mathematics

MDPI

Article

Fractional Order Sliding Mode Controller Based on Supervised Machine Learning Techniques for Speed Control of PMSM

Younes Zahraoui [1], Fardila M. Zaihidee [2,*], Mostefa Kermadi [3], Saad Mekhilef [3,4,5], Marizan Mubin [3], Jing Rui Tang [2] and Ezrinda M. Zaihidee [6]

[1] FinEst Centre for Smart Cities, Tallinn University of Technology, Ehitajate Tee 5, 19086 Tallinn, Estonia
[2] Faculty of Technical and Vocational, Sultan Idris Education University, Tanjong Malim 35900, Malaysia
[3] Power Electronics and Renewable Energy Research Laboratory (PEARL), Department of Electrical Engineering, Faculty of Engineering, University of Malaya, Kuala Lumpur 50603, Malaysia
[4] School of Software and Electrical Engineering, Faculty of Science, Engineering and Technology, Swinburne University of Technology, Victoria, VIC 3122, Australia
[5] Institute of Sustainable Energy, Universiti Tenaga Nasional (The National Energy University), Jalan Ikram-Uniten, Kajang 43000, Malaysia
[6] Centre for Mathematical Sciences, Universiti Malaysia Pahang, Gambang 26300, Malaysia
* Correspondence: fardila@ftv.upsi.edu.my

Abstract: Tracking the speed and current in permanent magnet synchronous motors (PMSMs) for industrial applications is challenging due to various external and internal disturbances such as parameter variations, unmodelled dynamics, and external load disturbances. Inaccurate tracking of speed and current results in severe system deterioration and overheating. Therefore, the design of the controller for a PMSM is essential to ensure the system can operate efficiently under conditions of parametric uncertainties and significant variations. The present work proposes a PMSM speed controller using machine learning (ML) techniques for quick response and insensitivity to parameter changes and disturbances. The proposed ML controller is designed by learning fractional-order sliding mode control (FOSMC) controller behavior. The primary purpose of using ML in FOSMC is to avoid the self-tuning of the parameters and ensure the speed reaches the reference value in finite time with faster convergence and better tracking precision. Furthermore, the ML model does not require the mathematical model of the speed controller. In this work, several ML models are empirically evaluated on their estimation accuracy for speed tracking, namely ordinary least squares, passive-aggressive regression, random forest, and support vector machine. Finally, the proposed controller is implemented on a real-time hardware-in-the-loop (HIL) simulation platform from PLECS Inc. Comparative simulation and experimental results are presented and discussed. It is shown from the comparative study that the proposed FOSMC based on ML outperformed the traditional sliding mode control (SMC), which is more commonly used in industry in terms of tracking speed and accuracy.

Keywords: machine learning; sliding mode control; permanent magnet synchronous motors; motor control; disturbance estimation

MSC: 68T07

Citation: Zahraoui, Y.; Zaihidee, F.M.; Kermadi, M.; Mekhilef, S.; Mubin, M.; Tang, J.R.; Zaihidee, E.M. Fractional Order Sliding Mode Controller Based on Supervised Machine Learning Techniques for Speed Control of PMSM. *Mathematics* **2023**, *11*, 1457. https://doi.org/10.3390/math11061457

Academic Editor: Denis N. Sidorov

Received: 19 January 2023
Revised: 6 February 2023
Accepted: 15 February 2023
Published: 17 March 2023

1. Introduction

Permanent magnet synchronous motors (PMSM) have great potential for high-precision applications due to their advantages in terms of high efficiency, high power density, and reliability [1]. However, the system uncertainties and unavoidable external disturbances may change the time-varying parameters with high-order complex dynamics in the PMSM speed control system [2]. In fact, the dynamical performance and steady-state performance are two important aspects of the PMSM's control system. Steady-state performance is

ensured when the controller can track the reference speed with minimum tracking error and the lowest torque ripple. In terms of dynamic performance, the controller must provide a quick convergence with the lowest overshoot possible [3].

Over the years, classical linear control, such as the proportional-integral-derivative (PID) controller, has been applied in industrial applications to control the PMSM drives due to its simple implementation [4]. Nevertheless, this type of controller has faced some issues because of the nonlinearity and load disturbance changes, which result in complicated precise speed regulation and unsatisfactory performance [5]. To enhance the control performance, many nonlinear control methods have been proposed for the PMSM drive, such as predictive control [6,7], finite-time control [3,8], backstepping control [9,10], disturbance rejection control [11,12], and adaptive control [13,14].

Among these controllers, much attention was given to sliding mode control (SMC), one of the most effective nonlinear control methods widely applied for speed regulation [15]. SMC can enhance the performance of the PMSM drive system's tracking speed, accuracy, and torque dynamics. Moreover, SMC guarantees consistent tracking although the system is exposed to internal parameter fluctuations or external disturbances. In addition, the remarkable features of SMC are its excellent accuracy and simplicity. Several studies of the SMC scheme have been conducted to enhance the dynamic performance of the PMSM speed regulation system, such as in [16,17].

Generally, in the SMC approach, a large switching gain is applied to suppress disturbances such as fluctuating load torque, which results in a discontinuous control law and substantial chattering with high-frequency oscillation [18]. To overcome the aforementioned obstacles in SMC, researchers proposed various methods to reduce the reaching phase and ensure to reach on a sliding surface [1]. For instance, in [19], the authors introduced a non-singular terminal sliding mode manifold for the speed loop. The proposed controller ensures the states reach the manifold in finite time and converge to the equilibrium point in a finite time. Thus, the controller could support the PMSM drive to reach the reference speed in a finite time, obtaining a faster convergence with excellent tracking accuracy. Moreover, this study proposed a composite terminal sliding mode control method based on a disturbance observer to eliminate chattering. The authors in [20] proposed a hybrid terminal sliding-mode observer based on the non-singular terminal sliding mode (NTSM) and the high-order sliding mode (HOSM) for the rotor position and speed regulation of the PMSM. The NTSM manifold and derivative estimator are applied to realize fast convergence and to obtain the derivative of the sliding-mode function. Meanwhile, a HOSM control law is used to ensure the stability of the observer and reduce the chattering without a low-pass filter. In [21], a super-twisting sliding mode (STSM) control technique combining generalized proportional integral observer (GPIO) was investigated. A GPIO was introduced to estimate the lumped disturbance to reduce the control gain's value and to enhance the PMSM system's robustness for satisfactory dynamical performance. In [22], a sliding-mode control method using a new sliding-mode reaching law (NSMRL) was presented. The controller consists of the system state variable and the power term in the sliding-surface function. The proposed technique effectively suppresses the inherent chattering and increases the velocity of the system state reaching the sliding surface. Based on NSMRL, the authors designed a sliding-mode speed controller (SMSC) to replace the traditional proportional-integral controller (PI). In [23], a continuous fast terminal sliding-mode control (CFTSMC) was introduced for the speed regulation of the PMSM drive system. The proposed technique has the capability to minimize the error quickly, thus providing higher-precision tracking and robustness against external disturbances. The extended state observer (ESO) estimates the system disturbances for feed-forward compensation and reduces the switching gain value. In [24], the conventional SMC was improved by changing its sliding manifold design to obtain a fractional order sliding-mode control (FOSMC) for the speed of a PMSM. The proposed FOSMC was designed with differentiation and integration of the sliding surface.

Besides the SMC methods, intelligent control techniques, such as fuzzy control and artificial neural network (ANN) control, were extensively applied for the PMSM speed regulator due to their strong optimization and capability to deal with uncertain and nonlinear problems. The authors in [25] proposed an adaptive interval type-2 fuzzy logic control scheme for high-performance PMSM. This technique utilizes the power of type-2 fuzzy logic systems based on adaptive control theory to enhance precision tracking and robustness against higher uncertainties. The proposed controller does not need electrical transducers. Therefore, no explicit current loop regulation is required, which yields a simplified control scheme. In [26], the synthesis and the properties of a neural speed controller trained online were introduced using the resilient backpropagation (RPROP) algorithm with ANN. The specific properties of the speed controller in the PMSM drive were automatically adapted and tuned. In [27], an intelligent control system using a recurrent wavelet-based Elman neural network (RWENN) for PMSM speed regulators was presented. The proposed intelligent optimal RWENN control system (IORWENNCS) incorporates an optimal controller, a RWENN controller, and a robust controller for optimal control and to reduce the quadratic performance index. To guarantee the stability of IORWENNCS, online adaptive control laws derived based on the optimal control technique and Lyapunov stability analysis are applied. In [28], an Elman neural network (ENN) based on a complementary sliding mode control (CSMC) was presented. The CSMC is applied by combining the integral sliding surface with the complementary sliding surface to minimize the chattering phenomenon. In addition, it was proposed that the ENN overcomes the problem of switching gain and boundary layer thickness due to its strong learning ability. Even though the enhanced SMC and the intelligent control techniques have a continuous law, which has the ability to eliminate the chattering problem, these methods need high switching gain during the multi-speed step change, which introduces chattering and large steady-state fluctuations in the speed regulation of the PMSM. Moreover, these types of controllers need parameter-tuning methods for a quick response [29].

