Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 277 (2026) 2585–2594

Procedia Computer Science

www.elsevier.com/locate/procedia

7th International Conference on Industry of the Future and Smart Manufacturing
(former International Conference on Industry 4.0 and Smart Manufacturing)

Online adaptation of an Echo State Network based Controller for Brushless-DC Motors via Modified FORCE Recursive Least Squares

Mariorosario Prist[a], Lorenzo Longarini[a], Alessandro Di Biase[b], Andrea Monteriù[a], Andrea Bonci[a,*]

[a] *Dept. of Information Engineering (DII), Polytechnic University of Marche, 60131 Ancona, Italy*
[b] *Dept. of Electrical and Information Engineering, Polytechnic of Bari, 70126 Bari, Italy*

Abstract

This paper addresses the growing role of Data Driven methods in industrial control, enabled by advances in sensor technology and Machine Learning toolkits. While neural networks have been successfully employed for plant identification, grey-box modelling, direct control synthesis, and pattern generation, their practical adoption remains limited by training costs and numerical stability issues. Echo State Networks (ESN) offer a compelling compromise: a fixed-weight reservoir captures rich nonlinear dynamics, and only a linear readout requires adaptation. However, existing ESN applications in control typically rely on offline training and still depend on a conventional controller to generate calibration data. To overcome these limitations, an ESN based closed loop controller has been introduced, whose output weights are updated online via a modified version of the First-Order Reduced and Controlled Error (FORCE) algorithm, a variant of Recursive Least Squares. During operation, the reservoir's spontaneous dynamics are tamed by feeding the network output back into its hidden units, ensuring a small error from the outset and requiring only modest weight adjustments thereafter. The results confirm that online adaptation preserves the computational efficiency of reservoir computing while extending its applicability to processes with uncertain or time-varying dynamics.

Keywords: Echo State Network; Recursive Least Square; First Order Reduced Controller Error; Control System; Deep Learning.

1. Introduction

The rapid growth in sensor technology allows engineers to collect large volumes of data from industrial plants. At the same time, modern machine-learning tools make it straightforward to extract useful information from these data and to train models that can be deployed in real time. As a result, data-driven methods are gaining importance in control-system design. Within this context, several Neural Network (NN) control strategies have been developed.

* Corresponding author. Tel.: +39-071 220 4666.
E-mail address: a.bonci@univpm.it

For example, Christofides [1] focused on identifying plant dynamics with a neural network and then embedding the learned model in a model-based framework, specifically Model Predictive Control (MPC). Instead, Scattolini [2], by contrast, adopted a hybrid or grey box approach where, starting from first-principles equations, he introduced a NN to capture those system terms that are unknown or hard to derive experimentally. These needs often arise when dealing with complex model-based dynamic systems, such as in autonomous robots [3, 4], aerospace [5, 6], vehicle dynamics [7, 8], and other areas. Other studies employ NN as direct controllers, deriving the control law solely from the plant's input–output data [9]. Within this context, Recurrent Neural Networks (RNNs) are particularly promising. Their internal feedback loops mirror the state evolution of the dynamical system being modeled, linking present outputs to past inputs in a way that naturally captures the plant's behavior. The RNNs have been shown to effectively capture complex dynamics, for example, the hysteresis of soft pneumatic actuators [10] and electroactive polymers [11]. They have been used to model the dynamics of a chemical reactor online with Long Short Term Memory (LSTM) [12]. Further applications concern model-free controls, where PID controllers are widely used in fields ranging from autonomous systems [13] to industrial applications. Here RNNs have been proposed to replace the PID in the speed control of a DC motor after expensive offline training [14]. However, large-scale industrial deployment is still limited by the significant data and computational resources required, together with persistent numerical-stability challenges. A promising compromise is offered by reservoir computing, and in particular by Echo State Networks (ESNs), which separate the dynamic representation, entrusted to a fixed-weight reservoir of neurons, from the learning process, which is confined to the output layer and can be solved using linear regression [15, 16, 17]. This drastically reduces the training cost and enables execution on hardware with limited resources. However, most contributions still require offline training. In some cases, the ESN is trained to replicate the behaviour of a PID to control temperature processes [18], to control a DC motor [19], or an electronic throttle body [20]. In addition, it has been used to act as an internal model in model-predictive control schemes [21, 22] or evolutionary variants [23, 24]. In all cases, the training is performed offline and the classical controller is indispensable for generating the initial dataset, meaning that replacing the controller does not avoid the initial design and calibration phase of the controller itself. Thus, an online learning mode could be more efficient and practical than batch algorithms for ESNs to deal with sequential data. In [25] was proposed the Least Mean Squares (LMS) based online learning algorithm to make the ESN work in an online and adaptive manner while [26] applied the online Gradient Descent algorithms where the network parameters of ESN are iteratively adapted to model the target system. A more efficient technique to apply the self-tuning characteristic of linear regulators to reservoirs, the First Order Reduced Controlled Error (FORCE) algorithm was introduced by Sussillo at al [27] that uses the Recursive Least Squares (RLS) method to update the ESN output weights online. In this scheme, the network's output is fed back into its hidden units during training, suppressing the chaos. Unlike standard gradient descent methods, FORCE keeps the output error low from the outset and needs only small, occasional weight updates to maintain that low error. In this context, the FORCE-RLS has been mainly used to train in real time the ESN for autonomous pattern generation tasks. For example, [28] proposed a transfer-RLS method and transfer-FORCE learning for simple and fast training of reservoir computing models for time series transformation tasks and temporal pattern generation tasks. While [29] introduced a composite FORCE learning of chaotic ESN for time series prediction. Instead, in the automation sector, the FORCE-RLS has been used in internal-model nonlinear control for pneumatic actuators [30], active vibro-isolation [31] and the Bayesian learning of inverse models for robotic trajectories [32]. As described before, most of these approaches focus mainly on the online ESN training for autonomous pattern generation tasks and there is a lack of investigation of the FORCE-RSL applied in a closed-loop controller where the ESN, throught the FORCE-RSL learns in real time to control a plant. Along this way, this study addresses this issue, the offline training of the ESN, by proposing an ESN-based controller whose output layer is updated in real time via FORCE-RLS. The fixed-weight reservoir provides a dynamic basis capable of representing nonlinearities, hysteresis, and dead time. Meanwhile, the RLS adapts the output coefficients using only the tracking error, eliminating the need for pre-recorded data. The result is a self-adaptive system that retains the computational efficiency of a traditional application of RNN models while incorporating the compensatory capabilities of neural networks. This makes it suitable for processes with uncertain or time-varying models. Summarizing, the contributions of this paper are as follows, a viable strategy is proposed that focuses exclusively on the novelty of the approach without, for now, making comparisons with other control strategies, while computational costs are evaluated.

