IEEE Transactions on Neural Networks and Learning Systems, Available online 22 November 2023
DOI:10.1109/TNNLS.2023.3333551

Drone's Objective Inference using Policy Error Inverse Reinforcement Learning

Adolfo Perrusquía, *Member, IEEE*, Weisi Guo, *Senior Member, IEEE*

***Abstract*—Drones are set to penetrate society across transport and smart living sectors. Whilst many are amateur drones that pose no malicious intentions, some may carry deadly capability. It is crucial to infer drone's objective to prevent risk and guarantee safety. In this paper, a policy error inverse reinforcement learning (PEIRL) algorithm is proposed to uncover the hidden objective of drones from on-line data trajectories obtained from cooperative sensors. A set of error-based polynomial features are used to approximate both the value and policy functions. This set of features are consistent with current on-board storage memories in flight controllers. The real objective function is inferred using an objective constraint and an integral inverse reinforcement learning batch least-squares rule. Convergence of the proposed method is assessed using Lyapunov recursions. Simulations studies using a quadcopter model are provided to demonstrate the benefits of the proposed approach.**

***Index Terms*—Drones, objective function, weight matrices, inverse reinforcement learning, online trajectories, policy error**

I. Introduction

Proliferation of cheaper drone technology has magnified the threat space for drone malicious attacks [1]–[3], e.g., spoofing/sniffing, and reconnaissance activities; to mention some of them. Reliable detection of drones and identifying its intention is critical to ensuring safeguarding society and national infrastructures against the most severe threats [4].

Current drone detection technologies focus on imagery data to distinguish drones from other objects and clutter in the airspace [5]–[7]. This is achieved by exploiting different sensing technologies such as: radar, lidar, point-cloud, and high-definition vision systems which are able to capture high-dimensional features associated to the drone structure [8], [9]. However, the snapshot data provided by these sensors do not capture the drone's hidden intent. Some authors were interested in recognizing human actions [10] from drone videos with interesting results [11]. Nevertheless, videos cannot inform anything about the drone's behaviour which can be risky for the recorded humans.

Other approaches are based on predictive models using time-series of the drone's positions, velocities, orientations, and control inputs trajectories [12]–[14]. These models use data history to predict the trajectory [15] that the drone will follow in future time steps. The most simple predictive algorithms are linear regression and support vector regression which are able to find a model from observational data. However, these methods are sensitive to the amount of data and its heterogeneity. Recurrent neural networks [16]–[18] are used as an alternative method for trajectory prediction using high-dimensional features of the time-series trajectories. Nevertheless, the predicted trajectories can diverge fast for noisy input data.

Gaussian processes [19], [20] and Bayesian methods [21], [22] combine the advantages of data-driven methods with kernel functions to estimate posterior distributions with high confidence prediction. The key idea is the design of adequate kernel functions that capture the drone physics to infer the future trajectory. If the drone physics is known in advance, then Kalman filters and its variants [23]–[25] can be used for trajectory prediction. However, it is well known that wrong physics model can cause large bias in the predicted trajectory.

Two different categories for intent prediction are distinguished in the previous methodologies: i) high-level intent classification associated to the drone's purpose of use, e.g., delivery, surveillance, etc., and ii) trajectory intent prediction which aims to predict the trajectory that the drone will follow in future time steps [26]. Despite predictive models give an insight about the drone's future trajectory, they do not uncover the hidden nature of intent. Therefore, a different approach is required to identify the drone's intent for any trajectory.

A. Related Work

In a control perspective, the intent of a drone is hidden within the flight controller, that is, the control law is designed to ensure the drone tracks a desired trajectory or destination by minimizing an objective function [27], [28]. This objective function defines the intention of the drone that we want to identify. This fact is consistent with reinforcement learning theory [29], [30] where the reward function is the most succinct, transferable and robust definition of the task [31] that the agent (in this case a drone) will perform. Here, the objective function and the reward function are equivalent. Hence, we are interested in extracting the objective function of the drone's controller to infer its intent for any trajectory destination.

There exists different methodologies to extract the objective function of controllers based on inverse optimal control (IOC) algorithms [32] and inverse reinforcement learning (IRL) techniques [33]. Whilst IOC is a model-based algorithm which is mainly used for linear systems under quadratic objective functions, IRL is a model-free algorithm which has been used for linear systems and for nonlinear systems under binary

This work was supported by the Royal Academy of Engineering and the Office of the Chief Science Adviser for National Security under the UK Intelligence Community Postdoctoral Research Fellowship programme.

A. Perrusquía and Weisi Guo are with the School of Aerospace, Transport and Manufacturing, Cranfield University, Bedford, UK
(e-mail: {Adolfo.Perrusquia-Guzman,Weisi.Guo}@cranfield.ac.uk).

reward functions [34]–[36], and quadratic structures. However, drone physics is strongly nonlinear which makes difficult to extract the objective function. Furthermore, binary rewards are not commonly used in low-level control architectures since they can generate discontinuous control policies that can lead to undesired performances or chattering effects.

Human Behaviour Learning (HBL) [37] has been used for experience inference. The idea of this approach is to infer experience to another system by combining expert knowledge with on-line interaction. Whilst the IRL aims to uncover the objective function of a system from data trajectories, the HBL algorithm finds the objective function that drives the system to behave as the expert system. However, the HBL approach is conservative and require some knowledge of the objective function and a linear dynamic model assumption. In addition, both HBL and IRL obtain multiple solutions for the objective function which confuse our understanding of the real user's intention.

Hierarchical primitive-based learning [38], imitation learning [39], and safety critical control architectures [40] have been adopted to extrapolate or infer optimal performances to similar systems under similar trajectories or tasks. However, the objective function is still hidden and require an additional algorithm (e.g., IOC/IRL) to extract it from the measurements of the system trajectories.

In view of the above, this paper proposes a policy error IRL (PEIRL) approach [39] for drone's objective function inference. The algorithm is inspired by the closed-loop input error technique of previous work [41] for parameter identification and trajectory inference. Here, the algorithm learns the hidden objective function associated to the observed trajectories using the input error between the real control policy and the policy obtained from an off-line reinforcement learning. The convergence of the proposed algorithm is assessed using Lyapunov recursions [42]. Simulation studies are carried out to demonstrate the benefits of the approach.

B. Contributions

The main features of the proposed work are the following:

- A novel policy error inverse reinforcement learning algorithm for drone's objective inference based on on-line data trajectories.
- The conservative assumptions of previous works are relaxed by avoiding prior knowledge of the objective function and linear model assumptions.
- Convergence to the exact objective function is guaranteed under a constrained batch least-squares rule.

The contribution of this work with respect to previous developments for drone intent prediction are the following:

1) The proposed approach does not require knowledge of the drone physics and high amount of data.
2) Instead of predicting future trajectory or classify the high-level intent, this approach extracts the hidden objective function which is associated to the main task that the drone aims to achieve (intent).
3) The proposed PEIRL combines data-driven with online learning to extract the exact objective function under function constraints.
4) Prior knowledge of the objective weights is avoided.
5) The approach does not require interaction with the real system to test new policies derived by new objective functions.

C. Outline of the paper and notations

The paper outline is as follows: Section II presents the drone's dynamics and preliminaries where the main assumptions and constraints are clearly stated. Section III develops the mission profile objective function extraction where the proposed PEIRL algorithm is designed in detail. Section IV reports simulations studies using a quadcopter model. Conclusions and future work are reported in Section V.

Throughout this paper, $\mathbb{N}$, $\mathbb{R}$, $\mathbb{R}^n$, $\mathbb{R}^{n\times m}$ denote the spaces of natural numbers, real numbers, real n-vectors, and real $n\times m$-matrices, respectively, $I_n \in \mathbb{R}^{n\times n}$ is an $n\times n$ identity matrix, $A>0$ defines a positive definite matrix, $\bar{\otimes}$ and $\otimes$ are the symmetric and standard Kronecker products; where $x\in\mathbb{R}^n,\ A\in\mathbb{R}^{n\times n}$ and $n,m\in\mathbb{N}$.

II. Drone's Dynamics and Preliminaries

The dynamic model of a drone verifies the following Euler-Lagrange equation [43]

$$M\ddot{q}+C(q,\dot{q})\dot{q}+G(q)=u, \tag{1}$$

where $M\in\mathbb{R}^{m\times m}$ is a symmetric and positive definite inertia matrix, $C(q,\dot{q})\in\mathbb{R}^{m\times m}$ denotes the centripetal and Coriolis forces, $G(q)\in\mathbb{R}^m$ is the gravity vector, $u\in\mathbb{R}^m$ defines the input vector, and $q,\dot{q},\ddot{q}\in\mathbb{R}^m$ are the position, velocity, and acceleration vectors.

Remark 1: The dynamic model (1) is a general notation for many Euler-Lagrange systems, e.g., rigid robots, mobile robots, drones, etc. However, for this approach it is required that the inertia matrix M to be constant or approximately constant in order to apply the proposed methodology. Drones are a special class of Euler-Lagrange systems that fulfils this requirement.

The drone model (1) in state space notation is written as

$$\dot{x}=f(x)+Bu, \tag{2}$$

where

$$f(x)=\begin{bmatrix}\dot{q}\\ -M^{-1}[C(q,\dot{q})\dot{q}+G(q)]\end{bmatrix}\in\mathbb{R}^n$$
$$B=\begin{bmatrix}0_m\\ M^{-1}\end{bmatrix}\in\mathbb{R}^{n\times m},\quad x=\begin{bmatrix}q\\ \dot{q}\end{bmatrix}\in\mathbb{R}^n,\quad n=2m.$$

Assumption 1: The drone is controlled by an unknown stabilizing optimal controller u^* under hidden mission profile objective function $\xi(x,x^d,u)$ and destination $x^d\in\mathbb{R}^n$.

