sensors

MDPI

Article

Neural Network Direct Control with Online Learning for Shape Memory Alloy Manipulators

Alfonso Gómez-Espinosa [1,*], Roberto Castro Sundin [2], Ion Loidi Eguren [3], Enrique Cuan-Urquizo [1,*] and Cecilia D. Treviño-Quintanilla [1]

1 Tecnologico de Monterrey, Escuela de Ingeniería y Ciencias, Ave. Epigmenio González 500, Fracc. San Pablo, Querétaro 76130, Mexico; cdtrevino@tec.mx
2 KTH Royal Insitute of Technology, 114 28 Stockholm, Sweden; rosun@kth.se
3 Escuela Politécnica Superior, Universidad Mondragón, 20500 País Vasco, Spain; ion.loidi@alumni.mondragon.edu
* Correspondence: agomeze@tec.mx (A.G.-E.); ecuanurqui@tec.mx (E.C.-U.); Tel.: +52-442-238-3302 (A.G.-E.)

Received: 12 April 2019; Accepted: 4 June 2019; Published: 6 June 2019

Abstract: New actuators and materials are constantly incorporated into industrial processes, and additional challenges are posed by their complex behavior. Nonlinear hysteresis is commonly found in shape memory alloys, and the inclusion of a suitable hysteresis model in the control system allows the controller to achieve a better performance, although a major drawback is that each system responds in a unique way. In this work, a neural network direct control, with online learning, is developed for position control of shape memory alloy manipulators. Neural network weight coefficients are updated online by using the actuator position data while the controller is applied to the system, without previous training of the neural network weights, nor the inclusion of a hysteresis model. A real-time, low computational cost control system was implemented; experimental evaluation was performed on a 1-DOF manipulator system actuated by a shape memory alloy wire. Test results verified the effectiveness of the proposed control scheme to control the system angular position, compensating for the hysteretic behavior of the shape memory alloy actuator. Using a learning algorithm with a sine wave as reference signal, a maximum static error of 0.83° was achieved when validated against several set-points within the possible range.

Keywords: shape memory alloys; artificial neural networks; control; manipulators

1. Introduction

Nonlinear systems have been an active research area over the past few decades, partly fueled by the needs of modern industry. As new actuators and materials are incorporated into industrial processes, additional challenges can be posed by their sometimes complex behavior. An example of this is the nonlinear hysteresis commonly found in shape memory alloys (SMAs) [1], certain nanocomposite materials [2], and micro-electromechanichal systems (MEMS) [3].

In applications where hysteresis behavior is present, the inclusion of a suitable hysteresis model in the control system design allows the controller to have better tracking of the system and may result in a reduced error response of the controlled variable [4–6]. The methods that have been used to model these behaviors are many, and include artificial neural networks (ANNs) [4], elliptical approximations of hysteresis loops [5], the Preisach model [6], ideal order hexagonal arrays adjusted using the Monte Carlo method [7], a modified Prandtl–Ishlinskii Model [8], and the Semilinear Duhem Model [9].

Although the use of a suitable model allows a better control of a hysteretic system, a major drawback is that each system responds in a unique way, meaning that the controllers characteristics, such as effectiveness, speed, number of cycles, etc., may vary from one system to another, even if

the control techniques are the same [10,11]. As an example, second-order models have been used to analyse and model output signal sensors. However, this process is time-consuming, and when completed, the resulting algorithm can only be applied successfully to the system it was specifically designed for [10]. Another previously used method for approximating and tracking the performance of a system, is to make use of ANNs. This has been done, for instance, for an inverted pendulum with an actuator displaying hysteretic behavior [11], and for an active magnetic bearing system [12]. Although ANNs are still under development in order to increase their accuracy in control system applications, they are frequently implemented due to their low computational cost [13].

Neural networks (NNs) have demonstrated their ability to identify, and compensate for, hysteresis in systems in which high precision is mandatory [14–17]; an adaptive wavelet NN controller with estimation of the friction force hysteresis was proposed for a piezo positioning mechanism [14], an ANN to identify, and compensate for, gear backlash hysteresis was tested for a precision position mechanism [15], a radial basis function neural network and a sliding control scheme was proposed for motion tracking control of piezoelectric actuators [16], and a dynamic recurrent NN with a proportional-derivative (PD) controller was implemented to identify and control a magneto restrictive actuator [17].

SMAs are recognized as promising smart materials for creating compact, high power-to-weight ratio actuators. However, position control of SMA actuators represents a major challenge for practical applications due to their nonlinear hysteretic behavior, producing steady-state errors when conventional controllers are used [18]. To overcome this problem, proportional-integral-derivate (PID) controllers with hysteresis models have been implemented [18–20]; for position control of SMA actuators using the generalized Prandtl–Ishlinskii inverse model [18], for micro-positioning control of SMA actuators by modeling the hysteresis using NNs [19], and for magnetic SMA actuators by using a radial basis function NN to obtain the Jacobian information of the system in order to adjust the controller parameters [20]. In addition, neural network controllers, previously trained to identify the system were proposed to control SMA actuators [21–24]; using the inverse of the ANN that replicate the dynamics of the SMA force actuator to implement the controller [21], implementing a model predictive controller based on a functional link ANN to control the linear memory metal actuator displacement [22], and realizing a recurrent neural model predictive, variable structure, controller designed to control a one degree of freedom (1-DOF) rotary manipulator actuated by an SMA wire [23,24]. The main disadvantages of these recurrent neural network controllers are the complexity of the control system, and the requirement of training the neural network, to identify the system, prior to the implementation of the control. During the identification of the system, a neural network captures the model of the plant. Later this plant model is utilized for the design of the controller. The control signal is only delivered to the plant, after these two stages are completed. Previous inputs/outputs data are gathered from the operation of the plant, and are utilized to train the neural network model, offline in batch mode [23]. Previous information from the controller outputs and from the plant outputs are the inputs utilized for the neural plant model, implementing a recurrent neural network structure [24].

In this work, a neural network direct control, with online learning, is developed for position control of shape memory alloy manipulators. The developed network consists of three inputs, four hidden layer perceptrons and one output, all using a sigmoidal activation function. In contrast to previous works, weight coefficients are updated online by using the actuator position data while the controller is applied to the system, without previous training of the neural network weights nor the inclusion of a hysteresis model. A real-time, low computational cost control system was implemented. Experimental evaluation was performed on a 1-DOF manipulator system actuated by a shape memory alloy wire and test results verify the effectiveness of the proposed control scheme to control the system angular position, compensating for the hysteretic behavior of the shape memory alloy actuator.

The remainder of the paper is organized as follows: Hysteresis nonlinearity is described in Section 2.1. Shape memory alloy nonlinear behavior is introduced in Section 2.2. Section 2.3 presents the

NN direct controller with online learning using back-propagation. Section 3 describes the experimental setup and the results are presented in Section 4. Finally, in Section 5, concluding remarks are provided.

