Federico Giai Pron

PMSM control

User guide

Giai Pron, Federico
3-17-2022

Contents

Figures

Symbols

Variables

Symbol	Name	Unit	Unit symbol
I	Current	Ampere	A
θ	Angular position	Radiant	rad
V	Voltage	Volt	V
ω	Angular speed	Radiant / second	rad/s
n	Angular speed	Rounds / minute	rpm
α	Angular acceleration	Radiant / square second	rad/s^2
p	Pole pairs number	1	1
$Mode$	Modality	1	1
$e_{Variable}$	Error between reference and feedback	1	1
C	Torque	Newton · meter	Nm
λ	Flux	Weber	Wb
R	Resistance	Ohm	Ω
L	Inductance	Farad	F
$k_{Variable}$	Constant	Variable	Variable
v	Reference voltage	1	1
$Carrier$	Carrier signal	1	1
S	Pulse	1	1
$bemf$	Back electromotive force	Volt	V
BW	Basic frequency	Hertz	Hz
t	Time / period	Second	s

Subscripts

Symbol	Name
0	Neutral
a	a-axis
b	b-axis
c	c-axis
α	α-axis
β	β-axis
d	d-axis
q	q-$axis$
m	Mechanical
e	Electrical
l	Load
f	Friction
CM	Common mode
s	Switching

Superscripts

Symbol	Name
$fdbck$	Feedback
cmd	Command
$ref,1$	Reference (from internal loop)
ref	Reference (selected)
$base$	Base
PM	Permanent magnets
$PMSM$	Permanent magnet synchronous motor
max	Maximum
DC	Direct current (/ bus)
FW	Flux-weakening
$MTPA$	Maximum torque per Ampere
$THIPWM$	Third harmonic injection

$CVCP$	Constant voltage – constant power
PSO	Particle swarm optimization method
$SVPWM$	Space vector modulation
FOC	Field oriented control
$CVCT$	Constant voltage – constant torque
$MTPV$	Maximum torque per Volt

Overview

As shown in Figure 1 - System description, the system under analysis consists of:

- User interface, with rotary switches, knobs, and scopes, that allow the user to select the operating mode of the inverter, to vary the command voltages / currents / torque / speed / angle, and to visualize the main simulation results
- Controller, that converts the command from the user into pulses to the inverter (S_{123456})
- Plant model, that simulates the behavior of the inverter and of the permanent magnet synchronous motor

The main contents of the package are:

- Automatic vectors / maps calibration based on PMSM parameters
- Automatic PIDs tuning
- Controller and plant model

The main contents of the model are:

- Clark and Park transformations
- Active short circuit
- Six switch open
- Field oriented control (FOC) with:
  - Voltage open loop control
  - Current closed loop control
  - Torque open loop control
  - Speed closed loop control
  - Angle closed loop control
- Third harmonic injection (THIPWM)
- Space vector modulation (SVPWM)
- Inverse Clark and Park transformation
- Maximum torque per ampere (MTPA)
- Flux weakening:
  - Maximum torque per volt (MTPV)
  - Constant voltage – constant torque (CVCT)
  - Constant voltage – constant power (CVCP)
- Inverter model
- Permanent magnet synchronous motor (PMSM) model

Note: as highlighted in the following chapters, some tools in the package have been "obscured". To access these contents, or to request a detailed description of each script / model, please, contact me by e-mail.

