Preprints.org

Article

Not peer-reviewed version

Reinforcement Learning for Plasma Control: A Proof of Concept for NTM Suppression

Luca Bonalumi *, Edoardo Alessi, Enzo Lazzaro, Silvana Nowak, Carlo Sozzi

Posted Date: 23 April 2026

doi: 10.20944/preprints202604.1616.v1

Keywords: nuclear fusion; artificial intelligence; MHD control; magnetic island; reinforcement learning

Preprints.org is a free multidisciplinary platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This open access article is published under a Creative Commons CC BY 4.0 license, which permit the free download, distribution, and reuse, provided that the author and preprint are cited in any reuse.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

Disclaimer/Publisher's Note: The statements, opinions, and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions, or products referred to in the content.

Article

Reinforcement Learning for Plasma Control: A Proof of Concept for NTM Suppression

Luca Bonalumi *, Edoardo Alessi, Enzo Lazzaro, Silvana Nowak and Carlo Sozzi

Institute for Science and Technology of Plasma (ISTP-CNR), Italy
* Correspondence: luca.bonalumi@istp.cnr.it

Abstract

Neoclassical tearing modes (NTMs) are magnetohydrodynamic instabilities that generate magnetic islands in tokamak plasmas, degrading confinement and potentially limiting high-performance operation. Their stabilization typically requires precise alignment and appropriate injection of electron cyclotron (EC) power beams, making real-time control a challenging task. In this work, we present a proof-of-principle study aimed at investigating the potential role of neural networks in the control of plasma instabilities. The objective is not to develop a device-specific controller, but rather to explore, within a synthetic environment, how a learning-based agent can autonomously discover effective stabilization strategies. To this end, a neural network controller is trained using reinforcement learning techniques, resulting in an intelligent and agnostic control system. The controller is defined as intelligent in the sense that it learns the optimal strategy directly from interaction with the environment, without being explicitly programmed or guided by a predefined control law. It is agnostic because it does not rely on equilibrium reconstruction or explicit knowledge of the deposition location relative to the island. Instead, it operates solely on feedback derived from a representation of the magnetic island width, using this information to adapt its actions. Two control tasks are considered: pure angular alignment and combined angular alignment with power control. This exploratory study establishes a framework for assessing the potential advantages of data-driven approaches in magnetic island control and provides a basis for future investigations aimed at improving alignment and suppression strategies in fusion plasmas.

Keywords: nuclear fusion; artificial intelligence; MHD control; magnetic island; reinforcement learning

1. Introduction

Neoclassical tearing modes (NTMs) are magnetohydrodynamic instabilities that arise on rational magnetic surfaces in tokamak plasmas. When destabilized, NTMs lead to the formation of magnetic islands that perturb the magnetic topology and degrade energy confinement. If left uncontrolled, their growth can significantly reduce plasma performance and, in severe cases, trigger a complete loss of equilibrium (disruption). The relevance of NTMs for tokamak operation is well documented. A survey of unintentional disruptions at the Joint European Torus (JET) performed by de Vries et al. [1] reported that approximately 16% of the analyzed disruptions were associated with the presence of NTMs, making them the single most frequent identified cause among the considered cases. Real time detection and control of NTMs are therefore of fundamental importance for ensuring stable plasma operation. This is particularly critical in high-performance regimes, characterized by elevated plasma pressure (high β), where the operational margins are reduced and the impact of magnetic islands is more severe. The issue becomes even more pressing for next-generation devices such as ITER, where the machine is designed to tolerate only a limited number of disruptions over its operational lifetime [2]. The amplitude of an NTM can be controlled by driving a localized current at the O-point of the magnetic island. This externally driven current compensates for the deficit in bootstrap current, which constitutes the primary drive of the island during its nonlinear evolution phase [3]. By restoring the

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

current profile locally, the destabilizing mechanism can be mitigated and the island growth arrested. In practice, this control strategy is implemented through the injection of electron cyclotron (EC) power beams, which enable localized current drive and heating. In future machines, NTM suppression will have more demanding requirements on ECH/CD alignment. Nowadays, real-time control system for NTM suppression typically usually on a sequence of coordinated operations: detection of the magnetic island, identification of its radial position through equilibrium reconstruction, and subsequent alignment of the EC launcher so that the injected power is deposited at the island O-point [4,5]. Achieving accurate alignment requires establishing the relationship between the beam deposition location and the poloidal injection angle α. The overall control architecture therefore depends on the consistent interaction of diagnostic measurements, equilibrium reconstruction, wave propagation modeling, and actuator control. In this work, we propose to investigate a neural-network-based controller trained through reinforcement learning for the stabilization of magnetic islands. Application of Reinforcement learning to NTM control has been performed before [6], but focusing on the β control to prevent the triggering conditions for a NTM.

Here we want to train a network to control the NTM by acting directly on the launcher and the power level injected into the plasma. The objective is not to provide a device-specific control solution, but rather to conduct a proof-of-concept study within a synthetic environment, with the aim of assessing both the potential and the limitations of this approach. For this reason we exploited the Proximal Policy Optimization (PPO) [7] algorithm, a policy-based, model-free reinforcement learning method that directly optimizes a stochastic policy through interaction with the environment. By avoiding the construction of an explicit model of the system dynamics, PPO reduces the dependence on a specific analytical representation of the plasma. The controller is trained to act on two control variables: the injected power and the poloidal injection angle. As feedback, the agent receives an estimate of the magnetic island amplitude. This can be obtained from the magnetic diagnostics as a consequence of the perturbation of the island on the radial magnetic field [8]. In this framework, the behavior of the controller is not dictated by an explicit control model. Rather, it is shaped through the reward function provided by the environment. By modifying the reward structure, different control objectives and constraints can be enforced, allowing the learning process to autonomously converge toward optimized stabilization strategies. In Section 2 the reinforcement learning framework is introduced. Description of the problem and the training process is performed in Section 3 and Section 4. In Section 5 the agent is optimized in order to minimize the power used to stabilized the magnetic island.

2. Reinforcement Learning Framework

The system under consideration, namely the magnetic island evolving under the plasma conditions, is a dynamical system whose behavior can be influenced through external actuators. The role of the controller is to manipulate the suppression system in order to drive the island toward a desired state, typically complete stabilization. Such a setting can be formally described within the framework of a Markov Decision Process (MDP). An MDP provides a mathematical representation of sequential decision-making in a dynamical environment, where a decision maker (agent) interacts with the system over time [9]. In this formalism, at each discrete time step t the decision maker observes the current state s_t of the system and selects an action a_t. The action influences the subsequent evolution of the dynamical system, leading to a new state s_{t+1}. At each time step t, the agent selects an action a_t according to a policy $\pi(a_t|s_t) = P(a_t \mid s_t)$ a function that specifies the probability of executing a certain action given that the system is in state s_t at time t. The objective of the decision maker is to select actions so as to optimize the system performance $G_\gamma(t)$ over future states. $\gamma \in (0,1]$ is a free-parameter, named the discount factor, that quantifies how the effect of an action taken at time t propagates into future states by weighting the contribution of subsequent rewards. For $\gamma = 0$, the system performance is determined solely by the instantaneous reward. Conversely, when $\gamma = 1$, all rewards throughout the process are weighted equally. The policy $\pi^*(a|s)$ that, at each time step

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

t, selects actions so as to maximize the expected return $G_\gamma(t)$ is referred to as the *optimal policy*. In the present context, the magnetic island dynamic and the actuators (power source, launching angle) status represent the environment, while the controller acts as the decision maker. The control problem is therefore recast as a sequential decision process in which the policy determines how the actuator should be operated based on the current state. Within this framework, the agent is implemented as a neural network which, by virtue of its well-known universal approximation capability [10], is able to approximate the optimal policy $\pi^*(a|s)$ through a dedicated training procedure known as reinforcement learning[11,12].

