

The Islamic University of Gaza
Deanery of Graduate Studies
Faculty of Engineering
Electrical Engineering Department



PARAMETERS IDENTIFICATION OF A PERMANENT MAGNET DC MOTOR

By

Mohammed S. Z. Salah

Supervisor

Prof. Dr. Muhammed Abdelati

“A Thesis Submitted in Partial Fulfillment of the Requirements for the
Degree of Master of Science in Electrical Engineering.”

1430 - 2009

Abstract

In this thesis, driver circuits to drive DC motor are designed and built. We take in our design the safety for any damage, the isolation from any interference and the capability to provide the sufficient current needed to drive the motor. Also, methods and algorithms used for parameters identification are addressed. Data acquisition module from National Instruments is a suitable module used for acquiring signals since it's compatible with MATLAB and LABVIEW. To identify the parameters of the motor, an experimental measurement of armature voltage, armature current and rotor speed are performed using the NIDAQ USB-6008 data acquisition module. Trials are performed to apply some of the methods on the motor practically to identify the motor parameters. DAQ toolbox in MATLAB/Simulink is used to acquire the test signals and perform analysis based on the nonlinear least square method or pattern search method which they are suitable. Finally, we designed a GUI to be user friendly and to automate all the process of identification, in addition to the feature for undergraduate and graduate students to apply experiments on control and machine labs. The extracted motor parameters from simulation produced consistent simulation results with experimental data. The proposed approaches are compared and both are proved to be simple, fast and accurate.

عنوان البحث بالعربية :

” استخلاص المعاملات الخاصة بالمحرك ذو التيار المستمر “

في هذا البحث تم تصميم وبناء دوائر التشغيل للمحرك ذو التيار المباشر المستخدم في الدراسة، حيث تم الأخذ بعين الاعتبار في التصميم وسائل الأمان لأي طارئ يحدث أو تداخل مغناطيسي وكذلك القدرة على تزويد التيار اللازم لتشغيل المحرك ، كما وتم دراسة الطرق المستخدمة في استخلاص المعاملات الخاصة بهذا النوع من المحركات، تم استخدام نموذج ملتقط العينات من National Instruments لمناسبتها للإستخدام لتوافقه مع MATLAB وكذلك LABVIEW ، ولاستخلاص هذه المعاملات تم بناء نظام تطبيقي لأخذ العينات الخاصة بجهد والتيار وسرعة المحرك باستخدام ملتقط العينات (NIDAQ USB-6008). كما وتم تحليل هذه العينات باستخدام برنامج MATLAB الذي يقوم بالاتصال وعمل المحاكاة اللازمة مع ملتقط العينات . تم إجراء عدة محاولات لتطبيق بعض الطرق التي تم دراستها . وفي النهاية تم تطبيق طريقة Nonlinear Least Square وكذلك طريقة Pattern Search في عملية استخلاص المعاملات لأنهما الطريقتان المناسبتان في إيجادها والمتوافقة مع MATLAB/Simulink ، تم أتمتة جميع هذه العمليات بتصميم شاشة محاكاة باستخدام GUI الموجود في MATLAB حتى يسهل عملية استخلاص المعاملات وإظهار الرسومات التحليلية المطلوبة ، مع إضافة عدة مزايا تمكن الطلاب والخريجين من تطبيق بعض التجارب في مجال التحكم والآلات . المعاملات التي تم استخلاصها أعطت نتائج مقبولة وتبين ذلك في تطابق الإستجابة بين النظام الحقيقي والنظام الافتراضي. الطريقتان المقترحتان تتميزان بالبساطة والسرعة وكذلك الدقة حيث تم عمل مقارنة بينهما.

Dedication

*To my family members who have been a constant source of
motivation, inspiration, and support.*

Acknowledgements

I would like to thank my thesis supervisor, Prof. Dr. Muhammed Abdelati, for his support, encouragement and professional assistance.

Special thanks to all other Islamic University staff members that I may have called upon for assistance especially Dr. Iyad Abu-Hudrouss and Dr. Mohammed Alhan-jouri, as their suggestions have helped with the development of this thesis.

I would also like to extend my gratitude to my family for the support they have given me. Also my thanks to my friends for the precious opportunity they gave me to study in.

