future internet

MDPI

Article

Electric Drive with an Adaptive Controller and Wireless Communication System

Mateusz Malarczyk, Mateusz Zychlewicz, Radoslaw Stanislawski and Marcin Kaminski *

Department of Electrical Machines, Drives and Measurements, Faculty of Electrical Engineering, Wroclaw University of Science and Technology, 19 Smoluchowskiego St., 50-372 Wroclaw, Poland
* Correspondence: marcin.kaminski@pwr.edu.pl

Abstract: In this paper, the problem of the remote control of electric drives with a complex mechanical structure is discussed. Oscillations of state variables and control precision are the main issues found in such applications. The article proposes a smart, IoT-enabled controller, which allows remote communication with a drive. To solve the problem of speed oscillations and to make the system robust to parameter uncertainty, an adaptive controller with two neural networks is designed. First, numerical tests are conducted in a Matlab/Simulink environment to examine the operation of the proposed control strategy. Afterwards, the obtained results are verified in a laboratory setup equipped with a 0.5 kW electric motor. Remote access is provided by a low-cost, ARM-based ESP32 microcontroller. Usually, virtual instruments used to communicate with remote devices require specific software, which may be expensive and pose compatibility problems. Therefore, the main contribution of the article is the creation of a low-cost, web-based Human-Machine Interface (HMI) with an asynchronous server utility provided by the ESP32 that allows remote control and data acquisition of electric drive state variables.

Keywords: IoT; remote control; adaptive speed control; neural networks

Citation: Malarczyk, M.; Zychlewicz, M.; Stanislawski, R.; Kaminski, M. Electric Drive with an Adaptive Controller and Wireless Communication System. *Future Internet* **2023**, *15*, 49. https://doi.org/10.3390/fi15020049

Academic Editors: Chan Hwang See, Kelvin Anoh, Yousef Dama, Simeon Keates and Raed A. Abd-Alhameed

Received: 30 December 2022
Revised: 19 January 2023
Accepted: 21 January 2023
Published: 28 January 2023

1. Introduction

1.1. Problem Formulation

The problem of precise speed control is still prevalent in industrial electrical drives with a complex mechanical structure [1]. Usually, analysis of control schemes is performed under the assumption that the connection between the motor and the load is rigid and does not negatively affect the drive. The elasticity of the shaft and gears can become a source of disturbance and oscillations [2,3]. It can lead to poorer product quality, damage to mechanical parts of the drive, or problems with stability. In the literature, two machines coupled with a flexible element are often referred to as a two-mass system and are the subject of numerous publications [4–6]. Examples of two-mass systems observed in the industry include rolling mills [7], wind turbines [8,9], and robotic manipulators [10–12]. Because of the requirement for high dynamics and the presence of oscillations, such applications must be controlled with more intricate control schemes.

PID cascade control is one of the most frequently used strategies due to its low computational power demand and the simplicity of practical implementation [13,14]. The gains of the controller can be adjusted with the pole distribution method or the modulus and symmetry criteria. However, the structure does not take any information about the load machine. It is not suitable for applications with varying speeds between the motor and the load [15]. In such cases, the problem can be solved by providing additional feedback from other variables [16] or using a state feedback controller [17,18]. In a state controller, all state variables (angular speed of the motor and load, and the torsional torque) are connected back to the speed controller to compensate for the elasticity of the coupling element. Nevertheless, both of these strategies suffer one significant disadvantage—they

are susceptible to changes in plant parameters. The gains of these controllers are set to work correctly for a selected operating point. The error margin is narrow; even slight changes in plant parameters or a sudden appearance of disturbances can lead to loss of control precision (reference speed tracking) [19].

To address the issue of parameter uncertainty, adaptive control schemes are often applied. Among the possible approaches, gain scheduling [20] and self-tuning methods [21] can be listed. One of the most efficient solutions is model reference adaptive control (MRAC) [22,23]. In MRAC, the difference between the model of the system and the actual output is fed to the adaptation algorithm, then new values of the controller are established. In this way, the controller remains correctly updated. Additionally, only approximate values of the plant parameters are needed to establish the initial value of the gains. There are two main downsides to this strategy [24]. The adaptation algorithm must be carefully chosen to ensure the stable operation of the drive, which is a difficult task. Furthermore, the use of the reference model can nullify the original non-linearities of the plant. This effect can negatively influence the dynamics of the system if the model is selected inappropriately.

The model-free approach relies on the use of intelligent structures, such as neural networks, which are widely used in different tasks related to electrical drives. They can be implemented to estimate state variables [25] or the state of charge (SoC) of electric vehicle batteries [26]. Deep learning algorithms are used to facilitate fault detection [27]. Neural structures are also applied to the task of controlling the angular speed of electric motors [28]. In this concept, gradient algorithms are used to adjust the weights of the network during drive operation. Adaptive properties are achieved through the use of online training. The design process includes the selection of the network structure and the learning coefficient. The stability of neural controllers can be proven through the use of the Lyapunov stability theorem [29]. The learning coefficient can be optimized using swarm inspired algorithms.

Effective optimization of the initial parameters of a neural network can significantly improve the drive's performance. This is especially important because no training is performed prior to starting the drive. With a wrong initiation point, the control structure can struggle to quickly adapt to the current state of the control structure. Incorrect setting of the weights and the learning rate can cause an oscillatory response [30]. The selection of more appropriate values can be facilitated by using nature-inspired metaheuristic optimization algorithms [31]. The major advantage of such algorithms lies in the fact that they do not need information about the cost function derivative to find better values. As a trade-off, reaching the global minimum is never guaranteed. One of the most common algorithms from this group is Particle Swarm Optimization (PSO) [32]. Other examples include Cuckoo Search (CS) [33,34], the Grey Wolf Optimizer (GWO) [35,36], Ant Colony Optimization (ACO) [37–39], or the Bat algorithm [40]. In this work, the Artificial Bee Colony (ABC) algorithm [41] was implemented to facilitate the proper selection of the parameters responsible for scaling the input of the proposed neural network.

Distributed measurement systems are developed to establish a connection between multiple units via the internet [42–44]. Remote access to information from a vastly distant instrument is often required. To obtain the measurements, virtual instruments (VIs) are often implemented. The VIs are usually created with the use of software dedicated to instrument control, such as LabVIEW™, LabWindows™/CVI, and Keysight VEE™. Using these environments forces the user to install the required software and may cause VI compatibility problems. To resolve this issue, a web-based Human-Machine Interface was created to collect and visualize the data acquired during the drive operation.

1.2. Related Work

Different approaches can be observed in the published papers. Most of the research is focused on the control structures and does not provide sufficient discussion on experimental data acquisition and visualization. On the other hand, some papers present complex designs of the front-end applications for data visualization without the examination of plant control. The authors mainly deal with proper data presentation [45,46], interaction

with the user by using haptic-feedback features [47,48] and the optimization of required hardware resources [49,50]. Both of the approaches lack the integrity of the process, the control structure, and its state variables. This makes them difficult to compare directly, as different targets are defined for various purposes. Because the idea of Industry 5.0 is becoming very popular, it is essential to focus on interactive smart products and enable human-machine and human-robot collaboration [51–53]. This can only be achieved with user-friendly interfaces and smart control structures (using artificial intelligence). Thus, the presented research addresses the smart control structure of the neural speed controller extended with online data presentation.

In this paper, a neural adaptive control structure is applied to control the speed of a two-mass drive. The experimental results obtained with an incremental encoder are sent to the end user via an ESP32 device. The visualization of the data is performed by a web-based HMI. In the first section, the mathematical description of the plant and the control structure is presented. Next, the working principle of the ADALINE neural predictor implemented in the proposed controller is discussed. The ABC algorithm is described later. The following section focuses on the numerical tests of the control structure performed in Matlab™/Simulink. Afterward, the experimental results are presented. They serve as a practical verification of the simulation studies. Then, ESP32-based data transmission and visualization HMI are shown. The article is concluded with a brief analysis of the results obtained and final comments.

2. Mathematical Model of the Control Structure and the Controller

The discussed control structure consists of two neural networks that are applied to a mechanical system with elastic joints. Such a construction is often referred to as a two-mass system. Expressions describing the dynamic properties of the drive can be formulated using a set of the following equations [4,54,55]:

$$T_1 s\omega_1 = T_e - T_s - T_{f1} \tag{1}$$

$$T_2 s\omega_2 = T_s - T_L - T_{f2} \tag{2}$$

$$T_c s T_s = \omega_1 - \omega_2 \tag{3}$$

where T_1, T_2, and T_c are the mechanical time constants of the motor, load and shaft, respectively, T_e, T_s are the electromagnetic torque and shaft torque, ω_1 and ω_2 are the rotor speeds of the motor and load, respectively, T_L is the load torque, and T_{f1} and T_{f2} are non-linear functions that describe the friction present in a real drive [30]:

$$T_{fi} = (c|\omega_i| + d)\mathrm{sgn}(\omega_i) \tag{4}$$

where c and d represent the viscous and Coulomb friction coefficients, respectively. It was assumed during the design process of the control structure that the current control loop (inner loop) is simplified to a first-order element and can be represented as a simple transfer function:

$$G_i(s) = \frac{1}{T_p s + 1}. \tag{5}$$

A simplified schematic of the control structure is presented in Figure 1. The first neural network presented (in blue in Figure 1) is a network composed of an input layer, a hidden layer, and an output layer. In addition, a hidden layer has a recurrent connection, which is called a context layer. The second one (represented in orange in Figure 1) represents the neural predictor of the feedback signal. Both networks work to achieve a reliable performance by the plant according to a reference value. The Elman network is described below, as the neural predictor is in Section 3.