The rest of this paper is organized as follows. In Section 2, the methodology, including a detailed description of the ESN architecture, the real-time FORCE-RLS training algorithm, and the performance metrics employed heve been described. Section 3 illustrates the application of the proposed approach to a specific case study and discusses the obtained results. Finally, Section 4 summarizes the main findings and outlines directions for future work.

2. Materials and Methods

The proposed controller uses an ESN coupled with a modified version of the online RLS learning algorithm to control the speed of electric motors with nonlinear dynamics and dead time introduced by the response time of the inverter or power electronics. The ESN reservoir, is generated with a sparse topology and fixed random weights. The system's adaptive intelligence is relegated to the linear reading layer, whose weights are updated step-by-step by the modified version of the RLS algorithm. This enables the controller to learn in real time without storing its history.

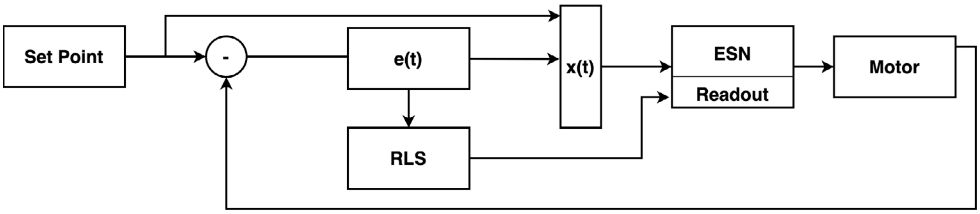

Fig. 1: Diagram Overview of the Closed Loop Control System that includes FORCE-RLS algorithm and the motor.

As shown in Figure 1 the proposed framework is composed by several parts, which include the ESN base controller, the readout coupled with the FORCE-RLS algorithm, the plant to be controlled and the setpoints generation both for training and for validation.

2.1. Echo State Network Base Controller

ESNs are a supervised-learning framework for RNN that addresses the optimisation challenges typically encountered in fully trainable RNNs. An ESN comprises a large, randomly connected layer (recurrent part of ESN), referred to as the reservoir, whose weights remain fixed after initialisation. External inputs, and optionally past outputs fed back into the network, excite the reservoir's high-dimensional nonlinear dynamics, embedding the input history in its evolving state. Learning is restricted to a subsequent linear readout stage, where only the output-layer weights are adjusted so that the network can reproduce a desired target signal.

Let $\mathbf{h}(t) \in \mathbb{R}^N$ denote the internal state of the reservoir at discrete time t. The reservoir dynamics are governed by the following two-stage update rule. First, the state $\mathbf{h}(t)$ is computed via a nonlinear transformation $f(.)$ applied to an affine combination of the current input $\mathbf{x}(t) \in \mathbb{R}^m$ and the previous reservoir state $\mathbf{h}(t-1)$:

$$\mathbf{h}(t) = f\left(\mathbf{W}_{\text{in}}\mathbf{x}(t) + \mathbf{W}_{\text{rec}}\mathbf{h}(t-1) + \mathbf{b}\right), \tag{1}$$

where $\mathbf{W}_{\text{in}} \in \mathbb{R}^{N\times m}$ and $\mathbf{W}_{\text{rec}} \in \mathbb{R}^{N\times N}$ are the input and recurrent weight matrices, respectively, and $\mathbf{b} \in \mathbb{R}^N$ is a bias vector. The function $f : \mathbb{R}^N \to \mathbb{R}^N$ is applied componentwise and is typically chosen as the hyperbolic tangent. Subsequently, the actual reservoir state $\mathbf{h}(t)$ is obtained via a leaky integration scheme, which blends the previous state $\mathbf{h}(t-1)$ with the state $\mathbf{h}(t)$ according to a leaking rate parameter $\alpha \in (0, 1]$:

$$\mathbf{h}(t) = (1-\alpha)\mathbf{h}(t-1) + \alpha\mathbf{h}(t). \tag{2}$$

For an Echo State Network (ESN) to operate reliably, the reservoir must be initialized so as to satisfy the *Echo State Property* (ESP), rigorously examined in [34, 35]. A widely adopted sufficient condition for ensuring the ESP is that the spectral radius of the recurrent weight matrix remains strictly below unity: $\rho(\mathbf{W}_{\text{rec}}) < 1$, where $\rho(\cdot)$ denotes the spectral radius. Enforcing this inequality guarantees that the reservoir dynamics are contractive, thereby preventing unbounded growth of the internal state and ensuring that the network's response is uniquely determined by the input history rather than by initial transients. With this property, it is imposed that the reservoir has a matrix $\mathbf{W}_{rec}$ with spectral radius $\rho(\mathbf{W}_{rec})$ less than 1. The output of the ESN at time t is computed by a linear readout mechanism that projects the current reservoir state $\mathbf{h}(t) \in \mathbb{R}^N$ onto the output space. Specifically, the output $\mathbf{y}(t) \in \mathbb{R}^p$ is given by $\mathbf{y}(t) = \mathbf{W}_{\text{out}}\,\mathbf{h}(t)$, where $\mathbf{W}_{\text{out}} \in \mathbb{R}^{p\times N}$ denotes the readout weight matrix. The parameters in $\mathbf{W}_{\text{out}}$ are the only components of the network subject to training, and they are typically estimated so as to minimize the prediction error with respect to a given target signal. This linear formulation enables efficient learning algorithms such as ridge regression or recursive least squares to be employed during both offline and online training phases. ESN can be trained either offline [36] or online [37]. In the offline setting, a commonly adopted technique is ridge regression, also referred to as Tikhonov regularization [38], which is defined as:

$$\mathbf{W}_{\text{out}} = \mathbf{Y}^T\mathbf{H} \cdot (\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I})^{-1}, \tag{3}$$

where $\mathbf{H} \in \mathbb{R}^{T\times N}$ is the matrix collecting the reservoir states over a training horizon of length T, $\mathbf{Y} \in \mathbb{R}^{T\times p}$ contains the corresponding target outputs, and $\lambda > 0$ is the regularization coefficient. This supervised approach, often referred to as one-shot learning, requires a pre-collected dataset for effective training. In contrast, for online learning, there is a continuous adaptation of the readout during interaction with the system to be controlled, making it particularly suitable for closed-loop control scenarios.

2.2. First Order Reduced Controlled Error Technique

Let $\boldsymbol{h}(t) \in \mathbb{R}^N$ be the state vector of the reservoir, let $r(t) \in \mathbb{R}$ be the target signal (the setpoint) at time t, and let $\boldsymbol{W}_{out}(t) \in \mathbb{R}^{p\times N}$ be the readout weight vector at time t. The goal of FORCE-RLS algorithm is to adapt $\boldsymbol{W}_{out}(t)$ online to minimize the exponentially weighted squared error:

$$J(\boldsymbol{w}) = \sum_{t=0}^{n} \gamma^{n-t}\left[r(t) - \boldsymbol{W}_{out}^{\top}\boldsymbol{h}(t)\right]^2 \tag{4}$$

where $\gamma \in (0, 1]$ is the forgetting factor which is intended to ensure the forgetting of data from the distant past, to give the possibility of tracking the statistical variation of observable data when updating the recursive equation for least squares estimation. So, according to [27], the recursive equation for updating the least square estimate $\mathbf{W}_{out}$ is:

$$\mathbf{W}_{out}(t) = \mathbf{W}_{out}(t-1) - e(t)\cdot\mathbf{P}(t)\mathbf{h}(t) \tag{5}$$

$$e(t) = r(t) - \mathbf{W}_{out}(t-1)^{\top}\mathbf{h}(t) \tag{6}$$

$$\boldsymbol{P}(t) = \frac{1}{\gamma}\left[\boldsymbol{P}(t-1) - \boldsymbol{k}(t)\boldsymbol{h}(t)^{\top}\boldsymbol{P}(t-1)\right] \tag{7}$$

$$\boldsymbol{k}(t) = \frac{\boldsymbol{P}(t-1)\boldsymbol{h}(t)}{\gamma + \boldsymbol{h}(t)^{\top}\boldsymbol{P}(t-1)\boldsymbol{h}(t)} \tag{8}$$

where $\boldsymbol{P}(t) \in \mathbb{R}^{N\times N}$ is the inverse correlation matrix (also called the covariance matrix), $e(t)$ is the instantaneous prediction error and $\boldsymbol{k}(t) \in \mathbb{R}^N$ is the RLS gain vector. $\boldsymbol{P}(t)$ is updated at the same time as the weights according to the Riccati equation for the RLS algorithm. The algorithm also requires an initial value for P, which is taken to be $\boldsymbol{w}(0) = \mathbf{0}$ and $\boldsymbol{P}(0) = \frac{1}{\delta}\boldsymbol{I}$ with $\delta > 0$ a small positive constant for numerical stability. The standard algorithm, as described in [27], compares the network output with the desired setpoint in order to minimize the tracking error. However, in the framework of the present paper, the network output is regarded as the control action applied to the

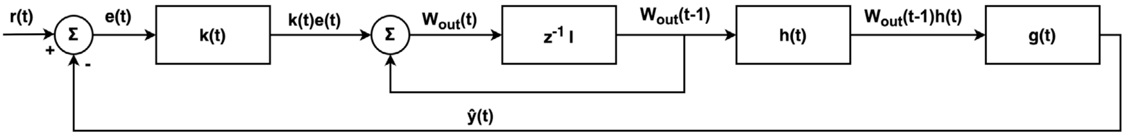

Fig. 2: Representation of the modified version of FORCE-RLS algorithm: Signal Flow Diagram.

controlled system (here a motor), therefore, the feedback signal is not the network output but the motor output, since the goal is to control the motor and not the ESN itself (see Figure 2). The following modified version is proposed:

$$e(t) = r(t) - g(\mathbf{w}_{out}(t-1)^{\top}\mathbf{h}(t)) \tag{9}$$

where $\boldsymbol{g}(y(t))$ is the function that converts the network's control signal into the measured motor output to be regulated.