2. Materials and Methods

2.1. Hysteresis

Hysteresis is a strongly nonlinear phenomena, where *strongly* indicates that it cannot be linearized. The non-ability of being linearized is a consequence of the *memory effect* in hysteretic systems, which means that the output of the system is dependent not only on the current input, but also on the previous state of the system [6,9]. Hysteresis can be found in many different areas, including structural mechanics, aerodynamics, and electromagnetics [9]. Because of the memory effect, an input–output mapping for a system with hysteresis seldom is *injective*, since one input often may result in two distinct outputs depending on the system history (illustrated in Figure 1). This of course implicates that hysteresis cannot be modeled in the sense of an ordinary mathematical function, but requires a more sophisticated framework. One of these frameworks was proposed by Preisach as a means to model the hysteresis found in magnets, and was further improved upon by mathematician Krasnoselskii to become what is now referred to as the Preisach model [25].

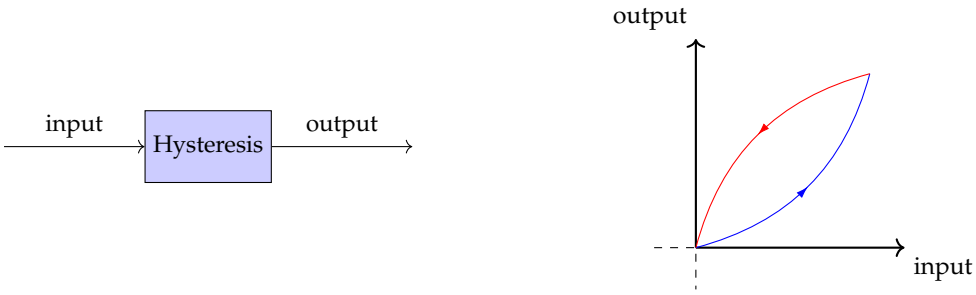


Figure 1. Typical input-output relation for a system with hysteresis.

The Classic Preisach Model

The model is most often represented by the *Preisach operator* $\hat{\Gamma}$, which can be expressed as a double integral of variables α, β

$$\hat{\Gamma} = \iint_{\alpha \geq \beta} \mu(\alpha, \beta) \hat{\gamma}_{\alpha\beta} \, \mathrm{d}\alpha \, \mathrm{d}\beta \,, \tag{1}$$

where $\mu(\alpha, \beta)$ is a weight function called the *Preisach function*, and $\hat{\gamma}_{\alpha\beta}$ is a fundamental hysteresis operator, often called *hysteron*. The hysteron is a mapping onto $\{-1,1\}$ with memory properties as seen in Figure 2 [26]. Using the Preisach operator, systems with hysteresis, like the one in Figure 1, can now be modeled by finding the correct weight function $\mu(\alpha, \beta)$.

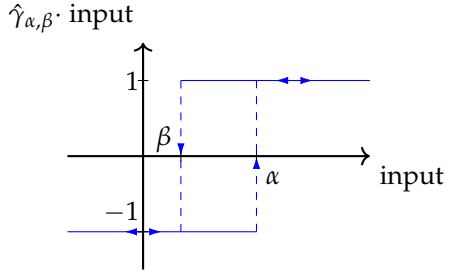


Figure 2. Definition of the fundamental hysteresis operator $\hat{\gamma}_{\alpha\beta}$. When the input is in the range (α, β), the output is positive if the input reached this range from above and negative if it was reached from below.

2.2. Shape Memory Alloys

Shape memory alloys, or SMAs, receive their name from their property of exerting a force to return to a certain *memorized* shape when heated, giving them the ability to convert heat into mechanical energy [23]. This effect is due to SMAs having two solid phases; a high temperature phase called *austenite* and a lower temperature phase called *martensite*, differentating themselves by having different crystal structures [24]. The austenite form is typically cubic and rigid, while the martensite usually is tetragonal, orthorhombic or monoclinic, giving it the possibility to change shape (e.g., stretch) upon being subjected to a force. Because of this, it is common to distinguish between the unstretched *twinned martensite* form and the stretched *untwinned* one. The relationship between the different phases can be seen in the stress–temperature-plane in Figure 3, for which it is also important to mention that the stress is sufficiently small to allow austenite form even under stress conditions when temperature is increased (seen in case (d)). For higher stress levels the austenite form could not be achieved even when increasing the temperature, but would rather stay in its detwinned martensite form (not shown in this figure) [27].

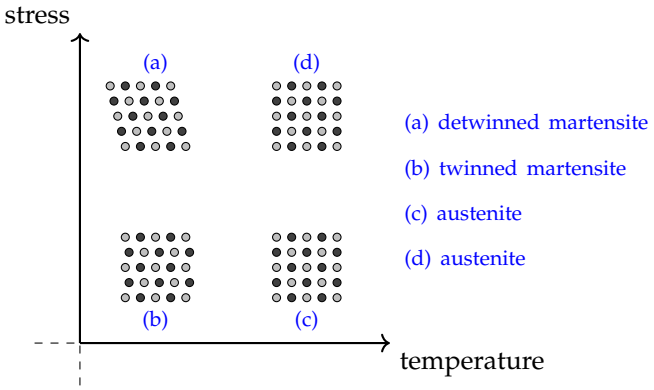


Figure 3. Stress–temperature plane showing schematic crystal structures of the different phases of a typical SMA .

The transition between the untwinned martensite form and the austenite is highly hysteretic, as shown by the experimental results for a Flexinol SMA wire in Figure 4. The wire was 310 mm long, ⌀ 0.13 mm and had a 50 g weight suspended to it.

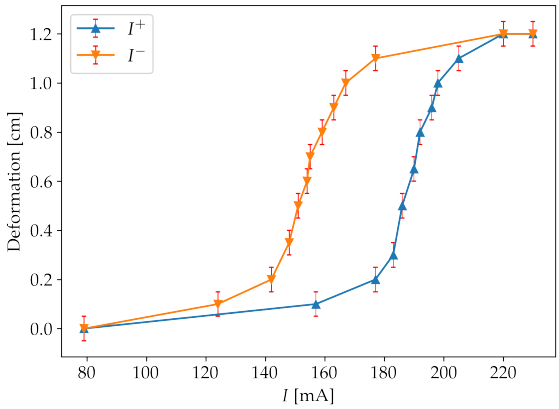


Figure 4. Plot showing deformation versus current for a Flexinol wire. I^+ denotes that the current is monotonically increasing, while I^- denotes a monotonical decrease.

2.3. Neural Networks

2.3.1. Feedforward Neural Networks with Sliding Window

In accordance with [28], a multilayer recurrent neural network is an extension of the classic perceptron that allows for more possibilities when it comes to the desired behavior, being able to approximate dynamic systems. The increased functionality comes from incorporating a number of so-called *hidden layers* in between the input and output layers, from a learning process called *back-propagation* and from making use of previous input/output values (i.e., recurrent neural network). Recurrent neural networks can approximate dynamic systems, represented by differential equations, however, feed forward neural networks can only approximate algebraic functions. Conventional recurrent neural networks incorporate temporal information in the hidden layer, by using previous output values as inputs for this same layer. On the other hand, feedforward neural networks with sliding windows can be implemented using previous input/output values as inputs for the input layer.

A hidden layer consists of a certain amount of perceptrons $h_1, \ldots, h_m$, each one connected to all of the perceptrons (or inputs) $h'_1, \ldots, h'_n$ in the preceding layer. Each of these connections are weighted by the coefficients w'_{ji}, where subscripts j, i indicate the connection between perceptrons/inputs h_j and h'_i. A general illustration is shown in Figure 5, where the input layer uses present values as well as previous values z_1, z_2, z_3 for a feedforward neural network structure with a sliding window.