2.1. Reinforcement Learning

In reinforcement learning, the policy $\pi_\theta(a_t|s_t)$ is represented by a neural network, where θ denotes the set of trainable parameters of the model. The network maps the current state of the system to a probability distribution over the available actions $\pi_\theta(a_t|s_t) = P_\theta(a_t \mid s_t)$. During training, the parameters θ are iteratively updated so as to optimize (minimize) a loss function $L(\pi_\theta(a_t|s_t))$. The update is performed by moving in the direction opposite to the gradient $\nabla_\theta L(\pi_\theta(a_t|s_t))$, according to standard gradient-based optimization procedures. The loss function is constructed such that its minimization corresponds to the maximization of the expected return $G_\gamma(t)$. The training process consists of a sequence of parameter updates, here referred to as iterations. At the i-th iteration, the policy is denoted as $\pi_\theta^{(i)}(a_t|s_t)$, reflecting the current set of parameters $\theta^{(i)}$.

As training progresses, successive updates modify the policy in order to increase the expected return. Under suitable conditions and for a sufficiently large number of iterations, the sequence of policies may converge toward an optimal policy: $\pi_\theta^{(i)}(a_t|s_t) \to \pi^*(a_t|s_t) \quad \text{as} \quad i \to \infty$. In this way, adjusting the network parameters through gradient descent leads the policy to progressively improve its long-term performance. This approach belongs to the class of policy-based methods, in which the policy itself is directly parameterized and optimized. In order to enlarge as much as possible the portion of the state–action space explored by the agent, each training iteration i includes a rollout phase during which multiple episodes are executed using the current policy $\pi_\theta^{(i)}$. An episode corresponds to one complete rollout of the agent–environment interaction, from initialization to termination. During each episode, trajectories are collected in the form of sequences of state–action transitions. More precisely, for $t = 1, \ldots, T_{\text{episode}}$, a trajectory is a set of $\left\{ s_t,\ a_t,\ \pi_\theta^{(i)}(a_t|s_t),\ G_\gamma(t) \right\}$, where T_{episode} denotes the episode length. These trajectories of an iteration constitute the dataset used to estimate the loss function $L(\pi_\theta^{(i)})$ and compute the gradient update of the policy parameters. The trajectories collected during the rollout phase are generated through the interaction between the agent and the environment. At each time step t within an episode, the agent selects an action a_t according to the current policy, the environment evolves to a new state, and a reward r_t is returned according to a predefined reward function. The reward r_t provides an instantaneous evaluation of the system performance at time t, based on criteria established in the formulation of the control objective. In this sense, the reward function encodes the performance metrics that define what constitutes desirable behavior for the controlled system. For each trajectory, the return $G_\gamma(t)$ is then computed as an aggregation of the reward r_t at each time step. In particular, it is defined as the discounted sum of future rewards:

$$G_\gamma(t) = \sum_{k=0}^{T_{\text{episode}}-t} \gamma^k r_{t+k}. \tag{1}$$

As summarized in Figure 1, reinforcement learning provides a framework for training an agent to follow an optimal control strategy for a given task, by iteratively adjusting its policy based on the rewards returned by the environment at each time step.

In the present work, the environment is represented by the numerical model that evolves the magnetic island dynamics, explicitly accounting for the influence of the injected power on its temporal evolution.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

The agent is implemented as a neural network composed of a Long Short-Term Memory (LSTM) layer, followed by fully connected layers.

The reward function is designed to promote island suppression: the agent receives positive reinforcement when the island width decreases and negative feedback otherwise. In addition, the reward includes penalty terms enforcing physical constraints, in particular limiting the admissible angular excursion of the launcher. The policy parameters are optimized using the Proximal Policy Optimization (PPO) algorithm.

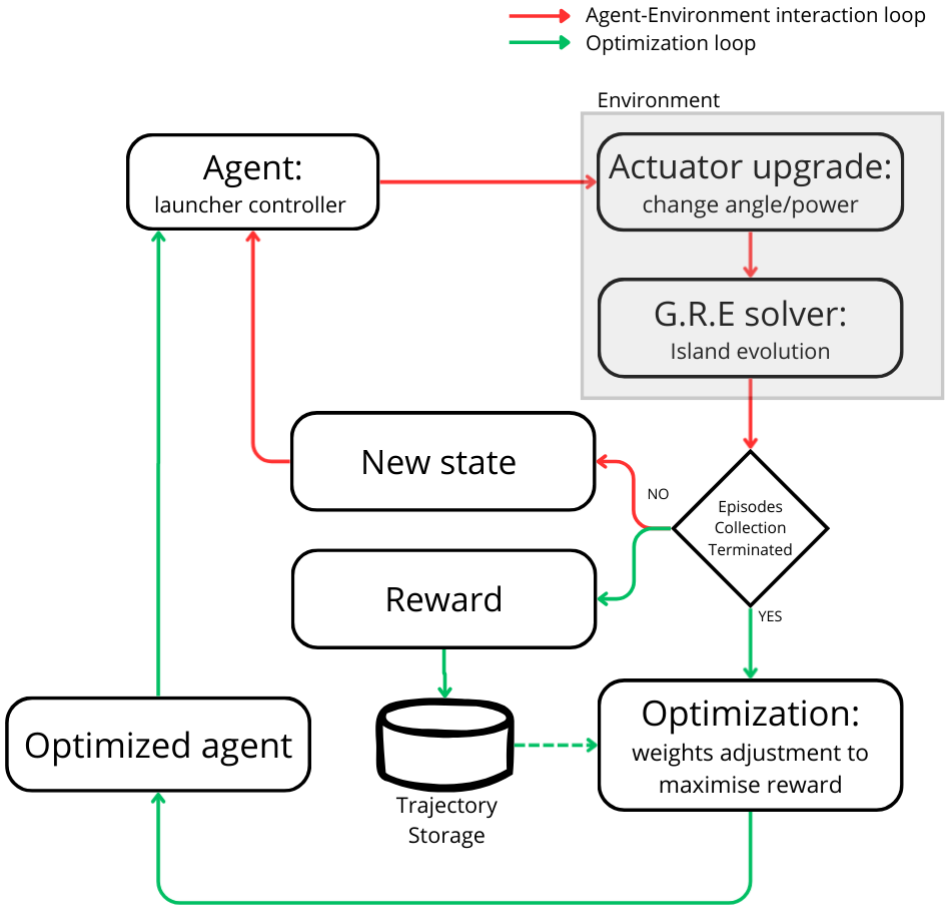

Figure 1. Illustration of the interaction loop between agent and environment (red) and the optimization loop that trains the agent (green)

2.2. Environment

The environment consists of a synthetic model that simulates the temporal evolution of the magnetic island. The dynamics are discretized in time steps of 10 ms. At each time step, the agent selects an action, corresponding to a modification of the injected power or the poloidal angle. This action is applied to the simulation environment, and its effect is incorporated into the island evolution over the subsequent 10 ms interval. The updated state of the island (width and growth rate) is then returned to the agent, closing the interaction loop. An simulation (episode) terminates either when a maximum of 300 time steps is reached, or when the island width decreases below a predefined threshold of 1 mm, which is taken as a condition for complete suppression. The evolution is based on the DTT [13] full power scenario [14], in which there will be 8MW of power coming from the upper launcher dedicated for the control and suppression of NTMs. The EC power deposition from the upper launcher used for NTM suppression is computed using the GRAY ray-tracing code [15]. A preliminary scan over the poloidal injection angle is performed to compute the EC power deposition profile corresponding to each angle. The resulting deposition profiles are subsequently retrieved during the simulation through a lookup table. The considered angles span the interval between 7° and 26°.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

Within this range (assumed to represent the admissible angular excursion of the launcher) the network is allowed to adjust the injection angle changing it by a quantity $\delta\alpha = 1^\circ$ at each step. The island evolution is computed under the assumption of a fixed plasma equilibrium. In particular, the feedback of the magnetic island on global plasma transport is neglected, so that the background equilibrium profiles are kept constant throughout the simulation. This assumption enables the construction of a computationally lightweight simulation environment, which can be iterated a large number of times during training. In the following section the physical model for the evolution of the NTM will be explained.

