


Article

Design and Implementation of a Machine-Learning Observer for Sensorless PMSM Drive Control

Dwi Sudarno Putra ^{1,2}, Seng-Chi Chen ^{1,*}, Hoai-Hung Khong ³  and Fred Cheng ⁴

¹ Department of Electrical Engineering, Southern Taiwan University of Science and Technology, Tainan 71005, Taiwan; da72b205@stust.edu.tw

² Department of Automotive Engineering, Universitas Negeri Padang, Padang 25132, Indonesia

³ Faculty of Electrical and Electronics Engineering, Ho Chi Minh City University of Transport, Ho Chi Minh City 70000, Vietnam; hungkhonghoai@hcmutrans.edu.vn

⁴ Imeier Green Technology Co., Ltd., New Taipei City 23511, Taiwan; fred@imeier.com.tw

* Correspondence: amtfcs123@stust.edu.tw; Tel.: +886-6-253-3131 (ext. 3324)

Abstract: Information about rotor positions is critical when controlling a permanent-magnet synchronous motor (PMSM). This information can be gathered using a sensor or through an estimation without using a sensor. This article discusses a machine learning technique for estimating rotor positions. The proposed machine learning observer was constructed using a modified Elman neural network as the main algorithm. The network was trained offline with training data obtained from PMSM field-oriented control simulations and was tested using a validation data set. The PMSM control simulation results revealed that the rotor position estimated through machine learning was comparable with the simulated rotor position; the average error was 0.0127 per unit position. Furthermore, the machine learning model was implemented in an experimental PMSM-control hardware platform. Both the simulation and experimental results indicate that the proposed machine learning observer has an acceptable performance.

Keywords: machine learning observer; rotor position estimation; sensorless motor control; modified Elman neural network



Citation: Putra, D.S.; Chen, S.-C.; Khong, H.-H.; Cheng, F. Design and Implementation of a Machine-Learning Observer for Sensorless PMSM Drive Control. *Appl. Sci.* **2022**, *12*, 2963. <https://doi.org/10.3390/app12062963>

Academic Editors: Javier Poza, Gaizka Ugalde and Gaizka Almandoz

Received: 30 December 2021

Accepted: 9 March 2022

Published: 14 March 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Electric motors have various uses in modern life, such as in household appliances, manufacturing, and transportation [1–4]. The permanent magnet synchronous motor (PMSM) is one of the numerous types of electric motors and has a high performance, strength, and torque. Therefore, numerous researchers have studied PMSMs; in particular, the control of PMSMs has attracted substantial attention. Field-oriented control (FOC) is the most frequently used strategy for PMSM control. Because the FOC scheme requires information about rotor positions, PMSMs are typically equipped with a sensor to read this parameter. However, for several reasons, the necessity to use a bulky sensor needs to be avoided to reduce production costs, eliminate reliability challenges, and minimize the effect of noise that can interfere with sensor performance; this method is called sensorless control.

Research on sensorless control strategies can be divided into two categories: (1) control during the initial startup and at a low speed [5–7] and (2) control at a medium or high speed [4,8,9]. The high-frequency injection technique is typically used for low speeds, whereas the back-electromotive force (EMF) method is used for high speeds.

Some research has suggested an alternative using artificial intelligence (AI) for sensorless control [10–16]. In 2003, Lachman et al., showed that neural networks (NNs) are limited by long computation times because NN performance is dominated by complex recursive computations [10].

As a subset of AI, machine learning (ML) has attracted attention due to its potential applications in many engineering disciplines, including powered electronics and motor

drives. ML based on an artificial NN (ANN) offers several advantages, such as a distributed design that enables it to handle multiple inputs and outputs, the ability to recognize nonlinear systems, and the ability to learn, generalize, and adapt. These characteristics indicate that ML can be used for motor control.

ANNs have been widely implemented for general motor control. In [11], Hoai used a radial-basis function self-tuning proportional–integral–derivative (PID) NN controller to control the velocity loop of a PMSM. Ramirez and Trujillo implemented a back-propagation NN (BPNN) training method to track the speed of a brushless direct-current (BLDC) motor [12]. In [13], a reactive power-based model-reference adaptive-system speed estimator and an adaptive neural network for a PMSM were used to construct a sensorless speed system.

Previously, AI has been used to predict rotor positions. Rajesh Kumar successfully used an ANN to estimate a rotor position angle; trapezoidal back-EMF information from a BLDC was employed as the ANN input [14]. In another study, the Scikit-learn computation software library was successfully used, along with two currents from the Clarke transform (I_α and I_β) as inputs, to facilitate a simulation of rotor positions [15]. The AdaBoost algorithm and Scikit-learn software libraries require enormous computational resources. Moreover, [16] described a ML observer based on a back-propagation algorithm.

The sensorless ML-based observer became interesting since it uses data from a well-running control process (data-driven) instead of a formula with specified motor parameters. A sliding mode observer, another type of observer, requires certain parameter values so that the mathematical process runs well [8,9,17,18]. While ML-based observers do not require detailed parameters, the proposed ML model needs to be trained using data obtained from a motor control that has been running well [15,16].

In [15,16], training data acquired from signal sampling at various motor-speed control operation points were used. The methods provided a favorable response for constant speed control but had difficulty with speed transitions for some implementations. These studies [15,16] used a multilayer feedforward configuration; this method has a disadvantage in that feedback data from previous positions cannot be obtained. To overcome this weakness, this paper proposes a scheme that utilizes feedback information—the previous value of the output layer—to represent the last rotor angle.

Figure 1 presents the architecture of the sensorless PMSM-FOC developed in this study. The startup process was performed using the Volt/Hz open-loop control method [19–23]. When the motor was operated at a low speed (for example, less than 400 rpm), it rotated in an open loop (switch mode 1). Thus, when the speed reference was set to a value greater than 400 rpm, the motor speed ramped up in an open loop until it reached 400 rpm. Switch mode 2 was then used, enabling a closed-loop sensorless control mode for speeds exceeding 400 rpm.

ML with the Elman NN (ENN) basic architecture was used in this study. The ENN includes a context layer to store an internal state; thus, it is more dynamic than the multilayer perceptron architecture [24]. The ENN can be considered to have additional memory neurons and local feedback [25]. Since its introduction in 1990, the ENN has been widely used in various fields [26–30], and many efforts have been made to optimize its performance [31–33]. Figure 2 presents a block diagram of the modified ENN. The dashed line indicates the feedback portion of the original ENN algorithm; the previous network output value is fed back to the input layer. This modified part is also the novel contribution of this work. The inputs are voltages and currents in the α - β axes (v_α , v_β , I_α , and I_β). The targeted outputs are the sine and cosine of the rotor-position angle θ . Figure 1 indicates that information about the rotor speed ω_e is also required during the FOC process. Consequently, in the ML-based observer block, the modified ENN, is combined with a phase-locked loop (PLL) to determine θ and ω_e . The PLL has been widely adopted to enhance estimator quality [11,18,34,35]. MATLAB 2021b was used to develop the modified ENN algorithms, design and simulate the control schemes, generate the embedded code for hardware implementation, and control and retrieve data from experimental platforms. As a

part of MATLAB, Simulink has a Motor Control Blockset that contains several fundamental motor-control function blocks, such as the Clark and Park Transformations, space vector generators, and electrical system models, such as those for inverters and motors [36]. Thus, motor-control simulations can be performed rapidly, precisely, and accurately. Because of the precision and accuracy of the simulation data, ML models trained on these data are thought to perform well when implemented on real hardware. The present experimental results were obtained using a hardware control platform based on digital signal processors (DSP) from Texas Instruments (TI). The platform was developed with MATLAB's embedded coder support package, which can efficiently create a control algorithm from a Simulink model [11,37]. ML has high computational demands; thus, determining the sample time was critical to the success of the hardware implementation.

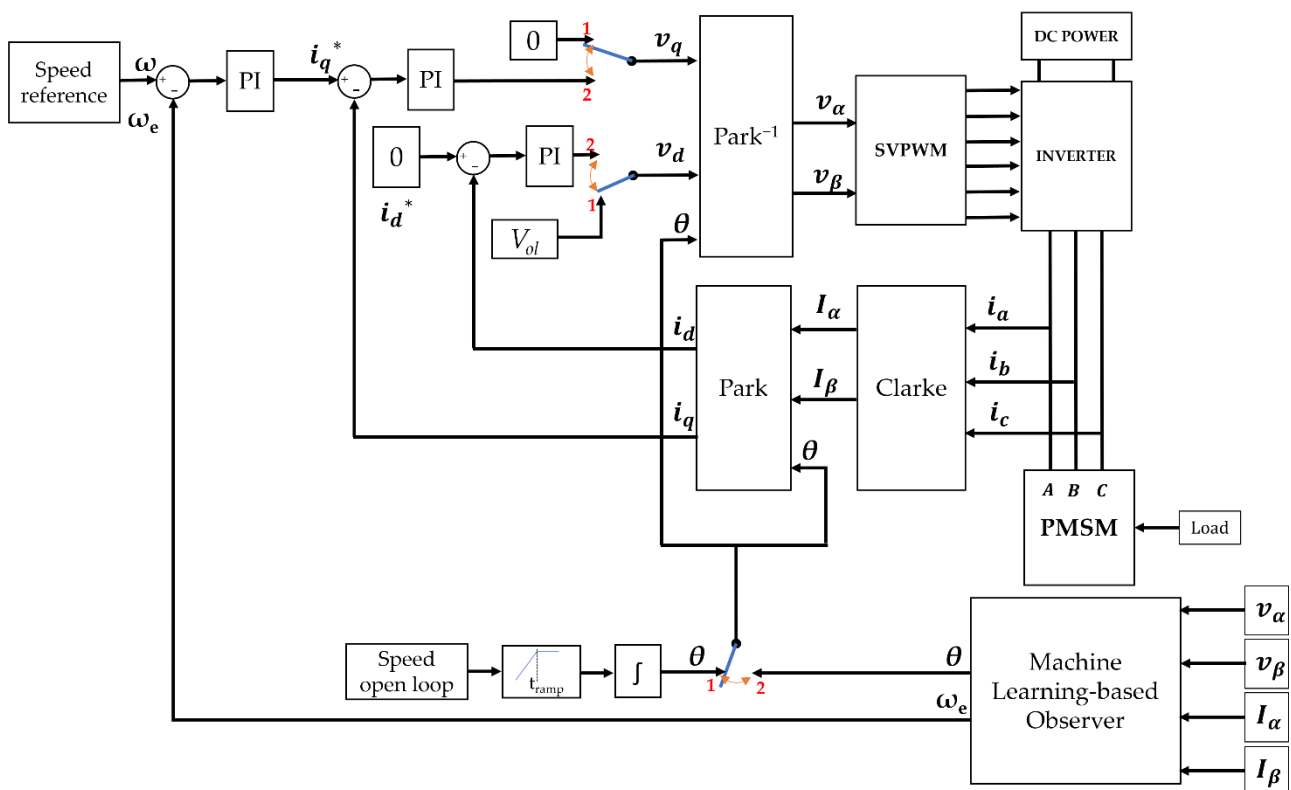


Figure 1. Proposed sensorless permanent magnet synchronous motor field-oriented control (PMSM-FOC) architecture.

