Journal of Machine Learning Research 25 (2024) 1-35 Submitted 1/24; Revised 7/24; Published 12/24

A New, Physics-Informed Continuous-Time Reinforcement Learning Algorithm with Performance Guarantees

Brent A. Wallace BAWALLA2@ASU.EDU
Department of Electrical Engineering
Arizona State University
Tempe, AZ 85287, USA

Jennie Si SI@ASU.EDU
Department of Electrical Engineering
Arizona State University
Tempe, AZ 85287, USA

Editor: George Konidaris

Abstract

We introduce a new, physics-informed continuous-time reinforcement learning (CT-RL) algorithm for control of affine nonlinear systems, an area that enables a plethora of well-motivated applications. Based on fundamental control principles, our approach uses reference command input (RCI) as probing noise to enable exploration in learning. With known physical dynamics of the environment, by leveraging on the Kleinman algorithm structure, and using state-action trajectory data, RCI provides a data-efficient optimal control solution under an infinite-horizon undiscounted cost. We show that our RCI-based CT-RL algorithm not only provides theoretical guarantees such as learning convergence, solution optimality, and closed-loop stability, but also well-behaved dynamic system responses. It is noted that our evaluations not only include extensive baseline and ablation studies using typical performance measures in RL, but also essential control-centric performance measures that are critical for real-life control applications. As a result, we demonstrate that our RCI-based CT-RL leads to new, SOTA control design and performance.

Keywords: Continuous-time reinforcement learning (CT-RL), adaptive/approximate dynamic programming (ADP), reference command input, persistence of excitation (PE), optimal control

1. Introduction

Continuous-time optimal control problems can be found in many important engineering and socioeconomic application domains such as aerospace (Stengel, 2022), waste water treatment (Yang et al., 2022), robotics (Craig, 2005), and economics (Caputo, 2005). Reinforcement learning (RL) emerged as a systematic method in the early 1980s (Barto et al., 1983; Sutton and Barto, 2018) to combat the curse of dimensionality by providing approximate dynamic programming solutions. Discrete-time (DT) RL algorithms (Si et al., 2004; Lewis et al., 2012b; Kiumarsi et al., 2018; Bertsekas, 2017; Liu et al., 2021) have shown extensive theoretical guarantees and demonstrations in applications. In particular, model-based DT-RL trajectory optimization algorithms (Landry et al., 2021; Atkeson and Morimoto, 2002) solve for the optimal value over the constraints of the nonlinear dynamics, simultaneously

accommodating 1) multiple cost structures, and 2) addition of application-specific control problem constraints. When the discrete-time dynamics are restricted to be polynomial, sum-of-squares approaches provide provable performance bounds on such systems (Summers et al., 2013). By contrast, CT-RL algorithms have seen fewer theoretical results and little applications successes. The CT-RL optimal control problem is considered more difficult than its DT-RL counterpart, as solving the Hamilton-Jacobi-Bellman (HJB) partial differential equation associated with CT problems is fundamentally more challenging than solving the Bellman difference equation for DT problems. To address the unique challenges of CT-RL problems, as a result, four main solution approaches have developed.

The first and intuitive approach is to discretize the continuous-time dynamics in order to approximate a CT problem by a DT problem. A Hamilton-Jacobi method by Kim et al. (2021) is developed for CT systems, and the proposed Q-learning framework requires discretizing the dynamics. Similarly, Haoran et al. (2020) propose stochastic control in a linear quadratic setting. The approach requires applying limiting arguments to a discretization. The neural ODE model-based RL framework (Yildiz et al., 2021) requires an Euler discretization of the value integral and using Monte Carlo sampling in training the critic.

The second approach resorts to LQR to eliminate the nonlinearity challenge. One of the seminal methods is developed by Bradtke et al. (1994). More recently, Possieri and Sassano (2022) develop a data-driven Q-learning approach to Kleinman's algorithm (Kleinman, 1968), but the method is limited to linear systems. Similarly, one of the early ADP works (Jiang and Jiang, 2012) is also limited to continuous-time LQR problems.

The third approach addresses CT-RL problems under general nonlinear (non-affine) dynamics, but only a handful of results are currently available. The ADP method in (Bian and Jiang, 2022) is one important work under this category. However, significant theoretical assumptions are imposed in order to prove algorithm convergence; practically, this translates to significant realization issues when the method is used to design CT-RL controllers, even on simple examples (Wallace and Si, 2024b). The method proposed in (Yildiz et al., 2021) treats general nonlinear dynamics described by an ODE. It addresses issues due to irregular and noisy data by using Bayesian neural ODE models to infer robust state differential information. Their CT-RL algorithm then relies on this model for inferred dynamics. Such an open-loop approach to control may seriously limit the applicability of the method. The authors of (Sandoval et al., 2023) address general nonlinear optimal control problems by using neural ODEs as feedback policies. Their approach, however, is limited to fixed initial condition and fixed final time. Together with the requirement of a known nonlinear dynamic model for numerical integration (at discrete time steps) for generating a forward state path, this method again has limited its applicability.

The methods dealing with general non-affine nonlinear dynamics, as discussed in the above, are important first steps to investigate the challenging CT-RL problems in a general setting, and some of them may have great potential to become meaningful methods. However, this approach requires significant further developments. This leads to a perhaps most studied fourth approach to nonlinear optimal control that deals with affine nonlinear systems. These methods allow closed-form solutions for the optimal policy in terms of the HJB solution (Lewis et al., 2012a). This approach has attracted by far the greatest attention including the foundational adaptive dynamic programming (ADP) works (Vrabie and Lewis, 2009; Vamvoudakis and Lewis, 2010; Jiang and Jiang, 2014) as well as the recent suc-

cesses in deep RL fitted value iteration (FVI) (Lutter et al., 2021, 2023b). When the affine nonlinear dynamics are assumed to be polynomial, which is rather restrictive, efficient sum-of-squares approaches have been developed which provide over- and under-approximations of the optimal value, as well as estimates of the region of attraction and policy optimality error (Yang et al., 2023). The significant advances in this area are not only due to the structural convenience of affine nonlinearity, but also are motivated by a broad applicability of such nonlinear systems, as a variety of compelling real-world applications admit such affine nonlinearity. For example, the Euler-Lagrange mechanisms can be found in many familiar systems, including robotic manipulators, wearable mechanical robots, marine navigation equipment, automatic machine tools, satellite-tracking antennas, UAVs, autofocus cameras, and many more. As such, recent CT-RL methods tailored to specific affine-nonlinear application domains have begun to achieve substantial performance, in for example aerospace systems (Wallace and Si, 2024a). For the same reason, our RCI-based CT-RL method also addresses optimal control of affine nonlinear dynamics with an aim to derive reliable design procedures that can be demonstrated on meaningful application problems.

2. Related Work

The current results on affine nonlinear systems generally fall into two classes of algorithms: adaptive dynamic programming (ADP), and actor-critic deep RL (DRL). The first school of adaptive dynamic programming (**ADP**) learns through optimal and adaptive control frameworks, oftentimes treating network weights as part of the adaptation parameters. These ADP approaches were developed largely within the scope of seminal works such as integral reinforcement learning (IRL) (Vrabie and Lewis, 2009), synchronous policy iteration (SPI) (Vamvoudakis and Lewis, 2010), robust ADP (RADP) (Jiang and Jiang, 2014), and continuous-time value iteration (CT-VI) (Bian and Jiang, 2022). There is actually a significant body of literature on ADP approaches to CT-RL control of nonlinear affine systems. But as shown in a recent comprehensive and systematic review of up-to-date ADP methods (Wallace and Si, 2024b), they are extensions revolving around the four central works. As a result of ADP's theoretical frameworks in adaptive and optimal control, Lyapunov arguments are available to prove qualitative properties including weight convergence and closed-loop stability results. Yet, quantitative results are few, as the proposed algorithms struggle to synthesis learning controllers in meaningful ways, and have only been evaluated on simple second order systems with known optimal solutions (Wallace and Si, 2024b).

The second deep RL (**DRL**) approach is the most recent and perhaps most promising to date, in particular the fitted value iteration (FVI) deep RL methods, which solve the Hamilton-Jacobi-Bellman/Isaacs (HJB/HJI) equations directly through approximation by deep networks, large datasets, and extensive training. The concept of applying black-box function approximation to solve the HJB traces back to the seminal work of Doya (2000). Subsequently, Tassa and Erez (2007) proposed a foundational value-function approximation framework using least-squares regression techniques for its neural network training. Recently, the novel continuous FVI (cFVI) (Lutter et al., 2021) and robust FVI (rFVI) (Lutter et al., 2023b) algorithms empirically exhibit low variance and control performance far surpassing that of prevailing ADP methods, and stand for the state-of-the-art currently. However, theoretical guarantees as those offered by ADP are yet to be developed.

Contributions. We propose a new model-based CT-RL learning approach with the following three contributions: 1) Our novel reference command input (RCI) enabled learning leverages fundamental feedback control principles to provide effective and efficient exploration for learning. 2) By taking advantage of Kleinman's structure and using state-action data-driven learning, we address nonlinear learning control problems with data efficiency. In addition, when the system physics afford a decentralized structure, RCI can make the algorithm even more efficient. 3) We provide theoretical guarantees of RCI CT-RL learning convergence, solution optimality, and system stability alongside comprehensive evaluations and comparisons to demonstrate RCI-enabled SOTA CT-RL results in optimal control of affine nonlinear systems.

3. Method

RCI requires knowledge of an affine nonlinear dynamics of the environment. Together with state-action trajectory data, it produces a sequence of policies $\{\mu_i\}_{i=1}^{\infty}$ which iteratively solve for the optimal control problem below.

3.1 Problem Formulation and Background

We consider the same affine nonlinear system as that considered in the above SOTA ADP and DRL methods:

$$\dot{x} = f(x) + g(x)u, \tag{1}$$

where $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^m$ is the control, $f : \mathbb{R}^n \to \mathbb{R}^n$, and $g : \mathbb{R}^n \to \mathbb{R}^{n\times m}$. As standard, we assume f and g are Lipschitz on a compact set $\Omega \subset \mathbb{R}^n$ containing the origin $x = 0$ in its interior, and that $f(0) = 0$. We consider the infinite-horizon undiscounted cost

$$J(x_0) = \int_0^{\infty} (x^T Q x + u^T R u)\, d\tau, \tag{2}$$

where $Q \in \mathbb{R}^{n\times n}$, $Q = Q^T \geq 0$ and $R \in \mathbb{R}^{m\times m}$, $R = R^T > 0$ are the state and control penalties. This Q-R formulation is the standard in nonlinear optimal control formulations (Lewis and Liu, 2012).

Kleinman's Algorithm for Linear Systems with Known Dynamic Model (Kleinman, 1968). We take advantage of some successive approximation concepts from Kleinman's algorithm to the proposed nonlinear RCI algorithm for data efficiency. Classical Kleinman's algorithm considers the linear time-invariant system $\dot{x} = Ax + Bu$. We assume that (A, B) is stabilizable and $(Q^{1/2}, A)$ is detectable, for well-posedness (Rodriguez, 2004). Kleinman's algorithm iteratively solves for the optimal LQR control $K^* = R^{-1}B^T P^*$, where $P^* \in \mathbb{R}^{n\times n}$, $P^* = P^{*T} > 0$ is the solution of the Riccati equation (Rodriguez, 2004), as follows. For iteration $i = 0, 1, \ldots$, on the current policy K_i, let $P_i \in \mathbb{R}^{n\times n}$, $P_i = P_i^T > 0$ be the solution of the algebraic Lyapunov equation (ALE)

$$(A - BK_i)^T P_i + P_i(A - BK_i) + K_i^T R K_i + Q = 0. \tag{3}$$

Then, P_i solved from (3) leads to the new policy $K_{i+1} \in \mathbb{R}^{m\times n}$ as

$$K_{i+1} = R^{-1}B^T P_i. \tag{4}$$

The following theorem is needed to prove the theoretical results of Section 4.

Theorem 1 (Convergence, Optimality, and Closed-Loop Stability of Kleinman's Algorithm (Kleinman, 1968)) *Suppose the initial policy K_0 is such that $A - BK_0$ is Hurwitz. Then we have the following:*

(i) $A - BK_i$ *is Hurwitz for all* $i \geq 0$.

(ii) $P^* \leq P_{i+1} \leq P_i$ *for all* $i \geq 0$, *and* $\lim_{i\to\infty} P_i = P^*$, $\lim_{i\to\infty} K_i = K^*$.

The following preparation is needed before introducing learning of the critic, which uses the bases Φ and weights c as formulated in Definition 2 and Proposition 3.

Definition 2 (Operators for Learning) *For a square symmetric matrix $P = P^T \in \mathbb{R}^{n\times n}$, define its "vectorization" $v(P)$ as*

$$v(P) = \big[p_{11},\, 2p_{12}, \ldots,\, 2p_{1n},\, p_{22},\, 2p_{23}, \ldots,\, 2p_{n-1,n},\, p_{nn}\big]^T. \tag{5}$$

We denote the dimension of the vector $v(P)$ as $\underline{n} \triangleq \frac{n(n+1)}{2}$. Given vectors $x, y \in \mathbb{R}^n$, define

$$\Phi(x,y) = \frac{1}{2}\big[2x_1y_1,\, x_1y_2 + x_2y_1, \ldots,\, x_1y_n + x_ny_1,\, 2x_2y_2, \ldots,\, 2x_ny_n\big]^T \in \mathbb{R}^{\underline{n}}. \tag{6}$$

Proposition 3 *The operators v (5) and Φ (6) satisfy the following:*

(i) v is a linear isomorphism of the symmetric matrices onto $\mathbb{R}^{\underline{n}}$; thus, for each $c \in \mathbb{R}^{\underline{n}}$, there exists a unique $P \in \mathbb{R}^{n\times n}$, $P = P^T$ such that

$$c = v(P). \tag{7}$$

(ii) Φ is a symmetric bilinear form. Furthermore, whenever $P \in \mathbb{R}^{n\times n}$, $P = P^T$, the following identity holds

$$\Phi^T(x,y)v(P) = x^TPy, \quad \forall\, x, y \in \mathbb{R}^n. \tag{8}$$

3.2 RCI Algorithm

Reference Command Input (RCI). Almost all ADP CT-RL algorithms require the persistence of excitation (PE) condition in proofs of algorithm properties (cf. Remark 5). PE is an analytical condition, but it is not constructive, nor is there an executable test procedure for PE for nonlinear systems (Shimkin and Feuer, 1987). In essence, PE means that in response to sufficiently exciting inputs the system states can be used in system identification and thus result in learning parameter convergence. This is in alignment with the notion of exploration in reinforcement learning. In almost all existing ADP works, to achieve PE it is standard practice to apply a probing noise d to the system (1) in a feedback control of the form $u = \mu(x) + d$, where $\mu : \mathbb{R}^n \to \mathbb{R}^m$ is a stabilizing policy. According to classical feedback control principles, a good feedback control attenuates plant input disturbances; thus, plant-input probing noise excitation is an inherently problematic

practice (Rodriguez, 2004) because good controllers attenuate plant input excitation. This leads to poor system excitation and data quality, the consequences of which are thoroughly analyzed in (Wallace and Si, 2024b). In particular, the resulting poor excitation leads to degraded algorithm numerics, thus poor quality in estimated value and policy. This often causes instability in the network weight updates, which often diverge and cause the closed-loop system to go unstable (Wallace and Si, 2024b).

