PROCEEDINGS OF ECOS 2023 - THE 36TH INTERNATIONAL CONFERENCE ON
EFFICIENCY, COST, OPTIMIZATION, SIMULATION AND ENVIRONMENTAL IMPACT OF ENERGY SYSTEMS
25-30 JUNE 2023, LAS PALMAS DE GRAN CANARIA, SPAIN

Enabling Reinforcement Learning for Flexible Energy Systems through Transfer Learning on a Digital Twin Platform

Carlotta Tubeuf[a], Felix Birkelbach[b], Anton Maly[c], Maximilian Krause[d] and René Hofmann[e]

TU Wien, Institute of Energy Systems and Thermodynamics, Vienna, Austria,
[a] carlotta.tubeuf@tuwien.ac.at, CA
[b] felix.birkelbach@tuwien.ac.at
[c] anton.maly@tuwien.ac.at
[d] maximilian.krause@tuwien.ac.at
[e] rene.hofmann@tuwien.ac.at

Abstract:

Pumped storage power plants compensate for fluctuations in the electricity grid and improve the stability through grid services. By increasing the flexibility of pumped storage power plants, they could compensate fluctuations in an even greater extent and thus accelerate the shift to a fully renewable energy system. One way to do this is to accelerate the switching between operating modes within pumped storage stations. For this, we propose to apply reinforcement learning (RL) to control the start and stop processes within a hydraulic machine. RL has been shown to outperform traditional optimal control methods, however, safety concerns are stalling research on applying RL for process control in safety-sensitive energy systems. To enable the safe and reliable transfer of the algorithm's learning strategy from a virtual test environment to the physical asset, we present a concept for applying RL via a digital twin platform. To demonstrate this concept, we set up a simulation model for the operating behavior during the start and stop processes of a lab-scale pump-turbine and validate it with experimental data. On this virtual representation, we test the application of RL to optimally control the blow-out process within pump-turbines. We present the structure of the deep Q-learning (DQN) RL algorithm we trained and the necessary problem formulations. Our results show that the DQN algorithm is suitable for finding the optimal operating strategy to blow-out the pump-turbine runner. We discuss the viability of our approach for the control of a pump-turbine and outline the next steps to test RL on a lab-scale model machine.

Keywords:

Reinforcement Learning, Digital Twin, Hydro Power, Process Control, Pump-Turbine, Transfer Learning.

1. Introduction

To reach the transition to a clean energy future, renewable energy systems, and especially wind and solar power technologies, will be expanded massively over the next years [1]. The energy sector is thus confronted with the growing share of volatile renewable energy systems in the grid. To balance out fluctuations, other energy sources and storage systems, such as pumped storage power plants, will need to increase not just in capacity but also in flexibility [2].

An approach to make pumped hydro storage systems more flexible is the acceleration of the switching between operating modes of pumped hydro machine units. Pumped storage power plants can be equiped with ternary sets, consisting of a pelton or francis runner and a storage pump, or with reversible pump turbines, where the machine unit can act as a turbine as well as as a pump. When switching from turbine to pump mode in pump-turbines, the runner is typically being blown-out, i.e. the water is being displaced by air, to minimize the start-up torque. This blow-out process is also necessary for both generator types when operating in synchronous condenser mode, which is used for compensating reactive power in the power grid [3].

To reveal optimization potential for faster changes of operating conditions in general and the blow-out process in particular, we propose to use a reinforcement learning (RL) algorithm for process control within the machine unit. RL is a type of machine learning (ML) in which an agent interacts directly with its environment. It aims to learn an optimal decision policy, guided by a scalar reward signal [4]. The application of RL in industrial control settings has received a lot of attention in recent years because it has been shown to outperform traditional optimal control methods [5]. However, safety concerns limit most use cases to simulated environments [6]. While research in the areas of robotics (e.g. [7, 8]) and manufacturing (e.g. [9]) seems to be leveraging the transition

https://doi.org/10.52202/069564-0289

of simulated to real-world applications of RL for process control, the implementation of an RL algorithm to critical infrastructure, such as hydropower systems, is still far from being realized [6].

To enable RL for flexible energy systems, we recently proposed a three-step learning method that uses transfer learning (TL) to transfer a pre-learned RL algorithm via a digital twin (DT) platform from a simple data model over the virtual representation of the machine to the real world machine unit [10]. Based on this method, which will be explained in Section 2.1., we will present the simulation model that was set up to act as the virtual representation of the lab-scale reversible pump-turbine where we plan to test the control of the blow-out process through RL in the future. We then describe the structure of the RL algorithm that we used and discuss our results for implementing the RL agent to control the blow-out process within the simulation model. Finally, we give an outlook on future research and draw a conclusion.