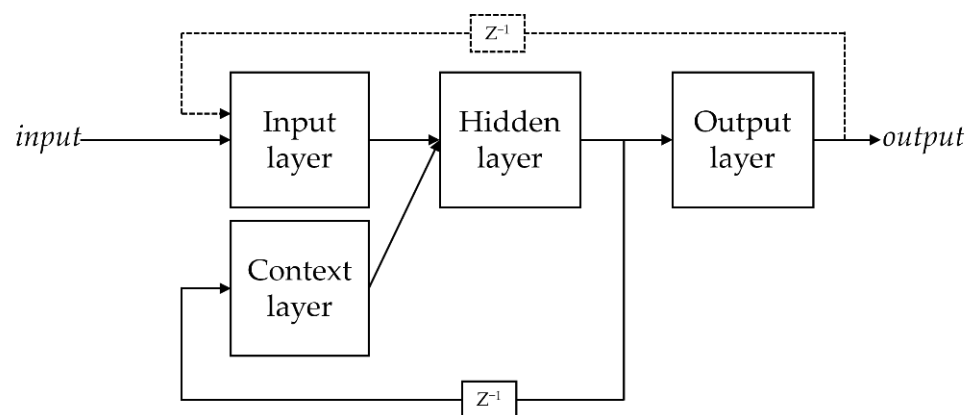


Figure 2. Modified Elman neural network (ENN) block diagram.

The following steps were performed to develop the model:

- An ML-based observer-training algorithm was developed on the basis of the modified ENN.
- Training and validation data were recorded from the sensed PMSM-FOC digital simulation performed with Simulink.
- The ML-based observer was trained and validated using the recorded data.
- The validated ML function block was implemented in a Simulink simulation for a sensorless PMSM-FOC.
- The ML-based observer was realized in a DSP-based hardware control scheme.

The remainder of this paper is organized as follows: Section 2 describes the PMSM drive system. Section 3 details the methodology for building the ML model, collecting learning data, and performing training and validations. The ML-based observer block is also described. In Section 4, the simulation results are presented and discussed. Section 5 presents the experimental results and discusses the implementation of the proposed algorithm. The final section concludes the paper.

2. PMSM Drive System

To control a PMSM using an ML-based observer, the observer need to be trained with data from a running PMSM control process. Training data were obtained from a simulation model instead of from a workbench control process.

In general, the mathematical model of a PMSM is expressed in the d - q synchronous rotating coordinate as follows:

$$\begin{cases} v_d = r_s i_d + L_s \frac{d}{dt} i_d - \omega_e L_s i_q \\ v_q = r_s i_q + L_s \frac{d}{dt} i_q + \omega_e L_s i_d + \omega_e \lambda_f \end{cases} \quad (1)$$

Here, v_d and v_q are the d - and q -axis voltages, respectively, and r_s is the winding resistance per phase of the stator. For a surface-mounted PMSM, $L_s = L_d = L_q$, where L_d and L_q are the d - and q -axis inductances, respectively; i_d and i_q denote the d - and q -axis currents, respectively; the rotational speed of magnet flux is denoted as ω_e ; and the permanent magnet flux linkage is denoted as λ_f .

Figure 1 indicates that i_d and i_q are obtained from Clarke and Park transformations of the phase currents. These currents are then controlled using proportional–integral (PI) current controls. The behavior is similar to a DC motor when the current i_d is set to zero. In mode 2, the torque produced by the PMSM can be expressed as follows:

$$T_e = \frac{3N_p}{4} \lambda_f i_q \triangleq K_t i_q \quad (2)$$

$$K_t = \frac{3N_p}{4} \lambda_f \quad (3)$$

Here, T_e is the electromagnetic torque, K_t is the torque constant, and N_p is the pole pairs. If T_L is the load torque, J is the inertia, and B is the friction constant, then the dynamic equation for the rotor speed (ω_r) of PMSM would be as follows:

$$\frac{d\omega_r}{dt} = \frac{1}{J} (T_e - T_L - B\omega_r) \quad (4)$$

If θ is the rotor position angle, the d - q axes of v_d and v_q can be returned to the α - β frame by using Park's inverse function:

$$v_\alpha = v_d \cos \theta - v_q \sin \theta \quad (5)$$

$$v_\beta = v_d \sin \theta + v_q \cos \theta \quad (6)$$

The three-phase currents (i_a , i_b , and i_c) from the PMSM are turned into two-phase α - β currents using the Clarke transform:

$$I_\alpha = i_a \quad (7)$$

$$I_\beta = \frac{1}{\sqrt{3}}i_a + \frac{2}{\sqrt{3}}i_b \quad (8)$$

The two-phase α - β currents are then transformed into currents in the d - q axis as i_d and i_q with the Park transform as follows:

$$i_d = I_\alpha \cos \theta + I_\beta \sin \theta \quad (9)$$

$$i_q = -I_\alpha \sin \theta + I_\beta \cos \theta \quad (10)$$

Figure 1 indicates that the FOC process comprises two control loops, typically called the inner loop and the outer loop. The inner loop control is the control of i_d and i_q . The outer loop control is the speed control. Both types of control are achieved using PI control strategies. Furthermore, a Simulink-based control simulation with sensors is performed to obtain the desired training data information for v_α , v_β , I_α , I_β , $\sin \theta$ and $\cos \theta$.

3. Proposed ML-Based Rotor Observer

3.1. Machine Learning

A flowchart of ML is presented in Figure 3. ML is the study of computer algorithms that can learn and develop on their own with experience and data [38].

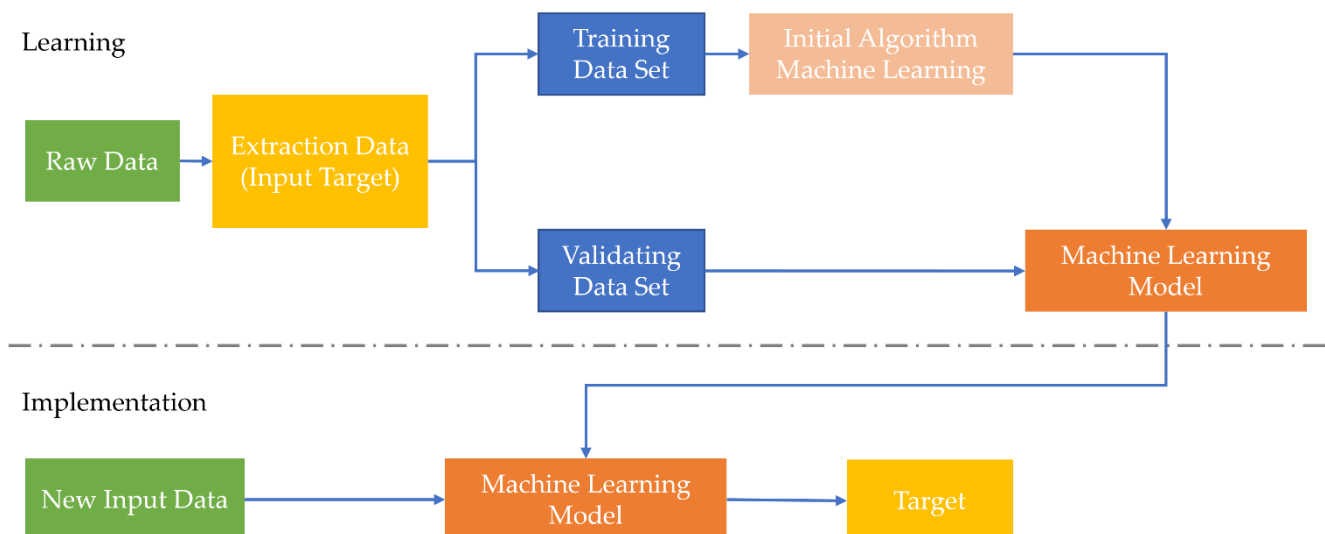


Figure 3. Machine learning (ML) flowchart.

An ML algorithm is initially a computational algorithm with a particular structure that provides random output. After each round of learning with training data, the algorithm capability is updated and the outputs are improved. A validation data set is then used to test the performance of the model. During testing, the ML model should give the correct output for each corresponding input with a low error rate. After updating its estimation capabilities, the model gives the desired output in accordance with previously learned input data features.

3.2. Modified ENN

In this study, the ENN was used as the core of the ML-based observer. The ENN is a network with memory neurons and a context layer [24]. The context-layer neurons store data generated by the hidden-layer neurons during each cycle; the stored data are then

used as inputs for the hidden-layer neurons in the next cycle. The context and hidden layers have the same number of neurons.

Figure 4 shows how values from the output layer are fed back to the input layer to improve ML performance. The number of input neurons is increased to handle this feedback. The initial four neurons (v_α , v_β , I_α and I_β) were increased to six neurons due to the additional feedback of $\sin \theta$ and $\cos \theta$.

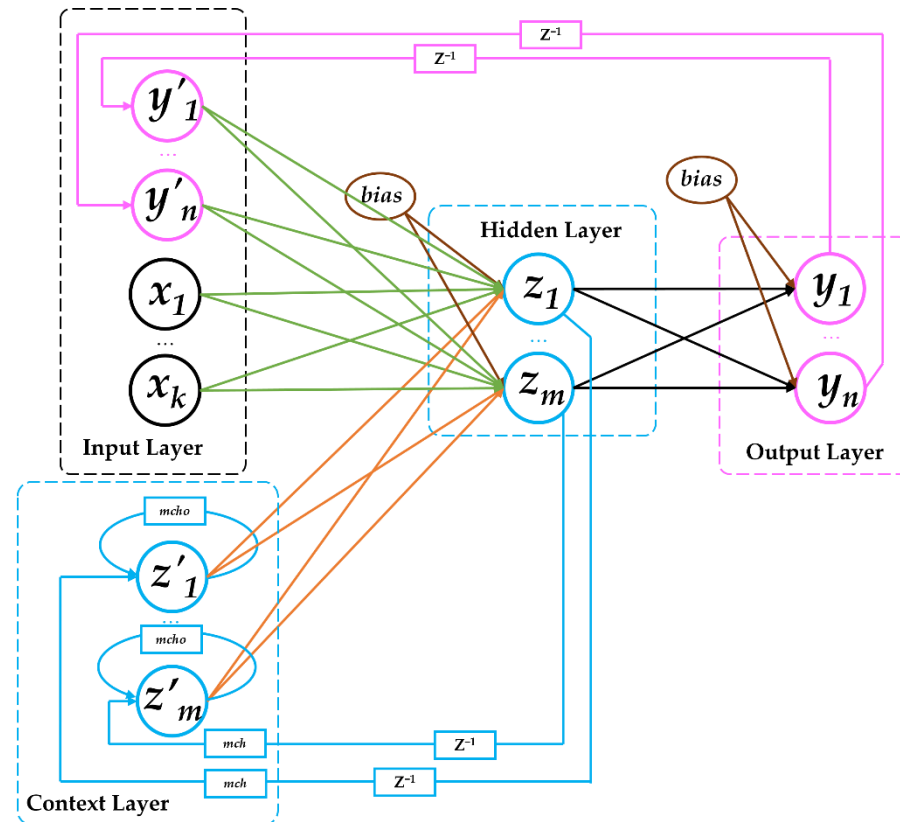


Figure 4. Scheme of the proposed modified ENN.

During the training process, the updating weight in this algorithm is the difference between the output neuron value and target output value. This process is the same as that in the BPNN. However, because the modified ENN has more input neurons and has a context layer, the required training time and execution time are longer than for the BPNN. Nonetheless, with the addition of this input, the observer's performance becomes much better because it can overcome the problem of speed transition.

