2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

Data-based Control of Partially-Observed Robotic Systems

Ran Wang[1*], Raman Goyal[1*], Suman Chakravorty[1], and Robert E. Skelton[1]

2021 IEEE International Conference on Robotics and Automation (ICRA) | 978-1-7281-9077-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICRA48506.2021.9561001

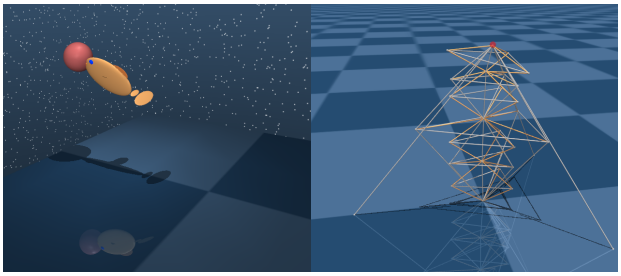
***Abstract*— This paper presents a data-based approach to control robotic systems with partially-observed feedback. First, an open-loop optimization problem is solved to generate the nominal trajectory and then a linear time-varying Autoregressive–Moving-Average (ARMA) model of the system is calculated along the trajectory from the output measurement data. The system is then described in information state, which contains input-output information of the past few steps. Finally, a feedback gain which is calculated by solving a specific LQG problem along the nominal trajectory. The separate design of the open-loop and the closed-loop problem is used following the Decoupled Data-based Control (D2C) approach. Simulation results are also shown for complex models with fluid-structure interaction in the presence of both process and measurement noise.**

***Index Terms*— Robotic path planning, learning and control, data-based control, partial-state feedback.**

I. Introduction

The classical literature on adaptively controlling unknown (stochastic) dynamical system provides rigorous analysis about the asymptotic performance and stability of the closed-loop system [1] [2], but it is mostly limited to linear systems or systems with finite state and control space. The optimal control of a possibly unknown nonlinear dynamical system with continuous state and action space is a significantly more challenging problem. The 'curse of dimensionality' associated with solving dynamic programming which is required for computing the optimal control, makes solving such problems computationally intractable for complex high-order systems. Moreover, *Learning to control* problems where the model of the system is unknown or too complex, also suffers from this computational complexity issue.

The past several years have seen significant progress in deep neural networks based reinforcement learning approaches for controlling unknown dynamical systems due to our inability to create accurate dynamical models for large complex systems [3], [4]. These approaches have been successfully used in many areas like playing games [5], locomotion [6], and robotic hand manipulation [7]. A number of new algorithms that show promising performance are proposed [8], [9], [10], 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 intensive. The training time for such algorithms is typically really large. Moreover, high variance and reproducibility issues on the performance are also reported [11].



(a) **Fish (27 States, 11 Observations)** (b) **Tensegrity Robotic Arm (150 States, 24 Observations)**

Fig. 1: Models simulated in MuJoCo in their terminal states.

We recently proposed a novel decoupled data-based control (D2C) algorithm for learning to control an unknown nonlinear dynamical system [12]. Our approach introduced a rigorous decoupling of the open-loop (planning) problem from the closed-loop (feedback control) problem. This decoupling allows for a highly sample 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 around the nominal trajectory and design an LQR controller for this linearized system. We have shown that the performance of the D2C algorithm is approximately optimal, in the sense that the decoupled design is near-optimal to second order in a suitably defined noise parameter [12]. In this paper, we propose an extension of the D2C algorithm which only needs partial observations, and not full state information. Also, the formulation only uses input-output data instead of state perturbation, which makes the extended algorithm more feasible for complex systems. The main idea is to generate Autoregressive–Moving-Average (ARMA) models along the nominal trajectory using input-output perturbation data and then writing the augmented linear time-varying system using the *information state* which is comprised of past several measurements and control inputs. This allows for the development of a *specific LQG* controller.

The ARMA models are written as the combination of autoregressive (AR) model and moving-average (MA) model and are mostly used in the statistical analysis of time series [13]. The autoregressive part contains the sum of past values of the variable itself ($z_t = \sum_{i=1}^{p} \alpha_{t-i} z_{t-i}$) and the moving-average part contains the accumulation of past few noise error terms. In the context of this paper, the moving-average part constitutes the input excitation added as i.i.d

* Equal Contribution. [1]R. Wang, R. Goyal, S. Chakravorty and R. E. Skelton are with the Department of Aerospace Engineering, Texas A&M University, Texas, USA. {rwang0417, ramaniitrgoyal92, schakrav, bobskelton}@tamu.edu. The codes can be found in our GitHub repository: https://github.com/rwang0417/icra2021_pod2c

noise ($z_t = \sum_{i=1}^{q} \beta_{t-i} u_{t-i}$). The ARMA models have been extensively used for modeling and forecasting in the field of statistics, economics, and finance [14]. Researchers have also used the nonlinear version of these input-output parametric models, known as, Nonlinear ARMA (NARMA) models for control of nonlinear systems for both deterministic and stochastic cases [15], [16]. Time-varying linear models were also proposed for input-output modeling of nonlinear functions by replacing it with a family of piece-wise linear functions [17]. Chemical industries have also been using this modeling idea for nonlinear model predictive control [18]. In this paper, we use the ARMA modeling technique to identify the linear state-space model around the nominal trajectory and show the condition to exactly match any linear state-space model.

