2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

# Multi-agent Model Predictive Control cooperation in a real eight-tank plant based on reinforcement learning

Óscar Aponte-Rengifo, Mario Francisco and Pastora Vega

*Abstract*— **In this work, a negotiation agent trained by deep reinforcement learning methodology is employed in order to achieve control objectives in a multi-agent cooperative distributed implementation of model-based predictive control (MPC). The negotiation agent is located in the upper layer of a hierarchical control architecture to achieve a consensus between the different local MPC controllers, providing weighting coefficients for the negotiation process between the control sequences provided by the local controllers. The training algorithm for the reinforcement learning of this agent is the deep deterministic policy gradient algorithm (DDPG). The validation of the negotiation agent has been performed successfully both in simulation and in a real laboratory plant composed of eight coupled water tanks, which is a non-linear system characterized by high interaction between subsystems.**

## I. Introduction

Complex large-scale processes are difficult to control using centralized MPC techniques, mainly due to the difficulty of obtaining an accurate centralized model of the process and the high computational capacity required for solving the optimization problems involved. Then, decentralized control emerges as a first approach to tackle these problems, because it considers local models and controllers for the different subsystems. However, many large-scale systems are made up of coupled interacting entities, and therefore it is necessary to adopt a distributed control scheme such as cooperative distributed model predictive control (DMPC), where information is exchanged between local MPC agents.

For the cooperative coordination of the local controllers and the different control signals that they provide, negotiation between agents is a useful technique, with the aim to achieve a reasonable consensus between them according to global control objectives. Moreover, consensus can be carried out by a negotiation agent that does not require the local models of the process and with the only additional communication steps than receiving as inputs the proposals of the local agents. Some techniques for negotiation are cooperative games, bargaining games or applying fuzzy logic inferences, among others [1]. They provide good results, but with some difficulties to incorporate uncertainty due to unmodeled dynamics and other unmeasured disturbances.

The deep reinforcement learning (DRL) method, based on successful reinforcement learning (RL) methodologies [2], leverage deep neural networks as an approximation function of the agent's behaviour to address problems involving large continuous state and action spaces [3], with no need of process models. Basically, the DRL algorithms seek to optimize the behavior of the agent only through an agent-environment interaction, where a policy $\pi$ defines the agent's behaviour. The procedure is simple; the agent sends an action to the environment, and as feedback the agent gets the new state representing the consequences of the action over the environment, and the reward, which quantifies the policy suitability regarding the considered objective.

Currently, DRL is involved in a variety of continuous control problems in the automatic control context, and in this work it has been applied to DMPC negotiation of an eight coupled tanks plant. The four coupled tanks plant is a well known benchmark to validate advanced control techniques due to its high interaction between subsystems [4]. Some recently methodologies related to our work are [5], where a DMPC based on fuzzy negotiation has been developed, and [6], with an extension of the previous DMPC to an eight tanks plant with control stability guarantees. Another approach is [7], where negotiation based on RL is applied to the eight tanks plant, with good results in simulation, using the Policy Gradient algorithm with discrete time actions. Some advanced control techniques applied to the real four tanks plant benchmark are for example [8], [9], [10], but any of them applies RL to the real plant.

In this work, a novel DRL methodology that uses deep deterministic policy gradient (DDPG) for training has been developed for setting up a negotiation agent in a cooperative DMPC [6], considering continuous action and state spaces and extending the work of [7]. One of the main advantages of this new negotiation procedure is that the agent does not require any prior knowledge of the system to work properly after some initial training. The proposed negotiation agent is trained and validated in simulation of an interconnected eight-coupled water tanks, and also validated in a real plant located in a laboratory of the University of Salamanca in Spain.

The deep deterministic policy gradient (DDPG) [11] is an algorithm for actor-critic RL architectures, which involves the deep Q-Network (DQN) [12] and the deterministic policy gradient algorithms (DPG) [13]. Basically, an actor-critic approach uses the Bellman equation to learn a Q-value function from DQN algorithm, and in turn, it employs the Q-value function to learn a policy from the DPG algorithm, where deep neural networks (DNN) are used as approximations functions of the Q-value function (critic) and the policy (actor).

The rest of the paper is structured as follows. Section 2

This work has been supported by projects PID2019-105434RB-C31 and TED2021-129201B-I00 of the Spanish Government.

All authors are in the Department of Computer Science and Automation, University of Salamanca, Spain. idu17344@usal.es, mfs@usal.es, pvega@usal.es

presents the general problem statement. Section 3 summarizes the methodology developed, including the DDPG algorithm for training the negotiation agent. Section 4 presents the case study, which is a system composed of eight interconnected water tanks. Section 5 details the application of the methodology to the current problem. Section 6 shows some training details of the proposed negotiation agent, and section 7 shows the simulation and experimental results, to end with section 8 with some conclusions.

