which solves the MDP in a "decoupled open loop-closed loop " fashion is briefly summarized. In Section IV, we propose a decoupled data-based control algorithm, with discussions of implementation problems. In Section V, we test the proposed approach using four typical benchmarking examples with comparisons to a state of the art RL technique.

II. PROBLEM FORMULATION

Consider the following discrete time nonlinear stochastic dynamical system: $x_{t+1} = h(x_t, u_t, w_t)$,where $x_t \in \mathbb{R}^n$, $u_t \in \mathbb{R}^p$ are the state and control vector at time k, respectively. The process noise w_t is assumed as zero-mean, uncorrelated Gaussian white noise, with covariance W.

The *optimal stochastic control* problem is to find the the control policy $\pi^o = \{\pi_0^o, \pi_1^o, \cdots, \pi_{T-1}^o\}$ such that the expected cumulative cost is minimized, i.e.,

$$\pi^o = \arg\min_{\pi} \ \tilde{J}^\pi(x_0), \quad \text{where,}$$

$$\tilde{J}^\pi(x_0) = \mathbb{E}_\pi \left[\sum_{t=0}^{T-1} c(x_t, u_t) + c_T(x_T) | x_0 \right], \tag{1}$$

$u_t = \pi_t(x_t)$, and $c(\cdot, \cdot)$ is the instantaneous cost function, and $c_T(\cdot)$ is the terminal cost function. In the following, we assume that the initial state x_0 is fixed, and denote $\tilde{J}^\pi(x_0)$ simply as $\tilde{J}^\pi$.

III. A NEAR OPTIMAL DECOUPLING PRINCIPLE

We first outline a near-optimal decoupling principle in stochastic optimal control that paves the way for the D2C algorithm described in Section IV.

We make the following assumption on the dynamics given in Section II:

$$x_{t+1} = f(x_t, u_t) + \epsilon w_t, \tag{2}$$

where $\epsilon < 1$ is a small parameter, i.e., the noise dependence is linear in the system dynamics, is Gaussian and white in time.

A. Linearization w.r.t. Nominal Trajectory

Consider a noiseless version of the system dynamics given by (2), $w_t = 0$ for all t. We denote the "nominal" state trajectory as $\bar{x}_t$ and the "nominal" control as $\bar{u}_t$, with the initial condition, $\bar{x}_0 = x_0$, known exactly. The resulting dynamics without noise is given by $\bar{x}_{t+1} = f(\bar{x}_t, \bar{u}_t)$. Let $\pi = (\pi_t)_{t=0}^{T-1}$ be a given control policy, i.e., $u_t = \pi_t(x_t)$, and thus, $\bar{u}_t = \pi_t(\bar{x}_t)$.

Assuming that $f(\cdot)$ and $\pi_t(\cdot)$ are sufficiently smooth, we can linearize the dynamics about the nominal trajectory. Denoting $\delta x_t = x_t - \bar{x}_t, \delta u_t = u_t - \bar{u}_t$, we can express,

$$\delta x_{t+1} = A_t \delta x_t + B_t \delta u_t + S_t(\delta x_t, \delta u_t) + \epsilon w_t, \tag{3}$$

$$\delta u_t = K_t \delta x_t + \tilde{S}_t(\delta x_t), \tag{4}$$

where $A_t = \frac{\partial f}{\partial x}|_{\bar{x}_t, \bar{u}_t}$, $B_t = \frac{\partial f}{\partial u}|_{\bar{x}_t, \bar{u}_t}$, $K_t = \frac{\partial \pi_t}{\partial x}|_{\bar{x}_t}$, and $S_t(\cdot, \cdot), \tilde{S}_t(\cdot)$ are second and higher-order terms in the respective expansions. Similarly, we can linearize the instantaneous cost $c(x_t, u_t)$ about the nominal values $(\bar{x}_t, \bar{u}_t)$ as,

$$c(x_t, u_t) = c(\bar{x}_t, \bar{u}_t) + C_t^x \delta x_t + C_t^u \delta u_t + H_t(\delta x_t, \delta u_t), \tag{5}$$

$$c_T(x_T) = c_T(\bar{x}_T) + C_T^x \delta x_T + H_T(\delta x_T), \tag{6}$$

where $C_t^x = \frac{\partial c}{\partial x}|_{\bar{x}_t, \bar{u}_t}$, $C_t^u = \frac{\partial c}{\partial u}|_{\bar{x}_t, \bar{u}_t}$, $C_T^x = \frac{\partial c_T}{\partial x}|_{\bar{x}_t}$, and $H_t(\cdot, \cdot)$ and $H_T(\cdot)$ are second and higher-order terms in the respective expansions.

Using (3) and (4), we can write the closed loop dynamics of the trajectory $(\delta x_t)_{t=1}^T$ as,

$$\delta x_{t+1} = \underbrace{(A_t + B_t K_t)}_{\bar{A}_t} \delta x_t + \underbrace{\{B_t \tilde{S}_t(\delta x_t) + S_t(\delta x_t, K_t \delta x_t + \tilde{S}_t(\delta x_t)))\}}_{\bar{S}_t(\delta x_t)} + \epsilon w_t, \tag{7}$$

where $\bar{A}_t$ represents the linear part of the closed loop systems and the term $\bar{S}_t(.)$ represents the second and higher-order terms in the closed loop system. Similarly, the closed loop incremental cost given in (5) can be expressed as

$$c(x_t, u_t) = \underbrace{c(\bar{x}_t, \bar{u}_t)}_{\bar{c}_t} + \underbrace{[C_t^x + C_t^u K_t]}_{\bar{C}_t} \delta x_t, + \underbrace{H_t(\delta x_t, K_t \delta x_t + \tilde{S}_t(\delta x_t))}_{\bar{H}_t(\delta x_t)}. \tag{8}$$

Therefore, the cumulative cost of any given closed loop trajectory $(x_t, u_t)_{t=0}^T$ can be expressed as,

$$J^\pi = \sum_{t=0}^{T-1} c(x_t, u_t = \pi_t(x_t)) + c_T(x_T)$$
$$= \sum_{t=0}^{T} \bar{c}_t + \sum_{t=0}^{T} \bar{C}_t \delta x_t + \sum_{t=0}^{T} \bar{H}_t(\delta x_t), \tag{9}$$

where $\bar{c}_T = c_T(\bar{x}_T), \bar{C}_T = C_T^x, \bar{H}_T(.) = H_T(.)$.

We first show the following result.

Lemma 1: The state perturbation equation

$$\delta x_{t+1} = \bar{A}_t \delta x_t + \bar{S}_t(\delta x_t) + \epsilon w_t$$

given in (7) can be equivalently characterized as

$$\delta x_t = \delta x_t^l + \bar{\bar{S}}_t, \ \ \delta x_{t+1}^l = \bar{A}_t \delta x_t^l + \epsilon w_t \tag{10}$$

where $\bar{\bar{S}}_t$ is an $O(\epsilon^2)$ function that depends on the entire noise history $\{w_0, w_1, \cdots w_t\}$ and δx_t^l evolves according to the linear closed loop system as above.

