A STATE OBSERVER FOR THE PERMANENT-MAGNET SYNCHRONOUS MOTOR

by

Lawrence Anthony Jones

B.S.E.E., UNIVERSITY OF NEW MEXICO
(1985)

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degrees of

ELECTRICAL ENGINEER
and
MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May, 1988

Signature of Author ....
Department of Electrical Engineering and Computer Science
May 16, 1988

Certified by .....
Jeffrey H. Lang
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Departmental Committee on Graduate Students

A State Observer for the Permanent-Magnet Synchronous Motor

by

Lawrence A. Jones

Submitted to the Department of Electrical Engineering and Computer Science on May 16, 1988 in partial fulfillment of the requirements for the degrees of Electrical Engineer and Master of Science in Electrical Engineering.

Abstract

An identity state observer for the smooth-rotor permanent-magnet synchronous motor is derived which estimates the electrical and mechanical states of the motor from measurements of its currents and voltages. The observer operates in the rotor frame and estimates direct and quadrature stator currents, rotor velocity, and rotor position. Since the rotor position is estimated, the rotor frame in which the observer operates is approximated using the latest rotor position estimate. The motor dynamics and the transformation into the estimated rotor frame are nonlinear and thus the observer and observer error dynamics are nonlinear. Simulations including measurement disturbances demonstrate that the observer converges in approximately one electrical cycle and is stable for very large perturbations. Small signal stability is analyzed from the linearized error model.

Hardware was constructed to test the operation of the observer with experimental data. The results of the experiments confirm the large signal stability of the observer and its ability to track the position of the rotor to within one mechanical degree. The steady-state accuracy of the observer is found to be less than the accuracy predicted by the simulations and the reason for this loss of accuracy is the presence of structured errors not included in the simulations. These structured errors include phase shifts in the analog channels, nonconcentricity of the rotor and stator, inaccuracy in the reference resolver, nonuniform sinusoidal winding of the stator, and non-constant stator inductance. Enhancements of the observer to account for these structured errors in order to improve the accuracy of the observer are derived.

Thesis Supervisor: Dr. Jeffrey H. Lang
Title: Associate Professor of Electrical Engineering

Contents

List of Figures

Acknowledgements

The author was supported by a National Science Foundation Fellowship during the course of this thesis. Any opinions, findings, conclusions, or recommendations expressed in this document are those of the author and do not necessarily reflect the views of the National Science Foundation. Additional support was provided by the Omron Electronics Corporation. The permanent-magnet motor drive used for experiments was donated to MIT by the Industrial Drives Division of the Kollmorgen Corporation.

Chapter 1

Introduction

Electric motors are traditionally used as actuators. However, they may also be used as sensors of the motion they actuate. This simultaneous operation as an actuator and sensor can be beneficial when additional motion sensors are too expensive, large, unreliable, or otherwise undesirable. In particular, the potential cost savings of eliminating a motion sensor can be great.

An electric motor can be successfully used as a sensor of its motion because this motion effects the voltages and currents measured at its terminals. Thus, the voltages and currents possess information concerning the motion. What is needed, then, is a means of extracting this information so as to estimate the desired motion. In this thesis, an estimator is developed which generates an estimate of the position and velocity of the motor. As shown in Figure 1.1, an application of this research is to replace the customary position sensor output to a closed-loop controller with the output of the estimator. Typical position sensors are resolvers and shaft encoders.

The particular type of motor considered in this thesis is the inverter-driven permanent-magnet synchronous motor. In order to obtain high performance, a precision position sensor is mounted on the motor shaft opposite the load coupling. The sensor provides position feedback, and, in lieu of a tachogenerator, a speed feedback signal is obtained by smoothing the derivative of the position.

There are several drawbacks to using direct position measurement. A sealed end-bell is often required to house the sensor and, for motors less than about 10 kW, this end-bell can comprise a significant proportion of the physical bulk of the motor. Multiple leads out of this end-bell are necessary to carry the sensor output signals. In the case of a resolver, a resolver-to-digital converter circuit is required to convert the resolver output into a digital position word for control purposes. The

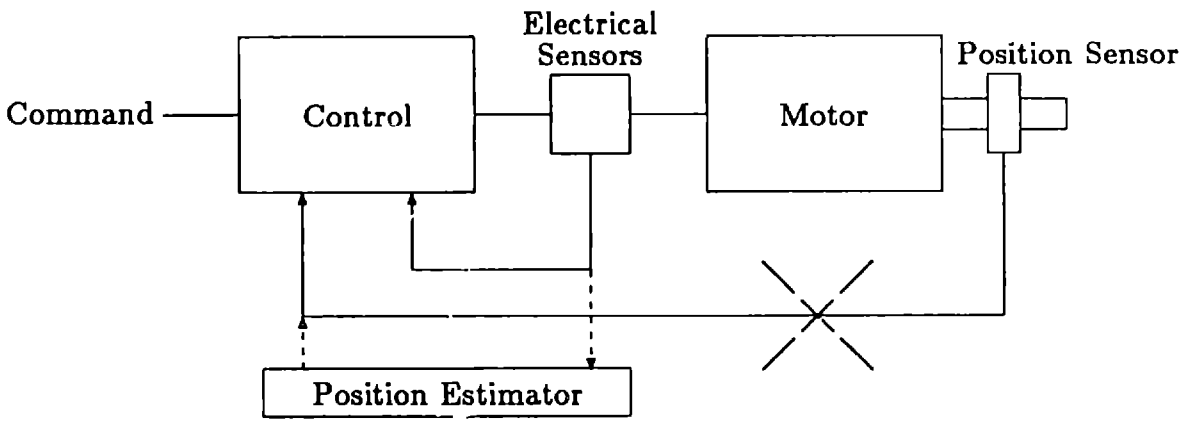


Figure 1.1: An application of a position estimator. The position estimator replaces the position sensor, eliminating the need for the sensor.

resolver and its converter together typically cost several hundred dollars.

The ultimate goal of the research presented in this thesis is to replace the direct position sensor with a position estimate derived from the stator voltages and currents. The sensor leads could be eliminated and thus system ruggedness would be improved since the only leads going to the motor would be the power leads. The position estimator considered is digital, and therefore, if a resolver is used, the resolver-to-digital circuit could also be eliminated, although an analog-to-digital converter is needed to interface the electrical sensors.

There are several drawbacks to replacing the direct position sensor with an estimator. The most significant is the reduction of control loop robustness. When a direct position sensor is used, stability of the feedback signal itself is not an issue. However, when an estimator is used, the stability and convergence rate of the estimator are important since the estimator is supplying the feedback signal. In particular, it is apparent that when the motor shaft is not moving, it is impossible to determine the shaft position if the inductance profile of the rotor is position independent. Some speed voltage caused by rotor motion is needed. If the estimator does not converge to an accurate estimate quickly, the controller will use incorrect feedback information and system performance will suffer.

Another drawback of position estimation is the need for additional electronics to implement the estimator. Sensors are needed to measure the voltages and currents.

The current sensors may already exist if the controller is controlling the current in the motor. In a digital environment, the estimator could be implemented by the same microprocessor that is used to implement the controller. This may require a more powerful microprocessor. Fortunately, microprocessors that are both inexpensive and capable of implementing both an estimator and controller are now available.

The research reported in this thesis divides into two categories. The first is the theoretical development of an algorithm to estimate the position of the motor from current and voltage measurements. The position estimator is developed from a system theory point of view with careful attention given to the structure of the underlying physical system. The development proceeds as follows. In Chapter 2, a model of the smooth-air-gap permanent-magnet synchronous motor is developed. A brief literature survey is provided in Chapter 3 as interesting examples of position estimators for various types of motors have been reported. The fundamental ideas of observer theory are presented as a viable method for estimating the position of the motor shaft. In Chapter 4, a new type of position estimator is developed and its excellent performance in simulation is demonstrated in Chapter 5. It is found to converge in approximately one electrical cycle independent of motor speed, providing the speed voltage is measurable above electrical noise. The form of this new estimator is pleasing in that it utilizes the symmetry present in the system to cope with the inherent nonlinearity of the system. Finally, the stability of the estimator is considered in Chapter 6.

The second category of research is the verification of algorithm performance using experimental data. The results of laboratory experiments conducted to verify the simulated estimator performance are presented. In Chapter 7, the experimental setup is described. In Chapter 8, the methods used to measure the system parameters are described. The results of these measurements are particularly interesting in that it was discovered that the motor used in the experiments was not electrically balanced and that the inductance profile of the rotor was not position independent. Modifications to the motor model necessary to account for the imbalance and inductance ripple are developed. The performance of the estimator driven by measured data is considered in Chapter 9. Finally, a summary and suggestions for future research are given in Chapter 10.

Information in the appendices includes a program for solving systems of differential equations, excerpts from the observer simulation program, a PC-MATLAB

macro for generating root-loci with respect to speed, details of the experimental setup, and a TMS32020 data collection routine. These appendices are referenced in the relevant chapters.

Chapter 2

State-Space Model

Rotating electric machines are well suited for the application of state-space descriptions. The mechanical motion of a rotating machine is generally simple and the electrical characteristics are easily described. The notation of state-space descriptions can be very compact, and for this reason alone one would think they would be widely used. Unfortunately, most textbooks do not present machine theory in the context of state-space descriptions and often do not utilize matrix notation; see Hancock [1], Jones [2], and Krause [3] for exceptions. The matrix exponential, so ubiquitous in linear system theory, may be used to further simplify the notation. In this review chapter, the motor models used in this thesis are derived from basic principles using matrix notation. For readers already familiar with motor models for permanent-magnet synchronous machines, the models used in this thesis are summarized at the end of this chapter in Section 2.6.

2.1 The $\alpha\beta 0$ Transformation

The matrix exponential is best applied to two-phase electrical machines. Most three-phase electrical machines have balanced wye or delta connected windings with no neutral connection and thus effectively have only two independent phases. A transformation that compacts the three-phase model to an equivalent two-phase description is not difficult to find. In fact, an infinite number of such transformations exist. However, it is desirable to utilize a particular transformation for the voltages and currents such that the representation of electrical power in the machine is preserved. As a result, the developed torques are invariant over the transformation, Hancock [1].

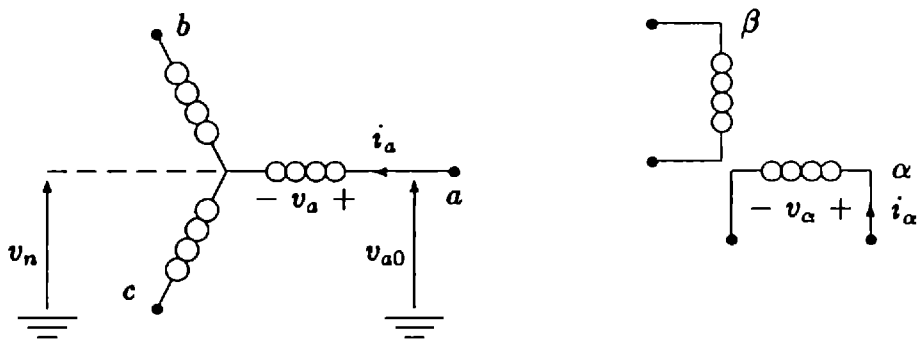


Figure 2.1: Wye connected three-phase stator and equivalent two-phase stator.

It would be convenient that the transformation be the same for both voltage and current. Consider a square transformation matrix C which transforms the variables $\tilde{\underline{i}}$ and $\tilde{\underline{v}}$ to a new set of variables $\underline{i}$ and $\underline{v}$

$$\underline{v} = \mathbf{C}\tilde{\underline{v}} \tag{2.1}$$

$$\underline{i} = \mathbf{C}\tilde{\underline{i}} \tag{2.2}$$

Power invariance is described by

$$\tilde{\underline{v}}^T\tilde{\underline{i}} = \underline{v}^T\underline{i} = \tilde{\underline{v}}^T\mathbf{C}^T\mathbf{C}\tilde{\underline{i}}. \tag{2.3}$$

Therefore,

$$\mathbf{C}^T\mathbf{C} = \mathbf{I} \tag{2.4}$$

and thus a necessary condition on the transformation matrix C is

$$\mathbf{C}^{-1} = \mathbf{C}^T. \tag{2.5}$$

Transformation matrix C must be Hermitian orthogonal. Equivalently, C must be unitary.

To determine a specific matrix C, consider the balanced three-phase system *abc* and the balanced two-phase system $\alpha\beta$ shown in Figure 2.1 where the a and α phases are aligned. The magnitude of the total air-gap magneto-motive force (mmf) for balanced currents in the three-phase system is $3NI/2$ where N is the number of turns per phase and I is the magnitude of the phase currents. In the two-phase system the magnitude of the total mmf is $\bar{N}\bar{I}$. To make the total mmf in

the balanced two-phase system be the same as the total mmf in the balanced three-phase system, the number of windings per phase in the two-phase system should be $\sqrt{3/2}$ times that in the three-phase system, and the current per phase $\sqrt{3/2}$ times that in the three-phase system, thus giving

$$\bar{N} = \sqrt{\frac{3}{2}}N \tag{2.6}$$

$$\bar{I} = \sqrt{\frac{3}{2}}I \tag{2.7}$$

so that the total two-phase mmf is $3NI/2$.

Resolving the two-phase mmf components along the three-phase axes, dividing by N, and utilizing the fact that in a wye connection $\sqrt{3}v_0 \equiv i_a + i_b + i_c = 0$ results in

$$\begin{bmatrix} i_\alpha \\ i_\beta \\ i_0 \end{bmatrix} = \begin{bmatrix} \sqrt{\frac{2}{3}} & -\sqrt{\frac{1}{6}} & -\sqrt{\frac{1}{6}} \\ 0 & \sqrt{\frac{1}{2}} & -\sqrt{\frac{1}{2}} \\ \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} \tag{2.8}$$

where i_0 is identically zero. Therefore, all of the information in the three-phase currents i_a, i_b, and i_c is contained in the two currents i_α and i_β. The current i_0 may be ignored. Note that this transformation satifies (2.5) and so it is power invariant.

Resolving the two-phase voltage vectors along the three-phase axes and utilizing the fact that for identical windings in a wye connection $\sqrt{3}v_0 \equiv v_a + v_b + v_c = 0$, also results in

$$\begin{bmatrix} v_\alpha \\ v_\beta \\ v_0 \end{bmatrix} = \begin{bmatrix} \sqrt{\frac{2}{3}} & -\sqrt{\frac{1}{6}} & -\sqrt{\frac{1}{6}} \\ 0 & \sqrt{\frac{1}{2}} & -\sqrt{\frac{1}{2}} \\ \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} \end{bmatrix} \begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} \tag{2.9}$$

where v_0 is identically zero, Leonhard [4, page 149]. The neutral terminal is normally not available and thus the phase voltages must be determined by measuring each terminal voltage with respect to some arbitrary ground. From Figure 2.1,

$$\begin{aligned} v_a &= v_{a0} + v_n \\ v_b &= v_{b0} + v_n \\ v_c &= v_{c0} + v_n \end{aligned} \tag{2.10}$$

Substituting (2.10) into (2.9) yields

$$\begin{bmatrix} v_\alpha \\ v_\beta \\ v_0 \end{bmatrix} = \begin{bmatrix} \sqrt{\frac{2}{3}} & -\sqrt{\frac{1}{6}} & -\sqrt{\frac{1}{6}} \\ 0 & \sqrt{\frac{1}{2}} & -\sqrt{\frac{1}{2}} \\ \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} \end{bmatrix} \begin{bmatrix} v_{a0} \\ v_{b0} \\ v_{c0} \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ \sqrt{3} \end{bmatrix} v_n \,. \tag{2.11}$$

Therefore, the transformation maps the three-phase voltages v_{a0}, v_{b0}, and v_{c0} into the two-phase voltages v_α and v_β, and the voltage v_0 which is proportional to the neutral voltage. Note that v_α and v_β are independent of v_n and therefore v_0 may be ignored and v_n need not be measured.

In summary, a power invariant transformation has been found that maps three-phase variables to two-phase variables and a zero sequence variable which may be ignored. It is the same for both currents and voltages and can also be shown to apply to flux linkages. The transformation matrix is

$$\mathbf{C} = \begin{bmatrix} \sqrt{\frac{2}{3}} & -\sqrt{\frac{1}{6}} & -\sqrt{\frac{1}{6}} \\ 0 & \sqrt{\frac{1}{2}} & -\sqrt{\frac{1}{2}} \\ \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} \end{bmatrix} \tag{2.12}$$

This transformation is commonly known as the $\alpha\beta 0$ transformation and, under the assumption of its use, only two-phase models will be considered henceforth.

2.2 Two-phase Model

A model of a general two-phase machine will now be developed from basic principles, Hancock [1]. Consider the set of four ideai windings shown in Figure 2.2. The $s1$ and $s2$ windings correspond with the $\alpha\beta$ windings in Figure 2.1 and are wound on the stator of the machine. The $r1$ and $r2$ windings are wound on the rotor of the machine which is free to rotate. The general forms of the terminal voltages are

$$v_{s1}(t) = R_{s1}i_{s1}(t) + \mathrm{p}[L_{s1s1}i_{s1}(t) + M_{s1s2}i_{s2}(t) + M_{s1r1}i_{r1}(t) + M_{s1r2}i_{r2}(t)] \tag{2.13}$$

$$v_{s2}(t) = R_{s2}i_{s2}(t) + \mathrm{p}[L_{s2s2}i_{s2}(t) + M_{s2s1}i_{s1}(t) + M_{s2r1}i_{r1}(t) + M_{s2r2}i_{r2}(t)] \tag{2.14}$$

$$v_{r1}(t) = R_{r1}i_{r1}(t) + \mathrm{p}[L_{r1r1}i_{r1}(t) + M_{r1s1}i_{s1}(t) + M_{r1s2}i_{s2}(t) + M_{r1r2}i_{r2}(t)] \tag{2.15}$$

$$v_{r2}(t) = R_{r2}i_{r2}(t) + \mathrm{p}[L_{r2r2}i_{r2}(t) + M_{r2s1}i_{s1}(t) + M_{r2s2}i_{s2}(t) + M_{r2r1}i_{r1}(t)] \tag{2.16}$$

where p is the time differential operator d/dt. The $s1$ winding and the $s2$ winding are magnetically orthogonal, and therefore their mutual inductance is zero. Similarly, the mutual inductance between the $r1$ and $r2$ windings is zero. Therefore

$$M_{s1s2} = M_{s2s1} = M_{r1r2} = M_{r2r1} \equiv 0\,. \tag{2.17}$$

If the stator windings are balanced, then

$$R_{s1} = R_{s2}, \qquad L_{s1s1} = L_{s2s2}\,, \tag{2.18}$$

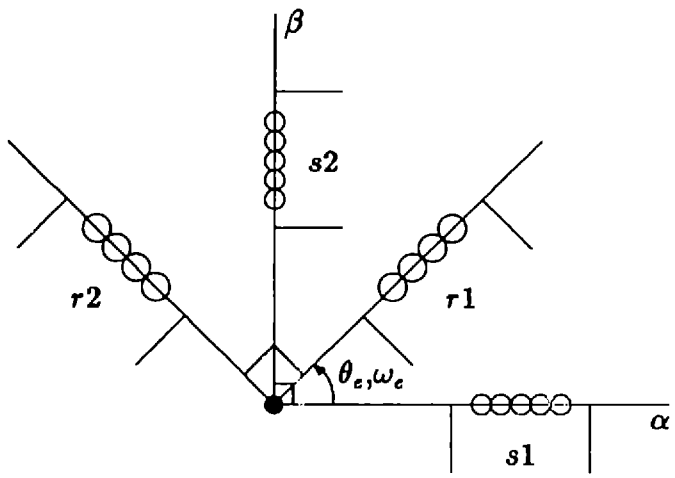


Figure 2.2: Generalized two-phase machine.

and if the rotor windings are balanced, then

$$R_{r1} = R_{r2}, \qquad L_{r1r1} = L_{r2r2}\,. \tag{2.19}$$

As the rotor turns, the mutual inductance between the rotor windings and the stator windings varies, with the maximum inductance occurring when the rotor and stator phases are aligned. Assuming that all the windings are spatially sinusoidal and that harmonics higher than the fundamental may be ignored, the mutuals between the stator and rotor may be written as

$$\begin{aligned} M_{s1r1} &= M\cos\theta_e, \qquad & M_{s1r2} &= -M\sin\theta_e, \\ M_{s2r1} &= M\sin\theta_e, \qquad & M_{s2r2} &= \;\; M\cos\theta_e \end{aligned} \tag{2.20}$$

where θ_e is a function of time.

Applying (2.17)–(2.20) to (2.13)–(2.16), and dropping all time arguments to simplify the notation results in

$$\begin{bmatrix} v_{s1} \\ v_{s2} \\ v_{r1} \\ v_{r2} \end{bmatrix} = \begin{bmatrix} R_s\mathbf{I} & \mathbf{0} \\ \mathbf{0} & R_r\mathbf{I} \end{bmatrix} \begin{bmatrix} i_{s1} \\ i_{s2} \\ i_{r1} \\ i_{r2} \end{bmatrix} \tag{2.21}$$

$$+ \begin{bmatrix} \mathrm{p}(L_s i_{s1}) & 0 & \mathrm{p}(M i_{r1}\cos\theta_e) & -\mathrm{p}(M i_{r2}\sin\theta_e) \\ 0 & \mathrm{p}(L_s i_{s2}) & \mathrm{p}(M i_{r1}\sin\theta_e) & \mathrm{p}(M i_{r2}\cos\theta_e) \\ \mathrm{p}(M i_{s1}\cos\theta_e) & \mathrm{p}(M i_{s2}\sin\theta_e) & \mathrm{p}(L_r i_{r1}) & 0 \\ -\mathrm{p}(M i_{s1}\sin\theta_e) & \mathrm{p}(M i_{s2}\cos\theta_e) & 0 & \mathrm{p}(L_r i_{r2}) \end{bmatrix}$$

Within the time derivatives in (2.21), it is apparent that if the rotor is turning then the differential equations are nonlinear in both θ_e and $\dot{\theta}_e$. Only if the shaft is locked in one position is (2.21) a linear, time-invariant system.

Equation (2.21) may be written more compactly by utilizing the matrix exponential, Brockett [5]. Define $\mathbf{J}$ as

$$\mathbf{J} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{2.22}$$

such that[1]

$$\exp(\ \mathbf{J}\theta_e) \ = \ \begin{bmatrix} \cos\theta_e & -\sin\theta_e \\ \sin\theta_e & \cos\theta_e \end{bmatrix} \tag{2.23}$$

$$\exp(-\mathbf{J}\theta_e) \ = \ \begin{bmatrix} \cos\theta_e & \sin\theta_e \\ -\sin\theta_e & \cos\theta_e \end{bmatrix}. \tag{2.24}$$

Utilizing the matrix exponential, (2.21) becomes

$$\begin{bmatrix} \underline{v}_s \\ \underline{v}_r \end{bmatrix} = \begin{bmatrix} R_s\mathbf{I} & 0 \\ 0 & R_r\mathbf{I} \end{bmatrix} \begin{bmatrix} \underline{i}_s \\ \underline{i}_r \end{bmatrix} + \mathrm{p}\left(\begin{bmatrix} L_s\mathbf{I} & M\exp(\mathbf{J}\theta_e) \\ M\exp(-\mathbf{J}\theta_e) & L_r\mathbf{I} \end{bmatrix} \begin{bmatrix} \underline{i}_s \\ \underline{i}_r \end{bmatrix} \right). \tag{2.25}$$

It is important to reconsider the assumptions that have been made in deriving this model:

- there is no magnetic coupling between the two stator windings,
- there is no magnetic coupling between the two rotor windings,
- the windings on both the rotor and the stator are assumed to be balanced, and
- the mutual inductances between the rotor windings and the stator windings are assumed to vary sinusoidally with rotor position.

Allowing for these assumptions, (2.25) is a valid model for any two-phase machine. In Section 2.4 it will be shown how this model may be specialized to a permanent-magnet, synchronous machine.

[1] A remarkable isomorphism exists between matrices of the form $a\mathbf{I} + b\mathbf{J}$ and complex numbers $a + jb$. Under certain conditions, matrices of the form $a\mathbf{I} + b\mathbf{J}$ may be treated *as if* they were complex numbers. Utilizing this isomorphism, (2.21) may be written even more compactly than with matrix exponentials, Novotny [6].

2.3 Torque Generation

If the shaft of the machine is stationary, (2.25) reduces to a set of linear, time-invariant differential equations. These LTI equations simply describe a special type of electric machine, namely a (possibly very inefficient) transformer. The electric machine of interest here is one that is designed to convert electric power into motion, or vice-versa. If the shaft is allowed to rotate, then it is necessary to include a model of the shaft dynamics in the overall model of the machine. These mechanical dynamics will be derived in this section.

Virtually all electric machines utilize magnetic fields to convert energy. Transformers convert energy from one electrical form to another electrical form (typically a voltage step up or down). Electric machines use magnetic fields to convert energy between electrical form and mechanical form. In motors, electrical energy stored in the windings is converted to mechanical energy which appears as torque during rotation of the motor shaft. In generators, the mechanical energy of the rotating shaft is converted to electrical energy in the windings. The torque necessary to turn the shaft in a motor, and the resisting torque the rotor in a generator experiences, is a result of the coupling of the magnetic fields produced by the windings on the stator with magnetic fields produced by the windings on the rotor, Woodson and Melcher [7]. How the fields couple is determined by the inductance matrix in (2.25) and it is what distinguishes the basic types of electric machines.

In a general machine such as that in Figure 2.2, the windings on the stator and on the rotor may be excited independently. The currents in the rotor are supplied through slip-rings mounted on the shaft. The mmf vector set up by these currents will always seek to align itself with the mmf vector set up by the currents in the windings of the stator in order to minimize potential energy. The difference between the mmf vectors produces a torque on the rotor. For the case of balanced sinusoidal excitation of the two-phase stator at a frequency of ω_s, and the balanced sinusoidal excitation of the two-phase rotor at frequency of ω_r, the two mmfs will only be traveling at the same angular speed in the air-gap of the machine if the speed of the turning rotor makes up the difference between the stator excitation frequency and rotor excitation frequency. Thus,

$$\omega_s - \omega_r = \omega\,. \tag{2.26}$$

This is the condition for time-average torque production. If ω_r and ω are constant,

then (2.26) is satisfied for only a single frequency ω_s. A machine in which ω_r is always constant is called a synchronous machine. One such example is the permanent-magnet synchronous motor discussed later in this thesis. A machine in which ω_r varies continuously in order to satisfy (2.26) is called an asynchronous machine.

Induction machines are asynchronous machines. In induction machines, there are no external connections to the rotor and excitation is provided by currents induced in the rotor by the stator mmf. Since this excitation is a result of conductors in the rotor crossing stator produced magnetic field lines, then if the rotor is rotating at the same speed as the stator mmf, no field lines will be crossed and the rotor will not be excited. Therefore, the rotor in induction machines must rotate at any speed *other* than that of the stator mmf. This difference in speeds is commonly referred to as the slip frequency.

As might be expected, hybrids exist such that classifying a machine as strictly synchronous or asynchronous is sometimes not possible. For example, machines that have slip-rings and normally run at synchronous speed often have special conductors embedded in the rotor such that above or below synchronous speed, induction takes place, thereby allowing the continued generation of torque. Therefore, in classifying machines it is more useful to use the inductance matrix in (2.25).

The energy delivered to or from the shaft of an electric machine is a result of the coupling of the electrical dynamics of the stator and rotor and therefore it is possible to derive this energy from (2.25) using conservation arguments. The derivation presented here is similar to that given by Woodson and Melcher [7].

The instantaneous power in the machine is

$$\begin{bmatrix} \underline{i}_s^T & \underline{i}_r^T \end{bmatrix} \begin{bmatrix} \underline{v}_s \\ \underline{v}_r \end{bmatrix} = \tag{2.27}$$

$$R \begin{bmatrix} \underline{i}_s^T & \underline{i}_r^T \end{bmatrix} \begin{bmatrix} \underline{i}_s \\ \underline{i}_r \end{bmatrix} + \begin{bmatrix} \underline{i}_s^T & \underline{i}_r^T \end{bmatrix} \mathrm{p} \left(\begin{bmatrix} L_s \mathbf{I} & M \exp(\mathbf{J}\theta_e) \\ M \exp(-\mathbf{J}\theta_e) & L_r \mathbf{I} \end{bmatrix} \begin{bmatrix} \underline{i}_s \\ \underline{i}_r \end{bmatrix} \right).$$

The first term on the right is the resistive power losses while the second term on the right is the sum of the power going into the magnetic fields of the windings and the power going out the shaft of the machine, that is,

$$\underline{i}^T \underline{v} = R \underline{i}^T \underline{i} + \frac{dW}{dt} + T_{em} \frac{d\theta}{dt} \tag{2.28}$$

where W is the energy stored in the windings and T_{em} is the electromagnetic torque

exerted on the shaft. Substituting (2.25) gives

$$\frac{dW}{dt} = \underline{i}^T \frac{d(\mathbf{L}\underline{i})}{dt} - T_{em}\frac{d\theta}{dt}. \tag{2.29}$$

Electrical energy is most conveniently added to the windings by changing $\underline{i}$ in time, not $\mathbf{L}\underline{i}$. The derivative relationship

$$\frac{d(\underline{i}^T\mathbf{L}\underline{i})}{dt} = \underline{i}^T\frac{d(\mathbf{L}\underline{i})}{dt} + \underline{i}^T\mathbf{L}^T\frac{d\underline{i}}{dt} \tag{2.30}$$

allows (2.29) to be re-expressed as

$$\frac{dW'}{dt} = \underline{i}^T\mathbf{L}^T\frac{d\underline{i}}{dt} + T_{em}\frac{d\theta}{dt} \tag{2.31}$$

where the coenergy W' is

$$W' = \underline{i}^T\mathbf{L}\underline{i} - W. \tag{2.32}$$

Multiplying through by dt gives an incremental expression for the coenergy

$$dW' = \underline{i}^T\mathbf{L}^T d\underline{i} + T_{em}d\theta. \tag{2.33}$$

In (2.32), it is apparent that the coenergy is a function of $\underline{i}$ and θ (since $\mathbf{L}$ is a function of θ) and so by the chain rule

$$dW' = \left.\frac{\partial W'}{\partial \underline{i}}\right|_{\theta} d\underline{i} + \left.\frac{\partial W'}{\partial \theta}\right|_{\underline{i}} d\theta. \tag{2.34}$$

Comparing (2.33) with (2.34), the electromagnetic torque experienced by the shaft is

$$T_{em} = \left.\frac{\partial W'}{\partial \theta}\right|_{\underline{i}}. \tag{2.35}$$

To find W', (2.34) is integrated. Under the assumptions that

$$T_{em}(\underline{i} = 0, \theta) = 0 \tag{2.36}$$

$$W'(\underline{i} = 0, \theta = 0) = 0 \tag{2.37}$$

the result is

$$W' = \frac{1}{2}\underline{i}^T\mathbf{L}\underline{i}. \tag{2.38}$$

Substituting W' into (2.35), the torque is thus

$$T_{em} = \frac{1}{2}\underline{i}^T\frac{\partial \mathbf{L}}{\partial \theta}\underline{i}. \tag{2.39}$$

With ideal windings there is no magnetic saturation and so $\mathbf{L}$ is a function of θ only and the partial may be replaced by a total derivative. Finally, for the inductance matrix in (2.25)

$$T_{em} = \frac{1}{2}\begin{bmatrix} \underline{i}_s^T & \underline{i}_r^T \end{bmatrix} \frac{d}{d\theta}\begin{bmatrix} L_s\mathbf{I} & M\exp(\mathbf{J}\theta_e) \\ M\exp(-\mathbf{J}\theta_e) & L_r\mathbf{I} \end{bmatrix}\begin{bmatrix} \underline{i}_s \\ \underline{i}_r \end{bmatrix}. \tag{2.40}$$

The electromagnetic torque is that which is necessary to counteract a load torque τ on the shaft, overcome the viscous friction torque $Bd\theta/dt$, overcome the directionless, constant, coulomb friction torque C, and accelerate the rotor which has inertia H. Summing these torques yields

$$T_{em} = H\frac{d^2\theta}{dt^2} + B\frac{d\theta}{dt} + C\frac{\omega}{|\omega|} + \tau. \tag{2.41}$$

Equation (2.41) is a model of the rotor dynamics where T_{em} is computed using (2.40). Note that it is nonlinear due to the absolute value in the coulomb friction term. By including this model with the model of the electrical dynamics derived in Section 2.2, the dynamics of a general electric machine, save the assumptions itemized in Section 2.2, are modeled completely.

2.4 The Permanent-Magnet Motor

The model (2.25) will now be specialized for a particular type of motor, namely the smooth-rotor surface-mounted permanent-magnet motor. The geometry is shown in Figure 2.3. In this type of motor, permanent magnets are used to provide the excitation on the rotor. Since the flux that they provide is fixed, they resemble windings with constant current sources, and therefore, w_r in (2.26) is zero. The material used to make the permanent magnets usually has high electrical resistivity, thereby precluding the production of torque via eddy currents. The permeability of the magnets is similar to air and so the stator sees a rotor surface of uniform permeability.

To specialize (2.25) to a machine of the form shown in Figure 2.3 it is necessary only to modify the inductance matrix. Let N equal the number of magnet pole pairs on the rotor. Each pair of poles is identical and therefore, if the shaft rotates $2\pi/N$ radians, the inductance matrix is exactly the same as before the rotation. In other words, for each 2π rotation of the shaft, the motor goes through N electrical cycles. The inductance matrix is periodic with an *electrical frequency*, ω_e, which is N times

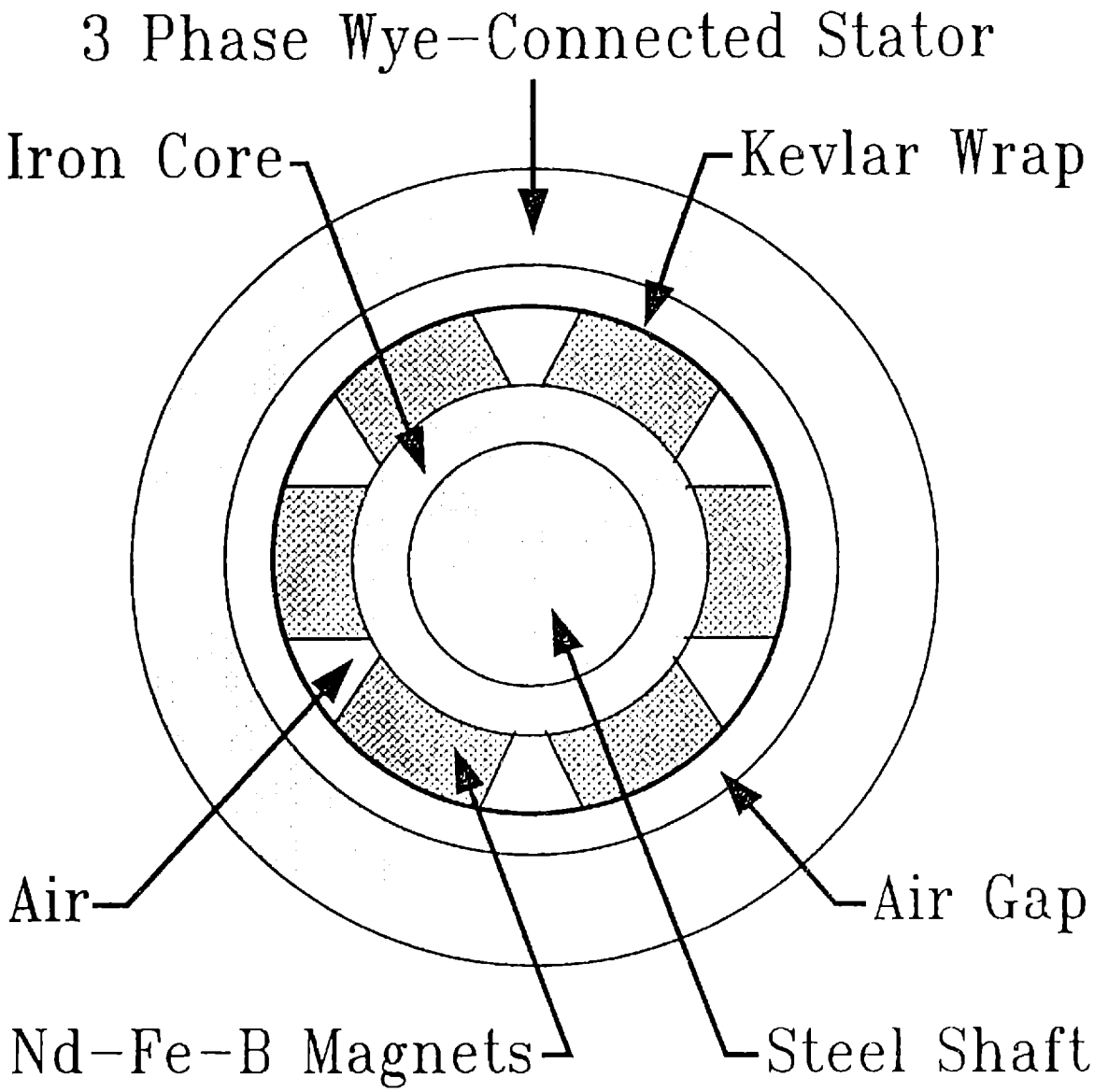


Figure 2.3: Geometry of a smooth-rotor surface-mounted permanent-magnet motor.

the *mechanical frequency* of the rotor. An *electrical angle*, θ_e, may be similarly defined as N times the *mechanical angle*. Define the units electrical radians (erad) such that

$$\theta_e = N\theta \text{ erad} \tag{2.42}$$

$$\omega_e = N\omega \text{ erad/s}. \tag{2.43}$$

Each north pole and it's adjacent south pole may be modeled as a single winding within an electrical cycle. In terms of Figure 2.2, current is present only in winding $r1$ on the rotor, and it is constant; the current in winding $r2$ is set to zero.

The rotor with surface mounted permanent magnets, shown in Figure 2.3, presents a surface of uniform permeability to the stator and therefore, as it rotates, the self-inductance of the stator windings is constant. Since the permeability of the permanent-magnet material is low, these self-inductances will be relatively small. In a non-ideal motor there may be some saturation in the metal of the motor, in which case the stator self-inductance, L_s, and coefficient of the mutual inductances, M, will not be constant. Permanent-magnet machines of this type are not generally operated with the bulk of the iron saturated and therefore the assumption of no saturation is valid.[2]

Normal operating conditions do not subject the permanent magnets to severe demagnetization and the material used to make the magnets is usually able to resist demagnetization during abnormal operating conditions. When this fact is combined with the fact that little eddy currents are generated in the rotor, it may be concluded that the model of the permanent magnets on the rotor, namely rotor windings with constant current excitation, is indeed valid. With the rotor current constant, there are no rotor electrical dynamics and therefore they may be dropped from the model (2.25). To simplify notation, the subscript s on the remaining variables and constants is dropped. This machine may be classified as synchronous, Leonhard [8].

To summarize, the inductance matrix in (2.25) is modified according to the following observations.

- The stator self-inductance L is constant since the rotor presents a surface of uniform permeability.

[2] Variable reluctance motors are an example of a type of machine often designed to operate with the iron heavily saturated.

- The inductance matrix is periodic with an electrical frequency $\omega_e = N\omega$.
- The magnet pole pairs on the rotor are modeled by single windings with constant current excitation i_r.
- Ideal windings have been assumed and thus saturation effects have been ignored. Including them would make L and M dependent on t and not allow them to be pulled out of the time derivative in (2.25).
- The rotor excitation is constant as the rotor turns and therefore there are no rotor electrical dynamics.

Applying the assumptions listed above, equation (2.25) reduces to

$$\underline{v} = R\underline{i} + L\frac{d\underline{i}}{dt} + MJN\omega \exp(JN\theta) \begin{bmatrix} 1 \\ 0 \end{bmatrix} i_{r1}\,, \tag{2.44}$$

and (2.41) to

$$\underline{i}^T MJN \exp(JN\theta) \begin{bmatrix} 1 \\ 0 \end{bmatrix} i_{r1} = H\frac{d^2\theta}{dt^2} + B\frac{d\theta}{dt} + C\frac{\omega}{|\omega|} + \tau\,. \tag{2.45}$$

Since i_{r1} is constant and scalar, define $K_m = Mi_{r1}$. Expressed in state-space form the model, (2.44) and (2.45), of a smooth rotor, permanent-magnet machine is

$$\frac{d\underline{i}}{dt} = -\frac{R}{L}\underline{i} - \frac{K}{L}N\omega \exp(JN\theta) \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \frac{1}{L}\underline{v} \tag{2.46}$$

$$\frac{d\omega}{dt} = -\frac{B}{H}\omega + \frac{KN}{H}\underline{i}^T \exp(JN\theta) \begin{bmatrix} 0 \\ 1 \end{bmatrix} - \frac{C}{H}\frac{\omega}{|\omega|} - \frac{1}{H}\tau \tag{2.47}$$

$$\frac{d\theta}{dt} = \omega\,. \tag{2.48}$$

The electrical dynamics are the current dynamics driven by the input voltage. Since currents may be directly measured in a real machine, it is convenient to define an output equation of precisely the current states

$$\underline{y} = \underline{i}\,. \tag{2.49}$$

The system (2.46)–(2.48) is a suitable state-space description of the permanent-magnet machine being considered. Note that the system is nonlinear in shaft position θ and shaft velocity ω. The nonlinear system is of the form

$$\frac{d\underline{x}}{dt} = \underline{f}(\underline{x}, \underline{u}) \tag{2.50}$$

Removing the θ nonlinearity by requiring θ to be constant is unacceptable since a motor is designed to convert electrical energy into motion. Removing the ω nonlinearity by requiring ω be constant would be an acceptable restriction. However, the model would still be nonlinear in θ.

The motion of most electric machines is single-axis rotation. Rotation about a single axis has a great deal of symmetry and this symmetry is apparent in (2.25) and (2.46)–(2.48), but not in the more general system equation (2.50). An important conclusion is that care must be taken in determining how general the system description may be without losing sight of valuable characteristics of the underlying system. The form of estimators and controllers derived by studying (2.25) and (2.40), which are a general description of rotating electric machines, may be very different from those derived by studying (2.50).

2.5 The *dq* Transformation

In this section a well-known transformation that arises naturally from (2.25) will be presented. The transfomation removes the θ_e nonlinearity without requiring that θ_e be constant. Once the transformation has been used to remove the θ_e nonlinearity, applying the restriction that ω be constant yields a linear system.

An orthogonal transformation corresponds to a rotation and/or reflection of one reference frame into another reference frame. Since the motion of the electric machines being considered is restricted to rotation, an orthogonal transformation might exist. This transformation could rotate the electrical variables in the frame of the stator into the rotating frame of the rotor.

Consider, for a moment, the form of a transformation that represents a single-axis rotation. Figure 2.4 shows axes $a1$-$r2$ rotating concentrically with respect to the fixed axes $\alpha\beta$. The components of I, projected on the $a1$-$a2$ axes, are i_{a1} and i_{a2}. The projections of these components on the $\alpha\beta$ axes are

$$i_\alpha = i_{a1}\cos\theta_e - i_{a2}\sin\theta_e \tag{2.51}$$

$$i_\beta = i_{a1}\sin\theta_e + i_{a2}\cos\theta_e\,. \tag{2.52}$$

In vector matrix form,

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \exp(\mathbf{J}\theta_e)\begin{bmatrix} i_{r1} \\ i_{r2} \end{bmatrix}. \tag{2.53}$$

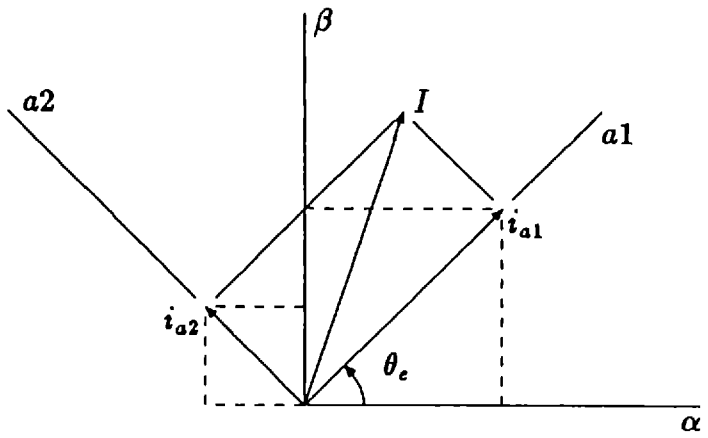


Figure 2.4: Rotation of the coordinate axes from $\alpha\beta$ to $a1$-$a2$.

Therefore, the matrix exponentials in (2.25) are precisely the transformations that rotate the variables onto a different reference frame.

Note that two reference frames are being used in (2.25). The matrix exponential in the stator equation refers the rotor variables from a frame rotating in the positive direction with respect to the stationary stator frame. From the rotor's point of view, the stator is rotating in the negative direction and, accordingly, the matrix exponential in the rotor equation refers the stator variables from a frame rotating in the negative direction with respect to a stationary rotor frame.

It might be advantageous to use only one reference frame. Referring all variables to either the fixed stator frame or the moving rotor frame would be acceptable. However, consider the more general case of referring both the stator and rotor variables to an arbitrary frame moving at a arbitrary angular speed. Figure 2.5 shows an arbitrary reference frame dq which is at an angle δ and rotating with an angular speed ω_δ. The d axis is called the *direct* axis and the q axis is called the *quadrature* axis. The rotor frame $r1r2$ is at an angle θ_e and rotating with an angular speed ω_e. To rotate the stator variables onto the dq frame, denoted with an overbar, define

$$\underline{\bar{i}}_s = \exp(-\mathrm{J}\delta)\underline{i}_s \tag{2.54}$$

$$\underline{\bar{v}}_s = \exp(-\mathrm{J}\delta)\underline{v}_s, \tag{2.55}$$

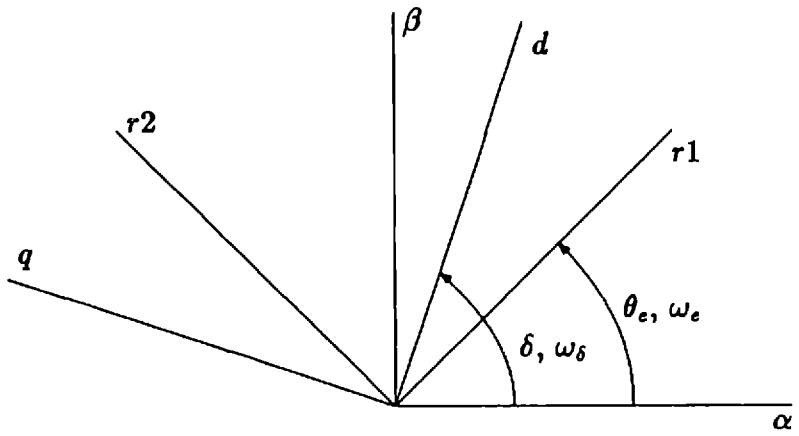


Figure 2.5: Rotation of axes $\alpha\beta$ and $r1r2$ onto arbitrary frame dq.

and to rotate the rotor variables onto the dq frame define $\gamma = \delta - \theta_e$ and

$$\underline{\bar{i}}_r = \exp(-\mathbf{J}\gamma)\underline{i}_r \tag{2.56}$$

$$\underline{\bar{v}}_r = \exp(-\mathbf{J}\gamma)\underline{v}_r\,. \tag{2.57}$$

Applying (2.54)–(2.57) to (2.25) yields

$$\begin{bmatrix} \underline{\bar{v}}_s \\ \underline{\bar{v}}_r \end{bmatrix} = \begin{bmatrix} R_s\mathbf{I} & 0 \\ 0 & R_r\mathbf{I} \end{bmatrix} \begin{bmatrix} \underline{\bar{i}}_s \\ \underline{\bar{i}}_r \end{bmatrix} + \mathrm{p}\left(\begin{bmatrix} L_s\mathbf{I} & M\mathbf{I} \\ M\mathbf{I} & L_r\mathbf{I} \end{bmatrix} \begin{bmatrix} \underline{\bar{i}}_s \\ \underline{\bar{i}}_r \end{bmatrix} \right) + \begin{bmatrix} L_s\mathbf{J}\omega_\delta & M\mathbf{J}\omega_\delta \\ M\mathbf{J}(\omega_\delta - \omega_e) & L_r\mathbf{J}(\omega_\delta - \omega_e) \end{bmatrix} \begin{bmatrix} \underline{\bar{i}}_s \\ \underline{\bar{i}}_r \end{bmatrix}. \tag{2.58}$$

If the rotor is rotating at a constant speed, and the arbitrary reference frame is rotating at some constant speed then (2.58) is a linear, time-varying system. The transformation has removed the nonlinearity. However, unless the system is periodically time-varying, (2.58) is still difficult to solve without resorting to numerical methods.

Equation (2.58) represents the general transformation of the generalized machine (2.25). This transformation is commonly known as the dq transformation. Choice of an appropriate dq reference frame depends on the machine being considered, Krause [3]. An appropriate dq reference frame for the motor considered in Section 2.4 is to fix it on the rotor such that $\omega_\delta = \omega_e$. Since the mmf vector on the rotor is independent of the stator and fixed by the permanent magnets, the rotor mmf

vector moves exactly with the rotor. If the rotor flux vector was not fixed on the rotor, as in an induction machine, then it would be more appropriate to make the arbitrary reference frame correspond to the rotor flux vector. When the reference frame is fixed to the rotor, the transformations (2.54) and (2.55) reduce to

$$\bar{\underline{i}} = \exp(-\mathbf{J}\theta_e)\underline{i} \tag{2.59}$$

$$\bar{\underline{v}} = \exp(-\mathbf{J}\theta_e)\underline{v}\,. \tag{2.60}$$

When these transformations are applied to the stator-frame model of the smooth-rotor, permanent-magnet motor (2.46)–(2.48), then (2.58) reduces to

$$\frac{d\bar{\underline{i}}}{dt} = -\left(\frac{R}{L}\mathbf{I} + \mathbf{J}N\omega\right)\bar{\underline{i}} - \frac{K}{L}N\omega\begin{bmatrix} 0 \\ 1 \end{bmatrix} + \frac{1}{L}\bar{\underline{v}}\,. \tag{2.61}$$

The transformation may also be applied to (2.47) to give

$$\frac{d\omega}{dt} = -\frac{B}{H}\omega + \bar{\underline{i}}^T\frac{KN}{H}\begin{bmatrix} 0 \\ 1 \end{bmatrix} - \frac{C}{H}\frac{\omega}{|\omega|} - \frac{1}{H}\tau\,. \tag{2.62}$$

If ω is constant then (2.61) and (2.62) are a linear, time-invariant system and thus easily solvable. Therefore, a transformation has been found that, under the restriction that the rotor is turning at constant speed, produces an LTI system.

The transformation where the reference frame is fixed on the rotor was first introduced in 1919 by Blondel [9]. During the following decade, Doherty [10] and Park [11] extended Blondel's work and the transformation became widely known as the Park transformation or the *dq* transformation. It revolutionized electric machine analysis since it transformed the intractable nonlinear problem into a linear problem. Transformations to other reference frames in order to convert the equations of other types of machines have appeared in the literature. In Krause [12], it is shown that all of these transformations can be derived from an arbitrary reference frame like that presented in this section. Transformations for some common machines have still not been found, e.g. for the variable reluctance machine. See Liu, et al. [13] for recent research results on the conditions necessary for a transformation to exist.

2.6 Summary

Due to the isolated neutral wye connection of the windings on the stator and rotor of electric machines, only two of the three phases are independent and therefore a

transformation may be found that reduces the three-phase model to a two-phase model. A power-invariant transform that accomplishes this reduction is

$$\begin{bmatrix} S_\alpha \\ S_\beta \\ S_0 \end{bmatrix} = \begin{bmatrix} \sqrt{\frac{2}{3}} & -\sqrt{\frac{1}{6}} & -\sqrt{\frac{1}{6}} \\ 0 & \sqrt{\frac{1}{2}} & -\sqrt{\frac{1}{2}} \\ \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} & \sqrt{\frac{1}{3}} \end{bmatrix} \begin{bmatrix} S_a \\ S_b \\ S_c \end{bmatrix} \tag{2.63}$$

where S is either a current or voltage variable. Using this transformation, only two-phase models need to be considered.

A two-phase state-space model for a smooth rotor, permanent-magnet motor is

$$\frac{d\underline{i}}{dt} = -\frac{R}{L}\underline{i} - \frac{K}{L}N\omega \exp(\mathbf{J}N\theta)\begin{bmatrix} 0 \\ 1 \end{bmatrix} + \frac{1}{L}\underline{v} \tag{2.64}$$

$$\frac{d\omega}{dt} = -\frac{B}{H}\omega + \frac{KN}{H}\underline{i}^T \exp(\mathbf{J}N\theta)\begin{bmatrix} 0 \\ 1 \end{bmatrix} - \frac{C}{H}\frac{\omega}{|\omega|} - \frac{1}{H}\tau \tag{2.65}$$

$$\frac{d\theta}{dt} = \omega . \tag{2.66}$$

The current vector $\underline{i}$ consists of two components in quadrature, namely i_α and i_β. The voltage vector $\underline{v}$ consists of two components in quadrature, namely v_α and v_β. The stator phase inductance and resistance are L and R, respectively. The magnet constant is K, and the number of magnet pole pairs on the rotor is N. Therefore, there are N electrical cycles for every mechanical cycle of the motor shaft. The rotor speed is ω and the rotor position is θ. The matrix $\mathbf{J}$ is given by

$$\mathbf{J} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} . \tag{2.67}$$

Finally, B is the viscous damping coefficient, H is the rotor inertia, τ is the load torque, and C is the coulomb friction coefficient. Note that coulomb friction always acts to retard rotor motion, independent of the direction of the rotor motion.

The voltages may be thought of as inputs to the system which drive the current dynamics while the currents may be thought of as measured outputs of the system. The state-space is of the form

$$\frac{d\underline{x}}{dt} = A\underline{x} + f(\underline{x}) + B\underline{u} \tag{2.68}$$

$$\underline{y} = C\underline{x} . \tag{2.69}$$

The system is nonlinear in both θ and ω.

The dq transformation

$$\bar{\underline{i}} = \exp(-\mathbf{J}\theta_e)\underline{i} \tag{2.70}$$

$$\bar{\underline{v}} = \exp(-\mathbf{J}\theta_e)\underline{v} \tag{2.71}$$

may be used to remove the nonlinearity in θ. The transformation projects the electrical variables onto a reference that rotates with the rotor. It is a special case of the dq transformation where the reference frame rotates at an arbitrary speed. Using the Park transformation, the system may be expressed such that it is nonlinear only in products of ω and $\underline{i}$.

$$\frac{d\bar{\underline{i}}}{dt} = -\left(\frac{R}{L}\mathbf{I} + \mathbf{J}N\omega\right)\bar{\underline{i}} - \frac{K}{L}N\omega\begin{bmatrix}0\\1\end{bmatrix} + \frac{1}{L}\bar{\underline{v}} \tag{2.72}$$

$$\frac{d\omega}{dt} = -\frac{B}{H}\omega + \bar{\underline{i}}^T\frac{KN}{H}\begin{bmatrix}0\\1\end{bmatrix} - \frac{C}{H}\frac{\omega}{|\omega|} - \frac{1}{H}\tau \tag{2.73}$$

$$\frac{d\theta}{dt} = \omega \tag{2.74}$$

If ω is assumed to be constant then the transformed system is linear, time-invariant.

Chapter 3

Literature Survey

The estimation of motor motion using measurements of electrical variables is not a new idea. A wide variety of methods have already been investigated in the literature. At one extreme are waveform detection methods that are commonly associated with wound-rotor and permanent-magnet synchronous motors, and variable-reluctance motors. These methods attempt to identify specific events such as peaks or zero crossings in the electrical waveforms that are the result of either the speed voltage of the rotating rotor, or changes in the mutual inductance between the stator and the rotor as the rotor moves. They have been successfully demonstrated for commutation needs, and may be implemented using inexpensive electronics, which is advantageous from a manufacturing viewpoint. However, they are not as accurate as possible because they do not utilize all the information present in the waveforms. For example, the fact that the speed voltage is always present while the rotor is moving means that speed voltage information which may reveal the position of the rotor is always present in the waveforms.

One way to extract all the information in the waveforms is to model the dynamics of the motor, drive these dynamics with the same input as is used to drive the real motor, and somehow ensure that errors between the modeled motor and real motor are minimized. If this can be done, then the states of the modeled motor will effectively summarize all the information in the waveforms up to the present time, and the model will accurately reflect the behavior of the real motor. This is the basis of a state observer. In addition, in a state observer, an output is defined as a combination of the states and this output is compared with the equivalent measured output of the real motor. Any error between these two signals is then used to correct the state trajectory of the observer. This corrective feedback may be used to design

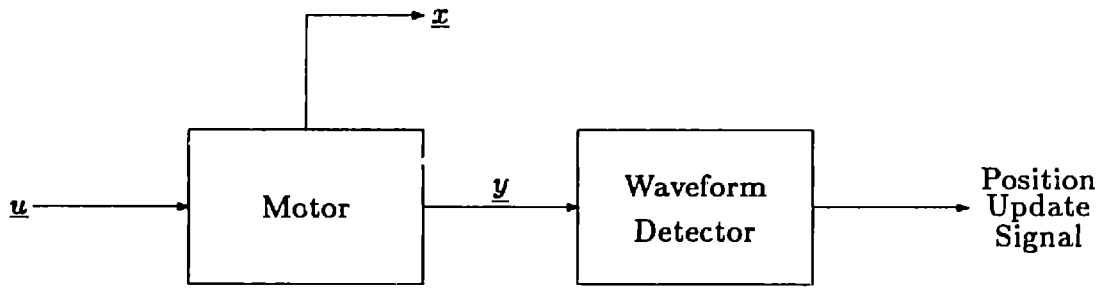


Figure 3.1: Estimation using waveform detection. The input to the motor is $\underline{u}$ and the output is $\underline{y}$. The dynamic states of the motor are $\underline{x}$. The waveform detector looks for specific events in the measured output $\underline{y}$ to produce a discrete position update signal.

an observer that converges quickly to zero error, or at least to ensure that observer errors are minimized even in the presence of noise and modeling errors.

For a smooth-rotor permanent-magnet synchronous motor, a state observer may be used to estimate the electrical and mechanical states. The estimated states are the two stator currents, the rotor velocity, and the rotor position. Thus the states of the observer are precisely the states desired for typical implementations of field-oriented, or vector, control, Leonhard [4].

3.1 Waveform Detection

The term *waveform detection* is used to describe those techniques that attempt to extract the shaft position from the electrical waveforms of the motor by identifying characteristics in the waveforms that are a result of the position and/or motion of the rotor. These techniques do not attempt to provide a continuous estimate of the shaft position but rather look for specific events in the measured waveforms and produce a discrete position update signal as shown in Figure 3.1.