The modified ENN algorithm depicted in Figure 4 was developed with the following steps:

- Step 1. The number of neurons in each layer is initially established. The input layer has six neurons ($k + n$), the output layer (n) has two neurons, and both the hidden layer and the context layer have seven neurons (m). The learning-rate value (lr) is 0.001.
- Step 2. The weights for each input layer (wil), context layer (wcl), and hidden layer (whl); the bias value of the hidden layer (bhl) and output layer (bol); and the memory coefficients of the context layer ($mcho$ and mch) are randomly generated.
- Step 3. In the first iteration of the training process, the context-layer neuron values must be initiated (z').
- Step 4. Each neuron from the input layer (x and y') and context layer (z') is distributed to the input of each neuron in the hidden layer. Finally, these values are summed with an appropriate bias value (vb). The summation results are then input into the activation function to obtain the output value of each hidden-layer neuron (z).

- Step 5. The output values of the hidden-layer neurons (z) are fed back to the context layer and become z' for the next iteration. Moreover, z is weighted (w) and is distributed to the next layer for each neuron in the output layer (y). All inputs from each hidden-layer neuron (z) are summed with an appropriate bias (wb). The summation results are then used on the activation function to obtain the value of each output-layer neuron (y).
- Step 6. The activation function calculation result for the output-layer neurons is the output of the network. This output is fed back to the input layer as y' and is compared with the target output to obtain an error value.
- Step 7. The weights and biases of the network are updated on the basis of the error values and learning rate.
- Step 8. Steps 4–7 are repeated until the desired number of training cycles (maximum epoch) is reached or the convergence condition is met.

Next, the generated model had to be tested using actual learning data. Data were obtained by running a sensed PMSM-FOC simulation on MATLAB Simulink.

3.3. Learning Process

In the learning step, ML requires both training data and validation data. The learning data must be gathered from a reliable source [16]. A sensed PMSM-FOC simulation based on MATLAB Simulink was performed and was used to acquire learning data (Figure 5).

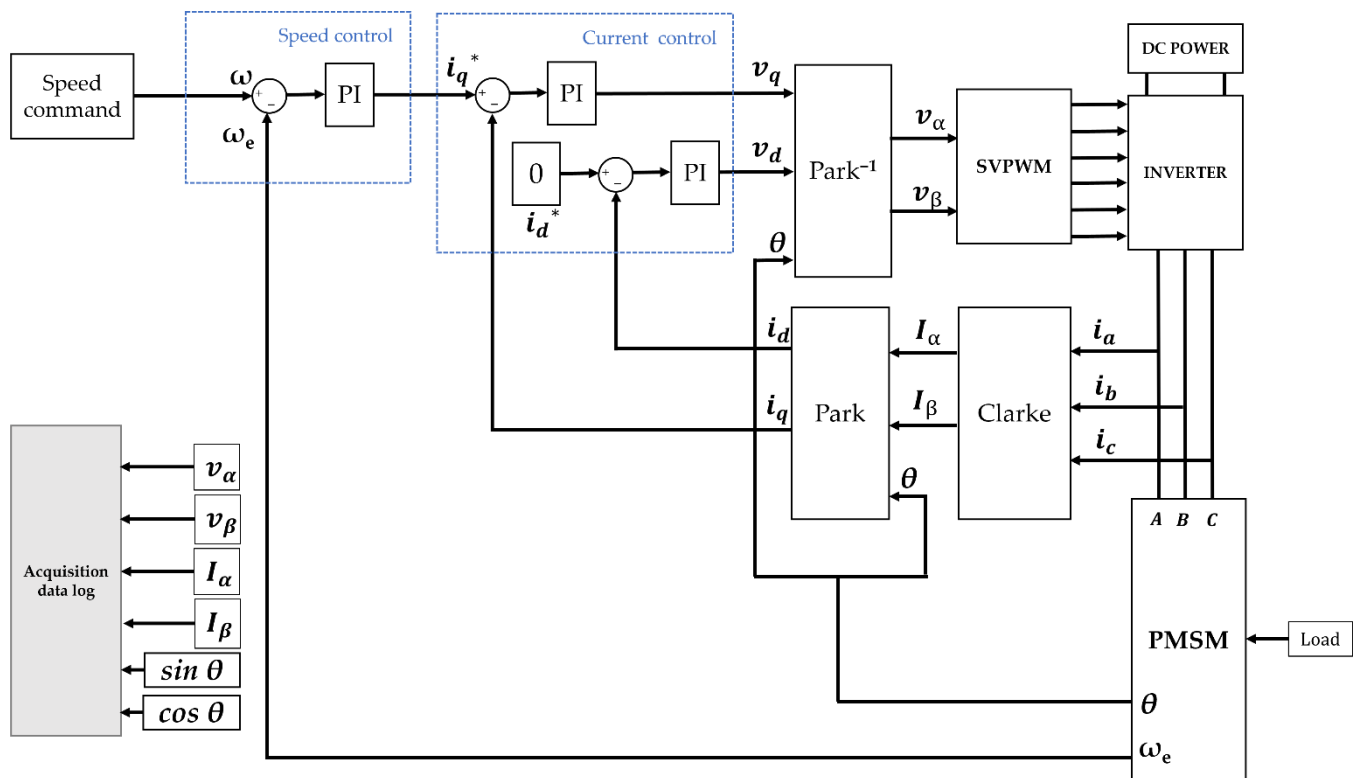


Figure 5. Sensed PMSM-FOC architecture for data acquisition.

The motor parameters are presented in Table 1. In the simulation, the PI control parameters for the current loop were $K_p = 1.7011$ and $K_i = 3098.9$, whereas the PI control parameters for the speed loop were $K_p = 0.4595$ and $K_i = 6.0661$. The sampling frequency was 10 kHz. To optimize the computational process, the values of several variables using a per-unit system (p.u system) were used to scale the international value system of unit or SI values as p.u-values from -1 to 1 [39].

Table 1. PMSM parameters.

Parameter	Value	Parameter	Value
Flux	0.0064 Wb	Rated Speed	3000 rpm
Inductance ($L_d = L_q$)	0.2 mH	Rated Voltage	24 V
Resistance (R_s)	0.36 Ohm	Rated Current	7.1 A
Inertia Coefficient (J)	7.06×10^{-6} Kg·m ²	Rated Torque	0.27 Nm
Friction Coefficient (B)	2.64×10^{-5} Kg·m ² /s	Pole Pair	4 Pairs

3.3.1. Training Data and Validation Data

A control system has a certain control range; this is also true in the present ML-based control system. The ML network must learn from training data that cover the entire range of motor speeds and torque control operations, including both speed and torque transitions. The speed variation commands and load variation profile for training data acquisition are shown in Figure 6a. Validation data were acquired using the profiles displayed in Figure 6b.

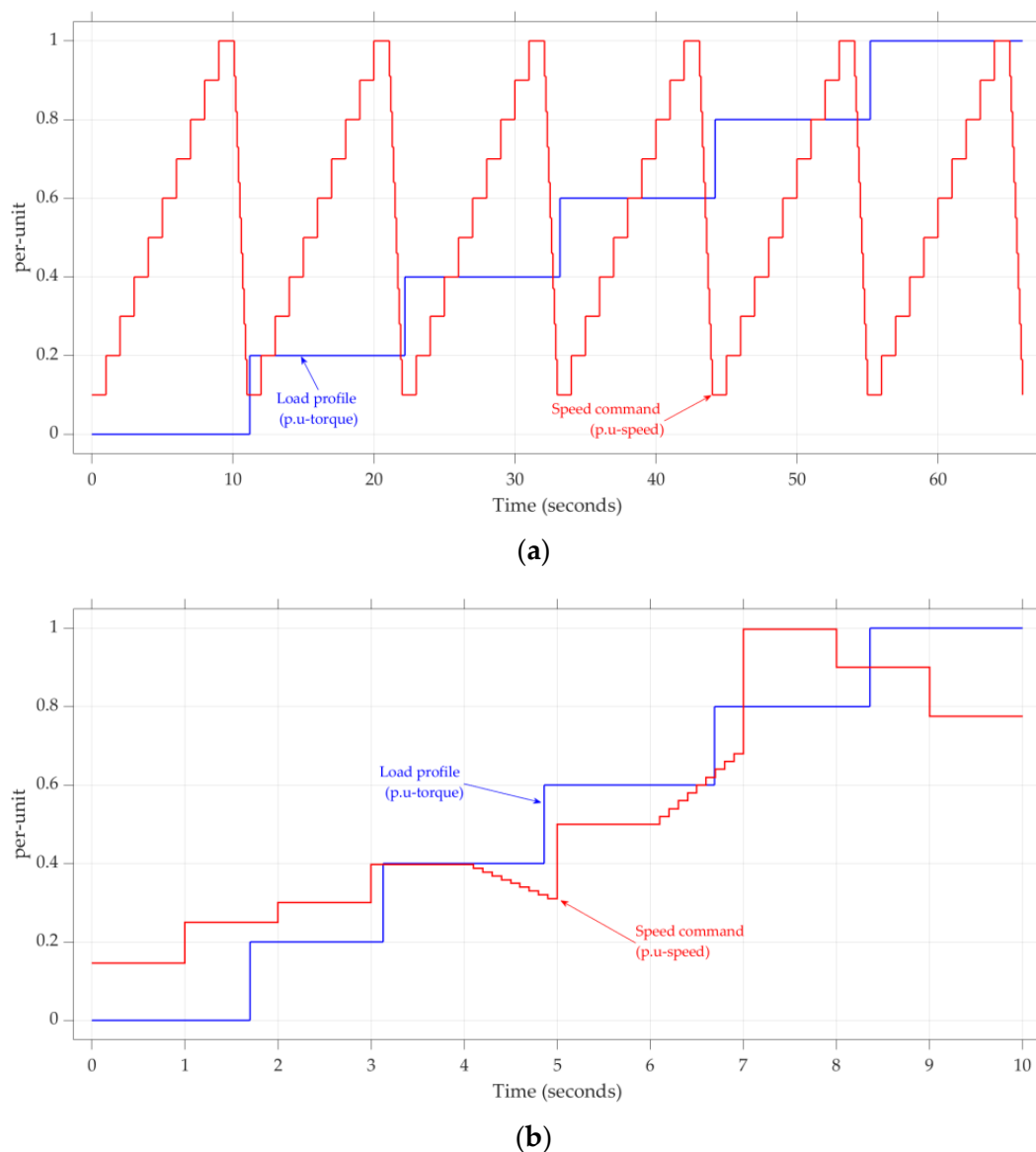


Figure 6. Speed commands and load profile settings for (a) training and (b) validation data acquisition.

For the training and validation processes, learning data were recorded from a simulation. An example of the acquired data is presented in Figure 7. Data were recorded and saved as arrays; each data array contained six values: v_α , v_β , I_α , I_β , $\sin \theta$ and $\cos \theta$. The sizes of the training and validation data arrays were 660.000 and 100.000, respectively.

