Master's thesis 2017:26

Electric Drive Speed Control for Varying Moment of Inertia

A Model Reference Adaptive Control Approach for Handling Changing Motor Dynamics

SIMON BJÖRKMAN, ALBIN DAHLIN

Department of Electrical Engineering
Division of Systems and Control
Chalmers University of Technology
Gothenburg, Sweden 2017

Electrice Drive Speed Control for Varying Moment of Inertia
A Model Reference Adaptive Control Approach for Handling Changing Motor Dynamics
SIMON BJÖRKMAN, ALBIN DAHLIN

Supervisor: Adam Maaninka, Aros Electronics
Examiner: Bo Egardt, Department of Electrical Engineering

Master's Thesis 2017:26
Department of Electrical Engineering
Division of Systems and Control
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2017

Abstract

This thesis presents an adaptive controller for regulating the speed of a permanent magnet synchronous motor that is robust against changes in moments of inertia. Model reference adaptive control is used as the approach for the control design. Both the recursive least squares method and the Kalman filter are presented, evaluated and discussed regarding their difference in performance when used as parameter estimation methods in the developed speed control algorithm. The advantages of an adaptive controller, compared to a PI controller, are demonstrated regarding disturbance rejection and reference tracking, when the moment of inertia is varying. The controller is concluded to be highly robust against changes in moment of inertia. Furthermore, the developed control algorithms are superior to the PI controller regarding disturbance rejection. When using the Kalman filter a small steady-state error is appearing which is not the case when using recursive least squares.

Keywords: Adaptive control, PMSM, Synchronous motor, Varying moment of inertia, MRAC, Speed control, Electric drive control

Acknowledgements

We would like to thank Aros electronics for allowing us to use their facilities, equipment and expertise throughout the project. A special thanks is directed to Adam Maaninka, our supervisor who has been a great support with his knowledge, positive attitude and enthusiasm. We would also like to thank Bo Egardt, who has advised and guided us.

Simon Björkman and Albin Dahlin, Gothenburg, June 2017

Contents

List of Figures

List of Tables

List of Symbols

δ_τ	Perturbation signal of torque reference [Nm]
ϵ	Prediction error
$\hat{\theta}$	Estimation of parameter vector
$\hat{b}$	Estimation of rotor frictional constant [Nms/rad]
λ	RLS forgettting factor
ω_m	Rotor mechanical speed [rad/s]
ω_m^*	Speed setpoint for controller [rad/s]
ω_r	Rotor electrical speed [rad/s]
ω_{ref}	Rotor speed for reference model [rad/s]
ψ_d	Stator flux in d-direction [Wb]
ψ_q	Stator flux in q-direction [Wb]
ψ_R	Back-emf constant [rad/Vs]
ψ_s	Stator flux [Wb]
τ_e	Produced electrical torque [Nm]
τ_e^*	Torque reference [Nm]
τ_e^u	Unperturbed torque reference [Nm]
τ_l	Torque load [Nm]
$\mathbf{i}$	Complex current vector [A]
$\mathbf{v}$	Complex voltage vector [V]
θ	Parameters vector
θ_m	Rotor mechanical angle [rad]
θ_r	Rotor electrical angle [rad]
φ	Regressor vector
a_{ref}	Pole of the first degree reference model
b	Rotor frictional constant [Nms/rad]
b_{ref}	Zero of the first degree reference model
i_d	Current in d-direction [A]
i_d^*	Current reference in d-direction [A]
i_q	Current in q-direction [A]
i_q^*	Current reference in q-direction [A]
J	Moment of inertia [kgm^2]
K	Correction gain
L_d	Inductance in d-direction [H]
L_q	Inductance in q-direction [H]
L_s	Stator inductance [H]
n_p	Number of pole pairs
P	Kalman filter/RLS algorithm covariance matrix
Q	Process covariance matrix in Kalman filter
R	Measurement covariance matrix in Kalman filter

R_s	Stator resistance [ohm]
T_ω	Sample time of speed controller [s]
v_d	Voltage in d-direction [V]
v_q	Voltage in q-direction [V]

1 Introduction

1.1 Problem background

Permanent Magnet Synchronous Motors (PMSM) are widely used in motor applications today due to their positive features, such as long life span as well as high efficiency and power density [1]. Aros electronics is a company that specializes in custom PM motor solutions that satisfy their customers' requirements. The most commonly used method for PMSM speed control, and the one used by Aros electronics, is a cascaded PI control system. This strategy is performing well during known circumstances. In some applications, however, such as in conveyor belts and robotic arms, the moment of inertia may be varying. With a varying moment of inertia, the dynamics of the system varies and hence a classical PI-controlled system will change its behavior. The changed operating condition is not what the PI controller was tuned for, which will lead to decreased performance [2]. An alternative controller that is more robust for cases when the moment of inertia is unknown or varying would be favorable in such applications. In order to better provide solutions for applications with changing dynamics, Aros is interested in developing a controller that has this feature. The design of such a controller is treated in this thesis.

1.2 Aim

The aim of the project is to develop a well-performing and robust PMSM speed controller that efficiently can compensate for changes in moment of inertia. The goal is that the resulting controller in this regard performs better than the PI regulator design that Aros is currently using. The quality of the controller is measured by the speed reference tracking considering step response qualities such as rise time, overshoot and settling time. Furthermore, an important factor is the ability to compensate for torque load and the disturbance rejection will thus be regarded as well.

1.3 Method

As a start, a rigorous literature study is performed, where the focus lies on the structure of, and existing control strategies for, the PMSM. With this information, a mathematical model of the PMSM is developed to be used for simulation purposes as well as during the control development. In order to make the simulation resemble the true system as accurately as possible, a system identification of the motor parameters is done. The developed model is then implemented in Matlab/Simulink, which is the tool used for simulation. The development of the control design is first carried out by analytically derive an expression for the control algorithm. The design procedure then enters an iterative process, improving the design by continuously test it in simulation and modifying the algorithm to achieve better results. In order to fairly and easily compare the different controllers and evaluate their performance, the test cases should be formed such that both the step response and disturbance rejection is examined for various moments of inertia. Once a controller with satisfying behavior is found from simulations, it is implemented in the physical system to be tested there. The iterative process is still carried out, now testing the controller both in simulation and the physical system. When a final control algorithm has been developed, a fine tuning of control parameter is performed before the final experiments are made. Finally, to evaluate the performance of the developed controller when implemented on the motor of subject, experiments are performed in a motor test bench.

1.4 Scope and boundaries

The focus of this project lies on developing a speed controller for a PMSM in the case of unknown or varying moment of inertia. The controller will regulate the speed of the motor using the voltage levels over the three phases as control signals. The transformation from desired voltage signal to phase voltage pulse width modulation (PWM) is already implemented in the programming library at Aros and this is utilized in the project. A hall-effect position sensor is used in order to measure the rotor position. The regulator is designed for a motor provided by Aros which is controlled using an Aros constructed microcontroller unit (MCU) with the central processing unit (CPU) Infineon XMC4500. The motor is modelled for simulation purposes and for use in the control design phase. However, the developed controller should be easy to adapt to other motors. The computational power of the CPU, along with the existing necessary code for motor control, provides the limitations for controller complexity. To optimize computational time for the implementation of the controller, such as using only integers, is not within the scope of the project. Only control development and implementation as a proof of concept is considered.

1.5 Outline

First, in Chapter 2 some background theory is given about the PMSM, the control scheme for model reference adaptive control as well as two different parameter estimation methods. In Chapter 3, the mathematical model of the PMSM is derived and identification of the motor parameter is presented. Chapter 4 describes the control design, where the development of the final controller is presented step by step. The results obtained from simulation and physical testing can be found in Chapter 5. In Chapter 6 the results and possible future work are discussed. Lastly, a conclusion of the project outcome can be found in Chapter 7.

2

Theory

In this chapter some background theory is presented for the reader to get an understanding of the system and the underlying methods used in the control algorithm. First, the operating principle of the permanent magnet synchronous motor is briefly described in Section 2.1. The concept of model reference adaptive control is then explained in Section 2.2. Finally, the recursive leasts squares method and the Kalman filter is covered in Section 2.3 and in Section 2.4, respectively.

2.1 Permanent magnet synchronous motor

2.1.1 Motor structure and operation

The PMSM is a three-phase machine and can be visualised as three static coils that are placed in a circle around a permanent magnet with 120° spacing. The magnet, called the rotor, can turn around its center. The three coils form the stator windings and are here denoted as phase a, b and c. Each winding can be modelled as an inductance in series with a resistance [3]. In this report, a motor with Y-configuration is used. Figure 2.1 shows a simple electrical diagram of the stator windings using this configuration. When a current runs through a coil, it creates a magnetic field according to Faraday's law [4]. The permanent magnet has its own magnetic field and the interaction of these fields creates a torque if the fields' opposite poles are not exactly aligned. Simplified, one could say that this effect is the strongest when the magnetic field created by the coils is 90 degrees ahead of the permanent magnet's north pole direction [3]. The idea is to make the two magnetic fields constantly repel each other in order to keep the rotor spinning.

2.1.2 Space vectors

Due to the placement of the coils, the phase currents can be illustrated in a phasor diagram as shown in Figure 2.2. By combining sign and magnitude of these currents, it is possible to create a current angle in any direction. However, it can easily be seen from Figure 2.1 and using Kirchoff's current law that the currents are linearly dependent according to

$$i_a + i_b + i_c = 0 \tag{2.1}$$

Because of this, it is possible to represent the three-phase motor using a two-phase

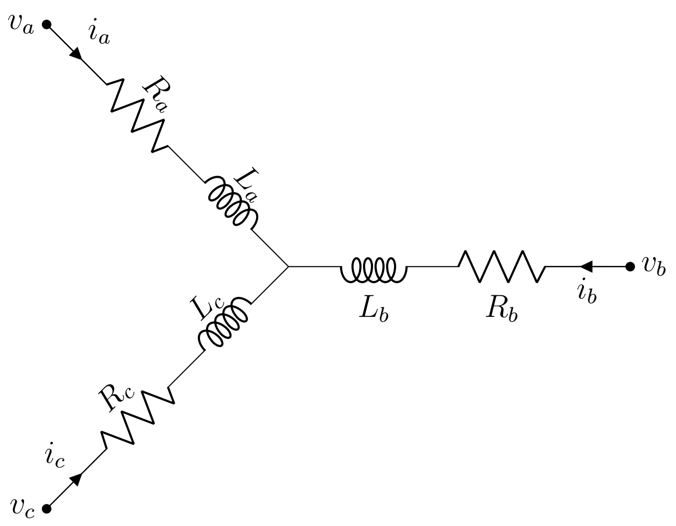

Figure 2.1: Stator windings diagram. Each phase can be modelled as a resistor in series with an inductance.

system with perpendicular axes. In the phasor diagram these are denoted α and β. It is convenient for modelling purposes to consider these as the real and imaginary axes in a complex plane, and the current vector becomes $\mathbf{i}^s = i_\alpha + ji_\beta$, called the *space vector*. The superscript s here denotes that the coordinates are stator fixed. It can however be seen as a real-valued vector $i^s = \begin{bmatrix} i_\alpha & i_\beta \end{bmatrix}^T$. This form is better suited for controller implementation, since imaginary numbers in this way are avoided. Transformation between the three-phase system and the $\alpha\beta$-system is straightforward with phasor arithmetic using

$$i_\alpha + ji_\beta = \frac{2}{3}K[i_a + e^{j2\pi/3}i_b + e^{j4\pi/3}i_c] \tag{2.2}$$

where K is a scaling constant that can be chosen arbitrarily [5]. It is used to make calculations easier in the two-phase system depending on the application. The three common choices are *Peak-value scaling* ($K = 1$), *RMS-value scaling* ($K = \frac{1}{\sqrt{2}}$) and *Power-invariant scaling* ($K = \sqrt{\frac{3}{2}}$). Throughout this report, the Power-invariant scaling is used. For the real-valued space vector the transformation is done with a constant transformation matrix

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = K \begin{bmatrix} \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} \\ 0 & \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} \tag{2.3}$$

2.1.3 dq-coordinates

For constant speed, the current vector in the phasor diagram is desired to rotate with the same speed as the rotor and have a phase lead of approximately 90 degrees compared to the rotor heading. For control purposes, it would be beneficial to use rotor-fixed coordinates since such a coordinate system would rotate with the same

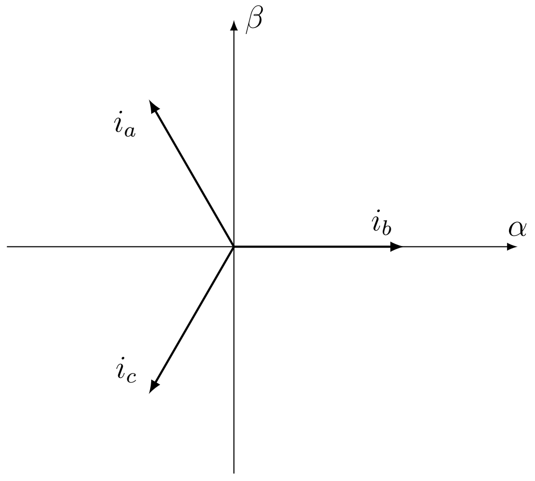

Figure 2.2: Phasor diagram. The total current vector resulting from the current in phase a, b and c can also be expressed as a vector in the 2-dimensional $\alpha\beta$ coordinate system.

speed as the rotor. A simpler model of the system can then be constructed where the currents are constant at constant speed rather than oscillating which is the case for the $\alpha\beta$-system. By defining the rotated current vector

$$\mathbf{i} = e^{-j\theta_r}\mathbf{i}^s \triangleq i_d + ji_q \tag{2.4}$$

this can be achieved [5]. This is denoted as the *dq* coordinate system, where i_d is the current in the rotor heading direction and i_q is the current in the perpendicular direction, always 90 degrees ahead of the magnet's north pole. Hence, i_q is the torque-creating current. Figure 2.3 shows the relation between the stator fixed $\alpha\beta$ coordinate system and the rotor fixed *dq* coordinate system. In (2.4), θ_r is the electrical angle, which is proportional to the rotor angle, θ_m, with the number of pole pairs, n_p. The rotor angle is defined as the angle between the rotor heading and the stator fixed α-axis. In order to go from the $\alpha\beta$-system to the dq-system, the rotor angle, θ_m, must hence be known. The transformation between these systems is a simple rotation as in (2.4) or for the real-valued vector representation using the rotational matrix

$$\begin{bmatrix} i_d \\ i_q \end{bmatrix} = \begin{bmatrix} \cos(\theta_r) & \sin(\theta_r) \\ -\sin(\theta_r) & \cos(\theta_r) \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} \tag{2.5}$$

Since the rotor-fixed dq-frame is convenient for control purposes and makes calculations more straightforward, it will be used throughout the rest of this report.

2.2 Model reference adaptive control

In model reference adaptive control (MRAC), the idea is to form a reference model with the desired closed-loop dynamics of the system. The control law is then formed to make the actual system behave as the reference model, given the same reference signals. Here, model reference control (MRC) is first explained where the system

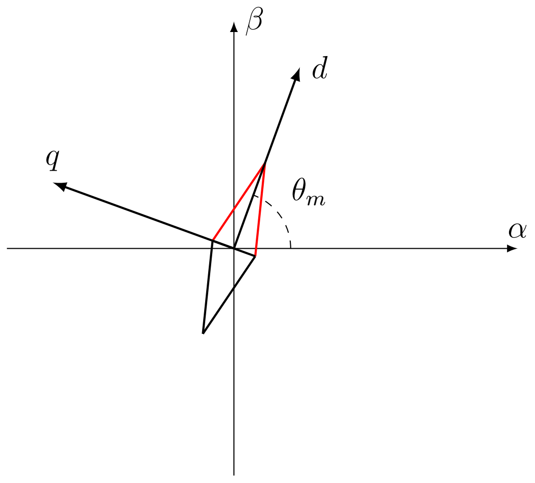

Figure 2.3: The dq coordinate system is rotor fixed with the d-axis aligned with the magnetic north pole.

dynamics are assumed to be static and known. Secondly, MRAC is presented, where the assumption of static and known dynamic parameters is removed. Only MRAC for first order systems is described, since this is sufficient for the reader to understand.