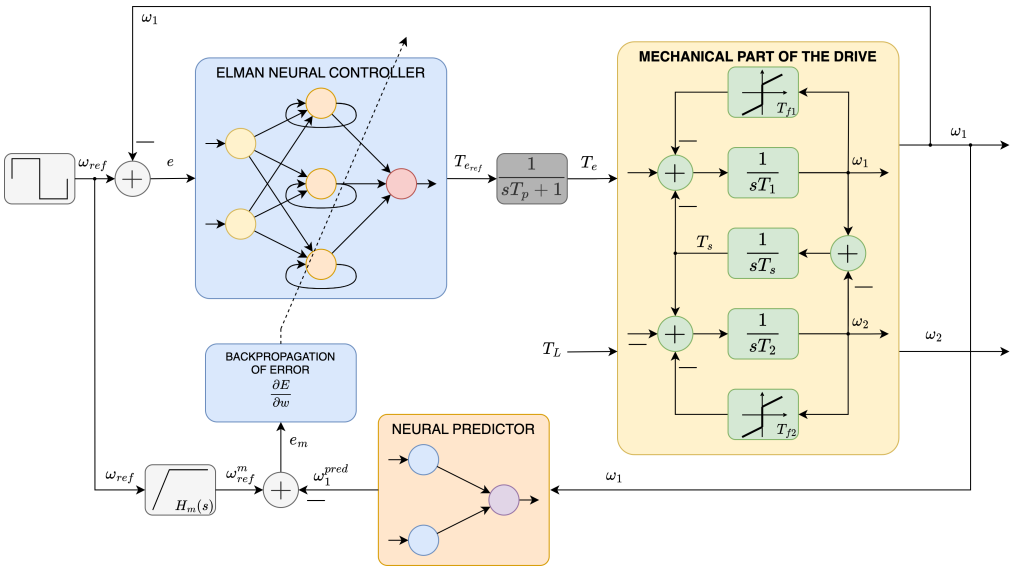


Figure 1. Simplified schematic diagram of adaptive control with two neural elements.

The Elman neural network consists of I inputs, N hidden neurons, and an output neuron that generates the reference value of the electromagnetic torque—T_{eref}. The input layer is a linear layer with i neurons. Equations for the input can be written as:

$$h_i^1(k) = x_i(k) \tag{6}$$

$$z_i^1(k) = f\left(h_i^1(k)\right) = h_i^1(k) \tag{7}$$

where h, z are the input and output node, respectively, i is the number of the input, x is the input signal, and k represents samples. The hidden layer, with a hyperbolic activation function, processes signals from the previous layer. In addition, in the Elman network, recurrent signals from context layer are introduced, which act as an additional memory that can learn supplementary features of the plant [56]. The equations for the hidden layer input and output signal can be written as follows:

$$h_j^2(k) = \sum_{i,j=1}^{N} \left(w_{ij}^1(k) z_i^1(k) + w_j^c(k) z_j^c(k)\right) \tag{8}$$

$$z_j^2(k) = f\left(h_j^2(k)\right) = \tanh\left(h_j^2(k)\right) \tag{9}$$

and for the context layer:

$$h_j^c(k) = z_j^2(k-1) \tag{10}$$

$$z_j^c(k) = f\left(h_j^c(k)\right) = h_j^c(k) \tag{11}$$

where w is a weight corresponding to the given layer and neuron and c stands for the context layer indicator. The output layer is also a linear layer, so its output can be presented similarly to the input layer:

$$h^3(k) = \sum_{j=1}^{N} w_j^2(k) z_j^2(k) \tag{12}$$

$$z^3(k) = f\left(h^3(k)\right) = h^3(k) = y_{nn} \tag{13}$$

The Elman neural network, presented in Figure 2 below, operates in on-line mode, which implies that it learns during the operation of the drive.

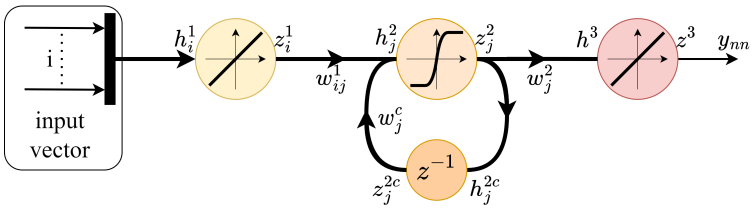


Figure 2. Detailed connections in a neural network with a context layer.

In this work, the gradient descent technique is used—weights between the first and second layer, second and third, and in the context layer, are subject to update their value during the running process of the drive. To calculate the new weights correctly, the cost function (14) should be defined (it should be differentiable). It tells the network how the weights should change and is the crucial part of the algorithm.

$$E(k) = \frac{1}{2}\left(\omega_{ref}^m(k) - \omega_1(k)\right)^2 \tag{14}$$

To ensure the stable work of a control structure, the reference speed signal is connected using the element described with the following formula [57]:

$$H_m(s) = \frac{\omega_r^2}{s^2 + 2\omega_r\xi s + \omega_r^2} \tag{15}$$

where ω_r is the resonant frequency and ξ is the damping coefficient, which changes the overshoot in case of ξ and defines the dynamics of the drive (ω_r). After calculating the cost function, updates for the weights can be processed. It should be started from the last layer, then the second to last layer is calculated, etc. This process is called backpropagation. The updates for parameters between the hidden and the output layer can be formulated as follows:

$$\Delta w_j^2(k) = \frac{\partial E(k)}{\partial w_j^2(k)} = \underbrace{\frac{\partial E(k)}{\partial y_{nn}(k)}}_{e_m(k)} \underbrace{\frac{\partial y_{nn}(k)}{\partial h^3(k)}}_{1} \underbrace{\frac{\partial h^3(k)}{\partial w_j^2(k)}}_{z_j^2(k)} \tag{16}$$

The context layer weight update takes a similar form:

$$\Delta w_j^c(k) = \frac{\partial E(k)}{\partial w_j^c(k)} = \underbrace{\frac{\partial E(k)}{\partial y_{nn}(k)}}_{e_m(k)} \underbrace{\frac{\partial y_{nn}(k)}{\partial h^3(k)}}_{1} \underbrace{\frac{\partial h^3(k)}{\partial z_j^2(k)}}_{w_j^2(k)} \underbrace{\frac{\partial z_j^2(k)}{\partial w_j^c(k)}}_{f'\left(h_j^2(k)\right)} \tag{17}$$

At last, update values for the weights between the input and the hidden layer should be calculated:

$$\Delta w_{ij}^1(k) = \frac{\partial E(k)}{\partial w_{ij}^1(k)} = \underbrace{\frac{\partial E(k)}{\partial y_{nn}(k)}}_{e_m(k)} \underbrace{\frac{\partial y_{nn}(k)}{\partial h^3(k)}}_{1} \underbrace{\frac{\partial h^3(k)}{\partial z_j^2(k)}}_{w_j^2(k)} \underbrace{\frac{\partial z_j^2(k)}{\partial w_{ij}^1(k)}}_{f'\left(h_j^2(k)\right)} \tag{18}$$

where term $f'\left(h_j^2(k)\right)$ means the calculation of the derivative of the compound function. New weights should be calculated before proceeding to the next step [58]:

$$w_j^2(k+1) = w_j^2(k) - \eta\Delta w_j^2(k) \tag{19}$$

$$w_j^c(k+1) = w_j^c(k) - \eta\Delta w_j^c(k) \tag{20}$$

$$w_{ij}^{1}(k+1) = w_{ij}^{1}(k) - \eta \Delta w_{ij}^{1}(k) \tag{21}$$

3. ADALINE Predictor

In the previous section, it was mentioned that the cost function requires motor speed in its calculations (14). As all signals are based on the current sample of the reference and motor speeds, if the motor speed is known in advance, the neural controller can take that into account and be more precise and robust in its work (due to faster reaction). To predict the value of a motor speed, a simple ADALINE model is proposed. The ADALINE model consists of simple linear function neurons. Data processing is performed similarly to classic neural networks—input values are multiplied by weights, then the sum of these values is taken. The advantage of this approach is the use of a model that can predict (with the proper connections of the signals) data without an actual plant model. The cost function [59] and output signal for an ADALINE model can be defined similarly to (13), (14):

$$E_{pred}(k) = \frac{1}{2}\Big(y_{pred}(k) - in(k)\Big)^2 \tag{22}$$

$$y^{pred}(k) = \sum_{\gamma=1}^{N_a} x_\gamma(k) w_\gamma^{pred}(k) \tag{23}$$

where x is the processed state variable, w^{pred} is the weight associated with a given input, $y^p red$ is the output of the predictor, in is the model input, and N_a is the number of inputs.