2. Methods

2.1. Reinforcement Learning on a Digital Twin Platform

When operating critical infrastructure, reliability and safety are crucial. Simultaneously, the replacement of standard controllers with RL algorithms to control processes is gaining in attention [6, 11, 12], which may be of interest for continuously revealing optimization potential and automating a flexible operation within energy systems. Hence, to enable the application of RL for process control within safety-sensitive energy systems, requirements for the trustworthiness of the RL algorithm need to be established and satisfied. Therefore, we propose a three-step learning method, that combines the benefits of RL and TL on a DT platform, as presented in our recent journal publication [10]. Figure 1 shows the concept for applying RL on a DT platform.

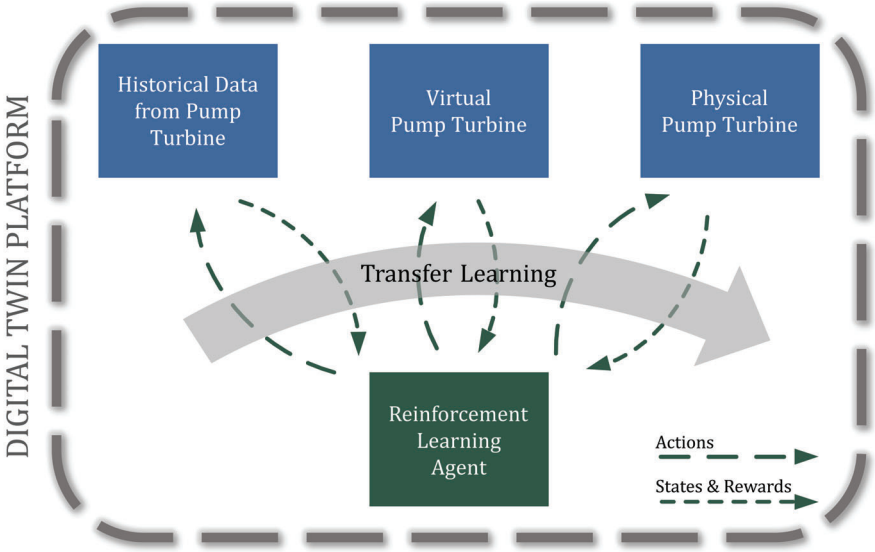


Figure 1: Concept for the application of reinforcement learning (RL) on a digital twin (DT) platform, adapted from [10].

Firstly, pre-training is done with a basic data model, followed by, secondly, training on an exact simulated replica of the actual asset. Thirdly, the pre-learned strategy is being adapted to the real machine unit. This pre-training can substantially reduce technical safety concerns related to learning and operating the actual power system by efficiently limiting the RL agent's action space. Figure 2 shows the general concept of a DT platform, as proposed by Kasper et al. [13], that integrates all three environments for the three-step RL approach: the historical data model, the virtual replication, and the physical unit, allowing TL to be used as a service to continuously enhance the RL agent's strategy.

2.2. Simulation Model

The virtual entity of our DT platform consists of a simulation model of the pump-turbine test rig at the laboratories of the Institute of Energy Systems and Thermodynamics (IET) at TU Wien (see Figure 3). The model pump-turbine consists of seven runner blades and is equipped with 20 guide vanes and 20 stay vanes. The simulation model is built using the Simscape language together with blocks from the Simscape standard libraries within the MATLAB/Simulink environment [14–16]. The model allows for the simulation of various operating

https://doi.org/10.52202/069564-0289

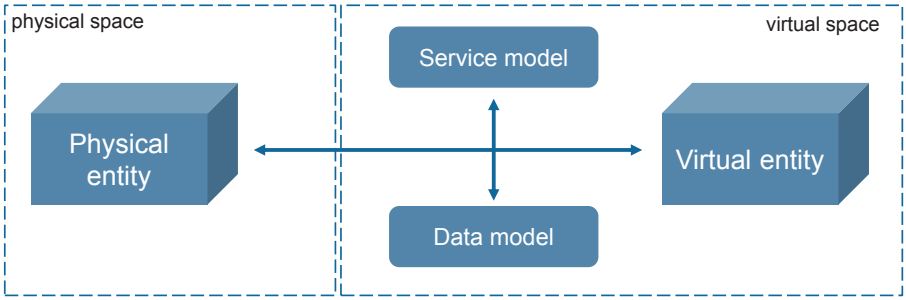


Figure 2: Schematic representation of a generic DT platform, adapted from [13].