The paper is organized as follows: Section II provides the optimal control problem formulation for the stochastic nonlinear system. Section III provides the condition on the ARMA model to exactly match the data generated using a linear system and Section IV gives the detailed D2C algorithm for the case of partially-observed states with measurement noise. Finally, the above-mentioned data-based control algorithm is used to control complex dynamical systems, including challenging cases of hard-to-model soft contact constraints and dynamic fluid interactions, with imperfect measurement of only a few of the states.

II. Problem Formulation

Let us write the non-linear dynamics in discrete time, noise perturbed state space form as follows:

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

where x_t is the state, u_t is the control, w_t is a white noise perturbation to the system, and ϵ is a small parameter that modulates the noise in the system, and the observation model is assumed as:

$$z_t = h(x_t) + v_t, \tag{2}$$

where v_t is the measurement noise. Suppose now that we have a finite horizon objective function:

$$J(x_0) = E\left[\sum_{t=0}^{T-1} c(x_t, u_t) + \phi(x_T)\right], \tag{3}$$

where $c(x, u)$ denotes a running incremental cost, $\phi(x)$ denotes a terminal cost function and $E[\cdot]$ is an expectation operator taken over the sample paths of the system. For the case of partially observed states, the objective is to design a feedback policy $u_t(Z_t)$ that minimizes the following cost function:

$$\min_{u_t(\cdot)} E\left[\sum_{t=0}^{T-1} c^Z(Z_t, u_t) + \phi^Z(Z_T)\right], \tag{4}$$

where Z_t is defined as the *information state* comprised of the past few observations z_i, and control inputs u_i, $Z_t = (z_t, z_{t-1}, \cdots, z_{t-q+1}, u_{t-1}, \cdots, u_{t-q+1})$. Notice that the cost function is chosen such that $c(x_t, u_t) = c^Z(Z_t, u_t)$ and $\phi(x_T) = \phi^Z(Z_T)$, in terms of the nominal information state Z_t defined above.

III. Partially-Observed Decoupled Data Based Control (D2C) Algorithm

In this section, we propose an extension to the so-called decoupled data-based control (D2C) algorithm [12]. The D2C algorithm is a highly data-efficient Reinforcement Learning (RL) method that has shown to be much superior to the state of the art RL algorithms such as the Deep Deterministic Policy Gradient (DDPG) in terms of data efficiency and training stability while retaining similar or better performance. In the following, we give a very brief introduction to the D2C algorithm (refer [12][19] for more details) and details the extension that allows for the generation of the near-optimal output feedback policy, i.e., the feedback as a function of past few observations. The main observation for the extension is that the partially observed case might be treated similarly to the fully observed case by noting that the information state in the problem comprises of past few observations and past few control inputs.

Remark 3.1: Please refer to Section IV for ARMA model design which allows for writing information state with finite past observations and control inputs.

Let us form a notion of nominal belief evolution by assuming $w_t, v_t = 0$ for all t, say $\bar{Z}_t = (\bar{z}_t, \bar{z}_{t-1}, \cdots, \bar{z}_{t-q+1}, \bar{u}_{t-1}, \cdots, \bar{u}_{t-q+1})$, and the deviations from the nominal belief state, say $\delta Z_t = (\delta z_t, \delta z_{t-1}, \cdots, \delta z_{t-q+1}, \delta u_{t-1}, \cdots, \delta u_{t-q+1})$, where $\delta z_t = z_t - \bar{z}_t$ and $\delta u_t = u_t - \bar{u}_t$ are the deviation from the nominal observation at time t. Again, assuming sufficient smoothness, any feedback law for such a problem can be represented as $\pi_t(Z_t) = \bar{u}_t + K_t \delta Z_t + S_t(\delta Z_t)$, where $\bar{u}_t$ is the nominal control sequence arising from $\pi_t(\cdot)$, K_t is the linear feedback term and $S_t(\cdot)$ are the higher order terms in the feedback law.

The D2C algorithm then proposes a 3 step procedure to approximate the solution to the above problem. First, a noiseless open-loop optimization problem is solved to find an optimal control sequence, $\bar{U}_T^*$, subject to the zero noise nominal dynamics: $x_{t+1} = f(x_t) + g(x_t)u_t$. Second, from rollouts, we estimate a linear perturbation model for the information state $\delta Z_t = A_{t-1}\delta Z_{t-1} + B_{t-1}\delta u_{t-1}$ around the nominal to obtain Linear-Time Varying (LTV) system. Third, an LQG controller for the above time varying linear system is designed whose time varying gain is given by K_t. Finally, the control applied to the system is given by $u_t = \bar{u}_t^* - K_t \delta Z_t$.

