Meccanica (2023) 58:537–553
https://doi.org/10.1007/s11012-023-01639-4

ORIGINAL PAPER

Optimization of the closed-loop controller of a discontinuous capsule drive using a neural network

Sandra Zarychta · Marek Balcerzak · Volodymyr Denysenko · Andrzej Stefański · Artur Dąbrowski · Stefano Lenci

Received: 31 August 2022 / Accepted: 22 January 2023 / Published online: 22 February 2023

Abstract In this paper the construction of a neural-network based closed-loop control of a discontinuous capsule drive is analyzed. The foundation of the designed controller is an optimized open-loop control function. A neural network is used to determine the dependence between the output of the open-loop controller and the state of the system. Robustness of the neural controller with respect to variation of parameters of the controlled system is analyzed and compared with the original optimized open-loop control. It is expected that the presented method can facilitate the construction of closed-loop controllers for which alternative methods are not effective, such as non-smooth or discontinuous ones.

Keywords Optimal control · Closed-loop · Neural network · Capsule drive · Discontinuous system · Non-smooth system

S. Zarychta · S. Lenci
Department of Civil and Building Engineering, and Architecture, Polytechnic University of Marche, via Brecce Bianche, 60131 Ancona, Italy

S. Zarychta (✉) · M. Balcerzak · V. Denysenko · A. Stefański · A. Dąbrowski
Division of Dynamics, Lodz University of Technology, Stefanowskiego 1/15, 90-537 Lodz, Poland
e-mail: sandra.zarychta@p.lodz.pl

1 Introduction

The capsule robots, often abbreviated to capsubots, are a class of capsule-shaped micro-robots able to explore the fields that are normally inaccessible to humans [1]. A particularly interesting subset of capsubots are devices propelled by an internal mechanical oscillator. The vibrating mass produces inertial forces which allow the whole capsule to move in the presence of friction. In this approach the external moving parts such as wheels, tracks, robotic legs or arms are no longer necessary [1, 2]. Such capsubots are truly intriguing from the practical point of view due to their enormous potential in medicine, engineering and other areas [1, 2]. Moreover, their rich dynamics [2] encompassing such phenomena as impacts, dry friction, etc. remains a broad research topic itself.

Wide range of research concerning the analysis and design of capsubots has already been performed. For instance, Guo et al. [2] presented a mesoscale prototype of a self-propelled vibro-impact capsule system as well as its optimization in terms of the average progression velocity, energy efficiency and power consumption. Huda and Yu [3] developed a control strategy for a capsubot, according to which the inner mass of the device—comprising two masses placed at the opposite ends of the cylindrical rod—is surrounded by a motor housing with a coil held in a shell. In [4], Liu et al. proposed a vibro-impact capsule containing a harmonically excited internal oscillator impacting a

massless plate suspended on a spring. Such arrangement causes the resultant horizontal force acting on the capsule to be asymmetric, which in the presence of dry friction enables the system to move forward. More detailed studies on the control function with regard to the progression rate, as well as the optimization of energy consumption, have been presented in [5–7]. A comprehensive bifurcation analysis of a vibro-impact system with the use of path-following methods accompanied by an experimental investigation can be found in [8]. Liu et al. [9] described a downscaled self-propelled vibro-impact capsule system with an ability to move precisely in a limited space, having a size equal to a market-leading gastrointestinal capsule endoscope. The capsule system includes two impact constraints with a linear bearing holding a T-shape magnet situated between them that restricts its linear motion. The dynamic analysis of the prototype, as well as the optimization of the progression speed and minimization of the required propulsive force are described.

Apart from the vibro-impact systems, a different layout of the capsule drives can be utilized. In particular, an interesting modification of the vibro-impact capsule can be obtained by replacing the mass-on-spring oscillator with a pendulum. In this case, propulsion of the capsule is caused by the interactions between friction, inertial forces produced by the swinging motion of the pendulum and the contact force between the capsule and the underlying surface [10]. Such arrangement seems to make the dynamics of the system somewhat more complex, because the contact force is dependent on the oscillations of the pendulum. Periodic locomotion principles and nonlinear dynamics of a pendulum-driven capsule system have been investigated in the works [10–12]. It is worth noting that in [12], Liu included a motion-generation strategy in the presence of viscoelasticity. The design and optimization of parameters of a predesigned control function profile for the pendulum capsule system have been considered in [12–14].

The existing methods of controller design applicable to capsubots use various approaches including the open-loop control, closed-loop feedback linearization or neural networks. In [1], Liu et al. presented three control approaches for capsubots. The first one involves an open-loop control, whereas the second utilizes a closed-loop control with a partial feedback linearization technique based on trajectory tracking. The last one, called a simple switch control, is a combination of the previous methods. The control profiles learned from the open and closed-loop control are used to move the capsubot effectively in the desired direction. In [3], Huda and Yu described a strategy for controlling a cylindrical rod, composed of two stages. The first one assumes the desired trajectory generation, whereas the second focuses on inner mass closed-loop control for a given desired trajectory with a partial feedback linearization approach. An adaptive trajectory tracking control method for a vibro-driven capsule system has been described by Liu et al. [13]. The implementation of an auxiliary input control variable establishing the non-collocated feedback loop, is constructed to cope with the parametric uncertainties. A comparison of the proposed approach with the classical one has been performed with the aid of a closed-loop feedback-tracking control system. The improvements in this method have been shown by Liu et al. [15]. This novel approach focuses on adding a neural network approximator and a robust compensator to an auxiliary control variable. The proposed design method with multi-layer neural networks and variable strategy structure as well as an adaptive tracking control scheme copes well with uncertainties such as a priori unknown parameters, approximation errors and disturbances [15]. In [16], Zarychta et al. described a novel Fourier series-based method of the open-loop optimal control estimation, applicable to discontinuous systems such as capsule drives. In [17], the problem of crossing a circular fold by a capsule robot has been discussed by Yan et al. The path following techniques have been utilized and the COCO software has been used in numerical studies.

Liao et al. [18] described the speed optimization of self-propelled capsule robot [9] in the varying frictional environment between the device and its supporting surface with the use of Six Sigma and Multi-Island genetic algorithms, having utilized the Monte Carlo approach for validation. In the literature we can find another interesting examples of the multi-objective optimization with Six Sigma as a controller for the genetic algorithm [19, 20] and its reliability analysis with the use of a Monte Carlo algorithm [21, 22].

There is seemingly little research on the application of neural networks in the optimal capsule drives control or other similar systems. Possibly, the direct application of the Reinforcement Learning technique [23–25] could be used to obtain an approximation of

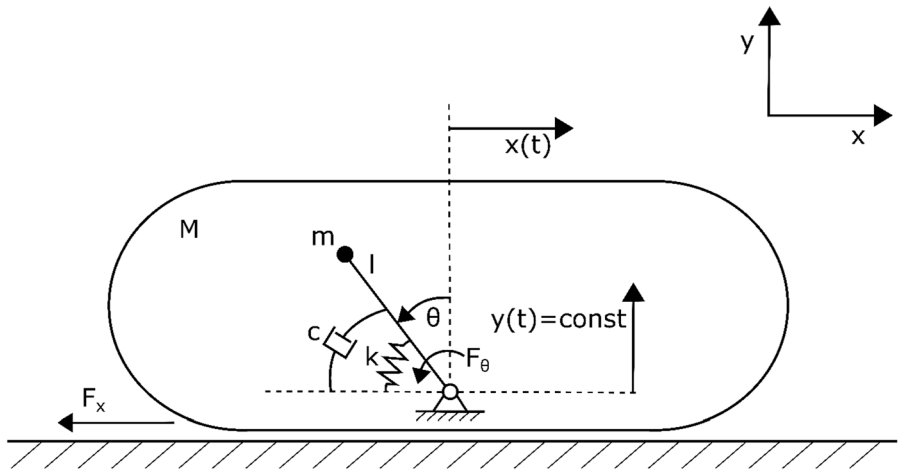

Fig. 1 Scheme of the capsule drive system. M—mass of the capsule, m—mass of the pendulum, l—length of the pendulum, θ—pendulum angle, k—spring stiffness, c—damping coefficient, F_θ—external torque acting on the pendulum, F_x—friction force, x(t), y(t)—coordinates of the capsule [16]

the optimal closed-loop controller. However, such an approach would require a lot of time and computational resources [23–25]. Therefore, in this work we propose a simpler option. Provided that the optimal open-loop control is determined, a neural network approach can be used to establish the dependence of the controller's output and the corresponding states of the controlled system. In such a manner, a closed-loop controller can be obtained, the action of which reflects the open-loop optimal control operation.

