Control of Urban Drainage Systems: Optimal Flow Control and Deep Learning in Action

Daniel Ochoa, Gerardo Riaño-Briceño, Nicanor Quijano and
Carlos Ocampo-Martinez

***Abstract*—A hierarchical control strategy is proposed to solve the optimal drainage problem in sewer systems by combining an optimization technique known as minimum scaled consensus control (MSCC) with the deep deterministic policy gradient (DDPG) algorithm. The MSCC strategy operates at the global control level, and is used to determine the flows of the hydraulic structures of the drainage system, such that the water is optimally distributed, i.e., wastewater flows are controlled to minimize saturation of water levels and/or flooding events, filling each of the drainage system components (e.g., pipes, tanks, wastewater treatment plants) proportionally to their capacity. On the other hand, the DDPG uses a model-free approach at the local control level, setting the drainage flows by operating valves and gates, without any knowledge of the inherent dynamics, so that it can be used to handle the nonlinearities of the system. Finally, a case study is presented to show the effectiveness of the proposed strategy.**

***Index Terms*—Minimum Scaled Consensus, Deep Deterministic Policy Gradients, Flow Control, Drainage Networks**

I. Introduction

NOWDAYS, flooding events are happening in most drainage infrastructures located in places that are affected by climate change, which causes negative effects both because of sanitary reasons and the deterioration other important infrastructures suffer. The reason behind these calamities comes mainly from the increase in rain intensities that have been occurring in the last decade due to climate change. One of the most appealing solutions to prevent flooding events is the usage of storm tanks, sensors, and flow control structures in order to efficiently distribute runoff flows among the components of the drainage system. Although several urban drainage systems (UDSs) count with these elements, most of the control tasks are conducted manually by operators.

As a result, a special interest has been posed in both the design and implementation of control systems to automate the UDS operation task [1], [2], [3]. Some of the already proposed strategies to operate UDSs in order to minimize flooding events include PID control [4], model predictive control (MPC) [2], optimal control, population-games-based control (PGBC) [5], among others. Nevertheless, most of the existing control strategies for large-scale UDSs are based on linear models or heuristics and do not explicitly consider the nonlinearities of the system and the phenomena characterized by them. Nonlinear control strategies that take into consideration these phenomena have been studied [1], [6] but do not suitably scale for large-scale UDSs due to their complexity.

On the other hand, there are precedents of the usage of machine learning for the real-time control of water-systems [7]. These approximations make use of artificial neural networks (ANNs) to obtain control actions and cope with the nonlinear behaviour, but are too dated to include the latest breakthroughs that have appeared in the field in regards to function approximation tasks [8] in order to improve their performance, thus not being competitive with respect to other control techniques. Even more, efforts have been put towards using machine learning techniques to forecast pluvial flooding [9] or waste-water indicators [10], but, to the best of our knowledge, have not been tightly integrated in the automatic control strategies so that they can improve with the additional information.

On this note, this paper focuses on the control problem of reducing flooding events in UDSs systems by leveraging recent advances in machine learning. For such end, the deep deterministic policy gradients (DDPG) [11] technique is studied.

The DDPG technique shows promising results in several complex tasks for systems whose dynamics cannot be easily represented by traditional models, and computational efficiency once the training procedure is carried out. Nevertheless, it demands a high computational burden to train the underlying neural networks (NNs). For this reason, it is not practical to use it directly to control large-scale systems such as the ones arising in the UDSs context. However, this need can be fulfilled by solving a linear programming problem that delivers the references to local DDPG controllers. Thus, both strategies can complement each other to be scalable and appropriate for the UDS control problem, taking into consideration its nonlinearities and its large-scale nature.

The main contributions of this paper are the design of a hierarchical control strategy that enables a better use of the existing system infrastructure and the novel usage of deep reinforcement learning, specifically of DDPG, in a water-systems setup.

D. Ochoa, G. Riaño-Briceño, and N. Quijano are with the Departament of Electrical and Electronic Engineering, Universidad de los Andes, Carrera 1^a No 19A-40, Bogotá, Colombia {de.ochoa1618, ga.riano949, nquijano}@uniandes.edu.co

C. Ocampo-Martinez is with the Automatic Control Department, Universitat Politècnica de Catalunya, Institut de Robòtica i Informàtica Industrial (CSIC-UPC), Llorens i Artigas, 4-6, 08028 Barcelona, Spain. cocampo@iri.upc.edu

This work has been partially funded by the Spanish projects DEOCS (ref. MINECO DPI2016-76493) and the Spanish State Research Agency through the María de Maeztu Seal of Excellence to IRI MDM-2016-0656.