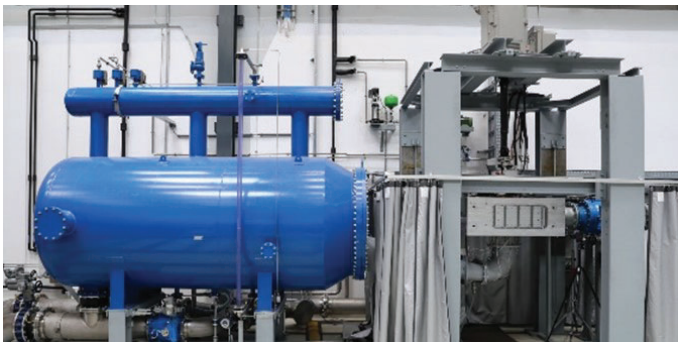


Figure 3: Test rig with scale reduced model of a radial pump-turbine at the hydraulic lab of IET, from [10].

conditions. The pump-turbine characteristics were modeled following the assumption of the affinity laws for the behavior of pumps and turbines with variable speeds. Equation 1 describes the linear dependence between the flow rate Q and the rotational speed n, and the cubic relation between n and the pressure difference built up by the pump-turbine Δp_{PT}.

$$\frac{Q_1}{Q_2} = \frac{n_1}{n_2} \approx \sqrt{\Delta p_{\mathrm{PT}}} \tag{1}$$

To make up for deviations between the simulated and the measured values for the flow rate Q, a representative fitting rate η, similar to an efficiency rate, depending on the rotational speed was introduced. The resulting Equation 2 for the pressure difference Δp_{PT} therefore comprises the efficiency rate $\eta(n)$, the scaling factors k_{PT} and k_ω, all compensating for actual flow conditions in the real machine unit, and the angular speed ω.

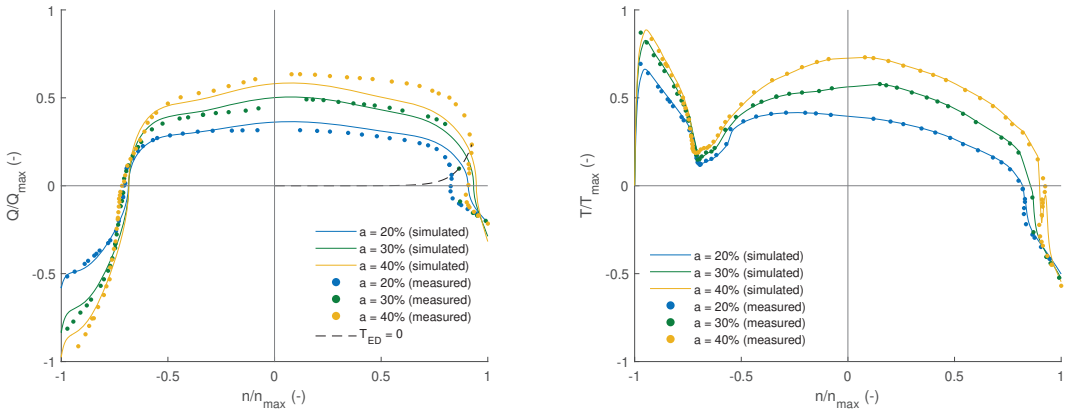
$$\Delta p_{\mathrm{PT}} = \eta(n) k_{\mathrm{PT}} (\omega - k_\omega)^2 \tag{2}$$

For the calculation of the shaft torque, the experimentally captured Tn- characteristics were directly implemented within the simulation model. To account for the influence of the water level in the draft tube cone on the torque, Equation 3 calculates the shaft torque T by multiplying the measured values T_{ch}, which are dependent on the rotational speed n and the the guide vane opening a, with the fraction between the actual water level x and the maximum water level in the draft tube cone x_{max}.

$$T = \frac{x}{x_{\mathrm{max}}} T_{\mathrm{ch}}(n, a) \tag{3}$$

The simulation model sufficiently replicates the measurements on the real model pump-turbine for the flow rate Q and shaft torque T for different guide vane openings a and rotational speeds n. Figures 4b and 4a show the comparison of the simulated curves with the measured data for the Qn- and Tn-characteristics, respectively.

