Received September 2, 2020, accepted September 8, 2020, date of publication September 18, 2020, date of current version October 1, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3025194

A Multi-Critic Reinforcement Learning Method: An Application to Multi-Tank Water Systems

JUAN MARTINEZ-PIAZUELO[1], (Graduate Student Member, IEEE), DANIEL E. OCHOA[2], (Member, IEEE), NICANOR QUIJANO[1], (Senior Member, IEEE), AND LUIS FELIPE GIRALDO[1], (Member, IEEE)
[1]Departamento de Ingeniería Eléctrica y Electrónica, Universidad de los Andes, Bogotá 111711, Colombia
[2]Department of Electrical, Computer and Energy Engineering, University of Colorado Boulder, Boulder, CO 80309 USA

Corresponding author: Juan Martinez-Piazuelo (jp.martinez10@uniandes.edu.co)

This work was supported in part by the Research Group GIAP.

ABSTRACT This paper investigates the combination of reinforcement learning and neural networks applied to the data-driven control of dynamical systems. In particular, we propose a multi-critic actor-critic architecture that eases the value function learning task by distributing it into multiple neural networks. We also propose a filtered multi-critic approach that offers further performance improvements as it eases the training process of the control policy. All the studied methods are evaluated with several numerical experiments on multi-tank water systems with nonlinear coupled dynamics, where control is known to be a challenging task. The simulation results show that the proposed multi-critic scheme is able to outperform the standard actor-critic approach in terms of speed and sensitivity of the learning process. Moreover, the results show that the filtered multi-critic strategy outperforms the unfiltered one under these same terms. This document highlights the benefits of the multi-critic methodology on a state of the art reinforcement learning algorithm, the deep deterministic policy gradient, and demonstrates its application to multi-tank water systems relevant for industrial process control.

INDEX TERMS Data-driven control, approximate dynamic programming, reinforcement learning, actor-critic methods, deep deterministic policy gradient, water-tank systems.

I. INTRODUCTION

With the recent advances in computational technologies, reinforcement learning (RL) [1] and neural networks have attracted considerable attention as a promising combination for the data-driven control of dynamical systems. RL offers a family of methods that, through trial-and-error interactions with the dynamical system, permit the model-free tuning of parameterized controllers. The use of neural networks, on the other hand, offers a flexible platform to represent such parameterized controllers for a vast range of problems. Hence, the combination of RL and neural networks not only endows engineers with very powerful control techniques, but also alleviates the model-based dependency that characterizes most classic control methods.

In the control community, the field that studies such combination is known as approximate dynamic programming (ADP). In ADP, ideas from optimal control and function approximation are combined to achieve optimization-based controllers under the model-free framework [2]–[4]. The most common approach within the ADP family uses two distinct neural networks, an actor and a critic, that are simultaneously trained through a policy iteration procedure [1], [5]. In particular, the critic is trained to approximate the value function that solves a Hamilton-Jacobi-Bellman equation, and the actor is trained to approximate the optimal control policy coupled to the solution given by the critic [5]. In the RL literature these approaches are termed as actor-critic methods, and, therefore, we refer as such in the rest of this document.

Due to the increasing popularity of RL and neural network methods, several recent works have studied actor-critic approaches as model-free control strategies. For instance, [6] proposes an actor-critic method for nonaffine nonlinear systems and demonstrates its application on a torsional pendulum; [7] uses a decentralized stochastic actor-critic algorithm to avoid collisions within large populations of mobile robots; and [8] proposes a hierarchical actor-critic scheme to simultaneously learn basic and compound skills required

The associate editor coordinating the review of this manuscript and approving it for publication was Youqing Wang.

for robotic navigation. Furthermore, [9] extends actor-critic methods to the maximum entropy model-free control framework, and [10] illustrates its application to challenging control problems. Finally, with the aid of deep-learning, [11] uses recurrent neural networks as a way to achieve memory-based controllers, and [12] uses them to transfer the control policy from simulation to reality.

Although several extensions and modifications of actor-critic approaches have been proposed, a common difficulty that remains a challenge is the data inefficiency inherent to the trial-and-error nature of RL methods. Approaches based on hierarchical learning have demonstrated that RL agents can benefit from more structured learning schemes, and that breaking a complex task into smaller simpler ones can ease the training process [8], [13]. Inspired by this last idea, in this work we propose an actor-critic method that eases the learning task of the critic neural network by distributing it into multiple critics. Under this scheme, each of the multiple critics focuses only on a certain component of the value function, and, therefore, individually solves a problem that is potentially simpler than the full value function approximation. In that sense, note that the proposed approach is fundamentally different from asynchronous actor-critic methods, e.g., asynchronous advantage actor-critic [14], where the learning process is improved by means of simultaneously training multiple instances of the same neural networks all having the (common) full learning objective. To the best of our knowledge, similar approaches to ours have been recently proposed in [15], [16], and [17]. As in the present work, [15] proposes a linear decomposition of the value function as a way to ease its approximation. Unlike us, however, they apply it to control problems with discrete action spaces. On the other hand, the work of [16] proposes an actor-critic method that trains an agent to solve various independent tasks and uses multiple critics to achieve simultaneous training, and the work of [17] uses multiple critics to split the learning process into different stages and applies the method to the stochastic actor-critic approach of [10]. In contrast, in the present work we explore single-task multi-input multi-output (MIMO) deterministic control problems and propose a multi-critic architecture to extend the deterministic actor-critic approach of [18]. Here, each output of the control system has to satisfy a given control objective and the objectives of all outputs have to be completed concurrently. We use multiple critics to ease the value function approximation in a distributed manner, and, we show that such aggregate of multiple critics indeed approximates the original value function without changing the learning objective. Furthermore, our multi-critic scheme not only eases the critic's training process, but also offers more flexibility in the way that the actor can be trained. To demonstrate this, we propose an heuristic filtered multi-critic approach that, with the assumption of shallow prior knowledge about the system's dynamics, is able to ease the actor's training and improve the learning process in terms of speed and sensitivity to the change of parameters. Such a filtered multi-critic method can be regarded as an actor-critic-attention approach, where the attention mechanism is fixed and is set based on some shallow prior knowledge of the system. Regarding attention-augmented actor-critic methods, some recent related works are the ones in [19] and [20]. Namely, the authors in [19] propose an actor-attention-critic scheme for multi-agent reinforcement learning, where each agent has an actor-critic structure and a central attention mechanism is used to select the relevant information for each agent at every time step. Similarly, the authors in [20] propose an actor-critic-attention framework, where an actor-critic controller is extended with an attention mechanism to integrate multiple sensory data. In contrast with the previous works, in this paper the attention mechanism is used to filter the back-propagated gradients coming from the multiple critics, so that each output neuron of the actor network can specialize on certain component of the approximated value function. As mentioned above, this filtered multi-critic scheme outperforms the unfiltered one in various aspects.

Unlike most of the previous works in RL, where the methods have been applied to game-like or movement-oriented problems, in this work we explore the application of RL to industrial processes. More specifically, in this work we study the data-driven control of multi-tank water systems. These systems are not only relevant for industrial process control, but are also characterized by nonlinear coupled dynamics that difficult the design of analytic controllers under the model-free framework. The application of RL to the control of water-tank systems has also been studied in [21] and [22]. In particular, the authors in [21] explore the combination of conventional data-driven control method and Q-learning for the control of a vertical two-tank system, and the authors in [22] consider a hierarchical control scheme where the deep deterministic policy gradient (DDPG) algorithm of [18] is applied to control the water-flows of an open channel system. In contrast with the previous works, in this paper we study the application of the proposed multi-critic schemes to the DDPG algorithm, and through several numerical experiments we evaluate our approaches both on a quadruple-tank process [23] and a vertical two-tank system. Furthermore, it is worth to highlight that our multi-critic approaches are compatible with the methods proposed in [21] and [22]. Hence, the developments presented in this work could further extend the aforementioned researches.

In summary, the contributions of this paper are as follows:

1) *Unfiltered multi-critic method:* we propose a deterministic actor-critic method that distributes the value function learning problem into multiple critics as a way to ease the learning task.
2) *Filtered multi-critic method:* we further propose an heuristic to ease the actor's learning task as well. Through numerical experiments, we show that this approach offers benefits in terms of speed and sensitivity of the RL training process.
3) *Numerical validation:* we perform several numerical experiments on the proposed multi-tank water systems.

We apply the multi-critic scheme on the state-of-the-art DDPG algorithm and validate the multi-critic improvements in multiple scenarios with different conditions.

The rest of this paper is organized as follows. Section II introduces the foundations of the single-critic actor-critic method and the DDPG algorithm. In Section III we develop the unfiltered multi-critic approach as well as the filtered multi-critic heuristic. Then, in Section IV we present the multi-tank water systems to be studied and explain how to apply the actor-critic approaches to such systems. In Section V we present the numerical experiments that validate our contributions. Finally, Section VI concludes the paper.