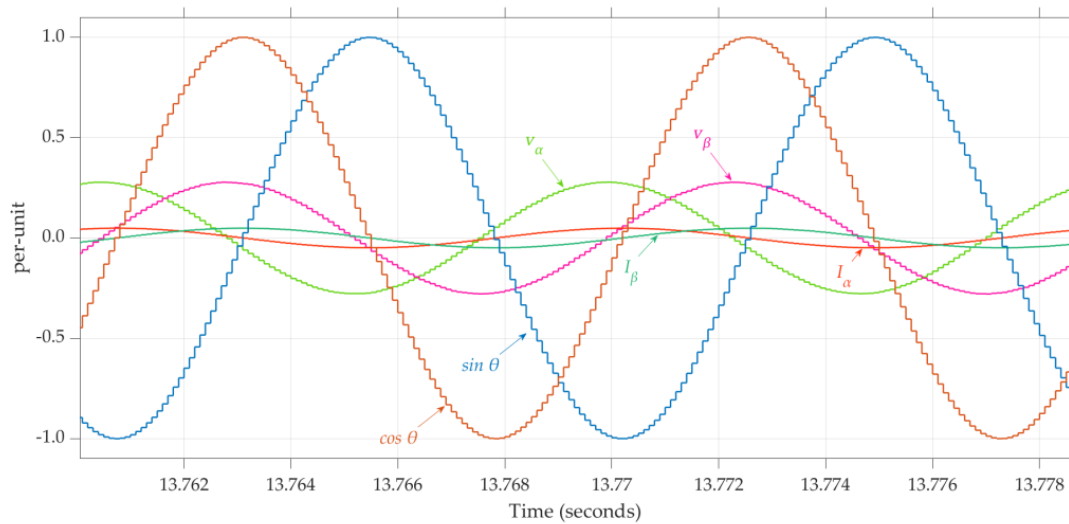


Figure 7. Example of recorded data.

3.3.2. Training and Validation

A set of training-routine programs was successfully created using MATLAB on the basis of the algorithm described in Section 3.2. The training process was executed several times to obtain the optimal training parameters and results. The optimal learning rate was 0.001, and the optimal number of neurons in the hidden layer was seven. The mean square error reached 3×10^{-6} ; the epoch error graph during training is presented in Figure 8. The trend of the graph shows that the ML model trains properly.

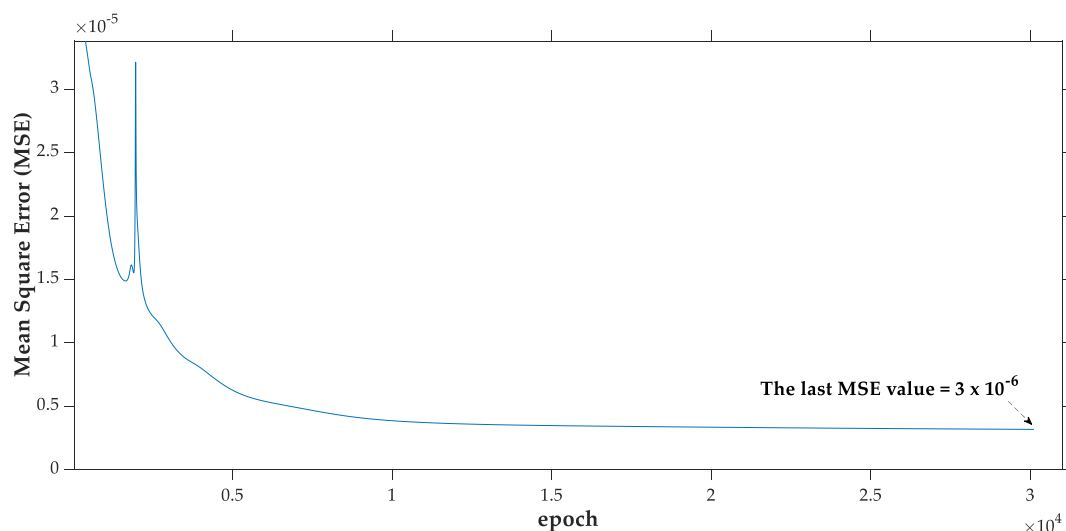


Figure 8. Mean square error performance curve.

The ML model's "brain" is the value of its weights, biases, and other coefficients at the end of the training process. A testing procedure with validation data can be used to confirm that the ML model has high performance for real data. Only steps 3, 4, and 5 (Section 3.2) were used during testing. A MATLAB testing routine program was created and successfully executed. Figure 9a,b presents the error histograms for the training and

validation data, respectively. Most errors in Figure 9a,b are close to zero; thus, the existing ML model is accurate and can be applied further as the core of the ML-based observer.

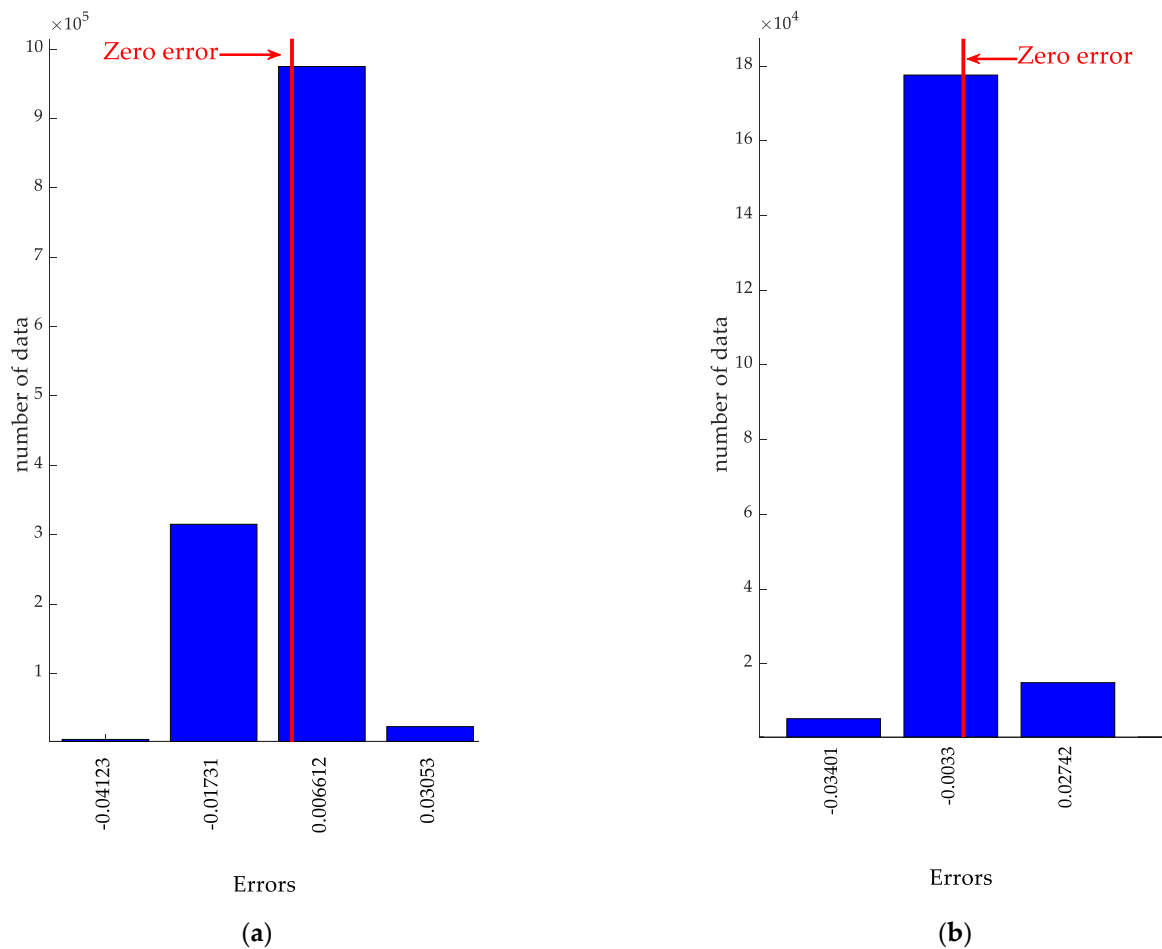


Figure 9. Error histogram for (a) training and (b) validation data testing.

3.4. ML-Based Observer

3.4.1. ML Function Block

The obtained ML model was used to estimate the values of $\sin \theta$ and $\cos \theta$. A function block of the model was constructed to implement the ML model in the simulation (Figure 10).

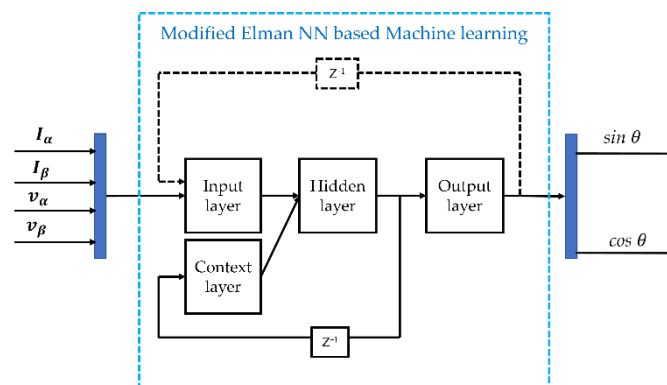


Figure 10. Function block architecture of the ML model.

3.4.2. PLL

The ML model output can be used to estimate $\sin \theta$ and $\cos \theta$; however, Figure 1 indicates that in sensorless control, the observer should be able to estimate the rotor position. Thus, a PLL was used to estimate the rotor position [11]. The PLL architecture is presented in Figure 11.

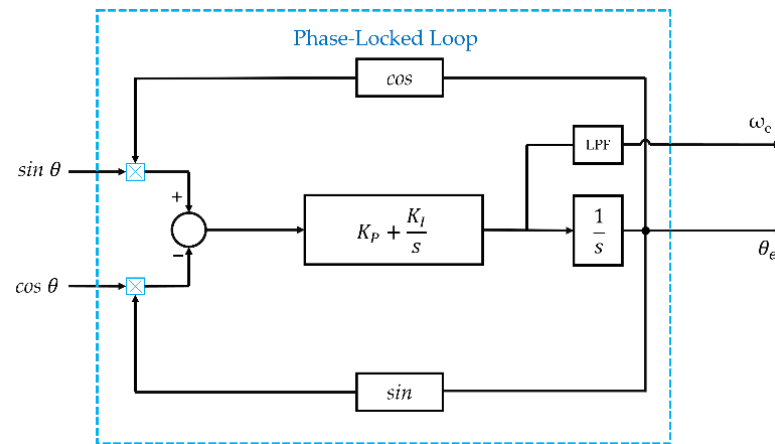


Figure 11. Phase-locked loop architecture.

The serial connection between the ML block and the PLL is the proposed ML-based observer.

4. Simulation Results

In this section, a simulated implementation of the proposed ML-based observer for a sensorless PMSM-FOC is discussed. The goal of the simulation was to determine the validity and performance of the proposed observer when it is implemented in a full-control process environment. The simulation was performed with MATLAB Simulink and was based on the architecture displayed in Figure 1. Full PMSM and inverter models are available in the Motor Control Blockset library of Simulink.

4.1. Rotor-Speed Simulation Results

Figure 12a presents the rotor-speed simulation results. The load was 0.1 p.u.-torque (0.027 Nm). From the standstill condition and for a predetermined time (4 s), the motor rotated in a ramp speed with the Volt/Hz strategy to the 0.1 p.u.-speed (410 rpm). When the reference speed exceeded 0.1 p.u.-speed, the sensorless closed-loop process was enabled.

