Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

Sondre Aasen

Control of Permanent Magnet Synchronous Motors for drones

Master's thesis in Engineering Cybernetics
Supervisor: Kristoffer Gryte
Co-supervisor: Jon Are Suul
June 2022

NTNU
Norwegian University of
Science and Technology

Sondre Aasen

Control of Permanent Magnet Synchronous Motors for drones

Master's thesis in Engineering Cybernetics
Supervisor: Kristoffer Gryte
Co-supervisor: Jon Are Suul
June 2022

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Engineering Cybernetics

NTNU
Norwegian University of
Science and Technology

Preface

This master's thesis is part of the fulfillment of the two-year Master of Science (M.Sc.) in Engineering Cybernetics at The Norwegian University of Science and Technology (NTNU) in Trondheim, Norway. The thesis is based on the work conducted in the subject TTK4551 Engineering Cybernetics, Specialization Project, regarding control of Permanent Magnet Synchronous Motors [1]. The work related to the master's thesis was carried out during the spring semester of 2022 in cooperation with Alva Industries, and was one of several project ideas suggested by the supervisors at NTNU. The thesis is written for readers with basic knowledge of electrical- and control systems.

Trondheim, 2022-06-08

Sondre Aasen

Acknowledgment

I would like to thank my family and girlfriend for their continued support. I am also grateful for all the help I have received from my supervisors Kristoffer Gryte and Jon Are Suul. Finally, I would like to thank Alva Industries for supplying important information needed in the thesis.

S.A.

Executive Summary

The research on fixed-wing vertical take-off and landing drones (VTOL) is growing rapidly, with an increasing focus on agile control systems and cost-reducing measures. This thesis aims to contribute to this research by presenting various sensorless control schemes that makes it possible to lock the propellers of a fixed-wing drone in order to prevent them from being damaged during takeoff and landings and to achieve better aerodynamic properties during flight to reduce energy expenditure.

With the aim of developing a sensorless control system to lock the drone propellers into a fixed position, a simulation environment was created using MATLAB's Simulink where a permanent magnet synchronous motor designed by Alva Industries could be simulated. By using a control system that was first developed in the specialization project, various estimation methods were implemented in the simulation environment to make the sensorless control system meet the requirements defined in the thesis. The implemented estimation methods were based on nonlinear observers, high frequency current injections, stator mounted Hall sensors and Kalman filters which was used as feedback to the control system in an effort to replace the need of encoders and in that way contribute to reduction of system costs.

After simulating and optimizing the control system using different estimation methods, it was found that only two of them gave satisfactory results. These were the ones based on the nonlinear observers and the Extended Kalman filter coupled with Hall sensors. By using the nonlinear observers as feedback, which performs poorly at low speeds, the propeller was magnetically locked into position by applying a fixed voltage to the stator windings. This method has its limitations as it is based on open loop control and therefore cannot guarantee that the propeller is locked into the desired position. On the other hand, the method involving the Extended Kalman filter coupled with Hall sensors is based on closed loop control and can to a greater extent ensure that the propeller is locked into the desired position. In addition, a commonly used method for estimating the rotor position at low speeds using high frequency current injections was found unsuitable for the motor designed by Alva due to small motor inductances.

In conclusion, two different methods were found that can lock the propeller into a fixed position without the use of an encoder. Nevertheless, only one of them can guarantee that the propeller is locked and remains locked in the correct position. For further research, it is proposed to verify the simulated results practically and conduct an energy analysis of the different control methods.

Contents

List of Tables

List of Figures

Acronyms

AC Alternating current

BLDC Brushless direct current motor

DC Direct current

DTC Direct torque control

EKF Extended Kalman filter

EMF Electromotive force

FOC Field-oriented control

HFI High frequency injection

IPMSM Interior mounted permanent magnet synchronous motor

KF Kalman filter

LTI Linear time invariant

PID Proportional-integral-derivative controller

PLL Phase locked loop

PMSM Permanent magnet synchronous motor

SNR Signal to noise ratio

SPMSM Surface mounted permanent magnet synchronous motor

SPWM Sinusoidal pulse width modulation

SRF Synchronous rotation frame

SVM Space vector modulation

PWM Pulse width modulation

UKF Unscented Kalman filter

VSI Voltage source inverter

VTOL Vertical take-off and landing

Chapter 1

Introduction

1.1 Background and motivation

Over the last decade, the use of drones has increased significantly in both industrial and private use. In order to utilize drones to the fullest, the selection of motors and control systems are of great importance. *Permanent magnet synchronous motors* (PMSM) is a widely used option for driving the propellers in drones due to their high efficiency and torque-to-weight ratio, and this thesis will explore the possibilities of simulating and controlling such a motor while driving a propeller.

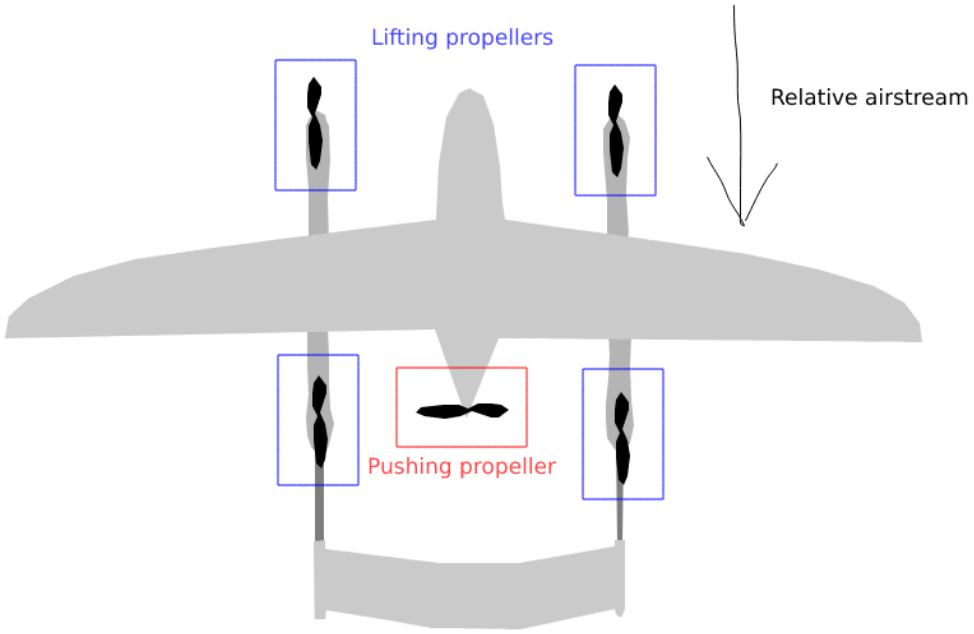


Figure 1.1: Illustration of a fixed-wing drone during horizontal flight operations

In addition to achieving precise motor control, it is also desired that the propellers not in use are able to lock themselves into fixed positions in order to improve the aerodynamics of the drone so that it runs more efficiently and in that way extend the range of the drone. Such a scenario is illustrated in figure 1.1 which shows a fixed-wing drone during horizontal flight where only the pushing propeller is being used. It can be seen that the drone will have the least air resistance when the lifting propellers are aligned in the same direction as the relative air stream and it is therefore desired to lock the lifting propellers in this orientation. Additionally, such a locking mechanism can be used as an equipment protective measure for the pushing propeller during take-off and landings in order to prevent the propeller from hitting any surroundings.

Usually when performing motor control for PMSMs, it is common to use an encoder to measure the speed and position of the rotor/propeller. However, the use of such measuring instruments increases system costs and may not always be available for use due to limited space and/or harsh operating environments. As a result of this, the thesis will also explore the possibilities of performing motor control using estimation methods such as nonlinear observers, high frequency current injections, extended Kalman filter and Hall sensors in an effort to replace the need of an encoder.

1.2 Literature study

In this section it is done a brief study of existing estimation methods of the rotor shaft-position, while also discussing the suitability of these methods in relation to the thesis. This study, along with the experience from the preliminary project thesis [1], will form the basis for the work that has been carried out over the semester.

Firstly, the article *Position Sensorless Permanent Magnet Synchronous Machine Drives—A Review* [24] was reviewed in order to get an overview of the different sensorless control schemes that have been developed over the years. From this article it was found that model-based estimators using the principle of back electromotive force work well at medium to high speeds, where the article by *Ortega et.al* [14] proposed a suitable model-based observer for our system. However, such estimators depend on the rotational speed of the rotor and will therefore work poorly for position control which will be used to lock the propellers of the drone. As a result, other methods involving high frequency current injections and Kalman filters were explored in an attempt to find estimators that can operate at low speeds. From this, a high frequency current injection-based method proposed by *Qiao et.al* [25] was found suitable for the surface-mounted PMSM simulated in this thesis. Additionally, the articles [7] and [12] proposed methods of implementing a Kalman filter which could be used to estimate the states of our system.

1.3 Objectives

The main objectives of this thesis are:

1. Acquire a fully functional model of a PMSM in Simulink
2. Develop a robust control system that allows for both velocity- and position control of the PMSM while using encoder feedback
3. Implement sensorless control where the encoder is replaced by estimators of the rotor-shaft speed and position without comprising the performance of the control system
4. Use the sensorless control system to lock the drone propeller in a fixed position when not being operated

1.4 Approach

The scientific approach in this thesis is to develop a simulation environment that allows for experimentation using different control methods and motor drives. In coordination with Alva Industries, it was desired to design the simulation environment for the PMSM as closely as possible to one of Alva Industries' own motors. This was done by using specified motor parameters supplied by Alva Industries together with knowledge of the load characteristics of a propeller.

1.5 Contributions

The main contribution of this thesis is to develop a simulation environment for control of PMSMs such that advanced testing can be simulated before it is performed practically. Additionally, the thesis contributes to the research of controlling drones as it explores various control methods that can have a cost-reducing effect without compromising performance. Finally, the thesis contributes to the research dedicated to extending the flight time of drones.

Chapter 2

Theoretical background

Chapter 2 introduces the different components of the system and describes key concepts needed to understand the work that has been carried out in this thesis.

2.1 Linear state-space representation

The dynamics of a linear and time-invariant system are described in state-space representation as:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \tag{2.1a}$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) \tag{2.1b}$$

where $\mathbf{x}(t)\ \epsilon\ \mathbb{R}^n$ are the states, $\mathbf{u}(t)\ \epsilon\ \mathbb{R}^p$ are the control inputs and $\mathbf{y}(t)\ \epsilon\ \mathbb{R}^q$ are the outputs of the model. The state matrix $\mathbf{A}\ \epsilon\ \mathbb{R}^{nxn}$, input matrix $\mathbf{B}\ \epsilon\ \mathbb{R}^{nxp}$ and output matrix $\mathbf{C}\ \epsilon\ \mathbb{R}^{qxn}$ are constant matrices [4].

2.2 Nonlinear state-space representation

The dynamics of a state-space model of a nonlinear system are described by:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}, \mathbf{u}) \tag{2.2a}$$

$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}, \mathbf{u}) \tag{2.2b}$$

where $\mathbf{f}\ \epsilon\ \mathbb{R}^n$ and $\mathbf{g}\ \epsilon\ \mathbb{R}^l$ are vector functions describing the nonlinear model [4].

2.3 Discretization of continuous LTI-systems

2.3.1 Exact discretization

In order to numerically analyze continuous LTI-state space systems, it is necessary to transform the system from a continuous-time state variable description into a discrete-time state variable description.

The following continuous-time LTI-state space system from section 2.1:

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \tag{2.3a}$$

$$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) \tag{2.3b}$$

can be discretized, assuming zero-order hold for the input $\mathbf{u}$, into:

$$\mathbf{x}[k+1] = \mathbf{A_d}\mathbf{x}[k] + \mathbf{B_d}\mathbf{u}[k] \tag{2.4a}$$

$$\mathbf{y}[k] = \mathbf{C_d}\mathbf{x}[k] \tag{2.4b}$$

where,

$$\mathbf{A}_d = e^{\mathbf{A}T} \tag{2.5a}$$

$$\mathbf{B}_d = \left(\int_{\tau=0}^{T} e^{\mathbf{A}\tau}\, d\tau\right)\mathbf{B} \tag{2.5b}$$

$$\mathbf{C}_d = \mathbf{C} \tag{2.5c}$$

and T is the fundamental sample time of the discretization [4].

2.3.2 Euler discretization

Exact discretization may sometimes be difficult to implement due to the heavy matrix exponential and integral operations involved. It is easier to calculate an approximate discrete model, which can be accurate for small enough time steps T. An example of such an approximate discrete model is found by using Euler's forward method:

$$\dot{\mathbf{x}} = \frac{\mathbf{x}[k+1] - \mathbf{x}[k]}{T} \tag{2.6}$$

Hence, the continuous-time state equation is approximately equal to:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x}[k] + \mathbf{B}\mathbf{u}[k] \tag{2.7}$$

such that the discretized system can be expressed as:

$$\mathbf{x}[k+1] = (\mathbf{I} + \mathbf{A}T)\mathbf{x}[k] + T\mathbf{B}\mathbf{u}[k] \tag{2.8}$$

This is known as the discretized system using *Euler discretization*. Expressed in the discrete form from equation 2.4, the system matrices is now defined as [4]:

$$\mathbf{A}_d = (\mathbf{I} + \mathbf{A}T) \tag{2.9a}$$

$$\mathbf{B}_d = T\mathbf{B} \tag{2.9b}$$

$$\mathbf{C}_d = \mathbf{C} \tag{2.9c}$$

2.4 The Kalman Filter

The *Kalman filter* is an algorithm that uses measurements observed over time, including random noise and other inaccuracies, and produces estimates of unknown variables that tend to be more accurate than those based on a single measurement alone. Kalman filters have been demonstrating its usefulness in various applications over the years due to its relatively simple form and small computational cost, and remains as one of the most powerful state estimation algorithms in present-day. The original Kalman filter was designed for linear dynamical systems where the process and measurement noise is assumed to be white, but several extensions of the filter has been developed since such as the *Extended Kalman filter* (EKF) and the *Unscented Kalman filter* (UKF) which are designed to work on nonlinear systems [12].

2.4.1 Problem definition

The Kalman filter is a recursive estimator, meaning that only the estimated state from the previous time step and the current measurement are needed to compute the estimate for the current state. The estimated states of the discrete-time Kalman filter is based on linear dynamical systems in state space format, and the process model defines the evolution of the states from time t_{k-1} to time t_k as:

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_k + \mathbf{w}_k \tag{2.10}$$

where $\mathbf{F}$ is the state transition matrix applied to the previous state vector $\mathbf{x}_{k-1}$, $\mathbf{B}$ is the control-input matrix applied to the control vector $\mathbf{u}_k$, and $\mathbf{w}_k$ is the process noise vector that is assumed to be zero-mean Gaussian with the covariance $\mathbf{Q}$, i.e., $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0},\ \mathbf{Q})$.

The process model is paired with the measurement model that describes the relationship between the states and the measurements at the current time step t_k as:

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k \tag{2.11}$$

where $\mathbf{z}_k$ is the measurement vector, $\mathbf{H}$ is the measurement matrix and $\mathbf{v}_k$ is the measurement noise vector that is assumed to be zero-mean Gaussian with the covariance $\mathbf{R}$, i.e., $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0},\ \mathbf{R})$.

The role of the Kalman filter is to provide an estimate of $\mathbf{x}_k$ at time t_k, given the initial estimate of $\mathbf{x}_0$, the series of measurement, $\mathbf{z_1}, \mathbf{z_2}, \ldots, \mathbf{z_k}$, and the information of the system described by $\mathbf{F}$, $\mathbf{B}$, $\mathbf{H}$, $\mathbf{Q}$ and $\mathbf{R}$. Although the covariance matrices are supposed to reflect the statistics of the noises, the true statistics of the noises is not known or not Gaussian in many practical applications. Therefore, $\mathbf{Q}$ and $\mathbf{R}$ are usually used as tuning parameters in order to achieve the desired performance of the filter [12].

2.4.2 Kalman Filter Algorithm

The Kalman filter consists of two stages: *prediction* and *update*. The algorithm of the Kalman filter is summarized as follows [12]:

Prediction step:

$$\hat{\mathbf{x}}_k^- = \mathbf{F}\hat{\mathbf{x}}_{\mathbf{k-1}}^+ + \mathbf{B}\mathbf{u}_k \tag{2.12a}$$

$$\mathbf{P}_\mathbf{k}^- = \mathbf{F}\mathbf{P}_{\mathbf{k-1}}^+\mathbf{F}^\mathbf{T} + \mathbf{Q} \tag{2.12b}$$

Update step:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_k^- \tag{2.13a}$$

$$\mathbf{K}_k = \mathbf{P_k}^-\mathbf{H}^\mathbf{T}(\mathbf{R} + \mathbf{H}\mathbf{P}_\mathbf{k}^-\mathbf{H}^\mathbf{T})^{-1} \tag{2.13b}$$

$$\hat{\mathbf{x}}_\mathbf{k}^+ = \hat{\mathbf{x}}_\mathbf{k}^- + \mathbf{K_k}\tilde{\mathbf{y}}_\mathbf{k} \tag{2.13c}$$

$$\mathbf{P}_\mathbf{k}^+ = (\mathbf{I} - \mathbf{K_k}\mathbf{H})\mathbf{P}_\mathbf{k}^- \tag{2.13d}$$

In the above equations, the hat operator means an estimate of a variable e.g. $\hat{\mathbf{x}}$ is an estimate of $\mathbf{x}$. The superscripts – and + denote predicted (prior) and updated (posterior) estimates, respectively. The predicted state estimate is evolved from the updated previous state estimate through the state-space model. The new term $\mathbf{P_k}$ is called the *state error covariance*-matrix and it estimates the uncertainty associated with the estimated state. The state covariance matrix consists of the variances associated with each of the state estimates and the correlation between the errors in the state estimates. One can observe that the error covariance becomes larger in the prediction step due to the summation of $\mathbf{Q}$. In practical terms, this means that the Kalman filter becomes more uncertain of the state estimate after the prediction step because of the process noise in the system.

In the update stage, the measurement residual $\tilde{\mathbf{y}}_\mathbf{k}$ is calculated first. The measurement residual is the difference between the true measurement $\mathbf{z_k}$ and the estimated measurement $\mathbf{H}\hat{\mathbf{x}}_\mathbf{k}^-$. The residual $\tilde{\mathbf{y}}_\mathbf{k}$ is then multiplied by the Kalman gain $\mathbf{K_k}$ to provide a correction to the predicted estimate. After the updated state estimate is obtained, the Kalman filter calculates the updated error covariance $\mathbf{P}_\mathbf{k}^+$ which will be used in the next time step. One can see that the error covariance becomes smaller in the update step, meaning that the filter is more certain of the state estimate after measurements have been utilized.

It is important to note that the Kalman filter is derived based on the assumption that the process and measurement models are linear, meaning that they can be expressed with the matrices $\mathbf{F}$, $\mathbf{B}$, and $\mathbf{H}$, and the process and measurement noise are Gaussian. Hence, a Kalman filter only provides optimal estimate if the assumptions are satisfied [12].

2.5 The Extended Kalman Filter

Suppose you have a nonlinear dynamic system where you are not able to define either the process model or measurement model using state-space representation as in 2.10 and 2.11. From the last section it is known that the original Kalman filter is designed for linear systems, meaning that a nonlinear system has to be linearized around an operating point before a Kalman filter can be applied. These linearizations become increasingly inaccurate once the system leaves the linearization point, ultimately rendering the Kalman filter dysfunctional as the prediction step will contribute to errors because of an inaccurate system model. The Extended Kalman filter provides us an efficient way of dealing with such nonlinear systems [12].

The extended Kalman filter can be viewed as a nonlinear version of the original Kalman filter that linearizes the nonlinear model about a current estimate. Suppose we have the following models for state transition and measurement:

$$\mathbf{x_k} = \mathbf{f}(\mathbf{x_{k-1}}, \mathbf{u_k}) + \mathbf{w_k} \tag{2.14}$$

$$\mathbf{z_k} = \mathbf{h}(\mathbf{x_{k-1}}) + \mathbf{v_k} \tag{2.15}$$

where $\mathbf{f}$ is the function of the previous state $\mathbf{x_{k-1}}$ and the control input $\mathbf{u_k}$ that provides the current state $\mathbf{x_k}$. $\mathbf{h}$ is the measurement function that relates the current state $\mathbf{x_k}$ to the measurement $\mathbf{z_k}$. $\mathbf{w_k}$ and $\mathbf{v_k}$ are Gaussian noises for the process model and the measurement model with covariance $\mathbf{Q}$ and $\mathbf{R}$, respectively.

Extended Kalman Filter Algoritm

In order to linearize the nonlinear model it is necessary to calculate the *Jacobian* matrix, a first-order partial derivative of a vector function with respect to a vector, of the nonlinear model in each time step as:

$$\mathbf{F}_{k-1} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\bigg|_{\hat{\mathbf{x}}^{+}_{\mathbf{k-1}}, \mathbf{u_k}} \tag{2.16}$$

$$\mathbf{H}_k = \frac{\partial \mathbf{h}}{\partial \mathbf{x}}\bigg|_{\hat{\mathbf{x}}^{-}_{\mathbf{k}}} \tag{2.17}$$

In doing so, the Jacobian matrix evaluated at the current estimate becomes the state transition matrix in the Extended Kalman filter. After the state transition matrix is defined, the Extended Kalman filter algorithm becomes very similar to that of the original Kalman filter.

Prediction step:

$$\hat{\mathbf{x}}_k^- = \mathbf{f}(\tilde{\mathbf{x}}_{\mathbf{k-1}}^+, \mathbf{u_k}) \tag{2.18a}$$

$$\mathbf{P_k^-} = \mathbf{F_{k-1}} \mathbf{P_{k-1}^+} \mathbf{F_{k-1}^T} + \mathbf{Q} \tag{2.18b}$$

Update step:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_k^-) \tag{2.19a}$$

$$\mathbf{K}_k = \mathbf{P_k}^- \mathbf{H_k^T} (\mathbf{R} + \mathbf{H_k} \mathbf{P_k^-} \mathbf{H_k^T})^{-1} \tag{2.19b}$$

$$\hat{\mathbf{x}}_\mathbf{k}^+ = \hat{\mathbf{x}}_\mathbf{k}^- + \mathbf{K_k} \tilde{\mathbf{y}}_\mathbf{k} \tag{2.19c}$$

$$\mathbf{P_k^+} = (\mathbf{I} - \mathbf{K_k} \mathbf{H_k}) \mathbf{P_k^-} \tag{2.19d}$$

The main difference from the Kalman filter is that the Extended Kalman filter obtains the predicted state estimate and predicted measurement by the nonlinear functions $\mathbf{f}(\mathbf{x_{k-1}}, \mathbf{u_k})$ and $\mathbf{h}(\mathbf{x_k})$, respectively. Additionally, the state transition matrix $\mathbf{F}_{k-1}$ is evaluated at every time step.

For good EKF performance the choice of the covariance matrices $\mathbf{Q}$, $\mathbf{R}$ and $\mathbf{P}_0$ is crucial. Covariance matrices give the statistical description of the model inaccuracy. Matrix $\mathbf{Q}$ represents the statistical description of the model, matrix $\mathbf{R}$ indicates the magnitude of measurement noise, matrix $\mathbf{P}_0$ contains the information of variances at the initial conditions and mainly affects the convergence rate of the EKF in the transient condition. Since these are usually unknown, in most cases the EKF matrices are designed and tuned by trial-and-error procedures [7].

2.6 Permanent Magnet Synchronous Motor

A *Permanent Magnet Synchronous Motor* is a brushless alternating current (AC) synchronous motor whose field excitation is provided by permanent magnets, and has a sinusoidal *back electromotive force* (EMF). PMSMs are typically used in applications requiring high-performance motor drives due to their high efficiency and torque-to-weight ratio, and is therefore suitable for driving the propellers of a fixed-wing drone. Given that the rotor is always magnetized due to the permanent magnets, it is only needed to generate a rotating magnetic field in the stator to be able to set the rotor in motion. This is achieved by applying a three phase voltage to the three phase winded stator, where each phase is shifted 120 degrees from another. By applying different voltages to each winding over time, it is possible to induce a rotating magnetic field in which the permanent magnetic poles of the rotor locks onto, ultimately producing torque and making the rotor rotate at the synchronous speed of the stator magnetic field [9].

PMSMs comes in different variations depending on how they are constructed. The two main variants are divided into: *surface-mounted PMSM* (SPMSM) and *interior-mounted PMSM* (IPMSM).

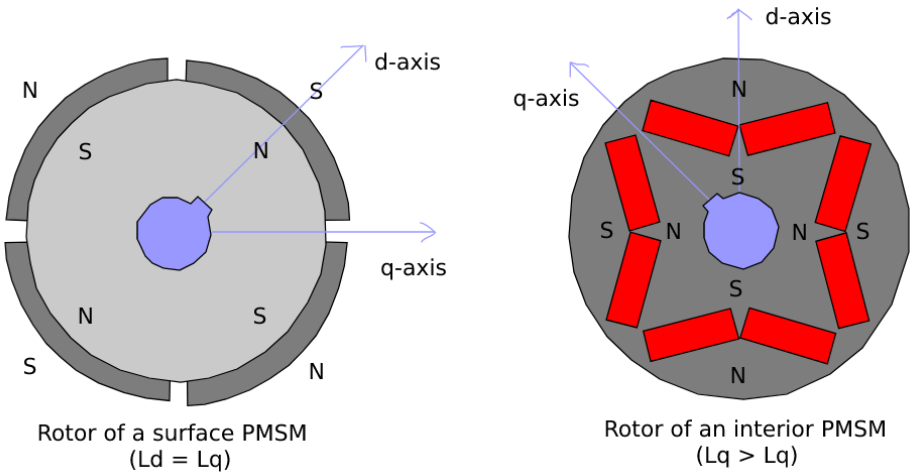


Figure 2.1: Cross-section of PMSM variations with inspiration from D. Ocen [19]

The difference between the two variations are related to how the permanent magnets are fixed to the rotor. SPMSMs have magnets mounted on the surface of the rotor, while IPMSMs have magnets embedded into the rotor as seen in figure 2.1. As a result of this, SPMSMs have a uniform air gap flux density making the motor non-salient, meaning that the motor inductances do not change depending on the rotor position. IPMSMs does not have a uniform air gap flux, eventually making the motor salient and giving rise to phenomena such as position dependent motor inductances and reluctance torque [9]. The motor which is simulated in this thesis is a SPMSM.

