Control Enhancement of Traction Electric Drives Using Neural Network Predictive Controller

Romain Cocogne, Sébastien Bilavarn, Mostafa El Mokadem, Khaled Douzane

▶ **To cite this version:**

Romain Cocogne, Sébastien Bilavarn, Mostafa El Mokadem, Khaled Douzane. Control Enhancement of Traction Electric Drives Using Neural Network Predictive Controller. ICCAD, May 2024, Paris, France. hal-04699443

HAL Id: hal-04699443
https://hal.science/hal-04699443v1

Submitted on 16 Sep 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Control Enhancement of Traction Electric Drives Using Neural Network Predictive Controller

ROMAIN COCOGNE[1], SÉBASTIEN BILAVARN[2], MOSTAFA EL MOKADEM[1], AND KHALED DOUZANE[1]

[1] *Silicon Mobility, Intel Corp, Sophia-Antipolis, France*
[2] *LEAT, Université Côte d'Azur, Sophia-Antipolis, France*

This paper investigates the use of a hybrid Recurrent Neural Network to reproduce the behavior of a nonlinear long-horizon Model Predictive Controller (MPC) used in traction motor drive systems. The goal is to assess the operational validity and control performances of such neural network based predictive controller (further referred to as *Full Neural Network MPC*), and to compare against Field Oriented Control (FOC), the current industry standard. Based on simulation results using a Model in the Loop (MiL) environment, it is shown that the proposed FNN-MPC can properly learn the behavior and characteristics of the nonlinear long-horizon MPC while being more computationally implementable, with still better control performance and quality than FOC.

1. INTRODUCTION AND BACKGROUND

Efficient control strategies of Interior Permanent Magnet Synchronous Motor (IPMSM) are currently an important challenge for high-quality real-time management of electric motors in many electric vehicle traction applications. Popular predictive control systems usually impose a huge computational burden and the need for expensive hardware, especially for long prediction horizons [1]. To overcome this, Artificial Neural Network (ANN)-based modeling methods started to be investigated. Despite a variety of papers addressing neural network based approximation of nonlinear MPC for a range of application domains (industrial processes, financial and economic models, robotics, etc.), there are currently few works investigating an application study to IPMSM as it is a recent field of research with the current democratization of Electric Vehicles (EV) and Hybrid Electric Vehicles (HEV).

A general state of the art considering how robust control of electric machines can be improved with neural networks is provided in [2], encompassing actual applications to control design, state estimation, signal processing, Pulse Width Modulation (PWM) synthesis, parameter and model identification. Besides replacing the control algorithm, some works have naturally started to investigate other benefits of neural networks such as sensorless position control [3] [4] or internal temperature estimation [5].

Among the works dealing more specifically with the control problem, three approaches can be considered for neural network based predictive controllers [1]. A first and widespread solution is to replace the MPC model with a neural network trained on the plant system. This is the easiest method because the network can be trained using data from a simple system identification experiment (for example using the *Matlab System Identification Toolbox™* [6]). The main advantage of this method is that for such complex nonlinear systems, an identification by the data can give sufficiently robust results compared to a linear approximation of the system [7]. The main drawback is the limited gains in processing power for long-horizon predictions, as the optimization algorithm is kept the same and consumes a lot of computing resources.

An illustration of this in an application to the MPC control of an IPMSM is described in [8]. A small Multi-Layer Perceptron (MLP), based on one hidden layer of seven neurons is designed and trained, with successful simulations showing improved control of a typical motor example. However, some simplifying assumptions about motor specifications (reducing the model to a Single Input Single Output system) are made to facilitate and allow the actual training procedure, which may restrict practical applications [8].

A second approach is to replace the optimization algorithm while keeping the formal prediction model. This type of substitute is more complex to implement because defining a dataset relevant to the MPC optimisation algorithm for proper representation and training is not a trivial task. No references could be found on this approach in the literature to the best of our knowledge.

The last approach is to replace the full MPC controller with a neural network (FNN-MPC). This technique is not as popular as the first method despite greater potential to reduce complexity by removing the need to solve complex optimization, state estimation and prediction problems in real-time [9]. This approach is less used because replacing the whole controller is more difficult, especially for nonlinear systems, and the resulting neural network may also lead to limited control performances and robustness.