The proof is given in Appendix A.

Now, we show the following important result.

Proposition 1: The mean and variance of the closed loop cost J^π obey the following relationships, where $\bar{J}^\pi = \sum_{t=0}^T \bar{c}_t$, and $\delta J_1^\pi = \sum_{t=0}^T \bar{C}_t \delta x_t^l$ (see (9)-(10)):

$$\tilde{J}^\pi = \mathbb{E}[J^\pi] = \bar{J}^\pi + O(\epsilon^2),$$
$$\text{Var}(J^\pi) = \underbrace{\text{Var}(\delta J_1^\pi)}_{O(\epsilon^2)} + O(\epsilon^4).$$

Proof: Using (10) in (9), we can obtain the cumulative cost of any sample closed loop trajectory as,

$$J^\pi = \underbrace{\sum_{t=0}^T \bar{c}_t}_{\bar{J}^\pi} + \underbrace{\sum_{t=0}^T \bar{C}_t \delta x_t^l}_{\delta J_1^\pi} + \underbrace{\sum_{t=0}^T \bar{H}_t(\delta x_t) + \bar{C}_t \bar{\bar{S}}_t}_{\delta J_2^\pi}. \qquad (11)$$

From (11), we get,

$$\begin{aligned} \tilde{J}^\pi &= \mathbb{E}[J^\pi] = \mathbb{E}[\bar{J}^\pi + \delta J_1^\pi + \delta J_2^\pi], \\ &= \bar{J}^\pi + \mathbb{E}[\delta J_2^\pi] = \bar{J}^\pi + O(\epsilon^2), \end{aligned} \qquad (12)$$

The first equality in the last line of the equations before follows from the fact that $\mathbb{E}[\delta x_t^l] = 0$, since its the linear part of the state perturbation driven by Gaussian white noise and by definition $\delta x_1^l = 0$. The second equality follows from the fact that δJ_2^π is an $O(\epsilon^2)$ function since $\bar{H}_t(\delta x_t)$ and $\bar{\bar{S}}_t$ are both $O(\epsilon^2)$ functions. Let $\delta\tilde{J}_2^\pi \equiv \mathbb{E}[\delta J_2^\pi]$. Noting that $\delta\tilde{J}_1^\pi \equiv \mathbb{E}[\delta J_1^\pi] = 0$, we obtain:

$$\begin{aligned} \text{Var}(J^\pi) &= \mathbb{E}[J^\pi - \tilde{J}^\pi]^2 \\ &= \mathbb{E}[\bar{J}^\pi + \delta J_1^\pi + \delta J_2^\pi - \bar{J}^\pi - \delta\tilde{J}_2^\pi]^2 \\ &= \mathbb{E}[\delta J_1^\pi + \delta J_2^\pi - \delta\tilde{J}_2^\pi]^2 \\ &= \text{Var}(\delta J_1^\pi) + \text{Var}(\delta J_2^\pi) + 2\mathbb{E}[\delta J_1^\pi(\delta J_2^\pi - \delta\tilde{J}_2^\pi)], \\ &= \text{Var}(\delta J_1^\pi) + \text{Var}(\delta J_2^\pi) + 2\mathbb{E}[\delta J_1^\pi \delta J_2^\pi], \end{aligned} \qquad (13)$$

where the last equality follows from the fact that $\mathbb{E}[\delta J_1^\pi] = 0$ and the fact that $\delta\tilde{J}_2^\pi$ is non-random. Since δJ_2^π is $O(\epsilon^2)$, $\text{Var}(\delta J_2^\pi)$ is an $O(\epsilon^4)$ function. It can be shown that $\mathbb{E}[\delta J_1^\pi \delta J_2^\pi]$ is $O(\epsilon^4)$ as well (see Lemma 2 in Appendix B). Finally $\text{Var}(\delta J_1^\pi)$ is an $O(\epsilon^2)$ function because δx_t^l is an $O(\epsilon)$ function. Combining these, we get the desired result. ■

The following observations can now be made from Proposition 1.

Remark 1 (Expected cost-to-go): Recall that $u_t = \pi_t(x_t) = \bar{u}_t + K_t\delta x_t + \tilde{S}_t(\delta x_t)$. However, note that due to Proposition 1, the expected cost-to-go, $\tilde{J}^\pi$, is determined to within $O(\epsilon^2)$ by the nominal control action sequence $\bar{u}_t$.

Remark 2 (Variance of cost-to-go): Given nominal control action $\bar{u}_t$, variance of the cost-to-go, which is $O(\epsilon^2)$, is determined to within $O(\epsilon^4)$ by the linear feedback term $K_t\delta x_t$.

B. Decoupled Approach for Feedback Control

Proposition 1 and the remarks above allow us to propose the following decoupled approach to stochastic nonlinear feedback control in the sense that the open-loop design is decoupled from the closed-loop design.

Open-Loop Design. First, we design an optimal (open-loop) control sequence $\bar{u}_t^*$ for the noiseless system. More precisely,

$$\begin{aligned} (\bar{u}_t^*)_{t=0}^{T-1} &= \arg\min_{(\bar{u}_t)_{t=0}^{T-1}} \sum_{t=0}^{T-1} c(\bar{x}_t, \bar{u}_t) + c_T(\bar{x}_T), \\ \bar{x}_{t+1} &= f(\bar{x}_t, \bar{u}_t), \bar{x}_0 = x_0. \end{aligned} \qquad (14)$$

Details of this open-loop design are discussed in Section IV.

Closed Loop Design. We find the optimal feedback gain K_t^* such that the variance of the linear closed loop system around the optimlal nominal path, $(\bar{x}_t, \bar{u}_t^*)$, is minimized.

$$\begin{aligned} (K_t^*)_{t=0}^{T-1} &= \arg\min_{(K_t)_{t=0}^{T-1}} \text{Var}(\delta J_1^\pi), \\ \delta J_1^\pi &= \sum_{t=0}^T \bar{C}_t \delta x_t^l, \\ \delta x_{t+1}^l &= (A_t + B_t K_t)\delta x_t^l + \epsilon w_t. \end{aligned} \qquad (15)$$

We characterize the approximate closed loop policy below.

Proposition 2: Construct a closed loop policy

$$\pi_t^*(x_t) = \bar{u}_t^* + K_t^* \delta x_t, \qquad (16)$$

where $\bar{u}_t^*$ is the solution of the open-loop problem (14), and K_t^* is the solution of the closed loop problem (15). Let π^o be the optimal closed loop policy. Then, $|\tilde{J}^{\pi*} - \tilde{J}^{\pi^o}| = O(\epsilon^2)$.Furthermore, among all policies with nominal control action $\bar{u}_t^*$, the variance of the cost-to-go under policy π_t^*, is within $O(\epsilon^4)$ of the variance of the policy with the minimum variance.