3.1.1 Examples of Waveform Detection

One of the earliest accounts of waveform detection (for which a U.S. patent was granted) is by Frus and Kuo [14]. This is a good, introductory paper on using waveform detection to control a single stack, variable-reluctance motor. The poten-

tial advantages of waveform detection are thoroughly explained. A single step of the motor is detected by monitoring the currents in the independent phases. These currents reflect the changing stator self inductances that depend on the position of the shaft. During point-to-point operation, the controller uses the number of steps taken to determine when to accelerate and decelerate the motor according to a preset acceleration profile. The details of how a step is determined from the current waveforms, which are missing in Frus and Kuo [14], were presented a year later by Kuo [15]. To detect position, the current in a phase that has just been turned off is monitored. As the current decays through the flyback circuit, local maxima will be observed as the the stator self inductance varies. The detection of one of these maxima indicates a successful step. The development of this method is continued in Lin and Kuo [16], and Kuo, et al. [17].

Pittet and Jufer [18] have reported an application of waveform detection to a single-phase permanent-magnet stepping motor. The decision of when to switch the polarity of the voltage applied to the single phase winding is determined by monitoring the difference between the current in a motor with a blocked rotor and the actual current in the motor. Hair [19] has reported the application of waveform detection to a two-phase two-pole permanent-magnet stepper motor. The technique takes advantage of the fact that the stator phase windings are independent. While one set of windings is energizing, the other set of windings is open-circuited so that the speed voltage can be monitored. Since the phase windings are in quadrature, the speed voltage is a simple sinusoid at the mechanical frequency of the motor. Therefore, a simple zero-crossing detect circuit is used to determine if the motor stepped as commanded.

An active form of waveform detection has been reported by Harris [20]. Using a four-pole three-phase variable-reluctance motor with independent stator phase windings, one phase is energized at a time and one of the off phases is actively probed to determine the self inductance of the stator. The inductance is determined by pulsing a voltage on one of the off phases and monitoring the resulting current ramps. As the minimum inductance point is approached, the current will exceed a threshold, indicating to the controller that it is time to switch the phases. An important advantage of this technique is that the control bandwidth is much higher than with a passive waveform detection technique. By varying the threshold, the controller can precisely set the switching angle which results in faster motor response. With a passive technique, the firing angle is only appoximate since the

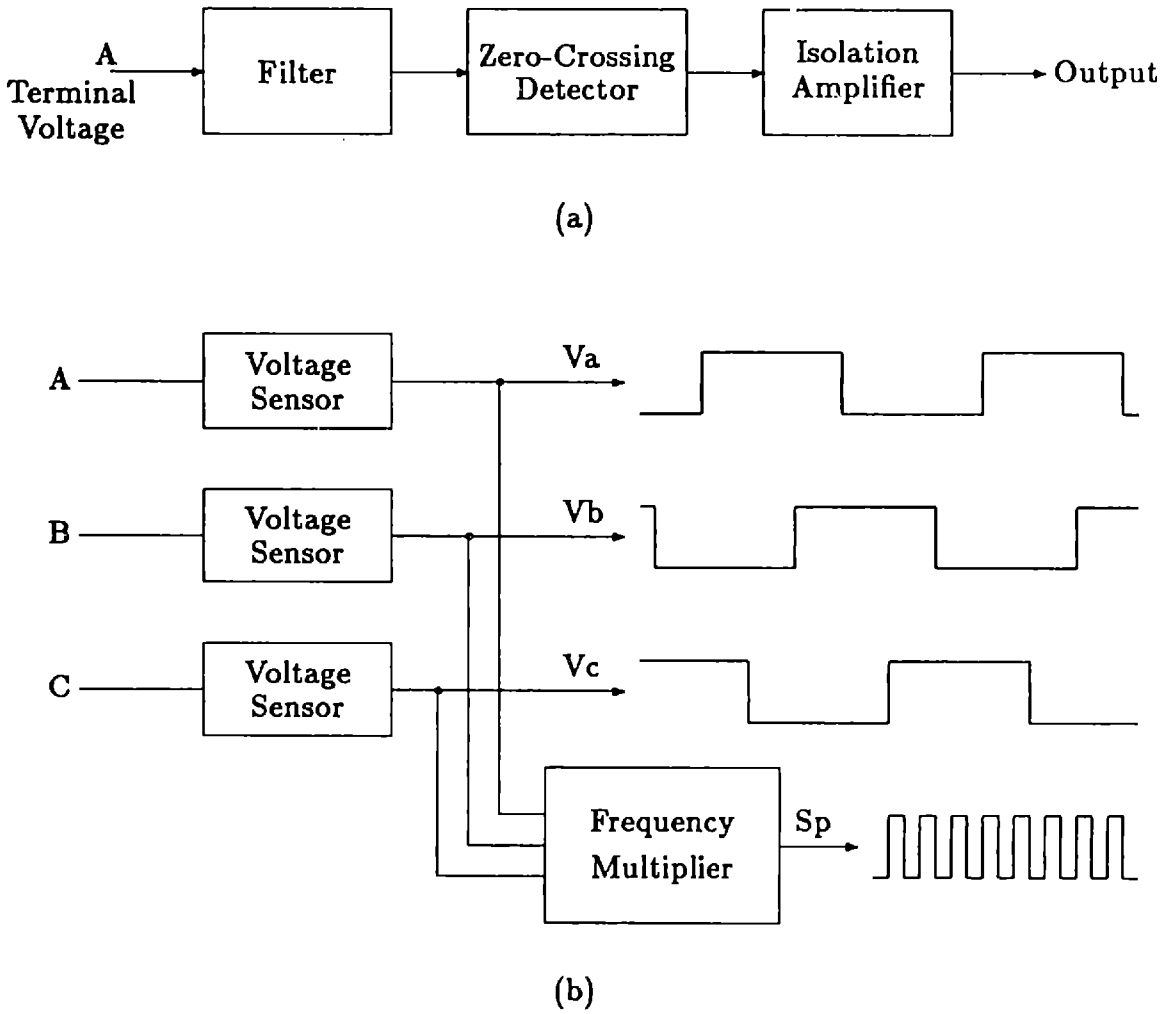


Figure 3.2: Waveform detection by voltage sensing: (a) voltage sensor, (b) signals used by the controller.

speed of the rotor may vary between detection events.

3.1.2 Case Study: Waveform Detection

Waveform detection applied to a wound-rotor salient-pole three-phase wye-connected synchronous motor driven by a current inverter has been reported by Le-Huy, et al. [21]. The technique is zero-crossing detection and is achieved by monitoring the phase voltages. A block diagram of a voltage sensor is shown in Figure 3.2a and a block diagram of all three sensors is shown in Figure 3.2b.

The signals Va, Vb, Vc, and Sp, provide position information for controlling the delay angle between the machine voltages and the thyristor gate signals. The control principle is as follows: Signal Sp has a frequency which is a large multiple of the frequency of the line-to-line voltages. When a line-to-line voltage goes positive, a counter, which is clocked by Sp, begins counting. When the counter times out, a gate signal is applied to the thyristor for the next phase in the direction of rotation. While the current is commutating from one phase to the next, the speed controller reloads the delay counter with a number appropriate to maintain the required torque angle. When the measured voltage falls negative, the counter begins counting again and when it times out, the gate signal is reset.

The signal Sp and the delay counter provide the information normally obtained from a position encoder. The resolution of the delay is determined by the multiplying factor used to obtain Sp. Note that the frequency of Sp is adjusted only at each zero-crossing detected in the line-to-line voltages and therefore, information about the current speed of the motor is added to Sp only at discrete points in time—a classic example of a shortcoming of waveform detection techniques. The correctness of the delay counter depends on Sp accurately reflecting the current speed of the motor. The accuracy of the effective position estimate is based on the motor not changing speed rapidly.

In a related paper, Davoine, et al. [22] present a technique for determining the position of the rotor at startup. The controller must know the initial position of the rotor in order to compute the correct initial firing order for starting in the desired direction with maximum torque. The initial position is computed by applying a step voltage to the wound rotor and monitoring the induced voltages at the terminals of the three stator phases. The motor is then started using forced commutation and accelerated until the phase voltage is sufficient to ensure thyristor blocking, at which time natural commutation may commence. This technique of determining the starting position of the rotor is clearly not applicable to the permanent-magnet motor described in Chapter 2 since there are no windings on the rotor. An alternative would be to use stator inductance measurements to determine the initial rotor position with a drawback being that electromagnetic torque would be applied to the rotor if the stator is wye-connected.

The scheme presented in this paper is an elegant example of waveform detection techniques. When natural commutation is occurring, the phase voltages provide an indicator of the position of the shaft just as the open-circuit speed voltage provides

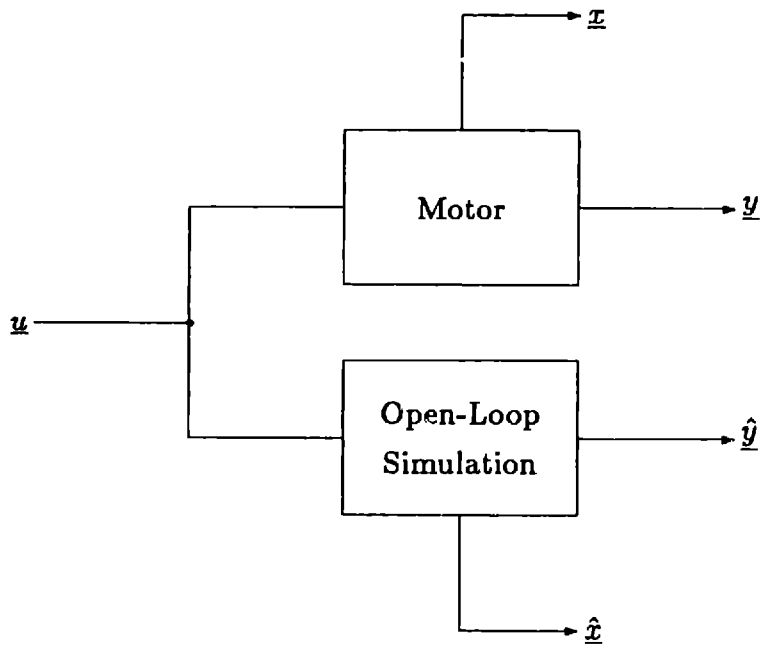


Figure 3.3: Open-loop simulation. The simulator is driven by the same input $\underline{u}$ which drives the motor and simulates the motor. Estimates $\hat{\underline{x}}$ and $\hat{\underline{y}}$ of the true state $\underline{x}$ and the true output $\underline{y}$, respectively, are produced.

an indicator of position when the phases are independent.

3.2 Open-Loop Simulations

Open-loop simulations attempt to compute a continuous estimate of the position of the motor shaft using a model of the motor. This approach is in contrast to waveform detection (Section 3.1) where an update to the position estimate is obtained only when a specific event in the monitored waveforms occurs. Therefore, an advantage of open-loop simulation over waveform detection is that if the simulation is accurate, the controller is provided with a continuous estimate of the position of the shaft. In Figure 3.3, a model of the motor is driven by the same input that drives the motor. The motor produces the variables $\underline{x}$ and $\underline{y}$ while the model produces the corresponding estimates $\hat{\underline{x}}$ and $\hat{\underline{y}}$.

How accurate the estimates $\hat{\underline{x}}$ and $\hat{\underline{y}}$ are depends on the sophistication of the

model. Simpler schemes compute position θ directly from measurements of the current waveforms. This approach is similar to direct flux estimation schemes used for vector control of induction motors. More sophisticated approaches (indirect estimation) include dynamics in the motor model. For example, rather than computing position directly from the noisy current measurements, the current measurements could be used to drive a differential equation for the shaft speed. Position could then be obtained by integrating the speed estimate. The smoothing action of the integrator would provide an improved position estimate. Even better estimates of the shaft position might be obtained by including the dynamics of the stator currents and producing estimates of the currents. These current estimates, rather than the noisy current measurements, could then be used to drive the speed dynamics.

3.2.1 Examples of Open-Loop Simulations

Endo, et al. [23] present a method for digitally computing a position estimate. The motor considered is a permanent-magnet stepper with independent phase windings. Since the windings are independent, the speed voltage in the off windings may be measured and the position of the rotor directly computed, just as a resolver-to-digital IC would compute a position given two sinewave signals in quadrature. This method is the logical next step from the speed voltage waveform detection method of Hair [19].

Hofer [24] illustrates how the current and voltage in a converter feeding a DC motor may be computed open-loop using on-off (i.e. not analog) information from the controller and converter. This technique is interesting in that no analog to digital converters are needed. The average converter current and voltage is computed using the firing angle, the current flow angle, and the overlap angle, all of which may be represented by binary signals. Unfortunately, Hofer refers to the algorithms as binary observers. As will be detailed in Section 3.3.1, an observer, by definition, must have corrective feedback on its state estimate. Hofer's binary observers do not contain corrective feedback and therefore should more appropriately be called binary simulators.

3.2.2 Case Study I: Open-Loop Simulation

Stiebler and Kiewe [25] give an interesting account of the technique of open-loop simulation as applied to a two-phase two-pole permanent-magnet stepper motor.

The rotor flux component is computed from measurements of the applied stator voltage and the resulting stator current. Since the flux vector is fixed to the rotor in a permanent-magnet motor, the position of the rotor is thus known. The flux estimate is computed using analog circuitry and then fed to a microcomputer which performs the motor control using the estimate. It is noted that by providing the controller with a continuous estimate of the shaft position, the switching angle may be continuously adjusted for the maximum possible acceleration.

Assume the two identical stator phases a and b which are in quadrature and thus do not couple magnetically. Ignore saturation. The flux linkages of the phases are

$$\lambda_a = \int (v_a - Ri_a)\,dt + \lambda_{ao} \tag{3.1}$$

$$\lambda_b = \int (v_b - Ri_b)\,dt + \lambda_{bo} \tag{3.2}$$

where

$$\lambda_a = Li_a + \lambda_r \cos\theta \tag{3.3}$$

$$\lambda_b = Li_b + \lambda_r \sin\theta \tag{3.4}$$

and λ_{ao} and λ_{bo} are the initial conditions of the flux. Position θ is defined to be zero when the rotor flux vector is aligned with phase a. Substituting (3.3) and (3.4) into (3.1) and (3.2) and rearranging yields

$$\lambda_r \cos\theta = \int (v_a - Ri_a)\,dt - Li_a + \lambda_{ao} \tag{3.5}$$

$$\lambda_r \sin\theta = \int (v_b - Ri_b)\,dt - Li_b + \lambda_{bo}\,. \tag{3.6}$$

For two-phase, full-step operation, the initial position is defined to be at $\theta_o = \pi/4 \bmod \pi/2$. Therefore, define the detent position as

$$\phi = \theta - \frac{\pi}{4}\,. \tag{3.7}$$

such that the initial position ϕ_o is zero. Substituting (3.7) into (3.5) and (3.6) and then adding the two equations yields

$$\Lambda = \lambda_r \cos\phi = \frac{1}{\sqrt{2}}\left(\int [v_a + v_b - R(i_a + i_b)]\,dt + L(i_a + i_b)\right) + \Lambda_o \tag{3.8}$$

where

$$\sqrt{2}\Lambda_o = L(i_{ao} + i_{bo}) + \sqrt{2}\lambda_r \cos\phi_o\,. \tag{3.9}$$

The initial currents i_{ao} and i_{bo} are measured at the beginning of the step. The computed value of rotor flux, Λ, may be used by the controller as a position signal since its value depends on θ. Note that this technique is a simple case of indirect vector control which is commonly used with induction motors. The position estimate implicit in Λ is subject to increasing errors during the integration since there is no corrective feedback. The integration is open-loop. Nevertheless, experimental results included in the paper show that even with a simple closed-loop controller which uses the estimated shaft position, the stepping motor can be operated at higher speeds than it can be when run open-loop.

3.2.3 Case Study II: Open-Loop Simulation

Watanabe, et al. [26] have reported a scheme to estimate rotor position by measuring the phase currents, computing the voltages using a model of the motor, and then computing position by comparing measured voltages with the computed voltages. The machine considered is a three-phase wye-connected two-pole salient permanent-magnet synchronous motor.

Assume two-phase axes $\alpha\beta$ fixed to the stator Also, assume the two-phase axes dq moving with the *estimated* rotor position. The voltage in the α axis, v_α, may be expressed in terms of the voltage in the d axis, v_d, via the inverse dq transformation (2.71). The inverse transformation is not exact since only the estimated position is available. The error between the measured α axis voltage and the computed α axis voltage is

$$e_{v_\alpha} = v_\alpha - (R + \mathrm{p}L_d)i_\beta - \omega L_q i_\alpha \tag{3.10}$$

where R is the phase resistance and L_d and L_q are the dq phase inductances. (In the model used in this thesis, L_d and L_q are identical since the rotor is smooth.) The symbol p is the time differential operator d/dt. If v_α is expressed using dq frame current relations, and i_α and i_β are expressed in the dq frame, the error reduces to

$$e_{v_\alpha} = K \sin\theta \tag{3.11}$$

where

$$K = (\mathrm{p}L_d - \mathrm{p}L_q)i_q + \omega(L_d - L_q)i_d - \omega K \tag{3.12}$$

Constant K is the magnet constant as in (2.64). Next, the voltage in the negative β axis, v_β, may be expressed in terms of the voltage in the dq axes via the negative

of the inverse dq tranformation. The error between the measured negative β axis voltage and the computed negative β axis voltage is

$$e_{v_\beta} = v_\beta + (R - \mathrm{p}L_d)i_\alpha - \omega L_q i_\beta \tag{3.13}$$

which may be expressed as

$$e_{v_\beta} = K \cos\theta \,. \tag{3.14}$$

Therefore, using (3.11) and (3.14), the rotor position θ may be computed as

$$\hat{\theta} = \arctan\left(\frac{e_{v_\alpha}}{e_{v_\beta}}\right) . \tag{3.15}$$

The accuracy of the position estimate depends on the accuracy of the measurements at each sample since past information is not used. For example, if a momentary disturbance occurs in a measurement, the position estimate will directly reflect this glitch since past information indicating that the current position estimate is accurate is not utilized. This problem is aggravated by the fact the computations include derivatives of the measurements. Discussions with Watanabe revealed that steady-state accuracies of approximately fifteen electrical degrees have been obtained in experiments.

Instead of simply performing a difference slope on the position estimate to obtain speed, Watanabe computes the variables, v_q, i_d, and i_q using the estimate $\hat{\theta}$ and then inverts the quadrature current equation. The quadrature current equation for a salient-pole machine is

$$v_q = (R + \mathrm{p}L_q)\, i_q - \omega L_d i_d + K\omega \tag{3.16}$$

(in contrast, recall that (2.72) is for a smooth-rotor machine) and therefore a speed estimate using estimated dq variables is

$$\hat{\omega} = \frac{\hat{v}_q - (R + \mathrm{p}L_q)\,\hat{i}_q}{K - L_d \hat{i}_d} \,. \tag{3.17}$$

Another scheme for the simpler case of a smooth-rotor permanent-magnet synchronous machine that is similar to Watanabe's involves measuring the currents and voltages and directly computing the voltages. The electrical dynmamics derived in Chapter 2, (2.64), are

$$\frac{d\,i_\alpha}{dt} = -\frac{R}{L} i_\alpha + \frac{K}{L} N\omega \sin N\theta + \frac{1}{L} v_\alpha \tag{3.18}$$

$$\frac{d\,i_\beta}{dt} = -\frac{R}{L} i_\beta - \frac{K}{L} N\omega \cos N\theta + \frac{1}{L} v_\beta \,. \tag{3.19}$$

The rotor position may be computed directly from these equations as

$$\theta = \frac{1}{N}\text{atan2}\left[-\frac{di_\beta}{dt} - \frac{R}{L}i_\beta + \frac{1}{L}v_\beta, \frac{di_\alpha}{dt} + \frac{R}{L}i_\alpha - \frac{1}{L}v_\alpha\right] \tag{3.20}$$

and rotor speed computed as

$$\omega = \frac{L}{KN}\sqrt{\left[\frac{di_\alpha}{dt} + \frac{R}{L}i_\alpha - \frac{1}{L}v_\alpha\right]^2 + \left[-\frac{di_\beta}{dt} - \frac{R}{L}i_\beta + \frac{1}{L}v_\beta\right]^2}. \tag{3.21}$$

This method, like Watanabe's, suffers from the need to take derivatives of the current.

3.3 Observer-Based Estimators

Observers are a natural extension of open-loop simulations. The improvement is to add a corrective feedback signal that ensures that the estimate being produced is at least close to the true value. The theory of observers for linear systems is well-developed and if the model for the process is known, designing an observer that will accurately track the state of a linear system is a relatively simple matter. Unfortunately, the models of AC motors are nonlinear and the observer theory for linear systems does not always apply. The term observer-based estimators is coined here to describe estimators that have the basic structure of an observer but that have been modified to the extent that they cannot truly be called observers.

3.3.1 Observer Theory

Observers for unmeasurable states have been studied since the earliest days of state-space theory. They originally were not called observers as the term wasn't introduced until the mid-1960's by Luenberger [27]. The well-known Kalman filter, Kalman [28], which appeared several years before Luenberger's approach, is an observer that is optimal when Gaussian process and measurement noise are present. Luenberger's contribution in considering the simpler deterministic problem was to show how a reduced-order observer could be constructed to estimate only those states that are unmeasurable. Luenberger's 1971 paper [29] is the definitive reference on linear time-invariant observers.

The general structure of an observer is shown in Figure 3.4. The observer is contained within the dashed box. Outside the dashed box is the plant which is the

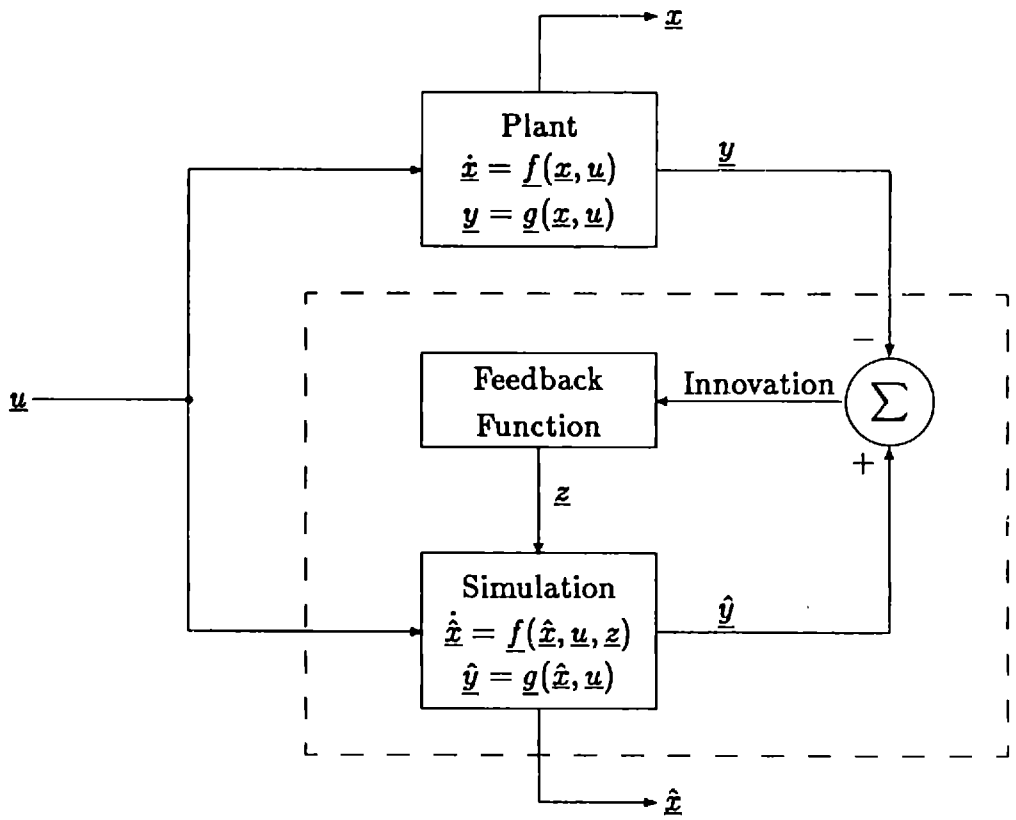


Figure 3.4: General structure of an observer system.

dynamic process producing the state $\underline{x}$ and the output $\underline{y}$. The plant is driven by the input $\underline{u}$.

The observer consists of three elements: the simulator, the innovation, and the feedback function. The simulator is a dynamic process that produces the state $\hat{\underline{x}}$ and the output $\hat{\underline{y}}$, which are estimates of the plant state $\underline{x}$ and and plant output $\underline{y}$. It is driven by the same input $\underline{u}$ that drives the plant. The simulator is usually a dynamic model of the plant. The simulator is simply the open-loop simulator in Figure 3.3. The improvement of an observer over an open-loop simulation is the use of corrective feedback based on innovation to improve the estimates $\hat{\underline{x}}$ and $\hat{\underline{y}}$. The corrective feedback is obtained by comparing the output of the simulator with the output of the plant. If the estimated output and the true output don't match, then an error signal, the innovation, is fed back into the simulator in such a way that $\hat{\underline{y}}$ evolves toward $\underline{y}$.

The plant and simulator are typically described using a set of differential equations and the differential variables in each are grouped together in the state $\underline{x}$ and $\hat{\underline{x}}$. If $\hat{\underline{x}}$ is intended to be an estimate of $\underline{x}$ then the observer is referred to as an identity observer. Note that $\hat{\underline{x}}$ is available for control purposes.

Linear Observers

Only the development of full-order identity linear observers will be presented here. See Luenberger [29] for a development of reduced-order general observers. Assume a linear system

$$\dot{\underline{x}} = \mathbf{A}\underline{x} + \mathbf{B}\underline{u} \tag{3.22}$$

$$\underline{y} = \mathbf{C}\underline{x} \tag{3.23}$$

where $\underline{x}$ is the state, $\underline{u}$ in the input, and $\underline{y}$ is the output. Assume another dynamic system to be used to observe the state $\underline{x}$,

$$\dot{\hat{\underline{x}}} = \mathbf{D}\hat{\underline{x}} + \mathbf{E}\underline{u} + \mathbf{K}\underline{y} \tag{3.24}$$

where $\mathbf{K}$ is a constant gain matrix and $\underline{y}$ is the measurement of the system being observed. For (3.24) to constitute a useful observer, the error between $\hat{\underline{x}}$ and $\underline{x}$ should decay to zero. Define $\underline{e} = \underline{x} - \hat{\underline{x}}$, then

$$\dot{\underline{e}} = \mathbf{D}\underline{e} + (\mathbf{A} - \mathbf{KC} - \mathbf{D})\underline{x} + (\mathbf{B} - \mathbf{E})\underline{u}\,. \tag{3.25}$$

If the error dynamics are to decay to zero then they should be independent of $\underline{x}$ and $\underline{u}$. This requires

$$\mathbf{D} = \mathbf{A} - \mathbf{KC} \tag{3.26}$$

$$\mathbf{E} = \mathbf{B} \tag{3.27}$$

which results in

$$\dot{\underline{e}} = (\mathbf{A} - \mathbf{KC})\underline{e}\,. \tag{3.28}$$

Substituting these requirements into (3.24) yields

$$\dot{\hat{\underline{x}}} = \mathbf{A}\hat{\underline{x}} + \mathbf{B}\underline{u} + \mathbf{K}(\underline{y} - \mathbf{C}\hat{\underline{x}})\,. \tag{3.29}$$

Immediately, it can be seen that the observer has the same dynamics matrix as the system being observed and that it is corrected by an innovation, or *residual*, which

is a comparison between the measured output and the observer output.[1] Written in a slightly different form, (3.29) is

$$\dot{\hat{\underline{x}}} = (\mathbf{A} - \mathbf{KC})\hat{\underline{x}} + \mathbf{B}\underline{u} + \mathbf{K}\underline{y}. \tag{3.30}$$

The eigenvalues of $\mathbf{A} - \mathbf{KC}$, and hence of the error dynamics, may be arbitrarily placed by proper choice of $\mathbf{K}$ as long as the pair $(\mathbf{A}, \mathbf{C})$ is completely *observable*. A proof of this fundamental requirement will not be given here; see Gopinath [30].

By properly choosing the gain matrix $\mathbf{K}$, the error dynamics, (3.28), can be made to decay at a rate independent of the system being observed and therefore $\hat{\underline{x}}$ can be made to converge to $\underline{x}$ as quickly as desired. The question arises as to how one should choose the gains, since large gains will also magnify any measurement noise. Additionally, excessive gains may result in actuator saturation and the excitation of unmodeled system dynamics. Guidance for the choice of optimal gains with respect to system and measurement noise is provided by the Kalman filter. If the system and measurements are linear and the offending noise is white and Gaussian, then it can be proven that the Kalman filter is the best of all possible filters, linear and nonlinear, thereby giving little reason to use any other structure. (See Kwakernaak [31] for an exhaustive account of the Kalman filter.) Choosing optimal gains in the presence of actuator saturation and the excitation of unmodeled system dynamics is a much more difficult issue since both side-effects are almost always nonlinear. The theory of optimal control attempts to address these and other nonlinearity issues.

Nonlinear Observers

The wide range of possible nonlinear system behavior has precluded the development of a general theory of systems equivalent to that available for linear systems. Linear systems are only a small subset of the set of all systems. Not surprisingly, the theory for nonlinear observers, beyond some stability results, is not well developed. In this section, some general results in nonlinear observer theory will be reviewed and in Section 3.3.2 some applications of nonlinear observers and system theory to electric machines will be presented.

[1] More generally, any system, of any dimension, with dynamics $(\mathbf{D}, \mathbf{E}, \mathbf{K})$, will track system $(\mathbf{A}, \mathbf{B})$ as long as a transformation $\hat{\underline{z}} = \mathbf{T}\underline{x}$ exists such that $\mathbf{TA} - \mathbf{DT} = \mathbf{KC}$ and $\mathbf{E} = \mathbf{TB}$, Luenberger [29]. It is convenient if the tranformation $\mathbf{T}$ is the identity since the observer state will thus have a one-to-one correspondence with the system state. An observer for which $\mathbf{T} = \mathbf{I}$ is called an identity observer.

The results for linear systems outlined in the previous section depended on expressing the error dynamics as a linear combination of the error variables. The error dynamics, explicitly expressed, could then be modified using state feedback. It is not, in general, possible to construct an explicit error system for a nonlinear system and thus designing observers for nonlinear systems is not straightforward.

The common thread found in most analyses of nonlinear systems is the use of small signal approximations via Taylor expansions. The use of this technique allows nonlinear systems to be analyzed as linear systems operating about a point in the state space. See, for example, Tsuji, et al. [32]. If the operating point changes, then the nonlinear system must be relinearized about the new position. For some nonlinear systems which satisfy special conditions, global linearizations that are independent of the operating point may be found. Global linearization is also known as external linearization. For an overview of global linearization (and many other topics), see Kokotovic [33]. For more specific results see Gully [34], Banks [35], Hunt, et al. [36], Krener and Isidori [37], and Reboulet and Champetier [38].

Probably the most common nonlinear estimator is the extended Kalman filter (EKF). It is the result when the Kalman filter is extended for use with nonlinear systems by continuously relinearizing the nonlinear system about the current estimate of the state. The derivation of the EKF may be found in Sage [39], Jazwinski [40], and Gelb [41], and thus will not be repeated here. The EKF is an observer that is optimal in the presence of uncorrelated, zero-mean, Gaussian, white noise. The corrective feedback is multiplied by a time-varying gain. The gain is optimal in the sense that it attempts to minimize the squared error between the estimated state and the true state and is computed using the partial derivatives of the system and output functions, evaluated at the current estimate of the state. These partials arise from Taylor expansions, to first order, of the nonlinear measurement and system functions. A variation on the EKF is to utilize higher order terms of the expansions; see Gelb [41]. This increases the complexity of the filter.

The EKF is not a particularly robust algorithm since the covariance matrix, which must be carried along to compute the optimal gain, is only an approximation which depends on $\underline{\hat{x}}$.[2] Many variations on the EKF may be found in the literature which attempt to get around this problem. An early example is by Pearson [42]. No

[2]When the system and measurements are linear, the EKF reduces to the Kalman filter. The covariance matrix in a Kalman filter is independent of the state $\underline{\hat{x}}$.

assumptions are made about the form of the system and measurement noise and the state estimate $\hat{\underline{x}}$ is recursively fit to the observations in a best least squares sense. An approximate solution of the least squares is found using invariant imbedding (see Section 3.3.3).

Another variation of the EKF is by Bestle [43]. In this work, the eigenvalues of the error system, linearized about the current operating point, are continuously placed. This is clearly similar to how the EKF functions except that Bestle's work may be viewed more in the context of the pole placement design of observers.

Song and Speyer [44,45,46] have developed the modified gain extended Kalman filter (MGEKF). This structure is noteworthy because it is shown in simulation to perform remarkably better than the EKF. Furthermore, its stability can be proven. The MGEKF applies only to the class of systems that have what are termed *modifiable* nonlinearities. Specifically, given the measurement function $h(x)$ and system function $f(x)$, then $f(x)$ is a modifiable nonlinear system function if there exists a time-varying function $\mathcal{H}$ such that

$$f(x) - f(\bar{x}) = \mathcal{H}\left[h(x), \bar{x}\right](x - \bar{x}) \ . \tag{3.31}$$

(See Song and Speyer [46, page 60] for a more detailed definition.) This requirement on the nonlinearities in the system and measurement allows the error dynamics to be linearized. The MGEKF is developed by modifying the EKF according to a least squares optimization, Speyer and Song [44].

Examination of the motor equations in Chapter 2 reveals that the nonlinearities in the stator-frame model are not modifiable while the nonlinearities in the rotor-frame model are modifiable. As will be considered in Chapter 4, the rotor-frame model is valid only if the rotor position is known exactly. If the rotor position is not measured and only an estimate is available, the dq tranformation will not be exact. This prevents the application of a MGEKF.

Halme, et al. [47] have reported a nonlinear filter that is derived by solving a two-point boundary value problem using the discrete minimum principle described by Selkäinaho, et al. [48]. The result strongly resembles the EKF. An application to a hydraulic lifter is presented but, unfortunately, the performance of the filter is found to be rather dismal.

3.3.2 Examples of Observer-Based Estimators

Many examples of observer-based estimators applied to electric machines may be found in the literature. An early example is Ueda, et al. [49], where a nonlinear estimator is applied to a smooth rotor synchronous generator on an infinite bus. The estimator is derived by expanding the system and measurement functions to first order about the current estimate of the state. The structure is simpler than that of an EKF because no attempt is made to minimize a stochastic cost criterion during the derivation. Rather, pole placement is used to guarantee convergence over a range of operating points.

A much simpler structure is developed by Okongwu, et al. [50]. A linear observer based on the linearized model of a wound-rotor synchronous generator is studied. Although the observer is designed for operation at a specific operating point, it is shown that it performs satisfactorily for small perturbations about a wide range of operating points.

Yamashita and Taniguchi [51] apply an observer that has the structure of a linear observer, that is, constant gains on linear innovation, to a nonlinear model of a synchronous generator. The derivation of the observer begins with the system

$$\frac{d\underline{x}}{dt} = \mathbf{A}\underline{x} + \underline{f}(\underline{x}) + \mathbf{B}\underline{u} \tag{3.32}$$

$$\underline{y} = \mathbf{C}\underline{x} + \underline{g}(\underline{x}) \tag{3.33}$$

and an observer

$$\frac{d\underline{z}}{dt} = \mathbf{A}\underline{z} + \underline{f}(\underline{z}) + \mathbf{G}\left[\underline{y} - \mathbf{C}\underline{z} - \underline{g}(\underline{z})\right] + \mathbf{B}\underline{u} \tag{3.34}$$

is designed. A procedure for optimally choosing the gain matrix $\mathbf{G}$ is presented. Limited simulation results are included. The simple structure of this observer is appealing, and is similar to stator-frame observer considered in Section 4.1 of this thesis. A potential drawback is that the Yamashita observer does not exploit the unique qualities of the nonlinearities in the generator, and therefore misses the potential for even faster converging estimates.

Observer-based estimators have also been studied for other types machines. For example, the use of linear observers for the estimation of rotor flux in induction machines is considered by Verghese and Sanders [52] and references therein. In this thoughtful, well-referenced paper, the concepts of direct and indirect rotor flux estimation, which are often confused in the literature on induction machines, are

elucidated. A *dq* transformed model of an induction motor with a constant speed assumption is used so that the system, and consequently the observer, are linear. This paper clearly illustrates the application of a linear observer to a practical system.

Lumsdaine, et al. [53] have developed an observer for the variable-reluctance motor where spatial variations in phase inductances affect the evolution of the flux in the phases as the rotor rotates. An observer is constructed that models the spatially-varying motor dynamics, and a very accurate position estimate is produced under a constant speed assumption. The structure of this observer is nonlinear since the motor dynamics and the feedback gain on the innovation are nonlinear functions of the estimated states. The feedback gain is a sinusoidal function and was motivated by the sinusoidal variation of the stator inductances. The experimental results reported in [53] are very encouraging.

3.3.3 Case Study: Observer-Based Estimators

In the extended Kalman filter, the nonlinear system is "fit" to the linear Kalman filter by linearizing it about the current state estimate. A significantly different approach is to find a solution to the nonlinear least square error problem directly. As one might suspect, finding a closed form solution is quite difficult. However, it is possible to find an approximate solution, and the technique used to do this is known as *invariant imbedding.*

If the time at which the system reaches its final state is assumed to approach infinity, then the solution to the original two-point boundary value problem may be considered to be imbedded in the infinite final time problem. The final state at infinity is held invariant and hence the name of the technique, invariant imbedding, Bellman [54], Sage [55]. The resulting estimator is similar to the EKF since they both attempt to minimize the estimate of the square state error and they both utilize a first order Taylor expansion of the state and measurement functions about the current estimate of the state. However, the invariant imbedding observer (IBO) also attempts to minimize the estimate of the square measurement error which results in an output error correction term in the covariance dynamics.

The very interesting IBO will not be derived here as an excellent presentation may be found in Gharban and Cory [56]. (Also see Sage [55] for more on invariant imbedding and Kirk [57] for an excellent presentation of the Euler-Lagrange method

which is used by Gharban and Cory.) The results are as follows. Consider the system and measurements

$$\dot{\underline{x}}(t) = \underline{f}(\underline{x}(t),t) + \mathbf{G}(\underline{x},t)\underline{w}(t) \tag{3.35}$$

$$\underline{z}(t) = \underline{h}(\underline{x}(t),t) + \underline{v}(t) \tag{3.36}$$

where $\underline{w}(t)$ and $\underline{v}(t)$ are uncorrelated, zero-mean, Gaussian, white-noise processes with covariance matrices $\mathbf{R}$ and $\mathbf{Q}$, respectively. An uncorrected simulation of this system would be

$$\dot{\hat{\underline{x}}} = \underline{f}(\hat{\underline{x}}(t),t) + \mathbf{G}(\underline{x},t)\hat{\underline{w}}\,. \tag{3.37}$$

An estimate of the measurement error is given by

$$\hat{\underline{v}} = \underline{z}(t) - \underline{h}(\hat{\underline{x}}(t))\,. \tag{3.38}$$

The problem is to minimize the quadratic cost function

$$J = \frac{1}{2}\int_{t_0}^{t_f} \left[\hat{\underline{v}}^T(t)\mathbf{R}^{-1}\hat{\underline{v}}(t) + \hat{\underline{w}}^T(t)\mathbf{Q}^{-1}(t)\hat{\underline{w}}(t)\right] dt \tag{3.39}$$

while satisfying (3.37) and (3.38). The solution of this problem uses the Euler-Lagrange technique, Kirk [57], to produce a two-point boundary value problem. The method of invariant imbedding is then used to find a recursive solution to the two-point boundary value problem. The resulting invariant imbedding observer is

$$\dot{\hat{\underline{x}}} = \underline{f}(\hat{\underline{x}}(t),t) + \mathbf{P}(t)\left.\frac{\partial \underline{h}^T(\underline{x}(t),t)}{\partial \underline{x}(t)}\right|_{\hat{\underline{x}}(t)} \mathbf{R}^{-1}\left[\underline{z}(t) - \underline{h}(\hat{\underline{x}}(t),t)\right] \tag{3.40}$$

$$\begin{aligned}\dot{\mathbf{P}}(t) = {} & \mathbf{P}(t)\left.\frac{\partial \underline{f}^T(\underline{x}(t),t)}{\partial \hat{\underline{x}}(t)}\right|_{\hat{\underline{x}}(t)} + \left.\frac{\partial \underline{f}(\hat{\underline{x}}(t),t)}{\partial \hat{\underline{x}}(t)}\right|_{\hat{\underline{x}}(t)} \mathbf{P}(t) \\ & + \mathbf{P}(t)\frac{\partial}{\partial \underline{x}(t)}\left[\frac{\partial \underline{h}^T(\underline{x}(t),t)}{\partial \underline{x}(t)}\mathbf{R}^{-1}(t)\left[\underline{z}(t) - \underline{h}(\underline{x}(t),t)\right]\right]\Bigg|_{\hat{\underline{x}}(t)} \mathbf{P}(t) \\ & + \mathbf{G}(\hat{\underline{x}}(t),t)\mathbf{Q}(t)\mathbf{G}^T(\hat{\underline{x}}(t),t)\end{aligned} \tag{3.41}$$

where $\mathbf{P}$ is a symmetric, positive definite covariance matrix for $\hat{\underline{x}}(t)$. Note that the IBO state estimate dynamics equation (3.40) is the same as the EKF state estimate dynamics equation, Gelb [41, page 188]. The error covariance dynamics of the IBO, (3.41), and EKF are similar but not identical. In particular, the invariant

imbedding produced a Hessian of $\underline{h}(\underline{x})$ which provides feedback correction in the error covariance dynamics. This feedback mechanism is not present in the extended Kalman filter which leads one to suspect that the IBO might perform better than the EKF. In any event, the feedback will almost certainly yield better error covariance dynamics when the initial error covariances are not known.

Gharban and Cory apply the IBO to a smooth-rotor synchronous generator. The model is fourth-order with two electrical states and two mechanical states. The IBO shares a striking disadvantage with the EKF and that is the necessity of carrying along a system of covariance dynamics which increases the order of the observer from four to fourteen. Although optimization with respect to system noise is clearly desirable, it incurs a heavy price. No consideration is made of the structure of the underlying nonlinear system. Rotating machines have strong symmetries that might be exploited to yield simpler structures. The invariant imbedding observer would have to be demonstrably superior to other methods before it could be seriously considered for real-time application. The limited simulation results provided by Gharban and Cory do not establish the IBO as a viable method.

Chapter 4

Proposed Observers

In this chapter, several observers which produce an estimate of the dynamic state of the motor are developed. The electrical variables available to drive the observer are measured in the stator frame and therefore an observer that operates in the stator frame is considered first. Next, observers that operate in the rotor frame are considered. It is found that in either frame, if corrective feedback is applied only in the electrical subdynamics, the performance of the observer is poor. The final observer considered, which is the used throughout the remainder of this thesis, incorporates corrective feedback in both the electrical and mechanical subdynamics through an insightful interpretation of the innovation process when the dq transformation is applied. The resulting state observer is one that could not have been developed from a general expression for a nonlinear system.

4.1 Stator-Frame Observer

The general form of the stator-frame model (2.64)–(2.66) is

$$\frac{d\underline{x}}{dt} = \underline{f}(\underline{x}, \underline{v}) \tag{4.1}$$

where the driving input is the stator-frame voltage $\underline{v}$ and the state $\underline{x}$ is

$$\underline{x} = \begin{bmatrix} \underline{i} & \omega & \theta \end{bmatrix}^T . \tag{4.2}$$

Define the output of the system to be the stator-frame currents

$$\underline{y} = \underline{i} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \underline{x} . \tag{4.3}$$

The system is driven by the voltage $\underline{v}$ and the resulting output is the current $\underline{i}$. Applying the stator-frame model of the motor to (3.34) yields a structure similar to that considered by Yamashita [51], discussed in Section 3.3.2. The corrective feedback is simply a set of gains applied to the differences between the estimated and measured currents. Denote estimated values with a superscript carat. The observer is

$$\frac{d\hat{\underline{i}}}{dt} = -\frac{R}{L}\hat{\underline{i}} - \frac{K}{L}N\hat{\omega}\exp(\mathbf{J}N\hat{\theta})\begin{bmatrix}0\\1\end{bmatrix} + \frac{1}{L}\underline{v} + \mathbf{G}_{\underline{i}}(\underline{i}-\hat{\underline{i}}) \tag{4.4}$$

$$\frac{d\hat{\omega}}{dt} = -\frac{B}{H}\hat{\omega} + \frac{KN}{H}\hat{\underline{i}}^T\exp(\mathbf{J}N\hat{\theta})\begin{bmatrix}0\\1\end{bmatrix} - \frac{C}{H}\frac{\hat{\omega}}{|\hat{\omega}|} - \frac{1}{H}\tau + \mathbf{G}_{\omega}(\underline{i}-\hat{\underline{i}}) \tag{4.5}$$

$$\frac{d\hat{\theta}}{dt} = \hat{\omega} + \mathbf{G}_{\theta}(\underline{i}-\hat{\underline{i}}) \tag{4.6}$$

How to choose the gains is a difficult problem since the system is nonlinear.

To explore the behavior of this observer (and other observers later in this chapter), simulations of the motor and observer were performed. The simulation software is described in Section 5.1. The parameters used in all the simulations are given in Figure 5.3 except when noted.

Consider the simulation results, shown in Figure 4.1, of the error between the observer (4.4)–(4.6) *with all gains zero* and the steady-state motor driven by voltages appropriate for operation at 1000 rpm. All simulator initial conditions except for the initial observer position are appropriate for steady-state operation at 1000 rpm. The initial observer position is set to be ahead of the initial motor position by 10% in electrical radians. The initial motor position is -1.56 electrical radians and so the initial observer position is set to -1.40 electrical radians. For the time shown in Figure 4.1, the motor turns 6.67 mechanical revolutions.

Simulations reveal that the response of (4.4)–(4.6) cannot be significantly improved with nonzero gains and that even finding gains for which the system remains stable is difficult. Consider a simulation using the same initial conditions as in Figure 4.1 but with nonzero gains. The position error response is shown in Figure 4.2. The response of the stator-frame observer is very slow. For larger perturbations the observer is unstable.

The effect of the corrective feedback in (4.4)–(4.6) is complicated by the fact that the equations are nonlinear. In a smooth-rotor permanent-magnet motor, where the magnets are mounted on the surface of the rotor (see Figure 2.3), the

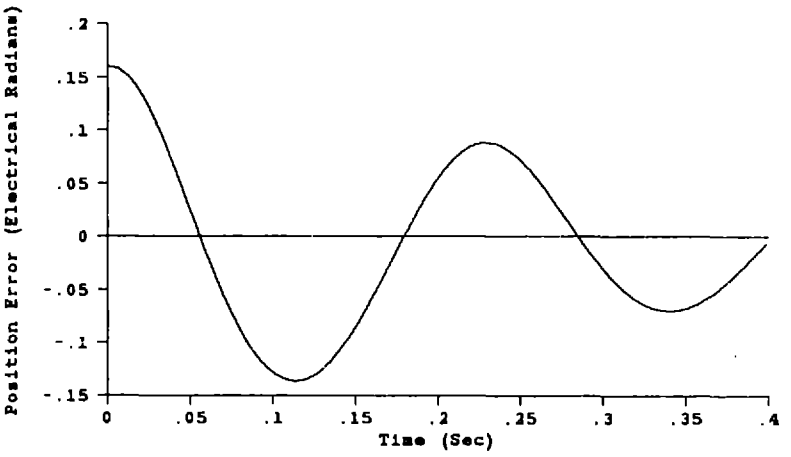


Figure 4.1: Stator-frame observer position error response with zero gains and the initial observer position set 10% ahead of the initial motor position operating at 1000 rpm.

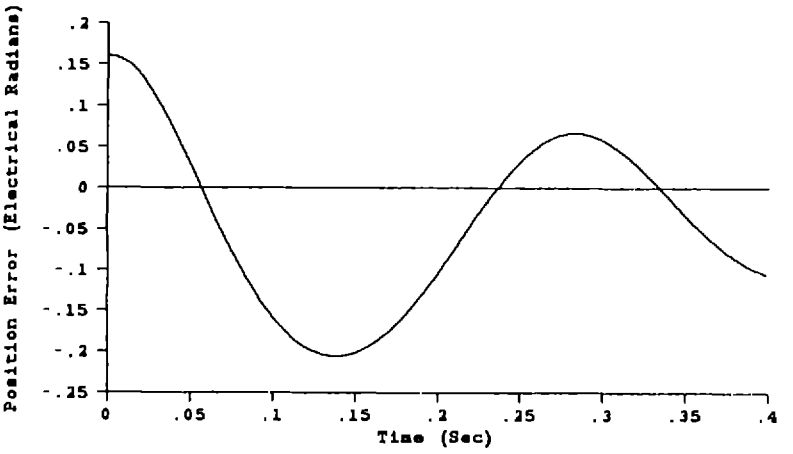


Figure 4.2: Stator-frame observer position error response with the initial observer position set 10% ahead of the initial motor position operating at 1000 rpm. $\mathbf{G_i} = \begin{bmatrix} 200 & 0; & 0 & 200 \end{bmatrix}$, $\mathbf{G_\omega} = \begin{bmatrix} 0 & 0 \end{bmatrix}$, $\mathbf{G_\theta} = \begin{bmatrix} 0 & 0 \end{bmatrix}$.

linear term due to the stator inductance in (4.4) (the first term on the right), is small compared to the speed voltage term (the second term on the right) and thus it is the nonlinearity in the equation that dominates its behavior. More important, however, is the fact that in this observer, error in the position estimate $\hat{\theta}$ enters into the electrical subdynamics only through the trigonometric functions sine and cosine and thus has no gain assigned to it. Better performance might be achieved if a meaningful gain could be attached to an error in the estimated position. Gains G_ω and G_θ were set to zero since values could not be found that would improve the performance of the observer. As a result, correction occurs only in the electrical variables. Since the electrical and mechanical subdynamics of the system are only loosely coupled (Verghese, et al. [58]), errors in position are corrected no faster than the mechanical time constant of the system.

4.2 Rotor-Frame Observers

The general form of the dq frame model of the motor (2.72)–(2.74) is

$$\frac{d\underline{\bar{x}}}{dt} = \underline{f}(\underline{\bar{x}}, \underline{\bar{v}}) \tag{4.7}$$

where the driving input is the rotor-frame voltage $\underline{\bar{v}}$ and the state $\underline{\bar{x}}$ is

$$\underline{\bar{x}} = \begin{bmatrix} \bar{i} & \omega & \theta \end{bmatrix}^T . \tag{4.8}$$

Note that the state, input, and nonlinear function are different in (4.7) than in (4.1). Define the output of the system to be the rotor-frame currents

$$\underline{\bar{y}} = \bar{i} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \underline{\bar{x}} . \tag{4.9}$$

The system is driven by the voltage $\underline{\bar{v}}$ and the resulting output is the current $\bar{i}$. A block diagram of the system (2.72)–(2.74) is shown in Figure 4.3.

It would appear in Figure 4.3 that the position θ does not feed back into the system and is unobservable from the current outputs. This is, in fact, true if the dq transformation is exact. The direct and quadrature currents are moving precisely with θ and thus do not contain any information about how θ is changing. The dq transformation will be exact, however, only if θ is known exactly at all times. If the position is measured, then it will be known at all times. However, if instead the position produced by the model in Figure 4.3 is used to perform the dq

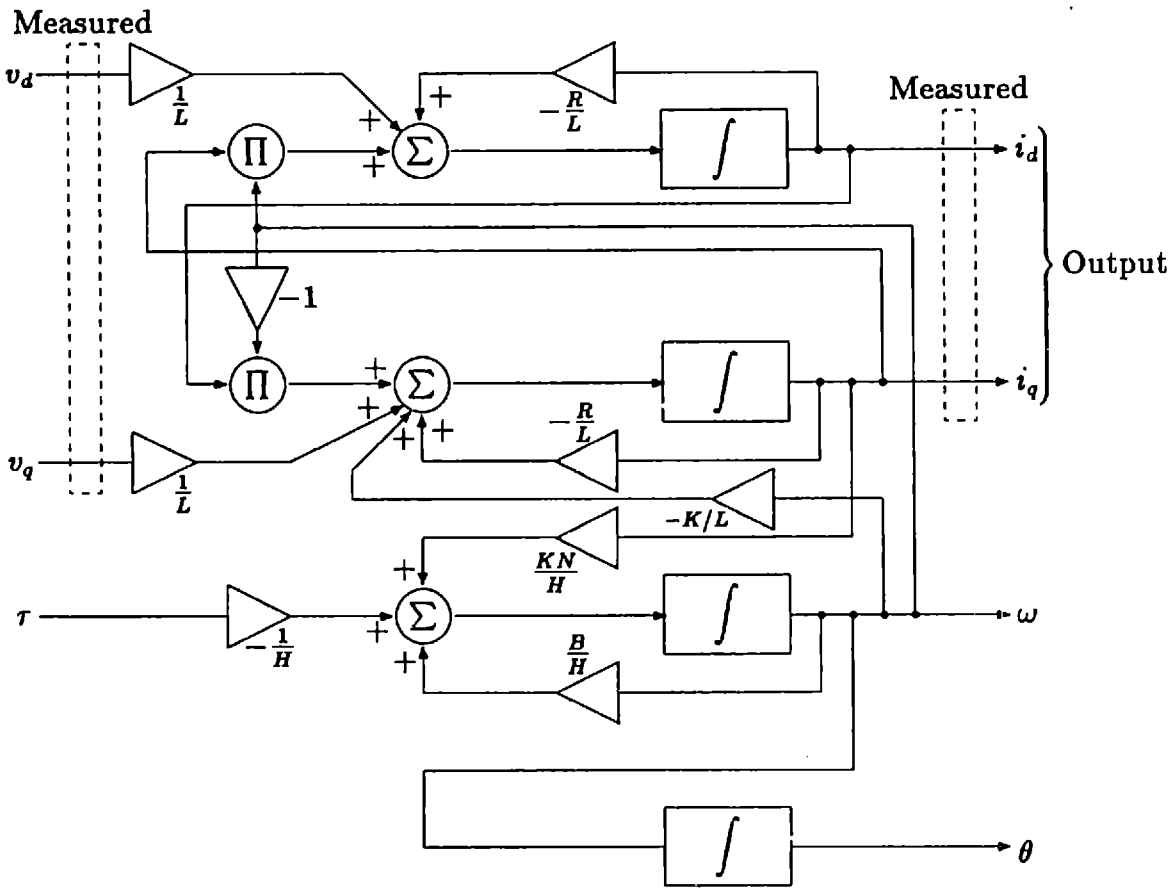


Figure 4.3: Block diagram of the dq transformed motor model. If true position is known such that $\underline{\bar{v}}$ is exact, then information about θ is not available in the output, i_d and i_q.

transformation, a feedback loop is closed back into the input voltages and position information will then be present in the output currents.

4.2.1 Estimated Rotor-Frame Observer

The position information needed to implement (2.72) and (2.73) for control purposes is usually obtained from an external position sensor. This position information is used to rotate the driving stator voltage into the rotor reference frame according to the dq transformation

$$\underline{\bar{v}} = \exp(-\mathbf{J}N\theta)\,\underline{v} \tag{4.10}$$

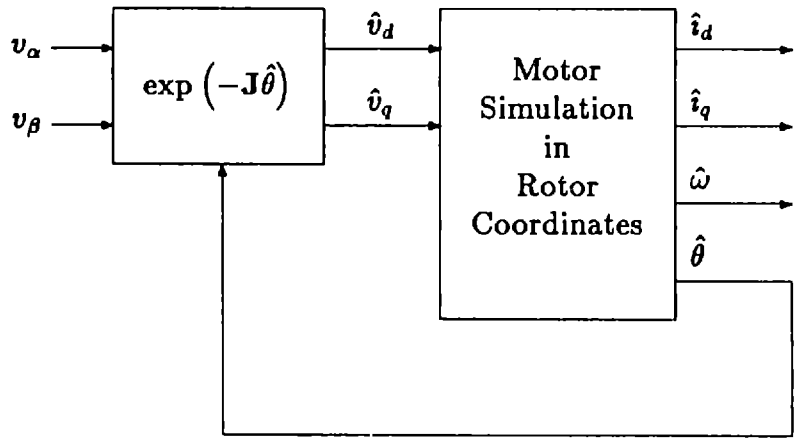


Figure 4.4: Estimated rotor-frame observer. The estimated position is used to perform the dq transformation. All of the states must be estimates since the transformed driving voltage is not exact. Signals v_α and v_β are measured.

$$\underline{\bar{i}} = \exp\left(-\mathbf{J}N\theta\right)\underline{i} \tag{4.11}$$

(see Section 2.5). Assume, however, that the rotor position is the estimated position produced by the motor simulation, rather than a measured position. The dq-transformed model (2.72)–(2.74), with estimated driving voltages, is

$$\frac{d\underline{\hat{i}}}{dt} = -\left(\frac{R}{L}\mathbf{I} + \mathbf{J}N\hat{\omega}\right)\underline{\hat{i}} - \frac{K}{L}N\hat{\omega}\begin{bmatrix}0\\1\end{bmatrix} + \frac{1}{L}\underline{\hat{v}} \tag{4.12}$$

$$\frac{d\hat{\omega}}{dt} = -\frac{B}{H}\hat{\omega} + \underline{\hat{i}}^T\frac{KN}{H}\begin{bmatrix}0\\1\end{bmatrix} - \frac{C}{H}\frac{\hat{\omega}}{|\hat{\omega}|} - \frac{1}{H}\tau \tag{4.13}$$

$$\frac{d\hat{\theta}}{dt} = \hat{\omega}\,. \tag{4.14}$$

If $\hat{\theta}$ is not precisely equal to θ, then the driving voltage will incorrectly drive the model, and thus all of the state estimates will contain some error. A block diagram of the estimated rotor-frame observer is shown in Figure 4.4. The incorrect voltage, however, will act to correct the error in $\hat{\theta}$. To see this, consider the vector diagram in Figure 4.5. The voltage driving the stator windings is V. This vector is rotating about the origin with respect to the fixed two-phase axes α and β. The direct and quadrature axes rotate with the voltage at the same frequency. If the true

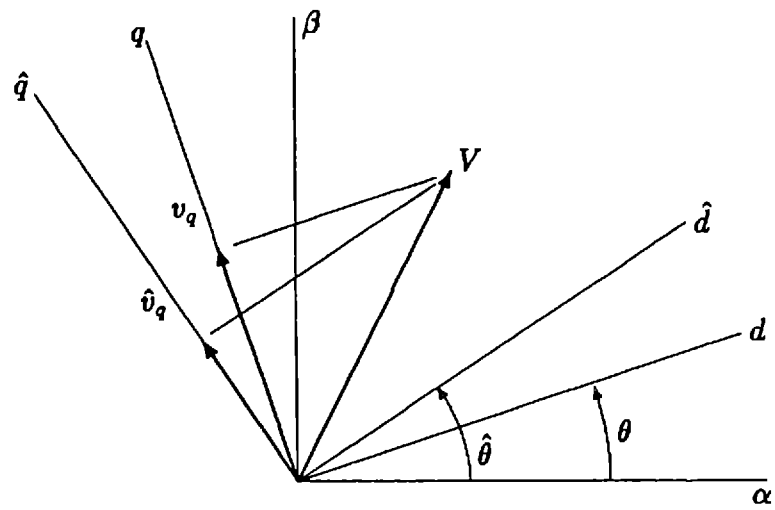


Figure 4.5: Driving voltage in the true and estimated rotor frames. Voltage V rotates with respect to the stationary $\alpha\beta$ axes. The true frame dq and the estimated frame $\hat{d}\hat{q}$ rotate with V.

position of the rotor is θ, then the voltage driving the quadrature current is v_q. If the estimated position of the rotor is $\hat{\theta}$ then the voltage driving the estimated quadrature current is $\hat{v}_q$. For the situation depicted in Figure 4.5, $\hat{\theta}$ is greater than θ, and therefore $\hat{v}_q$ is less than v_q. The result is that $\hat{i}_q$ will be driven less than i_q, and since the electromagnetic torque accelerating the rotor is proportional to the quadrature current, the estimated position $\hat{\theta}$ will fall back toward the true position θ and reduce the position error. Therefore, (4.12)–(4.14) is, in fact, an observer since there is corrective feedback. It shall be called the estimated rotor-frame observer.

