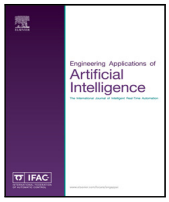
Engineering Applications of Artificial Intelligence 111 (2022) 104769

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Wind turbine pitch reinforcement learning control improved by PID regulator and learning observer

J. Enrique Sierra-Garcia [a,*], Matilde Santos [b], Ravi Pandit [c]

[a] Electromechanical Engineering Department, University of Burgos, 09006 Burgos, Spain
[b] Institute of Knowledge Technology, Complutense University of Madrid, 28040 Madrid, Spain
[c] Faculty of Science & Engineering, Anglia Ruskin University, Chelmsford, UK

ARTICLE INFO

Keywords:
Intelligent control
Reinforcement learning
Learning observer
Pitch control
Wind turbines

ABSTRACT

Wind turbine (WT) pitch control is a challenging issue due to the non-linearities of the wind device and its complex dynamics, the coupling of the variables and the uncertainty of the environment. Reinforcement learning (RL) based control arises as a promising technique to address these problems. However, its applicability is still limited due to the slowness of the learning process. To help alleviate this drawback, in this work we present a hybrid RL-based control that combines a RL-based controller with a proportional–integral–derivative (PID) regulator, and a learning observer. The PID is beneficial during the first training episodes as the RL based control does not have any experience to learn from. The learning observer oversees the learning process by adjusting the exploration rate and the exploration window in order to reduce the oscillations during the training and improve convergence. Simulation experiments on a small real WT show how the learning significantly improves with this control architecture, speeding up the learning convergence up to 37%, and increasing the efficiency of the intelligent control strategy. The best hybrid controller reduces the error of the output power by around 41% regarding a PID regulator. Moreover, the proposed intelligent hybrid control configuration has proved more efficient than a fuzzy controller and a neuro-control strategy.

1. Introduction

Coal-fired power plants have been identified as one of the significant causes of climate change. Although CO2 emissions are mainly produced by thermal power plants, these energy sources are still widely used nowadays. This also impacts not only the environment but also the health of the people. Indeed, each year more than 500,000 asthma episodes are estimated to be caused by air pollution worldwide (Green Peace, 2021). As a result, there is a widespread general consensus that renewable energy sources such as wind, hydro, and solar must be considered to mitigate climate change and reduce air pollution.

Year after year, wind energy usage is on the rise worldwide. After hydroelectric, it is currently the second most widely used renewable energy source (Our World in Data, 2020). Moreover, this upward trend appears to be continued until 2050 when it is expected to become the primary source of energy (IRENA, 2019). Consequently, research on wind energy and, particularly, on the efficiency of Wind Turbine (WT) control is encouraged to contribute this way to this sustainable trend.

The performance of a WT fully depends upon control systems applied on the turbine side and generator side. Different advanced and intelligent control strategies have been applied to these strongly non-linear and multi-input multi-output (MIMO) systems (Yang et al., 2016). From the control engineering point of view, several control actions can be identified in WT. The control of the pitch angle aims to stabilize the output power when the wind speed overpasses the rated wind speed. This is carried out by varying the pitch angle that changes the surface of the blades that the wind faces. The rotor angular speed is usually controlled by injecting current to the generator to seek the optimal power curve. Lastly, the yaw angle control modifies the orientation of the nacelle to match the main wind stream.

The control of the pitch angle of a WT is challenging due to the non-linearities of the system, the coupling of the variables, and the uncertainty of the measures. This control can be crucial for floating offshore wind turbines (FOWT), which are subjected to heavy external loads from mainly wind and waves that cause fatigue and vibrations. Even more, the pitch control action may affect the stability of the turbine (Tomás-Rodríguez and Santos, 2019). This has led to the application of artificial intelligent techniques to address this complex control problem (Sierra-García and Santos, 2021b). Among them, reinforcement learning (RL) is starting to be considered in this field. By the application of RL techniques, the control law can be improved online, reacting to changes in the system, facing uncertainties and disturbances. However, the learning process is usually prolonged, and

* Corresponding author.
E-mail address: jesierra@ubu.es (J.E. Sierra-Garcia).

https://doi.org/10.1016/j.engappai.2022.104769
Received 7 November 2021; Received in revised form 4 February 2022; Accepted 13 February 2022
Available online 4 March 2022

this strongly limits its applicability in control applications. Another drawback of the RL approach is that the controller performance is not that good during the initial learning phase, and the results may be pretty random due to it lacking enough experience to learn from.

In this work, a novel approach for WT pitch control based on a modified and enhanced RL strategy is proposed. On the one hand, RL architecture is complemented with a conventional proportional–integral–derivative (PID) controller. This PID regulator helps accelerate the training of the learning process. Indeed, its contribution has proved especially important during the first training episodes while the RL based control does not have any experience to learn from. On the other hand, the learning process is monitored by a learning observer. If it detects the system is no longer learning, it increases the exploration rate and enlarges the new defined exploration window. This allows us to consider new actions in a supervised way.

The new concept of the exploration window and its utility is introduced. In the RL process, when a random action is explored, instead of selecting an action among all the possible ones, the action is selected from a reduced set of activities within an exploration window defined around the best action. The action selector oversees this exploration window, and its size is determined by the learning observer. This method reduces the oscillations during the training and accelerates the learning performance.

The proposed hybrid control architecture based on RL has been evaluated and compared with a pure RL controller without the PID and with a conventional well-tuned PID controller. Besides, other intelligent control methods, such as fuzzy logic and neural networks-based controllers, have been applied to the same turbine for comparison purposes. The simulation results show that the learning process is speeded up with this new control configuration, and the oscillations are reduced during the training. The performance of the learning observer has also been compared against fixed epsilon and exponential epsilon decay exploitation strategies providing better results. This also improves the control of the WT as the controller learns faster to adapt to different wind profiles and speed operation ranges. Finally, the performance of the discrete RL (DRL) controller and its combinations with a P-controller or a PD-controller is better than the fuzzy controller and the neural controller.

In conclusion, the main contributions of this work can be summarized as follows:

- The development of a hybrid intelligent control architecture that combines the RL approach and the classical PID controller in a way that they complement each other. The RL is able to improve the system control performance as it learns and adapts the control to face uncertainties and disturbances. On the other hand, the PID helps accelerate the learning process of the control strategy. Thus, the hybrid controller results better than the pure RL or the best tuned PID, especially during the training phase.
- The inclusion of a learning observer in the control architecture. In RL, if the exploration rate is very large, the learning slows down; however, the risk of falling into local minima is big if it is very small. To find an adaptive balance, the learning observer monitors the learning process and adjusts the exploration rate and the exploration window in a dynamic way.
- The concepts of action selector, exploration window and their utility are introduced. In standard RL, the random actions are selected among all possible actions. However, in this approach the action is selected from a reduced set of actions within an exploration window defined around the best action. The action selector oversees this exploration window. This method helps reduce the oscillations during the training and accelerate the learning.

The rest of the paper is organized as follows. Section 2 presents a brief state of the art. In Section 3, the mathematical model of the WT used is described. The design of the improved RL controller and its components are presented in Section 4. Simulation results and comparisons with other intelligent control methods are discussed in Section 5. The paper ends with the conclusions and future works.

2. Related works

Intelligent control techniques such as fuzzy logic, neural networks, genetic algorithms, and RL have been successfully used to model and control engineering complex systems in different fields (Tzafestas, 2012; Sierra-García and Santos, 2021c; Santos, 2011; Trojaola et al., 2020).

There are some recent papers related to WT control inspired by RL. To mention some of them, an updated summary of deep RL for power system applications is given in Zhang et al. (2019). RL is used to regulate variable-speed WT in Fernandez-Gauna et al. (2017). It adapted traditional variable speed WT controllers to changing wind conditions in particular. The same authors used conditioned RL to investigate this difficult control scenario with vast state–action spaces (Fernandez-Gauna et al., 2018). A doubly-fed induction generator WT is online controlled using a policy iteration RL model and an adaptive actor-critic technique in Abouheaf et al. (2018). An RL-based artificial neural network for WT yaw control is presented in Saénz-Aguirre et al. (2019). The authors proposed a performance upgrade of this WT neuro–RL yaw control in a more recent study (Saenz-Aguirre et al., 2020). Tomin et al. presented adaptive control strategies in which a trained RL agent is used to extract the stochastic property of wind speed, and then the optimal policy is applied to the WT adaptive control design (Tomin et al., 2019). Passive RL solved by particle swarm optimization (PSO) policy was used by Hosseini et al. to control the pitch angle of a real WT using an adaptive type-2 neuro-fuzzy inference system with unsupervised clustering (Hosseini et al., 2020). Chen et al. also proposed a resilient WT controller based on RL and system status data that used adaptive dynamic programming (Chen et al., 2020). Deep RL and knowledge assisted learning were used to deal with the wake effect in a cooperative wind farm control in a related challenge (Zhao et al., 2020). Multi-agent deep RL has been recently applied to the energy field, mainly in cogeneration, to address coordinated problems within a multi-area system (Li et al., 2021a,b, 2022).

The use of hybrid intelligent controllers in WT has given good results; in many of the cases, the intelligent techniques have been combined with traditional PID regulators or with other conventional control solutions. An example is found in Iqbal et al. (2020), where the authors proposed a hybrid control system that merged a fuzzy system and a traditional model predictive controller. The goal of the fuzzy-based model-predictive controller of the pitch angle control was to reduce WT loads while increasing extracted power production. The Fuzzy Logic Controller (FLC) is particularly effective at dealing with the system non-linearity, whilst the predictive model controller aids stability and efficiency. A WT pitch PID-controller with parameters tuned using a fuzzy logic system was presented in Ngo et al. (2020). A small WT was used to test the method effectiveness. In addition, Sedighizadeh and Rezazadeh designed a RL-tuned adaptive PID controller for the same control objective (Sedighizadeh and Rezazadeh, 2008).

To track the maximum power, Sitharthan et al. proposed a hybrid maximum power point tracking (MPPT) control technique that predicts the effective wind speed and the ideal rotor speed of the wind power system. A radial basis function neural network was used, which was improved with a PSO algorithm (Sitharthan et al., 2020). Sarkar et al. 2020 developed a robust pitch control system for the rated WT power using PID controller for a wide range of simulated wind speeds (Sarkar et al., 2020). In addition, ant colony optimization, PSO, and classic Ziegler–Nichols algorithms have been applied to tune the PID controller parameters in order to achieve a steady output power within rated limits with variable wind speeds.

There are few articles on intelligent pitch management of large-scale WT. Rubio et al. described a fuzzy-logic-based pitch control system for

a 5 MW WT built on a semi-submersible platform (Rubio et al., 2019). The OC4 WT model was used to test it. The instantaneous value of the wind speed, filtered and normalized according to the nominal speed, is sent to the fuzzy controller, whose output is the pitch reference. For a variable speed WT, Abdelbaky et al. suggested restricted fuzzy-receding horizon pitch control (Abdelbaky et al., 2020). The model is transformed into a simple online quadratic optimization problem that requires less computational time to solve, and the controller ensures nominal stability. To test the mathematical model results, a 5MW offshore WT is used.