The aim of this study is to test and evaluate the aforementioned concept. For this purpose, an approximation of the open-loop optimal control of a pendulum capsule drive is performed by means of the Fourier series-based method described by us in [16]. After that, a neural network is used to determine the dependency between the output value of the optimized open-loop controller and the corresponding states of the capsule system. In such a manner, a closed-loop controller is obtained. Finally, the performance and robustness of the closed-loop neural controller are compared with the original open-loop one. The results show that the neural controller maintains the efficiency of the original and offers greater robustness against the uncertainty of the controlled system friction coefficient, which is actually one of the main limitations in the use of open-loop controllers.

We believe that such a solution can be an interesting option in the design and optimization of controllers used in the mechanical systems, including the discontinuous ones. Moreover, it is expected that the proposed method will facilitate the construction of closed-loop controllers of the systems for which an optimal open-loop control is available.

2 Mathematical model

The subject of this research is the pendulum capsule drive. This section presents a brief description of the system along with an approximation of its open-loop optimal control. The information presented below is a foundation for the new, neural network based, closed-loop controller of the device. A scheme of the pendulum capsule drive is presented in Fig. 1.

In the system under consideration, the propulsion of capsule is caused by the interactions between friction F_x, horizontal inertial forces produced by the swinging motion of the pendulum R_x and the reaction (contact) force between the capsule and the underlying surface R_y. Dynamics of the presented system is thoroughly described in [10–12]. In order to derive the motion equations of the capsule, one can either directly apply Newton's laws of motion or use the Lagrange approach. In the latter, the value of the constraint force R_y can be determined with the aid of a Lagrange multiplier [26].

A detailed derivation of the pendulum capsule motion equations can be found in our previous work [16]. Therefore, in this paper we present only a brief explanation. In the dimensional form, equations of motion of the capsule are as follows:

$$ml^2\ddot{\theta}(t) - ml\ddot{x}(t)\cos\theta(t) = mgl\sin\theta(t) - k\theta(t) - c\dot{\theta}(t) + F_\theta(t) \tag{1a}$$

$$(M+m)\ddot{x}(t) - ml\ddot{\theta}(t)\cos\theta(t) + ml\dot{\theta}^2(t)\sin\theta(t) = -F_x(t) \tag{1b}$$

$$R_y(t) = (M+m)g - ml\ddot{\theta}(t)\sin\theta(t) - ml\dot{\theta}^2(t)\cos\theta(t) \tag{1c}$$

where g is the gravitational acceleration and all the other symbols are described in the caption below Fig. 1. Then, the following non-dimensional quantities are used:

$$\Omega = \sqrt{\frac{g}{l}}, \tau = \Omega t, \gamma = \frac{M}{m}, z = \frac{x}{l}, \rho = \frac{k}{m\Omega^2 l^2}, \nu = \frac{c}{m\Omega l^2},$$
$$f_z = \frac{F_x}{m\Omega^2 l}, u = \frac{F_\theta}{m\Omega^2 l^2}, r_z = \frac{R_x}{m\Omega^2 l}, r_y = \frac{R_y}{m\Omega^2 l} \tag{2}$$

where t and τ correspond to dimensional and dimensionless time, respectively. Relations between derivatives with respect to t and τ are as follows.

$$\dot{x} = \frac{dx}{dt} = \frac{dx}{d\tau}\frac{d\tau}{dt} = \Omega\frac{dx}{d\tau} = \Omega x', \ddot{x} = \frac{d^2x}{dt^2}$$
$$= \frac{d}{dt}\left(\frac{dx}{dt}\right) = \frac{d}{d\tau}\left(\Omega\frac{dx}{d\tau}\right)\frac{d\tau}{dt} = \Omega^2\frac{d^2x}{d\tau^2} = \Omega^2 x'' \tag{3}$$

Using symbols and notation defined in formulas (2) and (3), equations of motion of the pendulum capsule drive can be presented in the following dimensionless, matrix form.

$$\begin{bmatrix} 1 & -\cos\theta(\tau) \\ -\cos\theta(\tau) & \gamma+1 \end{bmatrix}\begin{bmatrix} \theta''(\tau) \\ z''(\tau) \end{bmatrix}$$
$$= \begin{bmatrix} \sin\theta(\tau) - \rho\theta(\tau) - \nu\theta'(\tau) + u(\tau) \\ -\theta'^2(\tau)\sin\theta(\tau) - f_z(\tau) \end{bmatrix} \tag{4}$$

Further dimensionless quantities are contact force r_y, resultant horizontal load due to the pendulum's motion r_z, and the dimensionless Coulomb friction f_z, that are described by the following equations:

$$r_y(\tau) = (\gamma+1) - \theta''(\tau)\sin\theta(\tau) - \theta'^2(\tau)\cos\theta(\tau) \tag{5}$$

$$r_z(\tau) = \theta''(\tau)\cos\theta(\tau) - \theta'^2(\tau)\sin\theta(\tau) \tag{6}$$

$$f_z(\tau) = \begin{cases} \mu r_y(\tau) sgn\left[z'(\tau)\right] \leftrightarrow z'(\tau) \neq 0 \\ \mu r_y(\tau) sgn\left[r_z(\tau)\right] \leftrightarrow z(\tau) = 0 \wedge \left|r_z(\tau)\right| \geq \mu r_y(\tau) \\ r_z(\tau) \leftrightarrow z(\tau) = 0 \wedge \left|r_z(\tau)\right| < \mu r_y(\tau) \end{cases} \tag{7}$$

where μ is the friction coefficient. Equations (4)–(7) form the complete, dimensionless model of the capsule pendulum drive presented in Fig. 1.

Our previous paper [16] describes a numerical method that enables approximation of the optimal control in the form of a finite number of Fourier series terms (8).

$$u(\tau) = \frac{a_0}{2} + \sum_{k=1}^{K} a_k\cos(k\omega\tau) + \sum_{k=1}^{K} b_k\sin(k\omega\tau) \tag{8}$$

The detailed description of the method is beyond the scope of the current paper. Therefore, only the main idea is briefly explained here.

Using a finite number of the Fourier expansion harmonics (8), any periodic, bounded, piecewise continuous function can be approximated. The approximation accuracy increases with the number of harmonics, K. Therefore, Eq. (8) enables the parametrization of control function with the aid of $2K+3$ parameters: $a_0, a_1, \ldots, a_K, b_1, \ldots, b_K, \omega, K$. However, such parametrization cannot be effectively used to transform the optimal control problem into a nonlinear programming problem, i.e., it is not possible to simply optimize a_i, b_i parameters of the formula (8). The reason for this is the fact that the control function in all practical applications is bounded and there is no direct method to transform the restrictions imposed on the control function into the limits of values of the parameters a_i, b_i. The solution to this problem is explained in Fig. 2. One can notice that if the coefficients of subsequent harmonics are stacked in a vector $[a_1, b_1, \ldots, a_K, b_K]$, then the "span" of the resulting function depends on the *length* of such vector, whereas its "shape" changes with the *direction* of the vector.[1] The direction of a vector in $\mathbb{R}^{2K}$ can be described by a point on a unit sphere of dimension $(2K-1)$, which is specified by $2K-1$ spherical coordinates. Then, the location of the control function span within the frame specified by the set of allowable controls can be described by two additional numbers from the interval [0, 1]. Such transformation enables parametrization of the Fourier expansion (8) in terms of $(2K-1)$ spherical coordinates that specify the *shape* of the optimized function, 2 parameters

[1] The *span* of a function can be understood as the difference between supremum and infimum of its set of values. The *shape* of a bounded function is defined as its normalization to the interval [0, 1]. Graphical interpretation of these notions is presented in Fig. 2. Detailed discussion can be found in paper [16].