In that context, this paper proposes a speed control scheme using the ML technique, designed by learning fractional-order sliding mode control (FOSMC). The main objective of this work is to design an advanced non-linear control technique to solve the speed regulation control problem of the PMSM under a control input constraint. Moreover, the ML controller is investigated on a speed controller to avoid self-tuning of parameters, as well as to reduce the effects of parametric uncertainty and complicated load fluctuation. Several ML techniques are built, analyzed, and compared against the conventional SMC in a real-time hardware-in-the-loop (HIL) simulation environment using the RT Box from PLECS. The controllers are evaluated and compared under different testing profiles in terms of tracking speed and accuracy and reference overshoot.

This paper is organized into the following sections. In Section 2, the mathematical model of the PMSM and the speed controller are introduced. In Sections 3 and 4, the machine learning algorithm and the methodology are described. The relevant results and detailed analysis are given in Section 5. Finally, Section 6 concludes this study.

2. Mathematical Model of the Speed Regulation System of the PMSM

2.1. Dynamic Modeling of PMSM Drive

Under the synchronously rotating reference frame, the PMSM mathematical model in the *d*-*q* axis coordination can be presented as follows [30]:

$$\begin{cases} i'_d = \frac{-R_s}{L_d} i_d + n_p \omega i_q + \frac{u_d}{L_d} \\ i'_q = -n_p \omega i_d - \frac{R_s}{L_q} i_q - \frac{n_s \psi_f}{L_q} \omega + \frac{u_q}{L_q} \\ \dot{\omega} = \frac{1.5 n_p \psi_f}{J} i_q - \frac{B}{J} \omega - \frac{T_l}{J} \end{cases} \quad (1)$$

where the superscript $'$ denotes the derivation of a variable with respect to time. i'_d and i'_q are the derivative of the stator current components in the rotating frame. i_d and i_q are the *d*- and *q*-axis stator axis. u_d and u_q represent the *d*- and *q*-axis stator voltage. L_d and L_q

represent the d- and q-axis stator inductance. R_s describes the stator winding resistance. ω represents the angular velocity. B is the viscous friction coefficient. n_p represents the number of pole pairs in the PMSM. T_l is the load torque, and ψ_f represents the flux linkage. $\dot{\omega}$ represents the second-order angular velocity model of the PMSM. J is the moment of inertia. The output of the PMSM system is speed ω. From Equation (1), the PMSM dynamic equation can be described as:

$$\begin{aligned} \dot{\omega} &= \frac{1.5 n_p \psi_f}{J} i_q - \frac{B}{J}\omega - \frac{T_l}{J} \\ &= \frac{1.5 n_p \psi_f}{J} i_q^* - \frac{B}{J}\omega - \frac{T_l}{J} - \frac{1.5 n_p \psi_f}{J}(i_q^* - i_q) \\ &= b i_q^* + d(t) \end{aligned} \tag{2}$$

where $b = 1.5 n_p \psi_f / J$ and $d(t) = -(B\omega/J) - (T_l/J) - (1.5 n_p \psi_f / J)(i_q^* - i_q)$, which can be represented as the lumped disturbance, including the friction, the external load disturbance, and the tracking error of the d- and q-axis current loop.

In practice, the internal parameters of the PMSM drive are affected by the temperature. Thus, a limited scale is used to represent the range of internal parameter variations. The nominal value of the load torque must be limited. Therefore, the lumped disturbances i_d', i_q', and $\dot{\omega}$ represent the derivatives.

A block diagram of the PMSM drive system in this work is shown in Figure 1. In order to decouple the speed and currents, the vector control strategy of the reference current d-axis is set to 0. Two PI controllers, which are used to stabilize the $d-q$ axes current errors, are adopted in the two current loops.

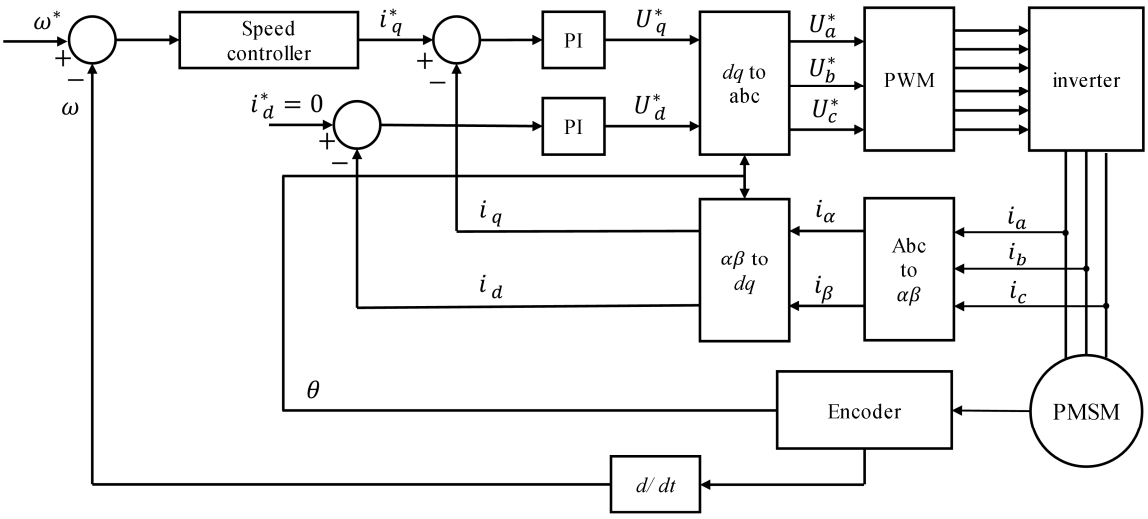

Figure 1. Block diagram of the PMSM vector control system.

2.2. Design of Fractional-Order Sliding Mode Control (FOSMC) for Speed Control of PMSM

The speed loop aims to obtain the optimal output signal to control the PMSM speed. The speed regulator ensures the motor speed ω is tracking the reference speed ω^* accurately by driving the tracking error asymptotically to zero. The output signal can be described as [30]:

$$e = \omega^* - \omega \tag{3}$$

where e is the asymptotical error, ω^* is the reference angular velocity, and ω is the real angular velocity. In SMC, the fractional-order sliding surface can be described as:

$$s = K_p e(t) + D_t^r e(t) \tag{4}$$

where K_p is a positive number, and D_t^r is a fractional order integral where $0 < r < 1$

$$\dot{s} = -w\, s - k_s \cdot sign(s) \tag{5}$$

The control law is given by [31]:

$$s = K_p e(t) + K_i D_t^{-\alpha} e(t) + K_d D_t^{\beta} e(t) \tag{6}$$

where ws and $k_s \in \mathbb{R}$, and $sign(.)$ is the sign function.

In this work, the fractional order sliding surface for FOSMC is given as [24]:

$$i_q(t) = (bK_p)^{-1} \begin{bmatrix} K_{i0} D_t^{1-\alpha} e(t) + K_{d0} D_t^{\beta+1} e(t) \\ +(w-a) K_p e(t) \\ +K_p e(t) + w K_{i0} D_t^{-\alpha} e(t) \\ +w K_{d0} D_t^{\beta} e(t) + K_s \cdot sign(s) \end{bmatrix} \tag{7}$$

where K_d, K_i, K_p are the fractional controller coefficients, and α and β are the order of fractional integration and differentiation, respectively. The order of fractional integration and differentiation helps to improve the stability, eliminate any steady-state error, and ensure that the controlled system converges to the desired setpoint.

3. Machine Learning Algorithms

The integration of machine learning (ML) into the control loop allows for predicting the behavior of non-linear dynamic systems. In other words, ML can find the optimal control signal by mapping between inputs and outputs based on a training dataset and then making predictions of the inputs that have never been trained. In order to find a reliable general representation of inputs and outputs, the training dataset for the generalization model must contain a complete representative set of data. In general, the ML dataset $S = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ is given to the learning method, with fixed and unknown distribution D, where N represents the size of the dataset. In each instance, x_i is a vector of the form $x_i = \left(x_i^1, \ldots, x_i^M\right)$. Each value $x_i^1, \ldots, x_i^M$ is relative to each feature $X_1, \ldots, X_M$. Y is a special feature called class. $Y_i, i = 1, \ldots, N$ represents a set of labels associated with each instance, x_i. If all sets $Y_i, i = 1, \ldots, N$ have only one value, the problem is considered a single label. Hence, the ML models have one output value for prediction. However, some ML architecture cannot be treated as a single-label problem. In this study, the output value from the speed controller has a single value. Therefore, the ML model is considered as a single label.

In practice, the ML models categorize based on their modeling capabilities, previous data assumptions, the number of model parameters and their update methods, and the runtime. Representative classes for the regression task in the ML model can be represented as linear models such as ordinary least squares and passive-aggressive regression, or non-linear regression, such as random forest and support vector machine. In this section, the ML models used for the regression are presented. Moreover, some notations required to formulate the ML models mathematically are briefly described.