II. ACTOR-CRITIC REINFORCEMENT LEARNING

In this section we introduce the main theory behind model-free RL and actor-critic methods. As is usual in the RL literature, we formulate the problem under the discrete-time Markov decision process framework.

Consider a time-invariant discrete-time system with state $\mathbf{x} = [x_1, \dots, x_{Nx}] \in \mathbb{R}^{Nx}$, with control input $\mathbf{u} = [u_1, \dots, u_{Nu}] \in \mathbb{R}^{Nu}$, and with unknown stochastic dynamics characterized by the transition distribution: $\mathbf{x}[k+1] \sim P\big(\mathbf{x}[k+1] \big| \mathbf{x}[k], \mathbf{u}[k]\big)$, for all k. Moreover, consider the discounted cost-to-go

$$G(k,T) = \sum_{i=k}^{T} \gamma^{i-k} C\big(\mathbf{x}[i], \mathbf{u}[i]\big), \quad (1)$$

where T is the time-horizon; the function $C : \mathbb{R}^{Nx} \times \mathbb{R}^{Nu} \to \mathbb{R}$ gives the cost of executing control action $\mathbf{u}[i]$ in state $\mathbf{x}[i]$ at time i; and $\gamma \in [0, 1)$ is a scalar discount factor that weights the importance of the cost $C(\cdot, \cdot)$ over time, and keeps $G(k, T)$ finite when $T = \infty$. Furthermore, consider the following assumptions.

Assumption 1 (Bounded State): There exists a finite $x_{max} \in \mathbb{R}$ such that $|x_i[k]| \leq x_{max}$, for all $i \in \{1, \dots, Nx\}$ and all k.

Assumption 2 (Bounded Input): There exists a finite $u_{max} \in \mathbb{R}$ such that $|u_i[k]| \leq u_{max}$, for all $i \in \{1, \dots, Nu\}$ and all k.

Assumption 3 (Bounded Cost): The cost $C(\mathbf{x}[k], \mathbf{u}[k])$ is finite for all k.

Remark 1: Notice that Assumption 1 is satisfied by any system with saturated states; Assumption 2 is satisfied if the output of the controller is saturated; and, finally, Assumption 3 can be satisfied with an appropriate design of the cost function $C(\cdot, \cdot)$.

Following, under this framework the goal is to find a feedback control policy that minimizes, in expectation, the discounted cost-to-go $G(k, T)$. Namely, a feedback control law $\mu : \mathbb{R}^{Nx} \to \mathbb{R}^{Nu}$ such that

$$\mu\big(\mathbf{x}[k]\big) = \arg\min_{\mathbf{u}[k]} \mathbb{E}_{\mu}\left[G(k,T) \big| \mathbf{x}[k], \mathbf{u}[k]\right], \quad \forall k, \quad (2)$$

where $\mathbb{E}_{\mu}[\cdot | \mathbf{x}[k], \mathbf{u}[k]]$ denotes the expectation over the closed-loop dynamics: $P\big(\mathbf{x}[k+1] \big| \mathbf{x}[k], \mu\big(\mathbf{x}[k]\big)\big)$, and is conditioned on the current state-action pair: $\mathbf{x}[k], \mathbf{u}[k]$. In this work, $\mu(\cdot)$ is represented by a Θ-parameterized neural network $\mu_{\Theta} : \mathbb{R}^{Nx} \to \mathbb{R}^{Nu}$, denoted as the actor, and is trained to approximate the minimizer of the expectation in (2). However, given that the dynamics of the system are assumed unknown, this expectation cannot be explicitly evaluated, and, instead, it has to be estimated from input-output data of the closed-loop system. For such, we define an additional Φ-parameterized neural network $Q^{\mu}_{\Phi} : \mathbb{R}^{Nx} \times \mathbb{R}^{Nu} \to \mathbb{R}$, which, for all times k, seeks to approximate the following Bellman equation:

$$\begin{aligned} Q^{\mu}_{\Phi}\big(\mathbf{x}[k], \mathbf{u}[k]\big) &= \mathbb{E}_{\mu}\left[G(k,T) \big| \mathbf{x}[k], \mathbf{u}[k]\right] \\ &= C\big(\mathbf{x}[k], \mathbf{u}[k]\big) + \gamma Q^{\mu}_{\Phi}\big(k+1\big), \end{aligned} \quad (3)$$

where $Q^{\mu}_{\Phi}\big(k+1\big) \triangleq Q^{\mu}_{\Phi}\big(\mathbf{x}[k+1], \mu_{\Theta}(\mathbf{x}[k+1])\big)$. This additional network is denoted as the critic and is iteratively trained to minimize the squared Bellman error (SBE):

$$\mathrm{SBE} = \frac{1}{2}\Big(y - Q^{\mu}_{\Phi}\big(\mathbf{x}[k], \mathbf{u}[k]\big)\Big)^2, \quad (4)$$

where the target y is set to be

$$y = C\big(\mathbf{x}[k], \mathbf{u}[k]\big) + \gamma \hat{Q}^{\mu}_{\hat{\Phi}}\big(\mathbf{x}[k+1], \hat{\mu}_{\hat{\Theta}}(\mathbf{x}[k+1])\big).$$

The target networks $\hat{\mu}_{\hat{\Theta}}(\cdot)$ and $\hat{Q}^{\mu}_{\hat{\Phi}}(\cdot, \cdot)$, with respective parameters $\hat{\Theta}$ and $\hat{\Phi}$, have the exact same structure as the actor and critic networks and are used to improve the stability of the learning process. Note that with the aid of the target networks, the target value y is decoupled from the parameters that we are learning, i.e., Θ and Φ. Hence, by delaying the updates of the target networks, the target value y does not change immediately with the updates of the main parameters Θ and Φ. Consequently, the use of target networks slows down the change of the target value y and improves the learning performance in practice. This heuristic approach was first proposed in [24] and has been widely accepted in the RL community. As in [18], in this work the parameters $\hat{\Theta}$ and $\hat{\Phi}$ are slowly copied from the main parameters Θ and Φ through soft updates of the form

$$\hat{\Theta} = \tau\Theta + (1-\tau)\hat{\Theta}, \quad \hat{\Phi} = \tau\Phi + (1-\tau)\hat{\Phi}, \quad (5)$$

where $\tau \in (0, 1]$ denotes the targets' learning rate. Once the expected cost under the current policy $\mu_{\Theta}(\cdot)$ is estimated, the policy can be updated to minimize such cost, and, in consequence, to become a better policy. For such, the parameters of the actor network are updated in the decreasing direction of the action-gradient that is back-propagated from the critic network [18], [25]. For time k that is:

$$\nabla\Theta \propto \nabla_{\Theta}\mu_{\Theta}\big(\mathbf{x}[k]\big) \nabla_{\mathbf{u}} Q^{\mu}_{\Phi}\big(\mathbf{x}[k], \mathbf{u}[k]\big)\Big|_{\mathbf{u}[k]=\mu_{\Theta}(\mathbf{x}[k])}. \quad (6)$$

Here, $\nabla\Theta \in \mathbb{R}^{N_{\Theta}}$ contains the gradients to be applied to the parameters of the actor network (N_{Θ} is the number of parameters of the actor network); the matrix $\nabla_{\Theta}\mu_{\Theta}(\mathbf{x}[k]) \in \mathbb{R}^{N_{\Theta} \times Nu}$ contains the gradients of each output of the actor network with respect to the actor's parameters; and

$\nabla_{\mathbf{u}} Q^{\mu}_{\Phi}(\mathbf{x}[k], \mathbf{u}[k]) \in \mathbb{R}^{Nu}$ contains the gradients of the critic network with respect to the control input $\mathbf{u}[k] = \mu_{\Theta}(\mathbf{x}[k])$.

The critic and actor networks described above are iteratively updated to evaluate and improve the control policy. This algorithm is known as deep deterministic policy gradient (DDPG) [18], and can be seen as a form of generalized policy iteration (GPI) [1], [5]. For the sake of clarity, Algorithm 1 summarizes the way that we use it in this work. Notice that the input-output data experiences that are obtained through the interaction with the system are saved in a memory buffer. Such memory buffer allows the algorithm to resample past experiences in the training process of the actor and critic networks, and, in consequence, to reduce the temporal correlation of the training data [24].