2.2.1 Model reference control

Consider a discrete first order system

$$y(k) = a_d y(k-1) + b_d u(k-1) \tag{2.6}$$

where y is the output signal, u is the control signal and where a_d and b_d are the dynamic parameters of the process. In MRC, a reference model is formed that models the desired dynamics from reference signal to output signal of the closed-loop system. In order to guarantee that the resulting controller becomes causal, the time delay of the closed-loop system needs to be at least the same as the delay of the open-loop system. A simple design of the reference model is to use the same order as the actual system model. That is,

$$y_{\text{ref}}(k) = a_{\text{ref}} y_{\text{ref}}(k-1) + b_{\text{ref}} r(k-1), \tag{2.7}$$

where r is the reference signal, y_{ref} is the reference model output and a_{ref} and b_{ref} are the dynamic parameters designed to make the model respond to reference signals as desired. To design the control law, the system model (2.6) is rearranged as [7]

$$y(k) = a_{\text{ref}} y(k-1) + (a_d - a_{\text{ref}}) y(k-1) + b_d u(k-1). \tag{2.8}$$

Since the aim in model reference control is to follow the reference model as close as possible, the tracking error, $e(k) = y(k) - y_{\text{ref}}(k)$, is of interest. The dynamics for the tracking error can be calculated by subtracting (2.7) from (2.8) giving

$$e(k) = a_{\text{ref}} e(k-1) + (a_d - a_{\text{ref}}) y(k-1) + b_d u(k-1) - b_{\text{ref}} r(k-1) \tag{2.9}$$

As the reference model parameter a_{ref} will be chosen such that the dynamics become stable, a sufficient condition to make the error converge exponentially to zero is to make the error dynamics become

$$e(k) = a_{\text{ref}} e(k-1) \tag{2.10}$$

This is achieved if the term $(a_d - a_{\text{ref}})y(k-1) + b_d u(k-1) - b_{\text{ref}} r(k-1)$ is zero which leads to the control law formulation

$$u(k) = \frac{(a_{\text{ref}} - a_d)}{b_d} y(k) + \frac{b_{\text{ref}}}{b_d} r(k) \tag{2.11}$$

2.2.2 MRAC

Using the control law (2.11) would give the same behaviour of the real system as the reference model. However, this is under the assumption that the process parameters a_d and b_d are exactly know. This is not the case in reality and to overcome this problem the certainty equivalence principle can be used [7]. This simply means that the unknown parameters are substituted by their estimations. The control law then becomes

$$u(k) = \frac{(a_{\text{ref}} - \hat{a}_d)}{\hat{b}_d} y(k) + \frac{b_{\text{ref}}}{\hat{b}_d} r(k) \tag{2.12}$$

where $\hat{a}_d$ and $\hat{b}_d$ are the estimated values of a_d and b_d respectively. Estimation of these parameters is therefore necessary. The estimation is performed online, meaning the parameters are updated continuously as the controller is running. This means that the control law changes during operation. The adaptation law may be performed in different ways, such as using the MIT rule [8], Lyapunov function analysis [9], Kalman filters, least squares methods or other filtering approaches. In this report, the recursive least squares method (RLS) and the Kalman filter are treated.

2.3 Recursive least squares

The method of least squares is a common concept to use when parameters for overdetermined systems are to be estimated. In this thesis, system models that are linear in parameters will be considered. The model for these types of systems can be expressed as

$$y(k) = \varphi^T(k)\theta + \epsilon(k) \tag{2.13}$$

Here y is the measured system output and φ is the regressor vector containing the known system signals that y depends on. The parameters that are to be estimated are contained in θ and ϵ is called the prediction error, which contains the difference between the predicted and actual output.

According to least squares, if there are N discrete time samples available, θ should be chosen such that the sum of all squared prediction errors

$$J(\theta) = \frac{1}{2}\sum_{k=1}^{N}\epsilon^2(k) \tag{2.14}$$

is minimized. However, in time-varying systems the parameters are not fixed at one true value but depend on time instance. The most recent errors are therefore more relevant than old ones. To account for this, a forgetting factor $\lambda \in (0,1)$ is introduced in the optimization problem as a weighing of the errors. The cost function (2.14) now becomes

$$J(\theta) = \frac{1}{2}\sum_{\tau=1}^{k}\lambda^{k-\tau}\epsilon^2(\tau), \tag{2.15}$$

where it can be seen that older error samples receive a higher exponent, making them less relevant. Since (2.15) is a convex function the minimum can be found by differentiation [10] according to

$$\frac{dJ(\theta)}{d\theta} = \sum_{\tau=1}^{k}\lambda^{k-\tau}\frac{d\epsilon(\tau)}{d\theta}\epsilon(\tau) = \sum_{\tau=1}^{k}\lambda^{k-\tau}\varphi(\tau)[y(\tau)-\varphi^T(\tau)\theta] = 0 \tag{2.16}$$

where the last equality can be realized from (2.13). The expression in (2.16) can be rewritten as

$$\sum_{\tau=1}^{k}\lambda^{k-\tau}\varphi(\tau)y(\tau) - \sum_{\tau=1}^{k}\lambda^{k-\tau}\varphi(\tau)\varphi^T(\tau)\theta = 0 \tag{2.17}$$

which leads to the RLS estimate

$$\hat{\theta}(k) = \Big[\sum_{\tau=1}^{k}\lambda^{k-\tau}\varphi(\tau)\varphi^T(\tau)\Big]^{-1}\sum_{\tau=1}^{k}\lambda^{k-\tau}\varphi(\tau)y(\tau). \tag{2.18}$$

Now, introduce the notation of the first term

$$P(k) = \Big[\sum_{\tau=1}^{k}\lambda^{k-\tau}\varphi(\tau)\varphi^T(\tau)\Big]^{-1} \tag{2.19}$$

which can be proven to be the covariance matrix of the estimate [11]. It can be seen that its inverse is

$$\begin{aligned} P^{-1}(k) &= \sum_{\tau=1}^{k}\lambda^{k-\tau}\varphi(\tau)\varphi^T(\tau) \\ &= \lambda\sum_{\tau=1}^{k-1}\lambda^{k-1-\tau}\varphi(\tau)\varphi^T(\tau) + \varphi(k)\varphi^T(k) \\ &= \lambda P^{-1}(k-1) + \varphi(k)\varphi^T(k). \end{aligned} \tag{2.20}$$

$P^{-1}(k)$ can thus be updated recursively using only its previous value and the new data, $\varphi(k)$. Using this information and the matrix inversion lemma, the recursive update of the covariance matrix can be shown to be [12]

$$P(k) = \frac{1}{\lambda}\left(P(k-1) - \frac{P(k-1)\varphi(k)\varphi^T(k)P(k-1)}{\lambda + \varphi^T(k)P(k-1)\varphi(k)}\right) \tag{2.21}$$

Now, with notation according to (2.19), the previous estimate can in accordance with (2.18) be written as

$$\begin{aligned}\hat{\theta}(k-1) &= P(k-1)\sum_{\tau=1}^{k-1}\lambda^{k-1-\tau}\varphi(\tau)y(\tau)\\ \Leftrightarrow \sum_{\tau=1}^{k-1}\lambda^{k-1-\tau}\varphi(\tau)y(\tau) &= P^{-1}(k-1)\hat{\theta}(k-1).\end{aligned} \tag{2.22}$$

With the use of this, and the fact that $\lambda P^{-1}(k-1) = P^{-1}(k) - \varphi(k)\varphi^T(k)$ from (2.20), (2.18) can be rewritten as

$$\hat{\theta}(k) = P(k)\left(\lambda\sum_{\tau=1}^{k-1}\lambda^{k-1-\tau}\varphi(\tau)y(\tau) + \varphi(k)y(k)\right) \tag{2.23}$$

$$= P(k)\left(\lambda P^{-1}(k-1)\hat{\theta}(k-1) + \varphi(k)y(k)\right) \tag{2.24}$$

$$= P(k)\left(\left(P^{-1}(k) - \varphi(k)\varphi^T(k)\right)\hat{\theta}(k-1) + \varphi(k)y(k)\right) \tag{2.25}$$

$$= \hat{\theta}(k-1) + P(k)\varphi(k)\left(y(k) - \varphi^T(k)\hat{\theta}(k-1)\right). \tag{2.26}$$

The resulting RLS algorithm for each iteration can therefore be described as

$$\begin{aligned}\hat{\theta}(k) &= \hat{\theta}(k-1) + K(k)\epsilon(k)\\ \epsilon(k) &= y(k) - \varphi^T(k)\hat{\theta}(k-1)\\ K(k) &= P(k)\varphi(k)\\ P(k) &= \frac{1}{\lambda}\left(P(k-1) - \frac{P(k-1)\varphi(k)\varphi^T(k)P(k-1)}{\lambda + \varphi^T(k)P(k-1)\varphi(k)}\right)\end{aligned} \tag{2.27}$$

2.4 Kalman filter

A widely used estimation method is the Kalman filter. It was first presented by R.E. Kalman and R.S. Bucy in [13] and is the optimal filter for a stochastic linear process with additive white Gaussian noise. The filter is based on a description of the process as a Markov chain, i.e. a discrete process where the next state of the system depends solely on the current, perturbed by a Gaussian error signal. Furthermore, a measurement model describing how the available measurement depends on the state is needed. Also, the measurement is modelled with a perturbation of a Gaussian noise signal. Hence, the dynamic and measurement models are defined as

$$x(k) = F_{k-1}x(k-1) + w(k) \tag{2.28}$$

$$y(k) = H_k x(k) + v(k) \tag{2.29}$$

Here x is the state that is estimated, $w(k) \sim \mathcal{N}(0, Q)$ is the process noise and $v(k) \sim \mathcal{N}(0, R)$ is the measurement noise. The matrix F_{k-1} is the state transition matrix and H_k is the observation model matrix. The Kalman filter is the optimal state estimator for this kind of system and can be divided into the following two

steps for each recursion [14].
Prediction step:

$$\hat{x}^-(k) = F_{k-1}\hat{x}(k-1) \tag{2.30}$$
$$P^-(k) = F_{k-1}P(k-1)F_{k-1}^T + Q \tag{2.31}$$

Update step:

$$\epsilon(k) = y(k) - H_k\hat{x}^-(k) \tag{2.32}$$
$$S = H_kP^-(k)H_k^T + R \tag{2.33}$$
$$K = P^-(k)H_k^TS^{-1} \tag{2.34}$$
$$\hat{x}(k) = \hat{x}^-(k) + K\epsilon(k) \tag{2.35}$$
$$P(k) = P^-(k) - KSK^T \tag{2.36}$$

The filter at each iteration computes the current optimal estimate, $\hat{x}(k)$, and the estimate covariance, $P(k)$, given the new measurement, $y(k)$. For a thorough derivation of the Kalman filter equations, the reader is referred to [13].

3

Modelling

In Section 2.1, the operating principle for the PMSM and the dq coordinate system was described. Using this as background information, the model of the PMSM, which is used for the control design and in simulations, is presented in this chapter. The mathematical model is first derived in Section 3.1 and in Section 3.2 the process of motor parameter identification is presented.

3.1 Mathematical model of PMSM

A mathematical model of the PMSM is given below, expressed in the rotor bound *dq* coordinate system, covered in section 2.1.3. The electrical dynamics, derived in Section 3.1.1, is combined with the mechanical dynamics, handled in Section 3.1.2, to result in the full dynamical expressions given in Section 3.1.3

3.1.1 Electrical dynamics

The input signal to a PMSM is the voltage over the stator windings. The voltage that does not dissipate in the winding resistance will produce a magnetic flux, according to the law of induction. The voltage equation can thus be written as [5]

$$\mathbf{v}^s - R_s\mathbf{i}^s - \frac{d\Psi_s^s}{dt} = 0 \tag{3.1}$$

where the superscript s denotes that the vectors are expressed in the stator bound coordinate system. Here, the stator flux Ψ_s^s is given by

$$\Psi_s^s = L_s\mathbf{i}^s + \Psi_R^s \tag{3.2}$$

where L_s is the stator inductance and Ψ_R^s is the rotor flux linkage created by the permanent magnets. The rotor flux vector in the stator coordinate system depends on the electrical angle of the rotor, θ_r, according to

$$\Psi_R^s = \psi_R e^{j\theta_r} \tag{3.3}$$

where the flux modulus ψ_R can be seen as a constant which depends on the permanent magnet. The derivative of the stator flux can, by inserting (3.3) into (3.2), be found to be

$$\frac{d\Psi_s^s}{dt} = L_s\frac{d\mathbf{i}^s}{dt} + j\omega_r\psi_R e^{j\theta_r}. \tag{3.4}$$

where ω_r is the time derivative of θ_r, i.e. electrical angular speed. Now, (3.1) can be expressed as

$$L_s \frac{d\mathbf{i}^s}{dt} = \mathbf{v}^s - R_s \mathbf{i}^s - j\omega_r \psi_R e^{j\theta_r}. \tag{3.5}$$

As described in Section 2.1.3 it is convenient to use synchronous, i.e. rotor fixed, coordinates when designing a controller for the PMSM. The electrical dynamics in (3.5) is therefore transformed into dq-coordinates by rotating all stator coordinates with the angle θ_r, giving

$$L_s \frac{d(e^{j\theta_r}\mathbf{i})}{dt} = jL_s\omega_r e^{j\theta_r}\mathbf{i} + L_s e^{j\theta_r}\frac{d\mathbf{i}}{dt} = e^{j\theta_r}\mathbf{v} - R_s e^{j\theta_r}\mathbf{i} - j\omega_r\psi_R e^{j\theta_r} \tag{3.6}$$

or equivalently

$$L_s e^{j\theta_r}\frac{d\mathbf{i}}{dt} = e^{j\theta_r}\mathbf{v} - R_s e^{j\omega_r}\mathbf{i} - jL_s\theta_r e^{j\theta_r}\mathbf{i} - j\omega_r\psi_R e^{j\theta_r}. \tag{3.7}$$

For convenience, (3.7) is multiplied with $e^{-j\theta_r}$ leading to

$$L_s \frac{d\mathbf{i}}{dt} = \mathbf{v} - R_s\mathbf{i} - jL_s\omega_r\mathbf{i} - j\omega_r\psi_R \tag{3.8}$$

which is described as a complex equation were $\mathbf{i} = (i_d + ji_q)$. That is, the imaginary numbers represents the q-direction of the current and real numbers represents the d-direction. As stated in Section 2.1.2 a real-valued vector representation is favorable for control implementation and (3.8) can instead be expressed as

$$L_s \frac{d}{dt}\begin{bmatrix} i_d \\ i_q \end{bmatrix} = \begin{bmatrix} v_d \\ v_q \end{bmatrix} - R_s \begin{bmatrix} i_d \\ i_q \end{bmatrix} - L_s\omega_r \begin{bmatrix} -i_q \\ i_d \end{bmatrix} - \begin{bmatrix} 0 \\ \psi_R\omega_r \end{bmatrix}. \tag{3.9}$$