A fuzzy control that considers as only input the wind speed was designed and simulated in Santoso et al. (2021). The fuzzy output was the blade angle. It was tested with an input random wind speed with a range of 7–20 m/s in a 20 kW WT. Jeon et al. presented a linear quadratic regulator based on fuzzy logic for the control of a variable-speed variable-pitch WT (Jeon and Paek, 2021). Simulations and wind tunnel tests were conducted. The main point of this proposal is that non only this control increases the power performance but also the structural stability of the WT compared with conventional PI control. Sahoo et al. also compared a PID, a FLC and fuzzy PID control to smooth the output power fluctuations of a WT by means of the pitch angle (Sahoo and Panda, 2022). An incremental PD-type fuzzy controller to generate the pitch angle reference of a large WT was designed in Serrano-Barreto et al. (2021). The performance of this control scheme on the NREL 5 MW floating offshore WT was compared with the internal GS-PI control that is provided within the FAST software with satisfactory results.

Neural networks have been also used in wind energy but mainly applied to fault diagnosis. This may be due to the lack of data that are necessary to train the network. In any case, some exceptions –though scarce– can be found in the literature, such as the following. Salem et al. implemented a pitch angle control system to control the power output of a small turbine at wind speeds above the rated speed (Salem et al., 2021). They used the neural network fitting function, using the relation between the power output from the WT model and the wind velocity to select the pitch angle. They implemented a feed-forward network with sigmoid hidden neurons and linear output neurons for the Power Coefficient (Cp) model at different rated power. An artificial neural network based settings with only pitch control, only voltage controllers, and with both pitch and voltage controllers was designed in Pamuji et al. (2021). The output of the pitch controller was the blade angle reference and the inputs the Cp and the Tip Speed Ratio (TSR), obtained from simulated models. Similarly, Reddak et al. designed an artificial neural network to implement a MPPT-pitch angle control strategy for controlling the WT (Reddak et al., 2021). Quite interestingly, Fan et al. simulated a new non-linear hybrid control approach based on the Adaptive Neuro-Fuzzy Inference System (ANFIS) and fuzzy logic control to regulate the pitch angle and maintain the captured mechanical energy at the rated value of a WT (Fan et al., 2021). In the controller, the reference value of the pitch angle was predicted by ANFIS according to the wind speed and the blade tip speed ratio. The proposed FLC provided feedback based on the captured power to modify the pitch angle in real-time. The effectiveness of the proposed hybrid pitch angle control method was verified on a 5 MW offshore WT under two different wind conditions. Finally, other ANFIS blade pitch control was proposed in Elsisi et al. (2021). However, as ANFIS requires a suitable dataset for training and testing, the paper also suggested an effective strategy to prepare a sufficient dataset using a new so-called mayfly optimization algorithm.

As it has been shown in state of the art, the works that use RL for WT pitch control are very limited. Hybrid controllers also appear as a possible solution for dealing with the complex challenges of WT control. The main difference between our proposed intelligent hybrid method and published methods is the combination of the RL approach with a PID to improve the performance, especially during the initial phase of the learning, and the implementation of a learning observer to handle the exploration rate dynamically.

Table 1
Parameters of the wind turbine model.

Parameter	Description	Value/Units
L_a	Inductance of the armature	13.5 mH
K_g	Constant of the generator	23.31
K_ϕ	Magnetic flow coupling constant	0.264 V/rad/s
R_a	Resistance of the armature	0.275 Ω
R_L	Resistance of the load	8 Ω
J	Inertia	6.53 kg m^2
R	Radius of the rotor	3.2 m
ρ	Density of the air	1.223 kg/m^3
K_f	Friction coefficient	0.025 N m/rad/s
$[c_1, c_2, c_3]$	C_p constants	[0.73, 151, 0.58]
$[c_4, c_5, c_6]$	C_p constants	[0.002, 2.14, 13.2, 18.4]
$[c_7, c_8, c_9]$	C_p constants	[18.4, −0.02, −0.003]
$[K_\theta, T_\theta]$	Pitch actuator parameters	[0.15, 2]
$[\alpha, \beta, \gamma, \tau]$	Blade filter constants	[0.55, 0.832, 1.17, 9]

3. Mathematical model of the wind turbine

In this work, a 7 kW WT is used as a basis. The mathematical model of this small WT is given by Eqs. (1)–(9). Its representation is shown in Fig. 1. The development of these expressions is further explained in Mikati et al. (2012), Sierra-García and Santos (2020b).

$$\ddot{\theta} = \frac{1}{T_\theta}\left[K_\theta\left(\theta_{ref} - \theta\right) - \dot{\theta}\right], \tag{1}$$

$$f_{blade}(s) = \frac{\beta \cdot s + \sqrt{2}}{\beta^2 \cdot s^2 \left(\sqrt{\left(\frac{2}{\alpha}\right)} + \sqrt{\alpha}\right) \cdot \beta \cdot s + \sqrt{2}} \cdot \frac{\gamma \cdot s + 1/\tau}{s + 1/\tau} \tag{2}$$

$$v_{ef} = f_{blade}(v_W) \tag{3}$$

$$\lambda = (w \cdot R)/v_{ef} \tag{4}$$

$$\lambda_i = \left[\left(\frac{1}{\lambda + c_8}\right) - \left(\frac{c_9}{\theta^3 + 1}\right)\right]^{-1} \tag{5}$$

$$C_p(\lambda_i, \theta) = c_1\left[\frac{c_2}{\lambda_i} - c_3\theta - c_4\theta^{c_5} - c_6\right] e^{-\frac{c_7}{\lambda_i}}, \tag{6}$$

$$\dot{w} = \frac{1}{2 \cdot J \cdot w}\left(C_p(\lambda_i, \theta) \cdot \rho\pi R^2 \cdot v_{ef}^3\right) - \frac{1}{J}\left(K_g \cdot K_\phi \cdot I_a + K_f \cdot w\right), \tag{7}$$

$$\dot{I}_a = \frac{1}{L_a}\left(K_g \cdot K_\phi \cdot w - (R_a + R_L)I_a\right), \tag{8}$$

$$P_{out} = R_L \cdot {I_a}^2 \tag{9}$$

Where L_a is the armature inductance (H), K_g is a dimensionless constant of the generator, K_ϕ is the magnetic flow coupling constant (V·s/rad), R_a is the armature resistance (Ω), R_L is the resistance of the load (Ω), considered in this study as purely resistive; w is the angular rotor speed (rad/s), I_a is the armature current (A), λ is the TSR which is dimensionless, and $[\alpha, \beta, \gamma, \tau]$ is the set of values of the filter that the blades implement.

The value C_p depends on the characteristics of the WT; J is the rotational inertia (kg m^2), R is the radius or blade length (m), ρ is the air density (kg/m^3), K_f is the friction coefficient (N.m/rad/s), K_θ and T_θ are dimensionless parameters of the pitch actuator, v_{ef} is the effective wind speed in the blades (m/s), and v_W is the wind speed measured by the anemometer sensor.

The parameters of the WT used in the simulations are shown in Table 1, extracted from (Mikati et al., 2012). It is a small WT that has been used in previous works, which allows us to compare the results.

4. Improved RL-based pitch controller

4.1. Architecture of the controller

RL is an artificial intelligence technique that mimics the way humans learn. It learns to perform tasks through rewards that reinforce

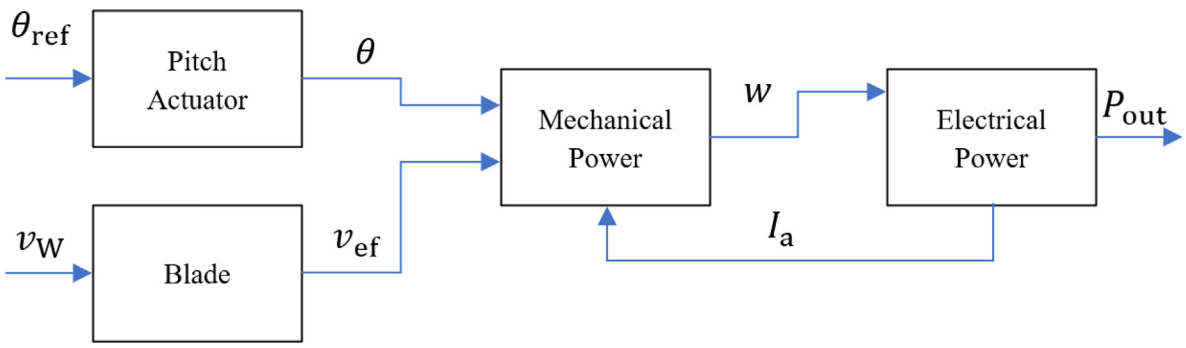


Fig. 1. Mathematical model representation of a wind turbine.

some actions (Sutton and Barto, 2018; Perrusquía and Yu, 2021). Typically, three elements are identified: an environment, an agent, and an interpreter. The agent, taking into consideration the state of the environment and the rewards previously obtained, selects the action with which it estimates a greater reward will be obtained. This action produces a change in the environment. The interpreter observes the effect of the action on the environment and provides feedback to the agent about the new state and the reward for the last action, thus closing the control loop. The relationship between the current state and the action to be taken is often called policy. This relationship may be discrete or continuous. If the policy is discrete, it is usually implemented by means of a table, and if it is continuous, it is quite common to use a neural network.

Certain similarities can be observed between this RL process and a traditional control loop: the environment would be equivalent to the system, the agent could be the controller, and the actuator and the interpreter act as sensor and an observer. Some authors consider that the interpreter is embedded in the agent or in the environment, but, in either case, the interpreter's role is always present.

Once the main concepts of the RL have been introduced, it is easy to identify some of their elements in the architecture of the proposed controller, shown in Fig. 2. This control architecture has some benefits; on the one hand, it can be applied to any WT, since the controller is able to learn online the best policy to control the WT regardless of its size and type (on-land, offshore), and it also adapts to different wind profiles; on the other hand, the action of the PID regulator helps accelerate the training. This is especially interesting during the first iterations as the learning control system has hardly any experience to learn from.

The architecture is made up of a state estimator, a reward calculator, a policy update mechanism, an action selector, a learning observer and a PID controller (Fig. 2). The state estimator calculates the current state, $s_t \in S$, considering the output power error P_{err}, and the wind speed V_{W}. The output power error P_{err} is the difference between the power reference P_{ref} (rated or maximum power) and the current output power P_{out}. The reward calculator uses the information at the current state s_t to obtain the reward $r_t \in \mathbb{R}$. The policy update mechanism modifies the table of the expected rewards $T : S \times A \rightarrow \mathbb{R}$, updating the reward associated to the previous state s_{t-1} and the previous action a_{t-1} based on the rewards.

The action selector chooses an action $a_t \in A$ based on table T, and it transforms this action into a pitch angle $\theta_R \in [0, \pi/2]$. At the end of each iteration, the learning observer checks if the controller is reducing the total error. If the error does not decrease, the learning observer adjusts the two input parameters of the action selector, ϵ and eW. As the pitch angle proposed by the action selector is not the ideal one, especially during the first learning iterations, the PID controller helps reduce the power error and stabilizes the output power around its rated value during that phase.

In order to study the stability of the controller, we have qualitatively analyzed the action of each controller separately. First, let suppose θ_R is constant; if the wind speed decreases, the output power goes down and the power error $P_{\text{err}} = P_{\text{ref}} - P_{\text{out}}$ grows. Therefore, the output of the PID controller increases, and the pitch reference decreases, $\theta_{\text{ref}} = \theta_{\text{R}} - \theta_{\text{PID}}$. As the pitch angle is smaller, the blade's surface exposed to the wind is bigger and the output power grows. Thus the power error is compensated, and the power remains stable.