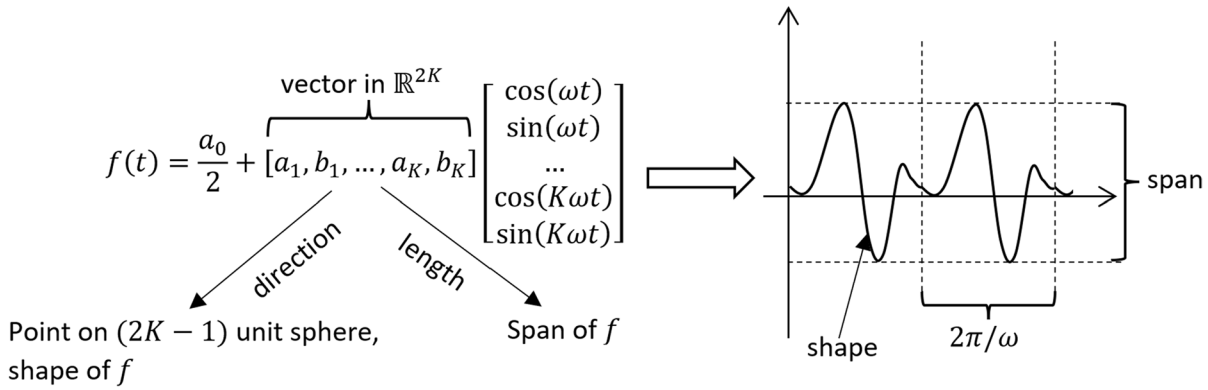

Fig. 2 Scheme of the Fourier series based method for optimal control approximation

governing its *span*, parameter ω, which influences the period and a fixed number of harmonics K. In such a manner, the function (8) is well-defined in terms of $2K+1$ parameters in fixed ranges, which enables their optimization, the parameter ω (which can be either optimized or selected beforehand), and the parameter K, which has to be selected a-priori. For more details, please refer to paper [16].

Using the algorithm described above, the control of the system (4)-(7) has been optimized with respect to the distance covered by the capsule within the dimensionless time interval $\tau \in [0, 100]$. In the optimization process, the following values of system parameters have been assumed: $\mu = 0.3, \rho = 2.5, \nu = 1.0, \gamma = 10$. Moreover, it has been asserted that the control $u(\tau)$ has to remain in the allowable range $[-4, 4]$. Under such assumptions, taking $K = 5$ harmonics in the formula (8), the following parameters of the approximate, open-loop optimal control have been obtained.

$$\begin{aligned} &a_0 \approx 1.62506, \omega \approx 1.64722 \\ &\left(a_1, a_2, a_3, a_4, a_5\right) \approx (-3.43222, -1.95285, -0.68182, 0.38493, 0.17389) \\ &\left(b_1, b_2, b_3, b_4, b_5\right) \approx (-0.41690, 0.12411, -0.10468, 0.13722, 0.27902) \end{aligned} \tag{9}$$

The control function (8) with parameters (9) and the resulting trajectory of the capsule's motion are presented in the 4th section of the paper. These results are the starting point for the current research. In the remaining part of this work, we are going to show that a neural network can learn from the open-loop solution (8) in order to form a closed-loop controller. Moreover, it will be demonstrated that, counterintuitively, such a neural network can outperform the original solution which served as the training set in the learning process (see Sect. 4).

3 Methods

In the following sections, the use of a Neural Network (NN) in the closed-loop controller design is presented. We base this process on the approximated solution of the open-loop optimal control function, for which we used our developed Fourier series-based method [16]. The objective of the NN is to return the value of optimized control for an arbitrary state of the controlled object (i.e., the pendulum capsule drive).

The research is divided into three stages. Within the preliminary research, we evaluate the performance of feedforward artificial neural network predictive models that approximate the dependencies between the optimized open-loop control and the corresponding state variables of the system. In order to achieve this, the multi-layer perceptron (MLP) has been created in the Python language, the architecture of which is described in Sect. 3.2. In this stage, we test the design of the MLP with regard to different activation functions for the hidden and output layer, along with the changing number of neurons. The process of NN training and the criteria used for

Fig. 3 The reference dataset for NN training

samples	x_0	x_1	x_2	x_3	time τ	target (optimal control) u
	features (system state variables)					
0	0.000000	0.000000	0.000000	-1.000000e-15	0.00	-3.88301
1	-0.000196	-0.039061	-0.000003	-5.512100e-04	0.01	-3.85803
2	-0.000779	-0.077407	-0.000011	-1.038640e-13	0.02	-3.83138
3	-0.001742	-0.115023	-0.000023	-1.462100e-03	0.03	-3.80320
.						
.						
.						
10091	-1.07457	-1.81183	-6.06491	-0.045840	100.00	3.25619

the model performance validation are described in Sect. 3.3.

In the second stage, the parameters of the NN predictive models with the top-scoring performance are implemented in the simulated controller of the pendulum capsule drive. Therefore, we are able to calculate the distance covered by the capsule system driven by the neural controller.

In the last part, the robustness of both controllers (the optimized open-loop and the neural closed-loop) concerning the varying coefficient of friction between the capsule and the underlying surface is examined. The performance of the controllers is tested in different conditions, from constant friction coefficient to large variations of this parameter.

To facilitate the comprehension of the presented method, all the above-described steps are performed in an Appendix using a much simpler example—a mathematical pendulum. We encourage the reader to analyze this additional material. Moreover, all the scripts created within the aforementioned stages of this study are available in a reference data repository [35].

3.1 The reference dataset

In this research, as a reference dataset we consider an optimized control trajectory of the pendulum capsule system along with its control calculated according to the formulas of the Fourier series-based algorithm presented in Sect. 2.

The reference dataset (see Fig. 3), consisting of approximately 10,000 control samples with six columns, contains the information about the system state variables $x_0 - x_3$, optimized, open-loop control u and dimensionless time τ. In each row of the dataset, one sample of the reference control is represented. Its inputs called features, stored as columns, refer to: x_0 —position of the pendulum θ, x_1—velocity of the pendulum $\dot{\theta}$, x_2—position of the capsule z, and x_3 velocity of the capsule z'. Since the control has to be independent of the capsule position and the time, the related variable x_2, along with the dimensionless time τ were dropped from the learning process. Therefore, we consider three features, i.e., state variables of the system (see formulas (4)–(7)) and the corresponding control u as our target variable that is going to be predicted by NN.

3.2 The Neural Network architecture

The design of NN architecture demands a few key decisions to be made, for instance to select the number of layers, number of neurons and activation functions. The number of hidden layers depends on the problem which we are dealing with. In this research we fix one hidden layer, as the given configuration is sufficient to approximate an arbitrary continuous function [27, 28], such as the optimal control (the target of our NN prediction). Moreover, we want to provide a simple method that does not require the design of the deep NN model.

Another problem to be solved is the number of neurons. This should be chosen in a way that avoids under or overfitting. In the first case where there are not enough neurons to train, the results may not be satisfactory and adequate to what has been expected. On the other hand, with too many neurons, the artificial neural network possesses an excessive number of parameters to be determined which makes it "remember" each data point separately and lose the data generalization property. Many rules of thumb exist for establishing the correct number of neurons in a hidden layer [27]. We decided to test one of them, where the number of hidden neurons is calculated as the sum of 2/3 the size of the input layer and the whole size of the output layer. In total, five different

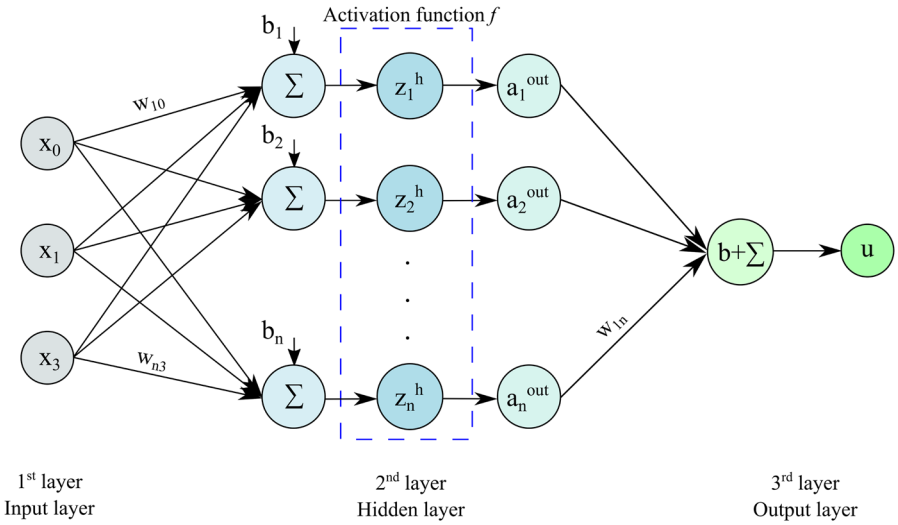

Fig. 4 Multi perceptron layer (MLP) architecture, where: n—total number of neurons in the hidden layer, x_0, x_1, x_3—system state variables and input activation for the hidden layer, $b, b_1, b_2, \ldots b_n$—bias for the current neuron in the hidden layer, $z_1^h, z_2^h, \ldots z_n^h$—value for the current neuron in the hidden layer, $a_1^{out}, a_2^{out}, \ldots a_n^{out}$—input activation for the output layer, u—output (control)

numbers of neurons (3, 5, 10, 30, and 50) are tested. The first value, equal to 3, is based on the rule of thumb, whereas the other four are chosen arbitrarily, basing on authors' experience.

