Software Impacts 15 (2023) 100479

Contents lists available at ScienceDirect

Software Impacts

journal homepage: www.journals.elsevier.com/software-impacts

Original software publication

Modelling the Field Oriented Control applied to a 3-phase Permanent Magnet Synchronous Motor

Juan Camilo Nustes [a,b,*], Danilo Pietro Pau [a], Giambattista Gruosso [b]

[a] System Research and Applications, STMicroelectronics, via C. Olivetti 2, Agrate Brianza, I-20864, Italy
[b] Department of Electronics, Information and Bioengineering, Politecnico di Milano, Milano, Piazza Leonardo da Vinci, 32 - 20133, Italy

ARTICLE INFO

Keywords:
Motor control
Simulink
Neural Networks
PID control
Field Oriented Control

ABSTRACT

This paper describes the software implementation of the Field Oriented Control (FOC) model applied to a 3-phase Permanent Magnet Synchronous Machine (PMSM) and its components. Furthermore, it explains how it can be used to generate the PMSM_FOC dataset made available through Data in Brief[1]. This dataset was generated using a Simulink representation for both the motor and the control scheme when a series of different speed targets are fed as input to the motor. Furthermore, this paper elaborates on the input signals designed for the dataset generation, while also proposing different scenarios where the Simulink model can be used for future motor control development.

Code metadata

Current code version	v1.1
Permanent link to code/repository used for this code version	https://github.com/SoftwareImpacts/SIMPAC-2023-50
Permanent link to reproducible capsule	https://codeocean.com/capsule/7793846/tree/v1
Legal code license	Apache 2.0
Code versioning system used	none
Software code languages, tools and services used	Created with Matlab 2021b, Simulink
Compilation requirements, operating environments and dependencies	Matlab 2020a or higher Motor Control Blockset DSP System Toolbox
If available, link to developer documentation/manual	https://github.com/juancnustes/FOC_PMSM/blob/main/Documentation
Support email for questions	juancnustes@gmail.com danilo.pau@st.com giambattista.gruosso@polimi.it

1. Introduction

This paper describes in detail the Simulink model for the Field Oriented Control (FOC) applied to a 3-phase Permanent Magnet Synchronous Machine (PMSM) based on the Trapezoidal Back EMF model presented in [1]. The software model of the control scheme implemented in Simulink is described in Section 2 and it is provided in the link C2 in the Code Metadata table above. The controller goal is to follow the speed reference set by different speed stimuli. The FOC control consists in two separate closed loops, one for speed control (outer loop), and one for the current or torque control (inner loop) as shown in Fig. 1. The control scheme is based on linear Proportional Integral Derivative PID regulators in both loops connected in cascade [2,3].

The FOC model is fed by various inputs to achieve speed control. Some of the inputs were designed to highlight the linear behaviour of the Proportional Integral Derivative (PID) controller when a fast or high magnitude speed variation occurs as in [4,5]. One goal of this model is to be able to process several input conditions that can be used to generate a dataset containing relevant scenarios and can be later implemented in approaches other than PID (e.g. Approaches

The code (and data) in this article has been certified as Reproducible by Code Ocean: (https://codeocean.com/). More information on the Reproducibility Badge Initiative is available at https://www.elsevier.com/physical-sciences-and-engineering/computer-science/journals.

* Corresponding author at: System Research and Applications, STMicroelectronics, via C. Olivetti 2, Agrate Brianza, I-20864, Italy.
E-mail addresses: juancamilo.nustes@mail.polimi.it (J.C. Nustes), danilo.pau@st.com (D.P. Pau), giambattista.gruosso@polimi.it (G. Gruosso).