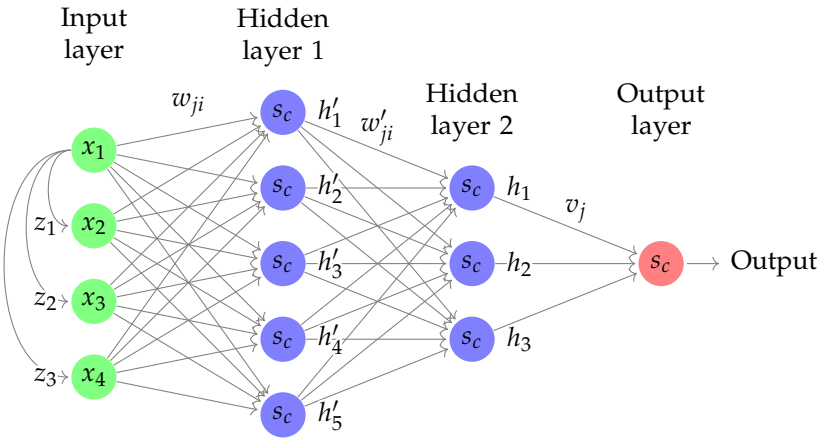


Figure 5. Schematic of a neural network with two hidden layers and a four element sliding window, where s_c denotes the sigmoid activation function and w_{ji}, w'_{ji}, v_j are weight coefficients.

For a neural network with one hidden layer and a single output, the inputs can be defined as x_i, the hidden layer perceptrons as h_j and the weights as follows

w_{ji} the weight of the connection between input x_i and hidden layer perceptron h_j
v_j the weight of the connection between hidden layer perceptron h_j and output perceptron.
it has been regarded as items, please confirm

2.3.2. Back-Propagation

Following the methodology of [28,29], the back-propagation algorithm can be used to train a multilayer neural network into reaching a desired behavior. This is done by iteratively changing the weights until the error is reduced to an acceptable level. A solution is thus any collection of weights w_{ji}, v_j that are able to achieve this. The back-propagation algorithm finds these weights by making use of the method of *gradient descent*. Using gradient descent the aim is to minimize the error function

$$E = \frac{1}{2}\sum_i |e_i|^2 = \frac{1}{2}\sum_i |y_{ri} - y_i|^2. \quad (2)$$

The gradient ∇E is only defined if E is continuous and differentiable, and thereby, the sigmoid $s_c : \mathbb{R} \to (0,1)$ given by

$$s_c(x) = \frac{1}{1+e^{-cx}}, \tag{3}$$

will be used as activation function, where c can be chosen arbitrarily and decides the steepness of the curve.

As a means of increasing the flexibility of the activation function, a so-called *bias*, θ, can be used. The role of the bias is to shift the activation function $s_c(x)$ in the positive or negative direction of the x-axis. A simple way of adding the bias is to expand the input vector $(x_1, \ldots, x_n)$ with 1, to get $(x_1, \ldots, x_n, 1)$ and the weight vector $\mathbf{w} = (w_1, \ldots, w_n)$ with $-\theta$, to get $w = (w_1, \ldots, w_n, -\theta)$. This permits the comfortable notation of a bias-shifted perceptron as

$$s_c\left(\sum_i w_i \cdot x_i - \theta\right) = \frac{1}{1+\exp\left[-c(\sum_i w_i \cdot x_i - \theta)\right]} = s_c(\mathbf{w} \cdot \mathbf{x}). \tag{4}$$

2.3.3. Calculation of the Gradient

For a neural network with one hidden layer and a single output, the gradient can be calculated in a straightforward manner using chain rule and the configuration of the control system, as seen in Figure 6 [28].

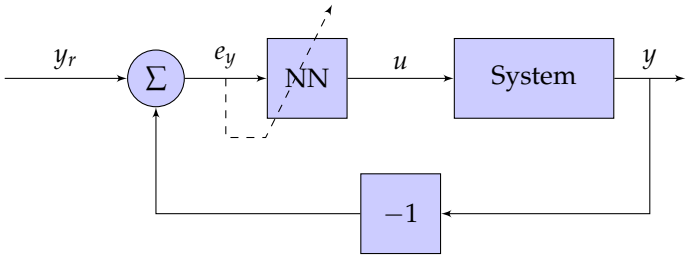


Figure 6. A block scheme of the control system.

Since the weights w_{ji}, v_j are the ones to be adjusted, the gradient is calculated with respect to these. Therefore, the gradient is defined

$$\nabla E = \begin{bmatrix} \frac{\partial E}{\partial v_j} \\ \frac{\partial E}{\partial w_{ji}} \end{bmatrix}, \tag{5}$$

with its partial derivatives

$$\frac{\partial E}{\partial v_j} = \frac{\partial E}{\partial e_y}\frac{\partial e_y}{\partial e_u}\frac{\partial e_u}{\partial u}\frac{\partial u}{\partial r}\frac{\partial r}{\partial v_j} = -e_y u(1-u)\frac{\partial e_y}{\partial e_u}h_j \tag{6}$$

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial e_y}\frac{\partial e_y}{\partial e_u}\frac{\partial e_u}{\partial u}\frac{\partial u}{\partial r}\frac{\partial r}{\partial h_j}\frac{\partial h_j}{\partial S_j}\frac{\partial S_j}{\partial w_{ji}} = -e_y u(1-u)v_j h_j(1-h_j)\frac{\partial e_y}{\partial e_u}x_i, \tag{7}$$

where the relations

$$r = \sum_j v_j h_j \tag{8}$$

$$S_j = \sum_i w_{ji} x_i, \tag{9}$$

along with the definitions seen in Figures 5 and 6 have been used, and where e_u denotes the error between the current control signal and the control signal required to control the system. Note that, with the lack of a mathematical model of the system, the error e_y cannot be expressed in analytical terms. Consequently, its partial derivative $\partial e_y / \partial e_u$ found in Equations (6) and (7), is unknown.

With the gradient explicitly calculated, the weights can now be adjusted iteratively as follows

$$v_j^{(t+1)} = v_j^{(t)} - \eta \frac{\partial E}{\partial v_j} = v_j^{(t)} + \eta h_j \delta^1 \frac{\partial e_y}{\partial e_u} \tag{10}$$

$$w_{ji}^{(t+1)} = w_{ji}^{(t)} - \eta \frac{\partial E}{\partial w_{ji}} = w_{ji}^{(t)} + \eta x_i \delta_j^2 \frac{\partial e_y}{\partial e_u}, \tag{11}$$

where η represents the learning rate and $\delta^1 = e_y u(1-u)$, $\delta_j^2 = \delta^1 v_j h_j (1-h_j)$.

Considering that

$$\frac{\partial e_y}{\partial e_u} = \operatorname{sgn}\left(\frac{\partial e_y}{\partial e_u}\right) \cdot \left|\frac{\partial e_y}{\partial e_u}\right| \tag{12}$$

and letting $\eta \cdot \mid \partial e_y / \partial e_u \mid \rightarrow \eta$, Equations (10) and (11) simplify to

$$v_j^{(t+1)} = v_j^{(t)} + \operatorname{sgn}\left(\frac{\partial e_y}{\partial e_u}\right) \eta h_j \delta^1 \tag{13}$$

$$w_{ji}^{(t+1)} = w_{ji}^{(t)} + \operatorname{sgn}\left(\frac{\partial e_y}{\partial e_u}\right) \eta x_i \delta_j^2, \tag{14}$$

where $\operatorname{sgn}(\partial e_y / \partial e_u)$, in accordance with [29], can be found experimentally for the system.