For the hierarchical scheme, the structure is composed of two levels as shown in Figure 1. First, there is the global control level that determines the minimum feasible volumes at each UDS component, in order to avoid flooding events. Second, there is the local control level that adjusts the flows of the system by manipulating gates according to the results of the upper control level.

At the global level, an optimization problem is proposed to find the minimum scaled consensus state (MSCS) [12]. The proposed controller guarantees convergence to the MSCS for the compartmental representation of the UDS. The controller provides the reference for local DDPG controllers that are in charge of operating gates to set the desired minimum volumes at certain drainage structures. The DDPG algorithm is trained using a linear model of the UDS, which is also used to determine the flow values that lead to the MSCS.

The remainder of this paper is organized as follows. In Section II, the control problem is described in detail as well as the model used to test the proposed hierarchical control strategy. Then, in Section III, the global level controller, which estimates the MSCS for the optimal allocation of water, and the DDPG local regulator, used to achieve such allocation, are described. Next, in Section IV, a case study is presented in order to assess the entire proposed strategy. In Section V, results of the proposed approach are compared to those obtained by using PID controllers instead of DDPG agents at the local control level, showing both the utility of the hierarchical control structure and the flexibility of the DDPG controllers for the studied setup. Finally, in Section VI, some concluding remarks are given alongside with future research perspectives.

II. Preliminaries

A. Problem Formulation

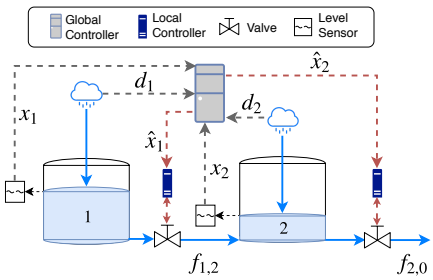

Fig. 1. Hierarchical control scheme for drainage systems adapted from [2]. A global controller retrieves information from rain and level sensors and sets the flow references for the local control level.

In this work, a hierarchical control scheme (see Figure 1) is proposed to dynamically adjust the flows of the UDSs, through the manipulation of valves and gates, in order to minimize flooding events. The proposed strategy is used to efficiently evacuate external inflows or disturbances d_i, such that flows are set according to a global control law that minimizes peak volumes in every storage or conveyance structure of the system.

A global control law is designed to fill the elements that conform the drainage infrastructure according to their maximum capacity $\bar{x}_i$, based on the conservation laws inherent to the dynamics of the system. Thus, a linear compartmental model is used for the global control level to determine the references for local DDPG controllers [11]. These controllers require information from the simulation environment, which is provided by the linear compartmental model, in order to determine the setting of valves and gates that relates to a flow reference $\hat{f}_{i,j}$.

The compartmental representation is used to model dynamical systems in which a resource is kept and transferred among storage units called compartments. The transference of the resource is here called a flow. Thus, a system may have external and/or internal flows. Every flow $f_{i,j}$ that goes from the i-th compartment to the j-th compartment is considered internal. External inflows are denoted by $f_{0,i}$ and external outflows are denoted by $f_{i,0}$.

Taking into consideration that the relationships between the volume stored in the compartments $x \in \mathbb{R}^n$, and the flows $f_{i,j}, f_{i,0} \in \mathbb{R}_+$ are considered linear, the compartmental system is represented by a triplet $(\mathcal{G}, \mathcal{D}, \Omega_{\mathcal{O}})$, where $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \Omega)$ is a weighted digraph, denominated the compartmental graph, with edges $(i,j) \in \mathcal{E}$ that represent internal flows $f_{i,j}$, nodes $i \in \mathcal{V}$ that represent compartments, and weights $\omega_{ij} \in \Omega$ denominated here as discharge coefficients of the system[1]. Additionally, $\mathcal{D} = \{d_i(\tau) \triangleq f_{0,i}(\tau) : (0,i) \in \mathcal{V}\}$ contains the external inflows, or disturbances, at time $\tau \in \mathbb{R}_+$, and $\mathcal{O} = \{(i,0) : (i,0) \in \mathcal{V}\}$ is the set of edges that represent external outflows with $\Omega_{\mathcal{O}} = \{\omega_{i,0} \in \mathbb{R}_+ : (i,0) \in \mathcal{O}\}$ the set that contains their associated discharge coefficients such that $f_{i,0}(\tau) = \omega_{i,0} x_i(\tau)$. Thus, the model for the global control level is composed of linear state equations for each compartment in the system as follows:

$$\dot{x}_i(\tau) = d_i(\tau) + \sum_{j \in \mathcal{U}_i} \omega_{i,j} x_j(\tau) - \sum_{j \in \mathcal{D}_i} \omega_{i,j} x_i(\tau) - \omega_{i,0} x_i(\tau), \quad \forall i \in \mathcal{V}, \tag{1}$$

with $\mathcal{U}_i = \{j \in \mathcal{V} : (j,i) \in \mathcal{E}\}$, $\mathcal{D}_i = \{j \in \mathcal{V} : (i,j) \in \mathcal{E}\}$ the sets of upstream and downstream neighbors of i respectively. Note that (1) can be written equivalently in the state-space representation

$$\dot{x}(\tau) = Ax(\tau) + d(\tau), \tag{2}$$

where $A = \mathcal{A}_\Omega^\top - \text{diag}(\mathcal{A}_\Omega \mathbb{1}_n + \tilde{\omega})$, with $\mathbb{1}_n$ a column vector of ones, $\tilde{\omega} = [\omega_{1,0}, \omega_{2,0}, ..., \omega_{n,0}]^\top$, and $\mathcal{A}_\Omega$ is the adjacency matrix of the compartmental graph.

Finally, it is worth noting that, in the scheme proposed in Figure 1 the global controller receives information from level sensors, instead of volume amounts. However, this is not an issue for the proposed formulation given that for all the components of the drainage system a relationship between volume and level can be proposed as described in [13]. Furthermore, despite the fact that in the global control level the system dynamics are treated in the continuous-time space, the local regulators are studied and implemented using a discrete-time scheme. The main reason behind this fact is

[1]The ratio between flow and volume is commonly known as discharge coefficient in the context of linear models for water systems [13].

that the global control level delivers the references to the local controllers at the beginning of the scenario and has minimal further interaction with the latter. Also, from a practical standpoint, the discrete implementation of the reinforcement learning controllers has proven to be both straightforward and highly effective for environments with continuous dynamics [14].

B. Reinforcement Learning

The reinforcement learning (RL) problem is defined on a framework for which an *agent* tries to maximize a reward function delivered by some *environment* by cleverly selecting actions from a pool of possible interactions called a *policy*. Each action u_t, when performed over the environment at time $t \in \mathbb{N}$, causes a change of state x_t and, under such transition, the named reward $r(x_t, u_t)$ is delivered to the agent. This relationship between agent and environment is usually modeled by a Markov decision process for which the transition dynamics between states are generally characterized to be probabilistic and memory-less. In order to cope with this stochastic nature, most RL algorithms try to maximize the action-value function

$$Q^{\pi}(x,u) = \mathbb{E}_{\pi}\left[\sum_{k=0}^{\infty} \gamma^k r(x_{t+k}, u_{t+k}) | x_t = x, u_t = u\right], \tag{3}$$

with $\gamma \in (0,1)$ a discount factor that weighs higher most recent rewards. This function specifies the expected reward to be received in the future given that the environment is in state x and the agent acts with u sampled from a policy π. It can be shown that this Q-function follows the Bellman equation (4) that characterizes it in a recursive fashion. In (4), $p(x_{t+1}, r_{t+1}|x_t, u_t)$ represents the transition dynamics of the environment and $p_{\pi}(u_{t+1}|x_{t+1})$ the probability of choosing u_{t+1} as the next action under policy π given that the state is x_{t+1}. If these dynamics were known, for a given policy the associated Q-function could be solved directly with the Bellman equation and a better policy could be proposed instead. By repeating the procedure, an optimal policy could be found.