The resistance in both directions can be modelled as equal, but the stator inductance cannot. The magnets in the rotor are not uniformly distributed over the whole surface, yielding sinusoidally varying inductance. The inductance on the d and q axes is defined as L_d and L_q respectively and L_s can therefore be expressed as a matrix according to

$$L_s = \begin{bmatrix} L_d & 0 \\ 0 & L_q \end{bmatrix}. \tag{3.10}$$

3.1.2 Mechanical dynamics

The rotor speed dynamics depend, according to Newton's second law, on the applied torques and the moment of inertia, J. The torques affecting the rotor are the produced electrical torque, τ_e, the load torque, τ_L and the frictional torque that is modelled to be proportional to the speed with the friction constant, b. Hence, the speed dynamics can be written as

$$J\frac{d\omega_m}{dt} = \tau_e - b\omega_m - \tau_l \tag{3.11}$$

Here ω_m is the mechanical rotor speed in rad/s, which is directly proportional to the electrical speed of the rotor, ω_r, by the number of pole pairs, n_p

$$\omega_r = n_p \omega_m \tag{3.12}$$

The produced electrical power can be written as [5]

$$P_e = \omega_r \mathrm{Im}\{\Psi_R^* \mathbf{i}\} \tag{3.13}$$

Since power is the torque times the angular speed [16], the torque can be obtained by

$$\begin{aligned}\tau_e &= \frac{P_e}{\omega_m} = n_p \ \mathrm{Im}\{\Psi_R^* \mathbf{i}\} = n_p \ \mathrm{Im}\{(\psi_d - j\psi_q)(i_d + j i_q)\} \\ &= n_p(\psi_d i_q - \psi_q i_d).\end{aligned} \tag{3.14}$$

All the magnetic flux from the rotor is directed in the magnet's north pole, i.e. in d-direction and since L_s is a matrix, the stator flux, $\psi_s = \begin{bmatrix}\psi_d \\ \psi_q\end{bmatrix}$, can, in accordance with (3.2), be written as

$$\psi_s = L_s i + \begin{bmatrix}\psi_R \\ 0\end{bmatrix} = \begin{bmatrix}L_d i_d + \psi_R \\ L_q i_q\end{bmatrix}. \tag{3.15}$$

Finally, this leads to the electrical torque equation

$$\tau_e = n_p(\psi_d i_q - \psi_q i_d) = n_p[\psi_R i_q + (L_d - L_q) i_d i_q]. \tag{3.16}$$

3.1.3 Total dynamical equations

Inserting (3.16) into (3.11) and (3.10) into (3.9) leads to the final dynamical equations for the PMSM

$$J\frac{d\omega_m}{dt} = n_p(\psi_R + (L_d - L_q) i_d) i_q - b\omega_m - \tau_L \tag{3.17}$$

$$\omega_r = n_p \omega_m \tag{3.18}$$

$$L_d \frac{di_d}{dt} = v_d - R_s i_d + L_q \omega_r i_q \tag{3.19}$$

$$L_q \frac{di_q}{dt} = v_q - R_s i_q - L_d \omega_r i_d - \psi_R \omega_r \tag{3.20}$$

3.2 System identification

To make the behaviour of the simulation as similar as possible to the physical system, it is important to know the motor parameters well. The rotor moment of inertia, $96 \times 10^{-6}\ kgm^2$, was supplied with enough accuracy to be used directly, but other data was either unavailable or inaccurate and those parameters therefore needed to be estimated.

3.2.1 Resistance

The resistance in the dynamical model developed is assumed to be equal for all stator phases. In order to estimate the stator resistance, two of the phases were connected to form one node. A current source was connected between this node and the third phase. The resulting electrical circuit can be seen in Figure 3.1. The resistance from node A to node B can, by the assumption $R_a = R_b = R_c \equiv R_s$, be expressed as

$$\frac{1}{\frac{1}{R_s} + \frac{1}{R_s}} + R_s = \frac{3}{2}R_s. \tag{3.21}$$

Ohm's law applied between node A and node B gives the relationship between the voltage, V, the current induced by the current source, I, and the total resistance

$$V = \frac{3}{2}R_s I \tag{3.22}$$

or equivalently

$$R_s = \frac{2V}{3I}. \tag{3.23}$$

The resistance can thus be estimated by setting a fixed current and measure the voltage using a voltmeter. When a current of 3 A was fed through the motor, a voltage of 87.6 mV was measured giving a resistance of 0.0195 ohm.

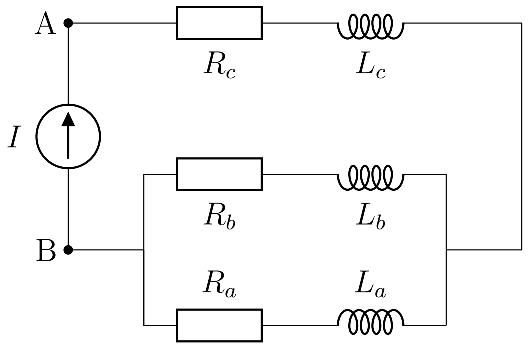

Figure 3.1: Electrical circuit for resistance identification.

3.2.2 Inductance

For identification of the inductances L_d and L_q, two of the phases are again connected to form an electrical circuit equivalent to Figure 3.2. According to [15], L_d can be found from the relationship $L_d = \frac{2}{3}L_{AB}$ when the electrical angle is zero, i.e $\theta_r = 0°$. Similarly, L_q is found when the electrical angle is 90°, i.e $L_q = \frac{2}{3}L_{AB}$ at $\theta_r = 90°$. In practice, this means that L_q is the highest value of L_{AB} when rotating the rotor for one period and L_d is the lowest. By measuring the inductance L_{AB} using an LCR meter, the rotor bound inductance was found to be $L_d = 83\ \mu H$ and $L_q = 170\ \mu H$.

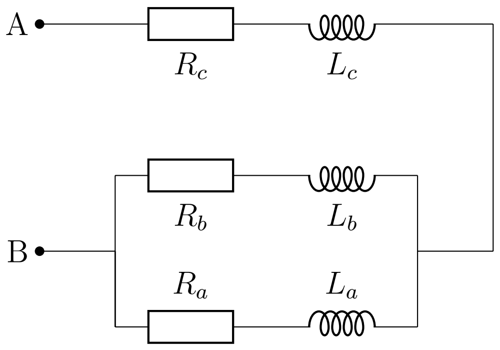

Figure 3.2: Electrical circuit for inductance identification.

3.2.3 Flux modulus

The flux modulus, or back-emf constant, can be determined by externally rotate the rotor while measuring the resulting back-emf voltage. At steady state, (3.5) can be rewritten as

$$V_{L-N} - R_s\mathbf{i}_s^s - \omega_r\psi_R = 0 \tag{3.24}$$

where V_{L-N} is the voltage over one phase to the neutral point. Since the voltage drop over the winding resistance is very small it can be neglected, resulting in the relationship

$$\psi_R = \frac{V_{L-N}}{\omega_r}. \tag{3.25}$$

However, it can be difficult to reach the neutral point of the motor. Instead, the line to line voltage, V_{L-L} can be used which relates to the line to neutral voltage according to $V_{L-L} = \sqrt{3}V_{L-N}$ giving

$$\psi_R = \frac{V_{L-L}}{\sqrt{3}\omega_r} \tag{3.26}$$

Using a power drill to rotate the rotor at a constant speed, the electrical frequency and the voltage was measured with an oscilloscope. In this way, the flux modulus was found to be 0.0091 Wb.

3.2.4 Frictional constant

The frictional coefficient, b, is a measure on how much breaking torque is created from an angular speed of the rotor and is modelled as a constant. From (3.17), it can be seen that at steady state, i.e. constant currents and constant speed, we have

$$n_p(\psi_R - (L_d - L_q)i_d)i_q - b\omega_m - \tau_L = 0. \tag{3.27}$$

By not applying any load torque, the friction constant, b, can be found to be

$$b = \frac{n_p(\psi_R - (L_d - L_q)i_d)i_q}{\omega_m}. \tag{3.28}$$

When the rotor was running at a constant speed, i_d and i_q was measured and the frictional factor could, using (3.28), be calculated to 4.2281×10^{-5} Nms/rad.

3.2.5 Measurement noise

When finding the magnitude of the measurement noise, a constant voltage vector was applied in order to get a constant rotational speed. The speed was then measured through the motor controller hardware, using the encoder. Over a sequence of 998 samples, or 33.7 seconds, the speed variance was 1.17 RPM when the mean speed was 992 RPM. The measurement noise variance is assumed to be independent on the mean speed and was thus identified as 1.17 RPM.

4

Control Design

This chapter presents the development of the controller. In Section 4.1, the general control structure is presented. The motor dynamics are used to obtain the system time constants, which indicates that the electrical and mechanical dynamics can be decoupled. In Section 4.1.1, the decoupled current controller is briefly described and in Section 4.1.2 the control signal remapping from torque setpoint to current setpoints is explained. The main work is presented in Sections 4.2 and 4.3, where the speed controller is described.

4.1 Control structure

The aim of the controller is to make the rotational speed of the PMSM, ω_m, follow a speed setpoint, ω_m^*, using the rotor bound voltage levels, v_d and v_q, as control signals. As can be seen in the dynamic equations for the PMSM in (3.17)-(3.20) the control signals do not have a direct impact on the rotational speed but instead an indirect influence through the currents i_d and i_q. In (3.17) the produced electromagnetic torque can be extracted as

$$\tau_e = n_p(\psi_R + (L_d - L_q)i_d)i_q \tag{4.1}$$

Inserting (4.1) into (3.17) leads to the expression

$$J\frac{d\omega_m}{dt} = \tau_e - b\omega_m - \tau_L \tag{4.2}$$

By considering τ_L an input disturbance, the model is in this way formulated as a first order single-input single-output system with transfer function

$$\frac{\omega_m(s)}{\tau_e(s)} = \frac{\frac{1}{b}}{\frac{J}{b}s + 1} \tag{4.3}$$

Hence, the mechanical time constant can be found as $T_s^\omega = \frac{J}{b} = 0.686$. In a similar manner, the electrical time constants can be derived to $T_s^d = \frac{L_d}{R} = 0.0043$ and $T_s^q = \frac{L_q}{R} = 0.0087$. As the electrical time constants are two orders of magnitudes smaller than the mechanical time constant, a separation of the controller into a cascaded control system can be done in order to simplify the control design. The inner loop controls the currents using the voltages as control signals, while the outer loop controls the speed using torque setpoint as control signal. The mechanical time constant depends on the moment of inertia and an extra load inertia would thus change it. However, since only an increase of the moment of inertia is possible, the time constant may only become larger and it is therefore no problem for the cascaded setup. Since the two controllers are decoupled, different sampling times can and should be used. The sampling time of the current controller is set to 250 μs, almost 20 times faster than the fastest time constant of the current dynamics. The speed controller is set to a sampling period that is ten times longer, i.e. 2.5 ms. This leads to a faster controller relative to the system time constant than for the current controller. This is motivated by the desire to quickly be able to compensate for torque loads.

A schematic diagram of the control structure is shown in Figure 4.1. In addition to the two controllers, a transformation from torque setpoint to current setpoints must be done between them. This transformation is described in Section 4.1.2.

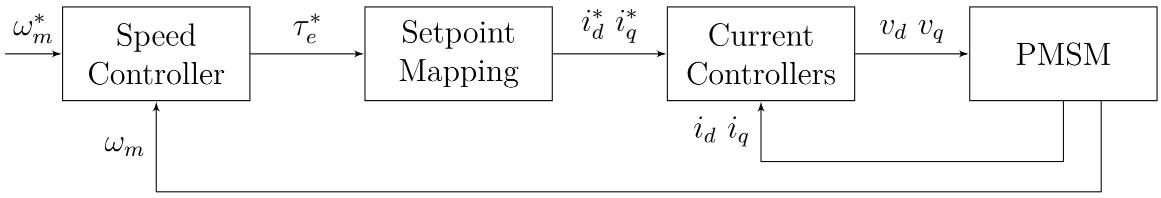

Figure 4.1: Control structure. A cascaded control is used with inner loop controlling current and outer loop controlling rotor speed. A mapping from torque setpoint to current setpoints between the speed and current controller is made.

4.1.1 Current controller

When the system dynamics are well-known, good performance for speed control of the PMSM system can be achieved with a PI regulator implementation in both the current and speed controller. The presumption is that only the moment of inertia is unknown and from (3.19) and (3.20) it can be seen that the electrical dynamics are not affected by a change in moment of inertia. With this as motivation, it is decided to stick with a PI regulator as current controller for both the d- and q-current loop.

4.1.2 Setpoint mapping

The control signal for the speed controller is a torque setpoint for the electromechanical torque. This desired torque is obtained by regulating the currents so that the actual torque

$$\tau_e = n_p(\psi_R + (L_d - L_q)i_d)i_q \tag{4.4}$$

tracks the torque setpoint. The currents are controlled by the current controllers, which need setpoint values for the currents, i_d and i_q. A mapping from torque setpoint to current setpoints is thus needed. For optimal use of the electrical power, the desired torque should be achieved by as low total current as possible. In other words, an optimization problem to minimize $i_d^2 + i_q^2$ with the constraint (4.4) should be solved for the mapping. However, the term $(L_d - L_q)$ is about one hundred times smaller than ψ_R and high currents in d-direction are therefore needed for this term to have a considerable impact on the resulting torque. A simple rule for the current setpoints is therefore to use all the current in the torque-creating q-direction instead. This leads to the mapping

$$\begin{aligned} i_d^* &= 0 \\ i_q^* &= \frac{\tau_e^*}{n_p \psi_R} \end{aligned} \tag{4.5}$$

4.2 Speed controller version 1

Using (4.2) as a model for the speed control, it can be seen that the relation from total resulting applied torque, $\tau_{tot} = \tau_e - \tau_L$, to speed, ω_m, is through a first order system with transfer function

$$\frac{\omega_m(s)}{\tau_{tot}(s)} = \frac{\frac{1}{J}}{s + \frac{b}{J}}. \tag{4.6}$$

An exact discretization of this model with sample time T_ω leads to the difference equation

$$\omega_m(k) = e^{-\frac{b}{J}T_\omega}\omega_m(k-1) - \frac{1}{b}(e^{-\frac{b}{J}T_\omega} - 1)\tau_{tot}(k-1) \tag{4.7}$$

By introducing $a_d = e^{-\frac{b}{J}T_\omega}$ and $b_d = -\frac{1}{b}(e^{-\frac{b}{J}T_\omega} - 1)$, (4.7) can be written as

$$\begin{aligned} \omega_m(k) &= a_d\omega_m(k-1) + b_d\tau_{tot}(k-1) \\ &= a_d\omega_m(k-1) + b_d\tau_e(k-1) - b_d\tau_L(k-1) \end{aligned} \tag{4.8}$$

This expression can then be used to parametrize the equation.