Define the error between the drone's trajectories and the desired destination as

$$e:=x-x^d. \tag{3}$$

Differentiating (3) with respect to time yields the following closed-loop error dynamics

$$\dot{e}=f(x)+Bu-\dot{x}^d. \tag{4}$$

A. The Optimal Control Problem

Assume that the stabilizing controller minimizes the following infinite horizon value function [44]

$$V(e) = \int_t^\infty (S(e) + u^\top R u) d\tau, \tag{5}$$

where $S(e) \geq 0 \in \mathbb{R}$ is a positive semi-definite weight function and $R = R^\top > 0 \in \mathbb{R}^{m\times m}$ is a positive definite weight matrix. Both $S(e)$ and R define the weights associated to the hidden mission profile objective function.

Assumption 2: Only measurements of the tracking error e, the states x, control input u, and the objective function $\xi(e,u)$ values are available. The drone's dynamics $f(\cdot)$ and B, and weight matrices $S(e)$ and R are unknown.

Assumption 3: The measurements used in this approach are obtained from cooperative sensors that are not embedded in the drone, e.g., radar, lidar, radio transponder, etc. These sensors do not have communication constraints that make difficult to obtain the necessary signals for the proposed approach. The measurements are assumed to be preprocessed using filters or low-rank approximation methods to attenuate noise, e.g., PCA or dynamic mode decomposition (DMD) [45] to lump together the noise in the principal components associated to the smallest eigenvalues. Unobservable states can be obtained using Kalman filter or closed-loop output error techniques [14].

Remark 2: Noise measurements are usually used as probing noise to ensure parameter estimates convergences on either on/off-policy architectures. This probing noise covers random noise, sinusoidal, exponential-decay signals, and so on. It has been proved in [46] that off-policy architectures have as outcome unbiased solutions despite the presence of noise.

Remark 3: The structure of the proposed objective function $\xi(e,u) = S(e) + u^\top R u$ gives a family of objective functions by defining different positive semi-definite functions $S(e)$. However, the solution of the optimal control problem may not exist for any objective function. The converse problem [30] can be used to find an optimal control policy by imposing an objective function structure and a desired solution for the Hamilton-Jacobi-Bellman equation. Nevertheless, this method requires to modify the dynamic model structure of the drone which is not feasible since the dynamics of the drone is fixed.

The Hamiltonian of the problem between (4) and the (5) is

$$H(e,u,\nabla V) = S(e) + u^\top R u + \nabla V(f(x) + Bu - \dot{x}^d), \tag{6}$$

and the optimal value function by

$$V^*(e) = \min_u \int_t^\infty (S(e) + u^\top R u) d\tau, \tag{7}$$

which verifies the following Hamilton-Jacobi-Bellman (HJB) equation

$$\min_u [H(e,u,\nabla V^*)] = 0, \tag{8}$$

where $\nabla = \frac{\partial}{\partial e}$ is the gradient of a function respect to the error e. The control policy that satisfies (8) is computed by differentiating the Hamiltonian (6) respect to u and equalling to zero as

$$\frac{\partial H}{\partial u} = 2u^\top R + \nabla V^* B = 0,$$

then the optimal control policy is

$$u^* = -\frac{1}{2} R^{-1} B^\top \nabla^\top V^*(e) := -\mathcal{K}(e), \tag{9}$$

where $\mathcal{K}(e) \in \mathbb{R}^m$ comprises the control policy structure. The HJB equation written in terms of the optimal control policy (9) verifies

$$\begin{aligned} \nabla V^*(e)\left(f(x) - \dot{x}^d\right) \quad & -\frac{1}{4}\nabla V^*(e) B R^{-1} B^\top \nabla^\top V^*(e) \\ & + S(e) = 0. \end{aligned} \tag{10}$$

B. Value function approximation

The solution of the HJB equation is hard to find even if we have complete knowledge of the drone's dynamics and the weight matrices of the objective function. Most ADP/RL algorithms use the Weierstrass high-order approximation theorem [44], [47] to approximate the optimal value function (5) and its gradient in terms of a complete independent basis set $\{\phi_i(x)\}$ as

$$\begin{aligned} V^*(e) &= W^{*\top}\phi(e) + \varepsilon_V, \\ \nabla V^*(e) &= W^{*\top}\nabla\phi(e) + \nabla\varepsilon_V, \end{aligned} \tag{11}$$

where $\phi(e) = [\phi_1(e), \cdots, \phi_N(e)]^\top : \mathbb{R}^n \to \mathbb{R}^N$ is a N-dimensional vector of basis functions, $W^* \in \mathbb{R}^N$ defines an optimal vector of weights, and $\varepsilon_V \in \mathbb{R}$ is a residual error due to the approximation. Then, the HJB equation written in terms of the basis function approximation (11) is

$$\begin{aligned} S(e) \quad & -\frac{1}{4} W^{*\top}\nabla\phi(e) B R^{-1} B^\top \nabla^\top \phi(e) W^* \\ & + W^{*\top}\nabla\phi(e)\left(f(x) - \dot{x}^d\right) = \varepsilon_H, \end{aligned} \tag{12}$$

where

$$\begin{aligned} \varepsilon_H = & -\nabla\varepsilon_V\left(f(x) - \dot{x}^d\right) + \frac{1}{2} W^{*\top}\nabla\phi(e) R^{-1} B^\top \nabla^\top \varepsilon_V \\ & + \frac{1}{4}\nabla\varepsilon_V B R^{-1} B^\top \nabla^\top \varepsilon_V \in \mathbb{R}, \end{aligned}$$

defines the residual error of the HJB which can be reduced if the set basis functions accurately approximate the optimal value function (7) such that $\varepsilon_V \approx 0$. Therefore, the optimal control policy is rewritten as

$$u^* = -\frac{1}{2} R^{-1} B^\top \nabla^\top \phi(e) W^* \approx -\mathcal{K}(e). \tag{13}$$

Assuming that the solution of the HJB equation (12) exists, then without loss of generality, we can equivalently write (13) as

$$u^* = -K\Phi(e), \tag{14}$$

where $K \in \mathbb{R}^{m\times M}$ is an optimal gain matrix and $\Phi(e) : \mathbb{R}^n \to \mathbb{R}^M$ is a vector composed of unique elements different to zero of the gradient $\nabla\phi(e)$. The Hamiltonian (6) in terms of (14) is written as

$$\begin{aligned} H(e,K) = \quad & S(e) + (K\Phi(e))^\top R K \Phi(e) \\ & + W^\top \nabla\phi(e)[f(x) - BK\Phi(e) - \dot{x}^d]. \end{aligned} \tag{15}$$

The parameterization in (14) is crucial for the proposed objective function extraction algorithm which is discussed in the next section.

III. Mission's Profile Objective Function Extraction

The proposed PEIRL scheme is depicted in Fig. 1. The measurements of the drone's trajectories feed two complementary learning [37] architectures based on a parameterization of the real control policy and an off-line reinforcement learning algorithm that obtains in each episode i new improved control policies based on iterative weight matrices S_i and R_i. The difference between the real and reinforcement learning control policies feeds an inverse reinforcement learning algorithm to estimate new improved weight matrices S_{i+1} and R_{i+1}. Subsequently these matrices are used in the off-line reinforcement learning algorithm and the procedure is repeated until the policy error is less than a proposed threshold.

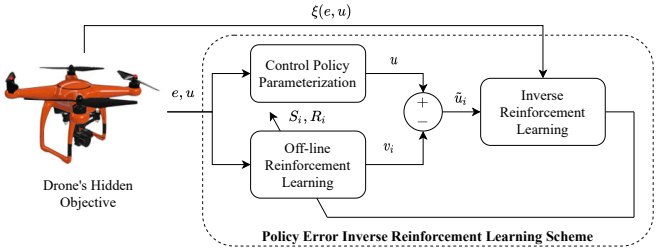


Fig. 1. Performance Objective Extraction Scheme

A. Policy Parameterization using On-line Data

The measurements of the drone's trajectories are stored in the following buffer matrices

$$\begin{aligned} U &= \begin{bmatrix} u(t-(p-1)T) & \cdots & u(t-T) & u(t) \end{bmatrix} \in \mathbb{R}^{m\times p} \\ X &= \begin{bmatrix} x(t-(p-1)T) & \cdots & x(t-T) & x(t) \end{bmatrix} \in \mathbb{R}^{n\times p} \\ E &= \begin{bmatrix} e(t-(p-1)T) & \cdots & e(t-T) & e(t) \end{bmatrix} \in \mathbb{R}^{n\times p} \end{aligned} \tag{16}$$

These terms construct a set of memory matrices that give feedback of the drone's performance under a hidden objective function. We can estimate the gain matrix K of (14) by solving the next batch least-squares (LS) rule

$$\begin{aligned} U &= -\widehat{K}\bar{\Phi}(E), \\ \widehat{K} &= -U\bar{\Phi}^\top(E)\left(\bar{\Phi}(E)\bar{\Phi}^\top(E)\right)^{-1}, \end{aligned} \tag{17}$$

where $\widehat{K} \in \mathbb{R}^{m\times M}$ is an approximation of the optimal gain matrix K due to the presence of noise and measurement errors, and $\bar{\Phi}(E) : \mathbb{R}^{n\times p} \to \mathbb{R}^{M\times p}$. So, the control policy is

$$u = -\widehat{K}\Phi(e). \tag{18}$$

Remark 4: The weight matrices associated to the objective function remain fixed in the drone's mission profile, that is, different destinations are reflected in the states and control input measurements which are consistent with the objective function values.

If we assume linear control policies, the basis functions are equivalent to the error states, i.e., $\Phi(E) = E$, then is easy to extract the desired destination as

$$X^d = X - E, \tag{19}$$

where $X^d \in \mathbb{R}^{n\times p}$ is a matrix containing the trajectories of the desired destination profile.

B. Off-line Reinforcement Learning

An off-line policy iteration algorithm [48] is used to approximate the real near optimal solution $V^*(e)$ under iterative weight matrices $S_i(e)$ and R_i. This model evaluates different weight matrices to find new policies which are close to the real drone's policy [49].