[1] https://data.mendeley.com/datasets/6tjkgtfnky

https://doi.org/10.1016/j.simpa.2023.100479
Received 31 January 2023; Received in revised form 9 February 2023; Accepted 13 February 2023

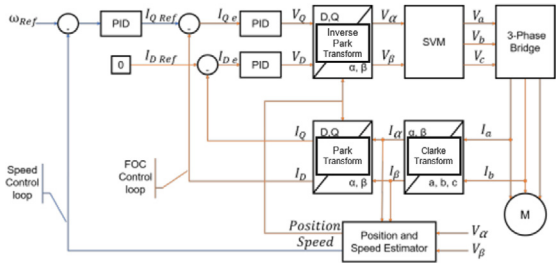

Fig. 1. Field Oriented Control Block-scheme representation.

Table 1
Motor parameters gathered from the Motor Profiler tool.

Parameter	Abbreviation	Value	Measure unit
Pole Pairs	p	7	–
Max. Speed	w_{max}	15000	Rpm
Nominal Current	I_n	1.20	Apk
Nominal DC voltage	V_{DC}	12.0	V
Stator Resistance	Rs	0.11	Ohm
Stator Inductance	Ls	0.018	mH
Back-Emf Constant	B-Emf	0.4	Vms/krpm

based on Machine Learning techniques such as on-device-learning and inferencing of Neural Networks) [6]. Other possibilities could be to use the model to verify the control scheme in different type of motors, calculate the maximum speed when a torque is applied, or to tune the controller and obtain a more aggressive controller with a lower settling time, by using difference type of signal processing (not limited to machine learning ones).

2. Development

2.1. FOC scheme

To develop the FOC control scheme inside the Simulink environment, the motor control blockset [7] must be installed, which includes pre-built blocks for the Clarke, Park, and Inverse Park transforms. To build the FOC control scheme it is needed to connect the blocks following the diagram shown in Fig. 1.

The Clarke transform gathers the 3-phase currents I_a, I_b, I_c that feed the motor and converts them to 2-phase currents I_α, I_β in a stationary reference frame [8]. The Park transform converts the 2-phase stationary currents I_α, I_β into the currents I_D, I_Q in a rotating DQ frame [8], these currents are then fed back to the PID regulators in the internal loop control.

The output of the PID regulators are the voltages V_D, V_Q which are still under the rotating reference frame, to bring the voltages back to a stationary reference frame, the Inverse Park transform is applied, and the voltages V_α, V_β are obtained.

The Space Vector Modulator (SVM) [9] is responsible for generating pulse width modulated signals to control the switches of an inverter, which then produces the required modulated voltage to drive the motor at the desired speed or torque.

The modulated voltage signals generated by the PWM are then fed to the motor inverter to produce the 3-phase AC voltage that feeds the Motor.

2.2. Motor model

To factor into the motor model inside the Simulink environment, a series of blocks inside the Motor control blockset [7] can be used. In this case, the "Surface mount PMSM" is the block used to simulate the motor behaviour. This block internally considers the motor equations for the non-sinusoidal back-EMF, flux linkage and peak currents [10],

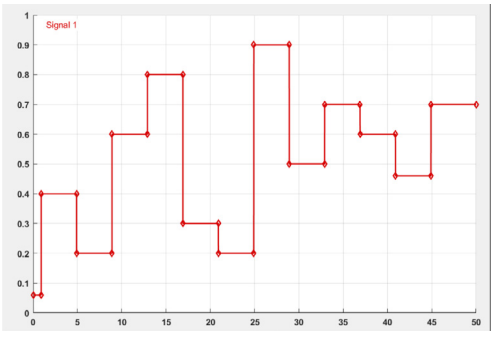

Fig. 2. Manually generated signal from the signal builder block.

and it can be fine-tuned with the desired motor parameters such as the pole pairs and maximum rated speed, in this paper, the parameters are presented in Table 1, which were obtained through the ST motor profiler [11]. The model implementation in Simulink is based on the sensorless FOC scheme developed by Mathworks [12].

2.3. Input Stimuli definition

The inputs can be defined to verify that the control scheme is working as expected, or to take the controller to its limit conditions. Furthermore, a set of various input shapes are proposed to generate a complete dataset considering both linear and non-linear situations. The input provides a speed reference for the FOC scheme, the inputs considered in this paper are Step, Ramp, Random signal, and manually generated signals.