Nevertheless, those dynamics are often assumed to be unknown for the RL algorithms and in order to evaluate the Q-function, interaction with the environment becomes the de-facto strategy. A drawback with the usage of the Bellman equation, and in general with the traditional RL algorithms, is that they are defined over discrete state and action spaces, which poses a problem for scalability to continuous action and state spaces. In order to overcome this dimensionality problem, two approaches are taken:

- **Value-function approximation**: The Q-function is represented by a functional form $Q(x,u) \approx Q(x,u|\eta^Q)$ parameterized by $\eta^Q \in \mathbb{R}^n$. The best parameter is found by minimizing a loss measure that indicates how far are the approximator's predictions from the sampled values of the value function.
- **Policy-Gradient methods**: The policy π is expressed in an explicit manner $\pi_{\theta^\pi}(x) = f(x|\theta^\pi)$, parameterized by θ^π, instead of obtaining it directly from the Q-function. To improve the policy, θ^π is adjusted so that a performance measure $J(\pi_{\theta^\pi})$ increases. Such performance measure is here considered to be

$$J(\pi_{\theta^\pi}) = \mathbb{E}_{\pi}\left[\sum_{k=0}^{\infty} \gamma^k r(x_k, u_k) | x_0 = x\right], \tag{5}$$

and the way of adjusting the parameter is to use gradient ascent over (5), i.e.,

$$\theta^\pi_{t+1} = \theta^\pi_t + \alpha \nabla J(\pi_{\theta^\pi}), \tag{6}$$

where $\alpha \in \mathbb{R}_+$ is the step length of each parameter update. When used in conjunction with a value-function approximator, the policy approximator receives the name of *actor* and the value-function one receives the name of *critic*.

In the following section, the description of an optimization problem is given for the global control level, and also details of the DDPG algorithm, are exposed.

III. Hierarchical Control

A. Global control level: Optimal Flow Distribution

Assume that it is required that the system achieves a final value $x^* = z\mathbb{1}_n$, where $z \in \mathbb{R}$, i.e., it is required that the system reaches output consensus [15]. To do so, it is necessary that the system has an equilibrium point in x^*. For the studied compartmental systems, which are positive and affine [16] due to the nature of the dynamics presented in (2), if the disturbances are constant and positive at every time t, i.e., $d(t) = d \in \mathbb{R}^n_+$, then Theorem 1 holds.

Theorem 1 (Positive Systems Stability [17])**.** *Consider a linear positive affine system characterized by (2), where A is a Metzler matrix and $d > 0$. If A is Hurwitz, then*

1) *The system has an unique equilibrium point $x^* \in \mathbb{R}^n$, which is the solution of $Ax^* + d = 0$.*
2) *The equilibrium point x^* is positive.*
3) *All the trajectories converge asymptotically to x^*.*

From the conservation law (1), it is guaranteed that A is a Metzler matrix [18] given that

$$\sum_{i \in \mathcal{V},\, i \neq j} \omega_{ji} + \omega_{i,0} = \sum_{i \in \mathcal{V},\, i \neq j} \omega_{ij}.$$

However, it is necessary to establish Assumption 1 to guarantee that A is Hurwitz.

Assumption 1. The compartmental system described by (2) is positive, affine, and output-connected, i.e., there exists a directed path from every compartment $i \in \mathcal{V}$ to a compartment j with $\omega_{j,0} > 0$.

Then, given that the conditions for stability have been established, there exists an equilibrium point x^* that is the solution of

$$Ax^* + d = 0. \tag{7}$$

$$Q^{\pi}(x_t,u_t)=\sum_{x_{t+1},r(x_t,u_t)} p(x_{t+1},r(x_t,u_t)|x_t,u_t)\left[r\left(x_t,u_t\right)+\gamma\sum_{u_{t+1}} p_{\pi}(u_{t+1}|x_{t+1})Q^{\pi}(x_{t+1},u_{t+1})\right] \tag{4}$$

Now, since $x^*=z\mathbb{1}_n$, condition (7) is equivalent to

$$\sum_{j\in\mathcal{V},\,i\neq j}\omega_{ji}-\left(\omega_{i,0}+\sum_{j\in\mathcal{V},\,i\neq j}\omega_{ij}\right)+\frac{d_i}{z}=0,\quad\forall\, i\in\mathcal{V}. \tag{8}$$

Additionally, the flows are physically restricted by minimum and maximum values

$$\omega_{ij}^{-}\leq\omega_{ij}\leq\omega_{ij}^{+},\quad\forall\, i,j\in\mathcal{V},\, i\neq j \tag{9}$$

$$\tilde{\omega}^{-}\leq\tilde{\omega}\leq\tilde{\omega}^{+}, \tag{10}$$

where $\omega_{ij}^{-},\omega_{ij}^{+}\in\mathbb{R}$, $\tilde{\omega}^{-},\tilde{\omega}^{+}\in\mathbb{R}_{+}^{|\mathcal{O}|}$.