4.2.1 Control law using three parameters

To design an MRAC based on the model presented in (4.8), a reference model of the same order is needed. The reference model is thus constructed as

$$\omega_{\text{ref}}(k) = a_{\text{ref}}\omega_{\text{ref}}(k-1) + b_{\text{ref}}\omega_m^*(k-1) \tag{4.9}$$

which will give a behavior of a first order system from the speed setpoint, ω_m^*, to the speed of the reference model, ω_{ref}. The reference model can be described by the transfer function

$$\frac{\omega_m(z)}{\omega_m^*(z)} = \frac{b_{\text{ref}}}{z - a_{\text{ref}}}. \tag{4.10}$$

Since a steady state gain of amplitude 1 from setpoint to actual speed is desired, the relationship $b_{\text{ref}} = 1 - a_{\text{ref}}$ must hold. Hence, only the pole placement of a_{ref}

is needed when designing the reference model. Subtracting the term $a_{\text{ref}}\omega_m(k-1)$ from both sides in (4.8) gives

$$\omega_m(k) - a_{\text{ref}}\omega_m(k-1) = (a_d - a_{\text{ref}})\omega_m(k-1) + b_d\tau_{tot}(k-1). \tag{4.11}$$

The error dynamics for the tracking error, $e(k) = \omega_m(k) - \omega_{\text{ref}}(k)$, can be calculated by subtracting (4.9) from (4.11)

$$\begin{aligned} e(k) - a_{\text{ref}}e(k-1) = (a_d - a_{\text{ref}})\omega_m(k-1) + b_d\tau_e(k-1) \\ - b_d\tau_L(k-1) - b_{\text{ref}}\omega_m^*(k-1). \end{aligned} \tag{4.12}$$

In order to make the error dynamics become

$$e(k) = a_{\text{ref}}e(k-1) \tag{4.13}$$

and hence make the error go to zero (since the reference model parameter a_{ref} will be chosen such that the dynamics are stable) the control law can be formed as

$$\tau_e^*(k) = \frac{1}{b_d}\Big((a_{\text{ref}} - a_d)\omega_m(k) + b_{\text{ref}}\omega_m^*(k) + b_d\tau_L(k)\Big). \tag{4.14}$$

In order for the control law (4.14) to be realizable, the unknown values of a_d, b_d and $\tau_L(k)$ must be estimated. By introducing a vector of regressors

$$\varphi^T(k) = [\omega_m(k-1)\ \tau_e^*(k-1)\ \ -1] \tag{4.15}$$

and a parameter vector

$$\theta^T(k-1) = [a_d\ \ b_d\ \ b_d\tau_L(k-1)], \tag{4.16}$$

(4.8) can be written as

$$\omega_m(k) = \varphi^T(k)\theta(k-1). \tag{4.17}$$

With this parametrization it is possible to iteratively estimate the unknown parameters a_d, b_d and $b_d\tau_L(k-1)$. The control signal is then calculated according to the control law (4.14) using the latest available estimates (4.16) of the unknown parameters. That is

$$\tau_e^*(k) = \frac{1}{\hat{\theta}_2(k)}\Big((a_{\text{ref}} - \hat{\theta}_1(k))\omega_m(k) + b_{\text{ref}}\omega_m^*(k) + \hat{\theta}_3(k)\Big) \tag{4.18}$$

where $\hat{\theta}_1(k)$, $\hat{\theta}_2(k)$ and $\hat{\theta}_3(k)$ are the first, second and third elements in $\hat{\theta}(k)$ respectively.

4.2.2 RLS based MRAC

The parameter estimation can be made using the RLS algorithm described in Section 2.3. The parameter vector (4.16) is thus estimated at each iteration using the

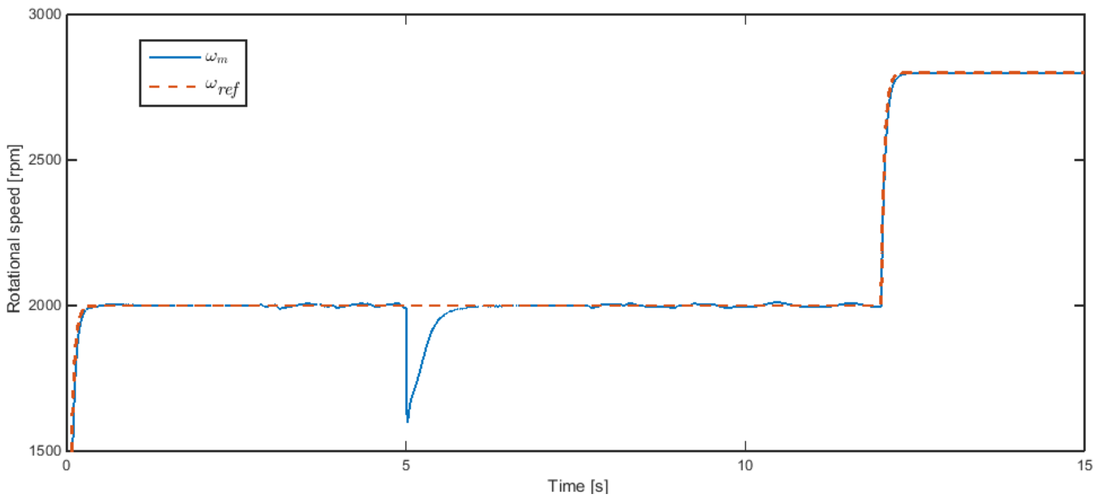

Figure 4.2: Performance of the speed controller using the RLS algorithm to estimate the parameters. The controller setup is $\lambda = 0.99$, $a_{\text{ref}} = 0.95$, $P_0 = 100\text{I}$ and $\hat{\theta}_0 = [1\ 100\ 0]^T$. The actual speed (solid blue line) follows the reference model speed (dashed red line) very well with the exception for the seconds right after the torque load step.

regressor vector (4.15) according to the RLS-algorithm

$$\epsilon(k) = \omega_m(k) - \varphi^T(k)\hat{\theta}(k-1) \tag{4.19}$$

$$P(k) = \frac{1}{\lambda}\Big(P(k-1) - \frac{P(k-1)\varphi(k)\varphi^T(k)P(k-1)}{\lambda + \varphi^T(k)P(k-1)\varphi(k)}\Big) \tag{4.20}$$

$$K(k) = P(k)\varphi(k) \tag{4.21}$$

$$\hat{\theta}(k) = \hat{\theta}(k-1) + K(k)\epsilon(k) \tag{4.22}$$

The responsiveness of the controller toward changed process dynamics and torque load is tuned using the forgetting factor λ. A lower value of λ gives a quicker response to changed dynamics and torque load but will at the same time lead to a controller that is less robust against noise. Furthermore, initial values for the estimated parameters, $\hat{\theta}_0$, and the covariance matrix, P_0, are needed.

By simulating the system when using the RLS based MRAC, it becomes evident that the controller performs quite well. Figure 4.2 shows the rotor speed, ω_m, during the simulation which starts with a setpoint step change to 2000 rpm with only the moment of inertia of the rotor, J_r, as the total moment of inertia, J. After 5 seconds a torque load step with amplitude 0.1 Nm is added. At 10 seconds a load inertia of $24J_r$ is added making the total moment of inertia to $25J_r$. After another 2 seconds, the setpoint is changed to 2800 rpm. It can be stated that the rotor speed follows the reference model speed, ω_{ref}, well for the whole sequence, with the exception for the seconds right after the torque load step.

4.2.3 Kalman filter based MRAC

In the RLS-algorithm the forgetting factor, λ, reflects the rate of adaptation for the parameters. However, speed control performance requires a faster response time towards changes in torque loads than in moment of inertia. It would therefore be beneficial to assign different adaptation rates for the parameters. This is possible using the Kalman filter, described in Section 2.4, by assigning different variances for the parameters in the process noise matrix, Q. As the changes in the parameters are unknown, a random walk model is used to describe their transition, leading to the dynamic model and measurement model according to

$$\theta(k) = \theta(k-1) + w(k) \tag{4.23}$$

$$\omega_m(k) = \varphi^T(k)\theta(k) + v(k) \tag{4.24}$$

Here $w(k) \sim \mathcal{N}(0, Q)$ captures the variations of the parameters and $v(k) \sim \mathcal{N}(0, R)$ the measurement noise. Applying these models for the Kalman filter equations (2.30)-(2.36) leads to the following algorithm.
Prediction step:

$$P^-(k) = P(k-1) + Q \tag{4.25}$$

Update step:

$$\epsilon(k) = \omega_m(k) - \varphi^T(k)\hat{\theta}(k-1) \tag{4.26}$$

$$S = \varphi^T(k)P^-(k)\varphi(k) + R \tag{4.27}$$

$$K = P^-(k)\varphi(k)S^{-1} \tag{4.28}$$

$$\hat{\theta}(k) = \hat{\theta}(k-1) + K\epsilon(k) \tag{4.29}$$

$$P(k) = P^-(k) - KSK^T \tag{4.30}$$

The measurement noise variance in the filter, R, is set according to the estimated value in Section 3.2.5 so that the only tuning parameter for the filter is the process noise matrix, Q. The control law derived in Section 4.2.1 is still used but now with the parameter estimation algorithm carried out according to (4.25)-(4.30).

With the possibility to assign different adaptation rates for the parameters, a quicker response for disturbance rejection can be obtained, as indicated by Figure 4.3, which shows the same simulation as Figure 4.2 but now with the Kalman filter as estimation method.

4.2.4 Perturbation signal

As stated earlier, the derived controllers seem to perform well at first glance. However, a problem arises at steady state when the setpoint and torque load is kept constant and the dynamics do not change. This can be seen in Figure 4.4, where the parameter estimations are shown for a simulation with setpoint kept at 2000 rpm and no torque load is added. Here, an exponential drift in the parameters appear, which in the end leads to an unstable behavior for the system. The drift can be

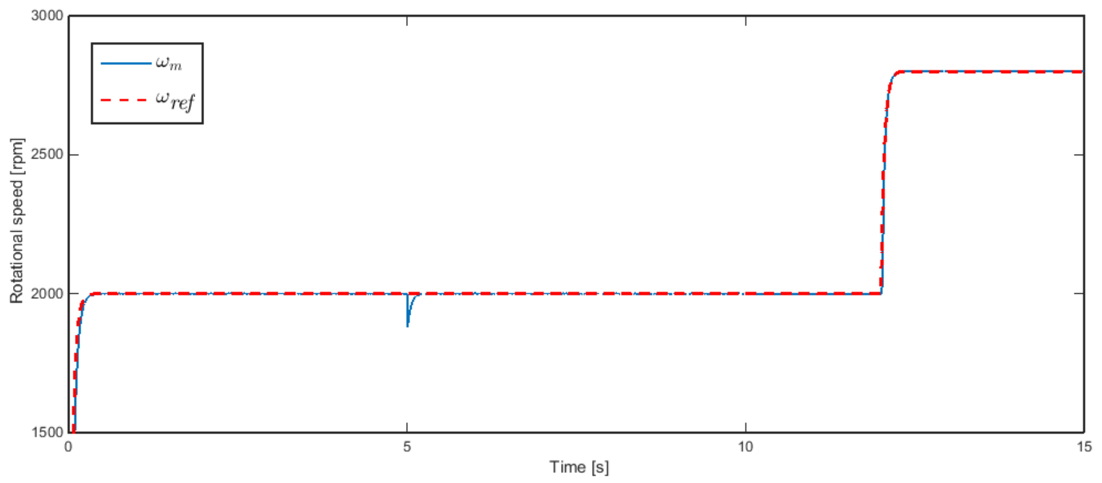

Figure 4.3: Performance of the speed controller when Kalman filter is used as parameter estimator for the standard test case. The controller setup is $R = 0.01$, $Q = \text{diag}\{0.001^2, 0.1^2, 0.1^2\}$, $a_{\text{ref}} = 0.95$, $P_0 = 100\text{I}$ and $\hat{\theta}_0 = [1\ \ 100\ \ 0]^T$. The actual speed (solid blue line) is now less affected by the load step at 5 seconds. The reference model speed is showed as a dashed red line.

explained by an insufficient excitation of the system [17], since at steady state the direction of the regressor vector, φ, does not change. In order to get a regressor vector that is richer in variations, a perturbation signal, δ_τ, is added to the calculated torque setpoint. The perturbation should affect the ideal closed-loop performance as little as possible while still ensuring persistent excitation of the system. A few different noise characteristics were investigated in order to achieve enough excitation while minimizing speed variance in steady state. The rather short cyclic torque perturbation signal

$$\delta_\tau = \begin{bmatrix} 0 & 1 & -2 & -1 & 2 & 0 & -1 & 2 & 1 & -2 \end{bmatrix} \times 10^{-3}\ \text{Nm} \tag{4.31}$$

was found to yield the most desirable result of the ones investigated, see Figure 4.5. The signal has zero mean and results in a low speed variance. One may argue that a more random signal would be favorable, and hence a pseudo-random number generator (PRNG) to distribute the perturbation signal was tested. The random variates from the PRNG was mapped to achieve different distributions of the perturbation signal, such as uniform and normal distribution. However, when the PRNG was used no improvement on the parameter estimates was found and the steady state speed variance increased, compared to when using the cyclic sequence. The sequence in Figure 4.5 was therefore chosen as perturbation signal.

The adaptive speed controller following this design is presented in Figure 4.6. The control law block computes the unperturbed control signal, τ_e^u, according to (4.18) which is added to the perturbation signal, δ_τ, to form the torque setpoint, τ_e^*. The parameter estimation is performed according to (4.19)-(4.22) for the RLS based MRAC and (4.25)-(4.30) for the Kalman filter based MRAC with the regressor vector formed as in (4.15).

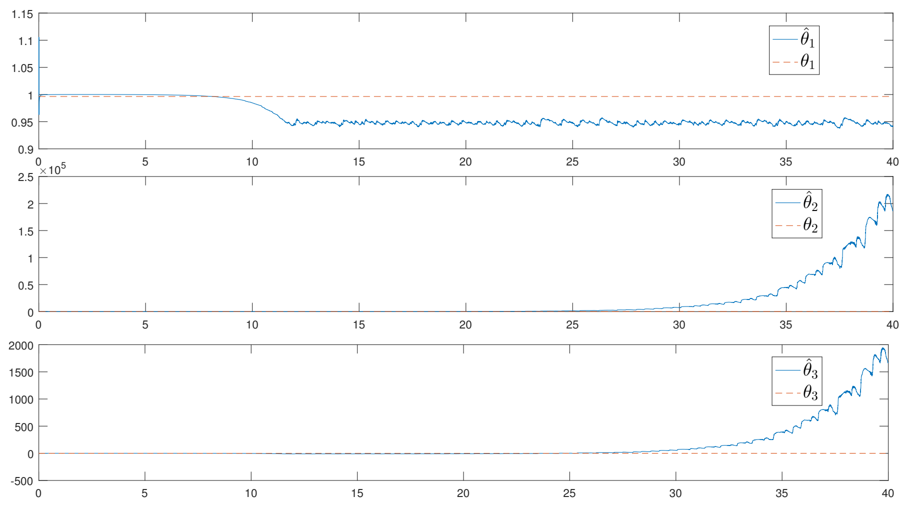

Figure 4.4: Parameter estimates during steady state at 2000 rpm using an unperturbed control signal. The estimated parameters are shown in solid blue lines and the true values in dashed red. Both $\hat{\theta}_2$ and $\hat{\theta}_3$ show an exponential drift.

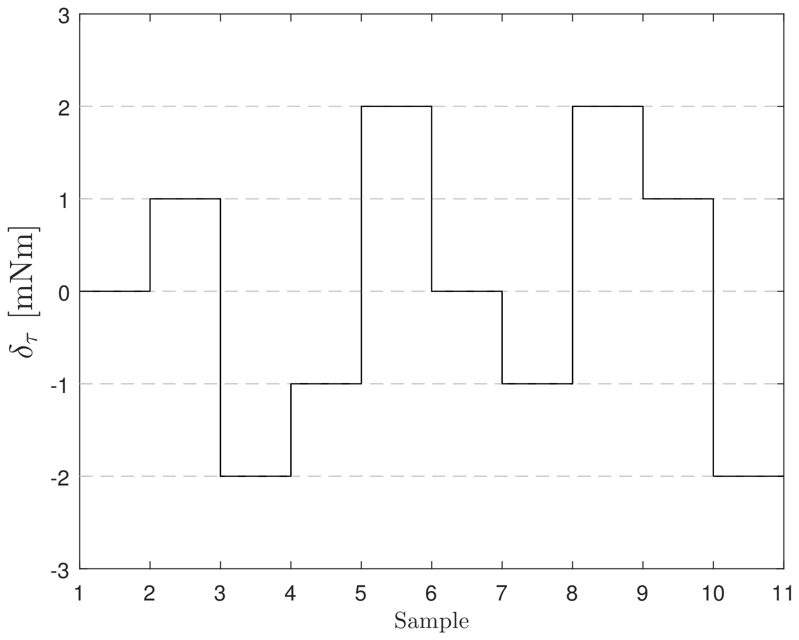

Figure 4.5: Perturbation signal sequence added to the control signal to ensure excitation of the system.

With the same simulation setup as for Figure 4.4, the drift of estimated parameters does not occur when the perturbation signal has been added, see Figure 4.7. Furthermore, the performance of the controller in the standard test case is the same as when not adding any perturbation.