[10] describes a FNN-MPC approach used to replace the full long-horizon finite control set model predictive control (FCS-MPC) in electrical drive systems. The problem here is simplified considering the controller output consists of a finite set of switching states, that can therefore be treated as a classification prob-

lem. A three layers MLP (9-25-8 topology) is used to learn the direct model predictive control problem of the power converter with a prediction horizon of 5. A rich dataset of reference signals which consists of chirp, sawtooth, and step signals are applied to the electrical machine controlled by a FCS-MPC controller for training the neural network. Its ability to operate in a real-time environment is demonstrated in a simulation using an IPMSM plant.

In more recent works, some authors have considered full MPC approximation employing larger Deep Neural Networks (DNNs), for real-time implementation of IPMSM torque tracking [11]. In this approach, the DNN is intended to learn the MPC functionality based on training data generated offline and with the controller stimulated in open-loop mode. The DNN topology is based on a MLP (5 layers of 20 neurons each) and trained with datatsets of 500,000 and 1,000,000 samples (with a random sampling strategy). Results show the ability of the DNN-approximate MPC to have a very good torque tracking capability on two continuous-time control set MPC (current tracking delta MPC, Del-MPC, torque tracking economic MPC, EMPC), and to potentially support long prediction horizons (up to 20 in their experiments).

IPMSM speed control based on FNN-MPC is also addressed in [12]. Aiming at the highly nonlinear characteristics of the MPC IPMSM control system, an Echo State Network (ESN, a type of recurrent neural network) is considered to replace a part of the global dual control loop structure of the full MPC (composed of current loop PI control and the speed loop controller). The ESN is trained and used in place of the original speed loop controller to predict the future speed. Simulation results show that the ESN prediction model is improved, with reduced overshoot and rise time compared to their reference controller, and have good dynamic control performance.

Replacing a long-horizon nonlinear MPC with a neural network to control an IPMSM system can bring distinct advantages, mainly in reducing the computational effort enough to get close to real-time processing [10]. However actual works are still in early stage with still many issues on the tangible contributions of neural network based MPC especially for IPMSM. Performance comparison with the FOC, the current standard for high performance control of IPMSM in the industry, is also lacking in the literature.

This paper addresses this, introducing a competitive long-horizon nonlinear MPC that can in theory outperforms a FOC, and investigates how a neural network trained on this MPC can be a more effective solution for real-time implementations

2. FNN-MPC DEFINITION AND DESIGN

A. IPMSM control

In electric drive technologies, the standard control approach is based on different levels of nested loops for torque (or speed) and current control [13]. The torque/speed request of the first loop is fed into lookup tables to compute the reference currents in the rotating dq-frame and supply values for the current loop [14]. The dq currents are calculated using a Clark / Park transformation on the measured abc phase currents. Since one goal of this work is to study the theoretical validity of the FNN-MPC as an IPMSM current controller, the modulation can be removed to simplify the problem and keep the full system in the dq reference frame. Differential equations of i_d and i_q can then be

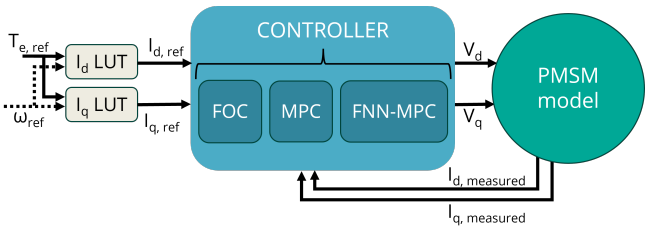

Fig. 1. IPMSM control system environment used for simulations

used to model the plant and serve as the prediction model in the nonlinear MPC.

We therefore address the block diagram of Fig. 1 where the control algorithm can be a FOC (used as the reference controller), a MPC or a FNN-MPC controller.