Definition 1 (Scaled Consensus [12])**.** A network in which the state equation of each node $i\in\mathcal{V}$ depends only on the local state $x_i(\tau)$ and the states $x_j(\tau)$ of the upstream neighbors $j\in\mathcal{U}_i$, achieves *scaled consensus* if the proportions between the state variables, i.e., $x_i/x_j\ \forall\, i,j\in\mathcal{V}$, reach specified constants in the asymptote. Formally, a network whose dynamics are described by (2) achieves scaled consensus to $[\kappa_1,...,\kappa_n]^{\top}$, with $\kappa_i>0\ \forall\, i\in\mathcal{V}$, if

$$\lim_{t\to\infty}\left(\kappa_1 x_1(\tau)-\kappa_j x_j(\tau)\right)=0,\qquad j=2,...,n, \tag{11}$$

for all the initial conditions $x(0)$.

Theorem 2. *The compartmental system, with dynamics defined by (2), achieves scaled consensus to* $x_f=x^*/\max\{x^*\}=[\kappa_1,...,\kappa_n]^{\top}$ *if*

$$\sum_{j\in\mathcal{V},\,i\neq j}\kappa_i\omega_{j,i}+\frac{d_i}{z}=\sum_{j\in\mathcal{V},\,i\neq j}\kappa_i\omega_{i,j}+\kappa_i\omega_{i,0},\quad\forall\, i\in\mathcal{V}. \tag{12}$$

Proof. Notice that (12) is equivalent to (8), thus, $x^*\in\mathbb{R}_{++}^n$ is an equilibrium point for all the initial conditions $x(0)$ of (2), since Theorem 1 holds. Moreover, since $\kappa_i=x_i^*/z$ and (12) hold, condition (11) holds, thus, the system achieves scaled consensus to x_f. □

Now, considering Definition 1, Theorem 2 holds for a system described by (1), and thus an optimization problem can be proposed to determine the value of the weights $\omega_{i,j}$, which implies that the system reaches a desired scaled consensus state x_f such that the the maximum peak volume $z=\max\{x^*\}$ is minimized. This problem is defined as the MSCS problem as follows:

$$\begin{aligned}\min_{z,\Omega,\bar{\omega}}\quad & z\\ \text{subject to}\quad & \text{(9), (10), and (12).}\end{aligned} \tag{13}$$

Once the MSCS is determined by solving (13), its value becomes the reference for the DDPG local controllers, which determines the setting of gates and/or valves, such that desired volumes are fixed in every reservoir of the UDS.

B. Local control level: Deep Deterministic Policy Gradients

The Deep Deterministic Policy Gradient (DDPG) [11] algorithm is used as the local flow regulator. This RL algorithm makes use of three components: a critic, an actor, and a *replay-memory*. These three components are constantly intertwined in order to find an optimal policy, and the associated action-value function, so that the agent can maximize the expected received reward from the environment. For such end, a pair of optimization problems are stated; one to update the critic parameters, and one to do so for the actor ones. The progressive update of such functional parameters until an optimum is obtained constitutes the *training* (or learning) stage of the DDPG algorithm. In order to sample information from the environment and progressively solve the optimization problems, DDPG uses policies of the form

$$\pi(x_t)=\mu(x_t|\theta^{\mu})+\mathcal{N}_t,$$

where $\mu(x|\theta^{\mu})$ is a deterministic policy, parameterized by a vector θ^{μ}, and $\mathcal{N}_t$ is noise sampled from an Ornstein-Uhlenbeck process. The addition of noise around the deterministic policy allows for the exploration of, otherwise, unvisited states so that more information is obtained to better estimate the Q-function.

With this in mind, the associated optimization problem for the critic consists in the minimization of the loss function given by

$$L(\eta^{Q})=E[(Q(x_t,u_t|\eta^{Q})-y_t))^2], \tag{14}$$

where

$$y_t=r(x_t,u_t)+\gamma Q(x_{t+1},\mu(x_{t+1}|\theta^{\mu})|\eta^{Q}). \tag{15}$$

This loss function specifies the sample error of the current Q-function, by using (4). On the other side, in order to find better policies that maximize (5), the functional parameters of the actor are updated with (6), taking the performance gradient as

$$\nabla_{\theta^{\mu}}J(\mu_{\theta^{\mu}})=E[\nabla_u Q(x,u|\eta^{Q})\nabla_{\theta^{\mu}}\mu(x|\theta^{\mu})]. \tag{16}$$

In addition, DDPG uses a replay-memory with which past experience is sampled in order to update both the value-function and policy approximators. As such, the parameter updates are not only taking into account the last sampled information but also past information stored in the aforementioned memory. This is done in order to break the temporal correlation between successive updates, which has been found to deter the improvement of performance for each policy update.