The update process principle remains the same as in the previous chapter. However, this time, the calculations are performed to optimize w^{pred} (it should be noted that the value of the learning rate β is a small value):

$$w_\gamma^{pred}(k+1) = w_\gamma^{pred}(k) + \beta\left[\left(\sum_{\gamma=1}^{N_a} x_\gamma(k) w_\gamma^{pred}(k) - in(k)\right) x_\gamma^{pred}(k)\right] \tag{24}$$

It can also be written as:

$$w_\gamma^{pred}(k+1) = w_\gamma^{pred}(k) + \beta \Delta w_\gamma^{pred}(k) \tag{25}$$

The update function is based on a gradient of the defined cost function for a predictor:

$$\Delta w_\gamma^{pred} = \frac{\partial E_{pred}(k)}{\partial w_\gamma^{pred}(k)} = \left[\left(\sum_{\gamma=1}^{N_a} x_\gamma(k) w_\gamma^{pred}(k) - in(k)\right) x_\gamma^{pred}(k)\right] \tag{26}$$

The neural predictor can be organized so that the output of one unit is the input of the second. Such a combination increases the prediction horizon by a number of connected units. If, for example, k units were connected as presented, the output of the entire predictor system would predict k future steps, which is presented in detail in [60]. An example system of connected predictor nodes is presented in Figure 3.

A stable training procedure for a neural network should allow small and smooth gradient changes and minimization of the defined error, E_{pred}. This process can be written as follows:

$$E_{pred}\Big(w_\gamma^{pred}(k)\Big) < E_{pred}\Big(w_\gamma^{pred}(k-1)\Big) \tag{27}$$

After modification and using Equation (25), it can be expressed as:

$$E_{pred}\Big(w_\gamma^{pred}(k) + \Delta w_\gamma^{pred}(k)\Big) < E_{pred}\Big(w_\gamma^{pred}(k-1)\Big) \tag{28}$$

The gradient value is presented with expression (26). The prediction error can also be defined:

$$e(k) = y_{ref}^{pred}(k) - y^{pred}(k) \tag{29}$$

The first part in (27) is a reference prediction value, which is achieved for the optimal weights value:

$$y_{ref}^{pred}(k) = \sum_{\gamma=1}^{nan} x_\gamma(k) w_{\gamma_{opt}}^{pred}(k) \tag{30}$$

After inserting Equations (30) and (23) into (29), it yields:

$$e(k) = w_{\gamma opt}^{pred} x_\gamma(k) - w_\gamma^{pred}(k) x_\gamma(k) = \Theta(k) x_\gamma(k) \tag{31}$$

where Θ is a weight adaptation error, and it can be expressed as:

$$\Theta(k) = w_{\gamma opt}^{pred} - w_\gamma^{pred}(k) \tag{32}$$

It should be noted that $w_{\gamma opt}^{pred}$ does not depend on time or samples because it is an optimal value. After combining expressions (25) and (32), the weight update error can be written as:

$$\Theta(k+1) = (1-\beta)\left(w_{\gamma opt}^{pred} - w_\gamma^{pred}(k)\right) \tag{33}$$

A proper analysis of the stability of the neural network should define a Lyapunov function. It can be defined as follows:

$$L(k) = \Theta^2(k) \tag{34}$$

This value should be minimized during the learning process, so for each iteration, the value of *L(k)* should be reduced:

$$L(k) < L(k-1) \tag{35}$$

Using expression (33), the previous equation can be rewritten as:

$$\left[\left(w_{\gamma opt}^{pred} - w_\gamma^{pred}\right)(1-\beta)\right]^2 < \left(w_{\gamma opt}^{pred} - w_\gamma^{pred}\right) \tag{36}$$

From the last inequality, a learning rate to ensure stability can be easily found. It is also known that a small value of a learning rate tends to increase the time needed to adapt the parameters, and a higher value may cause higher jumps, which result in spikes in the output signal. Depending on the application, an adaptive learning rate or a small constant value should be considered.

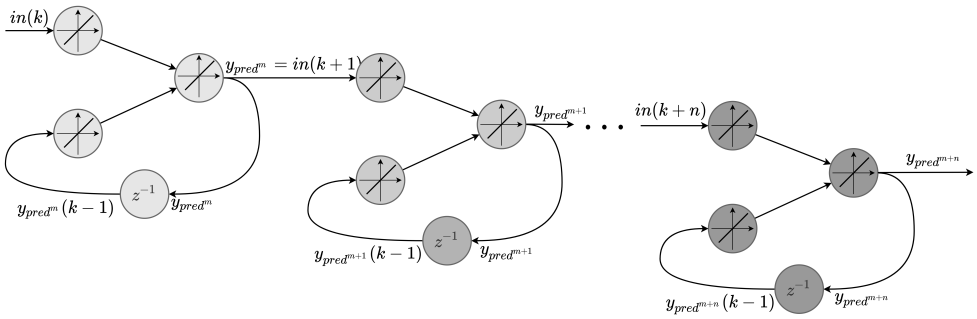


Figure 3. Concept of a series connection of the ADALINE-based elements.

4. Utilization of Artificial Bee Colony Algorithm

Artificial Bee Colony (Figure 4) is an algorithm inspired by a swarm, which consists of one queen, some male bees called drones, and many female workers. Bees can be divided into three main groups—employed, onlookers, and scouts [61]. The task of each member

of the group is different, but all of them are working for a greater good—their queen and the hive. Before any optimization is performed, the initialization process for the algorithm takes place.

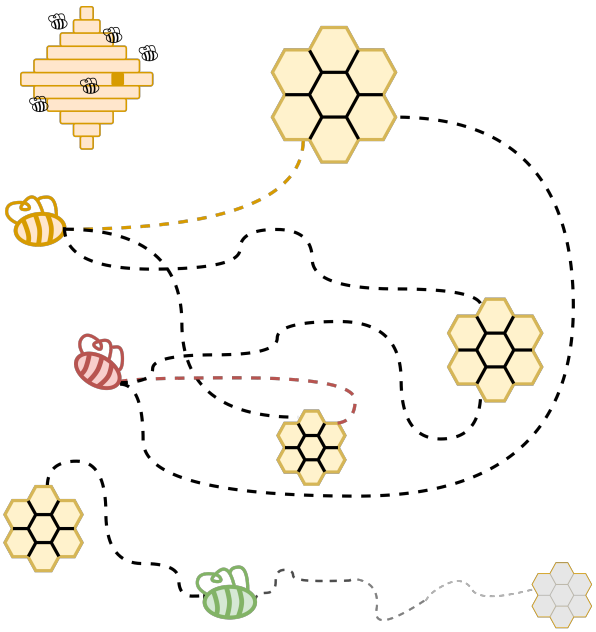


Figure 4. Visualization of the ABC algorithm.

A population of N bees is randomly generated using a pseudo-random generator it is schematically presented in the Figure 4 as a bee hive. Any member of a population should be constrained to the limits of the problem that is solved:

$$x_{ij} = x_i^{lb} + \text{rand}(0,1)(x_i^{ub} - x_i^{lb}) \tag{37}$$

where i is the dimension of a problem, j is the number of an individual bee, and x^{lb} and x^{ub} are the lower and upper bounds of the considered task. In the next step, the fitness function for the whole population is evaluated according to the equation:

$$fit_{ij} = \begin{cases} \frac{1}{1+f_{ij}} & , f_{ij} \geq 0 \\ 1 + |f_{ij}| & , f_{ij} < 0 \end{cases} \tag{38}$$

where f_{ij} is a fitness function defined by a user, and fit_{ij} is a benchmark defined in an algorithm itself [62]. Next, the first main part takes place—the employed bees phase (yellow bee), throughout which employed bees are looking for a new solution, using the expression:

$$x_{ij}^{new}(iter) = x_{ij}(iter) + \phi_{ij}\Big(x_{ij}(iter) - x_{kj}(iter)\Big) \tag{39}$$

where x_{ij}^{new} is a new solution found by an employed bee, x_{kj} is a randomly selected solution other than x_{ij}, $iter$ is a number of a current iteration of an algorithm, and ϕ_{ij} is a random number in a range $(-1, 1)$. The newly found solution x_{ij}^{new} is then compared to the old one x_{ij} according to the fitness value of each solution (38). If a new value is better (lower) than the previous one, it is exchanged so that the better value takes the place of the old one. However, if x_{ij}^{new} has a higher value, it is deleted from the population (the old value of a x_{ij} is kept in the population), and the abandoned solution counter (AC) is incremented. The

next phase starts with a probability calculation which tells if a bee should be exploited in this part. It is done according to a simple equation:

$$p^{OB} = \frac{fit_j}{\sum_{k=1}^{N} fit_k} \tag{40}$$

In the onlooker bees phase (red bee), the solutions with the highest probability are further exploited using an equation similar to (39):

$$x_{ij}^{new}(iter) = x_{ij}^{p_{OB}}(iter) + \phi_{ij}\left(x_{ij}^{p_{OB}}(iter) - x_{kj}(iter)\right) \tag{41}$$

where $x_{ij}^{p_{OB}}$ is a solution candidate for an onlooker phase with a chosen probability. This takes place until all onlookers find their solution—the food source. AC is also incremented in this phase. In the scout bees phase (green bee), the abandonment counter can be used. If its value exceeds the set limit, the bee becomes a scout bee, and then it is looking for a new source of food for its hive. In such a case, the Equation (37) is used. The end of the optimization process is evaluated by fulfilling at least one of the following criteria: reaching the maximum number of iterations, exceeding the calculation time, or obtaining small changes in the best value. The complexity of the ABC algorithm is proportional to the number of dimensions of the problem and the size of population, as well as the number of iterations of the main loop of the algorithm. The whole process is presented in Figure 5.