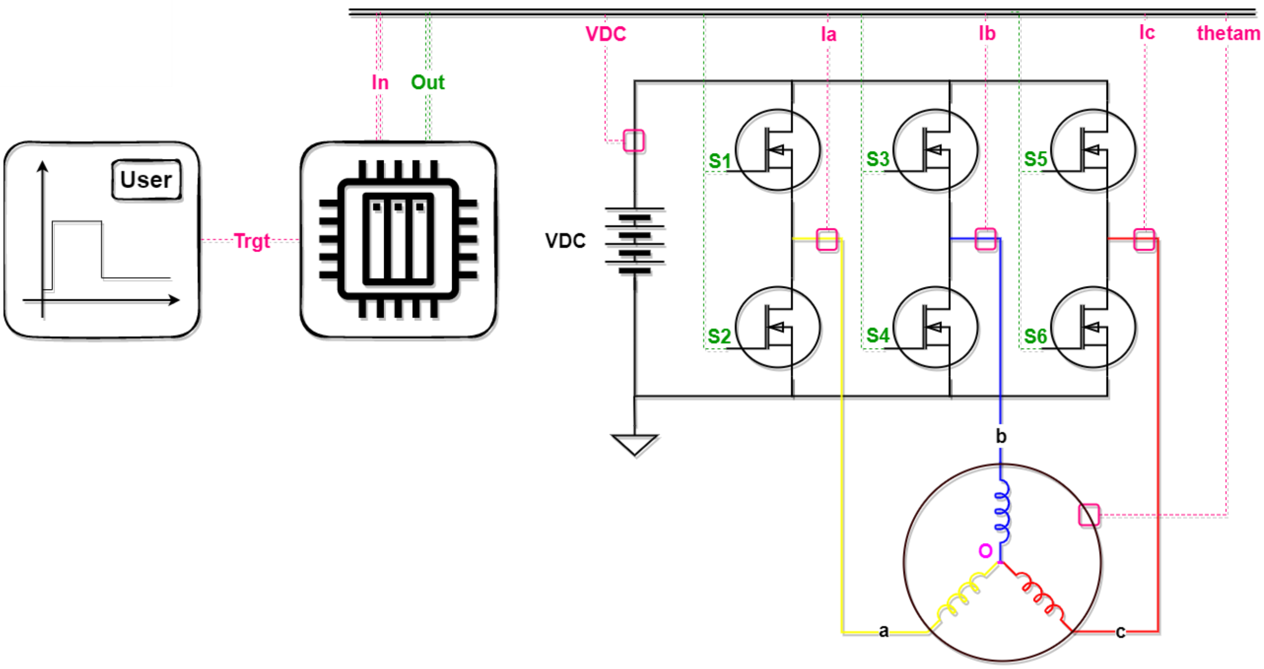

Figure 1 - System description

Tools

The package contains 5 tools, as shown in Table 1 - Tools within the package.

Tool	**Goal**
1_GenerateMap_IdqRef	Based on the PMSM parameters, generates: • Reference currents in the d-q frame as a function of the speed and of the reference torque • Maximum / minimum torque and power as a function of the speed
2_TunePID_Idq	Based on the PSO method, tunes the PID coefficients of the current closed loop
3_TunePID_omegam	Based on the PSO method, tunes the PID coefficients of the speed closed loop
4_TunePID_V	Based on the PSO method, tunes the PID coefficients of the voltage closed loop while in flux-weakening mode (not active by default)
5_SimulateInverter	Simulate the PMSM controller and plant

Table 1 - Tools within the package

1_GenerateMap_IdqRef

Run

- Open 1_GenerateMap_IdqRef / GenerateMap_IdqRef_Main_v03.m within the MATLAB environment and modify the PMSM parameters
- Run 1_GenerateMap_IdqRef / GenerateMap_IdqRef_Main_v03.m and wait some seconds for the results (the vectors / maps are generated with a high number of breakpoints, therefore the script can take some time to run)

Results

As shown in Figure 2 - 1_GenerateMap_IdqRef results – Reference currents in the d-q frame as a function of the speed and of the reference torque and Figure 3 - 1_GenerateMap_IdqRef results - Maximum / minimum torque and power as a function of the speed, the results that are obtained by running the script with the default parameters are:

- Reference currents in the d-q frame as a function of the speed and of the reference torque

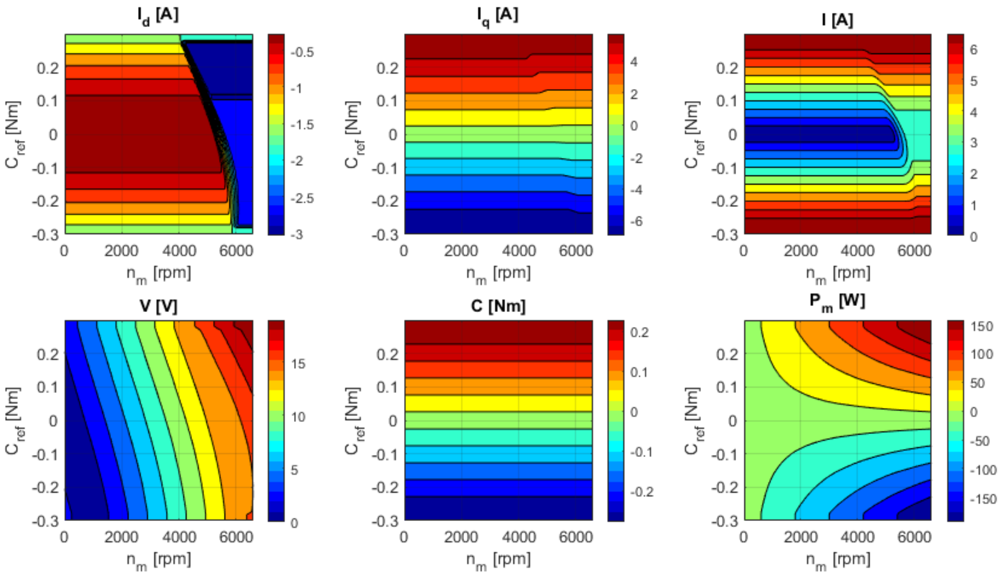

Figure 2 - 1_GenerateMap_IdqRef results – Reference currents in the d-q frame as a function of the speed and of the reference torque