Finally, it is important to say that in DDPG, both actor and critic are constructed as neural networks due to the capacity of abstraction they can achieve for complex functions such as those related with non-linear dynamic systems.

IV. Case Study

An open-channel pool setup [1], hereafter denominated the five-reservoir system, is next described. As presented in Figure 2, the system is composed of five sub-catchments that are affected by a precipitation event, the precipitation becomes runoff (marked in red), and then, it is transported through a system of five reservoirs in series (the values of the runoff flows that affect each reservoir are presented in Table I).

In order to control the flows of the system, linear gates are used [13]. By adjusting the setting of gates, the volumes stored at each reservoir are controlled in order to change the respective associated flows, as shown in Figure 2. Nevertheless, the last gate, i.e., the gate that controls the outflow from the fifth reservoir remains fixed (i.e., $\omega_{5,0} = 0.0276$).

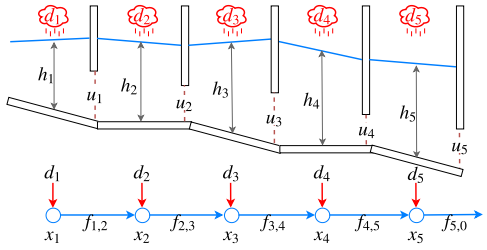

Fig. 2. Five open channels in series. The volume at each channel is controlled by using a gate. Each reservoir receives runoff water coming from the environment (red flows). The compartmental graph of the system is presented at the bottom.

TABLE I
Parameters of the five-reservoirs system

Max. Volume $[m^3]$	Discharge Coefficient $[1/s]$	Disturbance $[\ell/s]$
0.7	0.0838	5
3.23	0.0570	30
32.34	0.0064	20
75.41	0.0038	10
7.07	0.0276	20

To avoid flooding events, first, the MSCC algorithm is used to determine the minimum feasible volumes that can be achieved by manipulating the flows of the system. Once those volumes are determined, four DDPG controllers are used to determine the setting of gates in order to manipulate the flows of the system. The specific setting computed by the algorithm for each one of the gates is assumed to be between 0, closed, and 1, open. Thus, the actual control actions affecting the flow dynamics of the system. at a time t, are computed as follows:

$$u_{i,t}^{actual} = u_{i,t}^{DDPG} k_i x_{i,t}, \quad i \in \{1,2,3,4\}. \tag{17}$$

With the DDPG algorithm achieving the references, the efficient distribution of flows is guaranteed since the MSCS is reached.

A. DDPG configuration

The DDPG agents controlled the settings of the first four gates shown in Figure 2. Following the architecture described in [11], fully-connected neural networks with two hidden layers are used for the actor and the critic. The number of neurons used are 500 for the first hidden layer and 325 for the second one. Additionally, all the neurons used rectifier linear units [19] with the exception of the ones composing the output layer of the actor, which used a sigmoid activation function in order to bound the actions between 0 and 1. The state delivered to each of the DDPG agents, as the observation, contained all the current volumes and flows of the compartmental system.

With the named configuration, each one of those DDPG agents has as goal the maximization of the expected value of a reward function of the form

$$r_i(x_{i,t}, u_{i,t}) = -\left(\frac{x_{i,t} - \hat{x}_i}{\bar{x}_i}\right)^2, \qquad i \in \{1,2,3,4\}, \tag{18}$$

where $x_{i,t}$ is the current volume, $\hat{x}_i$ is the reference volume and $\bar{x}_i$ the maximum volume for the i-th reservoir. With this, the agents only care for locally achieving their control objectives.

V. Simulation Results

For the case study, the MSCS is given by $x^* = [0.3048, 1.4064, 14.0819, 32.8359, 3.0785]^\top$. By normalizing the values with respect to the maximum capacities of each reservoir, scaled consensus in terms of volume values is indeed output consensus in terms of the normalized volume values, i.e., $x_i^*/\bar{x}_i = z = 43.54\%$, for all $i \in \{1,2,3,4,5\}$.