The nonlinear drone's dynamics can be written in terms of the off-line control policy v_i as

$$\begin{aligned} \dot{e} &= f(x) - \dot{x}^d + B(u + v_i^j - v_i^j), \quad v_i^j = -K_i^j\Phi(e), \\ &= f(x) - \dot{x}^d - BK_i^j\Phi(e) + B(u + K_i^j\Phi(e)) \\ &= g(x) + B(u + K_i^j\Phi(e)) \end{aligned} \tag{20}$$

where $g(x) := f(x) - \dot{x}^d - BK_i^j\Phi(e)$, and K_i^j is the control gain matrix in iteration j of episode i. The Hamiltonian (15) is rewritten as an iterative equation as

$$\begin{aligned} H(e, K_i^j) =& S_i(e) + (K_i^j\Phi(e))^\top R_i K_i^j\Phi(e) \\ &+ W_i^{j\top}\nabla\phi(e)[f(x) - BK_i^j\Phi(e) - \dot{x}^d] = 0. \end{aligned} \tag{21}$$

The term in brackets in (21) is equivalent to (20), so

$$\begin{aligned} H(e, K_i^j) =& S_i(e) + (K_i^j\Phi(e))^\top R_i K_i^j\Phi(e) \\ &+ W_i^{j\top}\nabla\phi(e)[g(x) + B(u + K_i^j\Phi(e))]. \end{aligned} \tag{22}$$

Notice that the Hamiltonian (22) is not equal to zero and is not equivalent to (15) because is written in terms of $g(x)$ which has a different dynamics in comparison to $f(x)$. Therefore, the RL algorithms needs to compute the optimal weights and gain associated to the new dynamics. For this purpose, let define $K^{j+1}\Phi(e) = \frac{1}{2}R_i^{-1}B^\top\nabla^\top\phi(e)W_i^j$ and notice that the term $B(u+K_i^j\Phi(e))$ is regarded as an additional term that must be compensated to satisfy the Hamiltonian equality. Hence, the Hamiltonian of the off-line RL H_n is given by

$$\begin{aligned} H_n :=& S_i(e) + (W_i^j)^\top\nabla\phi(e)[g(x) + B(u + K_i^j\Phi(e))] \\ &+ \Phi^\top(e)(K_i^j)^\top R_i K_i^j\Phi(e) \\ &- 2(u + K_i^j\Phi(e))^\top R_i K_i^{j+1}\Phi(e) = 0. \end{aligned} \tag{23}$$

Integrating the Hamiltonian (23) in a time-interval $[t : t+T]$ gives the integral reinforcement learning equation [50]

$$\begin{aligned} &(\phi(e(t+T)) - \phi(e(t))^\top W_i^j \\ &\quad - 2\int_t^{t+T}(u + K_i^j\Phi(e))^\top R_i K_i^{j+1}\Phi(e)d\tau \\ &\quad = -\int_t^{t+T}\left(S(e) + \Phi^\top(e)(K_i^j)^\top R_i K_i^j\Phi(e)\right)d\tau. \end{aligned} \tag{24}$$

We construct a system of equations from κ samples of (24) measured along the drone's trajectories (2) as

$$\Psi\Theta = \Omega \tag{25}$$

where

$$\Theta = \begin{bmatrix} W_i^j \\ \text{vec}(K_i^{j+1}) \end{bmatrix} \in \mathbb{R}^{\iota},$$
$$\Psi = \left[z, -2[I_{\Phi\Phi}(I_M \otimes (K_i^j)^\top R_i) + I_{\Phi u}(I_M \otimes R_i)\right] \in \mathbb{R}^{\kappa \times \iota},$$
$$\Omega = -I_S - I_{\Phi\Phi}\text{vec}((K_i^j)^\top R_i K_i^j) \in \mathbb{R}^{\kappa},$$
$$z = \left[\phi(e(\tau))\Big|_t^{t+T}, \cdots, \phi(e(\tau))\Big|_{\mathcal{T}-T}^{\mathcal{T}}\right]^\top,$$
$$I_{\Phi\Phi} = \left[\int_t^{t+T} \Phi(e) \otimes \Phi(e) d\tau, \cdots, \int_{\mathcal{T}-T}^{\mathcal{T}} \Phi(e) \otimes \Phi(e) d\tau\right]^\top$$
$$I_S = \left[\int_t^{t+T} S_i(e) d\tau, \cdots, \int_{\mathcal{T}-T}^{\mathcal{T}} S_i(e) d\tau\right]^\top,$$
$$I_{\Phi u} = \left[\int_t^{t+T} \Phi(e) \otimes u d\tau, \cdots, \int_{\mathcal{T}-T}^{\mathcal{T}} \Phi(e) \otimes u d\tau\right]^\top,$$

where $\mathcal{T} := t + \kappa T$. Then the weights and control gain are computed by the following batch least-squares rule

$$\Theta \quad = (\Psi^\top \Psi)^{-1} \Psi^\top \Omega. \tag{26}$$

The weights W_i^j and gain matrix K_i^j define the optimal solution associated to the performance weights $S_i(e)$ and R_i. In the next section, the real and reinforcement learning policies are connected to obtain new and improved weight matrices $S_i(e)$ and R_i that are close to the real performance matrices.

C. Inverse Reinforcement Learning

Standard IRL algorithms use the information of reinforcement learning algorithm to estimate the weight matrices of the objective function. In this paper, the IRL algorithm uses the information of both the real and RL control policies in a complementary learning mechanism to estimate the real weight matrices.

First, notice that from the parameterization (14) it holds that

$$\nabla^\top \phi(e) W \equiv P \Phi(e), \tag{27}$$

where $P \in \mathbb{R}^{n \times M}$ is a matrix composed by the values of the weights W and the coefficients of the gradient $\nabla\phi$. Then for any control policy $u = -K\Phi(e)$ and using (27) we obtain that

$$\begin{aligned} u &= -\frac{1}{2} R^{-1} B^\top \nabla^\top \phi(e) W, \\ &= -\frac{1}{2} R^{-1} B^\top P \Phi(e) \\ &= -K\Phi(e), \quad K = \frac{1}{2} R^{-1} B^\top P. \end{aligned} \tag{28}$$

When the weights and gain matrices of the off-line RL converge, then we can extract information associated to the input dynamics B and the current weight matrix R_i as

$$\begin{aligned} K_i &= \tfrac{1}{2} R_i^{-1} B^\top P_i \\ 2K_i P_i^\dagger &= R_i^{-1} B^\top, \end{aligned} \tag{29}$$

where $P_i^\dagger \in \mathbb{R}^{M \times n}$ is the Moore-Penrose pseudoinverse of matrix P_i.

The policy error [41] between the real (18) and off-line RL control policies (20) is used to connect these control policies for the objective extraction. The policy error is defined as

$$\begin{aligned} \widetilde{u}_i &= v_i - u \\ &= -\tfrac{1}{2} R_i^{-1} B^\top \nabla^\top \phi(e) W_i + \widehat{K} \Phi(e). \end{aligned} \tag{30}$$

We want to minimize the following quadratic input error index

$$\widetilde{U}_i = \widetilde{u}_i^\top \widetilde{u}_i. \tag{31}$$

The above index is minimized when $v_i \to u$ as $t \to \infty$. In other words, we want that both the off-line RL and the real control policies be the same which means that the actual objective function can achieve the desired optimal performance. Notice that many objective functions can achieve the same performance of the real control policy. In the sequel of the paper, we provide an effective method to ensure convergence to the exact weight matrices.

From (30), it is clear that W_i is a free parameter that we can tune to obtain new and improved weight matrices $S_{i+1}(e)$ and R_{i+1}. The new weights $\mathcal{W}_i \in \mathbb{R}^N$ are updated by the following one-step gradient descent rule

$$\begin{aligned} \mathcal{W}_i &= W_i - \alpha \frac{\partial \widetilde{U}_i}{\partial W_i} \\ &= W_i + \alpha \nabla\phi(e) B R_i^{-1} \widetilde{u}_i \\ &= W_i + 2\alpha \nabla\phi(e) \left(K_i P_i^\dagger\right)^\top \widetilde{u}_i, \end{aligned} \tag{32}$$

where $\alpha > 0$ is the learning rate. The gradient of the value function approximation is equivalently written as

$$\nabla^\top \phi(e) \mathcal{W}_i = \mathcal{P}_i \Phi(e), \tag{33}$$

where $\mathcal{P}_i \in \mathbb{R}^{n \times M}$ is a rectangular matrix composed of the new weights in $\mathcal{W}_i$ and the coefficients of the gradient $\nabla\phi(e)$. Then, we can compute a control gain $\mathcal{K}_i \in \mathbb{R}^{m \times M}$ associated to the matrix $\mathcal{P}_i$ as

$$\mathcal{K}_i := K_i P_i^\dagger \mathcal{P}_i. \tag{34}$$

Following a similar procedure to the off-line RL algorithm, one can write

$$\begin{aligned} \dot{e} &= f(x) - \dot{x}^d + B(u + \nu_i - \nu_i), \qquad \nu_i = -\mathcal{K}_i \Phi(e), \\ &= h(x) + B(u + \mathcal{K}_i \Phi(e)), \end{aligned} \tag{35}$$

where $h(x) := f(x) - \dot{x}^d - B\mathcal{K}_i\Phi(e)$. The Hamiltonian associated to the new control gain $\mathcal{K}_i$ and weights $\mathcal{W}_i$ is given by

$$\begin{aligned} H(e, \mathcal{K}_i) :=& S_{i+1}(e) + (\mathcal{K}_i \Phi(e))^\top R_{i+1} \mathcal{K}_i \Phi(e) \\ &+ \mathcal{W}_i^\top \nabla\phi(e)[f(x) - B\mathcal{K}_i\Phi(e) - \dot{x}^d] = 0, \end{aligned} \tag{36}$$

where $S_{i+1}(e)$ and R_{i+1} are unknown weight matrices that satisfy the Hamiltonian equality. A procedure similar to the off-line RL is followed to obtain the Hamiltonian of the IRL H_s. The term in brackets in (36) is equivalent to (35), so

$$\begin{aligned} H(e, \mathcal{K}_i) :=& S_{i+1}(e) + (\mathcal{K}_i \Phi(e))^\top R_{i+1} \mathcal{K}_i \Phi(e) \\ &+ \mathcal{W}_i^\top \nabla\phi(e)[h(x) + B(u + \mathcal{K}_i\Phi(e))]. \end{aligned} \tag{37}$$

Let $\mathcal{K}_i\Phi(e) := \frac{1}{2}R_{i+1}^{-1}B^\top\nabla^\top\phi(e)\mathcal{W}_i$ and compensate the term $B(u+\mathcal{K}_i\Phi(e))$. Then the Hamiltonian H_s of the IRL is given by

$$\begin{aligned}H_s :=& S_{i+1}(e) + \mathcal{W}_i^\top\nabla\phi(e)[h(x)+B(u+\mathcal{K}_i\Phi(e))]\\ &- \nu_i^\top R_{i+1}\nu_i + 2u^\top R_{i+1}\nu_i = 0.\end{aligned} \tag{38}$$

Remark 5: The weight function $S_i(e)$ can be linearly parametrized as $S_i(e) := \vartheta_i^\top\psi(e)$, where $\vartheta \in \mathbb{R}^s$ is a vector of unknown weights and $\psi(e) : \mathbb{R}^n \to \mathbb{R}^s$ is a vector of known polynomial basis functions. In addition, the maximum degree of the basis functions in $\psi(e)$ matches with the maximum degree of the basis functions in $\phi(e)$.