The previously described neural networks suffice when we seek to approximate static functions. However, when the aim is to approximate a dynamic function, it is necessary to include the concept of time. This can be done by incorporating previous input/output values. For example, if $u(t)$ is the output at a time t, then an arbitrarily large collection of previous output values $u(t-T), u(t-2T), \ldots, u(t-k \cdot T)$, where T is the discretization time step and $k \in \mathbb{N}$, can be tracked. Using these previous output/input values as inputs, the neural network can obtain further understanding of the system dynamics and, for instance, know whether the output is increasing or decreasing, if the time derivate of the output is increasing or decreasing, etc.

3. Experimental Setup

A 1-DOF manipulator actuated by a ⌀ 0.13 mm Flexinol wire (DYNALLOY, Inc., 1562 Reynolds Ave., Irvine, CA, USA) was controlled using a pulse-width modulated current with a maximum of 230 mA that was fed by an Agilent e3631A power supply (5301 Stevens Creek Blvd., Santa Clara, CA, USA) set to an 8 V limit. The system controller consisted of a National Instruments myRIO-1900 (National Instruments Corporation, 11500 North Mopac Expy, Austin, TX, USA), whose 12-bit PWM output was connected to all eight amplifiers of a ULN2803 Darlington transistor array (Toshiba, 1-1, Shibaura 1-chome, Minato-ku, Tokyo, Japan) and a PDB181-K420K-103B potentiometer (Bourns, Inc.,

1200 Columbia Ave., Riverside, CA, USA). Several controlled parameters of the setup, including the weight of the manipulator arm and the maximum current allowed, were set so as to allow for repeatability, ensuring that the Flexinol wire was deformed reversibly [27]. With the current setup it was verified that this meant keeping the maximum strain below 4%. In our case, no two-way shape memory effect (SME) was observed and it was thus necessary to apply stress to the Flexinol wire in order to achieve any deformation.

To ensure the correctness of the angular position of the 1-DOF manipulator, the system was calibrated using an OptiTrack–Motive (NaturalPoint, Inc., 3658 SW Deschutes Street, Corvallis, OR, USA) motion capture system in a 16-cameras setup at a 240 Hz sampling rate (Appendix A). Figure 7 shows a photograph of the experimental setup. The dimensions of the manipulator are included in Appendix B.

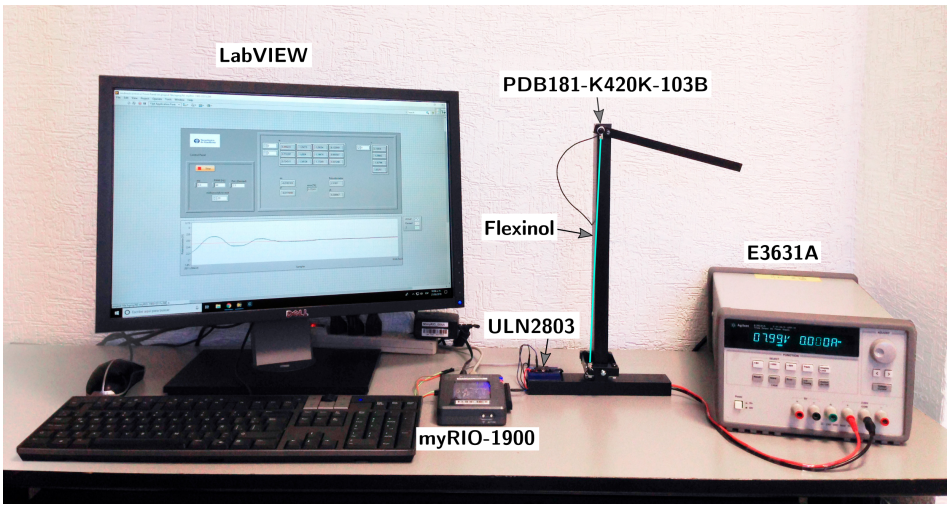


Figure 7. Photograph showing the experimental setup. The Flexinol wire has been marked in cyan.

The output signal from the potentiometer was fed into a feedback loop through the myRIO-1900 analog input with 12 bits of resolution, as seen in Figure 6. The complete experimental setup circuit is illustrated in Figure 8.

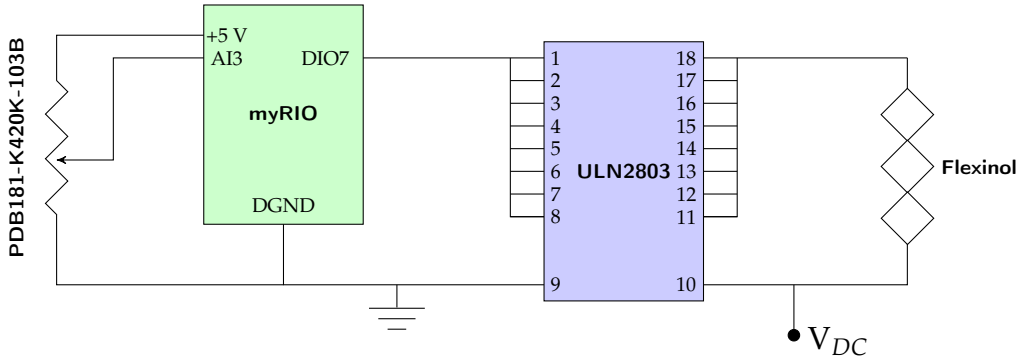


Figure 8. The circuit layout.

The wire used as an actuator was made of Flexinol—a nickel-titanium alloy that exhibits SMA properties. Information from the manufacturer is seen in Table 1.

Table 1. Flexinol wire characteristics.

Model No.	Diameter	Length	Operational Current	Transition Temperature	Pull Force	Resistance
STD-005-90	0.13 mm	305 mm	200 mA	90°	0.22 kg	0.75 Ω/cm

The software used to program the myRIO-1900 was NI LabVIEW 2017 (National Instruments Corporation, 11500 North Mopac Expy, Austin, TX, USA). A feedforward neural network with a three element sliding window (initialized to zero), consisting of 3 inputs, 4 hidden layer perceptrons and one output perceptron, was implemented and trained online using the back-propagation algorithm, without any prior training or knowledge of the system. The three utilized inputs, which were the present and two previous values of the output system error, were computed as the difference between the reference output value y_r to the measured output value y. Weights were adjusted for each iteration, until an acceptable error was reached. When no further adjustment was required, the learning procedure could be stopped. Optimization of the neural network architecture was made by performing experiments starting with a low number of hidden layer perceptrons and then increasing them one by one until no improvement of performance could be found in comparison to earlier experiments. The learning algorithm can be found in its entirety in Appendix C and is illustrated in the flowchart in Figure 9. The sample time of the controller was set to 20 ms and the PWM frequency to 60 Hz.