In this paper, the ABC was used to optimize the coefficients in the input layer of a neural network controller. Examples of different values of k_e and k_I are shown in Figure 6. Only one variable was changed at a time. The results prove the necessity of correct selection of the gains. The optimization was performed using the following fitness function consisting of three parts:

$$f = F_e + F_{de} + F_T \tag{42}$$

$$F_e = \int_{t=0}^{t_{sim}} t^2\left(\omega_{ref}(t) - \omega_1(t)\right)^2 dt \tag{43}$$

$$F_{de} = \int_{t=0}^{t_{sim}} \zeta \frac{d}{dt}\left(\omega_{ref}(t) - \omega_1(t)\right) dt \tag{44}$$

$$F_T = \int_{t=0}^{t_{sim}} \chi \frac{d}{dt} T_e dt \tag{45}$$

where t_{sim} is the duration of the simulation, and ζ and χ are empirically selected coefficients to achieve the desired speed response of the drive.

The maximum iteration count for an algorithm was set to 20, with the size of the population equal to $[20 \times D]$, where $D = 2$ is the dimension of the optimization task. Changes in the fit objective function can be observed in Figure 7 below. The process was started with random values for both variables. The lower bound was set to [0.1 0.1], lower values caused the process to throw an error, and an upper limit was set to [50 100]; this should ensure that the final values were in the proper range.

Exemplary transients gathered during the optimization process are depicted below in Figure 8. During the first iteration, high-frequency oscillations are visible during the simulation. It is much improved in the 5th iteration of the ABC—high overshoot is present in the initial phase of operation, and high noise is visible. The step response is refined during the optimization, which can be seen in the last iteration—high dynamics of the response with no oscillations in both static and dynamic states of the drive.

Figure 5. The Artificial Bee Colony algorithm (different phases are denoted with the same colors as in Figure 4).

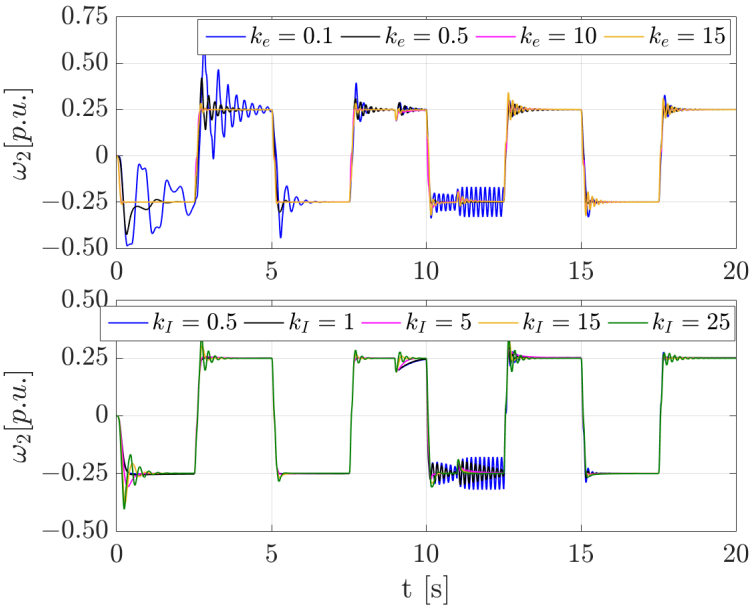


Figure 6. Influence of the input parameters of the neural network (k_e and k_I) on the final variable of the drive.

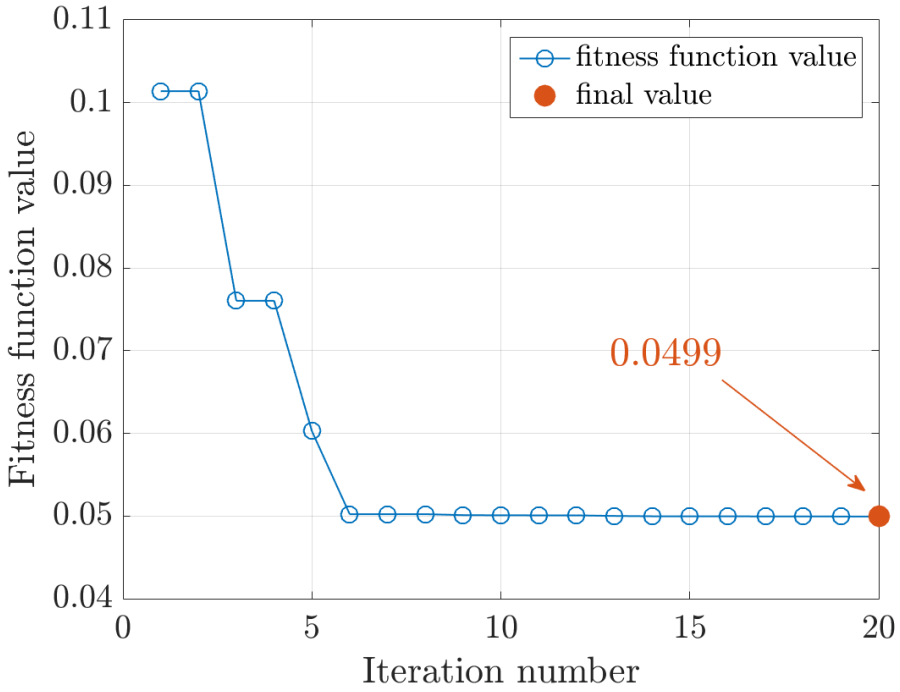


Figure 7. Fitness function values during the optimization process.

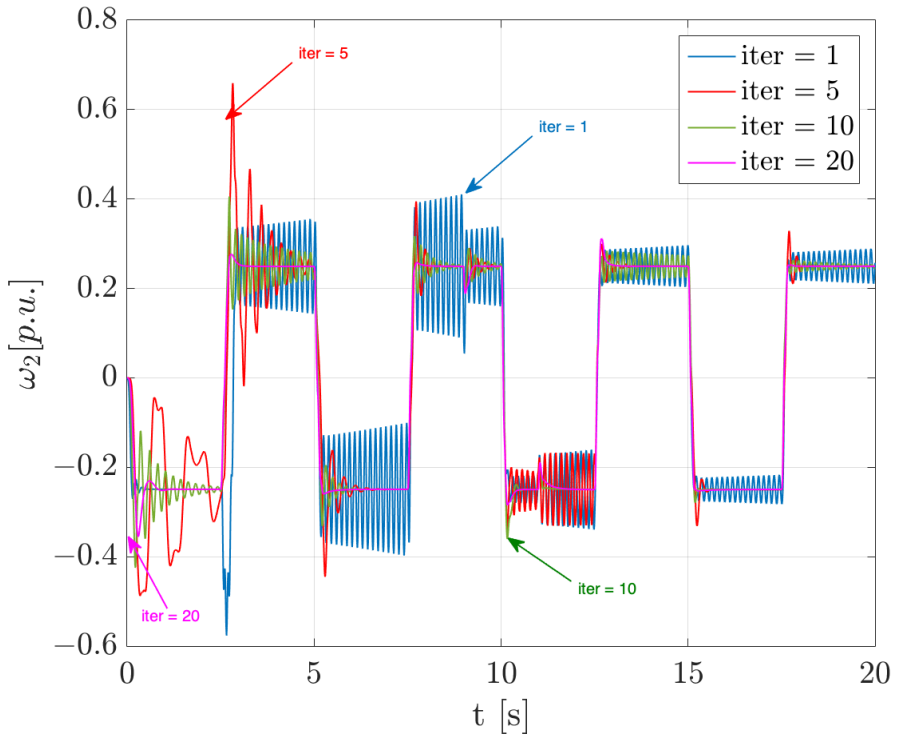


Figure 8. Transients of the load speed of the drive during the optimization process.

5. Simulation Tests

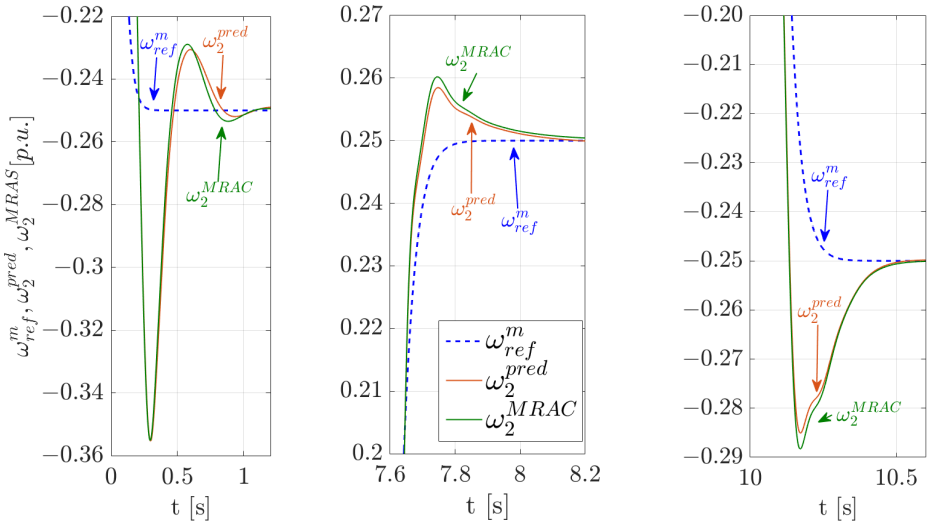
After the optimization of the gains of the input layer of the neural controller, tests of the optimized control structure were performed. Firstly, the work of the predictor was verified. The optimized control structure with a predictor applied was compared to the regular MRAC structure (with no predictor). All the tests were conducted for 20 seconds. The load torque was applied in the 9th second of the drive operation, and it was switched off in the 11th. Selected time frames from the simulation were chosen in the comparison. In

all parts of Figure 9, the load speed of the drive, after the predictor application, has a lower value of overshoot than the MRAC one. In Figure 9a, the startup process of the drive is presented; at first, the speeds match, and after some adaptation of the neural controller, the settling time is lower. After the adaptation of both networks (the Elman controller and the ADALINE predictor), it can be clearly seen that the latter one works better just before the load is switched on (Figure 9b) and after performing a reversion with a load torque applied (Figure 9c).