3.1. Ordinary Least Squares

Ordinary least squares (OLS) regression in the ML model is a generalized linear modeling technique, which is used to model a single-label problem that has been recorded on at least an interval scale. The OLS technique can be used with a single explanatory variable, multiple explanatory input, or categorical explanatory variables that have been correctly coded. The ordinary least squares regression model can find a linear relationship between the responses (input vector, $X \in \mathbb{R}$) and the predictor (output vector, Y). Consider the following linear regression model [32]:

$$Y_i = \hat{\alpha} + \hat{\beta} X_i + \varepsilon_i \tag{8}$$

where ε_i represents the error between the values of the responses and the predictors, which are also known as the residuals. In the ordinary least squares regression case, the sum of the squared residuals is given as:

$$\sum_{i=1}^{n} \left(Y_i - (\hat{\alpha} + \hat{\beta} X_i)\right)^2 \tag{9}$$

where $\hat{\alpha}$ and $\hat{\beta}$ are the parameter of the linear model that minimizes the sum of the residual. $\hat{\alpha}$ and $\hat{\beta}$ are defined by:

$$\hat{\alpha} = \overline{Y} - \hat{\beta}\overline{X}\hat{\beta} = \frac{\sum (X_i - \overline{X})(Y_i - \overline{Y})}{\sum (X_i - \overline{X})^2} \tag{10}$$

3.2. Passive-Aggressive Regression

The passive-aggressive algorithm offers an interesting regression closed-loop solution for control tasks. This ML technique uses a simple formulation within a regression framework, which aims to obtain a linear model track for every new instance by ensuring the correct regression of the present instance. This strategy demonstrates an accommodation between the passive behavior, where the algorithm tries to improve the model using the past data memory, and the aggressive behavior, where the algorithm attempts to predict the current values correctly [33]. The mathematical expression to describe the distance function and the ε-insensitive loss reading is given as follows [34]:

$$W_{t+1} = \arg\, min\left(\frac{1}{2}\|w - w_t\|_2^2 + C\xi\right) \tag{11}$$

$$\begin{aligned} &Subject\ to:\ \ell_\varepsilon(y_t, w{\cdot}\mathrm{f}_t) \le \xi \\ &\ell_\varepsilon(y_t, w{\cdot}\mathrm{f}_t) = \begin{cases} 0, \text{ if } |y - w{\cdot}\mathrm{f}_t| \le \varepsilon \\ |y - w{\cdot}\mathrm{f}_t|\ otherwise \end{cases} \end{aligned} \tag{12}$$

where ε is a constant to control the learning rate. C is a parameter to control the aggressiveness using the slack variable, ξ. If the prediction is within the ε-margin, the algorithm either remains "passive" in each update round or "aggressively" adjusts its weights to fit the new sample.

3.3. Random Forest

The random forest technique is an integrated ML algorithm developed from the bagging algorithm. As a non-linear model ensemble regression method, the random forest strategy is structured by a set of uncorrelated classification and decision regression trees [35]. The single decision trees may suffer from high variance in making optimal predictions due to the overfitting of the data [32]. The ensemble of decision trees is used to mitigate and fit a random sample of the given observations by replacing the random sample of features. Each sample is selected from the training dataset, and the features used are extracted randomly from all features at a certain rate of the set during the training process of the tree. After training the random forest model, the prediction values can be made by averaging the predictions from all the individual regression trees using Equation (13) [36]:

$$\hat{f}_{rf}^{B}(x) = \frac{1}{B_r}\sum_{b=1}^{B_r} T_B(x) \tag{13}$$

where $\hat{f}_{rf}^{B_r}(x)$ is the prediction of the regression model at the new point, x. $T_{B_r}(x)$ represents the output of the ensemble of the tree, and B_r is the total number of the tree.

The results of random forest training show the voting output for all decision trees. Compared to the other ML models, which require tedious parameter adjustments, the

random forest technique does not need parameter adjustments and promises to achieve nearly all desired results with high adaptability to the data and the parameters used [35].

3.4. Support Vector Machine

The support vector machine model has been extensively applied in control systems and forecasting models [37,38]. Support vector machines are very effective for solving non-linear problems and deal with a limited sample of training datasets. The structure risk minimization (SRM) principle in vector machines is used to reduce the upper bound of the generalization error to enhance the confidence level in the prediction model. The main concept behind the support vector machine is to apply a kernel function that performs non-linear mapping from the input space to higher-dimensional feature space and performs linear regression in this feature space. This method uses samples from the dataset to design the prediction function, which can be described as follows [39]:

$$y = w^T \phi(x) + b \tag{14}$$

where w represents the weight vector of the features, $\phi(x)$ is the mapping value between the inputs and outputs, x represent the inputs, and b is the intercept.

The regularized risk function is described in Equation (15) to estimate the coefficients w and b [39]:

$$\arg\, min \left(\frac{1}{2}\|w\|^2 + C\frac{1}{N}\sum_{i=1}^{N}(\xi_1 - \xi_1^*) \right) \tag{15}$$

$$Subject\ to: \begin{cases} Y_i - w\phi(X_i) - b \le \varepsilon + \xi_1 \\ w\phi(X_i) + b \le \varepsilon + \xi_1^*, i = 1, 2, \ldots, N. \\ \xi_1 \ge 0 \quad \xi_1^* \ge 0 \end{cases} \tag{16}$$

where $\|w\|$ is the regularized factor. C is the penalty parameter to determine the trade-off training and flatness errors, ε represents the intensity loss function, and ξ_1 and ξ_1^* are the slake variables. By applying the kernel function, Equation (15) can be represented as:

$$\arg\, min \begin{pmatrix} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) \cdot \\ K(X_i, X_j) - \varepsilon\sum_{i=1}^{N}(\alpha_i - \alpha_i^*) \\ + \sum_{i=1}^{N} Y_i(\alpha_i - \alpha_i^*) \end{pmatrix} \tag{17}$$

$$Subject\ to: \begin{cases} \sum_{j=1}^{N}(\alpha_i - \alpha_i^*) = 0 \\ \alpha_i, \alpha_i^* \in [0, C] \end{cases} \tag{18}$$

where α_i, α_i^* are Lagrange multipliers, and i and j are different samples. The prediction function can be written as below:

$$Y = f(X) = \left(\sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \cdot K(X_i, X_j) + b \right) \tag{19}$$

4. Methodology

This research aims to identify the input features in the speed controller among other output components. Therefore, the speed controller can more accurately track the speed reference under parametric uncertainties and significant variations. Figure 2 illustrates the design of the proposed methodology for the speed controller based on ML for PMSM drive. The major parts in the proposed methodology are: (i) data acquisition and dataset

construction; (ii) features' configuration; (iii) regression model configuration; (iv) model validation; and (v) real implementation and evaluation.

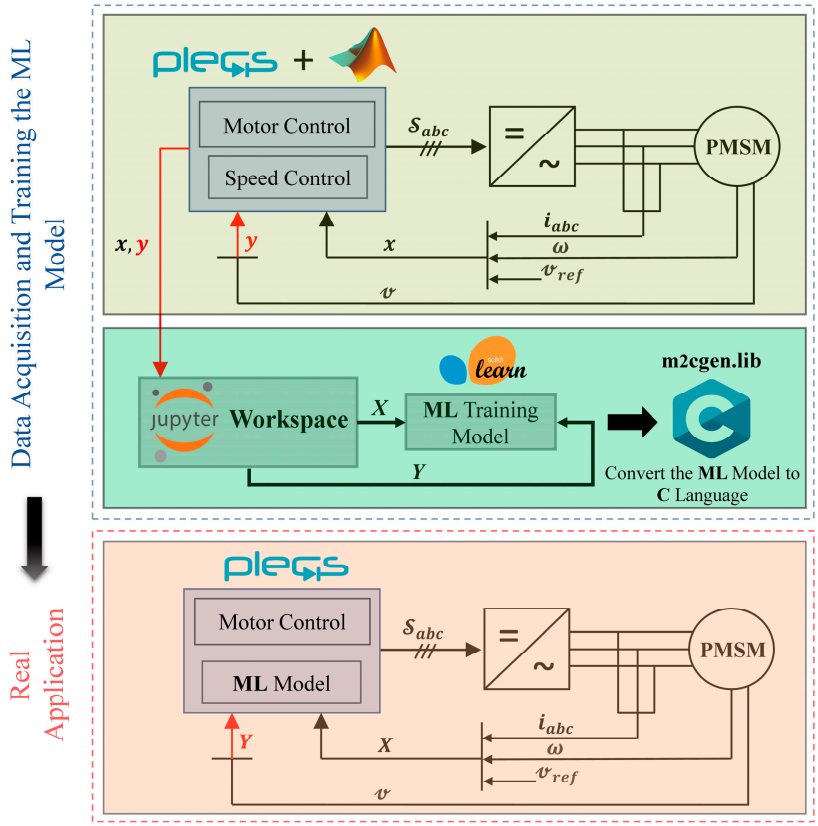

Figure 2. Simplified diagram of the controller design process from data acquisition at the test bench through model training, to the real-time implementation.

4.1. Data Acquisition and Dataset Construction

Data creation and data collection are essential components because data are the backbone of ML models. With an understanding of the problem from the first stage of the procedure, a dataset containing features relevant to the prediction of the speed tracking of PMSM is created and collected. In PMSM drive, obtaining an accurate and sufficient dataset to allow reliability in real-time for training, validating, and evaluating the ML model is challenging. Hence, any failure occurring in the system affects the collected dataset, which, in turn, affects the efficiency of the trained ML model. Therefore, a technique based on generating and training the dataset via simulations is used.

The FOSMC is integrated as a speed regulator to drive the PMSM at the rated load and different speed references to collect data for the ML model training. The output signal i of the FOSMC is identify by two different variables as described in Equation (3) (the error and the reference speed). The data are created and collected offline based on simulations using the PLECS blockset in the Matlab/Simulink environment. The data comprise the input value of the FOSMC controller, made up of the speed error and the speed reference as a response feature and the reference current d-axes as a predictor value (refer to Equations (3)–(7)). To gain a proper understanding of the dataset and its properties, initial activities on the dataset seek to create familiarity with the dataset, including identifying data quality issues and using exploratory data analysis to make first insights and discover interesting patterns.