The following subsections provide details for each of the above-mentioned steps of the D2C algorithm.

A. Open Loop Trajectory Optimization

A first-order gradient descent based algorithm is proposed here for solving the open-loop optimization problem with no noise ($\epsilon = 0$), where the underlying dynamics model is used as a blackbox, and the gradients are found from a sequence of rollout data with input perturbed by Gaussian noise. Let us denote the initial guess of the control sequence as $U^{(0)} = \{\bar{u}_t^{(0)}\}_{t=0}^{T-1}$, and the corresponding observations as

$\mathcal{Z}^{(0)} = \{\bar{z}_t^{(0)}\}_{t=0}^{T}$. The control policy is updated iteratively via:

$$U^{(n+1)} = U^{(n)} - \alpha \nabla_U \bar{J}|_{(\mathcal{Z}^{(n)}, U^{(n)})}, \tag{5}$$

where $U^{(n)} = \{\bar{u}_t^{(n)}\}_{t=0}^{T-1}$ denotes the nominal control sequence after n iterations. The vector $\bar{u}_t^{(n)}$ is the control value at step t and α is the step size parameter. $\bar{J}|_{(\mathcal{Z}^{(n)}, U^{(n)})}$ is the expected episodic cost under $U^{(n)}$ and the corresponding state trajectory $\mathcal{Z}^{(n)}$. To estimate the gradient $\nabla_U \bar{J}|_{(\mathcal{Z}^{(n)}, U^{(n)})}$, we 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, and the gradient vector is updated iteratively after each rollout. The expected episodic cost for the $(n+1)^{\text{th}}$ iteration is calculated by simulating a noiseless rollout with the control sequence $U^{(n)}$. The gradient vector is then estimated in a sequential manner as:

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

where j and n denote the j^{th} rollout of the $(n+1)^{\text{th}}$ iteration. $\bar{J}|_{(\mathcal{Z}^{(n)}, U^{(n)})}$ is the expected episodic cost of the $(n+1)^{\text{th}}$ iteration while $J|_{(\mathcal{Z}^{j,(n)}, U^{j,(n)})}$ denotes the cost of the j^{th} rollout under control sequence $U^{j,(n)}$ and the corresponding state trajectory $\mathcal{Z}^{j,(n)}$. Note that $U^{j,(n)} = \{\bar{u}_t^{(n)} + \delta u_t^{j,(n)}\}_{t=0}^{T-1}$ where $\{\delta u_t^{j,(n)}\}_{t=0}^{T-1}$ is the zero-mean, i.i.d Gaussian noise added as perturbation to the control sequence $U^{(n)}$. The scalar $\sigma_{\delta u}$ is the variance of the input perturbation and $\nabla_U \bar{J}|^{j+1}_{(\mathcal{Z}^{(n)}, U^{(n)})}$ denotes the gradient vector after j rollouts. The total rollout number r in one iteration is decided by the convergence of the gradient vector. After r rollouts, the control sequence is updated by Eq. (5) in which $\nabla_U \bar{J}|_{(\mathcal{Z}^{(n)}, U^{(n)})}$ is estimated by $\nabla_U \bar{J}|^{r}_{(\mathcal{Z}^{(n)}, U^{(n)})}$ using this iterative method. Repeat this until the cost converges and the optimized nominal control sequence is the result of the last iteration: $\{\bar{u}_t^*\}_{t=0}^{T-1} = \{\bar{u}_t^N\}_{t=0}^{T-1}$. The idea is basically averaging the gradient vector over all the rollouts in an iterative way. Note that the updated control can also be clamped at each iteration to enforce control constraints depending on the specific problem.

B. Linear Time-Varying System Identification using ARMA Model

Here we use the standard least square method to estimate the linear parameters from input-output experiment data. First start from the perturbed linear system about the nominal trajectory and estimate the system parameters α_{t-i} and β_{t-i} for $i = 1, \cdots, q$ from: $\delta z_t^{(j)} = \alpha_{t-1}\delta z_{t-1}^{(j)} + \cdots + \alpha_{t-q}\delta z_{t-q}^{(j)} + \beta_{t-1}\delta u_{t-1}^{(j)} + \cdots + \beta_{t-q}\delta u_{t-q}^{(j)}$, where $\delta z_t^{(j)}$ is the observed output with the control input perturbation vector $\delta u_t^{(j)}$ that we feed to the system at step t for the j^{th} simulation, where $j = 1, \cdots, N$ for a total of N simulations. All the perturbations are zero-mean, i.i.d, Gaussian noise with covariance matrix σI. The covariance σ is a $o(u)$ small value selected by the user. After running N simulations for each step and collecting the data, we can write the linear mapping between input-output perturbation as:

$$\mathbb{Z} = [\alpha_{t-1} \mid \alpha_{t-2} \mid \cdots \mid \alpha_{t-q} \mid \beta_{t-1} \mid \beta_{t-2} \mid \cdots \mid \beta_{t-q}]\mathbb{X},$$

and write out the components:

$$\mathbb{Z} = \begin{bmatrix} \delta z_t^{(1)} & \delta z_t^{(2)} & \cdots & \delta z_t^{(N)} \end{bmatrix},$$
$$\mathbb{X} = \begin{bmatrix} \delta z_{t-1}^{(1)} & \delta z_{t-1}^{(2)} & \cdots & \delta z_{t-1}^{(N)} \\ \vdots & \vdots & \ddots & \vdots \\ \delta z_{t-q}^{(1)} & \delta z_{t-q}^{(2)} & \cdots & \delta z_{t-q}^{(N)} \\ \delta u_{t-1}^{(1)} & \delta u_{t-1}^{(2)} & \cdots & \delta u_{t-1}^{(N)} \\ \vdots & \vdots & \ddots & \vdots \\ \delta u_{t-q}^{(1)} & \delta u_{t-q}^{(2)} & \cdots & \delta u_{t-q}^{(N)} \end{bmatrix}. \tag{6}$$

Finally, using the standard least square method, the linearized system parameters are estimated as:

$$[\alpha_{t-1} \mid \cdots \mid \alpha_{t-q} \mid \beta_{t-1} \mid \cdots \mid \beta_{t-q}] = \mathbb{Z}\mathbb{X}^{\mathsf{T}}(\mathbb{X}\mathbb{X}^{\mathsf{T}})^{-1}. \tag{7}$$

C. System Dynamics in Information State

After identifying the system parameters $\alpha_{t-1}, \cdots, \alpha_{t-q}$ and $\beta_{t-1}, \cdots, \beta_{t-q}$ for all $t = \{0 \cdots T\}$, using the least square method, now, we write the linear perturbation system in the information state as given in Eq. (8), where γ_{t-1} is given as: $\gamma_{t-1} = \beta_{t-1} + \beta_{t-2} + \cdots + \beta_{t-q}$, as the random noise sequence $\{w_{t-1}, w_{t-2}, \cdots, w_{t-q}\}$ is independent and assuming the noise to enter the same channel as the control input. This is based on the idea that if the output z_t depends on last q control inputs, then it would also depend on last

$$\underbrace{\begin{bmatrix} \delta z_t \\ \delta z_{t-1} \\ \delta z_{t-2} \\ \vdots \\ \delta z_{t-q+1} \\ \hline \delta u_{t-1} \\ \delta u_{t-2} \\ \delta u_{t-3} \\ \vdots \\ \delta u_{t-q+1} \end{bmatrix}}_{\delta Z_t} = \underbrace{\left[\begin{array}{ccccc|ccccc} \alpha_{t-1} & \alpha_{t-2} & \cdots & \alpha_{t-q+1} & \alpha_{t-q} & \beta_{t-2} & \beta_{t-3} & \cdots & \beta_{t-q+1} & \beta_{t-q} \\ 1 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ \vdots & & \ddots & & \vdots & \vdots & & \ddots & \vdots & 0 \\ 0 & 0 & \cdots & 1 & 0 & 0 & 0 & \cdots & 0 & 0 \\ \hline 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & & \ddots & & \vdots & \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 1 & 0 \end{array}\right]}_{A_{t-1}} \underbrace{\begin{bmatrix} \delta z_{t-1} \\ \delta z_{t-2} \\ \vdots \\ \delta z_{t-q+1} \\ \delta z_{t-q} \\ \delta u_{t-2} \\ \delta u_{t-3} \\ \vdots \\ \delta u_{t-q+1} \\ \delta u_{t-q} \end{bmatrix}}_{\delta Z_{t-1}} + \underbrace{\begin{bmatrix} \beta_{t-1} \\ 0 \\ \vdots \\ 0 \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}}_{B_{t-1}} \delta u_{t-1} + \underbrace{\begin{bmatrix} \gamma_{t-1} \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}}_{D_{t-1}} w_{t-1} \tag{8}$$

q disturbance terms. Now, we can use the identified A_{t-1}, B_{t-1} to design an LQG controller for the information state, i.e., $\delta u_t = K_t \delta Z_t$, where note that the current control input depends on the past observations as well as the control inputs.