Algorithm 1 GPI for DDPG-Like Actor-Critic Methods

```
1:  Initialization: start with an arbitrary initial parameterized control policy $\mu_{\Theta}(\cdot)$, and with an arbitrary initial parameterized value function $Q^{\mu}_{\Phi}(\cdot,\cdot)$ for the actor and the critic, respectively. Define update frequencies $f_{\mu}$ and $f_Q$ for the actor and critic, respectively.
2:  Beginning of training episode: set $k = 0$ and reset the initial state of the system.
3:  while $k \leq$ training steps do:
4:      Execute action $\mathbf{u}[k] = \mu_{\Theta}(\mathbf{x}[k])$ + exploration noise.
5:      Store in memory: $\mathbf{x}[k]$, $\mathbf{u}[k]$, $\mathbf{x}[k+1]$, $C(\mathbf{x}[k], \mathbf{u}[k])$.
6:      if mod $(k, f_Q) = 0$ then:
7:          Sample a batch $\mathcal{B}$ of tuples from the memory buffer.
8:          Perform one step update of the critic to minimize (4) on $\mathcal{B}$.
9:          Perform one step update of all the target networks using (5).
10:     end if
11:     if mod $(k, f_{\mu}) = 0$ then:
12:         Sample a batch $\mathcal{B}$ of tuples from the memory buffer.
13:         Perform one step update of the actor using (6) averaged over $\mathcal{B}$.
14:     end if
15:     $k \leftarrow k + 1$
16: end while
17: If maximum number of training episodes has been achieved, end. Else, return to step 2.
```

III. THE MULTI-CRITIC ACTOR-CRITIC ARCHITECTURE

In the previous section we introduced the main theory behind actor-critic methods through the RL approach. Now, we introduce the unfiltered and filtered multi-critic actor-critic architectures that depict our main contributions.

A. THE UNFILTERED MULTI-CRITIC APPROACH

Consider the case where the cost $C(\cdot,\cdot)$ is linearly separable into $Np > 1$ cost-partitions. More precisely, consider a cost $C(\cdot,\cdot)$ such that:

$$C\big(\mathbf{x}[k], \mathbf{u}[k]\big) = \sum_{p=1}^{Np} C_p\big(\mathbf{x}[k], \mathbf{u}[k]\big), \quad \forall k, \tag{7}$$

where $C_p : \mathbb{R}^{Nx} \times \mathbb{R}^{Nu} \rightarrow \mathbb{R}$ for all $p \in \mathcal{P}$, with $\mathcal{P} = \{1, \ldots, Np\}$. For instance, some cost-partitions might consider the error of certain components of the state with respect to some given reference signals, and other partitions might penalize large switches in the components of the control input. In any case, the fact that the cost $C(\cdot,\cdot)$ can be decomposed into Np partitions, as in (7), allows the decomposition of the critic network $Q^{\mu}_{\Phi}(\cdot,\cdot)$ into Np critic networks where each critic focus only on one cost-partition. The following proposition illustrates this result.

Proposition 1: Let $C(\cdot,\cdot)$ be a cost function that is linearly separable into Np partitions as in (7). Moreover, let $Q^{\mu}_{\Phi}(\cdot,\cdot)$ be a neural network approximating the expectation of the discounted cost-to-go of (1) as in (3). Then, the network $Q^{\mu}_{\Phi}(\cdot,\cdot)$ can be decomposed into Np networks as follows:

$$Q^{\mu}_{\Phi}\big(\mathbf{x}[k], \mathbf{u}[k]\big) = \sum_{p=1}^{Np} Q^{\mu}_{\Phi p}\big(\mathbf{x}[k], \mathbf{u}[k]\big), \quad \forall k, \tag{8}$$

where $Q^{\mu}_{\Phi p} : \mathbb{R}^{Nx} \times \mathbb{R}^{Nu} \rightarrow \mathbb{R}$ for all $p \in \mathcal{P}$.

Proof: Using the cost-partitions defined in (7), the discounted cost-to-go (1) can be decomposed into Np partitions:

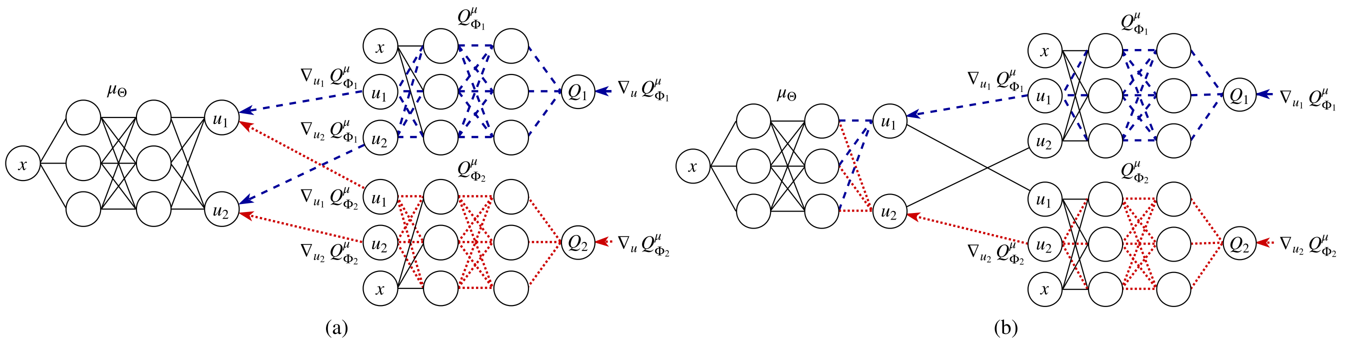
$$\begin{aligned} G(k,T) &= \sum_{i=k}^{T} \gamma^{i-k} C\big(\mathbf{x}[i], \mathbf{u}[i]\big) \\ &= \sum_{i=k}^{T} \gamma^{i-k} \sum_{p=1}^{Np} C_p\big(\mathbf{x}[i], \mathbf{u}[i]\big) \\ &= \sum_{p=1}^{Np} \sum_{i=k}^{T} \gamma^{i-k} C_p\big(\mathbf{x}[i], \mathbf{u}[i]\big) \\ &= \sum_{p=1}^{Np} G_p(k,T), \end{aligned}$$

where $G_p(k,T) = \sum_{i=k}^{T} \gamma^{i-k} C_p\big(\mathbf{x}[i], \mathbf{u}[i]\big)$. If we apply this partitioned cost-to-go in the first line of (3) and use the linearity of expectation, then, for all times k, the network $Q^{\mu}_{\Phi}(\cdot,\cdot)$ can be decomposed as

$$\begin{aligned} Q^{\mu}_{\Phi}\big(\mathbf{x}[k], \mathbf{u}[k]\big) &= \mathbb{E}_{\mu}\Big[G(k,T)\Big|\, \mathbf{x}[k], \mathbf{u}[k]\Big] \\ &= \mathbb{E}_{\mu}\left[\sum_{p=1}^{Np} G_p(k,T)\,\middle|\, \mathbf{x}[k], \mathbf{u}[k]\right] \\ &= \sum_{p=1}^{Np} \mathbb{E}_{\mu}\Big[G_p(k,T)\Big|\, \mathbf{x}[k], \mathbf{u}[k]\Big] \\ &= \sum_{p=1}^{Np} Q^{\mu}_{\Phi p}\big(\mathbf{x}[k], \mathbf{u}[k]\big). \end{aligned}$$

Since the equality holds exactly, the sum of the Np networks approximates the same expectation as the network $Q^{\mu}_{\Phi}(\cdot,\cdot)$. Thus, the overall learning objective is unchanged. This completes the proof. □

The previous result means that the expectation $\mathbb{E}_{\mu}\big[G(k,T)\big|\, \mathbf{x}[k], \mathbf{u}[k]\big]$ can be equally approximated by the



μ_Θ
x
u_1
u_2
$\nabla_{u_1} Q^\mu_{\Phi_1}$
$\nabla_{u_2} Q^\mu_{\Phi_1}$
$\nabla_{u_1} Q^\mu_{\Phi_2}$
$\nabla_{u_2} Q^\mu_{\Phi_2}$
$Q^\mu_{\Phi_1}$
$Q^\mu_{\Phi_2}$
Q_1
Q_2
$\nabla_u Q^\mu_{\Phi_1}$
$\nabla_u Q^\mu_{\Phi_2}$
(a)
(b)

FIGURE 1. Representation of how the gradients are back-propagated into the actor network in an illustrative two-critic multi-critic architecture: (a) unfiltered multi-critic, (b) filtered multi-critic. The blue (dashed) and red (dotted) lines represent network's connections with a flowing gradient that is exclusively associated to one of the critics. The black (solid) lines represent connections where gradients are not flowing or where the multi-critics' gradients are mixed.

single critic $Q^\mu_\Phi(\cdot,\cdot)$ or by the sum of Np critics $Q^\mu_{\Phi p}(\cdot,\cdot)$. The advantage of the later case, however, is that each of the multiple critics focus only on some component of the aforementioned expectation, i.e., $Q^\mu_{\Phi p}\big(\mathbf{x}[k],\mathbf{u}[k]\big) = \mathbb{E}_\mu\left[G_p(k,T)\big|\,\mathbf{x}[k],\mathbf{u}[k]\right]$, and, in consequence, the learning task of each of the Np critics is potentially simpler than the learning task of the overall critic network $Q^\mu_\Phi\big(\mathbf{x}[k],\mathbf{u}[k]\big) = \mathbb{E}_\mu\left[G(k,T)\big|\,\mathbf{x}[k],\mathbf{u}[k]\right]$. Thus, our multi-critic partition can be thought as a divide-and-conquer approach where the overall learning task is broken down into smaller problems, each to be solved by an independent dedicated neural network. Under this scheme, the p-th critic is trained to minimize the p-th squared Bellman error (p-SBE):

$$\text{p-SBE} = \frac{1}{2}\Big(y_p - Q^\mu_{\Phi p}\big(\mathbf{x}[k],\mathbf{u}[k]\big)\Big)^2, \tag{9}$$

where the target y_p is set to be

$$y_p = C_p\big(\mathbf{x}[k],\mathbf{u}[k]\big) + \gamma \hat{Q}^\mu_{\hat{\Phi}p}\big(\mathbf{x}[k+1], \hat{\mu}_{\hat{\Theta}}(\mathbf{x}[k+1])\big).$$