Contents

Abstract	ii
Dedication	iv
Acknowledgements	v
List of Tables	viii
List of Figures	ix
List of Nomenclatures	xi
List of Abbreviations	xii
1 Introduction and Literature Review	1
1.1 Research Motivation and Goal	1
1.2 Literature Review	1
1.3 Methodology	5
1.4 The Identification Process	6
1.4.1 Collecting Information about the System	7
1.4.2 Selecting a Model Structure to Represent the System	7
1.4.3 Matching the Selected Model Structure to the Measurements	9
1.4.4 Validating the Selected Model	9
1.5 Thesis Structure	9
2 DC Motors and Parameters Identification	10
2.1 History and Background	10
2.2 Categorization of Electric Motors	11
2.3 DC Motors	11
2.3.1 Brushed DC motors	12
2.4 Principle of Operation	13
2.4.1 Production of Torque	13
2.4.2 Generated E.M.F.	16
2.4.3 Motor Modeling and Simulation	17
2.5 Parameters Identification	19
2.5.1 Pseudo Inverse Technique	19

2.5.2	Nonlinear Least-Square Method	22
2.5.3	Pattern Search Algorithm	23
3	System Hardware Development	26
3.1	Introduction	26
3.2	Hardware Design	27
3.3	The MOSFET as a Switch	28
3.3.1	Introduction	28
3.3.2	Choosing a MOSFET	29
3.3.3	MOSFET Gate Driver	31
3.4	Pulse Width Modulation (PWM)	32
3.5	Power Supply	34
3.6	Voltage Doubler	35
3.7	Sensors and Feedback Elements	35
3.7.1	Introduction	35
3.7.2	Current Sensors	36
3.7.3	Shunt Resistors as a Current Sensor	37
3.7.4	Speed/Position Sensor	38
3.7.5	Voltage Sensing	39
4	Experimentation, Testing and Results	40
4.1	Experimental System	40
4.2	DC Motor Speed	41
4.3	Panel Meters	42
4.4	Sensing Circuit	43
4.5	Interfacing Circuit	44
4.6	Data Acquisition	45
4.7	Parameters Identification	46
4.7.1	GUI Interface	55
5	Conclusions and Future Work	60
5.1	Conclusions	60
5.2	Future Work	60
5.3	Summary of Contributions	61
	Bibliography	63
A	MALTLAB Codes	66
A.1	Acquiring Signals	66
A.2	Parameters Identification	71
A.3	Validation of Identification	73
B	Schematics	75

List of Tables

2.1	DC motor parameters.	20
3.1	Specification of experimental dc motor.	26
3.2	MOSFET Drivers.	31
3.3	Current sensing methods.	37
4.1	Interfacing circuit DB9 pinout	44
4.2	NI DAQ inputs and outputs.	46
4.3	Summary of parameter estimation process(NLS method)	57
4.4	Summary of parameter estimation results(NLS method)	57
4.5	Summary of parameter estimation process(Pattern method)	57
4.6	Summary of parameter estimation results(Pattern method)	58
5.1	Parameter estimation results	62

List of Figures

2.1	The electric motor invented by Michael Faraday.	12
2.2	Brushed DC motor.	13
2.3	Production of torque.	14
2.4	Production of magnetic field by permanent magnet.	14
2.5	Coil carrying current.	15
2.6	Split copper ring.	16
2.7	Cross section of dc motor	16
2.8	Complete equivalent model for dc motor	17
2.9	The Block diagram of dc motor	18
2.10	Experimental setup requirement	21
3.1	DC motor used in our experiment.	27
3.2	Architecture of the system	28
3.3	A Practical PWM Circuit.	33
3.4	Field driver circuit.	33
3.5	Armature driver circuit.	34
3.6	Power Box.	34
3.7	Voltage Doubler circuit.	35
3.8	Typical DC Motor Block Diagram.	36
3.9	Quadrature Encoder.	38
3.10	Voltage Divider Circuit.	39
4.1	General view of the experimental system	40
4.2	Samples of acquiring pulses.	42
4.3	Pulse Edges.	43
4.4	Digital panel meters.	43
4.5	Sensing circuit of the system.	44
4.6	PCB of interfacing circuit	45
4.7	The National Instruments USB-6008	45
4.8	Block diagram of DC motor	48
4.9	Armature voltage	49
4.10	Armature current response	50
4.11	Selecting Parameters for Estimation	50
4.12	Optimization options	51
4.13	Estimation process tab	52
4.14	Current response for the actual and estimated models	53
4.15	Speed response for the actual and estimated models	54