The activation function could be imagined as a filter that processes the values going through it and scales the output into the proper range. Without any activation function applied, the NN could learn only linear transformations. One of the exemplary activation functions used for the hidden layer is Rectified Linear Unit (ReLU), considered one of the most efficient due to its good resistance to vanishing gradients [29]. In this research, we also use the sigmoid (logistic) and hyperbolic tangent functions. The latter is preferred, since its gradients are not restricted to vary in the specified direction [30]. Moreover, when the output returned by the sigmoid function is close to zero, caused by the highly negative inputs, the process of the neural network prolongs and the probability of getting stuck in some local minima is higher [31]. For the output layer, a default option is a linear function (commonly used for regression problems). In this study, we test both: linear and sigmoid.

As the architecture of the NN model, including the different number of neurons as well as the activation functions for hidden and output layers, is established, we propose a default MLP model with n neurons in Fig. 4.

The proposed MLP consists of three layers: one input, one hidden, and one output layer. Please note that each layer is fully connected to the previous one via weight coefficients. For the input layer, the three features x_0, x_1, and x_3, referring to the position of the pendulum θ, the velocity of the pendulum $\dot{\theta}$, and the velocity of the capsule system z' respectively, are assigned. In the (single) hidden layer, consisting of n neurons, the activation function f is applied, being one of the following: the ReLU, sigmoid, or hyperbolic tangent. The output layer consists of one neuron. It is represented by the optimal control u of the pendulum capsule drive system being the target of NN model predictions. On the output, the linear and sigmoid function are applied.

Since the aim of the NN is to learn the relationship between the features (system state variables) and the target data (open-loop optimal control), formula (10) is introduced. It presents a way of calculating the output value of the chosen layer (in this case, the hidden layer) with the applied activation function f. Thus, the input for the next layer (in this case, the output layer) is known. This step is repeated till the last connection between the layer and the target is achieved.

$$a_i^{out} = f(z_i^h) = f\left(b_i^h + \sum\nolimits_{j\in\{0,1,3\}} x_j w_{ij}\right) \tag{10}$$

In formula (10), the following notation has been adopted: i—index of the current neuron in the hidden layer, $i \in \{1, \ldots, n\}$, n—total number of neurons in the hidden layer, j—index of the current neuron in the input layer, $j \in \{0, 1, 3\}$, h—hidden layer, f—the activation function applied in the hidden layer (e.g., ReLU, sigmoid, tanh), z_i^h—value for the current

neuron in the hidden layer, b_i^h—bias for the current neuron in the hidden layer, x_j—input for the hidden layer, represented by three system's state variables, a_i^{out}—current input for the output layer, w_{ij}—weight representing the connection between the jth input and the ith neuron in the hidden layer.

3.3 The Neural Network training process

Since the reference dataset (described precisely in Sect. 3.1.) is loaded and checked with respect to the missing values, the considered control features (x_0, x_1 and x_3) and the optimized open-loop control u are assigned to the vector $\boldsymbol{x}$ and the target variable u respectively. We split the dataset randomly into the separate training and test sets. It is worth noting that the smaller the test set, the more inaccurate the estimation of the generalization error becomes [31]. For this, we use 80% of the samples of the aforementioned features to fit the model and the remaining 20% of the unseen data for the performance evaluation. Dataset shuffling is additionally applied in this step in order to obtain the representative training and test sets. It also means that the created model is not determined by the data order. Moreover, it prevents getting stuck in cycles during the cost function optimization [31].

The way the loss gradients are used to update the parameters of the NN is specified by the optimizers [23]. In this research, we consider the adaptive moment estimation (Adam) optimizer combining the advantages of AdaGrad [32] and RMSProp [33] methods. The first one deals efficiently with sparse gradients, whereas the second works well in online as well as non-stationary settings and resolves some problems of the first one. More precise connections between these methods and Adam optimizer are described in [34]. In the chosen algorithm, hyperparameters are equipped with intuitive interpretation and typically do not require any tuning. The individual adaptive learning rates are calculated for different parameters based on the estimation of the first and second moments of the gradient [34].

We train each of the MLP models for 1000 epochs until the lowest generalization error is achieved. Then, the calculated weights and biases are used to predict the target variable of the unseen data. To reduce the risk of the randomness results, the training is repeated three times.

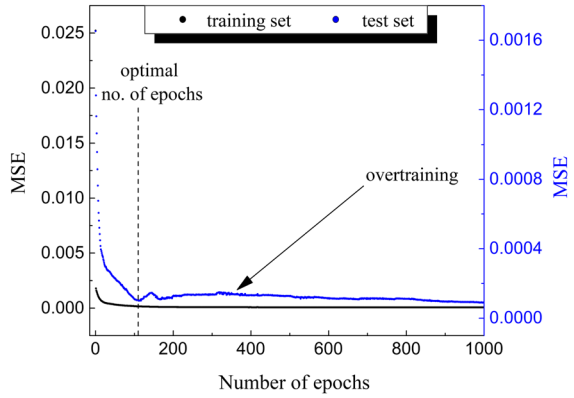

Fig. 5 The loss function value in 1000 epochs for the test set

Based on Fig. 5, showing the changes in the loss function for the training and test set, it can be noticed that the algorithm reaches convergence at the 114th epoch. Crossing that threshold, overtraining appears. To avoid this phenomenon, the loss values reaching the same level are monitored for ten epochs using the *EarlyStopping* class. Then, the training process is subsequently stopped.

The performance of the model is evaluated with regard to the loss function value and data fit on the test dataset. The first one is measured with the use of the Mean Squared Error (MSE) metric. This equals the average value of the Sum of Squared Errors cost function that is minimized to fit the model [31]. The MSE is calculated according to the following equation:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y^i - \hat{y}^i)^2 \tag{11}$$

where n—number of training samples, y^i—the true value of the current sample, $\hat{y}^i$—the predicted value of the sample.

Data fit evaluation of how far the predicted values are from the original ones is described using the coefficient of determination R^2 score, which shows the

Table 1 The NN predictive models' performance results concerning the various activation functions for the hidden and output layers and the different numbers of neurons

No. of neurons	Activation function of the hidden layer	Activation function of the output layer			
		None (linear)		Sigmoid	
		Data fit/loss function score			
		R^2	MSE	R^2	MSE
3	ReLU	0.985	0.002	0.985	0.0019
10		0.996	0.0005	0.995	0.0006
17		0.997	0.0003	0.998	0.0003
30		0.998	0.0002	0.998	0.0003
50		**0.999**	**0.0001**	**0.999**	**0.0002**
3	Sigmoid	0.991	0.0012	0.989	0.0014
10		0.996	0.0005	0.998	0.0003
17		0.997	0.0004	**0.999**	**0.0001**
30		0.997	0.0005	**0.999**	**0.0001**
50		0.998	0.0003	**0.999**	**0.0001**
3	Tanh (hyperbolic tangent)	0.965	0.0045	0.989	0.0015
10		0.996	0.0005	**0.999**	**0.0001**
17		0.998	0.0003	**0.999**	**0.0002**
30		0.998	0.0003	**0.999**	**0.0001**
50		0.996	0.0006	**0.999**	**0.0001**

fraction of response variance captured by the model [31].

4 Numerical results

The performance of NN predictive models has been analyzed for 30 different configurations of the MLP, presented in Sect. 3.2. The analysis included various activation functions for the hidden and output layers, along with changing numbers of neurons. The reasons for selecting the activation functions and the number of neurons have been presented in Sect. 3.2. The performance was measured with the use of MSE and R^2 parameters providing the information about the generalization of the error and data fit respectively. The results are presented in Table 1.

In the second part of the research we consider only NN models with performance measured by the R^2 and MSE equal to 0.999 and 0.0001/0.0002 respectively. Consequently, nine different configurations (bolded in Table 1) reach this result. It is worth noting that the highest scores are mainly obtained for the NN models

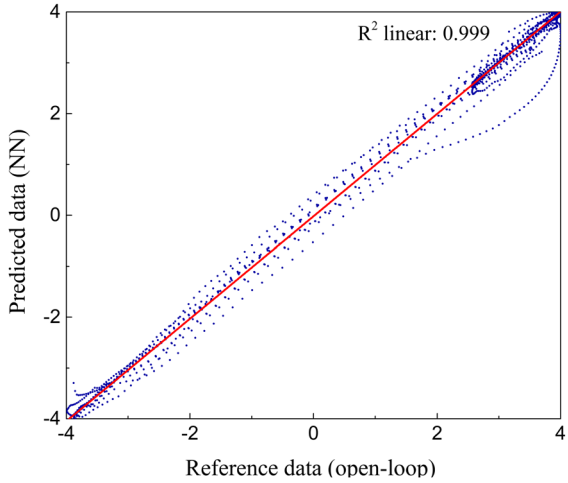

Fig. 6 The linear relationship between the learning and predicted data along with the R^2 coefficient value equal to 0.999 for the NN model consisting of 50 neurons and the applied ReLU and linear activation function for the hidden and output layer, respectively

with sigmoid and hyperbolic tangent activation functions in the hidden layer, whereas in the output one the sigmoid is applied. Moreover, top-scoring performance results with the aforementioned combination of activation functions in layers start from 10 neurons without significant changes during further increases in value. On the other hand, the rule of thumb tested in this study does not give satisfactory results, which could be related to the specific character of the considered system with the small number of input and output data. The process of NN training lasted an average of 100 epochs. However, in this study we do not focus on the training time criteria.