Power Circuit

The power circuit of the PMSM in this thesis consists of a battery used as power source and a three phased inverter that transforms the supply voltage from DC into AC.

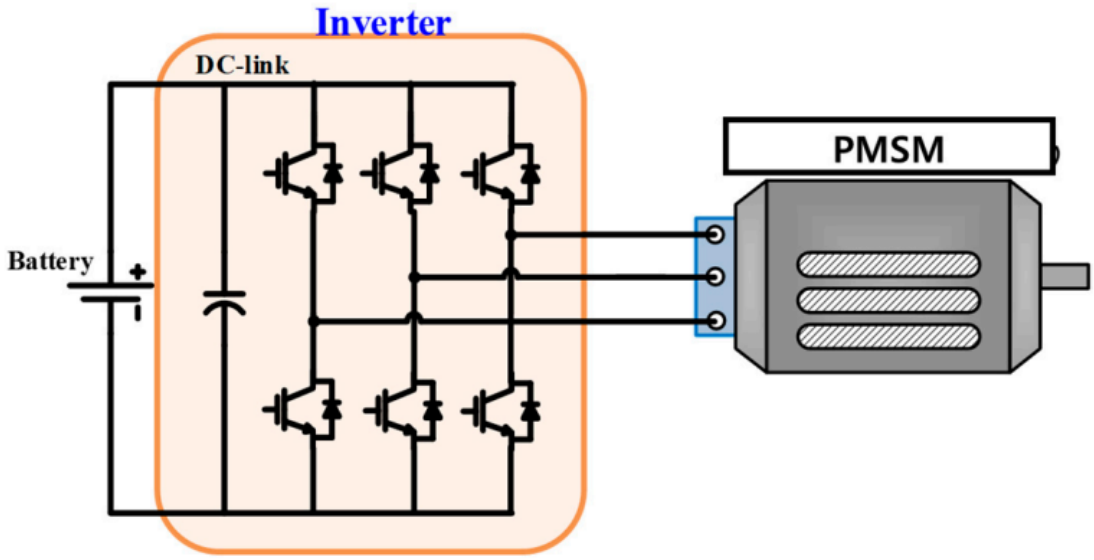


Figure 2.2: Illustration of a power circuit used in drones with permission from MDPI [13]

The three phase *Voltage Source Inverter* (VSI) used in this thesis consists of six transistors placed in a two-level topology as showed in figure 2.2 - with the goal of producing a three phased sinusoidal voltage where the frequency and amplitude is adjustable.

The inverter uses *Space Vector Modulation* (SVM) to emulate the three phase sinusoidal voltage. SVM is chosen over *sinusoidal pulse width modulation* (SPWM) due to its more efficient use of the DC-link voltage available from the battery and less harmonic distortion [11]. The sinusoidal voltages the inverter is meant to emulate are supplied from the control system shown in figure 2.4, where SVM is used in the PWM generator in order to determine the duty cycles of each transistor in the inverter.

2.6.1 Modelling PMSMs in Stationary *abc*-Reference Frame

The dynamics of PMSMs in the *abc*-reference frame can be found by inspecting the stator windings of the motor. Throughout the derivation of the mathematical model, the following assumptions are made: the three-phase system is balanced (sum of voltages and currents in the star point are always zero), magnetic saturation is neglected, the induced EMF is sinusoidal and eddy currents (currents due to change in magnetic fields) and hysteresis losses (losses due to reversal of magnetism in a ferromagnetic material) are negligible [10].

By using Kirchoff's circuit laws, it is possible to express the voltage balances of the stator by the following equations:

$$v_a = R_s i_a + \frac{d}{dt}\psi_a \tag{2.20a}$$

$$v_b = R_s i_b + \frac{d}{dt}\psi_b \tag{2.20b}$$

$$v_c = R_s i_c + \frac{d}{dt}\psi_c \tag{2.20c}$$

where $\mathbf{v_{abc}}$ is the voltage across each winding, $\mathbf{i_{abc}}$ the current in each winding and $\frac{d}{dt}\psi_{\mathbf{abc}}$ is the rate of change for the magnetic flux in each stator winding. In vector form this can be expressed as:

$$\mathbf{v_{abc}} = \mathbf{R_s i_{abc}} + \frac{d}{dt}\Psi_{\mathbf{abc}} \tag{2.21}$$

The three stator windings and the permanent magnet both contribute to the total flux linkage in each winding by:

$$\Psi_{\mathbf{abc}} = \mathbf{L_s i_{abc}} + \lambda_{\mathbf{s}} \tag{2.22}$$

where $\mathbf{L_s}$ is the self and mutual inductances of the stator and $\lambda_{\mathbf{s}}$ is the flux through the stator windings due to the permanent magnet. The permanent magnet fluxes $\lambda_{\mathbf{s}}$ linking the stator windings are defined as:

$$\lambda_{\mathbf{s}} = \begin{bmatrix} \lambda_m \cos(\theta_e) \\ \lambda_m \cos(\theta_e - 2\pi/3) \\ \lambda_m \cos(\theta_e + 2\pi/3) \end{bmatrix} \tag{2.23}$$

where λ_m is the permanent flux linkage and θ_e is the angle of the magnetic axis of the rotor in reference to the α-axis. The inductance matrix, $\mathbf{L_s}$, is expressed as:

$$\mathbf{L_s} = \begin{bmatrix} L_{aa} & L_{ab} & L_{ac} \\ L_{ba} & L_{bb} & L_{bc} \\ L_{ca} & L_{cb} & L_{cc} \end{bmatrix} \tag{2.24}$$

$L_{aa} = L_s + L_m \cos(2\theta_e)$
$L_{bb} = L_s + L_m \cos(2(\theta_e - 2\pi/3))$
$L_{cc} = L_s + L_m \cos(2(\theta_e + 2\pi/3))$
$L_{ab} = L_{ba} = -M_s - L_m \cos(2(\theta_e + \pi/6))$
$L_{bc} = L_{cb} = -M_s - L_m \cos(2(\theta_e + \pi/6 - 2\pi/3))$
$L_{ac} = L_{ca} = -M_s - L_m \cos(2(\theta_e + \pi/6 + 2\pi/3))$

where θ_e is the electrical rotor position, L_s is the stator self-inductance per phase, L_m is the stator inductance fluctation and M_s is the stator average mutual inductance [16].

Electro-mechanical relation

The mathematical model of the system can be completed by the electro-mechanical relation expressed by the torque equilibrium:

$$J\frac{d}{dt}\omega_m = \tau_m - B\omega_m - \tau_L \tag{2.25}$$

where J is the load inertia, ω_m is the rotor speed, τ_m is the motor generated torque, B is the viscous coefficient of the load and τ_L is the load torque.

The relation between the electrical angle of the rotor and the rotor speed is defined as:

$$\frac{d}{dt}\theta_e = N\omega_m \tag{2.26}$$

where θ_e is the electrical angle of the rotor, ω_m is the rotor speed and N is the number of pole pairs in the PMSM.

2.6.2 Modelling PMSMs in Synchronous *dq*-Reference Frame

So far it has been established a set of equations which describe the dynamics of PMSMs. However, these equations depend on the rotor position and makes the system quite involved to work with. If the system is transformed into a synchronous reference frame rotating with the magnetic axis of the rotor, all terms that involve the time-dependent rotor position (θ_e) will disappear. The transformation which enables this is called the *Clarke-Park Transformation* [18].

$$\mathbf{S_{dq0}} = \mathbf{TS_{abc}} \tag{2.27}$$

The Clarke-Park transform **T** is used to transform voltage, current, etc. from a stationary abc-reference frame into a rotating dq-reference frame. Here, **S** represents voltage, current, etc. in each of the respective reference frames. The *direct-quadrature*-reference frame is fixed to θ_e such that the d-axis is always aligned with the north pole of the rotor permanent magnet, while the q-axis remains orthogonal to the d-axis. The Clarke-Park transformation itself is defined as:

$$\mathbf{T} = \frac{2}{3}\begin{bmatrix} \cos(\theta_e) & \cos(\theta_e - 2\pi/3) & \cos(\theta_e + 2\pi/3) \\ -\sin(\theta_e) & -\sin(\theta_e - 2\pi/3) & -\sin(\theta_e + 2\pi/3) \\ 1/2 & 1/2 & 1/2 \end{bmatrix} \tag{2.28}$$

where θ_e is the angle of the magnetic axis relative to the a-axis of the stator. By applying the Clarke-Park transformation to the equations from 2.20, the dynamic model of the motor is expressed in the dq-reference frame. The set of equations then becomes:

$$v_d = R_s i_d + L_d \frac{d}{dt} i_d - N\omega_m i_q L_q \tag{2.29a}$$

$$v_q = R_s i_q + L_q \frac{d}{dt} i_q + N\omega_m (i_d L_d + \lambda_m) \tag{2.29b}$$

$$\tau_m = \frac{3}{2} N(i_q \lambda_m + i_d i_q (L_d - L_q)) \tag{2.29c}$$

$$J\frac{d}{dt}\omega_m = \tau_m - B\omega_m - \tau_L \tag{2.30}$$

where,

$L_d = L_s + M_s + 3/2L_m$ is the stator d-axis inductance.
$L_q = L_s + M_s - 3/2L_m$ is the stator q-axis inductance.
ω_m is the mechanical rotational speed.
N is the number of rotor permanent magnet pole pairs.
τ_m is the rotor torque.
λ_m is the permanent flux linkage.

Lastly, it is desired to transform the system into a nonlinear state-space representation as described in section 2.2:

$$\frac{d}{dt}i_d = -\frac{R_s}{L_d}i_d + \frac{L_q}{L_d}N\omega_m i_q + \frac{1}{L_d}v_d \tag{2.31a}$$

$$\frac{d}{dt}i_q = -\frac{R_s}{L_q}i_q - \frac{L_d}{L_q}N\omega_m i_d - \frac{\lambda_m N\omega_r}{L_q} + \frac{1}{L_q}v_q \tag{2.31b}$$

$$\frac{d}{dt}\omega_m = \frac{1}{J}\left(\frac{3}{2}N\left(i_q\lambda_m + i_d i_q(L_d - L_q)\right) - D_m\omega_m - T_L\right) \tag{2.31c}$$

$$\frac{d}{dt}\theta_e = N\omega_m \tag{2.31d}$$

where θ_e is the rotor electrical angle, $\mathbf{x}(t) = \begin{bmatrix} i_d & i_q & \omega_m & \theta_e \end{bmatrix}^T$ is the state vector and $\mathbf{u}(t) = \begin{bmatrix} v_d & v_q \end{bmatrix}^T$ is the input vector.

2.7 Field-Oriented Control

Field-Oriented Control (FOC), also known as vector control, is a technique used to control PM motors and AC induction motors. Field-oriented control remains today as one of the most commonly used control methods of PM motors due to its efficiency and smooth operations across the wide speed range, which is why it will be used to control the PMSM in this thesis. Other methods that also could have been used in order to control the PMSM include *direct torque control* (DTC) or *voltage-frequency scalar control* (V/f). These control methods can be easy to implement, but have the disadvantages of large torque ripples and poor dynamic performance, respectively [8].

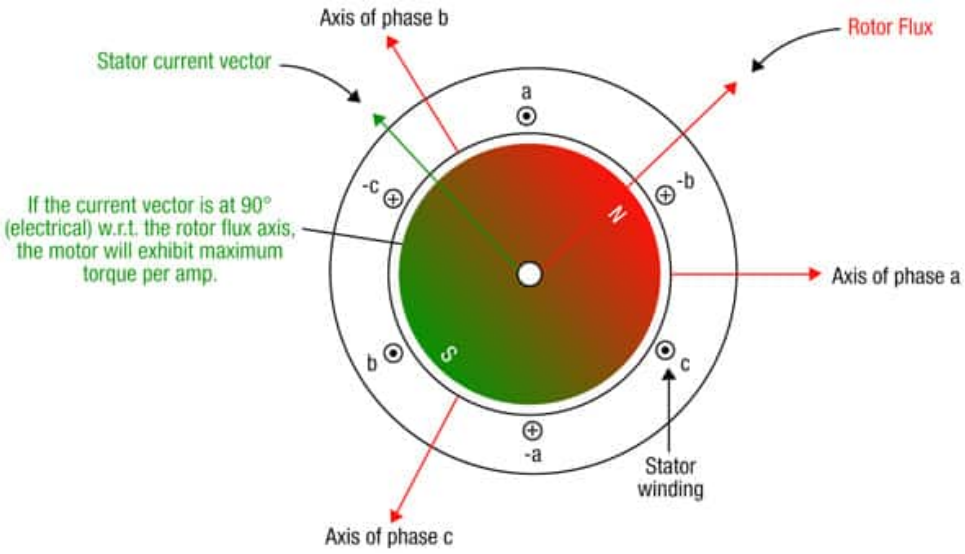


Figure 2.3: Cross section of a PMSM during FOC with permission from Texas Instruments [6]

In simple terms, field-oriented control is a motor control technique where the system is trying to orient the stator flux vector to a specific angle relative to the rotor flux vector, as shown in figure 2.3. The optimal angle of orientation depends upon what characteristic of the motor needs to be maximized. The most common use of field-oriented control is to maximize the motor's torque per amp. This is achieved when the stator flux vector is 90 degrees shifted away from the rotor flux vector. As the stator flux vector is determined by the motor currents, one can simply control the torque of the motor by controlling the motor currents [6].

In order to make use of field-oriented control, it is necessary to know the position of the magnetic axis of the rotor in order to know which direction the stator flux vector should be pointed. This can be found in different ways, two of which will be explored in this thesis:

- Using an encoder/Hall sensors to measure the position of the rotor
- Using an observer in order to estimate the position of the rotor

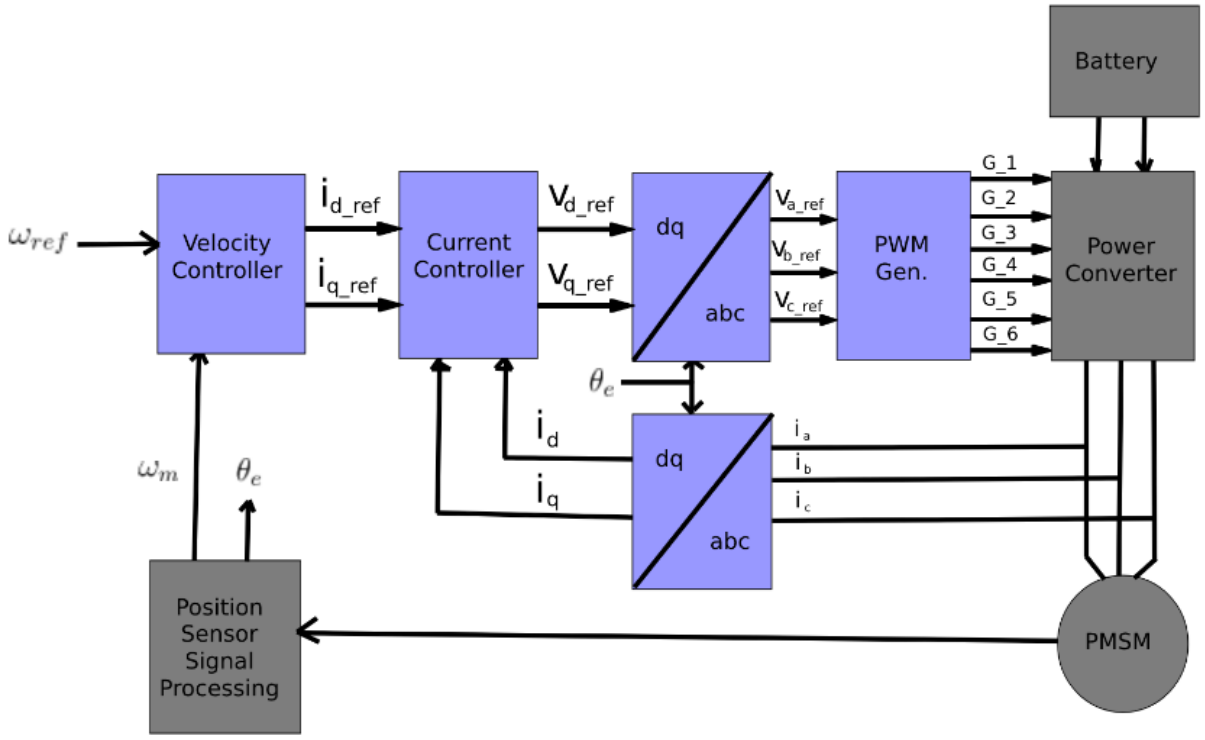


Figure 2.4: Block diagram of a FOC system with inspiration from MathWorks [15]

Figure 2.4 gives an insight to what a field-oriented control system might look like. The control system (blue) consists mainly of three parts: signal processing, the control loop and the PWM-generator. As seen in the figure above, there are two controllers in the control loop: the *current controller* and *the speed controller*. The current controller is considered as the *inner control loop* and uses two PI-controllers to control both motor torque i_q and motor flux i_d. The speed controller is considered as the *outer control loop* and uses a PI-controller to control the motor speed.

The physical system (grey) consists of a battery which supplies a three phase power inverter with DC voltage where the inverter transform the voltage from DC to AC. The inverter is controlled by the control system which regulates the power flow to the PMSM. The measurements that are taken from the physical system and used in the control system are: the motor currents, motor voltage and rotor position (only when encoder feedback is being used).

Signal processing and dq-transformations

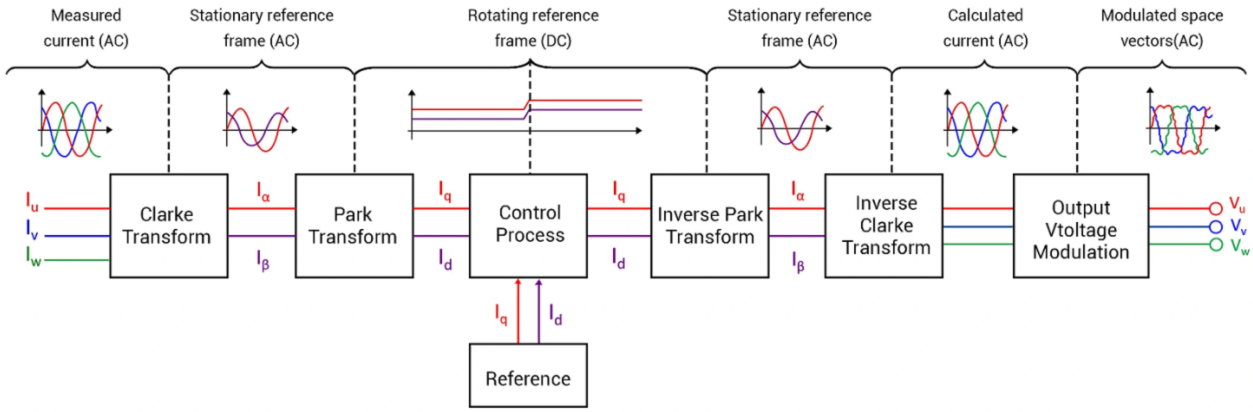


Figure 2.5: Transformations related to the control system with permission from Texas Instruments [2]

With the raw measurements taken from the physical system shown in figure 2.4, it is necessary to transform the current measurements into another reference frame before any form of control is implemented. It is desired that the stator flux vector is shifted 90 degrees from the magnetic axis of the rotor if one wants to maximize torque per amps. However, the measured motor currents are sinusoidal making it near impossible for a simple PI-controller to keep up with a sinusiodal reference. The solution to this problem is to transform the measured currents from the stationary *abc*-reference frame to the rotationary *dq*-reference frame where the magnetic axis of the rotor is the rotating reference as discussed in section 2.6.2. It turns out that in the *dq*-reference frame, the components of the stator flux vector remain constant, making it much simpler for a PI-controller to keep track of a reference.

Once the control process is completed, the output voltage from the controller is transformed back to the stationary *abc*-reference frame before it is applied to the motor via the inverter [21]. The field-oriented control process is visualized in figure 2.5.

Clarke-transformation

The transformations needed in order to transform the sinusoidal currents into constant DC-components are the Clarke- and Park-transformations. The Clarke-transformation transforms the three phase currents into two orthogonal components.

The α and β-components of the Clarke-transformation is given by:

$$i_\alpha = i_a \tag{2.32a}$$

$$i_\beta = \frac{1}{\sqrt{3}} i_a + \frac{2}{\sqrt{3}} i_b \tag{2.32b}$$

and the inverse Clarke-transformation is given by:

$$v_a = v_\alpha \tag{2.33a}$$

$$v_b = -\frac{1}{2} v_\alpha + \frac{\sqrt{3}}{2} v_\beta \tag{2.33b}$$

$$v_c = -\frac{1}{2} v_\alpha - \frac{\sqrt{3}}{2} v_\beta \tag{2.33c}$$

Park-transformation

The Park-transformation transforms the two orthogonal current components from the Clarke-transformation into two new orthogonal components, d and q. The interesting part is that the d-component is always aligned with the magnetic axis of the rotor. Since the q-component of the current is orthogonal to the d-axis, it becomes directly proportional to the induced motor torque. In other words, if one controls the magnitude of the q-current component then one controls the motor torque. Since the d-component does not contribute to torque, it is usually kept at zero unless it is desired to increase the motor speed above its design rating.

The d- and q-components of the Park-transformation is given by:

$$i_d = i_\alpha \cos\theta_e + i_\beta \sin\theta_e \tag{2.34a}$$

$$i_q = -i_\alpha \sin\theta_e + i_\beta \cos\theta_e \tag{2.34b}$$

and the inverse Park-transformation is given by:

$$v_\alpha = v_d \cos\theta_e - v_q \sin\theta_e \tag{2.35a}$$

$$v_\beta = v_q \cos\theta_e + v_d \sin\theta_e \tag{2.35b}$$

where θ_e is the angle between the magnetic axis of the rotor and the a-axis.

2.8 Incremental encoder

An *encoder* is a sensor which is mounted on the shaft of the rotor and gives information about the position of the rotor axis. The working principle of an incremental optical encoder consists of a disc mounted on the rotor-shaft of the motor with a given number of slits in it. On one side of the disc there is a light source, and on the other side light-receiving elements that produce pulses every time they detect light. It is common to use three light detecting elements. Two of the elements, namely A and B, are shifted by 90° and used to determine the direction of the rotation. This is done by inspecting if A lags B or if A leads B. The third element, Z, is used as an initialisation pulse and occurs once every revolution. The pulses are then used to trigger a counter which can be used to extract information about the position of the rotor axis. If A leads B, the counter increases and if A lags B, the counter decreases.

The angular position of the rotor-axis can be expressed as:

$$\theta_m = 2\pi \frac{P_c}{P_t} \tag{2.36}$$

where P_c is the number of pulses counted and P_t is the total number of slits.

There exists encoders using other working principles such as magnetic or absolute encoders. Absolute encoders have memory related to each slit such that it measures the absolute angle of the rotor-axis directly. Incremental encoders only measure the relative movement of the rotor shaft in regard to the start up position, meaning that incremental encoders have to use an external signal or the Z-pulse in order to measure the absolute angle of the rotor-axis [3].

2.9 Hall effect sensor

A *Hall effect sensor* detects the presence and magnitude of magnetic fields by using the Hall effect. These devices have a range of applications, with Hall effect sensors often used in automation systems to detect position, distance and speed. The working principle of Hall effect sensors is based on applying a constant current through a semiconductor. In the presence of a magnetic field perpendicular to the direction of the current, the charge carriers are deflected by the Lorentz force, producing a difference in electric potential between the two sides of the semiconductor. This voltage difference is proportional to the strength of the magnetic field.

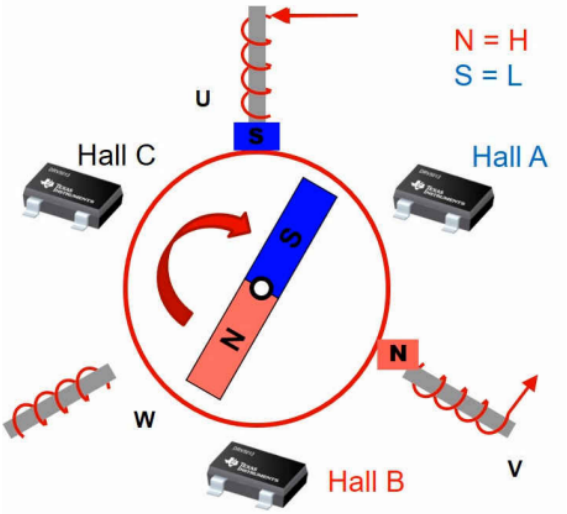


Figure 2.6: Placement of Hall sensors in a PMSM with permission from Texas Instruments [22]

There are two main types of Hall effect sensors: those with analogue outputs, and those that have digital outputs. Analogue sensors use a continuous voltage output that increases within a strong magnetic field and decreases in a weaker field. The digital output Hall sensor has a *Schmitt trigger*, which is a circuit that converts analog signals into digital signals. Due to the Schmitt trigger, when the magnetic flux passing through the Hall sensor exceeds the sensor's threshold value, the output of the Hall sensor switches from 'off' to 'on'. The digital Hall sensors can also be separated into two types depending on their working principles: bipolar and unipolar. Bipolar devices need a positive magnetic field (which comes from the south pole of a magnet) to operate them, and use the negative field (from the north pole) to release them. Unipolar devices only need a single magnetic south pole to both operate and release them as they move in and out of the magnetic field [20].

Figure 2.6 shows how digital, unipolar Hall sensors can be utilized to detect the position and rotational speed of the rotor.

2.10 Sensorless Permanent Magnet Synchronous Motor Drives

Due to the potential benefits in terms of system costs and improved reliability by eliminating the need for encoders in control systems, sensorless control methods have attracted much attention from both academic and industrial enterprises. Consequently, several sensorless control methods have been developed over the years for the whole speed range of PMSMs - each method with its own strengths and weaknesses.