4.16	Trajectories of estimated parameters	55
4.17	Parameters estimation results	56
4.18	Current response for the actual and estimated models	58
4.19	Speed response for the actual and estimated models	58
4.20	GUI interface	59
B.1	Field driver unit schematics.	76
B.2	Field driver unit layout.	77
B.3	Armature driver unit.	78
B.4	Armature driver unit layout.	79
B.5	Armature power circuit.	80
B.6	Field power circuit.	81
B.7	interfacing circuit.	82

List of Nomenclatures

I	Current.
B	Magnetic field flux density.
F	Force.
l	Length of wire.
T	Torque.
N	Number of turns.
T_m	Motor's torque.
K_t	Motor's torque constant.
e	Electromotive force.
λ	Flux linking the coil.
θ_m	Angle between the pole-faces.
ϕ	Magnetic flux linking the coil.
ω_m	Constant angular speed.
K_e	Back-EMF constant.
K_t	Torque constant.
R_a	Terminal resistance.
L_a	Armature inductance.
J	Moment of inertia of the rotor.
B	Viscous (damping) friction.
s	Laplace domain.
z	Z-transform domain.
W	Rotor speed.
V	Armature voltage.
R_{DS}	Drain to source MOSFET resistance.

List of Abbreviations

AC	Alternative Current.
AI	Analog Input.
AO	Analog Output.
CPR	Counts Per Revolution.
DAQ	Data Acquisition.
DAT	Data Acquisition Toolbox.
DC	Direct Current.
DSP	Digital Signal Processing.
EMF	Electromotive Force.
EMI	Electromagnetic Interference.
FET	Field Effect Transistor.
FPGA	Field Programmable Gate Array.
GUI	Graphical User Interface.
GPS	Generalized Pattern Search Algorithm.
IC	Integrated Circuit.
IOS	Input Current Offset.
LED	Light Emitting Diode.
MADS	Mesh Adaptive Search Algorithm.
MCU	Microcontroller Unit.
MOSFET	Metal Oxide Semiconductor Field Effect Transistor.
NI	National Instruments.
PCB	Printed Circuit Board.
PC	Personal Computer.
PD	Proportional Derivative.
PIC	Programmable Interface Controller.
PWM	Pulse Width Modulation.
USB	Universal Serial Bus.
VOS	Voltage Offset.

Chapter 1

Introduction and Literature Review

1.1 Research Motivation and Goal

In practical life and through building projects, we depend widely on using DC motors as the main component for producing motion and torque. These motors are taken from electronic devices such as printers, imaging machines, scanners and other old devices. The problem we face is that these motors does not have any industrial information. These information are the parameters which describe the mathematical model of motor to use it in simulation softwares such as MATLAB and LABVIEW. The mathematical model help predicting system behavior and designing system controller. Because we haven't the parameters of motor, we face a problem of implementing it or to precise control it. In our thesis we build the drivers and use a data acquisition module and MATLAB to estimate these parameters using nonlinear least square method. This method is an accurate method for estimation parameters by iterative procedure.

The goals for parameters identification are implementing an accurate mathematical models, design precise controllers, predict the closed loop behavior of the plant, researches validate manufacture supplied parameters and specify missing information.

1.2 Literature Review

We studied the modeling of dc motors for control applications [1], and we found that there are three different mathematical models of an armature controlled dc motors. These models are:

1. Precise nonlinear model.
2. Piecewise linear model.

3. Second-order linear model.

A mathematical model for a physical device must often reflect a compromise. It must not attempt to mirror the real device in such great detail that the model becomes cumbersome; on the other hand it should not be so simplified that predictions and explanations based on it are either trivial or far from reality.

In our thesis we used the second order linear model over other models due to its simplicity. The main difficulty with the nonlinear models is the requirement of numerical solution and the use of this model in those applications of adaptive and optimal control which require a digital computer. The second-order linear model assumes the following for ease of use:

1. The static friction is negligible and the frictional torque can be considered directly proportional to angular velocity.
2. The brush voltage drop is negligible.
3. Armature reaction can be neglected.
4. The resistance and the inductance of the armature can be regarded as constant.

We noticed from modeling that there is a variation of the inductance of the armature with armature current, so conventional methods for dc motor parameters identification is not accurate and lead to poor controlling.

Accurate mathematical models and their parameters are essential when designing controllers because they allow the designer to predict the closed loop behavior of the plant. Errors in parameter values can lead to poor control and instability. The conventional way of characterizing a dc motor is to perform a separate test for each parameter, but this is not only time consuming, but can yield misleading results if the parameters are measured under static or no load conditions. Therefore, estimation techniques must be used to estimate the unknown or inaccurate parameters values with precision. Estimation approaches can be divided into two categories: offline estimation and online estimation.

