



**ECE 730**  
**Control of Adjustable Speed Drives**

**Lab 2**  
**Vector Control of Permanent-Magnet  
Synchronous Machine**

**Due date:** Mar. 2, 2025 (before 11:59PM)  
Please upload report on Avenue to Learn:  
Assessments> Assignments> Lab 2

Late submissions won't be marked.

## Objective

- Implementing a vector control of a three-phase Permanent-Magnet Synchronous Machine using MATLAB-Simulink.

**Note-** You are supposed to be familiar with MATLAB-Simulink and are able to run your simulation either remotely or on your computer.

## Procedure

Figure 1 shows the block diagram of the current control loop in the framework of vector control of sinusoidal Permanent-Magnet synchronous machines (PMSM).

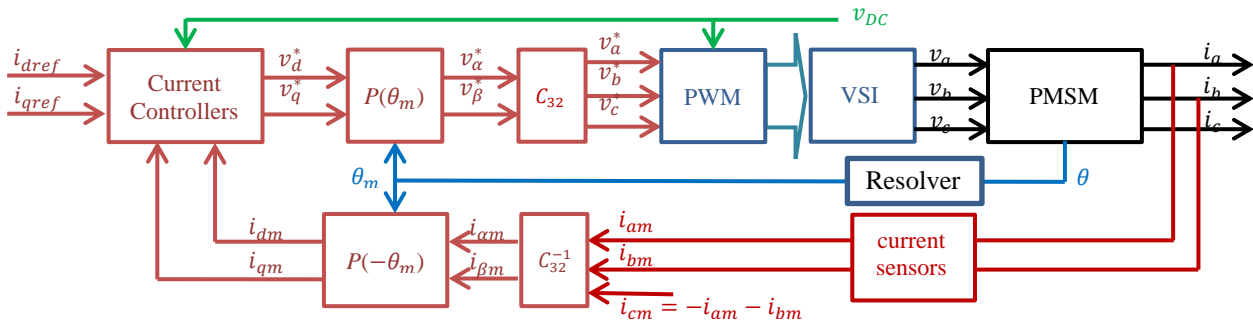


Figure 1- Vector control of PMSM: current control loop.

In this figure, the block PMSM has been already developed in Lab 1. Current and position measurements are supposed to be accurate and transformation matrices in the controller (blocks in red) are the same as those already used in the modeling of PMSM in Lab 1.

For testing the current controllers' performance in simulation, we assume in this lab that PWM+VSI is replaced by its average model:

$$G_{VSI} = 1 \quad (1)$$

As seen in Topic 3, the above block diagram can be simplified to the following under conditions discussed in Topic3:

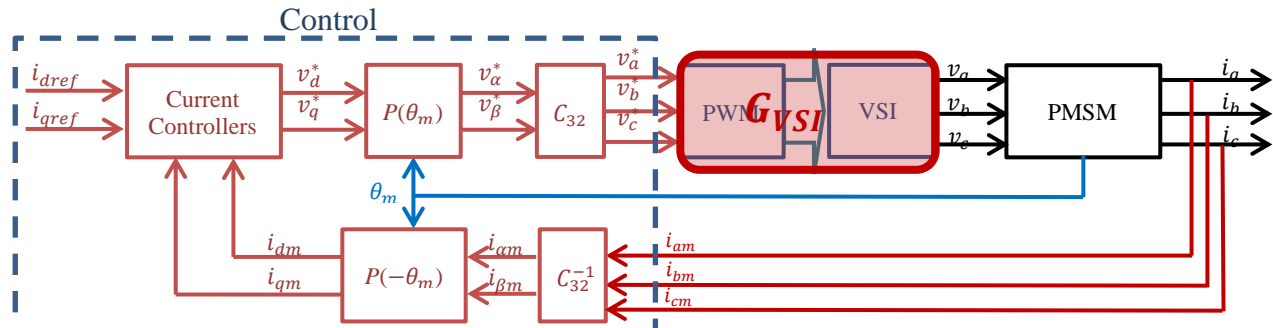


Figure 2- Vector control of PMSM: simplified current control loop.