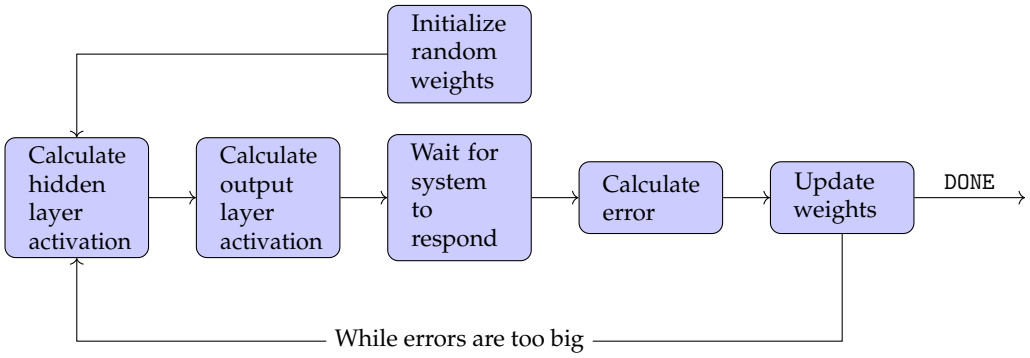


Figure 9. Flowchart illustrating the back-propagation algorithm.

4. Results

Prior to the experiments, a qualitative characterization of the system was performed. The results, confirming its hysteretic behaviour, can be seen in Figure 10.

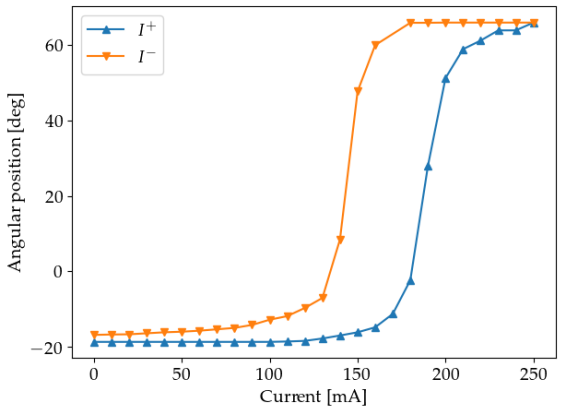


Figure 10. Characterization of the system. I^+ denotes that the current is monotonically increasing, while I^- denotes a monotonical decrease

Throughout every performed experiment, the Flexinol wire diamater as well as its original length and the ambient temperature remained constant. This is important since any change could affect the system behaviour [30].

The neural network coefficients were adjusted online by sending the manipulator to different set-points within the possible range, starting with the learning coefficient $\eta = 2$, a value deemed

appropriate by trial and error. The set-point was changed as soon as a low-error value was achieved and η reduced by a factor of 2 each time. The reduction factor 2 was also found by trial and error and corresponded to a value for which the weights were updated just enough to learn the new set-point without affecting the previously learnt ones. Finally, η was set to zero, thereby terminating the learning process. Online learning allows the controller to continue improving its performance for additional set-points and adapting to changes produced by system degradation and external disturbances. The learning process was terminated to evaluate the capability of the controller, limited to the knowledge acquired during the online learning, when being tested for set-points changes and load disturbances. It should, however, also be pointed out that the controller performed just as well—or even better—when η was kept bigger than zero, thus never terminating the learning. Results during a learning session, for the reference angular position y_r, the measured angular position y and the output system error (marked beneath in red), are presented in Figure 11.

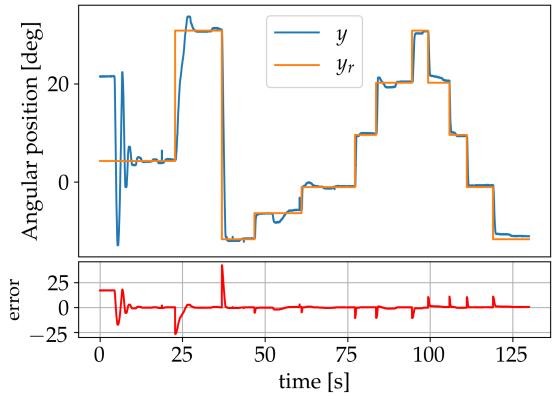


Figure 11. Learning session using set-points.

With η set to zero, the response of the system was verified for set-points $-12°$, $-1°$, $10°$, and $20°$, as seen in Figure 12. The maximum static error achieved was $1.28°$.

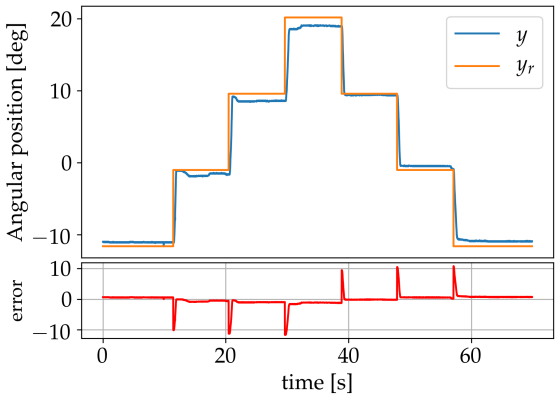


Figure 12. Results after learning using set-points.

A second learning procedure was implemented using a 0.01 Hz sine wave—ranging from $-20°$ to $31°$—as reference signal during learning. The learning coefficient η was primarily set to a value of 4 and gradually lowered to zero as the system started responding better to the reference signal. The learning process is illustrated in Figure 13. A validation done in the previous manner for set-points $-12°$, $-1°$, $10°$, $20°$, $31°$ can be seen in Figure 14 and resulted in a maximum static error of $0.83°$ for the set-point $-1°$. This error is reduced to $0.49°$ if we exclude the aforementioned set-point.

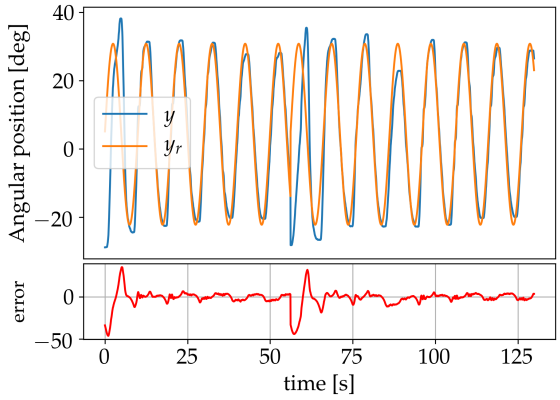


Figure 13. Learning session using a sine wave as reference signal.

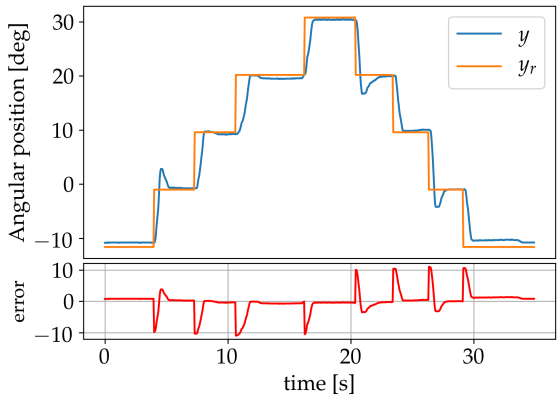


Figure 14. Results after learning.