In the increasing speed simulation, the speed step references were 0.18, 0.26, 0.32, and 0.48 p.u.-speed and continued to 0.36, 0.44, 0.54, and 0.6 p.u.-speed; the ramp speed was increased from 0.24 to 0.65 p.u.-speed. Most of the stepped reference-speed and ramping reference-speed values were not included in the training data (Figure 6); however, the simulated system still performed highly.

A similar result was observed in the decreasing speed simulation. Decreasing speed variations from 0.65, 0.55, 0.45, and 0.25 p.u.-speed and speed variations from 0.65 to 0.2 p.u.-speed were also not included in the training data. However, the proposed observer was still effective. Figure 12a reveals that the suggested method's rotor speed value and the actual rotor speed both smoothly tracked the given speed reference. Thus, the proposed observer method worked successfully in transition speeds.

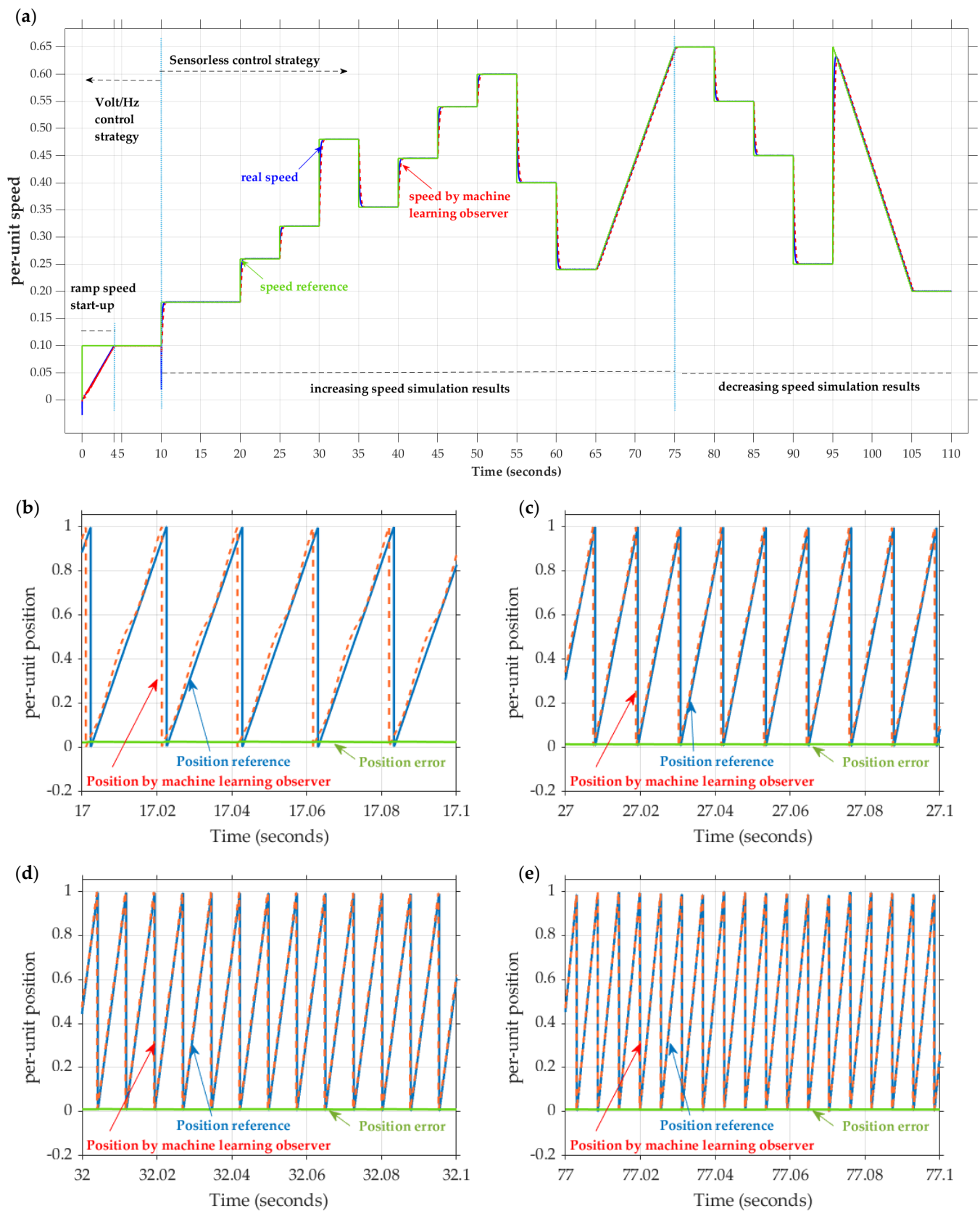


Figure 12. Simulation results for (a) rotor speed responses and for rotor positions at (b) 0.18, (c) 0.32, (d) 0.48, and (e) 0.65 p.u.-speed.

4.2. Rotor-Position Simulation Results

The performance of the proposed observer for determining the rotor position can be observed in Figure 12b–e. The position responses are presented at the four speeds of 0.18, 0.32, 0.48, and 0.65 p.u.-speed, respectively. Overall, the positions of the observer and encoder were almost identical; the average position error was close to zero (0.0127 p.u.-position).

Figure 12b, for example, presents the results for 0.18 p.u.-speed (738 rpm). The simulation result reveals that the time required for one electrical position cycle was 0.0203 s, and the time for one mechanical cycle was 0.0812 s. The rotation frequency was thus 12.4376 Hz, or 738.92 rpm. The other three speed calculation results are presented in Table 2.

Table 2. Speed calculations from rotor position response.

Speed (PU)	Speed (rpm)	Electrical Rotation Time (s)	Mechanical Rotation Time (s)	Speed Response (rpm)	Error rpm (%)
0.18	738	0.0203	0.0812	738.92	0.12
0.32	1312	0.0114	0.0456	1315.79	0.29
0.48	1968	0.0076	0.0304	1973.68	0.29
0.65	2665	0.0056	0.0224	2678.57	0.51

4.3. Current Response in the d - q Axis

Figure 13 presents the d - q current simulation responses. The proposed method is effective for the sensorless control strategy. In the sensorless mode, the current reference i_q^* was discovered to increase or decrease in accordance with the speed setting. For example, if the speed setting was increased from 0.26 p.u.-speed to 0.32 p.u.-speed, the motor required more current, causing an increase in the current i_q^* from 0.0308 to 0.0310 p.u.-current, and vice versa. In general, the current i_q smoothly followed the reference current i_q^* . The current i_d was close to zero, following the setting of the current reference i_d^* . A small spike in i_d was observed when the speed setting changed. A similar spike was observed for i_q^* and i_q in response to a stepped speed change. Larger speed changes resulted in larger spikes.

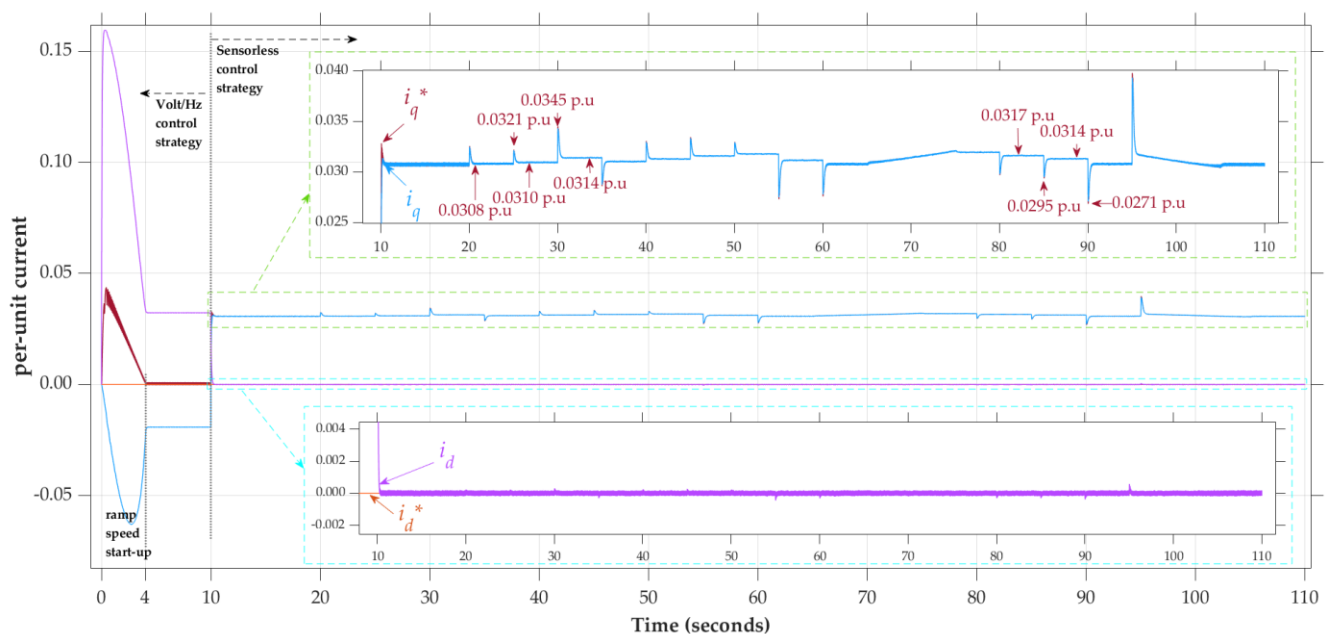


Figure 13. Simulation results for current responses in the d - q axis.

4.4. Simulation Response for a Varied External Load

The simulation response for a varied external load is presented in Figure 14. The load variation is given at 0.45 p.u.-speed. The motor was initially loaded with 0.1 of rated torque, and the load was increased to 0.2 of rated torque at $t = 25$ s. At $t = 35$ s, the load was reduced back to 0.1 of the rated torque.

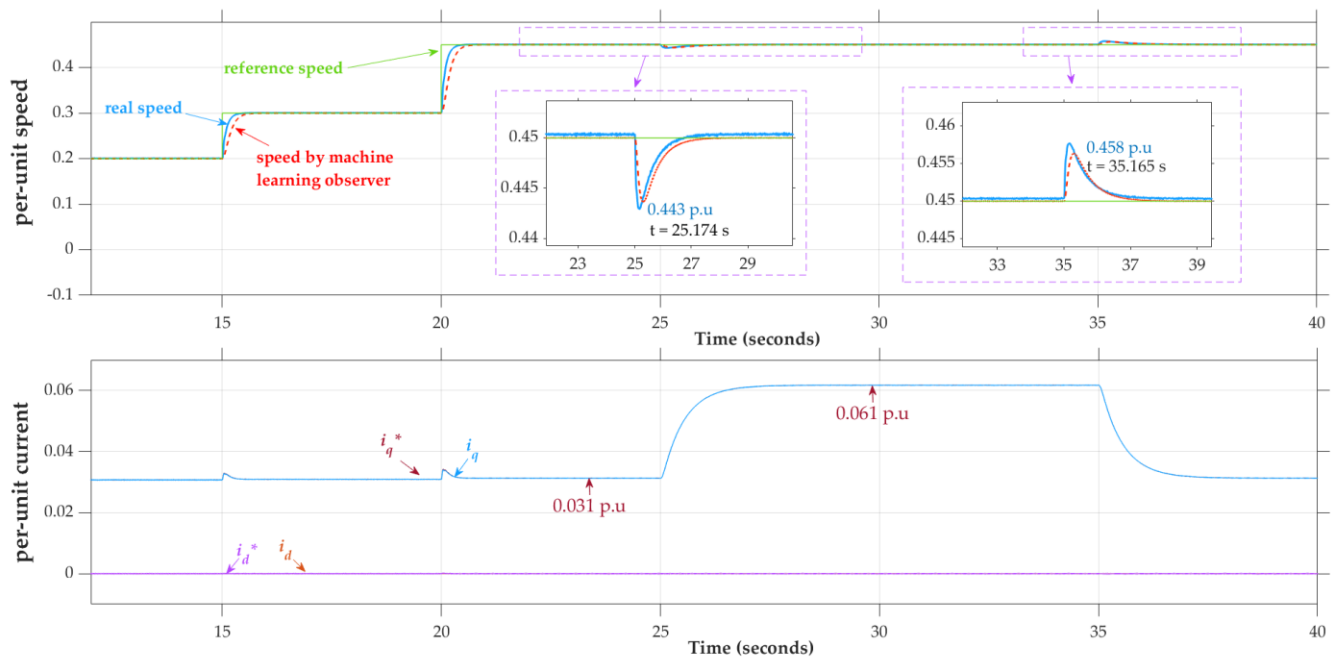


Figure 14. Speed and current simulation responses for external load variations (0.1 to 0.2 of rated torque) at 0.45 p.u.-speed.