## II. Problem statement

The system consists of subsystems $\mathcal{N} = \{1,2,...,N\}$ defined by the following local linear models,

$$x_l(t+1) = A_l x_l(t) + B_{ll} u_l(t) + w_l(t), \tag{1}$$

where $t \in \mathbb{N}_{0_+}$ is the time instant, $x_l \in \mathbb{R}^{q_l}$ the state vector, $u_l \in \mathbb{R}^{r_l}$ the input vector, the subsystem $l \in \mathcal{N}$, constrained on the convex sets that contain the origin $\mathscr{X}_l \triangleq \{x_l \in \mathbb{R}^{q_l} : A_{x,l}x_l \leq b_{x,l}\}$ and $\mathscr{U}_l \triangleq \{u_l \in \mathbb{R}^{r_l} : A_{u,l}u_l \leq b_{u,l}\}$, $A_l \in \mathbb{R}^{q_l \times q_l}$ and $B_{ll} \in \mathbb{R}^{q_l \times r_l}$ are matrices of proper dimensions. The coupling with other subsystems $j$ is defined as the measurable disturbances vector $w_l \in \mathbb{R}^{q_l}$, where $j$ refers to the set of neighbors $\mathcal{N}_l \triangleq \{j \in \mathcal{N} \setminus l : B_{lj} \neq 0\}$, i.e.,

$$w_l(t) = \sum_{j \in \mathcal{N}_l} B_{lj} u_j(t), \tag{2}$$

where $u_j \in \mathbb{R}^{r_j}$ is the input vector of subsystem $j$ and the matrix $B_{lj} \in \mathbb{R}^{q_l \times r_j}$ models the input coupling between $l$ and $j$. Furthermore, $w_l$ is bounded in a convex set $\mathscr{W}_l \triangleq \bigoplus_{j \in \mathcal{N}_l} B_{lj}\mathscr{U}_j$ due to system constraints. The neighborhood affected by agent $l$ is defined as $\mathscr{M}_l \triangleq \{j \in \mathcal{N} \setminus l : B_{jl} \neq 0\}$. Hence, from a general perspective, the overall system evolution can be formulated as follows

$$x_{\mathcal{N}}(t+1) = A_{\mathcal{N}} x_{\mathcal{N}}(t) + B_{\mathcal{N}} u_{\mathcal{N}}(t). \tag{3}$$

A multi-agent cooperative distributed MPC approach (Fig. 1) takes over the control of the subsystems $\mathcal{N}$, where the low-level control layer includes a fuzzy logic based negotiation procedure between each pair of local subsystems as detailed in [6]. The DRL negotiation agent is located in the upper-level control layer and gets from the low-level layer the $U_l^{f_m}(t)$ control sequences provided by each agent (MPC local controller) $l \in \mathcal{N}$ that participates in the negotiation; and with $m \in \{1,2,...,M_l\}$, where $M_l$ is the number of total control sequences available to agent $l$ because it can belong to different neighborhoods and the corresponding negotiation in the low-level layer will provide a different control sequence. Each control sequence is a vector of control signals with dimension equal to the prediction horizon $N_p$ of local MPC controllers.

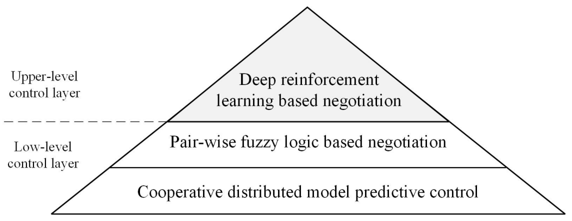
Upper-level control layer
Deep reinforcement learning based negotiation
Low-level control layer
Pair-wise fuzzy logic based negotiation
Cooperative distributed model predictive control

Fig. 1. Control architecture

### A. Control objectives

The objective of the negotiation agent is to provide weighting coefficients for the consensus among the control sequences $U_l^{f_m}(t) \in \left\{U_l^{f_1}(t), U_l^{f_2}(t), ..., U_l^{f_{M_l}}(t)\right\}$, in order to obtain the final control sequence $U_l^f$ to be applied in the plant. The consensus seeks to achieve the global control objectives of the DMPC based architecture, minimizing the overall system cost function $J_{\mathcal{N}}\left(x_{\mathcal{N}}(t), U_{\mathcal{N}}^f(t)\right)$ and satisfying the constraints, where $x_{\mathcal{N}} = (x_l)_{l \in \mathcal{N}}$ and $U_{\mathcal{N}}^f = \left(U_l^f\right)_{l \in \mathcal{N}}$. The cost function $J_{\mathcal{N}}$ is computed at each time instant $t$ based on the prediction errors of $x_{\mathcal{N}}$ and $u_{\mathcal{N}}$ concerning their references $x_{r_{\mathcal{N}}}$ and $u_{r_{\mathcal{N}}}$, respectively, over a future window of the prediction horizon length $N_p$, as it is typically selected for MPC control. Mathematically, it is stated as follows:

$$\begin{aligned} J_{\mathcal{N}}\left(x_{\mathcal{N}}(t), U_{\mathcal{N}}^f(t)\right) = \\ = \sum_{k=0}^{N_p-1} \left(x_{\mathcal{N}}(t+k) - x_{r_{\mathcal{N}}}(t+k)\right)^{\mathsf{T}} Q_{\mathcal{N}} \\ \left(x_{\mathcal{N}}(t+k) - x_{r_{\mathcal{N}}}(t+k)\right) + \\ + \sum_{k=0}^{N_p-1} \left(u_{\mathcal{N}}(t+k) - u_{r_{\mathcal{N}}}(t+k)\right)^{\mathsf{T}} R_{\mathcal{N}} \\ \left(u_{\mathcal{N}}(t+k) - u_{r_{\mathcal{N}}}(t+k)\right) \end{aligned} \tag{4}$$

were $Q_{\mathcal{N}}$ and $R_{\mathcal{N}}$ matrix weights are detailed in [6].

## III. Methodology

This section summarizes the algorithm used to train the negotiation agent for consensus, where the agent is approximated by a DNN with parameters $\theta$ that represents the actor in the DDPG algorithm. At the beginning of training, the critic DNN with Q-value function $Q(s,a;\phi)$ and the target critic DNN with Q-value function $Q_t(s,a;\phi_t)$ are initialized with the same random values $\phi_t = \phi$; in the same way, the actor $\pi(s;\theta)$ and the target actor $\pi_t(s;\theta_t)$ are also initialized with the same random parameters $\theta_t = \theta$. Training is made up of sequential episodes, each one comprising a number of time steps, until some convergence is achieved. A time step $t$ represents the interaction $s_t$-$a_t$-$r_t$ (current state, action given by the agent, and reward obtained). In the methodology proposed, the state is selected as a vector including all the current available control sequences for all local agents, where $t$ dependence has been omitted for brevity,

$$s_t = [U_1^{f_1}, U_1^{f_2}, ..., U_1^{f_{M_1}}, ..., U_l^{f_1}, U_l^{f_2}, ..., U_l^{f_{M_l}}]$$

Next, the action $a_t = \pi(s_t;\theta) + \xi$ is calculated using the current policy, where $\xi$ is a noise value from the Ornstein-Uhlenbeck noise model. The action $a_t$ is composed of the set of all negotiation coefficients,

$$a_t = [a_1^{f_1}, a_1^{f_2}, ..., a_1^{f_{M_1}}, ..., a_l^{f_1}, a_l^{f_2}, ..., a_l^{f_{M_l}}]$$

to give a specific weight to the available control sequences of each local agent. Therefore, the final sequences are obtained as:

$$U_l^f(t) = \textstyle\sum_{m=1}^{M_l} U_l^{f_m}(t) \cdot a_l^{f_m}$$

where $l \in \{1, ..., N\}$. Then, the first control signals of each sequence $U_l^f$ (as usual in MPC control) are implemented in the environment, composed of the system and the DMPC based control architecture with fuzzy negotiation, which generates the reward $r_t$ and the next state $s_{t+1}$. From the experience buffer that contains the stored experiences $(s_t, a_t, r_t, s_{t+1})$ a random mini-batch of $Z$ experiences $(s_t^E, a_t^E, r_t^E, s_{t+1}^E)$ is sampled. Then, the value function target $y_t$ is computed as the sum of the experience reward and discounted future reward:

$$y_t^E = r_t^E + \gamma Q_t(s_{t+1}^E, \pi_t(s_{t+1}^E); \phi_t). \tag{5}$$

The cumulative reward is calculated as follows. Firstly, the target actor selects the next action based on observation $s_{t+1}^E$ of the sampled experience. Secondly, the cumulative reward is obtained from $s_{t+1}^E$ and the action given by the target actor. Next, the critic parameters are updated minimizing the following function over all sampled $Z$ experiences,

$$L = \frac{1}{2Z}\sum_{E=1}^{Z}(y_t^E - Q(s^E, a^E; \phi))^2 \tag{6}$$

On the other hand, the parameters $\theta$ of the actor are updated through the sampled policy gradient applying the chain rule to the expected discounted reward to maximize its value, giving the following expression [11],

$$\frac{1}{Z}\sum_{E=1}^{Z}\nabla_a Q(s^E, a; \phi)\nabla_\theta \pi(s^E; \theta) \tag{7}$$

which includes the product of gradient of the critic output with respect to the action computed by the actor network and the gradient of the actor output with respect to the actor parameters, both gradients evaluated for $s^E$. Finally, a smoothing method at every time step with smoothing factor $\tau$ updates the target actor parameters $\theta_t = \tau\theta + (1-\tau)\theta_t$ and the target critic parameters $\phi_t = \tau\phi + (1-\tau)\phi_t$.