Based on the collected data, the next stages of the proposed methodology involve a series of activities focusing on cleaning, standardizing, and normalizing the dataset to

a form that is more suitable for analysis and model building. The activities include techniques for improving the performance of the ML model, such as processing and evaluation.

4.2. Features' Configuration

The collected data are loaded into the Jupyter Notebook environment using the Pandas library, and the features of response and predictor variables in the dataset are verified. Data quality checking is carried out to ensure the dataset is in a suitable format for the analysis. The dataset is checked for missing values in each column of the features and overall to make sure that the dataset does not contain missing values in any of the columns, along with a check of the data types of each column to make sure that all the variables are numerical. For the purpose of scaling, normalization and standardization techniques are used for data preprocessing.

Normalization is the operation for rescaling the dataset from the original range so that all values are within the range [0, 1]. Normalization requires knowing or correctly predicting the minimum and maximum predictor values, which can be predicted from the collected dataset. The normalization formula is given by:

$$x_{norm}^{(i)} = \frac{x^{(i)} - x_{\min}}{x_{\max} - x_{\min}} \tag{20}$$

where $x_{norm}^{(i)}$ is the normalized value. $x^{(i)}$ is the particular sample. $x_{\max}$ and $x_{\min}$ are the largest and the smallest values in the response column, respectively.

The standardization of the dataset involves rescaling the distribution of values so the mean of the observed values is 0 and the standard deviation is 1. The standardization process can be obtained using the equation below:

$$x_{std}^{(i)} = \frac{x^{(i)} - u_x}{\sigma_x} \tag{21}$$

where $x_{std}^{(i)}$ is the new value, u_x represents the sample mean of a particular feature column, and σ_x is the standard deviation value. Correlation analysis is performed to understand the relationships between the feature variables using a 3D plot. After establishing the dataset quality checks, the ML model is built in the next step.

4.3. Regression Model Configuration

The Sklearn library ML packages in the Jupyter Notebook environment are applied for the modeling, with an optimized implementation of the described ML models and some support packages to evaluate the ML model [40]. The ML model is implemented by simply calling the library from the Sklearn package to the relevant model and uploading the dataset. The proposed ML models are trained and evaluated for efficient training and model generalizability by splitting the dataset into two sets: 80% for training and 20% for testing. The dataset is split randomly to avoid the continuity of data that drives the system into a steady-state condition. Then, each ML model is configured using its specific parameters for training and testing.

4.4. Model Validation

The built models are evaluated on a testing dataset using the R-squared (R^2) score evaluation metric, which measures the relationship strength between the dependent variable and regression models on a convenient scale between 0 and 1. After training the ML models, the evaluation test is conducted to determine the goodness-of-fit of the trained models by calling the r2_score( ) function in Sklearn packages. The R^2 score reveals the scatteredness of data values over the regression line, referred to as the coefficient of determination. The score value is always between 0 and 1. A 0 score implies that the input value has no variability around its mean described by the model, while 1 implies that the input

value has all the variability around its mean. A high R^2 score represents the superiority of the ML model. The R^2 is given by:

$$R^2 = 1 - \frac{SS_{res}}{SS_{total}} \tag{22}$$

$$SS_{res} = \sum_{i=0}^{n} \left(y_i - \widehat{y}_i\right)^2 \tag{23}$$

$$SS_{res} = \sum_{i=0}^{n} \left(y_i - \overline{y}_i\right)^2 \tag{24}$$

where SS_{res} represents the residual sum of the squares, and SS_{total} describes the total sum of the squares. The boxplot tool is used to verify the presence of outliers during the exploratory evaluation of ML models.

4.5. Real-Time Implementation and Evaluation

After building and evaluating the ML models, the code is generated, and the ML models are uploaded to the hardware device for testing. The m2cgen package is a simple Python library that converts the trained ML models into different programming languages [41]. In this work, m2cgen is employed to convert the ML model from Sklearn to the C function. The m2cgen library is adaptable and simple to use and allows for easy uploading of the generated code to the PLECS RT-BOX device. The function written in C language represents the ML model of the speed controller for the PMSM drive. It consists of two inputs that demonstrate the speed error and the speed reference and a single output that represents the predicted reference current q-axis. The mean absolute error (MAE) is utilized to evaluate the proposed speed controller, which is characterized based on the error signal, $e(t)$, as denoted below:

$$MAE = \frac{1}{T}\int_{0}^{T} |e(t)|\, dt \tag{25}$$

5. Results and Analysis

To demonstrate the effectiveness of the proposed FOSMC based on ML techniques for speed control, simulation and experiments based on the PMSM drive system are extensively investigated in this section. The parameters of the PMSM are listed in Table 1.

Table 1. Parameters of the PMSM.

Parameter with Abbreviation	Value
Stator resistance, *Rs*	1.38 Ω
d-axis stator inductance, *Ld*	2.534 mH
q-axis stator inductance, *Lq*	2.534 mH
Magnetic pole, *p*	4
Moment of inertia, *J*	7.485×10^{-6} kg m^2
Flux linkage, λ	80 Wb
Viscous friction coefficient, *Bm*	0.0002 N ms/rad

As mentioned in the previous section, the dataset must be created based on the real behavior of the PMSM drive for the acquisition of data on the responses and predictor variables. Then, the simulation-generated dataset must be modified to replicate a real application environment. In order to create the dataset, different speed reference steps were proposed using FOSMC as a speed control. The speed reference steps were set from 100 rpm to 1000 rpm, with a 100 rpm increment (100→200→300→400→500→600→700→800→900→1000) and with a time duration of 0.03 s for each step. The sampling time of 4×10^{-6} s was proposed to collect one sample of data for the speed error, speed reference, and current ref-

erence q-axes at instant t. Consequently, the complete raw dataset contained approximately 75,000 samples for each speed reference step. To make a fair comparison, the variables for FOSMC were not tuned prior to this process to observe the steady-state performance of the proposed technique. The parameters of FOSMC are described in Table 2 [24].

Table 2. Parameters of the FOSMC.

Description	Symbol	Value
Order of fractional integration	α	0.35
Order of fractional differentiation	β	0.3
Integral gain	K_i	160
Proportional gain	K_p	10
Derivative gain	K_d	3

As an example of general preprocessing, the data collected are plotted and represented by a 3D graph, as illustrated in Figure 3, with the speed reference on the X-axis, speed error on the Y-axis, and the reference current d-axes on the Z-axis. It can be seen that the reference current in q-axes value in the constrained limits is in the range of $[-10, 10]$. Moreover, a high correlation can be observed between the speed reference, speed error, and the current reference q-axes; for example, the current reference q-axes correlate with the speed reference and the speed error.

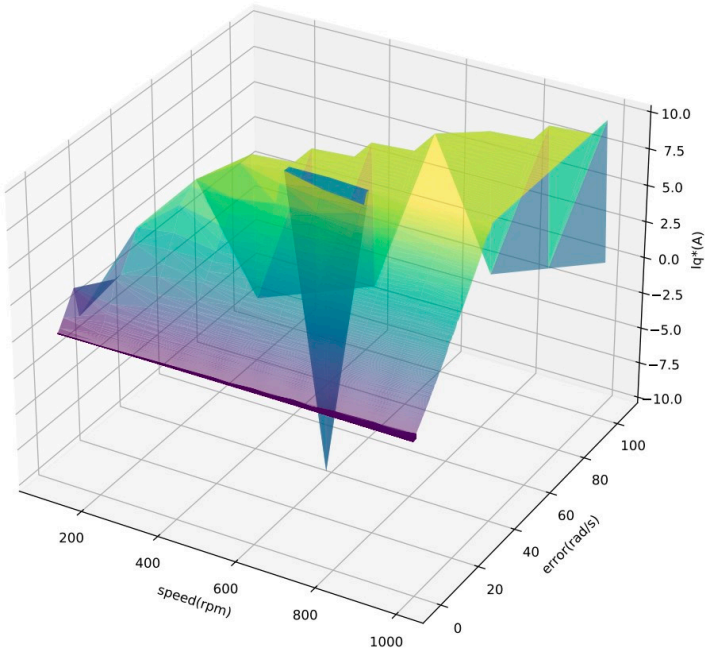

Figure 3. The reference current q-axes with different speed references and speed errors.

For the regression model configuration phase, the specific hyperparameters of the proposed ML model are described in Table 3. All hyperparameter interval bounds are manually selected to be in the range where the convergence of ML models' performance avoids overfitting.

After training the ML models, the results in terms of accuracy for each ML technique are shown in Figure 4. The accuracy is calculated by comparing the actual and the predicted values of the target variables. The total accuracy is summed over all the test samples that are randomly chosen from the dataset. In other words, the results report the sum of the testing dataset, which is correctly predicted out of the total testing data provided. Comparing the four models, the ML models based on RF and SVM show the best results, with an accuracy

of 0.996 and 0.994, respectively. Although the accuracy values of compared models are close, the models based on RF and SVM are highly appropriate for the non-linear dataset. These experimental results show the effectiveness of the proposed models to obtain the predictor value.