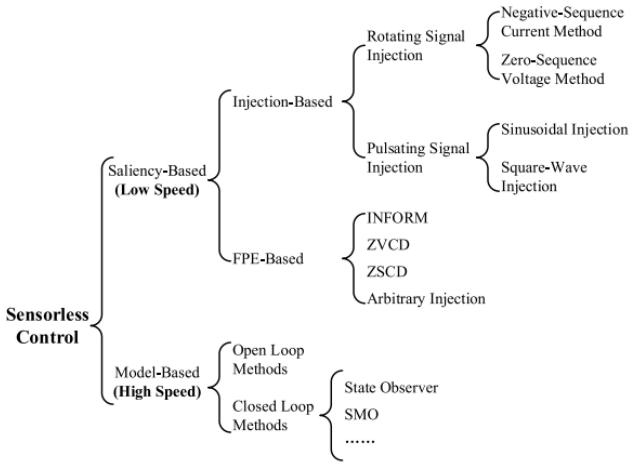


Figure 2.7: Overview of different sensorless control schemes [24]

Figure 2.7 gives an overview of the different sensorless control schemes that has been developed. It shows that the control methods are roughly split into two categories: *Saliency-Based* and *Model-Based* methods. For medium and high-speed applications, model-based methods use the techniques of back EMF or flux estimators to obtain the rotor position so that field-oriented control can be implemented. However, these methods fail in the low-speed region as the magnitude of the back EMF is directly proportional to the motor speed. Hence, the rotor position can not be estimated during low-speed motor applications due to low *signal-to-noise* ratio and other factors such as modeling uncertainty and inverter nonlinearities.

In order to expand sensorless control into the low- to-zero speed range, saliency-based methods have been developed using techniques such as *high-frequency signal injection* (HFI) in order to extract the rotor position. These methods take advantage of the salient property of PMSMs where the inductances change depending on the position of the rotor. This change in inductance can then be used to estimate the rotor position. However, there are negative effects to the injection of high frequent signals such as increased losses, torque ripples and accoustic noise. Besides, the maximum output voltage of an inverter at a higher operating speed range could possibly limit the additional injected signal. Consequently, it is recommended to use saliency-based methods with signal injection only at low and zero-speed ranges [24].

2.11 Model-Based Nonlinear Observer

In order to perform field-oriented control for medium and high-speed ranges, the nonlinear observer proposed by *Ortega et.al* [14] will be used to estimate the position and speed of the rotor. This nonlinear observer is chosen because it is suitable for SPMSMs and because it is the same method that has been physically implemented at NTNU for the control of a motor supplied by Alva Industries.

The nonlinear observer estimates the rotor position (θ_e) by using the dynamic equations of the PMSM introduced in section 2.6.1 together with voltage and current measurements from the motor. The motor dynamics in the $\alpha\beta$-reference are defined as:

$$L_s\dot{\mathbf{i}}_{\alpha\beta} = -R_s\mathbf{i}_{\alpha\beta} + \omega_e\psi \begin{bmatrix} \sin(\theta_e) \\ -\cos(\theta_e) \end{bmatrix} + \mathbf{v}_{\alpha\beta} \tag{2.37}$$

where $\mathbf{i}_{\alpha\beta}$ is the stator current, $\mathbf{v}_{\alpha\beta}$ is the motor terminal voltage, θ_e the electrical angle of the rotor, ω_e the electrical speed of the rotor, R_s is the stator resistance, L_s is the stator inductance and ψ_m is the permanent magnet flux linkage.

Furthermore, a new state observer is defined in [14] as:

$$\mathbf{x} = L_s\mathbf{i}_{\alpha\beta} + \psi_m \begin{bmatrix} \cos(\theta_e) \\ \sin(\theta_e) \end{bmatrix} \tag{2.38}$$

Let,

$$\mathbf{y} \equiv -R_s\mathbf{i}_{\alpha\beta} + \mathbf{v}_{\alpha\beta} \tag{2.39}$$

Then it follows from 2.37, 2.38 and 2.39 that:

$$\dot{\mathbf{x}} = L_s\dot{\mathbf{i}}_{\alpha\beta} - \omega_e\psi_m \begin{bmatrix} \sin(\theta_e) \\ -\cos(\theta_e) \end{bmatrix} = \mathbf{y} \tag{2.40}$$

Reducing the dynamics to the simple form $\dot{\mathbf{x}} = \mathbf{y}$. To introduce the nonlinear position observer, the vector function $\eta : \mathbb{R}^2 \to \mathbb{R}^2$ is defined as:

$$\eta(\mathbf{x}) = \mathbf{x} - L_s\mathbf{i}_{\alpha\beta} \tag{2.41}$$

In view of 2.38, its Euclidean norm is equal to:

$$\left\|\eta(\mathbf{x})\right\|^2 = \psi_m^2 \tag{2.42}$$

Consider then the nonlinear observer:

$$\dot{\hat{\mathbf{x}}} = \mathbf{y} + \frac{\gamma}{2}\eta(\hat{\mathbf{x}})\left[\psi_m^2 - \left\|\eta(\hat{\mathbf{x}})\right\|^2\right] \tag{2.43}$$

Using 2.38, one can define:

$$\begin{bmatrix}\cos(\hat{\theta}_e)\\ \sin(\hat{\theta}_e)\end{bmatrix} = \frac{1}{\psi_m}(\hat{\mathbf{x}} - L_s\mathbf{i}_{\alpha\beta}) \tag{2.44}$$

From this, one can extract $\hat{\theta}_e$ which is an estimate of the rotor position θ_e.

$$\hat{\theta}_e = \tan^{-1}\left(\frac{\hat{x_2} - L_s i_\beta}{\hat{x_1} - L_s i_\alpha}\right) \tag{2.45}$$

Speed observer

Given that an estimate of the rotor position is acquired, it can be further used to estimate the rotor speed. However, it is not desirable to obtain a speed estimate through numerical differentiation of the position estimates. Instead, a PLL-type PI tracking controller suggested in [14] is used to estimate the speed:

$$\dot{z_1} = K_p(\hat{\theta}_m - z_1) + K_i z_2 \tag{2.46a}$$

$$\dot{z_2} = \hat{\theta}_m - z_1 \tag{2.46b}$$

$$\hat{\omega}_e = K_p(\hat{\theta}_m - z_1) + K_i z_2 \tag{2.46c}$$

where $\hat{\omega}_e$ is an estimate of the rotor speed ω_e.

2.12 Low-Speed HFI Observer

Based on an observability analysis performed by *Vaclavek et.al* [23], an observability condition in order for SPMSMs to be fully observable is defined under the assumptions that the electrical parameters of the machine are known and constant, load torque and inertia is unknown and the rotor speed is constant or slowly varying:

$$\omega_e \neq 0 \tag{2.47}$$

This condition clearly demonstrates that model-based sensorless control techniques fail at speeds near zero in the case of SPMSMs, hence the need for saliency-based sensorless methods.

High-Frequency Square-Wave Voltage Injection

For the low-to-zero speed range, the observer from 2.11 will be replaced with the high-frequency current injection-based position observer proposed by *Qiao et.al* [25]. This observer is chosen because it is specifically designed for SPMSMs, uses a simple pulsating square-wave signal and allows for higher frequency injections in comparison to other similar methods such as high-frequency sinusoidal injections.

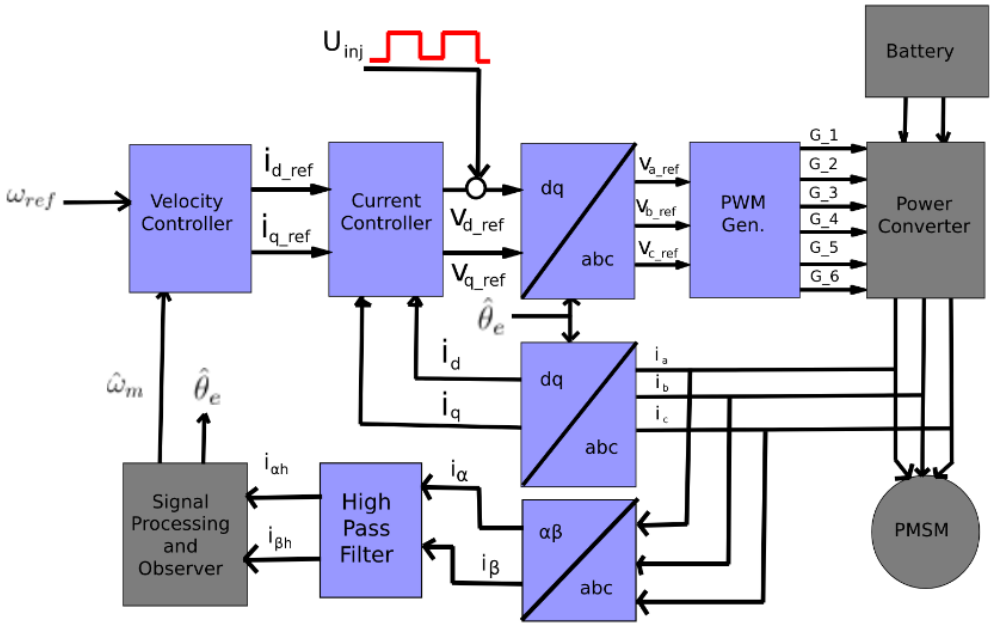


Figure 2.8: Block diagram of pulsating square-wave injection inspired from Qiao et.al [24]

Figure 2.8 shows the block diagram of the pulsating square-wave injection control scheme. The only thing added to the control system from section 2.7 is the square-wave voltage generator that injects voltage into the d-component of the PMSM supply voltage.

High-Frequency Impedance Model of an PMSM

The high-frequency current injection-based position observer is derived from the dynamic equations of the PMSM and uses current measurements from the motor to estimates the rotor position. The dynamics of a PMSM expressed in the dq-rotating reference frame is repeated below:

$$\begin{bmatrix} v_d \\ v_q \end{bmatrix} = \begin{bmatrix} R_s + L_s p & -\omega_{re} L_s \\ \omega_{re} L_s & R_s + L_s p \end{bmatrix} \begin{bmatrix} i_d \\ i_q \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_{re} \lambda_m \end{bmatrix} \tag{2.48}$$

where p is the derivative operator, v_d and v_q are the stator voltages, i_d and i_q are the stator currents, ω_{re} is the rotor electrical rotating speed in rad/s, L_s is the stator inductance, R_s is the stator resistance and λ_m is the permanent flux linkage.

If high-frequency pulsating voltage signals, $v_{d,h}$ and $v_{q,h}$, whose frequency is sufficiently higher than the rotor speed, are injected into the machine stator windings, high-frequency currents, $i_{d,h}$ and $i_{q,h}$, will be generated. Due to their high frequency, the derivatives of these signals can be quite large. Hence, when considering the high-frequency components of the system, the off-diagonal cross-coupling terms in 2.48 are sufficiently smaller than the diagonal terms and can be ignored. Similarly, in the low-speed region and at standstill, the back EMF term can also be neglected. As a result, the high-frequency model of the PMSM in the low-speed region can be expressed as:

$$\begin{bmatrix} v_{d,h} \\ v_{q,h} \end{bmatrix} = \begin{bmatrix} Z_{d,h} & 0 \\ 0 & Z_{q,h} \end{bmatrix} \begin{bmatrix} i_{d,h} \\ i_{q,h} \end{bmatrix} \tag{2.49}$$

where $Z_{d,h} = R_{d,h} + j\omega_h L_{d,h}$ and $Z_{q,h} = R_{q,h} + j\omega_h L_{q,h}$ are the d-axis and q-axis high-frequency impedances and ω_h is the frequency of the injected signals [25].

Injection of High-Frequency Pulsating Signals

In this section, the high-frequency impedance model expressed in 2.49 is used to derive the expressions for the induced high-frequency currents which is later used to estimate the rotor position. Firstly, the conventional position estimation scheme using pulsating sinusoidal voltage injection is briefly presented. That method is highly dependent on rotor saliency and is therefore not effective for SPMSMs. To solve this problem, a position estimation method independent on rotor spatial saliency is proposed by *Qiao et.al* [25]. The proposed method is first discussed based on a sinusoidal voltage injection, before a square-wave voltage injection scheme is put forward in order to improve the upper bandwidth of the sensorless speed control.

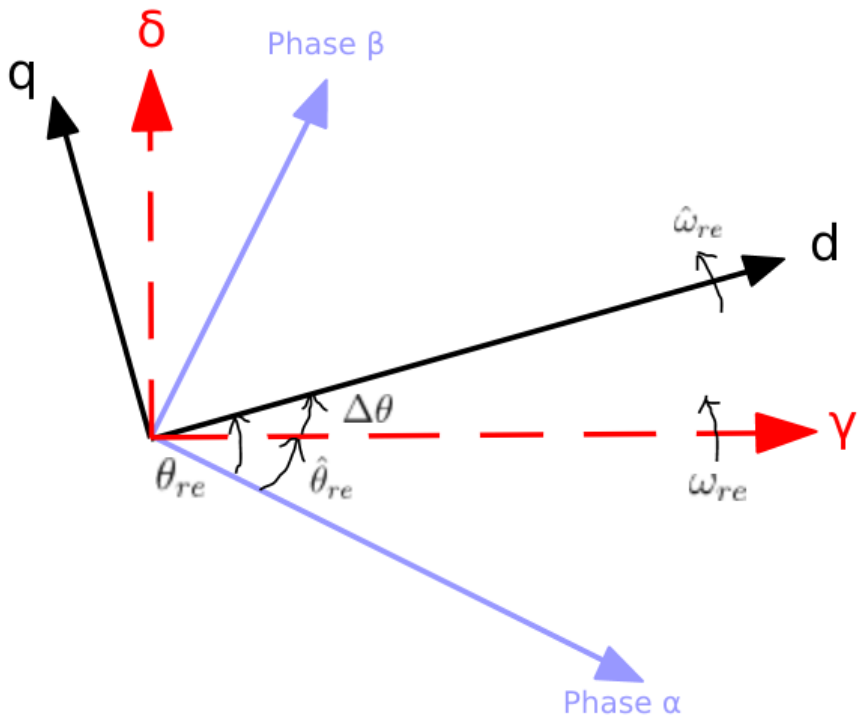


Figure 2.9: Relationships among the $\alpha-\beta$ stationary reference frame, the ideal $d-q$ rotating reference frame and the estimated $\gamma-\delta$ rotating reference frame with inspiration from Qiao et.al [25]

The angle between γ-axis and α-axis is defined as the estimated position, $\hat{\theta}_{re}$, as shown in figure 2.9. The error between the actual and estimated positions is denoted as $\Delta\theta$. A sinusoidal pulsating voltage vector described by 2.50 is injected into the estimated $\gamma-\delta$ rotating reference frame:

$$\mathbf{v}_{\gamma\delta,\mathbf{h}} = \begin{bmatrix} v_{\gamma,h} \\ v_{\delta,h} \end{bmatrix} = V_h \begin{bmatrix} \cos(\omega_h t) \\ 0 \end{bmatrix} \tag{2.50}$$

where ω_h and V_h are the frequency and amplitude of the injected voltage vector.

Projecting $\mathbf{V}_{\gamma\delta,\mathbf{h}}$ onto the dq-axes, the resulting voltage vector, $\mathbf{V}_{\mathbf{dq},\mathbf{h}}$, can be expressed as:

$$\mathbf{v}_{\mathbf{dq},\mathbf{h}} = \begin{bmatrix} v_{d,h} \\ v_{q,h} \end{bmatrix} = \begin{bmatrix} \cos(\Delta\theta) & \sin(\Delta\theta) \\ -\sin(\Delta\theta) & \cos(\Delta\theta) \end{bmatrix} \begin{bmatrix} v_{\gamma,h} \\ v_{\delta,h} \end{bmatrix} = v_{\gamma,h} \begin{bmatrix} \cos(\Delta\theta) \\ -\sin(\Delta\theta) \end{bmatrix} \tag{2.51}$$

According to 2.49 and 2.51, the induced high-frequency currents in the ideal dq-reference frame can be determined:

$$\begin{bmatrix} i_{d,h} \\ i_{q,h} \end{bmatrix} = v_{\gamma,h} \begin{bmatrix} \cos(\Delta\theta)/Z_{d,h} \\ -\sin(\Delta\theta)/Z_{q,h} \end{bmatrix} \tag{2.52}$$

In conventional HFI methods, the position information is extracted from the induced current signal in the estimated rotating reference frame as follows:

$$\begin{bmatrix} i_{\gamma,h} \\ i_{\delta,h} \end{bmatrix} = \begin{bmatrix} \cos(\Delta\theta) & -\sin(\Delta\theta) \\ \sin(\Delta\theta) & \cos(\Delta\theta) \end{bmatrix} \begin{bmatrix} i_{d,h} \\ i_{q,h} \end{bmatrix} = v_{\gamma,h} \begin{bmatrix} \frac{\cos^2(\Delta\theta)}{Z_{d,h}} + \frac{\sin^2(\Delta\theta)}{Z_{q,h}} \\ \frac{(Z_{q,h}-Z_{d,h})}{2Z_{d,h}Z_{q,h}} \sin(2\Delta\theta) \end{bmatrix} \tag{2.53}$$

From equation 2.53, it can be seen that $i_{\delta,h}$ is a function of the position tracking error $\Delta\theta$. However, the magnitude of $i_{\delta,h}$ depends on the rotor saliency. If the saliency is small, as in an SPMSM where,

$$Z_{q,h} - Z_{d,h} \ll Z_{q,h} + Z_{d,h}$$

the method suggested in 2.53 will not be effective for position estimation due to the low signal-to-noise ratio of the saliency related signal. To solve this problem, a better position observation method which has low dependence on the rotor saliency is needed for SPMSMs in the low-speed region.

Saliency Independent Position Observer

In the proposed method, the rotor position is obtained from the induced current vector, $\mathbf{i}_{\alpha\beta,\mathbf{h}}$, in the $\alpha\beta$-reference frame as follows:

$$\begin{aligned}\mathbf{i}_{\alpha\beta,\mathbf{h}} &= \begin{bmatrix} i_{\alpha,h} \\ i_{\beta,h} \end{bmatrix} = \begin{bmatrix} \cos(\theta_{re}) & -\sin(\theta_{re}) \\ \sin(\theta_{re}) & \cos(\theta_{re} \end{bmatrix} \begin{bmatrix} i_{d,h} \\ i_{q,h} \end{bmatrix} \\ &= V_h \cos(\omega_h t) \begin{bmatrix} \frac{\cos(\Delta\theta)}{Z_{d,h}} \cos(\theta_{re}) + \frac{\sin(\Delta\theta)}{Z_{q,h}} \sin(\theta_{re}) \\ \frac{\cos(\Delta\theta)}{Z_{d,h}} \sin(\theta_{re}) - \frac{\sin(\Delta\theta)}{Z_{q,h}} \cos(\theta_{re}) \end{bmatrix}\end{aligned} \tag{2.54}$$

If the position error $\Delta\theta$ is small enough such that $\sin(\Delta\theta) \approx 0$ and $\cos(\Delta\theta) \approx 1$, then 2.54 can be simplified as:

$$\begin{bmatrix} i_{\alpha,h} \\ i_{\beta,h} \end{bmatrix} \approx \frac{V_h \cos(\omega_h t)}{Z_{d,h}} \begin{bmatrix} \cos(\theta_{re}) \\ \sin(\theta_{re}) \end{bmatrix} \tag{2.55}$$

As shown in 2.55, if the rotating frequency of the machine is much smaller than the frequency of the injected signal, the envelopes of $\mathbf{i}_{\alpha\beta,\mathbf{h}}$ are position dependent signals. If the envelopes are extracted, the rotor position can be obtained.

Since for an SPMSM the difference between $Z_{d,h}$ and $Z_{q,h}$ can be neglected (i.e., $Z_{d,h} \approx Z_{q,h} \approx Z_{s,h}$, equation 2.54 can be simplified as follows:

$$\begin{bmatrix} i_{\alpha,h} \\ i_{\beta,h} \end{bmatrix} = \frac{V_h \cos(\omega_h t)}{Z_{s,h}} \begin{bmatrix} \cos(\theta_{re} - \Delta\theta) \\ \sin(\theta_{re} - \Delta\theta) \end{bmatrix} = \frac{V_h \cos(\omega_h t)}{Z_{s,h}} \begin{bmatrix} \cos(\hat{\theta}_{re}) \\ \sin(\hat{\theta}_{re}) \end{bmatrix} \tag{2.56}$$

Equations 2.55 and 2.56 are both simplified versions of 2.54. Although they are derived based on different assumptions, both of them indicate that the rotor position information can be directly obtained from the envelopes of $\mathbf{i}_{\alpha\beta,\mathbf{h}}$ if a high-frequency pulsating voltage vector is injected in the $\gamma - \delta$ reference frame:

$$\hat{\theta}_{re} = -\tan^{-1}\left(\frac{\tilde{i}_{\beta,h}}{\tilde{i}_{\alpha,h}}\right) \tag{2.57}$$

where $\tilde{i}_{\alpha,h}$ and $\tilde{i}_{\beta,h}$ are the envelopes of $i_{\alpha,h}$ and $i_{\beta,h}$.

Square-Wave Signal Injection

As described by 2.55 and 2.56 in the proposed method, to extract the rotor position information, only the envelopes of $\mathbf{i}_{\alpha\beta,\mathbf{h}}$ are needed. Since the envelopes are mainly constituted by the extremes of the current waveforms, it is more sensible to use a square-wave signal instead of a sinusoidal one. The highest frequency of the injected square-wave signal is at least more than twice the frequency of the sinusoidal signal, hence allowing the position observer to operate at higher rotational speeds. If the reference values of the voltages v_d and v_q can be updated twice per PWM cycle, the highest frequency of injected square-wave signal is equal to the PWM frequency.

A square-wave voltage vector expressed as:

$$\mathbf{v}_{\gamma\delta,\mathbf{h}} = \begin{bmatrix} v_{\gamma,h} \\ v_{\delta,h} \end{bmatrix} = V_h \begin{bmatrix} (-1)^n \\ 0 \end{bmatrix} \tag{2.58}$$

can be injected into γ and δ axes, where n is the index of the PWM cycles. When n is odd:

$$\begin{bmatrix} v_{d,h} \\ v_{q,h} \end{bmatrix} = V_h \begin{bmatrix} -\cos(\Delta\theta) \\ \sin(\Delta\theta) \end{bmatrix} \text{ and } \begin{bmatrix} i_{d,h} \\ i_{q,h} \end{bmatrix} = V_h \begin{bmatrix} -\cos(\Delta\theta)/Z_{d,h} \\ \sin(\Delta\theta)/Z_{q,h} \end{bmatrix} \tag{2.59}$$

Then:

$$\begin{bmatrix} i_{\alpha,h} \\ i_{\beta,h} \end{bmatrix} = V_h \begin{bmatrix} -\frac{\cos(\Delta\theta)}{Z_{d,h}}\cos(\theta_{re}) - \frac{\sin(\Delta\theta)}{Z_{q,h}}\sin(\theta_{re}) \\ -\frac{\cos(\Delta\theta)}{Z_{d,h}}\sin(\theta_{re}) + \frac{\sin(\Delta\theta)}{Z_{q,h}}\cos(\theta_{re}) \end{bmatrix} \tag{2.60}$$

If $Z_{d,h} \approx Z_{q,h} \approx Z_{s,h}$, equation 2.60 can be further simplified as:

$$\begin{bmatrix} i_{\alpha,h} \\ i_{\beta,h} \end{bmatrix} = \frac{V_h}{Z_{s,h}} \begin{bmatrix} -\cos(\theta_{re} - \Delta\theta) \\ \sin(\theta_{re} - \Delta\theta) \end{bmatrix} = \frac{V_h}{Z_{s,h}} \begin{bmatrix} -\cos(\hat{\theta}_{re}) \\ \sin(\hat{\theta}_{re}) \end{bmatrix} \tag{2.61}$$

When n is even:

$$\begin{bmatrix} v_{d,h} \\ v_{q,h} \end{bmatrix} = V_h \begin{bmatrix} \cos(\Delta\theta) \\ -\sin(\Delta\theta) \end{bmatrix} \text{ and } \begin{bmatrix} i_{d,h} \\ i_{q,h} \end{bmatrix} = V_h \begin{bmatrix} \cos(\Delta\theta)/Z_{d,h} \\ -\sin(\Delta\theta)/Z_{q,h} \end{bmatrix} \tag{2.62}$$

Then,

$$\begin{bmatrix} i_{\alpha,h} \\ i_{\beta,h} \end{bmatrix} = V_h \begin{bmatrix} \frac{\cos(\Delta\theta)}{Z_{d,h}}\cos(\theta_{re}) + \frac{\sin(\Delta\theta)}{Z_{q,h}}\sin(\theta_{re}) \\ -\frac{\cos(\Delta\theta)}{Z_{d,h}}\sin(\theta_{re}) + \frac{\sin(\Delta\theta)}{Z_{q,h}}\cos(\theta_{re}) \end{bmatrix} \tag{2.63}$$

If $Z_{d,h} \approx Z_{q,h} \approx Z_{s,h}$, equation 2.63 can be further simplified as:

$$\begin{bmatrix} i_{\alpha,h} \\ i_{\beta,h} \end{bmatrix} = \frac{V_h}{Z_{s,h}} \begin{bmatrix} \cos(\theta_{re} - \Delta\theta) \\ -\sin(\theta_{re} - \Delta\theta) \end{bmatrix} = -\frac{V_h}{Z_{s,h}} \begin{bmatrix} -\cos(\hat{\theta}_{re}) \\ \sin(\hat{\theta}_{re}) \end{bmatrix} \tag{2.64}$$

According to 2.61 and 2.64, the final expression for the $\mathbf{i}_{\alpha\beta,\mathbf{h}}$ can be expressed as:

$$\begin{bmatrix} i_{\alpha,h} \\ i_{\beta,h} \end{bmatrix} = \frac{V_h}{Z_{s,h}} \begin{bmatrix} \cos(\theta_{re} & -\Delta\theta) \\ -\sin(\theta_{re} & -\Delta\theta) \end{bmatrix} = (-1)^n \frac{V_h}{Z_{s,h}} \begin{bmatrix} -\cos(\hat{\theta}_{re}) \\ \sin(\hat{\theta}_{re}) \end{bmatrix} \tag{2.65}$$

Then by detecting the envelopes of the current components in 2.65, the estimated rotor position can be extracted as follows [25]:

$$\begin{cases} \tilde{i}_{\alpha,h} = -\frac{V_h}{Z_{s,h}} \cos(\hat{\theta}_{re}) \\ \tilde{i}_{\beta,h} = \frac{V_h}{Z_{s,h}} \sin(\hat{\theta}_{re}) \end{cases} \text{and } \hat{\theta}_{re} = -\tan^{-1}\left(\frac{\tilde{i}_{\beta,h}}{\tilde{i}_{\alpha,h}}\right) \tag{2.66}$$

where $\tilde{i}_{\alpha,h}$ and $\tilde{i}_{\beta,h}$ are the envelopes of $i_{\alpha,h}$ and $i_{\beta,h}$.