Proof: Let $\bar{J}^{\pi^o}$ denote the nominal cost of the optimal policy π^o, where recall from before that the nominal cost is the closed loop cost when all the noise inputs are identically zero. Then, we have $\tilde{J}^{\pi^*} - \tilde{J}^{\pi^o} = \tilde{J}^{\pi^*} - \bar{J}^{\pi^*} + \bar{J}^{\pi^*} - \tilde{J}^{\pi^o} \leq \tilde{J}^{\pi^*} - \bar{J}^{\pi^*} + \bar{J}^{\pi^o} - \tilde{J}^{\pi^o}$.The inequality in the last line above is due the fact that $\bar{J}^{\pi^*} \leq \bar{J}^{\pi^o}$, since the nominal control corresponding to π^*, $\bar{u}_t^*$, is the minimizer for the nominal optimal control (open-loop) problem. Now, using Proposition 1 for the policies π^* and π^o, we have that $|\tilde{J}^{\pi^*} - \bar{J}^{\pi^*}| = O(\epsilon^2)$, and $|\tilde{J}^{\pi^o} - \bar{J}^{\pi^o}| = O(\epsilon^2)$. Also, by definition, we have $\tilde{J}^{\pi^o} \leq \tilde{J}^{\pi^*}$, i.e., the expected cost of π^o is lower than that of π^* since π^o minimizes the expected cost over all feedback policies. Note that this is different from u_t^* minimizing the nominal (open-loop) cost of the system. Then, since $\tilde{J}^{\pi^*} - \tilde{J}^{\pi^o} \geq 0$, using the inequality above, we obtain: $|\tilde{J}^{\pi^*} - \tilde{J}^{\pi^o}| \leq |\tilde{J}^{\pi^*} - \bar{J}^{\pi^*} + \bar{J}^{\pi^o} - \tilde{J}^{\pi^o}| \leq |\tilde{J}^{\pi^*} - \bar{J}^{\pi^*}| + |\bar{J}^{\pi^o} - \tilde{J}^{\pi^o}| = O(\epsilon^2)$.A similar argument holds for the variance as well. ■

The closed loop cost function in (15) can be written as (after noting $\delta u_t = K_t \delta x_t$):

$$\begin{aligned} \text{Var}(\delta J_1^\pi) &= \mathbb{E}[\sum_{t,\tau=0}^T [\delta x_t^l \; \delta u_t] \mathcal{Q}_{t,\tau} \begin{bmatrix} \delta x_t^l \\ \delta u_t \end{bmatrix}], \\ \mathcal{Q}_{t,\tau} &= \begin{pmatrix} C_t^{x\intercal} C_\tau^x & C_t^{x\intercal} C_\tau^u \\ C_t^{u\intercal} C_\tau^x & C_t^{u\intercal} C_\tau^u \end{pmatrix}, \end{aligned} \qquad (17)$$

and $C_T^u = 0$. This problem is non-standard: a standard LQR problem only has a single sum instead of the double sum over time above. Albeit convex, there is no standard solution to the problem above. Therefore, we solve a standard LQR problem as a surrogate and the effect is one of reducing the variance of the cost-to-go.

Approximate Closed Loop Problem. We solve the following LQR problem for suitably defined cost function weighting factors Q_t, R_t:

$$\min_{(\delta u_t)_{t=0}^{T}} \mathbb{E}[\sum_{t=0}^{T-1} \delta x_t' Q_t \delta x_t + \delta u_t' R_t \delta u_t + \delta x_T' Q_T \delta x_t],$$
$$\delta x_{t+1} = A_t \delta x_t + B_t \delta u_t + \epsilon w_t. \quad (18)$$

The solution to the above problem furnishes us a feedback gain $\hat{K}_t^*$ which we can use in the place of the true variance minimizing gain K_t^*.

Remark 3: Proposition 1 states that the expected cost-to-go of the problem is dominated by the nominal cost-to-go. Therefore, even an open-loop policy consisting of simply the nominal control action is within $O(\epsilon^2)$ of the optimal expected cost-to-go. However, the plan with the optimal feedback gain K_t^* is strictly better than the open-loop plan in that it has a lower variance in terms of the cost to go. Furthermore, solving the approximate closed-loop problem using the surrogate LQR problem, we expect a lower variance of the cost-to-go function due to feedback, which is borne out empirically (see Fig. 3).

IV. Decoupled Data-Based Control (D2C) Algorithm

In this section, we propose a novel decoupled data-based control (D2C) algorithm The three-step framework to solve the stochastic feedback control problem may be summarized as follows: 1) solve the open-loop optimization problem using gradient descent with a black box simulation model of the dynamics, 2) identify the linearized time-varying system from input-output experiment data, and 3) design an LQR controller for the identified LTV system.

A. Open-Loop Trajectory Optimization

A first-order gradient descent-based algorithm is proposed here for solving the open-loop optimization problem given in (14), where the underlying dynamic model is used as a blackbox, and the necessary gradient estimates are found from a sequence of input perturbation experiment data using standard least squares.

Denote the initial guess of the control sequence as $U^{(0)} = \{\bar{u}_t^{(0)}\}_{t=1}^T$, and the corresponding states $\mathcal{X}^{(0)} = \{\bar{x}_t^{(0)}\}_{t=1}^T$. The control policy is updated iteratively via

$$U^{(n+1)} = U^{(n)} - \gamma_n \nabla_U \bar{J}|_{\mathcal{X}^{(n)},U^{(n)}}, \quad (19)$$

where $U^{(n)} = \{\bar{u}_t^{(n)}\}_{t=1}^T$ denotes the control sequence in the n^{th} iteration, $\mathcal{X}^{(n)} = \{\bar{x}_t^{(n)}\}_{t=1}^T$ denotes the corresponding states, and γ_n is the time varying step size parameter. As $\bar{J}|_{\mathcal{X}^{(n)},U^{(n)}}$ is the expected cumulative cost under control sequence $U^{(n)}$ and corresponding states $\mathcal{X}^{(n)}$, the gradient vector is defined as:

$$\nabla_U \bar{J}|_{\mathcal{X}^{(n)},U^{(n)}} = \begin{pmatrix} \frac{\partial \bar{J}}{\partial u_1} & \frac{\partial \bar{J}}{\partial u_2} & \cdots & \frac{\partial \bar{J}}{\partial u_T} \end{pmatrix} |_{\mathcal{X}^{(n)},U^{(n)}}, \quad (20)$$

which is the gradient of the average cumulative cost w.r.t the control sequence after n iterations. The following paragraph elaborates on how to estimate the above gradient.