A.1. IPMSM model

First, the set of differential equations of the IPMSM, which can later be used as plant and prediction models for the MPC control problem, has to be specified. The continuous-time model of the IPMSM in the synchronously rotating dq-frame can be expressed as follows [13], using the nomenclature of Table 1 :

$$\frac{di_d(t)}{dt} = -\frac{R_s}{L_d} i_d(t) + \omega_e(t)\frac{L_q}{L_d} i_q(t) + \frac{v_d}{L_d} \tag{1a}$$

$$\frac{di_q(t)}{dt} = -\frac{R_s}{L_q} i_q(t) - \omega_e(t)\frac{L_d}{L_q} i_d(t) - \omega_e(t)\frac{\psi}{L_q} + \frac{v_q}{L_q} \tag{1b}$$

The previous equations can be further discretized in time using the Euler method [13].

This results in a discrete-time model of the IPMSM, corresponding to the set of nonlinear equations :

$$x(k+1) = A_m x(k) + B_m u(k) + d_m \tag{2a}$$

$$y(k+1) = C_m x(k) \tag{2b}$$

Where

$$x(k) = \begin{bmatrix} i_d(k) \\ i_q(k) \end{bmatrix} \qquad A_m = \begin{bmatrix} 1 - T_s\frac{R_s}{L_d} & T_s\frac{\omega_e L_q}{L_d} \\ -T_s\frac{\omega_e L_d}{L_q} & 1 - T_s\frac{R_s}{L_q} \end{bmatrix}$$

$$u(k) = \begin{bmatrix} v_d(k) \\ v_q(k) \end{bmatrix} \qquad B_m = \begin{bmatrix} \frac{T_s}{L_d} & 0 \\ 0 & \frac{T_s}{L_q} \end{bmatrix}$$

$$d_m = \begin{bmatrix} 0 \\ -T_s\frac{\omega_e \psi}{L_q} \end{bmatrix} \qquad C_m = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

The values described in Table 2 are used for all the simulations and represent the characteristics extracted from a real electric motor used in an automotive traction system. Addressing the torque control problem in the following, we consider ω_e to be constant and assume $\omega_e = \omega_0$.

A.2. Model Predictive Control

The considered MPC scheme for the defined IPMSM model is depicted in Fig. 2. This controller will be further employed to generate training datasets for the neural network, and also to benchmark the different control techniques (FOC, MPC, FNN-MPC) in the validation study. Equations Eq. (2) are used in the nonlinear predictive model, where interior-point path-following

Table 1. Nomenclature of motor parameters

Name	Nomenclature	Unit
D-axis current	i_d	A
Q-axis current	i_q	A
D-axis voltage	v_d	V
Q-axis voltage	v_q	V
Electrical speed	ω_e	rad/s

Table 2. Constant motor parameters used for simulations

Name	Nomenclature	Value
DC-link Voltage	v_{dc}	$705V$
Number of pole pairs	N_{pp}	4
Maximum power	P_{max}	$93kW$
Stator resistance	R_s	$12m\Omega$
Flux linkage	ψ	$66mWb$
D-axis inductance	L_d	$153\mu H$
Q-axis inductance	L_q	$556\mu H$
Constant speed	ω_0	$8000rpm$
Sampling frequency	f_s	$100kHz$
Sampling time	T_s	$10\mu s$

Table 3. Constraints on motor parameters

Name	Nomenclature	Value
Maximum phase voltage	$v_{s_{max}}$	$407V$
Maximum phase current	$i_{s_{max}}$	$720A$
Maximum current slope	$i_{s_{max,slope}}$	$50000A/s$

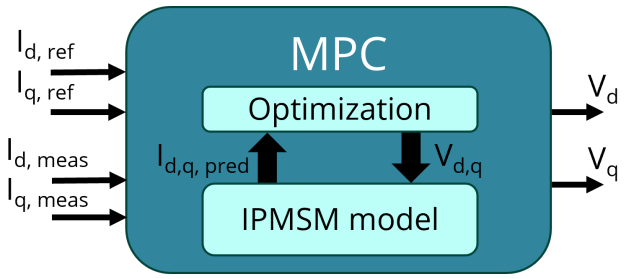

Fig. 2. General MPC scheme applied to IPMSM control

algorithms are used to solve the convex optimisation problem [15].