When simulations of the estimated rotor-frame observer are run, the results are disappointing. Consider the initial conditions used in Figure 4.1, where the initial observer position is set 10% ahead of the motor initial position for 1000 rpm. When these initial conditions are used with (4.12)–(4.14), the results shown in Figure 4.6 are obtained. The disappointing results are due to the fact that correction occurs only in the electrical variables. In fact, the phase lag of the position estimate causes it to converge somewhat *slower* than the mechanical time constant, as can be seen by comparing Figures 4.1 and 4.6.

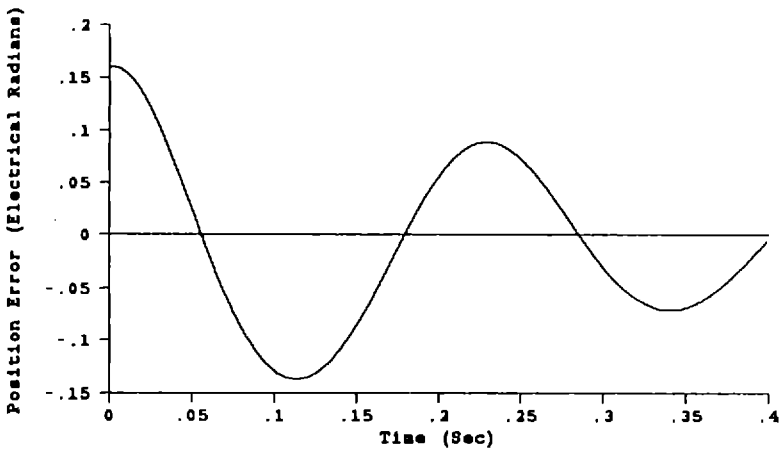


Figure 4.6: Estimated rotor-frame observer position error response with the initial observer position set 10% ahead of the initial motor position operating at 1000 rpm.

4.2.2 Theta-Computation Observer

If the dq transformation is not exact, position information will be present in the estimated direct and quadrature currents. The question arises as to how to extract and utilize this position information. The estimated rotor-frame observer (4.12)–(4.14), considered in the previous section, does not utilize the position information present in the dq currents. To extract the position, consider the following scheme. Define three positions:

- θ the true position,
- $\bar{\theta}$ the position predicted by the dq transformation, that is, the position that is required to transform the measured currents i_α and i_β into the estimated currents $\hat{i}_d$ and $\hat{i}_q$, and
- $\hat{\theta}$ the position estimated by the observer, that is, from $d\hat{\theta}/dt = \hat{\omega}$.

The exact transformation from rotor-frame estimated currents to measured stator-frame currents is

$$\underline{i} = \exp(\mathbf{J}N\bar{\theta})\hat{\underline{i}} \tag{4.15}$$

assuming $|\underline{i}| = |\hat{\underline{i}}|$. The position estimate $\bar{\theta}$ (which is *not* the same as $\hat{\theta}$) is the only unknown in this transformation. Define

$$\underbrace{\begin{bmatrix} i_\alpha & i_\beta \\ i_\beta & -i_\alpha \end{bmatrix}}_{I} = \begin{bmatrix} \cos N\bar{\theta} & -\sin N\bar{\theta} \\ \sin N\bar{\theta} & \cos N\bar{\theta} \end{bmatrix} \underbrace{\begin{bmatrix} i_d & i_q \\ i_q & -i_d \end{bmatrix}}_{\bar{I}} \tag{4.16}$$

such that

$$I = \exp(\mathbf{J}N\bar{\theta})\bar{I} \tag{4.17}$$

and therefore

$$\bar{\theta}\mathbf{I} = -\frac{1}{N}\mathbf{J}\ln(I\bar{I}^{-1}). \tag{4.18}$$

Define the driving voltage as

$$\bar{\underline{v}} = \exp(-\mathbf{J}N\bar{\theta})\underline{v}. \tag{4.19}$$

Substituting (4.18) gives

$$\begin{aligned} \bar{\underline{v}} &= \exp\left(-\mathbf{J}N\left[-\frac{1}{N}\mathbf{J}\ln\left(I\bar{I}^{-1}\right)\right]\right)\underline{v} \\ &= \bar{I}I^{-1}\underline{v}. \end{aligned} \tag{4.20}$$

With this driving voltage, an observer is thus

$$\frac{d\hat{\underline{i}}}{dt} = -\left(\frac{R}{L}\mathbf{I} + \mathbf{J}N\hat{\omega}\right)\hat{\underline{i}} - \frac{K}{L}N\hat{\omega}\begin{bmatrix} 0 \\ 1 \end{bmatrix} + \frac{1}{L}\bar{I}I^{-1}\underline{v} \tag{4.21}$$

$$\frac{d\hat{\omega}}{dt} = -\frac{B}{H}\hat{\omega} + \hat{\underline{i}}^T\frac{KN}{H}\begin{bmatrix} 0 \\ 1 \end{bmatrix} - \frac{C}{H}\frac{\hat{\omega}}{|\hat{\omega}|} - \frac{1}{H}\tau \tag{4.22}$$

$$\frac{d\hat{\theta}}{dt} = \hat{\omega} \tag{4.23}$$

where the voltages $\underline{v}$ are measured and the matrices I and $\bar{I}$ contain both measured and estimated currents.

The observer defined by (4.21)–(4.23) is nonlinear and the usual innovation process of subtracting a measured output from the estimated output to obtain a corrective feedback, as was shown in Figure 3.4, is not present. Implicitly, the nonlinear feedback in (4.21)–(4.23) provides corrective feedback. A gain matrix may be incorporated into the feedback as shown in Figure 4.7. The driving voltage

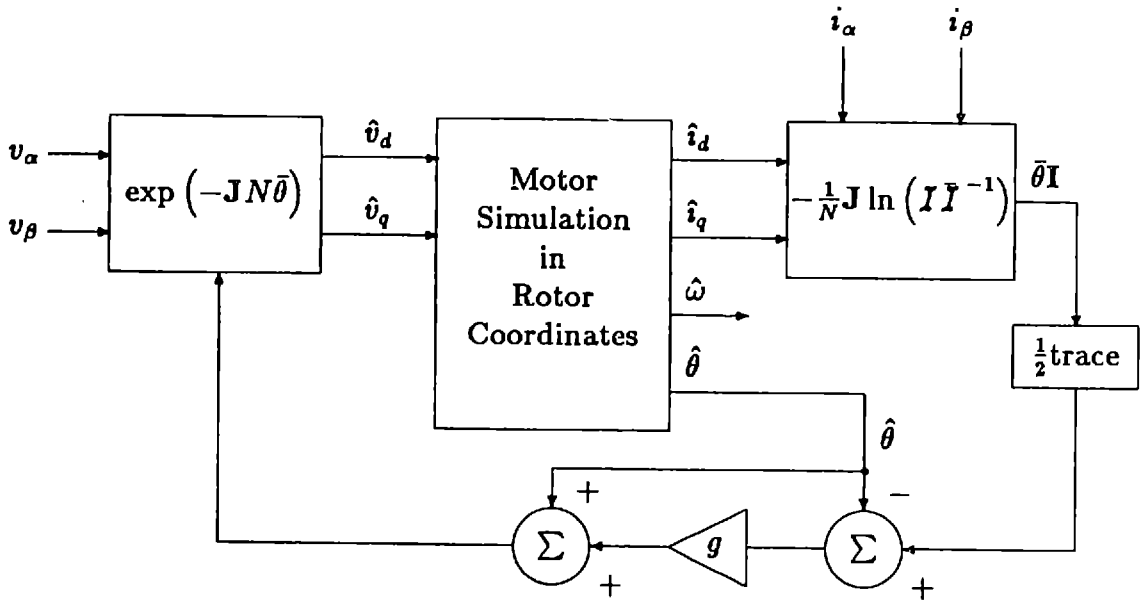


Figure 4.7: Theta-computation observer. Scalar gain g is applied to the computed value $\bar{\theta}$ to generate a position feeding into the dq transformation. Signals v_α, v_β, i_α, and i_β are measured.

for this structure is

$$\bar{\underline{v}} = \exp\left(-\mathbf{J}N\left[\hat{\theta} + g\left(\bar{\theta} - \hat{\theta}\right)\right]\right)\underline{v} \tag{4.24}$$

$$= \exp\left(-\mathbf{J}N\left[(1-g)\hat{\theta}\mathbf{I} - g\frac{1}{N}\mathbf{J}\ln\left(I\bar{I}^{-1}\right)\right]\right)\underline{v} \tag{4.25}$$

$$= \bar{I}^g I^{-g}\exp\left(-\mathbf{J}N(1-g)\hat{\theta}\right)\underline{v}. \tag{4.26}$$

The theta-computation observer using the driving voltages (4.26) is thus

$$\frac{d\hat{\underline{i}}}{dt} = -\left(\frac{R}{L}\hat{\underline{i}} + \mathbf{J}N\hat{\omega}\right)\hat{\underline{i}} - \frac{K}{L}N\hat{\omega}\begin{bmatrix}0\\1\end{bmatrix} + \frac{1}{L}\bar{I}^g I^{-g}\exp\left(-\mathbf{J}N(1-g)\hat{\theta}\right)\underline{v} \tag{4.27}$$

$$\frac{d\hat{\omega}}{dt} = -\frac{B}{H}\hat{\omega} + \hat{\underline{i}}^T\frac{KN}{H}\begin{bmatrix}0\\1\end{bmatrix} - \frac{C}{H}\frac{\hat{\omega}}{|\hat{\omega}|} - \frac{1}{H}\tau \tag{4.28}$$

$$\frac{d\hat{\theta}}{dt} = \hat{\omega} \tag{4.29}$$

and is shown in Figure 4.8.

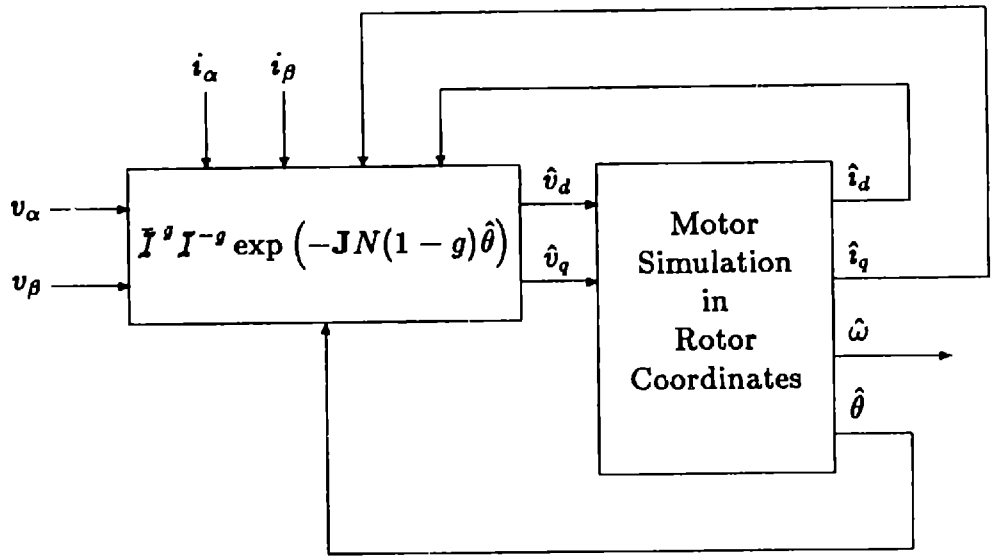


Figure 4.8: Simplified theta-computation observer. Signals v_α, v_β, i_α, and i_β are measured.

The form of the theta-computation observer is quite elegant. It contains only multiplications, thus obviating the need for sine tables in a real time implementation. Unfortunately, simulations of (4.27)–(4.29) indicate poor performance. A hunting transient with the initial observer position set 10% ahead of the 1000 rpm initial position is actually somewhat slower than the hunting transient shown in Figure 4.1. As with the previous observers considered, this observer is not stable in the presence of large perturbations.

4.2.3 Estimated-Innovation Observer

None of the observers considered thus far converge rapidly to the true position θ. Worse yet, they are unstable in the presence of large perturbations. One common feature of the observers considered is that corrective feedback is applied only in the electrical subdynamics. Thus, to accelerate the rotor quickly in the observer, large excursions of the electrical variables are required. The corrective feedback in the electrical subdynamics prevents large excursions of the electrical variables and thus the shaft will be left to converge at its natural mechanical frequency.

A new form of observer is needed, and the standard form of the identity observer

$$\frac{d\hat{\underline{x}}}{dt} = \underline{f}(\hat{\underline{x}}, \underline{u}) + \mathbf{G}(\underline{y} - \mathbf{C}\hat{\underline{x}}) , \tag{4.30}$$

provides guidance on how to proceed. In particular, the full gain matrix G provides for the inclusion of innovation in the mechanical subdynamics. By including a residual in (4.13), the rotor position will be directly affected by any error in the measured currents. The electrical and mechanical subdynamics will be tightly coupled. By closing a position feedback loop into the electrical subdynamics, errors will exist in the observer until the position estimate is correct.

The *dq* transformation is a vehicle used to feed position information into the electrical dynamics. Assuming the estimated currents $\hat{\imath}_d$ and $\hat{\imath}_q$ are correct, the measured stator currents i_α and i_β must transform precisely into the estimated *dq* frame using the position estimate. If they do not, then the position estimate must be incorrect and a resulting error signal should be generated to effect a correction.

Applying the ideas embodied in (4.30) to the *dq* transformed model (2.72)–(2.74), the estimated-innovation observer is

$$\frac{d\hat{\underline{i}}}{dt} = -\left(\frac{R}{L}\mathbf{I} + \mathbf{J}N\hat{\omega}\right)\hat{\underline{i}} - \frac{K}{L}N\hat{\omega}\begin{bmatrix} 0 \\ 1 \end{bmatrix} + \frac{1}{L}\hat{\underline{v}} + \mathbf{G}_{\underline{i}}(\check{\underline{i}} - \hat{\underline{i}}) \tag{4.31}$$

$$\frac{d\hat{\omega}}{dt} = -\frac{B}{H}\hat{\omega} + \hat{\underline{i}}^T\frac{KN}{H}\begin{bmatrix} 0 \\ 1 \end{bmatrix} - \frac{C}{H}\frac{\hat{\omega}}{|\hat{\omega}|} - \frac{1}{H}\tau + \frac{KN}{H}\mathbf{G}_\omega(\check{\underline{i}} - \hat{\underline{i}}) \tag{4.32}$$

$$\frac{d\hat{\theta}}{dt} = \hat{\omega} + \mathbf{G}_\theta(\check{\underline{i}} - \hat{\underline{i}}) . \tag{4.33}$$

This observer is severely nonlinear. Unlike a linear observer, the driving input is not a linear function. Furthermore, the measurement $\underline{y}$ in Figure 3.4 must be transformed into $\bar{\underline{y}}$ in the estimated rotor frame. These transformed measurements are denoted by an inverted superscript carat in (4.31)—(4.33),

$$\check{\underline{i}} = \exp(-\mathbf{J}N\hat{\theta})\underline{i} \tag{4.34}$$

where $\underline{i}$ is the current measured at the terminals of the motor. Although innovation is included in the position equation (4.33), *the gain matrix* $\mathbf{G}_\theta$ *will be set to zero* since including innovation in this equation introduces measurement noise directly into the position estimate, and the smoothing effect of the integration of the speed is lost.

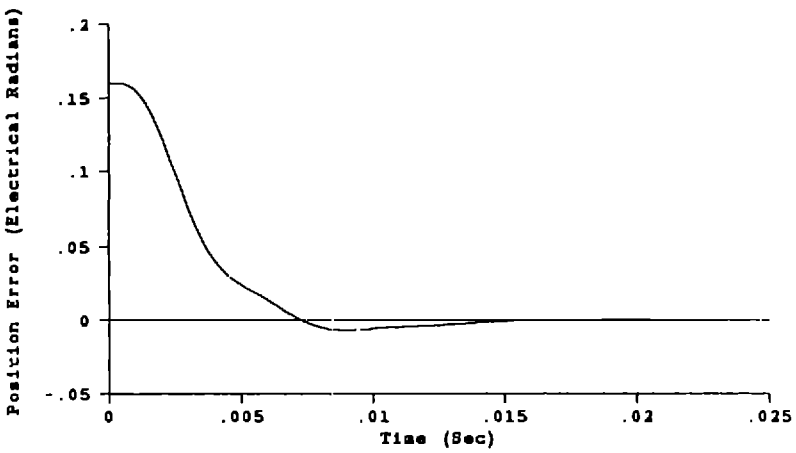


Figure 4.9: Estimated-innovation observer error response with the initial observer position 10% ahead of the 1000 rpm initial position. $\mathbf{G}_{\underline{\mathbf{i}}} = \begin{bmatrix} 200 & -100; & -100 & 200 \end{bmatrix}$, $\mathbf{G}_{\omega} = \begin{bmatrix} 100 & -300 \end{bmatrix}$.

As will be verified in the next chapter, this observer structure exhibits outstanding performance. For example, consider the familiar hunting transients. For the initial observer position 10% ahead of the 1000 rpm initial position, the resulting hunting transient is shown in Figure 4.9. Comparing these results with the hunting transient shown in Figures 4.1 reveals that the convergence rate has been improved by almost an order of magnitude. Additionally, the observer is stable for large perturbations.

4.3 Conclusions

Utilizing the *dq* transformation in an observer is useful because it provides a mechanism to effect corrective feedback in the mechanical subdynamics. The estimated-innovation observer (4.31)–(4.33) utilizes the *dq* transformation to obtain a corrective feedback that is dependent on error in both the mechanical and electrical variable estimates when in fact only the electrical variables are measured. The electromagnetic torque applied to the rotor is directly and linearly proportional to the quadrature current and thus it is possible to generate meaningful corrective feedback into the mechanical subdynamics using the error between the estimated-frame

quadrature current i_q and estimated quadrature current $\hat{i}_q$. The result is an observer that exhibits very good performance. It was shown in Section 4.2.3 that the position estimate produced by the estimated-innovation observer converges very quickly to the true position and that the observer is stable for large transients. In the next chapter, the performance of the estimated-innovation observer will be explored in depth.

Chapter 5

Simulated Performance

In Chapter 2, a model of a smooth-rotor permanent-magnet synchronous motor was developed. In Chapter 4, an observer based on this model was constructed to produce an accurate estimate of the motor dynamic state. This observer was referred to as the estimated-innovation observer. In this chapter, the mechanics of the simulation software are documented along with the results of simulations of the estimated-innovation observer.

5.1 Simulation Software

The motor models developed in Chapter 2 are nonlinear. The estimated-innovation observer is also nonlinear. Therefore, in order to simulate the motor and observer, software is needed to solve systems of simultaneous nonlinear differential equations. General purpose software packages such as SIMNON and MATRIX$_X$ are available for solving nonlinear systems. They tend to isolate the user from the details of the numerical simulation process. For pure numerical studies, this may be desirable. However, with the ultimate application of this research being the real-time microcomputer assembly-language implementation of the observer, the details of how the simulation is implemented are important. In a real-time implementation there is not enough time to step the observer using a sophisticated differential equation solver. In contrast, the simulation of the motor is necessary only in the non-real-time environment, and thus the solution of these differential equations can, and should, be performed with a sophisticated accurate algorithm.

The differential equation solving routine RKF45 was chosen to simulate the motor. RKF45 is based on the Runge-Kutta-Fehlberg fourth-fifth order algorithm

described by Fehlberg [59]. A robust computer implementation of this algorithm was written by Shampine and Watts [60] and its performance, compared to other algorithms, has been carefully documented by Shampine, et al. [61]. Forsythe, et al. [62], chose RKF45 as the best general-purpose differential equation solver available and provided a FORTRAN listing. A translation into Pascal may be found in Appendix A along with a suitable driver program.

RKF45 requires six evaluations of the state derivatives for each step. Therefore, a much simpler algorithm is needed to simulate the observer. The forward Euler formula is the simplest and requires only one evaluation of the state derivatives for each step. Its local error is proportional to the square of the step length h. The improved Euler formula (or Heun formula) requires two evaluations and its local error is proportional to the cube of h. The Euler formula is improved by using not just the state derivative at step n to project the state forward, but by projecting the state forward using an average of the state derivative at n and the state derivative at $n+1$. Since the state derivative at $n+1$ is unknown, it is estimated using the Euler formula. If the system of differential equations is

$$\frac{d\underline{x}}{dt} = \underline{f}\left(\underline{x}, t\right) \tag{5.1}$$

then the improved Euler formula is

$$\underline{x}_{n+1} = \underline{x}_n + \frac{1}{2}\left[\underline{f}\left(\underline{x}_n, t_n\right) + \underline{f}\left(\underline{x}_n + h\underline{f}\left(\underline{x}_n, t_n\right), t_n + h\right)\right] h\,. \tag{5.2}$$

Motor models are often stiff since they often contain fast electrical dynamics and slow mechanical dynamics. Low order, one-step methods, such as the improved Euler formula, may have difficulty converging when solving stiff systems. Occasionally, evidence will be seen in the simulations of the improved Euler formula having difficulty converging. See, for example, Section 5.2.1 and Figure 5.16.

RKF45 is a sophisticated, multi-step, differential equation solver. The error in a solution generated by RKF45 is a parameter specified by the user, and RKF45 will automatically take intermediate steps to keep the error within the specified limit. In contrast, the error using the improved Euler formula is proportional to the cube of the step-size h, where h is equivalent to the sampling period in a real-time system. The error can be arbitrarily reduced by reducing the step size.

The sampling frequency dictates how powerful the computational equipment must be in order to realize the observer in real-time. In typical current-controlled

synchronous motors, the voltage is pulse-width modulated at 8 *k*Hz in a fast inner current-control loop. Therefore, properly sampling the stator voltages at this level of detail for observer purposes would require a prohibitively fast sampling rate of at least 16 *k*Hz. The closed-loop bandwidth of the outer speed-control loop is approximately several hundred hertz. The much faster inner current-control loop, in combination with the high bus voltage, guarantees that the current commands from the speed-control loop will be followed in the motor. Looked at from another point of view, this means that the currents in motor relevant to observer performance are the response to a fundamental voltage waveform of only several hundred hertz.[1] Therefore, it is not necessary to measure the fast switching voltages, but rather their much slower averages. If roll-off bandwidth for anti-aliasing filters is included, the sampling bandwidth in a system that uses 8 *k*Hz switching should be about 2 *k*Hz. Therefore, an appropriate sampling frequency is 5 *k*Hz. The small stepsize of 200 μs ensures that the improved Euler formula will yield accurate results.

To simulate the performance of the observer off-line, it is necessary to simulate both the motor and the observer. A simple diagram of the structure of the simulation is shown in Figure 5.1. The observer is first stepped forward using the improved Euler formula and the voltages and currents measured at the end of the previous step. Second, the motor is stepped forward and the voltages and currents at the end of the step are stored. The observer is stepped forward first since it uses the specified motor initial conditions to take its first step. A more detailed account of the simulation structure is given in Figure 5.2. The driver program to run the simulation in Figure 5.2 is necessarily more complex than the driver program given in Appendix A. A complete simulation environment was written to produce the simulation results in this thesis. RKF45 is embedded in this simulation environment. The simulation environment includes the capability for interactively setting simulation parameters, plotting data, loading data from disk, and saving data to disk in Lotus 123 or PC-MATLAB format. Excerpts of this large program which implement the structure in Figure 5.2 are included in Appendix B.

The motor is simulated is the stator frame. Therefore, the electrical variables produced are precisely those that would be measured in a real system. The motor

[1]The fundamental voltage waveform cannot be applied directly to the motor windings because only a finite number of voltage vector positions are available from the inverter. By rapidly switching between vector positions, any intermediate position may be realized as the time average of the two positions.

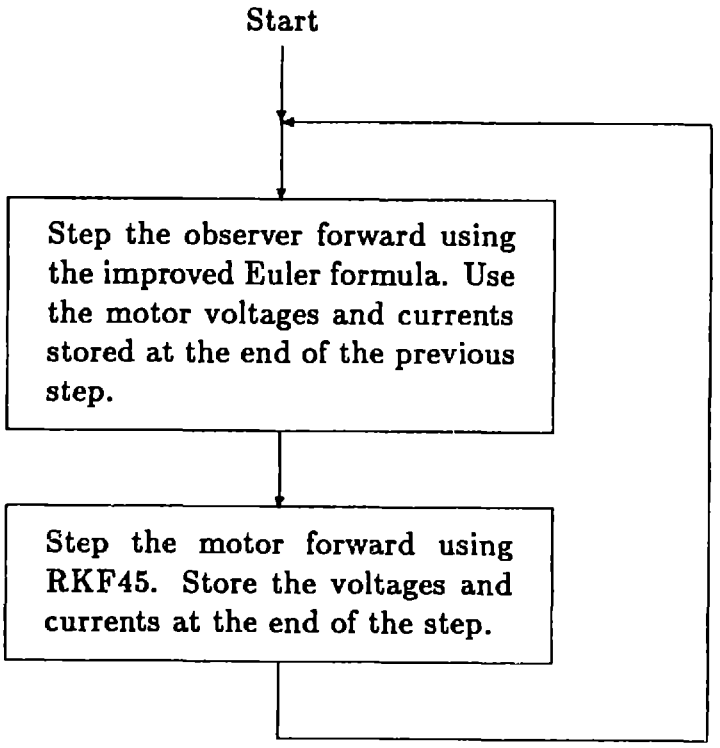


Figure 5.1: Basic structure of motor/observer simulation.

state vector is

$$\underline{x} = \begin{bmatrix} i_\alpha & i_\beta & \omega & \theta \end{bmatrix}^T . \tag{5.3}$$

The observer is simulated in an estimated dq frame and its state vector is

$$\hat{\underline{x}} = \begin{bmatrix} \hat{i}_d & \hat{i}_q & \hat{\omega} & \hat{\theta} \end{bmatrix}^T . \tag{5.4}$$

5.2 Observer Performance

Analytically predicting the large signal behavior of the estimated-innovation observer (4.31)–(4.33) is difficult. Therefore, simulations were performed to verify its large signal performance. The simulations may be broken into the following three major groups:

1. simulations where the measurements and physical parameters are exact,
2. simulations where the measurements are exact but the physical parameters are not, and

{Initialization}

1. Initialize miscellaneous variables.
2. If simulating the motor then
   - Clear all lists: motor state, observer state, voltages, and currents.
   - Create first list entries and fill with initial conditions.
3. If simulating the observer then
   - Clear observer state list. (Leave voltage and current lists intact.)
   - If voltage and currents lists not present then
     - set interrupted true
   - else
     - create first observer list entry and fill with initial conditions.

{Begin Simulation}

4. While more steps and no errors and not interrupted
   - If simulating the observer then
     - Step the observer forward using the improved Euler formula along with the voltage and current lists entries.
     - Save resulting state in a new list entry.
     - If not simulating the motor then move to the next voltage and current entries.
   - If simulating the motor then
     - Update the input voltage frequency. Change as necessary.
     - Step the motor forward using RKF45.
     - Store the results (including new voltages and currents) in new list entries.
     - If RKF45 reported an error then set error true.
   - If not enough memory available for the next step, then set error true.
   - If a key has been pressed, then set interrupted true.

Figure 5.2: Detailed structure of motor/observer simulation.

Physical Parameters

N	=	3	B	=	3.7×10^{-3} N·m·s·rad^{-1}
R	=	0.39 Ω	H	=	35.5×10^{-3} kg · m^2
L	=	0.444 mH	C	=	0.583 N·m
K	=	0.1105 V·s	τ	=	1.6 N·m

Observer Gains

$$\mathbf{G} = \left[\frac{\mathbf{G_I}}{\mathbf{G_\omega}} \right] = \left[\begin{array}{cc} 200 & -100 \\ -100 & 200 \\ \hline 100 & -300 \end{array} \right]$$

Simulation Parameters

Relative Error	=	0.00001	Initial Time	=	0.0
Absolute Error	=	0.00001	Step Size	=	200 μs
RKF45 Mode	=	1			

Figure 5.3: Parameters used in all simulations unless specifically noted otherwise.

3. simulations where the physical parameters are exact but the measurements are corrupted with noise.

Within these major groupings, several special cases are considered. Unless specifically stated otherwise, the parameters listed in Figure 5.3 were used in all the simulations. The physical parameters were obtained from the real motor system used in experiments described later in this thesis. The methods used to measure the parameters will be described in Chapter 8. In each of the simulations, the observer state estimates are compared to the motor states; the motor states are subtracted from the estimated states to generate estimation errors. These errors are the focus of most of the discussion in this chapter.

In all the simulations, the motor was controlled open-loop and the position of the rotor, real or estimated, was not used for control. Thus, only the performance

of the observer in isolation was studied. For closed-loop control using an observer, the controller would obtain position information from the observer and the two become coupled. In general, the performance of this system is much more complex. In the linear case, however, the separation theorem guarantees that the dynamics of the controlled system and the observer separate and thus analysis of the observer may proceed independent of the controller. It will be shown in Chapter 6 that the linearized estimated-innovation observer error behaves much like the nonlinear observer error for large perturbations. Therefore, in this thesis, an appeal is made to the separation theorem of a linear system to motivate and justify the study of the isolated observer behavior presented in the following sections.

5.2.1 Tests with Known Parameters

Step to Constant Speed from Rest

Assume the motor and its controller are running at a constant speed. One interesting question is if the observer can produce a position estimate that rapidly converges to the true shaft position when the observer is started from rest, that is with all initial conditions set to zero. This is interesting because the observer can always be started from rest, and because it represents a large transient. This subsection considers such tests.

The motor simulation is started using initial conditions appropriate for the desired steady-speed operation. The equations used to compute this steady-state are obtained from the dq model. Assuming the stator voltage vector is at $\theta = 0$ when $t = 0$, the values of the currents and the position of the shaft at $t = 0$ are

$$i_{do} = \frac{-KL(N\omega)^2 + \sqrt{\sigma V^2 - (\sigma i_{qo} + RKN\omega)^2}}{\sigma} \tag{5.5}$$

$$i_{qo} = \frac{1}{KN}(B\omega + C + \tau) \tag{5.6}$$

$$\theta_o = -\frac{1}{N}\left[\cos^{-1}\left(\frac{-\sigma i_{qo} - RKN\omega}{\sqrt{\sigma}V}\right) - \tan^{-1}\left(\frac{R}{LN\omega}\right)\right] \tag{5.7}$$

where

$$\sigma = R^2 + (LN\omega)^2 \tag{5.8}$$

and V is the amplitude of the two-phase voltage. The initial stator-frame currents are computed using the inverse Park transform

$$\begin{bmatrix} i_{\alpha o} \\ i_{\beta o} \end{bmatrix} = \exp\left(\mathbf{J}N\theta_o\right) \begin{bmatrix} i_{do} \\ i_{qo} \end{bmatrix}. \tag{5.9}$$

The estimated-innovation observer error response to the motor running at the constant speed of 100 rpm is shown in Figures 5.4–5.7. The error was obtained by subtracting the simulated motor response from the simulated observer response. The error responses are shown plotted against true shaft position as provided by the simulation of the motor. The observer position error response to the motor running at the constant speeds of 300 rpm, 1000 rpm, and 3000 rpm are shown in Figures 5.8–5.10, respectively. The position error response appears to be that of a linear system, namely a complex exponential where the damping varies inversely with speed. If this experiment is carried out at 4000 rpm, the error response neither converges nor diverges, but settles into a steady-state oscillation with no damping as shown in Figure 5.11. Performing this experiment at still higher speeds, the error responses diverge. In Chapter 6, a root locus with respect to speed is formed for the linearized error system and the positions of the poles accurately predict that the observer becomes unstable at high speeds.

As was noted in Chapter 4, the responses in Figures 5.7–5.10 are very fast, and the observer is stable for very large transients. Consider the response shown in Figure 5.10. The observer is locking on, from rest, to a motor that is turning at 3000 rpm. The fact that the observer doesn't even slip a pole pair is somewhat an artifact of this particular experiment. The initial position of the motor is -1.61 erad while the initial position of the observer is 0 erad and therefore the observer has some time to converge before the rotor sweeps past the zero position. It will be seen later that if the initial observer position is behind the true position, the observer will sometimes slip a pole pair.

The simulations indicate that the observer converges in approximately one electrical cycle[2] for midrange speeds where the damping is critical. The response is more sluggish at lower speeds. This is not surprising as there is less speed voltage in the current measurements for the observer to utilize. By increasing the innova-

[2]Recall from Chapter 2 that there are N electrical cycles per revolution of the motor shaft where N is the number of pole pairs on the rotor. The electrical frequency is defined as $\omega_e = N\omega$ and has units of electrical radians per second (erad/s). Therefore, the period of one electrical cycle is $T_e = 2\pi/(N\omega)$.

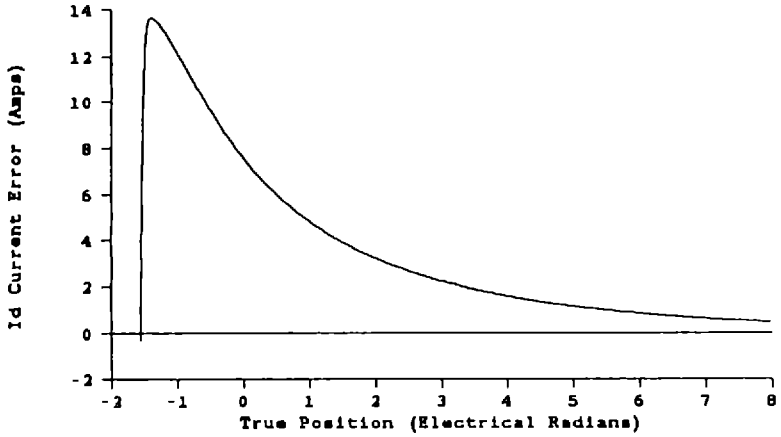


Figure 5.4: Observer direct current error response from rest for the motor running at a constant 100 rpm.

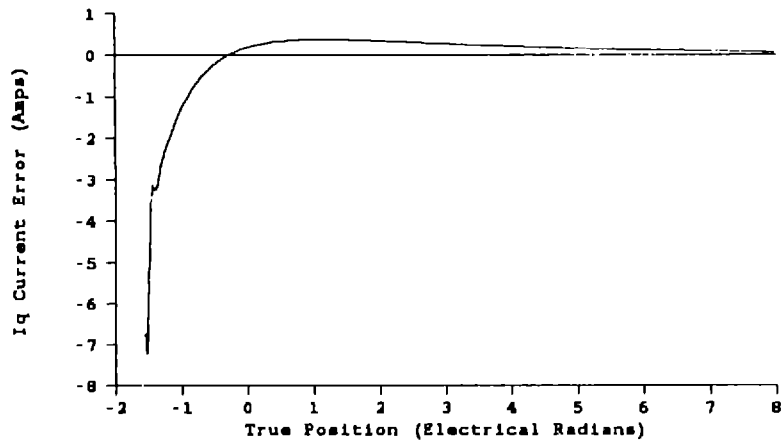


Figure 5.5: Observer quadrature current error response from rest for the motor running at a constant 100 rpm.

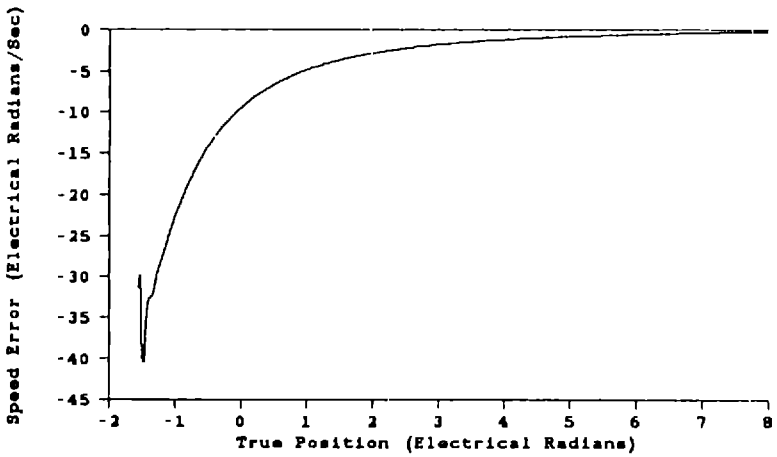


Figure 5.6: Observer speed error response from rest for the motor running at a constant 100 rpm.

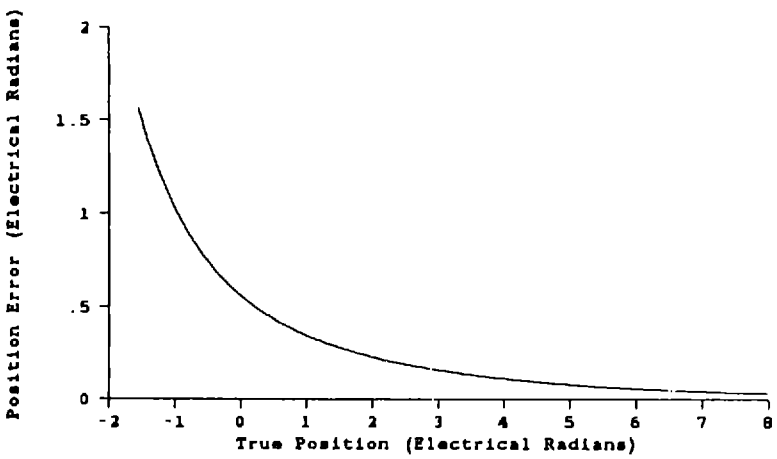


Figure 5.7: Observer position error response from rest for the motor running at a constant 100 rpm.

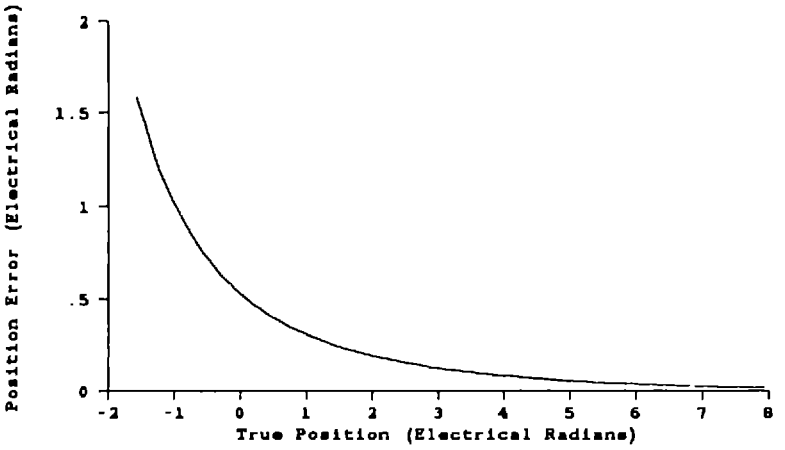


Figure 5.8: Observer position error response from rest for the motor running at a constant 300 rpm.

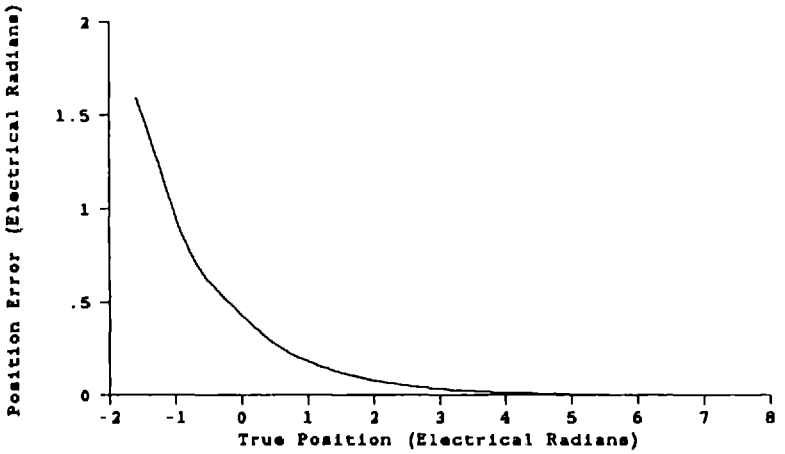


Figure 5.9: Observer position error response from rest for the motor running at a constant 1000 rpm.

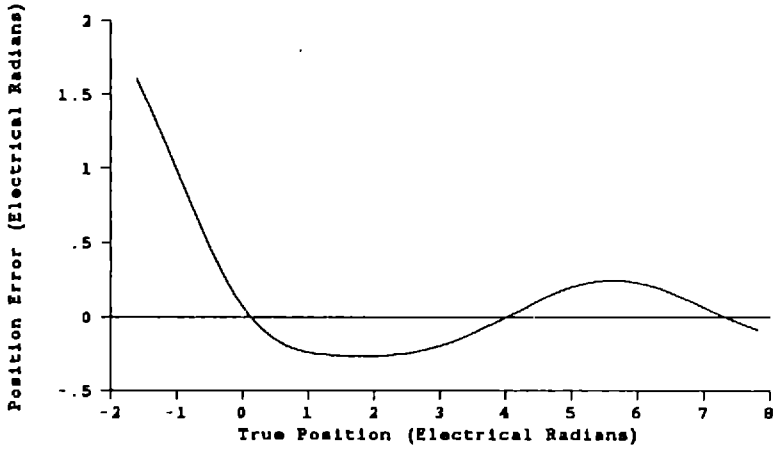


Figure 5.10: Observer position error response from rest for the motor running at a constant 3000 rpm.

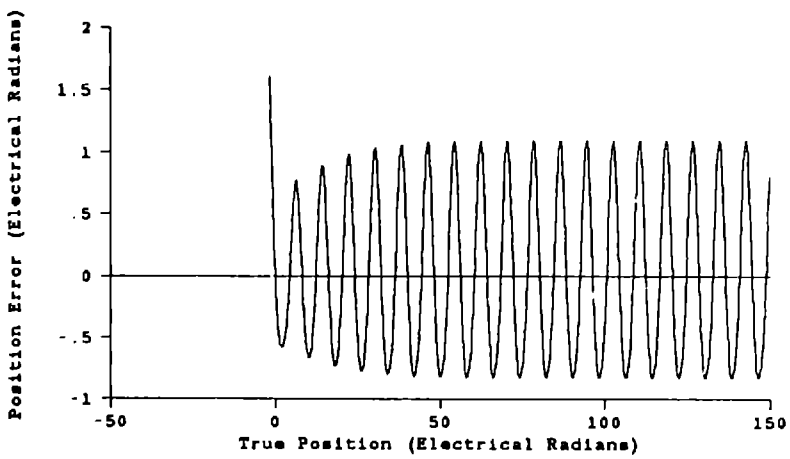


Figure 5.11: Observer position error response from rest for the motor running at a constant 4000 rpm.

tion gains, the convergence rate of the observer at slow speeds can be improved. However, the tradeoff is that at high speeds, where the speed voltage is large, the observer may become unstable. An improvement to the estimated-innovation observer might be the use of gain scheduling dependent on rotor speed.

Ramp to Constant Speed from Rest

It is not possible to determine the position of the rotor of a smooth-rotor permanent-magnet synchronous motor when the rotor is not rotating because the inductance profile of the rotor is independent of position. Therefore, during the initial startup of the motor, the initial estimated position may be substantially incorrect. Also at startup, there is a time when the motor is operating at slow speed where the observer is sluggish. Therefore, the second type of transient considered is to determine if the observer can follow a motor starting from rest. The assumption is initially made that a calibration routine is executed when the motor is first powered up where the phases are excited open loop by the controller and the observer is allowed to converge to an accurate estimate of the position. Therefore, on startup it is assumed that the state of the observer is identical to the state of the motor.

A ramping-frequency voltage is applied open-loop to the motor, and therefore the motor response is oscillatory. For example, if the voltage frequency is ramped from 0 to 31.42 electrical radians per second (100 rpm) at a rate of 1500 electrical radians per second per second (4775 rpm/s), the motor responds as shown in Figure 5.12. The large hunting transient that occurs when the frequency of the voltage stops ramping is indicative of the light coupling between the electrical and mechanical dynamics. The estimated-innovation observer error response for this ramp is shown in Figures 5.13–5.16. Note in Figures 5.14 and 5.16 how the quadrature current error is positive when the position error is negative. Since torque is proportional to the quadrature current, $\hat{\theta}$ is accelerated to catch θ when $\hat{\theta}$ is behind θ. The motor speed profiles for ramps to 300 rpm, 1000 rpm, and 3000 rpm and the corresponding observer position error responses are shown in Figures (5.17) and (5.18), (5.19) and (5.20), and (5.21) and (5.22), respectively. Note that the ramp to 3000 rpm does not have time to complete.

In each error response, a small high-speed oscillation and subsequent error in the observer estimate exists even though the initial conditions of the motor are precisely known. This is a result of the simple improved Euler formula used to implement

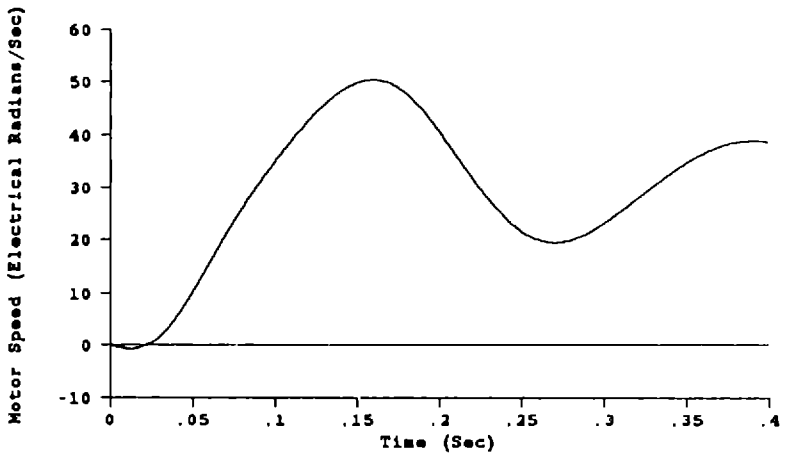


Figure 5.12: Speed of open-loop motor ramping to 100 rpm (31.4 erad/s) at 1500 erad/s^2. All initial conditions are zero.

the observer. The system is somewhat stiff and the algorithm initially has some difficulty converging.

Two factors affect the convergence of the observer. Note that if the ramp ends at a higher frequency, there is less error. As was seen earlier, the observer responds faster when it is driven at higher frequencies. Therefore, when the ramp ends at a higher frequency, the observer is able to converge faster. The second factor is the magnitude of the stator voltage is greater for the ramps to higher speeds. The observer is driven harder by the higher voltage and thus the stiffness of the equations is evident in the larger initial high-speed oscillation when ramping to a higher speed. Note that even though the error jumps to larger values initially, it is quickly reduced as the motor ramps up to speed because the higher voltage results in larger current errors and larger innovation. The net result is improved observer performance.

Next, the assumption that a calibration routine is executed at motor startup is relaxed. Consider the case of the motor ramping to 1000 rpm at 1500 erad/s from zero initial conditions while the observer is started from various initial rotor positions with all other observer states zero. Simulation results are shown in Figure 5.23. Since the electrical cycles are identical, the observer may slip an electrical cycle if the initial observer rotor position is sufficiently behind the true initial rotor

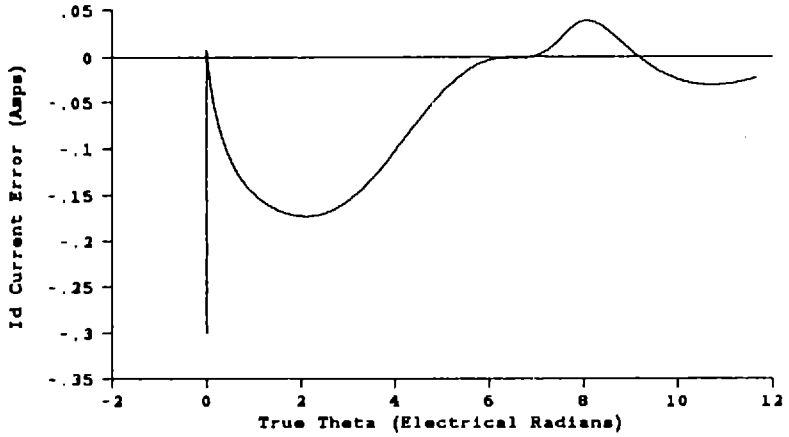


Figure 5.13: Observer direct current error response for the motor ramping to 100 rpm with initial rotor position known. All initial conditions are zero.

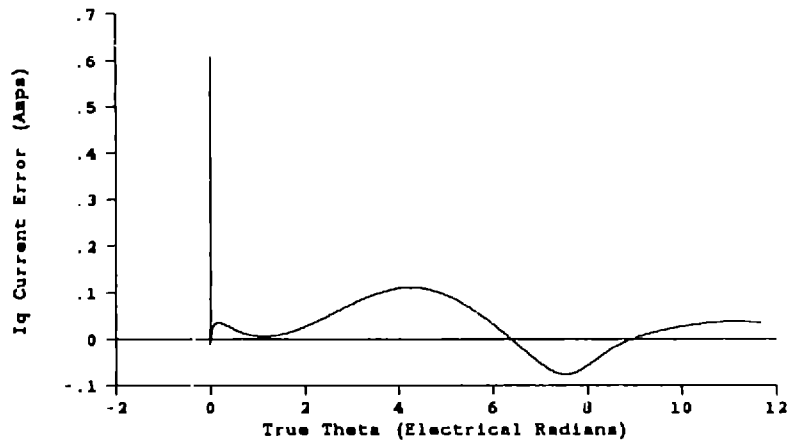


Figure 5.14: Observer quadrature current error response for the motor ramping to 100 rpm with initial rotor position known. All initial conditions are zero.

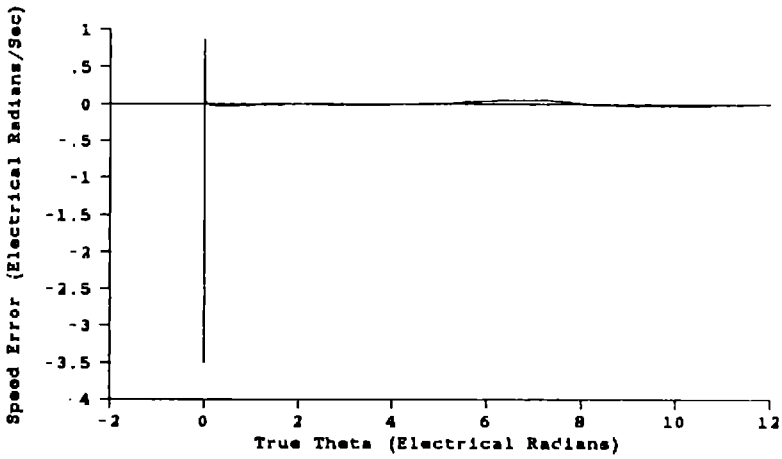


Figure 5.15: Observer speed error response for the motor ramping to 100 rpm with initial rotor position known. All initial conditions are zero.

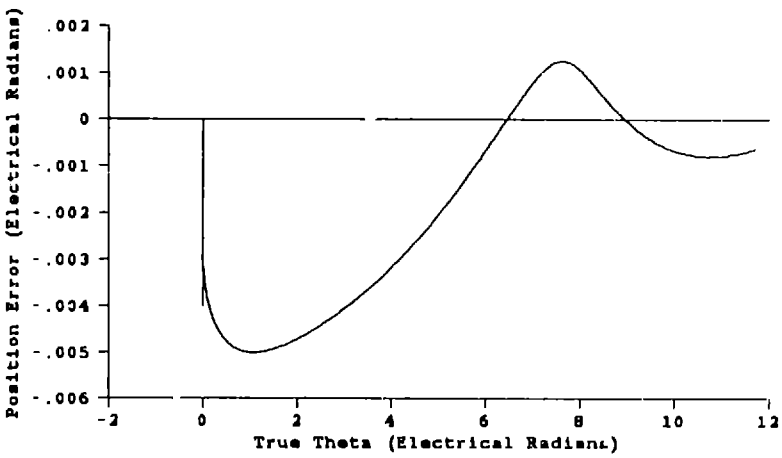


Figure 5.16: Observer position error response for the motor ramping to 100 rpm with initial rotor position known. All initial conditions are zero.

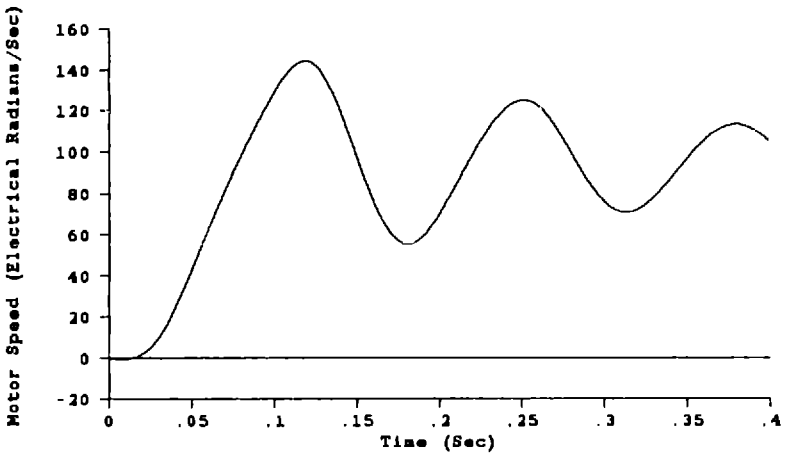


Figure 5.17: Speed of open-loop motor ramping to 300 rpm (94.3 erad/s) at 1500 erad/s^2. All initial conditions are zero.

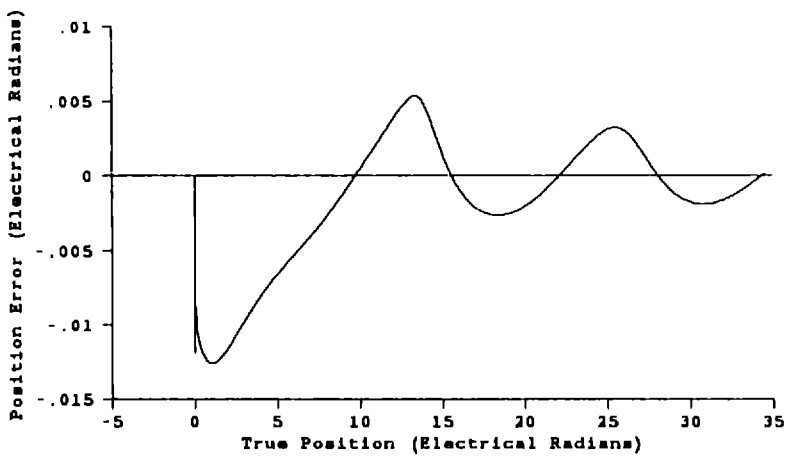


Figure 5.18: Observer position error response for the motor ramping to 300 rpm with initial rotor position known. All initial conditions are zero.

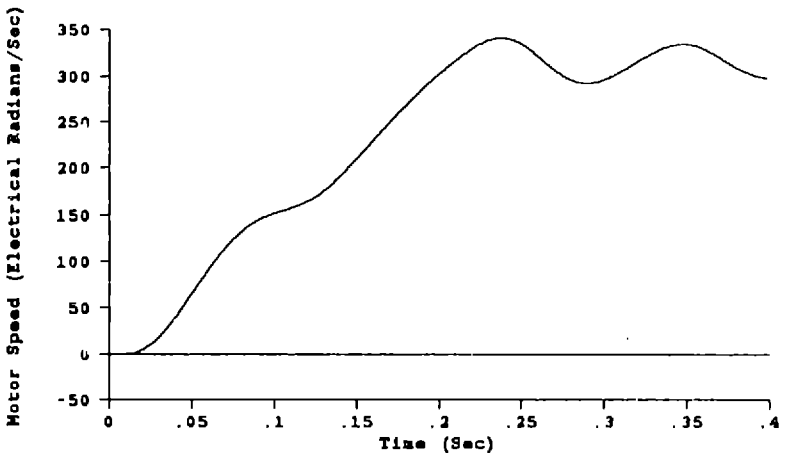


Figure 5.19: Speed of open-loop motor ramping to 1000 rpm (314.2 erad/s) at 1500 erad/s^2. All initial conditions are zero.

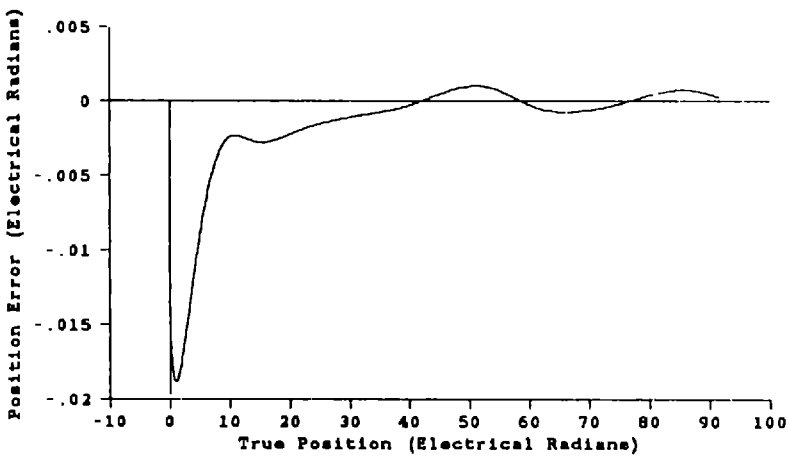


Figure 5.20: Observer position error response for the motor ramping to 1000 rpm with initial rotor position known. All initial conditions are zero.