2.13 Propellers

Propellers are used in drones to make them capable of flying and are connected to the rotor-shaft of the motor. The propellers create upward thrust that accelerates the drone, working as a load for the motor we are simulating. In this thesis it is therefore necessary to be able to model the load that the motor experiences from the propellers.

The torque from the propellers can be modeled by the following equations:

$$J = \frac{2\pi V_a}{\omega_m D} \tag{2.67a}$$

$$Q = \frac{\rho D^5}{4\pi^2}(C_{Q,0} + C_{Q,1} J)\omega_m^2 \tag{2.67b}$$

where J is the advanced ratio, V_a true airspeed, ω_m the rotor velocity, D the propeller diameter, ρ density of air, $C_{Q,0}$ and $C_{Q,1}$ the parametrization parameters of the torque coeffisient C_Q [5].

Chapter 3

Modelling

Chapter 3 describes the work that has been carried out over the course of this thesis. This includes the modelling of a PMSM using *Simulink* and the development of a sensorless control system that utilizes different estimators to control the simulated motor.

3.1 Modelling the Permanent Magnet Synchronous Motor

By using templates already developed by MathWorks [17], a great amount of time was saved during the modelling phase of the motor. If it were not for the existing templates, one would have to model the motor manually by using Simulink blocks and the equations from 2.29. As there already exists precise motor models available and that simulating the motor is the main scope of this thesis, it made sense to use the templates from MathWorks so that more time could be spent on researching the control system of the motor. However, the template provided by MathWorks was greatly modified in order resemble the motor designed by Alva Industries.

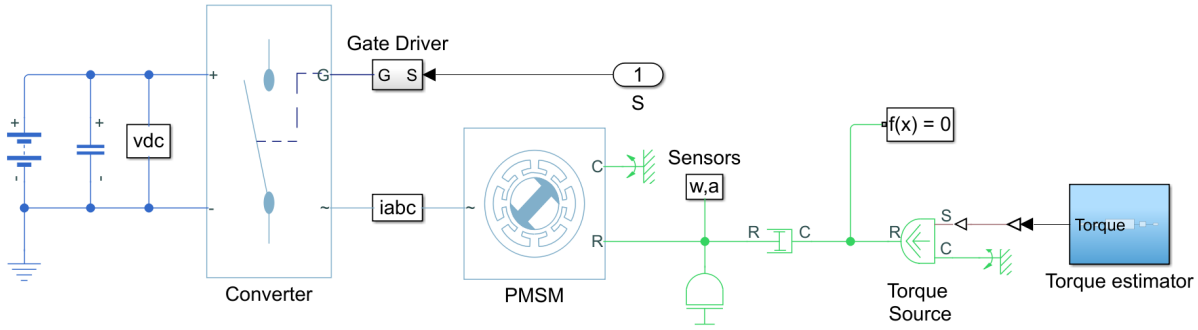


Figure 3.1: The physical motor model in Simulink

Once a functional motor model was acquired, one simply had to enter the specific motor parameters of the motor that was going to be simulated. The torque estimator was developed using the equations from section 2.13 regarding the modelling of propellers.

Power circuit

The power circuit consists of a battery and an inverter. The battery is modelled by a battery block integrated in Simulink while the three phase inverter is modelled by six ideal semiconductor switches in a two-level topology. The duty cycle of each ideal semiconductor switch is supplied from the SVPWM generator in figure 3.2.

3.2 Development of the control system

The control system is based on the principle of field-oriented control from section 2.7 and consists of multiple nested controllers dependent on the type of control one wishes to implement e.g. torque control, speed control etc. Regardless, field-oriented control must always have an inner control loop where the motor current is controlled which again controls the rotor torque and flux. The current-controller is considered by many as the most complex part of the control system because it is here the current is transformed from a stationary reference frame into a rotating reference frame in order to simplify the control process. Once the inner control loop controlling the torque and flux is established, it is possible to add outer control loops that control other entities such as rotor speed and position.

Current controller

The inner control loop controlling the motor current is realized by two separate PID-controllers, one controlling the rotor torque and another controlling the rotor flux. The quadrature current, i_q, has a reference which is determined by an outer loop (speed control) while the direct current, i_d, has a reference of zero as there is no desire to alter the rotor flux. The output voltages of the current controller is transformed back into the stationary abc-reference and used as a reference for the SVPWM generator which determines the duty cycles of each transistor in the DC/AC-inverter supplying the motor with power.

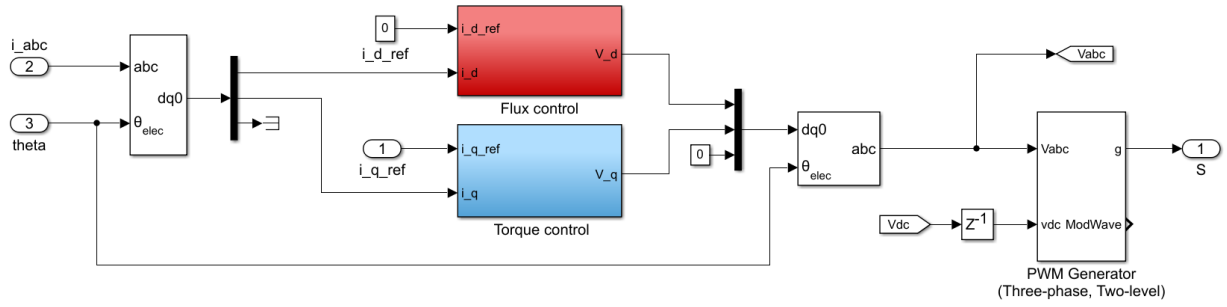


Figure 3.2: Current control

Speed- and position controller

The outer control loop controlling the rotor speed consists of a simple PID-controller where the speed reference is given by the user or another outer control loop such as a position controller. The output of the speed controller becomes the reference for the quadrature current, i_q in the inner control loop described in section 3.2.

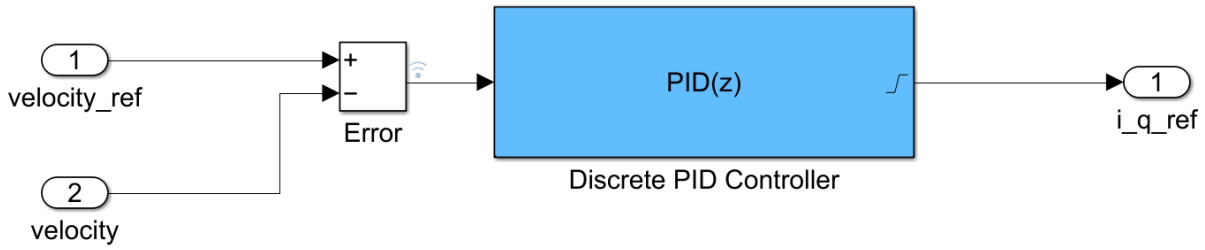


Figure 3.3: Speed control

If desired, position control can also be added. The position controller, which also consists of a PID-controller, outputs a speed reference for the speed controller in figure 3.3.

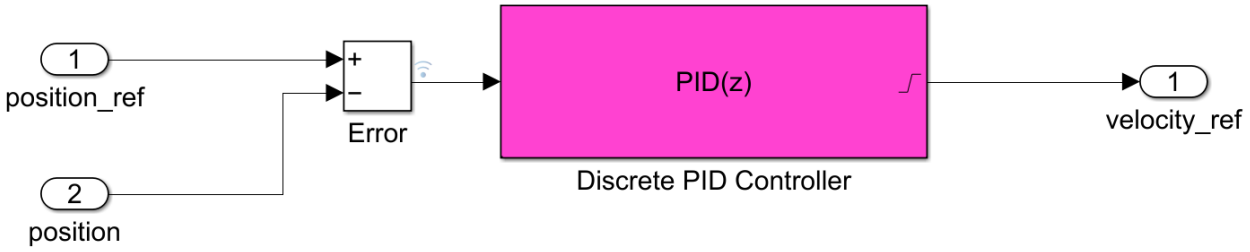


Figure 3.4: Position control

3.3 Implementing the Sensorless Observers

Given that encoders increase system costs and reduce the robustness of the system, it is desirable to explore the possibilities of using sensorless feedback i.e. using estimators instead of direct encoder measurements.

3.3.1 Implementing the Nonlinear Observer

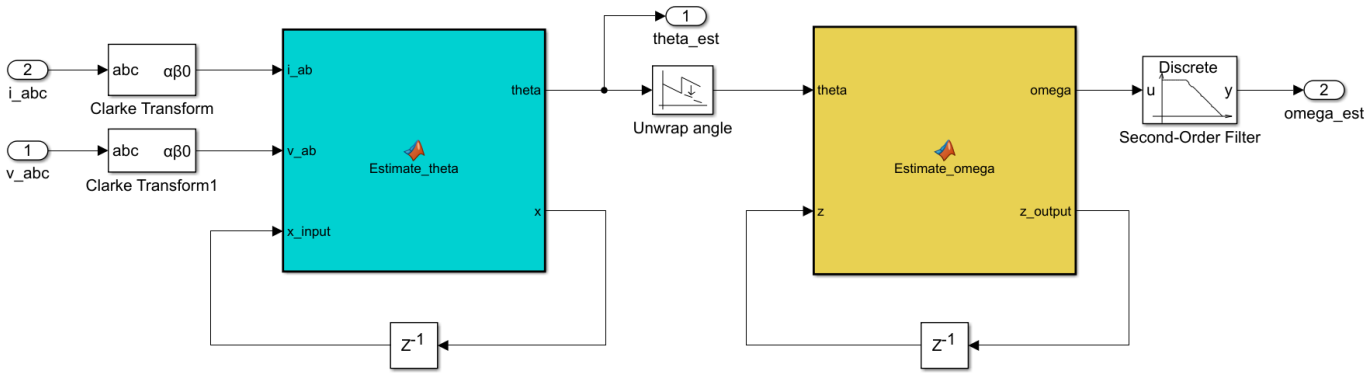


Figure 3.5: Nonlinear observer implemented in Simulink

The estimation of the rotor position $\hat{\theta}_e$ and rotor speed $\hat{\omega}_e$ using the nonlinear observer is implemented by using motor current- and voltage measurements together with the equations suggested in section 2.11.

The *unwrap angle*-function from the MATLAB-library is used to prevent large spikes in the speed estimate each time the rotor position jumps between $[0, 2\pi]$ or $[-\pi, \pi]$. In addition, a low-pass filter is added to the speed estimate in order to reduce the noise produced when deriving the rotor speed from the rotor position. Finally, the estimators of the rotor speed and position replace the encoder feedback in the control loops mentioned above.

3.3.2 Implementing the HFI Observer

Model-based sensorless control techniques, such as the nonlinear observer suggested above, fail at speeds near zero in the case of SPMSMs. As a result, high-frequency current injections is needed in order to perform sensorless control in the low-to-zero speed range. The square-wave HFI observer proposed in section 2.12 is implemented in the control system as shown below:

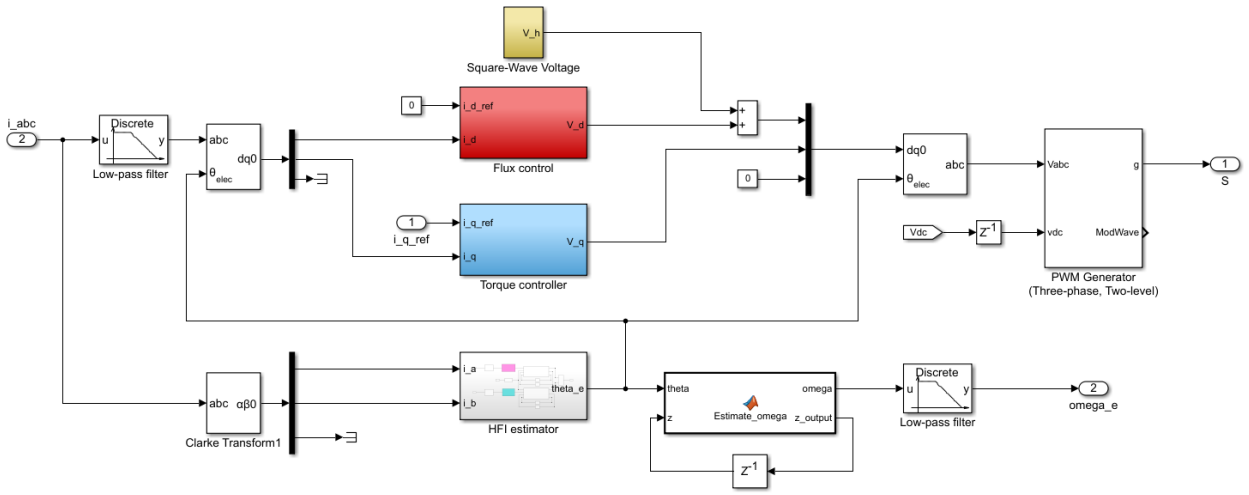


Figure 3.6: The HFI observer implemented into the control system

From the control system introduced in figure 3.2, it has now been added a square-wave signal which is injected into the d-axis voltage and a *HFI estimator*-block. The square-wave signal is the main component of the HFI scheme and allows us to obtain high-frequency current components which is used by the HFI estimator to estimate the rotor position. The low-pass filter added to $\mathbf{i_{abc}}$ is used to filter out these high-frequency currents so that they do not interfere with the performance of the control system.

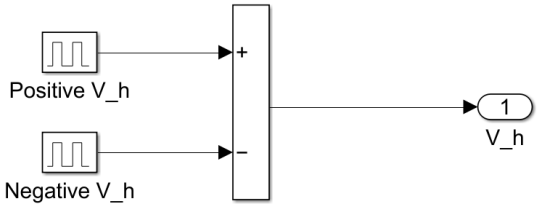


Figure 3.7: Modelling the square-wave voltage V_h

The high frequency square-wave voltage, V_h, is constructed as the sum of two square-wave generators that are in opposite phases and sign. In that way, a square-wave voltage alternating between $[-V_h, V_h]$ with frequency ω_h is created.

The HFI estimator-block which estimates the rotor position is shown below:

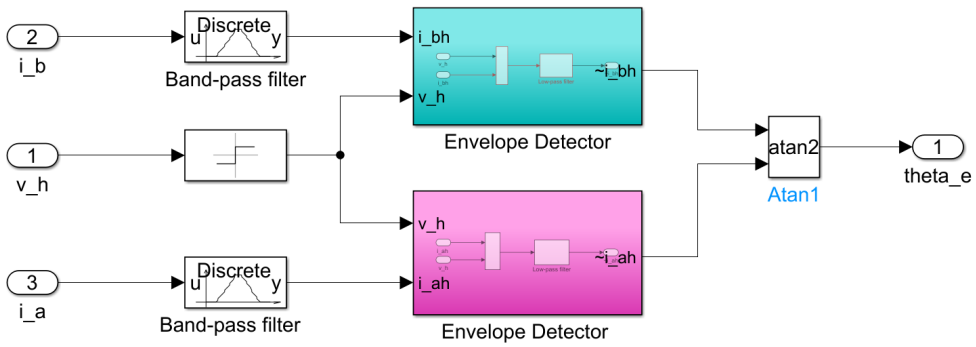


Figure 3.8: Estimation of the rotor position θ_e

The block inputs the currents i_α, i_β and the injection voltage V_h. The current signals are both passed through a band-pass filter with natural frequency ω_w so that the high-frequency current components $i_{\alpha,h}$ and $i_{\beta,h}$ can be isolated. The injection voltage V_h is used to demodulate the current signals and the envelope detector extracts the envelopes of the high-frequency current components.

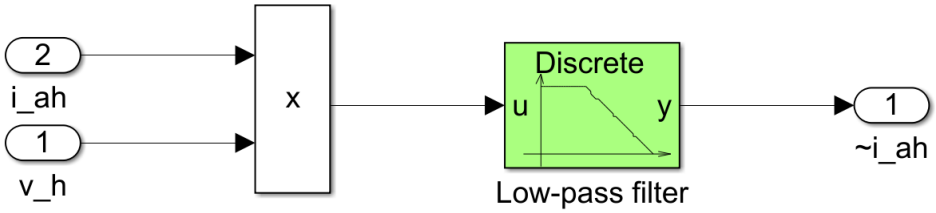


Figure 3.9: The envelope detector

The envelope detector demodulates the current signals by taking the product of the current signals and the sign of the injection voltage. Secondly, a second order low-pass filter is used to outline the current envelopes before the position estimate is calculated by using the *arctan*-function:

$$\hat{\theta}_{re} = \tan^{-1}\left(\frac{\tilde{i}_{\beta,h}}{\tilde{i}_{\alpha,h}}\right) \tag{3.1}$$

3.4 Implementing the Kalman Filter

The Extended Kalman Filter

Seeing that the dynamics of the PMSM described in section 2.6.1 are nonlinear, the use of the extended Kalman filter is necessary for this system. The dynamics are repeated here so that the matrices $\mathbf{F}_k$, $\mathbf{B}$, $\mathbf{H}$, $\mathbf{Q}$ and $\mathbf{R}$ can be derived:

$$\frac{d}{dt}i_d = -\frac{R_s}{L_d}i_d + \frac{L_q}{L_d}N\omega_m i_q + \frac{1}{L_d}v_d \tag{3.2a}$$

$$\frac{d}{dt}i_q = -\frac{R_s}{L_q}i_q - \frac{L_d}{L_q}N\omega_m i_d - \frac{\lambda_m N\omega_m}{L_q} + \frac{1}{L_q}v_q \tag{3.2b}$$

$$\frac{d}{dt}\omega_m = \frac{1}{J}\left(\frac{3}{2}N\left(i_q\lambda_m + i_d i_q(L_d - L_q)\right) - D_m\omega_m - T_L\right) \tag{3.2c}$$

$$\frac{d}{dt}\theta_e = N\omega_m \tag{3.2d}$$

Let the state vector $\mathbf{x}$, the input vector $\mathbf{u}$ and the output vector $\mathbf{y}$ be defined as:

$$\mathbf{x} = \begin{bmatrix} i_d & i_q & \omega_m & \theta_e \end{bmatrix}^T, \quad \mathbf{u} = \begin{bmatrix} v_d & v_q \end{bmatrix}^T, \quad \mathbf{y} = \begin{bmatrix} i_d & i_q \end{bmatrix}^T \tag{3.3}$$

If one assumes that the rotor velocity is constant during the sampling period T, one can instead set $\frac{d}{dt}\omega_m = 0$. The plant model is then defined as:

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} = \begin{bmatrix} -\frac{R_s}{L_d}i_d + \frac{L_q}{L_d}N\omega_m i_q \\ -\frac{R_s}{L_q}i_q - \frac{L_d}{L_q}N\omega_m i_d - \frac{\lambda_m N\omega_m}{L_q} \\ 0 \\ \omega_m \end{bmatrix}, \quad \mathbf{B}_c = \begin{bmatrix} \frac{1}{L_d} & 0 \\ 0 & \frac{1}{L_q} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.4}$$

As discussed in section 2.5, the extended Kalman filter linearizes the nonlinear model about the current estimate. Now that the plant model is defined in equation 3.4, it is necessary to calculate the Jacobian matrix in order to find the linearized state transition matrix $\mathbf{F}_{k-1}$ to be used in the Kalman filter:

$$\mathbf{F}_c = \left.\frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}}\right|_{x=x_{k-1}} = \begin{bmatrix} \frac{\partial f_1}{\partial i_d} & \frac{\partial f_1}{\partial i_q} & \frac{\partial f_1}{\partial \omega_m} & \frac{\partial f_1}{\partial \theta_m} \\ \frac{\partial f_2}{\partial i_d} & \frac{\partial f_2}{\partial i_q} & \frac{\partial f_2}{\partial \omega_m} & \frac{\partial f_2}{\partial \theta_m} \\ \frac{\partial f_3}{\partial i_d} & \frac{\partial f_3}{\partial i_q} & \frac{\partial f_3}{\partial \omega_m} & \frac{\partial f_3}{\partial \theta_m} \\ \frac{\partial f_4}{\partial i_d} & \frac{\partial f_4}{\partial i_q} & \frac{\partial f_4}{\partial \omega_m} & \frac{\partial f_4}{\partial \theta_m} \end{bmatrix} = \begin{bmatrix} -\frac{R_s}{L_d} & \frac{L_q}{L_d}N\omega_m & \frac{L_q}{L_d}Ni_q & 0 \\ -\frac{L_d}{L_q}N\omega_m & -\frac{R_s}{L_q} & -\frac{L_d}{L_q}Ni_d - \frac{\lambda_m N}{L_q} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{3.5}$$

Using Euler discretization from section 2.3.2, the state transition matrix $\mathbf{F_{k-1}}$, the input matrix $\mathbf{B}$ and the output matrix $\mathbf{H}$ is defined as:

$$\mathbf{F_{k-1}} = e^{\mathbf{F}_c T} \approx I + \mathbf{F_c} T = \begin{bmatrix} 1 - T\frac{R_s}{L_d} & T\frac{L_q}{L_d} N\omega_m & T\frac{L_q}{L_d} N i_q & 0 \\ -T\frac{L_d}{L_q} N\omega_m & 1 - T\frac{R_s}{L_q} & -T\frac{L_d}{L_q} N i_d - T\frac{\lambda_m N}{L_q} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & T & 1 \end{bmatrix} \tag{3.6}$$

$$\mathbf{B} = T\mathbf{B_c} = \begin{bmatrix} \frac{T}{L_d} & 0 \\ 0 & \frac{T}{L_q} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{3.7}$$

$$\mathbf{H} = \mathbf{H_c} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{3.8}$$

Furthermore, it is necessary to define the matrices $\mathbf{Q}$, $\mathbf{R}$ and $\mathbf{P}_0$. For simplicity, it is assumed that the matrices are diagonal:

$$\mathbf{Q} = \begin{bmatrix} Q_1 & 0 & 0 & 0 \\ 0 & Q_2 & 0 & 0 \\ 0 & 0 & Q_3 & 0 \\ 0 & 0 & 0 & Q_4 \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} R_1 & 0 \\ 0 & R_2 \end{bmatrix}, \quad \mathbf{P}_0 = \begin{bmatrix} P_1 & 0 & 0 & 0 \\ 0 & P_2 & 0 & 0 \\ 0 & 0 & P_3 & 0 \\ 0 & 0 & 0 & P_4 \end{bmatrix} \tag{3.9}$$

Adding of Hall sensors

If Hall sensors are used in the Kalman filter, then the output matrix $\mathbf{H}$ and measurement noise covariance matrix $\mathbf{R}$ will change from time to time as the Hall sensors do not continuously supply the Kalman filter with new measurements. When Hall pulses are registered in the Kalman filter, the matrices become:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{R} = \begin{bmatrix} R_1 & 0 & 0 & 0 \\ 0 & R_2 & 0 & 0 \\ 0 & 0 & R_3 & 0 \\ 0 & 0 & 0 & R_4 \end{bmatrix}$$

Between the Hall sensor pulses, the Kalman filter will only use current measurements in the update-step so that the matrices are reduced to:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad \mathbf{R} = \begin{bmatrix} R_1 & 0 \\ 0 & R_2 \end{bmatrix}$$

Implementing the Extended Kalman Filter

Finally, the algorithm defined in equations 3.10 and 3.11 is implemented in Simulink using MATLAB blocks:

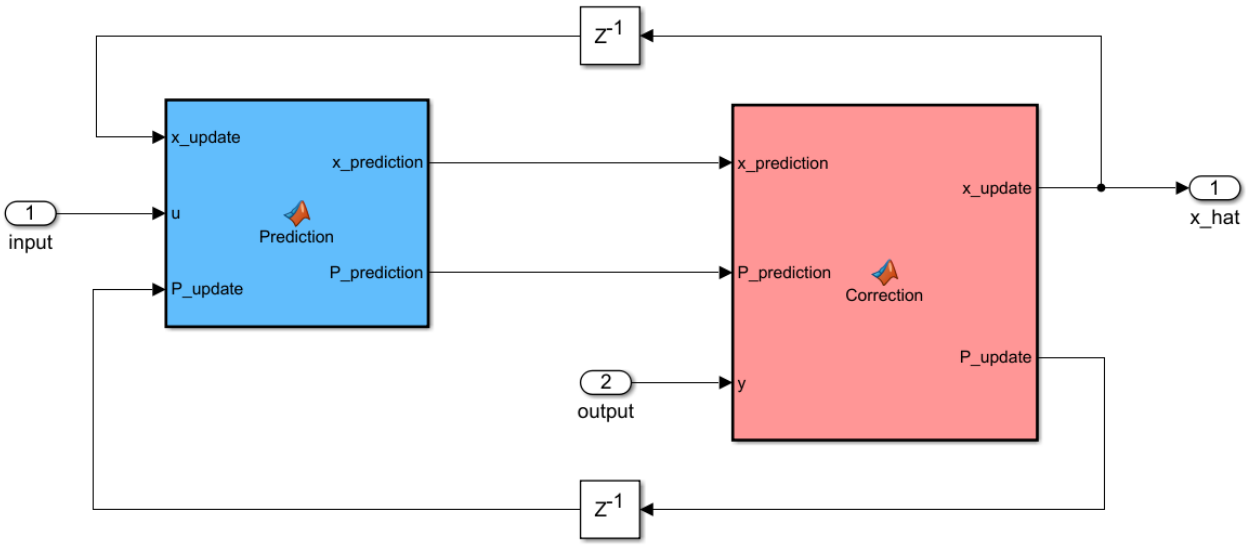


Figure 3.10: The Extended Kalman filter in Simulink

Prediction step:

$$\hat{\mathbf{x}}_k^- = \mathbf{f}(\tilde{\mathbf{x}}_{\mathbf{k-1}}^+, \mathbf{u_k}) \tag{3.10a}$$

$$\mathbf{P_k^-} = \mathbf{F_{k-1}} \mathbf{P_{k-1}^+} \mathbf{F_{k-1}^T} + \mathbf{Q} \tag{3.10b}$$

Update step:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_k^-) \tag{3.11a}$$

$$\mathbf{K}_k = \mathbf{P_k}^- \mathbf{H_k^T} (\mathbf{R} + \mathbf{H_k} \mathbf{P_k^-} \mathbf{H_k^T})^{-1} \tag{3.11b}$$

$$\hat{\mathbf{x}}_\mathbf{k}^+ = \hat{\mathbf{x}}_\mathbf{k}^- + \mathbf{K_k} \tilde{\mathbf{y}}_\mathbf{k} \tag{3.11c}$$

$$\mathbf{P_k^+} = (\mathbf{I} - \mathbf{K_k} \mathbf{H_k}) \mathbf{P_k^-} \tag{3.11d}$$

The initial state $\mathbf{x}_0$ and covariance matrix $\mathbf{P}_0$ are set by defining the initial conditions for each of the unit delay blocks.