D. Closed Loop Control Design with specific LQG method

Given the estimated perturbed linear system, we design a finite horizon, discrete time LQG [20] along the trajectory for each time step to minimize the cost function:

$$J = E\left[\delta Z_T^T Q_T \delta Z_T + \sum_{t=0}^{T-1} (\delta Z_t^T Q_t \delta Z_t + \delta u_t^T R_t \delta u_t)\right], \tag{9}$$

subject to $\delta Z_t = A_{t-1}\delta Z_{t-1} + B_{t-1}\delta u_{t-1} + D_{t-1} w_{t-1}$, $\delta Y_t = \delta Z_t + v_t$, where δY is the noisy measurement. Notice that this is a specific version of LQG where state is estimated in the presence of process and measurement noise only as opposed to estimating the states from smaller number of noisy measurements, i.e. ($C = I$): $L_t = P_t(P_t + V_t)^{-1}$, where P_t is solved in a forward propagation fashion from the Riccati equation:

$$P_{t+1} = A_t[P_t - P_t(P_t + V_t)^{-1} P_t]A_t^T + D_t W_t D_t^T, \tag{10}$$

with the initial condition $P_0 = E[\tilde{x}\tilde{x}^T]$. The feedback gain for the above problem are calculated as:

$$K_{t-1} = (R + B_{t-1}^T S_t B_{t-1})^{-1}(B_{t-1}^T S_t A_{t-1}), \tag{11}$$

where S_t is solved in a back propagation fashion from the Riccati equation:

$$S_t = A_t^T S_{t+1}[I - B_t(R_t + B_t^T S_{t+1} B_t)^{-1} B_t^T S_{t+1}]A_t + Q_t, \tag{12}$$

with final condition as $S_T = Q_T$. Finally, the closed-loop control policy is $u_t = \bar{u}_t^* - K_t \delta \hat{Z}_t$, where $\delta \hat{Z}_t = A_{t-1}\delta \hat{Z}_{t-1} + B_{t-1}\delta u_{t-1} + L_t(\delta Y_t - A_{t-1}\delta \hat{Z}_{t-1} - B_{t-1}\delta u_{t-1})$.

1) Simplified LQR design: For the case of noiseless measurements, a simplified LQR design can be used for the closed-loop feedback control with $u_t = \bar{u}_t^* - K_t \delta Z_t$ where K_t is calculated using eqs. (11) and (12) only.

IV. The ARMA Model for Design

This section provides the condition on the order ($q = p$) of the ARMA model for the exact match for a linear system. The output observation z_t is to be modeled using the past few observations and past few excitation inputs u_t, where excitation input is modeled as independent identically distributed random variables (i.i.d.) with zero mean.

Proposition 1: The ARMA model of the order q given by:

$$\begin{aligned}\delta z_t = \alpha_{t-1}\delta z_{t-1} + \cdots + \alpha_{t-q}\delta z_{t-q} \\ + \beta_{t-1}\delta u_{t-1} + \cdots + \beta_{t-q}\delta u_{t-q},\end{aligned} \tag{13}$$

exactly fits a linear model $\delta x_t = A\delta x_{t-1} + B\delta u_{t-1}$ with output $\delta z_t = C\delta x_t$ if the matrix: $\mathcal{O}_q = \begin{bmatrix} {A^{q-1}}^T C^T & {A^{q-2}}^T C^T & \cdots & A^T C^T & C^T \end{bmatrix}^T$ is full column rank.

Proof: Let us start by writing the output equation for past q time-steps as:

$$\underbrace{\begin{bmatrix} \delta z_{t-1} \\ \delta z_{t-2} \\ \vdots \\ \delta z_{t-q+1} \\ \delta z_{t-q} \end{bmatrix}}_{\mathcal{Z}_q} = \underbrace{\begin{bmatrix} CA^{q-1} \\ CA^{q-2} \\ \vdots \\ CA \\ C \end{bmatrix}}_{\mathcal{O}_q} \delta x_{t-q}$$
$$+ \underbrace{\begin{bmatrix} 0 & CB & \cdots & CA^{q-2}B & CA^{q-1}B \\ & 0 & CB & \cdots & CA^{q-2}B \\ & & \ddots & \vdots & \vdots \\ & & & 0 & CB \\ & & & & 0 \end{bmatrix}}_{\mathcal{H}_q} \underbrace{\begin{bmatrix} \delta u_{t-1} \\ \delta u_{t-2} \\ \vdots \\ \delta u_{t-q+1} \\ \delta u_{t-q} \end{bmatrix}}_{\mathcal{U}_q},$$

which can also be written as: $\mathcal{O}_q \delta x_{t-q} = \mathcal{Z}_q - \mathcal{H}_q \mathcal{U}_q$. A unique solution for δx_{t-q} exist if the matrix $\mathcal{O}_q$ is full column rank matrix and then the solution can be written as: $\delta x_{t-q} = \mathcal{O}_q^+(\mathcal{Z}_q - \mathcal{H}_q \mathcal{U}_q)$.