Table 3. Hyperparameters of the proposed ML models.

Regression Model	Hyperparameter	Parameters
Ordinary least squares (OLS)	copy_X	True
	fit_intercept	True
	n_jobs	None
	Normalize	False
Passive-aggressive regression (PR)	max_iter	100
	random_state	0
Random forest (RF)	max_depth	2
	random_state	0
	n_estimators	100
	max_leaf_nodes	100
Support vector regression (SVM)	kernel	RBF
	C	100
	gamma	0.1
	epsilon	0.1

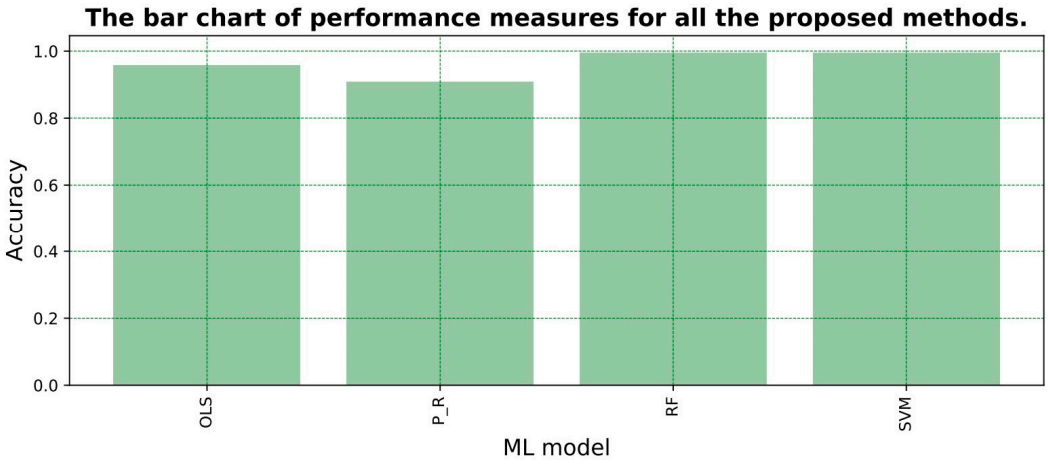

Figure 4. Bar chart of the performance measures for the proposed ML techniques.

To further prove the effectiveness of the trained models, the accuracy of the different speed steps for each ML model is shown in the form of a boxplot in Figure 5. The boxplots provide a pictorial representation of the effect of the speed reference steps on the accuracy of the trained ML models. The boxplots help us analyze each model and portray how accuracy varies over different speed reference steps. In the figure, the red line represents the median of the accuracy for each model and the two black horizontal lines connected with the black rectangle are, respectively, the non-outlying minimum and maximum values of the accuracy. It is observed that the SVM model is less accurate for some testing data. However, the SVM model accurately predicts the values in most speed references.

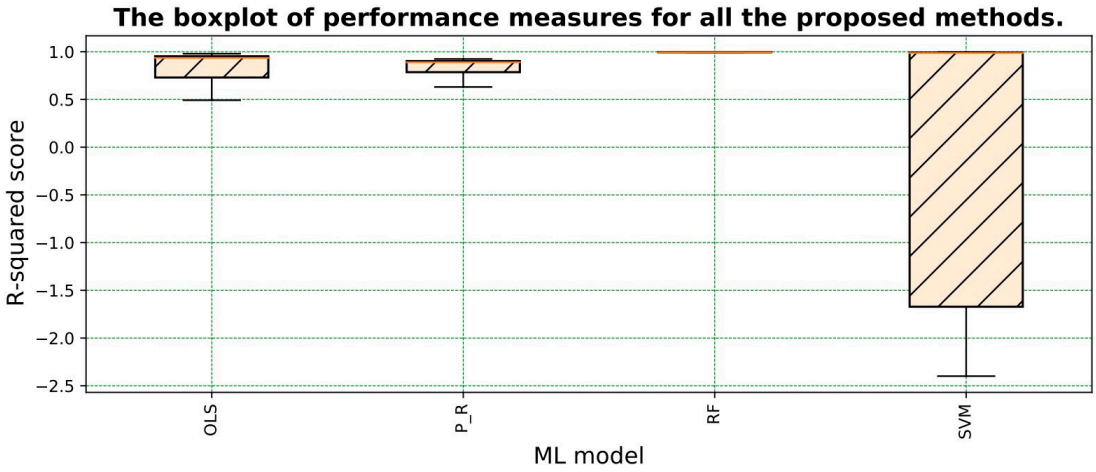

Figure 5. Boxplot of the performance measures for the proposed ML techniques.

5.1. Comparative Simulations

To further analyze the precision and effectiveness of the proposed techniques, simulation is carried out to select the efficiency of the proposed speed controller. Comparative simulations are performed among the proposed FOSMC based on ML models, and also with conventional SMC to verify the effectiveness.

The reference speed is set to 1000 rpm as a step signal and starts without a load. The speed tracking control responses of the FOSMC based on ML models and the SMC are shown in Figure 6. As can be seen from this figure that the speed of the PMSM system under the proposed FOSMC based on ML models has a better transient response profile than the conventional SMC as it performs with faster stability and without overshoot. When comparing the linear ML (OLS, PR) model with non-linear ML, as illustrated in Figure 6a, the speed controllers using OLS-FOSMC and PR-FOSMC experience less chattering and are more stable compared to RF-FOSMC and SVM-FOSMC. However, all the proposed ML models indicate that the control effects are excellent, with less error in the steady-state phase. On the other hand, the speed performance of the SMC is poor, which needs more time to regulate the reference speed and undergoes a considerable tracking error with large chattering while tracking the speed reference, as depicted in Figure 6b.

In addition, the graphical results are confirmed by calculating the error index. The obtained numerical values for the index metrics are compared in Table 4. It can be observed that the mean absolute error (MAE) index for the speed error plots under the OLS-FOSMC is reduced by 75% compared to that of the conventional SMC. Moreover, the MAE indexes for the speed errors using RF-FOSMC and SVM-FOSMC are reduced by 55%. However, the MAE index for the speed using PR-FOSMC is lower than that when the SMC is used. It can be concluded from these numerical results that the combined control of the speed and thrust force achieved under the OLS-FOSMC, RF-FOSMC, and SVM-FOSMC can deliver a faster speed response during start-up compared to the conventional SMC.

Table 4. Comparison of transient performance of the proposed controllers based on ML and the SMC using the MAE index.

MAE Index	OLS-FOSMC	PR-FOSMC	RF-FOSMC	SVM-FOSMC	SMC
	0.5967	5.426	0.7524	0.7522	2.2986

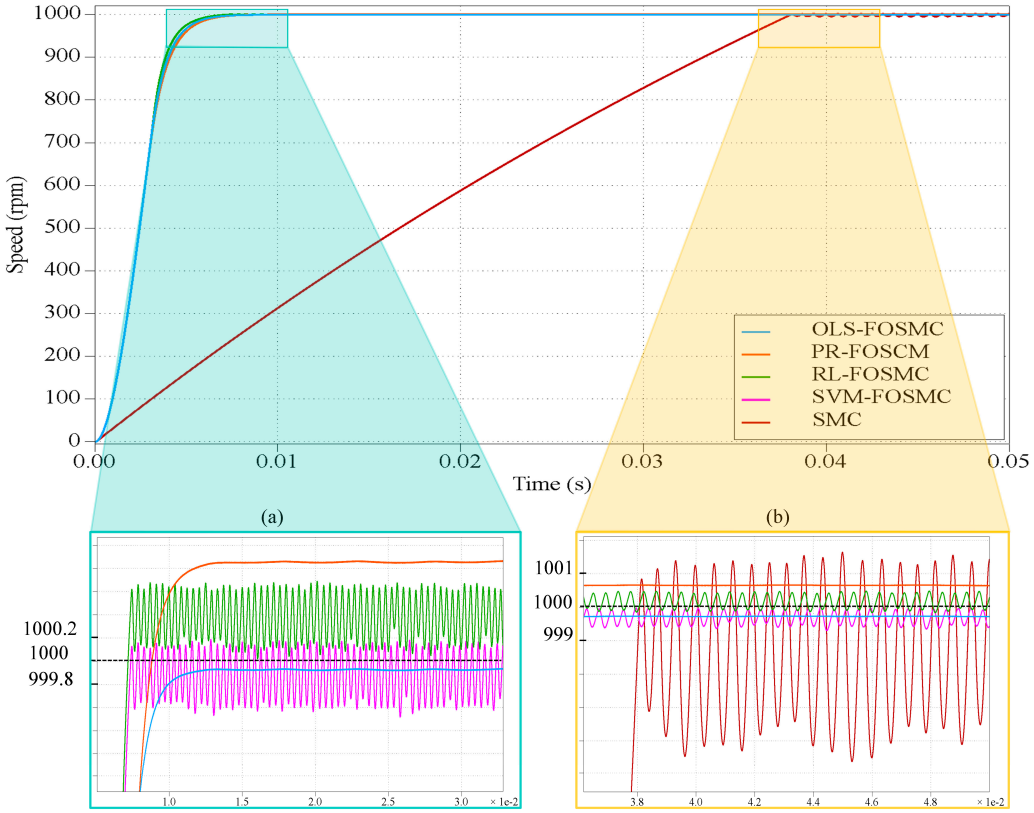

Figure 6. Speed responses of PMSM system under the proposed controllers based on ML and SMC.