The linear relationship between the reference and predicted data with R^2 coefficient equal to 0.999 is presented in Fig. 6. The results show the satisfactory level of the model fit for the top-scoring performance subject consisting of 50 neurons, along with the applied ReLU and linear activation function in the hidden and output layers.

Parameters of the nine top-scoring performances of NN models achieved in this stage are consequently implemented into the pendulum capsule drive controller. Then, the distance covered by the system in the dimensionless time interval is calculated and presented in Table 2.

Table 2 Distance covered by the pendulum capsule drive for the NN models with the highest performance scores obtained in the preliminary research along with their structure, i.e., the hidden and output layer activation functions, number of neurons, the data fit and loss function scores

Activation function of the hidden layer	Activation function of the output layer	No. of neurons	Data fit score	Loss function score	Covered distance
ReLU	**None (linear)**	**50**	**0.999**	**0.0001**	**6.135**
ReLU	Sigmoid	50	0.999	0.0002	5.998
Sigmoid	Sigmoid	17	0.999	0.0001	6.081
Sigmoid	Sigmoid	30	0.999	0.0001	6.082
Sigmoid	Sigmoid	50	0.999	0.0001	6.073
Tanh	Sigmoid	10	0.999	0.0001	6.115
Tanh	Sigmoid	17	0.999	0.0002	6.079
Tanh	Sigmoid	30	0.999	0.0001	6.076
Tanh	Sigmoid	50	0.999	0.0001	6.089

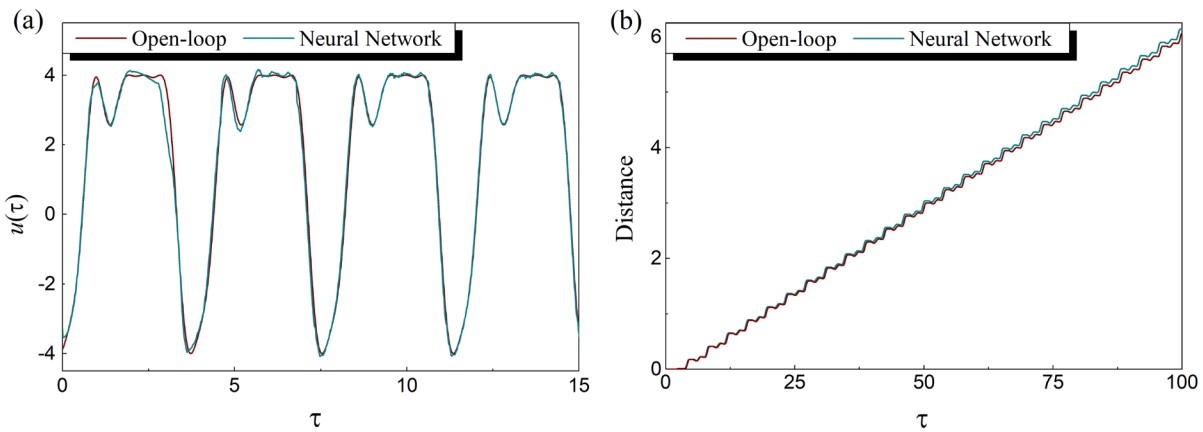

Fig. 7 The optimized control (**a**) with the corresponding distance covered by the pendulum capsule drive (**b**) for the open-loop and NN controller vs. dimensionless time (τ)

The analysis of the distance covered by the pendulum capsule drive in the assumed, dimensionless time interval revealed that the highest score is obtained for the NN model consisting of 50 neurons in the hidden layer, along with the ReLU applied as an activation function and the linear one used for the output. The achieved result is equal to 6.135, with 1.16% higher performance than the one from the open-loop control (6.065). The scores of other NN models differ insignificantly from the reference open-loop optimal control and are not considered in this study. In any case, it is worth observing that all (but one) provide a final displacement that is systematically larger than the reference one of 6.065, although the increment is minor.

The optimized control along with the distance covered by the pendulum capsule drive obtained in the dimensionless time interval for both controllers (open-loop and neural closed-loop), are presented in Fig. 7.

4.1 Perturbations in the system

The controllers' robustness against the variation of parameters of the system has been tested by introducing a variable friction coefficient between the capsule and the underlying surface. It has been assumed that the actual friction coefficient at each point is a uniformly distributed random value:

$$\mu_r \in [\mu - \Delta, \mu + \Delta] \tag{12}$$

Table 3 The results of the distance covered by the system in the environment with varying friction coefficient (Δ) range for the open-loop and neural closed-loop controller. The relative changes between the distance without and under perturbations, for both controllers, along with their performance in the uncertain frictional environment, according to Formulas (13) and (14)

Δ	Open-loop controller		Closed-loop Neural Network controller		Neural Network and open-loop comparison
	Distance without perturbation				
	6.065		6.135		
	Mean (±SD) distance affected by perturbations	Relative changes between non- and perturbed distance (%)	Mean (±SD) distance affected by perturbations	Relative changes between non- and perturbed distance (%)	Relative changes across both controllers' results (%)
0.00	6.065 (±0)	0	6.135 (±0)	0	1.16
0.01	6.069 (±0.0094)	0.07	6.139 (±0.0026)	0.06	1.16
0.02	6.074 (±0.012)	0.15	6.138 (±0.0044)	0.03	1.04
0.03	6.040 (±0.024)	−0.41	6.133 (±0.040)	−0.04	1.54
0.04	6.068 (±0.018)	0.06	6.159 (±0.038)	0.39	1.49
0.05	6.087 (±0.056)	0.37	6.130 (±0.015)	−0.09	0.70
0.06	6.042 (±0.034)	−0.38	6.108 (±0.040)	−0.45	1.10
0.07	5.999 (±0.031)	−1.08	6.139 (±0.033)	0.06	2.33
0.08	6.048 (±0.019)	−0.27	6.087 (±0.057)	−0.79	0.64
0.09	5.978 (±0.037)	−1.44	6.079 (±0.029)	−0.93	1.69
0.10	5.976 (±0.076)	−1.47	6.076 (±0.039)	−0.98	1.67
0.11	5.951 (±0.043)	−1.88	6.162(±0.086)	0.44	3.55
0.12	5.904(±0.054)	−2.66	6.151 (±0.32)	0.25	4.19
0.13	5.846 (±0.050)	−3.61	6.214 (±0.18)	1.28	6.29
0.14	5.873 (±0.11)	−3.16	6.114 (±0.085)	−0.35	4.10
0.15	5.864 (±0.062)	−3.32	6.058 (±0.11)	−1.27	3.31
0.16	5.775 (±0.098)	−4.78	6.179 (±0.068)	0.71	6.99
0.17	5.803 (±0.11)	−4.32	6.020 (±0.38)	−1.88	3.74
0.18	5.638 (±0.044)	−7.05	5.929 (±0.078)	−3.36	5.17
0.19	5.614 (±0.071)	−7.44	5.736 (±0.14)	−6.50	2.19
0.20	5.443 (±0.251)	−10.25	5.584 (±0.12)	−9.00	2.57

where μ is the nominal, assumed value of the friction coefficient and Δ is its maximal, absolute deviation. A different value of μ_r has been drawn for each interval of the capsule's path of motion of the dimensionless length equal to 0.1.

The subject of the evaluation was the NN predictive model with the highest result of the distance covered by the pendulum capsule drive in the interval of dimensionless time τ. The Δ parameter introducing perturbations, was varied from 0.00 to 0.20, thus changing the effective friction coefficient range simultaneously. For each value of the Δ parameter, the test has been repeated 3 times to reduce the risk of random results. The obtained mean scores, along with the corresponding standard deviation (SD) are presented in Table 3.