The step input is the starting case, as it only requires an initial and final speed value, as well as to set the time where the speed change occurs. The ramp input has a similar definition, but instead of setting a final speed, the slope of the ramp is specified, which is the incremental value of the speed each elapsed second.

The random signal generation is achieved with the matlab block "Random Source", included in the DSP Systems Toolbox [13] which randomly selects a mean value between a range set by the user, and then, a signal is generated around the randomly selected mean value. Finally, the input signal can be manually designed to have multiple speed changes with different magnitude and duration, the block used to accomplish this is the "Signal Builder" block, where the user can graphically design the signal shape and values to specifically test the controller in the critical conditions that the user wants to verify. An example of a signal generated with this block is shown in Fig. 2.

The step and ramp inputs are relevant in the control design to measure properties such as the settling time, the overshoot and the oscillations, while the random signal is useful to check how fast is the controller stabilization and to see if it is able to track a constantly varying reference. The manually built signal is useful to design specific boundary conditions where you expect the controller to fail. A combination of all these inputs leads to a very complete dataset that can be potentially used in training and validating Neural Networks.

2.4. Other potential applications of the software

The developed model opens the possibility for the user to test different types of controllers, motors, speed profile inputs, and applied torque values that conform the motor control system. As an example, a user could verify the control behaviour on a specific modelled motor by fine tuning the controller gains (via multiple pre computed set points) and verifying through the Simulink scope how the speed control changes. Also, there is the possibility to apply the control scheme to different type of motors just by changing the motor parameters in the Matlab script, as well as applying different torque loads. Other

Table 2
Possible user variations and their respective contribution to the overall modelled system.

Changed component	Effect
Speed input	Change of the motor response.
Controller gain	A different control scheme is implemented.
Motor parameters	The control scheme is implemented to a different motor.
Applied torque	Reduction of the maximum motor speed.

possibility, could be to define the speed profile required in a specific situation by defining a new input case and verify if the system is able to perform the task.

Finally, the user can define new speed profiles that could highlight corner conditions (e.g. further non linearities) of the PID controller in order to be included in the dataset published in Data in Brief so that this dataset can include more relevant input cases useful for developing new control strategies such as (but not limited to) a Machine Learning approach. In Table 2 are shown some of the system components that can be changed by the user and their respective contribution to the modelled system. In this way, the user can have an idea on how to modify the software to obtain a control scheme for a defined motor that better suits his particular interest.

3. How to run the model

The implementation of the FOC scheme in Matlab and published in the GitHub repository is provided in the link C2 in the Code Metadata table above, includes the following two files.

-The model initialization script "*Motor_script.m*" which computes the linear controller gains as well as assigns the motor and inverter parameters.

-The Simulink file "*FOCsimulation.slx*" which includes the input test cases, the FOC control scheme and the motor model.

Before running anything, verify that the "*motor control blockset*" [7] as well as "*DSP Systems Toolbox*" [13] add-ons are installed.

The first step is to run the *Motor_script* file, run it and verify that it compiled. Then, open the Simulink file *FOCsimulation*, where you find three main blocks, representing the inputs, control scheme and motor model described in Section 2 In the Simulink environment, open the "Speed tracking" scope and run the simulation. To verify that the controller is working properly, use the scope to visually confirm that the measured speed is in fact tracking the speed reference.

More test cases are included in the Simulink model, they are defined inside the Input main block. To change the speed reference. manually connect the desired input case and adjust the simulation time to the value specified in the input name [Ts = value]. After changing the target speed, run the Simulink file again, there is no need to run the .m file after changing the inputs. For a more detailed Step by step guide read the documentation paper provided in the link C8 in the Code Metadata table above.

Finally, to change the input torque, open the motor model main block, and change the constant value that is defining the input torque (currently set to zero to maximize motor speed). The motor parameters are defined in the "Motor_script.m" file, so is not recommended to change the parameters defined inside the Simulink block but on the initialization code itself.