In the cost function Eq. (3), the error between the reference current and the predicted current over the horizon is penalized, with voltage variations as well.

$$J = \sum_{k=0}^{N_p-1} ||y^*(k+1) - y(k+1)||_2^2 + \lambda_u ||\Delta u(k)||_2^2 \quad \textbf{(3a)}$$

$$||y^* - y||_2^2 = (y^* - y)^T I_2 (y^* - y) \quad \textbf{(3b)}$$

$$\Delta u(k) = u(k) - u(k-1) \quad \textbf{(3c)}$$

Where $\lambda_u = 0.01$, $N_p = 10$ being the prediction horizon and I_2 the identity matrix.

The control horizon is set to be the same as the prediction horizon. The long horizons are chosen for the enhanced control quality, with the consequence of increasing the computation time of the optimization algorithm. Finally, the following set of power and dynamic constraints Eq. (4), and the motor constraints of Table 3, must be satisfied at each step of the prediction.

$$\left.\begin{aligned} v_d^2 + v_q^2 &\leq v_{s_{max}}^2 \\ i_d^2 + i_q^2 &\leq i_{s_{max}}^2 \end{aligned}\right\} \quad \text{Saturation constraints} \quad \textbf{(4a)}$$

$$\Delta i_{d,q} < i_{s_{max,slope}} \quad \text{Dynamic constraints} \quad \textbf{(4b)}$$

B. Neural network substitution

The goal is to define and train a neural network to reproduce the behavior of the MPC defined previously. The chosen architecture of the FNN-MPC is inspired by the approach depicted in [9].

To identify which type of neural network model is capable to better emulate the MPC control policy of the IPMSM, we compare the RMSE (Root Mean Square Error) of the output vector $y = [v_d, v_q]$ for five types of neural networks: MLP (Multi-Layer Perceptron), CNN [16] (Convolution Neural Network), LSTM (Long Short-Term Memory), a mix between MLP and LSTM (LSTM-NN, as described in Fig. 3), and a mix between LSTM and CNN (LSTM-CNN). The RMSE was calculated after getting the best configuration possible for each topology, using the *Experiment Manager*™ of *Matlab*™ [6].

Results show that LSTM-NN has the smallest error with the original MPC, significantly outperforming other networks in terms of RMSE by a factor of 2 (Table 4). This result is consistent with the in-depth study done in [9]. Therefore this combination of LSTM with NN will be considered in the rest of the study,

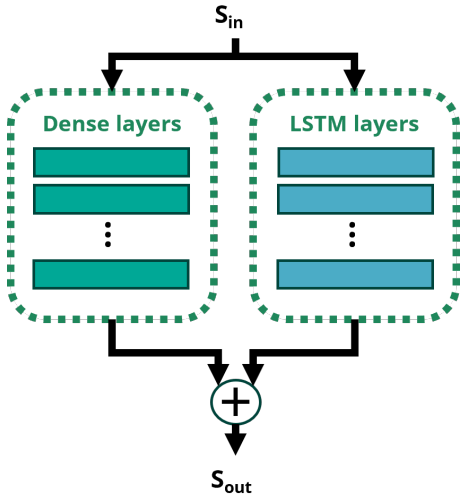

Fig. 3. Architecture of LSTM-NN

with a topology composed of two layers of 300 dense neurons per layer and one layer of 500 LSTM neurons, further completed with dropout and activation layers (ReLu for MLP and Sigmoid for LSTM), and input/output pre/post processing layers.

$$S_{in} = \begin{bmatrix} i_{dref}(k) & i_{qref}(k) \\ \dots & \dots \\ i_{dref}(k+p) & i_{qref}(k+p) \\ i_d(k-l) & i_q(k-l) \\ \dots & \dots \\ i_d(k) & i_q(k) \\ v_d(k) & v_q(k) \end{bmatrix} \quad \textbf{(5a)}$$

$$S_{out} = \begin{bmatrix} v_d(k+1) & v_q(k+1) \end{bmatrix} \quad \textbf{(5b)}$$

Where p is the prediction horizon of the MPC and l the length of the past history for LSTM neurons.

Equations Eq. (5) shows the construction of the network's input and output vectors. The input vector is built using the reference vector on the prediction horizon (from k to $k+p$), the measured current with an history of length l (used for the LSTM layers), and the voltage command at instant k. The ouput vector is simply the voltage command at instant $k+1$.

