Improving a Proportional Integral Controller with Reinforcement Learning on a Throttle Valve Benchmark

Paul Daoudi[1], Bojan Mavkov[2], Bogdan Robu[3], Christophe Prieur[3], Emmanuel Witrant[3], Merwan Barlier[1] and Ludovic Dos Santos[4]

***Abstract*— This paper presents a learning-based control strategy for non-linear throttle valves with an asymmetric hysteresis, leading to a near-optimal controller without requiring any prior knowledge about the environment. We start with a carefully tuned Proportional Integrator (PI) controller and exploit the recent advances in *Reinforcement Learning (RL) with Guides* to improve the closed-loop behavior by learning from the additional interactions with the valve. We test the proposed control method in various scenarios on three different valves, all highlighting the benefits of combining both PI and RL frameworks to improve control performance in non-linear stochastic systems. In all the experimental test cases, the resulting agent has a better sample efficiency than traditional RL agents and outperforms the PI controller.**

I. Introduction

Throttle valves are essential components in various industrial processes such as chemical plants, oil refineries, and power generation. Accurate control of these valves is crucial for maintaining the optimal flow rate of fluids and thus ensuring the efficient functioning of the entire system. However, controlling throttle valves is complex due to their non-linear behavior, stochasticity, and asymmetric hysteresis. These factors make it challenging to design an optimal controller that can handle the complexities of the valve system.

In this work, we focus on the regulation of a butterfly valve for car engines, crucial for modulating the amount of air supplied to the combustion chamber by adjusting a disc's rotation angle. Given the complexities induced by static friction and the non-linearities of the dynamics, the control community has explored various advanced strategies. For example, non-linear control approaches include an adaptive pulse control method leveraging a non-linear dynamic model accounting for friction and aerodynamic torque [1], a discrete-time sliding mode control for robust tracking [2], and a non-linear approach combining a Proportional-Integral-Derivative (PID) controller with a feedback compensator [3]. Additionally, adaptive control techniques [4], [5], [6], Linear Parameter-Varying (LPV) modeling, and mixed constrained H_2/H_∞ control strategies [7] have been proposed. Several works also exploit deep learning techniques. For instance, [8] presents a neural network-based sliding mode controller for electronic throttle valves. A control strategy using RL algorithms is advocated in [9] where the valve is controlled using the Deep Deterministic Policy Gradient (DDPG) [10] algorithm. However, the application of the algorithm is limited to simulations.

Our contribution extends the existing body of work by examining the combination of a classical PI control strategy with RL for throttle valve regulation. The RL framework is popular due to its ability to solve complex non-linear control problems without requiring an explicit model of the system [11], [12]. However, despite some exceptions such as balloon navigation [13] and plasma control in Tokamaks [14], RL has only been applied in simulated systems [15], [16], including the aforementioned work [9]. One major obstacle in directly applying RL to real-world systems is the need for a large amount of data and repetitive experiments to learn a good policy, as RL agents start in an unknown environment with no prior information available [17].

Only a few recent works combine Optimal Control (OC) and RL to reduce the number of data required by RL agents to learn a good policy [18].

The approaches of [19] and [20] propose a switching mechanism between an LQR controller and an RL agent, testing these approaches in simulated environments such as a pendulum and Acrobot. Some approaches embed a feedback controller within an RL policy to enhance sample efficiency [21], [22]. However, these methods do so without restraining the search space for the control input, nor without modifying the learning of the Q-function, which may be essential to avoid extrapolation errors [23], [24]. Similar to these previous methods, we aim to combine the strengths of OC and RL to build a controller for the throttle valve. For this, we adapt the recent area of Reinforcement Learning with Guides [25], [26], [27] to our setting, resulting in an algorithm that optimizes a Proportional Integral (PI) controller to guide the RL agent in navigating the control space efficiently. By reducing the search space for learning, the PI guide minimizes the data requirements and accelerates the learning process. We stress that our method is fundamentally different than those proposed by [28], [29], [30], which use RL to tune the gains of a feedback PI controller. Here, the PI controller is fixed and is used to guide the RL agent to improve its

[1] Paul Daoudi and Merwan Barlier are with the Noah's Ark Laboratory of Huawei Technologies, Paris, France. Email: paul.daoudi1@huawei.com and merwan.barlier@huawei.com

[2] Bojan Mavkov is with Université Côte d'Azur, CNRS, I3S, Nice, France. Email: bojan.mavkov@univ-cotedazur.fr

[3] Bogdan Robu, Christophe Prieur and Emmanuel Witrant are with Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, Grenoble, 38000, France Email: bogdan.robu@univ-grenoble-alpes.fr, christophe.prieur@gipsa-lab.fr and emmanuel.witrant@univ-grenoble-alpes.fr

[4] Ludovic Dos Santos is with Criteo AI Lab, Paris, France. Email: l.dossantos@criteo.com

Manuscript 112 submitted to 2024 8th IEEE Conference on Control Technology and Applications (CCTA). Received February 16, 2024.

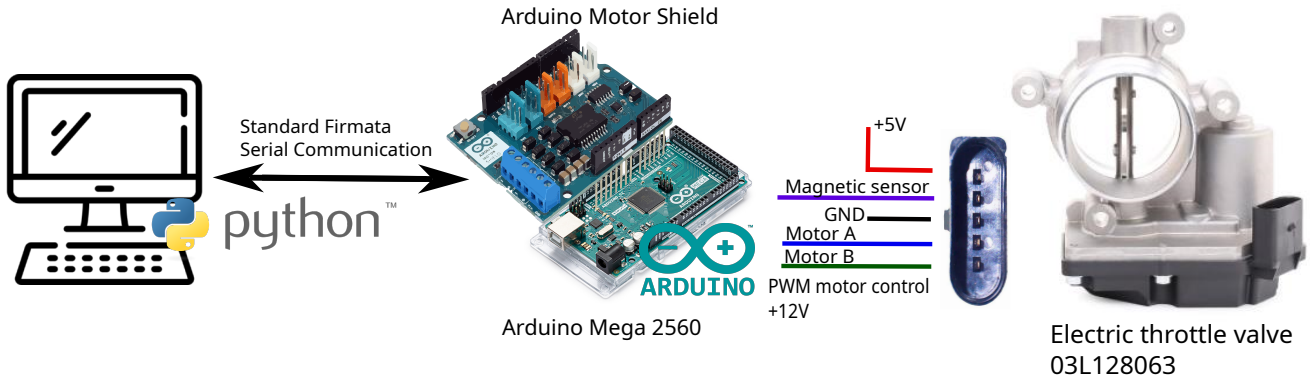

Fig. 1: Experimental test bench: hardware configuration and wiring of the electric throttle valve.

sample efficiency.