The prediction element is applied to the learning algorithm. For the test purposes, it was deactivated to check the difference. The fitness functions of the both structures are presented in Table 1.

Table 1. Fitness function comparison of analyzed control structures.

Structure	Fitness Function Values
MRAC	0.0525
MRAC with predictor	0.0499



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

Figure 9. Comparison of a control structure with and without a predictor: (**a**) starting operation of the drive, (**b**) and (**c**) selected parts of the operation with reversions.

Next tests, after the comparison, were performed for a different moment of inertia added to the different parts of the drive. Firstly, the nominal parameters were tested (transients of speeds and torques are presented in Figure 10). There is a small overshoot observed after the start of the drive. Due to the update operation of the internal weights of both neural networks, it is minimized after one reversion. The time constants of the motors in industrial applications (e.g., robotic arms) are changing under the operation. Figure 11 presents the transients gathered in the second test, where the load time constant T_2 was increased. No apparent changes are visible in the speeds other than a lower error of speed after applying the load torque, which was presumed. Higher values of torques are observable due to the fact that the controller tries to achieve the same speed as in the nominal parameter case, but higher values of inertia imply higher amplitude of torques.

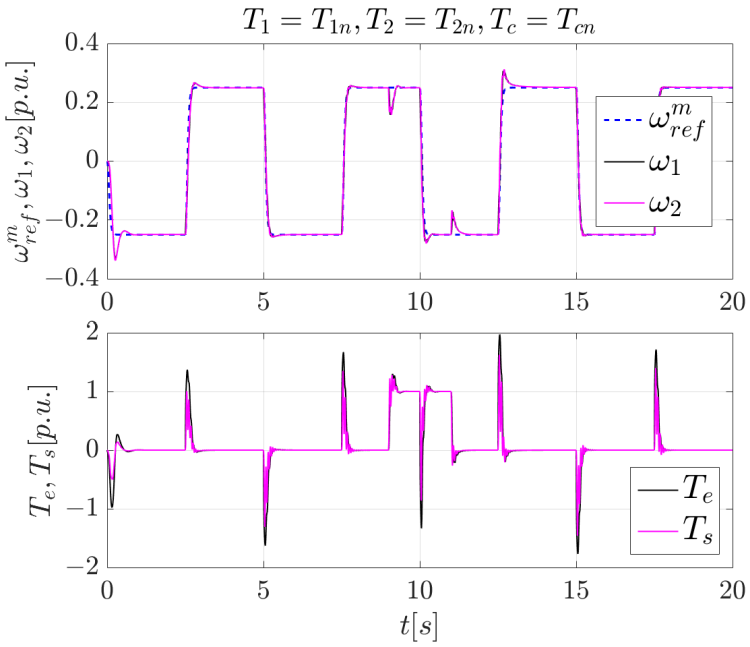


Figure 10. Transients of the speeds (w^m_{ref}, ω_1 and ω_2) and torques (T_e and T_s) of the two-mass drive for nominal parameters of the drive.

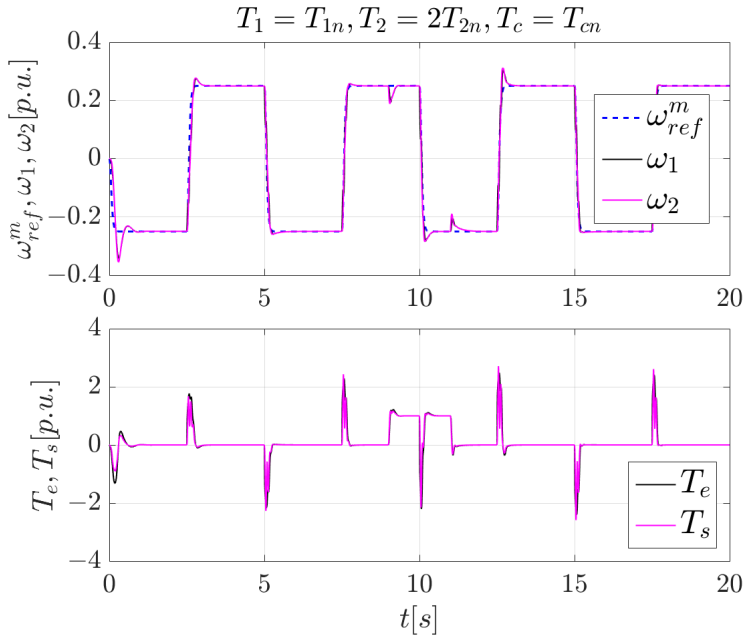


Figure 11. Transients of the speeds (ω^m_{ref}, ω_1 and ω_2) and torques (T_e and T_s) of the two-mass drive with an increased load time constant ($T_{2n} = 2T_{2n}$).

6. Experimental Verification

6.1. Description of the Laboratory Setup

After all the simulation tests were carried out, experimental verification on the laboratory stand took place. The drive consists of two DC motors coupled together with 600 mm long steel shaft. The stiffness time constant can be changed by exchanging the different shaft with different diameters, and additional flywheels can be added on the motor to increase the time constant of the load motor. It should be noted that the type of motor used in the experimental bench is insignificant since the work is focused around the

mechanical part of the drive. Thus, the same tests can be carried out using two coupled induction motors.

LEM sensors are introduced to measure the currents, and incremental encoders are also present on both machines for speed measurements. The drive is powered by an H-bridge, and an additional resistor is added to ensure the proper working of the load motor. The control algorithm was uploaded to a dSpace 1103 card with a DSP by a PC with a Windows operating system. The code was generated with use of a Matlab Embedded Function, which drastically simplifies the process of compiling the uploading the Simulink code to the dSpace card. All the data were observed in the part of the dSpace software called ControlDesk, where the virtual panel was created. The nominal parameters of the setup are presented in Table 2, and the laboratory setup is shown in Figure 12.

Table 2. Nominal parameters of the drive.

Parameter	Value
Motor nominal power	500 W
Load nominal power	500 W
Shaft length	600 mm
Shaft nominal diameter	5 mm
Encoder impulse	36,000 ppr

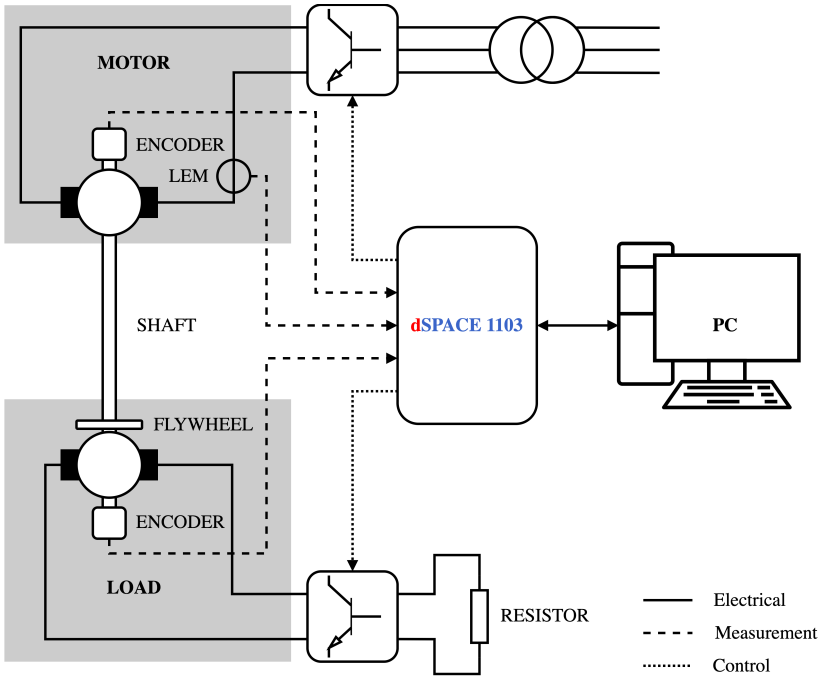


Figure 12. The laboratory setup.