We instead propose a reference command input (RCI) solution, also based on classical control principles, to excite the closed-loop system at the reference command input r (cf. Figure 1), which is favorable to the input disturbance d because the associated closed-loop map has larger magnitude response at low frequencies. This allows properly-designed input signals to pass through with little attenuation, thereby offering improved PE and excitation efficiency (Ogata, 1997; Rodriguez, 2004).

RCI is a new CT-RL method taking a holistic approach to most effectively and efficiently utilize known system dynamics information, fundamental feedback control principles, and learning based on observations from the system. As such, it enables SOTA CT-RL control design for affine nonlinear dynamical systems and leads to SOTA learning and control performance. Refer to Figure 1, which depicts the three key components of the new RCI method (bottom to top): 1) The feedback control loop that includes both the state feedback as the control policy, and also a feedback path for the reference command input. 2) The second component uses reference command input to enable effective and efficient exploration for learning. 3) The critic learning component is created so that it utilizes state trajectory data (x, u) to form the algorithm's learning update (see algorithm procedures below). Critically, RCI is compatible with current RL formulations. Since full state information is required in the optimal control problem, we may designate a subset of the state x as measurement variables $y \in \mathbb{R}^p$ for reference injection. Writing $x = \begin{bmatrix} y^T & x_r^T \end{bmatrix}^T$, where $x_r \in \mathbb{R}^{n-p}$ denotes the rest of the state, and denoting $e = r - y$ as the error signal, the control

$$u = \mu(e, x_r) = \mu(y, x_r) + \tilde{d}, \qquad \tilde{d} \triangleq \mu(e, x_r) - \mu(y, x_r), \tag{9}$$

is of the form $u = \mu(x) + \tilde{d}$. Thus, RCI can improve learning of existing CT-RL methods.

Critic Network Structure and Basis Selection. The critic is represented by

$$V(x) = \Phi^T(x, x)c_i, \tag{10}$$

where $c_i \in \mathbb{R}^{\underline{n}}$ in the form of Equation (7) are the weights of the critic, and Φ (6) contains the bases for critic approximation. Given $c_i \in \mathbb{R}^{\underline{n}}$ and from Proposition 3, we have that there exists a unique $P_i = P_i^T \in \mathbb{R}^{n \times n}$ such that $c_i = v(P_i)$. Thus, from (8), we have

$$V(x) = \Phi^T(x, x)c_i = \Phi^T(x, x)v(P_i) = x^T P_i x. \tag{11}$$

It is noted that we choose this quadratic approximation network in order to leverage the structure of the Q-R cost (2). It is also noted that this Q-R cost structure has been used in optimal control as a default. In-depth analysis and evaluations of this cost structure have proved it a highly effective means as a control objective in continuous state and control application problems (Lewis et al., 2012a; Bertsekas, 2005; Stengel, 2022; Rodriguez, 2004). Furthermore, even though we use Q-R cost structure, our approach is not an LQR method because we deal with affine nonlinear dynamics defined in Equation (1).

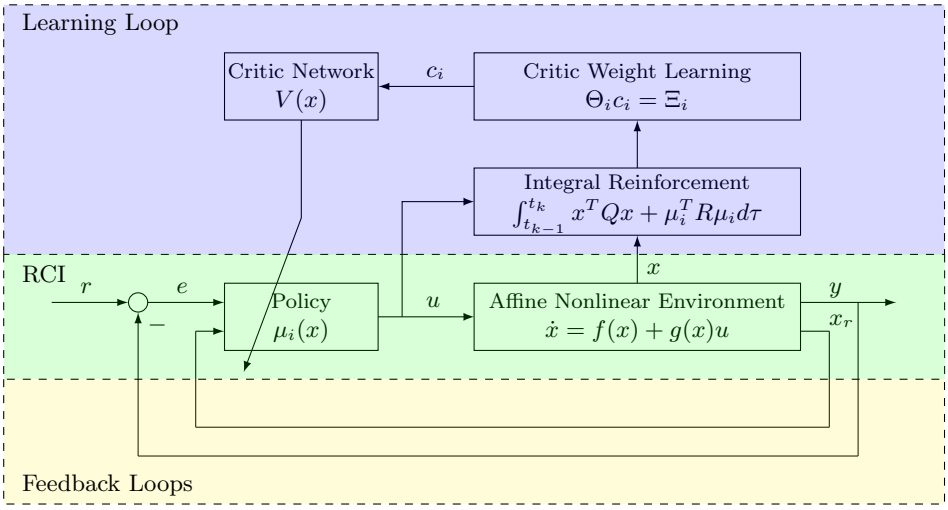

Figure 1: RCI enabled CT-RL learning control structure.

Control Policy. Having defined the critic structure, we form the control policy as follows

$$\mu_{i+1}(x) = -\frac{1}{2} R^{-1} g^T(0) \frac{\partial V}{\partial x}(x). \tag{12}$$

As in a general optimal control formulation (Lewis et al., 2012a), this policy solution is obtained from and for the affine nonlinear dynamics and utilizes gradient information of the value function approximator. It comes as a direct result of manipulating the HJB equation for control-affine systems. In addition, the policy structure (12) alongside the choice of quadratic critic bases (11) allows us to take advantage of Kleinman's control structure (Kleinman, 1968) in the subsequent algorithm formulation.

The New RCI CT-RL Design. We are now in a position to derive step-by-step our RCI-based CT-RL algorithm.

Step 1. Given iteration $i \geq 0$, let μ_i be the control policy at the i-th iteration, and let $t_0 < t_1$ be given. The critic network (output V) is designed to approximate the cost J in Equation (2). Defining the integral reinforcement signal ($\int_{t_0}^{t_1} x^T Q x + \mu_i^T(x) R \mu_i(x)\, d\tau$) as in (Vrabie and Lewis, 2009), then along environment trajectories, we have

$$V(x(t_0)) - V(x(t_1)) = \int_{t_0}^{t_1} x^T Q x + \mu_i^T(x) R \mu_i(x)\, d\tau. \tag{13}$$

The right-hand-side of (13) requires only state-action data (x, u). The learning goal is to minimize the approximation error $J - V$ in (13).

Step 2. For learning the critic weights c_i in Equation (10), we need latent parameters represented as $P_i = P_i^T$ in Proposition 3. Toward that, recasting (13) by differentiating the value function V along the trajectories of the nonlinear system (1), we have

$$V(x(t_1)) - V(x(t_0)) = \int_{t_0}^{t_1} \frac{d}{d\tau} \{V(x)\}\, d\tau = \int_{t_0}^{t_1} (f(x) + g(x)u)^T \frac{\partial V}{\partial x}(x)\, d\tau. \tag{14}$$

Using the identification $c_i = v(P_i)$ from Proposition 3, (14) can be written equivalently as

$$x^T(t_1)P_i x(t_1) - x^T(t_0)P_i x(t_0) = 2\int_{t_0}^{t_1} (f(x) + g(x)u)^T P_i x \, d\tau. \tag{15}$$

The differentiated value function equation (15) now incorporates the value function approximator structure V and its weights $c_i = v(P_i)$ alongside the nonlinear dynamics (f, g) (1) and the nonlinear trajectory data $\{x(t)\}_{t\in[t_0,t_1]}$ on the integral reinforcement interval $t \in [t_0, t_1]$.

Step 3. This step aims at incorporating the approximate cost V (10) of the policy μ_i to set up the stage for step 4 toward a value update, and thus the policy update μ_{i+1} (12). Construct a dynamic term ξ to be used in a reinforcement integral constant:

$$\xi(x) \triangleq Ax + B\mu_i(x), \quad A \triangleq \left.\frac{\partial\{f(x) + g(x)u\}}{\partial x}\right|_{x,u=0}, \quad B \triangleq \left.\frac{\partial\{f(x) + g(x)u\}}{\partial u}\right|_{x,u=0} \tag{16}$$

It is obtained by applying the current policy μ_i derived from the nonlinear dynamics (1) to its linearized system (A, B) (16) at state $x \in \mathbb{R}^n$. We now derive an equivalent expression for (15) which includes the current policy μ_i by adding an integral reinforcement constant based on ξ to both sides of (15):

$$x^T(t_1)P_i x(t_1) - x^T(t_0)P_i x(t_0) - 2\int_{t_0}^{t_1} \xi^T(x) \, P_i x \, d\tau = 2\int_{t_0}^{t_1} (f(x) + g(x)u - \xi(x))^T P_i x \, d\tau. \tag{17}$$

Notice that this algebraic manipulation from Equation (15) to (17) does not change the nature of the RCI procedure in directly addressing the nonlinear dynamics (1); indeed, (17) is algebraically equivalent to (15). Importantly, (17) still contains the derivative information of value V along nonlinear system trajectories $\{x(t)\}_{t\in[t_0,t_1]}$ from (1), and the integral reinforcement constant $\int_{t_0}^{t_1} \xi^T(x) \, P_i x \, d\tau$ in (17) is integrated over the nonlinear system trajectories $\{x(t)\}_{t\in[t_0,t_1]}$ to form part of the learning matrix Θ_i (23), which is then used to update the policy μ_{i+1} (12). Thus, the RCI policy μ_i is associated with the nonlinear dynamics (1), and not derived from a linearized (A, B) dynamics.

Step 4. We have prepared the value function equation (15) to include the current policy μ_i, we now inspect the value V from a perspective of the generalized Hamilton-Jacobi-Bellman (GHJB) equation (Beard and McLain, 1998) where V approximates the cost of current policy μ_i if and only if V satisfies the GHJB:

$$\left(f(x) + g(x)\mu_i(x)\right)^T \frac{\partial V}{\partial x}(x) + x^T Q x + \mu_i^T(x) R \mu_i(x) = 0. \tag{18}$$

This will allow us to integrate the key integral reinforcement equation (13) for a value and policy update.

Now, note that as $x \to 0$, we have $(f(x) + g(x)\mu_i(x)) \to \xi(x) = Ax + B\mu_i(x)$ at a rate $o(\|x\|)$. Thus, combining the value function approximator structure $V(x) = x^T P_i x$ with the GHJB equation (18) implies to the first order that

$$2\xi^T(x)P_i x + x^T Q x + \mu_i^T(x) R \mu_i(x) \approx 0. \tag{19}$$

Here, $\approx$ denotes first-order approximation. Returning to (17), the approximate GHJB equation (19) now shows that

$$x^T(t_1)P_ix(t_1) - x^T(t_0)P_ix(t_0) + \int_{t_0}^{t_1} x^TQx + \mu_i^TR\mu_i\,d\tau \approx 2\int_{t_0}^{t_1}(f(x)+g(x)u-\xi)^TP_ix\,d\tau,$$

or, rearranging

$$-2\int_{t_0}^{t_1}(f(x)+g(x)u-\xi(x))^TP_ix\,d\tau + \left[x^T(t_1)P_ix(t_1) - x^T(t_0)P_ix(t_0)\right] \approx -\int_{t_0}^{t_1} x^TQx + \mu_i^T(x)R\mu_i(x)\,d\tau. \quad (20)$$

The integral reinforcement equation (20) is now of the required form for learning: The right-hand-side of (20) incorporates the key integral reinforcement equation (13); crucially, it requires only knowledge of the current policy μ_i, trajectory data $\{x(t)\}_{t\in[t_0,t_1]}$ from the true nonlinear process, and the nominal model (f,g) (1) as opposed to exact knowledge of the actual nonlinear process, which cannot be known *a priori*. The left-hand-side contains the environment trajectory integral and difference data to form the critic weight update matrix in Step 5 below.

Step 5. Using the identification $c_i = v(P_i)$ from Proposition 3, and applying the algebraic identity (8) to the left-hand-side of (20) yields

$$\left[-2\int_{t_0}^{t_1}\Phi\big(f(x)+g(x)u-\xi(x),\,x\big)\,d\tau + \Phi\big(x(t_1)+x(t_0),x(t_1)-x(t_0)\big)\right]^T c_i \approx -\int_{t_0}^{t_1} x^TQx + \mu_i^T(x)R\mu_i(x)\,d\tau. \quad (21)$$

The equation (21) is of the final form required for a learning update, as all terms pertaining to the value function V via the weights c_i now appear as a linear regression to be solved in order to update the value function network weights. In particular, the terms in brackets on the left-hand-side will form a single row of the learning matrix Θ_i (23), multiplied on the right by the network weight vector c_i. Meanwhile, the integral reinforcement signal on the right-hand-side will form a single element of the learning vector Ξ_i (24), establishing an integral reinforcement learning update.

Step 6. We now use the integral reinforcement (21) (which comprises a single trajectory sample) to construct a value function weight update from $l \in \mathbb{N}$ such samples. Given a designer-selected state-action sample count $l \in \mathbb{N}$, sequence of sample instants $\{t_k\}_{k=0}^l$, and reference command input $r(t)$ (see discussions around Equation (9)), we apply $r(t)$ to system (1) under an initial stabilizing policy μ_0. In turn, we collect the resulting state-action trajectory data $\big\{\big(x(t),u(t)\big)\big\}_{t\in[t_0,t_l]}$. Applying (21) at the sample instants $\{t_k\}_{k=0}^l$ we arrive at a new updated weight from solving the following equation:

$$\Theta_i\,c_i = \Xi_i, \quad (22)$$

where the learning matrices $\Theta_i \in \mathbb{R}^{l \times \underline{n}}$, $\Xi_i \in \mathbb{R}^l$ are given from (21) by

$$\Theta_i = \begin{bmatrix} -2\int_{t_0}^{t_1} \Phi^T\big(f(x)+g(x)u-\xi(x),\, x\big)\, d\tau + \Phi^T\big(x(t_1)+x(t_0), x(t_1)-x(t_0)\big) \\ \vdots \\ -2\int_{t_{l-1}}^{t_l} \Phi^T\big(f(x)+g(x)u-\xi(x),\, x\big)\, d\tau + \Phi^T\big(x(t_l)+x(t_{l-1}), x(t_l)-x(t_{l-1})\big) \end{bmatrix}, \tag{23}$$

$$\Xi_i = -\begin{bmatrix} \int_{t_0}^{t_1} x^T Q x + \mu_i^T(x) R \mu_i(x)\, d\tau \\ \vdots \\ \int_{t_{l-1}}^{t_l} x^T Q x + \mu_i^T(x) R \mu_i(x)\, d\tau \end{bmatrix}. \tag{24}$$

Step 7. Having solved for the critic weights c_i (22), we update the policy μ_{i+1} from the updated value network $V(x) = \Phi^T(x,x)c_i$ via (12). The recursive process leads to the optimal solution of the affine nonlinear optimal control problem in Section 3.1.