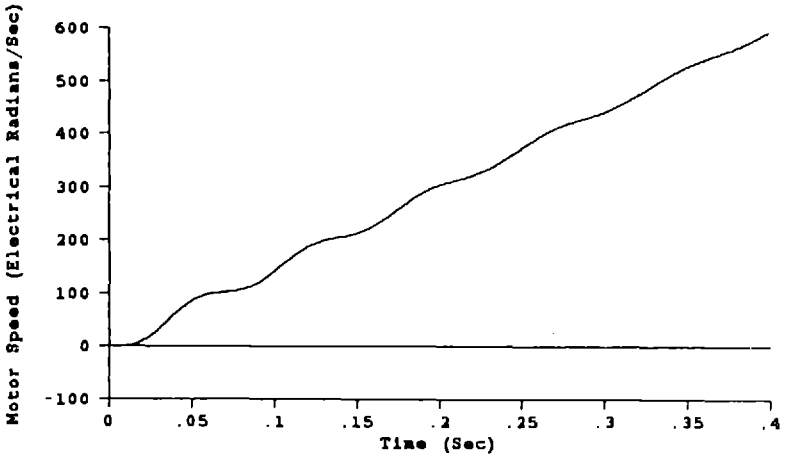


Figure 5.21: Speed of open-loop motor ramping to 3000 rpm (942.5 erad/s) at 1500 erad/s^2. All initial conditions are zero.

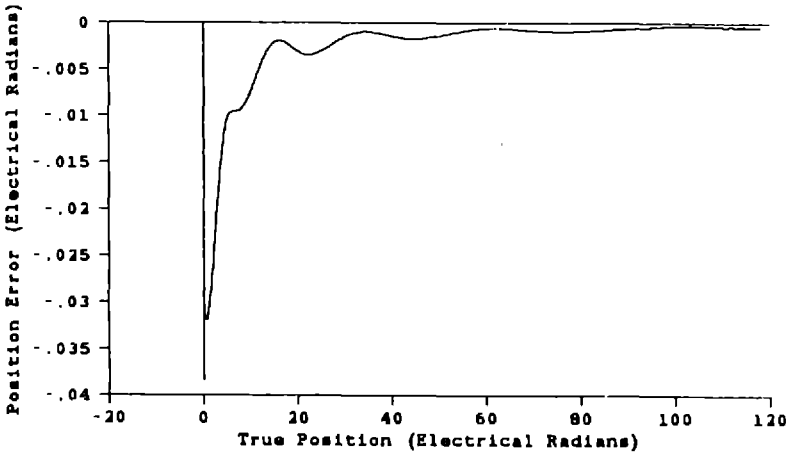


Figure 5.22: Observer position error response for the motor ramping to 3000 rpm with initial rotor position known. All initial conditions are zero.

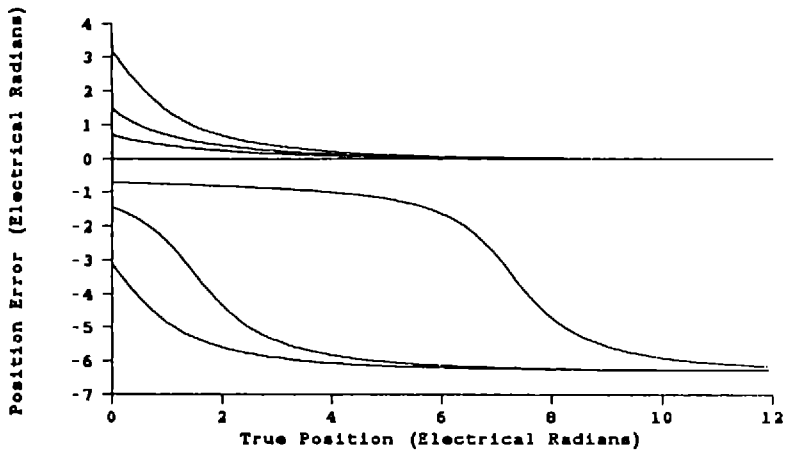


Figure 5.23: Observer position error response for the motor ramping to 1000 rpm with the initial position unknown. All other initial conditions are zero.

position. For many applications this behavior is acceptable. For servo applications it might not be. In addition, the observer may require more time to settle when it slips a pole pair.

Step Speed Change

Assume that the motor is running at constant speed and the observer is tracking the motor position exactly. The motor then executes a step speed change that the observer must track. The worst case is if the motor executes an infinitely fast step; the motor assumes the new speed instantaneously with no oscillations and the currents necessary to run the motor at the new speed are obtained instantaneously. An infinite-bandwidth controller would be needed to execute a perfect step along with a voltage source capable of delivering impulses and so, not surprisingly, a perfect step is not realizable.

The response of the observer to a step speed change illustrates the power of the estimated-innovation observer: the stator currents in the motor already exhibit the higher speed (the speed after the step) and thus, when transformed into the rotor frame using the more slowly increasing observer position estimate, generate a significant additional torque term in the speed equation *independent of the observer*

currents. Therefore, excursions in the driving angle of the voltage and innovation are not necessary and thus the resulting position error is very small.

To simulate the observer transient as a result of a perfect motor step, the observer initial conditions are set appropriate for the speed before the step and the motor initial conditions are set appropriate for the speed after the step. Initial conditions for the speed before the step are computed using (5.5)–(5.7), where it is assumed that the position of the voltage vector is at $\theta = 0$ when $t = 0$. Initial conditions for the speed after the step are computed using the same formulas except that the resulting initial shaft position, which is precisely the torque angle, will be different. Since there is never a discontinuity in the shaft position, a lead angle must be added to the voltage that is the difference between the new torque angle and the old torque angle. Therefore, the motor and observer will start from the same shaft position but at different speeds.

Simulations were run for steps of +10% from the base speeds of 100, 300, 1000, and 3000 rpm. The respective results are shown in Figures 5.24–5.27. As the base speed increases, the damping of the error response of the observer decreases until, at 3000 rpm, the response is highly oscillatory. This behavior is similar to that obtained for an observer step to constant speed from rest. At very high speeds (above 4000 rpm) the observer looses synchronism with the motor and becomes unstable. Stability at higher speeds may be obtained by using a different set of gains.

Conclusions

The estimated-innovation observer is able to converge to zero error in approximately one electrical cycle even for very large transients. The damping in the observer is inversely proportional to speed which means that at low speeds the observer is sluggish while at high speeds it is oscillatory and possibly unstable. Observer response is optimum when the damping is near critical. The gains should be chosen so that critical damping occurs in the range of speeds the motor is expected to be operated. Another possibility is to use gain-scheduling dependent on speed. This idea is discussed further in Chapter 6.

The observer is unable to converge if the motor is not turning and thus determining the starting position of the motor is a problem. If the observer has been previously allowed to converge, the stored final position may be used to start the

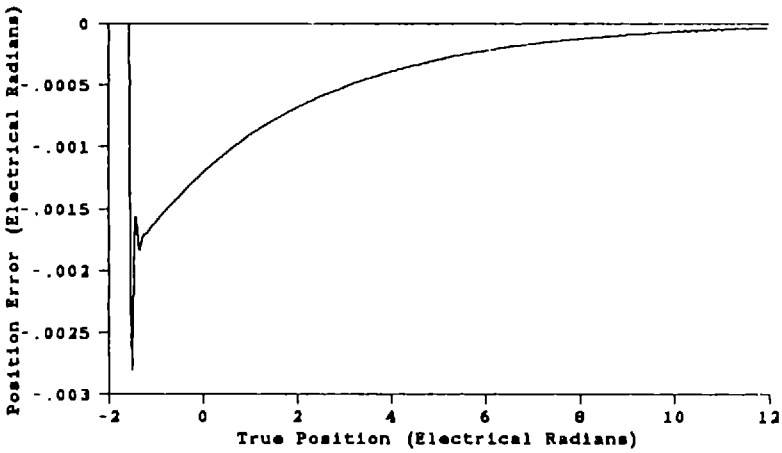


Figure 5.24: Observer position error response for the motor stepping from 100 to 110 rpm.

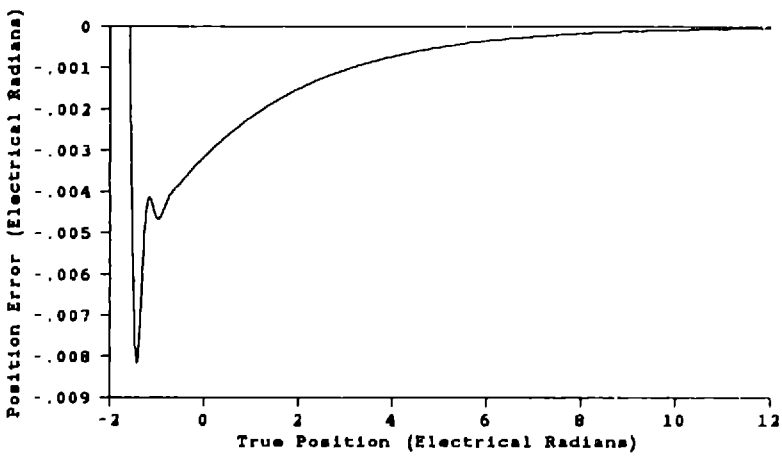


Figure 5.25: Observer position error response for the motor stepping from 300 to 330 rpm.

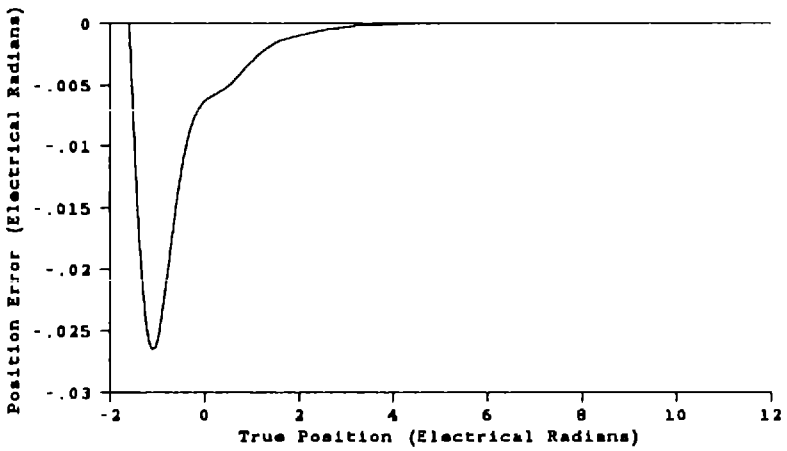


Figure 5.26: Observer position error response for the motor stepping from 1000 to 1100 rpm.

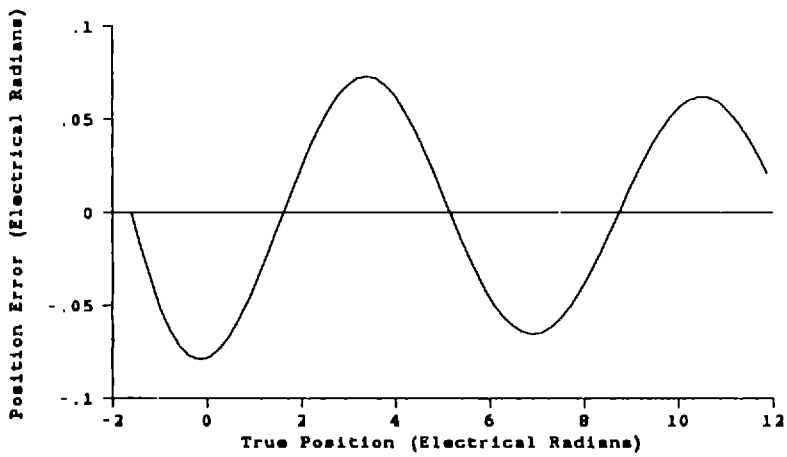


Figure 5.27: Observer position error response for the motor stepping from 3000 to 3300 rpm.

motor. Simulations indicate that the observer will accurately track the motor as it ramps up to speed from rest. If the observer has not been allowed to converge, the motor may be started from rest using open-loop control and the observer will converge to zero error with any initial shaft position. It may, however, slip a pole pair if the true position of the shaft is ahead of the initial guess. The observer converges after slipping a pole pair since it is unable to distinguish between electrical cycles. Therefore, when the motor is symmetrical, the observer may not converge to the nearest electrical cycle and thus it is impossible to provide a home position signal.

5.2.2 Tests with Unknown Parameters

An observer is based on a model which attempts to reproduce the output of the system being observed. Identity observers utilize models derived from the physics of the system such that the states of the observer have a one-to-one correspondence with the states in the physical system. The parameters of the model are measured, and if they are not accurately known, the model will not accurately reproduce the behavior of the system. As a result, the performance of the observer will suffer.

The model used by the estimated-innovation observer is the rotor-frame model of a two-phase motor (2.72)–(2.74). The parameters in this model are:

N	number of magnet pole pairs on the rotor;
R	two-phase stator resistance;
L	two-phase stator inductance;
K	rotor permanent-magnet constant;
B	viscous damping coefficient of the rotor and load;
H	inertia of the rotor and load;
C	coulomb friction of the rotor and load;
τ	load torque.

The constant N will not vary as it is determined by the physical structure of the motor. The stator inductance L is also primarily a function of the physical structure of the motor and therefore should not change in time. The viscous damping coefficient B and coulomb friction coefficient C are somewhat dependent on temperature but will be assumed to be constant since they contribute only small torque terms in the speed equation. The other parameters, R, K, H, and τ are investigated below. To investigate the effects of parameter errors on the performance of the observer, simulations of the motor stepping from 1000 to 1100 rpm were run with the parameters perturbed. The motor was assumed to be capable of executing a perfect step

as in Section 5.2.1.

Stator Resistance

As the temperature of the motor increases, the resistance of the stator windings will increase. As a result, the resistance parameter in the observer will be low. The observer response when the observer parameter R is 10% low is shown in Figure 5.28. The position error takes on a constant positive offset. This result may be explained by considering the observer equations (4.31)–(4.33). As long as the observer is in synchronism with the motor, the rotor speed ω in (4.32) must converge to zero average error. Therefore, the sum of the innovation and electromagnetic torque must equal the electromagnetic torque that was present before the resistance changed, and thus the currents must change from the unperturbed case. Since the magnitude of the driving voltage is constant, the only way the currents can be altered is by changing the angle at which the voltage and innovation drive the electrical subdynamics. Therefore, a steady-state error in the position estimate develops.

The resistance of the stator windings is typically very small, a few ohms or less. Measuring such small resistances is difficult, particularly if there are connectors in the paths of the windings. Therefore, the possibility exists of the resistance parameter in the observer being high. The observer response when the resistance is 10% high is shown in Figure 5.29. The position error takes on a constant negative offset. In both this case and the previous case of a low resistance parameter, the sign of the constant position error is dependent on the signs of the innovation gains used.

Magnet Constant

The magnetic field strength of most permanent-magnets is inversely proportional to temperature. As the temperature in a permanent-magnet motor goes up, the magnets become weaker, necessitating the use of more current to maintain torque. The additional current exacerbates the heating problem. As the magnet constant for the motor drops, the observer parameter K will become high. The observer response when the magnet constant K is 10% high is shown in Figure 5.30. If, due to mismeasurement, the magnet constant is 10% low, then the observer responds as shown in Figure 5.31. It is apparent that an error in the magnet constant causes

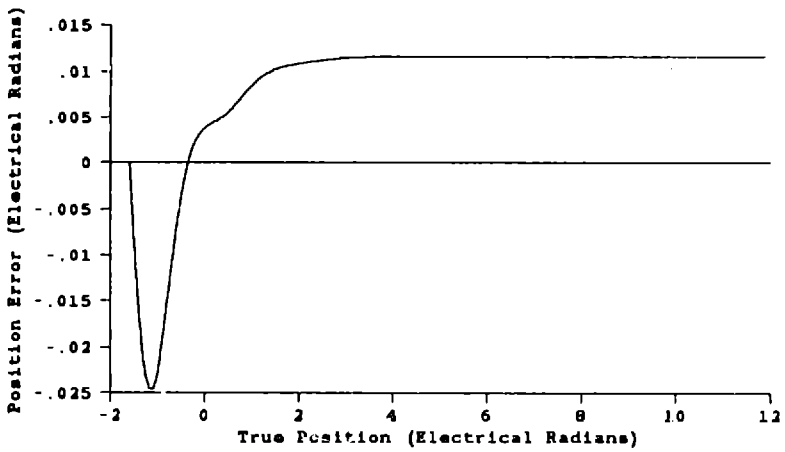


Figure 5.28: Observer position error response for the motor stepping from 1000 to 1100 rpm when the observer stator resistance R is 10% low.

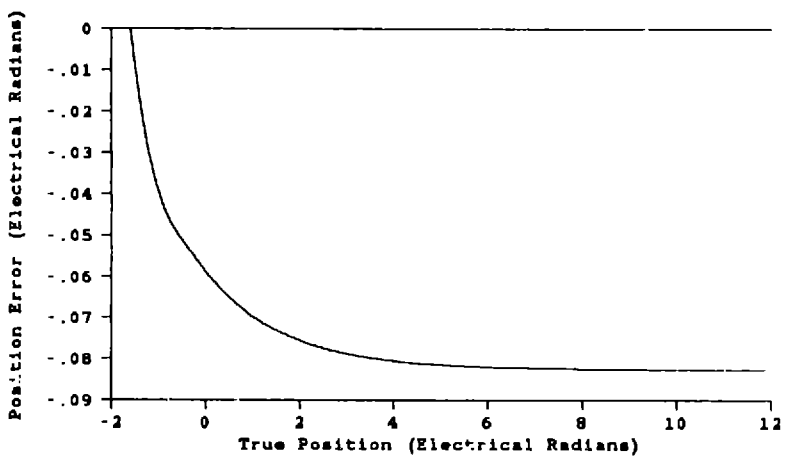


Figure 5.29: Observer position error response for the motor stepping from 1000 to 1100 rpm when the observer stator resistance R is 10% high.

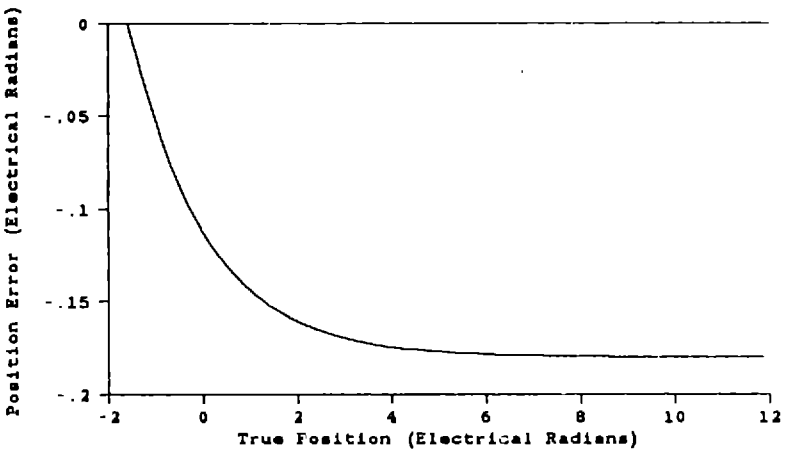


Figure 5.30: Observer position error response for the motor stepping from 1000 to 1100 rpm when the observer magnet constant K is 10% high.

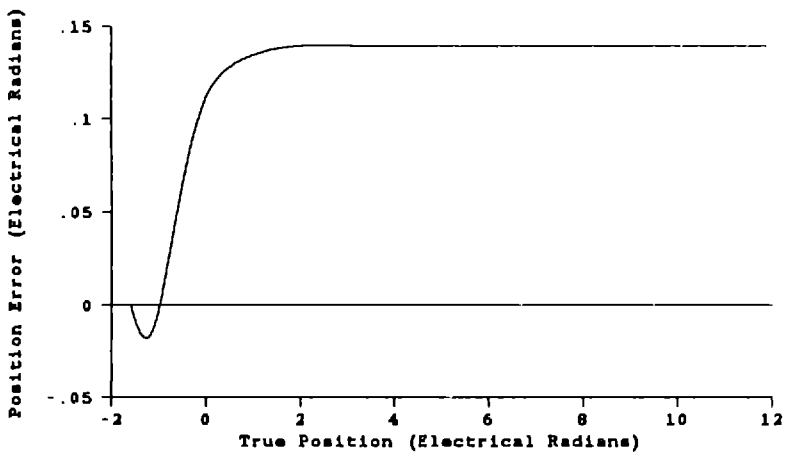


Figure 5.31: Observer position error response for the motor stepping from 1000 to 1100 rpm when the observer magnet constant K is 10% low.

a large position error. This is because the magnet constant appears in both the electrical and mechanical subdynamics. In particular, as the magnets weaken, the speed voltage is reduced. The currents must therefore change to compensate for both the lost electromagnetic torque and the reduced speed voltage. Changing the currents requires a change in the angle at which the voltage and innovation drive the current subdynamics.

Load Inertia

The inertia of the load connected to the shaft of the motor could change drastically. However, in the motor speed equation (2.73), the inertia H simply scales each term, and therefore the steady-state error of the observer should not be changed by an incorrect value for the parameter H. As the load inertia increases, the torques contributing to the acceleration of the rotor are scaled down. As the load inertia decreases, the torques are scaled up.

Consider the observer position error response shown in Figure 5.32 where the observer inertia is high by 500% . Since the magnitude of the torque terms are scaled down, the currents in the electromagnetic torque and innovation terms must show large excursions in order to respond to the motor step. Large excursions of the current will only occur if the driving angle of the voltage and innovation changes and thus there will be large excursions of the position error as seen in Figure 5.32.

When the observer inertia is low, the magnitude of the torque terms are scaled up. For example, when the observer inertia is low by 50% , the resulting position error response, shown in Figure 5.33, is very fast. Therefore, a small value of H should be used if the true value is unknown.

Load Torque

The load torque parameter τ is most likely not constant or even known. In many applications, the load torque on the motor ranges from zero to rated load. It is most important to consider the effect of observer load torque errors on observer performance.

It can be predicted that the observer response should be relatively insensitive to load torque errors. When a step response is required of the observer, the additional torque needed to increase the speed of the shaft is generated by the electromagnetic and innovation terms, independent of the load. A steady-state error will exist,

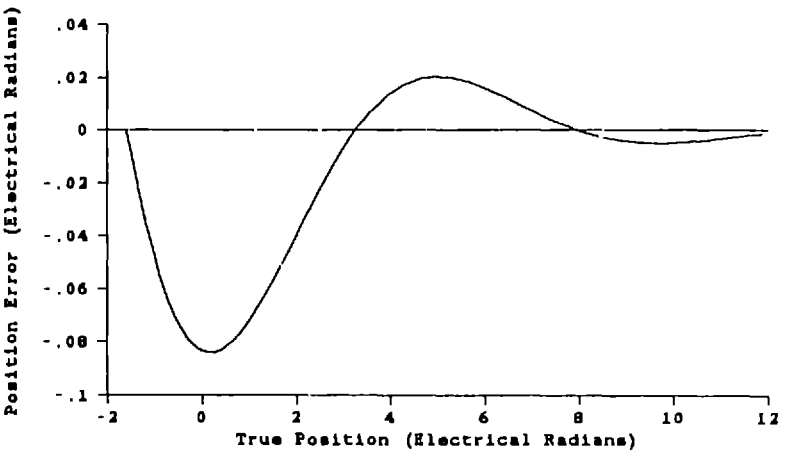


Figure 5.32: Observer position error response for the motor stepping from 1000 to 1100 rpm when the observer shaft inertia H is 500% high.

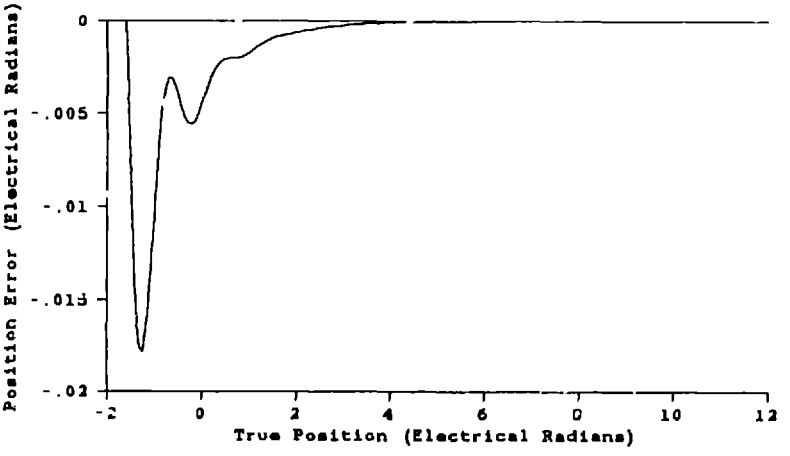


Figure 5.33: Observer position error response for the motor stepping from 1000 to 1100 rpm when the observer shaft inertia H is 50% low.

however, because for constant speed rotation, the incorrect observer load torque must be balanced by a steady-state error in the quadrature current, which will introduce additional torque in the electromagnetic and innovation terms. This current error is generated by adding an error to the driving angle of the voltage and innovation in the electrical subdynamics which shows up a steady-state error in the position estimate.

The observer position error response when the observer load torque is high by 100% is shown in Figure 5.34. The position error response when the observer load torque is low by 100% (that is $\tau = 0$) is shown in Figure 5.35. The responses in Figures 5.26 (where the load torque is correct), 5.34, and 5.35 are remarkably similar The notable difference is that when the observer load torque is incorrect there is a small steady-state position error. The fact that this error is small indicates that only a small change in the driving voltage and innovation is needed to provide the additional torque in the speed equation. Only a small error in the position is needed because the stator phase inductance is small. Currents ramp up quickly in the stator and it is the delicate balance between the speed voltage and the driving voltage that determines the magnitude of the currents. A slight perturbation of the speed voltage causes large changes in the currents, and thus only a small error in the position is needed. This phenomenon has important ramifications in a real-world implementation and is discussed further in Chapter 9.

Conclusions

The most critical observer parameter is the magnet constant K. If it is incorrect, steady-state observer performance deteriorates dramatically. Fortunately, it is relatively easy to accurately measure K by measuring the open-circuit speed voltage on the stator of the motor when it is driven by another motor. However, some types of magnets found in permanent-magnet synchronous motors, notably neodymium-iron-boron magnets, are sensitive to temperature and grow significantly weaker with increasing temperature. The magnet constant K may change 10% over the operating range of the motor as considered in Figures 5.30 and 5.31. One solution to this problem is to use magnet material that is not sensitive to temperature. Another solution is to adapt, on-line, to the changing magnet constant.

The two parameters most likely to have a wide variation over the operating range of the motor are the shaft inertia H and the load torque τ. Simulations

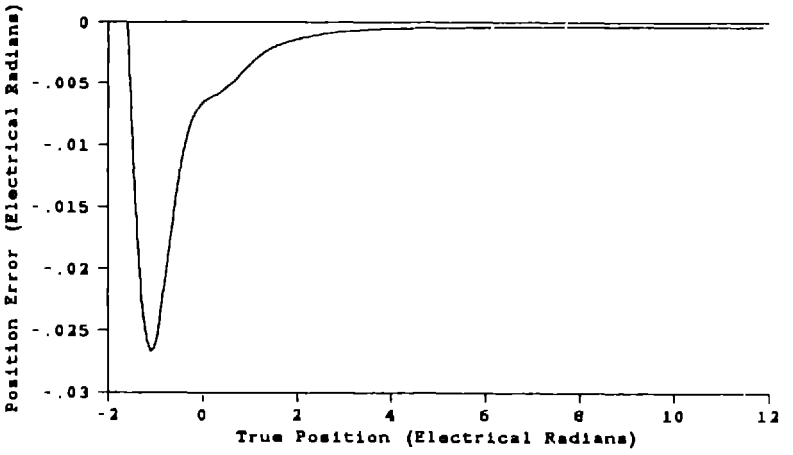


Figure 5.34: Observer position error response for the motor stepping from 1000 to 1100 rpm when the observer load torque τ is 100% high.

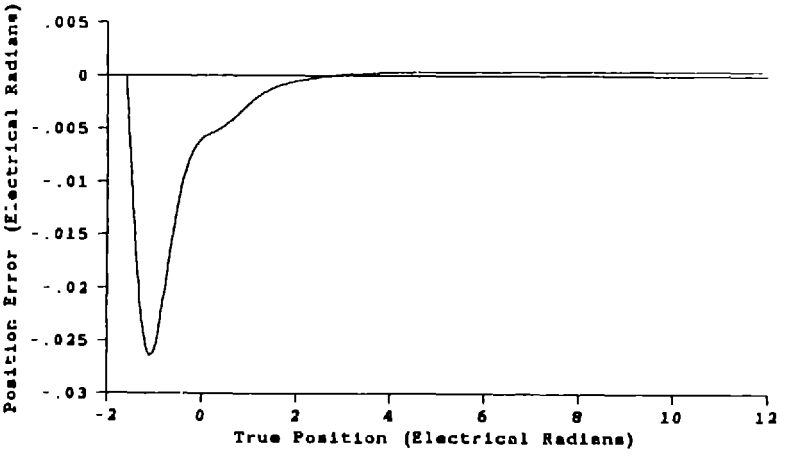


Figure 5.35: Observer position error response for the motor stepping from 1000 to 1100 rpm when the observer load torque τ is 100% low (that is $\tau = 0$).

verify the expected result that an incorrect shaft inertia parameter in the observer only affects the transient response of the observer. The observer is stable even for the large variations of H considered in Figures 5.32 and 5.33. The response of the observer is better when the observer inertia is below the true inertia and so if the operating inertia is unknown, it is best to choose the smallest expected value. The observer is also stable for large variations in load torque, although a small steady-state position error directly proportional to the load torque parameter error will develop. This steady-state error is small due to the sensitivity of the stator currents to changes in the relationship between the speed voltage and the stator voltages. If the load torque will vary during the operation of the observer, it is best to choose an intermediate value.

With the exception of H being too high, errors in the parameters do not effect the settling time of the observer. The observer settles in the time it takes for the shaft to turn four electrical radians. At 1100 rpm (the speed of the motor after the step), four electrical radians corresponds to 11.6 ms.

5.2.3 Noise Disturbances

The stator voltages and currents must be measured in order to implement the observer, and noise introduced by the measurement process may adversely affect the performance of the observer. It is important to study the performance of the estimated-innovation observer in the presence of measurement noise. The standard deviations of typical noise levels in zero-mean, Gaussian noise for the sensors described in Chapter 7 are 0.04 A and 0.2 V. For the purposes of simulation, high levels of noise are taken to be five times the typical levels, that is 0.20 A and 1.0 V.

The observer response to a perfect motor step from 1000 rpm to 1100 rpm, as in Figure 5.26, was used in the noise simulations. The observer error response with typical noise levels is shown in Figure 5.36. The steady-state standard deviation of the position error is 0.0015 erad. In terms of a mechanical revolution, this represents 1 part in 12570. The observer error response with high noise levels is shown in Figure 5.37. The steady-state standard deviation of the position error is 0.0076 erad, or 1 part in 2480. Five times more measurement noise yields five times more observer error.

To understand which measurements the observer is most sensitive to, consider a simulation where there is high level noise only in the current measurements. The

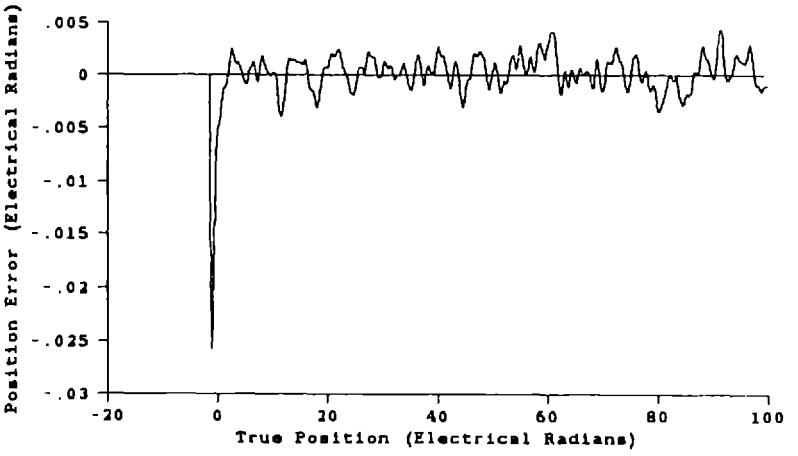


Figure 5.36: Observer position error response for the motor stepping from 1000 to 1100 rpm when typical levels of measurement noise are present in the current and voltage measurements.

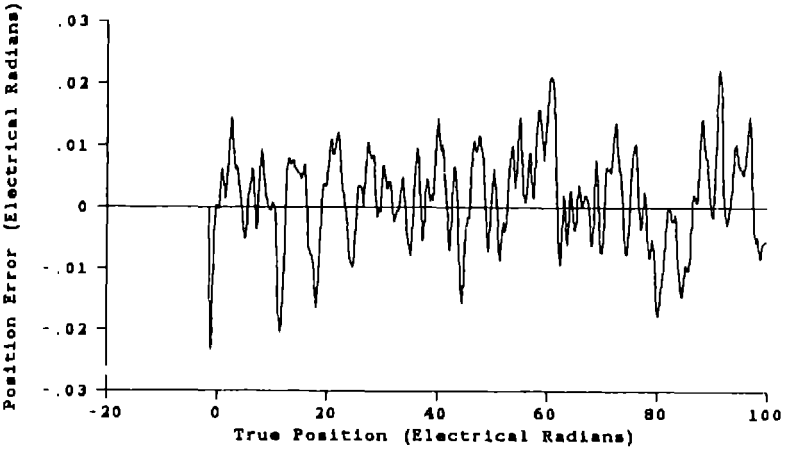


Figure 5.37: Observer position error response for the motor stepping from 1000 to 1100 rpm when high levels of measurement noise are present in the current and voltage measurements.

observer position error response is shown in Figure 5.38. The observer position error response for the high level of noise in the voltage measurements only is shown in Figure 5.39. The observer is more sensitive to noise in the voltage measurements.

Since the stator inductances are small (0.4 mH), the currents in the windings are determined by the very small difference between the driving voltage and the speed voltage. Any error in the measured voltage results in large current errors and thus large position errors. The deviation of the observer quadrature current from the noiseless trajectory when high level voltage measurement noise is present is shown in Figure 5.40. Errors in the measured current enter the dynamics through the innovation and thus do not cause such large errors. Of course, if the observer gains are large enough, substantial current errors will be generated. The deviation of the observer quadrature current from the noiseless trajectory when high level current measurement noise is present is shown in Figure 5.41. The current error is much less than the error with noisy voltage measurements, which predicts the smaller observer position error when noise is present only in the current measurements. The conclusion is that it is critical to perform accurate measurments of the voltage.

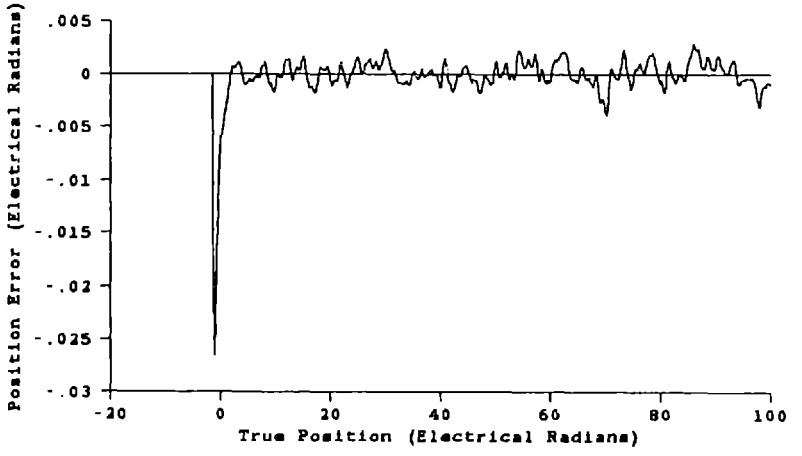


Figure 5.38: Observer position error response for the motor stepping from 1000 to 1100 rpm when high level noise is present in the current measurements. The voltage measurements are noiseless.

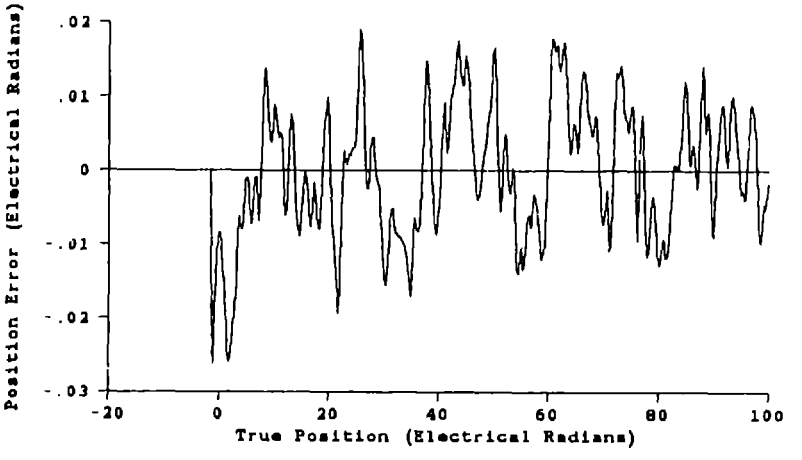


Figure 5.39: Observer position error response for the motor stepping from 1000 to 1100 rpm when high level noise is present in the voltage measurements. The current measurements are noiseless.

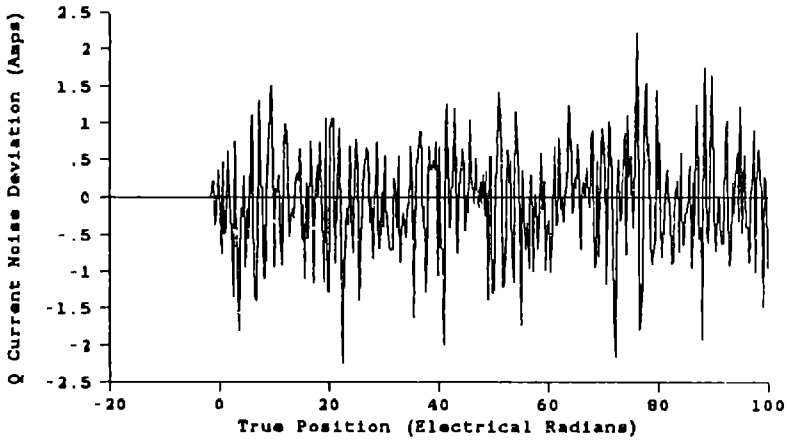


Figure 5.40: Difference between observer quadrature current when there is voltage measurement noise and observer quadrature current when there is no voltage measurement noise. Motor is stepping from 1000 to 1100 rpm.

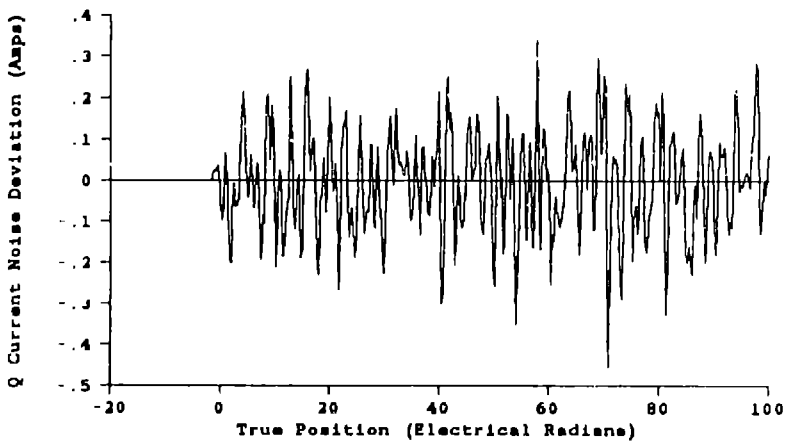


Figure 5.41: Difference between observer quadrature current when there is current measurement noise and observer quadrature current when there is no current measurement noise. Motor is stepping from 1000 to 1100 rpm.

Chapter 6

Stability

The stability of the observer is an important issue. A resolverless motor control system would depend on the observer to provide accurate position information. Loss of observer stability would result in erratic and possibly destructive motor operation. In this chapter, stability of the observer over a range of operating speeds will be considered. The observer is nonlinear and therefore the problem of determining its stability is difficult. The desired behavior of the observer is for the error between the estimated state and the true state to converge to a minimum. However, there are an infinite number of error minimums spaced 2π electrical radians apart since each electrical cycle is identical. Therefore, the observer error may converge to 0, 2π, 4π, etc., depending on the operating conditions. It is apparent then that a simple global stability argument for the observer does not exist. The problem is further complicated by the fact that in a nonlinear system the stability of the observer and the controller are interdependent. In a linear system, the separation theorem allows the stability of the observer and controller to be considered independently. Therefore, if the nonlinear observer is linearized, local observer stability may be determined independent of the local controller stability. It will be shown in this chapter that the local behavior of the linearized observer is surprisingly accurate even for large perturbations.

6.1 Error Dynamics

Rather than determine the stability of the observer itself, it is advantageous to determine the stability of its error dynamics. By using error dynamics, the objective of the stability analysis is always clear: the states in the error dynamics should decay

to the origin. Also, the error dynamics are naturally linearized about the origin. The observer estimation error is defined as

$$\underline{e} = \begin{bmatrix} \underline{e}_i & e_\omega & e_\theta \end{bmatrix}^T = \hat{\underline{x}} - \underline{x} \tag{6.1}$$

where $\underline{e}_i = \begin{bmatrix} e_d & e_q \end{bmatrix}^T$. The estimate $\hat{\underline{x}}$ is produced by the observer while $\underline{x}$ is the true state. By substituting (2.72)–(2.74) and (4.31)–(4.33) in (6.1), a nonlinear error system is obtained that depends on the operating point of the motor, $\bar{\underline{x}}$ and $\bar{\underline{v}}$. The error dynamics are

$$\begin{aligned} \frac{d\underline{e}_i}{dt} &= -\left(\frac{R}{L}\mathbf{I} + \mathbf{J}Ne_\omega\right)\underline{e}_i - \mathbf{J}N\left(\omega_e \underline{e}_i + e_\omega \bar{\underline{i}}\right) - \frac{K}{L}Ne_\omega \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ &\quad + \frac{1}{L}\left[\left(\exp(-\mathbf{J}Ne_\theta) - \mathbf{I}\right)\bar{\underline{v}}\right] + \mathbf{G}_{\underline{i}}\left[\left(\exp(-\mathbf{J}Ne_\theta) - \mathbf{I}\right)\bar{\underline{i}} - \underline{e}_i\right] \end{aligned} \tag{6.2}$$

$$\frac{de_\omega}{dt} = \frac{KN}{H}\left(e_q + \mathbf{G}_\omega\left[\left(\exp(-\mathbf{J}Ne_\theta) - \mathbf{I}\right)\bar{\underline{i}} - \underline{e}_i\right]\right) - \frac{B}{H}e_\omega \tag{6.3}$$

$$\frac{de_\theta}{dt} = e_\omega\,. \tag{6.4}$$

6.2 Linear Analysis

Linearizing the error dynamics (6.2)–(6.4) about zero error yields

$$\begin{bmatrix} \dot{\tilde{e}}_d \\ \dot{\tilde{e}}_q \\ \dot{\tilde{e}}_\omega \\ \dot{\tilde{e}}_\theta \end{bmatrix} = \begin{bmatrix} -\frac{R}{L} - G_{\underline{i}}(1,1) & N\omega - G_{\underline{i}}(1,2) & i_q \\ -N\omega - G_{\underline{i}}(2,1) & -\frac{R}{L} - G_{\underline{i}}(2,2) & -\frac{K}{L} - i_d \\ -G_\omega(1,1)\frac{KN}{H} & (1 - G_\omega(1,2))\frac{KN}{H} & -\frac{B}{H} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} G_{\underline{i}}(1,1)i_q - G_{\underline{i}}(1,2)i_d + \frac{1}{L}v_q \\ G_{\underline{i}}(2,1)i_q - G_{\underline{i}}(2,2)i_d - \frac{1}{L}v_d \\ \frac{KN}{H}(G_\omega(1,1)i_q - G_\omega(1,2)i_d) \\ 0 \end{bmatrix} \begin{bmatrix} \tilde{e}_d \\ \tilde{e}_q \\ \tilde{e}_\omega \\ \tilde{e}_\theta \end{bmatrix} \tag{6.5}$$

where a superscript tilde denotes a perturbation about equilibrium. Note that the linearized error dynamics depend on the operating point of the motor. The linearized error system describes the nonlinear error dynamics well over a surprisingly large range of operating conditions. For example, consider the results shown in Figure 6.1 where the motor has taken a perfect step from 1000 to 1100 rpm and the position error has been computed using the linearized error dynamics (6.5).

The corresponding results for the fully nonlinear estimated-innovation observer are shown in Figure 5.26.

A much larger transient is shown in Figure 6.2 where the observer is started from rest and converges to a motor running at a constant 1000 rpm. These results should be compared with those in Figure 5.9 which were computed using the nonlinear estimated-innovation observer. Even over a large transient the linearized dynamics closely approximate the nonlinear dynamics.

It was observed in Chapter 5 that the performance of the estimated-innovation observer was related to speed and, for the particular gain set used (see Figure 5.3), the observer settled to a steady-state oscillation at 4000 rpm and was unstable for higher speeds. This instability is accurately predicted in a pole plot of the linearized error system.[1] In Figure 6.3, the poles of (6.5) with the gains given in Figure 5.3 have been plotted for the speeds 100–5000 rpm. Two poles associated with 4000 rpm are extremely close to the $j\Omega$ axis and two poles associated with 5000 rpm are some distance into the right half-plane. At slow speeds there is a pole near zero which accounts for the slow, overdamped position error response.

As discussed in Chapter 4, it is desirable to tightly couple the electrical and mechanical subdynamics. If the subdynamics are not tightly coupled, rapid corrections to the electrical subdynamics will not effect rapid corrections to the mechanical subdynamics. It was further stated that an open-loop simulation would not have tightly coupled subdynamics. A pole plot of the linearized estimated-innovation observer with the gains set to zero is shown in Figure 6.4. Poles which may be associated with the electrical subdynamics are far to the left while poles which may be associated with the mechanical subdynamics are close to the $j\Omega$ axis. There is a wide separation of time scales (the system is stiff). In comparison, the effect of the innovation in the estimated-innovation observer is to draw the modes together.

Closer examination of (6.5) reveals more about the estimated-innovation observer. Note how $G_\omega(1,1)$ and $G_\omega(1,2)$ couple electrical errors and position error into the speed dynamics. In particular, $G_\omega(1,2)$ is associated with the torque-producing current i_q and is the source of the extra torque in the observer needed to quickly force the observer position into line with the motor position. Apparently, $G_\omega(1,2)$ is the most important gain. A pole plot of (6.5) where all the gains except $G_\omega(1,2)$ have been set to zero is shown in Figure 6.5. This single gain largely determines the shape the plot even when all the gains are used. The addition of $G_\omega(1,1)$

[1] See Appendix C for a listing of the program used to generate the pole plots.

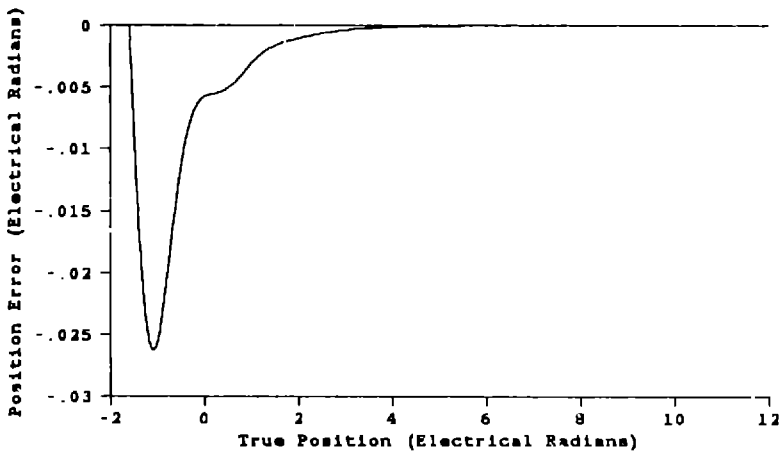


Figure 6.1: Linearized position error response for the motor stepping from 1000 to 1100 rpm.

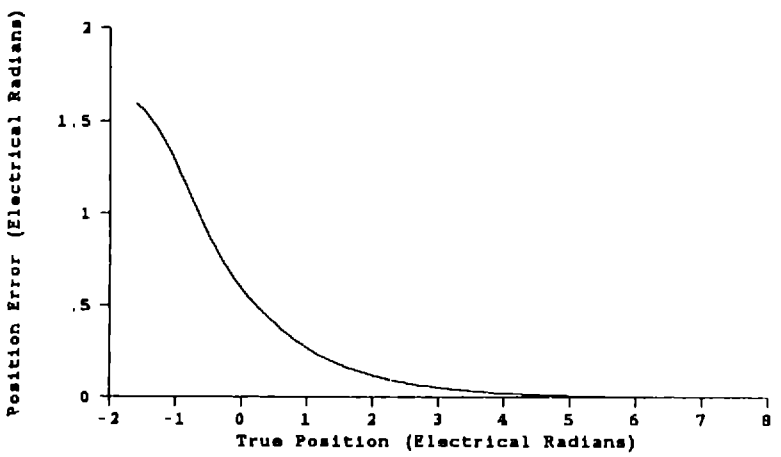


Figure 6.2: Linearized position error response from rest for the motor running at a constant 1000 rpm.

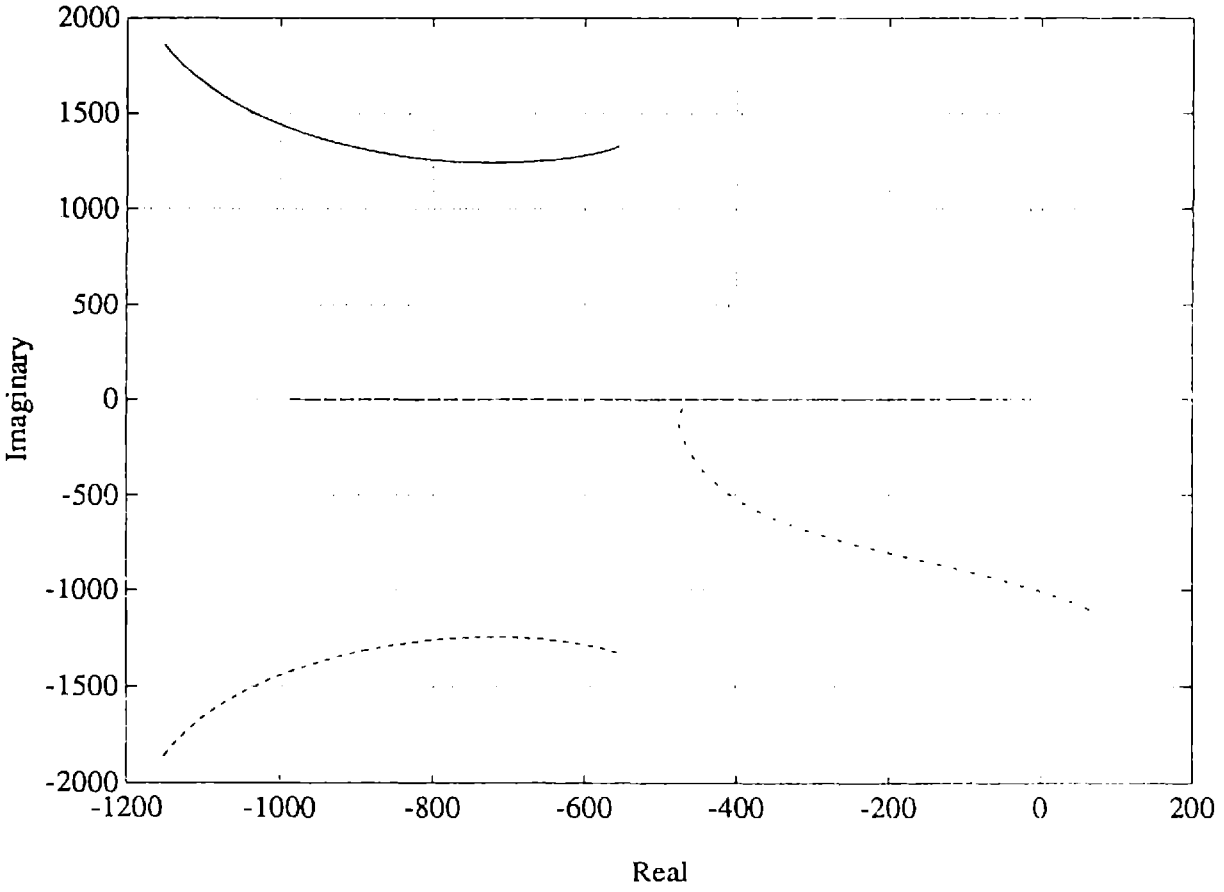


Figure 6.3: Pole plot of linearized system with full gain matrix for the speeds 100–5000 rpm. $\mathbf{G_I} = \begin{bmatrix} 200 & -100; & -100 & 200 \end{bmatrix}$, $\mathbf{G_\omega} = \begin{bmatrix} 100 & -300 \end{bmatrix}$.

makes the error response faster at slow speeds but also causes instability to occur sooner, as seen in Figure 6.6. Note that the pole plot predicts that this set of gains will give a faster error response than the full set used in Chapter 5. Reversing the sign of $G_\omega(1,1)$ results in instability at low speeds instead of high speeds.

The gains $\mathbf{G_I}$ associated with the electrical subdynamics are relatively unimportant since they do not effect corrections in the mechanical subdynamics. For example, if the the mechanical gains $\mathbf{G_\omega}$ are set to zero, the pole plot exhibits a wide separation of time scales as shown in Figure 6.7.

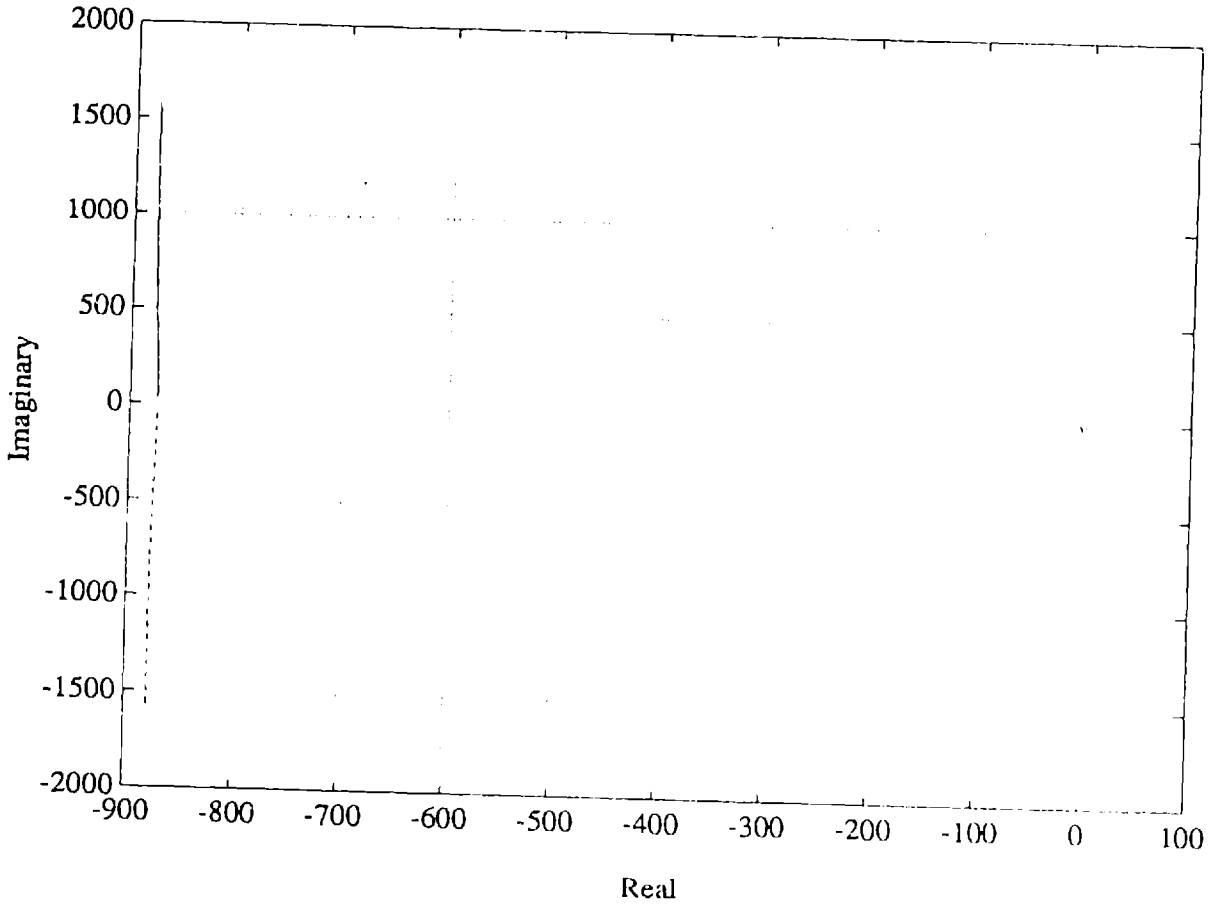


Figure 6.4: Pole plot of linearized system with zero gain matrix for the speeds 100–5000 rpm. $\mathbf{G}_{\mathbf{I}} = \begin{bmatrix} 0 & 0; & 0 & 0 \end{bmatrix}$, $\mathbf{G}_{\omega} = \begin{bmatrix} 0 & 0 \end{bmatrix}$.