We conduct empirical validation of our approach on the throttle valve by testing the PI controller, a state-of-the-art RL agent, and their combination under a variety of industry-relevant use cases. This empirical validation is repeated on three valves having the same commercial reference but possessing slightly different physical properties. Our results demonstrate that the combination of PI and RL produces an effective agent in all settings, achieving near-optimal control with fewer data samples compared to using a traditional RL agent alone.

The paper is structured as follows. Section II describes the experimental setup and outlines the physical properties of the considered throttle valves. Section III formulates our objectives and introduces two control methods that are commonly used for similar systems. Section IV proposes our hybrid control algorithm, blending the strengths of classical and RL methodologies. Finally, Section V presents our experimental findings, evaluating the strengths of the proposed controllers according to different criteria.

II. The Throttle Valve System

This section presents a detailed overview of our experimental setup for controlling throttle valves, explores their physical properties, and formalizes the objective.

A. Experimental setup

The throttle valves investigated have the commercial reference 03L128063 and included in an experimental test bench for control as detailled in [6]. They feature a rotational spring on the shaft of the valve plate, exerting a counteractive torque against the motor's torque to control the plate's angular position. In this study, we consider 3 valves coming from the above reference.

To regulate the opening angle, a Pulse Width Modulation (PWM) ranging from 0 to 100% is generated and modulates the input voltage from 0 to 12 Volts. The valves are all equipped with a magnetic angle sensor. We connect each valve to an Arduino Mega 2560® that controls the input and allows monitoring and analyzing the system performance. The motor control is enabled by the dual full-bridge driver *Arduino Motor Shield* attached to the Arduino board. In addition, each Arduino is connected to a Python interface that is necessary to build and train the RL agent. For all the valves, the sampling time for data acquisition and feedback control is 50 milli-seconds. The scheme of the experimental test bench is presented in Figure 1.

B. A non-linear stochastic system

This type of valve is particularly known for its non-linear stochastic dynamics with asymmetric hysteresis, which we confirm in the following experiment. We select an input range from 0% to PWM_{MAX}% and generate an increasing then decreasing sequence of steps covering this range, by increments of 5%, and apply it to the valve. Despite the valves coming from the same commercial reference, we notice that the required value PWM_{MAX} to turn the valve's angular plate to 0 degrees is different from one valve to another [6]. It is $\text{PWM}_{\text{MAX}} = 0.65\%$ for Valve 1, $\text{PWM}_{\text{MAX}} = 0.70\%$ for Valve 2 and $\text{PWM}_{\text{MAX}} = 0.45\%$ for Valve 3. The experiment is repeated 5 times, and we present the resulting data in Figure 2. It can be seen that for all valves we have an asymmetric hysteresis as well as the stochastic nature of the dynamics. For a given PWM input, the resulting angles vary significantly between each trial.

These results underlines the necessity of utilizing advanced control strategies to effectively manage the intricacies of the dynamics of the valve. In addition, it highlights that each valve requires a slightly different controller to work.

C. Problem statement and notations

Considering the experimental set-up and the stochastic nature of the valves' dynamics, the problem is modeled as a discounted Markov Decision Process (MDP) $(\mathscr{X}, \mathscr{U}, c, P, \gamma)$. Let $\Delta(\cdot)$ be the simplex on any given space $(\cdot)$, and define $c : \mathscr{X} \times \mathscr{U} \to \mathbb{R}$ the cost function, $P : \mathscr{X} \times \mathscr{U} \to \Delta(\mathscr{X})$ the transition probabilities, and $\gamma \in [0,1)$ the discount factor. At each time step t, the agent receives an observation $x_t \in \mathscr{X}$ and selects a control $u_t \in \mathscr{U}$. The system is then updated with $x_{t+1} \sim P(\cdot|x_t, u_t)$.

In this formalism, the objective is to find the optimal controller $\pi^* : \mathscr{X} \to \Delta(\mathscr{U})$ that minimizes the expected discounted cumulative cost $V^\pi(x) = \mathbb{E}_P\left[\sum_{t=0}^{\infty} \gamma^t c(x_t, u_t) | x_0 = x, u_t \sim \pi(\cdot|x_t)\right]$ for each observation

Manuscript 112 submitted to 2024 8th IEEE Conference on Control Technology and Applications (CCTA). Received February 16, 2024.