Integrating the Hamiltonian H_s in a time-interval $[t : t+T]$ gives

$$\begin{aligned}\int_t^{t+T}\psi^\top(e)d\tau\cdot\vartheta_{i+1} + \mathcal{W}_i^\top[\phi(e(t+T)) - \phi(e(t))]\\ + 2\int_t^{t+T}\mu_i^\top R_{i+1}\nu_i d\tau = 0,\end{aligned} \tag{39}$$

where $\mu_i(e) := u - \frac{1}{2}\nu_i(e)$. We construct a system of equations from ζ samples of (39) measured along the trajectories (2) as

$$\Sigma\Pi = \Upsilon, \tag{40}$$

where

$$\begin{aligned}\Pi &= \begin{bmatrix}\vartheta_{i+1}\\ \text{vec}(R_{i+1})\end{bmatrix} \in \mathbb{R}^\rho, \quad \rho = s + m^2,\\ \Sigma &= \begin{bmatrix}I_\psi & 2I_{\eta\nu}\\ I_{\psi\xi} & I_{uu}\end{bmatrix} \in \mathbb{R}^{2\zeta\times\rho},\\ \Upsilon &= \begin{bmatrix}I_{\phi\zeta}\mathcal{W}_i\\ \Xi\end{bmatrix} \in \mathbb{R}^{2\zeta},\\ I_{\phi\zeta} &= -\left[\phi(e(\tau))\Big|_t^{t+T}, \cdots, \phi(e(\tau))\Big|_{\mathcal{T}_\zeta-T}^{\mathcal{T}_\zeta}\right]^\top\\ I_{\eta\nu} &= \left[\int_t^{t+T}\mu_i\otimes\nu_i d\tau, \cdots, \int_{\mathcal{T}_\zeta-T}^{\mathcal{T}_\zeta}\mu_i\otimes\nu_i d\tau\right]^\top,\\ I_\psi &= \left[\int_t^{t+T}\psi(e)d\tau, \cdots, \int_{\mathcal{T}_\zeta-T}^{\mathcal{T}_\zeta}\psi(e)d\tau\right]^\top,\\ I_{\psi\xi} &= [\psi(e(t)), \cdots, \psi(e(t+\zeta T))]^\top,\\ I_{uu} &= [u(t)\otimes u(t), \cdots, u(t+\zeta T)\otimes u(t+\zeta T)]^\top,\\ \Xi &= \begin{bmatrix}\xi(t), & \cdots, & \xi(t+\zeta T)\end{bmatrix}^\top,\end{aligned}$$

with $\mathcal{T}_\zeta := t+\zeta T$. Notice that in this case, we collect samples of the real performance objective function $\xi(t)$ to avoid the existence of multiple solutions. The solution of (40) is

$$\Pi \quad = (\Sigma^\top\Sigma)^{-1}\Sigma^\top\Upsilon. \tag{41}$$

Theorem 1: The weights of the performance objective function can be exactly estimated by the proposed PEIRL algorithm as long as the policy error $\widetilde{u}$ converges to zero and if the IRL batch LS rule meets the objective function constraint Ξ.

Proof: Assume $\varepsilon_H = 0$. The Hamiltonian in (38) is equivalent to the following HJB equation

$$\begin{aligned}S_{i+1}(e) =& -\mathcal{W}_i^\top\nabla\phi(e)\left(f(x)-\dot{x}^d\right)\\ &+\frac{1}{4}\mathcal{W}_i^\top\nabla\phi(e)BR_{i+1}^{-1}B^\top\nabla^\top\phi(e)\mathcal{W}_i.\end{aligned} \tag{42}$$

Additionally, the Hamiltonian (23) is equivalent to the following HJB equation

$$\begin{aligned}S_{i+1}(e) =& -W_{i+1}^\top\nabla\phi(e)\left(f(x)-\dot{x}^d\right)\\ &+\frac{1}{4}W_{i+1}^\top\nabla\phi(e)BR_{i+1}^{-1}B^\top\nabla^\top\phi(e)W_{i+1}.\end{aligned} \tag{43}$$

Recall that $\mathcal{W}_i = W_i - \alpha\frac{\partial\widetilde{U}}{dW_i}$, then $\mathcal{W}_i$ converges to W_i as v_i approaches to u. In other words, we have that in the limit the following holds

$$\lim_{i\to\infty} v_i = u \Longrightarrow \lim_{i\to\infty}\mathcal{W}_i = W_i.$$

Equalling the HJB equations (42) and (43) gives

$$\begin{aligned}&\lim_{i\to\infty}\Big(-W_{i+1}^\top\nabla\phi(e)\left(f(x)-\dot{x}^d\right)\\ &\qquad+\frac{1}{4}W_{i+1}^\top\nabla\phi(e)BR_{i+1}^{-1}B^\top\nabla^\top\phi(e)W_{i+1}\Big) =\\ &\lim_{i\to\infty}\Big(-W_i^\top\nabla\phi(e)\left(f(x)-\dot{x}^d\right)\\ &\qquad+\frac{1}{4}W_i^\top\nabla\phi(e)BR_{i+1}^{-1}B^\top\nabla^\top\phi(e)W_i\Big).\end{aligned}$$

If we add $\frac{1}{4}W_i^\top\nabla\phi(e)R_i^{-1}B^\top\nabla^\top\phi(e)W_i$ in both sides of the above equality gives

$$\begin{aligned}&\lim_{i\to\infty}\Big(S_{i+1}(e)-\frac{1}{4}W_i^\top\nabla\phi(e)BR_{i+1}^{-1}B^\top\nabla^\top\phi(e)W_i\Big) =\\ &\qquad\lim_{i\to\infty}\Big(S_i(e)-\frac{1}{4}W_i^\top\nabla\phi(e)BR_i^{-1}B^\top\nabla^\top\phi(e)W_i\Big).\end{aligned}$$

The above equality has multiple solutions for different combinations between the weights $S_i(e)$ and R_i. However, the data collected of the real objective function Ξ restricts the possible solutions such as the only way that the above equality holds is when

$$\lim_{i\to\infty}S_i(e) = S(e), \qquad \lim_{i\to\infty}R_i = R. \tag{44}$$

This completes the proof. ■

Remark 6: This approach only requires two hyperparameters to tune: the learning rate α and the number of episodes i. The learning rate α is tuned manually since it depends on the drone's dynamics. A simple tuning approach is setting α small enough to avoid divergence in the gradient rule and smoothly increase its value to improve the learning performance as the number of episodes i increases. Adaptive learning rates using well-known optimizers, e.g., RMSprop or Adam can be used to adapt the learning rate automatically based on the gradient of the policy error $\widetilde{U}_i$. However, this is not used due to the simplistic nature of the one-step gradient descent rule.

IV. Simulation studies

In this section, we test the proposed algorithm using a quadcopter model. The quadcopter satisfies the following physics model [43]

$$\begin{bmatrix} \ddot{z} \\ \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} \frac{1}{m} c_\theta c_\phi F \\ \frac{1}{I_{xx}} \tau_\phi \\ \frac{1}{I_{yy}} \tau_\theta \\ \frac{1}{I_{zz}} \tau_\psi \end{bmatrix} - \begin{bmatrix} \frac{A_z}{m} \dot{z} \\ \frac{I_{yy}-I_{zz}}{I_{xx}} \dot{\theta}\dot{\psi} \\ \frac{I_{zz}-I_{xx}}{I_{yy}} \dot{\phi}\dot{\psi} \\ \frac{I_{xx}-I_{yy}}{I_{zz}} \dot{\theta}\dot{\phi} \end{bmatrix} - \begin{bmatrix} g \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

where z denotes the altitude and ϕ, θ, ψ denote the orientation, m is the mass of the quadcopter, I_{xx}, I_{yy}, I_{zz} are the moments of inertia, A_z is a drag force, g is the gravitational acceleration, F is the thrust force, τ_ϕ, τ_θ, and τ_ψ are the roll, pitch, and yaw torques. The parameters of the quadcopter are: $m = 0.468$ kg, $I_{xx} = I_{yy} = 4.856 \times 10^{-3}$ kgm^2, $I_{zz} = 8.801 \times 10^{-3}$ kgm^2, $A_z = 0.25$ kg/s, and $g = 9.81$ m/s^2.