The RCI-based CT-RL algorithm is summarized in Algorithm 1.

Algorithm 1 RCI-Based CT-RL.

1: **Hyperparameters:** Cost Q, R (2), number of data samples l, sample instants $\{t_k\}_{k=0}^{l}$, number of iterations i^*, reference command input r, initial stabilizing policy μ_0.
2: **Data Collection:** Apply reference command input $r(t)$ to system (1), starting from initial condition $x_0 \in \mathbb{R}^n$ simulating under policy μ_0, collecting state-action data $\big\{\big(x(t), u(t)\big)\big\}_{t\in[t_0,t_l]}$.
3: **Learning:**
4: **for** $i = 0, \ldots, i^* - 1$ **do**
5: Construct learning matrices Θ_i (23) and Ξ_i (24) from collected state action-data $\big\{\big(x(t), u(t)\big)\big\}_{t\in[t_0,t_l]}$ and current policy μ_i.
6: Perform critic network weight update c_i (22) from learning matrices Θ_i (23), Ξ_i (24).
7: Update critic network $V(x) \leftarrow \Phi^T(x,x)c_i$ (10).
8: Update policy μ_{i+1} via (12).
9: **end for**
10: **Termination:** Final policy μ_{i^*}.

4. Theoretical Results

Our choice of critic network structure V (10) and control policy structure μ_i (12) allows us to take advantage of classical control results in Kleinman's well-tested algorithm (Kleinman, 1968). These classical principles, combined with our use of state-action data (x, u) from the nonlinear environment in learning, enable us to develop our key convergence, optimality, and closed-loop stability guarantees.

Theorem 4 (Convergence, Optimality, and Closed-Loop Stability of RCI) *Suppose that the initial policy μ_0 of the form (12) is such that $\frac{\partial}{\partial x}\{f(x) + g(x)\mu_0(x)\}$ is Hurwitz at the origin (i.e., μ_0 is asymptotically stabilizing). Suppose also that the sample*

instants $\{t_k\}_{k=0}^{l}$ *are such that the integral reinforcement matrix* $\Theta \in \mathbb{R}^{l\times \underline{n}}$ *defined by*

$$\Theta = \begin{bmatrix} \int_{t_0}^{t_1} \Phi(x,x)\, d\tau & \cdots & \int_{t_{l-1}}^{t_l} \Phi(x,x)\, d\tau \end{bmatrix}^T \tag{25}$$

has full rank $\underline{n}$*. Then, identifying* $c_i = v(P_i)$ *(Proposition 3), RCI produces identical sequences of matrices* $\{P_i\}_{i=0}^{\infty}$ *as Kleinman's algorithm (Kleinman, 1968) to the first-order approximation if the Kleinman control sequence is produced based on a known linearized nonlinear process that is unknown to RCI. Thus, for RCI in the choice of critic bases* $\Phi(x,x) \in \mathbb{R}^{\underline{n}}$ *on the nonlinear system (1), we have:*

(i) $P^* \leq P_{i+1} \leq P_i$ *for all* $i \geq 0$*, and* $\lim_{i\to\infty} P_i = P^*$*, where* $P^* \in \mathbb{R}^{n\times n}$*,* $P^* = P^{*T} > 0$ *is the solution of the Riccati equation associated with* (A, B) *(16).*

(ii) $\lim_{i\to\infty} \mu_i = -K^*$ *in operator norm, where* $K^* \in \mathbb{R}^{n\times m}$ *is the optimal LQ control.*

Proof: An induction argument presented in Appendix B. ■

Insights: Theorem 4 is a theoretical guarantee of great practical utility to designers. It assures convergence and closed-loop stability of the RCI policy iterates $\{\mu_i\}_{i=0}^{\infty}$, as well as optimality of the limit policy to first-order approximation. We note that Kleinman's algorithm (Kleinman, 1968) only addresses fully-known linear dynamics (A, B) without model uncertainty, while RCI's learning and execution is on the actual nonlinear process with nonlinear uncertainties beyond the nominal system model (f, g) (1). As will be shown, RCI's use of nonlinear trajectory data $\{x(t)\}_{t\geq 0}$ generated by the actual physical process allows our learning-based design to converge to a policy which is much closer to optimal than the classical LQ controller (cf. Section 6). This is because the classical LQ design method does not capture 1) the system nonlinearity, and 2) possible model uncertainty, while both are fully accommodated by RCI. Another key advantage offered by RCI is that it replaces the formal PE requirement, for which there does not exist a closed-form test for nonlinear systems (Shimkin and Feuer, 1987), with a constructive test as shown in (25).

Remark 5 (Theoretical Results and Assumptions of SOTA CT-RL Works) *RCI, like the leading ADP-based works (Vrabie and Lewis, 2009; Vamvoudakis and Lewis, 2010; Jiang and Jiang, 2014; Bian and Jiang, 2022) provides theoretical results of convergence, optimality, and closed-loop stability. The SOTA FVIs (Lutter et al., 2021, 2023b) do not provide these guarantees. RCI's assumptions are among the least stringent in CT-RL, which we outline in detail in Appendix A.1. RCI requires standard stabilizability, detectability, and full-rank assumptions for well-posedness. Meanwhile, DRL FVIs (Lutter et al., 2021, 2023b) require partial derivative information of* f *and* g*. RCI's theoretical results are not global in nature, which is by design. Global asymptotic stability results for nonlinear systems generally require extremely stringent theoretical assumptions on both the structure of the environment dynamics and the knowledge of the dynamics. For instance, the global results of CT-VI (Bian and Jiang, 2022) require PE, existence and uniqueness of solutions to an uncountable family of finite-horizon HJB equations, and an initial globally asymptotically stabilizing policy. The ADP methods generally have the most stringent assumptions.*

Remark 6 (Decentralizable Environment for Further Data Efficiency) *Consider a decentralized environment (f, g) (1) with N separable control loops. To illustrate, we present $N = 2$ loops:*

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} f_1(x) \\ f_2(x) \end{bmatrix} + \begin{bmatrix} g_{11}(x) & g_{12}(x) \\ g_{21}(x) & g_{22}(x) \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}. \tag{26}$$

No assumptions are made on dynamic coupling between the loops; i.e., the loops may be fully coupled. Here, $x_j \in \mathbb{R}^{n_j}$, $u_j \in \mathbb{R}^{m_j}$ $(j = 1, \ldots, N)$ with $\sum_{j=1}^{N} n_j = n$ and $\sum_{j=1}^{N} m_j = m$. Such partitions appear in a variety of real-world applications such as robotic systems (Craig, 2005; Dhaouadi and Abu Hatab, 2013), helicopters (Enns and Si, 2002, 2003b,a), UAVs (Wang et al., 2016), and aircraft (Stengel, 2022; Dickeson et al., 2009a,b) In this case, the RCI learning rule (22) occurs in a decentralized fashion in each of the loops, thereby reducing problem dimensionality. This results in sequences of critic network weights $\{c_{i,j}\}_{i=0}^{\infty}$ and policies $\{\mu_{i,j}\}_{i=1}^{\infty}$ in each loop yielded by learning analogous to (22), now constructed with x_j instead of x, u_j instead of u, etc. Crucially, learning convergence, optimality, and closed-loop stability results analogous to Theorem 4 hold for the policies $\{\mu_{i,j}\}_{i=1}^{\infty}$ in each loop. For a complete formulation and proof of this result, see Section 6 of Supplemental Material.

5. Experiment Setup for Evaluations

Even though we have shown RCI-based CT-RL to have theoretical guarantees of learning convergence, solution optimality, and system stability in Section 4, these results are only qualitative. It is important to quantitatively and systematically evaluate how well-behaved the RCI control is in terms of both learning and control system performances. Our evaluations below include (typical) comprehensive evaluations of learning performance such as average return, learning success rate, and data efficiency. Additionally, we provide two new evaluations: controller optimality and control system performance, the latter of which is essential for continuous state and control applications problems, which require such evaluations as basic requirements of control system design.

5.1 Baselines

The evaluations involve four **baseline** methods, among which the Nominal LQ and the Optimal LQ are also used as **benchmarks** for assessing RCI policy optimality:

- "**Continuous FVI (cFVI)**": SOTA deep CT-RL method (Lutter et al., 2021).
- "**Robust FVI (rFVI)**": Robust variant of SOTA FVI (Lutter et al., 2023b).
- "**Nominal LQ**": The classical LQR design performed on the linearized model (A, B) of the nominal nonlinear system (f, g) (1). Its design does not take into account system nonlinearity and model uncertainty.
- "**Optimal LQ**": The classical LQR design performed on the linearization of the actual nonlinear process, where modeling errors are included as if the designer knows the uncertainty *a priori*. Its design does not take into account system nonlinearity.

Among the baseslines, the FVIs are used for comparisons for both learning performance and control system performance. Results involving the Nominal LQ and the Optimal LQ can be viewed as ablation studies, and the Optimal LQ also serves as a benchmark in order to measure control optimality.

5.2 Performance Measures

In these studies, we provide comprehensive evaluations of standard performance measures in: average return, learning success rate, generalization with respect to system initial conditions (ICs) and modeling error, and time and data efficiency as well as free parameter complexity. In addition, we analyze performance with respect to the following learning control measures:

- Cost Performance: The infinite-horizon cost $J(x_0)$ obtained via the integral (2) delivered by the policy in the nonlinear optimal control task. As a note, in controls conventions the cost $J(x_0) > 0$ is a positive number to be minimized (lower is better).
- Relative Cost Performance: The difference $J_{\star FVI}(x) - J_{RCI}(x)$ between the cost performance of the respective FVI algorithm (i.e., $\star =$ c or r) and RCI. Note that wherever this difference is positive, RCI delivers *better* performance than the respective FVI algorithm because this corresponds to RCI having a *lower* cost.
- Estimation Error: The difference $J(x) - V(x)$ between the cost $J(x)$ (2) at state $x \in \mathbb{R}^n$ and the value function approximation $V(x)$ (10) at $x \in \mathbb{R}^n$. Smaller difference is better (means that the critic is more accurate).
- Policy Optimality Error: The difference $\|\mu - K^*\|$ in operator norm between a given policy μ and the Optimal LQ policy K^* (i.e., K^* is optimal with respect to the actual physical process containing the model uncertainty). This is used to provide a quantitative measure of how much the RCI design based on nonlinear dynamics (1) improves over that on the linearized dynamics (A, B).

In addition, we measure the following closed-loop time-domain performance measures which are central to the continuous-time nonlinear dynamical control task studied:

- "**90% Rise Time** $t_{r,y,90\%}$: The time taken for the closed-loop response $y(t)$ to rise to 90% of its commanded value.
- "**1% Settling Time** $t_{s,y,1\%}$: The time taken for the closed-loop response $y(t)$ to settle within $\pm 1\%$ of its commanded value.
- "**Overshoot** $M_{p,y}$: The maximum by which the closed-loop response $y(t)$ exceeds the step reference command $r(t) \equiv r$. Expressed as: $M_{p,y} = \max_{t \geq 0} (y(t) - r)/r \ \times 100\%$.

Success of a Learning Trial. A learning trial is a "success" if the policy generated stabilizes the closed-loop system; specifically, if the state $\|x\| \leq 0.01$ at time $t = 25$ s.

5.3 Questions Addressed

Our evaluations aim to quantitatively address the following:

Q1: Does RCI CT-RL lead to learning convergence, control policy optimality, and closed-loop stability as theoretically guaranteed (cf. Theorem 4)? How close to optimal are the policies under modeling error, and how much does RCI learning improve solution optimality over a nominal classical LQ design?

Q2: How does RCI CT-RL learning performance (average return, learning variance, and success rate) compare to baseline FVIs?

Q3: How is RCI cost performance affected by system modeling errors in comparison to baseline FVIs?

Q4: How is RCI critic network estimation error affected by system modeling errors in comparison to baseline FVIs?

Q5: How well do RCI closed-loop time responses generalize with respect to system modeling

error when compared to baseline FVIs?
Q6: How data efficient is RCI compared to FVIs?