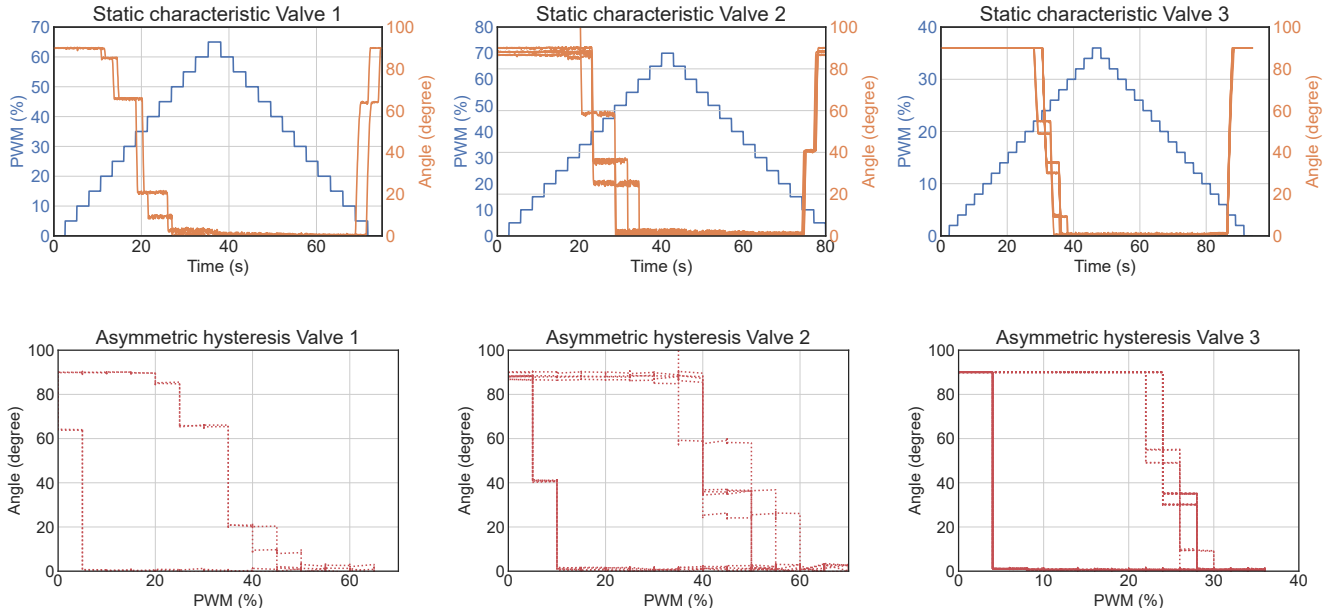

Fig. 2: Steady state analysis for the 3 valves.

$x \in \mathscr{X}$ with as few data samples as possible. The function V^π is known as the Value function of policy π, and we define the associated Q-value function as $Q^\pi(x,u) = \mathbb{E}_P\left[\sum_{t=0}^{\infty} \gamma^t c(x_t,u_t) | x_0 = x, u_0 = u, u_t \sim \pi(\cdot|x_t)\right]$.

The goal of this work is to develop a throttle valve controller $\pi : \mathscr{X} \to \Delta(\mathscr{U})$ that sets the angle of the valve's plate to a chosen angle from any initial position. The controller must not only perform optimally in nominal operating conditions by minimizing cumulative costs but also demonstrate robustness and stability under various real-world scenarios. Specifically, the controller must be able to reject external disturbances that may affect both the input and output signals, while exhibiting minimal overshoot. Bearing this information in mind, we associate the different components of the MDP with the valve. At time t, let α_t be the angle of the valve's plate and α_{ref_t} be the (desired) reference angle.

Furthermore:

- the observation x_t at time t is composed of the reference angle α_{ref_t}, the previous and current angles α_{t-1} and α_t, and the previous control input u_{t-1};
- the control u_t at time t is the PWM (%) input. To avoid saturation effects, we set it to $\mathscr{U} = [0, 0.8]$ for Valve 1 and 2, and to $\mathscr{U} = [0, 0.6]$ for Valve 3;
- the objective is to find a controller $\pi : \mathscr{X} \to \Delta(\mathscr{U})$ that sets the angle of the valve's plate to a chosen angle from any initial position. Hence, the cost function c is set to $c(x_t,u_t) = \|\alpha_t - \alpha_{\text{ref}_t}\|_2$;
- the controller must be optimal in all states and take into account long-term performance, so the discount factor γ is set to the high value of 0.99;
- the transition probabilities P are unknown;
- to enhance the convergence of RL algorithms, the system is made episodic, with each episode lasting for 100 time-steps, which represents 5 seconds. During each episode, a single random reference angle α_{ref_t} is generated, and the primary objective is to set the throttle valve's position to this specific angle.

III. Traditional techniques

In this section, we detail two traditional techniques that will be used to achieve our objective: regulate the opening of the throttle valve while being robust to the inherent noise and the stochasticity of the dynamics.

A. Discrete time PI Controller

The PI controller is commonly used in control systems design due to its simplicity and effectiveness in a wide range of systems. It can be expressed in discrete time, using the notations introduced in the previous section, as:

$$\pi_{\text{PI}}(x_t) = u_{t-1} - r_0\alpha_t - r_1\alpha_{t-1} + (r_0 + r_1)\alpha_{\text{ref}_t}. \quad (1)$$

In order to properly tune the gains for this system, we followed the steps proposed by the previous work [6], detailed below. This work found that the following auto-regressive model

$$\alpha_t = a\alpha_{t-1} + b_1 u_{t-1} + b_2 u_{t-2} \quad (2)$$

represents the dynamics of any valve from the considered commercial reference sufficiently well for control purposes. The model parameters a and b are tuned to fit the first order model using data describing the response of the valve from a sufficiently rich input signal (from the persistency of excitation point of view), generated by a 1022-long Pseudo Random Binary Sequence (PRBS) centered at 16%. Then, r_0 and r_1 are chosen such that the damping ratio ζ is set to 1 and the rising time t_R is 0.8 seconds.

Manuscript 112 submitted to 2024 8th IEEE Conference on Control Technology and Applications (CCTA). Received February 16, 2024.

For the valves considered in this paper, this experiment leads to the values of ($a = 0.78,\ b_1 = -0.18,\ b_2 = -0.23,\ r_0 = -2.28,\ r_1 = 1.83$) for Valve 1, ($a = 0.74,\ b_1 = -0.25,\ b_2 = -0.41,\ r_0 = -1.31,\ r_1 = 1.01$) for Valve 2, and ($a = 0.83,\ b_1 = -0.11,\ b_2 = -0.23,\ r_0 = -2.33,\ r_1 = 1.96$) for Valve 3.