Let us define a rollout to be an episode in the simulation that starts from the initial settings to the end of the horizon with a control sequence. For each iteration, multiple rollouts are conducted sequentially with both the expected cumulative cost and the gradient vector updated iteratively after each rollout. During one iteration for the control sequence, the expected cumulative cost is calculated as

$$\bar{J}|^{j+1}_{\mathcal{X}^{(n)},U^{(n)}} = (1 - \frac{1}{j})\bar{J}|^{j}_{\mathcal{X}^{(n)},U^{(n)}} + \frac{1}{j}(J|_{\mathcal{X}^{j,(n)},U^{j,(n)}}), \quad (21)$$

where j denotes the j^{th} rollout within the current iteration process of control sequence. $\bar{J}|^{j}_{\mathcal{X}^{(n)},U^{(n)}}$ is the expected cumulative cost after j rollouts while $J|_{\mathcal{X}^{j,(n)},U^{j,(n)}}$ denotes the cost of the j^{th} rollout under control sequence $U^{j,(n)}$ and corresponding states $\mathcal{X}^{j,(n)}$. Note that $U^{j,(n)} = \{\bar{u}_t^{(n)} + \delta u_t^{j,(n)}\}_{t=1}^T$ where $\{\delta u_t^{j,(n)}\}_{t=1}^T$ is the zero-mean, i.i.d Gaussian noise added as perturbation to the control sequence $U^{(n)}$. Then the gradient vector is calculated in a similar sequential manner as

$$\nabla_U \bar{J}|^{j+1}_{\mathcal{X}^{(n)},U^{(n)}} = (1 - \frac{1}{j})\nabla_U \bar{J}|^{j}_{\mathcal{X}^{(n)},U^{(n)}} +$$
$$\frac{1}{j\sigma_{\delta u}}(J|_{\mathcal{X}^{j,(n)},U^{j,(n)}} - \bar{J}|^{j+1}_{\mathcal{X}^{(n)},U^{(n)}})(U^{j,(n)} - U^{(n)}), \quad (22)$$

where $\sigma_{\delta u}$ is the variance of the control perturbation and $\nabla_U \bar{J}|^{j+1}_{\mathcal{X}^{(n)},U^{(n)}}$ denotes the gradient vector after j rollouts. After m rollouts, the control sequence is updated by equation (19) in which $\nabla_U \bar{J}|_{\mathcal{X}^{(n)},U^{(n)}}$ is estimated by $\nabla_U \bar{J}|^{m}_{\mathcal{X}^{(n)},U^{(n)}}$, and the procedure repeated till convergence.

B. Linear Time-Varying System Identification

The closed loop control design specified in (15) requires the knowledge of the parameters $A_t, B_t, 1 \leq t \leq T$, of the perturbed linear system. We propose a linear time variant (LTV) system identification procedure to estimate these parameters.

First start from perturbed linear system given by equation (18). Using only first order information, we estimate the system parameters A_t, B_t in the LTV form: $\delta x_{t+1} = \hat{A}_t \delta x_t + \hat{B}_t \delta u_t$.Now write out their components for each iteration in vector form as,

$$Y = [\delta x_{t+1}^0 \delta x_{t+1}^1 \cdots \delta x_{t+1}^{N-1}], \quad X = \begin{bmatrix} \delta x_t^0 & \cdots & \delta x_t^{N-1} \\ \delta u_t^0 & \cdots & \delta u_t^{N-1} \end{bmatrix},$$
$$Y = [\hat{A}_t \mid \hat{B}_t] X, \quad (23)$$

where N is the total iteration number. δx_{t+1}^n denotes the output state deviation, δx_t^n denotes the input state perturbations and δu_t^n denotes the input control perturbations at time t of the n^{th} iteration. All the perturbations are zero-mean, i.i.d, Gaussian random vectors whose covariance matrix is σI where I is the identity matrix and σ is a scalar. Note that here one iteration only has one rollout. Using the least squares method $\hat{A}_t$ and $\hat{B}_t$ can be calculated as follows:

$$[\hat{A}_t \mid \hat{B}_t] = YX'(XX')^{-1}, \quad (24)$$

The calculation procedure can also be done sequentially using recursive least squares. It is highly amenable to parallelization and is memory efficient.

C. Closed-Loop Control Design

Given the parameter estimate of the perturbed linear system, we solve the closed-loop control problem given in (18). This is a standard LQR problem. By solving the Riccati equation, we can get the closed-loop optimal feedback gain K_t^*. The details of the design are standard and are omitted here.

```
Algorithm 1: D2C Algorithm
1) Solve the deterministic open-loop optimization problem for optimal open loop nominal control sequence and trajectory $(\{\bar{u}_t^*\}_{t=1}^T, \{\bar{x}_t^*\}_{t=1}^T)$ using gradient descent method (Section IV-A).
2) Identify the LTV system $(\hat{A}_t, \hat{B}_t)$ via least square estimation (Section IV-B).
3) Solve the Riccati equations using estimated LTV system equation for feedback gain $\{K_t^*\}_{t=1}^T$ (Section IV-C).
4) Set $t = 1$, given initial state $x_1 = \bar{x}_1^*$ and state deviation $\delta x_1 = 0$.
while $t \leq T$ do
    $$\begin{aligned} u_t &= \bar{u}_t^* + K_t^* \delta x_t, \\ x_{t+1} &= f(x_t, u_t) + \epsilon w_t, \\ \delta x_{t+1} &= x_{t+1} - \bar{x}_{t+1}^* \end{aligned} \qquad (25)$$
    $t = t + 1$.
end while
```

D. Convergence of the D2C Algorithm

The Decoupled Data-Based Control (D2C) Algorithm is summarized in Algorithm 1. In the following, we provide a convergence analysis of the open-loop design and LTV identification parts of the D2C algorithm.

Proposition 3: Gradient Descent. Let the gradient $\nabla \bar{J}$ be Lipshitz continuous, i.e., $||\nabla \bar{J}(U_1) - \nabla \bar{J}(U_2)||| \leq L||U_1 - U_2||$, for some $L < \infty$, and the step size parameters in (19) satisfy $\sum_t \gamma_t = \infty$, and $\sum_t \gamma_t^2 < \infty$. Given $E||U^{(n)}||^2 < \infty$, the iterates in (19), $U^{(n)}$ almost surely converge to a set S, where $\nabla \bar{J} = 0$ on the set S.