Regarding the RL, in order to simplify the reasoning let us assume the number of error states is $n_e = 2$ and the number of derivative error states is $n_{de} = 2$, that is, the system is able to discriminate if the power error and its derivative are positive or negative. Therefore, the state estimator knows if the output power is getting closer or moving away from the reference, and the state of the system represents this. Now let suppose that the current state $st1$ indicates that the power error is positive ($P_{\text{ref}} > P_{\text{out}}$), and the output power is getting closer to the reference ($\dot{P}_{\text{err}} < 0$). If the action selector selects an action $a1$ that increases the output power, then the power error decreases, and the reward mechanism calculates a positive reward. This increases the Q value associated with the pair $(st1, a1)$, which enlarges the probability to select this action, $a1$. The state keeps being $st1$ as the power error is still positive. By contrast, if the action $a1$ decreases the output power, the reward would be negative, and the probability of selecting this action would decrease, and thus the state would change to $st2$. Iteration by iteration the frequency of the actions that push the output power towards the reference keeping the state $st1$ will grow, stabilizing the power around its rated value, and the other actions that move the system to $st2$ will tend to be rejected. Finally, if the power error is negative ($P_{\text{ref}} < P_{\text{out}}$), the output power gets closer to the reference when ($\dot{P}_{\text{err}} > 0$), then the reward is also positive. A similar explanation can be applied to states $st3$ and $st4$.

In addition, the effect of combining a RL controller and a PID is similar to the fact of having a lookup table and a PID regulator. The lookup table proposes a pitch reference accordingly to the wind: the larger the wind speed, the bigger the pitch angle. As this mapping function is not perfect, the PID contributes to reducing the mapping error between wind and blades angle. Previous works have shown how this approach improves the performance of the individual controllers (Sierra-García and Santos, 2021a). In this work, the role of the lookup table is played by the RL, the mapping relationship between the wind and the pitch angle is obtained by RL, and the PID helps reduce the errors while the controller is learning.

The operation of this control architecture can be formalized by the following expressions, Eqs. (10)–(18).

$$P_{\text{err}}\left(t_i\right) = P_{\text{ref}}\left(t_{i-1}\right) - P_{\text{out}}\left(t_{i-1}\right), \tag{10}$$

$$s_t = f_{\text{se}}(P_{\text{err}}\left(t_i\right), P_{\text{err}}\left(t_{i-1}\right), v_{\text{w}}\left(t_{i-1}\right)), \tag{11}$$

$$r_t = f_{\text{rc}}(P_{\text{err}}\left(t_i\right), P_{\text{err}}\left(t_{i-1}\right)), \tag{12}$$

$$T_{(s_{t-1},a_{t-1})}\left(t_i\right) = f_{\text{pu}}\left(r_t, T_{(s_{t-1},a_{t-1})}\left(t_{i-1}\right)\right) \tag{13}$$

$$\theta_{\text{R}}(t_i) = f_{\text{as}}(T_{(s_t)}\left(t_i\right), \epsilon\left(t_i\right), eW\left(t_i\right)), \tag{14}$$

$$\theta_{\text{PID}}\left(t_i\right) = \text{K}_{\text{p}} \cdot P_{\text{err}}\left(t_i\right) + \text{K}_{\text{d}} \cdot \frac{d}{dt} P_{\text{err}}\left(t_i\right) + \text{K}_{\text{I}} \cdot \int P_{\text{err}}\left(t_i\right) dt, \tag{15}$$

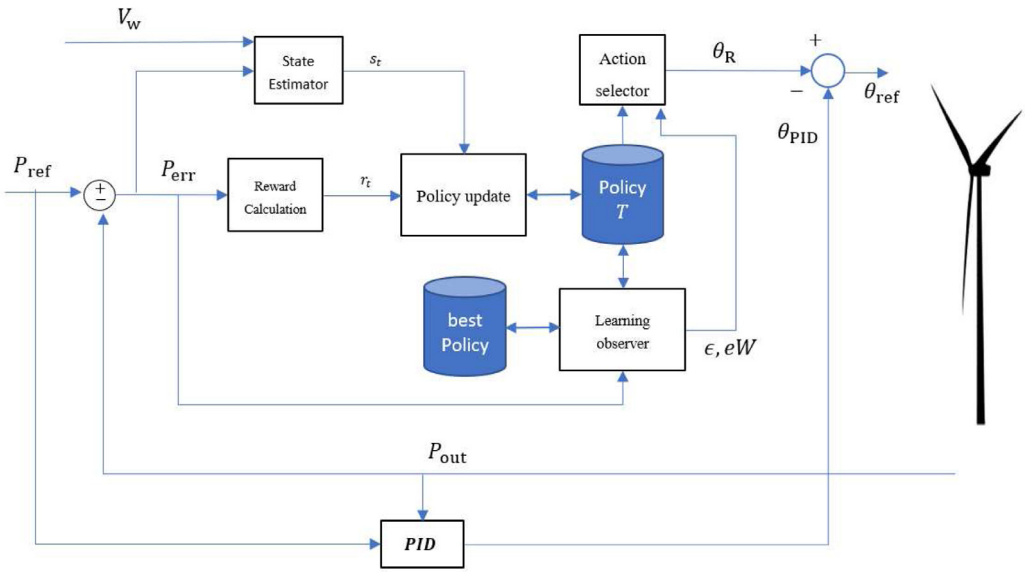


Fig. 2. Architecture of the reinforcement learning controller.

$$\theta_{ref}\left(t_i\right) = \theta_R\left(t_i\right) - \theta_{PID}\left(t_i\right), \tag{16}$$

$$\left[\epsilon\left(t_i\right), eW\left(t_i\right)\right] = \begin{cases} f_{LO}\left(P_{err}\left(t_i\right), P_{err}\left(t_{i-1}\right) \dots P_{err}\left(t_{i-N}\right)\right) & t_i \in T_{ep} \\ \left[\epsilon\left(t_{i-1}\right), eW\left(t_{i-1}\right)\right] & t_i \notin T_{ep}, \end{cases} \tag{17}$$

$$\left[T\left(t_i\right), bT(t_i)\right] = \begin{cases} f_{LOT}\left(P_{err}\left(t_i\right), P_{err}\left(t_{i-1}\right) \dots P_{err}\left(t_{i-N}\right)\right) & t_i \in T_{ep} \\ \left[T\left(t_{i-1}\right), bT\left(t_{i-1}\right)\right] & t_i \notin T_{ep} \end{cases}, \tag{18}$$

where $[K_p, K_D, K_I]$ are the tuning parameters of the PID, T_{ep} is the time when an episode finishes, $f_{se}: \mathbb{R}^3 \rightarrow S$ is the function implemented by the state estimator, $f_{rc}: \mathbb{R}^2 \rightarrow \mathbb{R}$ is the function implemented by the reward calculator, $f_{pu}: \mathbb{R}^2 \rightarrow \mathbb{R}$ is the function implemented by the policy updater, $f_{as}: \mathbb{R}^{|A|} \times \mathbb{R} \in [0,1] \times \mathbb{R} \in [0,1] \rightarrow \mathbb{R} \in [0, \pi/2]$ is the action selector function, $f_{LO}: \mathbb{R}^N \rightarrow \mathbb{R} \in [0,1] \times \mathbb{R} \in [0,1]$ is the function that the learning observer implements to update the pair $[\epsilon, eW]$, $f_{LOT}: \mathbb{R}^N \rightarrow \mathbb{R}^{|A|\cdot|S|} \times \mathbb{R}^{|A|\cdot|S|}$ is the function used by the learning observer to update table T, and N is the number of iterations.

The main elements of this control architecture are described in the following sections.

4.2. State estimator

The state of the system at time t is represented by a discrete value s_t, given by the value of the power error $P_{err}\left(t_i\right)$, its derivative, $dP_{err}\left(t_i\right)$, and the wind speed $v_w\left(t_i\right)$ (see Fig. 2). These signals are first discretized and then the state of the system is obtained by the combination of these discrete values, Eqs. (19)–(25).

$$dP_{err}\left(t_i\right) = \frac{P_{err}\left(t_i\right) - P_{err}\left(t_{i-1}\right)}{t_i - t_{i-1}}, \tag{19}$$

$$P_{err_S}\left(t_i\right) = MIN\left(e_{max}, MAX\left(P_{err}\left(t_i\right), e_{min}\right)\right), \tag{20}$$

$$dP_{err_S}\left(t_i\right) = MIN\left(de_{max}, MAX\left(\dot{P}_{err}\left(t_i\right), de_{min}\right)\right), \tag{21}$$

$$P_{err_D}\left(t_i\right) = DIV\left(P_{err_S}\left(t_i\right) - e_{min}, n_e\right), \tag{22}$$

$$dP_{err_D}\left(t_i\right) = DIV\left(dP_{err_S}\left(t_i\right) - de_{min}, n_{de}\right), \tag{23}$$

$$v_{wD}\left(t_i\right) = DIV\left(v_w\left(t_i\right) - v_{wmin}, n_v\right), \tag{24}$$

$$s_t = P_{err_D}\left(t_i\right) \cdot n_{de} \cdot n_v + dP_{err_D}\left(t_i\right) \cdot n_v + v_{wD}\left(t_i\right), \tag{25}$$

Where DIV denotes the integer division, $[e_{min}, e_{max}]$ is the range of the output power error, $[de_{min}, de_{max}]$ is the range of its derivative, v_{wmin} is the minimum wind speed, and $[n_e, n_{de}, n_v]$ are integer constants to adjust the number of states.

The selection of the number of states can influence the performance of the controller. Larger numbers give smaller discretization errors and, thus, smaller power errors. However, if the number of states is large, the frequency of repetition of the states is low, and this reduces the learning speed. To ensure a good balance, in this work, the numbers of states have been set to $\left[n_e, n_{de}, n_v\right] = [100, 50, 4]$, obtained after several simulations.

Estimating the state allows us to implement different actions in different cases: when the power error is small or large if this error is growing or decreasing, and considering high and low wind speed. These elements are key when controlling the pitch angle of a WT to get the maximum output power.

4.3. Reward calculator

As shown in Fig. 2, the reward calculator module receives the output power error and calculates the rewards r_t. There are several options to assign rewards to the actions carried out by the controller. For instance, a first logic approach would be to assign a reward as a function of the output power error regarding its nominal value. Thus, bigger errors, smaller rewards. However, depending on the policy update, the learning may not converge or does it very slowly (Sierra-García and Santos, 2020a). For this reason, in this work, we introduce rewards and penalties. If the output power approaches the nominal value, the action gets a reward; if it deviates from the right value, the action is penalized. The size of the reward/penalty is based on the derivative of the power error, as this value gives information about the speed at which the output power is approaching or moving away from the reference.