As the load was increased, a spike occurred in the rotor-speed reduction to the minimum value of 0.443 p.u.-speed at $t = 25.174$ s. Simultaneously, the current began to increase from 0.0313 to 0.0610 p.u.-current. Under the decreased load condition ($t = 35$ s), the rotor speed spiked to a maximum value of 0.458 p.u.-speed at $t = 35.165$ s, causing the current to decrease from 0.0610 to 0.0313 p.u.-current. Although a spike in the motor speed occurred in response to a change in the load, the motor speed could return to its initial value, indicating that the proposed system is robust to load variations.

5. Experimental Hardware Realization

The proposed method was implemented on a hardware platform. The platform is depicted in Figure 15. The platform comprised a motor drive control module, a PMSM motor with the parameters listed in Table 1, a generator, and an electronic load module.

The proposed control algorithm was implemented on a TI motor-drive control consisting of TI DSP LAUNCHXL-F28069M and TI BOOSTXL-DRV8301 motor drivers. This DSP had a 90-MHz CPU, 256 kB of flash memory, 96 kB of RAM, 16 enhanced pulse-width modulation (e-PWM) channels, and 16 analog-to-digital converter (ADC) channels. Additionally, the DSP supports various serial communication protocols, including SCI, SPI, and I2C. It enables real-time command and response data transmissions to and from hardware via a Simulink host file.

In the hardware implementation, the speed control and current loop had sampling frequencies of 1 kHz and 10 kHz, respectively. The inverter's PWM switching frequency was set at 10 kHz.

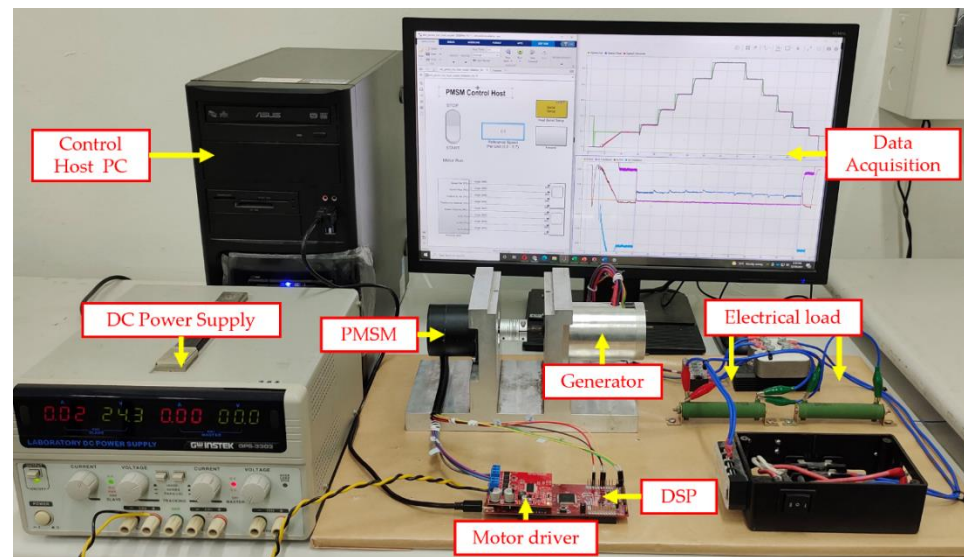


Figure 15. Experimental hardware platform.

5.1. Experimental Rotor-Speed Results

Figure 16a presents the experimental results for the rotor speed. The PMSM motor was attached to a generator but was not loaded. Initially, the motor was rotated at a ramp speed of 0.1 p.u.-speed by using the Volt/Hz method. Later, at $t = 15$ s, the reference speed was increased to 0.15 p.u.-speed, enabling the application of the sensorless control technique.

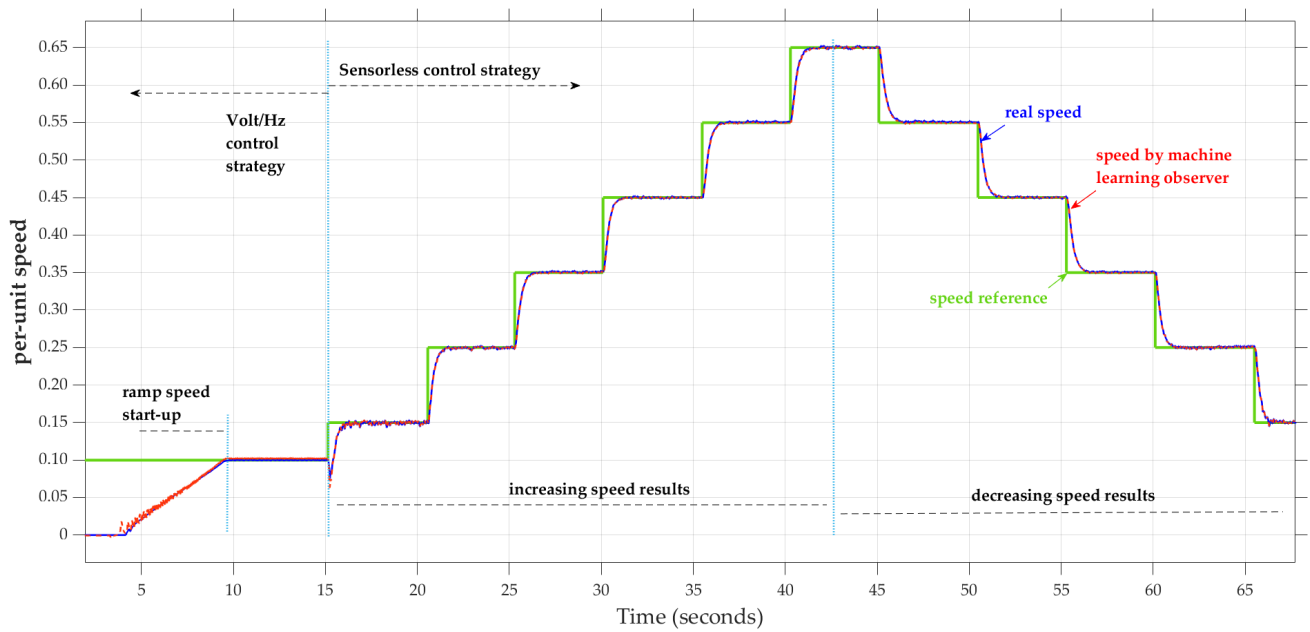
The variable speed-step references for the increasing speed experiment were 0.15, 0.25, 0.35, 0.45, 0.55, and 0.65 p.u.-speed. The experimental results revealed that both the actual motor speed and speed estimated by the proposed ML-based observer could successfully track the reference speed. The observer also successfully tracked the speed reference when the speed was reduced from 0.65 to 0.55, 0.45, 0.35, 0.25, and 0.15 p.u.-speed. Thus, the proposed observer method also worked successfully in transition speeds on this hardware realization.

5.2. Experimental Rotor-Position Results

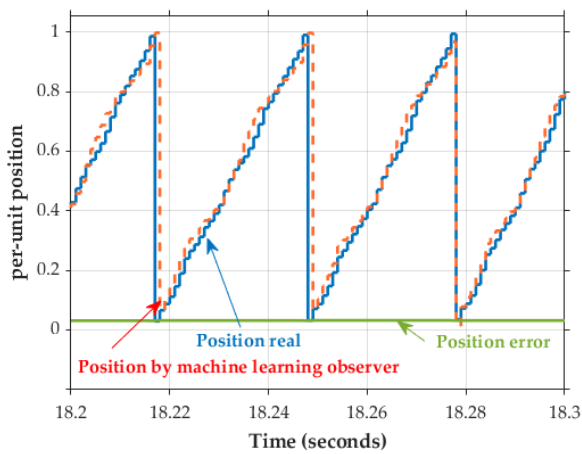
The proposed observer's performance in determining the rotor position on the experimental platform is presented in Figure 16b–e. The position responses are shown for four speed conditions: 0.15, 0.25, 0.45, and 0.65 p.u.-speed. In general, the observer's rotor positions and actual rotor positions nearly matched, and the average position error was close to zero (0.0607 p.u.-position). For example, Figure 16b presents the rotor position at 0.15 p.u.-speed (615 rpm). The duration of one electrical position cycle was 0.026 s; therefore, the time for one mechanical cycle was 0.104 s. The rotational frequency was 9.6154 Hz (576.92 rpm). The results for the remaining three speeds are presented in Table 3.

Table 3. Speed calculations from the rotor-position responses for the experimental platform.