Proof: The sample paths of the stochastic gradient descent algorithm (19) almost surely can be approximated asymptotically by the ODE $\dot{U} = -\nabla \bar{J}(U)$, due to the above assumptions and the fact that (22) is an unbiased estimator of the true gradient, and the convergence of the algorithm is almost surely determined by the limit points of the ODE (this follows from the so-called "ODE method" approach to Stochastic Approximation algorithms [15]). To characterize the limit points, choose the Lyapunov function $\bar{J}$ for the above ODE, then $\dot{\bar{J}} = -\nabla \bar{J} \cdot \nabla \bar{J} \leq 0$. Hence, $\bar{J}$ converges to a set S where $\nabla \bar{J} = 0$, proving the result. ■

Complexity of Stochastic Gradient Descent. The complexity of the gradient descent algorithm, per gradient descent step, is $O(pT)$, where p is the number of inputs, and T is the control horizon. However, due to the nonlinear cost function J, the convergence guarantees are only asymptotic, and the complexity of the whole algorithm is $O(K_\infty pT)$ where K_∞ is the steps to convergence which can vary with the initial guess and the learning parameter schedule.

Proposition 4: Convergence of LTV identification. The least squares estimates in (24), $[\hat{A}_t, \hat{B}_t] \rightarrow [A_t, B_t]$, as $N \rightarrow \infty$ in the mean square sense.

Proof: Without loss of generality, let the $cov(\delta x_t^{(i)}) = I_n$, and $cov(\delta u_t^{(i)}) = I_p$, where I_q denotes a $q \times q$ identity matrix. The least squares solution in (24) can be written as $[\hat{A}_t, \hat{B}_t] = Y^{(N)} X^{(N)\intercal} (X^{(N)} X^{(N)\intercal})^{-1}$, where $Y^{(N)} = [\delta x_{t+1}^{(1)} ... \delta x_{t+1}^{(N)}]$, $X^{(N)} = [\delta \bar{x}_t^{(1)}, \cdots \delta \bar{x}_t^{(N)}]$, and $\delta \bar{x}_t^{(i)} = [\delta x_t^{(i)}, \delta u_t^{(i)}]^\intercal$.
Using the Law of Large numbers, it is relatively straightforward to see that $\frac{1}{N} X^{(N)} X^{(N)\intercal} \rightarrow I_{n+p}$ as $N \rightarrow \infty$ almost surely. Let the noise after N steps be $V^{(N)}$ (which is zero mean with covariance I_N), i.e., $Y^{(N)} = [A_t, B_t] X^{(N)} + V^{(N)}$. Then, the error in the LS estimate is $V^{(N)} X^{(N)\intercal} (X^{(N)} X^{(N)\intercal})^{-1}$, which is zero mean and has covariance $(X^{(N)} X^{(N)\intercal})^{-1} \rightarrow \frac{1}{N} I_{n+p} \rightarrow 0$ as $N \rightarrow \infty$. Therefore, the LS estimate converges in mean square sense to the true parameter values. ■

Complexity of LTV identification. The key to the complexity of the above identification is how quickly does the sample covariance $\frac{1}{N} X^{(N)} X^{(N)\intercal} \rightarrow I_{n+p}$. It can be shown that $N = O(n + p)$ samples are good enough to get close to the limit with a very high probability if $n + p$ is large enough (Theorem 4.7.1 in [16]). We do not go into more details here due to space constraints but this is borne out by our empirical evidence. Thus, the complexity of the LTV idenitification is $O(n + p)T$ since we have T such identification steps.

Complexity of the D2C algorithm. The complexity of the open-loop design is $O(K_\infty pT)$ while that of the LTV identification, and hence the closed-loop design, is $O(n + p)T$. However, in general, the steps to convergence, $K_\infty >> n, p$, and thus, the training time of the D2C algorithm is overwhelmingly dominated by the open-loop part, an observation that is borne out by our empirical results that follow (see Table I).

V. Empirical Results

In this section, we compare the D2C approach with the well-known deep reinforcement learning algorithm - Deep Deterministic Policy Gradient (DDPG) [17]. For comparison, we evaluate both methods in the following three aspects: 1) Efficiency in training - the amount of time and storage required to achieve a desired task, 2) Robustness to noise - the deviation from the predefined task due to random noise in the process in the testing stage, and 3) Ease of training - the challenges involved in training with either of the data-based approaches. We tested our method with four benchmark tasks, all implemented in MuJoCo simulator [18]: Inverted pendulum, Cartpole, 3-link swimmer and 6-link swimmer (please see [19] for details). The state space ranges from 2 to 26 dimensions while the control space ranges from 1 to 6 dimensions in these examples. An off-the-shelf implementation of DDPG provided by *Keras-RL* [20] library has been customized for our simulations. For fair comparison, ‘episodic reward/cost fraction’ is considered with both methods. It is defined as the fraction of reward obtained in an episode during training w.r.t the nominal episodic reward (converged reward).

A. Performance Comparison

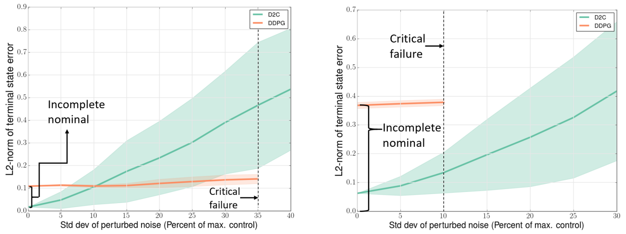
Training Efficiency: One way of measuring efficiency is to collate the times taken for the episodic cost (or reward) to converge during training. Plots in Fig. 1 show the training process with both methods on the systems considered. Each plot shows the training curve of one experiment. The curve marked as original is the actual training curve reflecting the original reward data. The one marked as filtered is the curve after smoothing out the spikes to show a better view of the reward trend as the training goes. Table I delineates the times taken for training respectively. As the system identification and feedback gain calculation in the case of D2C take only a small portion of time, the total time comparison in (Table I) shows that D2C learns the optimal policy substantially faster than DDPG, and hence, has a better training efficiency.

Robustness to noise: However, from plots in Fig. 2, it is evident that the performance of D2C is on par with or better than DDPG up to a certain level of noise. It may also be noted that the error variance in the D2C method increases abruptly when the noise level is higher than a threshold and drives the system too far away from the nominal trajectory that the LQR controller cannot fix it. This could be considered as a drawback for D2C. However, it must be noted that the range of noise levels (up until 100 % of the maximum control signal) that we are considering here is far beyond what is typically encountered in practical scenarios. Moreover, it must also be noted that the point at which the DDPG performance overtakes that of D2C, the performance of both methods is poor from the viewpoint of attaining the given task. In Fig. 3, we compare the episodic cost during testing between the open-loop policy applied along and the closed-loop policy of D2C. As expected, the closed-loop performance is much better than the open-loop performance albeit the closed-loop design is only a very small fraction of the training cost (see Table I).