3.5 Modelling the Hall effect sensors

A cost-effective way to obtain information about the rotor position is to use Hall sensors integrated in the stator of the motor which can then be used to detect the rotor's permanent magnets. In that way, the Hall sensors can be used to both detect the position of the rotor as well as estimating the speed. Such Hall effect sensors are implemented by comparing the rotor angle to the angles where the Hall sensors are mounted. Once the rotor is close to a Hall sensor, $|\theta_e - \theta_{hall}| < \alpha$, a boolean signal is set high.

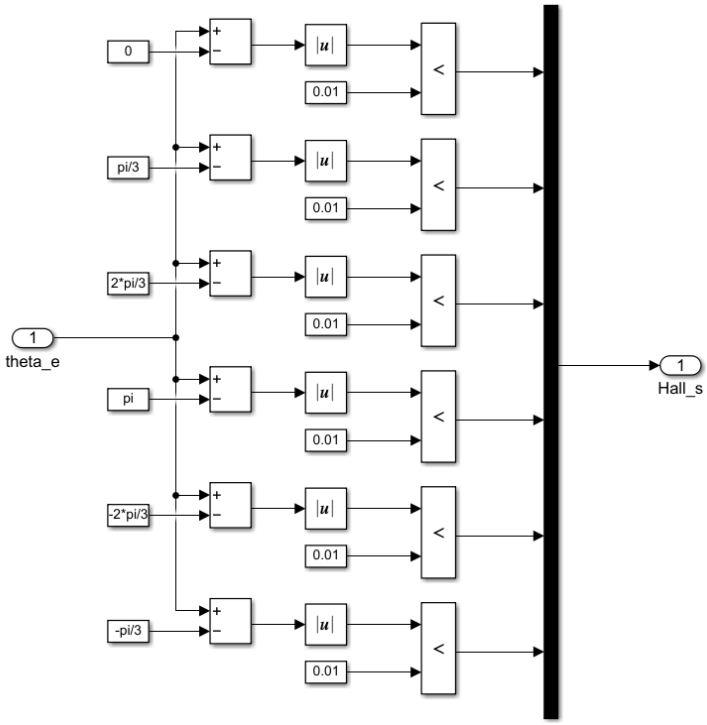


Figure 3.11: Six Hall sensors implemented in Simulink

In the figure above there is mounted six Hall sensors. Given that $\theta_e \, \epsilon \, [-\pi, \pi]$, this means that the six Hall sensors must be mounted at:

$$\theta_{hall} = \begin{bmatrix} 0 & \frac{\pi}{3} & \frac{2\pi}{3} & \pi & \frac{-2\pi}{3} & \frac{-\pi}{3} \end{bmatrix}.$$

The value of α is important because it gives the Hall sensors (simulink) enough time to detect the rotor position. The value of α depends on the characteristics of the Hall senor, which should be selected based on the nominal rotor speed. A high nominal rotor speed requires a large α and vice versa. By using a too small α, neither of the Hall sensors will work and by using a too large α, one will receive inaccurate measurements. In figure 3.11, the value of α was set to 0.01.

Speed Estimator based on Hall sensors

If Hall sensors are used to detect the position of the rotor, then it can also be useful to use the Hall sensors to estimate the rotor speed. This can be achieved by measuring the time between each Hall sensor pulse. Given that we know the distance the rotor must rotate before a new Hall sensor pulse occurs, the rotor speed can be estimated according to the following equation:

$$\hat{\omega}_m = \frac{\theta_h}{N \cdot t_h} \tag{3.12}$$

where $\hat{\omega}_m$ is the estimated mechanical rotor speed in radians per second, θ_h is the electrical angle between each Hall sensor in radians, N is the number of pole pairs and t_h is the measured time between each Hall sensor pulse.

The accuracy of the speed estimation depends on how many Hall sensors are used, meaning that one has to prioritize either accuracy or cost. In any case, at least two Hall sensors should be used so that you get both precise speed estimates and the ability to determine the direction the rotor is rotating. The implementation of a speed estimator using two Hall sensors are shown below:

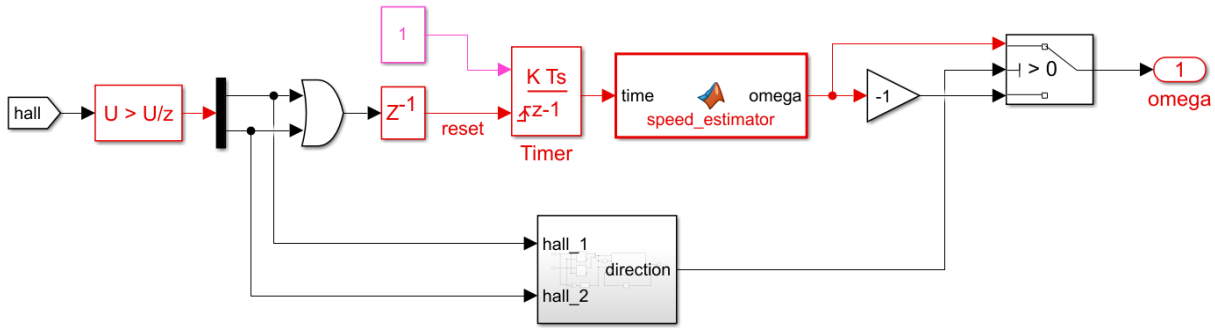


Figure 3.12: Simulink implementation of speed estimator based on Hall sensors

The speed estimator works in the way that one uses a timer that is reset after each Hall sensor pulse. Just before the timer is reset, the measured time is transferred to the MATLAB function *speed estimator* where the speed is estimated according to equation 3.12.

The direction the rotor rotates in is determined by the simulink blocks shown below:

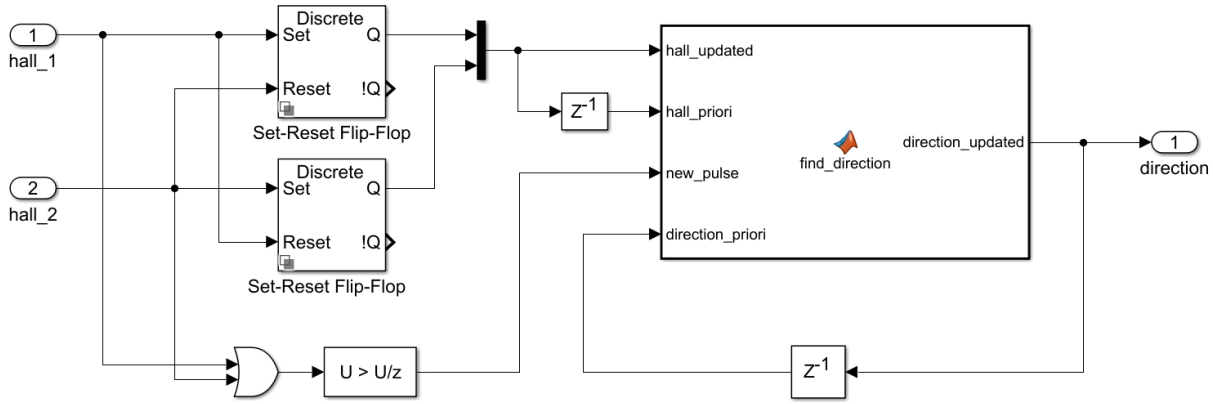


Figure 3.13: Logic used in order to determine rotational direction

Here, S-R latches are used in order to remember which Hall sensor was last active. In this way, you can determine the direction of rotation by checking if **hall 1** leads or lags **hall 2** or if the same Hall sensor becomes active twice in a row. By using more than two Hall sensors, it becomes even easier to determine the direction of rotation as the direction is determined by which Hall sensors become active in succession.

Position Estimator based on Hall sensors

It is also possible to implement a continuous position estimator based on Hall sensors by integrating the speed estimate. In this estimator, the position will be updated at each Hall sensor pulse but between the pulses the position is determined by the integrator. The following equation describe how the electrical rotor position is estimated between the Hall pulses:

$$\hat{\theta}_e^+ = \hat{\theta}_e^- + \hat{\omega}_e \cdot T_s \tag{3.13}$$

where $\hat{\theta}_e^+$ is the updated position estimate, $\hat{\theta}_e^-$ is the last position estimate, $\hat{\omega}_e$ is the speed estimate and T_s is the sample time.

Chapter 4

Simulation and results

Chapter 4 describes the results obtained from the simulations that have been carried out in this thesis. The simulation of the system is meant to reflect typical motor operations of a PMSM that is used to drive a drone propeller. This includes starting up the motor, running the motor at a given speed and locking the propeller into a fixed position when it is not in use. Firstly, an encoder was used as feedback to control the motor and later estimators using techniques such as nonlinear observers, HFI and EKF were used in an attempt to replace the encoder.

Parameters of the physical system

Table 4.1: Parameters used in the modelling of the Alva motor

Symbol	Parameter	Value	Unit
P_p	Peak power	3.1	kW
V_n	Nominal voltage	44	V
N	Pole pairs	14	
K_e	Back EMF constant (Y-wound)	8.03	mV/rpm
R_s	Phase resistance	19.9	mΩ
L_d	Direct axis inductance	2.64	µH
L_q	Quadrature axis inductance	2.64	µH
J_m	Rotor inertia	0.01	kgm^2
$C_{Q,0}$	Parametrization parameter of torque	0.0078	
$C_{Q,1}$	Parametrization parameter of torque	−0.0058	
D	Propeller size (diameter)	40	cm
V_a	Air speed (assumed constant)	20	m/s
ρ	Air density	1.225	kg/m^3
T_s	Fundamental sample time in Simulink	5	µs
f_{sw}	Switching frequency of inverter	20	kHz

4.1 Simulation using Encoder Feedback

The plots below show the performance of the control system while using direct encoder measurements as feedback.

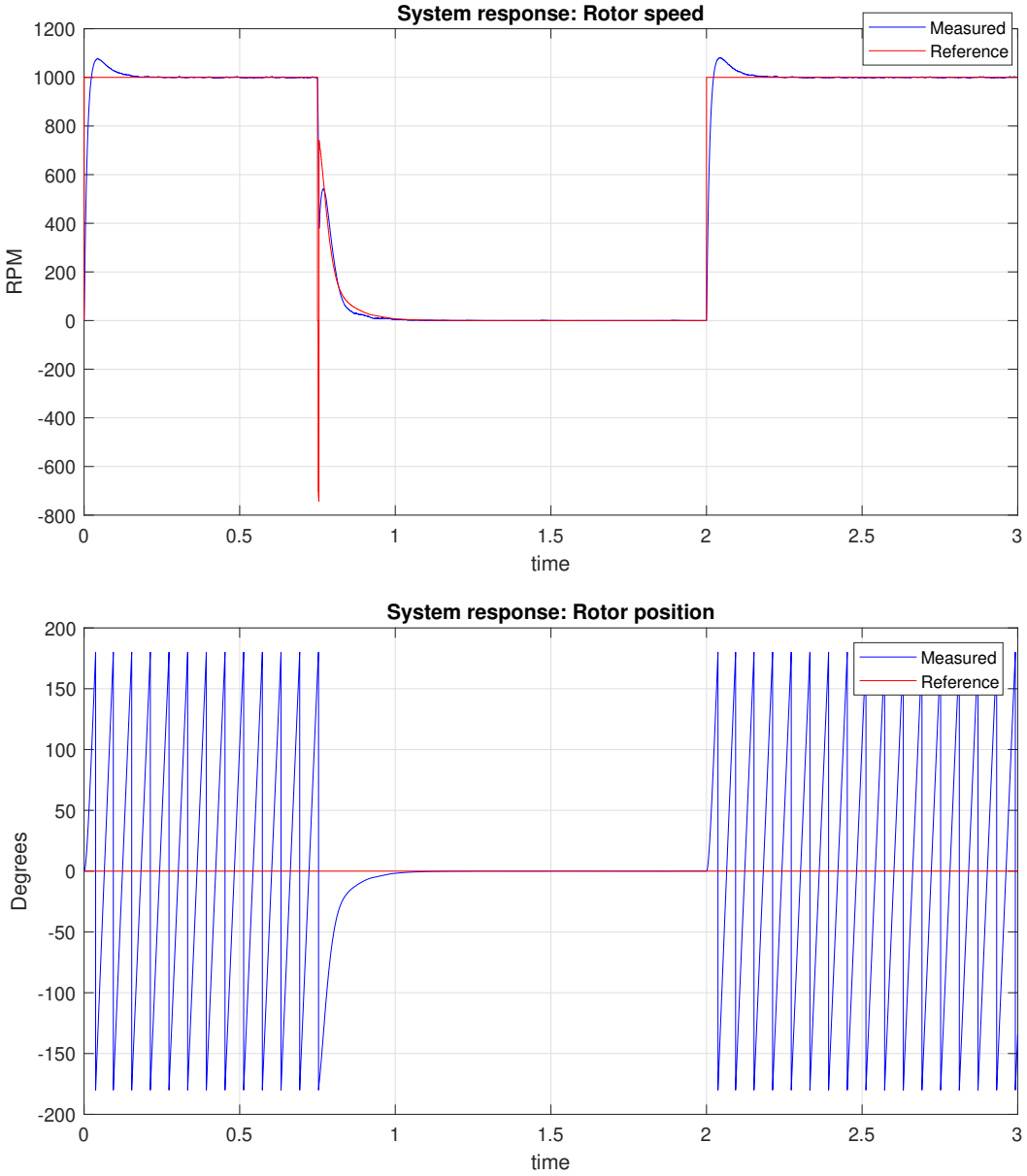


Figure 4.1: System response using encoder measurements

From figure 4.1 it can be seen that the speed controller works well with a fast response and little overshoot when the speed reference jumps from 0 to 1000 rpm. After 750 milliseconds, the motor is no longer needed and it is desired to control the rotor position so that it remains fixed at zero degrees. The controller then switches from speed control to position control with a refer-

ence of zero degrees. After approximately 250 milliseconds the reference is met, and the rotor is kept at zero degrees until the control system switches back to speed control at 2000 milliseconds.

It can be pointed out that the speed reference shortly jumps to -800 rpm due to the discontinuity that occurs when the angle has completed a full rotation. This has however little to say for the overall performance of the control system but could have been removed by slowing the motor down before switching to position control such that no angle discontinuities occur at this time or by implementing the method of *smallest signed angle* which finds the shortest distance between two angles (this method was unfortunately discovered after all the plots were made and therefore not included).

The tuning parameters of the control system used to achieve the desired system response is listed below:

Table 4.2: Parameters used to tune the control system

Symbol	Parameter	Value
K_{cp}	Proportional gain for inner current loop	1
K_{vp}	Proportional gain for outer speed loop	10
K_{vi}	Integrator gain for outer speed loop	200
K_{pp}	Proportional gain for outer position loop	25
T_{sc}	Sample time for current loop	10 µs
T_{sv}	Sample time for velocity loop	100 µs
T_{sp}	Sample time for position loop	1 ms

4.2 Simulation using Nonlinear Observer

The nonlinear observer from section 2.11 was simulated first in an attempt to replace the encoder. Before the observer could be used as feedback, it had several parameters that needed to be tuned so that the observer accurately estimates the measurements from the encoder. Using the control system described in the last section, the tuning process was completed by comparing the observer estimates to the measurements of the encoder until a satisfactory result was obtained.

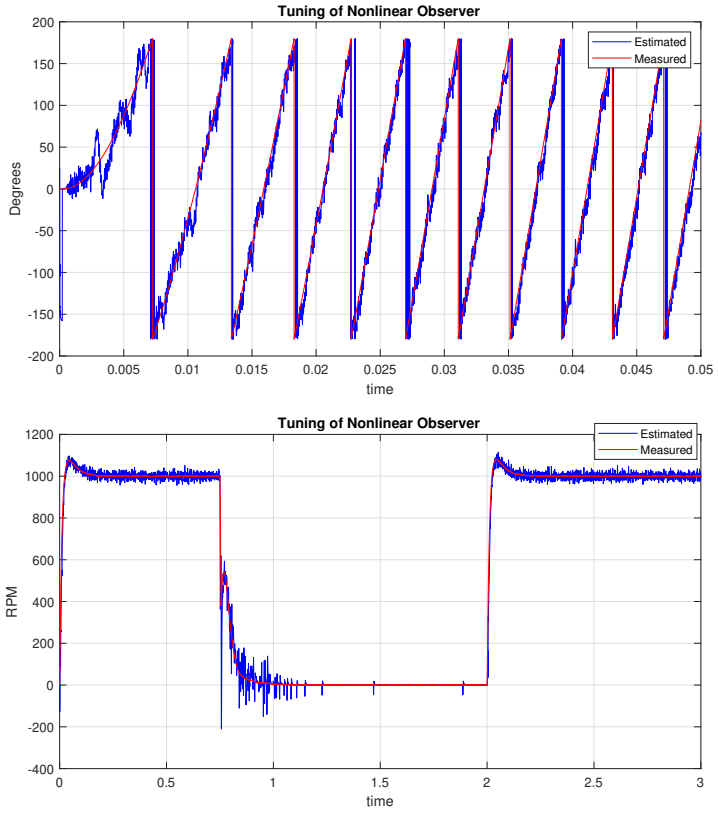


Figure 4.2: Estimation of rotor position and speed

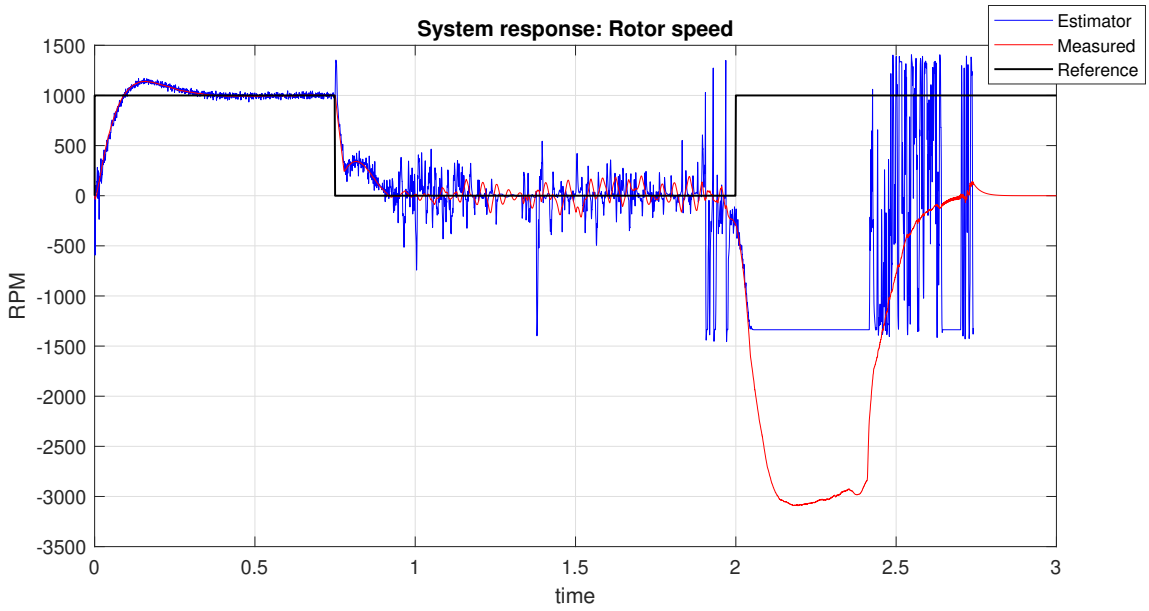
When increasing the observer gains, the estimates became more noisy and if they were reduced the estimates became more inaccurate. The tuning parameters of the observer was found to be optimal at:

Symbol	Parameter	Value
γ	Observer gain in theta estimation	10^9
K_p	Observer gain in omega estimation	500

From figure 4.2 it can be established that the observer was adequately tuned as it accurately estimates both the rotor position and speed with some noise present.

Simulation using Nonlinear Observer as feedback

The system was then simulated using the nonlinear observer as feedback instead of the encoder. The same control routine was performed and the results are shown in the figures below:



(a) Time axis is changed in the second figure to focus on the position estimate during standstill operations

Figure 4.3: Estimation of rotor speed and position

From figure 4.3 it can be seen that the nonlinear observers work well as feedback during high-speed operations, but fails as the speed gets close to zero during position control. This is expected as the model-based observers are dependent on the speed of the rotor, thus making them unsuitable for position control.

Replacing the Position Controller

In order to be able to lock the rotor/propeller in a fixed position while using the nonlinear observers as feedback, a new control algorithm had to be implemented to replace the position controller. By mounting a Hall sensor in the direction it is desired to lock the propeller, the rotor can be locked by applying a constant voltage to the stator windings once the Hall sensor detects the rotor. The simulation results using this control algorithm is plotted below:

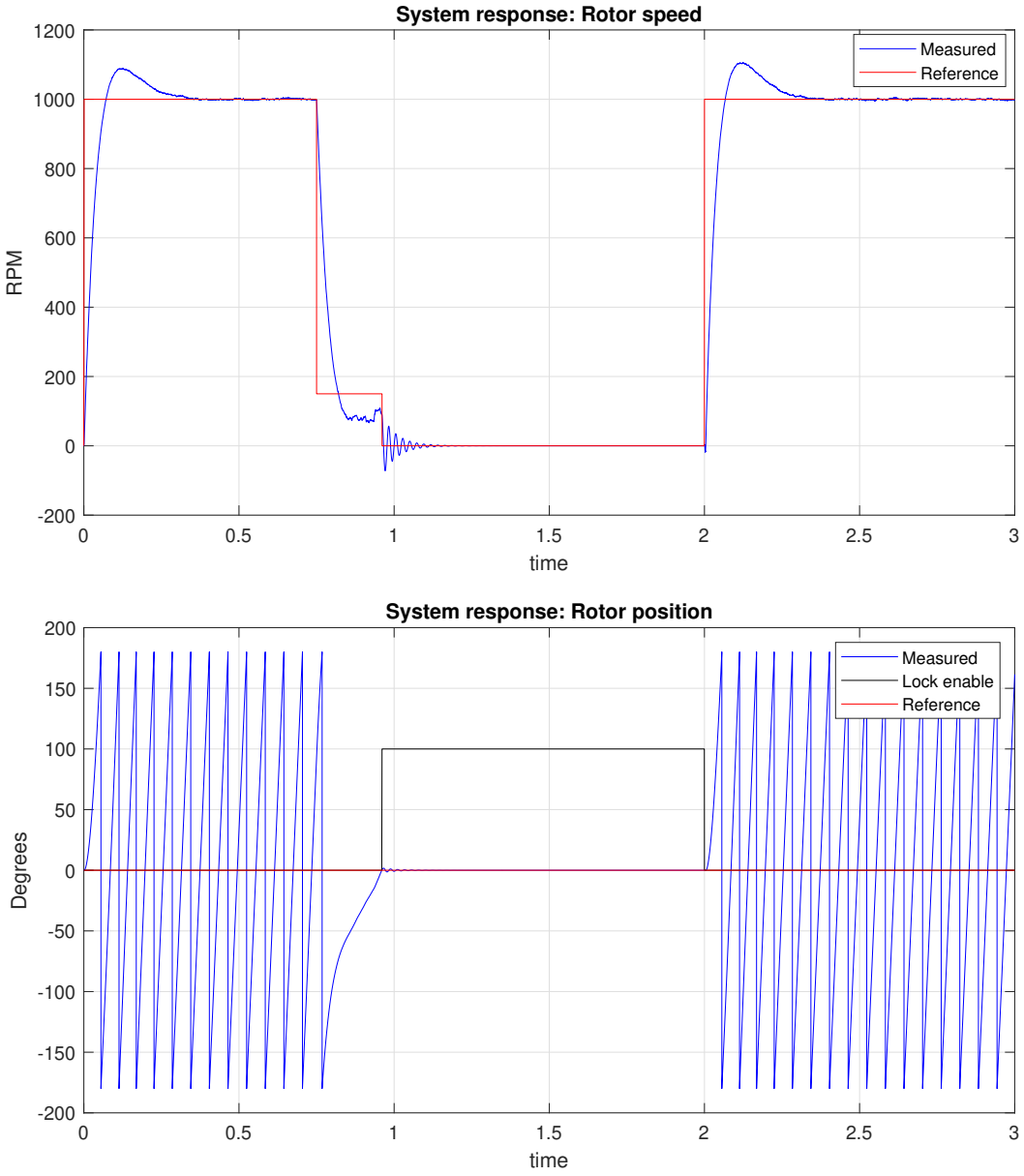


Figure 4.4: Estimation of rotor position and speed

From figure 4.4 it can be seen that the system response is still fast and accurate with an overshoot of approximately 8% and a settling time of 300 milliseconds while using estimators as feedback.

From around 750 milliseconds, it is desired to lock the rotor into a fixed position. The speed controller is then used to slow down the rotor as much as possible by using the lowest speed reference the nonlinear observer would allow (175 rpm) in order to reduce mechanical stress on the motor and ease the process of locking the rotor into the correct position. At around 950 milliseconds, the Hall sensor detects the rotor at zero degrees and the rotor is locked into position with no offset. At 2000 milliseconds, the motor is needed again and the control system switches back to speed control and operates normally.

The tuning parameters of the control system used to achieve the system response in figure 4.4 is listed below:

Table 4.3: Parameters used to tune control system using nonlinear observer as feedback

Symbol	Parameter	Value
K_{cp}	Proportional gain for inner current loop	1
K_{vp}	Proportional gain for outer velocity loop	3
K_{vi}	Integrator gain for outer velocity loop	35
T_{sc}	Sample time for current loop	10 µs
T_{sv}	Sample time for velocity loop	100 µs

4.3 Simulation using HFI Observer

The sensorless control method implemented in the last section satisfied the requirements set in the thesis regarding control of a drone propeller. However, there are some clear drawbacks to the rotor-locking control algorithm which it is desirable to improve. Firstly, the instantaneous locking of the rotor from driving speed may cause mechanical stress on the motor which ultimately may shorten the life span of the motor. Secondly, there are no ways of knowing whether or not the rotor locked into the correct position as the control algorithm is based on *open loop*-control where no feedback is used in the control system. This leads to cases where the constant locking voltage may be too low so that the rotor does not lock into the desired position, or that the voltage is too high so that unnecessary power is used to lock the rotor.