2.3. Controlled System (plant) description

The controlled system under consideration is represented by an electric motor which can be modeled by a first-order linear model with dead time, a zero-mean white-Gaussian measurement noise with standard deviation $\sigma = 0.002\,\omega_{\max}$ (0.2% of $\omega_{\max}$), and by two elementary nonlinearities that reflect its physical behavior: a minimum command activation threshold and saturation at non-negative speeds. This simplified yet representative model serves as the real-time simulation of the plant during online training, allowing the controller to compute the instantaneous tracking error and update its readout weights via FORCE-RLS under dynamically consistent conditions. In the continuous domain, the motor ideal dynamics is described by the transfer function:

$$G(s) = \frac{K}{1 + s\tau} \cdot e^{-sd} \tag{10}$$

where K is the motor static gain, τ is the mechanical time constant of the motor, and d the total dead time. For the numerical simulation, a discretization with sampling step T_s was adopted according to the zero-order hold scheme; the resulting state-space model is:

$$\begin{aligned} x[k+1] &= a\,x[k] + b\,u_d[k], \\ y[k] &= c\,x[k], \end{aligned} \qquad k \in \mathbb{N}. \tag{11}$$

where $x[k] \in \mathbb{R}$ denotes the state, $u_d[k] = u[k-N]$ is the input delayed by $N = d/T_s$ samples, $y[k] \in \mathbb{R}$ is the measured output,while $a = e^{-T_s/\tau}$, $b = K(1 - e^{-T_s/\tau})$, and $c = 1$ are the system's coefficients obtained by solving the continuous-time first-order model over one sampling interval T_s under the zero-order-hold assumption on the input. Before being applied to the model, the control signal u[k] undergoes two operations. First, a constant u_{min} is subtracted to represent the minimum pulse width modulation amplitude necessary to overcome static friction. Inputs lower than u_{min} are therefore nullified. Next, the output speed is constrained to $y[k] \geq 0$ since the motor only operates in one direction of rotation. Having established the mathematical model of the plant, we now specify the reference trajectories that will excite the closed loop during online training.

2.4. Reference setpoints for Online Training

In online ESN training, the trajectory of the reference setpoint plays the role of an excitation signal for the adaptive read-out. The convergence of the RLS update adopted in FORCE learning is guaranteed only if the regressor sequence, i.e. the reservoir state vector $\mathbf{h}(t)$, is persistently excited. This condition cannot be met when the system is trained with a single or narrowly confined set point pattern [39]. Varying the reference through steps, ramps, sinusoidal and stochastic trajectories forces the closed loop to visit a rich portion of its state–space, thereby exciting all dynamical modes of both the plant and the reservoir [40]. To train and validate the controller, a parametric signal generator module has been developed. The module produces four reference primitives that can be combined into multimodal sequences. Specifically, the following signals are generated: steps, where sequences of constant values are replicated for a number of samples equal to the desired duration; ramps, where a linear increase or decrease between two levels provides an indication of how the system responds dynamically to slow but persistent changes; and sinusoids, where harmonic signals are generated by specifying the amplitude, frequency, and number of cycles.

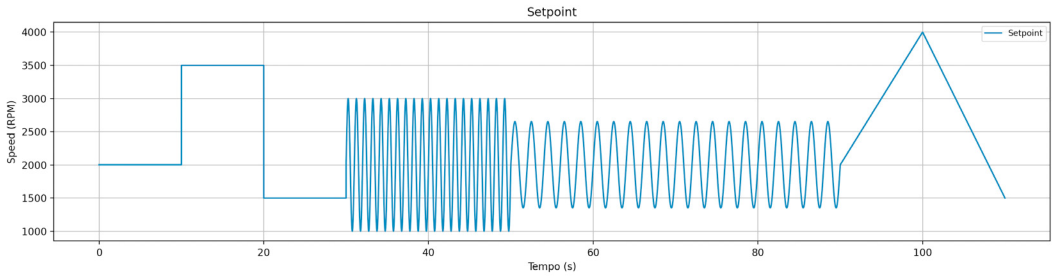

Fig. 3: Training setpoints. A combination of steps, sinusoidal waves, rapid ramp and rapid soak.

As shown in Figure 3, the primitives have been combined in a multimodal sequences. This allows the controller to be stressed with abrupt transitions, slow variations and periodic trends within a single experiment.

2.5. Evaluation Metrics

To comprehensively evaluate the effectiveness of the proposed method and correctly interpret the experimental results, some metrics have been identified, each of which describes a different aspect of the controller performance: the Mean Absolute Error (MAE), the Root Mean Square Error (RMSE) and Memory cost that quantifies the RAM occupied by the algorithm in order to evaluate the application on microcontrollers or edge devices with limited memory. These metrics enabled the correct analysis of the trade-offs between accuracy, stability, and computational requirements, based on application constraints.