2.2.1. NTM Evolution Model

The temporal evolution of the magnetic island width is modeled through a generalized Rutherford equation[16], which relates the growth rate of the island to the combined effect of destabilizing and stabilizing contributions acting on the local current density. In this framework, the island dynamic is governed by a first-order nonlinear ordinary differential equation of the form [17,18]:

$$\frac{dW}{dt} = \frac{r_s^2}{\tau_r g_1}\Big(\Delta'_0(W) + \Delta'_{bs}(W) - \Delta'_{GGJ}(W) - \Delta'_{stab}(W, P_{ec}, \alpha)\Big), \tag{2}$$

where W denotes the island width, r_s the radial position of the resonant surface, and τ_r the resistive time. The various Δ' terms represent the physical mechanisms contributing to the island evolution. The term $\Delta'_0(W)$ accounts for the classical tearing stability index [19] and decreases with increasing island width [20]. The bootstrap-driven contribution $\Delta'_{bs}(W)$ [21] represents the destabilizing effect associated with the reduction of bootstrap current inside the island due to profile flattening. Stabilizing contributions include the Green–Glasser–Johnson [22] curvature term $\Delta'_{GGJ}(W)$ and the active control term $\Delta'_{stab}(W, P_{ec}, \alpha)$. It is noteworthy that, while the Δ'_{GGJ} depends on stabilizing phenomena naturally occurring in a tokamak, the second term depends on the poloidal injection angle α and the injected power and models the effect of externally driven current and localized heating produced by electron cyclotron waves [23] .

The Figure 2 provides an overview of the model and the physical problem underlying Neoclassical Tearing Mode (NTM) evolution. The left panel illustrates the free evolution of the mode in the $(dW/dt, W)$ phase space, highlighting the phenomenology of magnetic island growth: as the island width increases, the growth rate progressively declines until it reaches zero, at which point the island attains a saturated amplitude and ceases to grow. It is worth noting that the physical model implies that the evolution of the mode is inherently a non-linear problem. Conversely, the right panel depicts the G_{CD} term. This parameter is proportional to Δ'_{stab} and accounts for the misalignment between the power deposition location (r_{dep}) and the NTM position (r_s). Notably, the deposition radius depends directly on the injection angle α. The relationship between r_{dep} and α, along with other α-dependent quantities (e.g., deposition width and current drive efficiency), was previously computed using the GRAY code and incorporated into the simulation via a look-up table. All the quantities entering the generalized Rutherford equation are computed from equilibrium parameters characterizing the reference plasma scenario. In the present study, these parameters are extracted from a DTT equilibrium obtained through integrated modeling, as the JETTO–JINTRAC suite [14]. The equilibrium profiles provide the necessary information on quantities such as the safety factor, pressure gradients, and geometric factors, which determine the various Δ' contributions governing the island evolution. In this configuaration the island saturates to a $W_{sat} \simeq 15$cm (as shown in Figure 2(left)) and the power required to fully suppress the saturated island is around 3.5MW using a continuous injection perfectly aligned with the O-point. The equilibrium quantities entering these terms are kept fixed, while the stabilizing contribution Δ'_{stab} depends on the control actions selected by the agent, namely the injected power and the poloidal injection angle. At each time step, the selected action modifies the stabilizing term, and the generalized Rutherford equation is integrated over a 10 ms interval to obtain the updated island width.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

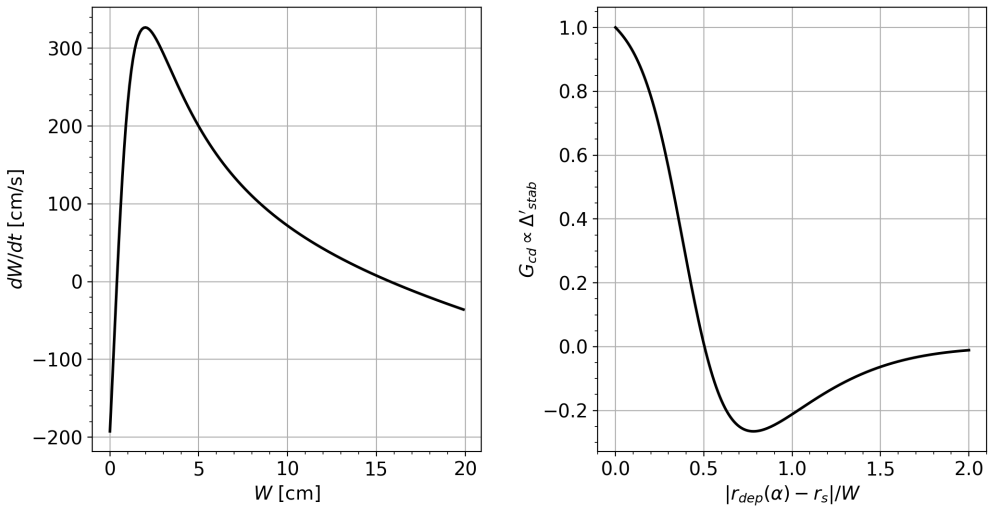

Figure 2. (Left) Plot of the Generalized Rutherford Equation in the phase space $(W, dW/dt)$ (Right) Plot of the G_{cd} function, proportional to Δ'_{stab}, it represents the reduction caused by a misalignment between the O-point of the island and the deposition.

2.2.2. Reward Function

The scalar reward is computed in the environment as the sum of three active components: (i) a distance-to-target variation term, (ii) a growth-rate magnitude term, and (iii) a launcher-angle threshold penalty. The total reward is defined as

$$r(t) = r_{\delta W}(t) + r_{\delta \dot{W}}(t) + p_{\alpha}(t). \tag{3}$$

The first component measures the variation of the absolute distance from the target width:

$$r_{\delta W}(t) = 5 \times 10^2 (W(t-1) - W(t)). \tag{4}$$

This term rewards reductions in the distance to $W = 0$. It is positive when the island approaches the target and negative when it moves away. Its magnitude depends on how fastly the island approaches to zero. The second component promotes large island growth-rate magnitude:

$$r_{\delta \dot{W}}(t) = -10\left(\left|\dot{W}(t-1)\right| - \left|\dot{W}(t)\right|\right) \tag{5}$$

This formulation encourages the agent to prioritize dynamically evolving regimes over steady-state saturation. This term represents a shaping potential $\Phi(t) = |\dot{W}(t)|$, a specific functional form that has been theoretically proven to preserve the optimal policy [24]. In practice, this term effectively eliminates the local minima that often lead the neural network to accept island saturation as an acceptable outcome.

An additional penalty term is introduced to discourage persistent attempts to move the launcher beyond its admissible angular limits. Let $n_{\text{viol}}(t)$ denote the number of consecutive time steps up to time t during which the agent attempts to move the launcher outside the allowed angular range. If such a violation occurs, the following penalty is applied:

$$p_{\alpha}(t) = -0.2\, n_{\text{viol}}(t). \tag{6}$$

If no violation is detected at time t, the counter is reset:

$$n_{\text{viol}}(t) = 1 \tag{7}$$

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

This mechanism introduces a progressively increasing cost for repeated boundary violations. Unlike a fixed penalty applied independently at each time step, the linear accumulation in n_{viol} penalizes persistent boundary-pushing behavior more severely than isolated events. From a control perspective, this term promotes the learning of admissible control strategies within operational constraints.

2.3. Training Algorithm

The Proximal Policy Optimization (PPO) algorithm [7] is a policy-based and model-free reinforcement learning method. This formulation naturally supports stochastic policies, which favor exploration of the action space. Moreover, policy-based methods allow the straightforward incorporation of intrinsic motivation mechanisms, such as curiosity-driven exploration [25], enabling the agent to discover novel strategies beyond those induced solely by the external reward. PPO is also a model-free algorithm. In this setting, the agent does not construct or approximate an explicit model of the environment dynamics, i.e., it does not estimate the transition probability distribution $P(s_{t+1} \mid s_t, a_t)$. This characteristic reduces the dependence of the controller on a specific analytical representation of the system dynamics. In complex systems such as plasmas, different operating conditions and physical parameters may lead to qualitatively different response of the system. Designing a controller based on an explicit model tailored to a particular configuration can therefore limit its applicability to other scenarios. A model-free approach mitigates this issue, as the policy is learned directly from interaction data rather than being tied to a predefined transition model of a specific plasma configuration. PPO is typically implemented within an actor–critic architecture [26]. In this framework, the learning system is composed of two complementary components: the *actor*, which represents the policy, and the *critic*, which evaluates its performance. In the present implementation, both components are incorporated within a single neural network using a multi-head structure. A shared part of the network processes the input state, after which two distinct output branches are defined. The actor head outputs the stochastic policy $\pi_\theta(a|s)$, i.e., the probability distribution over the available actions. The critic head provides an estimate of the expected return, commonly referred to as the value function:

$$V_\theta(s_t) = \mathbb{E}_{\pi_\theta}[G_\gamma(t) \mid s_t], \tag{8}$$

The critic therefore supplies an estimate of the long-term performance associated with a given state. The critic plays a central role in guiding the policy update. Its output $V_\theta(s_t)$ represents the expected return conditioned on the current state, i.e., the performance that the network anticipates when the system is in a given configuration. In other words, for a fixed state s_t, the critic estimates how well the agent is expected to perform in the future if it continues to follow the current policy. Based on this estimate, the advantage function is introduced to quantify how beneficial a specific action is compared to this expectation. For a given time step t, the advantage is defined as

$$A_t = G_\gamma(t) - V_\theta(s_t), \tag{9}$$

where $G_\gamma(t)$ is the realized (discounted) return and $V_\theta(s_t)$ is the critic's estimate. The advantage therefore measures the discrepancy between the actual performance obtained along a trajectory and the performance predicted by the critic. If $A_t > 0$, the executed action has led to better-than-expected performance and is thus advantageous. Conversely, if $A_t < 0$, the action has degraded performance relative to the critic's expectation. The magnitude of A_t determines how strongly the policy should be reinforced or penalized: the larger the positive difference between $G_\gamma(t)$ and $V_\theta(s_t)$, the more the corresponding action should be encouraged during the policy update. The advantage is directly used to calculate the loss function of the network.

2.3.1. Loss Function

The training objective of the network is defined as a weighted sum of three components:

$$L(t) = c_1 L_{\text{critic}}(t) + c_2 L_{\text{actor}}(t) + c_3(i) L_{\text{entropy}}(t), \tag{10}$$

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

where c_1, c_2, and c_3 are scalar coefficients controlling the relative contribution of each term. As in standard machine learning procedures, training proceeds by minimizing the total loss L through gradient-based optimization. The critic is trained to approximate the expected return associated with a given state. Its loss function therefore measures the discrepancy between the predicted value $V_\theta(s_t)$ and the empirical return $G_\gamma(t)$. This is typically expressed as a distance according to a chosen metric. We considered the mean squared error:

$$L_{\text{critic}}(t) = (V_\theta(s_t) - G_\gamma(t))^2, \tag{11}$$

where the sum runs over the collected trajectory samples and N denotes the number of training points. Minimizing L_{critic} encourages the critic to provide an increasingly accurate estimate of the return.

The actor loss is constructed to reinforce actions that yield positive advantage while preventing excessively large policy updates. In PPO, this is achieved through the clipped surrogate objective:

$$L_{\text{actor}} = -\min\left(\frac{\pi_\theta^{(i)}(a_t \mid s_t)}{\pi_\theta^{(i-1)}(a_t \mid s_t)} A_t,\ \text{clip}\left(\frac{\pi_\theta^{(i)}(a_t \mid s_t)}{\pi_\theta^{(i-1)}(a_t \mid s_t)}, 1-\epsilon, 1+\epsilon\right) A_t\right), \tag{12}$$

In this formulation, the sign of the advantage directly determines the direction of the policy update. If $A_t > 0$, the surrogate objective encourages an increase in the probability $\pi_\theta^{(i)}(a_t|s_t)$ of the sampled action, reinforcing behaviors that have produced better-than-expected returns. Conversely, if $A_t < 0$, the optimization step reduces the probability of that action, discouraging decisions that have led to lower-than-expected performance. In this way, the actor updates are guided by a relative performance signal: actions are not rewarded or penalized in absolute terms, but according to how much they improve or degrade performance with respect to the critic's expectation.

The clipping operator contrains the magnitude of policy updates between successive iterations. Specifically, the ratio $\frac{\pi^{(i)}(a_t|s_t)}{\pi^{(i-1)}(a_t|s_t)}$ is limited by the clipping operator, defined is function of the hyperparameter ϵ. This operation restricts the ratio to lie within the interval $[1-\epsilon,\ 1+\epsilon]$, thereby preventing excessively large deviations of the updated policy from the previous one. As a consequence, PPO achieves a balance between policy improvement and training stability: the advantage signal drives the direction of the update, while the clipping function ensures that the step size remains conservative.

The final contribution to the loss function is the entropy term, introduced to promote exploration during training. For a discrete action space with N_{actions} possible actions, the entropy of the policy at time t is defined as

$$L_{\text{entropy}} = -\frac{1}{N_{\text{actions}}} \sum_{i=1}^{N_{\text{actions}}} \pi_\theta^{(i)}(a_i \mid s_t) \log \pi_\theta^{(i)}(a_i \mid s_t). \tag{13}$$

The corresponding loss contribution is constructed so as to maximize this entropy. The entropy term prevents the probability distribution over actions from collapsing prematurely toward a nearly deterministic policy. Without this regularization, the agent may converge too early to a suboptimal strategy, thereby reducing exploration and increasing the risk of being trapped in a local optimum. By maintaining sufficient stochasticity in the action selection process, the entropy term supports a broader exploration of the control space during training. It is important to note that the coefficient determining the magnitude of the L_{entropy} depends on the iteration number i. The term c_3 decreases linearly from a maximum (larger exploration) at the begin of the training to a minimum (less exploration) reached at the 80% of the training process.

2.4. Neural Network Architecture

The input layer of the network is connected to a Long Short-Term Memory (LSTM) module [27,28] . LSTM units are a class of recurrent neural networks specifically designed to handle sequential data. They incorporate internal memory mechanisms that allow the network to conserve and update information over both short and long temporal horizons. Importantly, the parameters governing these

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

memory mechanisms are not predefined: the corresponding weights are learned during training, so that the network autonomously determines which features are relevant for the control task. In the present implementation, the LSTM layer is composed of 64 hidden units. The hidden state dimension therefore corresponds to 64 features, which are subsequently processed by the actor and critic branches.

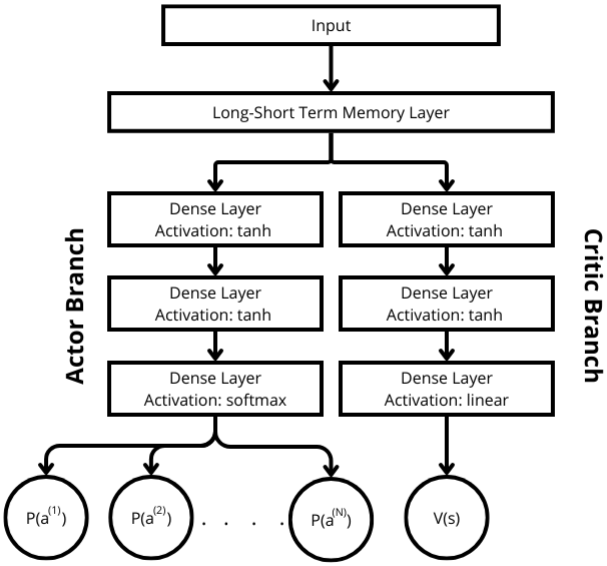

Figure 3. Illustration of the structure of the network. The input enters in the LSTM module and then the output of the layer is split in two branches. The Actor branches, composed by a sequence of Dense layers, outputs the probability of taking a certain action. The Critic branch, whose structure is the same as the actor branch, outputs the value used to calculate the loss function.

This configuration enables the network to maintain an expressive internal representation of the temporal evolution of the system.

Following the LSTM module, the network splits into two fully connected branches implementing the actor and the critic (multi-head architecture). Both branches take as input the LSTM output and consist of two dense layers with 64 neurons each and an hyperbolic tangent as activation function. The final layer of the actor branch (actor head) has N_{actions} output units and a softmax activation layer, whose output integrates to 1, consistent with the meaning of action probability distribution $\pi_\theta(a|s)$. The *critic* branch follows an analogous structure, but its output head consists of a single neuron with linear activation, returning the scalar value estimate $V_\theta(s)$.

3. Alignment Task with Fixed Power

In this configuration, the agent operates in a discrete action space composed of three possible commands acting on the poloidal launcher angle. At each time step, the agent can either increase the angle by a fixed increment $\delta\alpha$, decrease it by the same amount, or leave it unchanged.

Table 1. Discrete action space used in the angular-only control configuration.