The relative error for the simulated flow rate compared to the measured data is shown in Figure 5 for an exemplary guide vane opening of a = 30%. The deviations between simulated and measured data are comparable for other guide vane openings. Overall, the model is able to simulate three of the 4-quadrant characteristics (pump, pump brake and turbine) within acceptable relative error tolerances of $\pm$20%. Only when operating as turbine break and reverse pump, the relative error exceeds 20%. As our first use case for enabling increased flexibility within machine operation is on the pump start-up and the operation in synchronous condenser mode,



(a) *Qn*-characteristics (b) *Tn*-characteristics

Figure 4: Comparison of *Qn*- and *Tn*-characteristics from measured data with simulated curves for different guide vane openings.

the focus of the simulation is on operation as a pump and the blow-out process with speeds from $-n_{max}$ to around $-0.7n_{max}$. Deviations between the simulated and the experimentally measured data are thus more accepted in the other operation modes, resulting in non-sufficient representation of the operating range below the "zero torque" ($T_{ED} = 0$), the so called "S"-characteristic, as it can be seen in Figures 4 and 5. Here, the

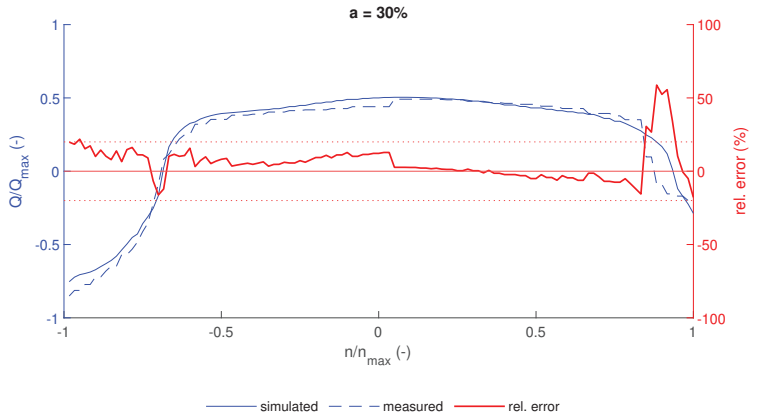


Figure 5: Relative error for the simulated flow rate compared to measurement data.

operating condition is not clearly defined and the area should be avoided in real operation. Nevertheless, the model is easily adaptable and correct representation of all pump-turbine operation modes needs to be ensured before the model is eligible as the complete virtual entity of the DT platform.

The blow-out of the runner is modeled through the lowering of the water level when air is blown into the draft tube. The machine is considered as blown-out as long as the water level is below the threshold $x_{\text{blow-out}}$. If the water level drops too low, air can leak into the tailwater vessel. Therefore, the water level should never fall below the critical level x_{crit}. Figure 6 illustrates the relevant water levels in the model pump-turbine, and Table 1 lists the according values.

2.3. Reinforcement learning algorithm

The objective of an RL agent is to interact with its environment to determine the best possible strategy, which is referred to as the optimal policy [4]. As illustrated in Figure 7, the agent makes a decision on what action to take at each time step t, resulting in a change in the environment. The environment's current state S_t is passed to

https://doi.org/10.52202/069564-0289

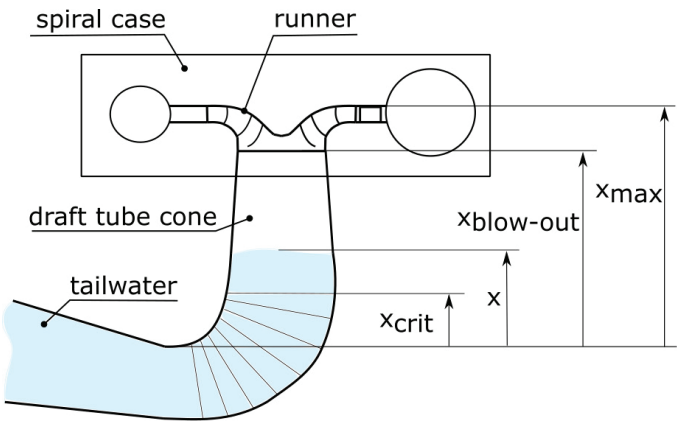


Figure 6: Illustration of relevant water levels inside pump-turbine.

Table 1: Water level values in draft tube of model pump-turbine.

x_{max}	0.807 m
$x_{blow-out}$	0.639 m
x_{crit}	0.227 m

the agent, along with a reward R_t that serves as a measure of the state's quality and provides feedback for the agent's learning algorithm. Based on the observation of the new state, the agent determines the subsequent action. By repeating this sequence, the agent acquires knowledge on how to effectively associate states with actions, with the objective of maximizing the cumulative rewards obtained over time (also known as return G). After a sufficient number of training sessions, this process leads to the development of the optimal policy π^*, with $\pi(A|S)$ indicating the probability of selecting action A when presented with state S [10].