5.4 Environments, Code, and Data

As CT-RL benchmarks are not as established as their DT-RL counterparts, our selection of the three environments used in this study are based on the following considerations: 1) The pendulum environment may be considered the most popular benchmark for CT-RL so far, as most studies have used it. It is also used in SOTA DRL evaluations of FVIs, we therefore used the identical setup as in (Lutter et al., 2021, 2023b). 2) The jet and the DDMR environments are from real physical systems. We use them to evaluate how SOTA CT-RL methods perform toward real-life meaningful applications. 3) As such, these environment selections are SOTA. A detailed comparison with leading ADP and DRL approaches to CT-RL is provided in Appendix A and Table 6. Physical motivations and insights of the models can be found in Sections 3–5 of Supplemental. The dynamics of each are given by

$$\begin{array}{rcl|rcl} \dot{\theta} & = & \omega & \dot{V} & = & \frac{m_c d}{\hat{m}}\omega^2 - \frac{2\overline{\beta}}{\hat{m}r^2}V + \frac{k_t}{\hat{m}k_g r}i_{a_r} + \frac{k_t}{\hat{m}k_g r}i_{a_l} \\ \dot{\omega} & = & \frac{mgL}{2I}\sin\theta + \frac{\tau}{I} & \dot{\omega} & = & \frac{-m_c d}{\hat{I}}\omega V - \frac{\overline{\beta}d_w^2}{2\hat{I}r^2}\omega + \frac{d_w k_t}{2\hat{I}k_g r}i_{a_r} - \frac{d_w k_t}{2\hat{I}k_g r}i_{a_l} \\ & & & \dot{i}_{a_r} & = & \frac{-k_g k_b}{l_a r}V - \frac{k_g k_b d_w}{2l_a r}\omega - \frac{r_a}{l_a}i_{a_r} + \frac{1}{2l_a}\overline{e}_a + \frac{1}{2l_a}\Delta e_a \\ & & & \dot{i}_{a_l} & = & \frac{-k_g k_b}{l_a r}V + \frac{k_g k_b d_w}{2l_a r}\omega - \frac{r_a}{l_a}i_{a_l} + \frac{1}{2l_a}\overline{e}_a - \frac{1}{2l_a}\Delta e_a \end{array} \tag{27}$$

$$\begin{bmatrix} \dot{V} \\ \dot{\gamma} \\ \dot{q} \\ \dot{\alpha} \end{bmatrix} = \begin{bmatrix} -D_V & -g\cos\alpha_e & 0 & 0 \\ \frac{L_V}{V_e} & 0 & 0 & \frac{L_\alpha}{V_e} \\ 0 & 0 & M_q & M_\alpha \\ \frac{-L_V}{V_e} & 0 & 1 & \frac{-L_\alpha}{V_e} \end{bmatrix} \begin{bmatrix} V \\ \gamma \\ q \\ \alpha \end{bmatrix} + \begin{bmatrix} T_{\delta_T} & 0 \\ 0 & 0 \\ 0 & M_{\delta_E} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_T \\ \delta_E \end{bmatrix}. \tag{28}$$

- **The pendulum** (27, left) is chosen identical to the model in the original FVI works (Lutter et al., 2021, 2023b). It has states $x = (\theta, \omega)$, where θ is the pendulum angle (measured zero pointing upward, positive counterclockwise), ω is the pendulum angular velocity, and the single-input control $u = \tau$ is the torque applied to the pendulum base.
- **The jet** aircraft (28) is derived from full-scale NASA wind tunnel data (Soderman and Aiken, 1971) and is a standard flight control benchmark (Stengel, 2022). It has states $x = (V, \gamma, q, \alpha)$, where V is the airspeed, γ is the flightpath angle (FPA), q is the pitch rate, and α is the angle of attack. It has controls $u = (\delta_T, \delta_E)$, where δ_T is the throttle setting (associated with the airspeed V in the translational loop $j = 1$), and δ_E is the elevator deflection (associated with the FPA γ and attitude q, α in the rotational loop $j = 2$).
- The differential drive mobile robot (**DDMR**) (27, right) is obtained from system ID on actual hardware (Mondal et al., 2020, 2019). The DDMR has states $x = (V, \omega, i_{a_r}, i_{a_l})$, where V is the velocity, ω is the angular velocity, and i_{a_r}, i_{a_l} are the right and left DC motor armature currents, respectively. The controls are $u = (\overline{e}_a, \Delta e_a)$, where $\overline{e}_a$ is the average of the armature voltages applied to the right and left DC motors (associated with the speed V in the translational loop $j = 1$), and Δe_a is the difference of the right/left voltages (associated with the rotational velocity ω in the rotational loop $j = 2$).

All Code/Data Available. All RCI code and datasets for this study are available in Supplemental and at (Wallace and Si, 2024c). All FVI results (Lutter et al., 2021, 2023b) are generated by the open-source code developed by the authors at (Lutter et al., 2023a).

5.5 Implementations

Training Procedure. An episode is initialized by resetting the environment and terminated at time T of the training horizon for collecting the state-action trajectory data (x, u). A trial is a complete training process that contains a series of consecutive episodes. RCI learning requires state-action trajectory data from a single episode which usually has on the order of $l = 100$ total samples for the three evaluated environments. This low data complexity allows RCI to learn online from the actual physical process. Deep RL FVIs require training data from over 5 million episodes (cf. Table 5), for details see (Lutter et al., 2021, 2023b). As a result, the only practical means of training FVI is in simulation.

Reference Command Input Selection. Numerical selections of the reference command inputs $r(t)$ can be found in Table 7 of Appendix C. We choose the reference command $r(t)$ based on classical input-output principles by placing the dominant frequency content near the peak of the closed-loop map from r to the output y (cf. Figure 1). This choice of frequency maximizes excitation efficiency (Ogata, 1997; Rodriguez, 2004).

Random Seeds. Training and evaluation for each of the methods are based on 20 seeds for random number generation (RNG): 0–19 for training, and 100–119 for evaluation. In the case of the FVIs, the seeds are used in the environment, Numpy, and PyTorch for number generation. For RCI, we have set MATLAB's master RNG seed for number generation.

Network Weight Initialization. For FVIs' deep networks, we use the identical network initialization procedure as in the original works (Lutter et al., 2021, 2023b); namely, Xavier normal distribution. The initialization gains of the layers can be found in Table 8 of Appendix C.2, the same as in previous studies (Lutter et al., 2021, 2023b). For RCI's quadratic network structure, we need only initialize the critic weights c_0 in (10). The setup of the uniform distributions used for all three environments is summarized in Appendix C.

Initializing the Environments. System initial conditions (ICs) for training and evaluation are generated using uniform distributions $\mathcal{U}$, where the ranges for the pendulum, jet aircraft, and DDMR cover the dynamics broadly, well beyond their linear regimes for evaluation as well. These distributions for each environment are provided in Appendix C. Note that, for the pendulum system, we have chosen the identical uniform distribution $\mathcal{U}$ for state initialization as in the original FVI studies (Lutter et al., 2021, 2023b). The only exception to the above uniform IC generation procedure is the systematic grid sweeps conducted in the RCI initial condition and modeling error generalization studies of Section 6. Here, for comprehensive evaluations, the ICs x_0 are swept over a grid of values $x_0 \in G_{x_0}$, which are likewise given in Appendix C.

Generalization under Modeling Error. Control performance under modeling errors is an essential measure of the quality of a controller. In real-life applications, the modeling error is usually unknown. Thus, learning provides both RCI and FVIs an opportunity to adapt to unmodeled dynamics in the environment. Here we consider several different scenarios to demonstrate system behaviors under different control designs given certain levels of modeling errors ν. This is a test of robustness or the ability to generalize of a controller. We thus report systematic results given $\nu = 0$–25% modeling errors. Implementation details are provided in Appendix C. Furthermore, Sections 3–5 of the Supplemental Material provide in-depth discussions of the dynamics of each of the environments studied, as well as the exact points structurally where the individual modeling error parameters enter the respective

dynamical equations. This work studies effects due to multiplicative model uncertainties, which is a standard formulation in CT-RL works (Lutter et al., 2023b, 2021), as well as classical control (Rodriguez, 2004) and aerospace applications (Stengel, 2022). However, by no means is RCI limited to multiplicative model uncertainties; indeed, nowhere is the model uncertainty structure required explicitly in Section 3. RCI is afforded this flexibility by collecting state-action data (x, u) directly from the nonlinear process with its embedded uncertainties, multiplicative or otherwise.

Evaluation Procedure. At each algorithm iteration, the respective policy of RCI, cFVI, and rFVI is evaluated over 100 episodes of the environment. The environment ICs are generated according to uniform distributions given in Appendix C. For display purposes of generating the surface plots in Figures 3 and 4, we then select a single seed for evaluation and present data for 0%, 10%, and 25% modeling errors. The resulting final policies are then evaluated with respect to their relative costs $J_{\star FVI}(x) - J_{RCI}(x)$ (cf. Section 5.2 for performance metric definitions) and value function approximation errors $J(x) - V(x)$ for all states x in evaluation grids G_x given in Appendix C.

Finally, for evaluating closed-loop time responses, we issue the following step reference commands: For the pendulum, we initialize it at full displacement (free hanging) $\theta_0 = \pi$, $\dot{\theta}_0 = 0$ and stabilize at the upright position $r(t) = 0$ rad. For the jet aircraft and DDMR, we initialize ICs to equilibrium $x_0 = x_e$. For the jet, we issue a command in velocity $y_1 = V$ of $r_1(t) = 110$ ft/s (i.e., 10 ft/s faster than equilibrium $V_e = 100$ ft/s), and we issue a command in flightpath angle $y_2 = \gamma$ of $r_2(t) = 1$ deg. For the DDMR, we issue a reference command in velocity $y_1 = V$ of $r_1(t) = 3$ m/s (i.e., 1 m/s faster than equilibrium $V_e = 2$ m/s), and we issue a reference command in angular velocity $y_2 = \omega$ of $r_2(t) = 30$ deg/s.

6. RCI Learning with Nominal LQ and Optimal LQ as Benchmarks

In this evaluation, we examine how RCI's convergence, optimality, and closed-loop stability guarantees (Section 4) generalize with respect to systematic sweeps of varying system initial conditions (ICs) and modeling errors (Section 5). This also serves as an ablation, as the Nominal LQ and Optimal LQ (see Section 5.1 for definitions) do not account for environment nonlinearities as RCI does.

Q1: RCI CT-RL leads to learning convergence, solution optimality, and closed-loop stability as theoretically guaranteed. RCI reaches controller optimality with a factor of ten improvement over that of the Nominal LQ design. We initialize the environments with the uniform distributions described in Section 5.5. A complete evaluation of RCI based on the nominal model ($\nu = 0\%$) and under modeling errors ($\nu = 10\%$ and $\nu = 25\%$) for all three environments can be found in Tables S1–S3 of Supplemental. For illustration, a representative subset of this data is reproduced in Table 1. As can be seen, RCI converges to the respective optimal policy with and without modeling error with little residual on the pendulum, jet, and DDMR environments.

Tables S1–S3 of Supplemental, with a subset reproduced in Table 1, summarize the optimality errors $\|\mu_{i^*} - K^*\|$ between the RCI final policies μ_{i^*} and the Optimal LQ policy K^*, even under severe modeling errors. Specifically, RCI provides a significantly improved policy over the Nominal LQR design (based on linearized dynamics) with over 90% reduction in optimality error. This result is in agreement with the theoretical guarantees (Theorem 4).

Note however, the Optimal LQ policy K^* provides a means to directly measure optimality of a RCI policy. This does not mean that RCI is an LQ design. As will be shown, the time responses of the Nominal LQ, Optimal LQ, and RCI are different, as they should be by design. Notice also that it is the RCI learning via a combination of nonlinear dynamic knowledge of the environment and the state-action trajectory data (x, u) that enables RCI's optimality edge over the nominal classical (Nominal LQ) design.

Table 1: RCI learning convergence and optimality properties with and without modeling error ($\nu = 0\%, 10\%, 25\%$)

System	ν	Nom LQ optimality error $\|K_0 - K^*\|$	RCI policy optimality error $\|\mu_{i^*} - K^*\|$	% error reduction RCI $\rightarrow$ Nom LQ
Pendulum	0%	0	1.75e-05±1.19e-05	N/A
	10%	1.04	0.03±5.47e-03	96.75±0.53
	25%	2.61	0.13±0.01	95.21±0.54
Jet	0%	0	1.93e-08±2.54e-09	N/A
	10%	0.05	1.71e-08±2.29e-09	99.99±7.06e-06
	25%	0.13	1.38e-08±1.94e-09	99.99±1.55e-06
DDMR	0%	0	1.19e-05±9.06-e06	N/A
	10%	0.68	0.06±0.03	91.06±4.07
	25%	1.74	0.10±0.06	94.25±3.23

7. Comparisons between RCI and Deep RL (FVIs)

This evaluation focuses on comparing RCI to current SOTA deep RL methods cFVI and rFVI (Lutter et al., 2021, 2023b). Training and evaluation procedures for all algorithms are described in detail in Section 5 and Appendix C.

Q2: RCI exhibits the fastest learning convergence of the three methods on all environments. Except for cFVI on the pendulum benchmark, RCI also delivers best return and lowest variance. The learning curves for RCI and FVIs are plotted in Figure 2 where RCI is trained out to FVIs iteration count for comparison purposes. The average return, variance, and success rate over 20 training seeds are tabulated in Table 2. As can be seen from Table 2, all methods successfully stabilize the closed-loop system for the three environments. The FVI algorithms exhibit overall consistent learning behavior which corroborates the original results (Lutter et al., 2021, 2023b). Additionally, cFVI edges out RCI on the pendulum task, delivering a higher average return and lower variance. However, for all environments RCI converges faster (within 10 iterations, cf. Figure 2) than the FVIs (usually 50–100 iterations). On the two higher-dimensional, multi-input jet aircraft and DDMR models studied, RCI exhibits the best average return and the lowest variance.

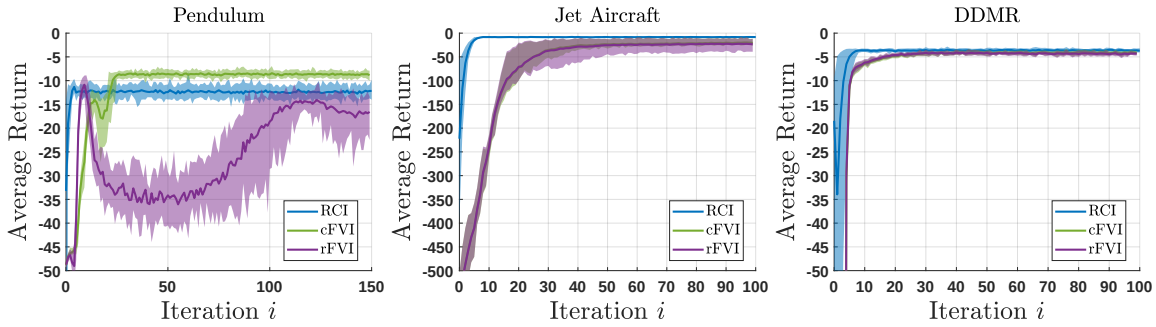

Figure 2: Learning curves of RCI and FVIs for the nominal ($\nu = 0\%$) pendulum (left), jet aircraft (middle), and DDMR (right). Note: Closer to 0 is better. The shaded areas cover the entire range of the evaluations over 20 seeds (100–119), represented the same way as in the original works (Lutter et al., 2021, 2023b).