C. Dataset generation and neural network training

We consider the environment defined in Fig. 4 to generate the D-dimension dataset $[S_{in}, S_{out}]^D$ that will be used to train the network. As the IPMSM is a complex nonlinear system and

Table 4. RMSE comparison of each neural network types

Model	RMSE
MLP	0.0245
LSTM	0.0222
CNN	0.0790
LSTM-NN	0.0120
LSTM-CNN	0.0285

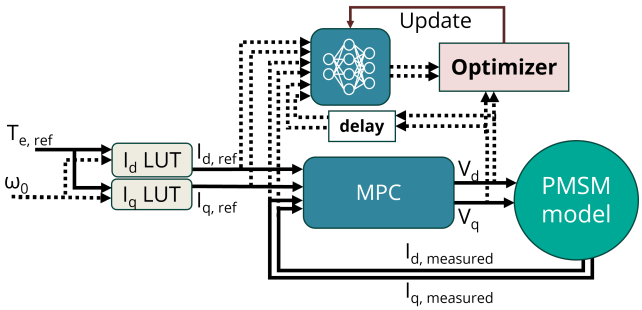

Fig. 4. Neural network training environment

considering long prediction horizons for MPC, it is challenging to create relevant datasets with a reasonable sample size. Limiting the amount of data used for learning is important because for this complex MPC problem, each sample may take a few seconds to be produced (Table 5) and this can quickly add up to an unrealistic amount of time. Therefore we limit the number of sample points to two million and only explore the *motor mode* (as opposed to the *generator mode*) inside the motor's map. The currents reference $i_{d,ref}$ and $i_{q,ref}$ are derived from two-dimensional look-up tables $LUT(speed, torque)$. The LUTs are pre-calculated using a FEM (Finite Element Model) tool. The nonlinear MPC is fed with a combination of pseudo-random steps, ramps and sinusoidal torque signals (Fig. 4) in a way to stimulate the system in the most operating states possible. Coverage of the operating range is supported using Latin Hypercube Sampling (LHS) which has shown better performance than random sampling or grid sampling [17]. Each stimulated signal is set for a duration of 2ms at the same sampling frequency as that of the MPC ($f_s = 100kHz$) in order to reflect a good representation and compromise between static and dynamic behaviors.

3. VALIDATION STUDY

A. Overview

The validation strategy is based on two steps: determine the operational validity of the proposed NN based controller to check if the FNN-MPC can correctly reproduce the behavior of the original MPC, and then compare performances against other reference controllers. In order to simulate the practical action in realistic EV drives, step and ramp of torque requests are applied to generate the IPMSM commands. Steps are used to highlight the advantages of MPC over FOC, and check if the FNN-MPC can keep these benefits. Ramps are used for more realistic motor simulations since this is how commands are generated under real operating conditions. A sampling frequency of 100kHz is chosen for the three controllers, as the goal in future work is to implement the solutions on a *Silicon Mobilty OLEA U* target, which is already capable of running an industrial application of FOC at 100kHz. It should be noted here that the speed is considered constant at ω_0 in all following simulations as a first assessment of the validity of the NN based control system. This will be extended to the whole possible range of speed in upcoming work.

A.1. Simulation environment

Simulations are based on *Matlab Simulink*™ with a MiL environment in *Rapid Acceleration* mode. Each controller is evaluated on the same experiment scenarios using the Key Performance Indicators (KPIs) of Table 5 and Table 6 to better assess the tangible benefits. KPIs are calculated from an average of each response for a total of 200 random 10ms scenarios. All measurements are conducted on a *12th Gen Intel(R) Core(TM) i7-1280P* CPU.

A.2. Reference controller

To assess more effectively the relevance of the FNN-MPC, we consider a reference controller for comparison. As the standard control method for AC synchronous and induction motors in the industry [18], the FOC is the best candidate for this task.

The FOC is based on two proportional integral regulators, followed by a decoupling layer. The reference currents are calculated using LUTs as described previously. The FOC used for comparison also features an anti-windup scheme on its outputs for better performances. No extra saturation is applied on the stator voltage (v_s) so that we can highlight the native saturation constraints available in the MPC. No flux weakening strategy is implemented as it will provide minimal performance improvement considering our simple simulation environment.