Now, the solution for output at time t can be written as:

$$\begin{aligned}\delta z_t = CA^q \delta x_{t-q} + CA^q B\delta u_{t-q} + CA^{q-1}B\delta u_{t-q+1} \\ + \cdots + CB\delta u_{t-1},\end{aligned} \tag{14}$$

where the unique solution for δx_{t-q} is substituted to get: $\delta z_t = CA^q(\mathcal{O}_q^+(\mathcal{Z}_q - \mathcal{H}_q\mathcal{U}_q)) + CA^q B\delta u_{t-q} + CA^{q-1}B\delta u_{t-q+1} + \cdots + CB\delta u_{t-1}$. Finally, substituting for $\mathcal{Z}_q$ and $\mathcal{H}_q$ gives:

$$\begin{aligned}\delta z_t = CA^q \mathcal{O}_q^+ \begin{bmatrix} \delta z_{t-1} \\ \delta z_{t-2} \\ \vdots \\ \delta z_{t-q+1} \\ \delta z_{t-q} \end{bmatrix} - CA^q \mathcal{O}_q^+ \mathcal{H}_q \begin{bmatrix} \delta u_{t-1} \\ \delta u_{t-2} \\ \vdots \\ \delta u_{t-q+1} \\ \delta u_{t-q} \end{bmatrix} \\ + \begin{bmatrix} CB & CAB & \cdots & CA^{q-1}B & CA^q B \end{bmatrix} \begin{bmatrix} \delta u_{t-1} \\ \delta u_{t-2} \\ \vdots \\ \delta u_{t-q+1} \\ \delta u_{t-q} \end{bmatrix},\end{aligned} \tag{15}$$

which gives the exact analytical solution for the ARMA parameters in terms of linear system matrices A, B and C. ■

Notice that if there exists a number q for which the matrix $\mathcal{O}_q$ is full column rank and then there always exists an exact fit for the ARMA model with sufficiently large enough q. This allows us to write linearized dynamics models at each step along the nominal trajectory in terms of ARMA parameters.

Corollary 4.1: ***Linear Time-Varying (LTV) System*** - The linear time-varying case: $\delta x_t =$

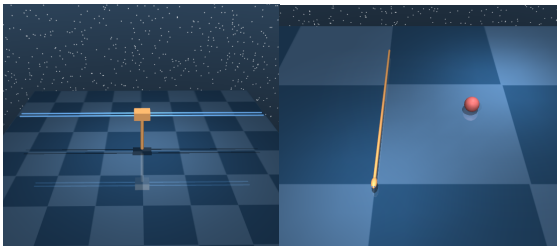
$A_{t-1}\delta x_{t-1} + B_{t-1}\delta u_{t-1}$, $\delta z_t = C_t \delta x_t$, follows directly from this with the condition of $\mathcal{O}_q = \left[A_{t-q}^T ... A_{t-1}^T C_t^T, \ \cdots, \ A_{t-q}^T C_{t-q+1}^T, \ C_{t-q}^T\right]^T$ to be full column rank.

Remark 4.1: For the examples we have tried, we observed that the q value does not change at each time-step and most likely would not change for physical/mechanical systems.

Corollary 4.2: For the case of mechanical systems with all the position/DOFs as the output feedback, the minimum value for q would be $q = 2$, which would allow for the exact fit for the ARMA model to be with only 2 past observations and inputs.

V. Empirical Results

We use MuJoCo, a physics engine [21], as a black-box to provide the data to design the nominal trajectory and closed-loop feedback gain. First, we list the details of the MuJoCo models used in our simulations.

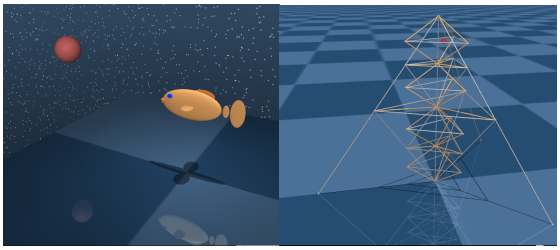


(a) Cart-Pole (b) 15-link Swimmer

Fig. 2: Models simulated in MuJoCo in their initial states.

Cart-Pole: The state of a four-dimension under-actuated cart-pole comprises the angle of the pole, cart's horizontal position, and their rates. Within a given horizon, the task is to swing-up the pole and balance it in the middle of the rail by applying a horizontal force on the cart. Figure 2(a) shows the initial position of the cart-pole system.

15-link Swimmer: The 15-link swimmer model has 17 degrees of freedom and together with their rates, the system is described by 34 state variables. The task is to solve the planning and control problem from a given initial state to the goal position located at the center of the ball. Controls can only be applied in the form of torques to the 14 joints. Hence, it is under-actuated by 3 DOF. Figure 2(b) shows the initial configuration of the swimmer.



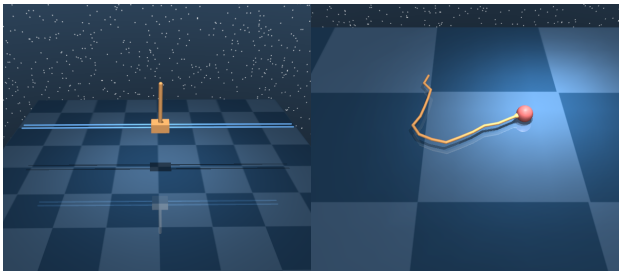
(a) Fish (b) Tensegrity Robotic Arm

Fig. 3: Models simulated in MuJoCo in their initial states.