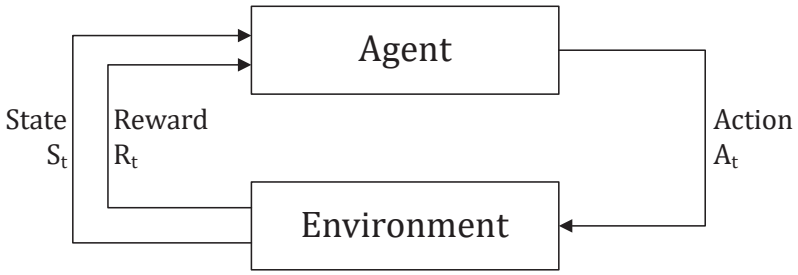


Figure 7: Agent-environment interaction within an RL algorithm, adapted from [4].

During training, an RL algorithm needs to balance between exploring the state space through selecting random actions and exploiting of actions that have already proven to yield high rewards. Only through exploiting on past actions, the agent's performance can be improved. However, prematurely focusing on exploitation may lead to the algorithm settling on a sub-optimal policy [17]. Therefore we use epsilon-greedy exploration following an epsilon decay function. In this way, the exploration probability ϵ decreases continuously during training, ensuring that the agent explores the whole state space in the beginning, but still manages to converge to the optimal policy when training progresses.

For our use case, we use a deep Q-network (DQN) training algorithm, which is a value-based RL algorithm that trains a so-called critic $Q(S, A; \phi)$ with parameters ϕ to predict the return for a given state S and action A. During training, the agent adjusts the parameters in ϕ. When training is finished, the optimal policy can be derived from the trained value function approximator, the critic $Q(S, A)$, with now tuned parameter values ϕ [18]. The general training algorithm for a DQN agent is described in Algorithm 1.

https://doi.org/10.52202/069564-0289

Algorithm 1 Deep Q-Network (DQN) Training Algorithm [18]

Initialize critic Q with random parameter values ϕ
for each episode **do**
  for each training time step **do**
    With probability ϵ select a random action A
    otherwise, select the action, for which the critic value function is greatest:
    $A = \arg\max_A Q(S, A; \phi)$
    Execute action A. Observe the reward R and the next state S'
    Store the experience (S, A, R, S') in the experience buffer
    Sample a random mini-batch of M experiences (S_i, A_i, R_i, S'_i) from the experience buffer
    if S'_i is a terminal state **then**
      Set value function target $y_i = R_i$
    else
      Set $y_i = R_i + \gamma \max_{A'} Q(S'_i, A', \phi)$
    end if
    Update the critic parameters ϕ by one-step minimization of the loss L across all sampled experiences:
    $L = \frac{1}{M} \sum_{i=1}^{M} (y_i - Q(S_i, A_i; \phi))^2$
    Update the probability threshold ϵ for selecting a random action based on the ϵ-decay rate
  end for
end for

3. Results

The goal of our use case is to have the RL algorithm control the blow-out process for pump start-up within the virtual model of our scale reduced pump-turbine in the lab of IET. In doing so, we seek to demonstrate how RL can enhance the flexibility of hydropower systems, by training the agent to minimize the usage of compressed air while blowing out the machine as fast as possible.

As RL algorithm, a DQN agent from the MATLAB Reinforcement Learning Toolbox [18] with the default vector Q-value deep neural network as critic was used. The DQN agent has a discrete action space, $A = [0, 1]$. If the action $A = 0$, no air is blown into the draft tube and if action $A = 1$, air gets blown into the draft tube with a constant pressure of p_{air}. At each time step, the agents receives the state of the environment $S = [x(t), A(t-\Delta t)]$ through a continuous value for the current water level x and the binary value for the previous action A. This informs the agent whether the water level is rising or falling, allowing it to make an informed decision on the next action. The reward at each training step is calculated with a reward function that consists of four weighted terms, as described in Equation 4. Hereby, $R_{\text{waterlevel}}$ is positive if the water level x is between $x_{\text{blow-out}}$ and x_{crit} and negative otherwise. R_{air} accounts for the penalty the agent receives every time $A = 1$, indicating that air is blown into the draft tube cone. $R_{\text{switching}}$ encourages mores stable operation through penalizing the agent whenever it switches the air valve. If the compressor's limit is reached, i.e. if the total mass flow of air blown into the draft tube during the whole training episode reaches the limit of $\dot{m}_{\text{air}} = 1\text{kg}$, the agent receives a high penalty $R_{\text{compressor}}$ and the episode is aborted.

$$R = w_1 R_{\text{waterlevel}} - w_2 R_{\text{air}} - w_3 R_{\text{switching}} - w_4 R_{\text{compressor}} \tag{4}$$

The final cumulative reward, the return, for one training episode can be calculated by summing up over the reward received during each training time step t (Equation 5).

$$G = \sum_{i=1}^{t_{\text{end}}} R(t) \tag{5}$$

All the important parameter and hyperparameter settings for the training of the DQN agent are listed in Table 2. Note that the guide vane opening $a = 1\text{mm}$ and not 0, as it would be expected during the blow-out process. This intentional gap between the guide vanes should indicate leakages, as they are common in real world constructions, leading to air dissipating and therefore the need to repeatedly blow air into the draft tube to remain in blown-out condition.