Ease of training: To elucidate the ease of training from an empirical perspective, the exploration noise that is required for training in DDPG mandates the system to operate with a shorter time-step than a threshold, beyond which the simulation fails due to an unbearable magnitude of control actions into the system. For this, we train both the swimmers in one such case (with $\Delta t = 0.01$ sec) till it fails and execute the intermediate policy. Fig. 4 shows the plot in the testing-stage with both methods. It is evident from the terminal state mean-squared error at zero noise level that the nominal trajectory of DDPG is incomplete and its policy failed to reach the goal. The effect is more pronounced in the higher-dimensional 6-link swimmer system (Fig. 4b), where the DDPG’s policy can be deemed to be downright broken. Note, from Table I, that the systems have been trained with DDPG for a time that is more than thrice with the 3-link swimmer and 4 times with the 6-link swimmer. Moreover, the starred entries in Table I indicate that DDPG failed to converge. On the other hand, under the same conditions, the seamless training of D2C results in a working policy with even greater data-efficiency.

TABLE I: Simulation parameters and training outcomes

System	Steps per episode	Time-step (in sec.)	Training time (in sec.)		
			D2C		DDPG
			Open-loop	Closed-loop	
Inverted Pendulum	30	0.1	12.9	< 0.1	2261.15
Cart pole	30	0.1	15.0	1.33	6306.7
3-link Swimmer	1600 800	0.005 0.01	7861.0 4001.0	13.1 4.6	38833.64 13280.7*
6-link Swimmer	1500 900	0.006 0.01	9489.3 3585.4	26.5 16.4	88160 15797.2*
Fish	1200	0.005	6011.2	75.6	124367.6



(a) 3-link swimmer

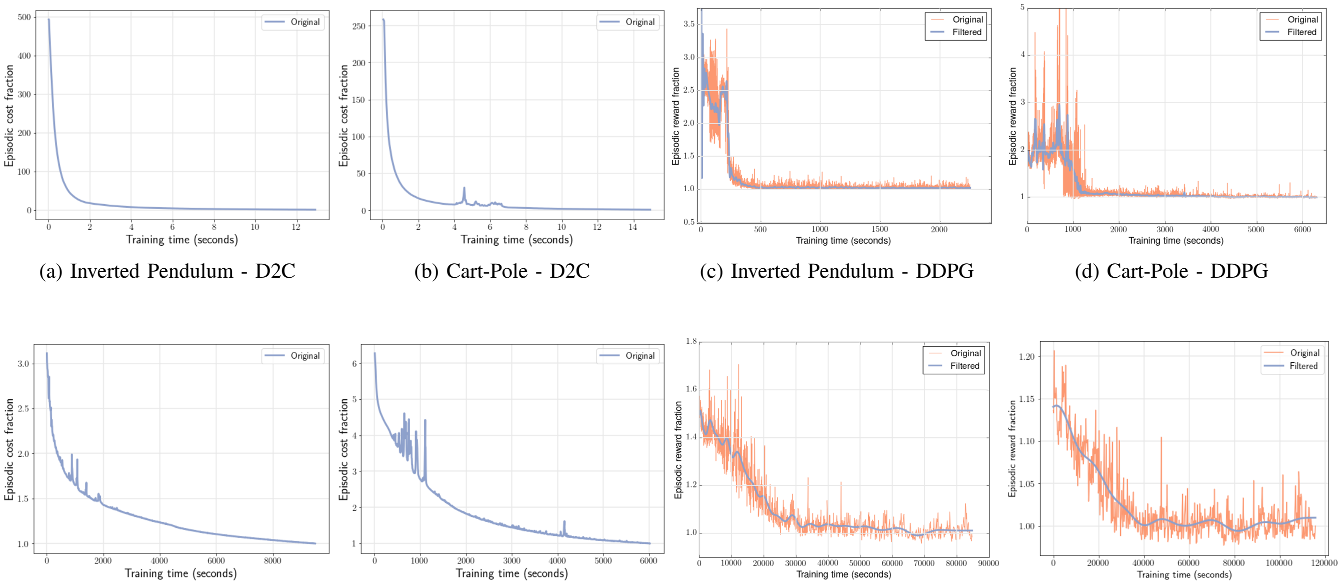
(b) 6-link swimmer

Fig. 4: D2C vs DDPG at $\Delta t = 0.01s$

VI. CONCLUSIONS

In this paper, we proposed a near-optimal control algorithm under fully observed conditions and showed that our method is able to scale-up to higher dimensional state-space without any knowledge about the system model. Due to the sequential calculation used in the open-loop optimization and the system identification, D2C is highly memory efficient and also convenient for parallelization. We tested its performance and compared them with a state-of-the-art deep RL technique - DDPG. From the results, our method has significant advantages over DDPG in terms of training efficiency and ease of training. This primarily stems from the far smaller parameter space, essentially open-loop sequences, that D2C searches over, as opposed to a complex parameterization like Deep Neural Nets for DDPG. The robustness of D2C is also better/ comparable in most cases but has scope for further improvement by employing more sophisticated feedback design and ensuring that the data efficiency is not compromised. We also believe further drastic reduction in the planning time can be achieved by parallelization and a more sophisticated parametrization and solution of the open-loop problem. Future work will focus on these aspects of the D2C technique.

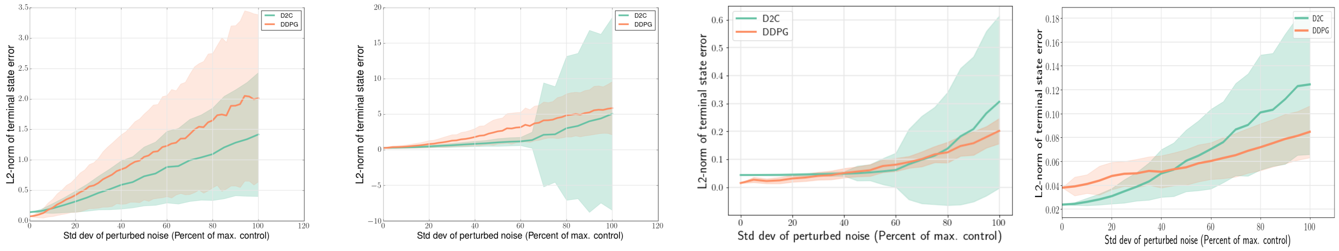
It is evident from the simulations that methods such as D2C are able to achieve their goals accurately whereas DDPG consumes an inordinate amount of time in ‘fine-tuning’ their behavior towards the goal. However, we also note that, by doing this, DDPG is tentatively exploring over the entire



(a) Inverted Pendulum - D2C (b) Cart-Pole - D2C (c) Inverted Pendulum - DDPG (d) Cart-Pole - DDPG