Action ID	Command	Description
0	$+\delta\alpha$	Increase launcher angle
1	$-\delta\alpha$	Decrease launcher angle
2	0	Keep launcher angle unchanged

The angular step $\delta\alpha$ defines the control resolution and remains constant throughout the training. The state (input) of the system is defined by the island width $W(t)$, its growth rate $\Delta W/\Delta t = \big(W(t) -$

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

$W(t-dt))/dt$, the last executed action $a(t-dt)$, and a binary flag indicating whether the launcher angle was modified at the previous time step (1 if moved, 0 otherwise). The power level injected in the plasma is taken fixed and equal to $P = 6$MW. Under these conditions, the learning task was reduced to a pure alignment problem: the agent was required to steer the deposition location toward the resonant surface without being responsible for managing the power. This configuration allows a clear assessment of the agent's capability to learn optimal angular steering strategies independently of power effects.

3.1. Training

During the training phase, the initial launcher angle was randomly sampled from intervals deliberately chosen far from the optimal alignment, namely $\alpha \in [7°, 12°]$ and $\alpha \in [18°, 23°]$. As a consequence, the agent was never initialized in the vicinity of the optimal angle during training. Training under these constraints was successful and showed stable convergence. Figure 4 shows the progresses throughout the training expressed in terms of the average reward on all the episodes.

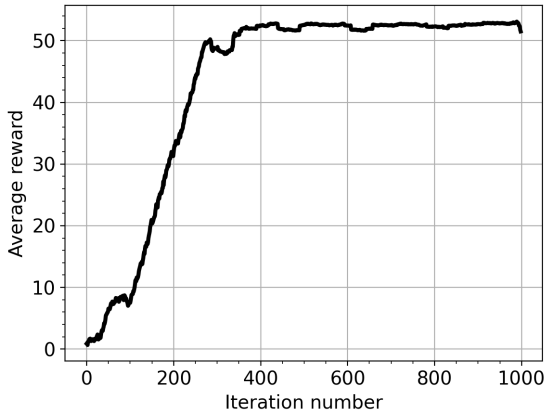

Figure 4. Average reward per iteration over the training process of the network in the angular-only control configuration.

In particular, a consistent optimal policy emerged after approximately 300–400 training iterations. Above this point the reward saturates at a value slightly above 50. Beyond this point the agent had learned a robust alignment strategy. Figure 5 shows the evolution of the individual reward components during training as a function of the iteration number. The green curve corresponds to the main stabilization term $r_{\delta W}$, the black curve represents the shaping contribution $r_{\delta \dot{W}}$ aimed at discouraging island saturation, and the red curve denotes the angular constraint penalty p_α. The dominant contribution throughout training is the main stabilization term (green), which drives the overall reward signal. Its evolution closely follows the trend of the total reward, exhibiting a progressive increase as the agent improves its alignment strategy and achieves more effective island suppression. The gradual growth of this component reflects the increasing ability of the policy to produce consistent reductions in island width. The angular penalty term (red) shows a decrease in magnitude during the early training phase. This indicates that the agent quickly learns to avoid persistent violations of the physical angular constraints. In practice, the network first focuses on identifying the admissible region of the control space, reducing the frequency of boundary-saturating actions. By restricting its effective exploration domain, the agent increases the probability of discovering physically meaningful stabilization strategies, which in turn contributes to the overall reward improvement. The different terms of the reward function converge to a positive value with slightly different timing. This behavior is consistent with its intended role: it is not designed to be the primary optimization driver, but rather to discourage strategies leading to island saturation (i.e., vanishing growth rate). Its comparatively smaller magnitude ensures that it acts as a regularization mechanism, guiding the dynamics without overriding the main stabilization objective.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

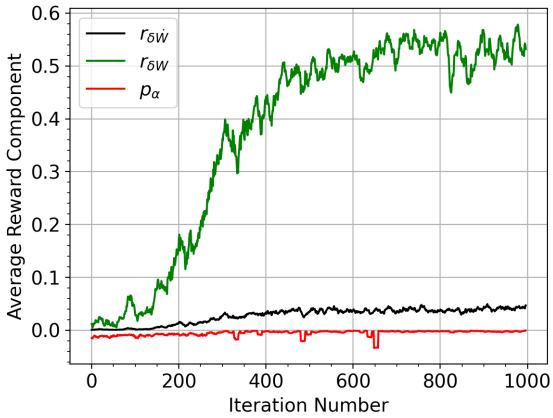

Figure 5. Average reward components versus training iteration:main stabilization term $r_{\delta W}$ (green), saturation-avoidance term $r_{\delta\dot{W}}$ (black), and angular constraint penalty p_α (red).

Figure 6 shows a representative simulation obtained after the training. At this stage, the training process is close to completion, and the agent is expected to have converged toward the optimal policy. The top panel displays the time evolution of the island width W, the middle panel the poloidal launcher angle at each time step, and the bottom panel the individual components of the reward function. During the first $\sim$ 5 time steps, the agent explores the angular space. In this phase, the launcher angle is progressively decreased, moving away from the optimal alignment, until the lower angular boundary is reached. At that point, the agent attempts to further decrease the angle; however, since the physical limit has been reached, the angle no longer changes. When the lower angular boundary is reached, the agent may still select the action corresponding to a further decrease of the launcher angle. However, the actuator saturates at the physical limit and the commanded action produces no actual angular displacement. This effect is explicitly observable by the agent through the `angle flag` included in the state vector, which is set to 1 only when the launcher angle is effectively updated and to 0 otherwise.

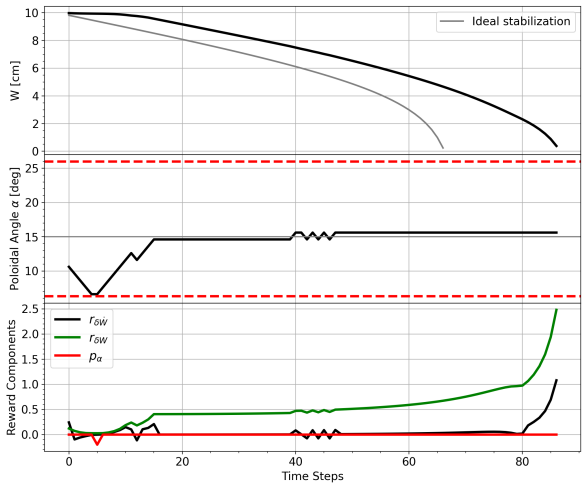

Figure 6. Typical behavior of the controller in the alignment task. (Top) Evolution of the island width (Middle) value of the injection poloidal angle α (Bottom) Components of the rewards

Therefore, the agent can learn during the training process that the selected control direction is not feasible because the action does not translate into state evolution, despite being repeatedly chosen. In addition, the reward signal includes the angular penalty term (red curve in the bottom panel), which

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

provides a negative reinforcement for persistent attempts to exceed the admissible angular range and further discourages this behavior. Subsequently, the agent reverses the steering direction and begins increasing the launcher angle. After approximately ten time steps, the island width starts to decrease, indicating that the deposition is moving toward the resonant surface. From this point onward, the agent consistently reinforces this control direction, steering the launcher toward the optimal alignment and eventually stabilizing around the corresponding angle. The reward components reflect this learning behavior. In the initial phase, strategies leading to island saturation are penalized, as evidenced by the negative contribution of the $r_{\delta\dot{W}}$ term (black line third plot). Once proper alignment is achieved, the main reward component becomes positive and progressively increases. A sharp growth of the total reward is observed when the island suppression accelerates, highlighting the dominance of the stabilization term as the island approaches complete suppression.