In order to improve the rotor-locking of the propeller, the low-speed HFI observer from section 2.12 was introduced to the control system in an effort to lock the rotor by using *closed loop*-position control instead. Given that the HFI observer should be able to obtain the position of the rotor for low- and zero speeds, it was expected a similar result from the control system as in figure 4.1 when encoder feedback was used.

Before the HFI observer could be used as feedback in the control system, several parameters had to be determined first. Again, these parameters are tuned while using encoder feedback during the tuning process. Using the following parameters during the tuning process, the estimation results over the next pages were obtained:

Table 4.4: Parameters used to tune HFI observer

Symbol	Parameter	Value
V_h	Amplitude of the injected voltage	20 V
ω_h	Frequency of injected voltage	500 Hz
ω_{lp}	Natural frequency of low pass filter	250 Hz

The rotor position is estimated in the HFI observer by the following equation:

$$\hat{\theta}_{re} = -\tan^{-1}\left(\frac{\tilde{i}_{\beta,h}}{\tilde{i}_{\alpha,h}}\right) \tag{4.1}$$

where $\tilde{i}_{\alpha,h}$ and $\tilde{i}_{\beta,h}$ are the envelopes of the high frequency current components $i_{\alpha,h}$ and $i_{\beta,h}$. The envelopes were extracted from the high frequency currents as shown below:

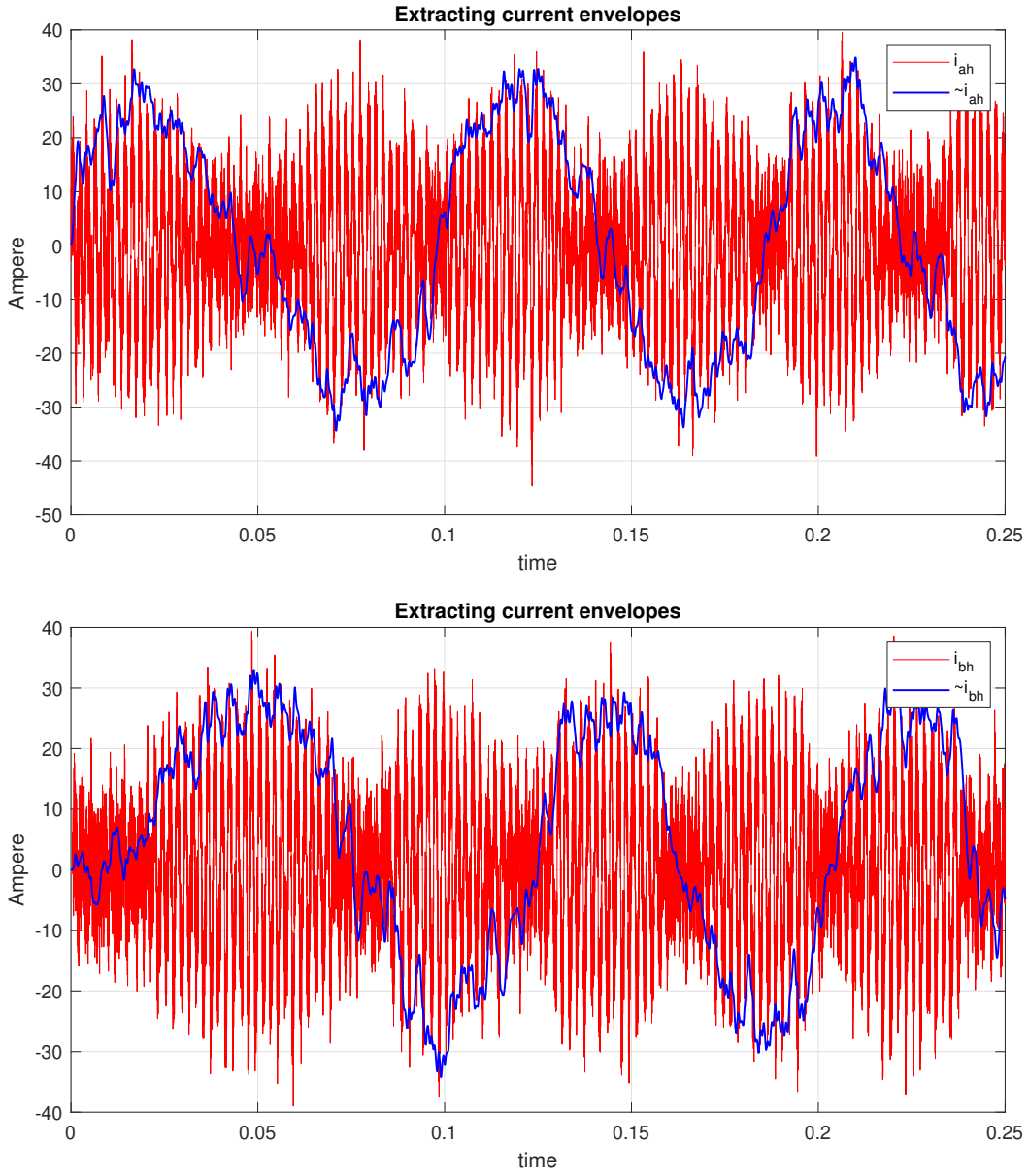


Figure 4.5: Envelope extraction from high frequency current components

Finally, the electrical position of the rotor was estimated by taking the inverse tangent of the current envelopes as described in equation 4.1:

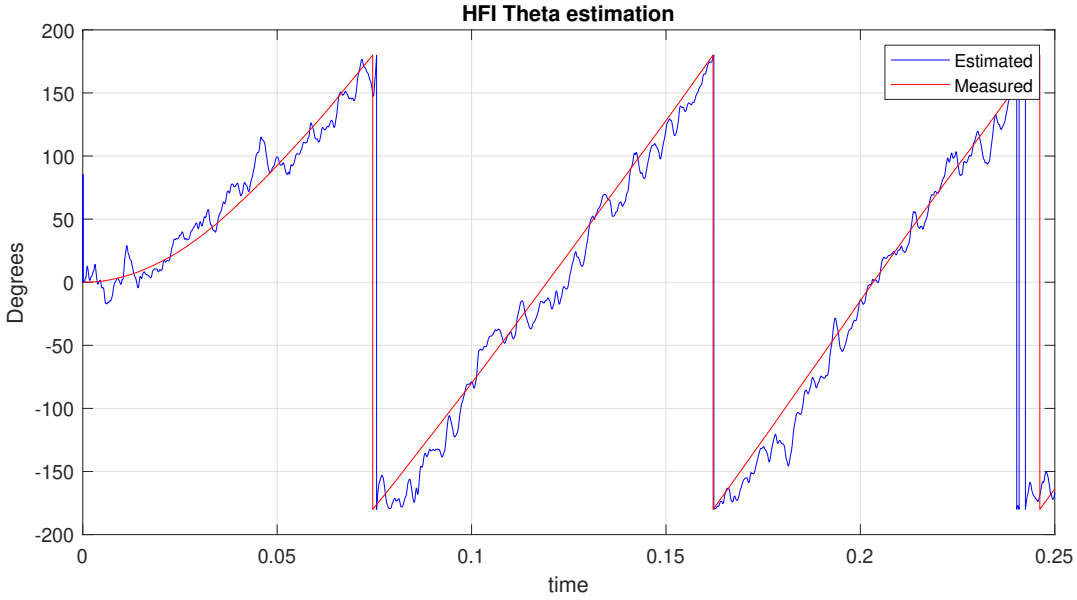


Figure 4.6: Estimation of rotor position using HFI

In order to estimate the rotor speed during low-speed operations, the same speed observer that was used for the nonlinear observer was implemented. This gave the following results:

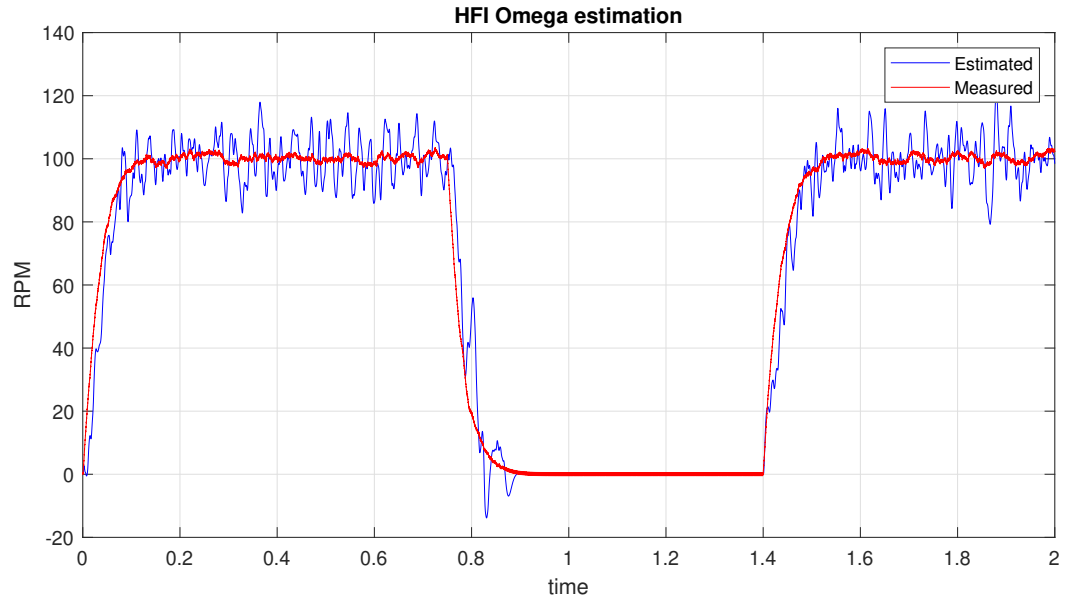


Figure 4.7: Estimation of rotor speed using HFI

Simulation using HFI observer as feedback

From the figures given in the last section, it could be confirmed that the HFI observer accurately estimated both the rotor position and speed and was ready to be used as feedback in the control system.

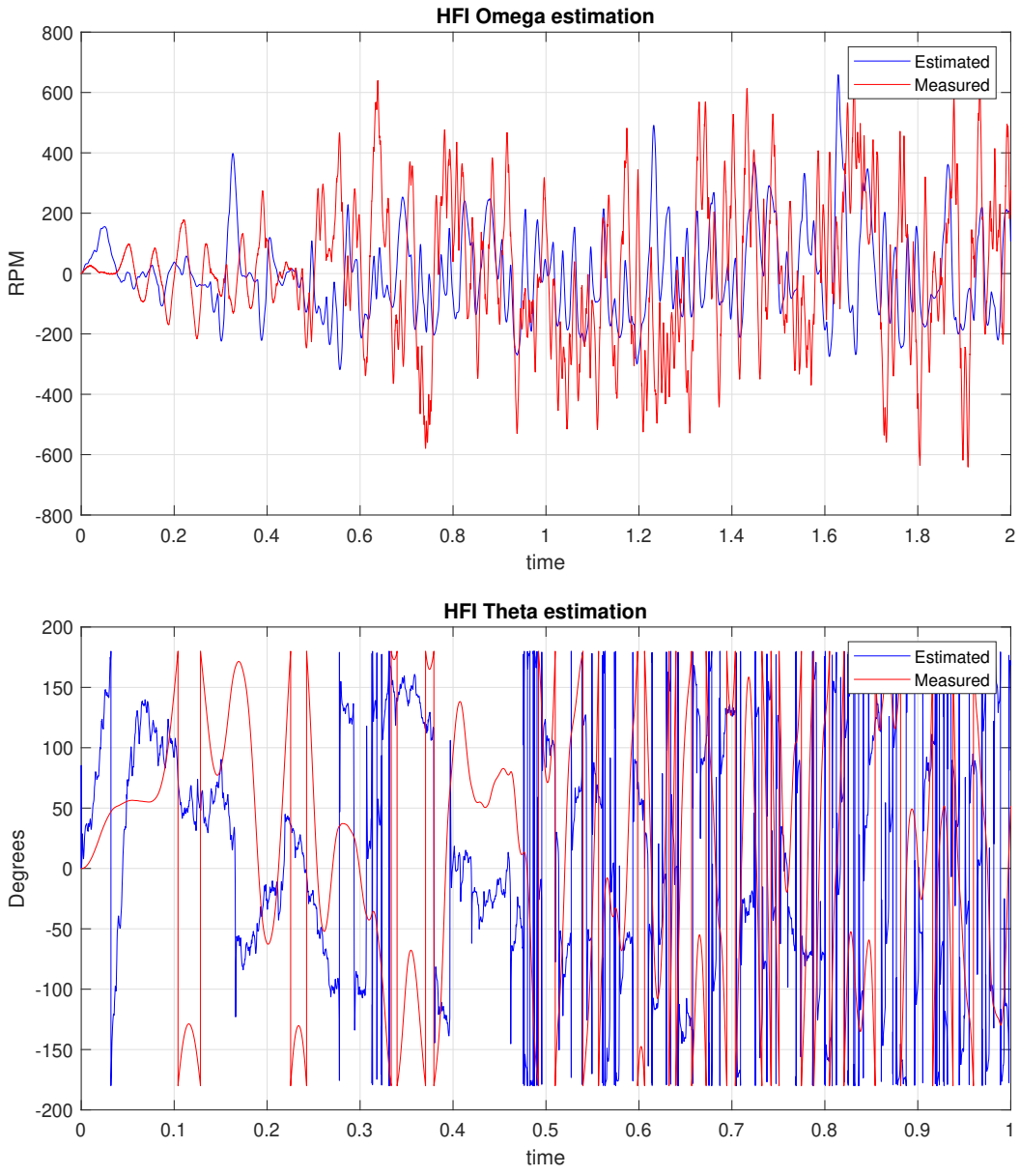


Figure 4.8: Estimation of rotor position and speed during observer feedback

From figure 4.8 it appears that the observer is not able to estimate the rotor position or speed while being used as feedback in the control system. By analyzing the plots of the high frequency current components in figure 4.9, it can be seen that the envelopes no longer have a sinusoidal shape and contain a great amount of noise.

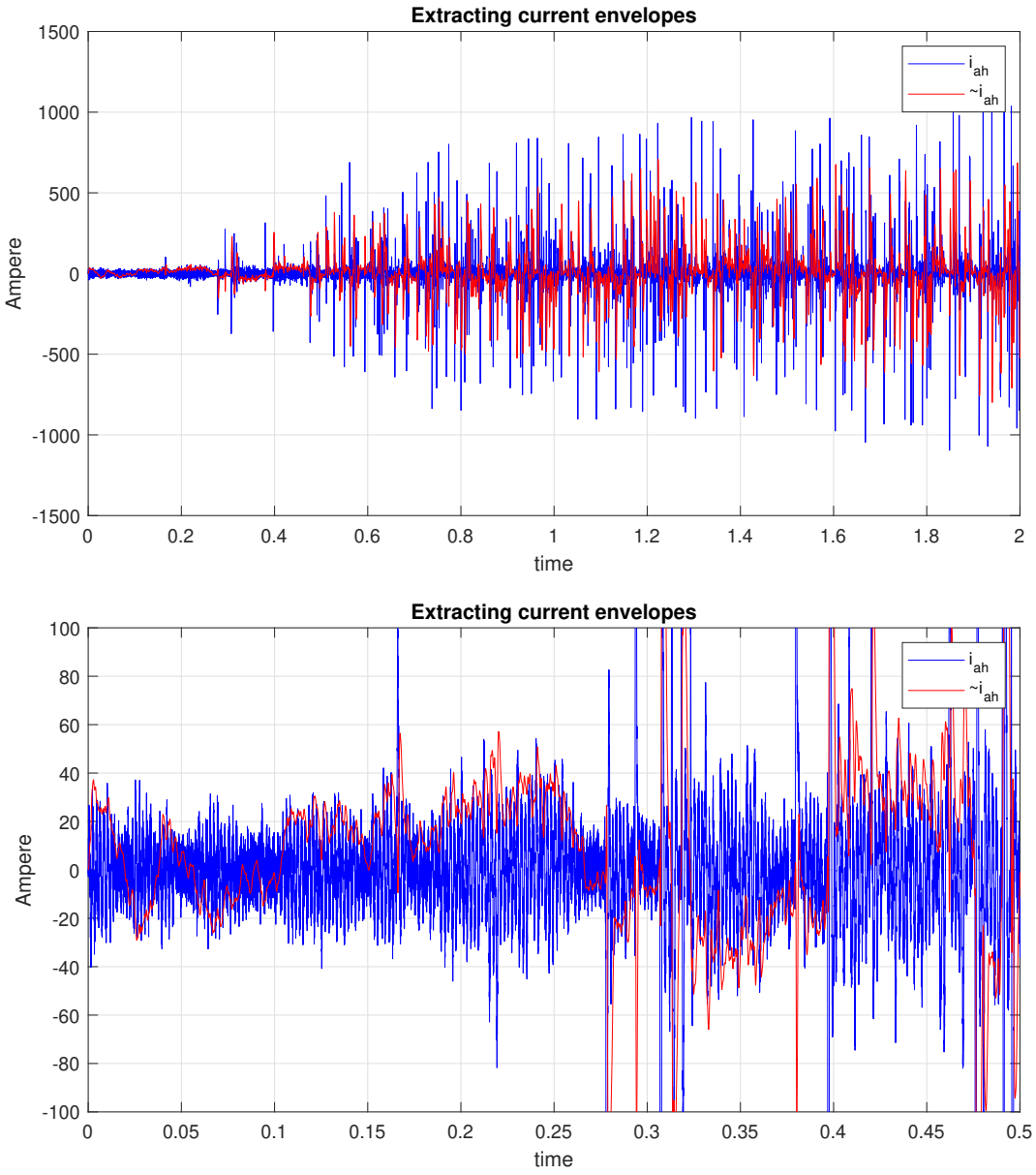


Figure 4.9: Envelopes of the high frequency current components

From the figures above it seems that the envelope signal is lost in all the noise, which again means that the rotor position can not be estimated. In order to fix this issue, the parameters set in table 4.4 were changed multiple times over. Unfortunately, neither of the changes ended up yielding a different result. Ultimately, it was tried to simulate the HFI observer on a different motor in order to find out what the problem might be.

Simulation using HFI Observer as feedback for a different PMSM

The template that was acquired from MathWorks already had a predefined PMSM which made it easy to simulate motor drives without having to know any specific motor parameters. By using these predefined parameters, the HFI technique from the last section could be tested on a different motor.

Table 4.5: Parameters for the predefined PMSM

Symbol	Parameter	Value	Unit
P_p	Peak power	14	kW
V_n	Nominal voltage	44	V
N	Pole pairs	2	
ψ_m	Permanent magnet flux linkage	0.04	Wb
R_s	Phase resistance	5	mΩ
L_d	Direct axis inductance	100	μH
L_q	Quadrature axis inductance	300	μH
J_m	Rotor inertia	0.01	kgm^2
$C_{Q,0}$	Parametrization parameter of torque	0.0078	
$C_{Q,1}$	Parametrization parameter of torque	−0.0058	
D	Propeller size (diameter)	40	cm
V_a	Air speed (assumed constant)	20	m/s
ρ	Air density	1.225	kg/m^3
T_s	Fundamental sample time in Simulink	5	μs
f_{sw}	Switching frequency of inverter	20	kHz

The HFI observer was also adjusted to fit the predefined motor. The following observer parameters was found appropriate:

Table 4.6: Parameters used to tune HFI observer for the predefined PMSM

Symbol	Parameter	Value
V_h	Amplitude of the injected voltage	20 V
ω_h	Frequency of injected voltage	1000 Hz
ω_{lp}	Natural frequency of low pass filter	500 Hz

The results from using HFI observer feedback are shown below:

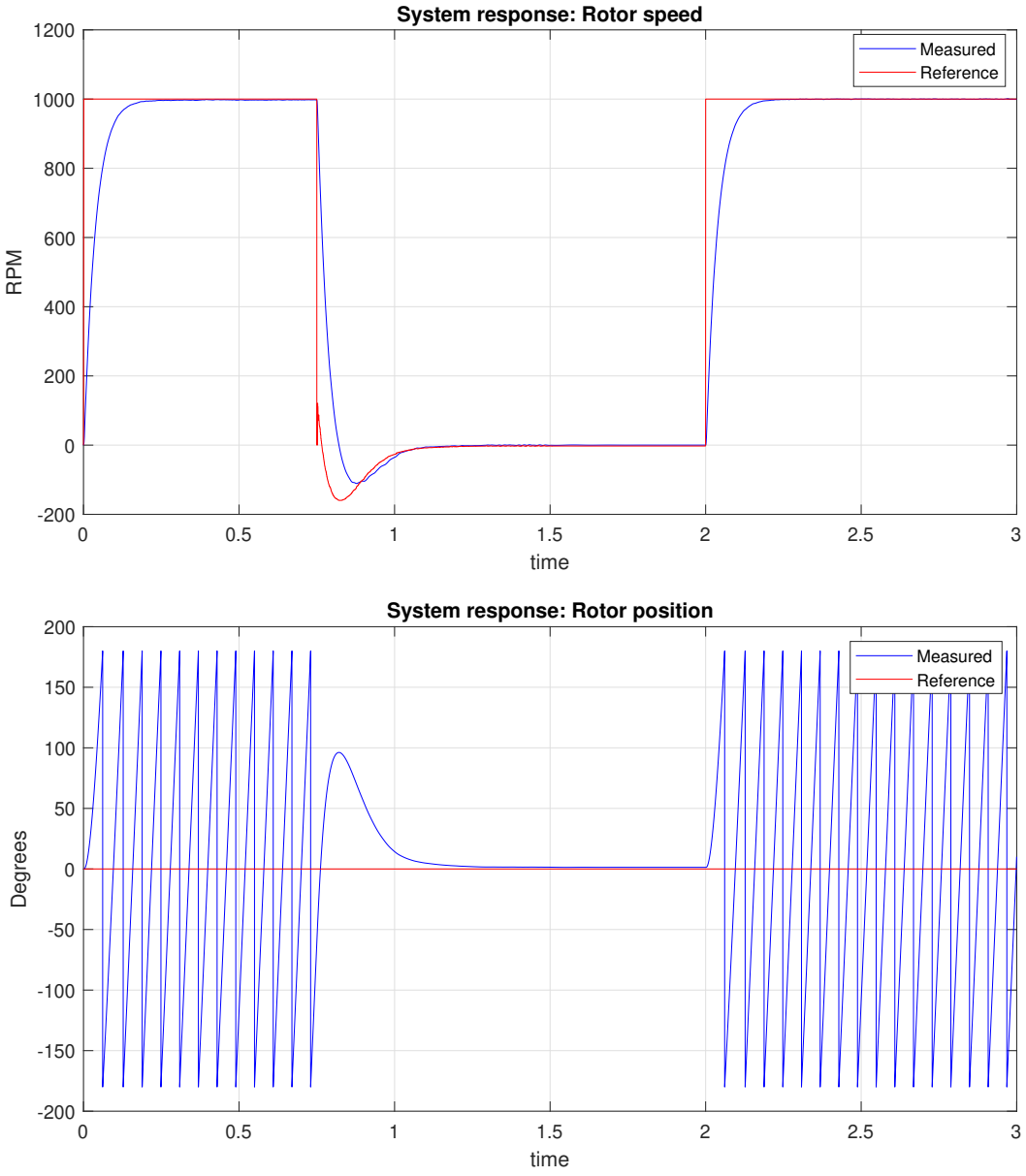


Figure 4.10: System response using HFI feedback

As seen in the figures above, both the speed and position-controller clearly works while using HFI observer feedback on another motor. The current envelopes and the position- and speed estimates from the HFI observer are plotted on the next couple of pages.

From figure 4.11 we see that the current envelopes remain sinusoidal while the HFI observer is being used as feedback:

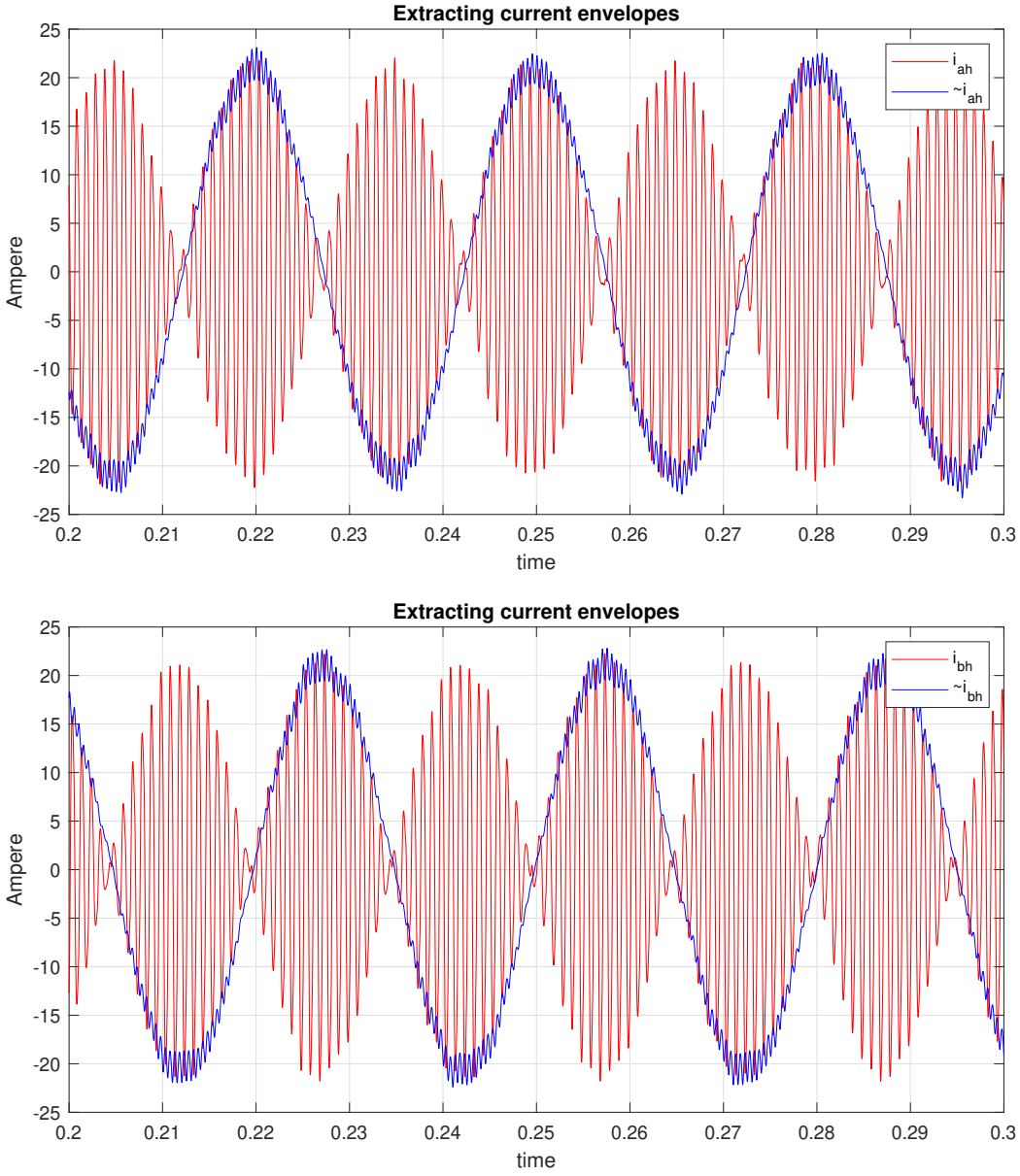


Figure 4.11: Envelopes of the high frequency current components

Having sinusoidal and well defined current envelopes gives rise to accurate position- and speed estimates as shown on next page.