6.3 Nonlinear Analysis

The nonlinear structure of the estimated-innovation observer precludes definitive statements based on an eigenvalue analysis. The strongest statements can be made using a summarizing function, such as a Liapunov function, to find the large-signal conditions where the system is stable. Unfortunately, no such function has yet been found, and, in fact, such a function may not exist. A summarizing function may not exist because a unique global error minimum does not exist. Each electrical cycle is identical and thus there are an infinite number of error minimums spaced 2π electrical radians apart. Attempting to find a summarizing function by limiting the range of operation to within a single electrical cycle will fail because the bound-

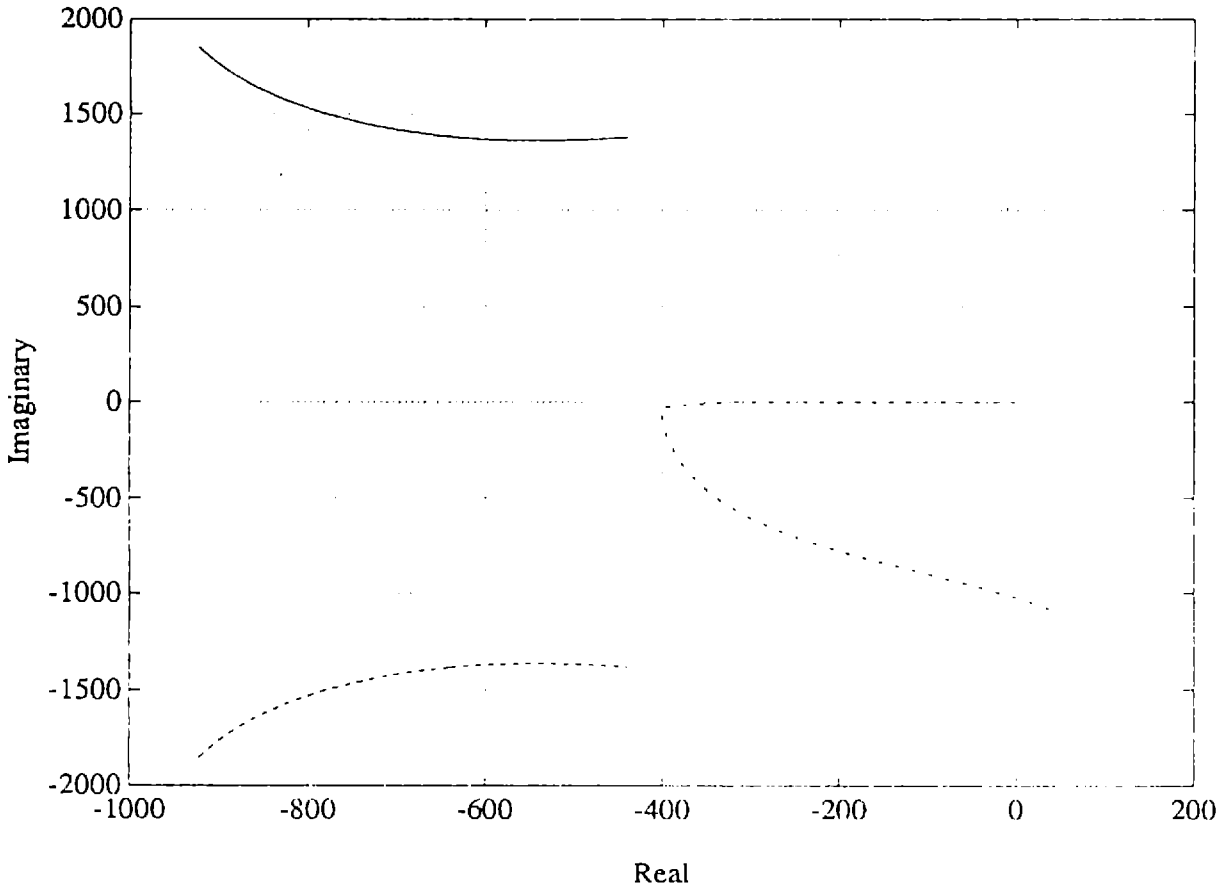


Figure 6.5: Pole plot of linearized system with $G_\omega(1,2) \neq 0$ for the speeds 100–5000 rpm. $\mathbf{G}_{\underline{\mathbf{r}}} = \begin{bmatrix} 0 & 0; & 0 & 0 \end{bmatrix}$, $\mathbf{G}_\omega = \begin{bmatrix} 0 & -300 \end{bmatrix}$.

ary between electrical cycles, that is, the boundary from which the observer may converge to within either electrical cycle, is not constant but rather depends on the operating conditions of the motor.

6.4 Gain Selection

The nonlinearity of the estimated-innovation observer results in the pole positions being dependent on speed. Therefore, constant gains cannot be selected to simply place the poles. Instead, the gains must be chosen to achieve the best pole locus over the range of speeds of interest.

An interesting possibility is to choose *nonconstant* gains such that the position

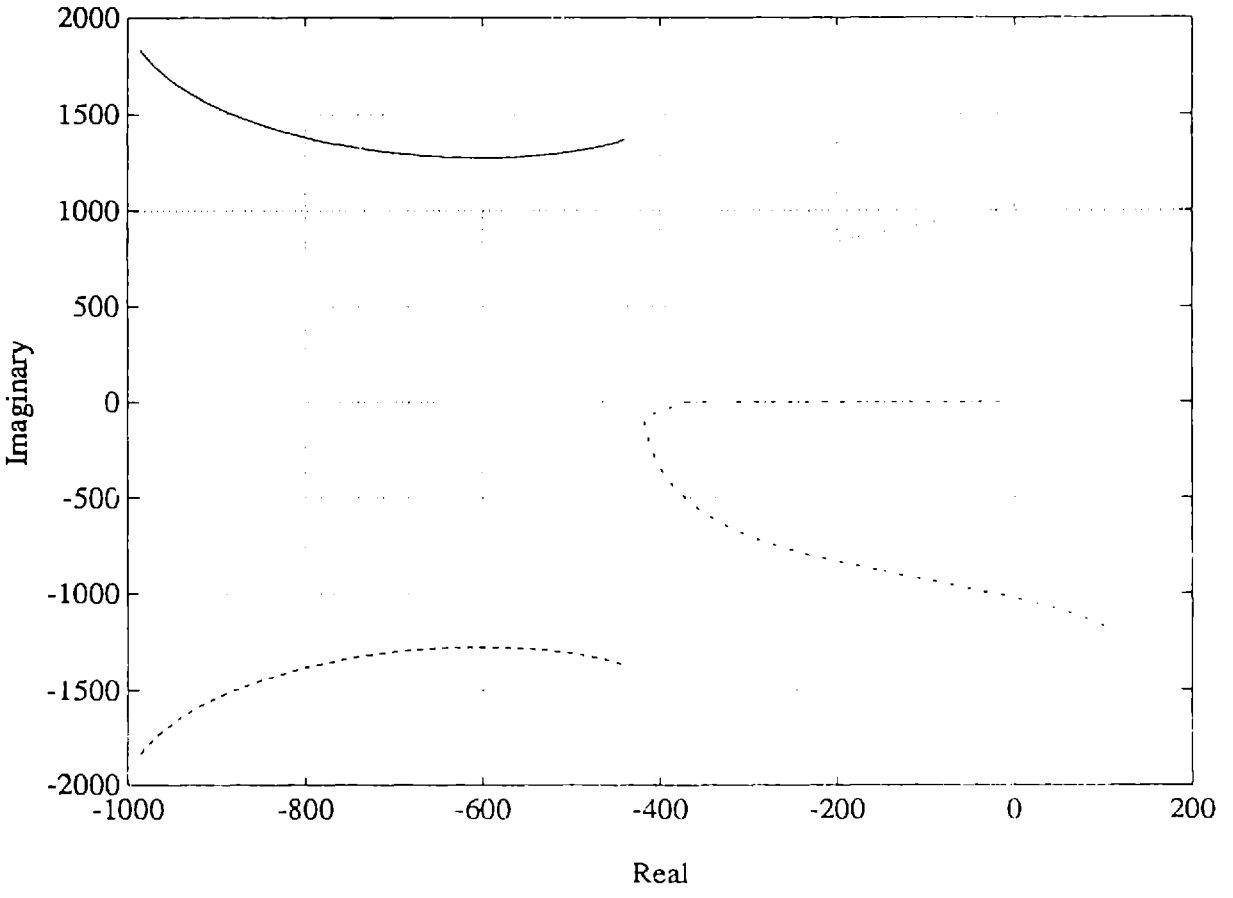


Figure 6.6: Pole plot of linearized system with mechanical subdynamics gains for the speeds 100–5000 rpm. $\mathbf{G}_{\underline{\mathbf{i}}} = \begin{bmatrix} 0 & 0; & 0 & 0 \end{bmatrix}$, $\mathbf{G}_{\omega} = \begin{bmatrix} 100 & -300 \end{bmatrix}$.

of the poles is independent of speed. The gains could be expressed as functions of the estimated and measured variables. An alternative would be to use precomputed sets of gains and establish a scheduling criterion.

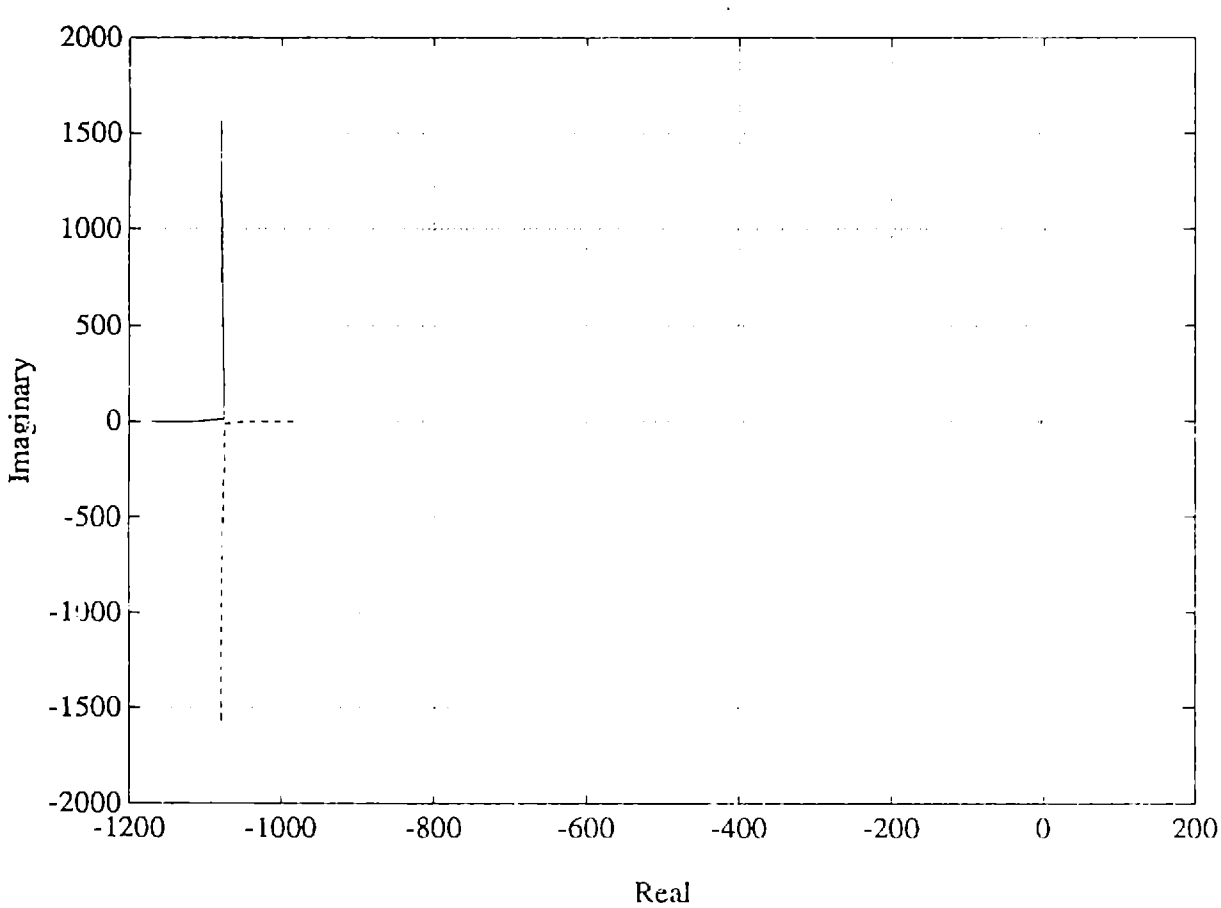


Figure 6.7: Pole plot of linearized system with electrical subdynamics gains for the speeds 100–5000 rpm. $\mathbf{G}_{\mathbf{I}} = \begin{bmatrix} 200 & -100; & -100 & 200 \end{bmatrix}$, $\mathbf{G}_{\omega} = \begin{bmatrix} 0 & 0 \end{bmatrix}$.

Chapter 7

Hardware

Hardware to implement the observer in real-time was constructed and tested. The system consists of a permanent-magnet synchronous motor and its inverter and controller coupled to a digital signal processor where the observer is implemented. In this chapter, the hardware and low-level software will be documented. Further details may be found in Appendix D.

7.1 Variable Frequency Drive

The permanent-magnet synchronous motor and drive electronics used for the initial tests of the observer were donated by Industrial Drives of Radford, Virginia. The system was set up as shown in Figure 7.1. Each major component will be discussed in turn.

7.1.1 Motor

The nameplate data for the Industrial Drives motor is:

Model	BR-3103-1001-B
Ser. Num.	86B176-9
Rated hp	3.70
Maximum Speed	5000 rpm
Rated Speed	5000 rpm
Stall Torque	5.8 #-ft
Peak Torque	25 #-ft
Rated Voltage	210 V l-to-l rms
Stall Current	15.9 A rms
Peak Current	70 A rms .

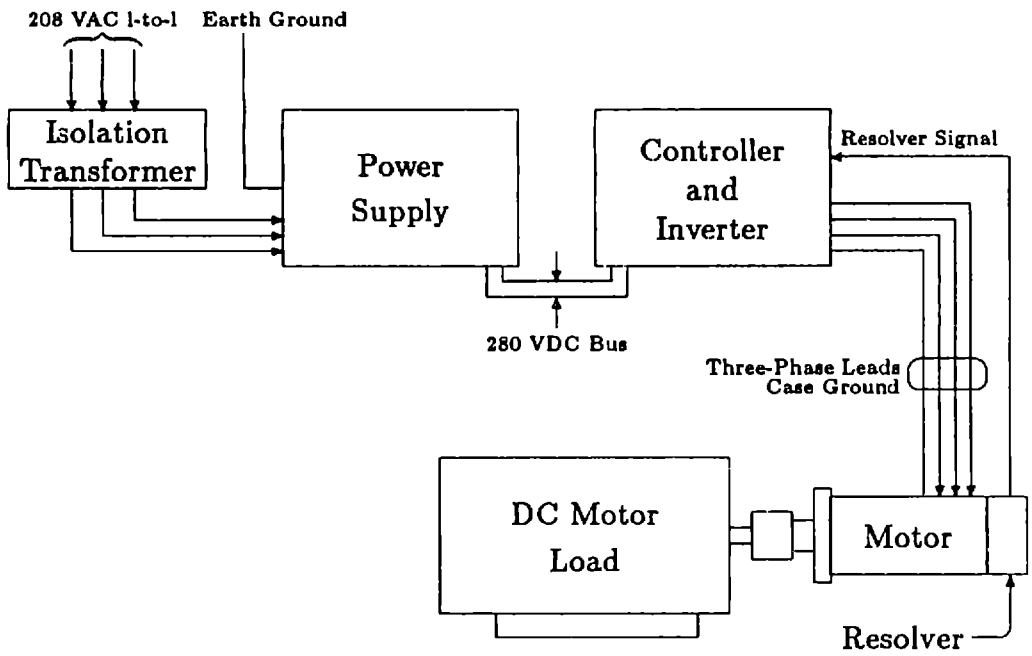


Figure 7.1: Motor system configuration.

Including the shaft, the motor is $11\frac{1}{4}$ inches long, and has a frontal area of $6\frac{1}{2}$in. $\times$ $6\frac{1}{2}$in. The rear endbell is $3\frac{5}{16}$ inches long and contains the rear shaft bearings, the resolver, the resolver connector, and the phase windings connector. The connectors are standard mil-spec.

The stator of the motor is "air-wound". The windings are embedded in epoxy such that the physical air-gap in the motor is uniform. The windings are wye-connected with an isolated neutral. A fourth connection on the phase windings connector is the case ground for the motor.

The rotor is constructed as in Figure 2.3. The inner shaft is steel with a jacket of permeable iron around it. Six neodymium-iron-boron permanent magnets are glued lengthwise to the iron. Finally, the rotor is wrapped in Kevlar that is rated to well beyond 6000 rpm in order to guarantee the magnets never separate from the rotor. The motor will be destroyed if a rotor magnet detaches as the brittle magnet material will disintegrate into a persistent, highly magnetized dust.

The parts of the motor are machined to tolerance and thus the press-fit shaft bearings have no adjustment. The frameless resolver depends on the machined

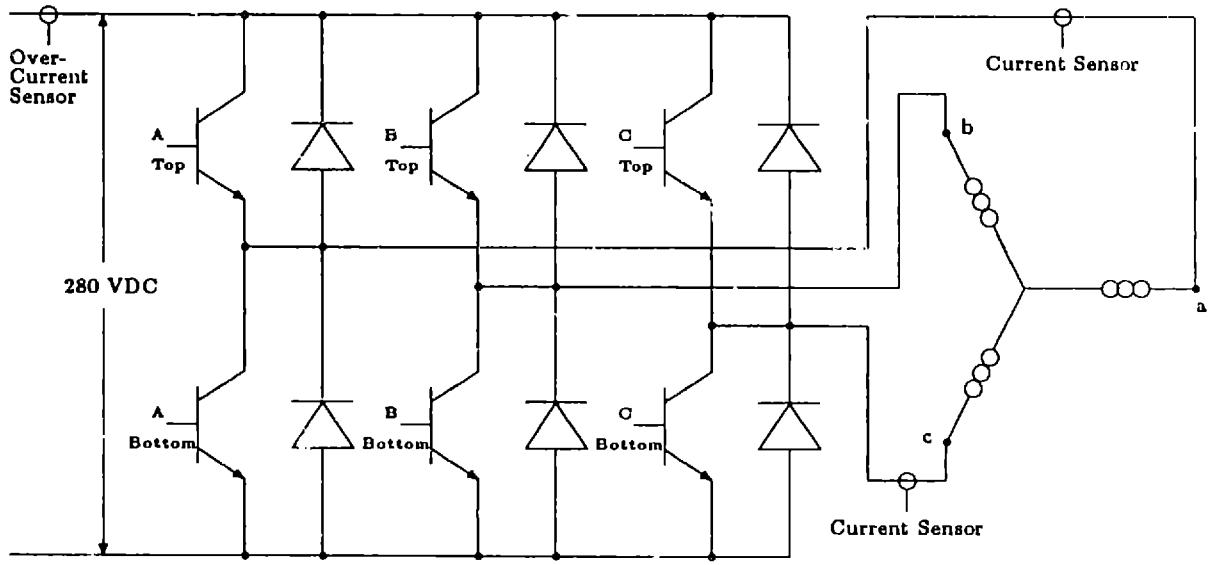


Figure 7.2: Full bridge inverter configuration.

tolerances of the motor. The rotor of the resolver is mounted directly on the motor shaft and the stator of the resolver mounts on the motor case. Therefore, the concentricity of the resolver rotor and its stator depends on the concentricity of the rotor and the motor case. The output of the resolver is a sine and cosine voltage that is used by a resolver-to-digital converter to produce a position word. The position word will only be accurate if the resolver rotor and stator are concentric.

7.1.2 Power Supply and Inverter

An isolation transformer is used between the power supply and the utility for safety. The power supply, model PSR3-208/50-01-002, converts the 208 V line-to-line from the utility to 280 V DC. The power supply is a full-wave rectifier with two large smoothing capacitors (2.2 mF each). It is rated at 50 Amps rms line current. A shunt regulator is included for use during regeneration that can dissipate 300 Watts. A remote regeneration resistor may be used if more dissipative capacity is needed.

The inverter, model BDS3-208/40-01-200, consists of full-bridges across the bus voltage for each phase using bipolar power transistors and is rated at 40 Amps rms per phase. The configuration is shown in Figure 7.2. Each transistor has a dedicated base drive where the drives for the three top transistors are flying. They

are powered from a multi-tap transformer which is powered by a regulated DC supply derived from 115 VAC. The base drives are coupled to logic signals from the control system using opto-isolators.

7.1.3 Control System

The control system in the BDS3 consists of two nested control loops: a fast, inner current-control loop and a slower, outer PI speed-control loop. Both control loops are implemented using analog electronics. A microcomputer is present on the circuit board and is used to produce a current command based on the measured position of the rotor. That is its only function.

The fast inner current-control loop is implemented with proportional control. The currents in phases a and c are measured and the current in phase b is computed with an op amp summer. (Recall that due to the wye connection, $i_b = -(i_a + i_c)$.) The measured currents are compared with commanded currents and the error signal is compared with an 8 kHz triangle wave using a saturating comparator. The resulting high-low pulse-width modulated signal is used to drive NAND gates which drive the optoisolators in the base drives. The high bandwidth of this control loop ensures that the real currents closely follow their commands.

The details of the pulse-width modulation (PWM) circuitry are important (see Voltage Sensors in Section 7.3.1) and interesting. In order to guarantee there will be no shoot-through, the PWM circuitry for each top transistor produces a high-low signal that is the logical inversion of the signal for the corresponding bottom transistor. Therefore, if the top transistor is on, then the bottom transistor is commanded off, and vice-versa. Additional circuitry is needed to account for the difference between the time needed to turn a power transistor on and the time needed to turn it off. Turning a power transistor on is relatively fast. Once on, it is driven heavily into saturation and therefore turning it off takes substantially longer, usually several microseconds or more. Therefore, an RC circuit is included in each PWM circuit that delays the other side of the bridge from turning on until the present side has completely turned off. In the BDS3 this delay is 15 μs.

The NAND gates used to signal the base drives may be inhibited by applying a logic zero to their second lead, thereby turning all the transistors off. The overcurrent sensor shown in Figure 7.2 uses this method to override the control system if excessive bus current is detected. Inhibiting the NAND gates may also be used by

the operator to completely disengage the control system from the motor.

In the BDS3, the current command for the inner current-control loop comes from the microcomputer through a digital-to-analog converter. The function of the microcomputer is to read the output of the resolver-to-digital converter and send a sinusoidal current command wave to the D/A converter at the appropriate torque angle for the error signal coming from the outer, speed-control loop. By using a microcomputer to supply this sine wave, the torque angle is easily changed based on the signals from the speed-control loop, thereby improving the performance of the motor. The microcomputer retrieves the torque angle from a lookup table and adds it to the measured position. The gain of the D/A converter is controlled by the speed-control loop thereby allowing the magnitude of the current command to be changed as necessary.

The PI speed control loop is clamped by back-to-back zener diodes thereby allowing it to saturate gracefully should the motor be unable to execute the command. Additionally, the velocity error loop may be shorted through front panel connections, thereby removing the integral action from the controller and severely reducing the proportional gain (see Appendix D). This condition is known as the torque-hold mode and is immensely useful in the laboratory as it "tames" the motor in order to perform various experiments.

7.1.4 DC Motor Load

A large DC motor is used to provide the load torque τ. It is rated at 4 hp at 3450 rpm. The windings are rated at 115 V and the armature current at 32 A. When the DC motor was used to drive the permanent-magnet motor, it was operated with separately exited windings. When used as a load torque, the field winding was excited and the armature current dissipated in a large power resistor. The procedure used to compute the load torque is given in Section 8.1.2.

7.2 Computer System

Due to the complexity of the observer, a real-time implementation requires substantial computing capability. Digital signal processors, which are tailored for high-speed integer arithmetic, can provide this capability. When incorporated into development systems, these processors are ideal for implementation and experimentation

with real-time algorithms such as observers.

7.2.1 Digital Signal Processor

Digital signal processors have become popular because they perform better than general-purpose microprocessors in many applications. Their popularity has resulted in a profusion of new processors on the market. Texas Instruments began the "DSP revolution" with its very successful TMS32010, introduced in 1983. In 1985, TI introduced the TMS32020, which is software compatible with the TMS32010.

The TMS32020 is a microprocessor in which features important in a general-purpose environment have been sacrificed in favor of features important in a numeric processing environment. For example, in a general-purpose environment, it is important to have a large number of addressing modes such that high-level languages may be implemented efficiently and easily. The TMS32020, has only three addressing modes.

With respect to control applications, the salient features of the TMS32020 are:

- 200 *ns* instruction cycle time,
- 544 words of on-chip data RAM,
- sixteen input and sixteen output channels,
- three external, maskable, user interrupts,
- bit manipulation and logical instructions,
- 0 to 16-bit scaling shifter,
- 32-bit ALU and accumulator, and
- single-cycle multiply/accumulate instructions.

Most of the instructions in the TMS32020 instruction set execute in one 200-*ns* cycle. Of those that execute in more than one cycle, most are multi-function instructions. The on-chip RAM speeds executions as many instructions execute faster when using it. The interrupts structure is extremely fast and interrupt service has been experimentally observed to begin approximately 400 *ns* after the interrupt is received. The 0 to 16-bit scaling shifter, useful for scaling variables while performing

integer arithmetic, can be used with no time penalty with most instructions. The 32-bit ALU and accumulator with single-cycle multiply/accumulate instructions is the TM320 series' original claim to fame. The single-cycle performance is actually achieved only for a series of multiply/accumulate instructions, but nevertheless, could be utilized in control applications. Single-cycle multiply/accumulate is ideal for implementing difference equations.

7.2.2 Development System

The popularity of the TMS320 series digital signal processors has resulted in the availability of a wide variety of support systems. Loughborough Sound Images of Great Britain manufactures a TMS32020 development system for use in IBM Personal Computers.[1] The salient features of this system with respect to laboratory development of real-time algorithms are that it:

- conveniently uses the PC as a front-end processor and file-server,
- contains a 17 μs A/D channel and 3 μs D/A channel,
- provides for the efficient use of the TMS32020 interrupts in conjunction with the A/D and D/A channels,
- allows access to 12 TMS32020 input and 12 TMS32020 output ports, each 16-bits,
- supports up to 128K words of zero wait-state external memory, and
- has complete software support with the Texas Instruments assembler and linker and the Loughborough debug monitor.

Software is written, assembled, and linked on the PC, and then downloaded onto the TMS32020 development system using the debug monitor. The debug monitor provides all the standard features typically found in debug monitors. After extensive experience with the system, it has been found to be reliable and capable.

[1] Available in the United States through Spectrum Signal Processing, Waltham, MA.

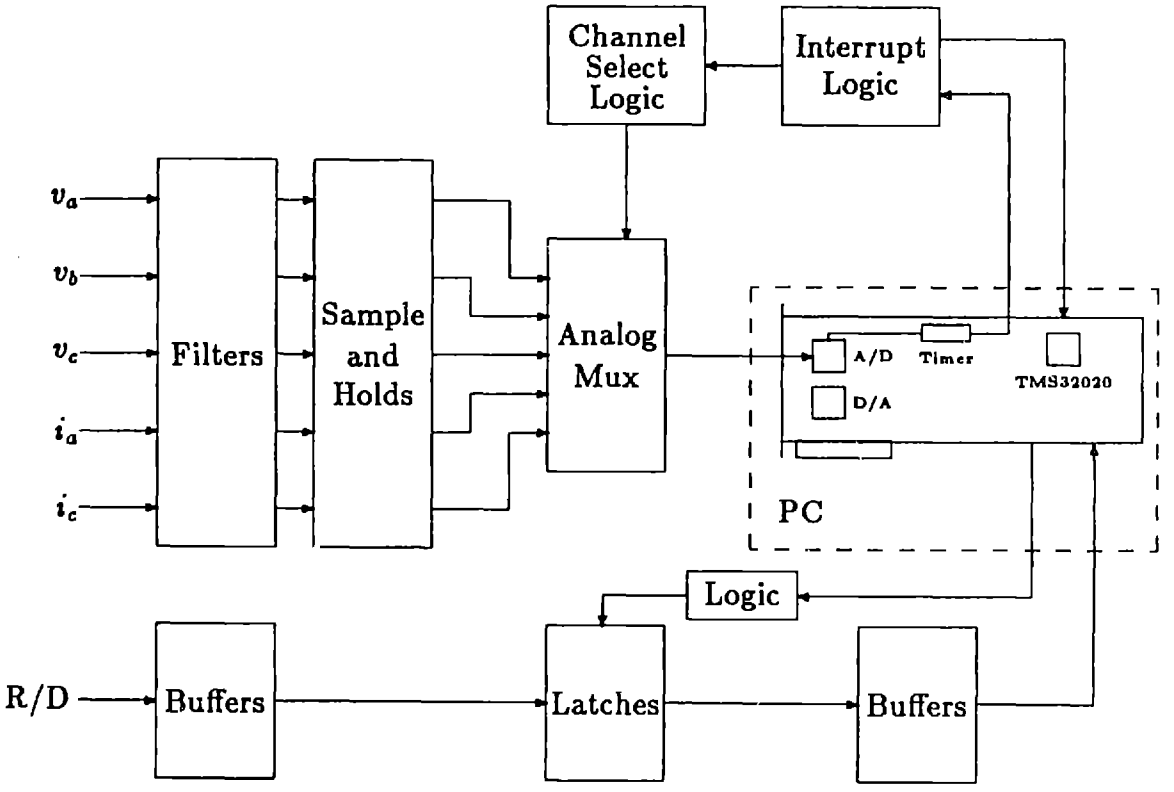


Figure 7.3: Interface between the motor system and computer system.

7.3 Interface Circuitry and Software

The TMS32020 development system contains a Burr-Brown PCM75 16-bit audio A/D converter. The analog input to the device is buffered with a voltage-follower circuit. Anti-aliasing filters and sample and hold circuits must be supplied externally. Additionally, since there is only one A/D converter, an analog multiplexer along with any associated logic circuitry must be supplied. In this section, the interface between the motor system and computer system will be described. A block diagram of the interface is shown in Figure 7.3.

7.3.1 Sensors

The electrical variables of the motor must be measured for the observer. The three stator voltages are measured and transformed to two-phase voltages using the $\alpha\beta 0$ transformation derived in Chapter 2. Two stator currents are measured and

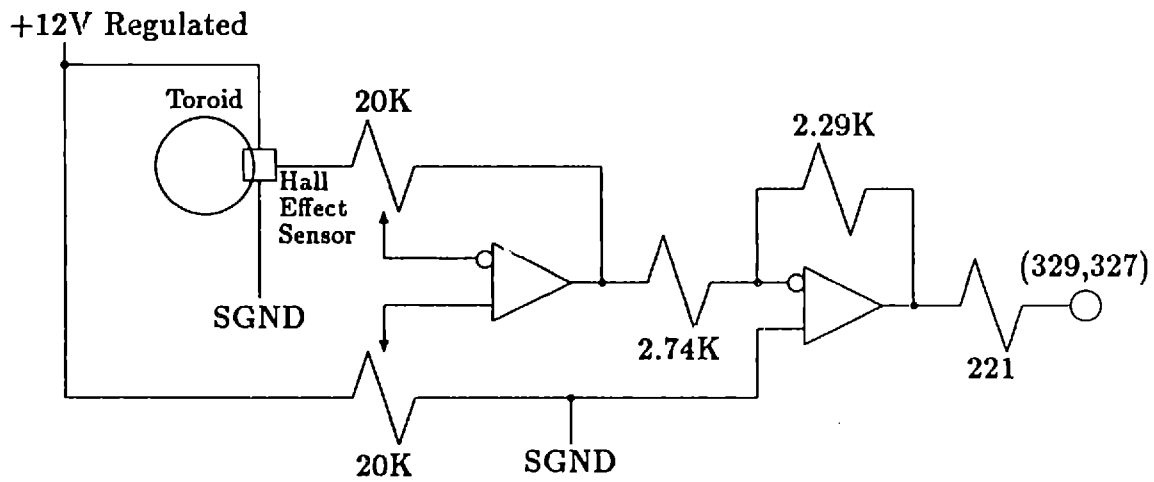


Figure 7.4: Circuitry used to measure the phase currents. The phase lead passes through the toroid a single time as it leaves the inverter. Current measurement outputs for i_a and i_b are available at probe points 329 and 327, respectively.

the third may be computed due the wye connection of the stator windings. The measured stator currents must also be transformed into two-phase currents using the $\alpha\beta 0$ transformation.

Current Sensors

The controller in the Industrial Drives system needs current measurements and therefore two current sensors are present as shown in Figure 7.2. These sensors are also used by the observer for current information. The sensors are Hall effect devices that measure the current in the phase leads going to the motor. Phases a and c are measured and an op amp summer is used to compute the current in phase b. The buffering circuitry for the Hall effect sensors is shown in Figure 7.4. The phase lead passes through the toroid a single time as it leaves the inverter. The outputs from the Hall effect sensors for phases a and c are available at probe points 329 and 327, respectively, on the Industrial Drives controller board.

Voltage Sensors

When a base drive commands a top transistor to be on, see Figure 7.2, the voltage on the phase lead quickly rises to the full bus voltage. When the base drive commands the top transistor off, the transistor turns off and the voltage on the winding goes low for some time independent of the bottom transistor while the winding discharges its current through a flyback diode. Therefore, the base drive signals are not true indications of the voltages on the windings. The voltages must be measured on the motor phase leads.

The voltages applied to the phase windings are pulse width modulated signals that are, ideally, either on or off. That is, the winding is connected to either the positive side of the bus or the negative side of the bus. Therefore, the phase voltages are simply high-low signals. A simple, inexpensive, and safe way to measure the voltages would be to use opto-isolators. The isolating LED (with a current-limiting resistor) could be connected from the phase lead to the low side of the bus, and thus the logic side would indicate high when the phase is connected to the high side of the bus and low when it is connected to the low side of the bus. With appropriate filtering to remove the PWM carrier frequency, the fundamental phase voltage is thus measured.

Experiments with the Industrial Drives motor revealed nonidealities in the voltage waveforms when the motor is operated at high speed and low torque. These nonidealities are a result of the delay between the time a transistor is turned off and its complement is turned on (refer to Section 7.1.3). If the complement transistor is not on by the time the flyback diode turns off, there will be no current in the winding and the phase voltage will be equal to the sum of the neutral voltage and the speed voltage. The neutral voltage takes on intermediate voltages between the bus rails dependent on the speed voltage and the voltages applied to the other phases. Examples of some of the nonidealities seen in the waveforms are shown in Figure 7.5.

Figure 7.5a shows the phase lead voltage when the bottom transistor turns off and the top transistor turns on when the speed voltage of the neutral with respect to that phase is negative. After the bottom transistor turns off, the voltage rises to the high side bus voltage as the current in the winding discharges through the top flyback diode. The current ramps to zero before the the top transistor turns on and the negative speed voltage seen at the phase lead causes the phase voltage

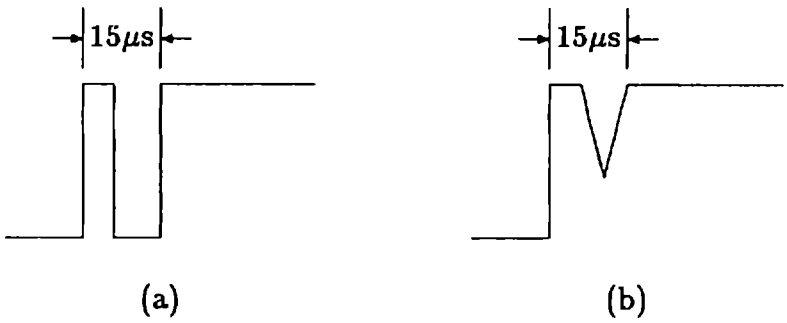


Figure 7.5: Examples of nonidealities seen in the voltage waveforms. In both cases, the top transistor is not turning on before the current in the phase has ramped down to zero. Until the top transistor turns on, the voltage at the phase lead is the sum of the neutral voltage and the speed voltage. (a) Large negative speed voltage. (b) Small speed voltage.

to bounce back down to the low side bus voltage turning the bottom flyback diode on. Finally, the top transistor turns on and the phase voltage rises to the high side bus voltage. The bounce from rail to rail is clean and therefore the opto-isolator method for measuring the voltage continues to work properly.

Figure 7.5b shows the bottom transistor turning off and the top transistor turning on when the speed voltage is at an intermediate value. As a result, the bounce is ragged since the speed voltage is not constant. In this situation, the opto-isolator method for measuring the voltages would not function properly since the opto-isolator does not linearly track the voltage. When the ragged edge occurs over many voltage chops, the error in the average of the opto-isolator measurement can become significant.

The inductance of the windings in the Industrial Drives motor is very small due to the large air-gap and surface-mounted magnets. Only a small voltage needs to be applied to a winding in the absence of speed voltage to obtain the desired current. A large bus voltage is necessary to overcome the speed voltage of the rotor magnets and it is the small difference between the impressed bus voltages and the speed voltage that determines the current in the windings. Unfortunately, the entire bus voltage must be measured even though it is only the last few volts that are driving the current in the phase. As a consequence, the voltage measurements must be

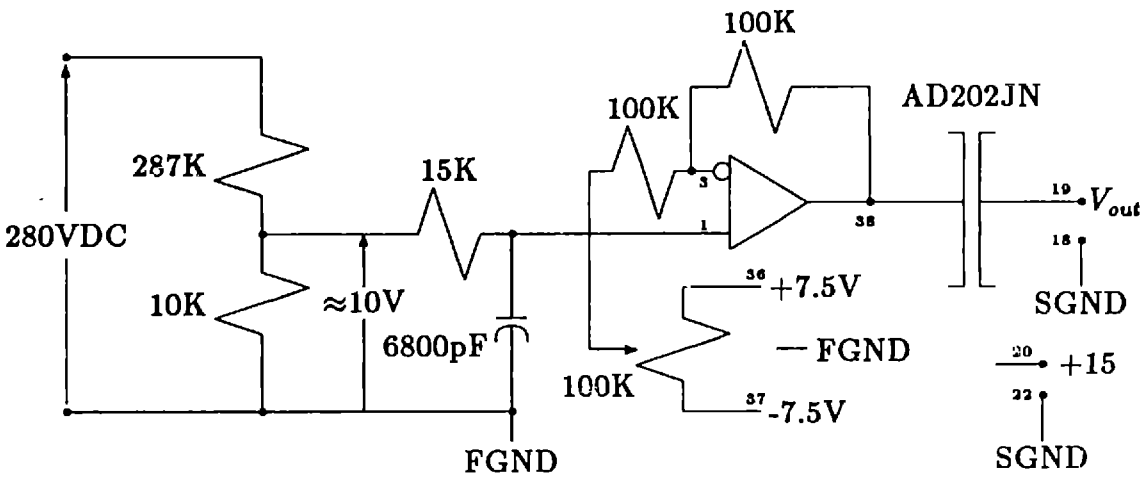


Figure 7.6: Isolation amplifier circuit for measuring the phase voltage.

extremely accurate and any error, such as that introduced by the ragged voltage chopping in the opto-isolators method, is unacceptable.

In order to more accurately measure the voltages, floating isolation amplifiers were used. The circuit is shown in Figure 7.6. It is desirable to utilize the full large signal swing of the amplifier. However, it is necessary to bandlimit the large signal input to prevent internal ringing. The RC filter in the input limits the amplifier input bandwidth to 1500 rad/s. The floating ± 7.5 V supply is used to add a negative offset. Note that the configuration of the circuit is noninverting and it has the high input impedance of a noninverting configuration. As the output of the floating amplifier changes, the offset voltage changes due to the changing amount of current being drawn through the offset voltage divider. The nonconstant offset voltage introduces an error in the measurements that was corrected for in the software used to process the measurements.

7.3.2 Filters

The signals from the current and voltage sensors must be filtered before they are sampled in order to prevent aliasing of unmodeled harmonics and high-frequency noise. Two two-pole filters were cascaded to form a fourth-order lowpass filter. The

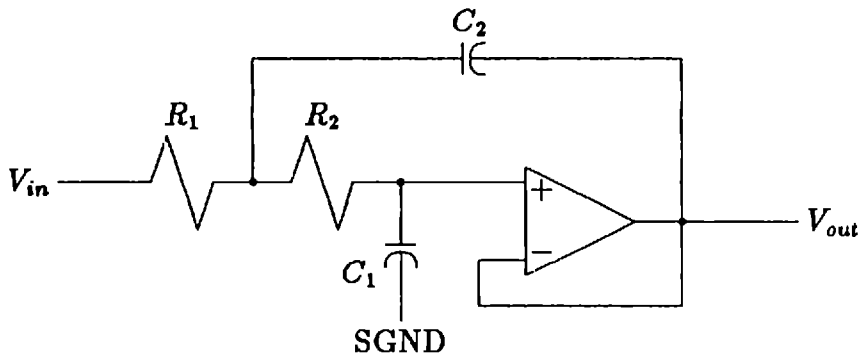


Figure 7.7: Low-pass filter circuit used to prevent aliasing.

two-pole filter circuit is shown in Figure 7.7. The cut-off frequency is

$$f_c = \frac{1}{2\pi\sqrt{R_1 R_2 C_1 C_2}} \tag{7.1}$$

and the damping factor is

$$\varsigma = \frac{1}{2}\frac{R_1 + R_2}{\sqrt{R_1 R_2}}\sqrt{\frac{C_1}{C_2}}. \tag{7.2}$$

The gain in the pass-band is unity due to the unity negative feedback.

For simplicity, assume $R_1 = R_2 = R$ and $C_2 = 2C_1 = C$. The values $R = 15k\Omega$ and $C = 0.01\mu$F were used in the experiments to produce a cut-off frequency of 750 Hz and critical damping. A cut-off of 750 Hz is appropriate since the maximum electrical frequency of the motor is 250 Hz and the Nyquist frequency of the observer is 2.5 kHz.

7.3.3 Sample and Holds

Sample and hold circuits (S/H) on each channel are simultaneously held at the beginning of each sample period. The held channels are then sequentially multiplexed to the A/D converter. The capacitor used in the S/H must be chosen so that the acquisition time is not long, and so that the held output does not droop significantly. At faster sampling rates the acquisition time is more critical and dictates the use of a smaller hold capacitor. At slower sampling rates the droop rate is critical and dictates the use of a larger capacitor. Tables provided by the manufacturer aid in the choice of an appropriate size capacitor. It is important to use a capacitor with

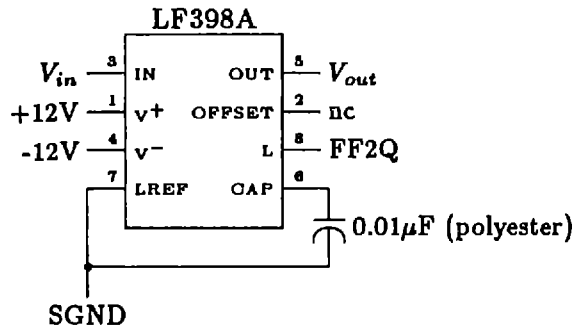


Figure 7.8: Sample and hold circuit used with each channel.

an appropriate dielectric. Hysteresis in the dielectric rules out the use of mylar or ceramic capacitors. Polypropylene and polystyrene capacitors are superior [63, page 7-9]. In the experiment, polyester capacitors were used. Figure 7.8 shows a schematic of the sample and hold circuit used with each channel. The LF398A sample and holds are powered by +12 V regulators (7812) and -12 V regulators (7912).

7.3.4 Channel Multiplexing

The TMS32020 development system has only one analog-to-digital conversion channel and therefore it is necessary to multiplex the analog channels to the input of the single A/D channel. The values held by the sample and hold circuits are sequentially presented to the A/D converter for conversion to a complemented two's complement.

The software overhead of collecting the channel data by multiplexing multiple channels through a single A/D converter can be minimized through careful design of the multiplexing circuitry. The multiplexing circuitry used in the experiment was designed to incur the minimum software overhead to collect the converted values from the analog channels. The circuitry is shown in Figure 7.9. The analog multiplexer is powered by the positive and negative 12 V regulators used for the sample and hold circuits.

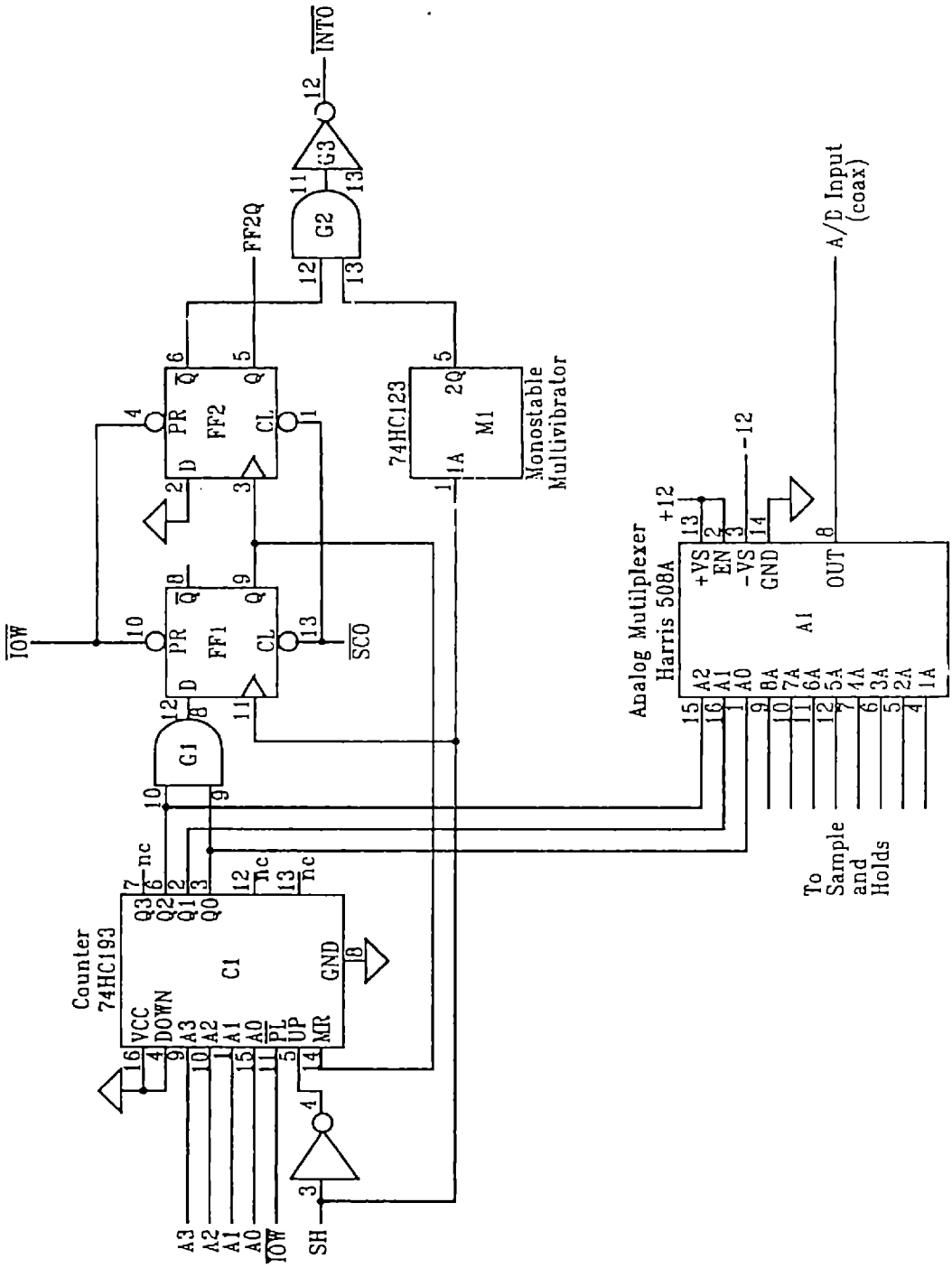


Figure 7.9: Analog channel multiplexing and interrupt circuitry.

Channel Select Logic

The counter C1 holds the current value of the channel to be converted. Its output selects the channel that the analog multiplexer passes to the A/D converter in the TMS32020 development system. The counter increments on the falling edge of signal SH from the A/D converter. Signal SH is set low by the A/D converter when it has completed a conversion. The counter is reset when the true output of FF1 goes high. FF1 goes high when the count reaches binary 5 and a high is clocked into FF1 on the rising edge of SH. Note that the capability to asynchronously load the counter from the address lines is provided by writing to the corresponding TMS32020 output port. This capability is not used.

Interrupt Logic

The function of the interrupt logic is to relieve the TMS32020 of the burden of keeping track of which channel is interrupting and if more channels need to be read. The interrupt logic generates the correct number of interrupts and presents the appropriate channel to A/D converter at the proper times. By placing this functionality in hardware, the interrupt service routines are extremely simple.

The output of the A/D converter in the development system is connected to ports 2 and 3 on the TMS32020. When the output word is read using port 3, another conversion is automatically begun. The A/D converter sets SH high to signal that a conversion is underway. The analog input must not change as long as SH is high.

A counter on the development system is available to the TMS32020 through Port 1. It was used as the sample clock in the experiment. When the counter finishes counting, signal $\overline{\text{SCO}}$ pulses low. The development system was configured (hardware link LK6a) to send this pulse to the TMS32020 as an $\overline{\text{INT1}}$ signal. Therefore, when $\overline{\text{INT1}}$ is unmasked, an interrupt 1 occurs at each counter timeout. The counter is set so that $\overline{\text{SCO}}$ pulses every 200 μs, yielding a 5 kHz sampling rate.

The timing diagram shown in Figure 7.10 greatly aids the explanation of how the interrupt logic in Figure 7.9 operates. Every 200 μs, the sample clock times out and $\overline{\text{SCO}}$ pulses low. The pulse clears FF1 and FF2, which releases the counter through MR, and allows $\overline{\text{INT0}}$ interrupt signals through gate G2. Clearing FF2 also signals the sample and hold circuits to hold their values. On the rising edge of $\overline{\text{SCO}}$, a conversion is begun by the A/D converter. A conversion takes 17 μs.

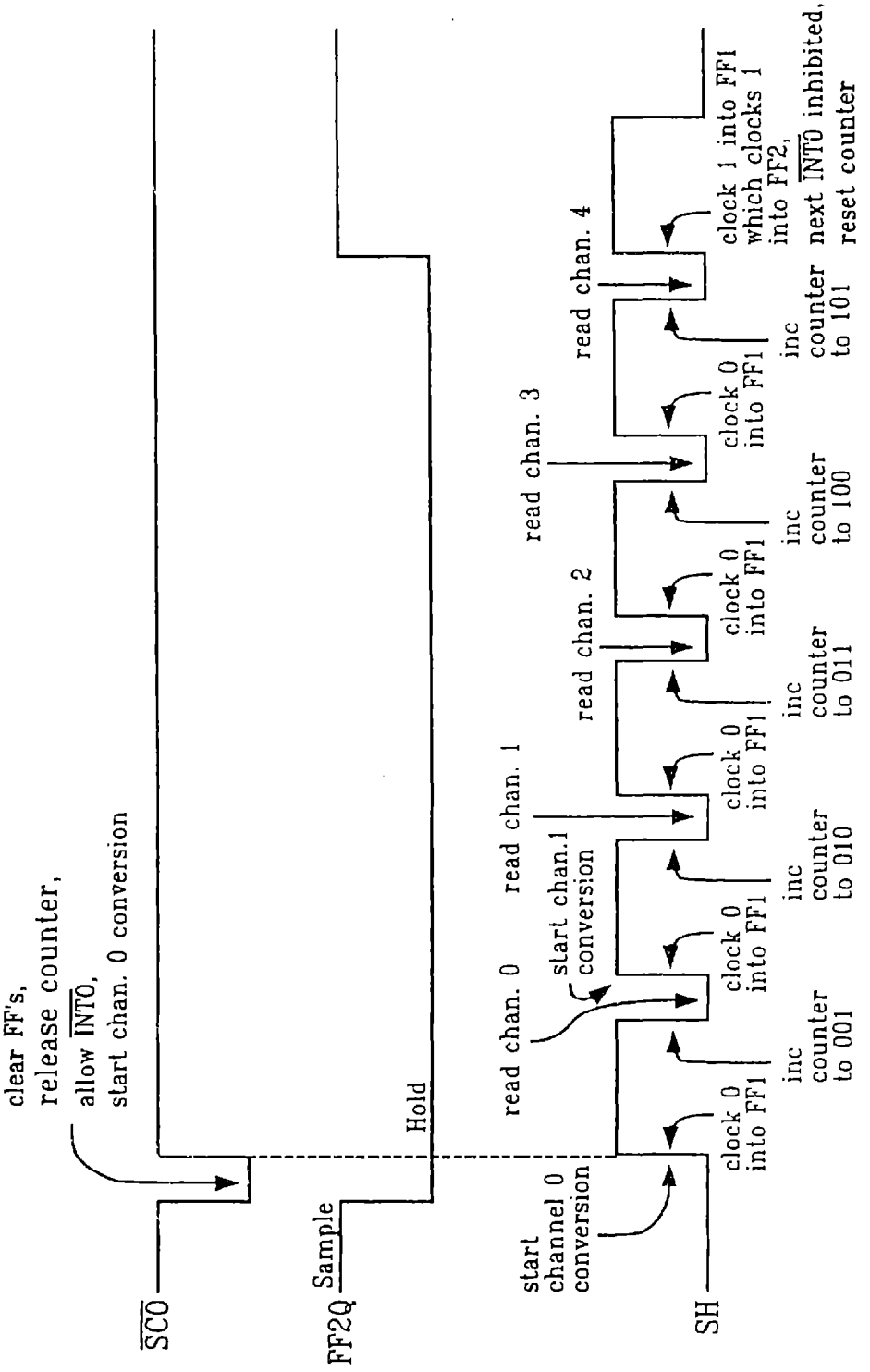


Figure 7.10: Interrupt logic timing diagram.

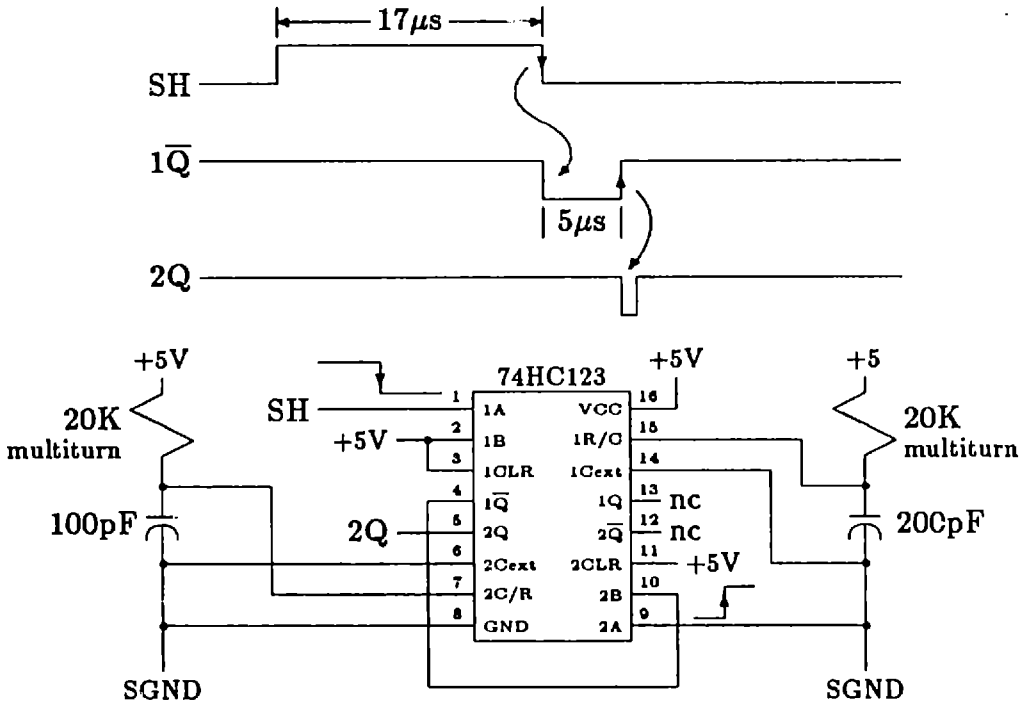


Figure 7.11: Monostable vibrator circuit for generating a 5 μs delay.

When the conversion is complete, SH falls and the channel counter is incremented. Approximately 5 μs later the output of the monostable vibrator M1 falls and the signal $\overline{\text{INT0}}$ causes an interrupt 0 on the TMS32020. The 5 μs delay is necessary to give the analog multiplexer time to stabilize. The monostable vibrator circuit is shown in Figure 7.11.

The service routine for interrupt 0 reads the conversion result from the A/D converter through port 3, and thus another conversion begins immediately. This process repeats until five channels have been read (0–4) and the counter reaches 5. A high is then present on the input to FF1. When the results of the channel 4 conversion are read, a conversion that will be ignored begins, and a high is clocked into FF1. When FF1Q goes high the counter is reset and a high is clocked into FF2. When FF2Q goes high, the sample and holds are released and any further $\overline{\text{INT0}}$ signals are blocked by gate G2. The system is now ready to repeat the sequence when the sample clock times out.

```
*************** INTERRUPT SERVICE ROUTINE 0 ***************

* AR4 points to the current channel storage bin.
* Page 0 assumed.
I0SERV  SACL   TEMPA     Save accumulator.
        SST    TEMPS     Save status register 0.
        LARP   4         Select pointer to channel bins.
        IN     *,3       Store data from A/D converter.
        LAC    *         Retrieve A/D data.
        NEG              A/D data is complemented 2's comp.
        SACL   *+        Store and point to next bin.
        LST    TEMPS     Restore status register 0.
        LAC    TEMPA     Restore accumulator.
        EINT             Enable interrupt.
        RET              Return.

*************** INTERRUPT SERVICE ROUTINE 1 ***************

I1SERV  LAR    AR4,STARTD  Load pointer to base of bins.
        EINT               Next interrupt will be an INT0'.
        RET                Return.
```

Figure 7.12: Interrupt service routines.

Interrupt Service Routines

It is critical that the TMS32020 spend as little time as possible servicing the interrupts to collect data from the analog channels. The main program that steps the observer forward in real-time requires most of each sample period to execute.

A data collection program may be found in Appendix E which illustrates the use of the TMS32020 with the interrupt logic of Figure 7.9. Example interrupt service routines are shown in Figure 7.12. Interrupt 1 occurs when the sample clock times out. The service routine sets auxiliary register AR4 to point to the address of the first channel storage bin. Approximately 21 μs later, the first $\overline{\text{INT0}}$ occurs. The interrupt 0 service routine uses the accumulator and the auxiliary register pointer ARP and therefore these registers must be saved. The data is fetched from port 3

and inserted into the current channel storage bin. The data is then negated since the output of the A/D converter is complemented two's complement. When the data is stored, AR4 is incremented so that it is pointing to the next storage bin when the interrupt 0 occurs. The hardware ensures that the correct number of such interrupts occur.

The interrupt service routines listed in Figure 7.12 are extremely fast. Every instruction except the port input instruction, IN, and the return instruction, RET, execute in a single cycle. The IN instruction requires two machine cycles plus a wait cycle, and the RET instruction requires two machine cycles. Interrupt service begins approximately 400 *ns* after the reception of an interrupt.[2] Therefore, service for interrupt 0 consumes 3.2 μs and service for interrupt 1 consumes 1.2 μs. Collecting the data from 5 analog channels thus consumes only 17.2 μs of processor time, which is 9% of the period at 5 *k*Hz.

The initialization software for starting the interrupt sequence described in this section is somewhat subtle and a listing of it may be found in Appendix E. The difficulty lies in the fact that the sample clock continuously produces $\overline{\text{INT1}}$ signals which are latched by the TMS32020. When interrupt 1 is first enabled, a undesired interrupt 1 will be serviced. The solution is to "clean" out this interrupt 1, along with any pending interrupt 0, by vectoring to a dummy service routine and then repointing the interrupt vectors to the real service routines. See Appendix E, page 228, for details.