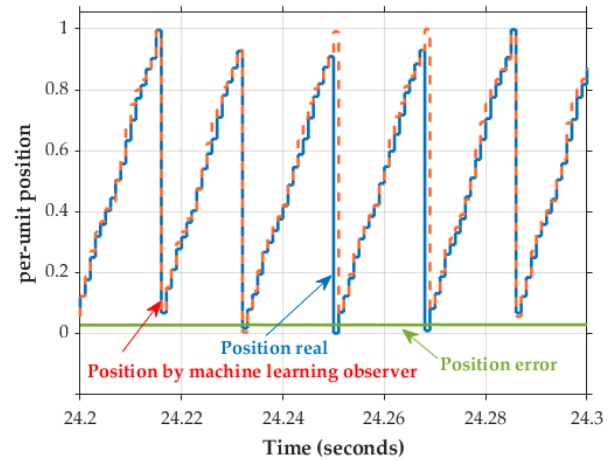
Speed (PU)	Speed (rpm)	Electrical Rotation Time (s)	Mechanical Rotation Time (s)	Speed Response (rpm)	Error rpm (%)
0.15	615	0.026	0.104	576.92	6.19
0.25	1025	0.015	0.060	1000.00	2.44
0.45	1845	0.008	0.032	1875.00	1.63
0.65	2665	0.006	0.024	2500.00	6.19



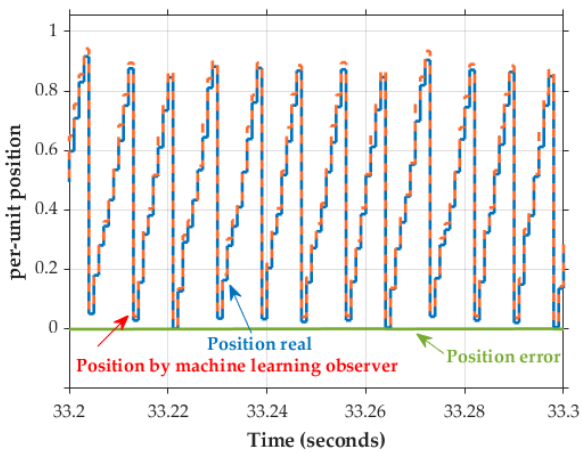
(a)



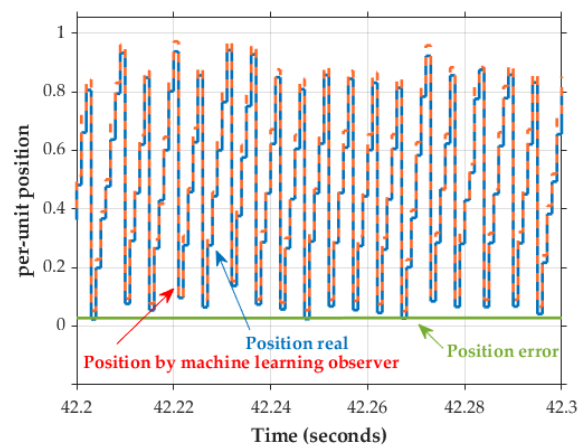
(b)



(c)



(d)



(e)

Figure 16. Experimental results for (a) rotor speed responses and for rotor positions at (b) 0.15, (c) 0.25, (d) 0.45, and (e) 0.65 p.u.-speed.

5.3. Experimental Current Response in the d - q Axis

On the basis of the experimental speed variation presented in Figure 16a, Figure 17 displays the experimental results for the d - q current responses. For example, if the speed setting was adjusted from 0.25 to 0.35 p.u.-speed, the current reference i_q^* increased from 0.0122 to 0.0129 p.u.-current. As the motor speed decreased—for example, from 0.55 to 0.45 p.u.—the current decreased from 0.0141 to 0.0134 p.u. The current i_q can generally track the current reference i_q^* . A spike in i_q^* and i_q was observed in response to the stepped speed adjustment. The experimental results indicated that the current i_d trace was close to zero relative to the reference i_d^* .

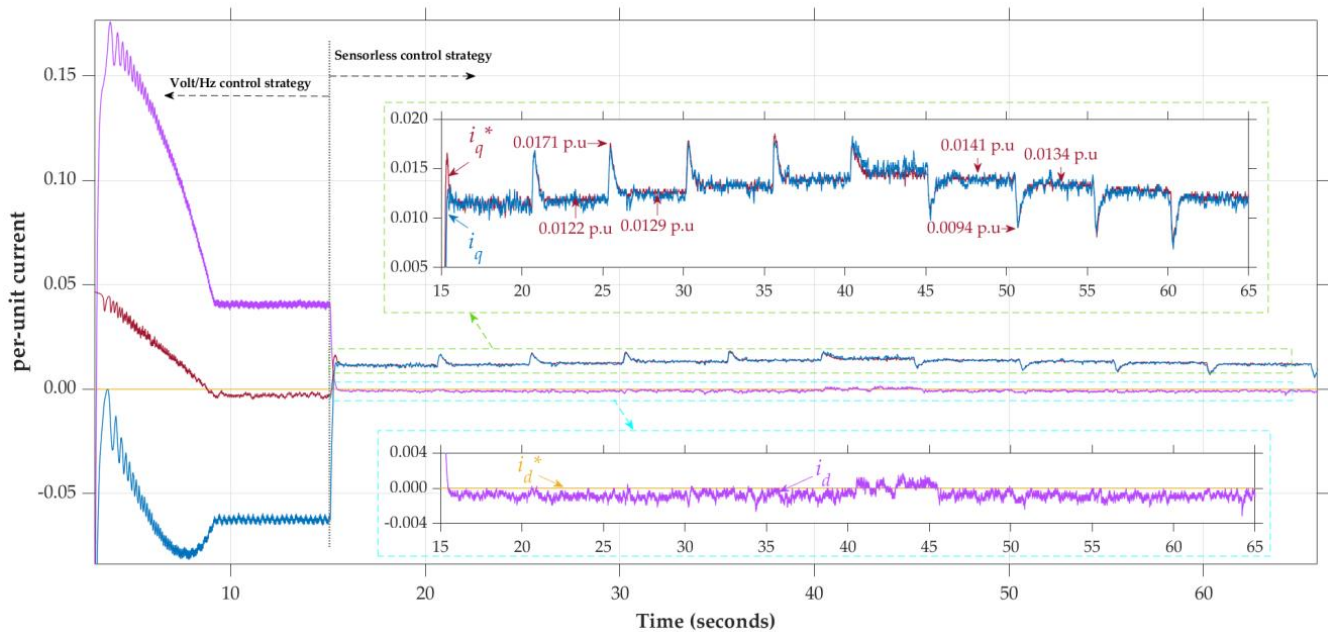


Figure 17. Experimental result for current response in the d - q axis.

5.4. Experimental Response of the Speed and Current with Varied External Load

The experimental response to varying external loads is presented in Figure 18. The motor was started using the Volt/Hz control technique; the generator was not loaded. The speed was then gradually increased from 0.1 to 0.25 p.u.-speed and finally to 0.4 p.u.-speed by using the proposed method. At $t = 26.26$ s, the generator was connected to an electrical load by using a $3\ \Omega$ resistor. The speed dropped temporarily but quickly recovered to 0.40 p.u.-speed. At $t = 26.39$ s, a minimum speed of 0.367 p.u.-speed was reached. After the load was connected, the currents i_q^* and i_q increased by 0.033 p.u.-current from 0.013 to 0.046 p.u.-current. At $t = 35.64$ s, the load was disconnected. A short spike in speed was observed on the speed response graph to a maximum at 0.440 p.u.-speed. The currents i_q^* and i_q immediately returned to 0.013 p.u.-current.

A similar result was observed when the speed was increased to 0.65 p.u.-speed and a $3\text{-}\Omega$ load was applied. The speed sharply decreased to a minimum of 0.590 p.u.-speed and sharply increased to a maximum of 0.710 p.u.-speed when the load was applied ($t = 50.90$ s) and removed ($t = 65.85$ s). Applying the load also caused increases in the currents i_q^* and i_q . These experiments revealed that the motor speed can recover quickly in response to a load, demonstrating that the proposed system is also robust when implemented in hardware.

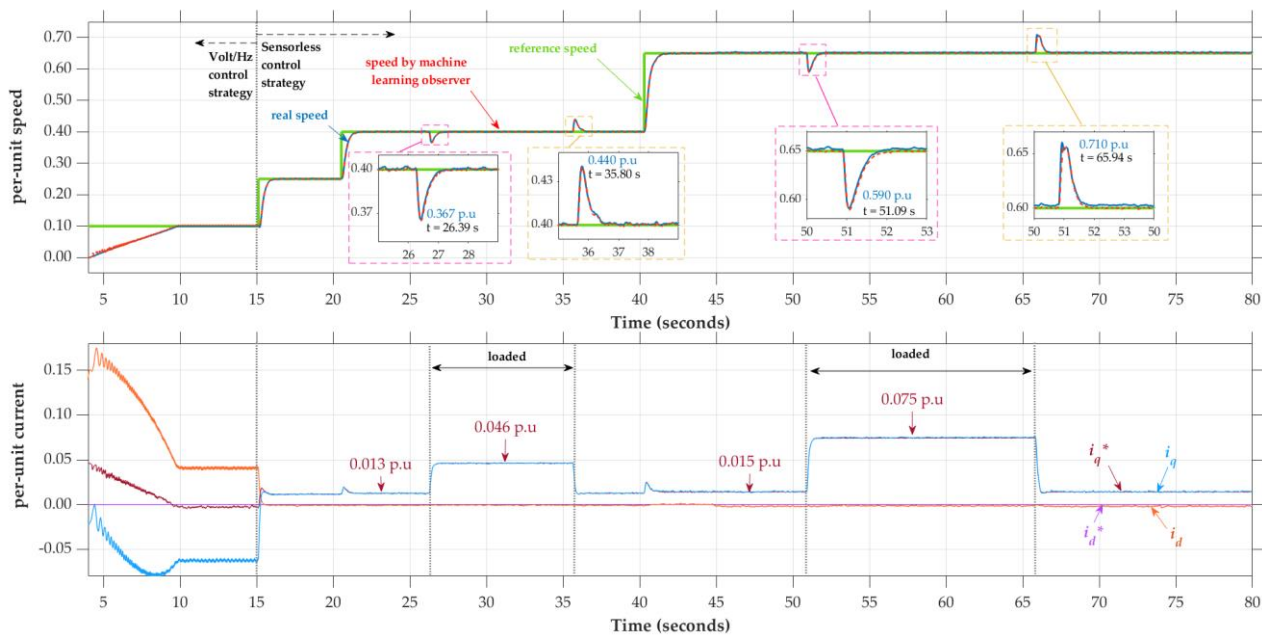


Figure 18. Experimental results for speed and current responses with external load variations at 0.40 and 0.65 p.u.-speed.

The next part discusses clarifying and verifying the roles of and linkages between the previous sections in realizing the PMSM sensorless control with ML-based observers. The proposed ML-based observer use modified the ENN as a basic algorithm of the ML model. The development of the ML model needs several steps. Section 3.2 outlined the steps. Then in Section 3.3.2, the ML model was trained and validated offline using a data-driven method, which was previously recorded from the simulation (Section 3.3.1). The validated ML model was attached to the ML function block (Section 3.4.1). The output of the function block was linked to the PLL function block (Section 3.4.2). The combination of these two function blocks formed the proposed ML-based observer. Then, the proposed ML-based observer was implemented in a full-control process environment to verify its performance. As a result, the simulation of the full-control process environment (Section 4) and experimental hardware (Section 5) confirmed that the proposed ML-based observer could be used in sensorless PMSM drive control. It can follow the transition of speed command changes well and is also robust to load changes.

6. Conclusions

An ML-based observer was successfully designed and implemented in a DSP-based sensorless PMSM-FOC process. A modified ENN algorithm, used as the ML model's basic architecture, was proven to overcome the speed transition problem. Data for training and validation were obtained from a simulated control process. The trained ML model combined with a PLL was the ML-based observer. The proposed observer was tested in a full-environment-control process simulation. An open-loop Volt/Hz startup strategy was used at the beginning of the control process, followed by the proposed sensorless control strategy. A hardware control platform based on DSP F28069M was used to experimentally implement the proposed observer. Compared to the actual rotor position, the rotor position estimated by the proposed ML-based observer has an average error of 0.0127 p.u.-position on simulation and 0.0607 p.u.-position on hardware implementation. Thus, the sensorless PMSM drive can be controlled well to follow the commanded speed and is robust to changes in load. Both the simulation and experimental results indicate that the proposed observer is effective as a sensorless PMSM-drive control system.

Author Contributions: D.S.P. wrote this article, designed the control method, implemented the hardware platform, drew the figures, and performed the simulation as well as the experiment. S.-C.C. supervised the study, coordinated the investigations, and checked the manuscript's logical structure. H.-H.K. validated the simulation and hardware experimental data and reviewed the writing and editing. F.C. provided practical suggestions and evaluated the feasibility of the application. All authors have read and agreed to the published version of the manuscript.