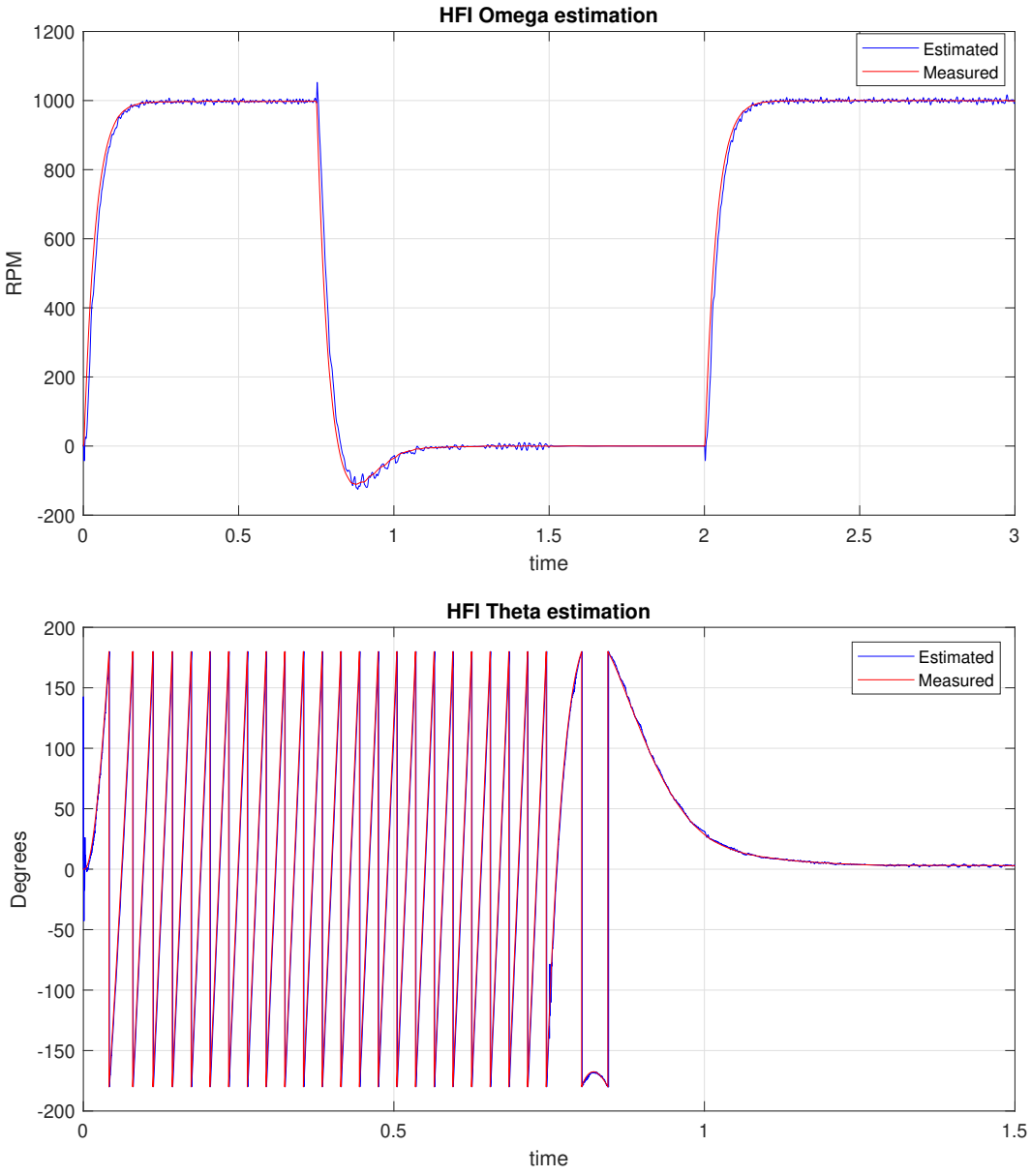


Figure 4.12: Estimation of rotor position and speed during observer feedback

From figure 4.12 it can be seen that the HFI technique works well for another motor. Besides the size difference and number of pole pairs, the only difference between the two motors is that Alva's motor is a SPMSM while the predefined motor is a IPMSM due to the fact that the direct- and quadrature axis inductances are not equal. In order to find out if the difference in direct- and quadrature inductance affects the HFI estimation, the system was simulated again using other values for the direct axis inductance.

By letting the direct axis inductance slowly approach the value of the quadrature axis inductance, it could be determined how the inductance difference affects the HFI estimation:

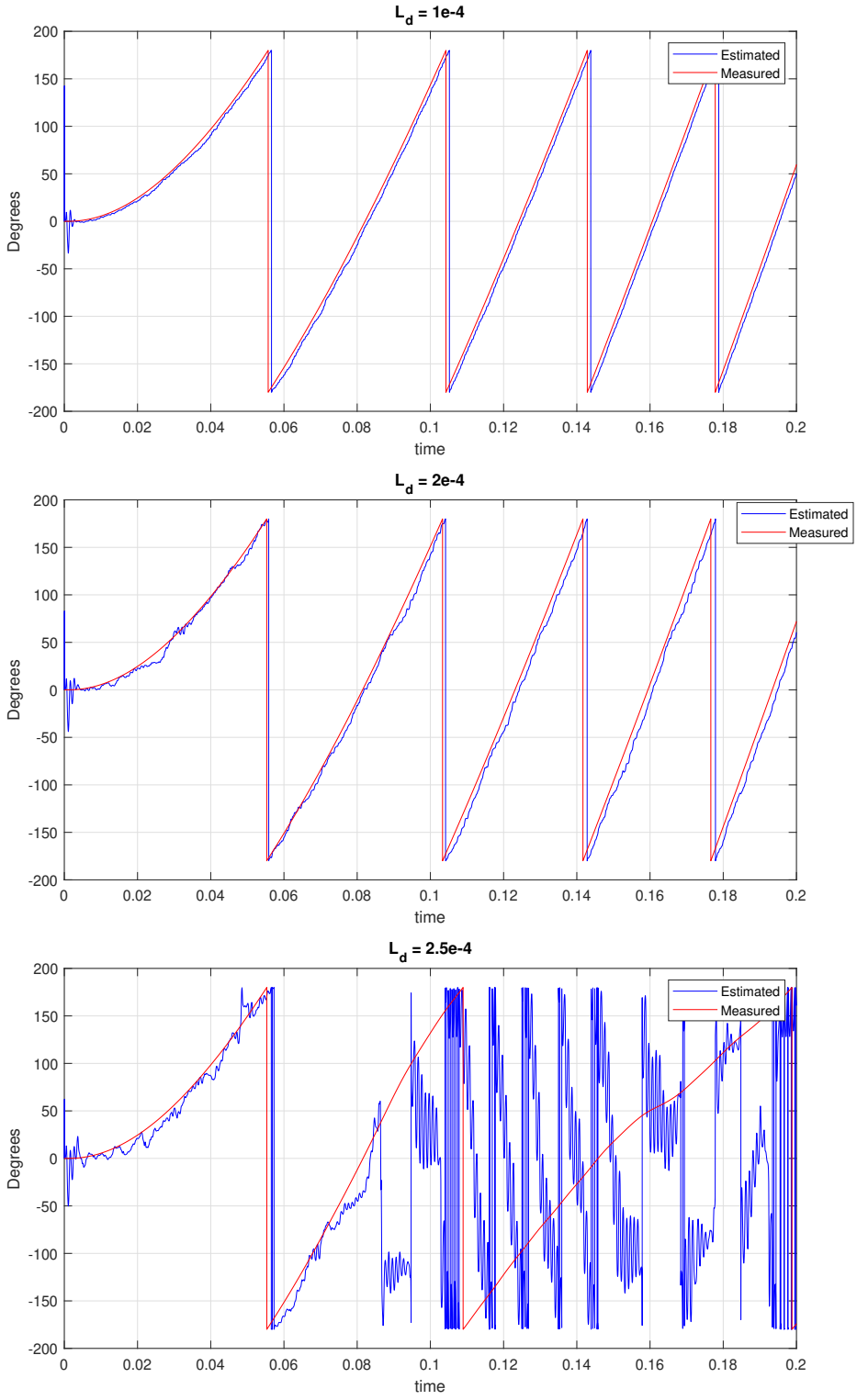


Figure 4.13: Estimation of rotor position for different values of L_d

From figures shown in 4.13, it can be seen that the accuracy of the HFI estimation decreases with decreasing inductance difference. Once the inductance difference ($L_q - L_d$) becomes less than 50 μH or 0.167 p.u, the observer becomes incapable of estimating the rotor position. The signal carrying the information of the rotor position from equation 2.66 is in fact dependent on the inductance difference, meaning that if the inductance difference is small then the position signal is easily lost to noise. This explains why the HFI observer worked fine for the motor designed by Alva Industries (SPMSM) during the tuning process in figure 4.7, but failed once the observer was used as feedback because of the introduced noise shown in figure 4.8.

Ultimately, this means that the results from the saliency independent HFI observer proposed by *Qiao et.al* [25] could not be replicated for the motor designed by Alva Industries due to small motor inductances.

4.4 Simulation using Hall Sensors

This far, sensorless methods involving both model-based and saliency-based observers have been used in order to control the motor during high and low-speed operations. However, only the model-based nonlinear observer along with a Hall sensor has yielded results. In the end, it seems that one has to abandon the idea of using sensorless methods in order to precisely control the motor designed by Alva Industries during low-speed operations.

Simulation using three Hall sensors

First, an attempt was made to obtain precise motor control during low-speed operations by using Hall sensors together with the integrator introduced in section 3.5 as feedback. By using three Hall sensors, a common configuration for *BLDC*-motors, the following results were obtained:

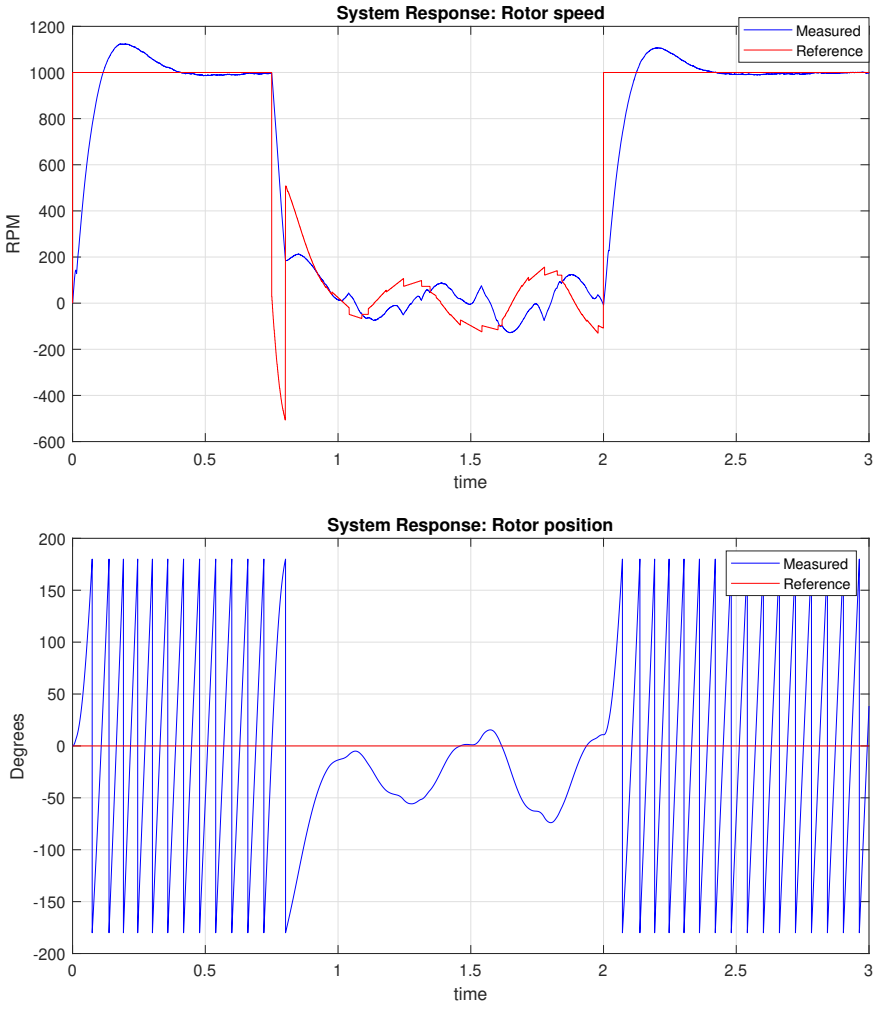


Figure 4.14: System response while using Hall sensor-based feedback

From figure 4.14 it can be seen that the Hall sensors work well for speed control, but that difficulties arise under position control. Due to the few pulses the Hall sensors provide at low speeds, the speed estimate becomes inaccurate which results in the rotor oscillating around the desired position with a maximum error of approximately 60°.

The estimated states provided by the Hall sensors compared to the measured states are plotted below:

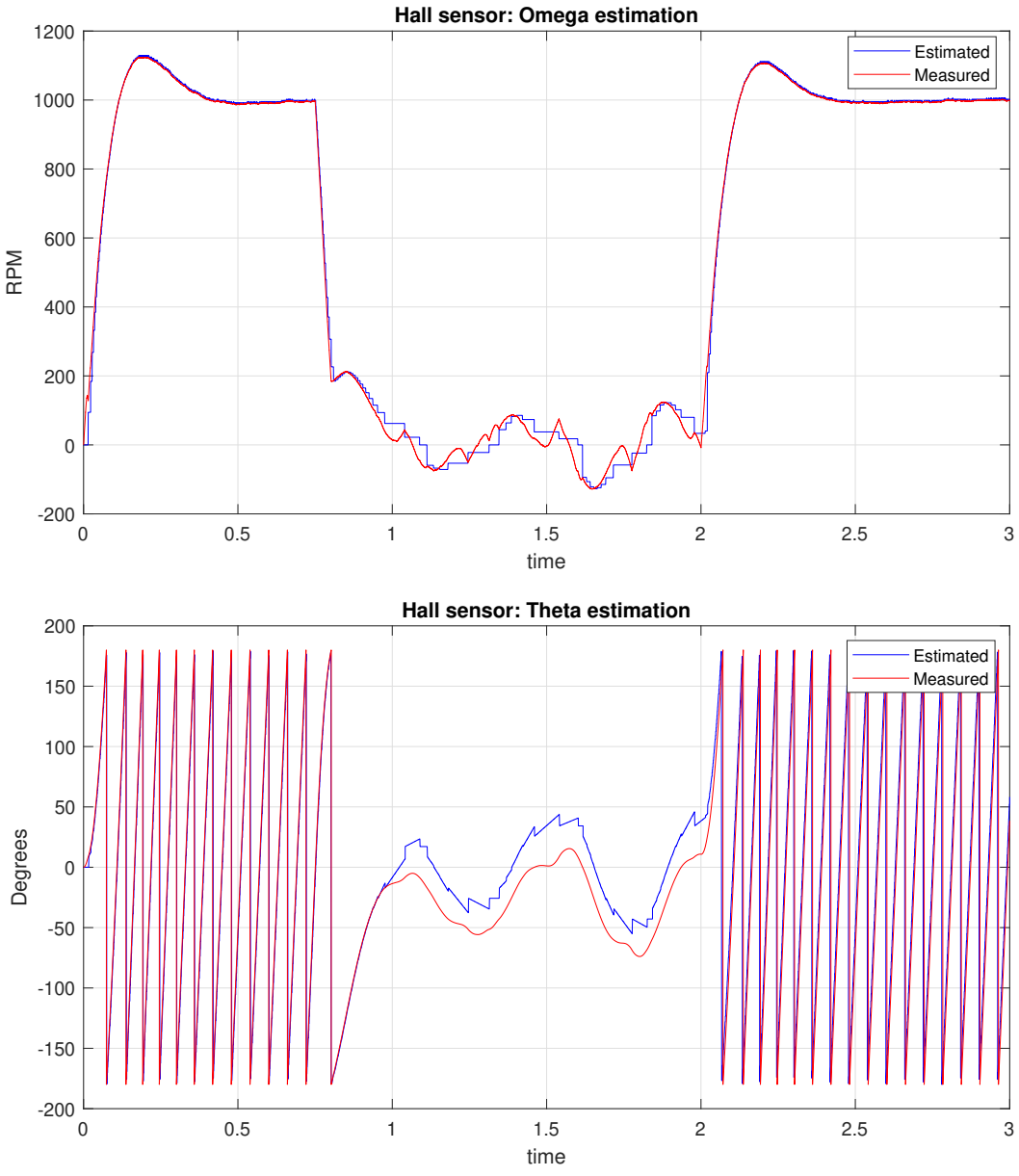


Figure 4.15: Estimation of rotor speed and position using Hall sensors

4.5 Simulation using Extended Kalman Filter and Hall sensors

When using Hall sensors and a integrator as feedback, we got a maximum error of approximately 60°. This is not necessarily good enough as 90°is the biggest error one can have when controlling the position of a propeller. One way to improve this is by combining the Hall sensors with an extended Kalman filter where current- and voltage measurements together with knowledge of the motor load are taken into account.

Simulation using EKF with a single Hall sensor

As with the other observers, the Extended Kalman filter also has to be tuned so that it fits the simulated system. This involves determining the inputs of the matrices $\mathbf{Q}$, $\mathbf{R}$ and $\mathbf{P_0}$. The system was simulated using encoder feedback in order to tune the Kalman filter:

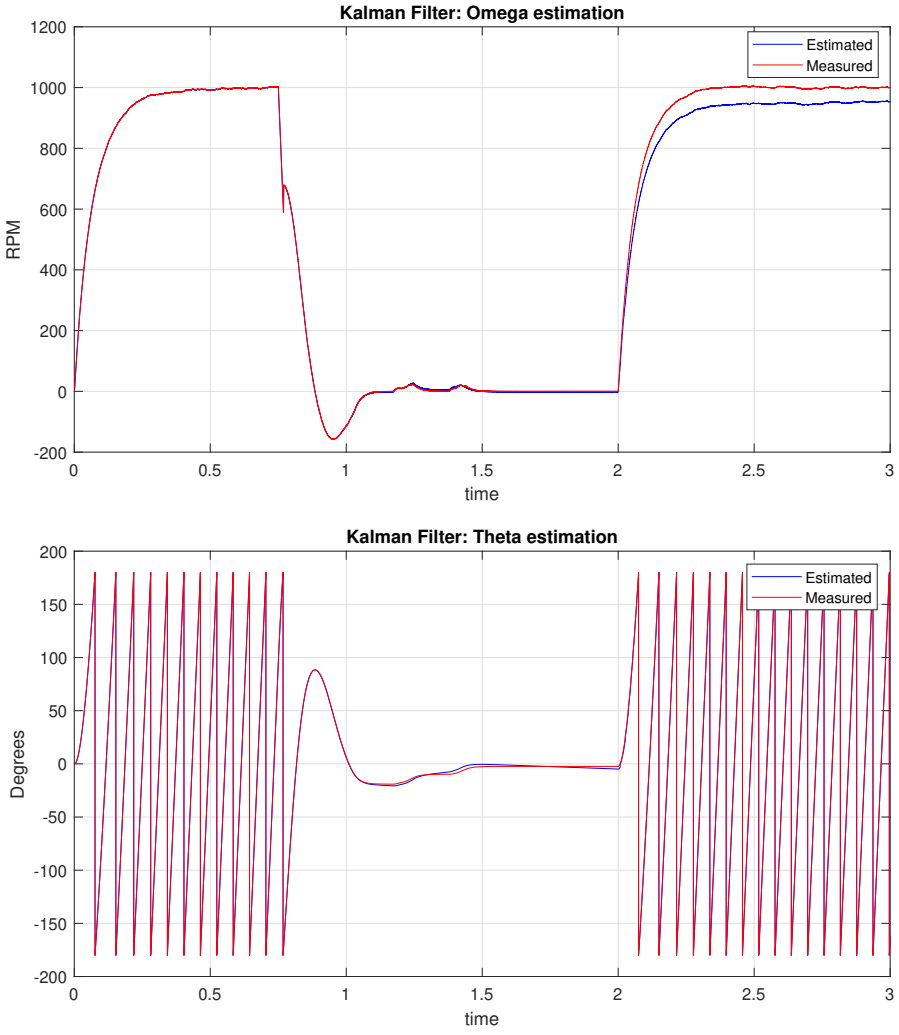


Figure 4.16: Estimation of rotor speed and position using EKF

Using trial and error, the following tuning parameters was found fitting for the extended Kalman filter using current measurements and a single Hall sensor to measure the rotor position:

$$\mathbf{Q} = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} 10^{-8} & 0 & 0 \\ 0 & 10^{-8} & 0 \\ 0 & 0 & 10^{-7} \end{bmatrix}, \quad \mathbf{P}_0 = \begin{bmatrix} 10^{-3} & 0 & 0 & 0 \\ 0 & 10^{-3} & 0 & 0 \\ 0 & 0 & 10^{-3} & 0 \\ 0 & 0 & 0 & 10^{-3} \end{bmatrix}$$

Given that the measurements have little noise in the simulated environment, the measurement covariance matrix **R** is small relative to the process noise covariance matrix **Q** - meaning that we want the Kalman filter to emphasize the measurements more than the predicted estimates.

From figure 4.17 it is visible that the EKF estimates both the rotor speed and position well during the first two seconds. However, the EKF does not have any measurements that can be used to correct deviations between the estimated and real rotor speed. Ultimately, this results in an incorrect speed estimate which in turn affects the position estimate since the position is estimated by integrating the rotor speed. Naturally, by having such deviations the EKF can not be used as feedback to the control system in its given form.

Simulation using EKF with two Hall sensors

From the results obtained using a single Hall sensor, the EKF clearly needed input related to the rotational speed of the rotor. Thus, two Hall sensors was used instead of one and the speed estimator based on Hall sensors from section 3.5 was implemented as well. After another tuning process, the new **R**-matrix was found to be:

$$\mathbf{R} = \begin{bmatrix} 10^{-8} & 0 & 0 & 0 \\ 0 & 10^{-8} & 0 & 0 \\ 0 & 0 & 6 \cdot 10^{-2} & 0 \\ 0 & 0 & 0 & 10^{-7} \end{bmatrix}$$

Yielding the following results while using the EKF as feedback:

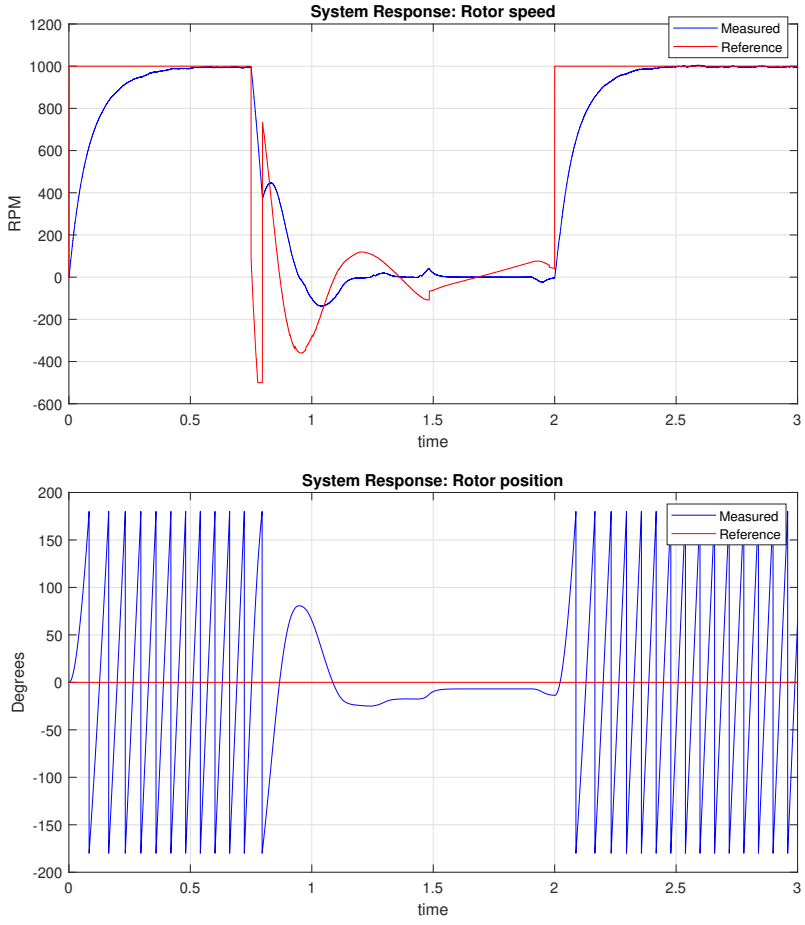


Figure 4.17: System reponse while using EKF as feedback

The system now works for both speed and position control. As there are few pulses from the Hall sensors at low speeds, a maximum error of approximately 20° occurs under position control.

It is also interesting to see how the Kalman filter estimates the speed and position of the rotor based on the measurements it has available.