3.2. Test and Discussion

The gray trace in Figure 6 (Top) represents the trajectory of the island in case of ideal stabilization, i.e. $\alpha = 15$ alinged with the O-point of the island, for the same power level. In this case the island is stabilized in around 0.65s. Figure 7 reports the mean stabilization time as a function of the initial launcher angle. For each initial condition, the stabilization time is computed as the average over a statistical ensemble of 20 independent episodes. A clear minimum is observed in the vicinity of $\alpha \simeq 15°$, which corresponds to the optimal alignment with the island O-point. When the initial angle is close to this value, the agent requires minimal corrective action, and stabilization is achieved in the shortest time. Moving away from the optimal angle in either direction results in a progressive increase of the stabilization time. A larger initial angular misalignment implies that the launcher must explore a wider radial region before reaching the optimal deposition location. Consequently, the agent requires additional control steps to identify and reach the stabilization configuration. It is noteworthy that the presence of a well-defined minimum around $\alpha \simeq 15°$ indicates that the learned policy extends coherently across the angular domain, rather than being confined to the regions explored during training. It is important to emphasize that the agent was not provided with any prior information regarding the optimal alignment position. This first test demonstrates that the reinforcement learning framework is capable of autonomously identifying the optimal geometric alignment strategy when the control space is reduced to angular steering alone. The emergence of a stable optimal policy indicates that the reward formulation provides a sufficiently informative and well-shaped learning signal. The agent is able to identify and reinforce the correct alignment strategy without requiring additional heuristic adjustments. Consequently, for the subsequent extensions of the control problem, no structural modification of the reward function is introduced.

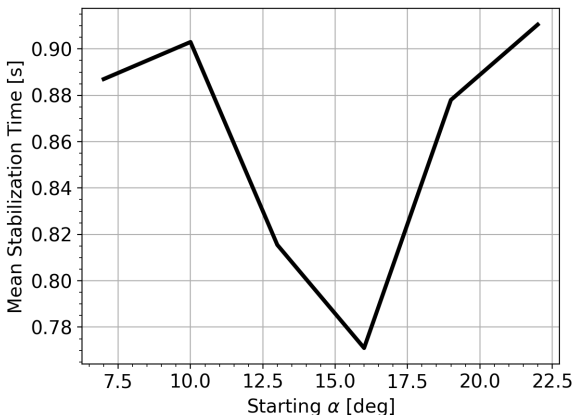

Figure 7. Average stabilization time calculated on a set of different starting angle α.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

4. Alignment Task with Power Control

In this extended configuration, the agent is allowed to control both the launcher angle and the injected EC power. The action space is therefore expanded to five discrete actions, acting either on the angular alignment or on the power level. At each time step, the policy may: increase the launcher angle by a fixed increment $\delta\alpha = 1^\circ$, decrease it by the same amount, increase the injected power by 1 MW, decrease it by 1 MW, or leave both quantities unchanged. The injected power is initialized to zero at the beginning of each episode. Consequently, the agent must not only identify the correct geometric alignment, but also determine when and how rapidly to increase the power in order to suppress the island. The enlargement of the action space increases the complexity of the control task. The state (input) of the system is defined by the island width $W(t)$, its growth rate $\Delta W/\Delta t = \big(W(t) - W(t-dt)\big)/dt$, the last executed action $a(t-dt)$, the previously injected power $P(t-dt)$, and a binary flag indicating whether the launcher angle was modified at the previous time step (1 if moved, 0 otherwise). Importantly, the reward formulation remains unchanged, so that any difference in learning behavior can be attributed to the higher-dimensional control space rather than to modifications of the optimization objective.

Table 2. Discrete action space used in the alignment task with power control.

Action ID	Command	Description
0	$+\delta\alpha = 1^\circ$	Increase launcher angle
1	$-\delta\alpha = -1^\circ$	Decrease launcher angle
2	$+\delta P = 1$MW	Increase injected power
3	$-\delta P = -1$ MW	Decrease injected power
4	/	No changes in power and angle

4.1. Training

Figure 8 shows the evolution of the average reward during training in the joint angular and power control configuration. Compared to the fixed-power case, the learning process appears significantly slower and less stable. The increased dimensionality of the action space introduces additional complexity in the exploration phase. As a consequence, the reward along the training exhibits stronger fluctuations and a more gradual improvement over iterations.

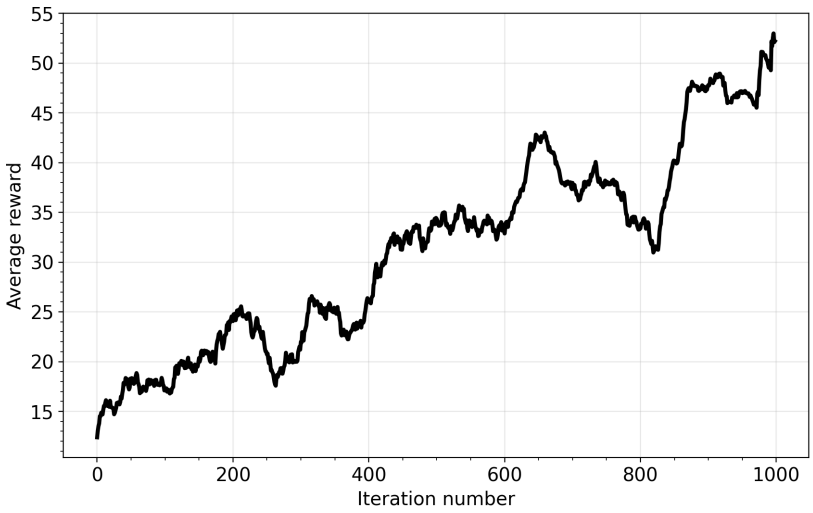

Figure 8. Average reward per iteration through a complete training process for the alignment task with power control.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

Performance comparable to the alignment-only configuration is reached only toward the final phase of training. Even in the late iterations, however, the reward does not display the same level of stability observed in the fixed-power case, suggesting that convergence is less robust under the current hyperparameter configuration. Despite these limitations, the overall trend remains positive: the average reward increases steadily throughout training, indicating that the agent successfully learns to coordinate angular steering and power. While the final performance is not yet optimal, the controller demonstrates clear improvement over iterations and establishes a viable basis for further refinement.

4.2. Test and Discussion

Figure 9 illustrates a representative example of the controller operating under the joint angular and power control configuration at the end of training. During the first $\sim$ 10 time steps, the agent increases the injected power in discrete steps of 1 MW, progressively ramping it up to the maximum available level of 8MW. In this initial phase, the launcher angle remains essentially unchanged, indicating that the policy prioritizes establishing a sufficient stabilization drive before refining the alignment. Once the power has reached its maximum value, the agent begins adjusting the launcher angle. A reduction of the angle by one increment $\delta\alpha$ produces a measurable decrease in the island width, indicating that the steering direction is correct. The policy then consistently continues to move the launcher in the same direction, applying successive 1° corrections until optimal alignment is reached. After convergence to the optimal angular region, the angle remains approximately constant while the island width decreases monotonically toward suppression.

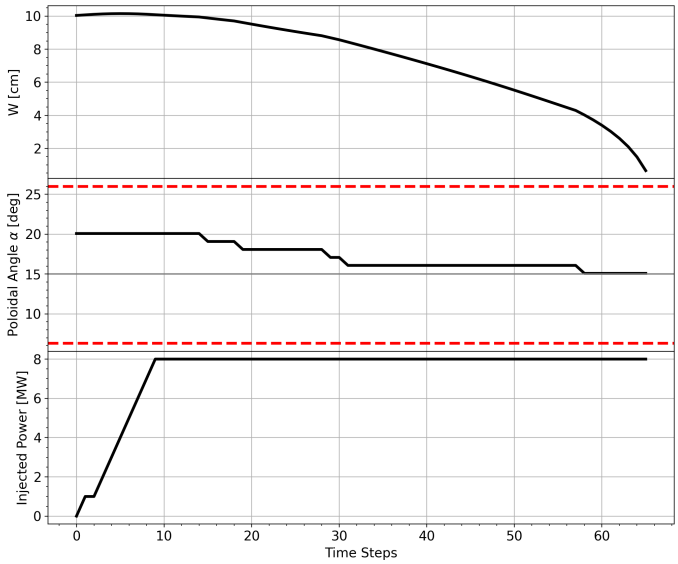

Figure 9. Typical behavior of the controller in the alignment task. (Top) Evolution of the island width (Middle) value of the injection poloidal angle α (Bottom) value of the injected power.