3. Results and Discussions

The ESN has been trained online with the FORCE-RLS algorithm so that it continuously adapts and produces a control signal that minimises the error between the set-point and the process output. The ESN employed in this study receives a two-dimensional input vector that contains the reference signal and the tracking error. Its reservoir comprises one hundred recurrent units arranged in four stacked sub-reservoirs, which enables the network to capture richer temporal relationships through successive nonlinear transformations. A leakage rate of 0.5 updates the internal states at an intermediate pace, striking a balance between memory retention and responsiveness to new information. The spectral radius is fixed at 0.9 so that the echo-state property is preserved while the reservoir remains dynamically expressive. The readout weights are updated online with a modified version of the FORCE-RLS algorithm. The experimental protocol consists of two phases: an initial training stage followed by a separate validation stage.

3.1. Training protocol

During training, the network is driven by a composite reference signal that combines step changes, linear ramps, and sinusoidal segments. This pattern is repeated for five epochs to progressively refine the read-out weights via the FORCE-RLS rule. Figure 4 depicts the training reference trajectory together with the network responses recorded at each epoch, illustrating the gradual improvement achieved over the five steps. In the figure, the time is reported in

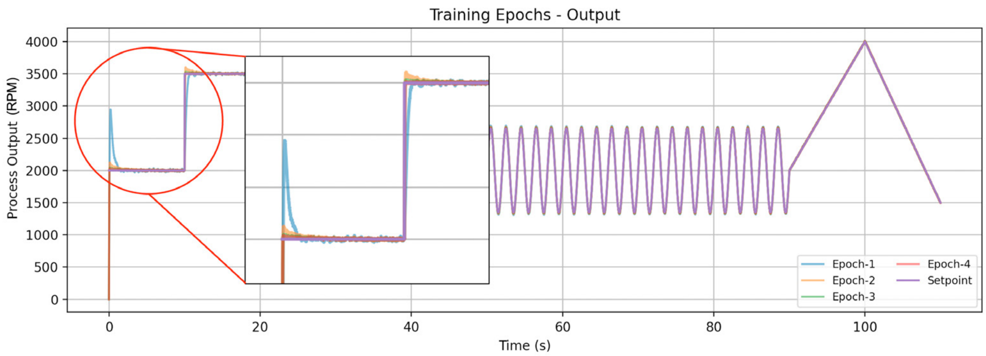

Fig. 4: Closed-loop motor-speed responses during successive training epochs.

seconds on the abscissa, whereas the ordinate gives the motor speed. The purple trace is the reference profile, and the blue, orange, green, and red traces are the motor speed outputs controlled by ESN after the first, second, third, and fourth epochs, respectively. At the initial step change, the first epoch response overshoots the target and then decays, while the following epochs progressively suppress the overshoot and shorten the settling interval. The magnified view emphasises that by the third step the response is already critically damped and aligns with the reference almost instantaneously. A second step, highlighted in the black window, confirms the same trend: the early epochs exhibit a small transient deviation that vanishes after further adaptation, demonstrating that the FORCE-RLS update drives the closed-loop pole toward the location predicted by the linear model. Taken together, the trajectories reveal a monotonic reduction of tracking error from the first to the fourth epoch and verify that the online FORCE-RLS procedure yields a stable and increasingly accurate controller capable of following steps, sinusoids, and ramps with uniform precision.

3.2. Validation protocol

To obtain a more comprehensive assessment of the online-trained controller, a validation experiment that records motor speed while the ESN tracks four distinct reference profiles has been devised. The sequence begins with a

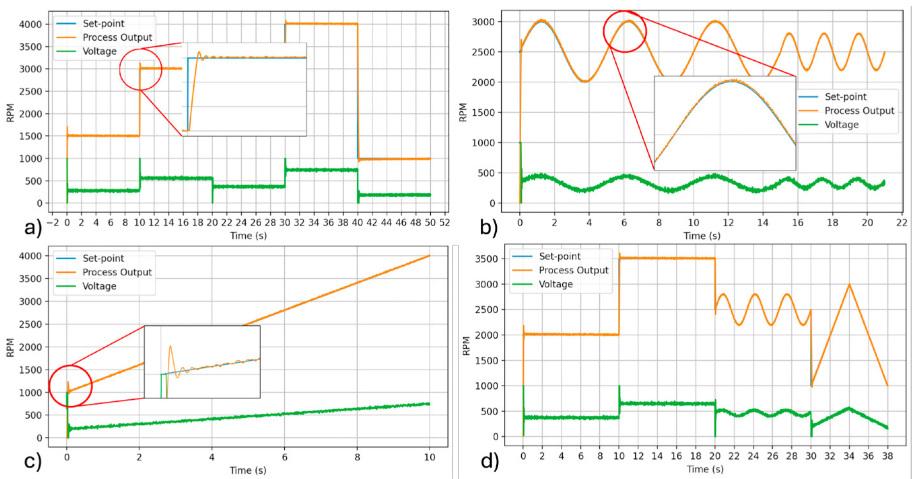

Fig. 5: Performance of the ESN-based control system across four different temperature set-point profiles a) Steps, b) Sinusoids, c) Ramp and d) multimodal sequences. The Voltage values are scaled from a range of 0-10 V to 0-1000 V only for visualization purpose.