B. Operational validity

Static error is used to give a measure of the relative accuracy of the control model in terms of torque response. The static error is measured starting from $T_{settling}$ and represents the deviation from the reference in steady state. First, it can be noted that static error of the FNN-MPC (0.170% for step, 0.105% for ramp) is close to that of MPC (0.0111% for step, 0.0002% for ramp), reflecting satisfying neural network approximation. However, the static error of FOC is large in comparison (1.954% for step, 1.084% for ramp). This error comes mostly from the integrator of the FOC. Indeed, starting at 2% of the reference at $T_{settling}$, the currents do not have sufficient time to reach 0% within the 10ms of simulation time. This could be improved by fine-tuning the integrator parameters but at the risk of degrading other KPIs such as the overshoot. As a result, FNN-MPC is about ten times more efficient in terms of static error.

In Fig. 5, the different traces reported show the responses of each controller to steps and ramps torque requests. The step response emphasize some inherent benefits of the MPC. As the torque reference is known on the prediction horizon, the MPC is able to anticipate the commands on the control horizon and optimize the response time. A similar behavior is observed for the FNN-MPC which clearly shows that the neural network correctly learns the prediction capability of MPC. In the traces of Fig. 5, the norm of the voltage vector (v_s) is plotted against $v_{s_{max}}$. For the step response (Fig. 5a), a saturation is visible for MPC resulting from the direct processing of motor constraints in the optimization function. As for the FOC, no saturation is present thus reflecting the need for another scheme to limit the output voltage norm. On its part, the FNN-MPC correctly reproduces the saturation capability, further confirming successful training for this control feature. On the other side, ramp responses comply with motor gradient constraints and no overshoot on the norm voltage is present (Fig. 5b). In this case, MPC and FNN-MPC both stick to the reference signal in dynamic mode, unlike FOC showing a sensitive response deviation (also reflected in the KPI response times).

Table 5. Controller performance comparison on a step response

		FOC	MPC	FNN-MPC
$T_{response}$	(ms)	0.303	0.149	0.152
$T_{settling}$	(ms)	0.404	0.161	0.164
Overshoot	(%)	0.055	1.379	0.755
Static error	(%)	1.954	0.011	0.170
T_{exec}	(s)	1.47	6.82	2.07

Table 6. Controller performance comparison on a ramp response

		FOC	MPC	FNN-MPC
$T_{response}$	(ms)	0.067	0.010	0.014
$T_{settling}$	(ms)	0.144	0.010	0.024
Overshoot	(%)	0.052	0.123	0.039
Static error	(%)	1.084	0.0002	0.105
T_{exec}	(s)	1.510	8.51	1.69

Execution time T_{exec} is where the limitations of the MPC are most visible. The execution time is the time taken by the MiL to simulate a 10ms signal. A longer T_{exec} means a greater controller complexity, which increase the difficulty of a real-time implementation. With the FOC as reference for performance, FNN-MPC and MPC are respectively 26% and 414% slower (averaged from step and ramp). Thanks to the corresponding processing complexity improvement, real-time implementation of the FNN-MPC can be considered using specialised processors or dedicated hardware (contrary to the MPC).

C. Performance comparison

In the following, we address a detailed analysis of performance results. The KPIs defined for this purpose are derived from simulations and reported in Table 5 and Table 6 for step and ramp input requests respectively. In terms of system response, it can be observed that on average, response times are at a comparable level for the FNN-MPC in regards to the original MPC (0.152ms vs. 0.149ms for step command, 0.014ms vs. 0.010ms for ramp). However compared to FOC, FNN-MPC improvement is more significant (2 times faster for step, 5 times for ramp). Settling times follow a very similar pattern, with a great correlation between MPC and FNN-MPC (0.161ms vs. 0.164ms for step, 0.010ms vs. 0.024ms for ramp), compared to 2.5 to 6 times improvement against FOC.