Offline techniques use specific test inputs, measure the corresponding output signals and then try to establish the relation between them. Online techniques use, for example, observers and Kalman filters to recursively estimate parameters. Neural networks can be used in either offline or online approaches.

There are many techniques and ordered from old to new, each technique has its own method for parameters identification. Each technique has advantages over the other and some have some disadvantages. The following techniques are listed below:

In 1973, W. Lord and J. H. Hwang identified eight practical servomotor models for separately excited dc servomotors, and the parameters of each model are clearly

defined. Testing procedures are given for determining a servomotor's model type and all the model parameters based only on the current response of the machine to a step input of armature voltage. Excellent agreement is obtained between the actual current responses of several servomotors and their predicted responses obtained from the corresponding mathematical models, indicating the efficacy of the technique [2].

In 1975, W. Lord and J. H. Hwang showed that linear modeling techniques are applicable to separately excited dc motors if the model parameters are obtained under dynamic operating conditions. They used Pasek's technique in order to identify the model type and all the model parameters from single current response of the motor to a step input of armature voltage [3].

In 1983, R. Schulz introduced a frequency response technique for measuring the parameters of high-performance dc motors. A second-order motor model, under certain conditions, is shown to be equivalent to a series resonant electrical circuit. Frequency response measurements of the motor, when treated as electrical impedance, form the basis of a measurement technique which has certain practical advantages. Results are compared to measurements made using conventional methods [4].

In 1988, A. D. Rajkumar and R. Somanatham proposed practical determination of direct current (dc) motor transients is generally possible only by using sophisticated transient recorders or storage oscilloscopes. In this paper a method is presented to determine dc motor transients using an 8085 based microcomputer. The scheme was tested in the laboratory and results verified [5].

In 1991, S. Weerasooriya and M. A. El-Sharkawi introduced an artificial neural network based high performance speed control system for a dc motor. The purpose is to achieve accurate trajectory control of the speed, specially when motor and load parameters are unknown. The unknown nonlinear dynamics of the motor and the load are captured by an artificial neural network. Performance of the identification and control algorithms are evaluated by simulating them on a typical dc motor model [6].

In 1992, Y. Jung, K. Cho, Y. Lim, J. Park and Y. Chang described efforts to develop a microcomputer-based parameter identification system for a brushless DC motor (BDCM). A BDCM is equal to a DC motor in the equivalent circuit. Therefore, Pasek's equation for the parameter determination of a DC motor is applicable to a BDCM. A new identification algorithm for BDCM parameters using Pasek's technique is developed. The algorithm is implemented on an IBM-PC/AT using the C language [7].

In 1996, D. M. Gillard and K. E. Bollinger described an investigation into the use of a multilayered neural network for measuring the transfer function of a power system for use in power system stabilizer (PSS) tuning and assessing PSS damping. The objectives are to quickly and accurately measure the transfer function relating the electric power output to the AVR PSS reference voltage input of a system with

the plant operating under normal conditions. In addition, the excitation signal used in the identification procedure is such that it will not adversely affect the terminal voltage or the system frequency [8].

In 2000 A. Rubbai and R. Kotaru proposed online identification and control of dc motor using learning adaptation. Use is made of the power of feedforward artificial neural networks to capture and emulate detailed nonlinear mappings, in order to implement a full nonlinear control law [9].

In 2000 X. Liu, H. Zhang, J. Liu and J. Yang described Parameter estimation based on block-pulse function series to estimate the continuous-time model of the motor. The electromechanical parameters of the motor can be obtained from the estimated model parameters [10].

In 2001, S. Saab and R. Abi Kaed-Bey showed that the parameters of a dc motor can be estimated experimentally by employing discrete measurements of an integrated dynamometer. The dynamometer outputs are the discrete measurements of the armature current, angular velocity, armature voltage (system input), and the torque developed by the motor. They employed least-squares algorithm to implement the parameter identification of dc motor without the use of a D/A converter and a power amplifier. A Kalman filter is also implemented, as a state observer, to estimate the angular acceleration and the derivative of the armature current. In addition, to improve the overall identification performance, the DC parameters were first estimated by decoupling the AC parameters using a DC input signal [11].