To better understand the impact of the Δ parameter variation on the distance covered by the pendulum capsule drive for the open-loop and neural closed-loop controller, the relative changes have been calculated (see Table 3). Within the first stage each of the controllers has been evaluated with respect to the changes between the distance without and under perturbations, according to the following formula:

$$\delta_C = \frac{z_p - z_0}{z_0} \cdot 100\% \tag{13}$$

where δ_c—the relative change for the open-loop or neural closed-loop controller (%), z_0—the distance covered by the system under the value of the Δ parameter equal to 0, z_p—the distance covered by the system under the non-zero value of the Δ parameter.

The known values of the distance covered by the pendulum capsule drive in the uncertain frictional environment led us to the performance comparison of both controllers in these conditions. The relative changes of the distance under perturbations between the open-loop and neural closed-controller have been calculated as follows:

$$\delta_P = \frac{z_{NN} - z_{OL}}{z_{OL}} \cdot 100\% \tag{14}$$

where δ_P—the relative change between the open-loop and neural closed-loop controller (%), z_{OL}—the distance covered by the system under the perturbations, for the open-loop controller, z_{NN}—the distance covered by the system under the perturbations, for the neural closed-loop controller.

Please note that the parameters of the open-loop controller were calculated with regard to the Formula (8) and presented in (9). On the other hand, the NN controller ones originate from the computation performed in the second stage of this research, according to Formula (10).

The distance covered by the pendulum capsule drive in the uncertain frictional environment presented in Table 3 shows that the NN controller is more resistant to friction changes than the open-loop controller. The first significant decrease in the distance appears much earlier in the open-loop controller, starting from a value of 0.07 corresponding to 0.10–0.13 in the NN one. The higher the Δ parameter value, the more noticeable the difference. The close-up look of the covered distance consideration for open-loop and NN controllers is presented in Fig. 8.

The impact of the varying coefficient of friction on the distance covered by the pendulum capsule drive calculated within the use of relative changes is presented in Figs. 9 and 10. The main observation is that the higher the Δ parameter value, the bigger decrease in the distance, especially for the open-loop control.

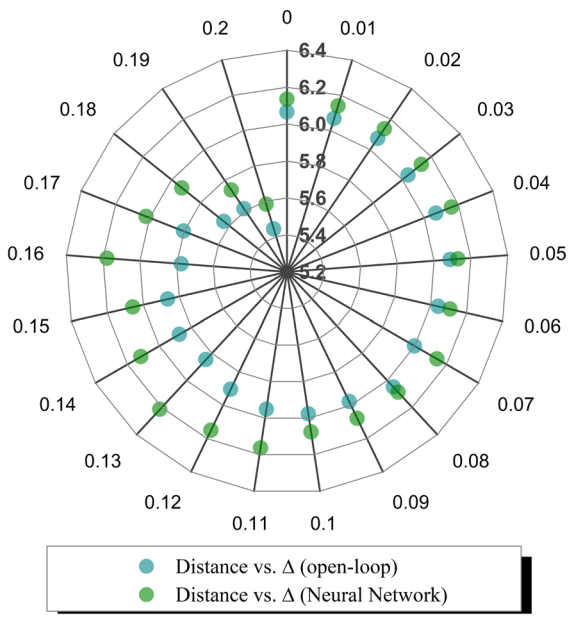

Fig. 8 Distance covered by the pendulum capsule drive in the uncertain frictional environment vs. the open-loop and neural network controller

It is worth nothing that as Δ varies from 0 to 0.16, the relative change of distance is of the order of 1% when the NN controller is applied. In contrast, the open-loop controller suffers a four times larger loss of efficiency for the same increase of Δ. The greatest noticeable change for both controllers appears with a Δ parameter value of 0.20, resulting in a 10.25% and 9% decrease in the distance for the open-loop controller and the NN one respectively. While for large values of the Δ parameter the performance of both controllers decreases (although to a different extent, as said), it is interesting to note that, quite surprisingly, for very small values of the Δ parameter the maximum distance increases with respect to the unperturbed case $\Delta = 0$ suggesting a kind of beneficial effect of uncertainties on μ.

The analysis of perturbed distances comparison performed for both controllers proves that the neural network controller is more resistant to changes occurring in the environment with a varying range of the coefficient of friction. Hence, the most significant discernible increase in performance is equal to + 7%.

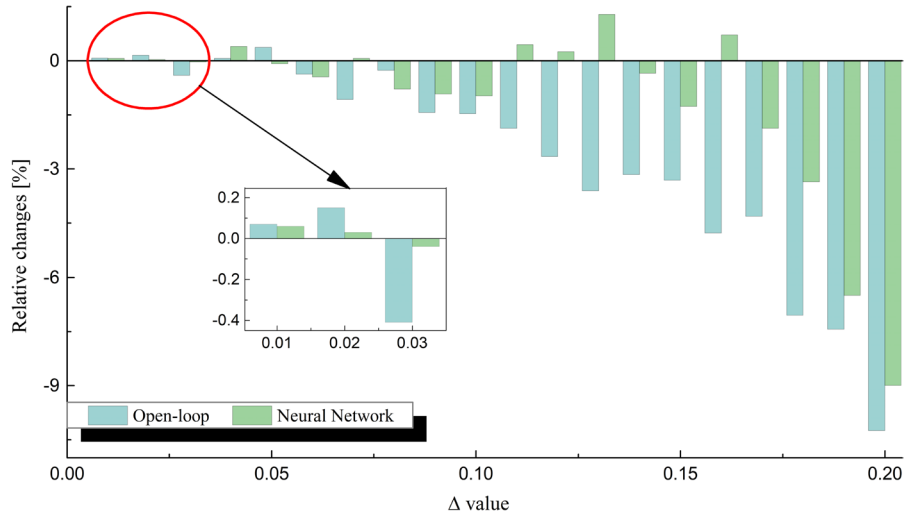

Fig. 9 Relative changes between the distance without and under perturbations (Δ) for the open-loop and NN controller

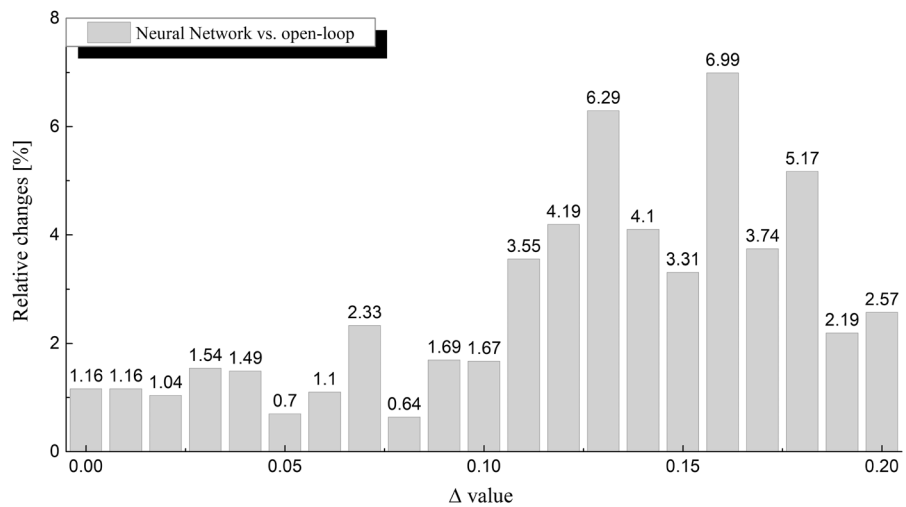

Fig. 10 The performance comparison of the open-loop and neural closed-loop controller in the environment with varying friction coefficient introduced by Δ parameter value

5 Summary and conclusions

In this study, a novel approach to the pendulum capsule drive control with the use of a neural network (NN) as a closed-loop controller is presented. The novelty in this research is the use of an optimized open-loop control function as the basis of the designed controller. The dependence between the output of the open-loop controller and the system state variables is determined by the NN.

One of the major aims of this research was to test and evaluate the robustness of the novel controller compared with the original open-loop control. Thus, the study was divided into three parts. In the preliminary research we created a default architecture of a NN predictive model that was tested for various configurations concerning different numbers of neurons, as well as the activation functions for hidden and output layers. Within this stage the performance of each model was measured with the R^2 and MSE tools. This allowed the selection of nine NN models with the highest correlation score between the reference and predicted data. The parameters of NN models gathered in this training have been implemented in the pendulum capsule drive simulated controller to calculate the distance covered by the system in the dimensionless time interval. The highest distance was achieved for the NN model consisting of 50 neurons

in the hidden layer along with the ReLU used as an activation function and a linear one applied for the output.