On the other hand, the actor network is trained using the same updates of (6), but adding the back-propagated gradients that come from the different critics. More precisely, for all k the parameters of the actor network are updated in the decreasing direction of the following gradient:

$$\nabla\Theta \propto \nabla_\Theta \mu_\Theta\big(\mathbf{x}[k]\big) \sum_{p=1}^{Np} \nabla_\mathbf{u} Q^\mu_{\Phi p}\big(\mathbf{x}[k],\mathbf{u}[k]\big)\Big|_{\mathbf{u}[k]=\mu_\Theta(\mathbf{x}[k])}, \tag{10}$$

with $\nabla\Theta \in \mathbb{R}^{N_\Theta}$; $\nabla_\Theta\mu_\Theta(\mathbf{x}[k]) \in \mathbb{R}^{N_\Theta\times Nu}$; and $\nabla_\mathbf{u} Q^\mu_{\Phi p}(\mathbf{x}[k],\mathbf{u}[k]) \in \mathbb{R}^{Nu}$, for all $p \in \mathcal{P}$. Furthermore, as before, the target networks are updated as in (5). As illustration, Fig. 1a shows an example of the described multi-critic architecture for the case when $Nu = Np = 2$. As we show in the results of Section V, the introduced multi-critic method is able to outperform the single-critic approach of Section II in terms of speed and sensitivity of the RL training process.

Remark 2: Notice that to select the number of critics, N_p, one should consider several factors. On one hand, the cost function $C(\cdot,\cdot)$ must have at least N_p linearly separable terms, so that $C(\cdot,\cdot)$ can be expressed as in (7). Hence, the number N_p depends on the considered cost function, which in turn depends on the controlled system as well as the control objective. Moreover, the number of critics N_p provides a trade-off between the simplicity of the cost-partitions and the number of neural networks that must be designed and trained. Namely, a bigger N_p leads to potentially simpler cost-partitions, but at the expense of more critic neural networks. Clearly, this is an important issue to consider when dealing with limited memory and processing resources.

Remark 3: It is worth mentioning that the recent work of [26] has proposed an algorithm called multi-critic DDPG. Nonetheless, the multiple critics in [26] are used as a form of redundant boosting where several estimations of the same value function are made (i.e., without any value decomposition). Although our multi-critic DDPG is given the same name as the one in [26], the two approaches are fundamentally different and should not be confused. We highlight this to avoid confusion between the acronyms used in the results of our work and the ones used in [26].

In the following section, we propose an additional heuristic modification to the multi-critic architecture that can lead to further improvements regarding its training performance.

B. THE FILTERED MULTI-CRITIC APPROACH

The multi-critic scheme presented in the previous section takes advantage of the linearly separable nature of the cost $C(\cdot,\cdot)$, and uses a distinct neural network to approximate the expected cost-to-go of each cost-partition $C_p(\cdot,\cdot)$. More precisely, the p-th critic seeks to approximate $\mathbb{E}_\mu\left[G_p(k,T)\big|\,\mathbf{x}[k],\mathbf{u}[k]\right]$, where $G_p(\cdot,\cdot)$ is defined as in (1) but with $C_p(\cdot,\cdot)$ instead of $C(\cdot,\cdot)$. Note that although the training process of the multiple critics is decoupled under this scheme, the back-propagated gradients that flow into each output neuron of the actor network consider all of the Np critics (see (10) and Fig. 1a). In this section we explore whether the training performance of the actor network can be further improved by filtering such back-propagated gradients.

By filtering we mean that some components of the back-propagated gradients that come from the multiple critics are ignored. To explore this, we modify the updates of the parameters of the actor network so that:

$$\nabla\Theta \propto \nabla_{\Theta}\mu_{\Theta}\big(\mathbf{x}[k]\big)\sum_{p=1}^{Np}\mathbf{F}_p\nabla_{\mathbf{u}}Q^{\mu}_{\Phi p}\big(\mathbf{x}[k],\mathbf{u}^*\big), \quad (11)$$

where all the shapes are the same as in (10); $\mathbf{u}^* \triangleq \mu_{\Theta}(\mathbf{x}[k])$; and $\mathbf{F}_p \in \mathbb{R}^{Nu\times Nu}$ is a diagonal matrix where the (i,i)-th element is 1 if $\partial Q^{\mu}_{\Phi p}(\mathbf{x}[k],\mathbf{u}[k])/\partial u_i \in \mathbb{R}$ is to be back-propagated to the actor network, and where the (i,i)-th element is 0 if $\partial Q^{\mu}_{\Phi p}(\mathbf{x}[k],\mathbf{u}[k])/\partial u_i$ is to be ignored.

Remark 4: Notice that if $\mathbf{F}_p$ is set as the $Nu \times Nu$ identity matrix for all $p \in \mathcal{P}$, then the update in (11) becomes equal to the one in (10).

The motivation behind this filtered multi-critic approach is that the matrices $\mathbf{F}_p$ can be designed so that each output neuron of the actor network is trained considering only certain cost-partitions. This could be desirable in the case where the different cost-partitions are dominated by distinct components of the control input. For instance, consider a deterministic nonlinear discrete-time system: $\mathbf{x}[k+1] = f(\mathbf{x}[k],\mathbf{u}[k])$, where $f : \mathbb{R}^{Nx}\times\mathbb{R}^{Nu} \to \mathbb{R}^{Nx}$ is a deterministic function that satisfies Assumption 1. Moreover, suppose that the j-th cost-partition is given by: $C_j(\mathbf{x}[k],\mathbf{u}[k]) = |r_i - x_i[k+1]|$, for all k, where $r_i \in \mathbb{R}$ is a reference signal to be achieved by $x_i[\cdot]$. If $f(\cdot)$ is such that $x_i[k+1]$ is mostly determined by the d-th control input: $u_d[\cdot]$, then the aforementioned cost-partition $C_j(\cdot,\cdot)$ is mostly dominated by $u_d[k]$, for all k. In consequence, it might be convenient to set the (d,d)-th term in $\mathbf{F}_j$ equal to one, and all other diagonal elements of $\mathbf{F}_j$ equal to 0. By doing so, only $\partial Q^{\mu}_{\Phi j}(\mathbf{x}[k],\mathbf{u}[k])/\partial u_d$ is back-propagated to the actor network from the j-th critic, and only the d-th output neuron of the actor receives such gradient. Given that the variance of the sum of random variables is equal to the sum of variances plus the corresponding co-variances, by weighting some multi-critics' gradients with zero, the overall variance of the summation in (11) is reduced. Clearly, such variance reduction leads to a bias increment regarding the actor's minimization objective in (2). Nevertheless, the results of Section (V) show that despite this additional bias, the filtered multi-critic approach is able to outperform the unfiltered one in terms of speed and sensitivity of the training process.

Remark 5: It is worth highlighting that the design of the filtering matrices $\mathbf{F}_p$ requires some knowledge of the system's dynamics. For instance, in the previous example we assumed that $f(\cdot)$ was such that $x_i[k+1]$ was mostly determined by $u_d[k]$ for all k. Therefore, the filtered multi-critic approach is not fully model-free as the unfiltered one. Nevertheless, note that the elements of the matrices $\mathbf{F}_p$ are either 0 or 1, and, in consequence, although some sense of dominance of the input-cost couplings is required, the exact values of such couplings are not needed. Thus, the required knowledge about the system's dynamics is relatively shallow.

For the sake of clarity, Algorithm 2 summarizes the implementation of the proposed multi-critic approaches, and Fig. 1b depicts an example of the filtered multi-critic scheme with $Np = Nu = 2$ and with filtering matrices given by:

$$\mathbf{F}_1 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad \mathbf{F}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}.$$

Remark 6: Note in Fig. 1b that the proposed gradient-filtering produces a decoupling of gradients in the output layer of the actor network. This leads to the specialization of each of the actor's output neurons on the minimization of a given critic, i.e., each actor's output neuron is trained to minimize a distinct cost-partition.