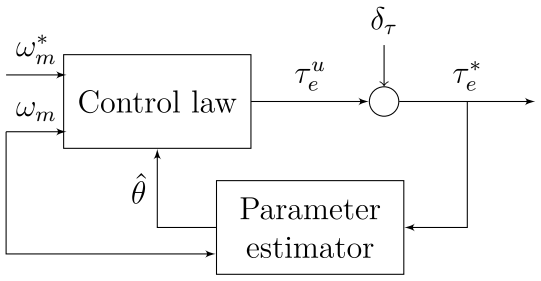

Figure 4.6: Speed controller scheme.

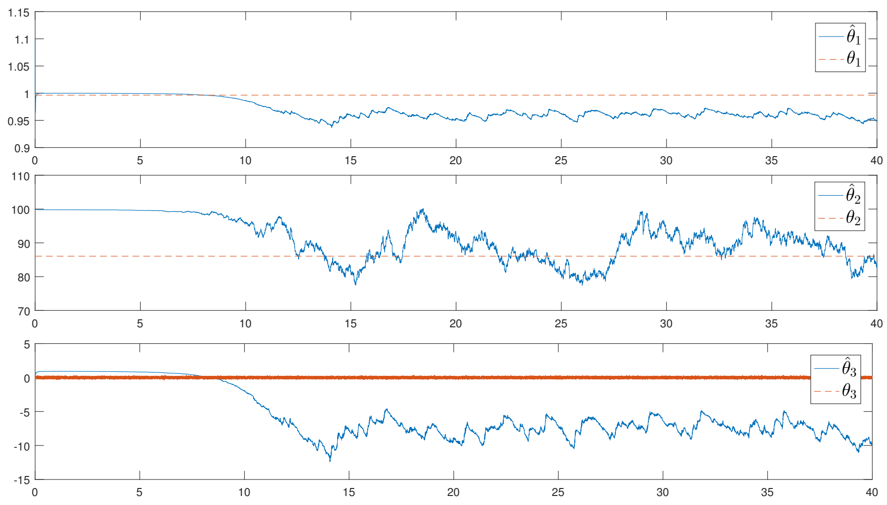

Figure 4.7: Parameter estimates during steady state at 2000 rpm when a perturbation is added to the control signal. The estimated parameters are shown in solid blue lines and the true values in dashed red.

4.3 Speed controller version 2

The controller derived in Section 4.2 showed very good performance in simulations, but when implementing it on the real system the parameter estimations escalated to large numbers leading to an unstable system. Since the only considered changes in the system are the moment of inertia and torque load, it should be possible to capture these variations in only two parameters. A two parameter estimator would also make the controller easier to tune and reduce its complexity. Still, the parameter estimations are performed using either the RLS described in Section 4.2.2 or the Kalman filter described in Section 4.2.3. The perturbation signal in Section 4.2.4 is also used to ensure persistence of excitation.

4.3.1 Control law using two parameters

To achieve a model using only two parameters some rearrangement of the original speed dynamics model (4.7) is needed. First, it should be noted that b_d can be expressed as

$$b_d = \frac{1}{b}(a_d - 1). \tag{4.32}$$

In the new parametrization, the speed difference between each sample is modelled instead of the actual speed. To achieve a model for the speed difference, the previous speed, $\omega_m(k-1)$, is subtracted from (4.7). This, together with (4.32), leads to the expression

$$\begin{aligned}\omega_m(k) - \omega_m(k-1) &= (a_d - 1)\omega_m(k-1) - \frac{1}{b}(a_d-1)(\tau_e(k-1) - \tau_L(k-1)) \\ &= (a_d - 1)\left(\omega_m(k-1) - \frac{1}{b}\tau_e(k-1)\right) + \frac{1}{b}(a_d-1)\tau_L(k-1).\end{aligned} \tag{4.33}$$

By assuming that an estimation of the frictional constant b is available, the regressor vector and parameter vector can be formed according to

$$\varphi^T(k) = \begin{bmatrix}\frac{1}{\hat{b}} & \omega_m(k-1) - \frac{1}{\hat{b}}\tau_e(k-1)\end{bmatrix} \tag{4.34}$$

$$\theta^T(k) = \begin{bmatrix}(a_d-1)\tau_L(k-1) & (a_d-1)\end{bmatrix} \tag{4.35}$$

and the prediction of the rotor speed becomes

$$\omega_m(k) = \omega_m(k-1) + \varphi^T(k)\theta(k). \tag{4.36}$$

Again using the MRAC approach to construct a control law, a reference model as in (4.9) is used. In order to obtain the tracking error dynamics, the term $a_{\text{ref}}\omega_m(k-1)$ is subtracted from the expression (4.36) to get

$$\omega_m(k) - a_{\text{ref}}\omega_m(k-1) = (1 - a_{\text{ref}})\omega_m(k-1) + \varphi^T(k)\theta(k). \tag{4.37}$$

A subtraction of the reference model (4.9) then leads to the tracking error dynamics

$$e(k) - a_{\text{ref}}e(k-1) = (1 - a_{\text{ref}})\omega_m(k-1) + \varphi^T(k)\theta(k) - b_{\text{ref}}\omega_m^*(k-1). \tag{4.38}$$

In the same way as in (4.12), the right hand side is desired to be set to zero, i.e.

$$(1 - a_{\text{ref}})\omega_m(k-1) + \frac{\theta_1(k)}{\hat{b}} + \theta_2(k)\left(\omega_m(k-1) - \frac{1}{\hat{b}}\tau_e(k-1)\right) - b_{\text{ref}}\omega_m^*(k-1) = 0. \tag{4.39}$$

Solving (4.39) and inserting the parameter estimates leads to the control law

$$\tau_e^u(k) = \frac{\hat{b}}{\hat{\theta}_2(k)}\left((\hat{\theta}_2(k) + 1 - a_{\text{ref}})\omega_m(k) - b_{\text{ref}}\omega_m^*(k) + \frac{\hat{\theta}_1(k)}{\hat{b}}\right). \tag{4.40}$$

The structure of the speed controller is still as shown in Figure 4.6, but now using (4.40) as control law and the parametrization according to (4.34) and (4.35).

To get a better understanding of how the estimates affect the control law, a reformulation can be done. Using the relationship of the model reference parameters, $b_{\text{ref}} = 1 - a_{\text{ref}}$, (4.40) becomes

$$\begin{aligned}\tau_e^u &= \frac{\hat{b}}{\hat{\theta}_2(k)}\left((\hat{\theta}_2(k) + 1 - a_{\text{ref}})\omega_m(k) - (1 - a_{\text{ref}})\omega_m^*(k) + \frac{\hat{\theta}_1(k)}{\hat{b}}\right) \\ &= \frac{\hat{b}(a_{\text{ref}} - 1)}{\hat{\theta}_2(k)}\Big(\omega_m^*(k) - \omega_m(k)\Big) + \hat{b}\omega_m(k) + \frac{\hat{\theta}_1(k)}{\hat{\theta}_2(k)}.\end{aligned} \tag{4.41}$$

The controller can hence be inferred to have three essential parts: a proportional gain of the speed error, $\frac{\hat{b}(a_{\text{ref}}-1)}{\hat{\theta}_2(k)}$, a feedback gain of the speed, $\hat{b}$, and a feedforward of the estimated torque load, $\frac{\hat{\theta}_1(k)}{\hat{\theta}_2(k)}$. The estimate of θ_1 is therefore only important in its relation to $\hat{\theta}_2$, while the magnitude of $\hat{\theta}_2$ determines the proportional gain.

4.3.2 Upper limit of θ

When implementing the controller in the real system, a problem arose when applying very large and quickly changing load torques. The estimation of θ_2 drifted towards zero and whenever it reached non-negative values the system became unstable. When examining the parameter a_d, it becomes obvious that the condition

$$a_d = e^{-\frac{b}{J}T_\omega} < 1 \tag{4.42}$$

holds since b, J and T_ω all are strictly positive values. From (4.42) and the definition $\theta_2 = a_d - 1$, it can be concluded that θ_2 must be negative. With this in mind, a condition of negativeness for updating the estimate of θ_2 is applied. In other words, if the next estimation for θ_2 is non-negative, only $\hat{\theta}_1$ is updated while $\hat{\theta}_2$ is kept at its previous value. Furthermore, $\theta_1 = \theta_2\tau_L$ and it is assumed that the load torque has a breaking effect on the rotor, i.e. $\tau_L \geq 0$. The upper bound for θ_1 is thus zero. In the same manner as for $\hat{\theta}_2$, a condition on updating $\hat{\theta}_1$ can be used in order to ensure that it never reaches potive values. The upper bound for $\hat{\theta}_1$ is important especially for the RLS based controller when a torque load step is applied. Without the bound, an oscillation of the control signal appears in the transient.

Inspecting (4.41) again, it can be seen that $\hat{b}(a_{\text{ref}} - 1)$ is negative. Since a positive proportional gain is desired, the necessity of a negative value for $\hat{\theta}_2$ also becomes evident.

4.3.3 Final control algorithm

The final controller follows the MRAC scheme presented in Section 4.3.1 with control law (4.40) and the parametrization according to (4.34) and (4.35). The calculated torque setpoint is then perturbed using the perturbation signal described in Section 4.2.4 in order to ensure excitation of the system. The parameter estimates are guaranteed to stay below their upper bounds using the conditions for updating their values described in Section 4.3.2. Two different methods are used for the parameter estimation, RLS and Kalman filter, which are described in Sections 4.2.2 and 4.2.3 respectively. The RLS based MRAC (RLS-MRAC) is described in Algorithm 1 and the Kalman filter based MRAC (KF-MRAC) is presented in Algorithm 2.

Algorithm 1: RLS based MRAC (RLS-MRAC)

```
   Input: $\omega_m(k)$, $\lambda$, $P_0$, $\hat{\theta}_0$, $a_{\text{ref}}$, $b_{\text{ref}}$, $\hat{b}$
   Output: $\tau_e^*(k)$
   Memory: $\hat{\theta}(k)$, $P(k)$, $\omega_m(k-1)$, $\tau_e^*(k-1)$
   Initialization: $P(k) = P_0$, $\hat{\theta}(k) = \theta_0$, $\omega_m(k-1) = 0$, $\tau_e^*(k-1) = 0$
1  Construct regressor vector as $\varphi^T(k) = [\frac{1}{\hat{b}} \quad \omega_m(k-1) - \frac{1}{\hat{b}}\tau_e(k-1)]$
2  Compute estimation error $\epsilon = \omega_m(k) - \varphi^T(k)\hat{\theta}(k-1)$
3  Update covariance matrix
     $P(k) = \frac{1}{\lambda}\left(P(k-1) - K\varphi^T(k)P(k-1)\right)$
4  Calculate correction gain
     $K = P(k)\varphi(k)$
5  Update parameter estimate
     $\hat{\theta}(k) = \hat{\theta}(k-1) + K\epsilon$
6  Ensure non-positiveness for $\hat{\theta}_1$
     if $\hat{\theta}_1(k) > 0$ then
     |  $\hat{\theta}_1(k) = \hat{\theta}_1(k-1)$
   end
7  Ensure negativeness for $\hat{\theta}_2$
     if $\hat{\theta}_2(k) \geq 0$ then
     |  $\hat{\theta}_2(k) = \hat{\theta}_2(k-1)$
   end
8  Compute unperturbed torque setpoint
     $\tau_e^u(k) = \frac{\hat{b}}{\hat{\theta}_2(k)}\left((\hat{\theta}_2(k) + 1 - a_{\text{ref}})\omega_m(k) - b_{\text{ref}}\omega_m^*(k) + \frac{\hat{\theta}_1(k)}{\hat{b}}\right)$
9  Add perturbation to the calculated torque setpoint
     $\tau_e^*(k) = \tau_e^u(k) + \delta_\tau, \quad \delta_\tau$ as in (4.31)
10 Set $\omega_m(k-1) \leftarrow \omega_m(k)$, $\tau_e^*(k-1) \leftarrow \tau_e^*(k)$, $\hat{\theta}(k) \leftarrow \hat{\theta}(k+1)$ and
     $P(k) \leftarrow P(k+1)$
11 At next iteration, start over from step 1
```

Algorithm 2: Kalman filter based MRAC (KF-MRAC)

```
   Input: $\omega_m(k)$, $Q$, $R$, $P_0$, $\hat{\theta}_0$, $a_{\text{ref}}$, $b_{\text{ref}}$, $\hat{b}$
   Output: $\tau_e^*(k)$
   Memory: $\hat{\theta}(k)$, $P(k)$, $\omega_m(k-1)$, $\tau_e^*(k-1)$
   Initialization: $P = P_0$, $\hat{\theta} = \theta_0$, $\omega_m(k-1) = 0$, $\tau_e^*(k-1) = 0$
1  Construct regressor vector as $\varphi^T(k) = [\frac{1}{\hat{b}} \quad \omega_m(k-1) - \frac{1}{\hat{b}}\tau_e(k-1)]$
2  Compute estimation error $\epsilon = \omega_m(k) - \varphi^T(k)\hat{\theta}(k-1)$
3  Perform prediction step in Kalman filter
     $P^-(k) = P(k-1) + Q$
4  Perform update step in Kalman filter
     $S = R + \varphi^T(k)P^-(k)\varphi(k)$
     $K = P^-(k)\varphi(k)S^{-1}$
     $P(k) = P^-(k) - KSK^T$
     $\hat{\theta}(k) = \hat{\theta}(k-1) + K\epsilon$
5  Ensure non-positiveness for $\hat{\theta}_1$
     if $\hat{\theta}_1(k) > 0$ then
       $\hat{\theta}_1(k) = \hat{\theta}_1(k-1)$
   end
6  Ensure negativeness for $\hat{\theta}_2$
     if $\hat{\theta}_2(k) \geq 0$ then
       $\hat{\theta}_2(k) = \hat{\theta}_2(k-1)$
   end
7  Compute next torque setpoint
     $\tau_e^u(k) = \frac{\hat{b}}{\hat{\theta}_2(k)}\left((\hat{\theta}_2(k) + 1 - a_{\text{ref}})\omega_m(k) - b_{\text{ref}}\omega_m^*(k) + \frac{\hat{\theta}_1(k)}{\hat{b}}\right)$
8  Add noise to the calculated torque setpoint
     $\tau_e^*(k) = \tau_e^u(k) + \delta_\tau, \quad \delta_\tau$ as in (4.31)
9  Set $\omega_m(k-1) \leftarrow \omega_m(k)$, $\tau_e^*(k-1) \leftarrow \tau_e^*(k)$, $\hat{\theta}(k) \leftarrow \hat{\theta}(k+1)$ and
     $P(k) \leftarrow P(k+1)$
10 At next iteration, start over from step 1
```

5

Results

This chapter presents the project results, evaluating the closed-loop performance for the developed controllers as well as for a PI controller. In Section 5.1, the simulation results can be found. The two MRAC approaches, presented in Section 4.3.3, are compared to the PI controller in terms of setpoint tracking and disturbance rejection. Since the true parameter values are known in simulation and not in the physical experiment, especially the estimation abilities of the two different methods are of interest. In Section 5.2, the controllers are compared more in depth when implemented and run on the physical system. Besides the setpoint tracking and disturbance rejection for different cases of moment of inertia, the sensitiveness of frictional constant estimation, $\hat{b}$, is investigated.