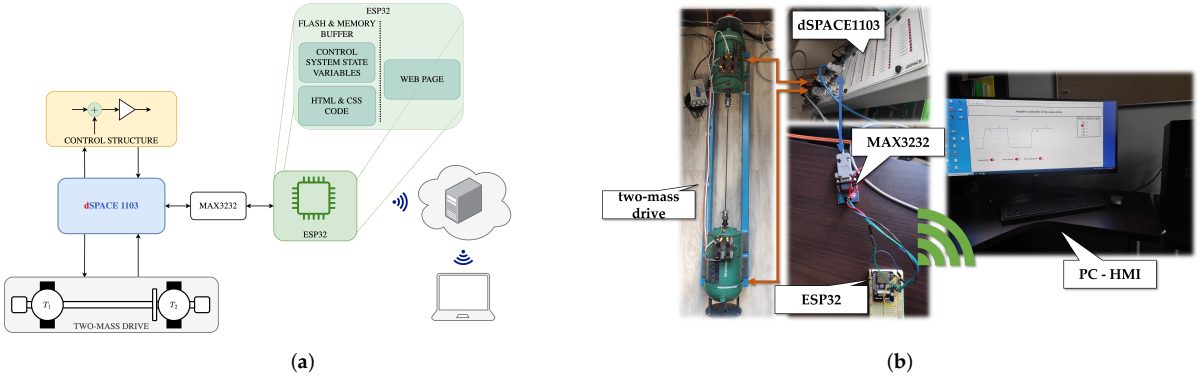
6.2. Remote Data Visualization

The design of the control system was extended with a remote Human-Machine Interface. The main task for the HMI was the visualization of the system state variables; the user interface additionally provides access to basic control, such as selecting the speed set-point and enabling/disabling the drive with a start&stop switch. In comparison to known HMI systems, which in most cases require special, licensed software and specialized hardware (e.g. additional sensors or measurement cards) for data acquisition and visualization [63–65], the proposed system broadcasts the desired data to any network-connected device. The developed system consists of the power electronics and sensors that were required for the drive operation. The state variables were extracted from the control hardware via the serial port and processed with a low-cost, highly adaptable dual-core

ESP32 microcontroller. The ESP32 incorporates a wide range of communication modules, both wired (I2C, I2S, SPI, UART) and wireless (Bluetooth Classic, Bluetooth Low Energy (BLE) and Wi-Fi), which makes it a universal tool that can be implemented in a variety of applications [66–68]. The embedded Wi-Fi module was used to provide remote access for end users. A simple web page was generated, thus enabling data monitoring through any device, such as mobile phones, tablets, or laptop PCs. The scenario assumes the situation with ESP32 working as an internet-connected web server (Figure 13). It enables cloud-based drive monitoring from any destination around the world—the approach especially valuable for maintenance staff in international companies. Not only industrial solutions can use the IoT-enabled machinery, but also research centers and universities may conduct experiments remotely. This is an essential in the home office and remote learning reality of pandemic. Because of the increasing necessity of ensuring data safety, advanced encryption is required. The access can be granted only for the authenticated users (3rd party services are no required). A parameter comparison between typical industrial HMI solutions and the device utilized in this article is presented in Table 3.

Table 3. Comparison of different devices providing HMI functionality [69–71].

Device	Siemens Simatic Panel Basic	ESP32	Omron NB-Series HMI
Power consumption	3 W	≈1 W	≈5 W
Screen size	up to 15″	end-user device dependent	up to 10″
Connectivity interfaces	Ethernet + Profinet	USB + Wi-Fi + Bluetooth	Ethernet + USB + RS232
User memory size	10 MB	4 MB (up to 16 MB)	128 MB
HMI design language/ required software	TIA PORTAL WINCC	HTML/CSS/C++	NB-Designer
Price	$$$	$	$$$



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

Figure 13. Remote data visualization with ESP32: dataflow (**a**), laboratory equipment (**b**).

The network framework was developed using an asynchronous web server ESP32 utility. It allows generating a responsive web page design based on HTML syntax. The communication between control system acquisition module and the web server in asynchronous mode was established in a request-based manner. Every reload of the web page or user activity (such as switch toggle) generates a defined request to access microcontroller variables. The asynchronous operation reduces the workload of the processor, as the data exchange was only conducted at the user request. The CSS style file and HTML code are stored in flash memory of the ESP32 with the utilization of SPIFFS (SPI Flash File System). Such an approach allows to generate a more complex graphical interface in comparison to the minimal code constrained by program memory. Moreover, the web page file was replaced in the microcontroller memory without the interference with the network configuration and data acquisition logic.

The designed HMI web page was linked with the system during the experimental tests. The speed transient was plotted from the buffered samples. The length of the single data

package was affected with the sampling frequency of the control structure. The captured screen can be found in Figure 14.

The most important element was the speed transient that covered most of the web page area. The Plotly library was used for this purpose. Not only did it allow data plotting, but it also enabled manipulation over the figure—zooming, panning and data cursor attaching. Furthermore, the interface provides the possibility to save the speed transient as an image.

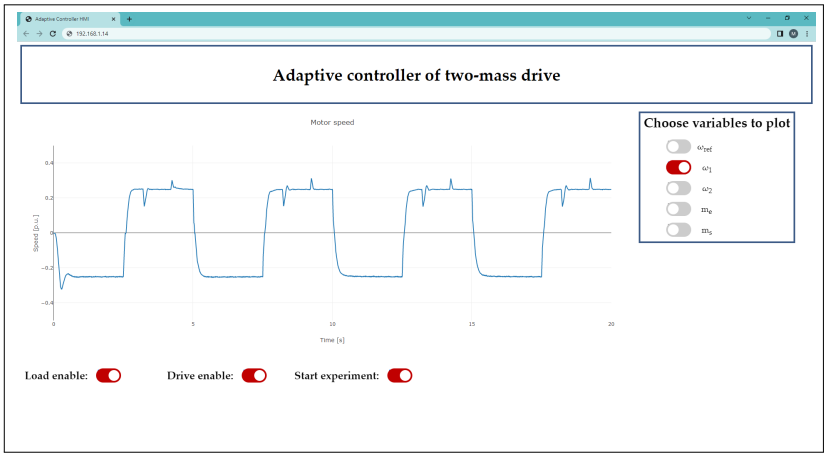


Figure 14. Graphical user interface.

6.3. Experimental Results

Experimental results are presented in Figures 15 and 16—for the nominal parameters of the drive and increased time constant of the load machine ($T_2 = 2T_{2n}$), respectively. Reference speed transients are assumed to be the same as in the simulation tests to reproduce the results best. Cycling reversions of the speed occur every 5 s, and the amplitude of the reference speed was set to 25% of the nominal value to prevent the limitation of the current. The load torque was also switched during the operation of the drive.

The experimental results are comparable to those obtained in the simulations. Overshoot can be noticed in the initial part of the operation. Reaction to the load is fast, and the drop of the speed is small. The results from Figure 16 show that increasing the load time constant slightly increases the overshoot during the first two reversions. Other than that, the transients are similar and comparable to the simulations for both tests.

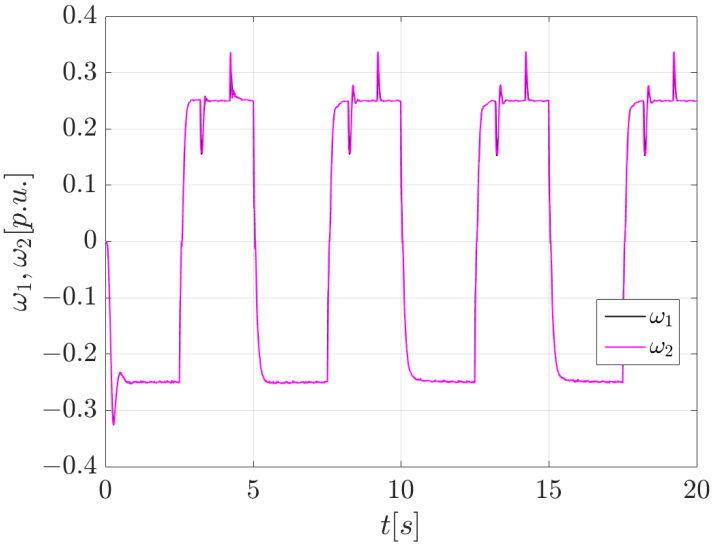


Figure 15. Transients of state variables—motor and load speeds (nominal parameters of two-mass system).

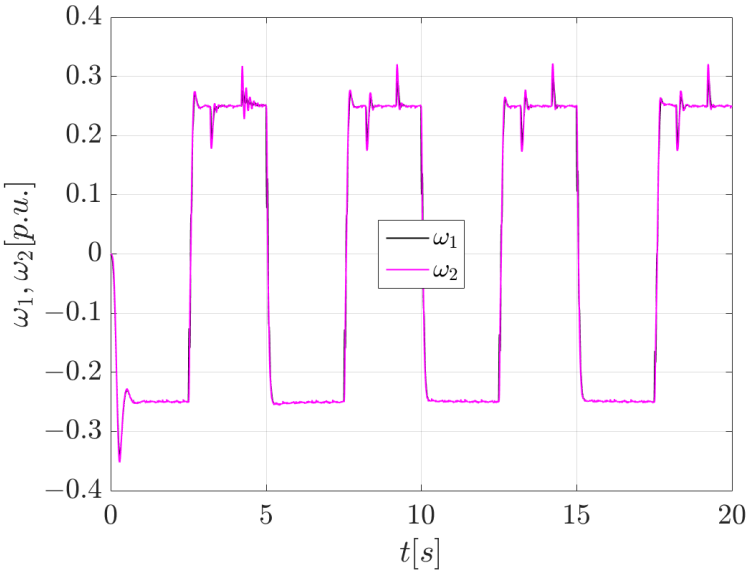


Figure 16. Transients of system signals achieved for an increased load time constant ($T_2 = 2T_{2n}$) of the drive.

7. Conclusions

In this paper, an IoT-enabled, ARM-based monitoring and control system was developed and applied to an electric drive with finite stiffness. An Elman recurrent neural network and a neural predictor were designed to control a drive with an elastic joint. The comparison of a drive operating with and without the predictor applied was also shown to confirm the superiority of two neural networks cooperating in a single speed control structure. The convenient and effective approach to optimization of the parameters (of the neural network) with the Artificial Bee Colony algorithm was provided. The presented results confirm that the use of a metaheuristic algorithm can facilitate the design process, since there is no given prescription for selection of correct values of coefficients. The developed data server enables online observation of the plant state variables and basic control of the electrical machines. The presented idea of coupling the smart controller utilizing elements of artificial intelligence and online data visualization is coherent with the Industry 5.0 concept. The novelty was not only provided by the original idea of synthesizing the controller structure and data presentation, but also by the implementation of a low-cost mainstream microcontroller. The data flow of the proposed system was described in detail. Thus, the framework for developing similar systems was established.