## IV. Case study

The plant is composed of eight interconnected water tanks (Fig. 2), where the upper tanks are the #3/4/7/8 tanks, which discharge into the lower tanks #1/2/5/6, which in turn discharge into large sinks that feed the pumps. The plant works by means of four pumps whose flows are distributed through the system with six three-way manual valves $\gamma_v$ with $v \in \{1, 2, ..., 6\}$.

The four subsystems that compose the plant are grouped as shown in Fig. 2: #1/3 as subsystem 1; #2/4 as subsystem 2; #5/7 as subsystem 3; and #6/8 as subsystem 4. The water level of the tank $n \in \{1, 2, ..., 8\}$ is expressed as $h_n$. The levels of the lower tanks are the controlled variables, $h_1$, $h_2$, $h_5$ and $h_6$, by the agents 1, 2, 3 and 4, respectively. The manipulated variables $q_1$, $q_2$, $q_3$ and $q_4$ are the pump flows for each subsystem. On the other hand, the operation point $h_n^p$ is $h_1^p = 0.10$, $h_2^p = 0.15$, $h_5^p = 0.10$, $h_6^p = 0.15$ for the controlled levels and $h_3^p = 0.07$, $h_4^p = 0.03$, $h_7^p = 0.025$, $h_8^p = 0.10$ for the upper levels, all expressed in meters. The operating point $q_l^p$ for the pumps flow rates are $q_1^p = 0.142$, $q_2^p = 0.421$, $q_3^p = 0.421$, $q_4^p = 0.140$, expressed in $m^3/h$.

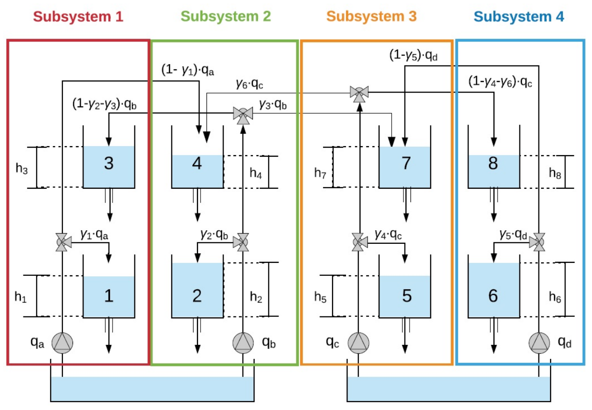

Fig. 2. Graphic representation of the system composed of eight interconnected water tanks divided into four subsystems [6].

The objective as a control tracking problem is to drive the state vector, $\widetilde{x}_{\mathcal{N}} = \left[h_n(t) - h_n^p\right]^\intercal$, to the reference state vector, $\widetilde{\mathcal{T}}_{\mathcal{N}} = \left[h_n^{\mathcal{T}}(t) - h_n^p\right]^\intercal$, where $h_n^{\mathcal{T}}$ is the references vector. The input vector is defined as $\widetilde{u}_{\mathcal{N}} = \left[q_l(t) - q_l^p\right]^\intercal$, where $q_l$ is the first value of the sequence $U_l^f(t)$. Moreover, the constraints on the state and input vectors are:

$$-h_n^p < \tilde{x}_n(t) \leq 0.08, \; -q_l^p < \tilde{u}_l(t) \leq 0.04,$$
$$\forall n \in \{1, 2, ..., 8\}, \; \forall l \in \{1, 2, 3, 4\}.$$

## V. Deep neural network negotiation framework

This section defines the environment and the state-action-reward for the RL method implemented as the negotiation agent in the distributed MPC framework. The environment is made up of the eight interconnected water tanks plant and the low-level control layer with the architecture based on cooperative DMPC. The DDPG agent will interact with this environment until the parameters of the actor DNN maximize the expected long-term reward. In order to simplify the negotiation procedure, the state $s_t$ is formed only by the first value of the control sequences to be negotiated in the upper-level control layer. In particular, for the subsystems *2* and *3*, two values are available for each subsystem, denoted as follows:

$$s_t = [u_2^{f_1}(t), u_2^{f_2}(t), u_3^{f_1}(t), u_3^{f_2}(t)].$$

The negotiation in the upper control layer will provide a specific weight to each of them in order to obtain the final control sequences for subsystems *2* and *3*. Therefore, the actor DNN outputs $a_t$ (actions), are two weighing coefficients

$$a_t = [a_2^{f_1}(t), a_3^{f_1}(t)] \text{ where } a_t \in [0,1].$$

The weighting coefficients for the other control sequences are calculated as follows because their total sum is 1:

$$\begin{aligned} a_2^{f_2}(t) &= 1 - a_2^{f_1}(t) \\ a_3^{f_2}(t) &= 1 - a_3^{f_1}(t) \end{aligned} \tag{8}$$