Algorithm 2 GPI for DDPG-Like Multi-Critic Actor-Critic Methods

```
1: **Initialization:** start with an arbitrary initial parameterized control policy $\mu_\Theta(\cdot)$, and with $Np$ arbitrary initial parameterized value functions $Q^\mu_{\Phi p}(\cdot,\cdot)$ for the actor and the multiple critics, respectively. Define update frequencies $f_\mu$ and $f_Q$ for the actor and critics, respectively.
2: **Beginning of training episode:** set $k = 0$ and reset the initial state of the system.
3: **while** $k \leq$ training steps **do:**
4:     Execute action $\mathbf{u}[k] = \mu_\Theta\big(\mathbf{x}[k]\big)$ + exploration noise.
5:     Store in memory: $\mathbf{x}[k]$, $\mathbf{u}[k]$, $\mathbf{x}[k+1]$, $C_p\big(\mathbf{x}[k],\mathbf{u}[k]\big)$ for all $p \in \mathcal{P}$.
6:     **if** mod $(k, f_Q) = 0$ **then:**
7:         Sample a batch $\mathcal{B}$ of tuples from the memory buffer.
8:         Perform one step update of each critic to minimize (9) on $\mathcal{B}$.
9:         Perform one step update of all the target networks using (5).
10:    **end if**
11:    **if** mod $(k, f_\mu) = 0$ **then:**
12:        Sample a batch $\mathcal{B}$ of tuples from the memory buffer.
13:        Perform one step update of the actor using (11) averaged over $\mathcal{B}$.
14:    **end if**
15:    $k \leftarrow k + 1$
16: **end while**
17: If maximum number of training episodes has been achieved, end. Else, return to step 2.
```

IV. APPLICATION TO MULTI-TANK WATER SYSTEMS

In this work we explore the data-driven control of multi-tank water systems as a testbed for our multi-critic approaches. These systems, which are relevant for industrial process control, are an attractive benchmark for the multi-critic architecture as they have multiple inputs and multiple outputs that are usually characterized by nonlinear coupled dynamics. In particular, we study the data-driven control of the quadruple-tank process benchmark of [23], and of a vertical two-tank system inspired by the parameters of [23]. Both systems are illustrated in Fig. 2 and in both cases the goal is to control the water-level heights of tanks 1 and 2 according to some given reference signals $r_1[k]$ and $r_2[k]$, respectively. We study the quadruple tank process as it allows a straightforward variable adjustment in the strength of the different input-state couplings, and, therefore, in the strength of dominance between the different input-cost couplings. On the other hand, we study the vertical two-tank system to further illustrate the advantages of our multi-critic approach when the switches of the control input are also penalized.

Following, the dynamics of the quadruple-tank process are given by

$$\begin{aligned}
x_1[k+1] &= x_1[k] + \frac{1}{A_1}\left(-q_1[k] + q_3[k] + \gamma_1 k_1 u_1[k]\right), \\
x_2[k+1] &= x_2[k] + \frac{1}{A_2}\left(-q_2[k] + q_4[k] + \gamma_2 k_2 u_2[k]\right), \\
x_3[k+1] &= x_3[k] + \frac{1}{A_3}\left(-q_3[k] + (1-\gamma_2) k_2 u_2[k]\right), \\
x_4[k+1] &= x_4[k] + \frac{1}{A_4}\left(-q_4[k] + (1-\gamma_1) k_1 u_1[k]\right),
\end{aligned}$$

while the dynamics of the vertical two-tank system are described by

$$\begin{aligned}
x_1[k+1] &= x_1[k] + \frac{1}{A_1}\left(-q_1[k] + k_1 u_1[k]\right), \\
x_2[k+1] &= x_2[k] + \frac{1}{A_2}\left(-q_2[k] + q_1[k] + k_2 u_2[k]\right).
\end{aligned}$$

In the previous equations, $x_i[k]$ is the water level height of the tank i at time k; A_i is the cross-section area of tank i; and $q_i[k] = a_i\sqrt{2gx_i[k]}$ for all i, where a_i is the cross-section area of the outlet hole of tank i, and g denotes the acceleration of gravity. The parameters k_1 and k_2 determine the flow of water at the pumps given the input voltages $u_1[k]$ and $u_2[k]$, respectively. For the quadruple-tank process, the parameters $\gamma_1 \in [0, 1]$ and $\gamma_2 \in [0, 1]$ establish how the water flows are split at the valves and determine the strength of the different input-output couplings. Thus, for the experiments on this system we evaluate different values of γ_1 and γ_2, but we select such values so that $\gamma_i > 0.5$ for $i = 1, 2$. By doing so we can deduce that $x_1[k+1]$ is more influenced by $u_1[k]$ than by $u_2[k]$, and, similarly, that $x_2[k+1]$ is more influenced by $u_2[k]$ than by $u_1[k]$. Moreover, even though the objective is to control only $x_1[k]$ and $x_2[k]$, in both cases we assume that the heights of all tanks are measurable. The parameters used for the simulation of both systems were inspired by the benchmark of [23] and are summarized in Table 1 to stay self-contained.

A. REINFORCEMENT LEARNING ON THE QUADRUPLE-TANK PROCESS

To apply the RL approaches of Sections II and III on the quadruple-tank process, we have to specify the cost $C(\cdot,\cdot)$ to be used during the training of the actor and the critic networks. Given that for this system the objective is to control the $x_1[k]$ and $x_2[k]$ to track some reference signals $r_1[k]$ and $r_2[k]$, respectively, we can set the overall cost $C(\cdot,\cdot)$ as:

$$\begin{aligned}
C(\mathbf{x}[k], \mathbf{u}[k]) = \alpha_1 \left| r_1[k+1] - x_1[k+1] \right| \\
+ \alpha_2 \left| r_2[k+1] - x_2[k+1] \right|,
\end{aligned}$$

where α_1 and α_2 are some non-negative scalars that weight each term in the cost. Here we can recognize that $C(\cdot,\cdot)$ is linearly separable into two cost-partitions. Namely:

$$\begin{aligned}
C_1(\mathbf{x}[k], \mathbf{u}[k]) &= \alpha_1 \left| r_1[k+1] + x_1[k+1] \right|, \\
C_2(\mathbf{x}[k], \mathbf{u}[k]) &= \alpha_2 \left| r_2[k+1] + x_2[k+1] \right|.
\end{aligned}$$

With the information so far we can apply both the single-critic method of Section II and the unfiltered multi-critic method of Section III-A to the quadruple-tank process. To apply the filtered multi-critic method of Section III-B, on the other hand, we need to know which input component has the greatest influence on each cost partition. Given that $\gamma_i > 0.5$ for $i = 1, 2$, we have that $u_1[k]$ has the greatest influence over $x_1[k+1]$, and that $u_2[k]$ has the greatest influence over $x_2[k+1]$. If we assume that we know this information, then we can deduce that $u_1[k]$ has more influence over $C_1(\mathbf{x}[k], \mathbf{u}[k])$ than $u_2[k]$, and that $u_2[k]$ has more influence over $C_2(\mathbf{x}[k], \mathbf{u}[k])$ than $u_1[k]$. Therefore, we can design the filtering matrices as:

$$\mathbf{F}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad \mathbf{F}_2 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

TABLE 1. Parameters of the water-tank systems.

Quadruple-tank process			
Parameter	Value	Parameter	Value
$A_1,\ A_3\ [cm^2]$	28	$k_1\ [cm^3/Vs]$	3.33
$A_2,\ A_4\ [cm^2]$	32	$k_2\ [cm^3/Vs]$	3.35
$a_1,\ a_3\ [cm^2]$	0.071	$g\ [cm/s^2]$	981
$a_2,\ a_4\ [cm^2]$	0.057	$\gamma_1,\ \gamma_2\ [-]$	*variable*
$x_{max}\ [cm]$	20	$u_{max}\ [V]$	10
$x_{min} [cm]$	0	$u_{min}\ [V]$	0
Vertical two-tank system			
Parameter	Value	Parameter	Value
$A_1,\ A_2\ [cm^2]$	32	$k_1,\ k_2\ [cm^3/Vs]$	1.66, 3.35
$a_1,\ a_2\ [cm^2]$	0.071	$g\ [cm/s^2]$	981
$x_{1max},\ x_{2max}\ [cm]$	10, 20	$u_{max}\ [V]$	10
$x_{min} [cm]$	0	$u_{min}\ [V]$	0

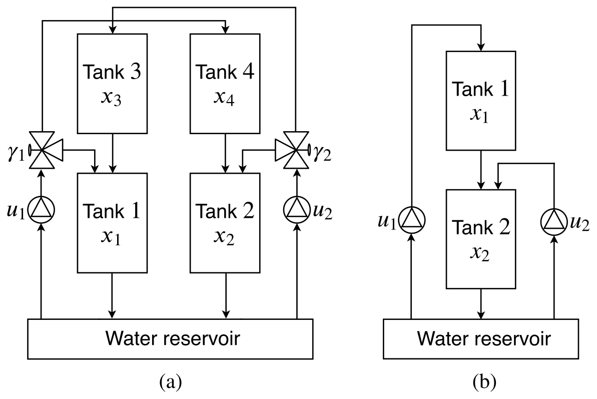


FIGURE 2. Water-tank systems. (a) Quadruple-tank process. (b) Vertical two-tank system.

It is worth mentioning that to track the desired reference signals $r_1[k]$ and $r_2[k]$, the state of the system has to be extended with information regarding the tracking error. More precisely, for the quadruple tank process we add two components to $\mathbf{x}[k]$ given by $x_5[k] = r_1[k] - x_1[k]$, and $x_6[k] = r_2[k] - x_2[k]$. Thus, for this system we have that $Nx = 6$, $Nu = 2$, and $Np = 2$.