In 2004, A. Dupuis, M. Ghribi and A. Kaddouri simplified the offline identification of motor parameters by proposing a new method based on optimization using a multiobjective elitist genetic algorithm. The non-dominated sorting genetic algorithm (NSGA-II) is used to minimize the error between the current and velocity responses of data and an estimated model. The robustness of the method is shown by estimating parameters of a DC motor in four different cases. Simulation results show that the method successfully estimates the motor parameters and is also capable of identifying a load torque simultaneously [12].

In 2005, R. Krneta, S. Antic, and D. Stojanovic investigated the issues involved in applying recursive least squares method in parameters identification of DC motor models. The validity of the proposed method was shown by simulation and experiments. By comparing a graphic of real motor speed and a graphics of speed of investigated models in a Z and S domains it can be concluded that a satisfying quality of DC motor parameters identification has been achieved [13].

In 2006, R. Garrido and R. Miranda proposed a new method for closed loop identification of position controlled dc servomechanisms. The loop around the servo is closed using a Proportional Derivative (PD) controller. A model of the servo is simultaneously controlled using a second PD controller. The error and its time derivative between the output of both, the real servo and its model, is employed for identifying the motor parameters which in turn are employed for updating the

model. Properties of the identification scheme are studied using Lyapunov stability theory [14].

In 2007, W. Aung described the analysis on modeling and simulink of DC motor and its driving system, hardware and software. For DC Motor Modeling, it can be analyzed with control techniques of Step response, Impulse response and Bode plot by using MATLAB Simulink. All data based on the internal circuit of a simple DC Motor and its features can be analyzed both by Control System design calculation and by MATLAB software [15].

In 2009, M. Hadeef and M. Rachid described a parameter identification method using inverse problem methodology is proposed. The minimization of the objective function with respect to the desired vector of design parameters is the most important procedure in solving the inverse problem. The conjugate gradient method is used to determine the unknown parameters, and Tikhonovs regularization method is then used to replace the original ill-posed problem with a well-posed problem [16].

1.3 Methodology

In this thesis, we made a study on different identification methods were used in order to get accurate parameters of dc motor compatible to our experimental model.

The convential way of characterizing a dc motor is to perform a separate test for each parameter, but this is not only time consuming, but can yield misleading results if the parameters are measured under static or no load conditions. So looking into another methods is needed to get accurate parameters need for good controlling. In our thesis there are various methods are studied which are Pasek's, frequency response, pseudo inverse and nonlinear least square methods.

Pasek's method is one of the earliest techniques used in parameters identification of dc motors. It determines a high-performance dc motor's mode model type and all the model parameters based only on the current response of the machine to a step input of armature voltage along with the steady state speed. But this method can introduce some instrumentation problems. The technique requires an accurate reading of two points of transient waveform, which can be difficult to do in presence of noise. Also this method measures a few points on the current time response curve making it very sensitive to current commutation noise, which renders the method inaccurate for low cost dc motors widely used in our industry.

An alternative method to Pasek's method is the frequency response method for determining the parameters of high performance dc motors. This method has the following features:

1. All significant motor parameters are determined at once in a dynamic and loaded condition.

2. No nonelectrical measurements are required (such as speed in Pasek's method).
3. Results are averaged out to minimize errors caused by noise.
4. Sophisticated instrumentation is not necessary.
5. The approach is readily adaptable to automatic testing.

Frequency response method determines the parameters of high performance dc motors by treating a second order motor model as an electrical impedance (RLC circuit), and by tuning the values of RLC circuit elements to match the response of dc motor and by some relations the parameters of dc motor are calculated. This method uses an ac signal with specific frequency about 1kHz. Unfortunately, this method is not suitable to the power drivers used in our experimental platform. Sensitivity to noise is another reason that discouraged us to utilize this method.

The pseudo inverse technique is a method capable of identifying an equivalent discrete transfer function of simple systems like motor, thermal system, mass damper systems etc with no zeros (only poles). This method assumes a single input single output system and unable to find the dc motor parameters and doesn't fit our objective [16].

Nonlinear least square method is a general approach in identification seeks to define an objective function that would reach its minimum. The physical system to be investigated is described in terms of parameters, and then, the objective function is minimized with respect to the parameters by an iterative procedure. At the minimum of the objective function, the values of the parameters describe the real structure of the physical system [17].

Pattern search algorithm is another method used in this thesis and applied practically on our experimental system. Pattern search is an attractive alternative to the genetic algorithm, as it is often computationally less expensive and can minimize the same types of functions. Finally, this method can be used to find the motor parameters with less error and compare it with previous methods.