We can construct two nonlinear systems to decouple the pose and orientation physics by defining

$$\dot{x}_1 = f_1(x_1) + B_1 u_1$$
$$\dot{x}_2 = f_2(x_2) + B_2 u_2,$$

where $x_1 = [z, \dot{z}]^\top$, $u_1 = c_\theta c_\phi F$, $x_2 = [\phi, \theta, \psi, \dot{\phi}, \dot{\theta}, \dot{\psi}]^\top$, $u_2 = [\tau_\phi, \tau_\theta, \tau_\psi]^\top$, and

$$f_1(x_1) = \begin{bmatrix} \dot{z} \\ -\frac{1}{m} A_z \dot{z} - g \end{bmatrix}, \quad B_1 = \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix},$$

$$f_2(x) = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ -\frac{I_{yy}-I_{zz}}{I_{xx}} \dot{\theta}\dot{\psi} \\ -\frac{I_{zz}-I_{xx}}{I_{yy}} \dot{\phi}\dot{\psi} \\ -\frac{I_{xx}-I_{yy}}{I_{zz}} \dot{\theta}\dot{\phi} \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \frac{1}{I_{xx}} & 0 & 0 \\ 0 & \frac{1}{I_{yy}} & 0 \\ 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix}.$$

Let focus first in the orientation physics. Consider the following quadratic cost index

$$V(e_2) = \int_t^\infty (e_2^\top S^o e_2 + u_2^\top R^o u_2) d\tau,$$

where $S^o \geq 0$ and $R^o > 0$ are unknown symmetric weight matrices and $e_2 = [\phi, \theta, \psi]^\top - [\phi^d . \theta^d, \psi^d]^\top$ is the orientation error and $\phi^d . \theta^d, \psi^d$ define the orientation destination. Then, the controller u_2 has the next structure $u_2 = -K^o e_2$, where $K^o \in \mathbb{R}^{3\times 6}$ is the unknown optimal control gain. The same procedure can be applied to the pose physics.

We can observe that the first nonlinear system can be expressed as a standard linear system with a disturbance as

$$\dot{x}_1 = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{1}{m} A_z \end{bmatrix} x_1 + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} (u_1 - mg).$$

The pose controller has the following structure $u_1 = -K^p e_1 + mg := \bar{K}^p \bar{e}_1$, where $K^p \in \mathbb{R}^{1\times 2}$ stands to the optimal control gain of a $\mathcal{H}_2$ control formulation, $\bar{K}^p = [-K^p, m]$ and $\bar{e}_1 = [e_1^\top, g]^\top$. The controller is designed to minimize the following quadratic cost index

$$V(e_1) = \int_t^\infty (e_1^\top S^p e_1 + u_1^\top R^p u_1) d\tau,$$

for some unknown symmetric matrices $S^p \geq 0$ and $R^p > 0$. Here $e_1 = z - z^d$ is the altitude error and z^d is the quadcopter's altitude destination.

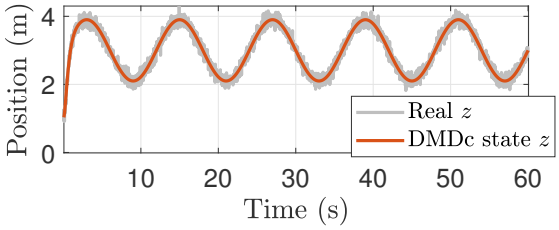


Fig. 2. Noise-free trajectory obtained from the DMDc method

Both the pose and orientation controllers are assumed to stabilize and track the desired destination with low-tracking error. The desired destination is $z^d = 3 + 0.9 \sin\left(\frac{\pi}{6} t\right)$, $\phi^d = \frac{1}{12} \pi \cos\left(\frac{\pi}{12} t\right)$, $\theta^d = \frac{1}{18} \pi \sin\left(\frac{\pi}{12} t\right)$, $\psi^d = 0$.

The optimal pose and orientation controllers are designed in terms of the following performance objective matrices

$$S^p = \begin{bmatrix} 100 & 50 \\ 50 & 100 \end{bmatrix}, \quad R^p = 1,$$
$$S^o = \text{diag}\{2, 2, 2, 1, 1, 1\}, \quad R^o = I_3.$$

These matrices are chosen arbitrarily to observe the effectiveness of the approach for different weight matrices structures, e.g., diagonal and non-diagonal positive semi-definite/definite matrices.

Thus, the approximate optimal control gains are

$$K^p \approx \begin{bmatrix} 10 & 10.2105 \end{bmatrix},$$
$$K^o \approx \begin{bmatrix} 1.414 & 0 & 0 & 1.007 & 0 & 0 \\ 0 & 1.414 & 0 & 0 & 1.007 & 0 \\ 0 & 0 & 1.414 & 0 & 0 & 1.012 \end{bmatrix}.$$

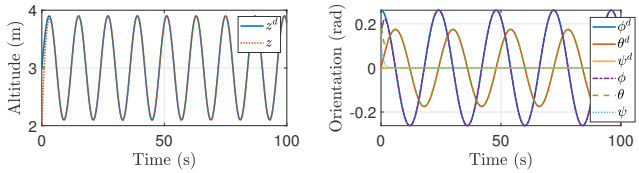
The states are corrupted by a Gaussian distributed noise with zero mean and variance 0.1. We measure the states and control inputs each 1 ms. A dynamic mode decomposition with control (DMDc) [51] is used to attenuate noise. The states of the DMDc output model are used instead of the real states to avoid biased parameter estimation. Fig. 2 shows the noise attenuation of the altitude trajectory.

Each 100 samples, the quadcopter's trajectories are stored in the following matrices $U_1 \in \mathbb{R}^{1\times 100}$, $U_2 \in \mathbb{R}^{3\times 100}$, $E_1 \in \mathbb{R}^{2\times 100}$, and $E_2 \in \mathbb{R}^{6\times 100}$. The real control gains are computed by the batch-LS rules

$$\widehat{\bar{K}}^p = -U_1 \bar{E}_1^\top (\bar{E}_1 \bar{E}_1^\top)^{-1}$$
$$\widehat{K}^o = -U_2 E_2^\top (E_2 E_2^\top)^{-1},$$

where $\widehat{\bar{K}}^p$ and $\widehat{K}^o$ are estimates of $\bar{K}^p$ and K^o, respectively. Here $\bar{E}_1 = \begin{bmatrix} E_1 \\ \vec{g} \end{bmatrix} \in \mathbb{R}^{3\times 100}$ with $\vec{g} = [g, \cdots, g] \in \mathbb{R}^{1\times 100}$. Fig. 3 shows the performance of the quadcopter using the above pose and orientation optimal controllers.

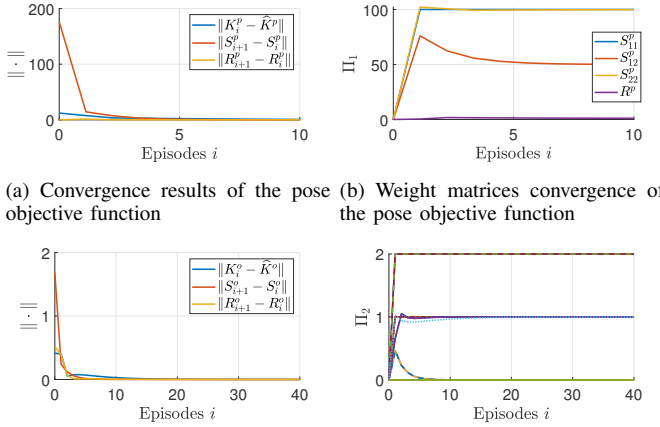
The hidden objective associated to matrices S^p, R^p, S^o, and R^o are extracted from real-time trajectories using the proposed PEIRL. The basis functions are proposed as quadratic functions, that is, $\phi(e_1) = e_1 \bar{\otimes} e_1$ and $\phi_2(e_2) = e_2 \bar{\otimes} e_2$. The



(a) Altitude trajectory (b) Orientation trajectories

Fig. 3. Quadcopter's trajectories

initial weight matrices for the off-line RL are set to: $S_i^p = I_2$, $R_i^p = 0.5$, $S_i^o = 0.5I_6$, and $R_i^o = 0.5I_3$. The learning rates are tuned manually until stable performances are achieved. The final learning rates are: $\alpha^p = 0.003$ and $\alpha^o = 3e^{-5}$. The number of parameters to extract are: $\Pi_1 = 4$ and $\Pi_2 = 27$ for the pose and orientation objective functions, respectively.



(a) Convergence results of the pose objective function (b) Weight matrices convergence of the pose objective function

(c) Convergence results of the orientation objective function (d) Weight matrices convergence of the orientation objective function

Fig. 4. Objective extraction results

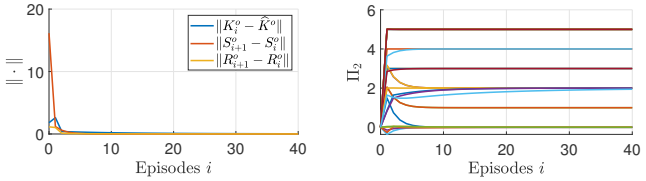
Fig. 4 shows the results of the control policies' objective extraction. Two different scenarios where considered in this case of study: diagonal weight matrices, and non-diagonal weight matrices. Fig. 4(a)-Fig. 4(b) show the results for the pose controller. The proposed PEIRL achieves convergence of the off-line RL control gain K_i^p to the real control gain $\widehat{K}^p$. In addition, the weight matrices S_i^p and R_i^p converge in the first seven episodes. Furthermore, these matrices converge to the real weight matrices (see Fig. 4(b)). On the other hand, the results for the orientation controller shows similar results to the pose controller, however they require more episodes to converge (see Fig. 4(c)-Fig. 4(d)). This is caused due to the selection of the learning rate, that is, large learning rates accelerate convergence of the algorithm, however they can produce instability in the long run. Hence, the selection of the learning rate is crucial to guarantee convergence and stability of the proposed PEIRL approach. The computational time per episode depends highly on the number of parameters to estimate and an user-defined threshold that determines when to stop either the RL or IRL loops. In this paper, we used a threshold of 0.01 for both the RL and IRL loops. The computational time per episode of the pose objective function is approximately 1.23 seconds, whilst for the orientation objective function is approximately 3.92 seconds.