7.3.5 Resolver Buffering and Read Select Logic

In order to verify the performance of the observer, it is necessary to read the true position of the shaft using the resolver. The resolver-to-digital converter used in the Industrial Drives system is an Analog Devices 1S24, which is a 12-bit converter suitable for speeds up to 10200 rpm. The digital output of this device is read through port 4 during the service for interrupt zero. The resolver buffering and read select logic are shown in Figure 7.13.

It is extremely important that the output drivers on the R/D converter not be damaged as the controller will destroy expensive power transistors if it receives erroneous position information from the resolver. Therefore, buffers were constructed

[2]Interrupt latency increases if the TMS32020 is executing a multicycle instruction when it is interrupted.

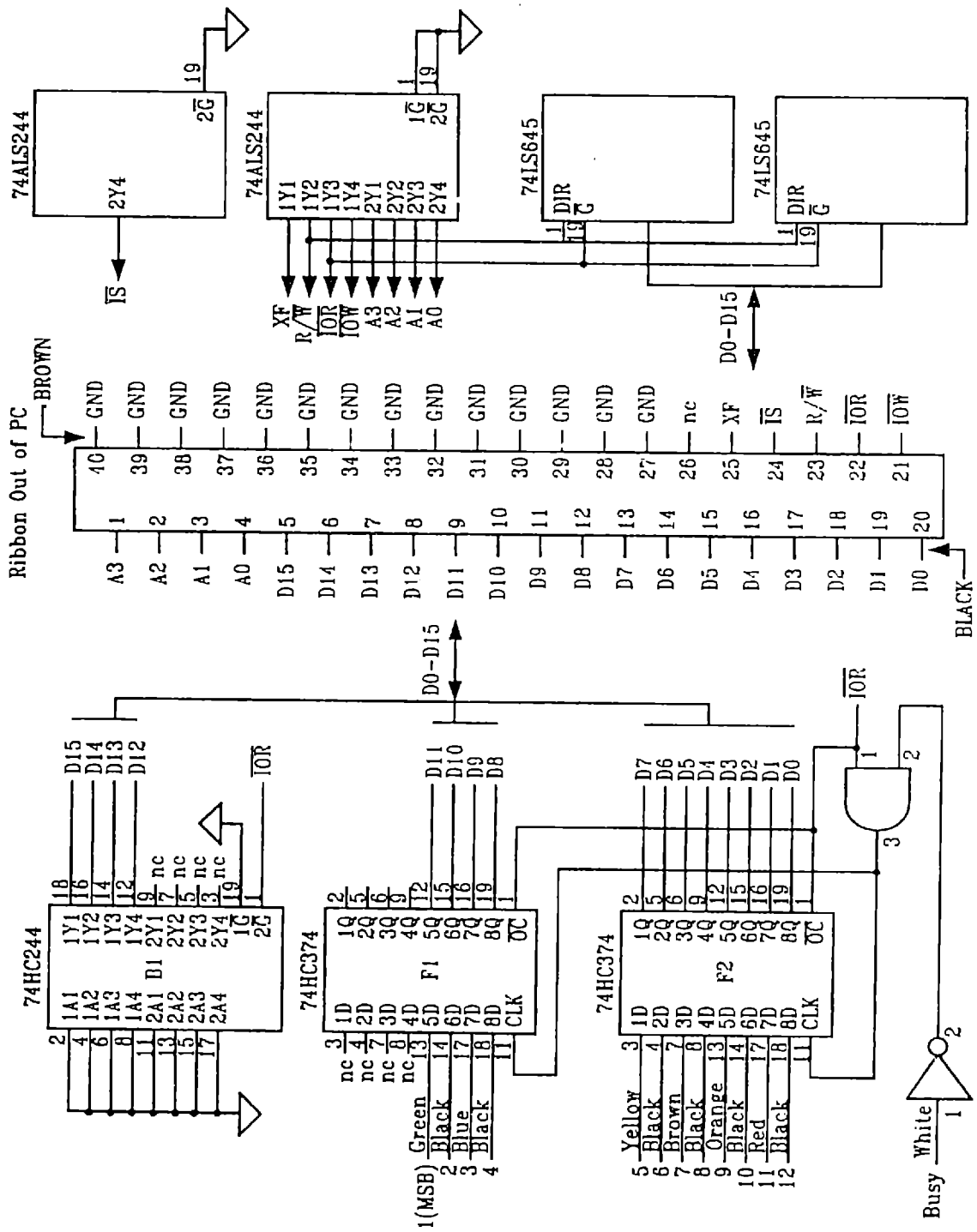


Figure 7.13: Resolver buffering and read select logic.

on a small circuit board and mounted next to the R/D converter for maximum protection. The 12 bits of resolver output are captured by F1 and F2 and placed on the lower 12 bits of the TMS32020 data bus. The remaining upper 4 bits of the bus are driven zero by buffer B1. A new position is available at each trailing edge of the BUSY signal from the R/D converter. Flip-flops F1 and F2, which capture the position information, are clocked by the BUSY signal. When reading the flip-flops, the clocking of data is suspended by overriding the BUSY signal with $\overline{\text{IOR}}$ from the TMS32020 development system.

Chapter 8

Experiment Calibration

The observers constructed in this research are all identity observers. The states of the observer correspond precisely to the physical states of the motor; the transformation between the motor and observer states is the identity matrix. The parameters in the observer, therefore, correspond directly with physical values that can be measured. In particular, the stator resistance, the stator inductance, the magnet constant, the viscous damping constant, the rotor inertia constant, the coulomb friction constant, and the load torque must be measured. In this chapter, the methods used to measure these parameters for one particular system are documented. The methods of current and voltage sensor calibration are also presented.

8.1 Parameter Measurements

8.1.1 Electrical Parameters

Since the neutral on most machines is not externally available, it is common practice to make line-to-line (l-l) measurements as shown in Figure 8.1a; the third phase is open. For a three-phase machine this yields three measurements, namely, a-to-b (ab), b-to-c (bc), and c-to-a (ca). Another three measurements may be obtained by making line-to-line-line (l-ll) measurements as shown in Figure 8.1b. The measurements are a-to-bc ($a\widehat{bc}$), b-to-ca ($b\widehat{ca}$), and c-to-ab ($c\widehat{ab}$).

Assuming a locked rotor ($\omega = 0$), the circuit for l-l measurements in Figure 8.1a yields

$$v = 2Ri + 2(\mathcal{L} + M)\frac{di}{dt} \tag{8.1}$$

where R is the three-phase resistance, $\mathcal{L}$ is the three-phase self-inductance, and

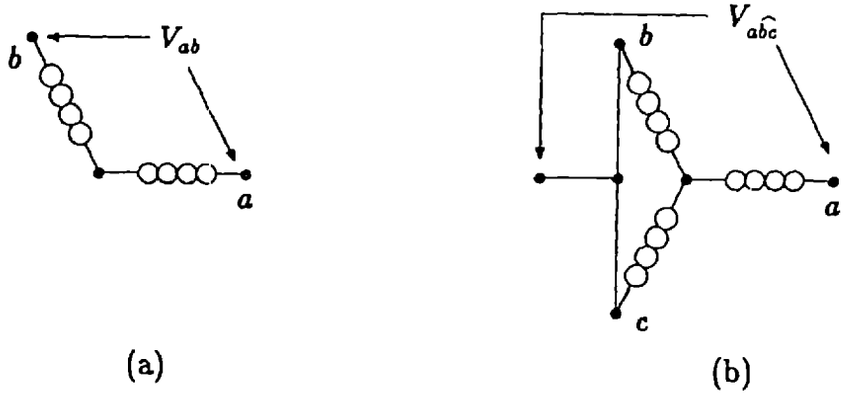


Figure 8.1: Stator measurements with wye connection and isolated neutral.

M is the mutual inductance between phases. The circuit for *l-ll* measurements in Figure 8.1b yields

$$v = \frac{3}{2}Ri + \frac{3}{2}(\mathcal{L} + M)\frac{di}{dt}. \tag{8.2}$$

Equations 8.1–8.2 are not sufficient to determine R, $\mathcal{L}$, and M.

Two-Phase Parameters

Line-to-line and line-to-line-line measurements measure three-phase stator inductance and resistance. The observers in Chapter 4 used two-phase models of the motor. Therefore, it is necessary to relate the three-phase parameters, which are measured, to the two-phase parameters. A wye-connected stator allows the use of a two-phase model via the $\alpha\beta 0$ transformation described in Section 2.1. Including the rotor flux, the three-phase model of a balanced stator is (see Fitzgerald, et al. [64])

$$\frac{d}{dt}\left\{\begin{bmatrix} \mathcal{L} & -M & -M & M_r\cos(N\theta) \\ -M & \mathcal{L} & -M & M_r\cos(N\theta - \frac{2\pi}{3}) \\ -M & -M & \mathcal{L} & M_r\cos(N\theta + \frac{2\pi}{3}) \end{bmatrix}\begin{bmatrix} i_a \\ i_b \\ i_c \\ i_r \end{bmatrix}\right\} = -\begin{bmatrix} R & 0 & 0 \\ 0 & R & 0 \\ 0 & 0 & R \end{bmatrix}\begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix}. \tag{8.3}$$

The rotor electrical dynamics are now ignored following the discussion in Section 2.4 and i_r is set to a constant so that $K = M_r i_r$. Distributing the derivative on the left yields

$$\begin{bmatrix} \mathcal{L} & -M & -M \\ -M & \mathcal{L} & -M \\ -M & -M & \mathcal{L} \end{bmatrix} \frac{d}{dt} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} = - \begin{bmatrix} R & 0 & 0 \\ 0 & R & 0 \\ 0 & 0 & R \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + KN\omega \begin{bmatrix} \sin(N\theta) \\ \sin(N\theta - \frac{2\pi}{3}) \\ \sin(N\theta + \frac{2\pi}{3}) \end{bmatrix} + \begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix}. \tag{8.4}$$

More compactly,

$$\mathcal{L} \frac{d\underline{i}_3}{dt} = -R\underline{i}_3 + KN\omega \begin{bmatrix} \sin(N\theta) \\ \sin(N\theta - \frac{2\pi}{3}) \\ \sin(N\theta + \frac{2\pi}{3}) \end{bmatrix} + \underline{v}_3 \tag{8.5}$$

where $\mathcal{L}$ is the inductance matrix, $\underline{i}_3$ is the vector of currents, and $\underline{v}_3$ is the vector of voltages in (8.4). Applying the $\alpha\beta 0$ transformation gives

$$\mathbf{C}\mathcal{L}\mathbf{C}^{-1} \frac{d\underline{i}_2}{dt} = -R\underline{i}_2 + \mathbf{C}KN\omega \begin{bmatrix} \sin(N\theta) \\ \sin(N\theta - \frac{2\pi}{3}) \\ \sin(N\theta + \frac{2\pi}{3}) \end{bmatrix} + \underline{v}_2 . \tag{8.6}$$

Computing the matrices and dropping the zero sequence terms and equation gives

$$\begin{bmatrix} \mathcal{L}+M & 0 \\ 0 & \mathcal{L}+M \end{bmatrix} \frac{d}{dt} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = - \begin{bmatrix} R & 0 \\ 0 & R \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \sqrt{\frac{3}{2}} KN\omega \begin{bmatrix} \sin(N\theta) \\ -\cos(N\theta) \end{bmatrix} + \begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix}. \tag{8.7}$$

Note that the inductance matrix in (8.7) involves terms $\mathcal{L} + M$. Therefore, it is not necessary to determine $\mathcal{L}$ and M uniquely, but rather only $\mathcal{L} + M$, which is precisely what *l-l* and *l-ll* measurements yield. Three-phase and two-phase resistances are equivalent. If the phases are balanced, then the required two-phase resistance and inductance may be determined with a single *l-l* or *l-ll* measurement. If the phases are not balanced, then it may be necessary to utilize all six available measurements. Finally, the two-phase magnet constant K is seen to be $\sqrt{(3/2)}$ times the three-phase magnet constant K.

Stator Resistance

Stator resistance was easily measured using an ohmmeter. Room temperature measurements of the Industrial Drives motor taken at the BDS3 connection block yielded

ab:	0.75Ω
bc:	0.75Ω
ca:	0.75Ω
abc:	0.59Ω
bca:	0.59Ω
cab:	0.59Ω

from which the parameter R was determined to be 0.38 Ω. When the motor was hot, *l-l* and *l-ll* resistance measurements yielded 0.80 Ω and 0.62 Ω, respectively, which yields a hot resistance of approximately 0.41 Ω, an 8% change. Although the resistance shows some sensitivity to temperature, it was found in Section 5.2.2 that the observer is relatively insensitive to errors in the resistance parameter and therefore R will be modeled as a time-invariant constant with a value of

$$\boxed{R = 0.39\ \Omega.}$$

Stator Inductance

Inductance measurements of the Industrial Drives motor were performed using a GenRad 1657 RLC Digibridge. Inductive core losses were determined to not be dominant since measurements of inductance using a series impedance bridge were more nearly in agreement at 100Hz and 1kHz than measurements of inductance using a parallel impedance bridge. Therefore, all measurements were taken in the series configuration. The Q was less than one at 100Hz and greater than two at 1kHz, and thus all measurements were taken at 1kHz.

It was assumed while developing the motor model in Chapter 2 that the rotor is magnetically smooth, that is, the rotor presents a constant permeability to each stator phase. In order to verify that the experimental motor has a magnetically smooth rotor, inductance measurements were taken for shaft positions from 0° to 360° at precise 10° increments. Six measurements were taken at each shaft position: *ab*, *bc*, *ca*, $\widehat{abc}$, $\widehat{bca}$, and $\widehat{cab}$. The results are shown in Figures 8.2 and 8.3.

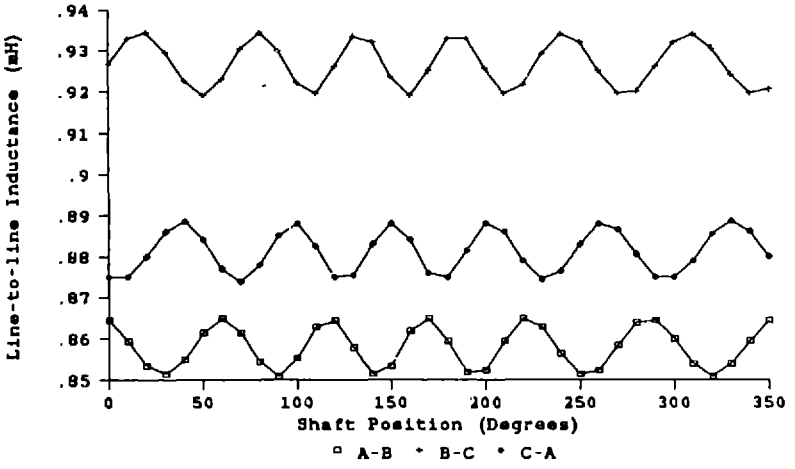


Figure 8.2: Measured line-to-line stator inductance as a function of shaft position.

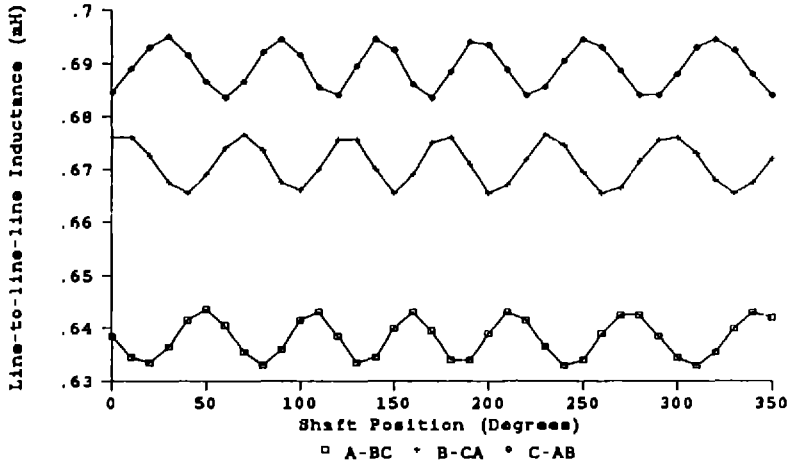


Figure 8.3: Measured line-to-line-line stator inductance as a function of shaft position.

There is a ripple in the measured inductance at six times the angular frequency of the fundamental mechanical speed. This ripple is caused by the six permanent magnets glued on the rotor. The permeability of the permanent magnet material is not exactly the same as the air and glue in the spaces between the permanent magnets. Therefore, the inductance changes as the permanent magnets pass under the stator. This difference of permeability is enhanced by the fact that with six evenly spaced rotor magnets and three evenly spaced stator phases, the same proportion of a magnet is under all three phases. Therefore, when making line-to-line measurements, the magnitude of the ripple seen is doubled. When making line-to-line-line, the magnitude of the ripple is magnified by 3/2. The data in Figures 8.2 and 8.3 exhibits this property. Within the set of *l-l* measurements, and within the set of *l-ll* measurements, the ripple is approximately constant. Consider, for example, the *ab* measurements. The magnitude of the measured ripple is 0.014 mH and therefore the true ripple is 0.007 mH, which is 0.82% of the average value of 0.8518 mH. In Chapter 9 it will be seen that there is an error in the observer position estimate at six times the mechanical frequency which is due to not modeling the inductance ripple. Modeling the ripple is considered in Section 8.3.

The data in Figures 8.2 and 8.3 reveals that the motor is also not electrically balanced. In a balanced motor, the *l-l* measurements would yield identical average inductances. The same may be said for *l-ll* measurements. The stator is probably not the source of the imbalance since it is wound by a machine and there are most likely precisely the same number of turns on each phase wound in identical patterns. A more likely cause of the imbalance is that the rotor is not centered in the stator. The rotor bearings in the Industrial Drives motor are not adjustable but rather pressed into a machined socket on the stator case. The stator is manufactured to tolerances such that the stator and rotor do not strike each other. Since the effective air-gap in a surface-mounted permanent-magnet motor is large, the physical air-gap is also allowed to be large and thus there is some room for the rotor to be nonconcentric with the stator. If the motor meets the required operating specifications with a nonconcentric rotor, then the manufacturing expense required to machine the stator case to tighter tolerances may not be justified. Not modeling the electrical imbalance caused by a nonconcentric rotor shaft will result in an error in the observer position estimate at the electrical frequency. An unbalanced model is developed in Section 8.4.

The average of the *l-l* and *l-ll* measurements yields the two-phase inductance

$$\boxed{L_s = \mathcal{L} + M = 0.444 \ mH.}$$

Magnet Constant

The magnet constant K is a measure of the strength of the magnets on the rotor and of the magnetic coupling between the rotor and the stator. The torque produced by the motor is directly proportional to the magnitude of K since the rotor is turned by the coupling of the magnetic fields of the stator with the magnetic fields of the rotor.

The magnet constant is easily measured by spinning the permanent-magnet motor with another motor and measuring the open-circuit voltage induced at the stator terminals. If the stator is sinusoidally wound, the induced voltages will be sinusoids at the electrical frequency of the motor. If the motor is balanced then the magnitudes of the three phase voltages will be equal and the phase angle between the voltages will be 120°.

The Industrial Drives motor was back-driven by a DC motor (see Figure 7.1) and the induced voltages were found to be very close to sinusoidal. The virtual lack of nonfundamental harmonics (third, fifth, seventh, etc.) is a result of the slotless construction of the stator as described in Section 7.1.1. Unfortunately, the magnitudes of the waveforms were not identical and they were not shifted by precisely 120°. The motor is not electrically balanced and this is due to the nonconcentric rotor discussed in the previous section. A model for a motor with a nonconcentric rotor is developed in Section 8.4.

The three-phase magnet constant is easily computed from (8.5) by utilizing the fact that the stator currents are zero in the open-circuit test. Assuming *l-l* measurements, the result is

$$K = \frac{|V_{l-l}|}{\sqrt{3}N\omega}. \tag{8.8}$$

Therefore, the two-phase magnet constant from motor measurements is

$$K = \frac{|V_{l-l}|}{\sqrt{2}N\omega}. \tag{8.9}$$

Line-to-line measurements were taken for 400 rpm to 2000 rpm in steps of 100 rpm. The data is shown in Figure 8.4. The average value of K is computed to be

$$\boxed{K = 0.1105 \text{ V·s.}}$$

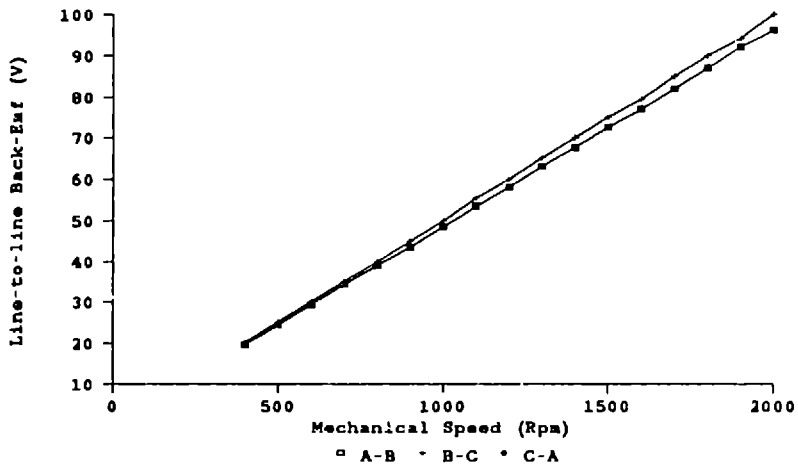


Figure 8.4: Measured line-to-line speed voltage as a function of speed.

The magnet constant was also measured when the motor was hot and the constant was found to have decreased by approximately 5%. Simulations in Section 5.2.2 showed that the observer is sensitive to variations in the magnet constant and therefore, for optimum performance over the operating range of the motor, an adaptive mechanism for updating the magnet constant in real-time would be most useful.

8.1.2 Mechanical Parameters

The mechanical dynamics were described by (2.65), which is repeated here as

$$\frac{d\omega}{dt} = -\frac{B}{H}\omega + \frac{KN}{H}\underline{i}_s^T \exp(\mathbf{J}N\theta)\begin{bmatrix}1\\0\end{bmatrix} - \frac{C}{H}\frac{\omega}{|\omega|} - \frac{1}{H}\tau\,. \tag{8.10}$$

The viscous damping coefficient B, the rotor inertia H, the coulomb friction constant C, and the load torque τ need to be determined. The mechanical parameters are not determined by separate tests but are instead determined simultaneously, as will be described below.

The second term on the right in (8.10) describes the electromagnetic torque produced by the motor. The current vector $\underline{i}_s$ is a state variable and thus its magnitude and angle are not a priori known. Therefore, it is desirable to perform tests where this term is not present. The first such test is a spin-down test. The

permanent magnet motor is coupled to the DC motor as shown in Figure 7.1. The DC motor is unpowered and open-circuited and the permanent-magnet motor is used to spin the two at a constant speed. The permanent-magnet is suddenly inhibited and the position of the rotor shaft is recorded while the two motors coast to a stop. The mechanical dynamics of this process are

$$\frac{d\omega}{dt} = -\frac{B}{H}\omega - \frac{C}{H}\frac{\omega}{|\omega|} \tag{8.11}$$

which has a solution

$$\theta(t) = \frac{H}{B}\left(\omega(0) + \frac{C}{B}\right)\left(1 - \exp(-\frac{B}{H}t)\right) - \frac{C}{B}t\,. \tag{8.12}$$

for $\omega(t) > 0$. Define

$$a[0] = \frac{B}{H} \tag{8.13}$$

$$a[1] = \frac{C}{B}\,. \tag{8.14}$$

Parameters $a[0]$ and $a[1]$ are determined by numerically fitting the spin-down data of position θ versus time t.

The spin-down data was obtained using the TMS32020 development system and the circuit shown in Figure 8.5. When the TMS32020 signal XF is set high, the inhibit signal INH into the Industrial Drives controller is pulled high which inhibits the gate signals to the power transistors, thereby inhibiting the inverter from driving the motor (see Section 7.1.3) and the motor stator is open circuited. The resolver-to-digital output from the Industrial Drives controller is then read and stored using the data collection routine listed in Appendix E as the coupled motors coast to a stop.

Equations (8.13) and (8.14) are not sufficient to uniquely determine the constants B, H, and C. Another test is needed. Consider spinning the two motors at a constant speed using the DC motor. From (8.10), the mechanical dynamics are thus

$$\tau_{DC} = B\omega + C\,. \tag{8.15}$$

Substituting $a[1]$ yields an expression for B, namely,

$$B = \frac{\tau_{DC}}{\omega + a[1]}\,. \tag{8.16}$$

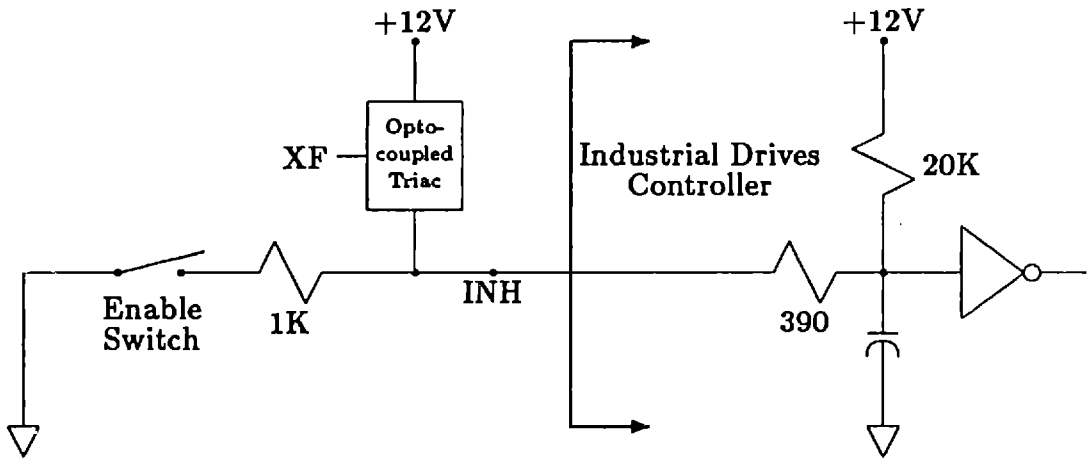


Figure 8.5: Inhibit override circuit. This ciruit allows the TMS32020 output line XF to override the inhibit switch and disable the Industrial Drives motor.

With B determined, H and C are determined from (8.13) and (8.14), respectively,

$$H = \frac{B}{a[0]} \tag{8.17}$$

$$C = Ba[1]\,. \tag{8.18}$$

The rotor speed in (8.16) is easily measured. Only the torque being produced by the DC motor is yet to be determined.

The torque produced by a DC motor is

$$\tau_{DC} = \frac{f(I_f,\omega)I_a}{\omega}\,, \tag{8.19}$$

where $f(I_f,\omega)$ is the armature speed voltage. It is difficult to get an accurate measure of the armature resistance when the DC motor is driving the permanent-magnet motor because the resistance seen at the terminals of the machine varies according to how the brushes are riding on the commutator. Therefore, $f(I_f,\omega)$ is determined as the speed voltage produced for a given field current at a given speed. Due to saturation of the iron in the motor, this speed voltage is a nonlinear function of field current and speed. Normalizing the speed voltage by the speed, (8.19) says that the torque produced by the DC motor is thus equal to the field flux times the armature current.

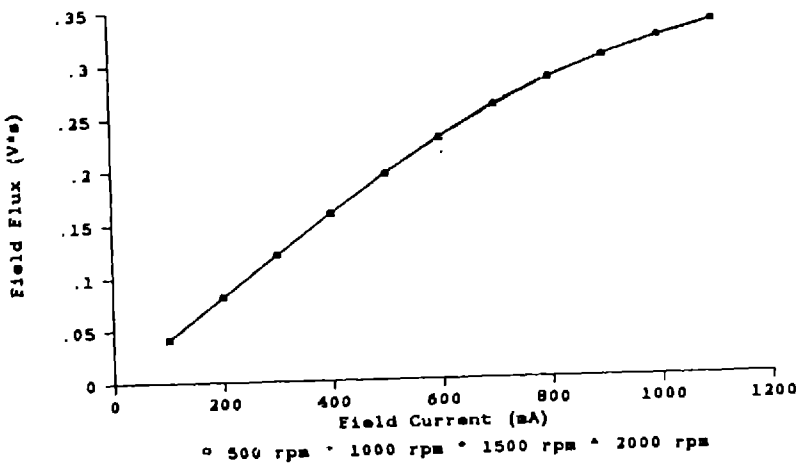


Figure 8.6: DC motor field flux as a function of field current.

The Industrial Drives motor was used to drive the DC motor at a constant speed while the field current was varied and the speed voltage at the armature terminals was measured. Plots of $f(I_f,\omega)/\omega$ for four different constant speeds are shown in Figure 8.6. The curves are virtually identical over a wide range of speed. When the DC motor is supplied with both a field and armature current then the torque can be determined using the flux curves to determine the flux at a specific field current and then multiplying the flux obtained from the curve by the measured armature current, as in (8.19).

Three spin-down tests were conducted. Position data was collected and fitted for each test. The results of these tests are

Initial Speed (rpm)	a[0] (s^{-1})	a[1] (s^{-1})
1000	0.1044	170.3
1500	0.1042	156.1
2000	0.1042	146.5
Average	0.1043	157.6

Three constant-speed permanent-magnet motor tests were conducted. The field and armature currents were measured and the DC motor torque was computed using the flux curves in Figure 8.6 and equation (8.19). The viscous damping coefficient B was then computed using (8.16), the load inertia H using (8.17), and the coulomb friction coefficient C using (8.18). The results of these tests are

Speed ω (rpm)	$I_f(A)$	$I_a(A)$	τ_{DC} (N·m)	B (N·m·s)	J (N·m·s^2)	C (N·m)
1000	0.320	3.19	1.02	0.00389	0.0373	0.613
1500	0.320	3.62	1.16	0.00368	0.0355	0.580
2000	0.320	4.12	1.32	0.00359	0.0344	0.566
Average				0.00370	0.0355	0.583

The DC motor load torque τ in (8.10) must be computed for each experiment. For example, assume a load torque of 1.6 N·m is desired. Set the field current at 1 A and use the flux curves in Figure 8.6 to find $f(I_f,\omega)/\omega$. Then, using (8.19),

$$I_a = \frac{1.6}{0.320} = 5.0 \text{ A}. \tag{8.20}$$

Therefore, at the constant speed of the experiment, if the armature resistance is adjusted so that the armature current it 5 A then the DC motor will supply the desired torque.

8.2 Sensor Calibration

The observer utilizes terminal measurements of stator phase voltages impressed on the motor and the resulting stator currents. The hardware used to measure these variables was described in Chapter 7. The meaning of the actual data received by the TMS32020 still needs to be determined.

The A/D converter on the development system is a 16-bit converter. However, the interior of a PC is an electromagnetically noisy environment, and values produced by the A/D converter are corrupted by noise. When the inputs to the anti-aliasing filters are grounded (see Figure 7.3), the values produced by the A/D converter are found to be corrupted with approximately 10 least-significant bits of noise. A sample of this noise is shown in Figure 8.7 where the two's-complement words from the A/D converter are scaled from -10 V to +10 V. The DC offset in each channel is easily corrected in software.

8.2.1 Current Sensors

Analog channels 3 and 4 were used to measure stator currents i_a and i_c. The sensors were calibrated by running known amounts of current through the toroids in the BDS3 (see Figure 7.4). A wire was threaded through the toroids and connected in series with a precision 1 Ω power resistor. The resistor was found to hold its

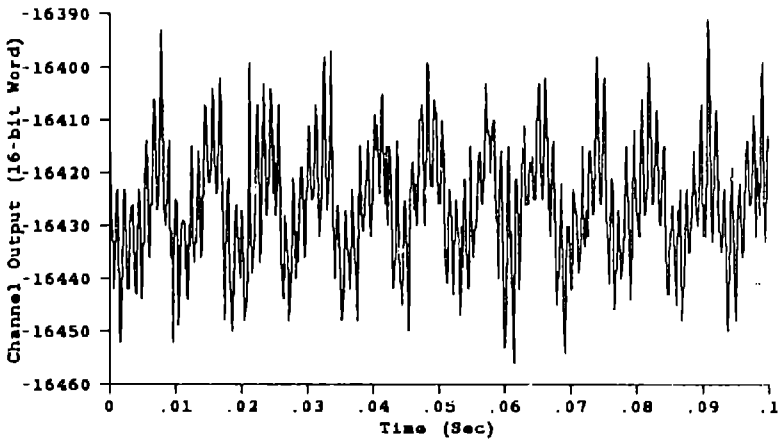


Figure 8.7: Analog channel 0 noise with the channel input grounded.

value over a wide temperature range. Data was taken for currents from -14 A to +14 A and the current sensors were found have a good linear response over this range. A positive voltage is registered for currents coming *out* of the motor. The model developed in Chapter 2 assumed that positive currents flow *in* to the motor. Therefore, the linear calibration function must have a negative slope.

8.2.2 Voltage Sensors

The voltage sensors were calibrated by applying a known voltage to the input of the filters (see Figure 7.3). As mentioned in Section 7.3.1, the voltage sensor response is nonlinear due to a dependence of the offset voltage on the output voltage. A typical response is shown in Figure 8.8. The bits deviation of the measurements from a best-fit line are plotted against input voltage. The nonlinearity was repeatable and is therefore corrected in software. A polynomial was fit to the error curve and this function is used to subtract the error.

The dynamic performance of the voltage sensors is difficult to judge since the desired fundamental voltage is embedded in pulse width modulation. An experiment was conducted to test the ability of the voltage sensors to faithfully reconstruct the fundamental harmonic from the pulse-width-modulated signal. A simple PWM circuit was constructed that operated over the range of -10 V to +10 V. The pre-

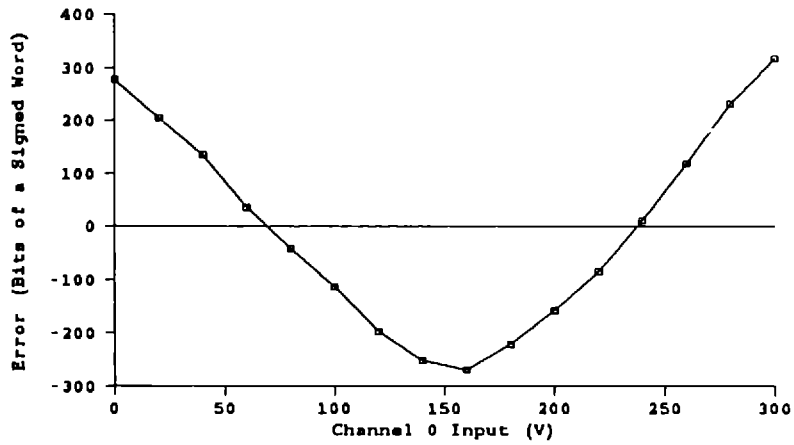


Figure 8.8: Voltage sensor nonlinearity. Difference between measured voltage and best-fit line expressed in bits.

cision resistor divider in the input to an isolation amplifier was bypassed and a sine wave, modulated at 8 *k*Hz, applied to the input. The value read from the A/D converter was immediately output through the D/A converter of the development system. Comparison of the D/A output with the applied input revealed a faithful reconstruction. The results of this test and the use of precision resistors in the divider network give confidence that the sensors faithfully reconstruct the fundamental voltage applied to the motor.

8.2.3 Resolver Output

A resolver is mounted on the Industrial Drives motor to provide position information for the controller. The output of this resolver was compared with the observer position estimate in order to measure of the performance of the observer.

The 12-bit resolver output is placed on the low 12 bits of the TMS32020 bus and read using I/O port 4 (see Figure 7.13). The model developed in Chapter 2 assumes that the zero position is when the rotor mmf vector within an electrical cycle is aligned with the mmf vector of the *a*-phase of the stator. The resolver reads $3F3_{16}$, 940_{16}, or $EA5_{16}$ under these conditions, corresponding to the three electrical cycles of the motor. Additionally, the model assumes that rotation from phase *a* to

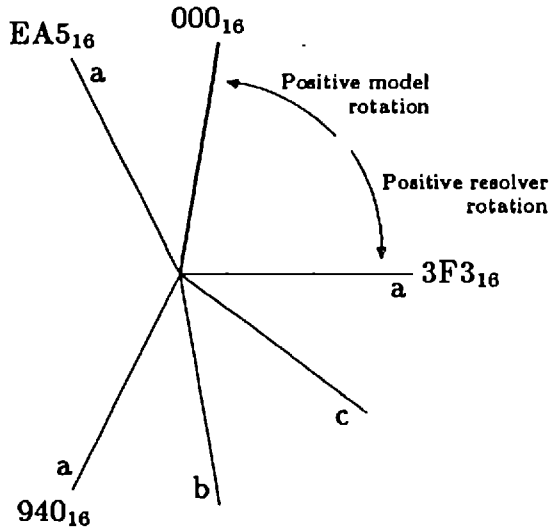


Figure 8.9: Resolver offset. One mechanical revolution is shown along with the corresponding resolver readings around the perimeter. Note that positive rotation for the resolver is from phase *a* to phase *c* whereas the model developed in Chapter 2 assumes positive rotation to be from phase *a* to *b*.

phase *b* is positive rotation. The output of the resolver assumes that the rotation from phase *a* to phase *c* is positive rotation. Therefore, the resolver output must be both shifted and reversed. A single electrical cycle is illustrated in Figure 8.9. The resolver readings at the three equilibrium positions were repeatable to 12 bits. The offsets from phase *a* at the three equilibrium positions are 88.86°, 88.13°, and 89.50°. All three offsets should be the same and the fact that they are not means that either the stator phase axes are not separated by precisely 120° or that the resolver is not reading the rotor position accurately. The accuracy of the resolver will be considered in Chapter 9.

8.3 Modeling Inductance Ripple

The model developed in Chapter 2 assumed that the stator inductance is independent of shaft position. As a result, the electromagnetic torque on the rotor is linearly proportional to the quadrature current. In Section 8.1.1 it was found

that the stator inductance in the experimental system was not independent of shaft position. Line-to-line measurements revealed a ripple in the inductance at twice the electrical frequency due to differences between the permeability of the magnets glued to the surface of the rotor and the air space between the magnets.

The performance of the estimated-innovation observer depends on the accuracy of the underlying motor model. For maximum performance, the motor model should include the observed ripple in the inductance. Including the inductance ripple leads to additional terms is the electrical subdynamics and an additional mechanism for the generation of torque. The new mechanism for the generation of torque is nonlinear with respect to the quadrature current and is a result of the varying reluctance of the rotor.

Consider the model of a three-phase system where ripple in the inductance matrix has been added

$$\frac{d}{dt}\left\{\begin{bmatrix} \mathcal{L}+P\cos(2N\theta) & -M-P\cos(2N\theta+\frac{\pi}{3}) & -M-P\cos(2N\theta-\frac{\pi}{3}) \\ -M-P\cos(2N\theta+\frac{\pi}{3}) & \mathcal{L}+P\cos(2N\theta+\frac{2\pi}{3}) & -M-P\cos(2N\theta-\pi) \\ -M-P\cos(2N\theta-\frac{\pi}{3}) & -M-P\cos(2N\theta-\pi) & \mathcal{L}+P\cos(2N\theta-\frac{2\pi}{3}) \end{bmatrix}\begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix}\right\} = -R\underline{i}_3 + KN\omega\begin{bmatrix} \sin(N\theta) \\ \sin(N\theta-\frac{2\pi}{3}) \\ \sin(N\theta+\frac{2\pi}{3}) \end{bmatrix} + \underline{v}_3 \tag{8.21}$$

$$\begin{aligned}\frac{d\omega}{dt} = & -\frac{B}{H}\omega - \frac{KN}{H}\begin{bmatrix}\sin(N\theta) & \sin(N\theta-\frac{2\pi}{3}) & \sin(N\theta+\frac{2\pi}{3})\end{bmatrix}\underline{i}_3 \\ & -\frac{PN}{H}\underline{i}_3^T\begin{bmatrix} \sin(2N\theta) & -\sin(2N\theta+\frac{\pi}{3}) & -\sin(2N\theta-\frac{\pi}{3}) \\ -\sin(2N\theta+\frac{\pi}{3}) & \sin(2N\theta+\frac{2\pi}{3}) & -\sin(2N\theta-\pi) \\ -\sin(2N\theta-\frac{\pi}{3}) & -\sin(2N\theta-\pi) & \sin(2N\theta-\frac{2\pi}{3}) \end{bmatrix}\underline{i}_3 \\ & -\frac{C}{H}\frac{\omega}{|\omega|} - \frac{1}{H}\tau\end{aligned} \tag{8.22}$$

$$\frac{d\theta}{dt} = \omega \tag{8.23}$$

where $K = Mi_r$. Using the $\alpha\beta 0$ transformation, the two-phase model is

$$\begin{aligned}\frac{d}{dt}\left\{\begin{bmatrix} \mathcal{L}+M+\frac{3}{2}P\cos(2N\theta) & \frac{3}{2}P\sin(2N\theta) \\ \frac{3}{2}P\sin(2N\theta) & \mathcal{L}+M-\frac{3}{2}P\cos(2N\theta) \end{bmatrix}\underline{i}\right\} \\ = -R\underline{i} + KN\omega\begin{bmatrix}\sin(N\theta) \\ -\cos(N\theta)\end{bmatrix} + \underline{v}\end{aligned} \tag{8.24}$$

$$\frac{d\omega}{dt} = \frac{KN}{H}\begin{bmatrix} -\sin(N\theta) & \cos(N\theta) \end{bmatrix}\underline{i} + \frac{PN}{H}\underline{i}^T\begin{bmatrix} -\sin(2N\theta) & \cos(2N\theta) \\ \cos(2N\theta) & \sin(2N\theta) \end{bmatrix}\underline{i} - \frac{B}{H}\omega - \frac{C}{H}\frac{\omega}{|\omega|} - \frac{1}{H}\tau \tag{8.25}$$

$$\frac{d\theta}{dt} = \omega \tag{8.26}$$

where $K = \sqrt{(3/2)}K$ and $P = (3/2)P$. Expanding the left side of (8.24) and applying the dq transformation

$$\underline{i} = \exp(\mathbf{J}N\theta)\underline{\bar{\imath}} \tag{8.27}$$

yields

$$\begin{bmatrix} L_D & 0 \\ 0 & L_Q \end{bmatrix}\frac{d\underline{\bar{\imath}}}{dt} = -R\underline{\bar{\imath}} + N\omega\begin{bmatrix} 0 & L_Q \\ -L_D & 0 \end{bmatrix}\underline{\bar{\imath}} - KN\omega\begin{bmatrix} 0 \\ 1 \end{bmatrix} + \underline{\bar{v}} \tag{8.28}$$

$$\frac{d\omega}{dt} = \underline{\bar{\imath}}^T\frac{KN}{H}\begin{bmatrix} 0 \\ 1 \end{bmatrix} + \frac{PN}{H}\underline{\bar{\imath}}^T\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\underline{\bar{\imath}} - \frac{B}{H}\omega - \frac{C}{H}\frac{\omega}{|\omega|} - \frac{1}{H}\tau \tag{8.29}$$

$$\frac{d\theta}{dt} = \omega \tag{8.30}$$

where

$$L_D = \mathcal{L} + M + \frac{3}{2}P \tag{8.31}$$

$$L_Q = \mathcal{L} + M - \frac{3}{2}P\,. \tag{8.32}$$

Unlike in the model for a perfectly smooth rotor, (2.72), there is a distinction between the direct and quadrature axis inductance. Additionally, the inductance ripple results in a nonlinear reluctance torque term in (8.29). Using this model, the estimated-innovation observer is thus

$$\frac{d\hat{\underline{\bar{\imath}}}}{dt} = -\left(\begin{bmatrix} \frac{R}{L_D} & 0 \\ 0 & \frac{R}{L_Q} \end{bmatrix} - \mathbf{J}N\hat{\omega}\begin{bmatrix} -\frac{L_D}{L_Q} & 0 \\ 0 & -\frac{L_Q}{L_D} \end{bmatrix}\right)\hat{\underline{\bar{\imath}}} - KN\hat{\omega}\begin{bmatrix} 0 \\ \frac{1}{L_Q} \end{bmatrix} + \begin{bmatrix} \frac{1}{L_D} & 0 \\ 0 & \frac{1}{L_Q} \end{bmatrix}\hat{\underline{\bar{v}}} + \mathbf{G}_{\underline{\bar{\imath}}}(\underline{\bar{\imath}} - \hat{\underline{\bar{\imath}}}) \tag{8.33}$$

$$\frac{d\hat{\omega}}{dt} = -\frac{B}{H}\hat{\omega} + \hat{\underline{i}}^T \frac{KN}{H}\begin{bmatrix} 0 \\ 1 \end{bmatrix} + \frac{PN}{H}\hat{\underline{i}}^T \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \hat{\underline{i}} - \frac{C}{H}\frac{\hat{\omega}}{|\hat{\omega}|} - \frac{1}{H}\tau$$

$$+ \mathbf{G}_\omega(\check{\underline{i}} - \hat{\underline{i}}) \tag{8.34}$$

$$\frac{d\hat{\theta}}{dt} = \hat{\omega} \tag{8.35}$$

This version of the estimated-innovation observer may be applied to permanent-magnet machines where the magnets are buried in the rotor rather than mounted on the surface of the rotor.

With an inductively smooth rotor, it is impossible to determine the position of the shaft when it is at rest. The inductance ripple may be used to determine the initial position of the shaft, although only to within an electrical cycle. In some applications, the fast convergence of the observer may obviate the need to determine the initial position.

8.4 Unbalanced Motor Model

In the previous section, it was shown how to model saliency of the rotor. The stator windings were assumed to be balanced and the rotor centered in the stator. In this section, a model will be developed for an unbalanced motor. An unbalanced motor may be the result of a nonconcentric rotor as described in Section 8.1.1.

Consider a balanced three-phase motor. The inductance matrix for the stator is

$$\mathcal{L} = \begin{bmatrix} \mathcal{L} & -M & -M \\ -M & \mathcal{L} & -M \\ -M & -M & \mathcal{L} \end{bmatrix}. \tag{8.36}$$

The instantaneous energy in the stator phases is

$$\frac{1}{2}\mathcal{L}i_a^2 + \frac{1}{2}\mathcal{L}i_b^2 + \frac{1}{2}\mathcal{L}i_c^2 + Mi_ai_b + Mi_bi_c + Mi_ci_a\,. \tag{8.37}$$

Now model the nonconcentricity of the rotor and the stator by distributing the same amount of energy in an unbalanced stator where the squared number of windings on phase a is modified by the factor μ_a, the squared number of windings on phase b by the factor μ_b, and the squared number of windings on phase c by the factor μ_c.

The self-inductances of each phase are thus modified by the same factors and the inductance matrix is

$$\mathcal{L}' = \begin{bmatrix} \mu_a\mathcal{L} & -M^{ab} & -M^{ac} \\ -M^{ab} & \mu_b\mathcal{L} & -M^{bc} \\ -M^{ac} & -M^{bc} & \mu_c\mathcal{L} \end{bmatrix}. \tag{8.38}$$

The energy in the unbalanced stator is

$$\frac{1}{2}\mu_a\mathcal{L}(i'_a)^2 + \frac{1}{2}\mu_b\mathcal{L}(i'_b)^2 + \frac{1}{2}\mu_c\mathcal{L}(i'_c)^2 + M^{ab}i'_a i'_b + M^{ac}i'_a i'_c + M^{bc}i'_b i'_c. \tag{8.39}$$

Without losing generality, assume the currents in the unbalanced stator to be

$$i'_a = i_a/\sqrt{\mu_a}, \qquad i'_b = i_b/\sqrt{\mu_b}, \qquad i'_c = i_c/\sqrt{\mu_c}, \tag{8.40}$$

such that the self terms in the balanced and unbalanced energy equations are equal. Substituting (8.40) into (8.39) and equating the result with (8.37) yields

$$M^{ab} = \sqrt{\mu_a\mu_b}M, \qquad M^{bc} = \sqrt{\mu_b\mu_c}M, \qquad M^{ac} = \sqrt{\mu_a\mu_c}M \tag{8.41}$$

and thus the inductance matrix for the unbalanced stator is

$$\mathcal{L}' = \begin{bmatrix} \mu_a\mathcal{L} & -\sqrt{\mu_a\mu_b}M & -\sqrt{\mu_a\mu_c}M \\ -\sqrt{\mu_a\mu_b}M & \mu_b\mathcal{L} & -\sqrt{\mu_b\mu_c}M \\ -\sqrt{\mu_a\mu_c}M & -\sqrt{\mu_b\mu_c}M & \mu_c\mathcal{L} \end{bmatrix}. \tag{8.42}$$

The unbalanced three-phase model is

$$\begin{bmatrix} \mu_a\mathcal{L} & -\sqrt{\mu_a\mu_b}M & -\sqrt{\mu_a\mu_c}M \\ -\sqrt{\mu_a\mu_b}M & \mu_b\mathcal{L} & -\sqrt{\mu_b\mu_c}M \\ -\sqrt{\mu_a\mu_c}M & -\sqrt{\mu_b\mu_c}M & \mu_c\mathcal{L} \end{bmatrix} \frac{d\underline{i}_3}{dt} = -R\underline{i}_3 + KN\omega \begin{bmatrix} \mu_a\sin(N\theta) \\ \mu_b\sin(N\theta - \frac{2\pi}{3}) \\ \mu_c\sin(N\theta + \frac{2\pi}{3}) \end{bmatrix} + \underline{v}_3 \tag{8.43}$$

$$\frac{d\omega}{dt} = -\frac{B}{H}\omega - \frac{KN}{H}\begin{bmatrix} \mu_a\sin(N\theta) & \mu_b\sin(N\theta - \frac{2\pi}{3}) & \mu_c\sin(N\theta + \frac{2\pi}{3}) \end{bmatrix}\underline{i}_3 - \frac{C}{H}\frac{\omega}{|\omega|} - \frac{1}{H}\tau \tag{8.44}$$

$$\frac{d\theta}{dt} = \omega \tag{8.45}$$

The $\alpha\beta 0$ transformation will not reduce this model from three-phase to two-phase. When the phases are not balanced, v_α and v_β do not decouple from v_0. The three-phase Park transformation may be applied (see Fitzgerald, et al. [64, page 365])

but it will not be clear how to apply innovation in a meaningful way in the resulting model. Additionally, the parameters μ_a, μ_b, and μ_c must be determined from experimental measurements. The process of determining these parameters is complicated by the presence of a wye-connected stator with an isolated neutral.

Chapter 9

Experiment Results

Data was collected using the hardware and software described in Chapter 7. This data was then used in place of the motor simulation to drive the observer simulation described in Chapter 5. In this way, the performance of the observer was verified when used with a real motor. This was done as offline processing so that the computational complexity of the observer was not studied. The results of these tests and an analysis of the observer error in the results are presented in this chapter.

9.1 Measured Waveforms

The voltages waveforms should be sinusoidal at a single frequency when the motor is running at constant speed unless the current-control loop saturates. Measurements of v_α[1] at 300 rpm and 3000 rpm are shown in Figures 9.1 and 9.2, respectively. Note the negation of the resolver position on the horizontal axis to account for the fact that the motor was running in the negative direction (which is the positive direction according to the resolver—see Figure 8.9). The voltages are noisy at low speed since the inertia of the rotor has less of a smoothing effect. At high speeds, it is apparent that several frequencies are dominant. A spectrum of the 3000-rpm voltage shown in Figure 9.2 is shown in Figure 9.3. The dominant frequency is the 150 Hz electrical frequency at which torque is produced with small contributions at the 50 Hz mechanical frequency, the 250 Hz fifth harmonic of the mechanical frequency, the 300 Hz sixth harmonic, and the 350 Hz seven harmonic. The components at frequencies other than the electrical frequency are the result of the complex interaction of the controller with a nonideal motor. Nevertheless, the components

[1]Recall that $v_\alpha = \sqrt{(2/3)}\, v_a - \sqrt{(1/6)}\, v_b - \sqrt{(1/6)}\, v_c$ in general.

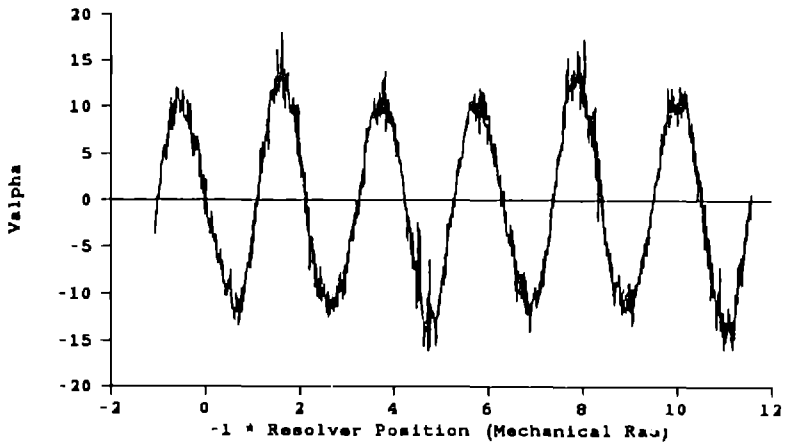


Figure 9.1: Measured two-phase voltage v_α. Speed = 300 rpm.

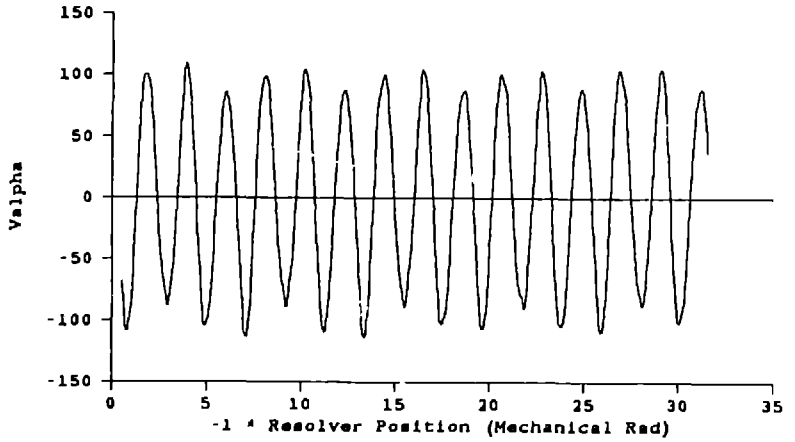


Figure 9.2: Measured two-phase voltage v_α. Speed = 3000 rpm.

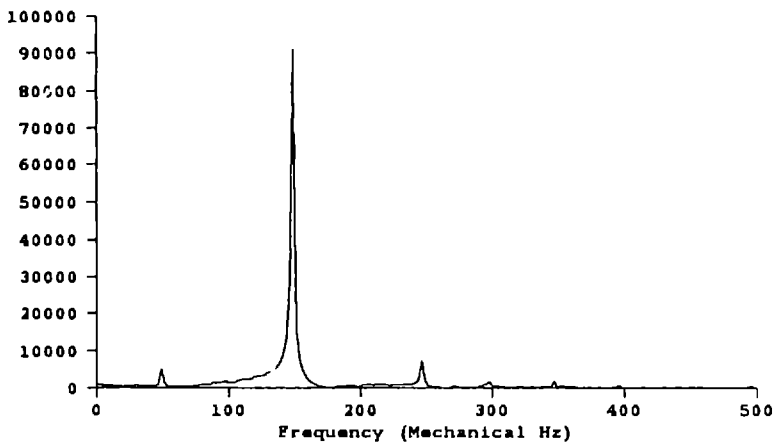


Figure 9.3: Spectrum of measured two-phase voltage v_α. Speed = 3000 rpm.

do appear at frequencies that may be associated with nonidealities of the motor. For example, the fifth and seventh harmonics may be associated with the imperfect sinusoidal winding of the stator. It is impossible to completely eliminate the odd harmonics, even in a slotless, air-wound stator. The sixth harmonic may be associated with the ripple in the inductance matrix and the harmonic at the mechanical frequency may be associated with the rotor not being concentric with the stator, as discussed in Chapter 8.

The current waveforms should also be sinusoidal at a single frequency when the motor is running at constant speed unless the current-control loop saturates. The outer speed-control loop commands the magnitude and torque angle of the sinusoids in order to maintain the commanded speed and the microprocessor in the controller synthesizes command sinusoids for the inner current-control loop. Measurements of i_α[2] at 300 rpm and 3000 rpm are shown in Figures 9.4 and 9.5, respectively. The time domain waveform at 300 rpm is quite abstruse. A spectrum of this waveform is shown in Figure 9.6 and the character of the signal is apparent. Although the 15 Hz electrical frequency at which torque is produced is dominant, there is a significant component at the mechanical frequency and at the fifth harmonic of the mechanical frequency. As with the voltage waveform, the fifth harmonic is a result

[2]Recall that $i_\alpha = \sqrt{(3/2)}\, i_a$ for a wye connection.

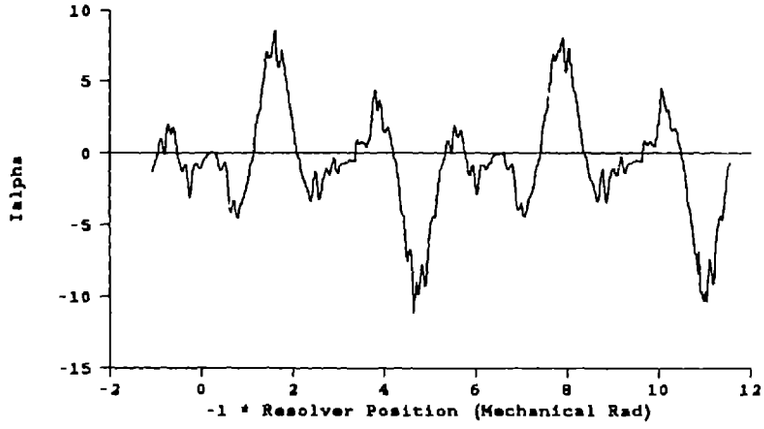


Figure 9.4: Measured two-phase current i_α. Speed = 300 rpm.

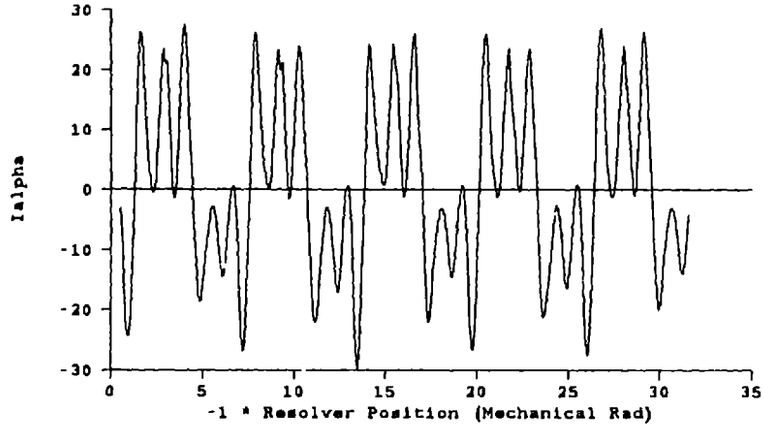


Figure 9.5: Measured two-phase current i_α. Speed = 3000 rpm.