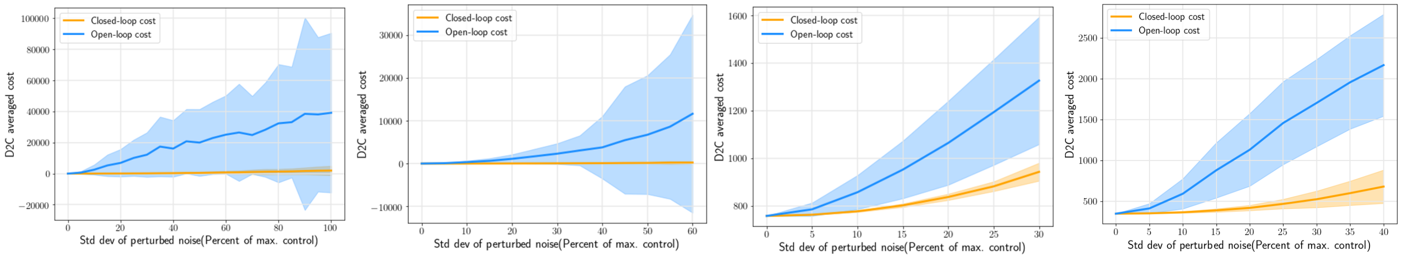
(e) 6-link Swimmer - D2C (f) Fish - D2C (g) 6-link Swimmer - DDPG (h) Fish

Fig. 1: Episodic reward fraction vs time taken during training



(a) Inverted Pendulum (b) Cart-Pole (c) 6-link Swimmer (d) Fish

Fig. 2: Terminal MSE vs noise level during testing



(a) Inverted Pendulum (b) Cart-Pole (c) 6-link Swimmer (d) Fish

Fig. 3: Averaged episodic reward fraction vs noise level during testing for D2C

state-space and can result in a better generic policy. Nevertheless, we hope that our approach signifies the potential of decoupling-based approaches such as D2C in a reinforcement learning paradigm and recognizes the need for more hybrid approaches that complement the merits of each.

References

[1] P. R. Kumar and P. Varaiya, *Stochastic systems: Estimation, identification, and adaptive control*. SIAM, 2015, vol. 75.

[2] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, “Mastering the game of go with deep neural networks and tree search,” *nature*, vol. 529, no. 7587, p. 484, 2016.

[3] 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.

[4] S. Levine, C. Finn, T. Darrell, and P. Abbeel, “End-to-end training of deep visuomotor policies,” *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[5] W. Yuhuai, M. Elman, L. Shun, G. Roger, and B. Jimmy, “Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation,” *arXiv:1708.05144*, 2017.

[6] S. John, L. Sergey, M. Philipp, J. Michael I., and A. Pieter, “Trust region policy optimization,” *arXiv:1502.05477*, 2017.

[7] 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.

[8] M. Falcone, “Recent results in the approximation of nonlinear optimal control problems,” in *Large-Scale Scientific Computing LSSC*, 2013.

[9] D. Jacobsen and D. Mayne, *Differential Dynamic Programming*. Elsevier, 1970.

[10] W. Li and E. Todorov, “Iterative linearization methods for approximately optimal control and estimation of non-linear stochastic system,” *International Journal of Control*, vol. 80, no. 9, pp. 1439–1453, 2007.

[11] D. Bertsekas, *Dynamic programming and optimal control*. Athena scientific Belmont, MA, 2012, vol. 2.

[12] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[13] A. Rajeswaran, K. Lowrey, E. V. Todorov, and S. M. Kakade, “Towards generalization and simplicity in continuous control,” in *Advances in Neural Information Processing Systems*, 2017, pp. 6550–6561.

[14] D. Yu, M. Rafieisakhaei, and S. Chakravorty, “Stochastic feedback control of systems with unknown nonlinear dynamics,” in *56th IEEE Conference on Decision and Control(CDC)*, 2017.

[15] H. Kushner and G. Yin, *Stochastic Approximation and Recursive Algorithms and applications*. Springer, NY, 2003.

[16] R. Versyhnin, *High Dimensional Probability: An Introduction with Application to Data Science*. Cambridge University Press, Cambridge, UK, 2018.

[17] T. Lillicrap *et al.*, “Continuous control with deep reinforcement learning,” in *Proc. ICLR*, 2016.

[18] T. Emanuel, E. Tom, and Y. Tassa, “Mujoco: A physics engine for model-based control,” *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012.

[19] T. Yuval and *et al.*, “Deepmind control suite,” *arXiv:1801.00690*, 2018.

[20] M. Plappert, “keras-rl,” https://github.com/keras-rl/keras-rl, 2016.

APPENDIX

A. Proof of Lemma 1

Proof: We proceed by induction. The first general instance of the recursion occurs at $t = 3$. It can be shown that: $\delta x_3 = \underbrace{(\bar{A}_2\bar{A}_1(\epsilon w_0) + \bar{A}_2(\epsilon w_1) + \epsilon w_2)}_{\delta x_3^l} + \underbrace{\{\bar{A}_2\bar{S}_1(\epsilon w_0) + \bar{S}_2(\bar{A}_1(\epsilon w_0) + \epsilon w_1 + \bar{S}_1(\epsilon w_0))\}}_{\bar{\bar{S}}_3}$. Noting that $\bar{S}_1(.)$ and $\bar{S}_2(.)$ are second and higher order terms, it follows that $\bar{\bar{S}}_3$ is $O(\epsilon^2)$. Suppose now that $\delta x_t = \delta x_t^l + \bar{\bar{S}}_t$ where $\bar{\bar{S}}_t$ is $O(\epsilon^2)$. Then: $\delta x_{t+1} = \bar{A}_{t+1}(\delta x_t^l + \bar{\bar{S}}_t) + \epsilon w_t + \bar{S}_{t+1}(\delta x_t) = \underbrace{(\bar{A}_{t+1}\delta x_t^l + \epsilon w_t)}_{\delta x_{t+1}^l} + \underbrace{\{\bar{A}_{t+1}\bar{\bar{S}}_t + \bar{S}_{t+1}(\delta x_t)\}}_{\bar{\bar{S}}_{t+1}}$. Noting that $\bar{S}_{t+1}$ is $O(\epsilon^2)$ and $\bar{\bar{S}}_{t+1}$ is $O(\epsilon^2)$ by assumption, the result follows. ■

B. Proof of Lemma 2

Lemma 2: Let δJ_1^π, δJ_2^π be as defined in (11). Then, $\mathbb{E}[\delta J_1^\pi \delta J_2^\pi]$ is an $O(\epsilon^4)$ function.