Further validation of the system performance after learning was made using triangular and sine wave forms as reference signals. The triangle wave form was set to a frequency of 0.04 Hz with an amplitude of 53° and the results are shown in Figure 15. The sinusoidals were set to an amplitude of 42° and to frequencies 0.005 Hz, 0.01 Hz and 0.02 Hz. Figure 16a shows the result for the learning frequency of 0.01 Hz and displays no significant phase lag. In Figure 16b the result for the 0.005 Hz sine wave is shown, neither displaying any significant signs of phase lag. Figure 17 shows the result for the 0.02 Hz sine wave, where a 0.3 s phase lag can be identified.

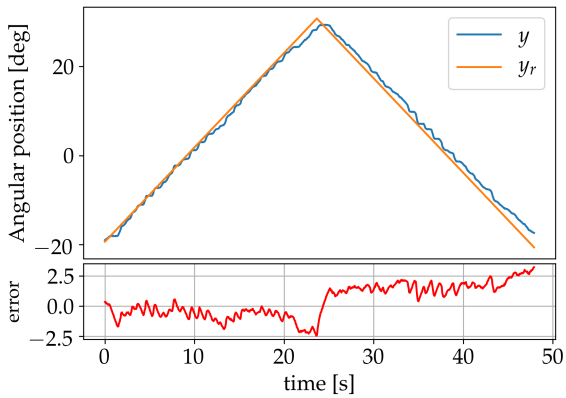
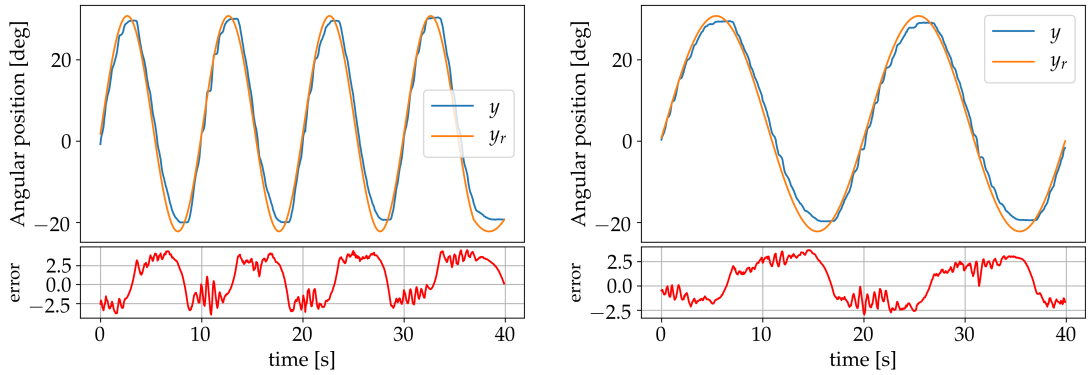


Figure 15. Results for a triangle wave form.



(**a**) (**b**)

Figure 16. (**a**) Results for a 0.01 Hz sine wave; (**b**) Results for a 0.005 Hz sine wave.

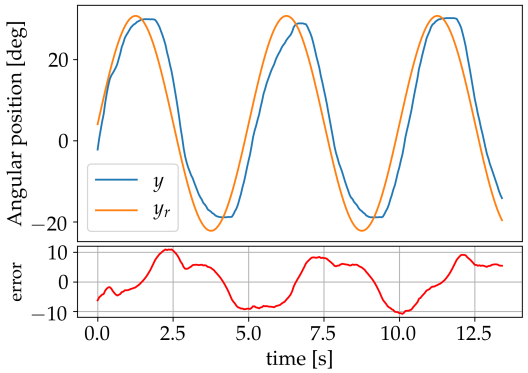
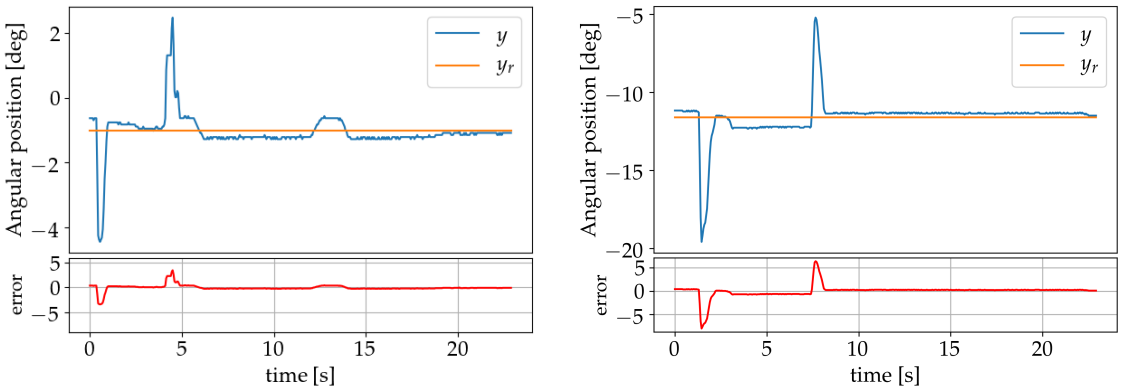


Figure 17. Result for a 0.05 Hz sine wave.

The effect of disturbances on the system was investigated by a sudden application of additional torque on the arm that was removed once the system had stabilized. This was done first by applying an extra 86% of torque (Figure 18a), and later an extra 143% (Figure 18b). In both cases, the perturbation was compensated for by the controller without additional learning to adjust the weights.



(**a**) (**b**)

Figure 18. Plots showing the results when applying an extra: (**a**) 85% torque; (**b**) 143% torque.

5. Conclusions

A real-time, low computational cost, neural network direct control, with online learning, was developed for position control of shape memory alloy manipulators. Neural network weight

coefficients were updated online by using the sensor position data while the controller was applied to the system, without previous training of the neural network weights nor the inclusion of a hysteresis model. Experimental evaluation was performed on a 1-DOF manipulator system actuated by a shape memory alloy wire and test results verify the effectiveness of the proposed control scheme to control the system angular position, compensating for the hysteretic behavior of the shape memory alloy actuator. Using a learning algorithm based on sending the actuator to distinct set-points within the possible range, a maximum static error of 1.28° was achieved when validating it against a staircase reference signal. This was later improved upon using a 0.01 Hz sine wave reference signal for learning, resulting in a maximum error of 0.83° for the same type of validation, or 0.49° for set-points above −12°. The effectiveness of the controller was also tested against a 0.04 Hz ramp and sine waves of frequencies 0.005-, 0.01- and 0.02 Hz without showing any severe signs of phase lag, with the worst case (0.02 Hz) presenting a lag of 0.3 s. The controller also showed its ability to compensate for load disturbances, when an additional torque of 143% was applied.

For future work, a detailed comparison with results for other existing techniques could be performed, including changing the control system and neural network architecture, as well as the utilized learning algorithm. In addition, a study of the effect of periodic disturbances on the system could be performed, as well as an investigation into if the system can be controlled without the use of sensors.

Author Contributions: Conceptualization, A.G.-E.; methodology, A.G.-E.; software, I.L.E. and R.C.S.; validation, R.C.S. and I.L.E.; formal analysis, A.G.-E., R.C.S. and I.L.E.; investigation, R.C.S., A.G.-E. and I.L.E.; resources, A.G.-E.; data curation, R.C.S.; writing—original draft preparation, R.C.S. and A.G.-E.; writing—review and editing, A.G.-E., E.C.-U., C.D.T.-Q. and R.C.S.; visualization, R.C.S.; supervision, A.G.-E.; project administration, A.G.-E.; funding acquisition, A.G.-E.