50 seconds segment containing four step changes, continues with a 22 seconds sinusoidal segment at two different frequencies, is followed by a 10 seconds rapid ramp, and concludes with a 38 seconds composite segment that blends

steps, fast ramps and sinusoidal portions. The resulting traces, collected in Figure 5, show that the ESN consistently drives the motor to the commanded speeds and maintains stable operation throughout. Subplots (a)–(d) in the Figure correspond to the four reference segments and highlight the controller's ability to accommodate abrupt transitions, periodic inputs, steadily varying commands and mixed patterns with comparable accuracy. In every subplot, the blue trace shows the reference set-point, the orange trace the measured process output, and the green trace the control effort. The subplot a) presents a step reference that spans the whole operating range. At each step the controller applies a brief voltage impulse that swiftly brings the speed toward the new target, after which the voltage settles to a steady value. The motor reaches every set-point in a short time with virtually no overshoot, confirming the closed-loop dynamics predicted by the first-order model. The subplot b) tests tracking of a smooth sinusoidal reference at two different frequencies. The ESN follows the amplitude with a mean error below three percent up to about one hertz. As the frequency rises a gradual phase lag appears, in line with both the reservoir bandwidth and the intrinsic low-pass nature of the motor. The subplot c) drives the speed along a ten-second ramp from idle to nominal value. The slope of the controlled speed coincides with the reference for the entire interval, indicating that the RLS term provides an effective integral-like action able to cancel static gain errors. Finally, the subplot d) combines all the experiments of previous subplots creating a more complex scenario. The unexpected variation generates a damped oscillation; nevertheless, the error remains bounded and the speed returns to its target within four seconds. The read-out continues to adjust, visible in the corrective oscillations of the voltage trace, and quickly restores the original performance without destabilising the reservoir. As highlighted in Figure 6, across all reference profiles the tracking

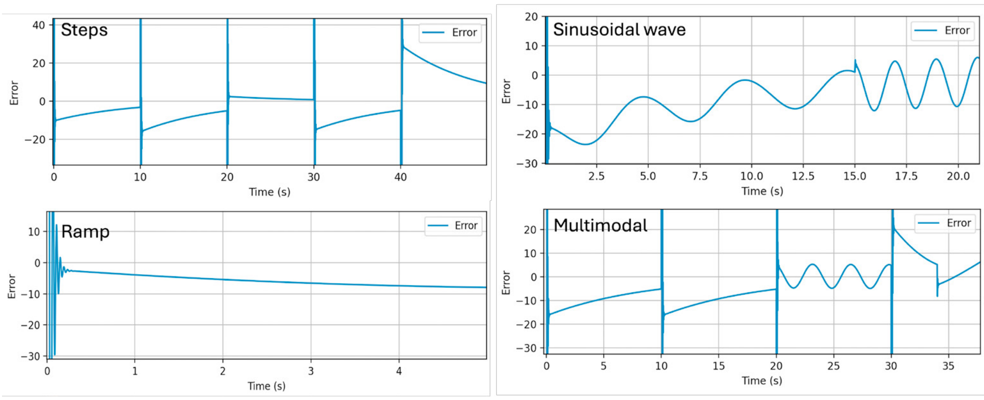

Fig. 6: Tracking error during the four validation scenarios: steps (top-left), sinusoid (top-right), ramp (bottom-left) and multimodal (bottom-right).

error remains within 40 rpm, which corresponds to 1% of the 4000 rpm operating range. Apart from the brief spikes that occur exactly at each set-point change, the error quickly settles to values below 0.5% of full scale, confirming the controller's precise steady-state regulation.

3.3. Performance

The values collected in Tab. 1 quantitatively confirm the observations already evident in the experimental traces. The mean absolute error is modest for all the examined profiles, with more pronounced peaks in large step varia-

Table 1: Evalution Metrics.

Signal	MAE	RMSE
Steps	8.6181	65.4680
Sinusoids	3.2154	28.8986
Ramp	3.8837	16.8793
Multimodal sequence	1.6050	21.6715

tions and troughs in multimodal sequences. This demonstrates the controller's ability to adapt to sudden changes and complex combinations of stimuli. The mean square error follows the same trend, indicating that even the largest deviations are contained and do not affect the stability of the system. The test results show that the entire control cycle, including state updating, FORCE-RLS adaptation and actuation, takes place in a few microseconds on a mid-range microcontroller. In terms of memory usage, the required space can be estimated by considering the dimensionality of the main elements involved. In particular, the overall requirement is dominated by the recurrent reservoir matrix which contains 10.000 coefficients represented in 32-bit floating point and occupies approximately 40 kB, as well as the covariance matrix of the RLS filter $P = 100 \times 100$, which is also made up of 10.000 elements and occupies a similar amount of space, around 40 kB. The remaining components the input matrix $W_{in} = 100 \times 2$ (approximately 0.8 kB), the output matrix $W_{out} = 100 \times 1$ (approximately 0.4kB), and the internal state vector $x = 100$ (approximately 0.4 kB) provide negligible contributions compared to the two main blocks. Therefore, during the online learning phase, the overall memory usage is approximately 82 kB, which falls well within the typical SRAM memory capacity of the mid-range microcontrollers used for the experiments. The ESN controller trained online with FORCE-RLS offers a fast, adaptive, and effective alternative to conventional speed regulators across diverse reference profiles.