Subsequently, the top-performing artificial NN model was tested in the last part of the study, considering the robustness of the controller and its reliability in the uncertain frictional environment introduced by the varying friction coefficient range between the capsule shell and the underlying surface. Without any perturbations the neural network controller achieved a 1.16% higher performance than the open-loop. The variations of the coefficient of friction range proved the NN more resistant to the perturbations occurring in the system with a maximum of +7% advantage over the open-loop controller. In fact, the changes between the distance without and under perturbations occurred much slower, and remained at the 1% level much longer. Meanwhile, in the open-loop controller the increasing value was constantly observed, exceeding some trial scores achieved in the NN controller at least 4 times.

Results presented in this study confirm that the NN controller works more efficiently compared to the original open-loop controller and proves the higher level of robustness in an environment where perturbations occur. It seems that the neural closed-loop controller could be an alternative option to the classic ones in many applications of the mechanical field, especially for non-smooth and discontinuous systems (as the one considered in this work). Moreover, it could significantly simplify the closed-loop controllers' systems design where only the open-loop control is available.

Acknowledgements This study has been supported by the National Science Centre, Poland, PRELUDIUM Programme (Project No. 2020/37/N/ST8/03448). This study has been supported by the National Science Centre, Poland under project No. 2017/27/B/ST8/01619. This paper has been completed while the first author was the Doctoral Candidate in the Interdisciplinary Doctoral School at the Lodz University of Technology, Poland. Sandra Zarychta, Marek Balcerzak and Prof. Andrzej Stefański declare that their research described in this work has been carried out during their scientific visit to Marche Polytechnic University, Ancona, in collaboration with Prof. Stefano Lenci.The majority of the work has been done during the first author's three months stay at Marche Polytechnic University, Ancona. We would like to thank Małgorzata Balcerzak for her invaluable assistance in proofreading this paper.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Data availability All the scripts created within this study are available in the research data linked to this paper [35].

Appendix

In this appendix the proposed method of training the closed-loop neural network controller by means of an existing open-loop optimal control function is presented on a simple, mathematical pendulum. The purpose of using such a trivial system is to explain the principle of the proposed algorithm as simply as possible. In this manner, the essence of the method can be easily shown without all the nuisances that

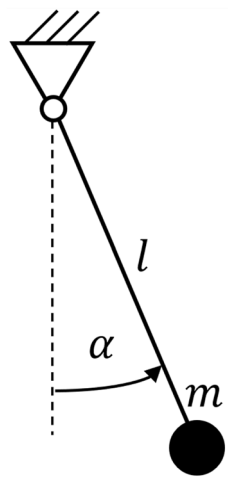

Fig. 11 Simple pendulum system

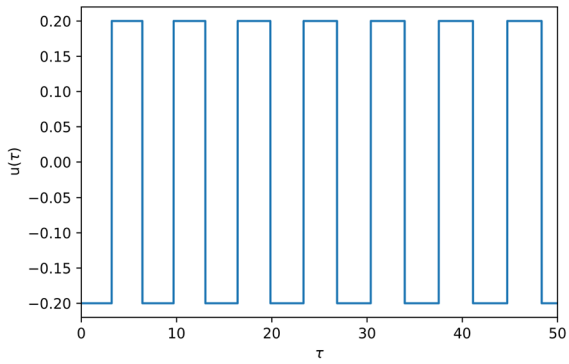

Fig. 12 Open loop optimal control of the pendulum system (Eq. 16)

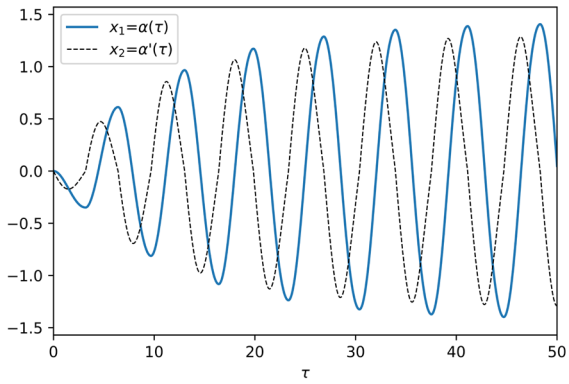

Fig. 13 Optimal trajectory of the pendulum system (Eq. 16)

may appear when the controlled object is more complex. The scheme of the analyzed system is presented in Fig. 11.

The equation of motion of the simple pendulum with a driving torque M and with a linear damping $-c\dot{\alpha}$ is as follows.

$$ml^2\ddot{\alpha} = -mgl\sin(\alpha) - c\dot{\alpha} + M \tag{15}$$

The following dimensionless quantities are introduced: $\tau = \omega t, \alpha' = \frac{d\alpha}{d\tau} = \frac{\dot{\alpha}}{\omega}, \alpha'' = \frac{d^2\alpha}{d\tau^2} = \frac{\ddot{\alpha}}{\omega^2}, \zeta = \frac{c}{2m\omega l^2}$, $u = \frac{M}{m\omega^2 l^2}$ where $\omega = \sqrt{\frac{g}{l}}$. Application of these symbols in the formula (15) yields the following dimensionless form of the equation of motion.

$$\alpha'' = -\sin(\alpha) - 2\zeta\alpha' + u \tag{16}$$

It is assumed that the control u is bounded: $u \in [-u_{max}, u_{max}]$. Moreover, the following

	t	x1	x2	u
0	0.00	0.00000	0.000000	-0.2
1	0.01	-0.00001	-0.001998	-0.2
2	0.02	-0.00004	-0.003992	-0.2
3	0.03	-0.00009	-0.005981	-0.2
4	0.04	-0.00016	-0.007966	-0.2

Fig. 14 The header and the initial rows of the dataset containing the optimal open-loop control $u(\tau)$ (Fig. 12) and the corresponding optimal trajectory (Fig. 13)

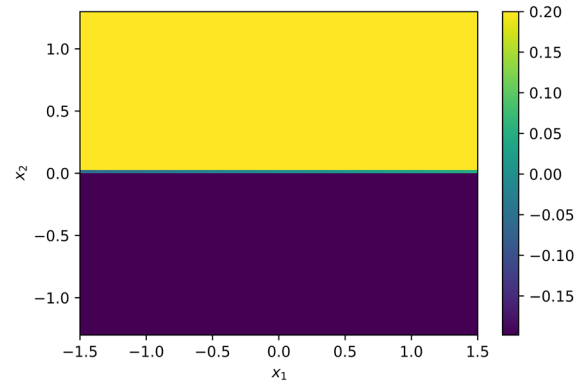

Fig. 15 Control values $u(x_1, x_2)$ returned by the neural network trained using the dataset presented in Fig. 14. The neural network easily detected that the control $u(x_1, x_2)$ must be simply equal u_{max} for positive x_2 and $-u_{max}$ for negative x_2

dimensionless values of the parameters have been adopted: $\zeta = 0.1, u_{max} = 0.2$.

Suppose that the objective is to maximize the amplitude of pendulum's swinging in a shortest possible time. Obviously, such goal is accomplished by increasing the mechanical energy of the pendulum which in turn is done by applying the maximum possible torque to the pendulum in the direction of its instantaneous rotational velocity. This strategy is a direct consequence of the work-energy principle [26]. In such a manner, the optimal control (see Fig. 12) along with the optimal trajectory (see Fig. 13) of the pendulum system (Eq. 16) can be easily generated. Now, the control $u(\tau)$, presented in Fig. 12, is a function of the dimensionless time τ only and can be considered an optimal open-loop control. All the generated data can be connected in one dataset containing

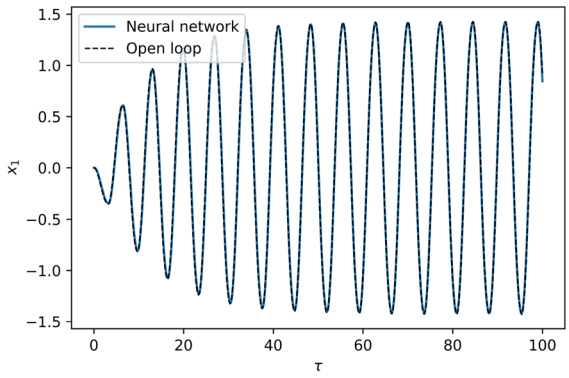

Fig. 16 Trajectories of the state variable $x_1 = \alpha(\tau)$ obtained with use of the neural network controller and by means of the open-loop optimal control. It can be noticed that both plots are indistinguishable

4 columns: dimensionless time τ, state variables $x_1 = \alpha(\tau), x_2 = \alpha'(\tau)$ and the optimal open-loop control $u(\tau)$ (see Fig. 14).