If both the power error and its derivative are positive or negative, it means that the output power deviates from the reference, and the action must be punished. On the other hand, if they have a different sign, the output power is approaching the reference, and this action must be rewarded. If the power error is zero, but its derivative is positive or negative, the output power deviates from the reference, and thus, it must be punished. These rules are mathematically summarized

in Eq. (26).

$$r_{\mathrm{v}}(t_i) = \begin{cases} -d\,P_{\mathrm{err_S}}(t_i) & P_{\mathrm{err_S}}(t_i) > 0 \\ -\left|d\,P_{\mathrm{err_S}}(t_i)\right| & P_{\mathrm{err_S}}(t_i) = 0 \\ d\,P_{\mathrm{err_S}}(t_i) & P_{\mathrm{err_S}}(t_i) < 0 \end{cases}, \tag{26}$$

$$r_t = \begin{cases} -r_{\mathrm{v}}(t_i) & sign\left(P_{\mathrm{err_S}}(t_i)\right) \neq sign\left(P_{\mathrm{err_S}}(t_{i-1})\right) \wedge \\ & \left|P_{\mathrm{err_S}}(t_i)\right| < \left|P_{\mathrm{err_S}}(t_{i-1})\right| \\ r_{\mathrm{v}}(t_i) & othercase \end{cases}, \tag{27}$$

A special case may happen when the power error changes its sign. If Eq. (26) is applied, a penalty will be always assigned because the new most recent value of the power error and its derivative have the same sign. However, if the latest value is closer to the reference than the previous one, it would be better to assign a reward. Thus, we use Eq. (27) to consider this case and correct the reward assignment. An auxiliary variable r_{v} is defined to simplify the equations. First, the signal r_v is obtained and then it is used to calculate the reward r_t, which is finally considered by the policy update module.

4.4. Policy update

The policy update modifies the policy considering the rewards given by the reward calculator. The policy of the RL is implemented as a table which associates an estimation of the expected reward $T_{(s_t,a_t)}$ to each pair of state and action (s_t, a_t). In this table, there will be so many rows as states and so many columns as actions. This means that if the system is at state s_t and the action selector executes a_t, it is expected to receive the reward/penalty of the associated cell in the table in the next iteration. The way to update the values of the policy table must consider the previous rewards and combine them with the latest one, r_t. Again, there are several options to do this. For instance, it is possible to consider only the latest one and discard the rest of the previous ones. This can be useful when the system fluctuates, but it is normally better to remember some previous rewards. To do so, an option is to calculate the average of some previous rewards, i.e., considering a moving average. Instead of adding the previous rewards it is possible to obtain it recursively, using the last value of the table according to equation Eq. (28),

$$T_{(s_{t-1},a_{t-1})}(t_i) = T_{(s_{t-1},a_{t-1})}(t_{i-1}) + \alpha\left(r_t - T_{(s_{t-1},a_{t-1})}(t_{i-1})\right), \tag{28}$$

Where α is the learning rate. This learning rate is equivalent to the inverse of the number of previous samples that is used to obtain the average of the rewards.

4.5. Action selector

According to Fig. 2, the action selector receives the current policy T, the exploration probability ϵ, and the exploration window size eW, and it calculates the corresponding pitch reference, θ_{R}. As shown in Fig. 3, the action selector has two paths to obtain the next action: one way is to calculate a greedy action, a_G, which considers the previous experiences, and another way is to calculate a random exploratory action, a_E. Depending on the position of the switch, one of these paths is used to choose the action that is finally executed, a_t. The position of the switch is randomly selected. To do it, a random number between 0 and 1 is generated. If the result is smaller than ϵ, the action a_E is selected. Otherwise, the greedy action a_G is chosen. Therefore, the probability of a random exploratory action is ϵ, and the probability of an action considering the experience is $(1-\epsilon)$. The parameter ϵ is adjusted by the learning observer.

The greedy action a_G is obtained based on the values of the policy table T. The state s_t is used to select the row of the table which implements the corresponding policy $T_{(s_t)}$. The content of the row can be accessed as a lookup table. The input of the table is the action, and the output is the expected reward. In standard RL, the greedy action is usually selected as the action with the maximum expected reward, a_{tBEST}. However, in order to reduce the error due to the discretization of the actions, in this approach the actions are weighted by the expected rewards in the table, giving as result a_G. For instance, if the table only had two discrete actions, $A = \{1, 2\}$, and these actions had the same expected rewards, the resulting weighted action 1.5 would be selected. This continuous action would be in the middle of the discrete actions 1 and 2. This way the discrete set of actions A is transformed into a continuous set of actions in the range $[0, |A|] \in \mathbb{R}$, where $|A|$ denotes the size of the discrete set A.

On the other hand, the exploratory action a_E is selected using the exploration window. This is a set of actions around the best action, a_{tBEST}, with size $eW \cdot |A|$. The parameter $eW \in \{\mathbb{R} \in [0, 1]\}$ is an input parameter adjusted by the learning observer. When a random action is explored, the action is chosen from a smaller range of options within the exploration window close to the best action, rather than from all the possible actions in A. This strategy facilitates the reduction of oscillations during the training and accelerates the learning.

In standard RL problems, the actions are not sorted, and there is not any relationship between them. However, in WT pitch control and in other control problems, the actions can be sorted. If the pitch angle increases the power decreases and vice versa. Therefore, if we rank the actions by considering the pitch angle, smaller actions provide smaller pitch and higher power. Hence, the actions that are nearly a_{tBEST} will have a larger probability of obtaining the best rewards. Therefore, the exploration window helps select a random action among the actions that a priori will get a better reward.

The operation of the action selector can be formalized by the following expressions Eqs. (29)–(33).

$$a_{\mathrm{G}} = (1/\sum_i T_{(s_t,i)}) \cdot \sum_i T_{(s_t,i)} \cdot i, \tag{29}$$

$$a_{\mathrm{t_{BEST}}} = arg_i MAX(T_{(s_t,i)}), \tag{30}$$

$$a_{\mathrm{E}} = a_{\mathrm{t_{BEST}}} + (rand() - 0.5) \cdot eW \cdot |A|, \tag{31}$$

$$a_t = \begin{cases} a_{\mathrm{G}} & rand() > \epsilon \\ a_{\mathrm{E}} & rand() \leq \epsilon, \end{cases} \tag{32}$$

$$\theta_{\mathrm{R}} = a_t \cdot (\pi/2)/|A|, \tag{33}$$

4.6. Learning observer

The learning observer is a module of the control strategy that receives as inputs the output power error and the policy (Fig. 2). It generates the best policy and adjusts the input parameters that configure the action selector. This module monitors the root mean squared error (RMSE) of the output power at the end of each training episode. If it detects that this error is growing, it modifies the inputs of the action selector to correct this tendency. This way it contributes to make faster the learning and limits the oscillations during the learning process.

As said, the action selector chooses a random action with a probability ϵ and a greedy action with a probability $(1-\epsilon)$. This is carried out to guarantee a proper exploration of the actions and to avoid falling into local minima. If ϵ is very small the actions are mainly based on the experience and the system may fall into a local minimum. However, if it is too high, the oscillations of the RMSE can be very large and the learning takes a long time. Thus, it is important to adjust this parameter properly. The learning observer oversees this; it checks the RMSE at the end of every episode; if the RMSE grows, then it increases the exploration in order to find better actions by enlarging the parameters ϵ and eW.

This module also compares the RMSE at the end of every episode with the best RMSE found (the smallest), $RMSE_{\mathrm{MIN}}$. If the current value is much greater than the best one, the current policy T is replaced by the best policy bT. This allows it to avoid large oscillations of the

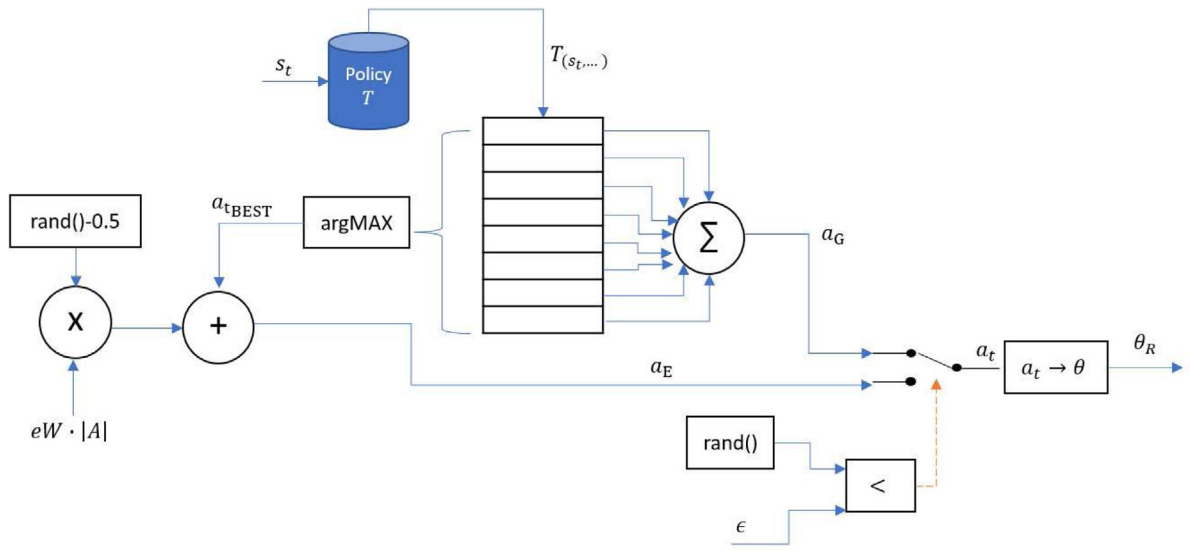


Fig. 3. Architecture of the action selector.

RMSE. In addition, every time that a best RMSE value is found, the best policy bT is stored.

Regarding the convergence of the learning observer, we must consider that the relationship of the pitch angle with the output power is inversely proportional. Thus, if the pitch increases the power decreases. On the other hand, it is well-known that for each state st, the RL aims to find the action that maximizes the expected rewards. In our case, the rewards are designed to select the actions that push the power to the reference and to reject those which move it away from it. For the states in which the power error is positive, large pitch angles tend to increase the error and provide negative rewards. Conversely, small pitch angles increase the output power and decrease the error as long as the reference is not exceeded. In another case, the power error jumps to negative values, the power moves away from the reference, and the rewards become negative. Therefore, the power error decreases until a local minimum is reached, and it grows from thereon. The precise action with which the minimum is reached depends on the wind speed, as higher wind speed gives smaller pitch angles. For each state st, this best action is searched by the RL algorithm.

This explanation may make it easier to understand how the learning observer works. This component oversees the exploration of the system. The selection of the exploration rate is a key issue in RL; if the exploration is low the actions are mainly greedy, and the best actions may not be explored. However, if the exploration is too high, the results are too random. In our approach, the system starts with an exploration rate ϵ low, thus almost all actions are greedy. The reward/punishment strategy tends to reduce the power error at each state and increases the frequency of the actions which receive positive rewards. This contributes to reduce the global RMSE episode by episode. However, this deceleration rate decreases as the power approaches the rated value because the size of the rewards is then smaller, and the RMSE tends to converge.

While the exploration rate is still low, the selected actions may be far from the optimum, and there comes a point at which the RMSE increases. Then the learning observer detects it and increases the exploration rate and the exploration window. The action selector chooses random actions within the exploration window, that is, actions whose distance to the current best action is less than the half the exploration window. Enlarging the level of exploration contributes to find actions closer to the optimum of each state, and this may reduce the global RMSE and helps the system converge. If the RMSE is reduced, the learning observer resets the exploration to exploit the new learned actions that have been found. In another case, if the exploration level were kept at the same value, the randomness of the results would grow.