5.1 Simulation results

Figure 5.1 shows the rotor speed from a simulation of the system using the RLS-MRAC described in Algorithm 1, the KF-MRAC described in Algorithm 2 and a PI controller. The simulation starts with a setpoint step change to 2000 rpm with only the moment of inertia of the rotor, J_r, as the total moment of inertia, J. After 5 seconds a torque load step with amplitude 0.1 Nm is added. At 10 seconds a load inertia of $24J_r$ is added making the total moment of inertia to $25J_r$. After another 2 seconds, the setpoint is changed to 2800 rpm. In this case, a_{ref} is set to 0.8 leading to a rise time (10% to 90%) for the reference model of 0.025 s and the PI controller is tuned to have the same rise time when the moment of inertia is J_r. Initial values for the estimation are $P_0 = \text{I}$ and $\hat{\theta}_0 = [0 \enspace -0.01]^T$ and the estimation of b is perfect, i.e. $\hat{b} = b$. The Kalman filter measurement noise matrix R is set to the identified measurement noise in Section 3.2.5, i.e. 0.01. The parameters $\lambda = 0.985$ and $Q_0 = [10^{-4} \enspace 10^{-6}]$ have been tuned to get the best performance for each controller.

The rise time and overshoot for the three controllers, both for the first and second setpoint step change, are presented in Table 5.1. The two adaptive controllers both quickly show the required step response, with a rise time of 0.025 s and adjust the dynamics after the changed moment of inertia. No overshoot appears at the second setpoint step and the rise time is only slightly longer. However, the step response using PI control after changed moment of inertia is much slower with the rise time of 0.38 s and results in an overshoot of 47.5%. This clearly indicates that the two adaptive controllers can compensate for changed moment of inertia and retain the

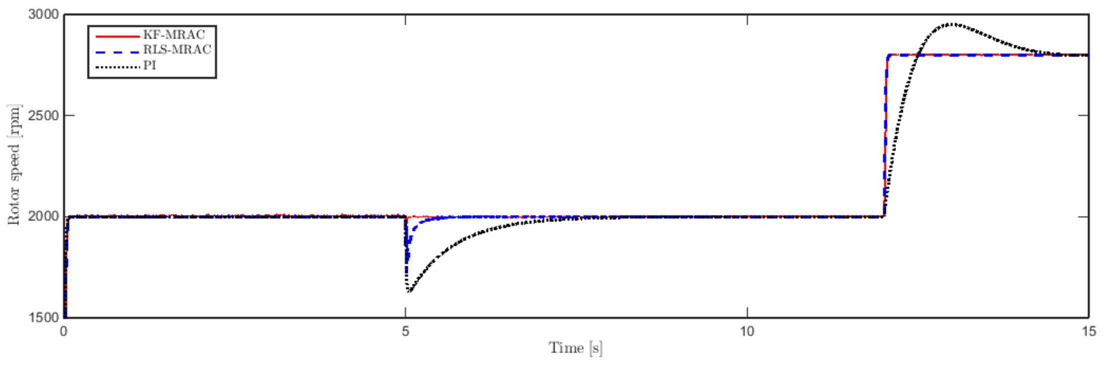

Figure 5.1: Rotor speed during standard test case using KF-MRAC (solid red line), RLS-MRAC (dashed blue line) and PI (dotted black line). Setpoint speed changes from 2000 to 2800 after 12 seconds.

Controller	Rise time 1 [s]	Overshoot 1 [%]	Rise time 2 [s]	Overshoot 2 [%]
KF-MRAC	0.025	0.2	0.035	0
RLS-MRAC	0.025	0.1	0.030	0
PI	0.025	0.1	0.38	47.5

Table 5.1: Step response characteristic values for the three controllers for the standard test case for step 1 from 0 to 2000 rpm at 0 seconds and step 2 from 2000 to 2800 rpm at 12 seconds.

desired closed-loop dynamics, something that the PI controller is incapable of.

The big difference between the two adaptive controllers is the disturbance rejection, where the KF-MRAC has the better performance. This can more clearly be seen in Figure 5.2 which shows a close-up look on the speed when the torque load is applied and by examining Table 5.2 which presents recovery time (time from load step to 99% of the steady-state speed is reached) and speed drop. The KF-MRAC both have a shorter recovery time, 0.025 s compared to 0.3 s, and a smaller speed drop, 94 rpm compared to 277 rpm.

The quick response for the KF-MRAC can be understood by examining the parameter estimations for the simulated sequence, shown in Figure 5.3. The parameter containing the information of the torque load, θ_1, is rapidly changed at 5 seconds when the load step is applied and the Kalman filter estimate adjusts immediately. A closer look at θ_1 right after the load step can be seen in Figure 5.4 where this

Controller	Recovery time [s]	Speed drop [rpm]
KF-MRAC	0.025	94
RLS-MRAC	0.300	277
PI	2.019	372

Table 5.2: Values for disturbance rejection for the three controllers. Recovery time is the time from load step to 99% of the setpoint speed is reached. Speed drop is the maximum deviation of the speed from setpoint speed.

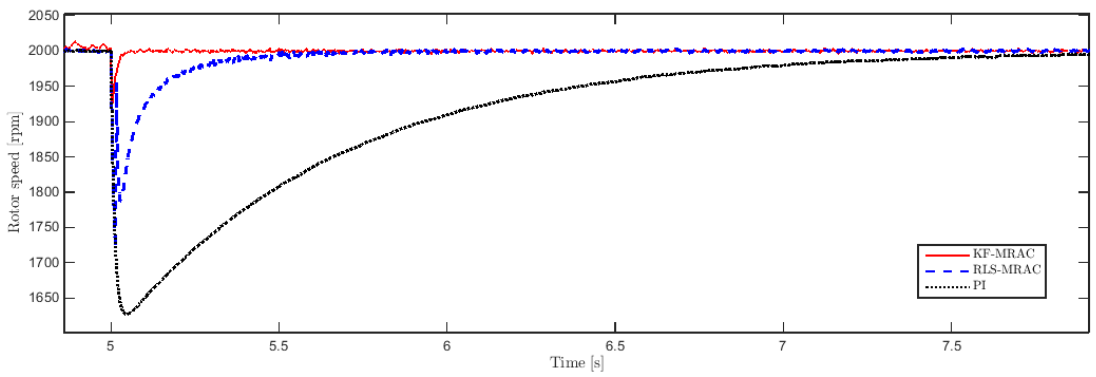

Figure 5.2: A closer look at the speed drop when the load step is applied. There is a clear performance difference of the KF-MRAC (solid red line), the RLS-MRAC (dashed blue line) and the PI (dotted black line).

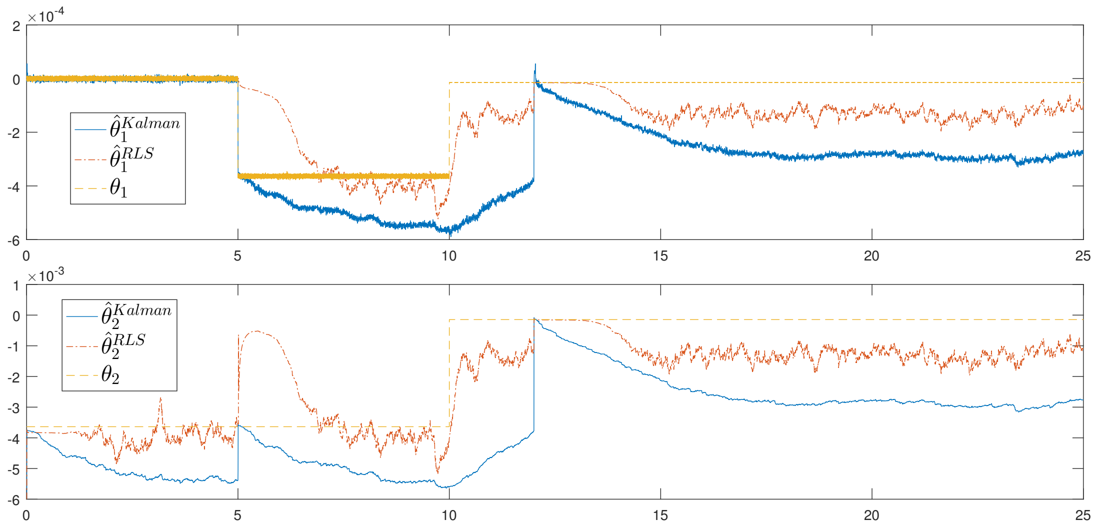

Figure 5.3: Parameters estimated in simulation using KF-MRAC (solid blue line) and RLS-MRAC (dash-dotted red line). The true parameters are shown as a dashed yellow line.

quick response becomes obvious. Looking again at Figure 5.3 it can be noted that the Kalman filter estimates seem more accurate shortly after a setpoint change or torque load step has occurred. However, when the system reaches steady state, the estimates start to drift down until settling at an offset from the true value. The RLS estimates move more slowly towards the true parameter values after a change but later on settle closer to the true values compared to the Kalman filter estimates.

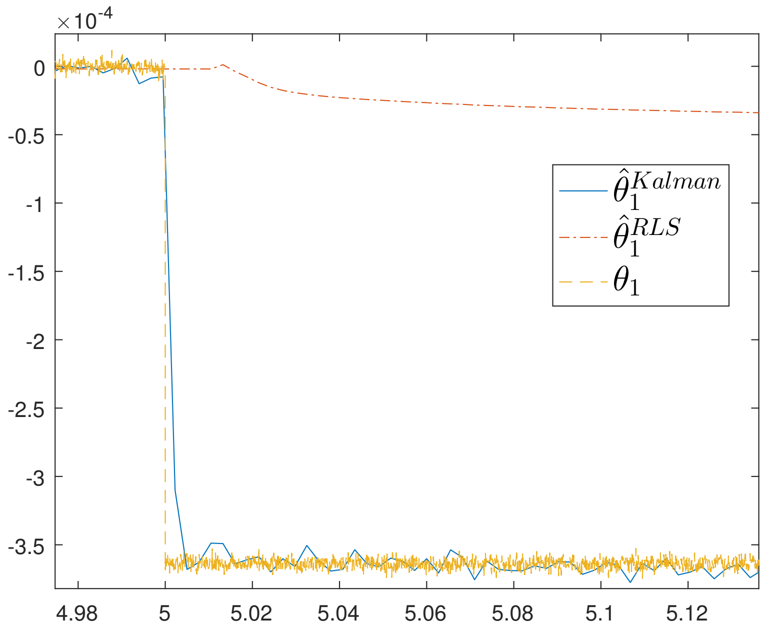

Figure 5.4: Estimation of θ_1 after the added torque load using Kalman filter (solid blue line) and RLS (dash-dotted red line). The true parameters is shown as a dashed yellow line.

5.2 Experimental results

The control design was tested using a motor test bench at Aros electronics. The test bench consisted of a motor that could apply load torques. The PMSM to be tested was attached to the axis of the test bench using a spring. With the additional motor, the inertia of the system was changed and thus now unknown. The total moment of inertia can be described as

$$J_1 = J_r + J_l \tag{5.1}$$

where J_r is the known moment of inertia of the subjected test rotor and J_l is the added moment of inertia of the load motor, which is estimated to be about the same size as J_r. In order to get a fair comparison between the controllers, the PI was tuned to get a step response similar to the reference model for the moment of inertia J_1. A reference model with $a_{\text{ref}} = 0.985$ was used, leading to a rise time of about 0.4 seconds. The initial values for the parameter estimates in both adaptive controllers were $\theta_0 = [0 \ \ -0.01]^T$ and $P_0 = 10^{-6}I$. The Kalman filter was tuned to $R = 0.01$, $Q = \text{diag}\{10^{-13},\ 5 \times 10^{-12}\}$ and the forgetting factor for the RLS was $\lambda = 0.985$. The estimation of b was set according to the identified value in section 3.2.4, i.e. 4.2281×10^{-5}.

To investigate the performance of the controllers for varying moment of inertia, the tests were performed using two different cases. First the tests were performed with the moment of inertia of the two rotors, J_1. For the second part of the tests, a steel

Controller	Rise time [s]	Overshoot [%]	Settling time [s]
KF-MRAC	0.38	0	0.94
RLS-MRAC	0.47	0.14	1.38
PI	0.36	0.59	0.83

Table 5.3: Step response characteristic values for the three controllers when the moment of inertia is J_1.

disc was attached to the axis in order to increase the moment of inertia. The added moment of inertia was 24 times bigger than the test rotor inertia, giving a total moment of inertia

$$J_2 = 25J_r + J_l \approx 13J_1. \tag{5.2}$$

The controllers were evaluated by their ability to follow the setpoint signal. This was done by comparing their steady-state speeds, their step responses and their ability to compensate for a torque load. The purpose of the tests was to demonstrate the ability of the developed controller to compensate for a varying moment of inertia and retain the original dynamics.

5.2.1 Step response

During the first test case, which can be seen in Figure 5.5, the moment of inertia was J_1 and a setpoint of 2000 rpm was given. Both the adaptive controllers directly follow the desired response, but a slightly slower response to reach the final value can be seen for the RLS-MRAC. As the PI controller is tuned to have the same rise time as the reference model for this moment of inertia, the similar behavior for the three controllers is expected. The rise time, overshoot and settling time (time to reach and stay within 1% of final value) for each controller are presented in Table 5.3. From this table, the similar dynamics for all controllers are easily seen.

However, a closer look at the steady-state speed, shown in Figure 5.6, reveals a problem with the KF-MRAC. Here it can be seen that the KF-MRAC produces a steady state speed offset of almost 10 rpm. This behavior has been present throughout the entire testing procedure, where the offset seems to have a linear relationship with the speed leading to an offset of about 0.5% of the setpoint. The cause for it has not been found, but a possible reason for the different steady-state behavior of the two estimation strategies is that they settle at different parameter estimation values. This can be seen in Figure 5.7 which illustrates the parameter estimates in the step response sequence. This difference in parameter estimation was also present in the simulation, as can be seen in Figure 5.3, where it showed that the RLS settled closer to the true parameter values.

Figure 5.8 demonstrates the step response for the same test but now with the moment of inertia changed to J_2. This clearly shows the changed behavior of the PI controller while the step responses for the adaptive controllers are similar to the previous case, thus indicating that the adaptive controllers have retained their dy-

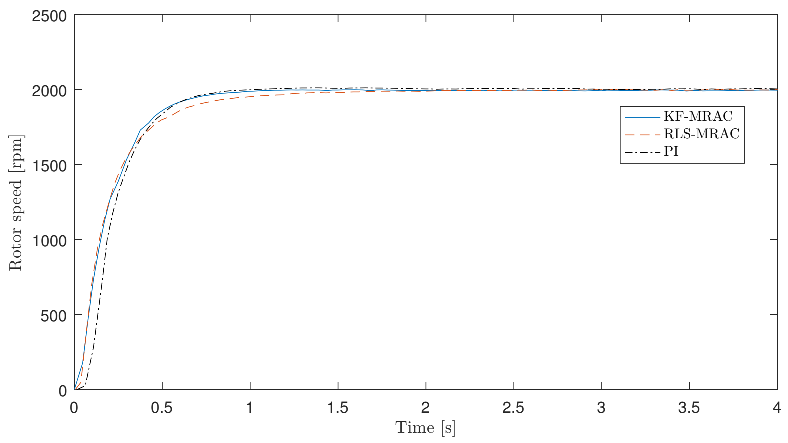

Figure 5.5: Step response for the case when the moment of inertia is J_1 using KF-MRAC (solid blue line), RLS-MRAC (dashed red line) and PI controller (dash-dotted black line).

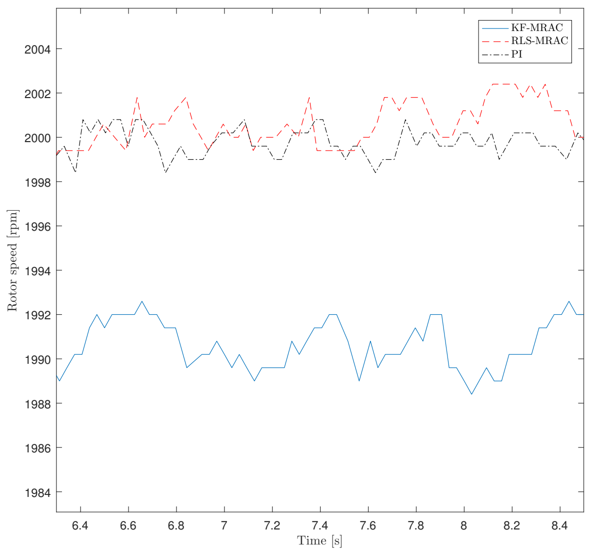

Figure 5.6: Steady-state speed for the KF-MRAC (solid blue line), the RLS-MRAC (dashed red line) and PI (dash-dotted black line). The controller using Kalman filter for estimation settles at an offset of about 10 rpm from the setpoint value.