Table 2: Average return and success rate on nominal model ($\nu = 0\%$) for three environments

Algorithm	Pendulum		Jet aircraft		DDMR	
	Success [%]	Return $[\mu \pm 2\sigma]$	Success [%]	Return $[\mu \pm 2\sigma]$	Success [%]	Return $[\mu \pm 2\sigma]$
RCI	100	-12.17 ± 2.30	100	-8.26 ± 1.21	100	-3.69 ± 0.51
cFVI	100	-8.85 ± 1.12	100	-20.50 ± 7.26	100	-4.12 ± 0.44
rFVI	100	-16.58 ± 5.36	100	-23.37 ± 16.04	100	-4.32 ± 0.46

Table 3: Training cost difference data $J_{\star FVI} - J_{RCI}$ (> 0: RCI better)

Alg	Data	Pendulum		Jet aircraft		DDMR	
		$\nu = 0\%$	$\nu = 25\%$	$\nu = 0\%$	$\nu = 25\%$	$\nu = 0\%$	$\nu = 25\%$
cFVI	min	-0.23	-0.02	0.00	0.00	1.11e-05	1.08e-05
	max	4.62e-04	0.51	8.04	8.57	0.27	0.41
	avg	-0.02±0.04	0.12±0.12	3.08±1.80	3.74±1.98	0.09±0.08	0.13±0.11
rFVI	min	-1.33e-06	-3.81e-04	0.00	0.00	1.77e-03	1.77e-03
	max	7.72	10.27	10.15	10.34	2.27	1.60
	avg	2.16±1.92	2.99±2.61	3.72±2.23	4.22±2.29	0.63±0.54	0.46±0.38

Q3: RCI cost performance is generally better than FVIs, and generalizes well to modeling error. Figure 3 illustrates the cost differences between respective FVIs and RCI, as summarized in Table 3. Some patterns emerge from Table 3: 1) The FVIs perform well. Indeed, the top left plot in Figure 3 shows that cFVI edges out RCI for the nominal pendulum far from the origin $x = 0$. However, when modeling error is introduced (top

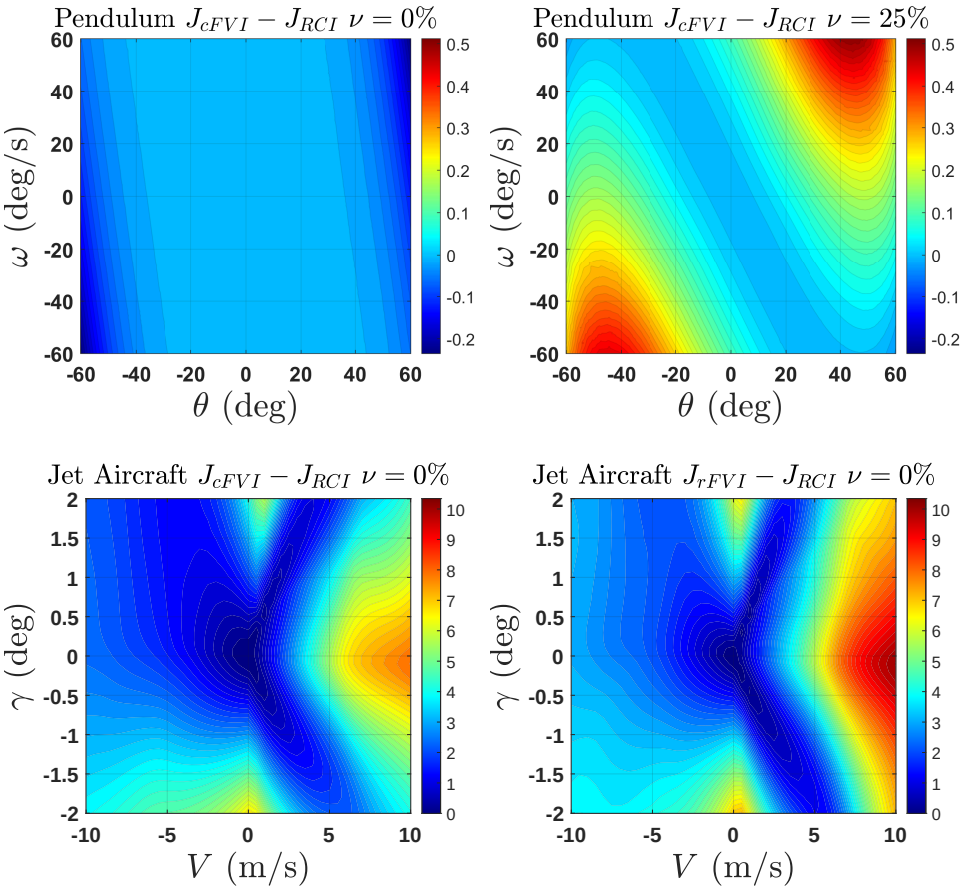

Figure 3: Top row: cFVI cost difference for the nominal pendulum model (left) and at 25% modeling error (right). Bottom row: cFVI (bottom left) and rFVI (bottom right) cost difference for the nominal jet aircraft model. Note that cFVI outperforms RCI on the nominal pendulum near the edges of the state domain (top left), but the advantage observed in these same regions erodes when modeling error is introduced (top right). Note from the bottom row that RCI delivers the lowest cost pointwise for the jet. Full plots can be found in Supplemental Figures S3–S5.

right), cFVI performance degrades, a trend we observe for all three environments (cf. Table 3 and Supplemental Figures S3–S5). By contrast, rFVI degradation is less pronounced, but at the cost of inferior overall performance. 2) RCI achieves the lowest cost for all three environments as modeling error ν increases, and generalizes the best over modeling errors (cf. Table 3). 3) For both multi-loop systems (i.e., the jet and DDMR), RCI achieves lowest cost pointwise and exhibits the best generalization to modeling error (cf. Table 3).

Q4: RCI critic estimation error exhibits best generalization to modeling error. Tables S5–S7 of Supplemental summarize the critic network approximation error performance $J - V$ of RCI and FVIs for all three environments, and this data is plotted in

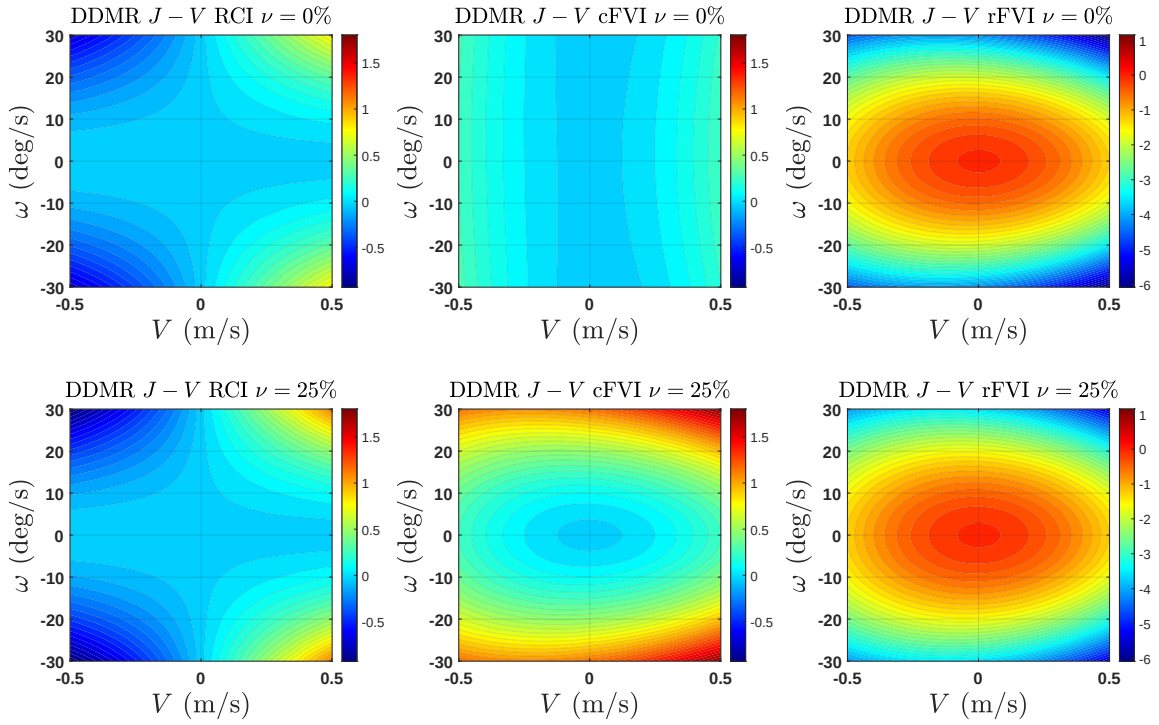

Figure 4: Critic NN approximation error $J(x) - V(x)$ for DDMR environment on the nominal model (top row) and at 25% modeling error (bottom row) for RCI (left), cFVI (middle), and rFVI (right). RCI exhibits the most consistent approximation behavior, varying little with modeling error. cFVI's critic is highly accurate on the nominal model, but degrades significantly with modeling error. rFVI struggles the most, consistently underestimating policy performance to a large degree. Full plots can be found in Supplemental Figures S6–S8.

Figures S6–S8 of Supplemental. We have reproduced some if this data here in Figure 4, which shows the critic network approximation error $J - V$ on the DDMR. As is the case with cost performance, cFVI does an excellent job of approximating its policy cost for the nominal model but experiences significant degradation. RCI exhbits the smallest critic network error when modeling error is introduced and the best generalization overall. On the DDMR, for instance, Table S7 of Supplemental shows that cFVI's worst-case critic estimation error increases by 445% from $\nu = 0\%$ to $\nu = 25\%$, as compared to RCI's 39%. Meanwhile, rFVI struggles to a larger degree than RCI or cFVI; however, rFVI's worst-case approximation improves from 6.08 at nominal to 5.36 at 25% modeling error, demonstrating favorable robustness/learning generalization. For comparison, RCI estimation error degrades, but only from 0.82 at nominal to 1.15 at 25% modeling error. The estimation error advantage for RCI on the jet is even more pronounced (Figure 5), exhibiting the best approximation and generalization. We attribute RCI's favorable performance on the jet to the well-established

successes of decentralized LQ control methods on aerospace applications; see, e.g, (Stengel, 2022; Dickeson et al., 2009a,b).

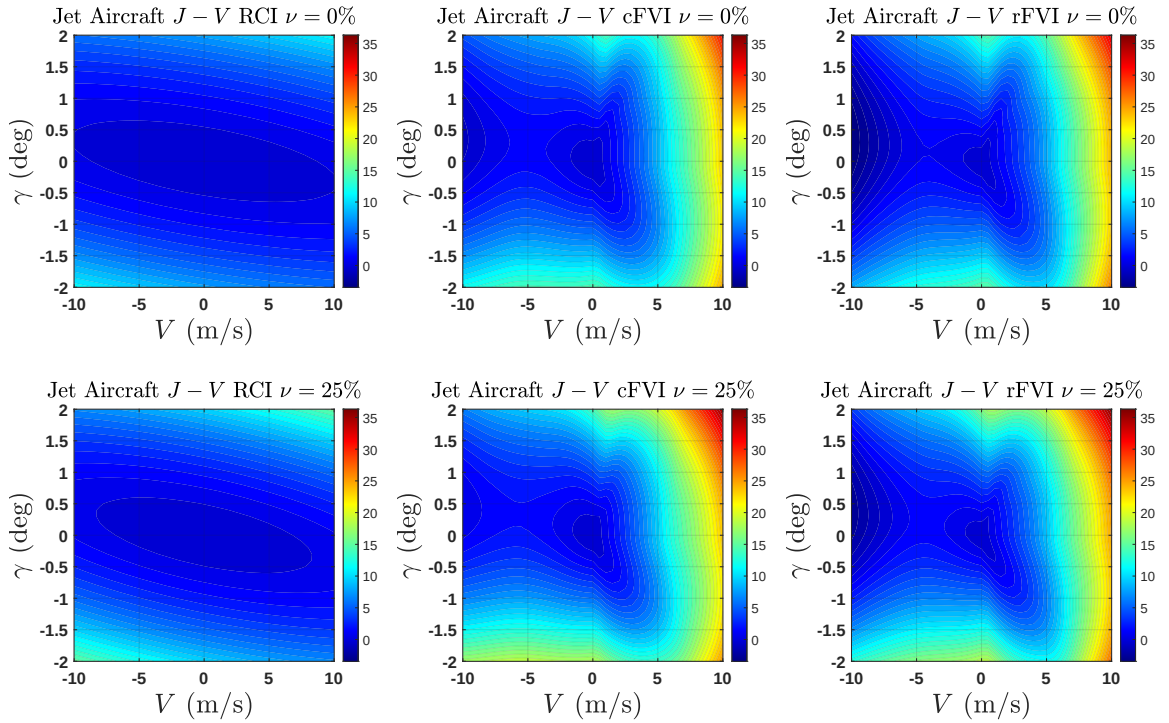

Figure 5: Critic NN approximation error $J(x) - V(x)$ for jet aircraft environment on the nominal model (top row) and at 25% modeling error (bottom row) for RCI (left), cFVI (middle), and rFVI (right). RCI exhibits the lowest approximation error and best generalization. cFVI and rFVI perform comparably to each other. Full plots can be found in Supplemental Figures S6–S8.

Q5: RCI closed-loop time responses are the fastest with least overshoot, best modeling error generalization. We issue step reference commands to the three environments in each of their output channels y and record the associated 1% settling time $t_{s,y,1\%}$, 90% rise time $t_{r,y,90\%}$, and percent overshoot $M_{p,y}$ in Table 4. Figure 6 displays closed-loop responses for all three environments at 25% modeling error. Overall, the FVI responses are sluggish compared to RCI. For instance on the jet aircraft in the FPA loop $y_2 = \gamma$ (cf. Table 4), cFVI and rFVI have 1% settling times $t_{s,\gamma,1\%}$ of 14.42 s and 17.88 s on the nominal model, two times slower than RCI at 7.17 s. Furthermore, the settling times for cFVI and rFVI degrade to 15.20 s and 18.58 s at 25% modeling error; by contrast, RCI's settling time actually improves to 6.44 s at 25% modeling error. On the two higher-dimensional, multi-loop jet aircraft and DDMR models, the FVIs also tend to exhibit large overshoot when compared to RCI (cf. right two plots of Figure 6 and Table 4).