## I. Current controller tuning:

Assume an accurate decoupling control is implemented:

$$\begin{cases} v_d^* = (u_d - L_q \cdot \omega \cdot i_{qm})/G_{VSI} \\ v_q^* = (u_q + L_d \cdot \omega \cdot i_{dm} + \Psi_f \cdot \omega)/G_{VSI} \end{cases} \quad (2)$$

We select PI controller to control  $i_d$  and  $i_q$ .

Q1) To achieve the following performances:

- Zero steady-state error
- No overshoot
- Closed-loop bandwidth  $f_{BW} = 500 \text{ Hz}$

tune PI controllers using pole-zero compensation technique:

$$\begin{aligned} K_{pd} &= \quad , K_{id} = \\ K_{pq} &= \quad , K_{iq} = \end{aligned}$$

**Note-** Machine parameters have been specified in Lab 1.

## II. Current controller simulation:




- 1) Implement PI controllers for  $i_d$  and  $i_q$  with the above calculated gains in Simulink.
- 2) Implement the decoupling control in Simulink (see Eq. (2)).
- 3) Consider  $i_{dref} = 0 \text{ A}$  and  $i_{qref}$  has a step change of  $+10 \text{ A}$  (from  $I_{q0}$  to  $I_{q0} + 10$ ) at  $t = 0.1 \text{ s}$ . Run the simulation and save the following results:  $i_d, i_q, v_d, v_q, \Omega$ .
- 4) Zoom on the above variables after the step change on  $i_{qref}$ . Measure the response time  $t_r$  (time from 0% to 95% of the step change magnitude). Is  $t_r$  consistent with your tuning in Section I?
- 5) Reconsider your current controller design with  $t_r = 5 \text{ ms}$  in Section I.Q1. Then go back to steps 3 and 4. Comment on the obtained results.
- 6) To consider parameter uncertainties and errors in the tuning (I.Q1) and the decoupling control, assume a stator resistance  $\hat{R}_s = 1.5 \times R_s$ ,  $\hat{L}_d = 2 \times L_d$  and  $\hat{L}_q = 0.5 \times L_q$ . Parameters with “^” are the nominal (known) parameters and used in the tuning of the controller. Then, calculate the controllers’ gains in Section I.Q1 and run steps 4 and 5. Save your results and comment on them.

## III. Discrete-time control

The current control is mostly implemented in a digital control board. Therefore, the block “Control” in Simulink must be triggered in a regular way at the sampling frequency. In this study,

the latter is set to  $f_s = 10 \text{ kHz}$ . Furthermore, the inherent delay of the discrete-time control introduces a coupling between  $d$ -axis and  $q$ -axis current control loops. This effect degrades the control performance and may lead to instability.

To realize the discrete-time controller:

- 7) A “Trigger” () must be added into the block “Control” and a “pulse generator” () must be integrated to the model, it provides rising (or falling) edge to the “Trigger”. The frequency of the pulse signal is equal to the sampling frequency ( $f_s$ ), i.e. its period should be  $T_s = 1/f_s$ . The pulse generator input time (t) comes from the clock ()
- 8) All continuous-time blocks in the control block, including integrators, if any, must be replaced by their discrete-time equivalents.
- 9) An angle shift must be implemented in the control to minimize the coupling effect due to the discrete-time control. Its role is to advance the rotor position angle of the voltage vector in such a way that the control voltage vector dq-components are well aligned with actual rotor dq-axes. Set the angle shift to  $\omega \cdot T_s/2$ , with  $T_s = 1/f_s$ .
- 10) Set  $T_s = 1/10,000$  in the MATLAB script to fix  $f_s = 10 \text{ kHz}$ .

Once the model is ready, run the following test:

- 11) Run the discrete-time controller under the same condition as with the continuous-time controller in Section II with no parameter error and  $f_{BW} = 500 \text{ Hz}$ . Watch the current control performance in terms of set-point tracking, response time, overshoot, and decoupling control.