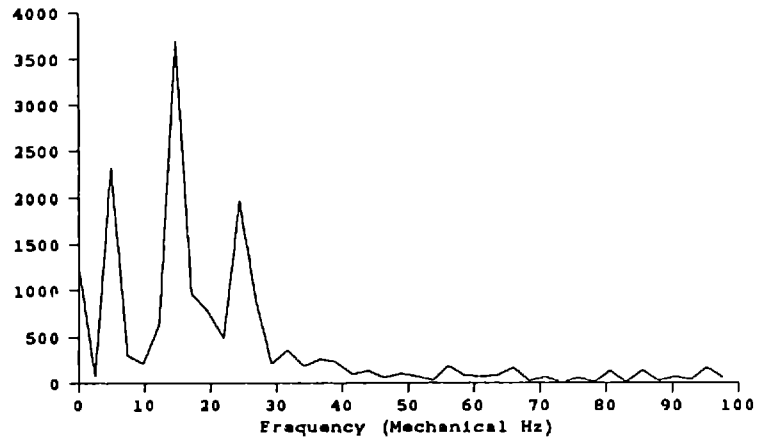


Figure 9.6: Spectrum of measured two-phase current i_α. Speed = 300 rpm.

of the imperfect winding of the stator and the mechanical fundamental is a result of the nonconcentricity of the rotor and stator.

It is interesting to note that the ratio of these components to the electrical frequency component is much larger in the current waveform than in the voltage waveform. Through the use of fractional pitch windings in the stator, the harmonics in the voltage can be substantially reduced compared to the the flux-density wave in the air-gap, Matsch [65]. The torque required to run the motor at a constant speed, however, is a direct result of the flux-density wave in the air-gap, and the flux-density wave in the air-gap is produced by the stator currents and the permanent magnets. Therefore, the effects of an imperfect sinusoidal winding of the stator is more pronounced in the current waveforms than in the voltage waveforms. Similarly, the effects of the nonconcentricity of the rotor and stator is also more pronounced in the current waveforms.

The existence of pronounced frequency components in the measured voltage and current waveforms at frequencies other than the electrical frequency is inconsistent with the motor model developed in Chapter 2. By the assumptions used to derive the model, the voltages necessary to drive motor at constant speed should be sinusoids at the electrical frequency. The resulting currents should also be sinusoids at the electrical frequency. The presence of unmodeled harmonics will cause the performance of the observer, which is based on the model, to suffer.

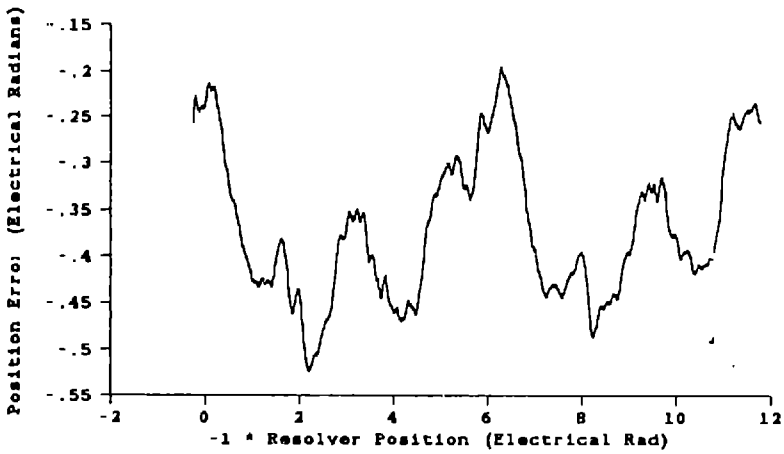


Figure 9.7: Observer position error response from rest for the real motor running at a constant 100 rpm. Load torque = 0.368 N·m.

9.2 Observer Response

The observer (4.31)–(4.33) was driven using measured voltages and currents in order to test its performance with a real motor rather than a simulated motor as in Chapter 5. Due to some operating limitations, the load torque applied to the motor was different during the various real-motor tests. The torque was measured and used in the observer during the offline tests. The value used is noted where appropriate. All tests were performed with the motor at approximately room temperature. Figures 9.7–9.10 illustrate the observer position error response using real motor data. The error is the difference between the position estimate produced by the observer and the position measured using the resolver mounted on the motor. These results are equivalent to those in Figures 5.7–5.9, that is, the observer position error response from rest for the motor running at constant speed.

Figure 9.7 illustrates the observer locking on to the motor running at the constant speed of 100 rpm. In this response, the initial observer error is very close to the steady-state error and so there is not a large transient. Note the bias of 0.36 electrical radians and the ripple with a standard deviation of 0.072 electrical radians. This standard deviation (rms) corresponds to 1.4 mechanical degrees. The observer position error response for another data set at 100 rpm is shown in Figure 9.8.

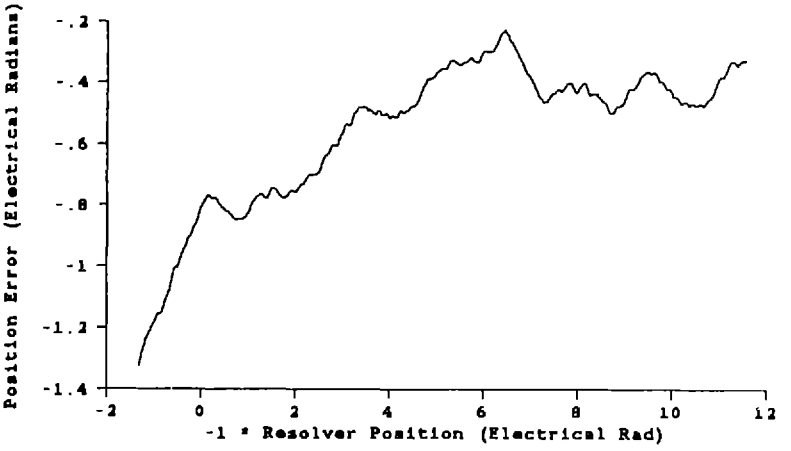


Figure 9.8: Observer position error response from rest for the real motor running at a constant 100 rpm. Load torque = 0.302 N·m.

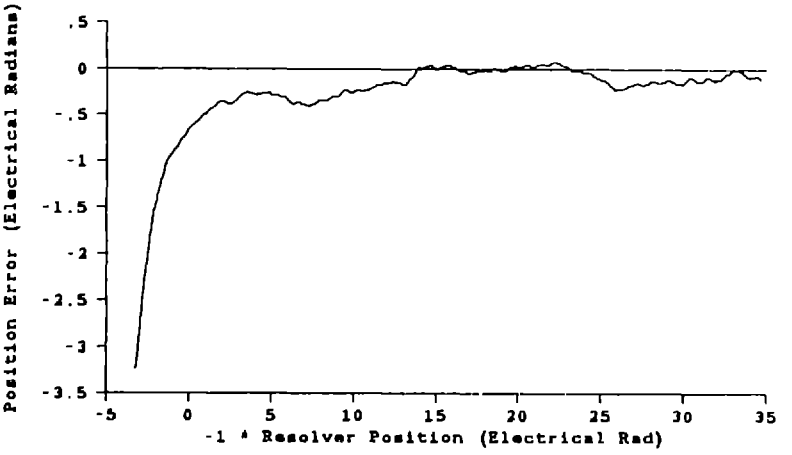


Figure 9.9: Observer position error response from rest for the real motor running at a constant 300 rpm. Load torque = 0.702 N·m.

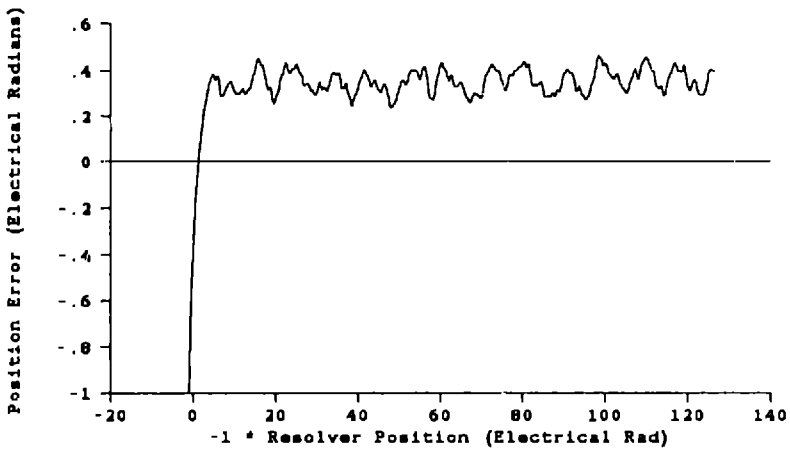


Figure 9.10: Observer position error response from rest for the real motor running at a constant 1000 rpm. Load torque = 0.969 N·m.

This response illustrates a large transient from an initial error of one electrical radian which lasts approximately 0.2 seconds. This time compares favorably with the simulated time in Figure 5.7.

Observer performance for constant motor speeds of 300 and 1000 are shown in Figures 9.9–9.10. A set of observer error responses for all four states for the motor running at 3000 rpm is shown in Figures 9.11–9.14. The large transients of the observer have approximately the same time constants as the simulations shown in Chapter 5 predicted. The rate of convergence is directly proportional to the speed of the motor. The least ripple is seen for 1000 rpm in Figure 9.10, and corresponds to 0.924 mechanical degrees rms.

The non-zero steady-state behavior was not predicted by the simulations. The error has discernable frequencies in it and does not appear to be Gaussian, as was used in the simulations. The bias and ripple also appear to be inversely proportional to speed. The sources of this error will be explored in the next section.

9.3 Analysis of Observer Error

A spectrum of the last 1024 values of observer error in Figure 9.14 is shown in Figure 9.15. The observer error is composed of five dominant frequencies: a com-

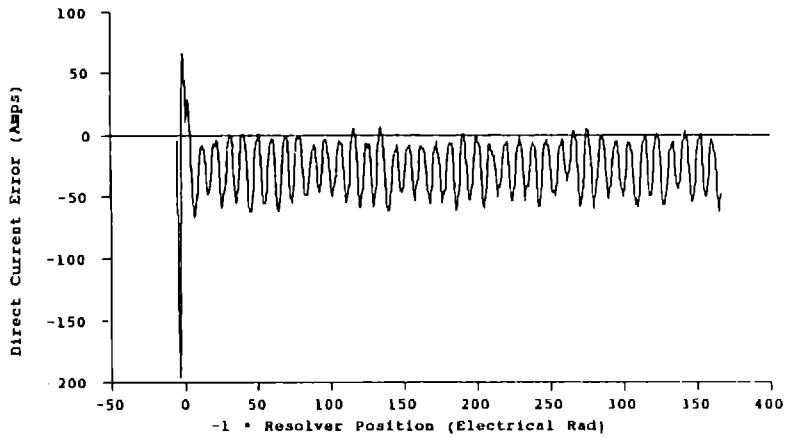


Figure 9.11: Observer direct current error response from rest for the real motor running at a constant 3000 rpm. Load torque = 1.17 N·m.

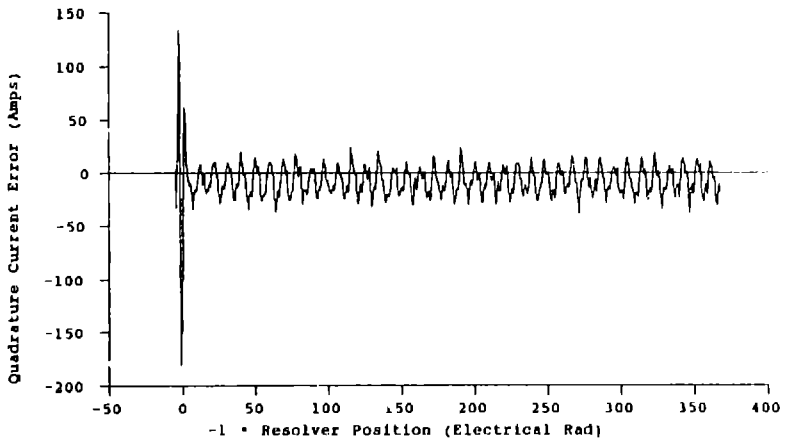


Figure 9.12: Observer quadrature current error response from rest for the real motor running at a constant 3000 rpm. Load torque = 1.17 N·m.

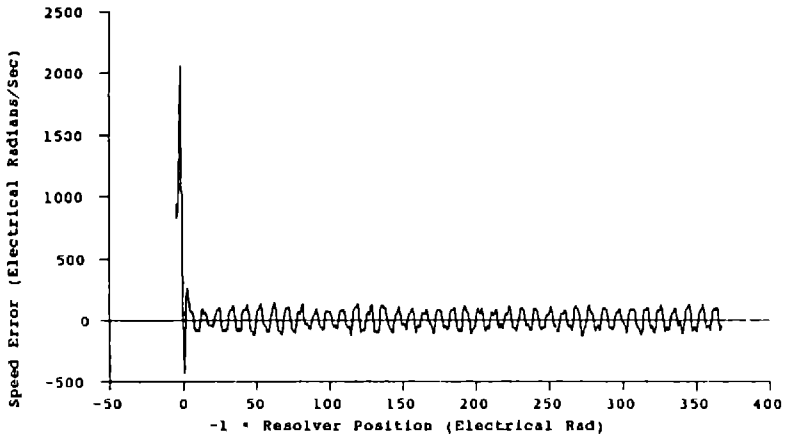


Figure 9.13: Observer speed error response from rest for the real motor running at a constant 3000 rpm. Load torque = 1.17 N·m.

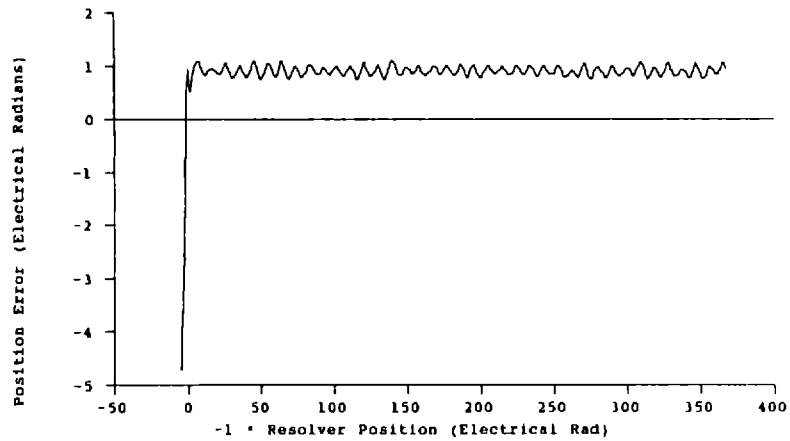


Figure 9.14: Observer position error response from rest for the real motor running at a constant 3000 rpm. Load torque = 1.17 N·m.

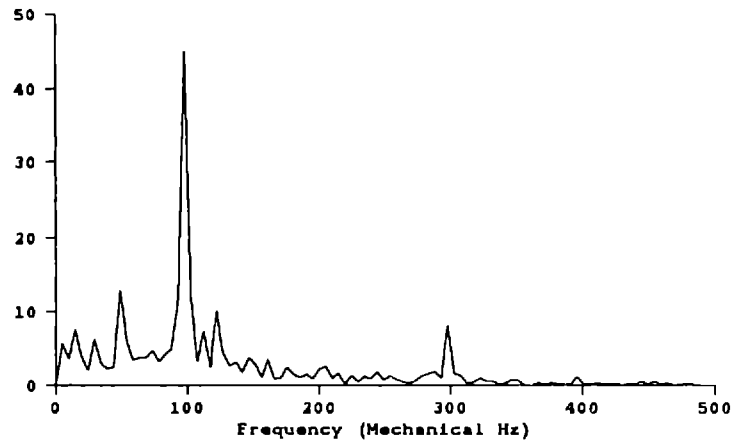


Figure 9.15: Spectrum of steady-state observer position error for the real motor running at a constant 3000 rpm. Zero frequency component suppressed to show harmonics.

ponent at 0 Hz, a component at the mechanical fundamental of 50 Hz, a second harmonic of the mechanical fundamental at 100 Hz, a fractional harmonic at 125 Hz, a sixth harmonic of the mechanical fundamental at 300 Hz. Each of these is discussed below.

The errors at the 50 Hz mechanical fundamental and the 300 Hz sixth harmonic are most likely due to motor modeling errors since the model in the observer exhibits only a 150 Hz third harmonic while operating at a constant 3000 rpm. The third harmonic results from having three magnet pole pairs on the rotor. In Chapter 8, it was found that the rotor of the motor was not centered in the stator and that the inductance profile of the stator was not independent of the position of the rotor. An asymmetric rotor and stator combination could result in once per revolution modeling errors, which would in turn drive a 50 Hz observer estimation error. Such errors could also result from modeling errors due to other manufacturing nonuniformities. The inductance ripple is most likely caused by the difference between the permeability of the permanent magnets and the air spaces between them. This would result in a sixth harmonic modeling error and hence a 300 Hz observer estimation error. Again some manufacturing nonidealities could also contribute to this error. The motor model in the observer used in this chapter, that is,

the ideal model derived in Chapter 2, did not account for any of these deviations from the ideal model. Presumably, including these deviations, perhaps as derived in Chapter 8, would eliminate these harmonics from the error.

The harmonic at 125 Hz is at five-halves the fundamental and its source is undetermined. It is most likely due to modeling error, perhaps arising through the beating between two other frequencies in the nonlinear observer. Note in Figure 9.5 the presence of a strong fifth harmonic. The motor model used in the observer cannot reproduce this harmonic for constant speed operation, clearly indicating that the model is not complete.

The zero-frequency bias component apparent in Figure 9.14, but suppressed in Figure 9.15 for plotting purposes, appears to be the result of phase shifts in the analog channels since the magnitude of the component is a function of speed. A significant source of phase shift is the isolation amplifiers used to measure the voltages. A similar source of phase shift is not present in the two channels used to measure the currents.

The simulation software presented in Chapter 5 was used to investigate the effects of phase shifts in the measurements. If the voltage waveforms are delayed by five samples, which corresponds to 1 ms at a sampling rate of 5 kHz, a constant offset is introduced into the observer position error response. This is shown in Figure 9.16. The value of the offset is -0.928 electrical radians. Note that the sign of the error is consistent with that of Figure 9.14 because the motor runs in opposite directions in the two experiments. If the motor is simulated at other (slower) speeds, the amount of delay in the simulated voltage waveforms needed to replicate the experimental offset is less, that is, the delay is a function of frequency. Apparently then, the constant, steady-state offset is a result of phase shifts in the analog channels.

The line-to-line inductance of the Industrial Drives motor is 0.4 mH. This small inductance means that the current ramps up and down very quickly when a voltage is applied. The simulated currents in the observer also ramp up and down very quickly in response to the measured voltages. Therefore, in order that the estimated currents be accurate, the measured voltages must be very accurate. The current in the stator is primarily the result of the difference between the applied voltage and the speed voltage applied to the phase inductance. In a motor with small phase inductances, this difference of voltages is very small. In a motor with strong magnets on the rotor, the speed voltage is large and thus the applied voltage must also be large. The speed voltage cannot be directly measured while the motor is running and

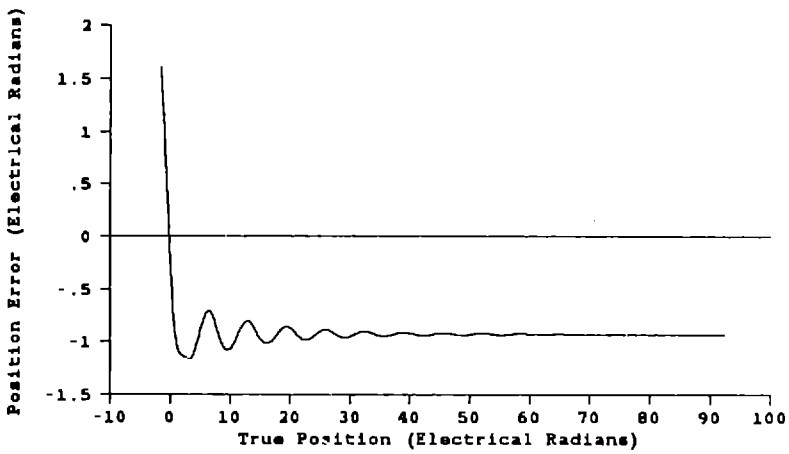


Figure 9.16: Observer position error response from rest for the simulated motor running at a constant 3000 rpm with the voltages delayed five samples. Load torque = 1.17 N·m.

is estimated by the observer in the quadrature current equation. The large applied voltage is measured. The implication is that the voltage measurements must be accurate to within less than 1 volt on a dynamic range of approximately 300 volts. This represents a demanding accuracy of less than 0.3%. The problem is further complicated by the fact that the fundamental voltage must be filtered from the pulse-width modulated voltages applied to the stator. Observer performance would undoubtedly improve in motors with larger stator inductances since the importance of making extremely accurate voltage measurments would be reduced.

The observer-estimated position is compared to the position measured using the resolver mounted on the motor shaft to form the observer error. The resolver is a frameless resolver meaning that the rotor of the resolver is mounted on the shaft of the motor and the stator of the resolver is mounted on the stator case of the motor. Since the rotor is not centered in the stator of the motor, the rotor of the resolver is not centered in the stator of the resolver. Visual examination of the resolver confirms this fact.

The 100 Hz second harmonic error in Figure 9.15 is most likely due to resolver measurement errors; the motor model in the observer is not the source of second harmonics of the mechanical fundamental. Experiments confirmed this hypothesis.

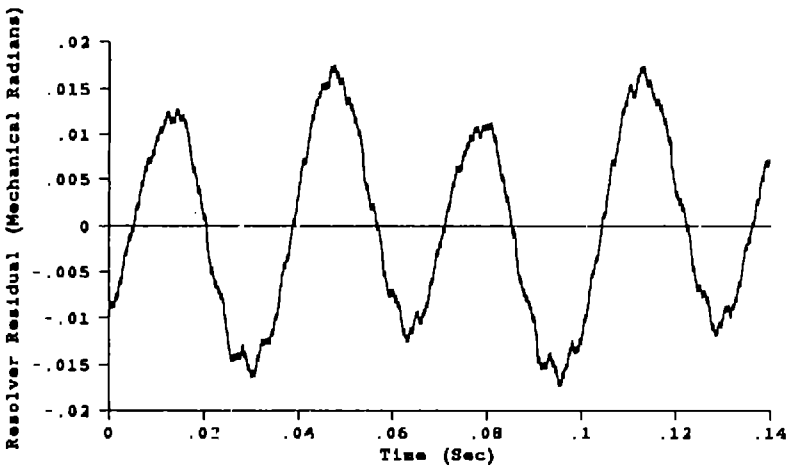


Figure 9.17: Residual between the measured position and a least squares linear fit line which reveals a second harmonic component.

If the DC motor is used to spin the permanent-magnet motor and resolver at a high constant speed, the residual between the measured position and a least squares linear fit line reveals a second harmonic resolver error. An example residual is shown in Figure 9.17. In this way, the resolver error was experimentally determined. The DC motor was used to spin the permanent-magnet motor at a high speed. A residual curve with respect to measured resolver position (0–4096) was then constructed. A polynomial was fit to this data and used to correct the measured position. The experiment results were found to be repeatable and the final correction curve was a fit of the average of five runs. This correction curve is shown in Figure 9.18. The correction curve was *not* used to compute the resolver error shown in Figures 9.7–9.10 and Figure 9.14, and application of this correction curve to the resolver data yields little improvement to the observer position error responses. Eliminating resolver error from the observer response is apparently more difficult than simply subtracting the measured error from the observer position error response. The reason for this is that the motor controller is using the output of the resolver to keep the speed constant. If the controller is forcing the speed of the motor to track the second harmonic, then second harmonic errors may appear in the observer position error response. Correcting for these errors in the resolver measurements will not improve the observer position error response since the second harmonic is

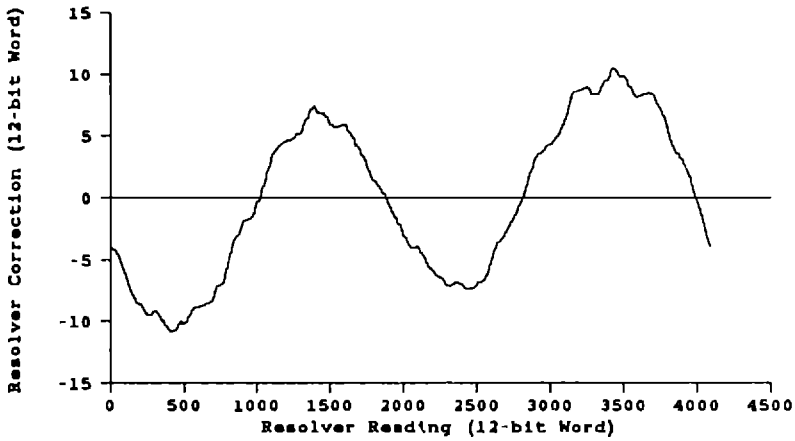


Figure 9.18: Resolver correction curve.

entering the observer through the current and voltage measurements. The observer will follow the error only if its bandwidth is substantially higher than that of the controller. The bandwidth of the observer may be increased by increasing the gains.

It is also possible that the resolver error depends on the load torque applied to the motor shaft. It is difficult to ascertain if the resolver error is dependent on load since the DC motor must be used to turn the assembly rather than be used as a load. The permanent-magnet motor cannot be used since it may be following the resolver error. This being the case, the correction curve shown in Figure 9.18 may not be applicable when the permanent-magnet motor is used as a drive and the DC motor as a load. The accuracy of the resolver is of paramount importance for certifying the accuracy of the observer. To pursue the issue of observer accuracy further requires a better resolver.

The effect of eliminating the errors described above may be approximated using Parseval's relation for the Discrete Fourier Transform. The rms ripple in electrical radians of the sampled time signal may be computed using

$$e_{rms} = \sqrt{\frac{1}{N^2} \sum_{k=0}^{N-1} |X[k]|^2} \tag{9.1}$$

where $X[k]$ is the DFT of $x(nT)$, $T = 200$ μs, $N = 1024$, and $X[0]$ is set equal to zero. The result of this computation for the DFT shown in Figure 9.15 is 0.0784

which is equal to 1.50 mechanical degrees. If the spectrum pairs corresponding to the errors described above, that is, the frequencies [48.8, 4951.2], [97.66, 4902.3], [122.1, 4877.9], and [297.85, 4702.1], are set to zero, the rms ripple is computed to be 0.0403 which is equal to 0.770 degrees. Therefore, eliminating the errors identified above will reduce the ripple in the steady-state observer error by approximately half.

9.4 Conclusions

The observer tests using experimental data were largely successful. The positive results showed that the observer can successfully follow very large transients and that the time constants of these transients are as predicted in the simulations. The largest transient was for the observer starting from rest with the motor running at 3000 rpm. Further, it was interesting to note that the linearized observer error dynamics predicted the behavior of such large scale transients rather well.

The test results revealed the steady-state observer error when driven by experimental data to be greater than predicted in the simulations. In Section 5.2.3, the steady-state error with high level noise was 0.146 mechanical degrees at 1000 rpm. This compares with 0.924 mechanical degrees at 1000 rpm in the experimental tests. However, the noise used in the simulations was unstructured Gaussian noise and analysis of the experimental errors revealed that they were due to *structured* noise in the measurements. The sources of error were identified as follows.

- An error at the mechanical fundamental frequency which is a result of not including once-per-revolution motor manufacturing nonuniformities in the observer model. This error could be eliminated by developing a more sophisticated model as suggested in Section 8.4.

- A sixth harmonic error which is a result of not including the ripple in the stator inductance in the observer model. The ripple is due to the air-gaps between the permanent magnets on the rotor. This error could be eliminated by using the more sophisticated motor model developed in Section 8.3.

- An error at five-halves the mechanical fundamental. The source of this error is undetermined, but it is most likely the result of modeling error. Hopefully, this error could be eliminated by including the relevant phenomenon in the observer model.

- Phase shifts in the analog measurement channels resulting in constant offsets in the steady-state error. This error could be eliminated by redesigning the analog channel circuits to yield identical, linear phase.

- A second harmonic error of the mechanical fundamental in the resolver measurement. This error could be eliminated by using a more accurate resolver and may be eliminated when the observer position estimate is used in place of the resolver to control the motor.

The experimental accuracy of the observer is probably sufficient to run the motor in speed control applications but may not be sufficient for servo applications. However, no fundamental limitiation on observer performance was discovered in the experiments and so the elimination of the errors as suggested would dramatically improve the performance of the observer possibly making it useful for servo applications.

Chapter 10

Summary and Future Research

10.1 Summary

The use of a state observer could eliminate the need for sensors to detect all states in high-performance variable speed drives. In particular, measurements of stator voltages and currents processed by such an observer may permit the elimination of the sensor used to sense the position and speed of the motor shaft. The observer developed in this thesis reconstructs the dynamic states of a smooth-rotor permanent-magnet synchronous motor in this way. The motor model used in this thesis was develop in Chapter 2 and is

$$\frac{d\underline{i}}{dt} = -\frac{R}{L}\underline{i} - \frac{K}{L}N\omega\exp(\mathbf{J}N\theta)\begin{bmatrix}0\\1\end{bmatrix} + \frac{1}{L}\underline{v} \tag{2.64}$$

$$\frac{d\omega}{dt} = -\frac{B}{H}\omega + \frac{KN}{H}\underline{i}^T\exp(\mathbf{J}N\theta)\begin{bmatrix}0\\1\end{bmatrix} - \frac{C}{H}\frac{\omega}{|\omega|} - \frac{1}{H}\tau \tag{2.65}$$

$$\frac{d\theta}{dt} = \omega. \tag{2.66}$$

The states of this fourth order system are two stator-frame currents $\underline{i} = \begin{bmatrix} i_\alpha & i_\beta \end{bmatrix}$, shaft speed ω, and shaft position θ.

The dq transformation,

$$\bar{\underline{v}} = \exp(-\mathbf{J}\theta)\,\underline{v} \tag{2.70}$$

$$\bar{\underline{i}} = \exp(-\mathbf{J}\theta)\,\underline{i} \tag{2.71}$$

removes the nonlinearity with respect to θ when θ is known. The resulting model

is

$$\frac{d\underline{i}}{dt} = -\left(\frac{R}{L}\mathbf{I} + \mathbf{J}N\omega\right)\underline{i} - \frac{K}{L}N\omega\begin{bmatrix}0\\1\end{bmatrix} + \frac{1}{L}\underline{v} \tag{2.72}$$

$$\frac{d\omega}{dt} = -\frac{B}{H}\omega + \underline{i}^T\frac{KN}{H}\begin{bmatrix}0\\1\end{bmatrix} - \frac{C}{H}\frac{\omega}{|\omega|} - \frac{1}{H}\tau \tag{2.73}$$

$$\frac{d\theta}{dt} = \omega \tag{2.74}$$

The states of this fourth order system are direct and quadrature stator currents $\underline{i} = \begin{bmatrix} i_d & i_q \end{bmatrix}$, shaft speed ω, and rotor position θ. The transformation is inexact if the position θ is not known. Without using the transformation, the model is nonlinear and it is difficult to determine how to apply the well-developed theory of linear observers. The *dq* transformation provides a structure for applying observer theory.

In the estimated-innovation observer presented in Chapter 4, the *dq* transformed model is driven by innovation which is the difference between the estimated output of the model $\hat{\underline{i}}$ and the measured currents transformed into the estimated frame $\check{\underline{i}}$,

$$\check{\underline{i}} = \exp(-\mathbf{J}N\hat{\theta})\underline{i}\,. \tag{4.34}$$

As long as the estimated theta $\hat{\theta}$ is not correct, the transformation of the measured currents will not be correct and thus the innovation will not be zero. The resulting observer is

$$\frac{d\hat{\underline{i}}}{dt} = -\left(\frac{R}{L}\mathbf{I} + \mathbf{J}N\hat{\omega}\right)\hat{\underline{i}} - \frac{K}{L}N\hat{\omega}\begin{bmatrix}0\\1\end{bmatrix} + \frac{1}{L}\hat{\underline{v}} + \mathbf{G}_{\underline{i}}(\check{\underline{i}} - \hat{\underline{i}}) \tag{4.31}$$

$$\frac{d\hat{\omega}}{dt} = -\frac{B}{H}\hat{\omega} + \hat{\underline{i}}^T\frac{KN}{H}\begin{bmatrix}0\\1\end{bmatrix} - \frac{C}{H}\frac{\hat{\omega}}{|\hat{\omega}|} - \frac{1}{H}\tau + \frac{KN}{H}\mathbf{G}_{\omega}(\check{\underline{i}} - \hat{\underline{i}}) \tag{4.32}$$

$$\frac{d\hat{\theta}}{dt} = \hat{\omega} + \mathbf{G}_{\theta}(\check{\underline{i}} - \hat{\underline{i}})\,. \tag{4.33}$$

Variables with a superscript carat are the observer estimates of the true states. Note that $\hat{\underline{v}}$ is the known stator voltages transformed into the estimated frame. The innovation depends nonlinearly on the observer state.

In Chapter 5, the motor and observer were simulated to establish the performance of the observer. The primary performance criterion was rapid convergence

of the observer position estimate $\hat{\theta}$ to the true motor position θ. The observer was found to converge in approximately one electrical cycle which was much faster than the hunting transient of an open-loop simulation. The observer was also found to be very robust for large perturbations and able to converge quickly from large disturbances. The transients examined were the observer starting from rest for a motor running at a constant speed, the observer and motor both starting from rest and ramping at a constant rate, the observer and motor both starting from rest and ramping at a constant rate with the initial observer shaft position unknown, and the observer following a step change of the motor speed.

The observer was also simulated with incorrect model parameters. The parameters R, K, H, and τ were perturbed one at a time and it was found that the observer was most sensitive to the magnet constant K. Large steady-state errors develop if the value of K was incorrect. If H was incorrect the transient error response of the observer was significantly slower. The observer was simulated with the voltage and current measurements corrupted with noise. It was found that the observer remains stable in the presence of relatively high levels of noise but that the steady-state performance suffers. It was also found that the observer was more sensitive to errors in the voltage measurements than to errors in the current measurements.

Stability of the observer was studied in a local sense via analysis of the linearized error dynamics in Chapter 6. The stability of the observer for a given set of gains can be accurately predicted using the linearized error dynamics even for large perturbations. The performance of the observer may also be predicted well, which provides guidance for choosing the observer gains. It was found that the poles of the linearized error dynamics predicted the instability of the observer at high speeds. Finally, it was found that the most important gains are the mechanical gains G_ω since they couple the electrical and mechanical subdynamics.

Experimental hardware was constructed to test the performance of the observer with a real motor rather than a simulated motor. The hardware and software was described in Chapter 7. It consisted of a wye-connected three-phase permanent-magnet synchronous motor where the magnets were mounted on the surface of the rotor. The motor was driven by a voltage inverter. It was controlled by a PI speed controller with hardware that converts the torque commands into sinusoidal current commands. The current commands were enforced in the motor by fast control loops that pulse-width modulate the voltages applied to the motor.

Hardware was constructed that may be used in a real-time implementation of

the observer. In this thesis, it was used to collect data for offline tests. This hardware consisted of analog channel multiplexing circuitry and interrupt control logic such that the burden on the microprocessor of collecting voltage and current measurements was minimal. The microprocessor used was a TMS32020 digital signal processor.

The process of measuring the voltage applied to the motor was complicated by the pulse-width modulation of the current-control loop. Only the fundamental frequency of the voltage was desired to drive the observer and so the PWM carrier frequency of 8 *k*Hz was filtered out. The experimental motor had very small stator inductances and so the currents in the phases were the result the small difference between the applied voltage and the speed voltage. The need for accurate voltage measurements was demonstrated in Chapter 5.

The physical parameters used by the observer were measured and the results were presented in Chapter 8. While measuring the parameters, it was discovered that the stator inductance of the experimental motor was not independent of the position of the rotor and that the motor was not electrically balanced. The ripple in the stator inductance was determined to be a result of differences between the permeability of the magnets mounted on the surface of the rotor and the air spaces between each magnet. It was determined that the motor was not electrically balanced due to the fact that the rotor and stator were not concentric. A model that accounts for the ripple in the stator inductance was developed and suggestions for modeling the nonconcentricity of the rotor and stator were given.

In Chapter 9, the performance of the observer used with data from the real motor was illustrated. The steady-state accuracy of the observer was found to be approximately one mechanical degree. This accuracy was below that predicted by the simulations. A spectral analysis of the error revealed the sources of the error. Components of error existed due to modeling error in the observer and to error in the reference resolver used to operate the permanent-magent motor and to verify the observer position estimate. It is believed that if the resulting inconsistencies between the real motor and its model were eliminated, the observer would perform much better. Removing the inconsistencies would require modifications to motor manufacturing or extended modeling efforts.

10.2 Future Research

In Chapter 8, the motor used in the experiments was found not to have a stator inductance profile independent of position and that the rotor was not concentric with the stator. The accuracy of the observer when used with real motor data could be improved by using the models presented in Chapter 8. Note that an observer based on the model derived in Section 8.3 could be used with a buried-magnet motor having a component of reluctance torque. The accuracy of the observer could also be improved by using more accurate voltage sensors. Due to the sensitivity of the observer to voltage measurement errors, it is important that the voltage measurements be very accurate. The observer should be implemented in real-time and incorporated into the motor control loop. The hardware described in Chapter 7 was designed for the real-time implementation of an observer. The digital position word produced by the observer could be used directly in place of the digital word the controller obtains from the resolver-to-digital converter. All of these improvements and extensions were being implemented at the time this thesis was being written.

In Chapter 5 it was found that the observer converged in approximately one electrical cycle. At slow speeds, the observer required more time to converge. At high speeds, the observer became unstable. The observer gains were chosen as a compromise between forcing the observer to converge rapidly at slow speeds while remaining stable at high speeds. The question of how to choose constant gains needs to be studied further. The procedure given by Yamashita and Taniguchi [51] for optimizing the gains in their stator-frame observer might be a useful place to start. If the gains are allowed to be nonconstant, gain scheduling could be used or a function could be found for the gains such that the performance of the observer is optimal over a range of speeds. The stability of the observer with constant gains needs to be studied further.

The observer in this thesis was developed for the permanent-magnet synchronous motor. The idea of estimated innovation arose from the use of the *dq* transformation which is applicable to a large class of machines. Liu, et. al [13] have studied the general requirements for a machine to be *dq* transformable. Using these results it may be possible to show that this observer may be used with any machine that is *dq* transformable.

Appendix A

RKF45 and Driver

The procedure RKF45 is used to solve systems of differential equations. It is based on the Runge-Kutta-Fehlberg fourth-fifth order algorithm. A listing of this procedure is provided in this appendix. How to use the code is carefully documented in the comments. A driver is needed to call RKF45 and a sample driver is provided beginning on page 196. Procedure RKF45 was used in the observer simulation program to simulate the motor as shown in Appendix B.

```
procedure rkf45( var neqn :integer; var y :savetype;
                 var t, tout, relerr, abserr :real;
                 var iflag :integer; var yp :savetype; var h :real;
                 var f1, f2, f3, f4, f5 :savetype;
                 var savre, savae, eps, u26 :real;
                 var nfe, kop, init, jflag, kflag :integer );


{    *-------------------------------------------------*
     !                                                 !
     !   FEHLBERG FOURTH-FIFTH ORDER RUNGE-KUTTA METHOD  !
     !                                                 !
     *-------------------------------------------------*

                        Written by
                 H. A. Watts and L. F. Shampine
     Sandia National Laboratories, Albuquerque, New Mexico.


     Translated from FORTRAN to Pascal by Lawrence A. Jones

  ---------------------------------------------------------------
  MODIFIED TO STORE EPS (MACHINE EPSILON) AND U26 IN THE PROCEDURE
  PARAMETER LIST. EPS AND U26 ARE NOT STORED IN THE ORIGINAL CODE
  AND IT IS NECESSARY TO STORE THEM SINCE THEY ARE NOT RECALCULATED
  EVERY CALL.  STORED IN VAR PARAMETERS EPS AND U26.    --LAJ
  ---------------------------------------------------------------

  (*Note: The structure of the FORTRAN program has been maintained
          in this translation to Pascal in order to preserve the
          numerical integrity of the routine. *)
```

RKF45 is primarily designed to solve non-stiff and mildly stiff differential equations when derivative evaluations are inexpensive. RKF45 should generally not be used when the user is demanding high accuracy.

Abstract

Procedure RKF45 integrates a system of neqn first order ordinary differential equations of the form

$$dy(i)/dt = f(t, y(1), y(2),..., y(neqn))$$

where the y(i) are given at time t. Typically the procedure is used as a one-step integrator to advance the solution a single step in the direction of tout. On return the parameters in the call list are set for continuing the integration. The user has only to call RKF45 again (and perhaps define a new value for tout). RKF45 calls procedure FEHL which computes an approximate solution over one step. RKF45 uses the Runge-Kutta-Fehlberg (4,5) method described in the reference:

E. Fehlberg. Low-Order Classical Runge-Kutta Formulas with Stepsize Control. NASA TR R-315. (Also in Computing, 6 (1970), pp. 61-71.).

The performance of RKF45 is illustrated in the reference:

L. F. Shampine, H. A. Watts, S. Davenport. Solving Non-Stiff Ordinary Differential Equations-- The State of the Art. SIAM Review, 18 (1976), pp. 376-471.

For a general discussion of RKF45 and a FORTRAN listing see:

G. E. Forsythe, M. A. Malcolm, C. B. Moler. Computer Methods for Mathematical Computations. Prentice-Hall, Inc. 1977, pp. 127-147.

The parameters represent:

```
neqn      :Number of equations to be integrated.
y(*)      :Solution vector at t.
t         :Independent variable.
tout      :Output point at which solution is desired.
relerr,
abserr    :Relative and absolute error tolerances for local
```

```
              error test. At each step the code requires that

                   abs(local error)  <=  relerr*abs(y) + abserr

              for each component of the local error and solution
              vectors.
     iflag    :Indicator for status of integration.
     yp,h,f1,
     f2,f3,f4,
     f5,savre,
     savae,
     nfe,kop,
     init,
     jflag,
     kflag    :Reals and arrays (of dimension neqn of SAVETYPE) to
               hold information internal to RKF45 which are
               necessary for subsequent calls.
```

First call to RKF45

The user must provide storage in his calling program for the arrays in the call list. THEY MUST BE DECLARED TO BE TYPE SAVETYPE! A procedure f, external to RKF45, performs the evaluation of the system equations to the right of the equal signs. The parameters of this routine are

```
     t  :Real.
     y  :Array of type savetype (which are of size neqn).
     yp :Array of type savetype (which are of size neqn).
```

The driver program calling RKF45 must initialize the following parameters:

```
     neqn    :Number of equations to be integrated (neqn >= 1).
     y(*)    :Vector of initial conditions.
     t       :Starting point of integration, must be a variable.
     tout    :Output point at which solution is desired. RKF45
              returns with iflag = 2 if continuation is possible.
     relerr,
     abserr  :Relative and absolute local error tolerances which
```

must be non-negative. Both relerr and abserr must be variables. The code should normally not be used with relative error control smaller than about 1.0e-8. To avoid limiting precision difficulties the code requires relerr to be larger than an internally computed relative error parameter which is machine dependent. In particular, pure absolute error is not permitted. If a smaller than allowable value of relerr is attempted, RKF45 increases relerr appropriately and returns control to the user before continuing the integration.

iflag :+1,-1 indicator to initialize the code for each new problem. Normal input is +1. The user should set iflag = -1 only when one-step integrator control is essential. In this case, RKF45 attempts to advance the solution a single step in the direction of tout each time it is called. Since this mode of operation results in extra computing overhead, it should be avoided unless needed.

Output from RKF45
---

y(*) :Solution at t.
t :Last point reached in integration.
iflag = 2 :Integration reached tout. Indicates successful return and is the normal mode for continuing integration.
= -2 :A single successful step in the direction of tout has been taken. Normal mode for continuing integration one step at a time.
3 :Integration was not completed because relative error tolerance was too small. Relerr has been increased appropriately for continuing.
= 4 :Integration was not completed because more than 32767 derivative evaluations were needed. This is approximately 500 steps.
= 5 :Integration was not completed because solution vanished making a pure relative error test impossible. Must use non-zero abserr to continue. Using the one-step integration mode for one step

```
            is a good way to proceed.
        =   6 :Integration was not completed because requested
            accuracy could not be achieved using smallest
            allowable stepsize.  User must increase the error
            tolerance before continued integration can be
            attempted.
        =   7 :It is likely that RKF45 is inefficient for solving
            this problem.  Too much output is restricting the
            natural stepsize choice.  Use the one-step
            integrator mode.
        =   8 :Invalid input parameters. This indicator occurs if
            any of the following is satisfied:

                  neqn <= 0
                  t = tout and iflag <> +1 or -1
                  relerr or abserr < 0
                  iflag = 0 or < -2 or > 8

    The following information is usually of no interest to the
    user but necessary for subsequent calls.

    yp          :First derivatives of the solution vector y at t.
    h           :Stepsize to be attempted on the next step.
    nfe         :Derivative evaluation counter.
```

Subsequent calls to RKF45

Procedure RKF45 returns with all information needed to continue the integration. If the integration reached tout, the user need only define a new tout and call RKF45 again. In the one-step integrator mode (iflag = -2) the user must keep in mind that each step taken is in the direction of the current tout. Upon reaching tout (indicated by changing iflag to 2), the user must then define a new tout and reset iflag to -2 to continue in the one-step integrator mode.

If the integration was not completed but the user still wants to continue (iflag = 3,4 cases), he just calls RKF45 again. With iflag = 3 the relerr parameter has been adjusted appropriately for continuing the integration. In the case of iflag = 4 the function

counter will be reset to 0 and another 32767 function evaluations are allowed.

However, in the case iflag = 5, the user must first alter the error criterion to use a positive value of abserr before integration can proceed. If he does not, execution is terminated.

Also, in the case iflag = 6, it is necessary for the user to reset iflag to 2 (or -2 when the one-step integration mode is being used) as well as increasing either abserr, relerr or both before the integration can be continued. If this is not done, execution will be terminated. The occurrence of iflag = 6 indicates a trouble spot (solution is changing rapidly, singularity may be present) and it often is inadvisable to continue.

If iflag = 7 is encountered, the user should use the one-step integration mode with the stepsize determined by the code or consider switching to the Adams codes DE/STEP,INTRP. If the user insists upon continuing the integration with RKF45, he must reset iflag to 2 before calling RKF45 again. Otherwise, execution will be terminated.

If iflag = 8 is obtained, integration can not be continued unless the invalid input parameters are corrected.

It should be noted that the variables and arrays yp, h, f1, f2, f3, f4, f5, savre, savae, nfe, kop, init, jflag, kflag contain information required for subsequent integration. Accordingly, they should not be altered. }

```
{-------------------------------------------------------------}

label 5, 10, 20, 25, 30, 40, 45, 50, 55, 60, 65, 70, 80, 85, 90,
      95, 100, 150, 200, 220, 240, 250, 260, 270, 300;

var
   hfaild, output :boolean;
   a, ae, dt, ee, eeoet, esttol, et, hmin, remin, rer, s, scale,
   tol, toln, epsp1, ypk :real;
   k, maxnfe, mflag :integer;
```

```
{Remin is the minimum acceptable value of relerr.  Attempts to
 obtain higher accuracy with this procedure are usually very
 expensive and often unsuccessful.}

{-------------------- ----------------------------------------}

procedure fehl( var neqn:integer; var y:savetype; var t, h:real;
                var yp, f1, f2, f3, f4, f5, s: savetype );

{Fehl integrates a system of neqn first order ordinary differential
equations of the form

            dy(i)/dt = f( t, y(1), y(2),..., y(neqn) )

 where the initial values y(i) and the initial derivatives yp(i)
 are specified at the starting point t.  Fehl advances the solution
 over the fixed step h and returns the fifth order (sixth order
 accurate locally) solution approximation at t+h in array s(i).
 F1,...,f5 are arrays of dimension neqn which are needed for
 internal storage.  The formulas have been grouped to control
 loss of significance.  Fehl should be called with an h not smaller
 than 13 units of roundoff in t so that the various independent
 arguments can be distinguished.}

label 221, 222, 223, 224, 225, 230;

var
   ch :real;
   k :integer;

begin {fehl}
    ch := h/4.0e0;
    for k := 1 to neqn do  f5[k] := y[k] + ch*yp[k];
    F( t+ch, f5, f1 );

    ch := 3.0e0*h/32.0e0;
    for k := 1 to neqn do
         f5[k] := y[k] + ch*( yp[k] + 3.0e0*f1[k] );
    F( t+3.0e0*h/8.0e0, f5, f2 );

    ch := h/2197.0e0;
    for k := 1 to neqn do f5[k] :=  y[k]
```

```
                                    + ch*( 1932.0e0*yp[k]
                                           + (7296.0e0*f2[k]
                                           -7200.0e0*f1[k]) );
    F( t+12.0e0*h/13.0e0, f5, f3 );

    ch := h/4104.0e0;
    for k := 1 to neqn do f5[k] := y[k] + ch*( (8341.0e0*yp[k] -
                                      845.0e0*f3[k])
                                    + (29440.0e0*f2[k] -
                                      32832.0e0*f1[k]));
    F( t+h, f5, f4 );

    ch := h/20520.0e0;
    for k := 1 to neqn do  f1[k] := y[k] + ch*( (-6080.0e0*yp[k] +
                                      (9295.0e0*f3[k] -
                                      5643.0e0*f4[k]) )
                                    + ( 41040.0e0*f1[k] -
                                      28352.0e0*f2[k]));
    F( t+h/2.0e0, f1, f5 );

    {Compute approximate solution at t+h.}

    ch := h/7618050.0e0;
    for k := 1 to neqn do s[k] := y[k] + ch*( (902880.0e0*yp[k] +
                                    (3855735.0e0*f3[k] -
                                    1371249.0e0*f4[k]) )
                                  + (3953664.0e0*f2[k] +
                                    277020.0e0*f5[k]) );

end; {fehl}

{------------------------------------------------------------------}

begin {*** RKF45 ***}

     remin := 1.0e-12;

     {The expense is controlled by restricting the number of
      function evaluations to be approximately maxnfe.  As set,
      this corresponds to about 500 steps.}

     maxnfe := 32767;
```

```
     {Check input parameters.}

     if neqn < 1 then goto 10;
     if ( (relerr < 0.0) or (abserr < 0.0 ) ) then goto 10;
     mflag := abs( iflag );
     if (mflag = 0) then goto 10;
     if (mflag > 8) then goto 10;
     if (mflag <> 1) then goto 20;

     {First call, compute the machine epsilon.}

     eps := 1.0e0;
5:   eps := eps/2.0e0;
     epsp1 := eps + 1.0e0;
     if (epsp1 > 1.0e0) then goto 5;
     u26 := 26.0e0 * eps;
     goto 50;

     {Invalid input.}

10:  iflag := 8;
     exit;

     {Check continuation possibilities.}

20:  if ( (t = tout) and (kflag <> 3) ) then goto 10;
     if (mflag <> 2) then goto 25;

     {Iflag = +2 or -2.}

     if (kflag = 3) then goto 45;
     if (init = 0) then goto 45;
     if (kflag = 4) then goto 40;
     if ( (kflag = 5) and (abserr = 0.0e0) ) then goto 30;
     if ((kflag = 6) and (relerr <= savre) and (abserr <= savae))
          then goto 30;
     goto 50;

     {Iflag = 3, 4, 5, 6, 7, or 8.}

25:  if (iflag = 3) then goto 45;
```

```
      if (iflag = 4) then goto 40;
      if ( (iflag = 5) and (abserr > 0.0e0) ) then goto 45;
30:   write('Integration cannot be continued since you did not');
      writeln( 'respond to ');
      write('the instructions pertaining to ' );
      writeln('iflag= 5, 6, 7, or 8!');
      halt;

      {Reset function evaluation counter.}

40:   nfe := 0;
      if (mflag = 2) then goto 50;

      {Reset flag value from previous call.}

45:   iflag := jflag;
      if (kflag = 3) then mflag := abs(iflag);

      {Save input iflag and set continuation flag value for
       subsequent input checking.}

50:   jflag := iflag;
      kflag := 0;

      {Save relerr and abserr for checking input on subsequent
       calls.}

      savre := relerr;
      savae := abserr;

      {Restrict relative error tolerance to be at least as large as
       2*eps+remin to avoid limiting precision difficulties arising
       from impossible accuracy requests.}

      rer := 2.0e0*eps + remin;
      if (relerr >= rer) then goto 55;

      {Relative error tolerance too small.}

      relerr := rer;
      iflag := 3;
      kflag := 3;
```

```
     exit;

55:  dt := tout - t;

     if (mflag = 1) then goto 60;
     if (init = 0) then goto 65;
     goto 80;

     {Initialization: Set initialization completion indicator,init.
                      Set indicator for too many output points,kop.
                      Evaluate initial derivatives.
                      Set counter for function evaluations, nfe.
                      Estimate starting stepsize.                  }

60:  init := 0;
     kop := 0;

     a := t;
     F(a, y, yp );
     nfe := 1;
     if (t <> tout) then goto 65;
     iflag := 2;
     exit;

65:  init := 1;
     h := abs(dt);
     toln := 0.0;
     for k := 1 to neqn do begin
        tol := relerr * abs( y[k] ) + abserr;
        if tol <= 0.0 then goto 70;
        toln := tol;
        ypk := abs( yp[k] );
        if (ypk * power( h, 5.0 ) > tol) then
             h := power( (tol/ypk), 0.2e0 );
70:  end;
     if (toln <= 0.0e0) then h := 0.0e0;
     h := amax1( h, u26*amax1( abs(t), abs(dt) ) );
     jflag := isign( 2, iflag );

     {Set stepsize for integration in the direction
      from t to tout.}
```

```
80:  h := sign( h, dt );

     {Test to see if RKF45 is being severely impacted by too
      many output points.}

     if abs(h) >= 2.0e0 * abs(dt) then kop := kop + 1;
     if (kop <> 32767 ) then goto 85;

     {Unecessary frequency of output.}

     kop := 0;
     iflag := 7;
     exit;

85:  if ( abs(dt) > u26*abs(t) ) then goto 95;

     {If too close to output point, extrapolate and return.}

     for k := 1 to neqn do  y[k] := y[k] + dt*yp[k];
     a := tout;
     F(a, y, yp );
     nfe := nfe + 1;
     goto 300;

     {Initialize output point indicator.}

95:  output := false;

     {To avoid premature underflow in the error tolerance function,
      scale the error tolerances.}

     scale := 2.0e0/relerr;
     ae := scale * abserr;

     {Step-by-step integration.}

100: hfaild := false;

     {Set smallest allowable stepsize.}

     hmin := u26 * abs(t);
```

```
      {Adjust stepsize if necessary to hit the output point.  Look
       ahead two steps to avoid drastic changes in the stepsize and
       thus lessen the impact of output point on the code.}

      dt := tout - t;
      if abs(dt) >= 2.0e0 * abs(h) then goto 200;
      if abs(dt) > abs(h) then goto 150;

      {The next successful step will complete the integration to the
       output point.}

      output := true;
      h := dt;
      goto 200;

150:  h := 0.5e0 * dt;

      {Core integrator for taking a single step.

       The tolerances have been scaled to avoid premature underflow
       in computing the error tolerance function et.  To avoid
       problems with zero crossings, relative error is measured
       using the average of the magnitudes of the solution at the
       beginning and end of a step.  The error estimate formula has
       been grouped to control loss of significance.  To distinguish
       the various arguments, h is not permitted to become smaller
       than 26 units of roundoff in t.  Practical limits on the
       change in the stepsize are enforced to smooth the stepsize
       selection process and to avoid excessive chattering on
       problems having discontinuities.  To prevent unnecessary
       failures, the code uses 9/10 the stepsize it estimates will
       succeed.  After a step failure, the stepsize is not allowed
       to increase for the next attempted step.  This makes the code
       more efficient on problems having discontinuities and more
       effective in general since local extrapolation is being used
       and extra caution seems warranted.

       Test number of derivative function evaluations.  If okay,
       try to advance the integration from t to t+h. }

200:  if (nfe <= maxnfe) then goto 220;
```

```
      {Too much work.}

      iflag := 4;
      kflag := 4;
      exit;

      {Advance an approximate solution over one step of length h.}

220:  fehl(neqn, y, t, h, yp, f1, f2, f3, f4, f5, f1);
      nfe := nfe + 5;

      {Compute and test allowable tolerances versus local error
       estimates and remove scaling of tolerances.  Note that
       relative error is measured with respect to the average of
       the magnitudes of the solution at the beginning and end of
       the step.}

      eeoet := 0.0e0;
      for k := 1 to neqn do begin
          et := abs( y[k] ) + abs( f1[k] ) + ae;
          if et > 0.0e0 then goto 240;

          {Inappropriate error tolerance.}

          iflag := 5;
          exit;

240:      ee := abs( (-2090.0e0*yp[k] + (21970.0e0*f3[k]
                                         - 15048.0e0*f4[k])) +
                    (22528.0e0*f2[k] - 27360.0e0*f5[k]) );
          eeoet := amax1( eeoet, ee/et );
      end;

      esttol := abs( h )*eeoet*scale/752400.0e0;

      if esttol <= 1.0e0 then goto 260;

      {Unsuccessful step...reduce the stepsize, try again.  The
       decrease is limited to a factor of 1/10.}

      hfaild := true;
      output := false;
```

```
      s := 0.1e0;
      if esttol < 59049.0e0 then s:= 0.9/power( esttol, 0.2e0 );
      h := s*h;
      if abs(h) > hmin then goto 200;

      {Requested error unattainable at smallest allowable stepsize.}

      iflag := 6;
      kflag := 6;
      exit;

      {Successful step...store solution at t+h and evaluate
       derivatives there.}

260:  t := t + h;
      for k := 1 to neqn do y[k] := f1[k];
      a := t;
      F(a, y, yp );
      nfe := nfe + 1;

      {Choose next stepsize.  The increase is limited to a factor of
       5.  If step failure has just occurred, next stepsize is not
       allowed to increase.}

      s := 5.0e0;
      if esttol > 1.889568e-4 then s:= 0.9e0/power( esttol, 0.2e0 );
      if hfaild then s := amin1( s, 1.0e0 );
      h := sign( amax1( s*abs(h), hmin ), h );

      {End of core integrator.}

      {Should we take another step? }

      if output then goto 300;
      if iflag > 0 then goto 100;

      {Integration successfully completed.}

      {One-step mode.}

      iflag := -2;
      exit;
```

```
     {Interval mode.}

300: t := tout;
     iflag := 2;
     exit;

end; {*** RKF45 ***}
```

```
program RKF45_Driver_Program;

{Written by Lawrence A. Jones.
 Only change those parameters marked off with comment lines!}

type
    {The dimension of the following array should be equal to the
     number of differential equations (number of states) in the
     system.}

  {--------------------------------}
    savetype = array[1..4] of real;
  {--------------------------------}

   file_stat  = ( read_it, write_it );
   fname_type = string[80];

var
   workkm1, workk6, workk7, workk8, workk9          :real;
   y, work1, workk1, workk2, workk3, workk4, workk5 :savetype;
   iwork1, iwork2, iwork3, iwork4, iwork5, iwork6   :integer;
   t, relerr, abserr, tfinal, tprint, tout          :real;
   neqn, iflag                                      :integer;
   error                                            :boolean;

   tfilename              :fname_type;
   tfil                   :Text;
   ch, direction          :char;
   dot, i                 :integer;
   result                 :boolean;

procedure Open_File( fname  : fname_type; var fil    : Text;
                     status : file_stat;  var result : boolean );
var
   okay, exist, badname :boolean;
   temp                 :char;

begin
     {$I-}              {turn off automatic file error checking}
     result := true;
     assign( fil, fname );
     reset( fil );
```

```
     exist := (IOresult = 0);
     if status = write_it then
          if exist then begin           {trying to write to a file}
               write( 'Overwrite? (y/n) ');          {that exists}
               repeat
                    readln( temp );
               until ( temp in ['y','n'] );
               if temp = 'y' then begin     {erase file and reopen}
                    close( fil );
                    erase( fil );
                    rewrite( fil );
                  end {then}
               else result := false;{user doesn't want file erased}
             end {then}
          else rewrite( fil )
     else if status = read_it then      {if exist then the file is}
         if not exist then begin                    {already reset}
              writeln( 'file not found' );
              result := false;
         end;
     {$I+}
end;

function power( a, b :real ):real;
begin
    power := exp( b*ln( a ) );
end;

function isign( a, b :integer ):integer;
begin
    if b < 0.0 then  isign := -a  else  isign := a;
end;

function sign( a, b :real ):real;
begin
    if b < 0.0 then  sign := -a  else  sign := a;
end;

function amax1( a, b :real ):real;
begin
    if a >= b then  amax1 := a  else  amax1 := b;
end;
```

```
function amin1( a, b :real ):real;
begin
    if a <= b then  amin1 := a  else  amin1 := b;
end;

procedure f( t:real; var y, yp:savetype );

{Place the equations to evaluate the state derivative, i.e. the
 right-hand side of the state equations in this procedure.  t is
 the current time, y is the current state, and yp is the new state
 derivative (i.e. yp = f(t,y) ). }

begin
 {-------------------------------------------------------------}
   yp[1] := ;
   yp[2] := ;
   yp[3] := ;
   yp[4] := ;
 {-------------------------------------------------------------}
end;

{$I c:\user\schnuff\prog\misc\rkf45}

begin {rkf45driver}
     writeln;
     writeln( '^C to abort' );
     repeat
          write('Target .M file ---> ');
          read( tfilename );
          dot := Pos( '.', tfilename ); {find the dot in tfilename}
          if dot = 0 then begin {user didn't use a dot,append a .M}
               tfilename := tfilename + '.M';
               writeln( '.M' );
            end
           else writeln;
          Open_File( tfilename, tfil, write_it, result );
     until result;
     write( tfil, 'Data = [' );
     writeln( 'Writing data...' );
   {-------------------------------------------------------------}
```

```
  {Initial time.}
  t := 0.0;

  {Number of equations (states) in your system.  This should
   agree with the array size in savetype! }
  neqn := 5;

  {Initial conditions of the states.}
  y[1] := -6.5519766;
  y[2] := -461.97532e-6;
  y[3] := -1;
  y[4] := 2.6447298e-3;
  y[5] := 0;

  {Solution tolerances.  These will dramatically influence the
   speed of the simulation.  Relerr=abserr=1.0e-5 --> fast,
   relerr= 1.0e-9 and abserr=0 --> slow.}
  relerr := 1.0e-5;
  abserr := 1.0e-5;

  {Final time.}
  tfinal := 0.14;

  {Output timestep.}
  tprint := 2e-4;

{-------------------------------------------------------------}

  {Mode:  1=normal mode, -1=one-step mode.  (See RKF45 source
   code for details.)}
  iflag := 1;

  tout := t;
  error := false;

  while (tout <= tfinal) and (not error) do begin

     rkf45( neqn, y, t, tout, relerr, abserr, iflag, work1,
            workkm1, workk1, workk2, workk3, workk4, workk5,
            workk6, workk7, workk8, workk9,
            iwork1, iwork2, iwork3, iwork4, iwork5 );
```

```
  {Format of output.}

{-------------------------------------------------------------}
writeln( t:10:4, y[1]:12:4, y[2]:12:4, y[3]:12:4, y[4]:12:4,
         y[5]:12:4 );
writeln( tfil, t:14, ' ',
              y[1]:14, ' ', y[2]:14, ' ', y[3]:14, ' ',
              y[4]:14, ' ', y[5]:14, ';' );
{-------------------------------------------------------------}

  {Error messages.}
  case iflag of
   -2 : tout := t + tprint;
    2 : tout := t + tprint;
    3 : begin
          writeln;
          write(  'Integration was not completed because ');
          writeln('relative error tolerance was too small.');
          write('Relerr has been increased and integration ');
          writeln('is continuing.');
          writeln;
        end;
    4 : begin
          writeln;
          write(  'Integration was not completed because ');
          writeln('more than 32767 derivative evaluations');
          write(  'were needed.  This is approximately 500 ');
          writeln('steps.  Continuing integration.');
          writeln;
        end;
    5 : begin
          writeln;
          write(  'Integration was not completed because ');
          writeln('solution vanished making a pure relative');
          write( 'error test impossible.  Abserr being set ');
          writeln('to 1.0e-4 and integration is');
          writeln('continuing.');
          writeln;
          abserr := 1.0e-4;
        end;
    6 : begin
```

```
            writeln;
            write(  'Integration was not completed because ');
            writeln('requested accuracy could not be');
            write(  'achieved using smallest allowable step');
            writeln('size.  Relerr being scaled by 10 and');
            writeln('integration is continuing.');
            writeln;
            relerr := 10*relerr;
            iflag := 2;
           end;
       7 : begin
            writeln;
            write(  'It is likely that RKF45 is inefficient ');
            writeln('for solving this problem!  Too much');
            write(  'output is restricting the natural step');
            writeln('size choice.  Continuing integration.');
            writeln;
            iflag := 2;
           end;
       8 : begin
            writeln;
            write(  'Invalid input parameters!  One of the ');
            writeln('following is true:');
            writeln;
            writeln('     neqn <= 0');
            writeln('     t=tout and iflag <> +1 or -1');
            writeln('     relerr or abserr < 0');
            writeln('     iflag = 0 or < -2 or > 8');
            error := true;
           end;
     end;
    end; {while}
    writeln( tfil, '];' );
    write( tfil, chr(26) );  {write a ^Z to end the text file}
    flush( tfil );           {flush the buffer to the disk}
    close( tfil );
    write( '...finished.' );
end. {rkf45driver}
```