Figure 8 shows the training progress of the DQN agent for learning how to control the blow-out process while balancing between rapidly reaching and remaining in blown-out operation mode and, simultaneously, minimizing the air mass blown into the draft tube. The large fluctuations for the first 200 training episodes represent the exploration phase of the learning algorithm, during which the agent chooses many random actions to explore the state space and evaluate the value function. As training progresses, the exploration probability ϵ decreases and the average return converges. The drops in the episode return curve in episodes 315 and 433 come from further random action decisions, as the probability for selecting a random action over the action for

https://doi.org/10.52202/069564-0289

Table 2: (Hyper-)Parameter settings for training of the DQN agent.

parameter	symbol	value
simulation options		
maximum simulation time	t_{sim}	120 s
air pressure	p_{air}	8 bar
guide vane opening	a	0.001 m
rotational speed motor-generator	n	0 rpm
agent options		
exploration probability	ϵ	0.5
minimum exploration probability	ϵ_{min}	0.001
exploration probability decay rate	ϵ_{decay}	0.0001
learn rate	γ	0.01
discount factor	α	0.99
mini batch size	M	64
training options		
maximum episodes		1000
averaging window length		20
stop training criterion = average return		91

which the critic value function is greatest, never decreases to 0 but converges to a minimum value ϵ_{min} = 0.1%. During numerous previous training sessions it was discovered that the highest achievable return is limited to G_{max} = 93. Therefore, a return of G_{stop} = 91 averaged over 20 training episodes was chosen as termination criterion. After the 542 training episodes, this criterion was reached and training was stopped.

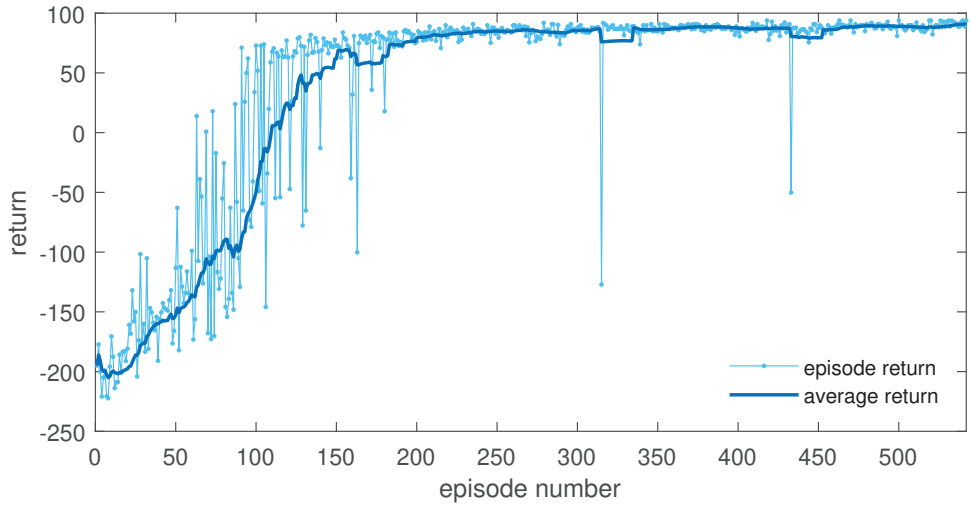


Figure 8: Episode and average return over training episodes.