Fish: The fish model moves in 3D space, the torso is a rigid body with 6 DOF. The system is described by 27 dimensions of states (including a set of quaternions) and 6 control channels. The task is to swim into the target ball. Controls are applied in the form of torques to the joints that connect the fins and tails with the torso. The rotation of the torso is described using quaternions. Figure 3(a) shows the initial configuration of the fish.

T2D1 Robotic Arm: The T_2D_1 tensegrity model is a 3D robotic arm [22]. It consists of 33 bars and 46 strings. The bars are connected by ball joints. The task is to reach the top node into the target ball from the given initial configuration in Fig. 3(b). Controls are applied in the form of tension in the strings, while the feedback is based on the coordinates of some of the nodes. The bars are shown as orange objects and the strings are shown as grey objects.

The final configuration of all the four models is given in Figs. 1 and 4. The final configurations are obtained at the end of the horizon with the partially-observed D2C algorithm. The videos for the simulation are given as supplementary files.



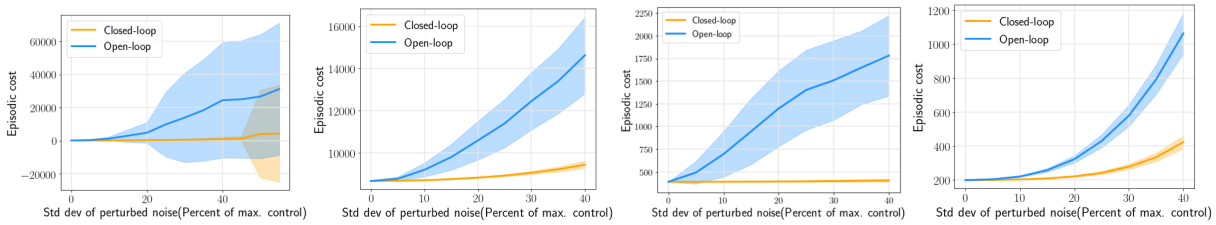
(a) Cart-Pole (b) 15-link Swimmer

Fig. 4: Models simulated in MuJoCo in their terminal states.

TABLE I: ARMA fitting parameters

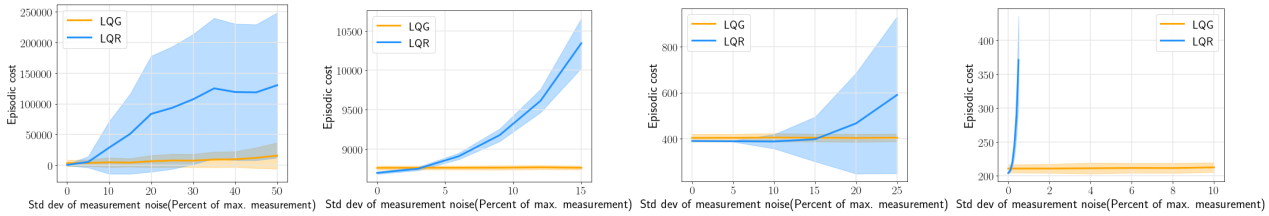
System	Steps per episode	Timestep (sec.)	q	State Number	Output Number
Cart-pole	30	0.1	2	4	2
15-link Swimmer	2400	0.005	5	34	10
Fish	1200	0.005	2	27	11
T2D1 Robotic Arm	200	0.01	3	150	24

The parameters for fitting the ARMA models for the four examples are given in Table I. The output number values represent the minimum number of state measurements needed to obtain a good fit of the ARMA model from the output data with q values representing the order of the ARMA model. Notice that a relatively smaller number of measurements are needed to control the structure with increasing complexity (higher number of states) of the model. The cart-pole is a classic underactuated robotics example where both cart's horizontal position and angle of the pole are needed as feedback. The 15-link swimmer and fish present the performance of our method when applied to high-dimensional multi-body robots in a fluid environment. The 15-link swimmer needs only angular positions of the 1st, 3rd, 5th, 7th, 9th, 11th, 13th, and 15th joints and the x,



(a) Cart-Pole (b) 15-link Swimmer (c) Fish (d) T2D1 Robotic Arm

Fig. 5: Averaged episodic reward vs process noise level with LQR as closed-loop feedback. Data collected from 200 rollouts at each noise level.



(a) Cart-Pole (b) 15-link Swimmer (c) Fish (d) T2D1 Robotic Arm

Fig. 6: Averaged episodic reward vs measurement noise for fixed process noise comparing LQG with LQR as closed-loop feedback. Data collected from 200 rollouts at each noise level.

y coordinates of the head. The fish needs only the angular positions of the fins and tails, the x, y, z coordinates of the head and the first quaternion element. The T_2D_1 robotic arm case shows the application to high-dimensional soft-body models. Here only the positions of 8 evenly chosen nodes are needed out of the total 25 nodes. Notice that velocity or rates feedback are not needed in the control design as the value of the state for rates can always be calculated from the past 2 observations of positions (refer to Corollary 4.2). The rule of thumb for measurement selection is that we only measure the positions that contain the most information and avoid redundant information.