Note that the purpose here is not to design the best possible PI controller but to show the advantages of combining both control and machine learning techniques, in comparison with using control or reinforcement learning techniques separately.

B. Approximate Policy Iteration

To have a reliable effective controller for the valve, we propose to use a traditional RL agent that uses the *Approximate Policy Iteration* scheme [31].

Most RL algorithms rely on the Bellman operator, defined as $\mathscr{B}^{\pi}[Q](x,u) = c(x,u) + \gamma \mathbb{E}_{x' \sim P(\cdot|x,u), u' \sim \pi(\cdot|x')}[Q(x',u')]$. This operator is a γ-contraction, so iteratively applying it to any function Q converges to its unique fixed point Q^{π}. However, since the transition probabilities P are unknown, the empirical Bellman operator $\hat{\mathscr{B}}$ that uses interactions with the environment to estimate this expectation is used instead.

In the Approximate Policy Iteration framework, both the policy π_{RL} and the Q functions are parametrized with the respective weights $\theta \in \Theta$ and $\omega \in \Omega$, and are thus denoted as π_{RL}^{θ} and Q^{ω}. Both estimates will work together until the policy converges to a near-optimal one. More precisely, at each epoch k, the agent collects data with the current policy $\pi_{\text{RL}}^{\theta_k}$, evaluates its associated Q-function via Approximate Policy Evaluation (APE) and improves its policy through Approximate Policy Improvement (API). Given a dataset $\mathscr{D} = \{(x_i, u_i, c_i, x_{i+1})_{i=1}^{N}\}$, $\hat{\mathbb{E}}$ the empirical expectation of the observation-control pair (x,u) induced by the data set $\mathscr{D}$, the scheme is formalized as:

$$Q^{\omega_{k+1}} \leftarrow \arg\min_{\omega \in \Omega} \hat{\mathbb{E}}\left[\left(Q^{\omega} - \hat{\mathscr{B}}^{\pi_{\text{RL}}^{\theta_k}}[Q^{\omega_k}]\right)^2\right], \qquad \text{(APE)}$$

$$\pi_{\text{RL}}^{\theta_{k+1}} \leftarrow \arg\min_{\theta \in \Theta} \hat{\mathbb{E}}_{x \sim \mathscr{D}, u \sim \pi_{\text{RL}}^{\theta}}[Q^{\omega_{k+1}}(x,u)]. \qquad \text{(API)}$$

This cycle of evaluation and improvement is commonly solved approximately with gradient descent and continues until the policy converges to an optimal one. This framework, not requiring prior knowledge from the environment beyond interaction data, can be applied as such to any system as long as it possesses the Markov property. Nonetheless, the absence of initial information makes it difficult to collect meaningful data at the early stages of learning. This leads to poor estimates of the Q function and the policy. As a result, agents necessitate extensive interactions to build accurate estimates and derive a near-optimal policy.

TD3 or Twin Delayed Deep Deterministic Policy Gradient [32], is a leading algorithm in this framework. Building upon the foundation laid by DDPG [10], it estimates the Q-function using two neural networks and introduces delayed policy updates to enhance the stability of learning.

IV. Combining both methods

On the one hand, the PI controller is easy to build and requires a minimal amount of data, but suffers from sub-optimality on non-linear systems due to its linear dependencies. On the other hand, RL agents often achieve near-optimality on complex systems but require a significant amount of data. Two primary reasons for this requirement are the poor initialization of the policy and the exhaustive search for the control u across the whole control space $\mathscr{U}$ at each time step.

To combine the advantages of both methods, we draw inspiration from the recent framework of *Reinforcement Learning with Guides* [25] and adapt one of the most recent and efficient techniques to our context: Perturbation Action Guided (PAG) [27]. For clarity purposes, we refer to the adaptation of PAG to our setting by PI-RL.

In this approach, to circumvent searching the entire control space $\mathscr{U}$ for the optimal control, the PI controller is incorporated to guide the training process. This not only reduces the search space but also steers the RL agent towards favorable parts of the environment, directly improving exploration and the quality of the gathered data. To do so, the PI-RL policy is centered around the PI controller and learns a perturbation ξ_{RL}^{ϕ}, with $\phi \in \Xi$ to guide it towards better actions. Formally, the new policy is written:

$$\pi_{\text{PI-RL}}^{\phi}(\cdot|x) = \pi_{\text{PI}}(x) + \xi_{\text{RL}}^{\phi}(\cdot|x). \qquad (3)$$

Unlike the traditional RL policy π_{RL}^{θ} that provides a control input u, the perturbation ξ_{RL}^{ϕ} aims solely to enhance the PI controller's performance. Thus, rather than defining the perturbation ξ_{RL}^{ϕ} over the entire control space $\mathscr{U}$, it is constrained to a smaller subset $S(\mathscr{U})$. This confines the search for optimal control to a vicinity around the PI controller, where the optimal control is more likely to be found, thereby improving the algorithm's sample efficiency. In practice, to reduce the search space that is originally $[0, 0.6]$ or $[0, 0.8]$ and keeping the differentiability property of the policy, we introduce a multiplicative factor $\eta \in [0,1]$, which scales the control input by a factor η.