Concerning overshoot, MPC has higher values than FOC because the original MPC cost function aims at minimizing the error on the current values for the whole horizon prediction, rather than reducing the overshoot. It is also interesting to note in the results that the percentage overshoot is lower for FNN-MPC (0.755% for step, 0.039% for ramp) than for MPC (1.379% for step, 0.123% for ramp). Indeed we should expect the FNN-MPC to overshoot more than the MPC as the behavior of the neural network is learned from the original MPC. This seeming contradiction comes from the construction of the training dataset, composed of 10000 step signals with an average overshoot of 0.52%. In the learning data, there is a higher representation

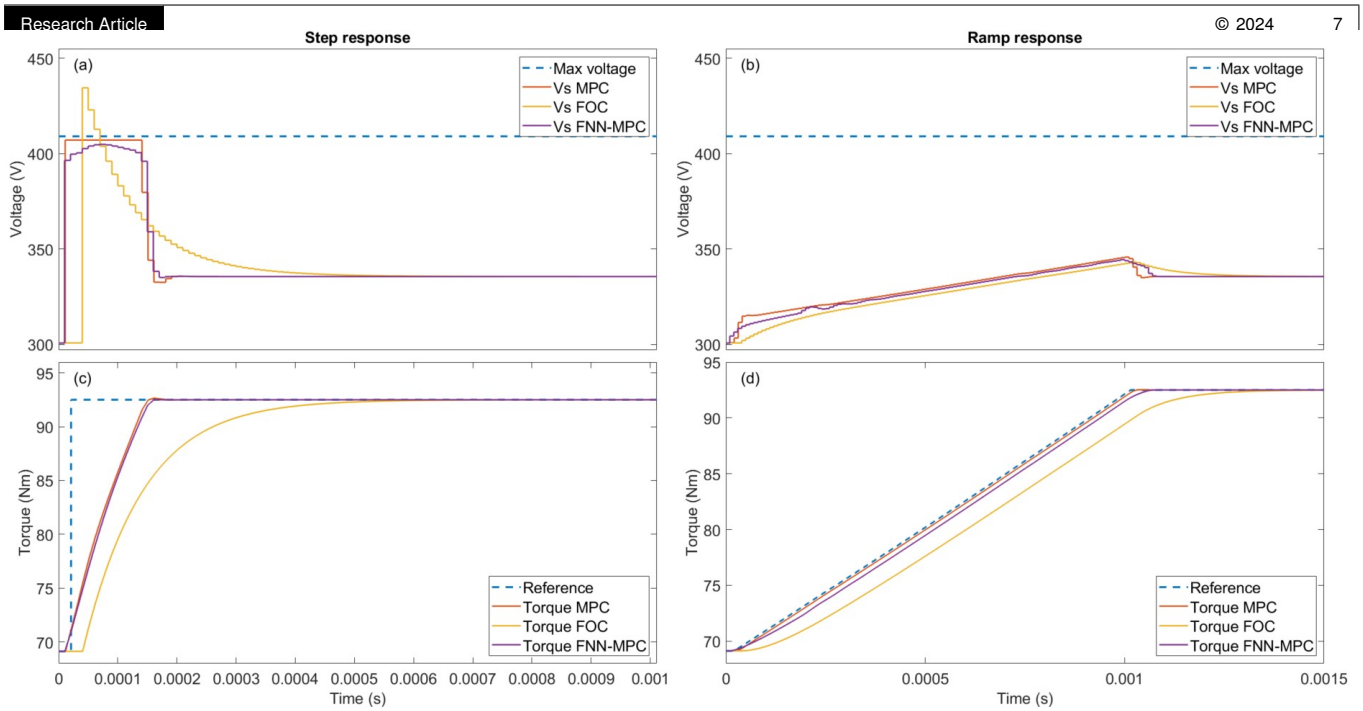

Fig. 5. Example of a step (c) and ramp (d) torque response for each controller. (a) and (b) are the corresponding v_s compared to $v_{s_{max}}$

of small steps (low percentage overshoot) compared to high steps (high percentage overshoot). This consideration was not anticipated at first but it can be exploited to better tune the FNN-MPC model in upcoming research. With minimum overshoot and despite a marginal reduction of performances compared to the MPC on previous performance KPIs ($T_{response}$, $T_{settling}$), the use of the FNN-MPC in place of the MPC in the control loop objectively brings additional control benefits.