Process noise only: Figure 5 shows the plots for the episodic cost of the four examples with the variation in the process noise. The figure compares the open-loop control policy and the LQR closed-loop control policy under different process noise levels. In the open-loop case, the process noise, which is added as the zero-mean Gaussian i.i.d. noise to every control channel at each step, drives the model off the nominal trajectory and results in the increased episodic cost. The standard deviation of the noise is proportional to the maximum control signal in the designed optimal control sequence for D2C. Notice that the mean episodic cost and the cost variance of the closed-loop policy, even with partial observation, is much smaller than that of the open-loop policy for all the examples for the entire noise level range. As mentioned earlier, the *simplified LQR design* is sufficient to robustly track the nominal trajectory for the case of process noise only. The maximum values of process noise in all the examples are found by constantly increasing the noise and finding a threshold value that will drive the system too far away from the nominal trajectory such that the final goal can not be achieved.

Both process and measurement noise: Figure 6 shows similar plots as shown in Fig. 5 except measurement noise is also added in the simulations along with the process noise. The plots are created by increasing the measurement noise in the system for a fixed value of process noise with standard deviation as 10% of the maximum control value. The *specific LQG method* is compared with the *simplified LQR design* as the closed-loop part of the algorithm. Notice that the episodic cost mean and variance for the LQR policy is smaller than the LQG policy when the measurement noise is smaller than a threshold and increased rapidly as we increase the measurement noise. When the measurement noise is small, the error between the fitted ARMA (linear) model and the true nonlinear model makes the LQG estimation less accurate than the measurement itself, while the LQR feedback based only on the accurate enough measurement performs well. As the measurement noise increases, the LQG policy benefits from the ARMA model and outperforms the LQR policy.

VI. Conclusions

In this paper, we presented a decoupled data-based approach to control complex robotic systems with partial observations. The paper shows that the exact linear state-space model can be matched by the q^{th}-order ARMA model generated using the input-output data. The ARMA model is then written as an LTV system in the information state which allows designing the closed-loop feedback design using only partially-observed states. A unique kind of LQG algorithm is then formulated where all the states are already available from the information state representation. The results for this data-based algorithm are also shown using complex models with fluid-structure interaction as analytical models for such systems are not available or can not be modeled accurately.

REFERENCES

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

[2] P. A. Ioannou and J. Sun, *Robust adaptive control*. Courier Corporation, 2012.

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

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

[5] 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.

[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] 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.

[8] 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.

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

[10] S. John, W. Filip, D. Prafulla, R. Alec, and K. Oleg, "Proximal policy optimization algorithms," *arXiv:1707.06347*, 2017.

[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] R. Wang, K. S. Parunandi, D. Yu, D. M. Kalathil, and S. Chakravorty, "Decoupled data based approach for learning to control nonlinear dynamical systems," *arXiv, also IEEE Transactions on Automatic Control (to appear)*, vol. abs/1904.08361, 2019. [Online]. Available: http://arxiv.org/abs/1904.08361

[13] G. E. Box, G. M. Jenkins, and G. C. Reinsel, *Time series analysis: forecasting and control*. John Wiley & Sons, 2011, vol. 734.

[14] M. Ghahramani and A. Thavaneswaran, "Financial applications of arma models with garch errors," *The Journal of Risk Finance*, 2006.

[15] I. Leontaritis and S. A. Billings, "Input-output parametric models for non-linear systems part i: deterministic non-linear systems," *International journal of control*, vol. 41, no. 2, pp. 303–328, 1985.

[16] I. J. Leontaritis and S. A. Billings, "Input-output parametric models for non-linear systems part ii: stochastic non-linear systems," *International journal of control*, vol. 41, no. 2, pp. 329–344, 1985.

[17] F. N. Chowdhury, "Input-output modeling of nonlinear systems with time-varying linear models," *IEEE Transactions on Automatic Control*, vol. 45, no. 7, pp. 1355–1358, 2000.

[18] E. Hernandez and Y. Arkun, "Control of nonlinear systems using polynomial arma models," *AIChE Journal*, vol. 39, no. 3, pp. 446–460, 1993.

[19] D. Yu, M. Rafieisakhaei, and S. Chakravorty, "Stochastic Feedback Control of Systems with Unknown Nonlinear Dynamics," in *56^{th} IEEE Conference on Decision and Control(CDC)*, 2017.

[20] A. Bryson and H. Y.-C., *Applied Optimal Control: Optimization, Estimation and Control*. Washington: Hemisphere Pub. Corp., 1975.

[21] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.

[22] R. Wang, R. Goyal, S. Chakravorty, and R. E. Skelton, "Model and data based approaches to the control of tensegrity robots," *IEEE Robotics and Automation Letters*, pp. 3846–3853, 2020.