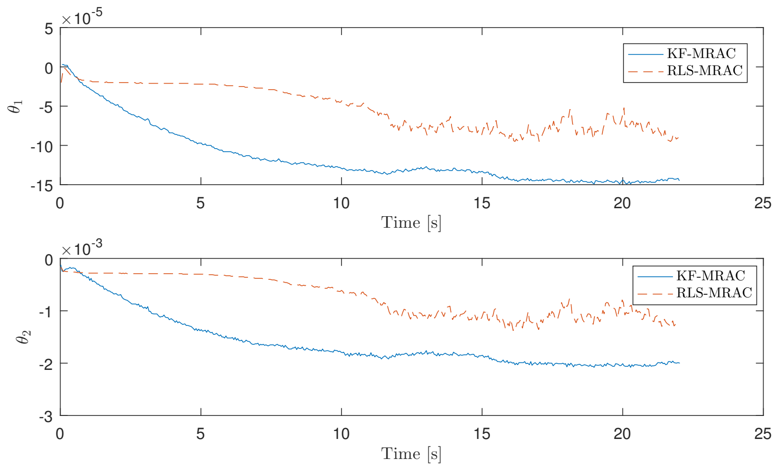

Figure 5.7: Parameter estimates during the step response sequence. The Kalman filter estimates (solid blue lines) settles at about twice the value of the RLS estimates (dashed red lines).

Controller	Rise time [s]	Overshoot [%]	Settling time [s]
KF-MRAC	0.45	0.14	0.91
RLS-MRAC	0.44	0.23	1.16
PI	0.66	12.02	3.78

Table 5.4: Step response characteristic values for the three controllers when the moment of inertia is J_2.

namics. The step response characteristic values are presented in Table 5.4. While difference in rise and settling time for the adaptive controllers is small, PI has an increase in rise time and settling time from 0.359 to 0.656 and from 0.827 to 3.78 respectively. Furthermore, the overshoot for the PI becomes 12 % while the adaptive controllers have a negligible overshoot.

5.2.2 Disturbance rejection

To evaluate the controllers' ability to reject load disturbances, a test was carried out where the motor was running at 2000 rpm and a torque load step was applied. For the case with the lower moment of inertia, J_1, a torque of 0.1 Nm was applied and for the bigger moment of inertia, J_2, a torque of 0.2 Nm was applied.

Figure 5.9 shows the test with moment of inertia J_1 and Table 5.5 presents performance values for the test. The test with moment of inertia J_2 is shown in Figure 5.10 and performance values for this test are presented in Table 5.6. From this, it is clear that both the MRACs are significantly better than the PI regarding disturbance rejection, even when moment of inertia is J_1 - the case which the PI controller was tuned for. Also here, the robustness against changed moment of inertia, when using the developed controllers, becomes obvious as the responses for the two cases of J_1 and J_2 are very similar. The response using the PI, however, is worse for the case

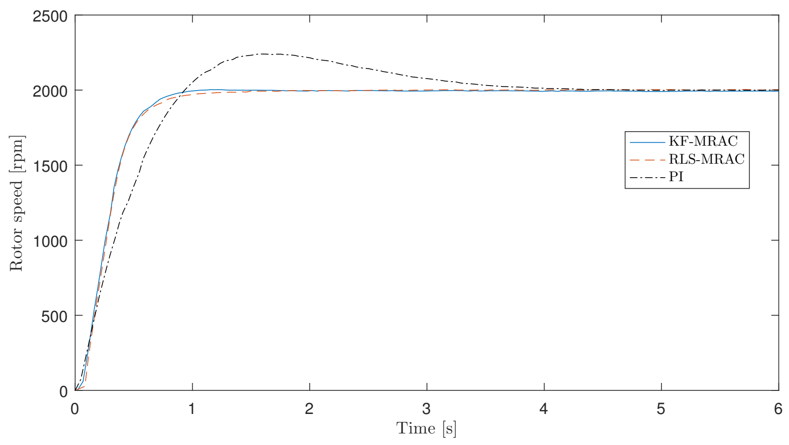

Figure 5.8: Step response for the case when the moment of inertia is J_2 using the KF-MRAC (solid blue line), the RLS-MRAC (dashed red line) and PI controller (dash-dotted black line).

Controller	Recovery time [s]	Speed drop [rpm]
KF-MRAC	0.36	86
RLS-MRAC	0.44	40
PI	1.9	200

Table 5.5: Values for disturbance rejection of a load step for the three controllers when the moment of inertia is J_1. Recovery time is the time from load step to 99% of the steady-state speed is reached. Speed drop is the maximum deviation of the speed from setpoint speed.

with higher inertia and again shows the dependency on moment of inertia.

The KF-MRAC is faster to fully compensate for the load, especially when moment of inertia is J_2, but the RLS-MRAC has a smaller speed drop. The fast compensation for the KF-MRAC can be understood by investigating the parameter estimates. From 4.41 it is clear that the estimation of torque load is $\hat{\tau}_L = \frac{\hat{\theta}_1}{\hat{\theta}_2}$. This factor is plotted in Figure 5.11, where the load step is made at 1 second. The Kalman filter estimate quickly finds the new higher value, while the RLS estimate more slowly converges to it. Still, the RLS-MRAC is able to reject the load better regarding

Controller	Recovery time [s]	Speed drop [rpm]
KF-MRAC	0.36	105
RLS-MRAC	0.75	38
PI	3.00	390

Table 5.6: Values for disturbance rejection of a load step for the three controllers when the moment of inertia is J_2. Recovery time is the time from load step to 99% of the steady-state speed is reached. Speed drop is the maximum deviation of the speed from setpoint speed.

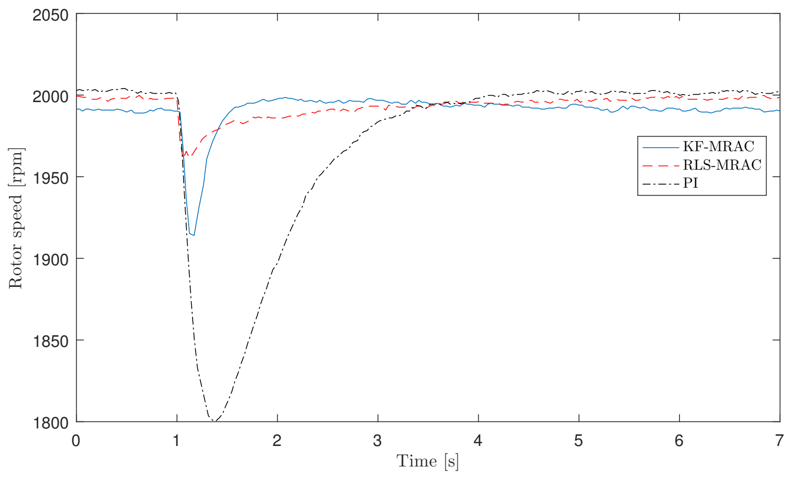

Figure 5.9: Load disturbance rejection for the case when the moment of inertia is J_1 using the KF-MRAC (solid blue line), the RLS-MRAC (dashed red line) and PI controller (dash-dotted black line).

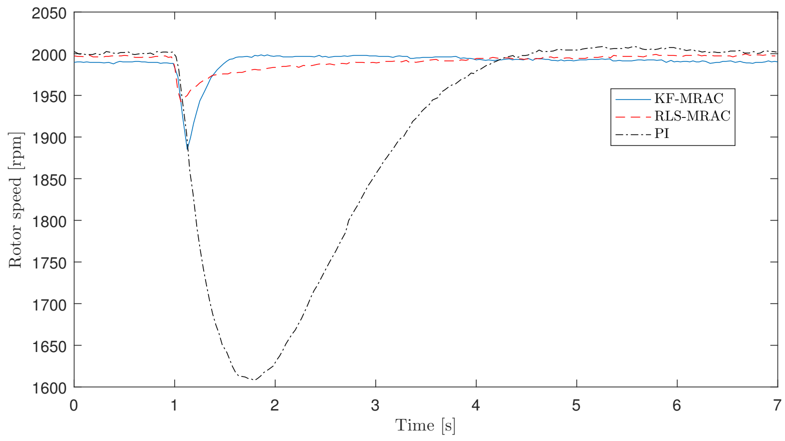

Figure 5.10: Load disturbance rejection for the case when the moment of inertia is J_2 using the KF-MRAC (solid blue line), the RLS-MRAC (dashed red line) and PI controller (dash-dotted black line).

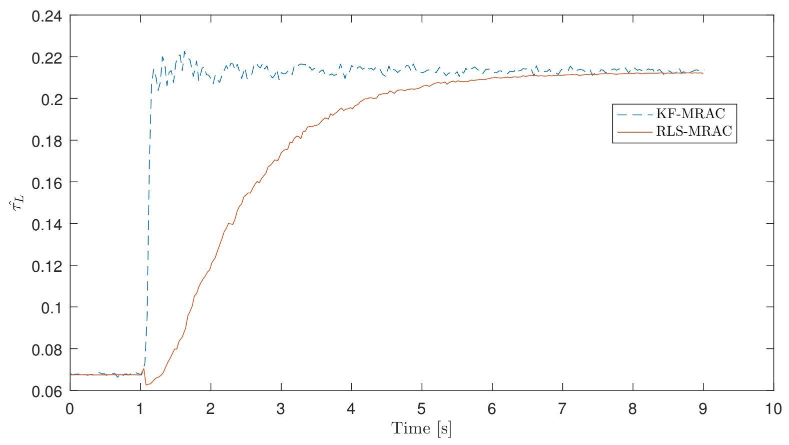

Figure 5.11: Load torque estimation during the load step. The Kalman filter estimation (dashed blue line) quickly settles at the new higher load value. The RLS estimation (solid red line) more slowly converge to the new load value.

the speed drop. The reason for this is its high increase of proportional gain. The proportional gain is $\frac{\hat{b}(a_{\text{ref}-1})}{\hat{\theta}_2}$, as seen in (4.41). In Figure 5.12 the proportional gain is shown, again the load step is made at 1 second. The gain shows a dramatic increase for the RLS-MRAC at the time of load step. This leads to a very aggressive controller against changes in speed which results in the small speed drop.

The high proportional gain, however, leads to a more nervous control signal. This can be seen in Figure 5.13 where the control signals for the three speed controllers are shown. Both the MRACs increase their torque reference more aggressively when the load step occurs, compared to the control signal of the PI. The KF-MRAC has a peak in the beginning but quickly settles at the new steady-state, while the RLS-MRAC oscillates more around it until the torque load estimate has settled and proportional gain has reduced.

5.2.3 Changed moment of inertia during runtime

In Section 5.2.1 the step response for different values of moment of inertia was investigated. However, to test the ability to adjust to changed moment of inertia when the motor is running and the estimates thus have settled for a certain moment of inertia, a different test is needed. For practical reasons, it was not possible to actually change the moment of inertia during runtime but instead an alternative test was performed to resemble this as good as possible. First, with the moment of inertia being J_1, the motor was run at 2000 rpm using the MRAC until the estimates settled at steady values. The parameter estimates and the covariance matrix were then saved. Then, the moment of inertia was changed to J_2 and the motor was again run at 2000 rpm, now using the PI as controller. With initial values for the MRAC set to the previously saved values for J_1, the motor controller was switched from PI to the MRAC. Two seconds after the switch, a setpoint step to 2500 rpm was performed. This was done for both the RLS-MRAC and the KF-MRAC. The

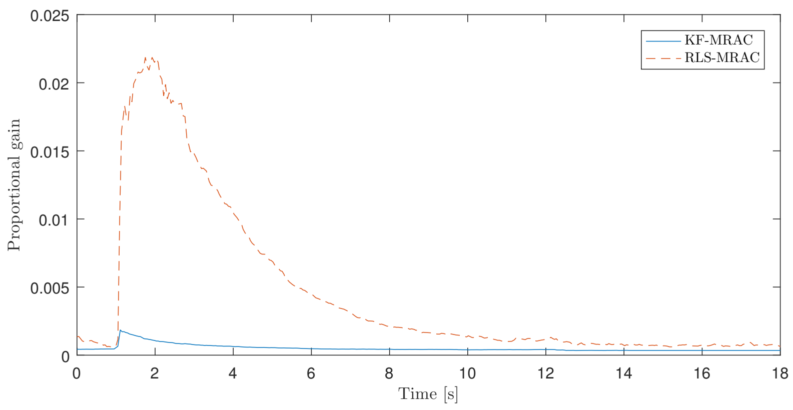

Figure 5.12: Proportional gain in control signal during the load step for the case of moment of inertia J_1. For the RLS-MRAC (dashed red line), the highest value during the transient is about 30 times its steady-state proportional gain. The KF-MRAC (solid blue line) show a less dramatic increase, reaching at most 5 times its steady-state value.

test was thus designed to resemble a sequence where the motor is run at 2000 rpm with the moment of inertia J_1, at a point the moment of inertia is changed to J_2 and two seconds after the change, a setpoint step to 2500 rpm is made.

To get a good picture of the sequence, the data from the first sequence using MRAC is merged with the second sequence. The first three seconds is with J_1 as moment of inertia running at steady state with a setpoint of 2000 rpm and from three seconds the moment of inertia is J_2. In Figure 5.14 the rotor speed is shown for the merged sequences. No deviation in speed from the steady-state value is appearing when the moment of inertia is changed for any of the controllers. Furthermore, they both react on the setpoint step change with satisfying dynamics. The rise time for the RLS-MRAC is exactly as desired, i.e. 0.4 seconds, while the KF-MRAC shows a somewhat slower behavior having a rise time of 0.43 seconds.

Figures 5.15 and 5.16 illustrates the parameter estimates for the merged sequences. The change of moment of inertia at 3 seconds is not immediately captured by any of the estimation methods, since a change of moment of inertia by itself will not affect the torque required to uphold the rotational speed. However, after five seconds when the setpoint step change occurs, the controllers quickly find new estimates. As can be seen in Figure 5.14, correct dynamics are obtained as a result. It is interesting to note that the same behaviour for the parameter estimates as in simulation can be seen when changing setpoint. At that time instant both the Kalman filter estimates and the RLS estimates jump up to about the same value. In simulation, this was known to be the true parameter value and this can be suspected to be the case also for the physical system. With the inertia J_2 and frictional constant $\hat{b}$, the expected estimate would be $\theta_2 = -1.4 \times 10^{-4}$. Comparing this value with Figure 5.16, it can be seen that the estimates approach this value immediately after the setpoint step change. The Kalman filter estimate rises to -2.1×10^{-4} and the RLS estimate to -1.5×10^{-4}. After the step, the estimates drift down to settle at values below the

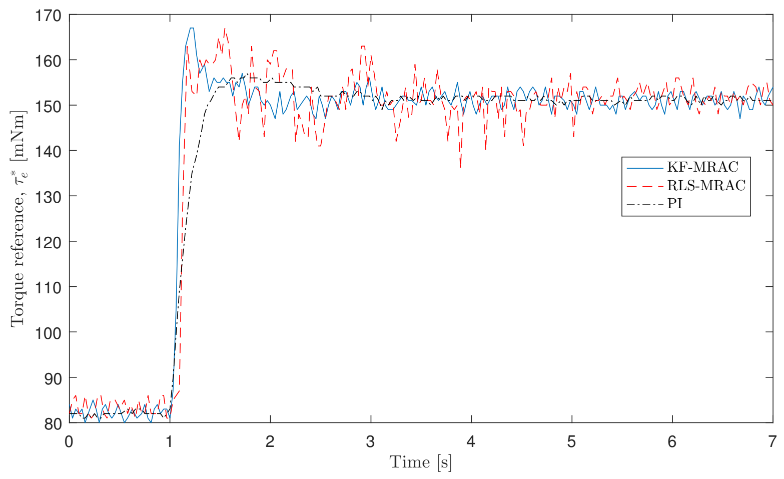

Figure 5.13: Torque setpoint signal during the load step. Both the KF-MRAC (solid blue line) and RLS-MRAC (dashed red line) has a more aggressive response to the load than the PI (dash-dotted black line). The control signal from the RLS-MRAC shows an oscillatory behaviour during the transient period after the load step.