Thus, the estimated MSCS became the reference for the local regulators (i.e., $\hat{x} = x^*$). By feeding the references to the agents, after training, the results shown in Figure 3 are obtained. The figure shows the results of the four DDPG agents modifying the gate settings so that the named references are reached. It is interesting to notice that, despite the fifth gate setting remained fixed, it is possible to achieve output consensus by solving the MSCS problem.

The dynamics and control actions having PID controllers as the local regulators are also shown in Figure 3. The PIDs are configured to regulate the error related to each reservoir to zero and a feedforward term is calculated, knowing the full-system dynamics, so that the steady-state could be more easily attained for each reference.

In order to explicitly quantify how well the strategies perform in achieving the whole reference set, a cumulative squared error (CSE) is established. It takes into account each of the associated reservoir error values as follows:

$$CSE(\mathbf{x}_t) = \sum_{k=0}^{t} \sum_{i=1}^{5} \left(\frac{x_{i,k} - \hat{x}_i}{\bar{x}_i}\right)^2 \left(\frac{\bar{x}_i}{\sum_i \bar{x}_i}\right). \tag{19}$$

The used performance measure is a weighted sum, with which either more or less importance is given to each individual error depending on how much the maximum volume of the associated reservoir constitutes of the maximum allowed volume for the UDS. This allows to judge the performance of the whole set of local individual controllers for their global impact on the solution. Thus, higher attention is paid to the regulation of the more critical reservoirs, i.e., the ones capable of storing greater amounts of water.

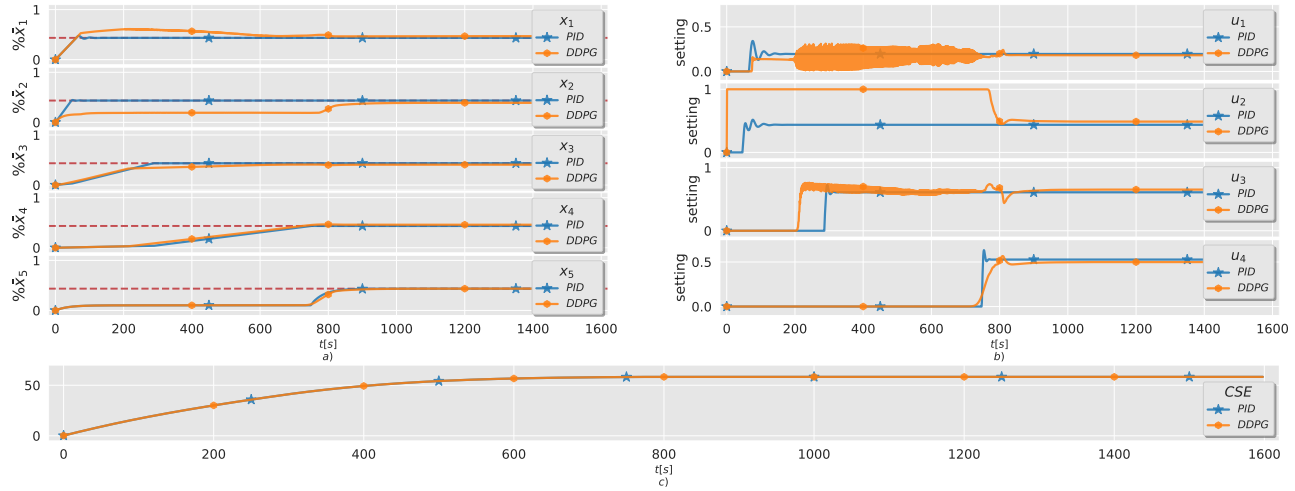

Fig. 3. DDPG local regulation: a) shows the relative volumes, with respect to associated maximum, of each one of the five reservoirs described for the case study. The dashed line, represents the desired references. Additionaly, b) shows each gate setting behaviour under the control of the respective agent. In c), a performance measure, as defined in (19), measures how far have the set of controlled reservoir volumes been from reaching the reference volumes.

As shown by the performance measure, both the DDPG and the PID controllers successfully achieve the MSCS in the UDS with similar performance, but the first does so without any explicit knowledge of the environment in comparison to the full information of the system dynamics used to make the PIDs work. This makes the extension of the usage of DDPG to nonlinear water distribution systems straightforward, situation which is not always attainable with the implementation of PIDs.