Proof: In the following, we suppress the explicit dependence on π for δJ_1^π and δJ_2^π for convenience. Recall that $\delta J_1 = \sum_{t=0}^T C_t^x \delta x_t^l$, and $\delta J_2 = \sum_{t=0}^T \bar{H}_t(\delta x_t) + C_t^x \bar{\bar{S}}_t$. As before, let us consider $\bar{\bar{S}}_3$. We have that $\bar{\bar{S}}_3 = \bar{A}_2\bar{S}_1(\epsilon w_0) + \bar{S}_2(\bar{A}_1(\epsilon w_0) + \epsilon w_1 + \bar{S}_1(\epsilon w_0))$. Note that $\epsilon w_0 = \delta x_1^l$ and $\bar{A}_1(\epsilon w_0) + \epsilon w_1 = \delta x_2^l$. Then, it follows that: $\bar{\bar{S}}_3 = \bar{A}_2 \begin{bmatrix} \delta x_1^{l\intercal} \bar{S}_{1,1}^{(2)} \delta x_1^l \\ \vdots \\ \delta x_1^{l\intercal} \bar{S}_{1,n}^{(2)} \delta x_1^l, \end{bmatrix} + \begin{bmatrix} \delta x_2^{l\intercal} \bar{S}_{2,1}^{(2)} \delta x_2^l \\ \vdots \\ \delta x_2^{l\intercal} \bar{S}_{2,n}^{(2)} \delta x_2^l, \end{bmatrix} + O(\epsilon^3)$, where the Hessian matrices $\{\bar{S}_{t,j}^{(2)}, j = 1, 2 \cdots n\}$ correspond to the second order term in the Taylor expansion of the n dimensional vector valued function $\bar{S}_t(.)$. A similar observation holds for $\bar{H}_3(\delta x_3)$ in that: $\bar{H}_3(\delta x_3) = \delta x_3^{l\intercal} \bar{H}_3^{(2)} \delta x_3^l + O(\epsilon^3)$, where $\bar{H}_t^{(2)}$ represents the Hessian matrix corresponding to the second order term in the Taylor expansion of the scalar valued function $\bar{H}_t(.)$. Therefore, from the above equations, it follows that we may write: $\bar{H}_t(\delta x_t) + C_t^x \bar{\bar{S}}_t = \sum_{\tau=0}^t \delta x_\tau^{l\intercal} Q_{t,\tau} \delta x_\tau^l + O(\epsilon^3)$, for suitably defined matrix coefficients $Q_{t,\tau}$. Therefore, it follows that $\delta J_2 = \sum_{t=0}^T \bar{H}_t(\delta x_t) + C_t^x \bar{\bar{S}}_t = \sum_{\tau=0}^T \delta x_\tau^{l\intercal} \bar{Q}_{T,\tau} \delta x_\tau^l + O(\epsilon^3)$, for suitably defined matrices $\bar{Q}_{T,\tau}$. Hence, $\delta J_1 \delta J_2 = \sum_{t,\tau=0}^T C_t^x(\delta x_t^l) \delta x_\tau^{l\intercal} \bar{Q}_{T,\tau} \delta x_\tau^l + O(\epsilon^4)$. Taking expectations on both sides: $\mathbb{E}[\delta J_1 \delta J_2] = \sum_{t,\tau=0}^T \mathbb{E}[C_t^x \delta x_t^l \delta x_\tau^{l\intercal} \bar{Q}_{T,\tau} \delta x_\tau^l] + O(\epsilon^4)$. Break $\delta x_t^l = (\delta x_t^l - \delta x_\tau^l) + \delta x_\tau^l$, assuming $\tau < t$. Then, it follows that: $\mathbb{E}[C_t^x \delta x_t^l \delta x_\tau^{l\intercal} \bar{Q}_{T,\tau} \delta x_\tau^l] = \mathbb{E}[C_t^x \delta x_\tau^l \delta x_\tau^{l\intercal} \bar{Q}_{T,\tau} \delta x_\tau^l]$, due to the independence of $\delta x_t^l - \delta x_\tau^l$ from δx_τ^l, and the fact that $\mathbb{E}[\delta x_t^l - \delta x_\tau^l] = 0$. Note that we may write $\delta x_\tau^l = \epsilon[\beta_{\tau-1}\omega_{\tau-1} + \cdots + \beta_0\omega_0]$, for suitably defined co-efficients $\beta_0, \beta_1 \cdots$. Therefore, it follows that: $\delta x_\tau^{l'} \bar{Q}_{T,\tau} \delta x_\tau^l = \epsilon^2 \sum_{k,l=0}^\tau \omega_k' \tilde{Q}_{kl}^{T,\tau} \omega_l$, for suitably defined matrices $\tilde{Q}_{kl}^{T,\tau}$. Therefore, it follows that $C_t^x \delta x_\tau^l \delta x_\tau^{l\intercal} \bar{Q}_{T,\tau} \delta x_\tau^l = \epsilon^3 \sum_{t_1,t_2,t_3=0}^{\tau-1} \sum_{i,j,k=1}^p \omega_{t_1}^i \omega_{t_2}^j \omega_{t_3}^k \alpha_{t_1,t_2,t_3}^{i,j,k}$, for suitably defined constants $\alpha_{t_1,t_2,t_3}^{i,j,k}$, where ω_t^i represents the i^{th} input noise term at time t and p is the total number of inputs to the system. Hence, $\mathbb{E}[C_t^x \delta x_\tau^l \delta x_\tau^{l\intercal} \bar{Q}_{T,\tau} \delta x_\tau^l] = \epsilon^3 \sum_{t_1,t_2,t_3=0}^{\tau-1} \sum_{i,j,k=1}^p \mathbb{E}[\omega_{t_1}^i \omega_{t_2}^j \omega_{t_3}^k] \alpha_{t_1,t_2,t_3}^{i,j,k}$. Note now that $\mathbb{E}[\omega_{t_1}^i \omega_{t_2}^j \omega_{t_3}^k] = 0$ unless $t_1 = t_2 = t_3$, regardless of i, j, k, since the noise is assumed to be white in time. If the input channels are uncorrelated, and the input noise standard Gaussian, it follows that $\mathbb{E}[\omega_s^i \omega_s^j \omega_s^k] = 0$ regardless of i, j, k since odd moments of a zero mean Gaussian variable are zero. Next, let us consider the case that the input channels are correlated. Then $\omega_s = \sqrt{W}\nu$, where W is the covariance of ω_s and ν is a Gaussian input vector that has identity covariance. Then, it follows that: $\mathbb{E}[\omega_s^i \omega_s^j \omega_s^k] = \sum_{i_1,i_2,i_3=1}^p \mathbb{E}[\nu_{i_1}\nu_{i_2}\nu_{i_3}] d_{i_1,i_2,i_3}$, for suitably defined coefficients d_{i_1,i_2,i_3}. However, due to our previous argument, $\mathbb{E}[\nu_{i_1}\nu_{i_2}\nu_{i_3}] = 0$ due to the noise input ν being spatially uncorrelated and Gaussian. Therefore, from the above above argument it follows that: $\mathbb{E}[C_t^x \delta x_\tau^l \delta x_\tau^{l\intercal} \bar{Q}_{T,\tau} \delta x_\tau^l] = 0$. Therefore, using the above fact, it follows that $\mathbb{E}[\delta J_1 \delta J_2] = O(\epsilon^4)$, thereby proving the result when $t > \tau$. An analogous argument as above can be repeated for the case when $\tau > t$. ■