The negotiation process to obtain the final control sequences, $U_2^f(t)$ and $U_3^f(t)$, employs the coefficients previously obtained:

$$\begin{aligned} U_2^f(t) &= U_2^{f_1}(t) \,.\, a_2^{f_1}(t) + U_2^{f_2}(t) \,.\, a_2^{f_2}(t) \\ U_3^f(t) &= U_3^{f_1}(t) \,.\, a_3^{f_1}(t) + U_3^{f_2}(t) \,.\, a_3^{f_2}(t) \end{aligned} \tag{9}$$

Note that no negotiation is required for subsystems #1 and #4 because they only have one neighbor, so $M_1 = M_4 = 1$. The objective of training any RL agent is to maximize the expected long-term reward, and taking into account that in our tracking control context the objective is to minimize the overall cost function Eq. 4, the reward $r_t$ is defined as:

$$r_t = -\left(J_{\mathcal{N}}(t)\right)^2 \tag{10}$$

## VI. Negotiation agent training

### A. Deep neural networks: Critic-Actor

The critic DNN provides the Q-value that indicates the reward expectations based on a state and the subsequent action. Consequently, the input of the critic DNN is divided into two paths: one for the state and one for the action. Both entries converge on a main path, resulting in a DNN specified as follows: The state path is composed of a Feature input layer (4 neurons), three Fully Connected Layer (FCL) as hidden layers; and the action path composed of a Feature input layer (2 neurons), one FCL; and the main path composed of two FCL, a ReLu function as activation function of the hidden layers and a FCL as output layer (1 neuron).

On the other hand, the DNN of the actor will determine the action that is given depending on the state. Hence, it is composed of a Feature input layer (4 neurons), three FCL as hidden layer, a ReLu function as activation function of the hidden layers, a FCL as output (2 neurons), a sigmoid function as activation function and an scaling layer (Scale=[0.8;0.8], Bias=[0.05;0.05]).

### B. Hyper-parameters

One of the challenges of RL framework when faced with large spaces of continuous actions in complex non-linear environments is to find the adequate values for the training hyper-parameters in the search of convergence towards the optimal policy. The critic learning rate = $10^{-4}$, actor learning rate = $10^{-5}$, variance exploration noise = [0.01 0.01] under Ornstein-Uhlenbeck noise values were obtained after multiple tuning iterations; the gradient threshold for the actor and critic is equal to 1, and the mini-batch training size $Z = 100$. On the other hand, the training consisted of 1000 episodes of 200 time steps in MATLAB with the Reinforcement Learning Toolbox. Due to difficulties for achieving convergence, each episode consists on a simulation keeping the same starting state and reference, provided that operation is around this state. Fig. 3 displays the evolution of the training with the average reward and the Q-value until policy convergence.

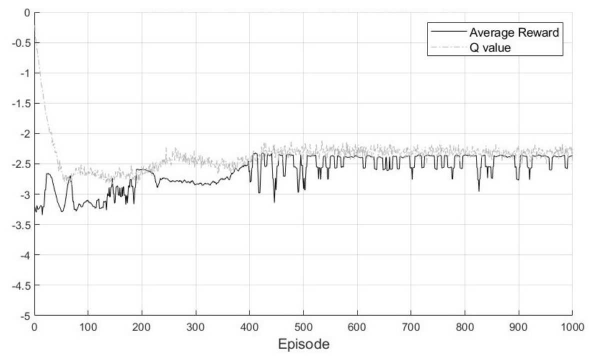

Fig. 3. Training of the negotiation agent: Average reward and Q-value for 1000 training episodes.

## VII. Validation in simulation and application to a real plant

The validation has been carried out under a reference tracking objective during $T = 600$ time steps in simulation and $T = 700$ time steps in the real plant, where there is a first reference step $\widetilde{\mathcal{T}_{\mathcal{N}}} = \widetilde{x}_{\mathcal{N}} + 0.2$ (meters) applied at $t = 1$ and a second reference step is $\widetilde{\mathcal{T}_{\mathcal{N}}} = \widetilde{x}_{\mathcal{N}} + 0.1$ (meters) applied at $t = 400$. All control sequences given by the DMPC are obtained with prediction horizon $N_p = 20$. This section also includes for comparison the results of a fuzzy logic based upper negotiation layer (FL), as detailed in [6].