5.2. Experimental Verification

In this section, experiments based on the PMSM drive system using a hardware-in-the-loop (HIL) setup are investigated in detail. As presented in Figure 7, the setup used for HIL verification purposes comprises an RT-Box 1 provided by Plexim, where the sampled time to model the controller is 6 µs and the switching frequency (fs) is 25 kHz. The connection of the RT-Box is described in [42,43]. The controller and PMSM drive parameters are set such that the parameters are equal to the parameters that were implemented in the simulation.

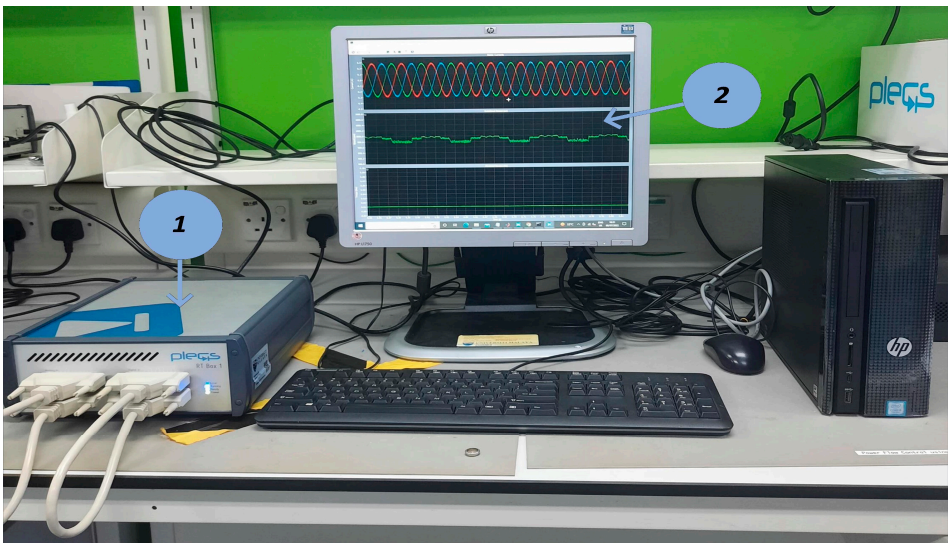

Figure 7. Hardware in-the-loop experimental setup. (1) PLECS RT-box, (2) desktop.

In order to show the superiority and robustness of the introduced speed controller based on ML, the following three scenarios are proposed:

5.2.1. Comparative Dynamic Response under Speed Reference

In this experiment, the system performance is tested under no load and the reference speed is set to be a step signal at 1000 rpm, as shown in Figure 8. The speed curves of the system under OLS-FOSMC, PR-FOSMC, RF-FOSMC, SVM-FOSMC, and SMC are exhibited in Figure 8. It can be noted that the speed fluctuation in a steady state with conventional SMC is much higher than that with the proposed controller based on ML. It is clearly illustrated in Figure 8a,b that the dynamic speed responses under the OLS-FOSMC and PR-FOSMC techniques have low fluctuation in the speed, with over speeds of +0.03% and +0.11%, respectively. The dynamic speed responses of RF-FOSMC and SVM-FOSMC are also presented in Figure 8c,d, where it can be seen that RF-FOSMC and SVM-FOSMC have small chattering, with over speeds of +0.09% and +0.04% and drop speeds of 0.06% and 0.05%, respectively. The conventional SMC has the highest speed fluctuation and an over speed and drop speed of +0.75% and 0.8%, respectively, as shown in Figure 8e. The comprehensive results demonstrate that the proposed FOSMC based on ML techniques has a better disturbance rejection capability, smaller speed drop, and better convergence than the conventional SMC.

5.2.2. Comparative Dynamic Response under a One-Speed Step Change

Figure 9 demonstrates the comprehensive comparative results of the four control methods using FOSMC based on ML and conventional SMC under a one-speed step change as a waveform of 2 min with a 50% duty cycle, considering the low- and high-speed responses are 1000 and 1200 rpm, respectively.

First, Figure 9a shows a much faster convergence of the speed tracking and dynamic performance when using OLS-FOSMC (i.e., the shorter settling time of 5 ms, overshoot of +0.03%) due to the behavior of the proposed ML model, which drives the non-linear system as linear during the steady-state phase and rejects any repeatable disturbances. Meanwhile, Figure 9b displays a convergence of the speed response when using PR-FOSMC (5.7 ms) and a higher speed tracking error compared to that shown in Figure 9a. However, Figure 9c,d illustrate the overshoot of +0.09% and +0.04% of the speed error during the transition when using RF-FOSMC and SVM-FOSMC at the instants 5.4 ms and 5.7 ms, respectively. Next, Figure 9e exhibits a low convergence to the speed reference (11 ms) and remarkable distortion during tracking of the speed error for the conventional SMC.

5.2.3. Comparative Dynamic Response under a Multi-Speed Step Change

This subsection verifies the comparative dynamic responses of the four control methods using FOSMC based on ML and conventional SMC under a multi-speed step change via comparative experimental results. In this experiment, the step response is set to 1000 rpm, 1200 rpm, and 1400 rpm, and the estimated values of the rotor speed are exhibited in Figure 10. The multi-speed step is changing as a waveform of 1.8 min with a 33% duty cycle in each step and a different initialization time considering the low- and high-speed responses are 1000 rpm, 1200 rpm, and 1400 rpm. The speed tracking performance of PMSM in this scenario is shown in Figure 10. Figure 10a illustrates an excellent convergence of the speed tracking and dynamic performance using OLS-FOSMC. Moreover, the overshoot and rise time are small in the speed transition, where the shorter settling time in each transition is 5 ms or 5.1 ms, and the overshoot is +0.03%. Figure 10b represents the convergence of the speed response during the transition when using PR-FOSMC (the shorter settling times of 5.7 ms and 5.6 ms in each transition). Figure 10c,d illustrate the shorter settling times of 5.5 ms and 5.7 ms during the second transition when using RF-FOSMC and SVM-FOSMC, respectively. Meanwhile, Figure 10e exhibits a low convergence to the speed reference in the transition (11 ms and 12 ms) and remarkable distortion when tracking the speed error for the conventional SMC.

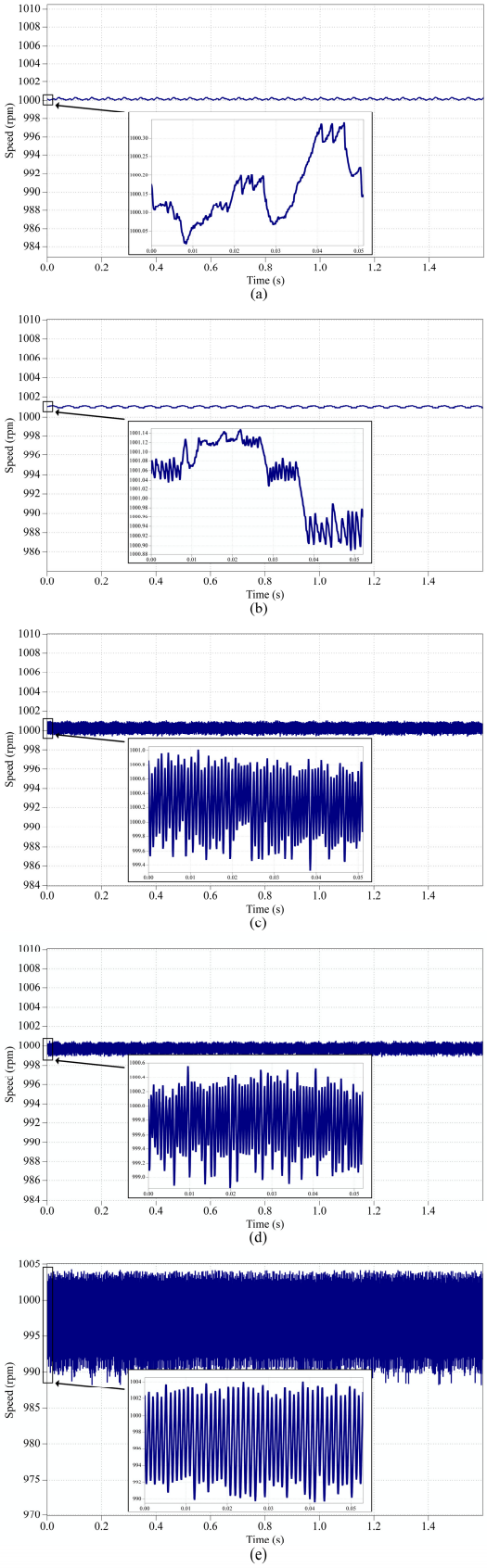

Figure 8. Experimental dynamic speed response of the PMSM under the no-load condition and the step reference speed of 1000 rpm using HIL. (**a**) OLS-FOSMC, (**b**) PR-FOSMC, (**c**) RF-FOSMC, (**d**) SVM-FOSMC, (**e**) SMC.