The implementation of a low-cost ESP32 chip proves that modern machinery can be monitored with simple devices without complex and expensive specialized HMI systems. Both cost-effective design and online data acquisition are essential to improve electric drive data availability in the industry. The advantage of the proposed HMI is that it was developed as a web page with a customized layout that can be adapted to requirements and can be opened with any device with access to the internet.

Author Contributions: Conceptualization, M.K.; methodology, M.K., R.S., M.M. and M.Z.; software, R.S., M.M. and M.Z.; validation, M.K., R.S., M.M. and M.Z.; formal analysis, M.K., R.S., M.M. and M.Z.; investigation, R.S., M.M. and M.Z.; resources, M.K.; data curation, R.S., M.M. and M.Z.; writing—original draft preparation, M.K., R.S., M.M. and M.Z.; writing—review and editing, M.K., R.S., M.M. and M.Z.; visualization, R.S. and M.Z.; supervision, M.K.; project administration, M.K.; funding acquisition, M.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

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

References

1. Blagodarov, D.A.; Grigorian, D.D.; Safonov, Y.M. The Approach to Design Control System for the Electric Drive with Flexible Mechanics. *Procedia Eng.* **2017**, *206*, 540–545. [CrossRef]
2. Zhan, B.; Jin, M.; Liu, J. Extended-state-observer-based adaptive control of flexible-joint space manipulators with system uncertainties. *Adv. Space Res.* **2022**, *69*, 8, 3088–3102. [CrossRef]
3. Zawarczynski, L.; Stefanski, T. Damping of torsional vibrations in electric drive with AC motor. In Proceedings of the 2018 International Interdisciplinary PhD Workshop (IIPhDW), Swinouscie, Poland, 9–12 May 2018; pp. 74–80. [CrossRef]
4. Luczak, D.; Pajchrowski, T. Application of Adaptive Neural Controller and Filter Tune for Multi-Mass Drive System. In Proceedings of the 20th European Conference on Power Electronics and Applications (EPE'18), Riga, Latvia, 17–21 September 2018; pp. 1–9.
5. Brock, S.; Luczak, D.; Nowopolski, K.; Pajchrowski, T.; Zawirski, K. Two approaches to Speed Control for Multi-Mass System with Variable Mechanical Parameters. *IEEE Trans. Ind. Electron.* **2017**, *64*, 3338–3347. [CrossRef]
6. Kaminski, M.; Orlowska-Kowalska, T. FPGA Implementation of ADALINE- Based Speed Controller in a Two-mass system. *IEEE Trans. Ind. Inform.* **2013**, *9*, 436–445. [CrossRef]
7. Dhaouadi, R.; Kubo, K.; Tobise, M. Two-degree-of-freedom robust speed controller for high-performance rolling mill drives. *IEEE Trans. Ind. Appl.* **1993**, *29*, 919–926. [CrossRef]
8. Zhu, H.; Yoshida, S. Disturbance observer-based torsional vibration damper for variable-speed wind turbines. *IFAC J. Syst. Control* **2020**, *14*, 100112. [CrossRef]
9. Li, R.; Huang, Q. Model of Resonance Predictive in Pitch System of Wind Turbine Based on Vector Fitting. In Proceedings of the 2021 International Conference on Power System Technology (POWERCON), Haikou, China, 8–9 December 2021; pp. 1220–1223. [CrossRef]
10. Li, J.; Wang, D.; Wu, X.; Xu, K.; Liu, X. Vibration prediction of the robotic arm based on elastic joint dynamics modeling. *Sensors* **2022**, *22*, 6170. [CrossRef]
11. Madsen, E.; Rosenlund, O.S.; Brandt, D.; Zhang, X. Comprehensive modeling and identification of nonlinear joint dynamics for collaborative industrial robot manipulators. *Control Eng. Pract.* **2020**, *101*, 104462. [CrossRef]
12. Liu, L.; Yao, W.; Guo, Y. Prescribed Performance tracking control of a free-frying flexible-joint robot space robot with distrubances under input saturation. *J. Frankl. Inst.* **2021**, *358*, 4571–4601. [CrossRef]
13. Zhang, G.; Furusho, J. Speed control of two-inertia system by PI/PID control. *IEEE Trans. Ind. Electron.* **2000**, *47*, 603–609. [CrossRef]
14. Brock, S.; Gniadek, M. Analysis of input shaping and PID-controller interaction structures for two-mass systems. In Proceedings of the 16th International Conference on Mechatronics—Mechatronika 2014, Brno, Czech Republic, 3–5 December 2014; pp. 625–630. [CrossRef]
15. Thomsen, S.; Hoffmann, N.; Fuchs, F.W. PI Control, PI-Based State Space Control, and Model-Based Predictive Control for Drive Systems With Elastically Coupled Loads—A Comparative Study. *IEEE Trans. Ind. Electron.* **2011** *58*, 3647–3657. [CrossRef]
16. Choudhary, S.; Sharma, S.K.; Shrivastava, V. Modelling of speed controller for industrial applications: A two mass drive system. In Proceedings of the 2016 IEEE 7th Power India International Conference (PIICON), Bikaner, India, 25–27 November 2016; pp. 1–5. [CrossRef]
17. Saarakkala, S.E.; Hinkkanen, M. State-Space Speed Control of Two-Mass Mechanical Systems: Analytical Tuning and Experimental Evaluation. *IEEE Trans. Ind. Appl.* **2014**, *50*, 3428–3437. [CrossRef]
18. Hori, Y.; Iseki, H.; Sugiura, K. Basic consideration of vibration suppression and disturbance rejection control of multi-inertia system using SFLAC (state feedback and load acceleration control). *IEEE Trans. Ind. Appl.* **1994**, *30*, 889–896. [CrossRef]
19. Pajchrowski, T.; Siwek, P.; Wojcik, A. Application of the Reinforced Learning Method for adaptive electric drive control with variable parameters. In Proceedings of the IEEE 19th International Power Electronics and Motion Control Conference (PEMC), Gliwice, Poland, 25–29 April 2021; pp. 687–694. [CrossRef]
20. Dallolio, A.; Overaas, H.; Johansen, T.A. Gain-scheduled steering control for a wave-propelled unmanned surface vehicle. *Ocean Eng.* **2022**, *257*, 111618. [CrossRef]
21. Azahar, M.I.P.; Irawan, A.; Ismail, R.M.T.R. Self-tuning hybrid fuzzy sliding surface control for pneumatic servo system positioning. *Control Eng. Pract.* **2021**, *113*, 104838. [CrossRef]
22. Usama, M.; Kim, J. Improved self-sensing speed control of IPMSM drive based on cascaded nonlinear control. *Energies* **2021**, *14*, 2205. [CrossRef]
23. Pajchrowski, T. The direct drive with variable moment of inertia in the structure of the reference model. In Proceedings of the 2014 16th International Power Electronics and Motion Control Conference and Exposition, Antalya, Turkey, 21–24 September 2014; pp. 695–700. [CrossRef]
24. Zhang, D.; Wei, B. A reiew on model reference adaptive control of robotic manipulators. *Annu. Rev. Control* **2017**, *43*, 188–198. [CrossRef]
25. Talebi, H.A.; Patel, R.V.; Wong, M. A Neural-Network Based Observer for Flexible-Joint Manipulators. *IFAC Proc.* **2002**, *35*, 317–322. [CrossRef]
26. Hong, J.; Wang, Z.; Chen, W.; Wang, L.Y.; Qu, C. Online joint-prediction of multi-forward-step battery SOC using LSTM neural networks and multiple linear regression for real-world electric vehicles. *J. Energy Storage* **2020**, *30*, 101459. [CrossRef]