The previous explanation can be formalized by the following equations, Eqs. (34)–(39).

$$RMSE\,(ep) = \sqrt{\frac{1}{N_t}\sum \left(P_{out}\,(t_i) - P_{ref}\,(t_i)\right)^2}, \tag{34}$$

$$[RMSE_{\mathrm{MIN}}(ep), RMSE_{\mathrm{MIN}_2}(ep)] = \begin{cases} \left[RMSE\,(ep), RMSE_{\mathrm{MIN}}\,(ep-1)\right] \\ \qquad if\ RMSE\,(ep) < RMSE_{\mathrm{MIN}}\,(ep) \\ \left[RMSE_{\mathrm{MIN}}\,(ep-1), RMSE\,(ep)\right] \\ \qquad if\ RMSE_{\mathrm{MIN}}\,(ep) < RMSE\,(ep) < RMSE_{\mathrm{MIN}_2}\,(ep) \\ \left[RMSE_{\mathrm{MIN}}\,(ep), RMSE_{\mathrm{MIN}_2}\,(ep)\right] \\ \qquad if\ RMSE_{\mathrm{MIN}_2}\,(ep) < RMSE\,(ep), \end{cases} \tag{35}$$

$$\epsilon(ep) = \begin{cases} \epsilon\,(ep-1) + \mathrm{inc}_\epsilon & if\ RMSE\,(ep) \geq RMSE_{\mathrm{MIN}_2} \\ 0 & if\ RMSE\,(ep) < RMSE_{\mathrm{MIN}_2}, \end{cases} \tag{36}$$

$$eW(ep) = \begin{cases} eW(ep-1) + \mathrm{inc}_{\mathrm{eW}} & if\ RMSE\,(ep) \geq RMSE_{\mathrm{MIN}_2} \\ 0 & if\ RMSE\,(ep) < RMSE_{\mathrm{MIN}_2}, \end{cases} \tag{37}$$

$$bT(ep) = \begin{cases} T\,(ep) & if\ RMSE\,(ep) < RMSE_{\mathrm{MIN}} \\ bT\,(ep-1) & if\ RMSE\,(ep) \geq RMSE_{\mathrm{MIN}}, \end{cases} \tag{38}$$

$$T(ep) = \begin{cases} bT(ep) & if\ RMSE\,(ep) > \mathrm{k_T} \cdot RMSE_{\mathrm{MIN}} \\ T(ep-1) & if\ RMSE\,(ep) \leq \mathrm{k_T} \cdot RMSE_{\mathrm{MIN}}, \end{cases} \tag{39}$$

Where ep is the current training episode, N_t is the number of training samples in every episode, $[\mathrm{inc}_\epsilon, \mathrm{inc}_{\mathrm{eW}}]$ are parameters to adjust how fast the exploration grows, and $\mathrm{k_T}$ is a parameter to control when the policy must be reset to bT.

5. Results and discussion

This RL-based control strategy has been applied to the model of a real small WT. Simulation results have been obtained using Matlab/Simulink software. The duration of each simulation is 100 s. The sample time, T_s, varies in order to reduce the discretization error, its maximum value is 20 ms. The control period is 250 ms.

To evaluate the performance of this proposed hybrid controller, it has been compared with a conventional PID regulator, Eq. (40), with a control system based on pure RL without the PID controller, Eq. (41), with a FLC, Eq. (43), and with a neural network controller, Eq. (44).

$$\theta_{\mathrm{PID}}\,(t_i) = \frac{\pi}{4} - \mathrm{K_p}\left[P_{\mathrm{err}}\,(t_i) + \mathrm{K_d} \cdot \frac{d}{dt}P_{\mathrm{err}}\,(t_i) + \mathrm{K_i} \cdot \int P_{\mathrm{err}}\right] \tag{40}$$

$$\theta_{\mathrm{DRL}}\left(t_i\right) = f_{\mathrm{as}}(T_{(s_t)}\left(t_i\right), \epsilon\left(t_i\right), eW\left(t_i\right)), \tag{41}$$

$$\theta_{\mathrm{DRL-PID}}\left(t_i\right) = f_{\mathrm{as}}\left(T_{(s_t)}\left(t_i\right), \epsilon\left(t_i\right), eW\left(t_i\right)\right) - \theta_{\mathrm{PID}}\left(t_i\right), \tag{42}$$

$$\theta_{\mathrm{FLC}}\left(t_i\right) = \frac{\pi}{4} - f_{\mathrm{FLC}}\left(P_{\mathrm{err}}\left(t_i\right), V_{\mathrm{w}}(t_{i-1})\right), \tag{43}$$

$$\theta_{\mathrm{NEU}}\left(t_i\right) = \frac{\pi}{4} - f_{\mathrm{NEU}}\left(P_{\mathrm{err}}\left(t_i\right), \dot{P}_{\mathrm{err}}\left(t_i\right)\right), \tag{44}$$

Where $[\mathrm{K_p}, \mathrm{K_d}, \mathrm{K_i}]$ are the tuning parameters of the PID that have been obtained by trial and error. Their values are $[\pi/4000, 0.2, 01]$.

The FLC is implemented by a Takagi–Sugeno structure with two inputs, P_{err} and V_{W}, and one output, θ_{ref}. Three Gaussian fuzzy sets are assigned to the fuzzy input P_{err}, namely Negative, Zero and Positive, uniformly distributed in the range [−500, 500] W, and width 175 W. The speed V_{DL} is defined by three uniformly distributed Gaussian fuzzy sets in the interval [12.25, 13] m/s, the width is 0.132 m/s. Its labels are Low, Medium and High. The output is a singleton that can take three values: - $\pi/4$, 0, and $\pi/4$ (rad).

On the other hand, the neural controller is implemented by a radial basis function neural network with two inputs: P_{err} and $\dot{P}_{\mathrm{err}}$. The input range of P_{err} is adjusted to [−1100,1100] and the input range of $\dot{P}_{\mathrm{err}}$ is set to [−1000, 1000]. The number of hidden neurons is 25. The learning algorithm of the neural network is described in Sierra-García and Santos (2020b).

5.1. Performance of the controller

The performance of the proposed hybrid controller is tested with different wind speed profiles: a noisy constant wind speed with an average speed value of 12.625 m/s and a signal-noise ratio of 30 dB; a sinusoidal signal with an amplitude 0.375 m/s, period of 50 s, and an average value of 12.625 m/s; and a sawtooth signal, with the same amplitude, average value, and period. Different controllers obtained with combinations of the constants $[\mathrm{K_p}, \mathrm{K_d}, \mathrm{K_i}]$ have been combined with the DRL and tested, particularly, a PID-controller, P-controller, PD-controller, and a PI-controller.

Figs. 4–6 show the wind speed signal (left) and the output power obtained with different pitch control strategies (right). The rated value is represented with a dashed black line. The light blue lines represent the output power when the pitch reference is 90° and the green ones when the reference is fixed to 0°. The dark blue lines show the results obtained with the PID, and the red lines the results given by the PID with $\mathrm{K_i} = 0$ (PD). The yellow lines show the results of the RL controller without PID, called DRL, and the purple lines the performance of the hybrid controller, i.e., RL combined with the PID or PD (DRL-PID or DRL-PD), depending on which one gives better performance. The RL controller has been trained during 100 episodes and the figures show the results of the best values at the end of the training. In these three figures, 4–6, the green line and the light blue line help identify the boundaries of the signals since when the pitch reference is 0°, the output power is maximum and when it is 90°, the wind power is minimum.

In Fig. 4b it is possible to observe how the best performance is obtained with the DRL controller (yellow) and the DRL-PD controller (purple) with a noisy constant wind speed (Fig. 4a). In this case, the best RMSE was obtained with $\mathrm{K_i} = 0$, that is, with a PD regulator. The four controllers can stabilize the output power around the rated value, however the PID and the PD reacts slower. In this case the difference between the DRL and the DRL-PD is not very noticeable.

In Fig. 5a, the sinusoidal shape of the wind speed (Fig. 5b) can be seen in the power limits (light blue and green lines). The minimum error is obtained with the DRL-PD followed by the pure DRL. Again, the best results are obtained with $\mathrm{K_i} = 0$. The biggest differences between the DRL and the DRL-PD are observed in the signal peaks.

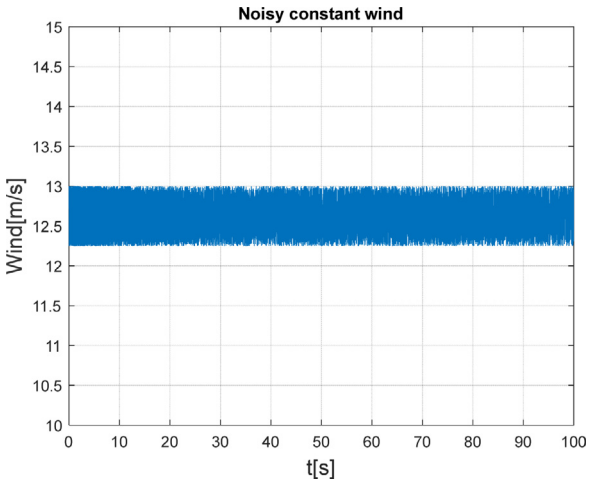


Fig. 4a. Noisy constant wind speed.

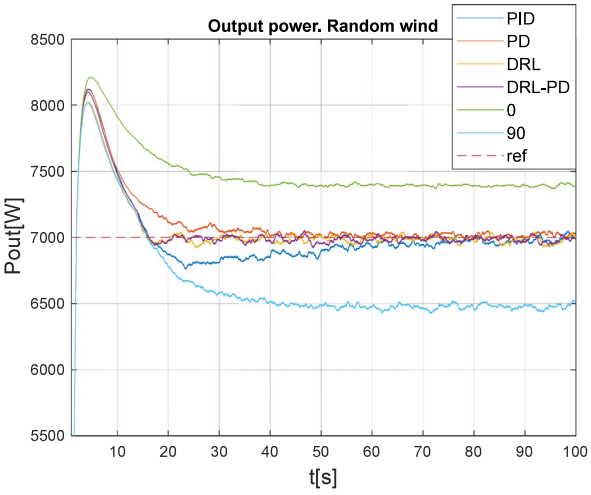


Fig. 4b. Output power for noisy wind speed.

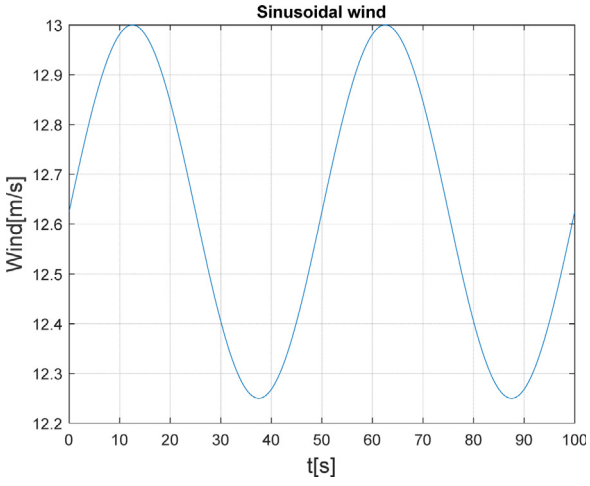


Fig. 5a. Sinusoidal wind speed.

Fig. 6b presents the same comparison when the wind is a sawtooth signal (Fig. 6a). Again, the best performance is obtained by the RL controller combined with the PID, although in this case the improvement given by the PID in terms of RMSE is smaller than in the case of the

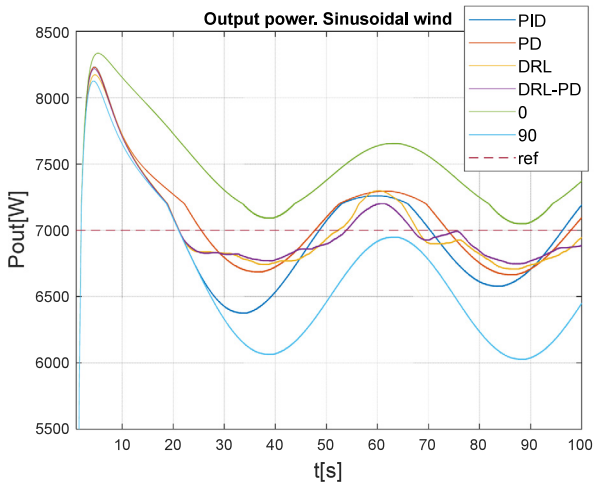


Fig. 5b. Output power for sinusoidal wind speed.