Appendix B

Simulation Program

A program was written to simulate the motor and/or the observer. The motor is simulated using RKF45, given in Appendix A. The observer is simulated using the improved Euler method. In addition to running simulations of the motor and observer, the simulation program provides an interface that allows the user to set simulation parameters, examine and plot the simulation data, retrieve data from disk, and save data to disk in Lotus 123 or PC-MATLAB format. The program is somewhat lengthy and is not included here in its entirety in the interests of brevity. Listed below are excerpts from the program corresponding to the Figure 5.2. The procedures and functions included are

`Compute_I`	computes output current from the state;
`Compute_V`	computes the driving voltage for the motor;
`F`	evalutates the motor state derviative for RKF45;
`Step_the_Observer`	steps the observer forward one step;
`Compute_Obs_Deriv`	computes the observer state derivative;
`Integrate_Observer`	integrates the observer forward in time;
`Run_Simulation`	simulates the motor and observer.

```
function Compute_I( state:state_record; phase:phase_type ):real;
{Computes the output of the state space system from the state
 passed to it in the parameter list.}
begin
     with motor_parameters do begin
          case phase of
               phaseA : Compute_i := state.direct;
               phaseB : Compute_i := state.quadrature;
          end;
     end;
end;
{-------------------------------------------------------------------}
function Compute_V( theta :real; phase :phase_type;
                                 max_phase_voltage :real):real;
{Computes the open-loop stator voltages used to drive the motor.}
begin
  case phase of
    phaseA:Compute_v := max_phase_voltage*cos(theta);
    phaseB:Compute_v := max_phase_voltage*cos(theta-two_pi_over_3);
    phaseC:Compute_v := max_phase_voltage*cos(theta+two_pi_over_3);
  end;
end;
{-------------------------------------------------------------------}
procedure F(time:real; var y, yp:savetype; var global_parms:real);
{This procedure is called by RKF45 to calculate the state
 derivative, i.e. the right-hand side of the state equations.
 Variable time is the current time, y is the current state, and yp
 is the state derivative (i.e. yp = f(y,t) ).}
var
   Va, Vb, Vc : real;
begin
   RKF_theta := RKF_theta + e_omega*(time - last_time);
   Va  := Compute_V( RKF_theta, phaseA, max_phase_voltage );
   Vb  := Compute_V( RKF_theta, phaseB, max_phase_voltage );
   Vc  := Compute_V( RKF_theta, phaseC, max_phase_voltage );
   last_time := time;
   with motor_parameters do begin
       {Stator frame 2-phase currents driven by
        stator frame 3-phase voltages.}
        yp[1] :=  -( Rs/Ls )*y[1]
                  + ( K*y[3]/Ls )*sin( y[4] )
                  + ( 1/Ls )*(sqrt23*Va - sqrt16*Vb - sqrt16*Vc);
```

```
        yp[2] :=  -( Rs/Ls )*y[2]
                  - ( K*y[3]/Ls )*cos( y[4] )
                  + ( 1/Ls )*(sqrt12*Vb - sqrt12*Vc);
        yp[3] :=   ( K*N*N/H )*(-y[1]*sin(y[4]) + y[2]*cos(y[4]))
                  - ( B/H )*y[3]
                  - (C*N/H)*sign(1,y[3])
                  - N*tau_L/H;
        yp[4] :=   y[3];
   end;
end;

{-----------------------------------------------------------------}
procedure Step_the_Observer( var time :real; t_end :real;
                             cva, cvb, cvc, cia, cib :io_pointer;
                             current_obs_state,
                             new_obs_state :state_pointer );
{- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -}
function Compute_Obs_Deriv( state      :state_type;
                            old_state  :state_record;
                            theta_park :real;
                            var va, vb, vc, i1, i2 :real  ):real;
var
   delta_d, delta_q :real;
begin
   delta_d :=  i1*cos(theta_park) + i2*sin(theta_park)
                                          - old_state.direct;
   delta_q := -i1*sin(theta_park) + i2*cos(theta_park)
                                          - old_state.quadrature;
   with observer_parameters do begin
     case state of {2-phase rotor frame currents;
                    3-phase stator frame voltages}
       direct    :Compute_Obs_Deriv := -( Rs/Ls )*old_state.direct
                                + old_state.w*old_state.quadrature
                                + ( 1/Ls )*(  cos( theta_park )*(
                                                sqrt23*va
                                              - sqrt16*vb
                                              - sqrt16*vc )
                                            + sin( theta_park )*(
                                                sqrt12*vb
                                              - sqrt12*vc ))
                                + gain[1,1]*( delta_d )
                                + gain[1,2]*( delta_q );
```

```
      quadrature:Compute_Obs_Deriv :=
                                -( Rs/Ls )*old_state.quadrature
                               - old_state.w*old_state.direct
                               - ( K/Ls )*old_state.w
                               + ( 1/Ls )*( -sin( theta_park )*(
                                                  sqrt23*va
                                              - sqrt16*vb
                                              - sqrt16*vc )
                                         + cos( theta_park )*(
                                                  sqrt12*vb
                                              - sqrt12*vc ))
                               + gain[2,1]*( delta_d )
                               + gain[2,2]*( delta_q );
       w          :Compute_Obs_Deriv :=
                        ( K*N*N/H )*( old_state.quadrature
                                         + gain[3,1]*( delta_d )
                                         + gain[3,2]*( delta_q ))
                               - ( B/H )*old_state.w
                               - (C*N/H)*sign(1,old_state.w)
                               - N*tau_L/H;
       theta      :Compute_Obs_Deriv := old_state.w;
     end; {case}
   end; {with}
end; {Compute_Obs_Deriv}
{- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -}
procedure Integrate_Observer;
var
   state_deriv, next_state :state_record;
   delta_t, Va, Vb, Vc, I1, I2 :real;
begin
     delta_t := t_end - time;
     {With noise.}
     (*Va := cva^.value + grng( 0.0, 0.20, idum );
     Vb := cvb^.value + grng( 0.0, 0.20, idum );
     Vc := cvc^.value + grng( 0.0, 0.20, idum );
     I1 := cia^.value + grng( 0.0, 0.04, idum );
     I2 := cib^.value + grng( 0.0, 0.04, idum );*)
     {Without noise.}
     Va := cva^.value;
     Vb := cvb^.value;
     Vc := cvc^.value;
     I1 := cia^.value;
```

```
  I2 := cib^.value;

  (* Improved Euler's Method *)

state_deriv.direct := Compute_Obs_Deriv(
                          direct, current_obs_state^.state,
                          current_obs_state^.state.theta,
                          Va, Vb, Vc, I1, I2 );
state_deriv.quadrature :=
                 Compute_Obs_Deriv(
                          quadrature, current_obs_state^.state,
                          current_obs_state^.state.theta,
                          Va, Vb, Vc, I1, I2 );
state_deriv.w := Compute_Obs_Deriv(
                          w, current_obs_state^.state,
                          current_obs_state^.state.theta,
                          Va, Vb, Vc, I1, I2 );
state_deriv.theta := Compute_Obs_Deriv(
                          theta, current_obs_state^.state,
                          current_obs_state^.state.theta,
                          Va, Vb, Vc, I1, I2 );

next_state.direct     := current_obs_state^.state.direct
                             + delta_t*state_deriv.direct;
next_state.quadrature := current_obs_state^.state.quadrature
                             + delta_t*state_deriv.quadrature;
next_state.w          := current_obs_state^.state.w
                             + delta_t*state_deriv.w;
next_state.theta      := current_obs_state^.state.theta
                             + delta_t*state_deriv.theta;

new_obs_state^.state.direct :=  current_obs_state^.state.direct
                              + (delta_t/2)*( state_deriv.direct
                              + Compute_Obs_Deriv( direct,
                                    next_state,
                                 current_obs_state^.state.theta,
                                       Va, Vb, Vc, I1, I2 ) );
new_obs_state^.state.quadrature :=
                             current_obs_state^.state.quadrature
                           + (delta_t/2)*( state_deriv.quadrature
                           + Compute_Obs_Deriv( quadrature,
                                 next_state,
```

```
                                    current_obs_state^.state.theta,
                                    Va, Vb, Vc, I1, I2 ) );
  new_obs_state^.state.w :=   current_obs_state^.state.w
                            + (delta_t/2)*( state_deriv.w
                            + Compute_Obs_Deriv( w,
                                    next_state,
                                    current_obs_state^.state.theta,
                                          Va, Vb, Vc, I1, I2 ) );
   new_obs_state^.state.theta :=  current_obs_state^.state.theta
                                + (delta_t/2)*( state_deriv.theta
                                + Compute_Obs_Deriv( theta,
                                    next_state,
                                    current_obs_state^.state.theta,
                                    Va, Vb, Vc, I1, I2 ) );
end;  {Integrate Observer}
{- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -}
begin {Step_the_Observer}
     Integrate_Observer;
     new_obs_state^.state.time := t_end;
     if not motor_on then time := t_end;
end;  {Step_the_Observer}

{-------------------------------------------------------------------}
procedure Run_Simulation;
var
     k              :integer;      {Number of steps counter.}
     interrupted    :boolean;      {Stop the simulation.}
     i              :integer;      {Local counter.}
     saved_RKF_mode :integer;
begin {Run_Simulate}
     Box( 15, open, low );    {Simulator message box.}
     Goto_window( 0 );
     Goto_window( 15 );       {Simulator step box.}
     ClrScr;
     LowVideo;
     Goto_window( 17 ); ClrScr;
     GotoXY( 1, 1 );
     writeln('No. of steps = ', number_of_steps:5 );
     write( 'Current step = ');

     {Initialize random number seed for noise generation.}
     idum := -( 1 + trunc(random(1)) );
```

```
gliset := 0;

{Initialize the array that passes the states to RKF45.}
RKF45_state[1] := initial_motor_state.direct;
RKF45_state[2] := initial_motor_state.quadrature;
RKF45_state[3] := initial_motor_state.w;
RKF45_state[4] := initial_motor_state.theta;

saved_RKF_mode := iflag;
t := t_initial;
tout := t + tprint;
error := false;
interrupted := false;

if motor_sim_on then begin
      {If the motor simulator is running then we want to
       initialize the motor state list and the voltage and
       current lists.  In particular, we reclaim the heap
       space by disposing of all lists.}

      Goto_window( 15 );
      ClrScr; NormVideo; writeln( 'Disposing of lists.' );
      write( 'Patience please.' );
      Dispose_State_List( first_motor_state );
      Dispose_State_List( first_observer_state );
      Dispose_VI_List( first_va );
      Dispose_VI_List( first_vb );
      Dispose_VI_List( first_vc );
      Dispose_VI_List( first_ia );
      Dispose_VI_List( first_ib );
      ClrScr;

      {Create the first data node in the required lists.}
      new( new_motor_state );
      new_motor_state^.next_state := nil;
      current_motor_state := new_motor_state;
      first_motor_state := new_motor_state;
      with first_motor_state^.state do begin
           direct := initial_motor_state.direct;
           quadrature := initial_motor_state.quadrature;
           w        := initial_motor_state.w;
           theta    := initial_motor_state.theta;
```

```
    time     := t_initial;
end;

e_omega := w_steady_state;{driving voltage vector speed}
e_theta := 0;          {driving voltage vector position}

{Compute and store the currents generated by this
 initial value of the state.  NOTE how this yields the
 output currents as a function of the step just
 completed.  The observer will, as it now stands,
 use the value of voltage driving the motor at the end
 of the LAST step and the currents computed at the end
 of the LAST step.}

new( new_ia );
new_ia^.next := nil;
current_ia := new_ia;
first_ia := new_ia;
current_ia^.value:=Compute_I(current_motor_state^.state,
                             phaseA);
new( new_ib );
new_ib^.next := nil;
current_ib := new_ib;
first_ib := new_ib;
current_ib^.value:=Compute_I(current_motor_state^.state,
                             phaseB);
{Store phase voltage inputs that will drive the motor
 during the NEXT time step.}

new( new_va );
new_va^.next := nil;
current_va := new_va;
first_va := new_va;
current_va^.value := Compute_V( e_theta, phaseA,
                                max_phase_voltage );
new( new_vb );
new_vb^.next := nil;
current_vb := new_vb;
first_vb := new_vb;
current_vb^.value := Compute_V( e_theta, phaseB,
                                max_phase_voltage );
new( new_vc );
```

```
        new_vc^.next := nil;
        current_vc := new_vc;
        first_vc := new_vc;
        current_vc^.value := Compute_V( e_theta, phaseC,
                                        max_phase_voltage );
        if data_viewer_on and m_viewer_on then begin
            Goto_window( 9 );       {Viewer Box}
            GotoXY( 1, 8 );
            InsLine;
            LowVideo;
            write(t:10,'    ',first_motor_state^.state.direct:10,
                  '  ',
                   first_motor_state^.state.quadrature:10, '  ',
                   first_motor_state^.state.w:10, '  ',
                   first_motor_state^.state.theta:10 );
            Goto_window( 0 );
        end;
    end; {if motor_sim_on}

    if observer_sim_on then begin
        {If the observer simulator is on then we want to
         initialize the observer state list.  Note that if only
         the observer is on then the voltage and current lists
         will not be initialized (that's done only if the motor
         is on) since it is assumed that current and voltage
         lists already exist (usually from experimentally
         collected data).  If the current and voltage lists
         don't exist the simulation terminates immediately.}

        Goto_window( 15 );
        ClrScr; NormVideo; writeln( 'Disposing lists.' );
        Dispose_State_List( first_observer_state );
        ClrScr;
        if (first_va = nil) or (first_vb = nil) or
           (first_vc = nil) or (first_ia = nil) or
           (first_ib = nil) then begin
                Goto_window( 15 );
                GotoXY( 1, 2 );
                NormVideo;
                ClrScr; write('End of V or I list.');
                Goto_window( 0 );
                interrupted := true;
```

```
        error := true;
    end
else begin
    {Create the first data node.}
    new( new_observer_state );
    new_observer_state^.next_state := nil;
    current_observer_state := new_observer_state;
    first_observer_state := new_observer_state;
    with first_observer_state^.state do begin
        direct    := initial_observer_state.direct;
        quadrature:= initial_observer_state.quadrature;
        w         := initial_observer_state.w;
        theta     := initial_observer_state.theta;
        time      := t_initial;
    end;

    if motor_sim_on or (not continued) then begin
        current_va := first_va;
        current_vb := first_vb;
        current_vc := first_vc;
        current_ia := first_ia;
        current_ib := first_ib;
       end
    else if continued then begin
              first_va := current_va;
              first_vb := current_vb;
              first_vc := current_vc;
              first_ia := current_ia;
              first_ib := current_ib;
    end;
    if not continued then begin
        old_va := first_va^.value;
        old_vb := first_vb^.value;
        old_vc := first_vc^.value;
    end;
    if data_viewer_on and o_viewer_on then begin
       Goto_window( 9 );    {Viewer Box}
       GotoXY( 1, 8 );
       InsLine;
       LowVideo;
       write(t:10,     ',
             first_observer_state^.state.direct:10,
```

```
                    ' ',
                    first_observer_state^.state.quadrature:10,
                     ' ',
                    first_observer_state^.state.w:10, '  ',
                    first_observer_state^.state.theta:10 );
                Goto_window( 0 );
            end;
        end; {else}
    end; {if observer_sim_on}

    k := 1;
    date := GetDate;
    sim_time := GetTime;
    Write_Title;


    {--------- Begin the simulation ---------}

    while (k <= number_of_steps) and (not error) and
      (not interrupted) do begin

       Goto_window( 17 );   {Write current step number on screen.}
       NormVideo;
       GotoXY( 16, 2 ); write( k:5 );
       Goto_window( 0 );

       {We first simulate the observer, if it's on, because it
        uses the value of the output current computed at the
        end of the last simulation step. Therefore, we should
        update the observer first since running the motor forward
        a time step will yield the values of current at the END of
        THIS time step.}

       if observer_sim_on then begin
          {Solve observer equations forward one step.  A very
           simple forward integration algorithm is used here as
           opposed to the very sophisticated algorithm used for
           the motor simulation.  The reason is that in real time
           only a simple integration will be possible and
           therefore we should run the observer here like we
           ultimately will in real time.  The motor simulation on
           the other hand should be as accurate as possible here
```

```
 in order to accurately model the real-world motor.}

if (current_va = nil) or (current_vb = nil) or
   (current_vc = nil) or (current_ia = nil) or
   (current_ib = nil) then begin
       Goto_window( 15 );
       GotoXY( 1, 2 );
       NormVideo;
       ClrScr; write('End of V or I list.');
       Goto_window( 0 );
       interrupted := true;
       error := true;
   end
else begin
     {Create another node in the observer state list to
      receive the results of this next observer step.}
     new( new_observer_state );
     new_observer_state^.next_state := nil;
     current_observer_state^.next_state :=
                                      new_observer_state;
     {Don't move current_observer_state yet as we need
      it to step the observer forward.}

     Step_the_Observer(t, tout, current_va, current_vb,
                       current_vc,current_ia,current_ib,
                       current_observer_state,
                       new_observer_state );

     {The theta state can be allowed to simply ramp or
      it can be mod 2Pi.  Past simulation experience has
      shown that it doesn't matter which way you do it.
      It's generally easier to process the data
      afterward, however, if the angle is simply allowed
      to ramp up since the error plots will not have
      spikes.  Include the following code for 2Pi
      operation.}
     (*if new_observer_state^.state.theta > Pi then
          new_observer_state^.state.theta :=
              new_observer_state^.state.theta - 2*Pi;*)
     current_observer_state := new_observer_state;
     old_va := current_va^.value;
     old_vb := current_vb^.value;
```

```
            old_vc := current_vc^.value;

            {Note that we don't have to load the states into
             the list as in the motor simulation because
             Step_the_Observer does this for us.  Also note
             that even though we've simulated the observer
             before the motor, it is desired to print the motor
             states for this step first instead of the current
             observer state so that we will see motor data,
             observer data, motor data, observer data, etc.
             Therefore, we will print the current observer
             state after the motor is simulated (i.e. see
             below).}

            if (not motor_sim_on) then  {We have to maintain
                     the V and I lists.  Otherwise, the motor
                     simulator does this as it creates the V
                     and I lists.  We also have to maintain the
                     time.}
            begin
                 current_va  := current_va^.next;
                 current_vb  := current_vb^.next;
                 current_vc  := current_vc^.next;
                 current_ia  := current_ia^.next;
                 current_ib  := current_ib^.next;
                 current_aux := current_aux^.next;
                 tout := t + tprint;
            end;
       end; {else}
    end; {if observer_sim_on}

    if motor_sim_on then begin

       (*if e_omega < 314.1592654 then  begin {ramping omega}
            e_omega := e_omega + 1500*tprint; end
       else e_omega := e_omega;*)
       e_theta := e_theta + e_omega*tprint;

       {Solve motor equations forward in time one step.  RKF45
        is a Runge-Kutta-Fehlberg Fourth-Fifth order
        differential equation solver.  It is a very
        sophisticated routine and will take intermediate steps
```

```
 if necessary in order to maintain the local error less
 than relerr and the global error less than abserr.  It
 is used to solve the motor equations in order to
 provide the best possible real world modeling of the
 motor.}

var_relerr := relerr;
var_abserr := abserr;


RKF45( neqn, RKF45_state, t, tout, var_relerr,
       var_abserr, iflag, work1, workkm1, workk1,
       workk2, workk3, workk4, workk5, workk6, workk7,
       workk8, workk9, iwork1, iwork2, iwork3, iwork4,
       iwork5 );

{Store these new values of the states, which represent
 the state of the system at the END of this time step,
 in the motor list.}
new( new_motor_state );
new_motor_state^.next_state := nil;
current_motor_state^.next_state := new_motor_state;
current_motor_state := new_motor_state;

{Unload RKF45 array in preparation for next call.}
{Include the following lines for mod 2Pi operation.}
(*if RKF45_state[ 4 ] > Pi then
     RKF45_state[ 4 ] := RKF45_state[ 4 ] - 2*Pi;*)
with current_motor_state^.state do begin
     direct     := RKF45_state[ 1 ];
     quadrature := RKF45_state[ 2 ];
     w          := RKF45_state[ 3 ];
     theta      := RKF45_state[ 4 ];
     time       := t;
end;

{Store phase voltage inputs used to drive the NEXT time
 step.  Note that if the observer is on then on the next
 pass through the while it will use these voltages.
 Again, these are the voltages "sampled" at the
 beginning of the next time step.}
```

```
new( new_va );
new_va^.next := nil;
current_va^.next := new_va;
current_va := new_va;
current_va^.value := Compute_V( e_theta, phaseA,
                                    max_phase_voltage );

new( new_vb );
new_vb^.next := nil;
current_vb^.next := new_vb;
current_vb := new_vb;
current_vb^.value := Compute_V( e_theta, phaseB,
                                    max_phase_voltage );

new( new_vc );
new_vc^.next := nil;
current_vc^.next := new_vc;
current_vc := new_vc;
current_vc^.value := Compute_V( e_theta, phaseC,
                                    max_phase_voltage );
{Compute and store the currents generated by the present
 value of the state.  Note that these are the currents
 that would be available to the observer at the
 beginning of the next time step.}

new( new_ia );
new_ia^.next := nil;
current_ia^.next := new_ia;
current_ia := new_ia;
current_ia^.value:=Compute_I(current_motor_state^.state,
                                                phaseA);

new( new_ib );
new_ib^.next := nil;
current_ib^.next := new_ib;
current_ib := new_ib;
current_ib^.value:=Compute_I(current_motor_state^.state,
                                                phaseB);

if data_viewer_on and m_viewer_on then begin
    Goto_window( 9 );       {Viewer Box}
    GotoXY( 1, 8 );
    InsLine;
    NormVideo;
    write(current_motor_state^.state.time:10, '    ',
          current_motor_state^.state.direct:10, '   ',
```

```
                current_motor_state^.state.quadrature:10,'  ',
                current_motor_state^.state.w:10, '  ',
                current_motor_state^.state.theta:10 );
            Goto_window( 0 );
        end;

      {Now determine if the RKF45 was able to complete the
       requested time step successfully by examining iflag.
       If not, send a message and take action appropriate for
       the error.}
      case iflag of
           2              : ;
          -2,3,4,5,6,7,8 : begin
                               Goto_window( 15 );
                               ClrScr;
                               NormVideo;
                           write( 'RKF45 is reporting iflag = ',
                                      iflag );
                               Goto_window( 0 );
                               interrupted := true;
                               error := true;
                            end;
      end;
    end; {if}  {if motor_sim_on}

    if observer_sim_on then begin
         if data_viewer_on and o_viewer_on then begin
                Goto_window( 9 );    {Viewer Box}
                GotoXY( 1, 8 );
                InsLine;
                NormVideo;
                write(current_observer_state^.state.time:10,
                   '  ',
                   current_observer_state^.state.direct:10,
                   '  ',
                   current_observer_state^.state.quadrature:10,
                   '  ',
                   current_observer_state^.state.w:10, '  ',
                   current_observer_state^.state.theta:10 );
                Goto_window( 0 );
         end; {if}
    end; {if}
```

```
   {Check to see if there's any more memory.  Recall that
    MaxAvail returns a negative number if there's more than
    64K of heap memory.  Therefore, we adjust the value
    returned by MaxAvail if necessary.}
   heap_available := MaxAvail;
   if heap_available < 0 then heap_available :=
                                    heap_available + 65536.0;

   {Now we see if there's enough heap memory to accomodate
    another motor list entry AND another observer list entry.
    Recall that MaxAvail returns the number of free paragraphs
    where a paragraph equals 16 bytes and also recall that
    Turbo-87 uses 8 byte reals (!).}
   if (heap_available < 6) then begin
         Goto_window( 15 );
         GotoXY( 1, 2 );
         NormVideo;
         ClrScr; write('OUT OF MEMORY!');
         Goto_window( 0 );
         interrupted := true;
         error := true;
   end;
   if KeyPressed then interrupted := true;
   tout := t + tprint;
   k := k + 1;
end; {while}

{Write simulation completion status in the simulation message
 window.}
Goto_window( 15 );
if interrupted then begin
     GotoXY( 1, 2 ); NormVideo; write( 'Simulation Stopped' );
     Whistle;
  end
else begin
     Goto_window( 15 );
     ClrScr; NormVideo; writeln( 'Simulation Complete' );
     write( 'Heapspace = ', heap_available:10, ' paragraphs' );
     Whistle;
end;
Goto_window( 0 );
```

```
      iflag := saved_RKF_mode;
end; {Run_Simulation}
```

Appendix C

Root Locus Software

```
function [rootsr,rootsi,w]=rootloc(wlo, whi, n, g)

% [rootsr,rootsi,w] = rootloc( wlo, whi, n, g )
%
% This PC-MATLAB m-file computes a root locus with respect to speed
% of the linearized error dynamics of the estimated-innovation
% observer given in Section 6.1.  The motor is assume to be
% controlled such that Id is zero.  The inputs are the range of
% speeds over which the locus should be calculated in the form of a
% minimum speed, wlo, and a maximum speed, whi.  Parameter n is the
% number of speeds within the given range at which to compute the
% poles.  Parameter g is a 3x2 MATRIX of observer gains.  The
% outputs are vectors of the real part of the locus and imaginary
% part of the locus and the mechanical speeds at which these roots
% were computed.

% Define the motor constants

K = 0.1105;
L = 0.444e-3;
N = 3;
R = 0.39;
B = 3.7e-3;
C = 0.583;
H = 3.55e-2;
Tau = 1.6;
id = 0;

deltaw = (whi-wlo)/(n-1);
```

```
for i = wlo:deltaw:whi, w = [w i]; end
for i = w,

        % Compute the motor steady-state.

        omega = (1/60)*2*pi;      % Mechanical speed.
        we    = N*omega;          % Electrical speed.
        iq    = (B*omega)/(K*N) + Tau/(K*N) + C/(K*N);
        sigma = R^2 + (omega*L*N)^2;
        V     = sqrt((K^2*omega^4*L^2*N^4...
                              + (iq*sigma + R*K*omega*N)^2)/sigma);
        alpha =  -acos( (-iq*sigma - R*K*omega*N)...
                                              /(V*sqrt(sigma)) )/N...
                 + atan( R/(omega*L*N) )/N;   % Mechanical angle.
        alphae = N*alpha;   % Electrical torque angle.
        Vd =  V*cos(alphae);
        Vq = -V*sin(alphae);

        % Compute the A matrix.

        A      =  zeros(4,4);
        A(1,1) = -(R/L) - g(1,1);
        A(1,2) =  we - g(1,2);
        A(1,3) =  iq;
        A(1,4) =  g(1,1)*iq - g(1,2)*id + (1/L)*Vq;
        A(2,1) = -we - g(2,1);
        A(2,2) = -(R/L) - g(2,2);
        A(2,3) = -(K/L) - id;
        A(2,4) =  g(2,1)*iq - g(2,2)*id - (1/L)*Vd;
        A(3,1) = -g(3,1)*K*N^2/H;
        A(3,2) =  (1 - g(3,2))*K*N^2/H;
        A(3,3) = -B/H;
        A(3,4) =  K*N^2*(g(3,1)*iq - g(3,2)*id)/H;
        A(4,1) =  0;
        A(4,2) =  0;
        A(4,3) =  1;
        A(4,4) =  0;

        % Compute the eigenvalues of the A matrix and store in
        % output vectors.

        eigenval = eig(A);
```

```
        rootsr = [rootsr;real(eigenval)];
        rootsi = [rootsi;imag(eigenval)];

        % Prepare to plot the locus.

        root1r = [root1r;real(eigenval(1))];
        root2r = [root2r;real(eigenval(2))];
        root3r = [root3r;real(eigenval(3))];
        root4r = [root4r;real(eigenval(4))];
        root1i = [root1i;imag(eigenval(1))];
        root2i = [root2i;imag(eigenval(2))];
        root3i = [root3i;imag(eigenval(3))];
        root4i = [root4i;imag(eigenval(4))];

end

% Plot the locus.

plot(root1r,root1i,root2r,root2i,root3r,root3i,root4r,root4i);
xlabel( 'Real' );
ylabel( 'Imaginary' );
%title( 'Root Locus With Respect To Speed' );
grid
```

Appendix D

Hardware Details

TMS32020 Development System

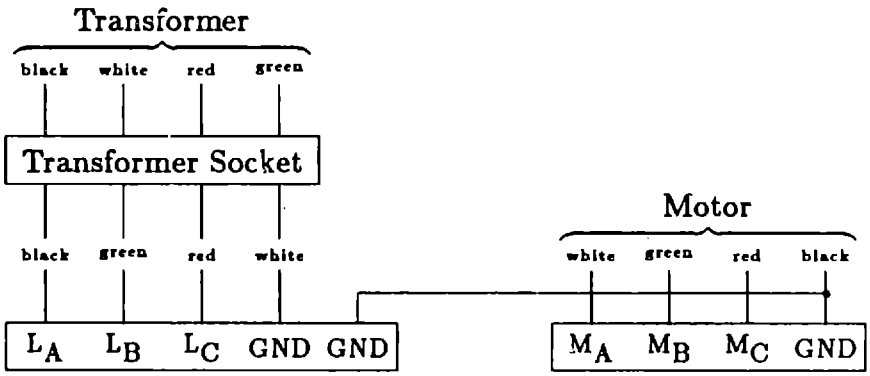
External Program RAM	>20 – >201F
External Data RAM	>400 – >23FF >8000 – >9FFF

When using the monitor, the user program RAM is >400 – >201F and the data memory locations >60, >61, and >62 are not available. The on-board interval timer has been selected as the sample clock and therefore link LK6a is present and link LK6b is absent. The white ribbon cable from the TMS32020 development system is:

Line 7		$\overline{\text{INT0}}$
Line 10		GND
Line 16		$\overline{\text{SCO}}$
Line 17		SH
Line 20	(green)	GND

Isolation Transformer

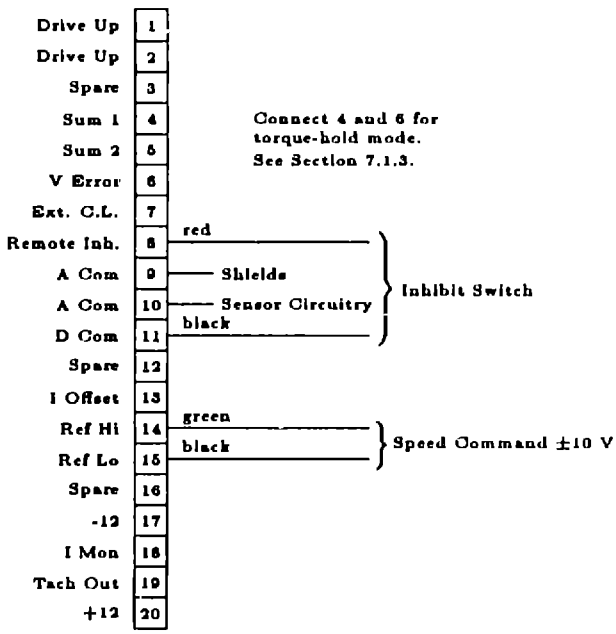
Primary:	Jumper	1 and 3, 2 and 4, 5 and 7, 6 and 8, 9 and 11, 10 and 12,	
	Come in on	1, 4, and 8	
Secondary:	Come out on	X7, X8, X9	for 181 V *l-l*
		X4, X5, X6	for 203 V *l-l*
		X1, X2, X3	for 224 V *l-l*



Connector 59

sin hi	1	red
sin lo	2	black
cos lo	3	black
cos hi	4	green
shield com	5	shields
ref hi	6	white
ref lo	7	black
shield com	8	shield
spare	9	nc
spare	10	nc

Connector 151



Appendix E

Data Collection Software

The following TMS32020 assembly language routine was written to collect motor data for use in the offline tests of the observer. The program is assembled using the Texas Instruments TMS32020 Assembler. The resulting .MPO file is loaded into the Loughborough development system using the Loughborough monitor. The program begins at >400 in the program memory. To stop the program, a breakpoint should be set at one of the NOPs in the background task.

The code as listed collects 2000 points from the resolver and six analog channels. The code may be easily modified to collect only 1000 points. The six analog channels are connected such that channels 0–5 are Va, Vb, Vc, Ia, Ib, and Bus, respectively. The sixth channel, Bus, is obtained by moving the wire connecting pin 9 of G1 and pin 3 of C1 so that it connects pin 9 of G1 with pin 2 of C1 in Figure 7.9. A voltage sensor was implemented for the sixth channel to measure the voltage of the DC bus. Collecting 1000 points, the program stores the data in the following data memory locations:

Resolver	>0400 – >07E7	
Channel 0	>07E8 – >0BCF	
Channel 1	>0BD0 – >0FB7	
Channel 2	>0FB8 – >139F	
Channel 3	>13A0 – >1787	
Channel 4	>1788 – >1B6F	
Channel 5	>1B70 – >1F57	(if used)

If 2000 points are collected, the program stores the data in the following data memory locations:

Resolver	>0400 – >0BCF	
Channel 0	>0BD0 – >139F	
Channel 1	>13A0 – >1B6F	
Channel 2	>1B70 – >233F	
Channel 3	>8000 – >87CF	
Channel 4	>87D0 – >8F9F	
Channel 5	>8FA0 – >976F	(if used)

The gap in memory between channels 3 and 4 is a result of the use of 8Kx8 static RAMs in the development system instead of 32Kx8 static RAMs.

```
        IDT   'MULTICHAN'

        OPTION DUNLST

*************************
*                       *
*       MULTICHAN       *
*                       *
*************************

* This is a TMS32020 assembly language routine.
* It collects data from the resolver and six analog
* channels using multiplexing hardware. The channels are
* Va, Vb, Vc, Ia, Ic, Bus.
* The Texas Instruments TMS32020 Assembler should be
* used to assemble this routine.

* Written by Lawrence A. Jones.

** Register usage.

*  ARO  Offset from data segment to data segment.
*  AR4  Interrupt data pointer.

** Address constants.

PAGE0   EQU   0          For access to on-chip registers.
CURPGE  EQU   4          Corresponds to >200 (on-chip).
TIM     EQU   2          On-chip time register.
PRD     EQU   3          Period reg for on-chip timer.
IMR     EQU   4          Interrupt mask register.

TMPORT  EQU   PA1        Access to Pacific timer.
ADPORT  EQU   PA2        A-to-D port (non-repeating).
ARPORT  EQU   PA3        Analog repeat port.
DAPORT  EQU   PA2        D-to-A port.
RSPORT  EQU   PA4        Resolver to digital port.

** Data constants.

TIMVAL  EQU   >FC19  External timer value for  5KHz sampling.
*TIMVAL  EQU   >FE0D  External timer value for 10KHz sampling.
```

```
*TIMVAL  EQU   >FF39  External timer value for 25KHz sampling.
*TIMVAL  EQU   >FF9D  External timer value for 50KHz sampling.
ICLR0    EQU   >FFC1  Interrupt mask to enable INT0 only.
ICLR1    EQU   >FFC2  Interrupt mask to enable INT1 only.
ICLR01   EQU   >FFC3  Interrupt mask to enable INT0 and INT1.

* Relative data page map.

         DORG  >0
I1CNT    DATA  $
STARTD   DATA  $
TEMP1    DATA  $

* Set INT0' and INT1' vectors.

         AORG  >2
I0VECT   B     I0INIT     See initialization code comments
I1VECT   B     I1INIT     for I0INIT and I1INIT.

         AORG  >400       Start of available prog RAM.
         DINT             Ensure that interrupts are disabled.

         B     INIVAR     Begin initialization.

*********************** MAIN PROGRAM ***********************

*---------------- Variable Initialization -----------------

INIVAR   EQU   $
         LDPK  CURPGE     Set page pointer to current page.
         LRLK  AR4,>3FF   (Starting location of the data - 1)
         SAR   AR4,STARTD
*         LRLK  AR0,1000   Each segment holds 1000 points.
         LRLK  AR0,2000   Each segment holds 2000 points.
         LACK  >0
         SACL  I1CNT      First INT1' yet to occur.
         RSXM             Unsigned arithmetic.
         RXF

*---------------- Interrupt Initialization ----------------
```

```
INIINT  EQU   $
        LALK  TIMVAL     Set external timer at PA1.
        SACL  TEMP1
        OUT   TEMP1,TMPORT

* Wait for the next INT1' as the new timer value isn't
* loaded until the timer times out.

        LALK  ICLRO1     Clear IMR for INTO' and INT1'.
        LDPK  PAGEO      Set page to on-chip registers.
        SACL  IMR        Unmask INTO and INT1.
        LDPK  CURPGE     Set page back to external RAM.
        EINT             Enable INTO's and INT1's.

IWAIT   LAC   I1CNT     I1CNT incremented in I1INIT routine.
        SBLK  >2        Wait until I1INIT has been executed
        BNZ   IWAIT       twice.

        LRLK  AR1,>258  Delay ~120us to let the remaining
        LARP  AR1          INTO's run out.
DELAY   BANZ  DELAY,*-

* Now reload the INT1' vector to point to I1SERV
* which is the main INT1' service routine.

        LALK  I1SERV     Load address of service routine.
        SACL  TEMP1      Store in data memory.
        LALK  I1VECT+1   Point to INT1' service vector.
        TBLW  TEMP1      Poke the new address.

* Now reload the INTO' vector to point to IOSERV
* which is the main INTO' service routine.

        LALK  IOSERV     Load address of service routine.
        SACL  TEMP1      Store in data memory.
        LALK  IOVECT+1   Point to INTO' service vector.
        TBLW  TEMP1      Poke the new address.

* The interrupt generating hardware and software is now
* initialized.  The next timeout of the sample clock will
* begin the proper functioning of the interrupt hardware
* and thus the algorithm must begin next.  Process begins
```

```
* on upcoming SCO!

*---------------------- Background Task ---------------------

WAIT    B     WAIT       Wait for interrupts to arrive.

DONE    EQU   $          Terminate.
        DINT
        NOP              NOPs are so that this location may
        NOP              be easily found during disassembly.
        NOP
        NOP
IDLE    B     IDLE

*--------------- INTERRUPT SERVICE ROUTINE 0 ---------------

IOSERV EQU    $
       IN     *,ARPORT   Get data from analog channel.
       LAC    *
       NEG               A/D data is complemented 2's comp.
       SACL   *0+
* The following code is used only when 2000 points from
* 6 channels are stored.  Four of the channels are stored in
* low RAM and the last two are stored in high RAM.  The
* following code bumps the pointer up to high RAM for the
* last two channels.
       SAR    AR4,TEMP1  Store interrupt data pointer.
       ZAC
       LAC    TEMP1      Load interrupt data pointer.
       SBLK   >2340      Pointer beyond low memory?
       BLZ    NOBUMP
       SBLK   >1000      Pointer already in high memory.
       BGEZ   NOBUMP     Fetch interrupt data pointer again.
       SAR    AR4,TEMP1
       ZAC
       LAC    TEMP1
       ADLK   >5CC0      Bump the pointer.
       SACL   TEMP1
       LAR    AR4,TEMP1
NOBUMP EINT
       RET
```

```
*--------------- INTERRUPT SERVICE ROUTINE 1 ---------------

I1SERV EQU   $

* We have < 17us to complete the service of this interrupt
* before the first INTO occurs.

MORE   LAC   STARTD      Fetch pointer.
       ADLK  >1
       SACL  STARTD
       SBLK  >BDO        Check if end of dat space- 2000 pts
       BZ    DONE
       LAR   AR4,STARTD  Reload pointer into AR4.
       LARP  AR4
       IN    *0+,RSPORT  Get shaft position from R/D.
       EINT              Next interrupt will be an INTO'.
       RET

* The external timer was continually latching interrupts
* into the IFR during initialization and therefore we will
* clean the first interrupt out, let the second INT1' occur
* and then begin on the second INT1'.

IOINIT EQU  $
       IN   TEMP1,ARPORT
       EINT
       RET

I1INIT EQU  $
       LAC   I1CNT
       ADLK  >1
       SACL  I1CNT      Indicate that I1INIT has occurred.
       EINT
       RET

*===============================================================

       END
```

Bibliography

[1] N. N. Hancock. *Matrix Analysis of Electrical Machinery.* Pergamon Press, New York, second edition, 1974.

[2] Charles V. Jones. *The Unified Theory of Electrical Machines.* Butterworth & Co. (Publishers) Ltd., London, 1967.

[3] Paul C. Krause. *Analysis of Electric Machinery.* McGraw-Hill Book Company, New York, 1986.

[4] W. Leonhard. *Control of Electrical Drives.* Springer-Verlag, New York, 1985.

[5] Roger W. Brockett. *Finite Dimensional Linear Systems.* John Wiley and Sons, Inc., New York, 1970.

[6] D. W. Novotny and J. H. Wouterse. Induction machine transfer functions and dynamic response by means of complex time variables. *IEEE Trans. on Power Apparatus and Systems*, PAS-95(4):1325–1335, July/August 1976.

[7] Herbert H. Woodson and James R. Melcher. *Electromechanical Dynamics, Part I: Discrete Systems.* John Wiley & Sons, Inc., New York, 1968.

[8] W. Leonhard. Microcomputer control of high dynamic performance ac-drives—a survey. *Automatica*, 22(1):1–19, 1986.

[9] A. Blondel and F. Carbenay. Méthode d'analyse expérimentale des propriétés des alternateurs. *Revue Générale de l'Électricite*, 5:811–819 and 843–855, June 1919.

[10] R. E. Doherty and C. A. Nickle. Synchronous machines: an extension of blondel's two reaction theory. *AIEE Transactions*, 45:912–942, June 1926.

[11] R. H. Park. Two-reaction theory of synchronous machines—generalized method of analysis—part I. *AIEE Trans.*, 48:716–727, July 1929.

[12] P. C. Krause and C. H. Thomas. Simulation of symmetrical induction machinery. *IEEE Trans. on Power Apparatus and Systems*, 84:1038–1053, November 1965.

[13] X. Z. Liu, G. C. Verghese, J. H. Lang, and M. K. Onder. Extending the Blondel-Park transformation to generalized electrical machines. In *Proc. 12th IMACS World Congress*, Paris, July 1988.

[14] J. R. Frus and B. C. Kuo. Closed-loop control of step motors without feedback encoders. In *Proc. of the 5th Ann. Symp. on Incremental Motion Control Systems and Devices*, pages CC–1–CC–11, Urbana, IL, 1976.

[15] B. C. Kuo. On current detection in variable-reluctance step motors. In *Proc. of the 6th Ann. Symp. on Incremental Motion Control Systems and Devices*, pages 205–220, Urbana, IL, 1977.

[16] W. C. Lin and B. C. Kuo. Waveform detection of permanent-magnet step motors, part I. In *Proc. of the 8th Ann. Symp. on Incremental Motion Control Systems and Devices*, pages 227–241, Urbana, IL, 1979.

[17] B. C. Kuo. Waveform detection of permanent-magnet step motors, part II. In *Proc. of the 8th Ann. Symp. on Incremental Motion Control Systems and Devices*, pages 243–256, Urbana, IL, 1979.

[18] A. Pittet and M. Jufer. Closed-loop without encoder of electromagnetic step motors. In *Proc. of the 7th Ann. Symp. on Incremental Motion Control Systems and Devices*, pages 37–44, Urbana, IL, 1978.

[19] V. D. Hair. Direct detection of back emf in permanent-magnet step motors. In *Proc. of the 12th Ann. Symp. on Incremental Motion Control Systems and Devices*, pages 211–219, Urbana, IL, 1983.

[20] Walter D. Harris. *Practical Indirect Position Sensing for a Variable Reluctance Motor.* Master's thesis, Massachusetts Institute of Technology, May 1987.

[21] H. Le-Huy, A. Jakubowicz, and R. Perret. A self-controlled synchronous motor drive using terminal voltage sensing. In *IAS Annual Meeting Conference Record*, pages 562–569, Cincinnati, OH, 1980.

[22] Jacques Davoine, Robert Perret, and Hoang Le-Huy. Operation of a self-controlled synchronous motor without a shaft position sensor. *IEEE Trans. on Industry Applications*, IA-19(2):217–222, 1983.

[23] T. Endo, F. Tajima, K. Iizuda, and H. Uzuhashi. Brushless motor without a shaft-mounted position sensor. *Trans. of IEEE of Japan*, 105-E(1/2):1–7, 1985.

[24] K. Hofer. Sensorless fourquadrant drives. In *18th Annual IEEE Power Electronics Specialists Conference*, pages 607–615, Blacksburg, VA, 1987.

[25] M. Stiebler and M. Kiewe. Closed-loop control of a step motor without using a rotor postion encoder. In *ICEM*, Lausanne, Switzerland, 1984.

[26] Hiroshi Watanabe, Takashi Isii, and Tomoo Fujii. Dc-brushless servo system without rotor position and speed sensor. In *Proceedings of IECON '87: Motor Control and Power Electronics*, pages 228–234, Cambridge, MA, November 1987.

[27] David G. Luenberger. Observing the state of a linear system. *IEEE Trans. on Military Electronics*, MIL-8:74–80, April 1964.

[28] R.E. Kalman. A new approach to linear filtering and prediction problems. *Trans. ASME (J. Basic Engineering)*, 82D(1):35–45, March 1960.

[29] David G. Luenberger. An introduction to observers. *IEEE Trans. on Aut. Control*, AC-16(6):596–602, December 1971.

[30] B. Gopinath. On the control of linear multiple input-output systems. *Bell Syst. Tech. J.*, March 1971.

[31] Huibert Kwakernaak and Raphael Sivan. *Linear Optimal Control Systems*. Wiley-Interscience, New York, 1972.

[32] Setsuzo Tsuji, Hitoshi Takata, Ryuzo Ueda, and Shigeo Takata. Second-order observer for nonlinear systems from discrete noiseless measurements. *IEEE Trans. on Aut. Control*, AC-22:105–112, February 1977.

[33] P. V. Kokotovic. Recent trends in feedback design: an overview. *Automatica*, 21(3):225–236, 1985.

[34] Sol W. Gully. Aerospace applications of nonlinear observer theory. In *Proc. of the 18th IEEE Conf. on Decision and Control*, pages 630–631, Fort Lauderdale, FL, 1979.

[35] S. P. Banks. A note on non-linear observers. *Int. J. Control*, 34(1):185–190, 1981.

[36] L. R. Hunt, Renjeng Su, and George Meyer. Global transformations of nonlinear systems. *IEEE Trans. on Aut. Control*, AC-28(1):24–31, January 1983.

[37] Arthur J. Krener and Alberto Isidori. Linearization by output injection and nonlinear observers. *Systems & Control Letters*, 3:47–52, 1983.

[38] C. Reboulet and C. Champetier. A new method for linearizing non-linear systems: the pseudolinearization. *Int. J. Control*, 40(4):631–638, 1984.

[39] Andrew P. Sage and James L. Melsa. *Estimation Theory with Applications to Communications and Control. McGraw-Hill Series in Systems Science*, McGraw-Hill Book Company, New York, 1971.

[40] Andrew H. Jazwinski. *Stochastic Processes and Filtering Theory.* Volume 64 of *Mathematics in Science and Engineering*, Academic Press, New York, 1970.

[41] Arthur Gelb, editor. *Applied Optimal Estimation.* The M.I.T. Press, Cambridge, MA, 1974.

[42] J. B. Pearson. On nonlinear least-square filtering. *Automatica*, 4:97–105, 1967.

[43] D. Bestle and M. Zeitz. Canonical form observer design for non-linear time-variable systems. *Int. J. Control*, 38(2):419–431, 1983.

[44] Jason L. Speyer and Taek L. Song. A comparison between the pseudomeasurement and extended kalman observers. In *Proc. of the 20th IEEE Conf. on Decision and Control*, pages 324–329, San Diego, CA, December 1981.

[45] Taek L. Song and Jason L. Speyer. A stochastic analysis of a modified gain extended kalman filter with applications to estimation with bearings only measurements. *IEEE Trans. on Aut. Control*, AC-30(10):940–949, October 1985.

[46] Taek L. Song and Jason L. Speyer. The modified gain extended kalman filter and parameter identification in linear systems. *Automatica*, 22(1):59–75, 1986.

[47] A. Halme, J. Selkäinaho, and J. Soininen. Adaptive control with nonlinear filtering. *Automatica*, 21(4):453–463, 1985.

[48] J. Selkäinaho, A. Halme, and F. Behbehani. An efficient nonlinear filter with application experiences on multivariable adaptive control and fault detection. In *IFAC Workshop on Adaptive Systems in Control and Signal Processing*, Pergamon Press, New York, 1983.

[49] R. Ueda, H. Takata, S. Nakagaki, and S. Takata. On the estimation of transient state of power system by discrete nonlinear observer. *IEEE Trans. on Power Apparatus and Systems*, PAS-94(6):2135–2140, November/December 1975.

[50] E. H. Okongwu, W. J. Wilson, and J. H. Anderson. Optimal state feedback control of a microalternator using an observer. *IEEE Trans. on Power Apparatus and Systems*, PAS-97(2):594–602, 1978.

[51] K. Yamashita and T. Taniguchi. On the estimation of the transient state of a synchronous machine by an optimal observer. *Int. J. Control*, 41(2):417–428, 1985.

[52] George C. Verghese and Seth R. Sanders. Observers for faster flux estimation in induction machines. In *16th Annual IEEE Power Electronics Specialists Conference*, Toulouse, France, June 1985.

[53] A. Lumsdaine, J. H. Lang, and M. J. Balas. A state observer for variable reluctance motors. In *Proc. of the 15th Ann. Symp. on Incremental Motion Control Systems and Devices*, Urbana, IL, 1986.

[54] R. E. Bellman, H. Kagiwada, R. Kalaba, and R. Sridhar. *Invariant Embedding and Nonlinear Filter Theory.* RAND Research Memorandum RM-4374-PR, RAND Corp., December 1964.

[55] Andrew P. Sage. *Optimum Systems Control.* Prentice-Hall, Inc., Englewood Cliffs, NJ 07632, first edition, 1968.

[56] C. K. Gharban and B. J. Cory. Non-Linear dynamic power system state estimation. *IEEE Trans. on Power Systems*, PWRS-1(3):276–283, August 1986.

[57] Donald E. Kirk. *Optimal Control Theory: An Introduction.* Prentice-Hall, Inc., Englewood Cliffs, NJ 07632, 1970.

[58] George C. Verghese, Jeffrey H. Lang, and Leo F. Casey. Analysis of instability in electrical machines. *IEEE Trans. on Industry Applications*, IA-22(5):852–864, September/October 1986.

[59] E. Fehlberg. *Low-Order Classical Runge-Kutta Formulas with Stepsize Control.* Technical Report TR R-315, NASA. (also in Computing, 6 (1970) pp. 61–71).

[60] L. F. Shampine and H. A. Watts. *Practical Solution of Ordinary Differential Equations by Runge-Kutta Methods.* Technical Report SAND76-0585, Sandia National Laboratories Report, 1976.

[61] L. F. Shampine, H. A. Watts, and S. Davenport. Solving non-stiff ordinary differential equations— the state of the art. *SIAM Review*, 18:376–471, 1976.

[62] George E. Forsythe, Michael A. Malcolm, and Cleve B. Moler. *Computer Methods for Mathematical Computations.* Prentice-Hall, Inc., Englewood Cliffs, NJ 07632, 1977.

[63] *Linear Databook.* National Semiconductor Corporation, 2900 Semiconductor Drive, Santa Clara, CA 95051, 1982.

[64] A. E. Fitzgerald, Charles Kingsley, Jr., and Stephen D. Umans. *Electric Machinery.* McGraw-Hill Book Company, New York, fourth edition, 1983.

[65] Leander W. Matsch. *Electromagnetic and Electromechanical Machines.* Harper & Row, Publishers, Inc., New York, second edition, 1977.