1.4 The Identification Process

Each identification session consists of a series of basic steps. Some of them may be hidden or selected without the user being aware of his choice [18]. Clearly, this can result in poor or suboptimal results. In each session the following actions should be taken:

- Collecting information about the system.
- Selecting a model structure to represent the system.

- Choosing the model parameters to fit the model as well as possible to the measurements: selection of a "goodness of fit" criterion.
- Validating the selected model.

Each of these points is discussed in more detail next.

1.4.1 Collecting Information about the System

If we want to build a model for a system, we should get information about it. This can be done by just watching the natural fluctuations, but most often it is more efficient to set up dedicated experiments that actively excite the system. In the latter case, the user has to select an excitation that optimizes his own goal (for example, minimum cost, minimum time, or minimum power consumption for a given measurement accuracy) within the operator constraints. The quality of the final result can depend heavily on the choices that are made.

1.4.2 Selecting a Model Structure to Represent the System

A choice should be made within all the possible mathematical models that can be used to represent the system. Again a wide variety of possibilities exist, such as:

- Parametric versus nonparametric models:

In a parametric model, the system is described using a limited number of characteristic quantities called the parameters of the model, whereas in a nonparametric model the system is characterized by measurements of a system function at a large number of points. Examples of parametric models are the transfer function of a filter described by its poles and zeros and the motion equations of a piston. An example of a nonparametric model is the description of a filter by its impulse response at a large number of points.

Usually it is simpler to create a nonparametric model than a parametric one because the modeler needs less knowledge about the system itself in the former case. However, physical insight and concentration of information are more substantial for parametric models than for nonparametric ones.

- White box models versus black box models:

In the construction of a model, physical laws whose availability and applicability depend on the insight and skills of the experimenter can be used (Kirchhoff's laws, Newton's laws, etc.). Specialized knowledge related to different scientific fields may be brought into this phase of the identification process. The modeling of a loudspeaker, for example, requires extensive understanding of mechanical, electrical, and acoustical phenomena. The result may be a physical model, based on comprehensive knowledge of the internal functioning of the system. Such a model is called a white box model. Another approach is to extract a black box model from the data. Instead

of making a detailed study and developing a model based upon physical insight and knowledge, a mathematical model is proposed that allows sufficient description of any observed input and output measurements. This reduces the modeling effort significantly. For example, instead of modeling the loud-speaker using physical laws, an input-output relation, taking the form of a high-order transfer function, could be proposed. The choice between the different methods depends on the aim of the study: the white box approach is better for gaining insight into the working principles of a system, but a black box model may be sufficient if the model will be used only for prediction of the output. Although, as a rule of thumb, it is advisable to include as much prior knowledge as possible during the modeling process, it is not always easy to do it. If we know, for example, that a system is stable, it is not simple to express this information if the polynomial coefficients are used as parameters.

- Linear models versus nonlinear models:

In real life, almost every system is nonlinear. Because the theory of nonlinear systems is very involved, these are mostly approximated by linear models, assuming that in the operation region the behavior can be linearized. This kind of approximation makes it possible to use simple models without jeopardizing properties that are of importance to the modeler. This choice depends strongly on the intended use of the model. For example, a nonlinear model is needed to describe the distortion of an amplifier, but a linear model will be sufficient to represent its transfer characteristics if the linear behavior is dominant and is the only interest.

- Linear-in-the-parameters versus nonlinear-in-the-parameters:

A model is called linear-in-the-parameters if there exists a linear relation between these parameters and the error that is minimized. This does not imply that the system itself is linear. For example, $\epsilon = y - (a_1 u + a_2 u^2)$ is linear in the parameters a_1 and a_2 but describes a nonlinear system. On the other hand,

$$\epsilon(j\omega) = Y(j\omega) - \frac{a_0 + a_1 j\omega}{b_0 + b_1 j\omega} U(j\omega) \quad (1.1)$$

describes a linear system but the model is nonlinear in the b_0 and b_1 parameters. Linearity in the parameters is a very important aspect of models because it has a strong impact on the complexity of the estimators if a (weighted) least squares cost function is used. In that case, the problem can be solved analytically for models that are linear in the parameters so that an iterative optimization problem is avoided.