4. Impacts

The modelled system can be manipulated by the users to simulate their particular test scenarios, including critical corner cases, and verify the controller for a specific motor. The model also has the capability to generate data at different simulation points in order to build a dataset that mimics the behaviour of any given PMSM just by adjusting the motor parameters inside the Simulink model, and then repeating the data generation process. The generated data has great value as it mimics the motor behaviour and brings the possibility to gather new data without needing to physically have the motor. Finally, the acquired data has great potential to develop different motor control approaches which not necessarily rely on classic linear controllers such as PID regulators, for example, it brings the possibility to develop control schemes based on Machine Learning approaches as proposed in [14,15] as well as other type of signal processing algorithm the user may conceive.

5. Future developments

The authors plan to provide updates on new input signals that can work as test cases where the linear behaviour of the PID controller is highlighted, in this way a more complete dataset from the motor can be obtained. The authors are also aware that the "Signal builder" may not be compatible with some blocks that may be part of your control scheme.

CRediT authorship contribution statement

Juan Camilo Nustes: Conceptualization, Methodology, Software, Formal analysis, Investigation, Writing – original draft, Writing – review & editing. **Danilo Pietro Pau:** Conceptualization, Validation, Writing – original draft, Writing – review & editing, Supervision, Project administration. **Giambattista Gruosso:** Conceptualization, Validation, Writing – original draft, Writing – review & editing, Supervision, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] B.P. Kumar, C.M.C. Krishnan, Comparative study of different control algorithms on Brushless DC motors, in: 2016 Biennial International Conference on Power and Energy Systems: Towards Sustainable Energy, PESTSE, 2016, pp. 1–5.
[2] N. Federici, D. Pau, N. Adami, S. Benini, Tiny reservoir computing for extreme learning of motor control, in: 2021 International Joint Conference on Neural Networks, IJCNN, 2021, pp. 1–8.
[3] F. Korkmaz, et al., Comparative performance evaluation of FOC and DTC controlled PMSM drives, in: 4th POWERENG, 2013, pp. 705–708.
[4] A. Flah, S. Lassaad, An improved PMSM drive architecture based on BFO and neural network: Regular paper, in: IJARS, Vol. 10, 2013.
[5] H.P.H. Anh, et al., Advanced speed control of PMSM motor using neural FOC method, in: 2018 4th International Conference GTSD, 2018, pp. 696–701.
[6] S. Husach, R. Yatsiuk, Research on neural network vector control system for induction motor, in: 2020 IEEE Problems of Automated Electrodrive. Theory and Practice, PAEP, 2020, pp. 1–4.
[7] Mathworks. Motor control blockset. URL: https://www.mathworks.com/products/motor-control.html.
[8] C.J. O'Rourke, M.M. Qasim, M.R. Overlin, J.L. Kirtley, A geometric interpretation of reference frames and transformations: dq0, clarke, and park, IEEE Trans. Energy Convers. 34 (4) (2019) 2070–2083.
[9] D.O. Neacsu, Space vector modulation - An introduction – Tutorial at IECON2001, 2001.
[10] Parag Kshirsagar, R. Krishnan, High-efficiency current excitation strategy for variable-speed nonsinusoidal back-emfpmsm machines, IEEE Trans. Ind. Appl. 48 (6) (2012) 1875–1889.
[11] STMicroelectronics. ST Motor profiler. URL: https://wiki.st.com/stm32mcu/wiki/STM32MotorControl:STM32_MC_Motor_Profiler.
[12] Sensorless FOC scheme. Mathworks. URL: https://ww2.mathworks.cn/help/mcb/gs/sensorless-foc-pmsm-smo-fo.html.
[13] DSP systems toolbox. Mathworks. URL: https://www.mathworks.com/products/dsp-system.html.
[14] S. Suman, et al., Novel approach of speed control of PMSM driveusing neural network controller, 2016, pp. 2780–2783.
[15] K. Cheon, et al., On replacing PID controller with deep learning controller for DC motor system, J. Autom. Control Eng. 3 (2015) 452–456.