The perturbation is learned similarly to a control input in traditional RL, that is by minimizing its associated Q^{ω} function. We stress here that the APE step has to be modified to take into account the policy change. Note that the expectation in the Bellman operator is with respect to $\pi_{\text{PI-RL}}^{\phi_k}$, not π_{θ_k}. This avoids the overestimating problem that may be encountered when the Q-function estimates observation-control pairs that are not described by the data set $\mathscr{D}$ [32], [27]. The process becomes the PI-Approximate Policy Iteration, where the APE-API steps are modified to be PI-APE and PI-API, defined as:

$$Q^{\omega_{k+1}} \leftarrow \arg\min_{\omega \in \Omega} \hat{\mathbb{E}}\left[\left(Q^{\omega} - \hat{\mathscr{B}}^{\pi_{\text{PI-RL}}^{\phi_k}}[Q^{\omega_k}]\right)^2\right], \qquad \text{(PI-APE)}$$

$$\xi_{\text{RL}}^{\phi_{k+1}} \leftarrow \arg\min_{\phi \in \Xi} \hat{\mathbb{E}}_{x \sim \mathscr{D}, u \sim \pi_{\text{PI-RL}}^{\phi}}[Q^{\omega_{k+1}}(x,u)]. \qquad \text{(PI-API)}$$

Manuscript 112 submitted to 2024 8th IEEE Conference on Control Technology and Applications (CCTA). Received February 16, 2024.

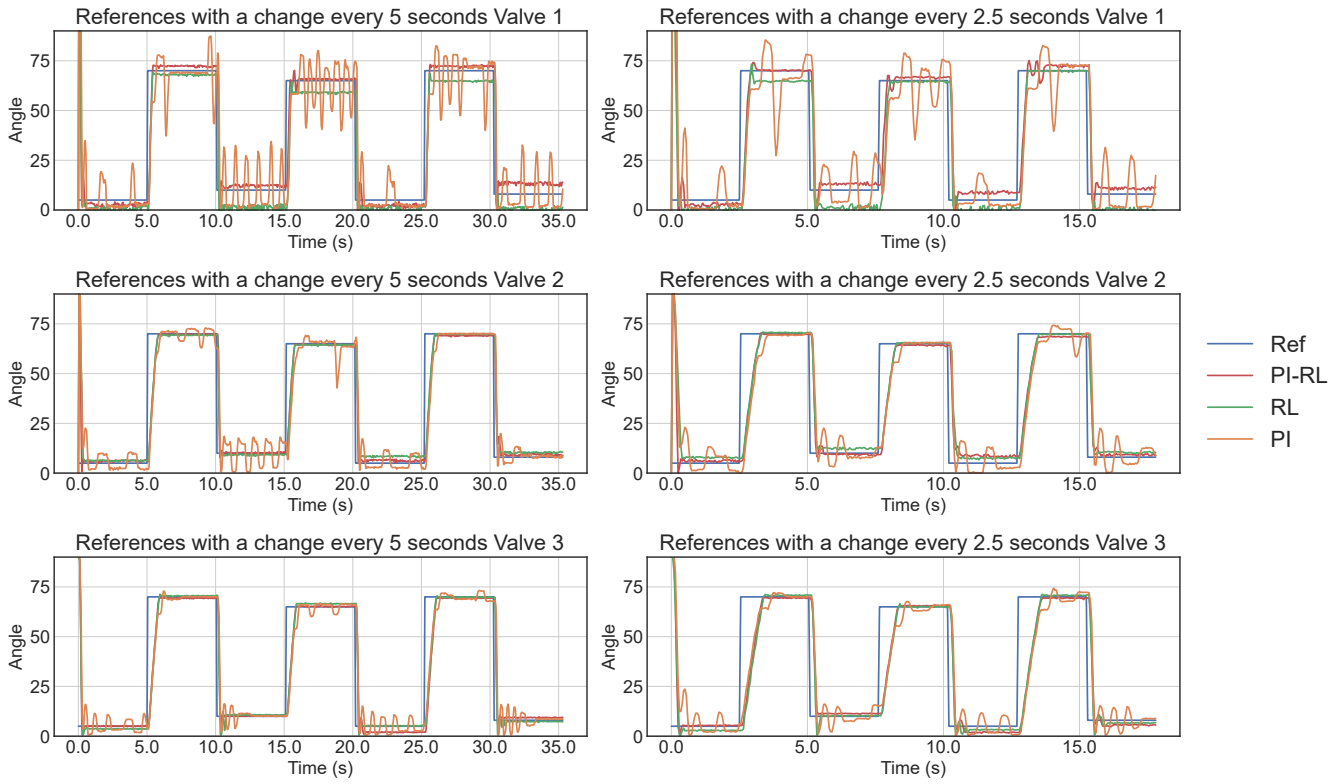

Fig. 3: Comparison of the different agents under different scenarios when all agents have access to the real outputs. Both figures display the evolution of the angles over time with a reference change every X seconds, set to 5 seconds for the left figures and to 2.5 seconds for the right figures.

Similarly to the traditional RL agent, both the Q-function and the perturbation ξ_{RL}^{ϕ} are parametrized with neural networks that are learned with gradient descent. The scaling term η is set to 0.5 for both valves reducing by two the search space. The subspace of $\mathscr{U}$ is hence set to $S(\mathscr{U}) = [0, 0.4]$ for the first two valves and to $S(\mathscr{U}) = [0, 0.3]$ for the last valve.

The whole algorithm can be found in Pseudo-Code 1.

Algorithm 1 PI-RL

Select damping ratio ζ, rising time t_R and Φ
Create PI controller
Initialize Q^{ω_0} and $\xi_{\text{RL}}^{\phi_0}$
for $k \in (0, \dots, K)$ **do**
  Gather data with $\pi_{\text{PI-RL}}^{\phi_k}$ from Eq. (3)
  Add data to the replay buffer $\mathscr{D}$
  Sample a batch from $\mathscr{D}$
  Update $Q^{\omega_{k+1}}$ with gradient descent on Eq. (PI-APE)
  Update ξ_{ϕ}^{k+1} with gradient descent on Eq. (PI-API)
end for

V. Comparison of the different controllers on the experimental test benches

We conduct a series of experiments on various setups. We first perform a nominal comparison between the controllers, where we test the controllers' performance on a simple task: tracking a reference signal with sufficient time to adjust. We then test them on more challenging tasks, including adapting to quick reference changes and robustness with respect to noise in the control input or in the output observation.