Funding: The authors would like to acknowledge the financial support of Tecnologico de Monterrey, in the production of this work.

Acknowledgments: We thank Fabiola Díaz Nieto who provided insight and expertise that greatly assisted on National Instrument hardware and LabVIEW programming. The authorsalso wish to thank Alethia Jocelyn Delgado De la Paz for the 1-DOF manipulator 3D modeling, Guillermo Salvador Barrón Sánchez for the angular position calibration using the motion capture system, Arturo Gómez Sierra for the photography artwork, and Erika Nataly Santana Aguilar, Rocío Morfín Díaz, Ramon Ariel Ivan Muñoz Corona, and Omar Enrique Trejo Díaz for their assistance during the earlier stages of this project.

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

Abbreviations

The following abbreviations are used in this manuscript:

(A)NN	(artificial) neural network
1-DOF	one degree of freedom
MEMS	micro-electromechanical systems
MLP	multilayer perceptron
PID	proportional-integral-derivative
PWM	pulse-width modulation
SMA	shape memory alloy

Appendix A. Motion Capture System

The setup of the motion capture system is seen in Figures A1 and A2 and resulted in the calibration equation

$$y = 53x - 128, \tag{A1}$$

with a linear correlation coefficient $R = 0.9997$, where y is the angular position and x the potentiometer voltage.

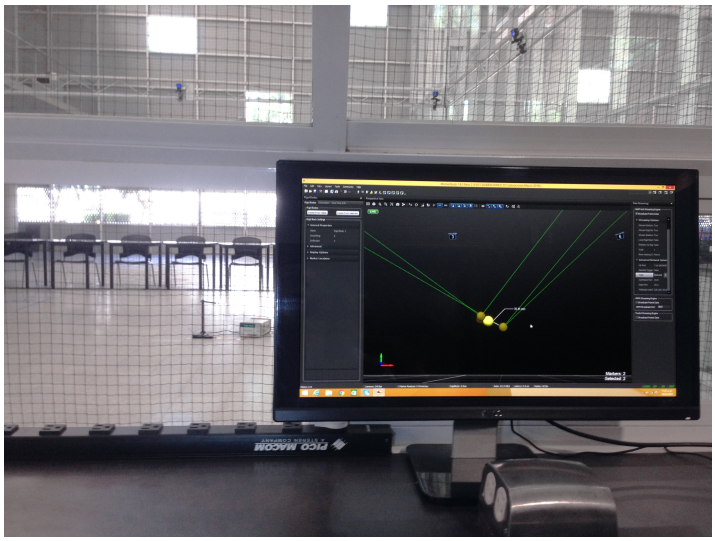


Figure A1. The setup of the motion capture system.

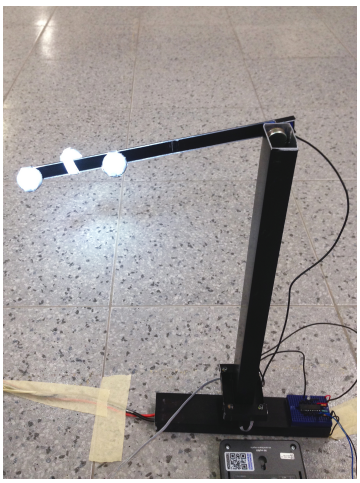
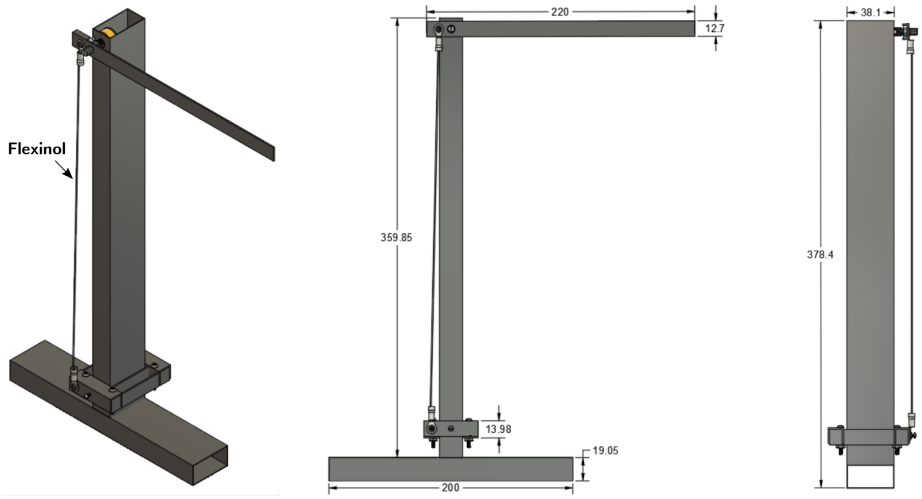


Figure A2. Setup of the reflective markers.

Appendix B. 1-DOF Manipulator



(**a**) (**b**) (**c**)

Figure A3. Schematic showing the dimensions in millimeters of the 1-DOF manipulator used in experiments: (**a**) Perspective view; (**b**) Front view; (**c**) Side view.

Appendix C. Back-Propagation

Listing 1: The back-propagation algorithm in pseudocode.

```
# SETUP
define vector "inputs" = [x_1,...,x_n]
define vector "correct_outputs" = [d_1,...,d_n]
define boolean "DONE" = FALSE
initialize float "eta" > 0
initialize float "tolerance" > 0
initialize random weights "w_ij" and "v_j"
initialize float "t" > 0

# TRAINING
while not DONE:
 |  for x_k in inputs and d_k in correct_outputs:
 |   |  # Calculate hidden layer activation
 |   |  h_j = sigmoid( sum_over_i(x_i*w_ji) )
 |   |
 |   |  # Calculate output layer activation
 |   |  u   = sigmoid( sum_over_j(h_j*v_j) )
 |   |
 |   |  # Let the system respond
 |   |  # before calculating error
 |   |  wait t seconds
 |   |
 |   |  # Calculate error
 |   |  e_y_k = x_k-d_k
 |   |
 |   |  # Update weights
 |   |  w_ji += eta*e_y_k*u*(1-u)*h_j
 |   |  v_j  += eta*e_y_k*u*(1-u)*h_j*(1-h_j)*v_j*x_i
 |  end for
 |
 |  if e_y_k < tolerance for all k:
 |      DONE = TRUE
end while
```

References

1. Zhou, M.; Zhang, Q. Hysteresis model of magnetically controlled shape memory alloy based on a PID neural network. *IEEE Trans. Magn.* **2015**, *51*, 1–4.
2. Chinni, F.; Spizzo, F.; Montoncello, F.; Mattarello, V.; Maurizio, C.; Mattei, G.; Bianco, L.D. Magnetic hysteresis in nanocomposite films consisting of a ferromagnetic AuCo alloy and ultrafine Co particles. *Materials* **2017**, *10*, 717. [CrossRef] [PubMed]
3. Tan, Q.; Kang, H.; Xiong, J.; Qin, L.; Zhang, W.; Li, C.; Ding, L.; Zhang, X.; Yang, M. A wireless passive pressure microsensor fabricated in HTCC MEMS technology for harsh environments. *Sensors* **2013**, *13*, 9896–9908. [CrossRef] [PubMed]
4. Tu, F.; Hu, S.; Zhuang, Y.; Lv, J.; Wang, Y.; Sun, Z. Hysteresis Curve Fitting Optimization of Magnetic Controlled Shape Memory Alloy Actuator. *Actuators* **2016**, *5*, 25. [CrossRef]