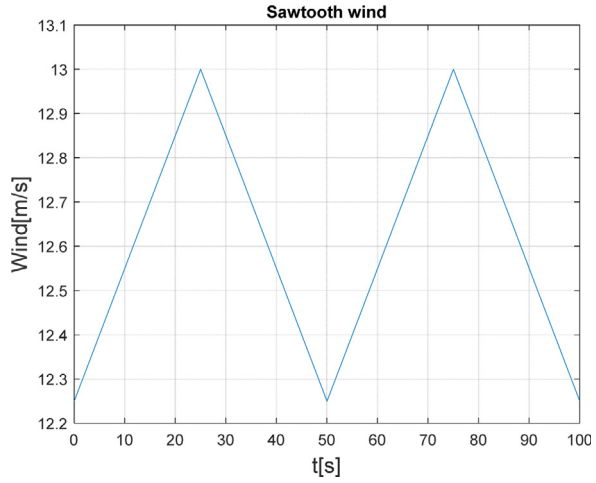


Fig. 6a. Sawtooth wind speed.

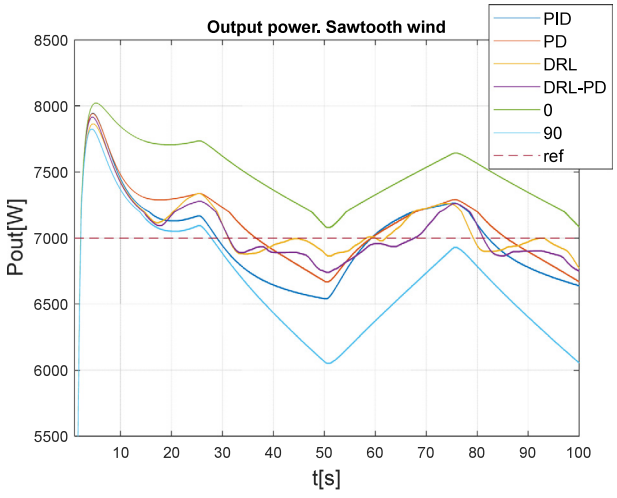


Fig. 6b. Output power for sawtooth wind speed.

sinusoidal wind. However, it can be seen how the DRL-PID controller is able to better dampen power fluctuations caused by changes in the wind than the DRL control.

In addition to these figures, numerical results have also been obtained. Table 2 shows the root mean squared error (RMSE) of the output power. The columns represent the results with the controllers: PID, P (PID with K_i and K_d equal to 0), PI (PID with $K_d = 0$), PD (PID with $K_i = 0$), DRL, DRL-PID, DRL-P, DRL-PI, and DRL-PD. The last row indicates the average value of the previous rows. The last columns show the values obtained for the same turbine and the same wind profiles with a fuzzy controller (FUZ) and a neuro-control (NEU).

According to Table 2, for every wind speed profile, the RMSE is smaller when the DRL or any of its variants is applied. The best DRL variant (DRL-P) provides a mean RMSE that is 26% smaller than the mean RMSE of the best PID; even more, in the case of the sawtooth wind profile this reduction is up to 35%. Another interesting result is that the DRL control, even without PID, gives better results than the PID regulators. The combination of the PID and DRL improves the performance with respect to the DRL control for almost all wind profiles, being the random wind the only exception. The integral term of the PID only improves the control for the sawtooth wind speed profile. It seems that this term slows down the learning of the RL controller.

The results obtained by the DRL controller and its combinations with P-controller or PD-controller are better than the results given by the other intelligent control methods tested, the fuzzy controller and the neural network controller.

The RL controller learns how to adapt the control law online. On the contrary, the PID is tuned for specific operating conditions and considering certain system parameters; thus, any changes in the system affect its performance. To illustrate this, we have changed the resistance load of the WT. The previous experiments were performed with a resistance of 8 ohm. We changed the resistance to 7.84 ohm, just a 2%, and the RMSE for the sawtooth wind increased from 250 W to 272 W, that is, 8%. Changing the resistance to 7.6 ohm (5% of variation) the RMSE grows up to 331 W, a 32% increase. Finally, a value of 7.2 ohm (10% decrease) makes the RMSE grows up to 421 W, that is, a 68% bigger. However, in all these cases, the performance of the DRL and the other combinations of control methods can cope with them and keep a good performance.

These results allow us to conclude that the performance of the WT pitch control improves when a conventional PID is combined with a DRL strategy. In the next section, it will be shown how the learning can also be improved.

5.2. Improvement of the DRL control strategy

The PID improves the control in terms of reducing the RMSE of the output power, while the RL controller is learning. These experiments have been carried out combining the DRL with a PID-controller, a P-controller, a PD-controller, and a PI-controller. The gains of the controller are the same as in the previous section. The RMSE is obtained at the end of each episode.

Figs. 7–9 (left) show the evolution of the RMSE while the system is learning. The figures, on the right, show the RMSE overtime when the best policy found by the learning observer is applied. The color code is the same as before.

On the one hand, these figures confirm the results presented in Table 2. The DRL control gives smaller RMSE than the best PID for all wind speed profiles. The incorporation of a PID to the DRL control strategy reduces the RMSE during the first episodes. This improvement is about 150–200 W for all the wind profiles during the first episode. For instance, the pure RL starts with an RMSE of 600 W with sinusoidal wind; when it is combined with a PID, the error starts around 450 W. This means a 25% RMSE reduction. The DRL combined with the P-controller or with the PD-controller keeps this difference up to an episode when it starts to decrease. This makes the controller converge much faster. However, this difference decreases from the first episode if the DRL controller is combined with the PI or the PID. In fact, the

Table 2
Comparison of the RMSE of the output power [W] for different control strategies and wind speed profiles.

Wind profile	PID	P	PI	PD	DRL	DRL-PID	DRL-P	DRL-PI	DRL-PD	FUZ	NEU
Noisy constant	125,99	104,5	134,15	105,25	**71,62**	131,25	79,07	149,49	75,88	84,61	232,53
Sinusoidal	339,95	265,29	347,51	263,88	231,29	312,94	**199,63**	320,26	206,09	214,58	295,13
Sawtooth	250,34	224,45	253,45	223,64	157,81	151,68	**144,4**	147,71	163,81	186,31	205,33
Mean	238,76	198,08	245,03	197,59	153,57	198,62	**141,03**	205,82	148,59	161,83	244,32

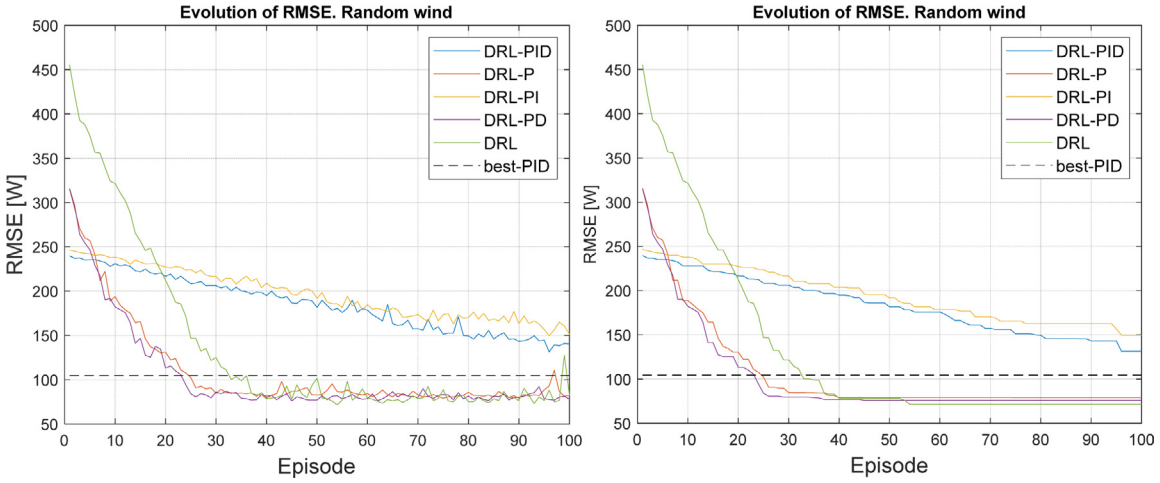


Fig. 7. Evolution of the RMSE for constant noisy wind, last policy (left), best policy (right).

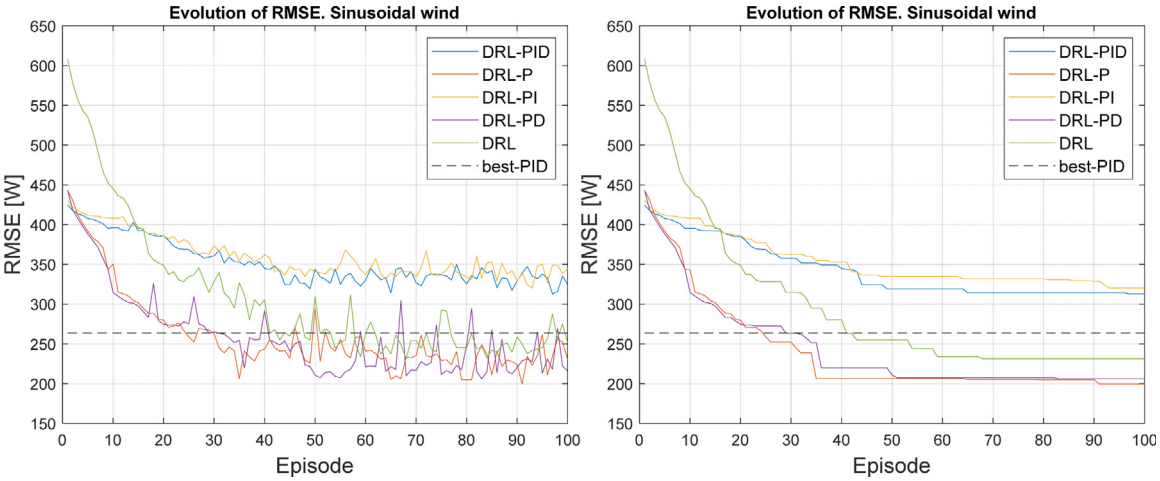


Fig. 8. Evolution of the RMSE for sinusoidal wind, last policy (left), best policy (right).

performance of the pure DRL overpasses the DRL-PI or DRL-PID when the number of episodes is large for random and sinusoidal wind speed profiles. It seems that the integral term of the PID controller slows down the response and the learning.

The most significant and most frequent peaks in the RMSE occur with sawtooth wind speed profile. On the contrary, the most regular learning is observed with constant noisy wind; in this case, the RMSE converges at episode 25 for DRL-P and DRL-PD, and at episode 40 for DRL. This represents an acceleration of 37%.