4. Conclusion and future works

The above results show that a reservoir computer whose read-out is adapted online with a modified FORCE-RLS rule can regulate motor speed with the same accuracy and effectiveness ordinarily achieved by conventional controllers, yet without relying on a detailed plant model, a offline dataset and an external supervisory loop. The fixed recurrent core supplies a rich nonlinear basis, while the RLS update confines numerical adaptation to a low-dimensional parameter vector, preserving both computational efficiency and echo-state stability. Training on a mixed reference sequence of steps, ramps, and sinusoids over five epochs yields a monotonic reduction of tracking error, and the subsequent validation confirms uniform performance across abrupt, periodic, and steadily varying commands. These results indicate that the proposed ESN architecture offers a practical data-driven alternative for processes whose dynamics are uncertain or slowly time-varying, extending the operational scope of reservoir computing from off-line identification to real-time closed-loop control. Although the present results are encouraging, a more detailed examination of the mathematical structure of the ESN and the FORCE-RLS are required to guarantee long-term stability, to quantify closed-loop performance, and to assess computational overhead. Future work should develop rigorous proofs of stability and convergence for the online read-out adaptation, extend the analysis to a wider range of real-world physical systems and operating conditions, and establish bounds that relate reservoir size to tracking accuracy and numerical cost.

Acknowledgements

This work has been supported by the Italian PON ’RICERCA E INNOVAZIONE’ 2014-2020 - AZIONE IV.6 ’CONTRATTI DI RICERCA SU TEMATICHE GREEN’.

This work was also partially supported by European Project Digital Europe “EDIH4Marche” (European Digital Innovation Hub for Marche), Call DIGITAL-2021-EDIH-01, Grant Agreement No. 101084027 on topic DIGITAL-2021-EDIH-INITIAL-01 (DIGITAL Simple Grants Acton)

References

[1] Z. Wu, A. Tran, D. Rincon, P. D. Christofides, “Machine learning-based predictive control of nonlinear processes. part i: Theory,” *AIChE Journal*, vol. 65, no. 11, 2019.

[2] S. Pozzoli, M. Gallieri, R. Scattolini, “Tustin neural networks: a class of recurrent nets for adaptive MPC of mechanical systems,” *IFACPapersOnLine*, vol. 53, no. 2, pp. 5171–5176, 2020.

[3] H. He, J. Gray, A. Cangelosi, Q. Meng, T. M. McGinnity and J. Mehnen, “The Challenges and Opportunities of Human-Centered AI for Trustworthy Robots and Autonomous Systems,” *IEEE Transactions on Cognitive and Developmental Systems*, vol. 14, pp. 1398-1412, 2020.

[4] A. Bonci, F. Gaudeni, M. C. Giannini, and S. Longhi, “Robot Operating System 2 (ROS2)-Based Frameworks for Increasing Robot Autonomy: A Survey”, *Applied Sciences*, 2020.

[5] A. K. Raz, E. P. Blasch, C. Guariniello and Z. T. “Mian, An overview of systems engineering challenges for designing AI-enabled aerospace systems,” *AIAA Scitech 2021 Forum*, pp. 0564, 2020.