The simulation results for the blow-out process when controlled by the trained DQN agent are shown in Figure 9. The rotational speed of the motor-generator was set to n = 0rpm, for simulating the machine at standstill. Since a leakage between the rotational guide vanes of a = 1mm was assumed, the flow rate Q is not 0, but a constant water flow of roughly 18 l/s was calculated by the simulation model. The final strategy of the RL algorithm for controlling the blow-out process can be seen in the sub-figures for the action, the mass flow rate and the water level over time. Air gets blown into the draft tube for a certain amount of time (A = 1). Subsequently, the air valve is being closed again (A = 0) and the water level rises again due to leakage effects. When the threshold $x_{blow-out}$ is reached, the valve opens again (A = 1). This sequence is repeated until the maximum simulation time of t_{sim} = 120s is reached.

https://doi.org/10.52202/069564-0289

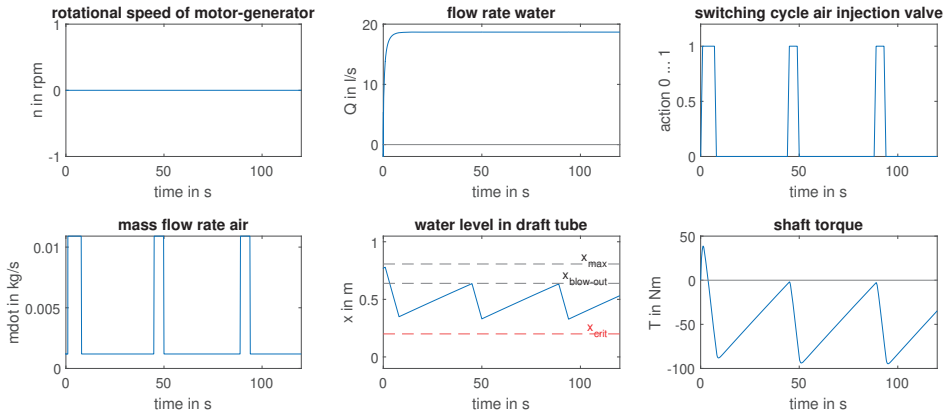


Figure 9: Simulation results for the blow-out process.

4. Discussion

The control of the blow-out process within the simulation model of a model pump-turbine through a DQN RL agent was successfully implemented. However, the algorithm' optimal policy does not yet reveal much optimization potential. Nevertheless it results in a rather logical blow-out sequence that could have also been achieved easily by a simple hysteresis controller. The results therefore show that the DQN agent and our RL problem formulation is suitable for finding the optimal blow-out operation strategy.

Considering that the current water level in the draft tube of the model pump-turbine at our lab facilities is hard to be measured directly, future research will deal with finding a solution for representing the blow-out conditions in the simulation model more accurately. Then, the control of the process will most probably increase in complexity and the optimal policy won't be as straightforward. We assume TL to be beneficial in transferring the DQN agent's policy to handle model adaptations.

Further, in this work we only considered the blow-out process during machine downtime. Blowing out a rotating runner, as it is being done for operation in synchronous condenser mode, may result in different findings. Investigations of different operation parameters are thus recommended and will be a part of our future research.

5. Conclusion

We presented a method to enable the use of RL for process control in pumped hydro storage systems. Concerns for letting a ML algorithm interact with safety-sensitive industrial equipment are among the biggest research barriers for RL for process control. We argue that through the transfer of a pre-learned RL algorithm through the use of a DT platform, safety concerns can be reduced and reliability of the RL algorithm's policy can be increased. In this paper, we showed the first results for the training of a DQN RL agent on a simulation model of a reversible pump-turbine, which acts as the virtual entity of our DT platform. The model was fitted to represent the behavior of a lab-scale model pump-turbine, which is located at the laboratory of the IET at TU Wien with satisfactory accuracy. Training of the RL algorithm was successfully carried out. The results confirm the expected optimal operation of the blow-out process. Future research will address increasing model complexity, exploration of different action and state spaces for the learning of the agent and, ultimately, the transfer of the RL algorithm to the model machine in the lab and use the autonomously learned optimal strategy to control the blow-out process.

Acknowledgments

The authors thank Christian Bauer, head of the Research Unit of Fluid Flow Machinery, for his guidance and advice. The authors acknowledge the support of this work through the women's promotion program of the Faculty of Mechanical and Industrial Engineering (MWBF) at TU Wien.

Abbreviations

DQN Deep Q-Learning

DT Digital Twin

IET Institute of Energy Systems and Thermodynamics, TU Wien

https://doi.org/10.52202/069564-0289

ML Machine Learning

RL Reinforcement Learning

TL Transfer Learning

Nomenclature

a guide vane opening, m

A action

G return

k_{PT} fitting constant, $\mathrm{Pas}^2/\mathrm{rad}^2$

k_ω fitting constant, rad/s

L loss

M mini batch size

$\dot{m}$ mass flow rate, kg/s

n rotational speed, rpm

p pressure, bar

Q flow rate, m^3/s

R reward

S state

t time step, s

t_{sim} simulation time, s

T shaft torque, Nm

T_{ED} torque factor, 1

w weighting factor

x water level, m

y value function

Greek symbols

α discount factor

ϵ exploration probability

ϵ_{decay} exploration probability decay rate

η efficiency

γ learn rate

ϕ Q-learning parameter

Δ difference

ω angular speed, rad/s

Subscripts and superscripts

air air

blow-out blow-out condition

ch characteristic

crit critical

max maximum

min minimum

PT pump-turbine

References

[1] T. Krutzler, H. Wiesenberger, C. Heller, M. Gössl, G. Stranner, A. Storch, H. Heinfellner, R. Winter, M. Kellner, and I. Schindler, "Szenario Erneuerbare Energie 2030 und 2050," tech. rep., Umweltbundesamt, Wien, 2016.