However, during the initial phase of the learning, the performance of the DRL is not so good because the intelligent controller does not have enough experience to learn from. Hence, the combination with the PID gives better results than on its own. In Fig. 7, the RMSE of the pure DRL (green line), without PID, starts at 500 W while with the PID (red, purple, orange), the initial error is around 350 W. This reduction of the error in the first episodes is the main contribution of the PID in the control scheme. Besides, the DRL without PID crosses the best PID line (dashed line) around episode 30. Meanwhile the red and purple lines (DRL with P or PD) cross the best PID line at around episode 10, that is, three times faster.

Therefore, the PID can improve the performance of the RL controller and accelerate the learning process. But if the gains of the PID are not well-tuned, the PID action may be counterproductive. To ensure its effectiveness, it is necessary to select small Ki values for the PID. If the Ki gain is too large, the PID may slow down the controller and decelerate the learning. Indeed, the best results have been obtained by the combination of RL and P or PD controller.

5.3. Effects of the learning observer

As explained before, the learning observer adjusts ϵ and eW in order to increase the actions space if it detects the learning is not improving from one episode to another. To evaluate the improvement of this approach, the evolution of the RMSE obtained with the DRL controller with the learning observer is compared with other standard RL strategies, such as keeping constant ϵ or ϵ-exponential decay. For a fair

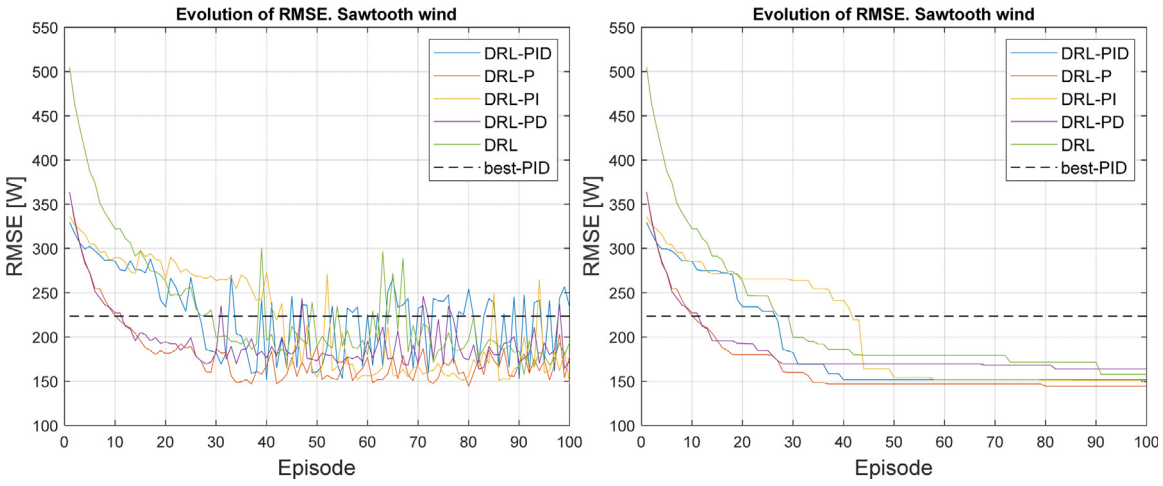


Fig. 9. Evolution of the RMSE for sawtooth wind, last policy (left), best policy (right).

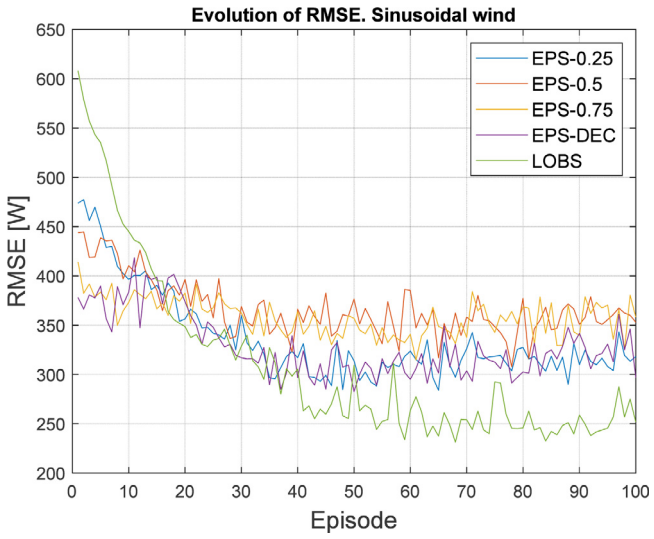


Fig. 10. Comparison of evolution of RMSE with different strategies that update ϵ.

comparison, in these experiments the DRL is used without combining it with a PID. The wind speed profile is sinusoidal with the parameters of the previous sections.

The exponential decay updates ϵ by Eq. (45), where ep denotes the episode.

$$\epsilon\,(ep) = e^{-0.05\cdot(ep-1)} \qquad 1 \geq ep \geq 100 \tag{45}$$

Fig. 10 shows the comparison of the evolution of the RMSE when different strategies to update ϵ are used. The parameter ϵ (EPS) is set to 0.25 (blue line), 0.5 (red line) and 0.75 (yellow line). The purple line shows the results when it is updated by the exponential decay (EPS-DEC), and the green one when it is updated by the learning observer (LOBS).

In the first episode, the RMSE obtained with the learning observer is larger than with the other approaches; however, this error decreases much faster. Indeed, from episode 40 the RMSE is much smaller with the learning observer. In addition, smoother and more regular learning is also shown, especially up to episode 40. As expected, during the first episodes a larger exploration produces better performance as the controller does not have enough experiences to learn from, and the best strategy is to explore all possible options. As the number of episodes increases, the controller gains more experience, and better results are obtained by lowering the exploration. Thus EPS-0.25 and EPS-DEC give a lower RMSE.

However, narrowing the exploration space is not enough to update the performance of the learning observer. The EPS-DEC reduces the exploration up to 0.0068 and still the RMSE obtained with LOBS is not reached. This may be explained by the effect of eW; EPS-DEC selects an action among all possible actions in the range $[0, \pi/2]$. However, the learning observer adjusts the size of the exploration window eW, and the random action is selected among the actions closest to the known best one, that is, the actions within the exploration window.

That is, the learning observer helps not only improve the convergence of the learning but also to get better results regarding the control, as the error is smaller.

6. Conclusions and future works

Because of the non-linearities, coupling between variables, and uncertainty of the environment, controlling the pitch angle of a WT is a challenging task. This has inspired researchers to investigate on intelligent control approaches as a solution to this control problem (Jove et al., 2021). One of these techniques that has been recently applied to the control of wind devices is RL, which implements different ways of learning using rewards that reinforce certain actions.

One of the drawbacks of RL is its slow convergence, which limits its applicability in real control problems. In this work, a novel control architecture that seeks to improve the performance of RL-based control

Table 3
List of abbreviations.

Abbreviation	Description
ANFIS	Adaptive Neuro-Fuzzy Inference System
Cp	Power Coefficient
DRL	Discrete Reinforcement Learning
EPS-DEC	Epsilon decay
FLC	Fuzzy Logic Controller
LOBS	Learning Observer
MPPT	Maximum Power Point Tracking
PID	Proportional–integral–derivative
PSO	Particle Swarm Optimization
RL	Reinforcement Learning
RMSE	Root Mean Squared Error
TSR	Tip Speed Ratio
WT	Wind Turbine

Table 4
List of control parameters.

Parameter	Description	Value
$[K_p, K_D, K_I] \in \mathbb{R}^3$	Tuning parameters of the PID	$[\pi/4000, 0.2, 01]$.
$[e_{min}, e_{max}] \in \mathbb{R}^2$	Range of the power error	[−1000,7000]
$[de_{min}, de_{max}] \in \mathbb{R}^2$	Range of the derivative of the power error	[−1000,1000]
$v_{w\,min} \in \mathbb{R}$	Min wind velocity	12
$inc_e \in \mathbb{R}$	Number of discrete states for P_{err_D}, dP_{err_D} and V_{wD}	[100,50,4]
$\alpha \in \mathbb{R}$	Learning rate	0.01
$inc_\epsilon \in \mathbb{R}$	Epsilon increment	0.01
$inc_{eW} \in \mathbb{R}$	Exploration window increment	0.01
$k_T \in \mathbb{R}$	Reset policy threshold	1.2

Table 5
List of variables.

Variable	Description	Units
$\theta_{ref} \in \mathbb{R}$	Pitch angle reference input of WT	rad
$\theta \in \mathbb{R}$	Pitch angle output of WT	rad
$\theta_R \in \mathbb{R}$	Pitch angle output of RL controller	rad
$\theta_{PID} \in \mathbb{R}$	Pitch angle output of PID controller	rad
$f_{blade} : \mathbb{R} \to \mathbb{R}$	Function of the blade filter	m/s
$v_{ef} \in \mathbb{R}$	Effective wind velocity	m/s
$v_W \in \mathbb{R}$	Measured wind velocity	m/s
$\lambda \in \mathbb{R}$	Tip Speed Ratio	–
$C_p : \mathbb{R} \to \mathbb{R}$	Power coefficient function	–
$w \in \mathbb{R}$	Angular rotor speed	rad/s
$I_a \in \mathbb{R}$	Armature current	A
$P_{ref} \in \mathbb{R}$	Power reference	W
$P_{err} \in \mathbb{R}$	Power error	W
$P_{out} \in \mathbb{R}$	Output power	W
$P_{err_S} \in \mathbb{R}$	Bounded P_{err}	W
$dP_{err_S} \in \mathbb{R}$	Bounded $\dot{P}_{err}$	W/s
$P_{err_D} \in \mathbb{R}$	Discrete P_{err}	W
$dP_{err_D} \in \mathbb{R}$	Discrete $\dot{P}_{err}$	W/s
$V_{wD} \in \mathbb{R}$	Discrete v_W	m/s
$S \subseteq \mathbb{N}$	Set of discrete states	–
$A \subseteq \mathbb{N}$	Set of actions	–
$s_t \in S$	State at time t	–
$a_t \in A$	Action at time t	–
$r_t \in \mathbb{R}$	Reward at time t	–
$a_G \in A$	Greedy action	
$a_E \in A$	Exploration action	
$a_{BEST} \in A$	Best action	
$T_{ep \in \mathbb{R}}$	Time when an episode finish	s
$f_{se} : \mathbb{R}^3 \to S$	Function implemented by the state estimator	–
f_{rc}: $\mathbb{R}^2 \to \mathbb{R}$	Function implemented by the reward calculator	–
f_{pu}: $\mathbb{R}^2 \to \mathbb{R}$	function implemented by the policy updater	–
f_{as}	Action selector function f_{as}: $\mathbb{R}^{\lvert A\rvert} \times \{\mathbb{R} \in [0,1]\}^2 \to \mathbb{R} \in [0, \pi/2]$	rad
$f_{LO} : \mathbb{R}^N \to \{\mathbb{R} \in [0,1]\}^2$	Function implemented by the learning observer to update $[\epsilon, eW]$	–
$f_{LOT} : \mathbb{R}^N \to \left\{\mathbb{R}^{\lvert A\rvert \cdot \lvert S\rvert}\right\}^2$	Function used by the learning observer to update tables	–
$T_{(s_t,a_t)} \in \mathbb{R}^{\lvert s\rvert \lvert A\rvert}$	Table which associates an estimation of the expected reward to each pair of state and action (s_t, a_t).	–
$bT_{(s_t,a_t)} \in \mathbb{R}^{\lvert s\rvert \lvert A\rvert}$	Best table $T_{(s_t,a_t)}$	–
$ep \in \mathbb{N}$	Episode	–
$\epsilon \in \{\mathbb{R} \in [0,1]\}$	Probability to select a random action	–
$eW \in \{\mathbb{R} \in [0,1]\}$	Exploration window	–

in WT is presented. The proposal initially applies the standard RL strategy: reward calculation, state estimator, policy updating, etc., that are here re-defined for the particular control problem. Then, the proposed intelligent RL control is improved by combining it with a PID regulator and a learning observer.