5. Nasiri-Zarandi, R.; Mirsalim, M. Finite-element analysis of an axial flux hysteresis motor based on a complex permeability concept considering the saturation of the hysteresis loop. *IEEE Trans. Ind. Appl.* **2016**, *52*, 1390–1397.
6. Li, Z.; Zhang, X.; Su, C.Y.; Chai, T. Nonlinear control of systems preceded by Preisach hysteresis description: A prescribed adaptive control approach. *IEEE Trans. Control Syst. Technol.* **2016**, *24*, 451–460. [CrossRef]
7. Vázquez, M.; Nielsch, K.; Vargas, P.; Velázquez, J.; Navas, D.; Pirota, K.; Hernandez-Velez, M.; Vogel, E.; Cartes, J.; Wehrspohn, R.; et al. Modelling hysteresis of interacting nanowires arrays. *Phys. B Condens. Matter* **2004**, *343*, 395–402. [CrossRef]
8. Dong, R.; Tan, Y. A modified Prandtl–Ishlinskii modeling method for hysteresis. *Phys. B Condens. Matter* **2009**, *404*, 1336–1342. [CrossRef]
9. Oh, J.; Bernstein, D.S. Semilinear Duhem model for rate-independent and rate-dependent hysteresis. *IEEE Trans. Autom. Control* **2005**, *50*, 631–645.
10. Almassri, A.; Wan Hasan, W.; Ahmad, S.; Shafie, S.; Wada, C.; Horio, K. Self-calibration algorithm for a pressure sensor with a real-time approach based on an artificial neural network. *Sensors* **2018**, *18*, 2561. [CrossRef]
11. Liu, Z.; Lai, G.; Zhang, Y.; Chen, X.; Chen, C.L.P. Adaptive neural control for a class of nonlinear time-varying delay systems with unknown hysteresis. *IEEE Trans. Neural Netw. Learn. Syst.* **2014**, *25*, 2129–2140. [PubMed]
12. Liu, Z.; Lai, G.; Zhang, Y.; Chen, C.L.P. Adaptive neural output feedback control of output-constrained nonlinear systems with unknown output nonlinearity. *IEEE Trans. Neural Netw. Learn. Syst.* **2015**, *26*, 1789–1802.
13. Fulginei, F.R.; Salvini, A. Neural network approach for modelling hysteretic magnetic materials under distorted excitations. *IEEE Trans. Magn.* **2012**, *48*, 307–310. [CrossRef]
14. Lin, F.J.; Shieh, H.J.; Huang, P.K. Adaptive wavelet neural network control with hysteresis estimation for piezo-positioning mechanism. *IEEE Trans. Neural Netw.* **2006**, *17*, 432–444. [CrossRef] [PubMed]
15. Seidl, D.R.; Lam, S.L.; Putman, J.A.; Lorenz, R.D. Neural network compensation of gear backlash hysteresis in position-controlled mechanisms. *IEEE Trans. Ind. Appl.* **1995**, *31*, 1475–1483. [CrossRef]
16. Liaw, H.C.; Shirinzadeh, B.; Smith, J. Robust neural network motion tracking control of piezoelectric actuation systems for micro/nanomanipulation. *IEEE Trans. Neural Netw.* **2009**, *20*, 356–367. [CrossRef] [PubMed]
17. Cao, S.; Wang, B.; Zheng, J.; Huang, W.; Weng, L.; Yan, W. Hysteresis compensation for giant magnetostrictive actuators using dynamic recurrent neural network. *IEEE Trans. Magn.* **2006**, *42*, 1143–1146.
18. Sayyaadi, H.; Zakerzadeh, M.R. Position control of shape memory alloy actuator based on the generalized Prandtl–Ishlinskii inverse model. *Mechatronics* **2012**, *22*, 945–957. [CrossRef]
19. Asua, E.; Etxebarria, V.; García-Arribas, A. Neural network-based micropositioning control of smart shape memory alloy actuators. *Eng. Appl. Artif. Intell.* **2008**, *21*, 796–804. [CrossRef]
20. Zhou, M.; Zhang, Q.; Wang, J. Feedforward-feedback hybrid control for magnetic shape memory alloy actuators based on the Krasnosel'skii-Pokrovskii model. *PLoS ONE* **2014**, *9*, e97086. [CrossRef]
21. Ghasemi, Z.; Nadafi, R.; Kabganian, M.; Abiri, R. Identification and Control of Shape Memory Alloys. *Meas. Control* **2013**, *46*, 252–256. [CrossRef]
22. Tai, N.T.; Ahn, K.K. A hysteresis functional link artificial neural network for identification and model predictive control of SMA actuator. *J. Process Control* **2012**, *22*, 766–777. [CrossRef]
23. Nikdel, N.; Nikdel, P.; Badamchizadeh, M.A.; Hassanzadeh, I. Using neural network model predictive control for controlling shape memory alloy-based manipulator. *IEEE Trans. Ind. Electron.* **2014**, *61*, 1394–1401. [CrossRef]
24. Nikdel, N.; Badamchizadeh, M.A. Design and implementation of neural controllers for shape memory alloy–actuated manipulator. *J. Intell. Mater. Syst. Struct.* **2015**, *26*, 20–28. [CrossRef]
25. Janičić, V.; Ilić, V.; Pjevalica, N.; Nikolić, M. An approach to modeling the hysteresis in ferromagnetic by adaptation of Preisach model. In Proceedings of the 2014 22nd Telecommunications Forum Telfor (TELFOR), Belgrade, Serbia, 25–27 November 2014; pp. 761–764.
26. Wang, X.; Sun, T.; Zhou, J. Identification of preisach model for a fast tool servo system using neural networks. In Proceedings of the 2008 IEEE International Symposium on Knowledge Acquisition and Modeling Workshop, Wuhan, China, 21–22 December 2008; pp. 232–234.

27. Sun, L.; Huang, W.M.; Ding, Z.; Zhao, Y.; Wang, C.C.; Purnawali, H.; Tang, C. Stimulus-responsive shape memory materials: A review. *Mater. Des.* **2012**, *33*, 577–640. [CrossRef]
28. Hernández-Alvarado, R.; García-Valdovinos, L.G.; Salgado-Jiménez, T.; Gómez-Espinosa, A.; Fonseca-Navarro, F. Neural network-based self-tuning PID control for underwater vehicles. *Sensors* **2016**, *16*, 1429. [CrossRef] [PubMed]
29. Cui, X.; Shin, K.G. Direct control and coordination using neural networks. *IEEE Trans. Syst. Man Cybern.* **1993**, *23*, 686–697.
30. An, L.; Huang, W.M.; Fu, Y.Q.; Guo, N. A note on size effect in actuating NiTi shape memory alloys by electrical current. *Mater. Des.* **2008**, *29*, 1432–1437. [CrossRef]