Finally, we note that as corroborated by Section 6, RCI recovers the closed-loop performance of the optimal policy for all systems and outperforms the nominal classical LQ

Table 4: Closed-loop performance generalization to modeling error $\nu = 0\%, 10\%, 25\%$

System	Alg	$t_{s,y,1\%}$ (s)			$t_{r,y,90\%}$ (s)			$M_{p,y}$ (%)		
	ν	0%	10%	25%	0%	10%	25%	0%	10%	25%
Pendulum $y = \theta$	RCI	1.16	1.16	1.18	0.67	0.71	0.74	0.00	0.01	0.07
	cFVI	1.80	1.80	1.95	1.10	1.19	1.36	0.00	0.02	0.19
	rFVI	1.52	1.54	1.53	0.85	0.88	0.90	0.00	0.03	0.03
	Opt LQ	1.16	1.22	1.28	0.67	0.70	0.74	0.00	0.02	0.01
	Nom LQ	1.16	0.99	2.12	0.67	0.65	0.71	0.00	0.65	3.09
Jet Aircraft $y_1 = V$	RCI	14.42	14.62	14.41	9.61	9.41	9.31	0.09	0.10	0.11
	cFVI	18.58	18.33	17.97	9.92	9.98	10.07	0.00	0.00	0.00
	rFVI	19.20	18.96	18.83	10.28	10.35	10.59	0.00	0.00	0.00
	Opt LQ	14.42	14.62	14.41	9.61	9.41	9.31	0.09	0.10	0.11
	Nom LQ	14.42	14.36	14.27	9.61	9.47	9.51	0.09	0.10	0.12
$y_2 = \gamma$	RCI	7.17	6.99	6.44	4.43	4.49	4.44	0.25	0.36	0.69
	cFVI	14.42	14.67	15.20	3.91	3.99	4.16	14.68	16.53	19.96
	rFVI	17.88	18.18	18.58	4.00	4.10	4.23	15.63	17.41	20.60
	Opt LQ	7.17	6.99	6.44	4.43	4.49	4.44	0.25	0.36	0.69
	Nom LQ	7.17	6.67	8.81	4.43	4.39	4.42	0.25	0.37	1.11
DDMR $y_1 = V$	RCI	5.33	5.33	5.33	3.83	3.83	3.83	0.43	0.43	0.43
	cFVI	5.45	5.46	5.52	3.94	3.96	3.77	0.55	0.55	0.55
	rFVI	5.77	5.78	5.88	3.75	3.67	3.77	0.10	0.10	0.10
	Opt LQ	5.33	5.33	5.33	3.83	3.83	3.83	0.43	0.43	0.43
	Nom LQ	5.33	5.33	5.33	3.83	3.83	3.83	0.43	0.43	0.43
$y_2 = \omega$	RCI	6.68	6.87	7.44	1.24	1.19	1.19	16.93	18.76	21.06
	cFVI	12.81	12.12	11.69	1.28	1.22	1.16	26.02	30.74	38.93
	rFVI	17.51	17.51	17.67	1.53	1.51	1.38	12.06	13.81	16.72
	Opt LQ	6.68	7.26	7.43	1.24	1.19	1.18	16.93	18.65	21.11
	Nom LQ	6.68	6.66	6.15	1.24	1.18	1.13	16.93	20.45	26.64

design. As in Section 6, we can clearly see from Figure 6 that the designer will face degraded closed-loop performance if they opt for the nominal classical design, which is corroborated numerically in Table 4. RCI successfully learns the nonlinearity and model uncertainty present in the nonlinear environment (f, g) (1), allowing it to recover optimal performance.

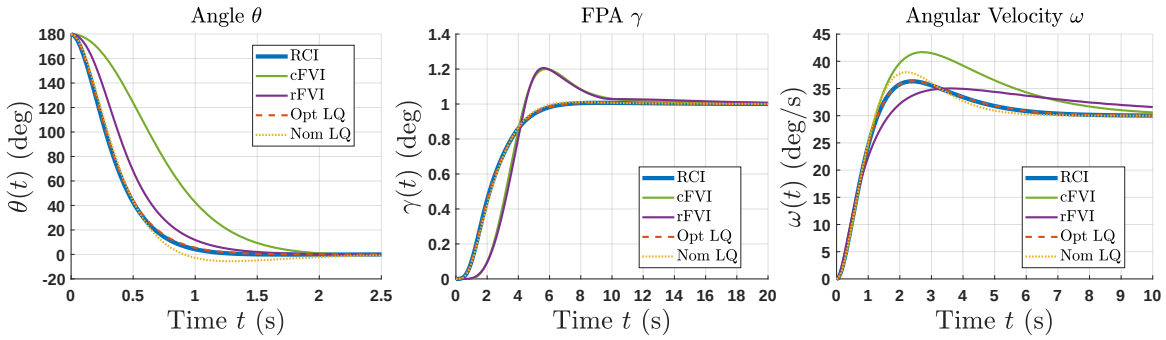

Figure 6: Closed-loop responses at 25% modeling error. Left: pendulum swing-up from natural hanging position (i.e., $\theta = 180°$). Middle: jet aircraft 1 deg step FPA command. Right: DDMR 30 deg/s step angular velocity command. Overall, the RCI responses tend to be faster than the FVIs with less overshoot. Full plots can be found in Supplemental Figures S9–S11.

Q6: RCI reduces algorithm time/data complexity by multiple orders of magnitude. Table 5 lists key algorithm complexity parameters for RCI and FVI. To illustrate, we examine the ratio of RCI/FVI for these parameters on the DDMR model. For number of trajectory episodes required: RCI/FVI = 1/5,000,000, data sample complexity: 1/6,000,000, training epochs: 1/400, and training time: 1/3,000.

Table 5: Algorithm time/data complexity

Parameter	Pendulum		Jet Aircraft		DDMR	
	RCI	cFVI/rFVI	RCI	cFVI/rFVI	RCI	cFVI/rFVI
# Episodes	1	1.05e+07 /3.84e+06	1	5.12e+06	1	5.12e+06
# Trajectory Data Samples	15	3.45e+08 /1.73e+08	45	2.30e+08	35	2.30e+08
# Epochs as in FVIs (Lutter et al., 2021, 2023b)	5	2,000 /3,000	5	2,000	5	2,000
Avg Training Time (s)	0.17	6.88e+03 /8.98e+03	4.25	8.18e+03 /7.98e+03	2.58	6.30e+03 /6.04e+03

8. Conclusion and Discussion

We introduce a new, model-based RCI CT-RL design in the same context as the current ADP and SOTA deep RL CT-RL methods, which address affine nonlinear dynamic control problems. Our RCI approach leverages the known affine nonlinear dynamics, an innovative design of input excitation to enable learning exploration, and Kleinman control structures. We use a different approach to exploration during learning, as it is realized that explo-

ration is essential yet can be problematic at times. This is because effective exploration requires injecting the right level of exploration noise at the right time and within the physical constraints of the application at hand. **On the Utility of RCI,** our new reference command input design approach has given us the opportunity to leverage the available, possibly inaccurate, dynamics knowledge while exploring in the uncharted territory. The corresponding design takes shape of Kleinman's structure while using state-action trajectory data to learn. The former, Kleinman's solution framework, inherently provides us with well-behaved system performance such as quick settling time, small overshoot, high-frequency noise rejection, etc. The latter, state-action trajectory based learning, enables us to capture unmodeled dynamics or model uncertainty while retaining all these nice properties. As such, our innovative RCI-based CT-RL makes the best use of available affine nonlinear model dynamics and learning from data. This is why RCI achieves consistent learning, well-behaved system time responses, and practically-observed and theoretically-guaranteed convergence, optimality, and stability. In the meantime, our RCI-based approach also achieves a nice trade-off between exploration and exploitation, which again is facilitated by effective exploration with reference command input and by using physically-informed knowledge of the environment. The exploitation or optimization is also efficiently achieved by our proper use of a quadratic control performance structure, which is a standard practice in classical optimal control involving continuous states and actions.

Among the three CT-RL approaches in the same class of affine nonlinearity, **RCI** presents theoretical guarantees, and its learning performance at least matches, and often outperforms, the current SOTA deep RL FVIs in terms of average return, critic network approximation accuracy, closed-loop time-domain performance, algorithm data/time efficiency, and generalization to modeling error. Yet, RCI's efficiency requires knowledge of the environment, and RCI only considers Q-R cost structures (2). Meanwhile, **ADP** presents strong analytical results, and some ADP methods do not require knowledge of the environment (f, g). But ADP methods have not been proven for meaningful applications, as only evaluations of toy systems with known optimal solutions are available. Furthermore, ADPs generally restrict to Q-R cost as well. Finally, deep RL **FVIs** are learning-driven methods with significant empirical promise and generalizability, as independently verified by the new SOTA evaluations we conduct on these algorithms in Section 7. These methods also consider flexible cost structures including dense/sparse costs. However, FVIs require the most dynamic knowledge of the three classes, and theoretical results are yet to be developed. Finally, we would like to mention that while a majority of CT-RL results are developed for addressing affine nonlinear dynamics, great efforts are needed to develop designs for fully nonlinear dynamic environments, even extensive evaluations of and further developments based on currently available methods can be helpful to gain fundamental insights on addressing CT-RL optimal control of general nonlinear dynamics.

Acknowledgments

This work was supported in part by the NSF under Grants 1563921, 1808752, and 2211740. Brent A. Wallace was also supported by the NSF under Graduate Research Fellowship Grant 026257-001.

Appendix A. Background

A.1 Theoretical Assumptions Required by SOTA CT-RL Addressing Affine Nonlinear Dynamics

As shown below, RCI is among the least restrictive in CT-RL in its theoretical assumptions. As a note, all methods require that (f, g) (1) be Lipschitz near origin to assure well-posedness of solutions to the system differential equations.

RCI (present work):

- Stabilizability of the linearization (A, B) of the nonlinear system (f, g) (1), and $(Q^{1/2}, A)$ detectable (for well-posedness of regulation problem, cf. Appendix A.1)
- Full column rank of integral reinforcement matrix Θ (25)
- Initial stabilizing policy

FVIs (Lutter et al., 2021, 2023b):

- f and g are smooth in their partial derivatives in the state x and model uncertainty parameters θ, and these partials are all known *a priori*
- Undiscounted problem $\gamma = 1$ can be approximated by discounted problem $0 < \gamma < 1$
- Discrete-time running cost $r(x, u)$ can be approximated by continuous-time counterpart: $r(x, u) = \Delta t\, r_c(x, u)$ with sample time Δt
- Strict convexity of action penalty g_c
- Availability of convex conjugate function to action penalty g_c
- Higher-order terms in Taylor series expansion of optimal value V^* are negligible
- Existence of an *a priori* state grid $x \in \mathcal{D}$ to contain trajectories to for fitting procedure
- Trajectories leaving the grid $x \in \mathcal{D}$ can be instantaneously re-initialized to the previous position inside the grid

Integral Reinforcement Learning (IRL) (Vrabie and Lewis, 2009):

- There exists a sequence of sampling instants $t_0 < t_1 < \cdots < t_l$ such that the IRL regression matrix has full rank
- Chosen bases approximate optimal value and its gradient uniformly on compact sets
- Basis functions for critic network are linearly-independent
- Initial stabilizing policy

Synchronous Policy Iteration (SPI) (Vamvoudakis and Lewis, 2010):

- Existence and uniqueness of least-squares solution to approximate HJB equation
- PE assumption on various learning signals
- Chosen bases approximate optimal value and its gradient uniformly on compact sets
- Chosen bases approximate optimal policy uniformly on compact sets
- Basis functions for critic network are linearly-independent
- Basis functions for actor network are linearly-independent
- Initial stabilizing policy

Robust ADP (RADP) (Jiang and Jiang, 2014):

- Optimal value can be bounded from above and below by *a priori* known class $\mathcal{K}_\infty$ functions
- Existence of *a priori* known compact set Ω_0 for which the closed-loop system under the initial policy is invariant with respect to the probing noise d
- PE assumption on various learning signals

- Chosen bases approximate optimal value and its gradient uniformly on compact sets
- Chosen bases approximate optimal policy uniformly on compact sets
- Basis functions for critic network are linearly-independent
- Basis functions for actor network are linearly-independent
- Initial stabilizing policy

Continuous-Time Value Iteration (CT-VI) (Bian and Jiang, 2022):

- Existence and uniqueness of solutions to an uncountable family of finite-horizon HJB equations
- Properness of each solution to the finite-horizon HJB equation
- Convergence of family of solutions of finite-horizon HJB equation to the infinite-horizon HJB solution
- Invariance of closed-loop state trajectory to *a priori* compact set
- Initial *globally asymptotically stabilizing* policy
- PE assumption on various learning signals
- Chosen bases approximate optimal value and its gradient uniformly on compact sets
- Chosen bases approximate optimal policy uniformly on compact sets
- Chosen bases approximate optimal Hamiltonian uniformly on compact sets
- Basis functions for critic network are linearly-independent
- Basis functions for actor network are linearly-independent
- Basis functions for Hamiltonian network are linearly-independent

A.2 Environments Studied by SOTA CT-RL Works

We provide an overview of the environments studied in the evaluations of the leading CT-RL works in Table 6 below. As can be seen, the proposed environments are SOTA.

Table 6: Environments in SOTA CT-RL evaluations

Alg	System	Order	# inputs	Source of model parameters
RCI	Pendulum	$\longrightarrow$	$\longrightarrow$	Identical to FVIs as benchmark
	Jet Aircraft (new in CT-RL)	4	2	Full-scale NASA wind tunnel data (Soderman and Aiken, 1971)
	DDMR (new in CT-RL)	4	2	System ID on actual hardware (Mondal et al., 2020, 2019)
FVIs	Pendulum	2	1	Quanser STEM curriculum resources (Quanser, 2018)
	Cart Pendulum	4	1	
	Furatura Pendulum	4	1	
IRL	Simple Academic	2	1	Non-physical, optimal solutions known *a priori* (Remark 1)
	Simple Academic	2	1	
SPI	Simple Linear	3	1	Non-physical LQR example
	Simple Academic	2	1	See IRL above
RADP	Simple Engine	2	1	Non-physical for illustration
	Simple Power Bus	2	1	Non-physical for illustration
CT-VI	Simple Academic	2	1	See IRL above
	Simple Robot Arm	4	2	Non-physical for illustration

Remark 1: Many of the leading ADP works (Vrabie and Lewis, 2009; Vamvoudakis and Lewis, 2010; Jiang and Jiang, 2014; Bian and Jiang, 2022) study simple academic second-order examples which are constructed such that the optimal value and policy are polynomial functions known *a priori* in closed form, and for which the bases chosen can achieve exact approximation. The CT-VI work (Bian and Jiang, 2022) does study a robotic arm example, but the model is simplified and the parameter values are chosen academic for illustration. Similar trends hold for the examples studied in RADP (Jiang and Jiang, 2014).