1.4.3 Matching the Selected Model Structure to the Measurements

Once a model structure is chosen (e.g., a parametric transfer function model), it should be matched as well as possible with the available information about the system. Mostly, this is done by minimizing a criterion that measures a goodness of the fit. The choice of this criterion is extremely important because it determines the stochastic properties of the final estimator. Many choices are possible and each of them can lead to a different estimator with its own properties. Usually, the cost function defines a distance between the experimental data and the model. The cost function can be chosen on an ad hoc basis using intuitive insight, but there also exists a more systematic approach based on stochastic arguments.

1.4.4 Validating the Selected Model

Finally, the validity of the selected model should be tested: does this model describe the available data properly or are there still indications that some of the data are not well modeled, indicating remaining model errors? In practice, the best model (= the smallest errors) is not always preferred. Often a simpler model that describes the system within user-specified error bounds is preferred. Tools will be provided that guide the user through this process by separating the remaining errors into different classes, for example, unmodeled linear dynamics and nonlinear distortions. From this information, further improvements of the model can be proposed, if necessary. During the validation tests it is always important to keep the application in mind. The model should be tested under the same conditions as it will be used later. Extrapolation should be avoided as much as possible. The application also determines what properties are critical.

Good understanding of the intended applications helps to set up good experiments, and is very important to make the proper simplifications during the model-building process.

1.5 Thesis Structure

There are five chapters in this thesis. Chapter 1 provides introduction and Literature review. Chapter 2 concerns about dc motor principles and modeling. Chapter 3 presents system hardware development. Chapter 4 provide experimental system, tests and results. Finally, in the last chapter conclusions and future work are given.

Chapter 2

DC Motors and Parameters Identification

2.1 History and Background

An electric motor is a device using electrical energy to produce mechanical energy, nearly always by the interaction of magnetic fields and current-carrying conductors. The reverse process, that of using mechanical energy to produce electrical energy, is accomplished by a generator or dynamo. Traction motors used on vehicles often perform both tasks [19].

Electric motors are found in myriad uses such as industrial fans, blowers and pumps, machine tools, household appliances, power tools, and computer disk drives, among many other applications. Electric motors may be operated by direct current from a battery in a portable device or motor vehicle, or from alternating current from a central electrical distribution grid. The smallest motors may be found in electric wristwatches. Medium-size motors of highly standardized dimensions and characteristics provide convenient mechanical power for industrial uses. The very largest electric motors are used for propulsion of large ships, and for such purposes as pipeline compressors, with ratings in the thousands of kilowatts. Electric motors may be classified by the source of electric power, by their internal construction, and by application.

The physical principle of production of mechanical force by the interaction of an electric current and a magnetic field was known as early as 1821. Electric motors of increasing efficiency were constructed throughout the 19th century, but commercial exploitation of electric motors on a large scale required efficient electrical generators and electrical distribution networks.

2.2 Categorization of Electric Motors

The classic division of electric motors has been that of Alternating Current (AC) types vs Direct Current (DC) types. This is more a de facto convention, rather than a rigid distinction. For example, many classic DC motors run on AC power, these motors being referred to as universal motors.

Rated output power is also used to categorize motors, those of less than 746 Watts, for example, are often referred to as fractional horsepower motors (FHP) in reference to the old imperial measurement.

The ongoing trend toward electronic control further muddles the distinction, as modern drivers have moved the commutator out of the motor shell. For this new breed of motor, driver circuits are relied upon to generate sinusoidal AC drive currents, or some approximation of. The two best examples are: the brushless DC motor and the stepping motor, both being poly-phase AC motors requiring external electronic control, although historically, stepping motors (such as for maritime and naval gyrocompass repeaters) were driven from DC switched by contacts.

Considering all rotating (or linear) electric motors require synchronism between a moving magnetic field and a moving current sheet for average torque production, there is a clearer distinction between an asynchronous motor and synchronous types. An asynchronous motor requires slip between the moving magnetic field and a winding set to induce current in the winding set by mutual inductance; the most ubiquitous example being the common AC induction motor which must slip in order to generate torque. In the synchronous types, induction (or slip) is not a requisite for magnetic field or current production (eg. permanent magnet motors, synchronous brush-less wound-rotor doubly-fed electric machine).

2.3 DC Motors

A DC motor is designed to run on DC electric power. Two examples of pure DC designs are Michael Faraday's homopolar motor (which is uncommon), and the ball bearing motor.

A homopolar motor has a magnetic field along the axis of rotation and an electric current that at some point is not parallel to the magnetic field. The name homopolar refers to the absence of polarity change. Homopolar motors necessarily have a single-turn coil, which limits them to very low voltages. This has restricted the practical application of this type of motor. Figure of this motor is shown in Figure 2.1.