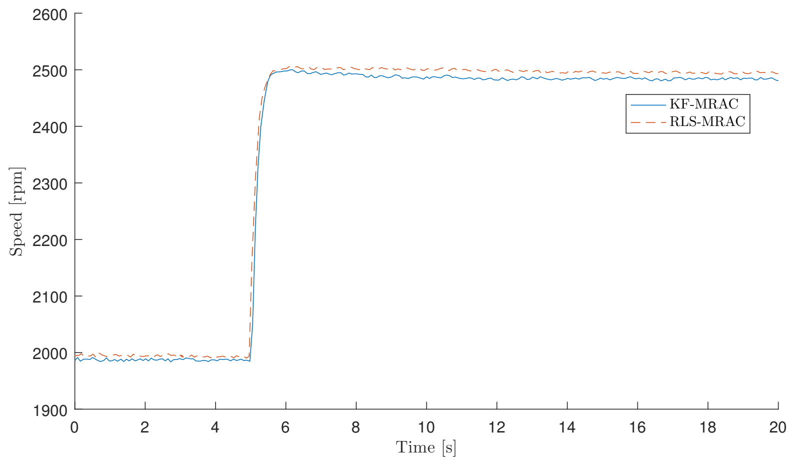

Figure 5.14: Rotor speed during a sequence of changed moment of inertia. At 3 seconds the moment of inertia is changed from J_1 to J_2 and at 5 seconds a setpoint step from 2000 rpm to 2500 rpm is made. Both the KF-MRAC (solid blue line) and the RLS-MRAC (dashed red line) show high robustness against changed moment of inertia.

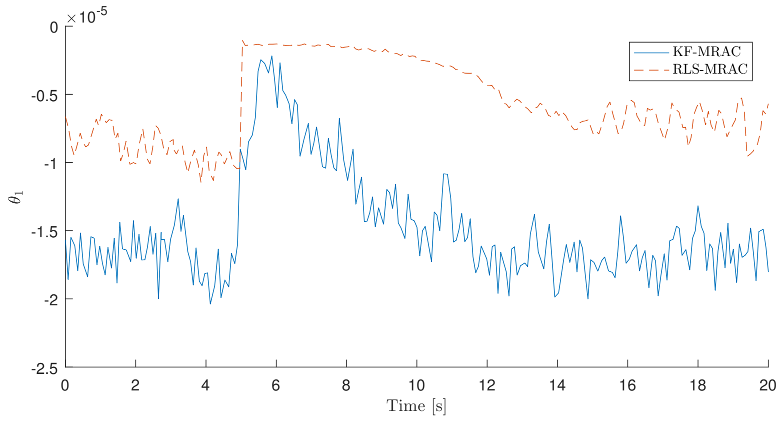

Figure 5.15: Estimation of θ_1 during the sequence of changed moment of inertia. At the time of the setpoint step both estimates directly rise to nearly the same value. Quickly after the step, the Kalman filter estimate (solid blue line) drifts down to the same offset as before. The RLS estimate (dashed red line) more slowly settles at its offset, which is at a higher level than the Kalman filter estimate.

assumed true value. The Kalman filter estimate settles further below than the RLS estimate.

5.2.4 Sensitivity for various $\hat{b}$

Since $\hat{b}$ is a part of the regressor vector, the parameter estimations depend on it. It is thus important to have a reasonable estimation of the frictional constant, b. To investigate how sensitive the controllers are for different values of $\hat{b}$, a step response test was performed using various values. The step response is shown in Figure 5.17 and Figure 5.18 using KF-MRAC and RLS-MRAC, respectively. Besides the identified value of b, 4.2281×10^{-5}, the estimate is set to both two times as big and half the value. In both cases the controller behaves similarly when $\hat{b}$ is approximately b and $0.5b$, with just a little bit slower step response for $\hat{b} \approx 0.5b$. Another difference can be found by closely examining the steady-state offset for the KF-MRAC which is 3 rpm bigger for $\hat{b} \approx 0.5b$. For $\hat{b} \approx 2b$ the offset becomes positive and an overshoot also occurred in the step response. This was also the case for the RLS-MRAC when $\hat{b} \approx 2b$, seen in Figure 5.18. However, this offset is still smaller than the one received using KF-MRAC. When larger or smaller values of $\hat{b}$ is used, such as $4b$ and $0.25b$, the system becomes unstable.

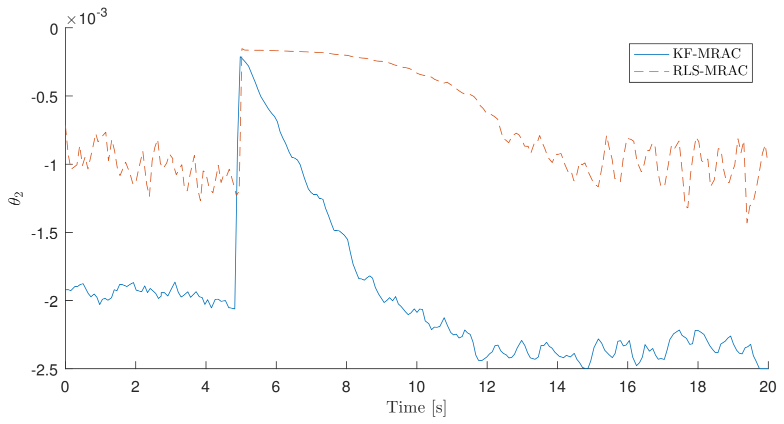

Figure 5.16: Estimation of θ_2 during the sequence of changed moment of inertia. The behavior is similar to the one for $\hat{\theta}_1$. Both the Kalman filter estimate (solid blue line) and the RLS estimate (dashed red line) rise to a value close to the expected true value, -1.4×10^{-4}, at the time of setpoint change.

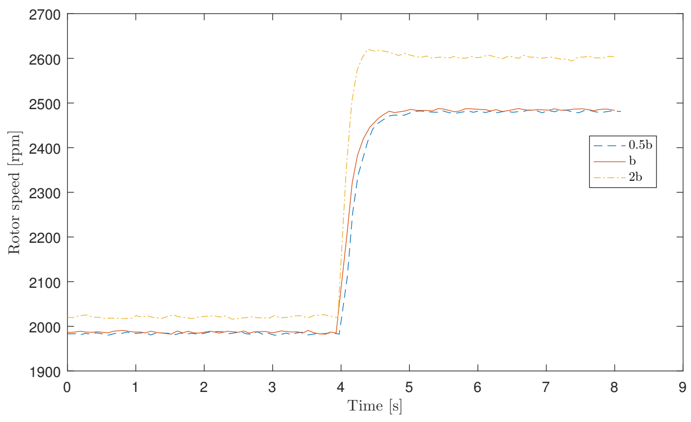

Figure 5.17: Step response from 2000 to 2500 rpm using KF-MRAC for different $\hat{b}$.

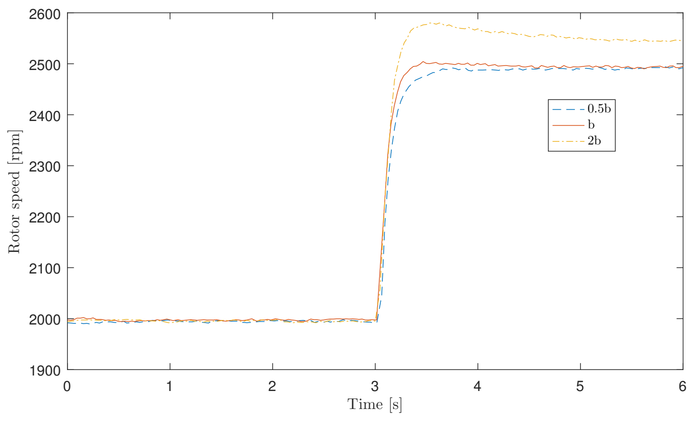

Figure 5.18: Step response from 2000 to 2500 rpm using RLS-MRAC for different $\hat{b}$.

6 Discussion and Future Work

The two controllers developed in the project perform equally well regarding the step response and adapt to big changes in moments of inertia in a way that a PI controller cannot do. With the ability to estimate the plant dynamics, the defined reference model can always be traced closely, meaning that the behavior toward changes in setpoint is independent on the moment of inertia.

When it comes to torque load compensation, the MRACs show a much better performance than the PI. It can be concluded that the KF-MRAC is faster to fully compensate for an applied torque load, which is due to its ability to quickly estimate the new load. With the RLS-MRAC, however, the speed drop is smaller because of the high increase of proportional gain. This comes with the price of a more nervous control signal in the transient period after the load step. The PI controller has the smoothest control signal, but this also leads to a much slower response to added torque load. For an increased moment of inertia, the PI controlled motor shows an even deeper speed drop and is slower on its way back to the speed setpoint. This dependence on moment of inertia is not something that is apparent for the adaptive controllers. Here, again, the developed controllers thus show their ability to retain the dynamics for varying moment of inertia. The reason for the quality difference in disturbance rejection for the two adaptive controllers can be explained by the possibility to assign different adaptation rates for the two parameters in the Kalman filter. By modifying the RLS to make the tuning of different adaptation rates possible, the disturbance rejection may be enhanced to faster regain its speed. In [18] an RLS scheme is presented where multiple forgetting factors are used, leading to more degrees of freedom for tuning the adaptation rates which might make this possible.

The KF-MRAC has a steady state speed offset at constant speeds. The offset is not very big, about 0.5% of the setpoint for all of the speed span, but it is nonetheless an important drawback. In some applications, the robustness against changed moment of inertia and quick response to torque loads may be more important, but for applications where good steady-state tracking of the speed is crucial the KF-MRAC presented here is not an alternative. An explanation for this behavior has not been established, but one hypothesis is that the different parameter estimations for the two controllers is the reason. The estimates using Kalman filter and RLS settle at different values and simulations indicate that the RLS estimates are closer to the true values. In simulation, the parameter offset for the Kalman filter did not result in any offset in speed but when applied on the actual system this was the case. In the physical experiments, when physical phenomena that are not modelled appear, the effect of the estimation offset may become more important leading to tracking errors. For stable parameter estimation, excitation of the system is necessary and the added perturbation signal is crucial. However, the system might still not be excited enough. When significant changes was made for the system, such as setpoint steps and torque load changes, the true parameter values were temporarily found but the estimates then quickly started to drift to an offset. This indicates that more excitation of the system leads to better estimations. A possible improvement can be made by thoroughly investigating the role of the perturbation signal, with the goal to excite the system while ensuring minimal disturbance on the output. Another possible fix could be to only estimate θ_2 (and indirectly the moment of inertia) when a setpoint step change occurs, since changes in moment of inertia are difficult to detect at constant speeds. This method would lock the moment of inertia estimate shortly after the new speed setpoint is reached, i.e. when the value is believed to be close to the true value. Yet another solution for the steady-state offset might be to introduce integrating effect in the control law. This may, however, worsen its quick responsiveness to disturbances.

For the two controllers presented in this report an estimate of the frictional constant, b, is needed. The choice of $\hat{b}$ has a considerable effect on the behaviour of the controller and the identification of b thus becomes rather important. In the real system, the frictional constant may not actually be constant but rather dependent on the rotor speed. Since b is considered constant in the controllers and the controllers are fairly sensitive to different values for $\hat{b}$, it becomes obvious that the nonlinear traits of the friction may be an issue. One possible solution for this is to use lookup tables for $\hat{b}$ for different speeds, in an effort to obtain better performance for the entire speed range. A possibly even better solution for getting rid of the sensitivity of $\hat{b}$ would be to estimate it online. The first version of the speed controller did not rely on any identification of the frictional constant, but instead indirectly estimated it using a third parameter. In simulations, the desired closed-loop performance using this controller was successfully achieved. Despite the promising behavior in simulations, no stable controller was found when trying to implement this method on the real system. As it would be highly advantageous to not rely on an estimation of the frictional constant, further investigations on the implementation issues are of interest.

7 Conclusion

The two developed motor control methods, RLS-MRAC and KF-MRAC, are much more robust against changes in moment of inertia compared to a PI controller. With both methods, a defined reference model can be followed accurately for a large span of moments of inertia. They also perform very well against torque load disturbances, where the RLS-MRAC has the lowest speed drop while the KF-MRAC has a slightly bigger drop but is instead faster to regain its speed. The KF-MRAC has a more stable control signal, compared to the RLS-MRAC control signal which is prone to oscillations immediately after a load change. Excitation of the system is crucial in order for the parameter estimates not to drift and make the closed-loop system stable. A perturbation signal is added to the torque setpoint in the presented algorithms, leading to stable parameter estimation. Even so, a steady-state speed offset is appearing for the KF-MRAC. This is a drawback that the RLS-MRAC does not have.

Bibliography

[1] Chandler, J. "PMSM technology in high performance variable speed applications." Control Design White Paper, 2006

[2] Su, Y.X., Chun H.Z., Bao Y.D. "Automatic disturbances rejection controller for precise motion control of permanent-magnet synchronous motors." IEEE Transactions on Industrial Electronics 52.3: 814-823, 2005

[3] Krishnan, R. "Permanent magnet synchronous and brushless DC motor drives". CRC Press, 2010

[4] Dobbs, E. R. "Basic Electromagnetism". Springer Netherlands, 1993

[5] Harnefors, L. "Control of variable-speed drives". Applied Signal Processing and Control, Department of Electronics, Mälardalen University, 2002.

[6] Egardt, B. "Nonlinear and adaptive control". Department of Signals and Systems, Chalmers University of Technology, 2016

[7] Ding, Z. "Nonlinear and adaptive control systems". Vol. 84. IET, 2013.

[8] Khoshnood, A., Roshanian, J., Khaki-Sedig A. "Model reference adaptive control for a flexible launch vehicle." Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering 222.1: 49-55, 2008

[9] Rastogi, E.,Prasad L.B. "Performance Analysis of Model Reference Adaptive Control Using Lyapunov Approach for A Dynamical System." i-Manager's Journal on Electrical Engineering 9.2, 2015

[10] Andréasson, N., Evgrafov, A., Gustavsson, E., Nedelková, Z., Patriksson, M., Sou, K.C., Önnheim, M. "An Introduction to Continuous Optimization, 3rd edition", Studentlitteratur, 2013

[11] Sjö, A. "Updating techniques in recursive least-squares estimation", 1992

[12] Swanson, D.C. "Signal Processing for Intelligent Sensor Systems with MATLAB®". CRC Press, 2011

[13] Kalman, R.E., Bucy, R.S. "New results in linear filtering and prediction theory." Journal of basic engineering 83.3: 95-108, 1961

[14] Särkkä, S. "Bayesian filtering and smoothing. Vol. 3". Cambridge University Press, 2013

[15] Bobek, V. "Pmsm electrical parameters measurement." Freescale Semiconductor, 2013.

[16] Rashid, M.H. "Power electronics handbook: devices, circuits and applications". Academic press, 2010

[17] Åström, K.J., Wittenmark, B. "Adaptive control". Courier Corporation, 2013

[18] Vahidi, A, Stefanopoulou, A. Peng, H. "Recursive least squares with forgetting for online estimation of vehicle mass and road grade: theory and experiments." Vehicle System Dynamics 43.1: 31-55, 2005