- Maximum / minimum torque and power as a function of the speed

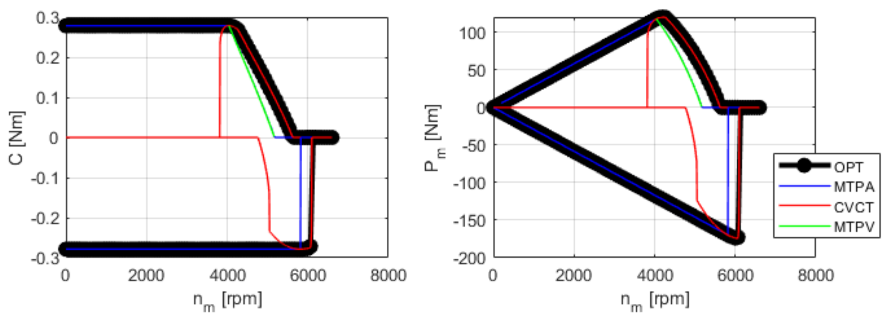

Figure 3 - 1_GenerateMap_IdqRef results - Maximum / minimum torque and power as a function of the speed

Note: to access the results of the run, it is necessary to use the "not obscured" files. *In this case, the tool outputs a structure, OUT, which contains the* I_{dq}^{ref} *maps (OUT.Id and OUT.Iq as a function of OUT.nm and OUT.C_ref) and the* $C^{ref,max/min}$ *vectors (OUT.C_ref_max and OUT.C_ref_min as a function of OUT.nm).*

2_TunePID_Idq, 3_TunePID_omegam, 4_TunePID_V

Run

- Copy all the files contained in *2_TunePID_Idq* / *3_TunePID_omegam* / *4_TunePID_V* (one of the folders only) and paste them in *5_SimulateInverter* / *Open.*
- Open *5_SimulateInverter* / *Open* / *Parameters.m*, comment the following lines and modify the PMSM parameters.
  % 1 - clear all
  % 2 - close all
  % 3 – clc
- Open *5_SimulateInverter* / *Open* / *Model.slx*, select the desired operating mode, and regulates the command voltages / currents / torque / speed / angle based on the PID to be tuned
  - To tune the PID coefficients of the current closed loop, for example, I selected *Mode_cmd* equal to 3 (torque control), and I chose a well-defined *C_cmd* profile
  - To tune the PID coefficients of the speed closed loop, for example, I selected Mode_cmd = 4 (speed control), and I chose a well-defined *omegam_cmd* profile

  Uncomment the following *To Workspace* blocks in *5_SimulateInverter* / *Open* / *Model.slx*:
  - $e_{I_{dq}}$ (TunePID_Idq)
  - V_{dq0}^{ref} (TunePID_Idq)
  - e_{ω_m} (TunePID_omegam)
  - C^{ref} (TunePID_omegam)
  - e_V (TunePID_V)
- Open *5_SimulateInverter* / *Open* / *TunePID_Idq.m / TunePID_omegam.m / TunePID_V.m* and modify the PSO parameters. Note that, depending on them, the optimization can bring more or less time
- Open *5_SimulateInverter* / *Open* / *ObjFun_fun.m*, where the objective function is calculated based on the signals coming from the Simulink model, and modify it if necessary
- Run *5_SimulateInverter* / *Closed* / *TunePID_Idq.m* / *TunePID_omegam.m* / *TunePID_V.m* and wait until the end of the optimization or when you think the results are sufficiently good

Note: to uncomment the To Workspace blocks in the Model.slx file it is necessary to use the "not obscured" model. Without it, it is not possible to run the optimization scripts.