The following metrics are defined for comparison: the sum of the integral squared errors, $ISE = \sum ISE_n$ where $ISE_n = \int_0^T (h_n^{\mathcal{T}} - h_n)^2 dt$, for $n \in \{1,2,5,6\}$ concerning the controlled levels, and the total global cost function (Eq. 4) along the evaluation period, $J = \sum_0^T J_{\mathcal{N}}\left(x_{\mathcal{N}}(t), U_{\mathcal{N}}^f(t)\right)$.

### A. Validation in simulation

According to Fig. 4, the difference between the water levels obtained with the upper-layer negotiation based on fuzzy logic (FL) and the proposed negotiation of this work (RL) is small, both showing good tracking behavior, with $ISE_{RL} = 0.0340$ and $ISE_{FL} = 0.0317$. For the corresponding first value of the final control sequences (Fig. 5), a certain difference is observed between those obtained with FL and RL negotiation. The values of the total global cost function (Eq. 4) are $J_{RL} = 1.5993$ and $J_{FL} = 1.3656$.

The final control sequences $U_2^f(t)$ and $U_3^f(t)$ are a consequence of the consensus between the first value of the control sequences shown in Fig. 6, in which a big difference is observed in transients between the sequences that have to negotiate; $U_2^{f1}(t)$ with $U_2^{f2}(t)$, and $U_3^{f1}(t)$ with $U_3^{f2}(t)$.

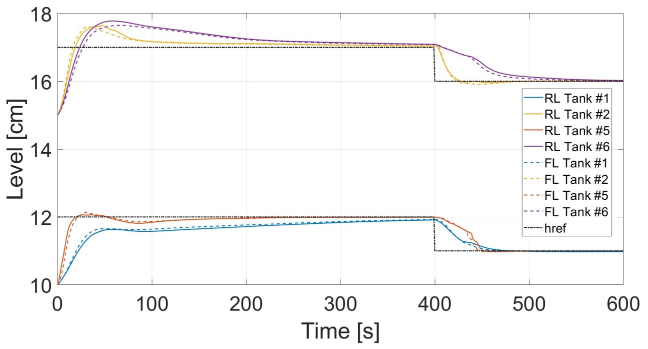

Fig. 4. Levels of controlled tanks #1-2-5-6, under fuzzy logic based negotiation (FL) and reinforcement learning based negotiation (RL).

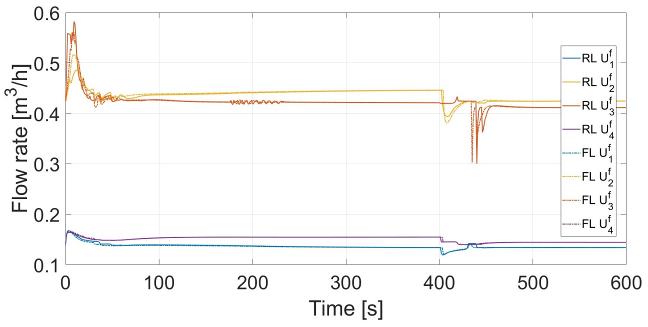

Fig. 5. Evolution of the first value of the final control sequences $U_1^f(t)$, $U_2^f(t)$, $U_3^f(t)$ and $U_4^f(t)$, after FL and RL negotiation.

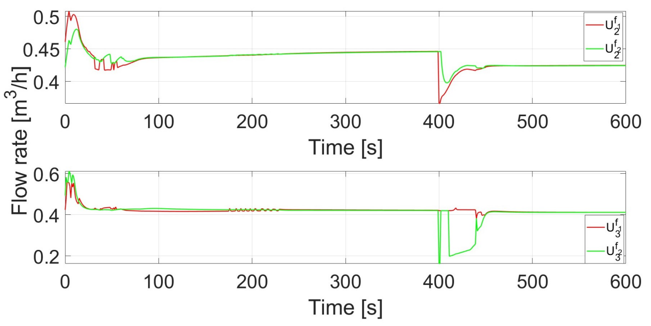

Fig. 6. Evolution of the first value of the control sequences for consensus, which form the state $s_t$ for RL negotiation: $u_2^{f1}(t)$, $u_2^{f2}(t)$, $u_3^{f1}(t)$ and $u_3^{f2}(t)$

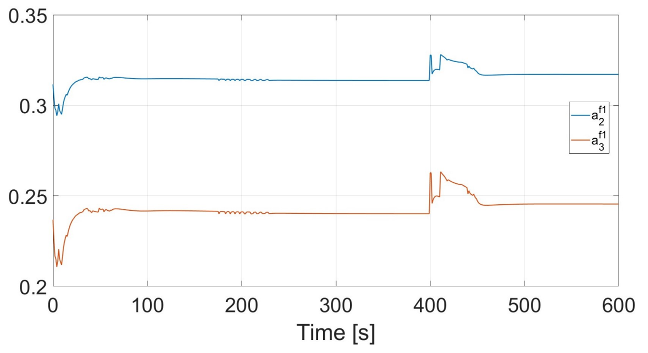

Fig. 7. Evolution of the negotiation coefficients provided by the DNN, $a_t = \left\{a_2^{f1}, a_3^{f1}\right\}$