We further test the approach using non-diagonal weight matrices for the orientation controller design. In this case the real weights and gain are:

$$S^o = \begin{bmatrix} 5 & 4 & 3 & 2 & 1 & 0 \\ 4 & 5 & 3 & 1 & 2 & 0 \\ 3 & 3 & 5 & 4 & 3 & 0 \\ 2 & 1 & 4 & 5 & 2 & 0 \\ 1 & 2 & 3 & 2 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 5 \end{bmatrix}, \quad R^o = 2I_3,$$

$$K^o \approx \begin{bmatrix} 1.396 & 0.702 & 0.911 & 1.552 & 0.324 & 0.001 \\ 0.706 & 1.413 & 0.681 & 0.324 & 1.552 & 0.001 \\ -0.231 & -0.094 & 1.098 & 0 & 0 & 1.587 \end{bmatrix}.$$

The results are exhibited in Fig. 5. Notice that the samples in Ξ give a constraint to the batch LS rule to guarantee convergence to the exact weight matrices of the mission profile objective function.

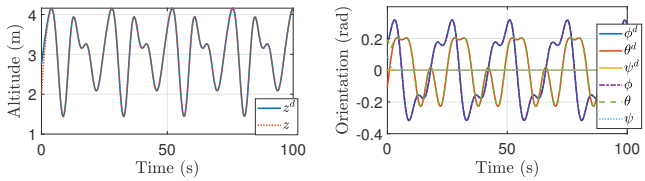


(a) Convergence results (b) Weight matrices convergence

Fig. 5. Orientation objective function results under non-diagonal weight matrices

A. *Robustness results*

First we verify the claim that the proposed technique holds for any drone's trajectories. The drone's desired trajectories are modified to $z^d = 3 + 0.9\sin(\frac{\pi}{6}t) - \frac{1}{2}\cos(\frac{\pi}{4}t) + \frac{1}{4}\cos(\frac{\pi}{3}t)$, $\phi^d = \frac{1}{12}\pi\cos(\frac{\pi}{12}t) - \frac{1}{36}\pi\cos(\frac{\pi}{4}t) + \frac{1}{45}\sin(\frac{\pi}{6}t)$, $\theta^d = \frac{1}{18}\pi\sin(\frac{\pi}{12}t) + \frac{1}{36}\pi\sin(\frac{\pi}{4}t) - \frac{1}{30}\pi\cos(\frac{\pi}{6}t)$, $\psi^d = 0$. The same non-diagonal weight matrices are used for the optimal control design. In addition, we simulate different sensor's sampling time to demonstrate the algorithm robustness, i.e., we measure the states each 0.02 seconds. Fig. 6 exhibit the tracking of the altitude and attitude trajectories.

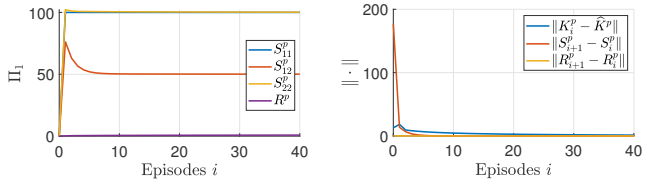


(a) Altitude trajectory (b) Attitude trajectories

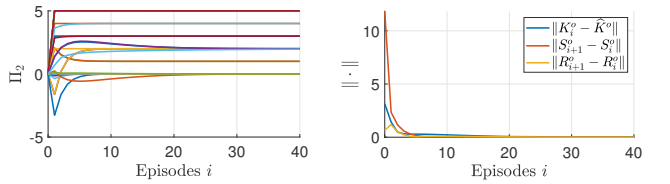
Fig. 6. New trajectories testing

The initial weight matrices of the proposed PEIRL are set to $S_i^p = 0.1I_2$, $R_i^p = 0.1$, $S_i^o = 10I_6$, and $R_i^o = I_3$. The results of the objective inference are shown in Fig. 7. The results show that the same expert's weight matrices are obtained despite the trajectories are modified and measured with a different

sampling time. It is observed that the initial weight matrix R_0 can cause divergence of the proposed PEIRL if it is not close to the exact weight matrix. One solution is to decrease the value of the learning rate α which can cause a slow learning inference or set R_0 close to its real value.



(a) Weight matrices convergence of the pose objective function (b) Convergence results of the pose objective function



(c) Weight matrices convergence of the orientation objective function (d) Convergence results of the orientation objective function

Fig. 7. Robustness results for different drone's desired destinations

B. Sensitivity Analysis

We perform a sensitivity analysis of the proposed PEIRL. One of the main advantages of the proposed approach is that the learning rate α is the only hyperparameter to tune. We test five different learning rates and compare them to obtain more insights of the proposed approach. In addition, we also test the sensitivity of the learning rate α under different initial weight matrices Fig. 8 shows the convergence curves of the error between the estimated kernel matrix $\mathcal{P}_i$ with the optimal P matrix under different initial weight matrices.

Similarly to standard gradient descent rules, the learning rate must be chosen small enough to avoid divergence or convergence to a local minima. Notice that for small initial weights, the PEIRL finds rapidly the solution for relatively large learning rates (see Fig. 8(a)). On the other hand, for relatively large initial weight matrices, the PEIRL requires more episodes to converge despite of using the same learning rates (see Fig. 8(b)). We observe that the initial weight values play a major role in the convergence of the algorithm. Here, for large initial weight matrix S_0 and small weight matrix R_0, the algorithm shows a fast convergence with similar behaviours across all the learning rates. Since R_i is small in the first episodes, then the applied control policy will be large. In consequence, the policy error will be large such that its gradient accelerates the learning phase of the PEIRL.

C. Comparison results

We compare the proposed methodology with a human behaviour learning (HBL) algorithm proposed in [37] and a data-driven IRL algorithm proposed in [52]. We assume that R^p and R^o are known in advance to implement the HBL and IRL algorithms, that is, $R^p = 1$ and $R^o = I_3$. The same initial weight matrices and learning rates are used in this experiment. Here we adapt the IRL algorithm to match to the proposed nonlinear formulation. We consider the first simulation case using diagonal weight matrices. Fig. 9 and Fig. 10 show the objective extraction results for both the pose and orientation objective functions using the HBL and IRL algorithms. The inferred weight matrices of each algorithm are

$$S^p_{HBL} = \begin{bmatrix} 100 & 103.19 \\ 103.19 & 100 \end{bmatrix}, \quad S^p_{IRL} = \begin{bmatrix} 99.97 & 106.41 \\ 106.41 & 100 \end{bmatrix},$$

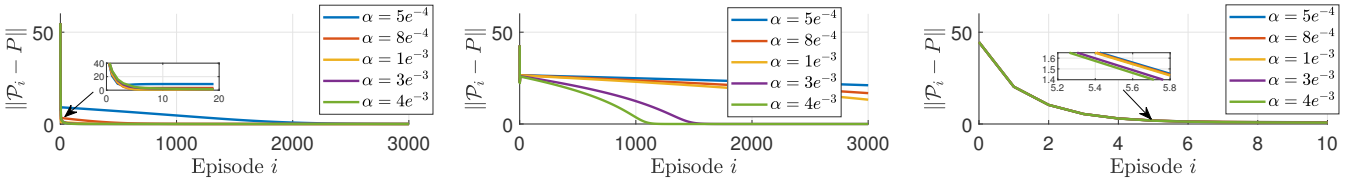
$$S^o_{HBL} = \begin{bmatrix} 2 & 0 & 0 & 0.921 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0.921 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0.926 \\ 0.921 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0.921 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0.926 & 0 & 0 & 1 \end{bmatrix},$$

$$S^o_{IRL} = \begin{bmatrix} 2 & 0 & 0 & 1.417 & 0 & 0 \\ 0 & 2 & 0 & 0 & 1.417 & 0 \\ 0 & 0 & 1.94 & 0 & 0 & 1.405 \\ 1.417 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1.417 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1.405 & 0 & 0 & 1 \end{bmatrix}.$$

Notice that in both cases the weight matrices do not converge to their real values despite the matrix R is known in advance. Furthermore, the matrices S^p_j, $j = HBL, IRL$ are not positive definite and can pose an undesired objective function. Both the HBL and IRL can obtain multiple solutions due to the lack of constraints. These solutions define different intentions that obtain the desired behaviour imposed by the user. However, we cannot identify which is the real objective function that the user uses in the control design and, in consequence, biased conclusions or assumptions are obtained. Here, the constraint proposed in this approach fix this problem by forcing the PEIRL algorithm to converge to the exact weight matrices. Additionally, the constraint allows to infer both weight matrices and relax the conservative assumptions of both HBL and IRL. In terms of the computational time, both HBL and IRL require less time to infer the objective function from the data in comparison to the PEIRL. These differences of time are derived due to the incorporation of the objective constraint in the IRL loop. The computational time per episode of both pose and orientation objectives are 0.76 and 3.05 seconds for the HBL and 0.73 and 2.96 seconds for the IRL. This defines a trade-off between computation time and parameters' inference accuracy that, in view of the results, is acceptable due to the relatively fast convergence results.

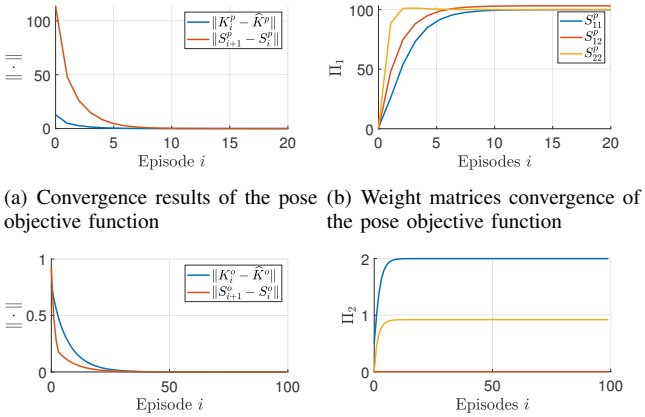
D. Safety Critical Control Application

The objective function inference problem has several applications in the security domain. One of them is safety critical control whose aim is to ensure the safe implementation of control and AI models in real-world applications. In this scenario, the objective function acts as an experience inference architecture that injects a desired behaviour or performance to another system.