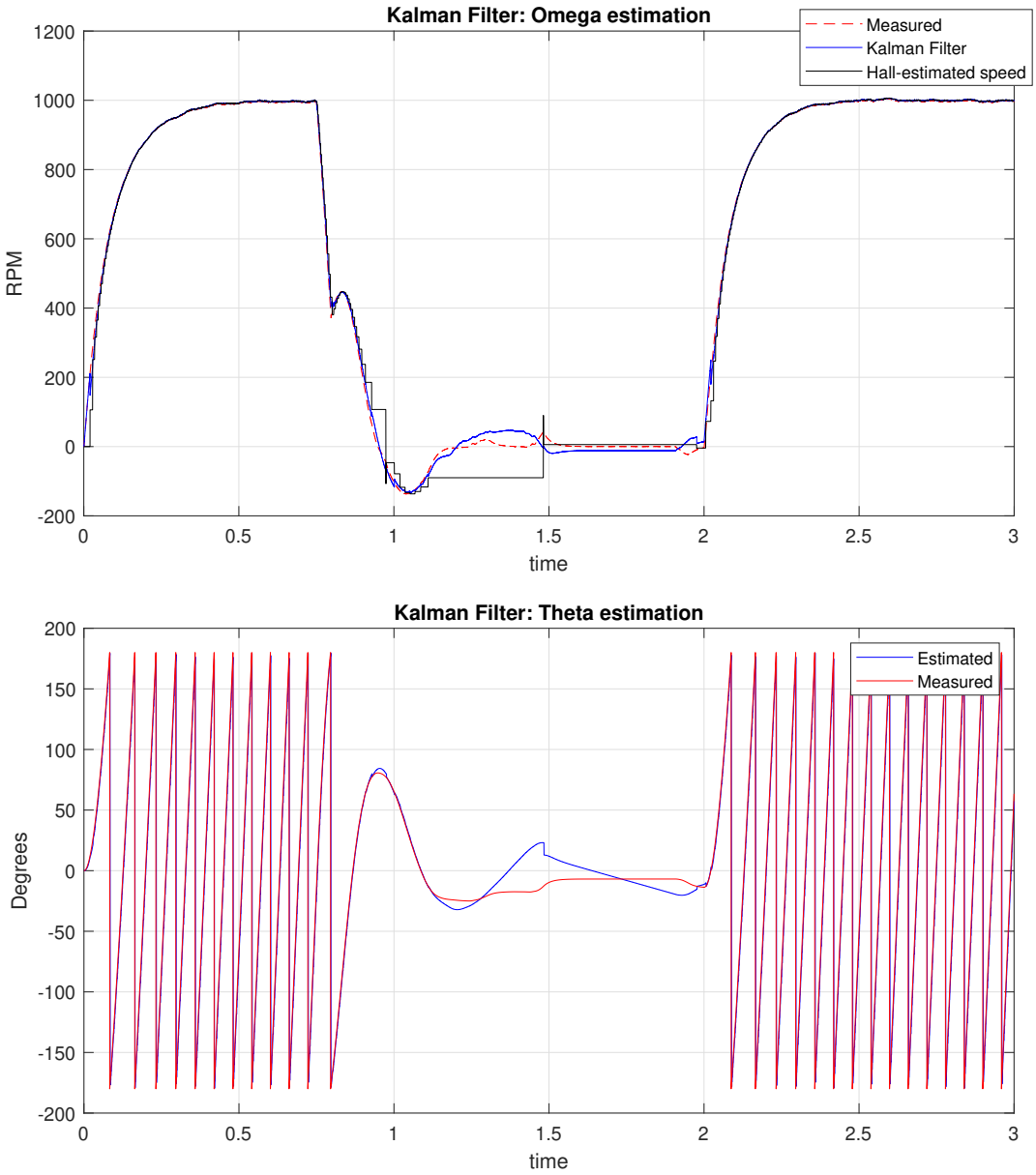


Figure 4.18: Estimation of rotor speed and position using EKF

As seen in the top figure of 4.18, the Kalman filter uses the speed measurements from the Hall sensors to improve the speed estimate. As the speed decreases, the filter gets fewer and fewer measurements which in turn leads to more inaccurate estimates. In the second figure we see how the rotor position is both overestimated and underestimated by the Kalman filter during position control due to few Hall sensor pulses, ultimately leading to the error seen in figure 4.17.

Simulation using EKF with three Hall sensors

By using three Hall sensors, it is possible to get even more accurate speed and position estimations. Since an extra Hall sensor was being used, the speed estimation would become more precise which in turn lowers the R-value associated with the speed measurements in the noise matrix **R**:

$$\mathbf{R} = \begin{bmatrix} 10^{-8} & 0 & 0 & 0 \\ 0 & 10^{-8} & 0 & 0 \\ 0 & 0 & 2 \cdot 10^{-2} & 0 \\ 0 & 0 & 0 & 10^{-7} \end{bmatrix}$$

This gives rise to the following system response:

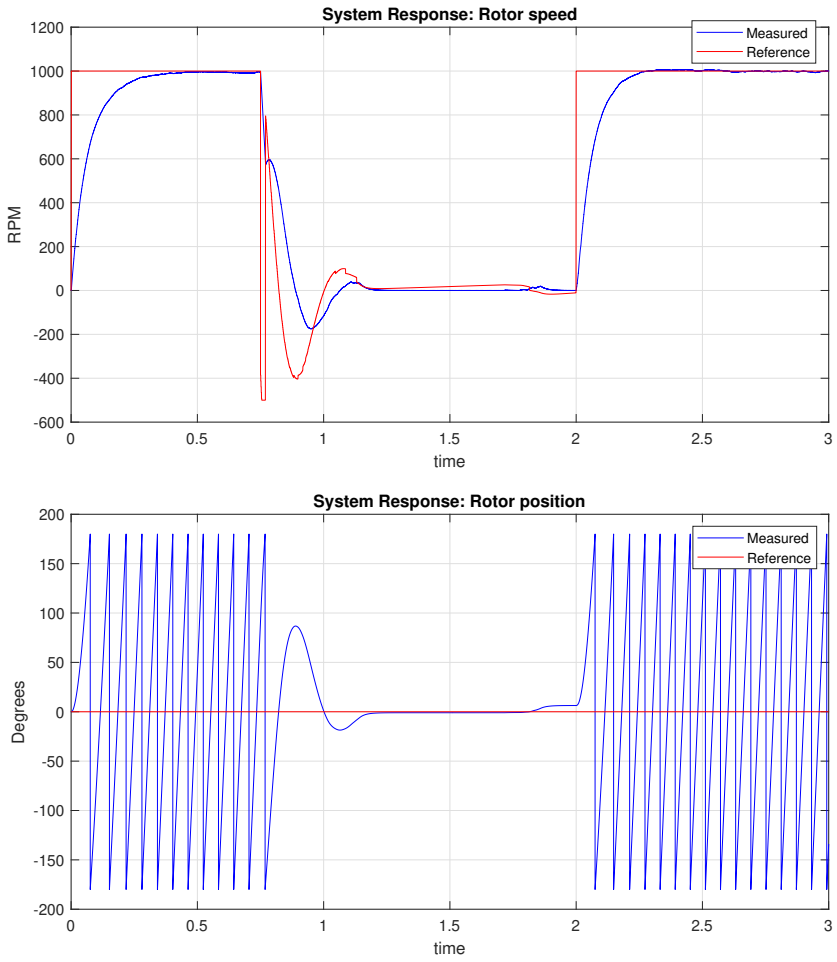


Figure 4.19: System reponse while using three Hall sensors and EKF as feedback

From figure 4.19 it can be seen that the speed controller has a faster and smoother response and that the error during position control is almost eliminated with a maximum error of 8°.

The plots below compare the real states to the estimated ones:

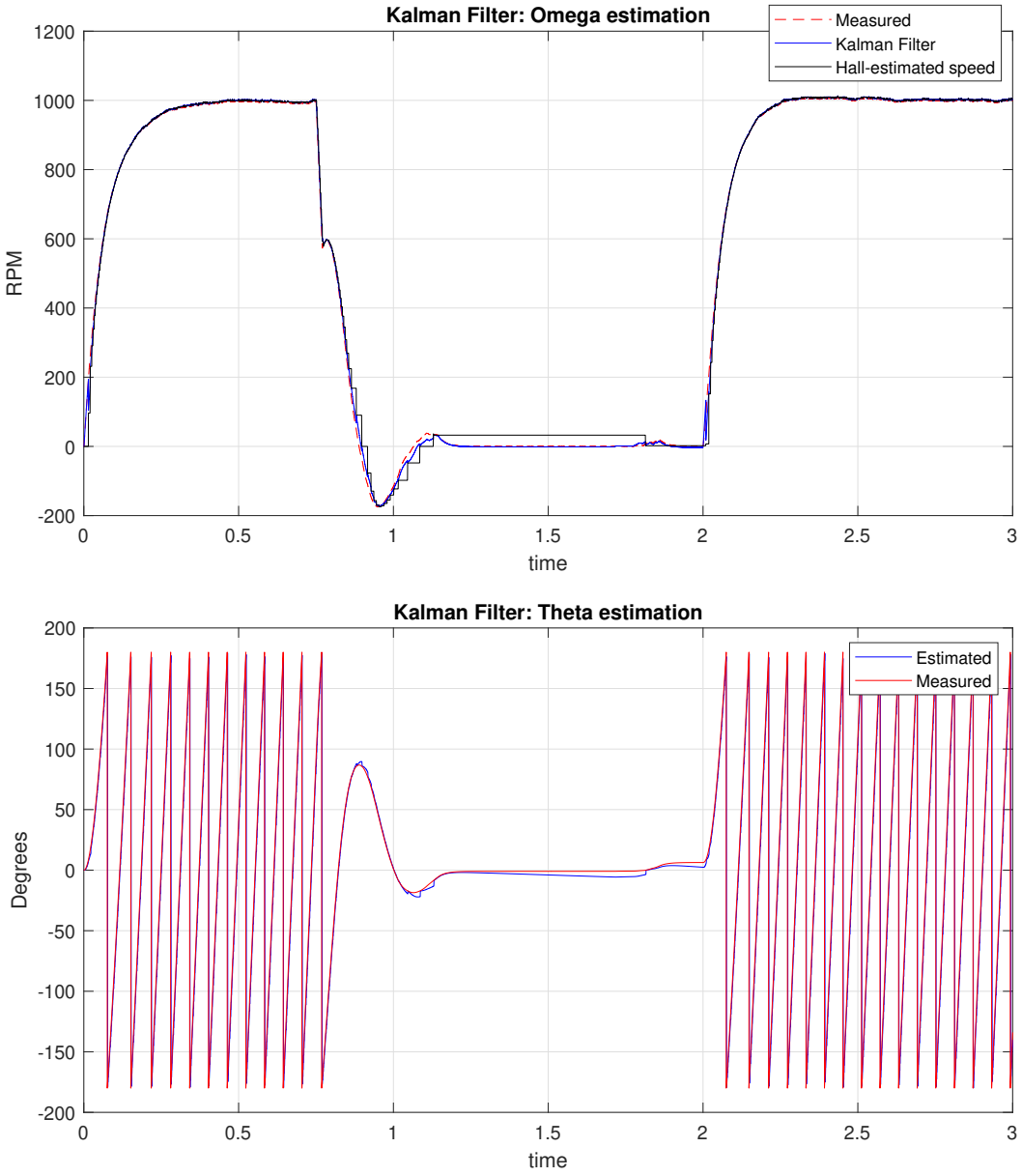


Figure 4.20: Estimation of rotor speed and position using EKF and three Hall sensors

From figure 4.20 it can be seen that the speed estimate has improved, especially about low speeds. This makes the position estimate more accurate so that the position error in figure 4.19 is kept within an acceptable limit.

Based on the results in this section, the position error during the locking-phase can be reduced by adding more and more Hall sensors until a satisfactory result is obtained.

Chapter 5

Conclusion and future work

Chapter 5 will conclude the work conducted in the thesis, before presenting recommendations for future work.

Conclusion

The purpose of the project was to develop a sensorless control system for a permanent magnet synchronous motor used to drive a propeller so that both the system costs as well as the energy efficiency of a drone were improved. As presented in Chapter 4, it was found that only two of the proposed estimation methods gave satisfactory control results. These were the ones based on the nonlinear observers and the Extended Kalman filter coupled with Hall sensors. By using the nonlinear observers as feedback, which performs poorly at low speeds, the propeller was magnetically locked into position by applying a fixed voltage to the stator windings. This method has its limitations as it is based on open loop control and therefore cannot guarantee that the propeller is locked into the desired position. On the other hand, the method involving the Extended Kalman filter coupled with Hall sensors is based on closed loop control and can to a greater extent ensure that the propeller is indeed locked into the desired position. From the simulations, it was found that the EKF control algorithm had a maximum error of 8°during the locking phase of the propeller, which was well within the requirements set in this thesis.

A surprising result was that the high frequency current injection-control scheme failed as this was predicted to be the solution to the shortcomings of the nonlinear observer during low-speed operations. As a result, the extended Kalman filter coupled with the Hall sensors was used as an alternative solution.

Recommendations for Future Work

Recommendations for future work involve determining the accuracy of the simulated motor model relative to the physical motor. This can be done by comparing results from the simulated system and the physical system under equal operating conditions. In addition, it is also recommended to test the developed control methods physically in order to verify the results from the thesis. Finally, it would be interesting to see the results of an analysis related to how much energy is saved by using the different control algorithms developed in this thesis in order to reduce wind resistance of a fixed-wing drone.

Bibliography

[1] S. Aasen. Control of Permanent Magnet Synchronous Motors for drones - Unpublished specialization project, 2021.

[2] V. Anand and S. Khobe. Smoothing EV powertrain performance with a field-oriented control algorithm, 2021. URL https://www.embedded.com/smoothing-ev-powertrain-performance-with-a-field-oriented-control-algorithm/.

[3] M. Anderson. What is the difference between incremental and absolute encoders?, June 2019. URL https://realpars.com/absolute-vs-incremental-encoder/.

[4] C.-T. Chen. *Linear System Theory and Design*. Oxford University Press, Inc., USA, 3rd edition, 1998. ISBN 0195117778.

[5] E. M. Coates, A. Wenz, K. Gryte, and T. A. Johansen. Propulsion System Modeling for Small Fixed-Wing UAVs. pages 748–757, June 2019. ISSN 2575-7296. doi: 10.1109/ICUAS.2019.8798082.

[6] N. Cravotta. Increasing Motor Performance with Field-Oriented Control, December 2011. URL https://www.digikey.ee/en/articles/increasing-motor-performance-with-field-oriented-control.

[7] J. Dilys, V. Stankevič, and K. Łuksza. Implementation of Extended Kalman Filter with Optimized Execution Time for Sensorless Control of a PMSM Using ARM Cortex-M3 Microcontroller. *Energies*, 14(12), 2021. ISSN 1996-1073. doi: 10.3390/en14123491. URL https://www.mdpi.com/1996-1073/14/12/3491.

[8] S. Dwivedi, M. Laursen, and S. Hansen. Voltage vector based control for PMSM in industry applications. 07 2010. doi: 10.1109/ISIE.2010.5637742.

[9] Electrical Baba. Permanent Magnet Synchronous Motor (PMSM) – Construction and Working Principle, June 2016. URL https://electricalbaba.com/permanent-magnet-synchronous-motor-pmsm-construction-working-principle/.

[10] W. Han. Simulation model development of electric motor and controller. 2017.

[11] M. Kazmierkowski, R. Krishnan, F. Blaabjerg, and J. Irwin. *Control in Power Electronics: Selected Problems*. Academic Press Series in Engineering. Elsevier Science, 2002. ISBN 9780080490786. URL https://books.google.no/books?id=E7_6Hpv8LOYC.

[12] Y. Kim and H. Bang. Introduction to Kalman Filter and Its Applications. 2019. doi: 10.5772/intechopen.80600. URL https://doi.org/10.5772/intechopen.80600.

[13] S.-T. Ko, S.-S. Park, and J.-H. Lee. Regenerative Battery Charging Control Method for PMSM Drive without a DC/DC Converter. *Electronics*, 8(10), 2019. ISSN 2079-9292. doi: 10.3390/electronics8101126. URL https://www.mdpi.com/2079-9292/8/10/1126.

[14] J. Lee, J. Hong, K. Nam, R. Ortega, L. Praly, and A. Astolfi. Sensorless Control of Surface-Mount Permanent-Magnet Synchronous Motors Based on a Nonlinear Observer. *Power Electronics, IEEE Transactions on*, 25:290 – 297, 03 2010. doi: 10.1109/TPEL.2009.2025276.

[15] MathWorks. PMSM Field-Oriented Control, . URL https://se.mathworks.com/help/physmod/sps/ref/pmsmfieldorientedcontrol.html.

[16] MathWorks. Permanent magnet synchronous motor with sinusoidal flux distribution, . URL https://se.mathworks.com/help/physmod/sps/ref/pmsm.html.

[17] MathWorks. PMSM Position Control, . URL https://www.mathworks.com/help/physmod/sps/ug/pmsm-position-control.html.

[18] D. Ocen. Direct Torque Control of a Permanent Magnet synchronous Motor, 2005. URL http://kth.diva-portal.org/smash/record.jsf?pid=diva2%3A582454&dswid=-503.

[19] PowerSim. Motor Control Design Suite. URL https://powersimtech.com/resources/tutorials/motor-control-design-suite/.

[20] RS Components. Everything You Need To Know About Hall Effect Sensors. URL https://ie.rs-online.com/web/generalDisplay.html?id=ideas-and-advice/hall-effect-sensors-guide.

[21] Y. Solbakken. Vector Control for Dummies, March 2017. URL https://www.switchcraft.org/learning/2016/12/16/vector-control-for-dummies.

[22] Texas Instruments. Brushless DC Motor Commutation Using Hall-Effect Sensors. URL https://www.ti.com/lit/an/slvaeg3a/slvaeg3a.pdf?ts=1650981547260&ref_url=https%253A%252F%252Fwww.google.com%252F.

[23] P. Vaclavek, P. Blaha, and I. Herman. AC Drive Observability Analysis. *IEEE Transactions on Industrial Electronics*, 60(8):3047–3059, 2013. doi: 10.1109/TIE.2012.2203775.

[24] G. Wang, M. Valla, and J. Solsona. Position Sensorless Permanent Magnet Synchronous Machine Drives—A Review. *IEEE Transactions on Industrial Electronics*, 67(7):5830–5842, 2020. doi: 10.1109/TIE.2019.2955409.

[25] Y. Zhao, Z. Zhang, C. Ma, W. Qiao, and L. Qu. Sensorless control of surface-mounted permanent-magnet synchronous machines for low-speed operation based on high-frequency square-wave voltage injection. In *2013 IEEE Industry Applications Society Annual Meeting*, pages 1–8, 2013. doi: 10.1109/IAS.2013.6682519.

Appendix A

MATLAB code

A.1 Initializing parameters of the model

```
%% Parameters for PMSM motor model

clear;
%% Machine Parameters
Vnom = 44;          % Motor voltage                    [V]
Ld   = 2.64e-6;     % Stator d-axis inductance         [H]
Lq   = 2.64e-6;     % Stator q-axis inductance         [H]
Rs   = 19.9e-3;     % Stator resistance per phase      [Ohm]
BEMF = 8.03e-3;     % Back-EMF constant wye-wound      [V/rpm]
p    = 14;          % Number of pole pairs
rad  = 30/pi;       % BEMF from V/rpm to V*s/rad
psim = BEMF*rad/p; % Permanent magnet flux linkage    [Wb]
Cdc  = 0.001;       % DC-link capacitor                [F]
Jm   = 0.01;        % Inertia motor                    [kg*m^2]

% First order approximation of propeller torque coeffisient:
Cq0  = 0.0078;      % Constant coeffisient
Cq1  = -0.0058;     % First order coeffisient

D    = 0.4;         % Propeller size (diameter)        [m]
rho  = 1.225;       % Air density                      [kg/m^2]
Va   = 20;          % Air speed assumed constant       [m/s]

```

```
24 Vdc_on = 24;        % Voltage threshold to activate the inverter
25
26 %% Control Parameters
27 Ts    = 5e-6;       % Fundamental sample time                [s]
28 fsw   = 2e4;        % PMSM drive switching frequency         [Hz]
29
30 Tsi   = 1e-5;       % Sample time for current control loops [s]
31 Tso   = 1e-4;       % Sample time for outer control loop     [s]
32 Tpc   = 1e-3;       % Samle time for position control        [s]
```

A.2 Estimation of rotor position - nonlinear observer

```
function [theta, x] = Estimate_theta(i_ab, v_ab, x_input)
dt = 5e-6;                % Step-length

L = 3/2 * 2.64e-6;        % Stator inductance
R = 3/2 * 0.0199;         % Stator resitance

L_ia = L * i_ab(1);       % Inductive voltage a
L_ib = L * i_ab(2);       % Inductive voltage b
R_ia = R * i_ab(1);       % Resistive voltage a
R_ib = R * i_ab(2);       % Resistive voltage b

lambda_2 = 0.0055^2;      % Flux linkage squared
gamma_half = 1e9*0.5;     % Observer gain

% Error:
err = lambda_2 - ((x_input(1) - L_ia)^2 + (x_input(2) - L_ib)^2);
% State Observer 1:
x1_dot = -R_ia + v_ab(1) + gamma_half * (x_input(1) - L_ia) * err;
% State Observer 2:
x2_dot = -R_ib + v_ab(2) + gamma_half * (x_input(2) - L_ib) * err;

% Forward euler of state observers:
x = [x_input(1) + x1_dot * dt,  x_input(2) + x2_dot * dt];

theta = atan2(x(2) - L_ib, x(1) - L_ia); % Estimation of theta
```

A.3 Estimation of rotor speed - nonlinear observer

```
function [omega, z_output] = Estimate_omega(theta, z)
dt = 5e-6;      % Step-length

Kp = 500;       % Observer gain P
Ki = 0;         % Observer gain I

% State observer 1:
z1_dot = Kp * (theta - z(1)) + Ki * z(2);

% State observer 2:
z2_dot = theta - z(1);

% Forward euler of state observers:
z_output = [z(1) + z1_dot * dt; z(2) + z2_dot * dt];

omega = Kp * (theta - z(1)) + Ki * z(2); % Estimation of omega
```

A.4 Prediction step of EKF

```
function [x_prediction, P_prediction] = Prediction(x_correction,
P_correction, u)

% Machine Parameters:
Ld    = 2.64e-6;     % Stator d-axis inductance          [H]
Lq    = 2.64e-6;     % Stator q-axis inductance          [H]
Rs    = 19.9e-3;     % Stator resistance per phase       [Ohm]
BEMF  = 8.03e-3;     % Back-EMF constant wye-wound       [V/rpm]
p     = 14;          % Number of pole pairs
rad   = 30/pi;       % BEMF from V/rpm to V*s/rad
psim  = BEMF*rad/p; % Permanent magnet flux linkage     [Wb]
Jm    = 0.01;        % Inertia motor                     [kg*m^2]
Ts    = 5e-6;        % Fundamental sample time           [s]

% Propeller parameters:
Cq0 = 0.0078;        % Constant coeffisient
Cq1 = -0.0058;       % Linear coeffisient
D   = 0.25;          % Propeller diameter
rho = 1.225;         % Air density
Va  = 20;            % Air speed

% Calculating load torque:
if x_correction(3) ~= 0                % Avoid dividing by zero
    Jad = 2*pi*Va/(x_correction(3)*D); % Advanced ratio
    torque = rho*D^5/(4*pi^2)*(Cq0 + Cq1*Jad)*x_correction(3)^2;
else
    torque = 0;
end
```

```
% x = [i_d, i_q, omega, theta], u = [v_d, v_q]

% Prediction step:
x1 = x_correction; % Create a shorter var. to make room in script

K1 = Ts*[-Rs/Ld*x1(1) + Lq/Ld*p*x1(3)*x1(2) + 1/Ld*u(1)
      -Rs/Lq*x1(2) - Ld/Lq*p*x1(3)*x1(1) - psim*p*x1(3)/Lq + 1/Lq*(2)
       1/Jm*(3/2*p*(psim*x1(2)+(Ld-Lq)*x1(1)*x1(2)) - torque)
       x1(3)*p];

x_prediction = x_correction + K1;

% Calculate Jacobian:
F = [1-Ts*Rs/Ld Ts*p*x1(3) Ts*p*x1(2) 0
     -Ts*p*x1(3) 1-Ts*Rs/Lq -Ts*p*x1(1)-Ts*psim*p/Lq 0
     0 0 1 0
     0 0 Ts*p 1];

% Define the Q-matrix:
Q = diag([0.1 0.1 0.1 0.1]);

% Update error covariance matrix:
P_prediction = P_correction*F*P_correction' + Q;
end
```

A.5 Update step of EKF

```
function [x_correction, P_correction]= Correction(x_prediction, P_predicti

% y = [i_d, i_q, omega], where omega is estimated by the Hall sensors

% Only i_d and i_q are continously measured:
y_measurements = [y(1); y(2)];

% Define the H-matrix:
H = [1 0 0 0
     0 1 0 0];

% Define the R-matrix:
R = diag([1e-8 1e-8]);

% If a hall pulse is detected, change the H and R-matrices:
if hall_sensor
    H = [1 0 0 0
         0 1 0 0
         0 0 1 0
         0 0 0 1];

     R = diag([1e-8 1e-8 2e-2 1e-7]);
end

% Measured rotor position is determined by the different hall sensors:
if hall_sensor(1)
    y_measurements = [y(1); y(2); y(3); 0];

elseif hall_sensor(2)
    y_measurements = [y(1); y(2); y(3); 2*pi/3];

elseif hall_sensor(3)
    y_measurements = [y(1); y(2); y(3); -2*pi/3];
end

```

```
36 % Calculate Kalman-gain
37 K = P_prediction*H'*(H*P_prediction*H' + R)^(-1);
38
39 % Update step:
40 x_correction = x_prediction + K*(y_measurements - H*x_prediction);
41
42 % Update error covariance matrix:
43 P_correction = (eye(4)-K*H)*P_prediction*(eye(4)-K*H)' + K*R*K';
44 end
```

A.6 Hall-based estimator

```
function [theta, omega]= Hall_estimator(hall, theta_in, time)

Ts = 5e-6; % Fundamental sample time

if time == 0                    % Avoid division by zero
    omega = 0;
else
    omega = 2*pi/(3*14*time); % mech.speed = distance/time*num of pp
end

% Any hall pulses will update the position estimate:
if hall(1)
    theta = 0;

elseif hall(2)
    theta = 2*pi/3;

elseif hall(3)
theta = -2*pi/3;

% Between hall pulses, the position is estimated by:
else
    theta = theta_in + omega*Ts;

    % Boundaries of the rotor angle:
    if theta > pi
        theta = -pi;
    elseif theta < -pi
        theta = pi;
    end
end
end
```

Sondre Aasen

Control of Permanent Magnet Synchronous Motors for drones

NTNU
Norwegian University of
Science and Technology