[2] M. Gimeno-Gutiérrez and R. Lacal-Arántegui, "Assessment of the European potential for pumped hydropower energy storage based on two existing reservoirs," *Renewable Energy*, vol. 75, pp. 856–868, Jan. 2015.

[3] A. Maly and C. Bauer, "Experimental investigation of a free surface oscillation in a model pump-turbine," *IOP Conference Series: Earth and Environmental Science*, vol. 774, p. 012068, Jan. 2021. Publisher: IOP Publishing.

[4] R. S. Sutton and A. Barto, *Reinforcement learning: An introduction*. Adaptive computation and machine learning, Cambridge, Massachusetts; London, England: The MIT Press, second edition ed., Jan. 2018.

[5] J. Shin, T. A. Badgwell, K.-H. Liu, and J. H. Lee, "Reinforcement Learning – Overview of recent progress and implications for process control," *Computers & Chemical Engineering*, vol. 127, pp. 282–294, Jan. 2019.

[6] R. Nian, J. Liu, and B. Huang, "A review On reinforcement learning: Introduction and applications in industrial process control," *Computers & Chemical Engineering*, vol. 139, p. 106886, Jan. 2020.

[7] H. Ju, R. Juan, R. Gomez, K. Nakamura, and G. Li, "Transferring policy of deep reinforcement learning from simulation to reality for robotics," *Nature Machine Intelligence*, vol. 4, pp. 1077–1087, Dec. 2022. Number: 12 Publisher: Nature Publishing Group.

[8] E. Salvato, G. Fenu, E. Medvet, and F. A. Pellegrino, "Crossing the Reality Gap: A Survey on Sim-to-Real Transferability of Robot Controllers in Reinforcement Learning," *IEEE Access*, vol. 9, pp. 153171–153187, 2021.

[9] J. Li, D. Pang, Y. Zheng, and X. Le, "Digital Twin Enhanced Assembly Based on Deep Reinforcement Learning," in *2021 11th International Conference on Information Science and Technology (ICIST)*, pp. 432–437, IEEE, 2021.

[10] C. Tubeuf, F. Birkelbach, A. Maly, and R. Hofmann, "Increasing the Flexibility of Hydropower with Reinforcement Learning on a Digital Twin Platform," *Energies*, vol. 16, p. 1796, Jan. 2023. Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

[11] N. P. Lawrence, M. G. Forbes, P. D. Loewen, D. G. McClement, J. U. Backström, and R. B. Gopaluni, "Deep reinforcement learning with shallow controllers: An experimental application to PID tuning," *Control Engineering Practice*, vol. 121, p. 105046, Apr. 2022.

[12] S. Spielberg, R. Gopaluni, and P. Loewen, "Deep reinforcement learning approaches for process control," in *2017 6th International Symposium on Advanced Control of Industrial Processes (AdCONIP)*, pp. 201–206, May 2017.

[13] L. Kasper, F. Birkelbach, P. Schwarzmayr, G. Steindl, D. Ramsauer, and R. Hofmann, "Toward a Practical Digital Twin Platform Tailored to the Requirements of Industrial Energy Systems," *Applied Sciences*, vol. 12, p. 6981, Jan. 2022. Number: 14 Publisher: Multidisciplinary Digital Publishing Institute.

[14] The MathWorks Inc., "MATLAB version: 9.13 (R2022b)," 2023.

[15] The MathWorks Inc., "Simulink version: 10.6 (R2022b)," 2023.

[16] The MathWorks Inc., "Simscape version: 5.4 (R2022b)," 2023.

https://doi.org/10.52202/069564-0289

[17] J. Langford, “Efficient Exploration in Reinforcement Learning,” in *Encyclopedia of machine learning and data mining* (C. Sammut and G. I. Webb, eds.), Springer Reference, pp. 389–392, New York, NY: Springer, second edition ed., Jan. 2017.

[18] The MathWorks Inc., “Reinforcement Learning Toolbox,” 2023.