(a) Initial weights $S = 0.1I_4$ and $R = 0.1$ (b) Initial weights $S = 10I_4$ and $R = 2$ (c) Initial weights $S = 10I_4$ and $R = 0.1$

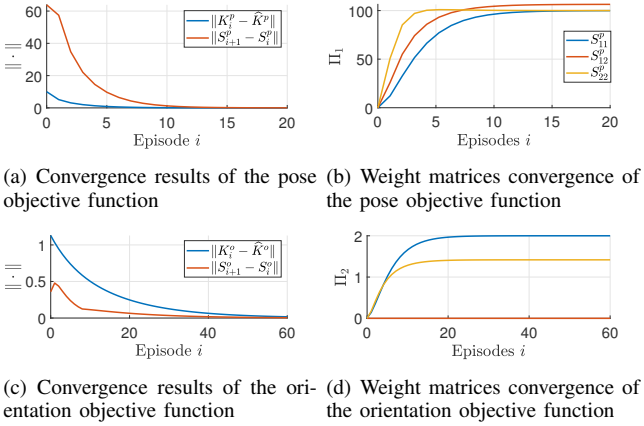
Fig. 8. Sensitivity analysis of the learning rate under different initial weight matrices



(a) Convergence results of the pose objective function (b) Weight matrices convergence of the pose objective function

(c) Convergence results of the orientation objective function (d) Weight matrices convergence of the orientation objective function

Fig. 9. Objective function extraction results using HBL



(a) Convergence results of the pose objective function (b) Weight matrices convergence of the pose objective function

(c) Convergence results of the orientation objective function (d) Weight matrices convergence of the orientation objective function

Fig. 10. Objective function extraction results using IRL

Fig. 11 shows the diagram of the proposed safety critical control application which exhibits a similar architecture to the objective inference problem of Fig. 1. The main differences are the incorporation of expert data and that an on-line RL architecture is used. The expert data defines the data of an expert behaviour that we aim to infer to the real drone by modifying its objective function. We used on-line RL rather than off-line RL due to the nature of the safety critical control problem which requires an on-line implementation. In this implementation, the gains of the expert data are approximately

$$K_e^p \approx \begin{bmatrix} 3.5355 & 3.7339 \end{bmatrix},$$
$$K_e^o \approx \begin{bmatrix} 2.236 & 0 & 0 & 0.722 & 0 & 0 \\ 0 & 2.236 & 0 & 0 & 0.722 & 0 \\ 0 & 0 & 2.236 & 0 & 0 & 0.734 \end{bmatrix},$$

which are obtained using the following weight matrices

$$S^p = \begin{bmatrix} 25 & 5 \\ 5 & 25 \end{bmatrix}, \quad R^p = 2,$$
$$S^o = \text{diag}\{10, 10, 10, 1, 1, 1\}, \quad R^o = 2I_3.$$

The real drone is controlled using the following approximated gains

$$K^p \approx \begin{bmatrix} 1.4142 & 1.5902 \end{bmatrix},$$
$$K^o \approx \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

The learning rate is manually tuned until a fast convergence result is obtained. The final learning rates are: $\alpha^p = 0.003$ and $\alpha^o = 3e^{-5}$. The objective function inference results are shown in Fig. 12. Similarly to the previous experiments, the PEIRL reduces the policy error by inferring the objective function of the expert data to the real drone. Fig. 13 shows the results of the experience inference algorithm for safety critical control.

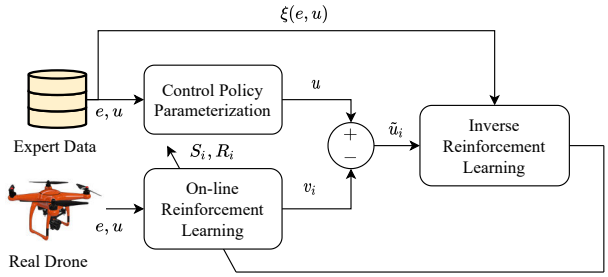
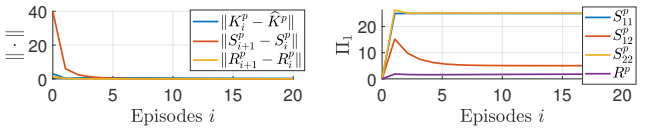


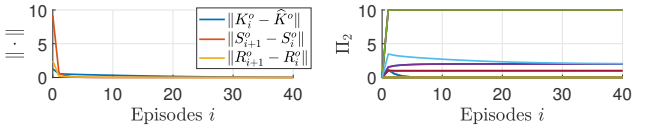
Fig. 11. Safety Critical Control Scheme

The results of Figs. 12-13 show that the objective function is the one that determines the closed-loop performance of the drone. Here, the safety critical control based on the objective function inference algorithm is capable to infer the objective function and obtain the same expert performance. This task is becoming relevant in several applications that involve multi-agent systems and imitation learning. In contrast to HBL and IRL, the inferred objective function is stable and provides the accurate information of the real objective intent. Future work will study the experience inference of drones' policies with different structures and parameters.



(a) Convergence results of the pose objective function

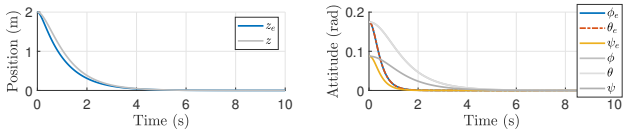
(b) Weight matrices convergence of the pose objective function



(c) Convergence results of the orientation objective function

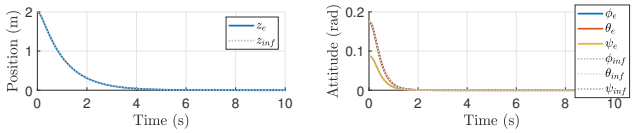
(d) Weight matrices convergence of the orientation objective function

Fig. 12. Objective function inference for Safety Critical Control



(a) Initial Altitude performance

(b) Initial Attitude performance



(c) Altitude Inferred performance

(d) Attitude inferred performances

Fig. 13. Experience Inference results for Safety Critical Control

V. Conclusions

This paper reports a policy error inverse reinforcement learning algorithm for drone's objective inference using online data trajectories. The error and control input of the drone's trajectories are used to extract the hidden objective under an iterative mechanism inspired by the closed-loop input error technique. A set of basis functions is used to linearly parametrize the nonlinear control policy applied to the drone. Two off-line reinforcement learning/inverse integral reinforcement learning algorithms are used to generate new policies and objective weights until they converge to their respective real values. Convergence to the exact weight matrices is achieved under a batch-LS rule with an objective constraint. Convergence of the algorithm is verified using Lyapunov recursions. Simulations studies and comparisons using a quadcopter model are carried out to verify the stability, hyperparameter sensitivity, and robustness of the approach for different weight matrices. A safety critical control application is given to demonstrate the benefits of the PEIRL in experience inference tasks.

Further work will exploit the merits of the proposed technique to design control informed algorithms to enhance the robustness and stability of data-driven trajectory prediction algorithms. Different objective functions that consider constraints in the input and output signals in a H_2-control sense will be further analysed. Other future research vector consists in the objective extraction of robust control policies to incorporate aerodynamical effects at high velocities. Measurements obtained from non-cooperative sensors and/or communication issues pose a great challenge in safety and security sectors. This is also a concern for future work.

References

[1] J.-P. Yaacoub, H. Noura, O. Salman, and A. Chehab, "Security analysis of drones systems: Attacks, limitations, and recommendations," *Internet of Things*, vol. 11, p. 100218, 2020.

[2] Y. Zhou, K. G. Vamvoudakis, W. M. Haddad, and Z.-P. Jiang, "A secure control learning framework for cyber-physical systems under sensor and actuator attacks," *IEEE Transactions on Cybernetics*, vol. 51, no. 9, pp. 4648–4660, 2020.

[3] A. Tahir, J. Böling, M.-H. Haghbayan, H. T. Toivonen, and J. Plosila, "Swarms of unmanned aerial vehicles—a survey," *Journal of Industrial Information Integration*, vol. 16, p. 100106, 2019.

[4] A. Perrusquia, W. Guo, B. Fraser, and Z. Wei, "Uncovering drone intentions using control physics informed machine learning," *PREPRINT (Version 1) available at Research Square*, 2023.

[5] G. Lykou, D. Moustakas, and D. Gritzalis, "Defending airports from uas: A survey on cyber-attacks and counter-drone sensing technologies," *Sensors*, vol. 20, no. 12, p. 3537, 2020.

[6] I. Guvenc, F. Koohifar, S. Singh, M. L. Sichitiu, and D. Matolak, "Detection, tracking, and interdiction for amateur drones," *IEEE Communications Magazine*, vol. 56, no. 4, pp. 75–81, 2018.

[7] B. Taha and A. Shoufan, "Machine learning-based drone detection and classification: State-of-the-art in research," *IEEE access*, vol. 7, pp. 138 669–138 682, 2019.

[8] B. K. Kim, H.-S. Kang, and S.-O. Park, "Drone classification using convolutional neural networks with merged doppler images," *IEEE Geoscience and Remote Sensing Letters*, vol. 14, no. 1, pp. 38–42, 2016.

[9] H.-S. Shin, D. Turchi, S. He, and A. Tsourdos, "Behavior monitoring using learning techniques and regular-expressions-based pattern matching," *IEEE transactions on intelligent transportation systems*, vol. 20, no. 4, pp. 1289–1302, 2018.

[10] A. Perrusquía and W. Yu, "Human-behavior learning for infinite-horizon optimal tracking problems of robot manipulators," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 57–62.

[11] W. Sultani and M. Shah, "Human action recognition in drone videos using a few aerial training examples," *Computer Vision and Image Understanding*, vol. 206, p. 103186, 2021.

[12] H. Zhang, Y. Yan, S. Li, Y. Hu, and H. Liu, "Uav behavior-intention estimation method based on 4-d flight-trajectory prediction," *Sustainability*, vol. 13, no. 22, p. 12528, 2021.