A notable feature of this behavior is that the agent maintains the injected power at its maximum level throughout the stabilization phase. While this strategy is effective in achieving suppression, it is not necessarily optimal from an energetic standpoint. The early and sustained ramp-up of power suggests that the current policy prioritizes rapid stabilization over power efficiency. The optimal policy can be improved for instance by delaying the power ramp-up until near-optimal alignment is achieved, thereby reducing unnecessary power usage while preserving stabilization performance. Figure 10 reports the success rate, defined as the percentage of runs leading to complete island suppression, as a function of the initial launcher angle. For each angular value, 50 independent simulations were performed and the number of successful suppressions was recorded. For initial angles up to

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

approximately 17°–18°, the controller achieves a success rate between 96% and 100%, indicating robust performance over a broad portion of the angular domain. In this region, the policy is generally able to identify the correct steering direction and coordinate alignment with power modulation effectively. However, a marked degradation in performance is observed for initial angles above 20°, where the success rate drops to 92 %

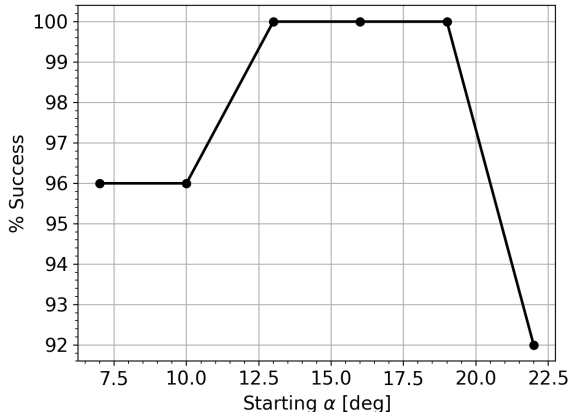

Figure 10. Stabilization success as function of the poloidal initial starting angle α.

The observed asymmetry in performance indicates that further optimization of the training procedure is required. In particular, additional fine-tuning of hyperparameters or an improved exploration strategy may enhance stability and increase the success probability for more challenging initial conditions. Figure 11 shows the mean stabilization time as a function of the initial poloidal angle α. For each starting angle, the reported value is obtained by averaging over an ensemble of 50 independent trajectories. The results indicate that the average suppression time is approximately 1 s for a broad range of initial conditions. However, a clear increase in stabilization time is observed when the initial angle exceeds $\sim 20°$, highlighting a degradation in control performance for large initial misalignments.

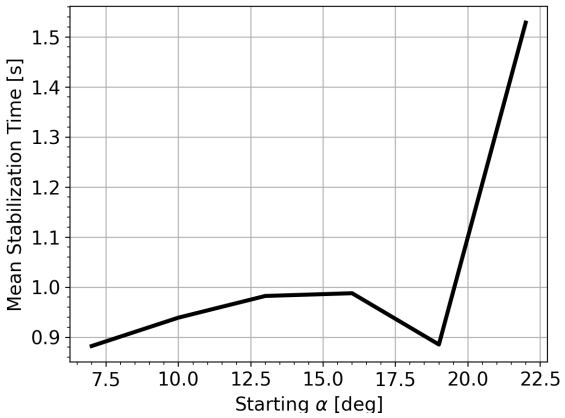

Figure 11. Stabilization time as function of the poloidal initial starting angle α. The stabilization time is calculated on a statistical population of 50 trajectories per each angle.

5. Power Optimization

In order to promote a more energy-efficient control strategy, the reward function was further modified by introducing an additional cost term penalizing the use of high injected power. The

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

additional power cost is formulated as a hinge-like function, i.e., a piecewise-linear term that remains zero below a prescribed threshold and increases beyond it:

$$C_p = -\lambda \max\left(0, P - P_{\text{th}}\right)^2, \tag{14}$$

where P is the injected power, P_{th} is a predefined threshold, and λ controls the strength of the penalty. The maximum power available for the stabilization is 8MW and the threshold power above which the network is penalized, is set to be at 40 % of the maximum power. The quadratic dependence ensures moderate power exceedances are only mildly discouraged, whereas significantly high power levels are strongly penalized. This modification is intended to discourage unnecessarily large power levels while still allowing the agent to exploit them when strictly required for stabilization.

Starting from the previously trained network controlling both poloidal angle and injected power, a new training phase was performed with the modified reward function. In this way, the agent was allowed to refine its strategy under the additional constraint of power minimization, without altering the underlying architecture or learning algorithm.

Figure 12 illustrates a representative stabilization trajectory obtained with the reward function including the quadratic power penalty. The simulation starts from an initial poloidal angle of 10°. The upper panel shows the evolution of the island width, which is progressively reduced until complete suppression. The middle panel reports the evolution of the launcher poloidal angle, while the lower panel displays the injected EC power. During the initial phase, the agent performs the alignment process while increasing the injected power only up to approximately 4 MW. Once the launcher approaches the optimal alignment, the power is incremented more, first by 1 MW and finally reaching the maximum level only in the final stage of the suppression process.

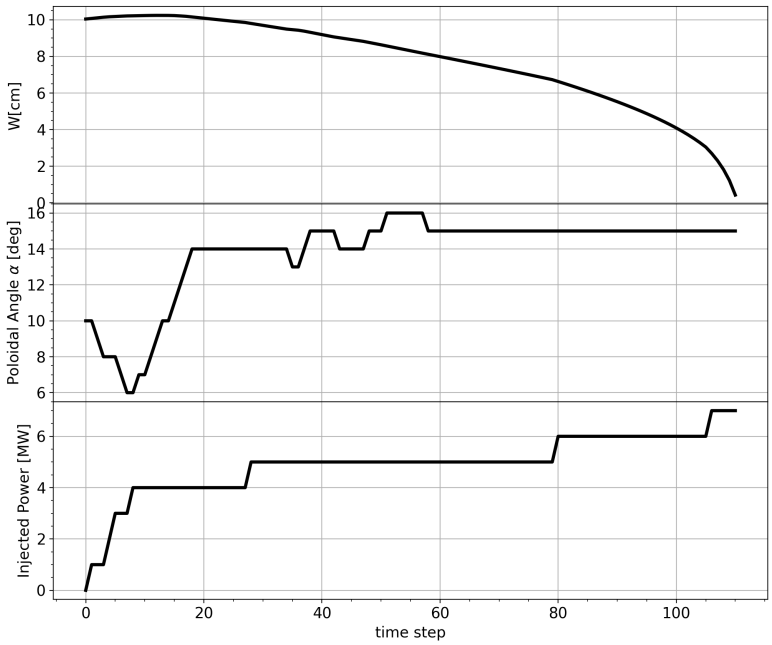

Figure 12. Trajectory of the island width (Top), the injection angle (Middle) and the power level (Bottom) in an episode for the optimized policy

Figure 13 presents a comparison of the mean energy expenditure as a function of the initial poloidal angle, with and without power optimization. For each initial condition, the reported value corresponds to the average energy spent during the stabilization process. The optimized controller consistently reduces the total energy consumption across all considered starting angles. The effect is

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

particularly pronounced for initial angles larger than 20°, where the non-optimized strategy exhibits a marked increase in energy usage, while the optimized controller maintains significantly lower values.

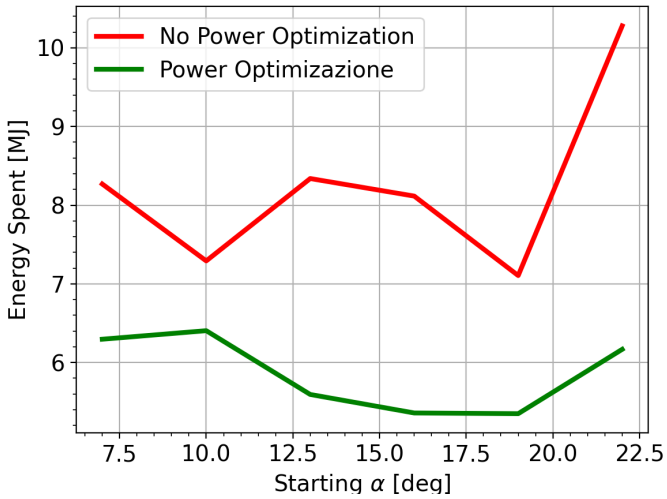

Figure 13. Energy consumption for island suppression as a function of the poloidal starting angle α. Here is presented a comparison between optimized (green line) and non optimized network (red line)