B. REINFORCEMENT LEARNING ON THE VERTICAL TWO-TANK SYSTEM

As with the quadruple-tank process, in this system we also seek to track some reference signals $r_1[k]$ and $r_2[k]$ in $x_1[k]$ and $x_2[k]$, respectively. However, in addition to that, we also explore the penalization on switches of the control input.[1] For that, we set the cost $C(\cdot, \cdot)$ as

$$\begin{aligned} C(\mathbf{x}[k], \mathbf{u}[k]) = \alpha_1 &\left| r_1[k+1] - x_1[k+1] \right| \\ + \alpha_2 &\left| r_2[k+1] - x_2[k+1] \right| \\ + \alpha_3 &\left| u_1[k] - u_1[k-1] \right| \\ + \alpha_4 &\left| u_2[k] - u_2[k-1] \right| . \end{aligned}$$

As before, α_i is a non-negative scalar to weight the i-th term of the cost. From such cost definition we can identify several possible cost-partitions. In particular, in this work we consider $Np = 2$ and set the partitions as follows:

$$\begin{aligned} C_1(\mathbf{x}[k], \mathbf{u}[k]) &= \alpha_1 \left| r_1[k+1] - x_1[k+1] \right| \\ &\quad + \alpha_3 \left| u_1[k] - u_1[k-1] \right| , \\ C_2(\mathbf{x}[k], \mathbf{u}[k]) &= \alpha_2 \left| r_2[k+1] - x_2[k+1] \right| \\ &\quad + \alpha_4 \left| u_2[k] - u_2[k-1] \right| . \end{aligned}$$

Furthermore, from the structure of the system (see Fig. 2b) we can expect $C_1(\cdot, \cdot)$ to be more influenced by $u_1[k]$, and $C_2(\cdot, \cdot)$ to be more influenced by $u_2[k]$. Thus, we set the filtering matrices as:

$$\mathbf{F}_1 = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad \mathbf{F}_2 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}.$$

To cope with the objectives defined by the aforementioned costs, for the vertical two-tank system we have to extend the state $\mathbf{x}[k]$ with information regarding the tracking error and the previously executed control input $\mathbf{u}[k-1]$. More precisely, for this system we add four components to $\mathbf{x}[k]$ such that $x_3[k] = r_1[k] - x_1[k]$, $x_4[k] = r_2[k] - x_2[k]$, $x_5[k] = u_1[k-1]$, and $x_6[k] = u_2[k-1]$. Therefore, for the vertical two-tank system we have that $Nx = 6$, $Nu = 2$, and $Np = 2$.

V. NUMERICAL EXPERIMENTS

In this section we implement both the standard and the multi-critic actor-critic approaches on the water systems of Section IV. For the standard actor-critic we implement the DDPG algorithm of [18], and for the multi-critic approaches we modify the DDPG following the guidelines of Sections III-A and III-B to turn it into the unfiltered multi-critic DDPG (MCDDPG) and into the filtered multi-critic DDPG (FMCDDPG), respectively. For the sake of clarity, we present the implementation details and the simulation results in Sections V-A and V-B, respectively.

A. IMPLEMENTATION DETAILS

For all the experiments of this work, the actor neural networks have two fully connected hidden layers with 20 neurons and *tanh* activation functions. Similarly, the actors' output layers have 2 neurons representing the two-components of the systems' input $\mathbf{u}[\cdot]$, and have a *tanh* activation to bound the control actions within $[-1, 1]$. In the interaction with the system, however, the outputs of the actors are scaled and shifted to be within $[u_{min}, u_{max}]$ of each system (see Table 1). The critic networks, on the other hand, are defined under two different schemes that seek a fair comparison among single-critic and multi-critic approaches. Since the multi-critic approach introduces an additional neural network, we do not consider totally fair to perform the comparison against a standard actor-critic with a single critic of the same size as the ones of the multi-critic strategy. Therefore, we explore two schemes. In the first scheme, used for the multi-critic approaches and for the DDPGs tagged as small, we use critics with two hidden layers of 40 neurons. In the second scheme, used for the DDPGs tagged as big, we double the size of the critic's hidden layers to have 80 neurons. In both cases, all the hidden layers have *tanh* activation functions and the output layers have a single neuron with a linear activation. For all critics, as in [18], the input state $\mathbf{x}[\cdot]$ is received at the first hidden layer, while the input action $\mathbf{u}[\cdot]$ is received at the second hidden layer. This is to mitigate the vanishing gradient effect in the gradients that are back-propagated to the actor network. Moreover, all inputs to the neural networks are normalized within the range $[-1, 1]$ to avoid ill-conditioned feature vectors.

Remark 7: It is worth noting that the contribution of this work is not on the specific architectures of the neural networks but rather on the proposed multi-critic schemes. The aforementioned neural network sizes were selected so that both the single-critic and multi-critic approaches were able to achieve competitive levels of performance on the considered systems. Nevertheless, with additional fine-tuning of hyper-parameters, or better design of the feature vectors, it might be possible to reduce the size of the neural networks if required.

Regarding the cost functions, in all cases we set the weight coefficients α_i so that the magnitude of each tracking error term in the cost is bounded under 1, and each switching penalty term in the cost is bounded under 0.5. Moreover, given that the multi-critic approaches break the cost $C(\cdot, \cdot)$ into N_p partitions and uses each partition to train a different critic network, the scale of the cost seen by each of the multiple critics is different than the one seen by the single-critic of the DDPG approach (recall that the single-critic approach uses the sum of all cost-partitions). As has been discussed in [27], the scale of the cost (reward) signal might affect the performance of the learning algorithm. Hence, to isolate this variable, we train the single-critic DDPGs using both the sum

[1]This kind of penalization has been studied before in other optimization-based control methods as model predictive control.

and the mean over cost-partitions. We tag these agents as sum and mean, respectively.

In this work, all the neural networks are implemented in Python with the Tensorflow deep-learning framework [28]. For all experiments, we train the agents over 1500 episodes of 400 and 200 steps of $1s$ for the quadruple-tank process and the vertical two-tank system, respectively. Furthermore, we repeat all training procedures for 5 different random seeds as a way to measure reproducibility. For all cases we set the discount factor γ in (1) as 0.99, we use the Adam optimizer [29] with a learning rate of 0.001 for both the actor and critic networks, and we set the targets' learning rate τ to be 0.001. Regarding the parameters of Algorithms 1 and 2, we set the batch size $|\mathcal{B}|$ to be 128, we set the maximum size of the memory buffer as 20000, and we set the update frequencies f_Q and f_μ such that the critic and actor are updated 10 and 1 times per episode, respectively. That way the critic networks are trained faster than the actors and thus provide more accurate gradients. Moreover, in all cases we clip the critics' targets $\hat{Q}^{\mu}_{\hat{\Phi}}(\cdot,\cdot)$ within $[-10, 10]$ to mitigate over-confident bootstrapped estimates. Lastly, for the exploration noise we use a Ornstein-Uhlenbeck random process [30] with $\theta = 0.15$ and with σ linearly annealed from 0.15 to 0.06 over the training episodes.

B. SIMULATION RESULTS

In order to evaluate the performance of the different actor-critic algorithms, we save the partial control policies obtained after every 100 episodes of training, and evaluate each of them on a fixed set of 50 randomly selected episodes[2] of $300s$. The initial conditions of such 50 evaluation episodes are sampled randomly from a finite set of initial conditions that are distributed over a wide region of the possible water-levels of the tanks. Moreover, the desired reference levels of the controlled tanks are selected to ensure their attainability, i.e., to guarantee that the desired reference levels are feasible. Unless stated otherwise, the evaluation is performed under the cost (1) averaged over the set of 50 episodes. As an illustration, Fig. 3 depicts the training performance of all the actor-critic approaches applied to an instance of the quadruple-tank process with $\gamma_1 = \gamma_2 = 0.6$. The solid lines show the mean performance taken over the random seeds and the shadows depict its standard deviation over the different seeds. Fig. 3a shows that with 1500 training episodes, the single-critic architectures with small critics cannot match the mean performance of the multi-critic approaches. Furthermore, the performance of the single-critic architectures has far more variance than the one of the multi-critic approaches. Fig. 3b, on the other hand, presents the comparison against the single-critic approaches with doubled critic's sizes. In this case we observe that, at the end of training, the DDPG under the sum cost-scheme is able to match the mean performance of