Funding: This research received funding from Imeier Green Technology Co., Ltd. with contract no. 14001110024.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Chung, S.-U.; Moon, S.-H.; Kim, D.-J.; Kim, J.-M. Development of a 20-Pole–24-Slot SPMSM With Consequent Pole Rotor for In-Wheel Direct Drive. *IEEE Trans. Ind. Electron.* **2016**, *63*, 302–309. [\[CrossRef\]](#)
2. Park, H.-I.; Choi, J.; Jeong, K.-H.; Cho, S. Comparative analysis of surface-mounted and interior permanent magnet synchronous motor for compressor of air-conditioning system in electric vehicles. In Proceedings of the 2015 9th International Conference on Power Electronics and ECCE Asia (ICPE-ECCE Asia), Seoul, Korea, 1–5 June 2015. [\[CrossRef\]](#)
3. Mocanu, R.; Onea, A. Phase resistance estimation and monitoring of PMSM used in electrical vehicles. In Proceedings of the 2014 18th International Conference on System Theory, Control and Computing (ICSTCC), Sinaia, Romania, 17–19 October 2014; pp. 512–519.
4. Nak, H.; Gülbahce, M.O.; Gokasan, M.; Ergene, A.F. Performance investigation of extended Kalman filter based observer for PMSM using in washing machine applications. In Proceedings of the 2015 9th International Conference on Electrical and Electronics Engineering (ELECO), Brno, Czech Republic, 24–26 November 2015; pp. 618–623.
5. Consoli, A.; Scarcella, G.; Testa, A. Industry application of zero-speed sensorless control techniques for PM synchronous motors. *IEEE Trans. Ind. Appl.* **2001**, *37*, 513–521. [\[CrossRef\]](#)
6. Hyunbae, K.; Lorenz, R.D. Carrier signal injection based sensorless control methods for IPM synchronous machine drives. In Proceedings of the Conference Record of the 2004 IEEE Industry Applications Conference, Seattle, WA, USA, 3–7 October 2004; 39th IAS Annual Meeting. Volume 2, pp. 977–984.
7. Raca, D.; Garcia, P.; Reigosa, D.; Briz, F.; Lorenz, R.D. Carrier Signal Selection for Sensorless Control of PM Synchronous Machines at Zero and Very Low Speeds. In Proceedings of the 2008 IEEE Industry Applications Society Annual Meeting, Edmonton, AB, Canada, 5–9 October 2008; pp. 1–8.
8. Qiao, Z.; Shi, T.; Wang, Y.; Yan, Y.; Xia, C.; He, X. New Sliding-Mode Observer for Position Sensorless Control of Permanent-Magnet Synchronous Motor. *IEEE Trans. Ind. Electron.* **2013**, *60*, 710–719. [\[CrossRef\]](#)
9. Kim, H.; Son, J.; Lee, J. A High-Speed Sliding-Mode Observer for the Sensorless Speed Control of a PMSM. *IEEE Trans. Ind. Electron.* **2011**, *58*, 4069–4077. [\[CrossRef\]](#)
10. Lachman, T.; Mohamad, T.R.; Teo, S.P. Sensorless position estimation of switched reluctance motors using artificial neural networks. In Proceedings of the IEEE International Conference on Robotics, Intelligent Systems and Signal Processing, Changsha, China, 8–13 October 2003; Volume 1, pp. 220–225.
11. Hoai, H.-K.; Chen, S.-C.; Than, H. Realization of the Sensorless Permanent Magnet Synchronous Motor Drive Control System with an Intelligent Controller. *Electronics* **2020**, *9*, 365. [\[CrossRef\]](#)
12. Ramírez-Cárdenas, O.-D.; Trujillo-Romero, F. Sensorless Speed Tracking of a Brushless DC Motor Using a Neural Network. *Math. Comput. Appl.* **2020**, *25*, 57. [\[CrossRef\]](#)
13. Sai Shiva, B.; Verma, V.; Khan, Y.A. Q-MRAS-Based Speed Sensorless Permanent Magnet Synchronous Motor Drive with Adaptive Neural Network for Performance Enhancement at Low Speeds. In *Innovations in Soft Computing and Information Technology*; Chattopadhyay, J., Singh, R., Bhattacharjee, V., Eds.; Springer: Singapore, 2019; pp. 103–116.
14. Kumar, R.; Padmanaban, S.V. An Artificial Neural Network Based Rotor Position Estimation for Sensorless Permanent Magnet Brushless DC Motor Drive. In Proceedings of the IECON 2006—32nd Annual Conference on IEEE Industrial Electronics, Paris, France, 6–10 November 2006; IEEE: Paris, France, 2006; pp. 649–654.
15. Makni, Z.; Zine, W. Rotor position estimator based on machine learning. In Proceedings of the IECON 2016—42nd Annual Conference of the IEEE Industrial Electronics Society, Florence, Italy, 23–26 October 2016; IEEE: Florence, Italy, 2016; pp. 6687–6692.
16. Zine, W.; Makni, Z.; Monmasson, E.; Idkhajine, L.; Condamine, B. Interests and Limits of Machine Learning-Based Neural Networks for Rotor Position Estimation in EV Traction Drives. *IEEE Trans. Ind. Inform.* **2018**, *14*, 1942–1951. [\[CrossRef\]](#)
17. Liu, Y.; Fang, J.; Tan, K.; Huang, B.; He, W. Sliding Mode Observer with Adaptive Parameter Estimation for Sensorless Control of IPMSM. *Energies* **2020**, *13*, 5991. [\[CrossRef\]](#)

18. An, Q.; An, Q.; Liu, X.; Zhang, J.; Bi, K. Improved Sliding Mode Observer for Position Sensorless Control of Permanent Magnet Synchronous Motor. In Proceedings of the 2018 IEEE Transportation Electrification Conference and Expo, Asia-Pacific (ITEC Asia-Pacific), Bangkok, Thailand, 6–9 June 2018; pp. 1–7.
19. Wu, R.; Slemon, G.R. A permanent magnet motor drive without a shaft sensor. *IEEE Trans. Ind. Appl.* **1991**, *27*, 1005–1011. [\[CrossRef\]](#)
20. Ogasawara, S.; Akagi, H. An approach to position sensorless drive for brushless DC motors. *IEEE Trans. Ind. Appl.* **1991**, *27*, 928–933. [\[CrossRef\]](#)
21. Lee, K.-W.; Kim, D.-K.; Kim, B.-T.; Kwon, B.-I. A Novel Starting Method of the Surface Permanent-Magnet BLDC Motors Without Position Sensor for Reciprocating Compressor. *IEEE Trans. Ind. Appl.* **2008**, *44*, 85–92. [\[CrossRef\]](#)
22. Itoh, J.-I.; Nomura, N.; Ohsawa, H. A comparison between V/f control and position-sensorless vector control for the permanent magnet synchronous motor. In Proceedings of the Proceedings of the Power Conversion Conference-Osaka 2002 (Cat. No.02TH8579), Osaka, Japan, 2–5 April 2002; IEEE: New York, NY, USA, 2002; Volume 3, pp. 1310–1315.
23. Perera, P.D.C.; Blaabjerg, F.; Pedersen, J.K.; Thogersen, P. A sensorless, stable V/f control method for permanent-magnet synchronous motor drives. *IEEE Trans. Ind. Appl.* **2003**, *39*, 783–791. [\[CrossRef\]](#)
24. Elman, J.L. Finding structure in time. *Cogn. Sci.* **1990**, *14*, 179–211. [\[CrossRef\]](#)
25. Sastry, P.S.; Santharam, G.; Unnikrishnan, K.P. Memory neuron networks for identification and control of dynamical systems. *IEEE Trans. Neural Netw.* **1994**, *5*, 306–319. [\[CrossRef\]](#) [\[PubMed\]](#)
26. Cacciola, M.; Megali, G.; Pellicanó, D.; Morabito, F.C. Elman neural networks for characterizing voids in welded strips: A study. *Neural Comput. Appl.* **2012**, *21*, 869–875. [\[CrossRef\]](#)
27. Wan, W.; Xu, H.; Zhang, W.; Hu, X.; Deng, G. Questionnaires-based skin attribute prediction using Elman neural network. *Neurocomputing* **2011**, *74*, 2834–2841. [\[CrossRef\]](#)
28. Bhushan, B.; Singh, M.; Hage, Y. Identification and control using MLP, Elman, NARXSP and radial basis function networks: A comparative analysis. *Artif. Intell. Rev.* **2012**, *37*, 133–156. [\[CrossRef\]](#)
29. Alhmoud, L.; Nawafleh, Q. Short-Term Load Forecasting for Jordan Power System Based on NARX-ELMAN Neural Network and ARMA Model. *IEEE Can. J. Electr. Comput. Eng.* **2021**, *44*, 356–363. [\[CrossRef\]](#)
30. Kamanditya, B.; Kusumoputro, B. Elman Recurrent Neural Networks Based Direct Inverse Control for Quadrotor Attitude and Altitude Control. In Proceedings of the 2020 International Conference on Intelligent Engineering and Management (ICIEM), London, UK, 17–19 June 2020; pp. 39–43.
31. Jia, W.; Zhao, D.; Shen, T.; Tang, Y.; Zhao, Y. Study on Optimized Elman Neural Network Classification Algorithm Based on PLS and CA. *Comput. Intell. Neurosci.* **2014**, *2014*, e724317. [\[CrossRef\]](#)
32. Al-Jamali, N.A.S.; Al-Raweshidy, H.S. Modified Elman Spike Neural Network for Identification and Control of Dynamic System. *IEEE Access* **2020**, *8*, 61246–61254. [\[CrossRef\]](#)
33. Li, P.; Chi, Q.; Wan, K.; Zhang, X.; Yang, C.; Wang, S. Intelligent Substation Communication Network Traffic Predicting Based on Improved Elman Neural Network. In Proceedings of the 2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE), Xiamen, China, 18–19 October 2019; pp. 1841–1845.
34. Kung, Y.-S.; Risfendra, R.; Lin, Y.-D.; Huang, L.-C. FPGA-realization of a sensorless speed controller for PMSM drives using novel sliding mode observer. *Microsyst. Technol.* **2018**, *24*, 79–93. [\[CrossRef\]](#)
35. Lu, Q.; Quan, L.; Zhu, X.; Zuo, Y.; Wu, W. Improved Sliding Mode Observer for Position Sensorless Open-Winding Permanent Magnet Brushless Motor Drives. *Prog. Electromagn. Res. M* **2019**, *77*, 147–156. [\[CrossRef\]](#)
36. Motor Control Blockset Documentation. Available online: <https://www.mathworks.com/help/releases/R2021a/mcb/index.html> (accessed on 21 November 2021).
37. Nicola, C.-I.; Nicola, M.; Selișteanu, D. Sensorless Control of PMSM Based on Backstepping-PSO-Type Controller and ESO-Type Observer Using Real-Time Hardware. *Electronics* **2021**, *10*, 2080. [\[CrossRef\]](#)
38. Mitchell, T.M. *Machine Learning*; McGraw Hill: New York, NY, USA, 1997.
39. Per-Unit System-MATLAB & Simulink. Available online: <https://www.mathworks.com/help/mcb/gs/per-unit-system.html> (accessed on 16 December 2021).