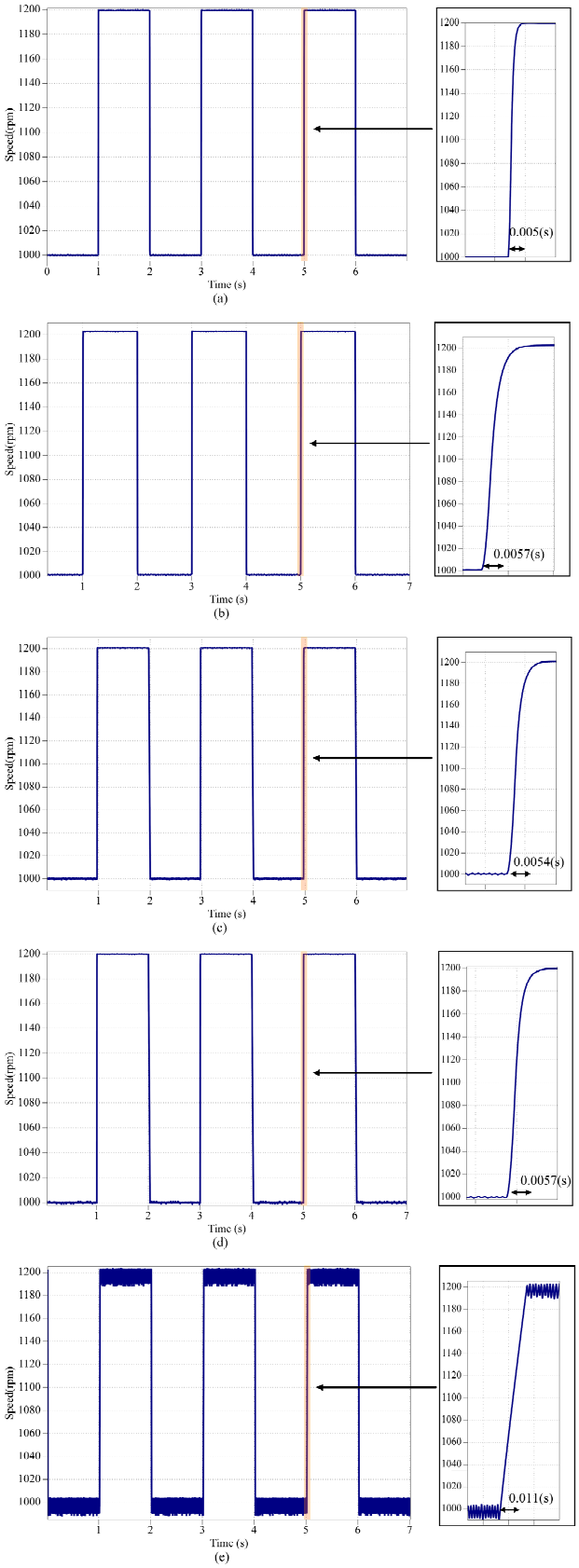

Figure 9. Experimental dynamic speed response of the PMSM under the no-load condition and two-step reference speed using HIL. (**a**) OLS-FOSMC, (**b**) PR-FOSMC, (**c**) RF-FOSMC, (**d**) SVM-FOSMC, (**e**) SMC.

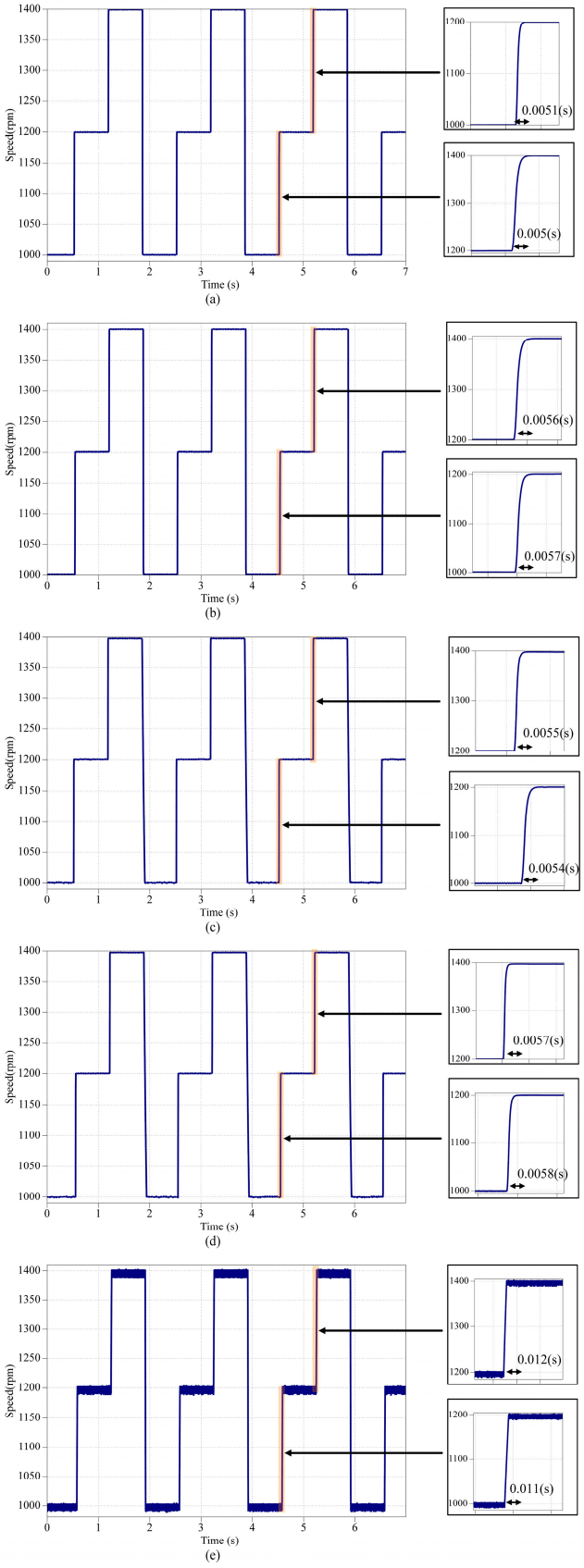

Figure 10. Experimental dynamic speed response of the PMSM under the no-load condition and different-step reference speed using HIL. (**a**) OLS-FOSMC, (**b**) PR-FOSMC, (**c**) RF-FOSMC, (**d**) SVM-FOSMC, (**e**) SMC.

6. Conclusions

A speed controller based on machine learning (ML) was successfully designed and implemented for PMSM drive. The proposed ML controller was designed by learning the behavior of the fractional order sliding mode control (FOSMC). Different ML algorithms were presented and applied as the ML model's basic architecture, which were proven to overcome self-tuning and the speed transition problem to drive the PMSM. The dataset for training, validation, and testing was obtained through a simulated control process. The trained ML models were based on an FOSMC speed controller. The proposed speed controllers were tested in a full environment-control process simulation. The superiority of the proposed controller was confirmed through simulations and experiments using HIL, and the results showed that the designed method could perform satisfactorily, with a fast transient response and good disturbance rejection. Using the proposed controller, the PMSM drive can accurately track the reference speed with the industrial requirement of 0.1% error. The conventional SMC method has inferior characteristics and fluctuation in the steady-state speed of the PMSM and a lower convergence rate. The ML controller was investigated with a speed regulator to avoid self-tuning of parameters (learning the tuning parameters during the training) and reduce the effect of parametric uncertainties on the speed variation. However, it is still uncertain how well this type of controller may be generally applied across the different environments of the PMSM or even in different implementations that require an enormously large dataset exhibiting this diversity.

Author Contributions: Conceptualization, Y.Z. and F.M.Z.; Methodology, Y.Z. and M.K.; Software, Y.Z.; Validation, Y.Z. and F.M.Z.; Formal analysis, M.K.; Investigation, E.M.Z.; Writing—review & editing, Y.Z.; Visualization, Y.Z. and J.R.T.; Supervision, F.M.Z., S.M. and M.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Ministry of Higher Education, Malaysia (MOHE) through Fundamental Research Grant Scheme (FRGS/1/2021/TK0/UPSI/02/1).

Data Availability Statement: Not Applicable.

Conflicts of Interest: Not Applicable.

References

1. Zaihidee, F.M.; Mekhilef, S.; Mubin, M. Robust Speed Control of PMSM Using Sliding Mode Control (SMC)—A Review. *Energies* **2019**, *12*, 1669. [CrossRef]
2. Yang, J.; Chen, W.-H.; Li, S.; Guo, L.; Yan, Y. Disturbance/Uncertainty Estimation and Attenuation Techniques in PMSM Drives—A Survey. *IEEE Trans. Ind. Electron.* **2016**, *64*, 3273–3285. [CrossRef]
3. Du, H.; Wen, G.; Cheng, Y.; Lu, J. Design and Implementation of Bounded Finite-Time Control Algorithm for Speed Regulation of Permanent Magnet Synchronous Motor. *IEEE Trans. Ind. Electron.* **2020**, *68*, 2417–2426. [CrossRef]
4. Ma, Y.; Li, D.; Li, Y.S.; Yang, L.C.A.F.T. A Novel Discrete Compound Integral Terminal Sliding Mode Control with Disturbance Compensation for PMSM Speed System. *IEEE/ASME Trans. Mechatron.* **2021**, *27*, 549–560. [CrossRef]
5. Hou, Q.; Ding, S.; Yu, X. Composite Super-Twisting Sliding Mode Control Design for PMSM Speed Regulation Problem Based on a Novel Disturbance Observer. *IEEE Trans. Energy Convers.* **2020**, *36*, 2591–2599. [CrossRef]
6. Chai, S.; Wang, L.; Rogers, E. A Cascade MPC Control Structure for a PMSM with Speed Ripple Minimization. *IEEE Trans. Ind. Electron.* **2012**, *60*, 2978–2987. [CrossRef]
7. Mynar, Z.; Vesely, L.; Vaclavek, P. PMSM Model Predictive Control with Field-Weakening Implementation. *IEEE Trans. Ind. Electron.* **2016**, *63*, 5156–5166. [CrossRef]
8. Yang, J.-Z.; Li, Y.-X.; Tong, S. Adaptive NN finite-time tracking control for PMSM with full state constraints. *Neurocomputing* **2021**, *443*, 213–221. [CrossRef]
9. Sun, X.; Yu, H.; Yu, J.; Liu, X. Design and implementation of a novel adaptive backstepping control scheme for a PMSM with unknown load torque. *IET Electr. Power Appl.* **2019**, *13*, 445–455. [CrossRef]
10. Linares-Flores, J.; Garcia-Rodriguez, C.; Sira-Ramirez, H.; Ramirez-Cardenas, O.D. Robust Backstepping Tracking Controller for Low-Speed PMSM Positioning System: Design, Analysis, and Implementation. *IEEE Trans. Ind. Inform.* **2015**, *11*, 1130–1141. [CrossRef]