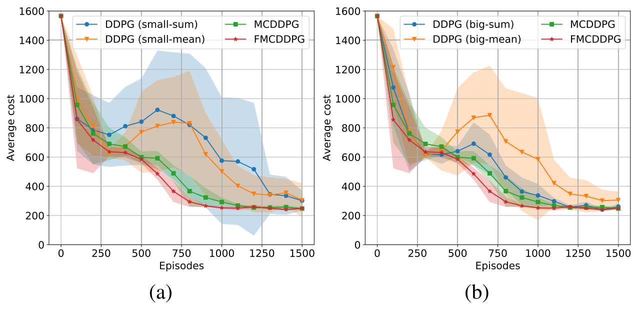


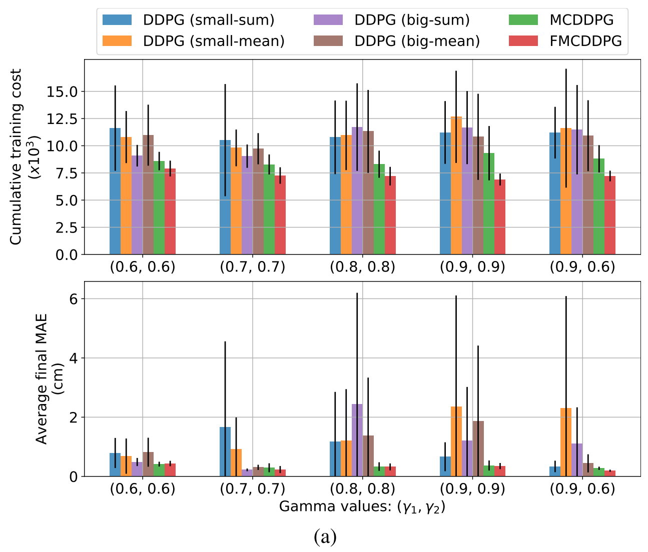
FIGURE 3. Evaluation of the trained control policies at different training episodes. (a) Comparison of multi-critic approaches against DDPGs with small critics. (b) Comparison of multi-critic approaches against DDPGs with big critics.

the multi-critic ones. Nonetheless, the multi-critic approaches display a faster performance improvement in terms of the number of training episodes. Similarly, if we compare the training performance of the unfiltered multi-critic against the filtered one, we see that the FMCDDPG outperforms the MCDDPG in terms of such learning speed. We attribute such improvement to a better credit assignment provided by the filters applied to the gradients that are back-propagated from the multiple critics. Namely, with the aid of such filters, the back-propagated gradients received by each output neuron of the actor network consider only certain component of the approximated value function. Thus, in the filtered case, each output neuron of the actor network is not only able to specialize on a particular component of the approximated value function (c.f., Remark 6), but is also less affected by the performance of the other output neurons of the actor network. Therefore, the applied filters play the role of an attention mechanism that, for the considered multi-tank systems, leads to a better learning performance both with respect to the single-critic and unfiltered multi-critic approaches.

To further validate the previous results, we repeat the training-testing procedure of Fig. 3 for various instances of the quadruple-tank process with different values of γ_1 and γ_2. In particular we use 4 symmetric values with $\gamma_1 = \gamma_2$, ranging from 0.6 to 0.9, and an additional asymmetric one with $\gamma_1 = 0.9$ and $\gamma_2 = 0.6$. We select such values in order to test the methods under different strengths of the input-output couplings, while considering only minimum phase instances of the system.[3] The results of these experiments are summarized in the bar-plots of Fig. 4a. The top bar-plots show the cumulative cost over the partial policies of the training procedure (i.e., the cost under the curves of Fig. 3 but for the corresponding system's instances). Since the performance at episode 0 is the same for all experiments, what these plots seek to measure is the speed of learning over the training episodes. The bottom bar-plots, on the other hand, evaluate the final control policies that result at the end of each training procedure. The evaluation in this case is performed under

[2]For reproducibility and comparison purposes, the precise settings of the 50 evaluation episodes used to test our approaches are available online at (copy and paste the link): https://drive.google.com/drive/folders/1-UwYgXWXMtq6hzmHsbvxIR9hDN_OFeYb?usp=sharing.

[3]We leave the study of the non-minimum phase case for future work as it involves additional difficulties that are out of the scope of this research.



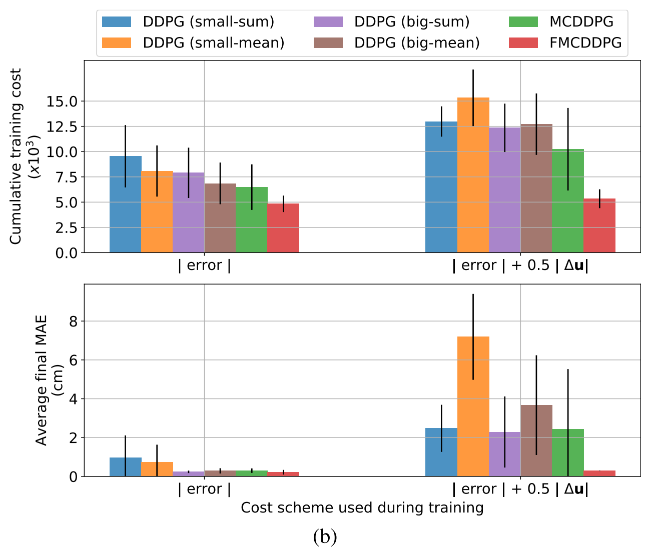


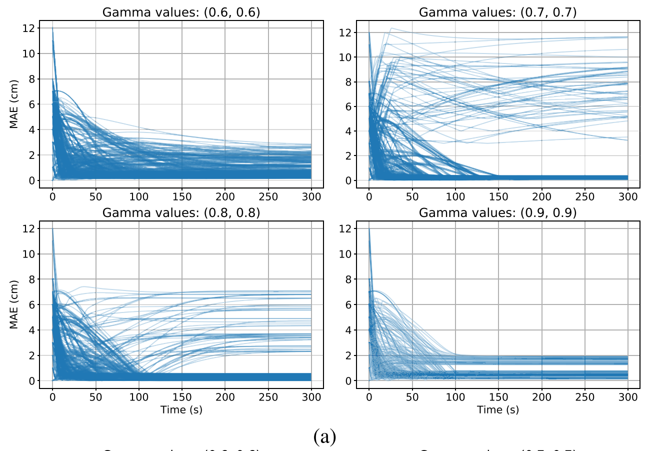
FIGURE 4. Summary of training (top) and final (bottom) performances for: (a) different instances of the quadruple-tank process, (b) different cost-schemes in the vertical two-tank system. The evaluation cost does not consider the $\Delta \mathbf{u}[k] = |\mathbf{u}[k] - \mathbf{u}[k-1]|$ terms. This ensures a comparable metric for both experiments of (b).

the average final mean absolute error metric. This metric is defined as the average over the 50 evaluation episodes, and over the control policies obtained with the 5 different random seeds, of the mean absolute error (MAE) at the final time step $k = T = 300s$ of each evaluation episode. The MAE$[\cdot]$ in this case is defined as:

$$\text{MAE}[k] = \frac{1}{2}\left|r_1[k] - x_1[k]\right| + \frac{1}{2}\left|r_2[k] - x_2[k]\right|, \quad \forall k.$$

The results of Fig. 4a show that the multi-critic approaches not only achieve, on average, the smallest cumulative costs, but also are the only ones that consistently achieve low final tracking errors across all instances of the quadruple-tank process. This is an important result as it displays an improvement in training speed and sensitivity to change in parameters when compared to the single-critic methods. Regarding the different variants of the single-critic approach, from the results of Fig. 4a we cannot establish a significant trend as all of the single-critic agents display high variance. In contrast, we do recognize that the filtered multi-critic outperforms the unfiltered one in all cases. Note that although both of the multi-critic approaches are able to achieve low tracking errors, the FMCDDPG achieves the best average performance in the least number of training episodes and with the least variance in all cases.

To illustrate the closed-loop performance that is achieved at the end of training of the studied actor-critic methods, Fig. 5 shows a set of MAE$[k]$ trajectories, with $0 \leq k \leq 300$, for different evaluation episodes, different random seeds and different instances of the quadruple-tank process. Figs. 5a and 5b display the results regarding the DDPG small-sum and FMCDDPG agents, respectively. Clearly, the FMCDDPG achieves a better controller more consistently, as it is able to bring the errors close to zero in all cases. As was expected from the results of Fig. 4a, the final errors tend towards zero but does not achieve zero. An interesting future work might



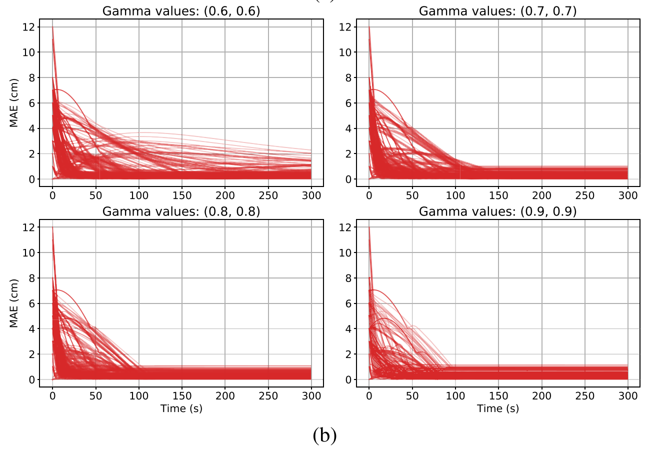


FIGURE 5. MAE$[k]$ trajectories, with $0 \leq k \leq 300$, for different evaluation episodes, different random seeds and different instances of the quadruple-tank process. (a) Final policy of the DDPG (small-sum). (b) Final policy of the FMCDDPG.

focus on the addition of integral action as a way to guarantee such zero-error at the steady state.