This hybrid controller has been shown more effective in terms of error regarding the rated power output. Even more, the combination of the DRL with a PID regulator reduces the error of the output power during the first episodes of the training. In addition, the learning converge is accelerated by around 37%. That allows the implementation of the control strategy in real-time. On the other hand, the hybrid controller, particularly the combination of the DRL with a proportional controller, reduces the RMSE of the wind power in about 41% regarding a well-tuned PID. With the best found configuration of the hybrid controller, the performance reaches the 98% in terms of power.

Besides, the definition of a learning observer that monitors the training and adjusts accordingly the exploration rate has been proved useful to get these improvements. It also smoothers the learning curve and gives a more accurate output power.

Among other possible future works, we may highlight the application of this control configuration to large WT, using the corresponding models. Another further step would be the extension of this approach to reduce the vibrations simultaneously to the power error control. This will be especially interesting for floating WT. Finally, it would also be desirable to implement the controller in a WT prototype.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The findings of this study have been generated by the equations and parameters cited within the article.

Funding statement

This work was partially supported by the Spanish Ministry of Science, Innovation and Universities under MCI/AEI/FEDER Project number RTI2018-094902-B-C21.

Appendix. List of abbreviations, variables, and control parameters

See Tables 3–5.

References

Abdelbaky, M.A., Liu, X., Jiang, D., 2020. Design and implementation of partial offline fuzzy model-predictive pitch controller for large-scale wind turbines. Renew. Energy 145, 981–996.

Abouheaf, M., Gueaieb, W., Sharaf, A., 2018. Model-free adaptive learning control scheme for wind turbines with doubly fed induction generators. IET Renew. Power Gener. 12 (14), 1675–1686.

Chen, P., Han, D., Tan, F., Wang, J., 2020. Reinforcement-based robust variable pitch control of wind turbines. IEEE Access 8, 20493–20502.

Elsisi, M., Tran, M.Q., Mahmoud, K., Lehtonen, M., Darwish, M.M., 2021. Robust design of ANFIS-based blade pitch controller for wind energy conversion systems against wind speed fluctuations. IEEE Access 9, 37894–37904.

Fan, Y.J., Xu, H.T., He, Z.Y., 2021. Smoothing the output power of a wind energy conversion system using a hybrid non-linear pitch angle controller. Energy Explor. Exploit. 01445987211041779.

Fernandez-Gauna, B., Fernandez-Gamiz, U., Grana, M., 2017. Variable speed wind turbine controller adaptation by reinforcement learning. Integr. Comput.-Aided Eng. 24 (1), 27–39.

Fernandez-Gauna, B., Osa, J.L., Graña, M., 2018. Experiments of conditioned reinforcement learning in continuous space control tasks. Neurocomputing 271, 38–47.

Green Peace, 2021. https://es.greenpeace.org/es/trabajamos-en/cambio-climatico/carbon/. Last (Accessed 06 November 2021).

Hosseini, E., Aghadavoodi, E., Ramírez, L.M.F., 2020. Improving response of wind turbines by pitch angle controller based on gain-scheduled recurrent ANFIS type 2 with passive reinforcement learning. Renew. Energy.

Iqbal, A., Ying, D., Saleem, A., Hayat, M.A., Mehmood, K., 2020. Efficacious pitch angle control of variable-speed wind turbine using fuzzy-based predictive controller. Energy Rep. 6, 423–427.

IRENA, 2019. Future of wind: Deployment, investment, technology, grid integration and socio-economic aspects (a global energy transformation paper), international renewable energy agency, Abu Dhabi. https://www.irena.org/-/media/Files/IRENA/Agency/Publication/2019/Oct/IRENA_Future_of_wind_2019.pdf. Last (Accessed 06 November 2021).

Jeon, T., Paek, I., 2021. Design and verification of the LQR controller based on fuzzy logic for large wind turbine. Energies 14 (1), 230.

Jove, E., González-Cava, J.M., Casteleiro-Roca, J.L., Alaiz-Moretón, H., Baruque, B., Leitão, P., et al., 2021. An intelligent system for harmonic distortions detection in wind generator power electronic devices. Neurocomputing 456, 609–621.

Li, J., Yu, T., Yang, B., 2021a. A data-driven output voltage control of solid oxide fuel cell using multi-agent deep reinforcement learning. Appl. Energy 304, 117541.

Li, J., Yu, T., Zhang, X., 2022. Coordinated load frequency control of multi-area integrated energy system using multi-agent deep reinforcement learning. Appl. Energy 306, 117900.

Li, J., Yu, T., Zhang, X., Li, F., Lin, D., Zhu, H., 2021b. Efficient experience replay based deep deterministic policy gradient for AGC dispatch in integrated energy system. Appl. Energy 285, 116386.

Mikati, M., Santos, M., Armenta, C., 2012. Modelado y simulación de un sistema conjunto de energía solar y eólica para analizar su dependencia de la red eléctrica. Rev. Iberoam. Autom. E Inf. Ind. 9 (3), 267–281.

Ngo, Q.V., Chai, Y., Nguyen, T.T., 2020. The fuzzy-PID based-pitch angle controller for small-scale wind turbine. Int. J. Power Electron. Drive Syst. 11 (1), 135.

Our World in Data, 2020. https://ourworldindata.org/renewable-energy. Last (Accessed 06 November 2021).

Pamuji, F.A., Ulil'Azmi, M., Riawan, D.C., 2021. MPPT of 1.5 kW wind turbine with pitch and voltage control based on artificial neural network. In: 2021 International Seminar on Intelligent Technology and Its Applications. SITIA, IEEE, pp. 40–45.

Perrusquía, A., Yu, W., 2021. Identification and optimal control of nonlinear systems using recurrent neural networks and reinforcement learning: An overview. Neurocomputing.

Reddak, M., Nouaiti, A., Gourma, A., Berdai, A., 2021. MPPT and pitch angle based on neural network control of wind turbine equipped with DFIG for all operating Wind Speed Regions. In: International Conference on Digital Technologies and Applications. Springer, Cham, pp. 1421–1432.

Rubio, P.M., Quijano, J.F., López, P.Z., et al., 2019. Intelligent control for improving the efficiency of a hybrid semi- submersible platform with wind turbine and wave energy converters. Rev. Iberoam. Autom. E Inf. Ind. 16 (4), 480–491.

Saénz-Aguirre, A., Zulueta, E., Fernández-Gamiz, U., Lozano, J., Lopez-Guede, J.M., 2019. Artificial neural network based reinforcement learning for wind turbine yaw control. Energies 12 (3), 436.

Saenz-Aguirre, A., Zulueta, E., Fernandez-Gamiz, U., Ulazia, A., Teso-Fz-Betono, D., 2020. Performance enhancement of the artificial neural network-based reinforcement learning for wind turbine yaw control. Wind Energy 23 (3), 676–690.

Sahoo, S., Panda, G., 2022. Control and comparison of power for a variable-speed wind turbine using fuzzy PID controller. In: Advanced Computational Paradigms and Hybrid Intelligent Computing. Springer, Singapore, pp. 385–393.

Salem, M.E., El-Batsh, H.M., El-Betar, A.A., Attia, A.M., 2021. Application of neural network fitting for pitch angle control of small wind turbines. IFAC-PapersOnLine 54 (14), 185–190.

Santos, M., 2011. An application approach of intelligent control. Rev. Iberoam. Autom. E Inf. Ind. RIAI 8 (4), 283–296.

Santoso, D.B., Pangestu, A.B., Latifa, U., Fauzi, A., Zahro, L., 2021. Pitch angle control of a wind turbine using fuzzy logic control. IOP Conf. Ser. Earth Environ. Sci. 830 (1), 012073.

Sarkar, M.R., Julai, S., Tong, C.W., Uddin, M., Romlie, M.F., Shafiullah, G.M., 2020. Hybrid pitch angle controller approaches for stable wind turbine power under variable wind speed. Energies 13 (14), 3622.

Sedighizadeh, M., Rezazadeh, A., 2008. Adaptive PID controller based on reinforcement learning for wind 677 turbine control. In: Proc. World Academy of Science, Engineering and Technology. Vol. 27. pp. 257–262.

Serrano-Barreto, C.L., Sierra-García, J.E., Santos, M., 2021. Intelligent hybrid controllers for the blade angle of floating wind turbines. In: International Workshop on Soft Computing Models in Industrial and Environmental Applications. Springer, Cham, pp. 461–470.

Sierra-García, J.E., Santos, M., 2020a. Exploring reward strategies for wind turbine pitch control by reinforcement learning. Appl. Sci. 10 (21), 7462.

Sierra-García, J.E., Santos, M., 2020b. Performance analysis of a wind turbine pitch neuro controller with unsupervised learning. Complexity 2020.

Sierra-García, J.E., Santos, M., 2021a. Lookup table and neural network hybrid strategy for wind turbine pitch control. Sustainability 13 (6), 3235.

Sierra-García, J.E., Santos, M., 2021b. Redes neuronales y aprendizaje por refuerzo en el control de turbinas eólicas. Rev. Iberoam. Autom. E Inf. Ind. 18 (4), 327–335.

Sierra-García, J.E., Santos, M., 2021c. Switched learning adaptive neuro-control strategy. Neurocomputing 452, 450–464.

Sitharthan, R., Karthikeyan, M., Sundar, D.S., Rajasekaran, S., 2020. Adaptive hybrid intelligent MPPT controller to approximate effectual wind speed and optimal rotor speed of variable speed wind turbine. ISA Trans. 96, 479–489.

Sutton, R.S., Barto, A.G., 2018. Reinforcement Learning: An Introduction. MIT Press.

Tomás-Rodríguez, M., Santos, M., 2019. Modelling and control of floating offshore wind turbines. Rev. Iberoam. Autom. E Inf. Ind. 16 (4), 381–390.

Tomin, N., Kurbatsky, V., Guliyev, H., 2019. Intelligent control of a wind turbine based on reinforcement learning. In: 2019 16th Conf. on Electrical Machines, Drives and Power Systems ELMA 2019. IEEE, pp. 1–6.

Trojaola, I., Elorza, I., Irigoyen, E., Pujana-Arrese, A., Calleja, C., 2020. The effect of iterative learning control on the force control of a hydraulic cushion. Logic J. IGPL.

Tzafestas, S.G. (Ed.), 2012. Methods and Applications of Intelligent Control. Vol. 16. Springer Science & Business Media.

Yang, B., Jiang, L., Wang, L., Yao, W., Wu, Q.H., 2016. Non-linear maximum power point tracking control and modal analysis of DFIG based wind turbine. Int. J. Electr. Power Energy Syst. 74, 429–436.

Zhang, Z., Zhang, D., Qiu, R.C., 2019. Deep reinforcement learning for power system applications: An overview. CSEE J. Power Energy Syst. 6 (1), 213–225.

Zhao, H., Zhao, J., Qiu, J., Liang, G., Dong, Z.Y., 2020. Cooperative wind farm control with deep reinforcement learning and knowledge assisted learning. IEEE Trans. Ind. Inf..