Fig. 7 shows the negotiation coefficients given by the actor DNN output $a_t = \left\{a_2^{f1}, a_3^{f1}\right\}$, that will be used to calculate the other two weighting coefficients using Eq. 8, and with all of them the final control sequences. The coefficients vary over time depending on the discrepancy between the negotiation control values (Fig. 6), showing constant behavior when there is no conflict between the control values involved in the negotiation.

## B. Application to a real plant

In this section, some results of the methodology applied to a real eight-coupled tanks lab plant are shown, considering the same cooperative DMPC-based control architecture in the lower layer, and the same tracking problem used for simulation results, except for the evaluation time $T = 700$ steps. The evolution of the levels in the controlled tanks (Fig. 8), shows that both negotiation upper layers perform a good reference tracking, with $ISE_{RL} = 0.0544$ and $ISE_{FL} = 0.0511$, where $ISE_n = \int_{t=5}^{T} (h_n^{\mathcal{T}} - h_n)^2 dt$. As for the first value of the final control sequences (Fig. 9), some differences can be observed between FL and RL, which are larger than in simulation results. The values of the total global cost function (Eq. 4) are $J_{RL} = 1.3913$ and $J_{FL} = 0.9576$, being $J = \sum_{t=5}^{T} J_{\mathcal{N}}$. Regarding the first value of the sequences to be agreed upon (Fig. 10), discrepancies are observed between the pairs involved in negotiation. The DNN outputs of the negotiation agent (Fig. 11) change over time depending on the variations in the sequences to be negotiated, as expected.

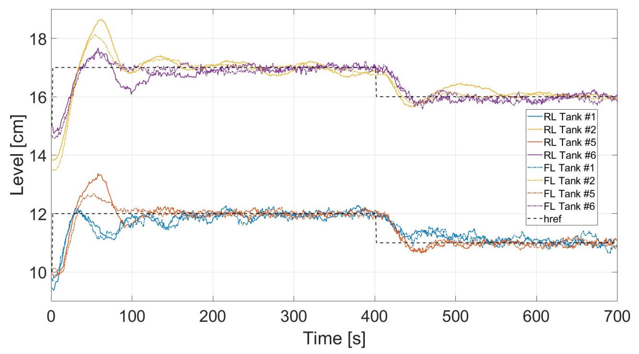

Fig. 8. Levels of controlled tanks #1-2-5-6, under fuzzy logic based negotiation (FL) and reinforcement learning negotiation (RL).

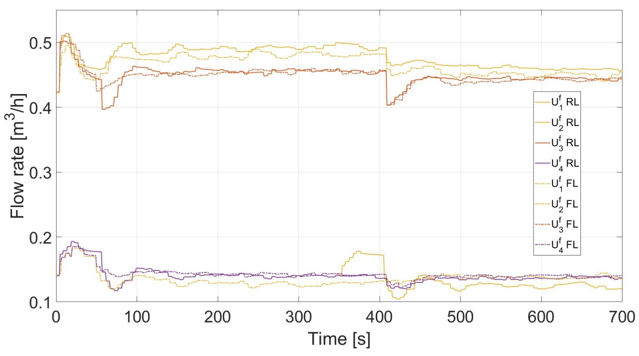

Fig. 9. Evolution of the first value of the final control sequences $U_1^f(t)$, $U_2^f(t)$, $U_3^f(t)$ and $U_4^f(t)$, after FL and RL negotiation

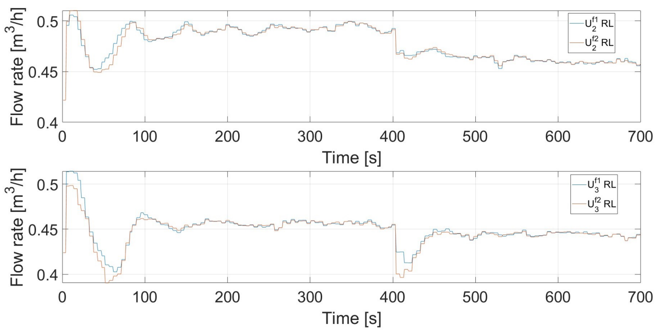

Fig. 10. Evolution of the first value of the control sequences for consensus, which form the state $s_t$: $u_2^{f1}(t)$, $u_2^{f2}(t)$, $u_3^{f1}(t)$ and $u_3^{f2}(t)$.