In order to validate the previous results on a different system, we implement all the studied actor-critic strategies on the vertical two-tank system as well. In this case, however, we test two different cost schemes with and without the penalization in the switches of the control input (i.e., penalizing or not the $\Delta\mathbf{u}[k] = |\mathbf{u}[k] - \mathbf{u}[k-1]|$ terms in the cost). As shown in Fig. 4b, the multi-critic approaches are again the ones with the fastest average learning speeds in all the experiments. Moreover, the filtered multi-critic strategy outperforms the unfiltered one in both cases. Here, we observe that the addition of the $\Delta\mathbf{u}[k]$ terms does produce a significant drop in performance for all strategies besides the FMCDDPG. We have found that the addition of the $\Delta\mathbf{u}[k]$ term in the cost is a practical way to avoid large switches in the control policies of RL agents. Therefore, an algorithm robust to the addition of such term is an attractive approach for a vast range of RL-based control applications, and, as is shown in the bottom of Fig. 4b, the FMCDDPG does not display a significant drop in training performance when the $\Delta\mathbf{u}[k]$ terms are included.

VI. CONCLUDING REMARKS

In this paper we have demonstrated the application of neural-network-based controllers to the model-free control of multi-tank water systems with unknown nonlinear dynamics. We have proposed an unfiltered multi-critic RL method that is able to consistently outperform the standard single-critic approach in all our studied systems. Furthermore, we have proposed a filtered multi-critic method which, with the assumption of shallow prior knowledge about the system's dynamics, is able to improve the training performance beyond the former multi-critic approach. It is worth to highlight that the proposed multi-critic schemes do not change the structure of the actor network, and, therefore, do not increase the controller complexity compared to the one of the standard single-critic method.

Future work should focus on the application of the multi-critic architectures to different systems and different cost schemes, and on the exploration of how to further improve the data efficiency of the multi-critic training process so that online training implementations could be feasible on modern cyber-physical systems.

REFERENCES

[1] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, vol. 1, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.

[2] D. E. Kirk, *Optimal Control Theory: An Introduction*. Chelmsford, MA, USA: Courier Corp., 2012.

[3] L. Busoniu, R. Babuska, B. D. Schutter, and D. Ernst, *Reinforcement Learning and Dynamic Programming Using Function Approximators*. Boca Raton, FL, USA: CRC Press, 2017.

[4] F.-Y. Wang, H. Zhang, and D. Liu, "Adaptive dynamic programming: An introduction," *IEEE Comput. Intell. Mag.*, vol. 4, no. 2, pp. 39–47, May 2009.

[5] B. Kiumarsi, K. G. Vamvoudakis, H. Modares, and F. L. Lewis, "Optimal and autonomous control using reinforcement learning: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 6, pp. 2042–2062, Jun. 2018.

[6] D. Wang and J. Qiao, "Approximate neural optimal control with reinforcement learning for a torsional pendulum device," *Neural Netw.*, vol. 117, pp. 1–7, Sep. 2019.

[7] P. Long, T. Fanl, X. Liao, W. Liu, H. Zhang, and J. Pan, "Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 6252–6259.

[8] Z. Yang, K. Merrick, L. Jin, and H. A. Abbass, "Hierarchical deep reinforcement learning for continuous action control," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5174–5184, Nov. 2018.

[9] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, Jul. 2018, pp. 1861–1870.

[10] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine, "Soft actor-critic algorithms and applications," 2018, *arXiv:1812.05905*. [Online]. Available: http://arxiv.org/abs/1812.05905

[11] N. Heess, J. J. Hunt, T. P. Lillicrap, and D. Silver, "Memory-based control with recurrent neural networks," 2015, *arXiv:1512.04455*. [Online]. Available: http://arxiv.org/abs/1512.04455

[12] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 1–8.

[13] K. Frans, J. Ho, X. Chen, P. Abbeel, and J. Schulman, "Meta learning shared hierarchies," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–11.

[14] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.

[15] H. V. Seijen, M. Fatemi, J. Romoff, R. Laroche, T. Barnes, and J. Tsang, "Hybrid reward architecture for reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5392–5402.

[16] S. Cabi, S. G. Colmenarejo, M. W. Hoffman, M. Denil, Z. Wang, and N. de Freitas, "The intentional unintentional agent: Learning to solve many continuous control tasks simultaneously," in *Proc. Mach. Learn. Res.*, vol. 78, 2017, pp. 207–216.

[17] D. Akimov, "Distributed soft actor-critic with multivariate reward representation and knowledge distillation," 2019, *arXiv:1911.13056*. [Online]. Available: http://arxiv.org/abs/1911.13056

[18] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2016, pp. 1–14.

[19] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2961–2970.

[20] E. Barati and X. Chen, "An actor-critic-attention mechanism for deep reinforcement learning in multi-view environments," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 2002–2008.

[21] M.-B. Radac, R.-E. Precup, and R.-C. Roman, "Data-driven model reference control of MIMO vertical tank systems with model-free VRFT and Q-learning," *ISA Trans.*, vol. 73, pp. 227–238, Feb. 2018.

[22] D. Ochoa, G. Riano-Briceno, N. Quijano, and C. Ocampo-Martinez, "Control of urban drainage systems: Optimal flow control and deep learning in action," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2019, pp. 4826–4831.

[23] K. H. Johansson, "The quadruple-tank process: A multivariable laboratory process with an adjustable zero," *IEEE Trans. Control Syst. Technol.*, vol. 8, no. 3, pp. 456–465, May 2000.

[24] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[25] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. ICML*, 2014, pp. 387–395.

[26] J. Wu, R. Wang, R. Li, H. Zhang, and X. Hu, "Multi-critic DDPG method and double experience replay," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2018, pp. 165–171.

[27] H. P. van Hasselt, A. Guez, M. Hessel, V. Mnih, and D. Silver, "Learning values across many orders of magnitude," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 4287–4295.

[28] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, and S. Ghemawat. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. [Online]. Available: https://www.tensorflow.org/

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–15.

[30] G. E. Uhlenbeck and L. S. Ornstein, "On the theory of the Brownian motion," *Phys. Rev.*, vol. 36, pp. 823–841, Sep. 1930.

JUAN MARTINEZ-PIAZUELO (Graduate Student Member, IEEE) received the B.S. and M.S. degrees in electronic engineering from the Universidad de los Andes, Bogotá, Colombia, in 2017 and 2019, respectively.

He is currently an Instructor with the Universidad de los Andes. His current research interests include feedback control, reinforcement learning, and game theory and optimization.

DANIEL E. OCHOA (Member, IEEE) received the B.S. degree *(cum laude)* in electronic engineering, the B.S. degree in physics, and the M.S. degree in electronic and computer engineering from the Universidad de los Andes, Bogotá, Colombia, in 2017 and 2019, respectively.

His current research interests include geometric control, nonlinear optimization, reinforcement learning, and quantum control.

NICANOR QUIJANO (Senior Member, IEEE) received the B.S. degree in electronics engineering from Pontificia Universidad Javeriana (PUJ), Bogotá, Colombia, in 1999, and the M.S. and Ph.D. degrees in electrical and computer engineering from The Ohio State University, in 2002 and 2006, respectively.

In 2007, he joined the Electrical and Electronics Engineering Department, Universidad de los Andes (UAndes), Bogotá, where he is currently a Full Professor and the Director of the Research Group in Control and Automation Systems (GIAP). He has co-advised the Best European Ph.D. thesis in control systems in 2017. He is the coauthor of the Best Paper of *ISA Transactions* in 2018. His current research interests include hierarchical and distributed optimization methods using bio-inspired and game-theoretical techniques for dynamic resource allocation problems, especially in energy, water, agriculture, and transportation.

Dr. Quijano was a Program Committee Member of several international conferences, such as the IEEE CDC in 2011 and IFAC NOLCOS in 2016. He was a member of the Board of Governors with the IEEE Control Systems Society (CSS) in 2014. He served as the Chair for the IEEE CSS, Colombia, from 2011 to 2013. He serves as an Associate Editor for the IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY and *Journal of Modern Power Systems and Clean Energy*. He served as a part of the editorial board for some journals such as *Journal of Artificial Intelligence Research*.

LUIS FELIPE GIRALDO (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from The Ohio State University, Columbus, OH, USA, in 2016.

He has been an Assistant Professor with the Universidad de los Andes, Bogotá, Colombia, since 2016. He is currently a member with the Center for Artificial Intelligence Research and Education CINFONIA. His current research interests include design and analysis of interconnected dynamical systems, machine learning, and humanitarian engineering.