VI. Concluding Remarks and Future Work

A hierarchical control strategy composed of a minimum scaled consensus control (MSCC) strategy and deep deterministic policy gradients (DDPG) controllers, as global and local level controllers respectively, has been proposed. Such approach is tested in a linear open-channel pool case study, showing the effectiveness of the proposal for the urban drainage system (UDS) control problem. The local DDPG regulators achieve near-optimal performance for the references established by the MSCC without any explicit knowledge of the environment. Furthermore, the work here presented introduces a novel usage of the deep reinforcement learning procedure, as in the context of control of water distribution structures with multiple inputs and multiple outputs has not yet been found in the explored literature.

As future work, the implementation of the hierarchical control structure acting over scenarios involving non-linear dynamics is a natural extension. Additional randomization in the environment with which the DDPG agents interact during training could also prove beneficial making them more robust to disturbances and environment changes during the testing phase.

References

[1] X. Litrico and V. Fromion, *Modeling and control of hydrosystems*. Springer Science & Business Media, 2009.

[2] C. Ocampo-Martinez, *Model predictive control of wastewater systems*. Springer Science & Business Media, 2010.

[3] J. Barreiro-Gomez, N. Quijano, and C. Ocampo-Martinez, "Constrained distributed optimization: A population dynamics approach," *Automatica*, vol. 69, pp. 101–116, 2016.

[4] M. Schütze, A. Campisano, H. Colas, W. Schilling, and P. A. Vanrolleghem, "Real time control of urban wastewater systems—where do we stand today?" *Journal of hydrology*, vol. 299, no. 3-4, pp. 335–348, 2004.

[5] N. Quijano, C. Ocampo-Martinez, J. Barreiro-Gomez, G. Obando, A. Pantoja, and E. Mojica-Nava, "The role of population games and evolutionary dynamics in distributed control systems: The advantages of evolutionary game theory," *IEEE Control Systems*, vol. 37, no. 1, pp. 70–97, 2017.

[6] Q. Wu, X. Litrico, and A. M. Bayen, "Data reconciliation of an open channel flow network using modal decomposition," *Advances in Water Resources*, vol. 32, no. 2, pp. 193–204, 2009.

[7] S. Mounce, "A comparative study of artificial neural network architectures for time series prediction of water distribution system flow data," in *Machine Learning in Water Systems-AISB Convention 2013*. Sheffield, 2013, pp. 5–12.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[9] J. Noymanee, N. O. Nikitin, and A. V. Kalyuzhnaya, "Urban pluvial flood forecasting using open data with machine learning techniques in pattani basin," *Procedia Computer Science*, vol. 119, pp. 288 – 297, 2017, 6th International Young Scientist Conference on Computational Science, YSC 2017, 01-03 November 2017, Kotka, Finland.

[10] F. Granata, S. Papirio, G. Esposito, R. Gargano, and G. de Marinis, "Machine learning algorithms for the forecasting of wastewater quality indicators," *Water*, vol. 9, no. 2, p. 105, 2017.

[11] 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 *Proceedings of the 4th International Conference on Learning Representations (ICLR-2016)*, 2016.

[12] S. Roy, "Scaled consensus," *Automatica*, vol. 51, pp. 259–262, 2015.

[13] V. Te Chow, *Open channel hydraulics*. McGraw-Hill Book Company, Inc; New York, 1959.

[14] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel, "Benchmarking deep reinforcement learning for continuous control," *International Conference on Machine Learning*, pp. 1329–1338, 2016.

[15] Z. Zhao and Z. Lin, "Global leader-following consensus of a group of general linear systems using bounded controls," *Automatica*, vol. 68, pp. 294–304, 2016.

[16] P. De Leenheer and D. Aeyels, "Stabilization of positive linear systems," *Systems & Control Letters*, vol. 44, no. 4, pp. 259–271, 2001.

[17] F. Bullo, *Lectures on Network Systems*, 1st ed. CreateSpace, 2018, with contributions by J. Cortes, F. Dorfler, and S. Martinez. [Online]. Available: http://motion.me.ucsb.edu/book-lns

[18] C. Briat, "Sign properties of Metzler matrices with applications," *Linear Algebra and its Applications*, vol. 515, pp. 53–86, 2017.

[19] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, vol. 15. PMLR, 11–13 Apr 2011, pp. 315–323.