Results

Example of results can be found in *2_TunePID_Idq* / *TunePID_Idq_Results.txt* and *3_TunePID_omegam* / *TunePID_omegam_Results.txt.*

5_SimulateInverter

Run

- Open *5_SimulateInverter* / *Open* / *Parameters.m* within the MATLAB environment and modify the PMSM parameters. Note that, if the PMSM parameters are modified, then it is necessary to adjust the calibration vectors / maps by following the procedure shown in *1_GenerateMap_IdqRef*
- Open *5_SimulateInverter* / *Open* / *Model.slx* within the Simulink environment
- Select the command mode (*Mode_cmd*) through the corresponding rotary switch and modify the command voltages (*Vd_cmd* and *Vq_cmd*) / currents (*Id_cmd* and *Iq_cmd*) / torque (*C_cmd*) / speed (*nm_cmd*) / angle (*thetam_cmd*) through the corresponding knob
- Open the *Vdq0*, *Idq0*, *C*, *nm* and *global* scopes to look at the simulation results

Results

An example of results is shown in Figure 4 - 5_SimulateInverter results. This has been obtained by setting *Mode_cmd* equal to 3 (torque control) and modifying the command torque (*C_cmd*) through the corresponding knob. The significant signals to be looked at are:

- *Vdq0_ref*, i.e. the reference voltages in the d-q frame actuated by the inverter
- *Idq0_fdbck*, i.e. the feedback currents in the d-q frame
- *C_cmd* and *C_fdbck*, i.e. the commanded and estimated torques
- *nm_fdbck*, i.e the feedback speed

The signals *Vdq0_cmd*, *Idq0_cmd*, and *nm_cmd* are equal to 0 because not in voltage / torque / speed control.

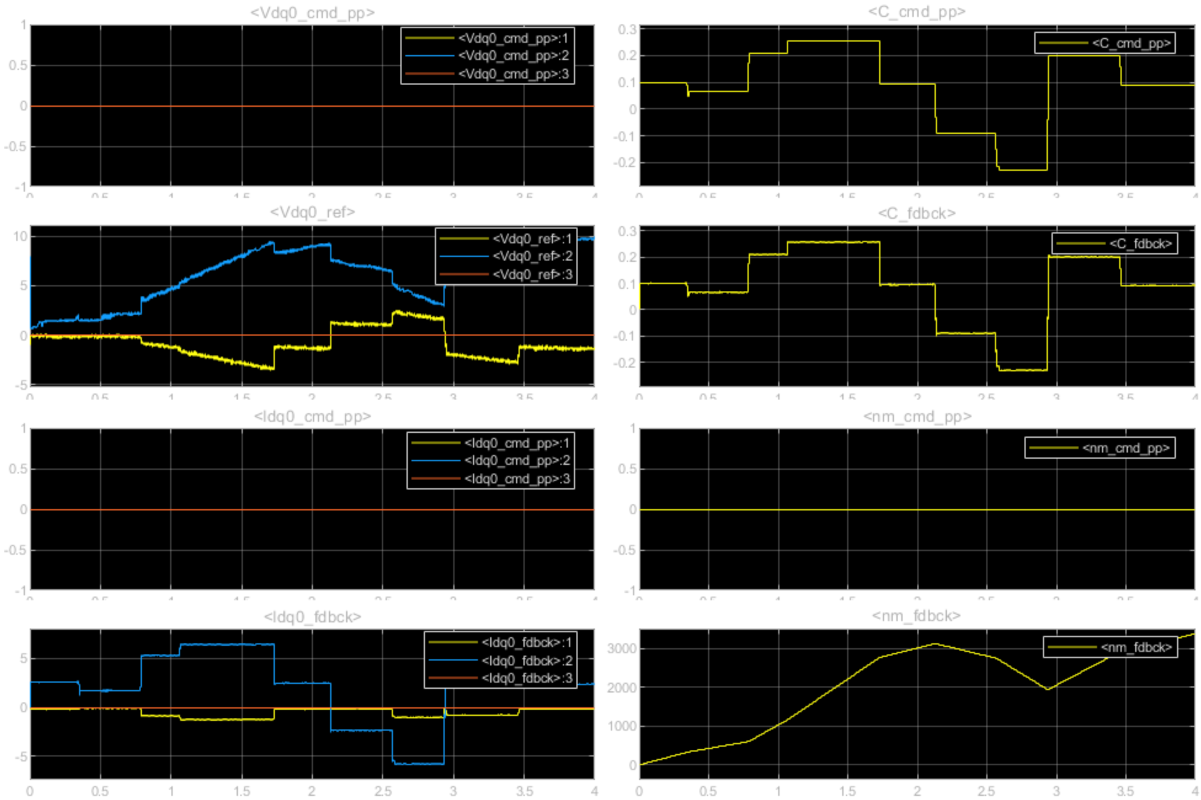

Figure 4 - 5_SimulateInverter results