Appendix B. Proofs of Theoretical Results

B.1 Proof of Theorem 4

Suppose that the sample count $l \in \mathbb{N}$ and sample times $\{t_k\}_{k=0}^{l}$ are such that Θ (25) has full rank $\underline{n}$. Note that given the choice of quadratic critic bases (10) and policy structure (12), the initial stabilizing policy μ_0 may be expressed in the form $\mu_0(x) = -K_0 x$ for some $K_0 \in \mathbb{R}^{m\times n}$. Thus, the hypotheses imply that $A - BK_0$ is Hurwitz. We now proceed by induction on i.

Suppose it has been proved for iteration $i \geq 0$ that $A - BK_i$ is Hurwitz and that $\mu_i(x) = -K_i x$ as generated from Kleinman's algorithm. We first claim the hypotheses imply that the least-squares matrix $\Theta_i \in \mathbb{R}^{l\times \underline{n}}$ (22) also has full column rank $\underline{n}$. For suppose $v(P) \in \mathbb{R}^{\underline{n}}$ is such that $\Theta_i v(P) = 0$. Examining (17), and proceeding through the derivation in Section 3.1, after applying the identity (8) we note for *any* symmetric matrix that $\Theta_i v(P) = \Theta v(N)$, where $N \in \mathbb{R}^{n\times n}$, $N = N^T$ is given by

$$N = (A - BK_i)^T P + P(A - BK_i). \tag{29}$$

However, (29) is itself an ALE. Furthermore, since $N = N^T$ and since $A-BK_i$ is Hurwitz by hypothesis, (29) has the unique solution $P = \int_0^\infty e^{(A-BK_i)^T t}(-N)e^{(A-BK_i)t}\,dt$ (Rodriguez, 2004). Meanwhile, the full rank of Θ and that $\Theta v(N) = 0$ imply $v(N) = 0$, or $N = 0$. Since $N = 0$, we have by the above that $v(P) = 0$. We have shown that Θ_i has trivial right null space, hence full column rank $\underline{n}$.

Having established that Θ_i has full rank, we now claim that $P_i \in \mathbb{R}^{n\times n}$, $P_i = P_i^T > 0$ (uniquely) solves the ALE (3) if and only if $c_i = v(P_i)$ satisfies the least-squares regression (22) at equality. The forward direction was already proved in the derivation (14)–(21) of Section 3.1. The key here is that since $\mu_i(x) = -K_i x$ has been established, the first-order approximation of the GHJB equation (19) holds at equality; indeed, (19) collapses to Kleinman's ALE (3) in this case.

Conversely, suppose $v(P) \in \mathbb{R}^{\underline{n}}$ is such that the least-squares regression (22) is minimized. Since Θ_i has full column rank, $v(P) \in \mathbb{R}^{\underline{n}}$ is unique. Now, letting $P_i = P_i^T > 0$ be the (unique) solution of the ALE (3), the forward direction establishes that $v(P_i) \in \mathbb{R}^{\underline{n}}$ satisfies (22) at equality. Thus, $v(P) = v(P_i)$, whence $P = P_i$ (Proposition 3) and the result is proved.

Having established the preceding, the proof now follows by induction on the algorithm iteration i. ■

Appendix C. Additional Implementation Details

Hardware. These studies were performed in MATLAB R2022b, on an NVIDIA RTX 2060, Intel i7 (9th Gen) processor. All numerical integrations in this work are performed in MATLAB's adaptive `ode45` solver to ensure solution accuracy.
Software. All RCI code and datasets developed for this work is available in Supplemental and at (Wallace and Si, 2024c). All FVI results (Lutter et al., 2021, 2023b) were generated from the open-source repository developed by the authors (Lutter et al., 2023a).

C.1 Additional Training/Evaluation Procedure Details

This section provides additional implementation details for the training/evaluation procedures discussed in Section 5.5. We encourage that the reader first see Section 5.5 to get a general orientation of the procedures.
Network Weight Initialization. For the FVI algorithms' deep networks, we use the identical network initialization procedure as the original development works (Lutter et al., 2021, 2023b); namely, Xavier normal distribution (initialization gains can be found in Table 8 of Appendix C.2). For RCI's simpler quadratic network structure, we need only initialize the critic weights c_0 in (10). We initialize each element of the critic weight vector $c_0 \in \mathbb{R}^l$ in the following uniform distributions for the pendulum, jet aircraft, and DDMR, respectively:

$$c_0 \sim \mathcal{U}(-10, 10), \tag{30}$$

$$c_0 \sim \begin{cases} \mathcal{U}(-25, 25), & \text{for weights in translational loop } j = 1 \\ \mathcal{U}(-1, 1), & \text{for weights in rotational loop } j = 2 \end{cases}, \tag{31}$$

$$c_0 \sim \begin{cases} \mathcal{U}(-25, 25), & \text{for weights in translational loop } j = 1 \\ \mathcal{U}(-10, 10), & \text{for weights in rotational loop } j = 2 \end{cases}. \tag{32}$$

IC Generation – Training. System ICs for training are generated via the following uniform distributions $\mathcal{U}$ for the pendulum, jet aircraft, and DDMR, respectively:

$$x_0 \sim \mathcal{U}(\pm\pi \text{ rad}, \pm 8 \text{ rad/s}), \tag{33}$$

$$x_0 \sim \mathcal{U}(\pm 15 \text{ ft/s}, \pm 3 \text{ deg}, \pm 30 \text{ deg/s}, \pm 10 \text{ deg}), \tag{34}$$

$$x_0 \sim \mathcal{U}(\pm 3 \text{ m/s}, \pm 90 \text{ deg/s}, \pm 0 \text{ A}, \pm 0 \text{ A}). \tag{35}$$

As a note: For the pendulum system, we have chosen the identical uniform distribution $\mathcal{U}$ (33) for state initialization as is chosen in the original FVI benchmark studies (Lutter et al., 2021, 2023b). The only exception to the above IC generation procedure is the systematic grid sweeps conducted in the RCI initial condition and modeling error generalization studies of Section 6. Here, the initial conditions x_0 are swept over a grid of values $x_0 \in G_{x_0}$. The IC grids G_{x_0} for the pendulum, jet aircraft, and DDMR are given respectively as:

$$G_{x_0} = [-\tfrac{\pi}{3} : \tfrac{\pi}{6} : \tfrac{\pi}{3}] \text{ rad } \times [-\tfrac{\pi}{3} : \tfrac{\pi}{6} : \tfrac{\pi}{3}] \text{ rad/s}, \tag{36}$$

$$G_{x_0} = [90 : 2 : 110] \text{ ft/s } \times [-2 : 0.5 : 2] \text{ deg}, \tag{37}$$

$$G_{x_0} = [1.5 : 0.125 : 2.5] \text{ m/s } \times [-30 : 5 : 30] \text{ rad/s}, \tag{38}$$

where all other ICs for the higher-order jet aircraft and DDMR environments are set to zero. Note that for all systems in this work, the IC grid G_{x_0} is centered about the respective equilibrium point x_e (cf. Sections 3–5 of Supplemental for discussion of equilibria of each system).

IC Generation – Evaluation. For the learning curves plotted in Figure 2, at each algorithm iteration the return of the trained policies is evaluated over 100 episodes of the environment. For evaluation, system ICs for training are generated via the following uniform distributions $\mathcal{U}$ for the pendulum, jet aircraft, and DDMR, respectively:

$$x_0 \sim \mathcal{U}(\pm\pi \text{ rad}, \pm 0.01 \text{ rad/s}), \tag{39}$$

$$x_0 \sim \mathcal{U}(\pm 10 \text{ ft/s}, \pm 2 \text{ deg}, \pm 0.01 \text{ deg/s}, \pm 0.01 \text{ deg}), \tag{40}$$

$$x_0 \sim \mathcal{U}(\pm 0.5 \text{ m/s}, \pm 30 \text{ deg/s}, \pm 0 \text{ A}, \pm 0 \text{ A}). \tag{41}$$

As a note: For the pendulum system, we have chosen the identical uniform distribution $\mathcal{U}$ (39) for state initialization as is chosen in the original FVI benchmark studies (Lutter et al., 2021, 2023b).

For display purposes of generating the surface plots in Figures 3 and 4, we evaluate the final polices of a single trial for each method over the following evaluation grids $x \in G_x$ for the pendulum, jet aircraft, and DDMR, respectively:

$$G_x = [-60 : 0.5 : 60] \text{ deg } \times [-60 : 0.5 : 60] \text{ deg/s}, \tag{42}$$

$$G_x = [90 : 2 : 110] \text{ ft/s } \times [-2 : 0.1 : 2] \text{ deg}, \tag{43}$$

$$G_x = [1.5 : 0.125 : 2.5] \text{ m/s } \times [-30 : 5 : 30] \text{ rad/s}, \tag{44}$$

where all other ICs for the higher-order jet aircraft and DDMR environments are set to zero. Note that for all systems in this work, as is the case with the IC grids G_{x_0} (36)–(36) the evaluation grids G_x (42)–(42) are centered about the respective equilibrium point x_e (cf. Sections 3–5 of Supplemental for discussion of equilibria of each system). It is these grids which are used to generate the surface plots of Figures 3 and 4, and the corresponding tabular data in Table 3.

Modeling Error Generation. In the modeling error generalization studies of Sections 6 and 7, modeling error ν is swept over a grid of values $\nu \in G_\nu$. The modeling error grids G_ν for the pendulum, jet aircraft, and DDMR are given respectively as:

$$G_\nu = [1 : 0.01 : 1.25], \tag{45}$$

$$G_\nu = [1 : -0.025 : 0.75], \tag{46}$$

$$G_\nu = [1 : 0.025 : 1.25]. \tag{47}$$

The direction of the perturbation (i.e., $\nu > 1$ or $\nu < 1$) is chosen to maximize the difficulty of the respective learning problem (cf. Sections 3–5 of Supplemental for in-depth discussion).

In the RCI combined IC/modeling error generalization study of Section 6, RCI is run over the combined trial space $(x_0, \nu) \in G_{x_0} \times G_\nu$, where the respective IC grids G_{x_0} for each environment are given by (36)–(38) and the respective modeling error grids G_ν for each environment are given by (45)–(47). This corresponds to 1,620 learning trials for the pendulum (81 ICs x_0 × 20 modeling errors ν), 1,089 trials for the jet (99 x_0 × 11 ν), and 1,287 trials for the DDMR (117 x_0 × 11 ν).

C.2 Hyperparameter Selections

C.2.1 Shared Hyperparameters

State, Control Penalty Gains. For the pendulum, we use identical penalty selections to those in the original FVI studies (Lutter et al., 2021, 2023b); namely,

$$Q_1 = \texttt{diag}(1, 0.1), \qquad R_1 = 0.5. \tag{48}$$

For the jet aircraft, consider the decentralized design framework described in Section 4 of Supplemental. We choose the following cost structure

$$\begin{aligned} Q_1 &= \texttt{diag}(0.005, 0.05), & R_1 &= 5, \\ Q_2 &= \texttt{diag}(0.5, 1, 0, 0), & R_2 &= 1. \end{aligned} \tag{49}$$

These state/control penalties were chosen to yield optimal LQ controllers achieving nominal closed-loop step response specifications comparable to existing benchmarks (Stengel, 2022): A 90% rise time in speed $t_{r,V,90\%} = 9.297$ s and FPA $t_{r,\gamma,90\%} = 4.52$ s, a 1% settling time in speed $t_{s,V,1\%} = 14.47$ s and FPA $t_{s,\gamma,1\%} = 7.20$ s, percent overshoot in speed $M_{p,V} = 0.09\%$ and FPA $M_{p,\gamma} = 0.25\%$.

For the DDMR, consider the decentralized design framework described in Section 5 of Supplemental. We choose the following cost structure

$$\begin{aligned} Q_1 &= 10I_2, & R_1 &= 0.75, \\ Q_2 &= \texttt{diag}(25, 7.5), & R_2 &= 1. \end{aligned} \tag{50}$$

These state/control penalties were chosen to yield optimal LQ controllers achieving nominal closed-loop step response specifications comparable to existing benchmarks (Mondal et al., 2020, 2019): A 90% rise time in speed $t_{r,V,90\%} = 3.778$ s and angular velocity $t_{r,\omega,90\%} = 1.27$ s, a 1% settling time in speed $t_{s,V,1\%} = 5.556$ s and angular velocity $t_{s,\omega,1\%} = 6.73$ s, percent overshoot in speed $M_{p,V} = 0\%$ and angular velocity $M_{p,\omega} = 16.9\%$.

C.2.2 RCI

Initial Stabilizing Policy. For the pendulum, we use the initial stabilizing policy

$$K_{0,1} = \begin{bmatrix} 13.5108 & 5.8316 \end{bmatrix}, \tag{51}$$

which we obtained from cost structure selections $Q_1 = \texttt{diag}(0.5, 0.25)$, and $R_1 = 0.01$. For the jet aircraft in loop j ($j = 1, 2$), we use the initial stabilizing policies

$$K_{0,1} = \begin{bmatrix} 0.0316 & 0.1168 \end{bmatrix}, \tag{52}$$

$$K_{0,2} = \begin{bmatrix} -1.7321 & -3.4191 & -0.3427 & -0.9709 \end{bmatrix}, \tag{53}$$

which we obtained from a decentralized design with cost structure selections $Q_1 = 0.01I_2$, $R_1 = 10$, $Q_2 = \texttt{diag}(1.5, 2.5, 0, 0)$, and $R_2 = 0.5$. For the DDMR in loop j ($j = 1, 2$), we use the initial stabilizing policies

$$K_{0,1} = \begin{bmatrix} 2.2361 & 3.4966 \end{bmatrix}, \tag{54}$$

$$K_{0,2} = \begin{bmatrix} 8.6603 & 12.4403 \end{bmatrix}, \tag{55}$$

which we obtained from a decentralized design with cost structure selections $Q_1 = 5I_2$, $R_1 = 1$, $Q_2 = \texttt{diag}(7.5, 2.5)$, and $R_2 = 0.1$. See Table 7 for remaining RCI hyperparameters.