27. Pietrzak, P.; Wolkiewicz, M.; Orlowska-Kowalska, T. PMSM Stator Winding Fault Detection and Classification Based on Bispectrum Analysis and Convolutional Neural Network. *IEEE Trans. Ind. Electron.* **2022**, *70*, 5192–5202. [CrossRef]
28. Ding, S.; Peng, J.; Zhang, H.; Wang, Y. Neural network-based adaptive hybrid impedance control for electrically driven flexible-joint robotic manipulators with input saturation. *Neurocomputing* **2021**, *458*, 99–111. [CrossRef]
29. Rego, R.C.B.; Meneghetti, F.; de Araujo, U. Lyapunov-based continuous-time nonlinear control using deep neural network applied to underactuated systems. *Eng. Appl. Artif. Intell.* **2022**, *107*, 104519. [CrossRef]
30. Zychlewicz, M.; Stanislawski, R.; Kaminski, M. Grey Wolf Optimizer in Design Process of the Recurrent Wavelet Neural Controller Applied for Two-Mass System. *Electronics* **2022**, *11*, 177. [CrossRef]
31. Hannan, M.A.; Ali, J.A.; Mohamed, A.; Hussain, A. Optimization techniques to enhance the performance of induction motor drives: A review. *Renew. Sustain. Energy Rev.* **2018**, *81*, 1611–1626. [CrossRef]
32. Kennedy, J.; Eberhart, R. Particle Swarm Optimization. In Proceedings of the 1995 IEEE International Conference on Neural Networks, Perth, Australia, 27 November–1 December 1995; pp. 1942–1948. .
33. Shirzadeh, M.; Amirkhani, A.; Tork, N.; Taghavifar, H. Trajectory tracking of a quadrotor using a robust adaptive type-2 fuzzy neural controller optimized by cuckoo algorithm. *ISA Trans.* **2021**, *114*, 171–190. [CrossRef] [PubMed]
34. Knypinski, L.; Kuroczycki, S.; Marquez, F.P.G. Minimization of torque ripple in the brushless DC motor using Constrained Cuckoo Search Algorithm. *Electronics* **2021**, *10*, 2299. [CrossRef]
35. Lin, C.-H. Permanent-Magnet Synchronous Motor Drive System Using Backstepping Control with Three Adaptive Rules and Revised Recurring Sieved Pollaczek Polynomials Neural Network with Reformed Grey Wolf Optimization and Recouped Controller. *Energies* **2020**, *13*, 5870. [CrossRef]
36. Knypinski, L.; Paweloszek, K.; Menach, Y.L. Optimization of low-power line-start PM motor using Gray Wolf Metaheuristic Algorithm. *Energies* **2020**, *13*, 1186. [CrossRef]
37. Colorni, A.; Dorigo, M.; Maniezzo, V. Distributed optimization by ant colonies. In Proceedings of the 1st European Conference on Artificial Life, York, UK, 11–13 November 1991; pp. 131–142.
38. Szczepanski, R.; Tarczewski, T.; Erwinski, K.; Grzesiak, L.M. Comparison of constraint-handling techniques used in Artificial Bee Colony Algorithm for auto-tuning of state feedback speed controller for PMSM. In Proceedings of the 15th International Conference on Informatics in Control, Automation and Robotics, Porto, Portugal, 29–31 July 2018 ; pp. 269–276. [CrossRef]
39. Tarczewski, T.; Niewiara, L.J.; Grzesiak, L.M. Artificial bee colony based state feedback position controller for PMSM servo-drive—The efficiency analysis. *Bull. Pol. Acad. Sci. Tech. Sci.* **2020**, *68*, 997–1007. [CrossRef]
40. Yang, X.S.; Gandomi, A.H. Bat algorithm: A novel approach for global engineering optimization. *Eng. Comput.* **2012**, *29*, 464–483. [CrossRef]
41. Balasubramani, K.; Marcus, K. A Comprehensive review of Artificial Bee Colony Algorithm. *Int. J. Comput. Technol.* **2013**, *5*, 15–28. [CrossRef]
42. Pianegiani, F.; Macii, D.; Carbone, P. An open distributed measurement system based on an abstract client-server architecture. *IEEE Trans. Instrum. Meas.* **2003**, *52*, 686–692. [CrossRef]
43. Winiecki, W.; Karkowski, M. A new Java-based software environment for distributed measuring systems design. *IEEE Trans. Instrum. Meas.* **2002**, *51*, 1340–1346. [CrossRef]
44. Guo, Z.; Chen, P.; Zhang, H.; Jiang, M.; Li, C. IMA: An Integrated Monitoring Architecture With Sensor Networks. *IEEE Trans. Instrum. Meas.* **2012**, *61*, 1287–1295. [CrossRef]
45. Singh, H.V. P.; Mahmoud, Q.H. NLP-Based Approach for Predicting HMI State Sequences towards Monitoring Operator Situational Awareness. *Sensors* **2020**, *20*, 3228. [CrossRef] [PubMed]
46. Vuckovic, M.; Schmidt, J.; Ortner, T.; Cornel, D. Combining 2D and 3D Visualization with Visual Analytics in the Environmental Domain. *Information* **2022**, *13*, 7. [CrossRef]
47. Wang, G.; Ren, G.; Hong, X.; Peng, X.; Li, W.; O'Neill, E. Freehand Gestural Selection with Haptic Feedback in Wearable Optical See-through Augmented Reality. *Information* **2022**, *13*, 566. [CrossRef]
48. Sun, Z.; Zhu, M.; Lee, C. Progress in the Triboelectric Human–Machine Interfaces (HMIs)-Moving from Smart Gloves to AI/Haptic Enabled HMI in the 5G/IoT Era. *Nanoenergy Adv.* **2021**, *1*, 81–120. [CrossRef]
49. Juneja, A.; Das, N.N. Big Data Quality Framework: Pre-Processing Data in Weather Monitoring Application. In Proceedings of the 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), Faridabad, India, 14–16 February 2019; pp. 559–563. [CrossRef]
50. Sigüenza, Á.; Díaz-Pardo, D.; Bernat, J.; Vancea, V.; Blanco, J.L.; Conejero, D.; Hernández Gómez, L. Sharing Human-Generated Observations by Integrating HMI and the Semantic Sensor Web. *Sensors* **2012**, *12*, 6307–6330. [CrossRef]
51. Mourtzis, D.; Angelopoulos, J.; Panopoulos, N. A Literature Review of the Challenges and Opportunities of the Transition from Industry 4.0 to Society 5.0. *Energies* **2022**, *15*, 6276. [CrossRef]
52. Akundi, A.; Euresti, D.; Luna, S.; Ankobiah, W.; Lopes, A.; Edinbarough, I. State of Industry 5.0—Analysis and Identification of Current Research Trends. *Appl. Syst. Innov.* **2022**, *5*, 27. [CrossRef]
53. Nahavandi, S. Industry 5.0—A Human-Centric Solution. *Sustainability* **2019**, *11*, 4371. [CrossRef]
54. Szczepanski, R.; Kaminski, M.; Tarczewski, T. Auto-Tuning Process of State Feedback Speed Controller Applied for Two-Mass System. *Energies* **2020**, *13*, 3067. [CrossRef]

55. Pajchrowski, T.; Janiszewski, D. Control of multi-mass system by on-line trained neural network based on Kalman filter. In Proceedings of the 17th European Conference on Power Electronics and Applications (EPE'15 ECCE-Europe), Geneva, Switzerland, 8–10 September 2015; pp. 1–10. [CrossRef]
56. El-Sousy, F.F.M. Intelligent Optimal Recurrent Wavelet Elman Neural Network Control System for Permanent-Magnet Synchronous Motor Servo Drive. *IEEE Trans. Ind. Inform.* **2013**, *9*, 1986–2003. [CrossRef]
57. Derugo, P.; Szabat, K.; Zawirski, T.P.K. Fuzzy Adaptive Type II Controller for Two-Mass System. *Energies* **2022** *15*, 419. [CrossRef]
58. Widrow, B.; Stearns, S.D. *Adaptive Signal Processing*; Prentice Hall: Upper Saddle River, NJ, USA, 1985; ISBN 0-13-004029-0.
59. Garanayak, P.; Naayagi, R.T.; Panda, G. A High-Speed Master-Slave ADALINE for Accurate Power System Harmonic and Inter-Harmonic Estimation. *IEEE Access* **2020** *8*, 51918–51932. [CrossRef]
60. Kaminski, M. Adaptive Controller with Neural Signal Predictor Applied for Two-Mass System. In Proceedings of the 2018 23rd International Conference on Methods & Models in Automation & Robotics (MMAR), Miedzyzdroje, Poland, 27–30 August 2018; pp. 247–252. [CrossRef]
61. Karaboga, D.; Basturk, B. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *J. Glob. Optim.* **2007**, *39*, 459–471. [CrossRef]
62. Xiang, W.-L.; Li, Y.-Z.; He, R.-C.; An, M.-Q. Artificial bee colony algorithm with a pure crossover operation for binary optimization. *Comput. Ind. Eng.* **2021**, *152*, 107011. [CrossRef]
63. Aranburu, E.; Lasa, G.; Gerrikagoitia, J.K.; Mazmela, M. Case study of the experience capturer evaluation tool in the design process of an industrial HMI. *Sustainability* **2020**, *12*, 6228. [CrossRef]
64. Sitompul, T.A. Human–Machine Interface for remote crane operation: A review. *Multimodal Technol. Interact.* **2022**, *6*, 45. [CrossRef]
65. Zheng, S.; Zhang, Q.; Zheng, R.; Huang, B.-Q.; Song, Y.-L.; Chen, X.-C. Combining a multi-agent system and communication middleware for smart home control: a universal control platform architecture. *Sensors* **2017**, *17*, 2135. [CrossRef] [PubMed]
66. Duobiene, S.; Ratautas, K.; Trusovas, R.; Ragulis, P.; Šlekas, G.; Simniškis, R.; Račiukaitis, G. Development of wireless sensor network for environment monitoring and its implementation using SSAIL technology. *Sensors* **2022**, *22*, 5343. [CrossRef] [PubMed]
67. Mandza, Y.S.; Raji, A. IoTivity cloud-enabled platform for energy management applications. *IoT* **2022**, *3*, 73–90. 10.3390/iot3010004 [CrossRef]
68. Aghenta, L.O.; Iqbal, M.T. Low-cost, open source IoT-based SCADA system design using Thinger.IO and ESP32 thing. *Electronics* **2019**, *8*, 822. [CrossRef]
69. *ESP32 Series Datasheet Version 4.1*; Espressif Systems: Shanghai, China, 2022.
70. Balde, A.Y.; Bergeret, E.; Cajal, D.; Toumazet, J.-P. Low Power Environmental Image Sensors for Remote Photogrammetry. *Sensors* **2022**, *22*, 7617. [CrossRef] [PubMed]
71. *Programmable Terminal NB Series/NB-S Series Datasheet*; Omron: Kyoto, Japan, 2021.