To assess the effectiveness of the different controllers, we use the Mean Square Error (MSE) between the current angle and the objective. It is formally defined with sequences $\alpha = (\alpha_0, \alpha_1, \dots)$ and $\alpha_{\text{ref}} = (\alpha_{\text{ref}_0}, \alpha_{\text{ref}_1}, \dots)$ as:

$$MSE(\alpha, \alpha_{\text{ref}}) = \|\alpha - \alpha_{\text{ref}}\|_2 . \tag{4}$$

This metric seems appropriate as it encompasses all of the costs during one scenario.

In all our experiments, both RL-based algorithms use TD3. The Q-functions and the policies π_{RL}^{θ} and ξ_{RL}^{θ} are parametrized with neural networks with 2 hidden layers of size 64 and use the ReLU activation function.

A. Nominal comparison and reaction to quick reference changes

We compare the performance of all agents for tracking a reference signal through a series of experiments. First, we perform a comparison when the angles change every 5 seconds, and one with quicker reference changes every 2.5 seconds. All results are summarized in Figure 3.

a) PI controller: In all scenarios, we observe that the PI controller, while efficient, encounters difficulties in accurately tracking the reference signal. The controller requires

Manuscript 112 submitted to 2024 8th IEEE Conference on Control Technology and Applications (CCTA). Received February 16, 2024.

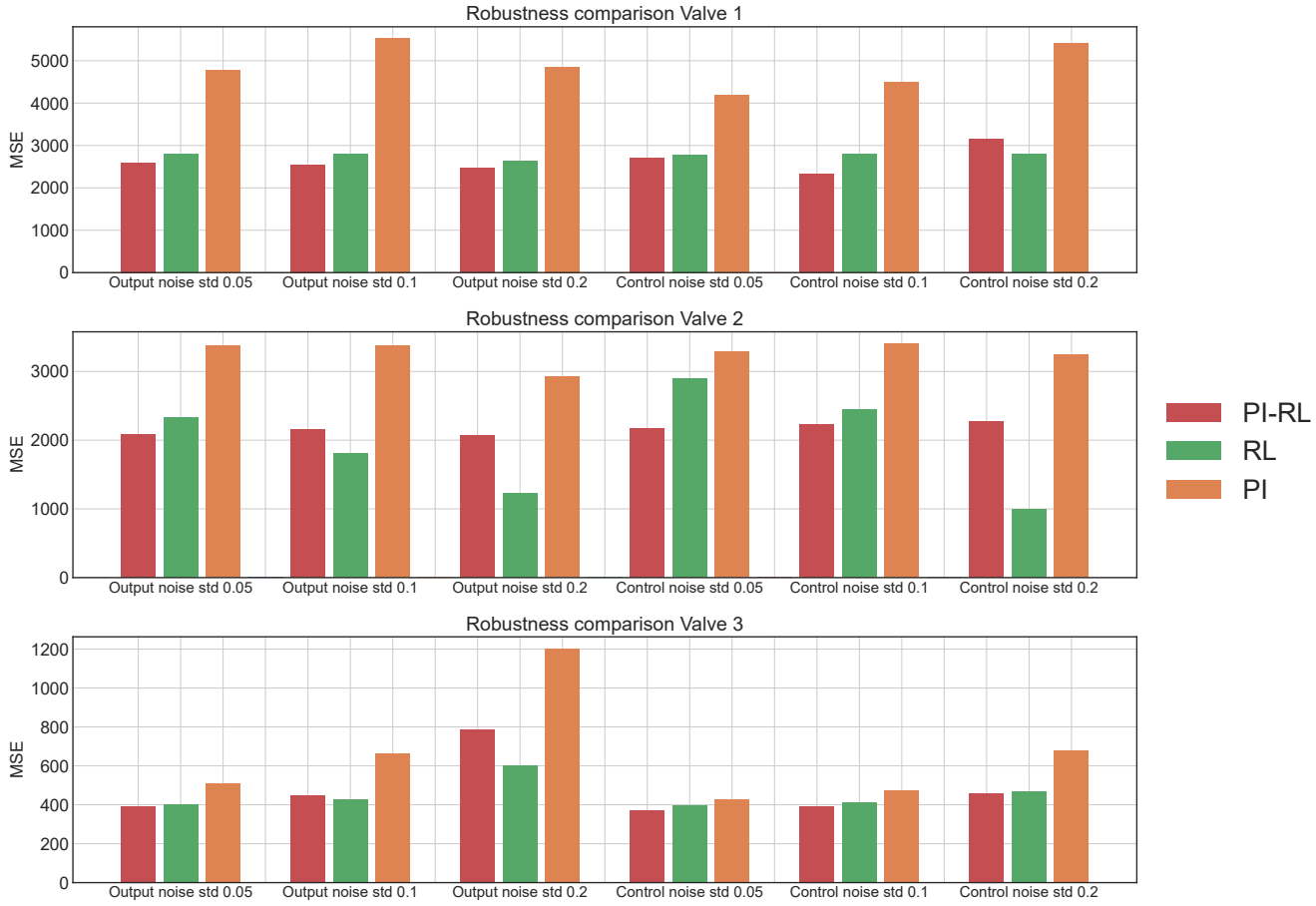

Fig. 4: Comparison of the different agents under noisy outputs or noisy controls with different standard deviations. Each bar represents the Mean-Squared-Error (MSE) between the reference and the angle of the valve.

time to accumulate errors before adjusting to the reference. Due to the system's complexity and the dry friction effect, it may cause the valve's plate to overshoot the target. The PI controller would then need additional time to adjust, and potentially select overly aggressive inputs once more. This results in oscillations around the reference angle, particularly evident in the first valve that seems the most complicated to control. The other valves also present such oscillations, especially for low-angle references. This reveals the need for more sophisticated controllers and not solely relying on a linear policy.

b) RL-based agents: On the other hand, both RL-based agents demonstrate remarkable success in precisely tracking the reference signal in all scenarios, including the most challenging ones. For example, they can track all extreme reference values with a change every 2.5 seconds, which is twice as fast as the rate they were trained for. This level of performance is also observed for low-angle references that proved to be challenging for the PI controller. The controls provided by the RL-based agents are stable in all settings, with no instances of unstable behavior observed. In some cases, a slight tracking bias is observed where the agents reach a point close to but not precisely at the reference. This can be attributed to the RL agents' goal of minimizing the Q function, which is an expectation over the system's stochastic dynamics. To address this, a more appropriate RL formulation may be needed, which considers higher-order moments of the cumulative future cost distribution or steady-state errors in a refined way. We leave this for future work. Despite these minor biases, both RL-based agents demonstrate a near-optimal performance in all tasks for all valves.