11. Zurita-Bustamante, E.W.; Sira-Ramirez, H.; Linares-Flores, J.; Ramirez-Cardenas, O.D.; Contreras-Ordaz, M.A.; Guerrero-Castellanos, J.F. On the active disturbance rejection control of the permanent magnet synchronous motor. In Proceedings of the 2018 IEEE Power Energy Conference Illinois, PECI 2018, Champaign, IL, USA, 22–23 February 2018; Volume 2018, pp. 1–6. [CrossRef]
12. Zuo, Y.; Mei, J.; Jiang, C.; Yuan, X.; Xie, S.; Lee, C.H.T. Linear Active Disturbance Rejection Controllers for PMSM Speed Regulation System Considering the Speed Filter. *IEEE Trans. Power Electron.* **2021**, *36*, 14579–14592. [CrossRef]
13. Barkat, S.; Tlemçani, A.; Nouri, H. Noninteracting Adaptive Control of PMSM Using Interval Type-2 Fuzzy Logic Systems. *IEEE Trans. Fuzzy Syst.* **2011**, *19*, 925–936. [CrossRef]
14. Jie, H.; Zheng, G.; Zou, J.; Xin, X.; Guo, L. Speed Regulation Based on Adaptive Control and RBFNN for PMSM Considering Parametric Uncertainty and Load Fluctuation. *IEEE Access* **2020**, *8*, 190147–190159. [CrossRef]
15. Wang, H.; Li, S.; Lan, Q.; Zhao, Z.; Zhou, X. Continuous terminal sliding mode control with extended state observer for PMSM speed regulation system. *Trans. Inst. Meas. Control* **2016**, *39*, 1195–1204. [CrossRef]
16. Wai, R.-J. Total sliding-mode controller for PM synchronous servo motor drive using recurrent fuzzy neural network. *IEEE Trans. Ind. Electron.* **2001**, *48*, 926–944. [CrossRef]
17. Han, Y.-S.; Choi, J.-S.; Kim, Y.-S. Sensorless PMSM drive with a sliding mode control based adaptive speed and stator resistance estimator. *IEEE Trans. Magn.* **2000**, *36*, 3588–3591. [CrossRef]
18. Liu, J.; Li, H.; Deng, Y. Torque Ripple Minimization of PMSM Based on Robust ILC via Adaptive Sliding Mode Control. *IEEE Trans. Power Electron.* **2017**, *33*, 3655–3671. [CrossRef]
19. Li, S.; Zhou, M.; Yu, X. Design and Implementation of Terminal Sliding Mode Control Method for PMSM Speed Regulation System. *IEEE Trans. Ind. Inform.* **2011**, *9*, 1879–1891. [CrossRef]
20. Feng, Y.; Zheng, J.; Yu, X.; Truong, N.V. Hybrid Terminal Sliding-Mode Observer Design Method for a Permanent-Magnet Synchronous Motor Control System. *IEEE Trans. Ind. Electron.* **2009**, *56*, 3424–3431. [CrossRef]
21. Hou, Q.; Ding, S. GPIO Based Super-Twisting Sliding Mode Control for PMSM. *IEEE Trans. Circuits Syst. II Express Briefs* **2020**, *68*, 747–751. [CrossRef]
22. Xu, B.; Zhang, L.; Ji, W. Improved Non-Singular Fast Terminal Sliding Mode Control with Disturbance Observer for PMSM Drives. *IEEE Trans. Transp. Electrif.* **2021**, *7*, 2753–2762. [CrossRef]
23. Xu, W.; Junejo, A.K.; Liu, Y.; Islam, R. Improved Continuous Fast Terminal Sliding Mode Control with Extended State Observer for Speed Regulation of PMSM Drive System. *IEEE Trans. Veh. Technol.* **2019**, *68*, 10465–10476. [CrossRef]
24. Zaihidee, F.M.; Mekhilef, S.; Mubin, M. Application of Fractional Order Sliding Mode Control for Speed Control of Permanent Magnet Synchronous Motor. *IEEE Access* **2019**, *7*, 101765–101774. [CrossRef]
25. Chaoui, H.; Khayamy, M.; Aljarboua, A.A. Adaptive Interval Type-2 Fuzzy Logic Control for PMSM Drives with a Modified Reference Frame. *IEEE Trans. Ind. Electron.* **2017**, *64*, 3786–3797. [CrossRef]
26. Pajchrowski, T.; Zawirski, K.; Member, S.; Nowopolski, K. Neural Speed Controller Trained Online by Means of Modified RPROP Algorithm. *IEEE Trans. Ind. Inform.* **2015**, *11*, 560–568. [CrossRef]
27. 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.* **2012**, *9*, 1986–2003. [CrossRef]
28. Jin, H.; Zhao, X. Complementary Sliding Mode Control via Elman Neural Network for Permanent Magnet Linear Servo System. *IEEE Access* **2019**, *7*, 82183–82193. [CrossRef]
29. Zhang, B.; Pi, Y.; Luo, Y. Fractional order sliding-mode control based on parameters auto-tuning for velocity control of permanent magnet synchronous motor. *ISA Trans.* **2012**, *51*, 649–656. [CrossRef] [PubMed]
30. Xu, W.; Jiang, Y.; Mu, C. Novel Composite Sliding Mode Control for PMSM Drive System Based on Disturbance Observer. *IEEE Trans. Appl. Supercond.* **2016**, *26*, 1–5. [CrossRef]
31. Fei, J.; Wang, Z.; Pan, Q. Self-Constructing Fuzzy Neural Fractional-Order Sliding Mode Control of Active Power Filter. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, 1–12. [CrossRef]
32. Kirchgassner, W.; Wallscheid, O.; Bocker, J. Data-Driven Permanent Magnet Temperature Estimation in Synchronous Motors with Supervised Machine Learning: A Benchmark. *IEEE Trans. Energy Convers.* **2021**, *36*, 2059–2067. [CrossRef]
33. Jorge, J.; Paredes, R. Passive-Aggressive online learning with nonlinear embeddings. *Pattern Recognit.* **2018**, *79*, 162–171. [CrossRef]
34. Von Krannichfeldt, L.; Wang, Y.; Hug, G. Online Ensemble Learning for Load Forecasting. *IEEE Trans. Power Syst.* **2020**, *36*, 545–548. [CrossRef]
35. Wu, H.; Li, W. Downscaling Land Surface Temperatures Using a Random Forest Regression Model with Multitype Predictor Variables. *IEEE Access* **2019**, *7*, 21904–21916. [CrossRef]
36. Javeed, A.; Zhou, S.; Yongjian, L.; Qasim, I.; Noor, A.; Nour, R. An Intelligent Learning System Based on Random Search Algorithm and Optimized Random Forest Model for Improved Heart Disease Detection. *IEEE Access* **2019**, *7*, 180235–180243. [CrossRef]
37. Pourjafari, E.; Reformat, M. A Support Vector Regression Based Model Predictive Control for Volt-Var Optimization of Distribution Systems. *IEEE Access* **2019**, *7*, 93352–93363. [CrossRef]
38. Khakifirooz, M.; Chien, C.F.; Chen, Y.J. Dynamic Support Vector Regression Control System for Overlay Error Compensation with Stochastic Metrology Delay. *IEEE Trans. Autom. Sci. Eng.* **2020**, *17*, 502–512. [CrossRef]

39. Ahmad, M.W.; Mourshed, M.; Rezgui, Y. Tree-based ensemble methods for predicting PV power generation and their comparison with support vector regression. *Energy* **2018**, *164*, 465–474. [CrossRef]
40. Barupal, D.; Fiehn, O. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *127*, 2825–2830. [CrossRef]
41. Available online: https://github.com/BayesWitnesses/m2cgen (accessed on 5 October 2022).
42. Available online: https://www.plexim.com/sites/default/files/demo_models_categorized/rtbox/sensorless_vector_control_pmsm.pdf (accessed on 5 October 2022).
43. Available online: https://www.plexim.com/sites/default/files/demo_models_categorized/rtbox/sensorless_vector_control_pmsm.zip (accessed on 5 October 2022).