6. Conclusions

In this work, a reinforcement-learning-based controller for magnetic island stabilization has been investigated within a synthetic environment. The objective was to explore, in a controlled and interpretable setting, the potential of neural-network-based strategies for the suppression of neoclassical tearing modes. The controller developed in this work is intentionally blind: it does not receive any information about the deposition location, the position of the resonant surface, or the relative alignment between launcher and island. The input state is solely represented by the magnetic island amplitude and its growth rate implementing a kind of optimum seeking RL-based controller. Differently from 'optimum seeking' algorithms, different features can be performed by the architecture here proposed. The controller takes into account the limited operational angular range of the launcher, as implemented in Sec.3. Further, it can be trained also to deal with power sources, as in Section 4. In Section 5, it is shown that it is possible to adopt additional policy as saving power consumption for the training of the controller. The results show that, during the training process, the agent is able to successfully converge toward an optimal policy through repeated interaction with the environment. The learning dynamics are driven by the maximization of the return associated with each action, as defined by the prescribed reward function. By iteratively updating its parameters to increase long-term performance, the network autonomously discovers effective stabilization strategies without the need for externally imposed control laws. Two control tasks were considered. In the first case, the agent was trained to perform pure angular alignment, demonstrating rapid convergence toward an optimal stabilization strategy. In the extended configuration, where both angular steering and power control were included, the learning process became more complex but still led to coherent stabilization behaviors. The third development of the agent demonstrated the possibility of optimizing power usage within the control strategy. By introducing a quadratic cost associated with high injected power levels, the reward structure was modified so as to explicitly penalize excessive energy expenditure. As a consequence, the agent adapted its behavior, progressively biasing its strategy toward minimizing the utilized power while preserving stabilization capability. This modification resulted in a significant reduction of the total energy required for mode suppression, highlighting the flexibility of the reward-based framework in shaping the controller behavior according to specific operational objectives. Further work is required to assess the response of the network to modifications of the environment. In particular, it will be necessary to evaluate the robustness of the learned policy under perturbations such

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

as variations in density and temperature, shifts of the resonant surface, or the presence of measurement errors. Additionally, a more realistic representation of the actuator dynamics should be implemented, especially accounting for delays between measurement acquisition and action execution, which may significantly influence the agent's behavior.

Author Contributions: Conceptualization, L.B.,E.A. and C.S.; methodology, L.B.; software, L.B.; validation, L.B., E.A. and C.S.; formal analysis, L.B.; investigation, L.B.; writing—original draft preparation, L.B.; writing—review and editing, L.B.,E.A.,C.S.,E.L. and S.N.; visualization, L.B.; supervision E.A.,C.S.,E.L. and S.N.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author(s).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. De Vries, P.; Johnson, M.; Alper, B.; Buratti, P.; Hender, T.; Koslowski, H.; Riccardo, V.; Contributors, J.E. Survey of disruption causes at JET. *Nuclear fusion* **2011**, *51*, 053018.
2. Lehnen, M.; Aleynikova, K.; Aleynikov, P.; Campbell, D.; Drewelow, P.; Eidietis, N.; Gasparyan, Y.; Granetz, R.; Gribov, Y.; Hartmann, N.; et al. Disruptions in ITER and strategies for their control and mitigation. *Journal of Nuclear materials* **2015**, *463*, 39–48.
3. Sauter, O.; Henderson, M.; Ramponi, G.; Zohm, H.; Zucca, C. On the requirements to control neoclassical tearing modes in burning plasmas. *Plasma Physics and Controlled Fusion* **2010**, *52*, 025002.
4. Stober, J.; Barrera, L.; Behler, K.; Bock, A.; Buhler, A.; Eixenberger, H.; Giannone, L.; Kasparek, W.; Maraschek, M.; Mlynek, A.; et al. Feedback-controlled NTM stabilization on ASDEX Upgrade. In Proceedings of the EPJ Web of Conferences. EDP Sciences, 2015, Vol. 87, p. 02017.
5. Humphreys, D.; Ferron, J.; La Haye, R.; Luce, T.; Petty, C.; Prater, R.; Welander, A. Active control for stabilization of neoclassical tearing modes. *Physics of Plasmas* **2006**, *13*.
6. Seo, J.; Kim, S.; Jalalvand, A.; Choi, W.H.; Linder, B.; Kolemen, E. Avoiding fusion plasma tearing instability with deep reinforcement learning. *Nature* **2024**, *626*, 746–751. https://doi.org/10.1038/s41586-024-07024-9.
7. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* **2017**.
8. Wilson, H. Neoclassical tearing modes. *Fusion science and technology* **2002**, *41*, 107–116.
9. Puterman, M.L. Markov decision processes. *Handbooks in operations research and management science* **1990**, *2*, 331–434.
10. Hornik, K.; Stinchcombe, M.; White, H. Multilayer feedforward networks are universal approximators. *Neural networks* **1989**, *2*, 359–366.
11. Kaelbling, L.P.; Littman, M.L.; Moore, A.W. Reinforcement learning: A survey. *Journal of artificial intelligence research* **1996**, *4*, 237–285.
12. Belousov, B.; Abdulsamad, H.; Klink, P.; Parisi, S.; Peters, J. Reinforcement learning algorithms: analysis and applications **2021**.
13. Ambrosino, R.; et al. DTT-Divertor Tokamak Test facility: A testbed for DEMO. *Fusion Engineering and Design* **2021**, *167*, 112330.
14. Casiraghi, I.; et al. Core integrated simulations for the Divertor Tokamak Test facility scenarios towards consistent core-pedestal-SOL modelling. *Nuclear Fusion* **2023**, *63*, 036017.
15. Farina, D. A quasi-optical beam-tracing code for electron cyclotron absorption and current drive: GRAY. *Fusion Science and Technology* **2007**, *52*, 154–160.
16. Rutherford, P.H.; et al. Nonlinear growth of the tearing mode. *Physics of Fluids* **1973**, *16*, 1903.
17. Lazzaro, E.; Borgogno, D.; Brunetti, D.; Comisso, L.; Fevrier, O.; Grasso, D.; Lutjens, H.; Maget, P.; Nowak, S.; Sauter, O.; et al. Physics conditions for robust control of tearing modes in a rotating tokamak plasma. *Plasma Physics and Controlled Fusion* **2017**, *60*, 014044.
18. Zohm, H. Stabilization of neoclassical tearing modes by electron cyclotron current drive. *Physics of plasmas* **1997**, *4*, 3433–3435.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 23 April 2026 doi:10.20944/preprints202604.1616.v1

19. Furth, H.; Rutherford, P.; Selberg, H. Tearing mode in the cylindrical tokamak. *The Physics of Fluids* **1973**, *16*, 1054–1063.
20. White, R.B.; Monticello, D.; Rosenbluth, M.N.; Waddell, B.V. Saturation of the tearing mode. Technical report, Princeton Univ., N.J. (USA). Plasma Physics Lab., 1976. https://doi.org/10.2172/7136568.
21. Gorelenkov, N.; Budny, R.; Chang, Z.; Gorelenkova, M.; Zakharov, L. A threshold for excitation of neoclassical tearing modes. *Physics of Plasmas* **1996**, *3*, 3379–3385.
22. Glasser, A.; Greene, J.; Johnson, J. Resistive instabilities in general toroidal plasma configurations. *The Physics of Fluids* **1975**, *18*, 875–888.
23. De Lazzari, D.; Westerhof, E. On the merits of heating and current drive for tearing mode stabilization. *Nuclear Fusion* **2009**, *49*, 075002.
24. Ng, A.Y.; Harada, D.; Russell, S. Policy invariance under reward transformations: Theory and application to reward shaping. In Proceedings of the Icml. Citeseer, 1999, Vol. 99, pp. 278–287.
25. Pathak, D.; Agrawal, P.; Efros, A.A.; Darrell, T. Curiosity-driven exploration by self-supervised prediction. In Proceedings of the International conference on machine learning. PMLR, 2017, pp. 2778–2787.
26. Konda, V.; Tsitsiklis, J. Actor-critic algorithms. *Advances in neural information processing systems* **1999**, *12*.
27. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural computation* **1997**, *9*, 1735–1780.
28. Staudemeyer, R.C.; Morris, E.R. Understanding LSTM - a tutorial into Long Short-Term Memory Recurrent Neural Networks. *CoRR* **2019**, *abs/1909.09586*, [1909.09586].

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

© 2026 by the author(s). Distributed under a Creative Commons CC BY license.