B. Robustness to external noise

We also conduct experiments to assess the robustness of the controllers in the presence of noise, both in the output and in the control. To this end, we test the controllers under varying noise levels and analyze their performance.

To better visualize the effects of noise on the controllers' performance, we plot in Figure 4 the Mean Squared Error (MSE) rather than the signals and the reference as in the previous section.

In comparison to the PI controller, both RL-based agents demonstrate superior performance over the PI controller in all scenarios. Once again, it is evident that the first valve is the most complicated to control as attested by the PI performances. It is also essential to note that, under specific conditions of noise, e.g. in Valve 2 with a control noise with

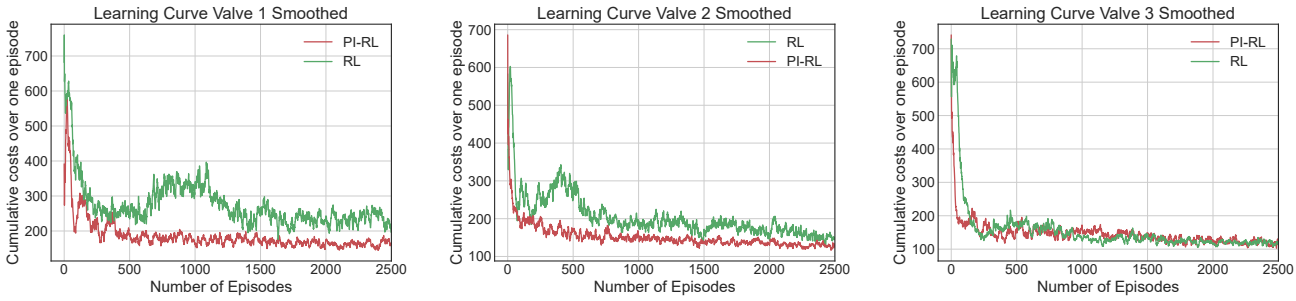

Fig. 5: Learning curves of the different RL-based agents for the 3 valves. We train both agents during 2500 episodes, representing 3 hours with our computer without the need to use a GPU.

a standard deviation of 0.2 and in Valve 1 with an output noise of standard deviation 0.2, the PI-RL agent exhibits a higher loss than the traditional RL agent. This result may be attributed to the fact that the PI-RL agent is centered around the PI controller which is inadequate in these scenarios. Consequently, the reduced search space is not sufficient to yield an optimal controller, and a more sophisticated approach may be needed to address the challenges posed by these specific noise conditions. Nevertheless, we note that the perturbation ξ_{RL}^{ϕ} can improve the PI controller up to a certain point.

In summary, our robustness analysis reveals that both RL-based agents perform well in noisy environments while the PI controller's performance declines more rapidly. This declination may affect the PI-RL agent as the search space is too narrow to recover a near-optimal policy. These results suggest that the PI-RL agent offers promising results in low-to-moderate noise environments, where the PI controller can still provide a relevant control signal. However, in more challenging noise conditions where the PI controller's performance is limited, traditional RL may be better suited to provide a more robust and effective controller.

C. Sample efficiency

As previously shown, both the PI-RL and classical RL agents achieve similar performances in controlling the throttle valve. One important advantage of the combined PI and RL approach over the classical RL approach is that it requires fewer data to learn even though the PI controller may not be the optimal one. This can be seen in Figure 5, where we plot the cumulative costs over the training process for both controllers for the 3 different valves.

We observe that the PI-RL approach achieves a low cumulative cost faster than the RL approach, indicating that it was able to learn the control task more efficiently. This is because PI-RL leverages the prior knowledge of the PI controller to reduce the search space and guide the RL exploration, which helps to reduce the exploration time and improve the sample efficiency. After a sufficient amount of data, the traditional agent finally catches the performance of the combined agent. This is less obvious in Valve 3, which seems to be the simplest valve to control as observed in Figure 3 and Figure 4. Indeed, the RL agent converges almost as fast as the PI-RL agent to the same near-optimal policy.

The combination of PI and RL, therefore, benefits from the advantages of both worlds. The PI controller provides a good indication of where the optimal control is located, while RL can refine it toward better controls.

VI. Conclusion

In this work, we show that the combination of reinforcement learning and optimal control can be a powerful approach to designing near-optimal controllers for throttle valve systems. By adapting the recent area of Reinforcement Learning with Guides to the throttle valve system, we can build a near-optimal controller that reduces the data requirement of traditional RL agents and overcomes the limitations of the PI controller. Our approach is validated on three different valves with slightly different dynamics, all demonstrating the advantages and drawbacks of all controllers.

This provides evidence that the integration of OC techniques with RL can lead to significant improvements in data efficiency, making it a promising avenue for future research in other complex systems.

References

[1] C. Canudas de Wit, I. Kolmanovsky, and J. Sun, "Adaptive pulse control of electronic throttle," in *Proceedings of the 2001 American Control Conference*, vol. 4. IEEE, 2001, pp. 2872–2877.

[2] U. Ozguner, S. Hong, and Y. Pan, "Discrete-time sliding mode control of electronic throttle valve," in *Proceedings of the 40th IEEE Conference on Decision and Control*, vol. 2. IEEE, 2001, pp. 1819–1824.