[13] H. Ahn, H.-L. Choi, M. Kang, and S. Moon, "Learning-based anomaly detection and monitoring for swarm drone flights," *Applied Sciences*, vol. 9, no. 24, p. 5477, 2019.

[14] A. Perrusquía and W. Guo, "A closed-loop output error approach for physics-informed trajectory inference using online data," *IEEE Transactions on Cybernetics*, vol. 53, no. 3, pp. 1379–1391, 2022.

[15] F. M. Bianchi, S. Scardapane, S. Løkse, and R. Jenssen, "Reservoir computing approaches for representation and classification of multivariate time series," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 5, pp. 2169–2179, 2020.

[16] K. Saleh, M. Hossny, and S. Nahavandi, "Intent prediction of pedestrians via motion trajectories using stacked recurrent neural networks," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 4, pp. 414–424, 2018.

[17] A. Perrusquía and W. Yu, "Identification and optimal control of nonlinear systems using recurrent neural networks and reinforcement learning: An overview," *Neurocomputing*, vol. 438, pp. 145–154, 2021.

[18] C. Legaard, T. Schranz, G. Schweiger, J. Drgoňa, B. Falay, C. Gomes, A. Iosifidis, M. Abkar, and P. Larsen, "Constructing neural network based models for simulating dynamical systems," *ACM Computing Surveys*, vol. 55, no. 11, pp. 1–34, 2023.

[19] H. Oh, H.-S. Shin, S. Kim, A. Tsourdos, and B. A. White, "Airborne behaviour monitoring using gaussian processes with map information," *IET Radar, Sonar & Navigation*, vol. 7, no. 4, pp. 393–400, 2013.

[20] R. Rezaie and X. R. Li, "Gaussian conditionally markov sequences: Modeling and characterization," *Automatica*, vol. 131, p. 109780, 2021.

[21] J. Liang, B. I. Ahmad, M. Jahangir, and S. Godsill, "Detection of malicious intent in non-cooperative drone surveillance," in *2021 Sensor Signal Processing for Defence Conference (SSPD)*. IEEE, 2021, pp. 1–5.

[22] R. Rezaie and X. R. Li, "Destination-directed trajectory modeling, filtering, and prediction using conditionally markov sequences," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, no. 2, pp. 820–833, 2020.

[23] P. Becker, H. Pandya, G. Gebhardt, C. Zhao, C. J. Taylor, and G. Neumann, "Recurrent kalman networks: Factorized inference in high-dimensional deep feature spaces," in *International Conference on Machine Learning*. PMLR, 2019, pp. 544–552.

[24] G. Revach, N. Shlezinger, X. Ni, A. L. Escoriza, R. J. Van Sloun, and Y. C. Eldar, "Kalmannet: Neural network aided kalman filtering for partially known dynamics," *IEEE Transactions on Signal Processing*, vol. 70, pp. 1532–1547, 2022.

[25] C. Wang, H. Han, J. Wang, H. Yu, and D. Yang, "A robust extended kalman filter applied to ultrawideband positioning," *Mathematical Problems in Engineering*, vol. 2020, 2020.

[26] B. Fraser, A. Perrusquía, D. Panagiotakopoulos, and W. Guo, "Hybrid deep neural networks for drone high level intent classification using non-cooperative radar data," in *2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*. IEEE, 2023, pp. 1–6.

[27] W. Yu and A. Perrusquía, "Simplified stable admittance control using end-effector orientations," *International Journal of Social Robotics*, vol. 12, no. 5, pp. 1061–1073, 2020.

[28] A. Perrusquía, "Solution of the linear quadratic regulator problem of black box linear systems using reinforcement learning," *Information Sciences*, vol. 595, pp. 364–377, 2022.

[29] N. Ab Aza, A. Shahmansoorian, and M. Davoudi, "From inverse optimal control to inverse reinforcement learning: A historical review," *Annual Reviews in Control*, vol. 50, pp. 119–138, 2020.

[30] K. Vamvoudakis and F. L. Lewis, "On-line actor-critic algorithm to solve the continuous-time infinite horizon optimal control problem," *Automatica*, vol. 46, pp. 878–888, 2010.

[31] A. Perrusquía and W. Guo, "Cost inference of discrete-time linear quadratic control policies using human-behaviour learning," in *2022 8th International Conference on Control, Decision and Information Technologies (CoDIT)*, vol. 1. IEEE, 2022, pp. 165–170.

[32] H. El-Hussieny and J.-H. Ryu, "Inverse discounted-based lqr algorithm for learning human movement behaviors," *Applied Intelligence*, vol. 49, no. 4, pp. 1489–1501, 2019.

[33] W. Xue, B. Lian, J. Fan, P. Kolaric, T. Chai, and F. L. Lewis, "Inverse reinforcement q-learning through expert imitation for discrete-time systems," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[34] A. Y. Ng, S. Russell *et al.*, "Algorithms for inverse reinforcement learning." in *International Conference on Machine Learning*, vol. 1, 2000.

[35] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first International Conference on Machine learning*, 2004.

[36] S. Levine, Z. Popovic, and V. Koltun, "Nonlinear inverse reinforcement learning with gaussian processes," *Advances in neural information processing systems*, vol. 24, 2011.

[37] A. Perrusquía, "A complementary learning approach for expertise transference of human-optimized controllers," *Neural Networks*, vol. 145, pp. 33–41, 2022.

[38] M.-B. Radac and T. Lala, "A hierarchical primitive-based learning tracking framework for unknown observable systems based on a new state representation," in *2021 European Control Conference (ECC)*. IEEE, 2021, pp. 1472–1478.

[39] J. Ramírez, W. Yu, and A. Perrusquía, "Model-free reinforcement learning from expert demonstrations: a survey," *Artificial Intelligence Review*, vol. 55, no. 4, pp. 3213–3241, 2022.

[40] A. Perrusquía and W. Guo, "Optimal control of nonlinear systems using experience inference human-behavior learning," *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 1, pp. 90–102, 2023.

[41] A. Perrusquía, R. Garrido, and W. Yu, "Stable robot manipulator parameter identification: A closed-loop input error approach," *Automatica*, vol. 141, p. 110294, 2022.

[42] F. L. Lewis, D. Vrabie, and K. G. Vamvoudakis, "Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers," *IEEE Control Systems Magazine*, vol. 32, no. 6, pp. 76–105, 2012.

[43] A. Perrusquía and W. Guo, "Closed-loop output error approaches for drone's physics informed trajectory inference," *IEEE Transactions on Automatic Control*, 2023.

[44] B. Kiumarsi, K. G. Vamvoudakis, H. Modares, and F. L. Lewis, "Optimal and autonomous control using reinforcement learning: A survey," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 6, pp. 2042–2062, 2017.

[45] S. L. Brunton and J. N. Kutz, *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2022.

[46] Y. Yang, Z. Guo, H. Xiong, D.-W. Ding, Y. Yin, and D. C. Wunsch, "Data-driven robust control of discrete-time uncertain linear systems via off-policy reinforcement learning," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 12, pp. 3735–3747, 2019.

[47] A. Perrusquía and W. Yu, "Neural $\mathcal{H}_2$ control using continuous-time reinforcement learning," *IEEE Transactions on Cybernetics*, vol. 52, no. 6, pp. 4485–4494, 2022.

[48] S. A. A. Rizvi and Z. Lin, "Reinforcement learning-based linear quadratic regulation of continuous-time systems using dynamic output feedback," *IEEE Transactions on Cybernetics*, vol. 50, no. 11, pp. 4670–4679, 2019.

[49] A. Perrusquía, "Human-behavior learning: A new complementary learning perspective for optimal decision making controllers," *Neurocomputing*, vol. 489, pp. 157–166, 2022.

[50] H. Modares, F. L. Lewis, and M.-B. Naghibi-Sistani, "Integral reinforcement learning and experience replay for adaptive optimal control of partially-unknown constrained-input continuous-time systems," *Automatica*, vol. 50, no. 1, pp. 193–202, 2014.

[51] J. L. Proctor, S. L. Brunton, and J. N. Kutz, "Dynamic mode decomposition with control," *SIAM Journal on Applied Dynamical Systems*, vol. 15, no. 1, pp. 142–161, 2016.

[52] B. Lian, V. S. Donge, F. L. Lewis, T. Chai, and A. Davoudi, "Data-driven inverse reinforcement learning control for linear multiplayer games," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Adolfo Perrusquía received the B.Eng. degree in Mechatronic Engineering from the National Polytechnic Institute (UPIITA-IPN) in 2014, and the M.S. and Ph.D. degrees, both in Automatic Control from the Automatic Control Department at the CINVESTAV-IPN in 2016 and 2020, respectively. He is currently a lecturer at the School of Aerospace, Transport and Manufacturing, Cranfield University, and a former UK-IC Postdoctoral Research Fellow. He is a member of the IEEE Computational Intelligence Society. His main research of interest focuses on robotics, mechanisms, machine learning, reinforcement learning, nonlinear control, system modeling and system identification.

Weisi Guo (received the M.Eng. degree in engineering, and the M.A. and Ph.D. degrees in computer science from the University of Cambridge, Cambridge, U.K., in 2005, 2011, and 2011, respectively. He is a Chair Professor of Human Machine Intelligence with Cranfield University, Cranfield, U.K. He has published over 180 papers and is a PI on a number of molecular communication research grants.

Prof. Guo's research has won him several international awards. He was a Turing Fellow at the Alan Turing Institute.

2023-11-22

þÿ D r o n e s o b j e c t i v e i n f e r e n c e u
error inverse reinforcement learning

Perrusquía, Adolfo

IEEE

þÿ P e r r u s q u í a A , G u o W . (2 0 2 3) D r o n e s o b j e c t i v e i n f e r e n c e u s i n g p o l i c y e r
reinforcement learning. IEEE Transactions on Neural Networks and Learning Systems.
Available online November 2023
https://doi.org/10.1109/TNNLS.2023.3333551