4. CONCLUSION

In this paper, a hybrid RNN based approach for the control of traction motor drive systems is developed to overcome the computational complexity of a long-horizon nonlinear MPC. Neural Network design, training and validation have been explained in detail with special focus on dataset creation to obtain usable training and validation sets for the defined ANN. Finally, the resulting FNN-MPC strategy has been simulated and compared with a reference field-oriented controller and a MPC in *Matlab, Simulink*™ [6]. Validation study have shown that the ANN is capable of correctly approximating the nonlinear IPMSM electrical drive control system. And while being nearly five times faster than the MPC, the FNN-MPC still globally improve the control quality against the FOC reference controller.

Future work will essentially address implementation issues related to the FNN-MPC definition (training at constant speed) and simulation with gradually more realistic conditions (modulation, inverter and noise models) using *Hardware in the Loop* (HIL) implementation targeting the *Silicon Mobility OLEA(r) U FPCU* with programmable hardware.

REFERENCES

1. Max Schwenzer and Thomas Bergs and Muzaffer Ay and Dirk Abel, Review on model predictive control, an engineering perspective, The International Journal of Advanced Manufacturing Technology, 2021.
2. S. Zhang, O. Wallscheid, and M. Porrmann, Machine learning for the control and monitoring of electric machine drives: Advances and trends, IEEE Open Journal of Industry Applications, 2023.
3. M. Zolfaghari, S. A. Taher, and D. V. Munuz, Neural network-based sensorless direct power control of permanent magnet synchronous motor, Ain Shams Engineering Journal, 2016.
4. Q. Gao and C. Zong, Rotor position estimation method of pmsm based on recurrent neural network, Elsevier CPESE, 2021.
5. J. Li and T. Akilan, Global attention-based encoder-decoder lstm model for temperature prediction of permanent magnet synchronous motors, IEEE arXiv, 2022.
6. T. M. Inc., Matlab version: 23.2.0 (r2023b), Natick, Massachusetts, United States, 2023. [Online]. Available: https://www.mathworks.com
7. P. Sørensen, M. Nørgaard, O. Ravn, and N. Poulsen, Implementation of neural network based non-linear predictive control, Elsevier Neuro-computing, 1999.
8. K. Premkumar and V. Priya, Speed control of permanent magnet synchronous motor using neural network model predictive control, Journal of Energy Systems, 2020.
9. S. Spielberg, P. Kumar, A. Tulsyan, B. Gopaluni, and P. Loewen, A deep learning architecture for predictive control, Elsevier IFAC, 2018.
10. I. Hammoud, S. Hentzelt, T. Oehlschlaegel, and R. Kennel, Long-horizon direct model predictive control based on neural networks for electrical drives, 46th Annual Conference of the IEEE Industrial Electronics Society (IECON 2020), 2020.
11. M. Abu-Ali, F. Berkel, M. Manderla, S. Reimann, R. Kennel, and M. Abdelrahem, Deep learning-based long-horizon mpc: Robust, high performing, and computationally efficient control for pmsm drives, IEEE transactions on power electronics, 2022.
12. H. Mao, X. Tang, and H. Tang, Speed control of pmsm based on neural network model predictive control, transactions of

the institute of measurement and control, Sage Transactions of the Institute of Measurement and Control, 2022.
13. M. Bendjedia, Synthèse d'algorithmes de commande sans capteurs de moteurs pas à pas et implantation sur architecture programmable, Ph.D. dissertation, Université de Franche-Compte, 2007.
14. J. G. Cintron-Rivera, S. N. Foster, C. A. Nino-Baron, and E. G. Strangas, High performance controllers for interior permanent magnet synchronous machines using look-up tables and curve-fitting methods, IEEE International Electric Machines and Drives Conference, 2013.
15. S. Boyd and L. Vandenberghe, Convex optimization, Cambridge university Press, 2004.
16. A. B. Risum and R. Bro, Using deep learning to evaluate peaks in chromatographic data, Elsevier Talanta, 2019.
17. M. Tahkola, J. Keränen, D. Sedov, M. F. Far, and J. Kortelainen, Surrogate modeling of electrical machine torque using artificial neural networks, IEEE Access, 2020.
18. H. Chen, X. Gong, Y.-F. Hu, Q.-F. Liu, B.-Z. Gao, and H.-Y. Guo, Automotive control: the state of the art and perspective, Science Direct Acta Automatica Sinica, 2013.