Table 7: RCI hyperparameter selections

Hyperparameter	Pendulum	Jet Aircraft		DDMR	
	Loop $j=1$	Loop $j=1$	Loop $j=2$	Loop $j=1$	Loop $j=2$
Sample Period $T_{s,j}$ (s)	1	2	0.5	4	1
Number of Samples l_j	15	15	30	20	15
Final Iteration i_j^*	5	5	5	5	5
Ref Cmd r_j	$10\sin(\frac{2\pi}{10}t)$	$5\sin(\frac{2\pi}{50}t)$	$0.1\sin(\frac{2\pi}{2.5}t)$	$2\sin(\frac{2\pi}{10}t)$	$5\sin(\frac{2\pi}{50}t)$
(deg | m/s, deg | m/s, deg/s)	$+5\sin(\frac{2\pi}{5}t)$	$+10\sin(\frac{2\pi}{25}t)$	$+0.1\sin(\frac{2\pi}{1.5}t)$	$+\sin(\frac{2\pi}{5}t)$	$+5\sin(\frac{2\pi}{5}t)$ $+5\sin(\frac{2\pi}{2.5}t)$
Initial Policy $K_{0,j}$	(51)	(52)	(53)	(54)	(55)

Table 8: cFVI, rFVI hyperparameter selections

Hyperparameter	Pendulum		Jet Aircraft		DDMR	
	cFVI	rFVI	cFVI	rFVI	cFVI	rFVI
Time Step (s)	0.008	0.008	0.008	0.008	0.008	0.008
Time Horizon (s)	5	5	20	20	5	5
Discounting γ	0.99	0.99	0.99	0.99	0.99	0.99
Network Dimension	$[3\times 96]$	$[3\times 96]$	$[3\times 96]$	$[3\times 96]$	$[3\times 96]$	$[3\times 96]$
# Ensemble	4	4	4	4	4	4
Activation	Tanh	Tanh	Tanh	Tanh	Tanh	Tanh
Learning Rate	1e-5	1e-5	3e-5	3e-5	3e-5	3e-5
Weight Decay	1e-6	1e-6	1e-6	1e-6	1e-6	1e-6
Hidden Layer Gain	1.41	1.41	1.41	1.41	1.41	1.41
Output Layer Gain	1.00	1.00	1.00	1.00	1.00	1.00
Output Layer Bias	-0.1	-0.1	-0.1	-0.1	-0.1	-0.1
Diagonal Softplus Gain β_L	1.0	1.0	7.5	7.5	1.0	1.0
Batch Size	256	128	256	256	256	256
# Batches	200	200	200	200	200	200
Eligibility Trace	0.85	0.85	0.85	0.85	0.85	0.85
n-step Trace Weight	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4
# Iterations	100	150	100	100	100	100
# Epochs/Iteration	20	20	20	20	20	20
State Adversary $\|\xi_x\|_{\max}$	0.0	0.025	0.0	0.025	0.0	0.025
Action Adversary $\|\xi_u\|_{\max}$	0.0	0.1	0.0	0.1	0.0	0.1
Model Adversary $\|\xi_\theta\|_{\max}$	0.0	0.15	0.0	0.1	0.0	0.009
Obs Adversary $\|\xi_o\|_{\max}$	0.0	0.025	0.0	0.025	0.0	0.025

C.2.3 cFVI, rFVI

Hyperparameter selections for cFVI and rFVI can be found in Table 8. These parameter selections are overall quite standard and have indeed demonstrated great learning performance successes on second-order, unstable systems in previous studies (Lutter et al., 2021, 2023b). As with our selections of the pendulum model structure and parameters (cf. Section 3 of Supplemental), for our pendulum studies we have selected hyperparameters identical

to those of the original cFVI/rFVI evaluations (Lutter et al., 2021, 2023b), with two exceptions. In (Lutter et al., 2021, 2023b), the authors use a logcos control penalty function scaled so that its curvature at the origin $u = 0$ is $2R$; i.e., so that its curvature agrees with that of a quadratic penalty $u^T Ru$. In order to make comparisons consistent across the methods studied, we have applied the standard quadratic control penalty $u^T Ru$ for all methods. Likewise, the authors in (Lutter et al., 2021, 2023b) wrap the penalty function of the pendulum angle state to be periodic in $[0, 2\pi)$, a practice which we have dropped for consistency of comparison. Finally, due to these changes we observed that more iterations were necessary for rFVI to converge in training the pendulum system (cf. Figure 2), so we increased its iteration count from 100 previously (Lutter et al., 2021, 2023b) to 150 here.

References

C. Atkeson and J. Morimoto. Nonparametric representation of policies and value functions: A trajectory-based approach. In *Adv. Neur. Info. Proc. Syst. (NeurIPS)*, volume 15, pages 1–8, November 2002.

A. G. Barto, R. S. Sutton, and C. W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Trans. Syst., Man, Cybern.*, 13(5):835–846, October 1983. doi: 10.1109/TSMC.1983.6313077.

R. W. Beard and T. T. McLain. Successive galerkin approximation algortihms for nonlinear optimal and robust control. *Int. J. Contr.*, 71:717–743, 1998.

D. P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, Belmont, MA, USA, 3 edition, 2005.

D. P. Bertsekas. Value and policy iterations in optimal control and adaptive dynamic programming. *IEEE TNNLS*, 28(3):500–509, March 2017.

T. Bian and Z.-P. Jiang. Reinforcement learning and adaptive optimal control for continuous-time nonlinear systems: A value iteration approach. *IEEE TNNLS*, 33(7): 2781–2790, July 2022. doi: 10.1109/TNNLS.2020.3045087.

S. J. Bradtke, B. E. Ydstie, and A. G. Barto. Adaptive linear quadratic control using policy iteration. In *1994 IEEE ACC*, pages 3475–3479, June 1994.

M. R. Caputo. *Foundations of dynamic economic analysis: optimal control theory and applications.* Cambridge University Press, Cambridge, UK, 2005.

J. J. Craig. *Introduction to Robotics: Mechanics and Control.* Pearson Education, Upper Saddle River, NJ, USA, 3 edition, 2005.

R. Dhaouadi and A. Abu Hatab. Dynamic modelling of differential-drive mobile robots using lagrange and newton-euler methodologies: A unified framework. *Adv. Robo. Auto.*, 2(2):1–7, January 2013.

J. Dickeson, A. A. Rodriguez, S. Sridharan, J. Benevides, and D. Soloway. Decentralized control of an airbreathing scramjet-powered hypersonic vehicle. In *AIAA Guid., Nav.,*

Contr. Conf. Exhib., pages 1–25, August 2009a. doi: 10.2514/6.2009-6281. AIAA 2009-6281.

J. Dickeson, A. A. Rodriguez, S. Sridharan, D. Soloway, A. Korad, J. Khatri, J. Benavides, A. Kelkar, and J. Vogel. Control-relevant modeling, analysis, and design for scramjet-powered hypersonic vehicles. In *AIAA/DLR/DGLR Int. Space Planes Hyper. Syst. Tech. Conf.*, pages 1–45, October 2009b. doi: 10.2514/6.2009-7287. AIAA 2009-7287.

K. Doya. Reinforcement learning in continuous time and space. *Neural Comp.*, 12(1): 219–245, January 2000.

R. Enns and J. Si. Apache helicopter stabilization using neural dynamic programming. *AIAA J. Guid., Contr., & Dyn.*, 25(1):19–25, January 2002.

R. Enns and J. Si. Helicopter flight-control reconfiguration for main rotor actuator failures. *AIAA J. Guid., Contr., & Dyn.*, 26(4):572–584, July 2003a.

R. Enns and J. Si. Helicopter trimming and tracking control using direct neural dynamic programming. *IEEE TNN*, 14(4):929–939, July 2003b.

W. Haoran, T. Zariphopoulou, and X. Y. Zhou. Reinforcement learning in continuous time and space: A stochastic control approach. *J. Mach. Learn. Res. (JMLR)*, 21(1): 9363–9396, January 2020.

Y. Jiang and Z.-P. Jiang. Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics. *Automatica*, 48(10):2699–2704, October 2012.

Y. Jiang and Z.-P. Jiang. Robust adaptive dynamic programming and feedback stabilization of nonlinear systems. *IEEE TNNLS*, 25(5):882–893, January 2014. doi: 10.1109/TNNLS.2013.2294968.

J. Kim, J. Shin, and I. Yang. Hamilton-Jacobi deep Q-learning for deterministic continuous-time systems with Lipschitz continuous controls. *J. Mach. Learn. Res. (JMLR)*, 22(1): 9363–9396, September 2021.

B. Kiumarsi, K. G. Vamvoudakis, H. Modares, and F. L. Lewis. Optimal and autonomous control using reinforcement learning: A survey. *IEEE TNNLS*, 29(6):2042–2062, June 2018.

D. Kleinman. On an iterative technique for Riccati equation computations. *IEEE TAC*, 13 (1):114–115, February 1968. doi: 10.1109/TAC.1968.1098829.

B. Landry, H. Dai, and M. Pavone. SEAGuL: sample efficient adversarially guided learning of value functions. *Learning for Dyn. and Contr.*, 144:1105–1117, June 2021.

F. L. Lewis and D. Liu. *Reinforcement Learning and Approximate Dynamic Programming for Feedback Control*. John Wiley & Sons, New York, NY, USA, 2012.

F. L. Lewis, D. Vrabie, and V. L. Syrmos. *Optimal Control*. John Wiley & Sons, 3 edition, 2012a.

F. L. Lewis, D. Vrabie, and K. G. Vamvoudakis. Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers. *IEEE Contr. Syst. Mag.*, 32:76–105, 2012b.

D. Liu, S. Xue, B. Zhao, B. Luo, and Q. Wei. Adaptive dynamic programming for control: A survey and recent advances. *IEEE Trans. Syst., Man, Cybern.*, 51(1):142–160, January 2021.

M. Lutter, S. Mannor, J. Peters, D. Fox, and A. Garg. Value iteration in continuous actions, states and time. In *Proc. 38th Int. Conf. Mach. Learn. (ICML)*, volume 139, pages 7224–7234, July 2021. URL `https://proceedings.mlr.press/v139/lutter21a.html`.

M. Lutter et al. Continuous & robust fitted value iteration. `https://github.com/milutter/value_iteration`, 2023a. Accessed: 2023-01-12.

M. Lutter et al. Continuous-time fitted value iteration for robust policies. *IEEE Trans. Patt. Anal. Mach. Intel.*, 45(5):5534–5548, May 2023b. doi: 10.1109/TPAMI.2022.3215769.

K. Mondal, A. A. Rodriguez, S. S. Manne, N. Das, and B. A. Wallace. Comparison of kinematic and dynamic model based linear model predictive control of non-holonomic robot for trajectory tracking: Critical trade-offs addressed. In *Proc. IASTED Int. Conf. Mech. Contr.*, pages 9–17, December 2019. `https://doi.org/10.2316/P.2019.860-017`.

K. Mondal, B. A. Wallace, and A. A. Rodriguez. Stability versus maneuverability of non-holonomic differential drive wheeled robot: Focus on aggressive position control strategies. In *2020 IEEE CCTA*, pages 388–395, August 2020. `https://doi.org/10.1109/CCTA41146.2020.9206155`.

K. Ogata. *Modern Control Engineering*. Prentice Hall, Upper Saddle River, NJ, USA, 3 edition, 1997.

C. Possieri and M. Sassano. Q-learning for continuous-time linear systems: A data-driven implementation of the Kleinman algorithm. *IEEE Trans. Syst., Man, Cybern. – A: Syst.*, 52(10):6487–6497, October 2022.

Quanser. Quanser courseware and resources. `https://www.quanser.com/solution/control-systems/`, 2018.

A. A. Rodriguez. *Analysis and Design of Multivariable Feedback Control Systems*. CONTROL3D, Tempe, AZ, USA, 2004.

I. L. Sandoval, P. Petsagkourakis, and E Antonio del Rio-Chanona. Neural odes as feedback policies for nonlinear optimal control. *IFAC-PapersOnLine*, 56(2):4816–4821, January 2023.

N. Shimkin and A. Feuer. Persistency of excitation in continuous-time systems. *Syst. Contr. Lett.*, 9(3):225–233, September 1987.

J. Si, A. G. Barto, W. B. Powell, and D. C. Wunsch. *Handbook of Learning and Approximate Dynamic Programming*. Wiley, Piscataway, NJ, USA, 2004.

P. T. Soderman and T. N. Aiken. Full-scale wind-tunnel tests of a small unpowered jet aircraft with a T-tail. *NASA-TN-D-6573*, November 1971.

R. F. Stengel. *Flight Dynamics*. Princeton University Press, Princeton, NJ, USA, 2 edition, 2022.

T. H. Summers, K. Kunz, N. Kariotoglou, M. Kamdarpour, S. Summers, and J. Lygeros. Approximate dynamic programming via sum of squares programming. In *Proc. 2013 IEEE ECC*, pages 191–197, July 2013.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, USA, 2 edition, 2018.

Y. Tassa and T. Erez. Least squares solutions of the HJB equation with neural network value-function approximators. *IEEE TNN*, 18(4):1031–1041, July 2007.

K. G. Vamvoudakis and F. L. Lewis. Online actor–critic algorithm to solve the continuous-time infinite horizon optimal control problem. *Automatica*, 46(5):878–888, 2010. doi: 10.1016/j.automatica.2010.02.018.

D. Vrabie and F. L. Lewis. Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems. *Neur. Net.*, 22(3):237–246, 2009. doi: 10.1016/j.neunet.2009.03.008.

B. A. Wallace and J. Si. Reinforcement learning control of hypersonic vehicles and performance evaluations. *AIAA J. Guid., Contr., & Dyn.*, 47(12):2587–2600, December 2024a. `https://doi.org/10.2514/1.G008225`.

B. A. Wallace and J. Si. Continuous-time reinforcement learning control: A review of theoretical results, insights on performance, and needs for new designs. *IEEE TNNLS*, 35(8):10199–10219, August 2024b. `https://doi.org/10.1109/TNNLS.2023.3245980`.

B. A. Wallace and J. Si. RCI (JMLR 2024). `https://github.com/bawalla2/JMLR-2024`, 2024c.

P. Wang, Z. Man, Z. Cao, J. Zheng, and Y. Zhao. Dynamics modelling and linear control of quadcopter. In *IEEE Int. Conf. Adv. Mech. Syst. (ICAMechS)*, pages 498–503, November 2016.

L. Yang, H. Dai, A. Amice, and R. Tedrake. Approximate optimal controller synthesis for cart-poles and quadrotors via sums-of-squares. *IEEE Robo. & Auto. Lett.*, 8(11): 7376–7383, November 2023.

Q. Yang, W. Cao, W. Meng, and J. Si. Reinforcement-learning-based tracking control of waste water treatment process under realistic system conditions and control performance requirements. *IEEE Trans. Syst., Man, Cybern. – A: Syst.*, 52(8):5284–5294, August 2022.

C. Yildiz, M. Heinonen, and H. Lähdesmäki. Continuous-time model-based reinforcement learning. In *Proc. 38th Int. Conf. Mach. Learn. (ICML)*, pages 12009–12018, July 2021.