Now, the closed-loop optimized controller can be created by training a neural network on the dataset presented in Fig. 14. In such network, the state variables $x_1 = \alpha(\tau), x_2 = \alpha'(\tau)$ are the input values and the corresponding values of the control function $u(\tau)$ are the output values (the target of our prediction). In such a manner, the dependence of the optimal control on time is removed. Instead, an approximate closed-loop optimal control, i.e., the function $u(x_1, x_2)$, can be obtained.

In the presented case, the underlying dependence between u and x_1, x_2 is trivial:

u attains the maximum value u_{max} for positive x_2 and the minimum value $-u_{max}$ for negative x_2, being independent from x_1. Therefore, almost any one-layer neural network can detect such an obvious relationship. In the numerical tests, a one-hidden-layer network has been used. Similarly as in the main part of the paper, the hidden layer contained neurons with "ReLU" activation function whereas the output layer uses the sigmoid [24, 31]. It turned out that even *one* neuron in the hidden layer enables detection of the proper relationship. The heatmap depicting the control value u for different states x_1, x_2 is presented in Fig. 15 whereas comparison of trajectories with the open-loop optimal control and the closed-loop neural network-based one is presented in Fig. 16.

References

1. Liu Y, Yu H, Yang TC (2008) Analysis and control of a capsubot. IFAC Proc Vol 17(1 PART 1):2–3. https://doi.org/10.3182/20080706-5-KR-1001.1613
2. Guo B, Liu Y, Birler R, Prasad S (2019) Self-propelled capsule endoscopy for small-bowel examination: proof-of-concept and model verification. Int J Mech Sci 174(December):2020. https://doi.org/10.1016/j.ijmecsci.2020.105506
3. Huda MN, Yu H (2015) Trajectory tracking control of an underactuated capsubot. Auton Robots 39(2):183–198. https://doi.org/10.1007/s10514-015-9434-3
4. Liu Y, Wiercigroch M, Pavlovskaia E, Yu H (2013) Modelling of a vibro-impact capsule system. Int J Mech Sci 66:2–11. https://doi.org/10.1016/j.ijmecsci.2012.09.012
5. Páez Chávez J, Liu Y, Pavlovskaia E, Wiercigroch M (2016) Path-following analysis of the dynamical response of a piecewise-linear capsule system. Commun Nonlinear Sci Numer Simul 37:102–114. https://doi.org/10.1016/j.cnsns.2016.01.009
6. Liu Y, Islam S, Pavlovskaia E, Wiercigroch M (2016) Optimization of the vibro-impact capsule system. Stroj Vestnik/Journal Mech Eng 62(7–8):430–439. https://doi.org/10.5545/sv-jme.2016.3754
7. Maolin L, Yao Y, Yang L (2018) Optimization of the vibro-impact capsule system for promoting progression speed. MATEC Web Conf 148:1–5. https://doi.org/10.1051/matecconf/201814810002
8. Liu Y, PáezChávez J, Guo B, Birler R (2020) Bifurcation analysis of a vibro-impact experimental rig with two-sided constraint. Meccanica 55(12):2505–2521. https://doi.org/10.1007/s11012-020-01168-4
9. Liu Y, Páez Chávez J, Zhang J, Tian J, Guo B, Prasad S (2020) The vibro-impact capsule system in millimetre scale: numerical optimisation and experimental verification. Meccanica 55(10):1885–1902. https://doi.org/10.1007/s11012-020-01237-8
10. Liu P, Yu H, Cang S (2018) On the dynamics of a vibro-driven capsule system. Arch Appl Mech 88(12):2199–2219. https://doi.org/10.1007/s00419-018-1444-0
11. Liu P, Yu H, Cang S (2015) On periodically pendulum-driven systems for underactuated locomotion: a viscoelastic jointed model. In: 2015 21st international conference on automation and computing: automation, computing and manufacturing for new economic growth, ICAC 2015, no September, pp 11–12. https://doi.org/10.1109/IConAC.2015.7313936
12. Liu P, Huda MN, Tang Z, Sun L (2020) A self-propelled robotic system with a visco-elastic joint: dynamics and motion analysis. Eng Comput 36(2):655–669. https://doi.org/10.1007/s00366-019-00722-3
13. Liu P, Yu H, Cang S (2018) Optimized adaptive tracking control for an underactuated vibro-driven capsule system. Nonlinear Dyn 94(3):1803–1817. https://doi.org/10.1007/s11071-018-4458-9
14. Liu P, Yu H, Cang S (2018) Trajectory synthesis and optimization of an underactuated microrobotic system with dynamic constraints and couplings. Int J Control

Autom Syst 16(5):2373–2383. https://doi.org/10.1007/s12555-017-0192-7
15. Liu P, Yu H, Cang S (2019) Adaptive neural network tracking control for underactuated systems with matched and mismatched disturbances. Nonlinear Dyn 98(2):1447–1464. https://doi.org/10.1007/s11071-019-05170-8
16. Zarychta S, Sagan T, Balcerzak M, Dabrowski A, Stefanski A, Kapitaniak T (2022) A novel, Fourier series based method of control optimization and its application to a discontinuous capsule drive model. Int J Mech Sci 219(January):107104. https://doi.org/10.1016/j.ijmecsci.2022.107104
17. Yan Y, Zhang B, Páez J, Liu Y (2022) Optimising the locomotion of a vibro-impact capsule robot self-propelling in the small intestine. Commun Nonlinear Sci Numer Simul 114:106696. https://doi.org/10.1016/j.cnsns.2022.106696
18. Liao M, Zhang J, Liu Y, Zhu D (2022) Speed optimisation and reliability analysis of a self-propelled capsule robot moving in an uncertain frictional environment. Int J Mech Sci 221(November 2021):107156. https://doi.org/10.1016/j.ijmecsci.2022.107156
19. Baril C, Yacout S, Clément B (2011) Design for Six Sigma through collaborative multiobjective optimization. Comput Ind Eng 60(1):43–55. https://doi.org/10.1016/j.cie.2010.09.015
20. Asafuddoula M, Singh HK, Ray T (2015) Six-sigma robust design optimization using a many-objective decomposition-based evolutionary algorithm. IEEE Trans Evol Comput 19(4):490–507. https://doi.org/10.1109/TEVC.2014.2343791
21. Zhang X, Lu Z, Cheng K, Wang Y (2020) A novel reliability sensitivity analysis method based on directional sampling and Monte Carlo simulation. Proc Inst Mech Eng Part O J Risk Reliab 234(4):622–635. https://doi.org/10.1177/1748006X19899504
22. Zhou L, Cai G, Yang J, Jia L (2010) Monte-Carlo simulation based on FTA in reliability analysis of door system. In: 2010 2nd international conference on computer and automation engineering (ICCAE), vol 5, no 1, pp 713–717. https://doi.org/10.1109/ICCAE.2010.5451338
23. Chollet F (2018) Deep learning with Phyton
24. Géron A (2017) Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow (2019, O'reilly)
25. Raschka S, Mirjalili V (2019) Python machine learning: machine learning & deep learning with Python, Scikit-Learn and TensorFlow 2, 3rd edn, no January 2010
26. Taylor JR (2005) Classical mechanics. University Science, p 2005
27. Heaton JT (2005) Introduction to neural networks with Java. Heaton Research Inc
28. Cybenko G (1989) Approximation by superpositions of a sigmoidal function. Math Control Signals Syst 2(4):303–314. https://doi.org/10.1007/BF02551274
29. Glorot X, Bordes A, Bengio Y (2011) Deep sparse rectifier neural networks. J Mach Learn Res 15
30. Sharma S, Sharma S, Athaiya A (2020) Activation functions in neural networks. Int J Eng Appl Sci Technol 04(12):310–316. https://doi.org/10.33564/ijeast.2020.v04i12.054
31. Raschka S (2015) Python machine learning: unlock deeper insights into machine learning with this vital guide to cutting-edge predictive analytics. www.packtpub.com
32. Duchi J, Hazan E, Singer Y (2010) Adaptive subgradient methods for online learning and stochastic optimization. In: COLT 2010—23rd international conference on learning theory, vol 12, pp 257–269
33. Tieleman T, Hinton G (2012) Lecture 6.5-RMSProp, COURSERA. Neural Netw Mach Learn Tech Rep
34. Kingma DP, Ba JL (2015) Adam: a method for stochastic optimization. In: 3rd international conference on learning representations ICLR 2015—conference track proceedings, pp 1–15
35. Zarychta S, Balcerzak M, Denysenko V, Stefanski A, Dabrowski A, Lenci S (2022) Optimization of the closed-loop controller of a discontinuous capsule drive using a neural network. Mendeley Data, V1. https://doi.org/10.17632/rvcxkbnx3h.1