[3] J. Deur, D. Pavkovic, N. Peric, M. Jansz, and D. Hrovat, "An electronic throttle control strategy including compensation of friction and limp-home effects," *IEEE Transactions on Industry Applications*, vol. 40, no. 3, pp. 821–834, 2004.

[4] D. Pavković, J. Deur, M. Jansz, and N. Perić, "Adaptive control of automotive electronic throttle," *Control Engineering Practice*, vol. 14, no. 2, 2006.

[5] X. Jiao, J. Zhang, and T. Shen, "An adaptive servo control strategy for automotive electronic throttle and experimental validation," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 11, pp. 6275–6284, 2014.

[6] E. Witrant, I. D. Landau, and M.-P. Vaillant, "A data-driven control methodology applied to throttle valves," *Control Engineering Practice*, vol. 139, p. 105634, 2023.

[7] S. Zhang, J. J. Yang, and G. G. Zhu, "LPV modeling and mixed constrained H_2/H_∞ control of an electronic throttle," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 5, pp. 2120–2132, 2014.

Manuscript 112 submitted to 2024 8th IEEE Conference on Control Technology and Applications (CCTA). Received February 16, 2024.

[8] M. Barić, I. Petrović, and N. Perić, "Neural network-based sliding mode control of electronic throttle," *Engineering Applications of Artificial Intelligence*, vol. 18, no. 8, pp. 951–961, 2005.

[9] R. Siraskar, "Reinforcement learning for control of valves," *Machine Learning with Applications*, vol. 4, p. 100030, 2021.

[10] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *Proceedings of the 4th International Conference on Learning Representations (ICLR)*, 2016.

[11] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

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

[13] M. G. Bellemare, S. Candido, P. S. Castro, J. Gong, M. C. Machado, S. Moitra, S. S. Ponda, and Z. Wang, "Autonomous navigation of stratospheric balloons using reinforcement learning," *Nature*, vol. 588, no. 7836, pp. 77–82, 2020.

[14] J. Degrave, F. Felici, J. Buchli, M. Neunert, B. Tracey, F. Carpanese, T. Ewalds, R. Hafner, A. Abdolmaleki, D. de Las Casas *et al.*, "Magnetic control of tokamak plasmas through deep reinforcement learning," *Nature*, vol. 602, no. 7897, pp. 414–419, 2022.

[15] W. Koch, R. Mancuso, R. West, and A. Bestavros, "Reinforcement learning for uav attitude control," *ACM Transactions on Cyber-Physical Systems*, vol. 3, no. 2, pp. 1–21, 2019.

[16] S. Tunyasuvunakool, A. Muldal, Y. Doron, S. Liu, S. Bohez, J. Merel, T. Erez, T. Lillicrap, N. Heess, and Y. Tassa, "dm_control: Software and tasks for continuous control," *Software Impacts*, vol. 6, p. 100022, 2020.

[17] G. Dulac-Arnold, N. Levine, D. J. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester, "Challenges of real-world reinforcement learning: definitions, benchmarks and analysis," *Mach. Learn.*, vol. 110, no. 9, pp. 2419–2468, 2021.

[18] B. Recht, "A tour of reinforcement learning: The view from continuous control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, pp. 253–279, 2019.

[19] S. Gillen, M. Molnar, and K. Byl, "Combining deep reinforcement learning and local control for the acrobot swing-up and balance task," in *Proceedings of the 2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 4129–4134.

[20] S. Zoboli, V. Andrieu, D. Astolfi, G. Casadei, J. S. Dibangoye, and M. Nadri, "Reinforcement learning policies with local lqr guarantees for nonlinear discrete-time systems," in *Proceedings of the 2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 2258–2263.

[21] T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine, "Residual reinforcement learning for robot control," in *Proceedings of the 2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 6023–6029.

[22] R. Zhang, P. Mattsson, and T. Wigren, "Aiding reinforcement learning for set point control," *22nd World Congress of the International Federation of Automatic Control (IFAC)*, 2023.

[23] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *Proceedings of the International Conference on Machine Learning (ICML)*. PMLR, 2019, pp. 2052–2062.

[24] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 1179–1191, 2020.

[25] M. Zimmer, P. Viappiani, and P. Weng, "Teacher-student framework: a reinforcement learning approach," in *AAMAS Workshop Autonomous Robots and Multirobot Systems*, 2014.

[26] R. Agarwal, M. Schwarzer, P. S. Castro, A. C. Courville, and M. Bellemare, "Reincarnating reinforcement learning: Reusing prior computation to accelerate progress," *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, pp. 28 955–28 971, 2022.

[27] P. Daoudi, B. Robu, C. Prieur, L. Dos Santos, and M. Barlier, "Enhancing reinforcement learning agents with local guides," in *Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AMMAS)*, 2023.

[28] M. Sedighizadeh and A. Rezazadeh, "Adaptive pid controller based on reinforcement learning for wind turbine control," in *Proceedings of world academy of science, engineering and technology*, vol. 27. Citeseer, 2008, pp. 257–262.

[29] I. Carlucho, M. De Paula, and G. G. Acosta, "An adaptive deep reinforcement learning approach for mimo pid control of mobile robots," *ISA transactions*, vol. 102, pp. 280–294, 2020.

[30] N. P. Lawrence, G. E. Stewart, P. D. Loewen, M. G. Forbes, J. U. Backstrom, and R. B. Gopaluni, "Optimal pid and antiwindup control design as a reinforcement learning problem," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 236–241, 2020.

[31] D. P. Bertsekas, "Approximate policy iteration: A survey and some new methods," *Journal of Control Theory and Applications*, vol. 9, no. 3, pp. 310–335, 2011.

[32] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proceedings of the International Conference on Machine Learning*. PMLR, 2018, pp. 1587–1596.