[6] A. Zaoui, D. Tchuente, S. F. Wamba and B. Kamsu-Foguem, "Impact of artificial intelligence on aeronautics: An industry-wide review," *Journal of Engineering and Technology Management*, vol. 71, 2024.
[7] A. Bonci, R. De Amicis, S. Longhi, E. Lorenzoni and G. A. Scala, "Motorcycle's lateral stability issues: Comparison of methods for dynamic modelling of roll angle," *In Proceedings of the 20th International Conference on System Theory, Control and Computing*, pp. 607-612, 2016.
[8] A. Bonci, S. Longhi and G. A. Scala, "Towards an All-Wheel Drive Motorcycle: Dynamic Modeling and Simulation," *IEEE Access*, pp. 112867-112882, 2020.
[9] P. Yan, D. Liu, D. Wang, H. Ma, "Data-driven controller design for general mimo nonlinear systems via virtual reference feedback tuning and neural networks," *Neurocomputing*, vol. 171, pp. 815–825, 2016.
[10] W. Sun, N. Akashi, Y. Kuniyoshi, and K. Nakajima, "Physics-informed recurrent neural networks for soft pneumatic actuators," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6862–6869, 2022.
[11] Z. Jiang, Y. Li, and Q. Wang, "Intelligent feedforward hysteresis compensation and tracking control of dielectric electro-active polymer actuator," *Sensors and Actuators A: Physical*, vol. 341, pp. 113581, 2022.
[12] K. Zarzycki, and M. Ławryńczuk, "LSTM and GRU neural networks as models of dynamical processes used in predictive control: a comparison of models developed for two chemical reactors," *Journal of Sensors*, vol. 21, no. 16, pp. 5625, 2021.
[13] Bonci, A.; Di Biase A.; Giannini, M. C.; and Longhi, S. Yaw rate-based PID control for lateral dynamics of autonomous vehicles, design and implementation. *IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA).* **2023**, pp 1-8.
[14] J. Wu, J. Huang, Y. Wang, and K. Xing, "RLS-ESN based PID control for rehabilitation robotic arms driven by PM-TS actuators," *In Proceedings of the 2010 International Conference on Modelling, Identification and Control*, pp. 511–516, 2010.
[15] H. Jaeger, "The 'echo state' approach to analysing and training recurrent neural networks," *GMD Technical Report*, vol. 148, no. 34, 2001.
[16] H. Jaeger, "Adaptive nonlinear system identification with echo state networks," *Advances in Neural Information Processing Systems*, 2002.
[17] H. Jaeger, "Short term memory in echo state networks," *GMD Forschungszentrum Informationstechnik - Technical Report*, 2001.
[18] A. Bonci, L. Longarini, S. Longhi, G. Pompei, and M. Prist, "Process regulation control using Echo State Networks: an ESN-based deep neural network approach for PID control," *Procedia Computer Science*, vol. 253, pp. 2369–2376, 2025.
[19] M. Salmen, and P. Plöger, "Echo State Networks used for Motor Control, *Conference on Robotics and Automation*, vol. 18, 1953-1958, 2005.
[20] W. D'Amico, M. Farina and G. Panzani, "Recurrent Neural Network controllers learned using Virtual Reference Feedback Tuning with application to an Electronic Throttle Body," *European Control Conference (ECC)*, pp. 2137-2142, 2022.
[21] J. Huang, J. Qian, L. Liu, Y. Wang, C. Xiong, and S. Ri, "Echo state network based predictive control with particle swarm optimization for pneumatic muscle actuator," *Journal of the Franklin Institute*, vol. 353, no. 12, pp. 2761–2782, 2016.
[22] M. Wang, Z. Chen, K. Guo, X. Yu, Y. Zhang, L. Guo, and W. Wang, "Millimeter-level pick and peg-in-hole task achieved by aerial manipulator," *IEEE Transactions on Robotics*, vol. 40, pp. 1242–1260, 2023.
[23] Y. C. Bo, and X. Zhang, "Online adaptive dynamic programming based on echo state networks for dissolved oxygen control," *Applied Soft Computing*, vol. 62, pp. 830–839, 2018.
[24] A: Bonci, L. Fredianelli, R. Kermenov, L. Longarini, S. Longhi, G. Pompei, M. Prist, and C. Verdini, "DeepESN neural networks for industrial predictive maintenance through anomaly detection from production energy data," *Applied Sciences*, vol. 14, 2024.
[25] S. Wen, R. Hu, Y. Yang, T. Huang, Z. Zeng and Y. -D. Song, "Memristor-Based Echo State Network With Online Least Mean Square," *in IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 9, pp. 1787-1796, 2019.
[26] X. Yao, Z. Wang and H. Zhang, "Prediction and identification of discrete-time dynamic nonlinear systems based on adaptive echo state network," *Neural Networks*, vol. 113, pp. 11-19, 2019
[27] D. Sussillo and L. F. Abbott, "Generating coherent patterns of activity from chaotic neural networks," *Journal of Neuron*, pp. 544–57, 2009.
[28] H. Tamura, and G. Tanaka, "Transfer-RLS method and transfer-FORCE learning for simple and fast training of reservoir computing models," *Official journal of the International Neural Network Society*, vol. 143, pp. 550–563, 2021.
[29] Y. Li, K. Hu, K. Nakajima and Y. Pan, "Composite FORCE Learning of Chaotic Echo State Networks for Time-Series Prediction," *41st Chinese Control Conference (CCC)*, pp. 7355-7360, 2022.
[30] J. Wu, Y. Wang, J. Huang, and H. Zhou, "Nonlinear Internal Model Control Using Echo State Network for Pneumatic Muscle System," *Journal of Comput.*, vol. 7, no. 12, pp. 3060–3067, 2012.
[31] H. Ogawa, and Y. Takahashi, "Echo state network based model predictive control for active vibration control of hybrid electric vehicle powertrains," *Applied Sciences*, vol. 11, no. 14, pp. 6621, 2021.
[32] A. Polydoros, L. Nalpantidis, and V. Krüger, "Advantages and limitations of reservoir computing on model learning for robot control," *In IROS Workshop on Machine Learning in Planning and Control of Robot Motion*, 2015.
[33] H. Jaeger, and H. Haas, "Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication," *Journal of Science*, vol. 304, pp. 78-80, 2004.
[34] C. Gallicchio, and A. Micheli, "Echo state property of deep reservoir computing networks," *Journal of Cognitive Computation*, vol.9, 2017.
[35] I.B., Yildiz, H. Jaeger, and S. Kiebel, "Re-visiting the echo state property," *Journal of Neural Networks*, vol.35, pp. 1–9, 2012.
[36] M. Lukoševičius, "A Practical Guide to Applying Echo State Networks," *in Neural Networks Book*, pp. 659-686, 2012.
[37] H. Jaeger, "Adaptive nonlinear system identification with echo state networks," *in Proc. Int. Conf. Neural Inf. Process. Syst*, pp. 609–616, 2002.
[38] A. G. Hart, J. L. Hook and J. H. P. Dawes, "Echo State Networks trained by Tikhonov least squares are L2(μ) approximators of ergodic dynamical systems," *Physica D: Nonlinear Phenomena*, vol. 421, pp. 132882, 2021.
[39] A. Simpkins, "System Identification: Theory for the User, 2nd Edition (Ljung, L.; 1999) [On the Shelf]," in IEEE Robotics & Automation Magazine, vol. 19, no. 2, pp. 95-96, 1999.
[40] D. Sussillo, and L. F. Abbott, "Transferring learning from external to internal weights in esn with sparse connectivity." *PLoS One*, 2012.