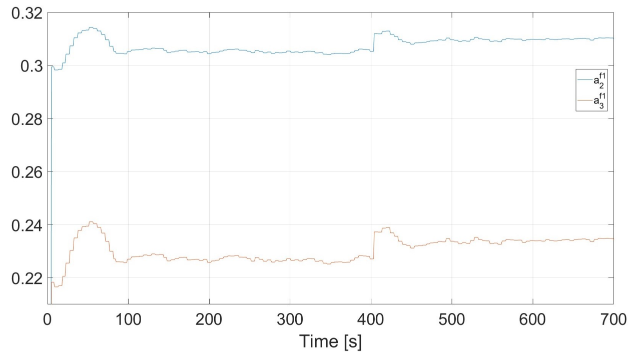

Fig. 11. Evolution of the negotiation coefficients provided by the DNN, $a_t = \left\{a_2^{f1}, a_3^{f1}\right\}$.

## VIII. Conclusion

In this work, we have successfully applied a negotiation agent trained by DRL methodology in a real plant made up of eight interconnected water tanks. The agent participates in the upper control layer of a cooperative DMPC with fuzzy logic negotiation among local agents. It should be noted that the similarity between the simulation model and the real plant means that the sequences to be negotiated during training are similar to the sequences to be negotiated during application in the real process. This is important because the agent's actions during the real application must be governed by the same objective imposed during training. The tracking validation in the real plant shows good results despite relevant plant model mismatch due to unmodeled issues, noisy measurements and other unmeasured disturbances. This plant model mismatch is also responsible for the existing differences between simulation and experimental results.

On the other hand, the big challenge of tuning the hyperparameters for training the negotiation agent has motivated the consideration of environment simulations with fixed conditions to ease the training convergence. The results for negotiation are good despite of the mentioned issue because the operating conditions are in the region chosen for training.

The proposed RL negotiation methodology has been compared to FL negotiation, showing no significant improvement in the current operating conditions, but further work in other situations involving more transients would increase the benefits. Anyway, the employed negotiation agent has the advantage of not requiring any prior knowledge of the system and no more communication steps than receiving the sequences to be negotiated from the DMPC local agents.

## References

[1] J. M. Maestre and R. R. Negenborn, Eds., *Distributed Model Predictive Control Made Easy*, ser. Intelligent Systems, Control and Automation: Science and Engineering. Springer Netherlands, 2014, vol. 69.

[2] R. S. Sutton and A. G. Barto, *Reinforcement Learning, second edition: An Introduction*. MIT Press, 2018.

[3] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," vol. 34, no. 6, pp. 26–38, 2017, IEEE Signal Processing Magazine.

[4] K. Johansson, "The quadruple-tank process: a multivariable laboratory process with an adjustable zero," vol. 8, no. 3, 2000, pp. 456–465, IEEE Transactions on Control Systems Technology.

[5] M. Francisco, Y. Mezquita, S. Revollar, P. Vega, and J. F. De Paz, "Multi-agent distributed model predictive control with fuzzy negotiation," vol. 129, pp. 68–83, expert Systems with Applications.

[6] E. Masero, M. Francisco, J. M. Maestre, S. Revollar, and P. Vega, "Hierarchical distributed model predictive control based on fuzzy negotiation," vol. 176, p. 114836, 2021, expert Systems with Applications.

[7] O. Aponte-Rengifo, P. Vega, and M. Francisco, "Deep reinforcement learning agent for negotiation in multi-agent cooperative distributed predictive control," *Applied Sciences*, vol. 13, no. 4, p. 2432, 2023.

[8] I. Alvarado, D. Limon, D. Muñoz de la Peña, J. M. Maestre, M. A. Ridao, H. Scheu, W. Marquardt, R. R. Negenborn, B. De Schutter, F. Valencia, and J. Espinosa, "A comparative analysis of distributed MPC techniques applied to the HD-MPC four-tank benchmark," vol. 21, no. 5, 2011, pp. 800–815, journal of Process Control.

[9] L. Orihuela, P. Millán, C. Vivas, and F. R. Rubio, "Suboptimal distributed control and estimation: application to a four coupled tanks system," vol. 47, no. 8, pp. 1755–1771, international Journal of Systems Science.

[10] X. Meng, H. Yu, J. Zhang, T. Xu, and H. Wu, "Liquid level control of four-tank system based on active disturbance rejection technology," vol. 175, p. 109146, 2021, measurement.

[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," 2015, arXiv preprint. arXiv:1509.02971.

[12] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," vol. 518, no. 7540, pp. 529–533, 2015, nature.

[13] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proceedings of the 31st International Conference on Machine Learning*. PMLR, 2014, pp. 387–395.