A ball bearing motor is an unusual electric motor that consists of two ball-bearing-type bearings, with the inner races mounted on a common conductive shaft, and the outer races connected to a high current, low voltage power supply. An alternative construction fits the outer races inside a metal tube, while the inner



Figure 2.1: The electric motor invented by Michael Faraday.

rices are mounted on a shaft with a non-conductive section (e.g. two sleeves on an insulating rod). This method has the advantage that the tube will act as a flywheel. The direction of rotation is determined by the initial spin which is usually required to get it going.

By far the most common DC motor types are the brushed and brushless types, which use internal and external commutation respectively to create an oscillating AC current from the DC source so they are not purely DC machines in a strict sense.

2.3.1 Brushed DC motors

The classic DC motor design generates an oscillating current in a wound rotor, or armature, with a split ring commutator, and either a wound or permanent magnet stator. A rotor consists of one or more coils of wire wound around a core on a shaft; an electrical power source is connected to the rotor coil through the commutator and its brushes, causing current to flow in it, producing electromagnetism. The commutator causes the current in the coils to be switched as the rotor turns, keeping the magnetic poles of the rotor from ever fully aligning with the magnetic poles of the stator field, so that the rotor never stops (like a compass needle does) but rather keeps rotating indefinitely (as long as power is applied and is sufficient for the motor to overcome the shaft torque load and internal losses due to friction, etc.). Figure of this motor is shown in Figure 2.2.

Many of the limitations of the classic commutator DC motor are due to the need for brushes to press against the commutator. This creates friction. At higher speeds, brushes have increasing difficulty in maintaining contact. Brushes may bounce off the irregularities in the commutator surface, creating sparks. (Sparks are also created inevitably by the brushes making and breaking circuits through the rotor coils as the brushes cross the insulating gaps between commutator sections. Depending on the commutator design, this may include the brushes shorting together adjacent sections and hence coil ends momentarily while crossing the gaps.

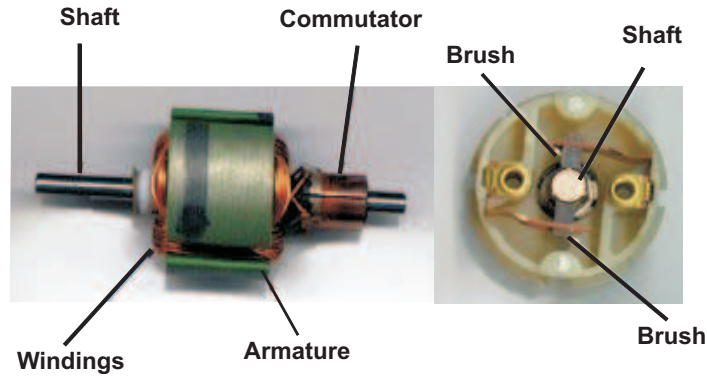


Figure 2.2: Brushed DC motor.

Furthermore, the inductance of the rotor coils causes the voltage across each to rise when its circuit is opened, increasing the sparking of the brushes.) This sparking limits the maximum speed of the machine, as too-rapid sparking will overheat, erode, or even melt the commutator. The current density per unit area of the brushes, in combination with their resistivity, limits the output of the motor. The making and breaking of electric contact also causes electrical noise, and the sparks additionally cause RFI. Brushes eventually wear out and require replacement, and the commutator itself is subject to wear and maintenance (on larger motors) or replacement (on small motors). The commutator assembly on a large machine is a costly element, requiring precision assembly of many parts. On small motors, the commutator is usually permanently integrated into the rotor, so replacing it usually requires replacing the whole rotor.

Large brushes are desired for a larger brush contact area to maximize motor output, but small brushes are desired for low mass to maximize the speed at which the motor can run without the brushes excessively bouncing and sparking (comparable to the problem of "valve float" in internal combustion engines). (Small brushes are also desirable for lower cost.) Stiffer brush springs can also be used to make brushes of a given mass work at a higher speed, but at the cost of greater friction losses (lower efficiency) and accelerated brush and commutator wear. Therefore, DC motor brush design entails a trade-off between output power, speed, and efficiency/wear. Types of DC motors are permanent magnet, separately, shunt, series and compound.

2.4 Principle of Operation

2.4.1 Production of Torque

Consider a wire carrying a current I suspended in a magnetic field of uniform flux density B , as shown in Figure 2.3, where the direction of the current is at right angles to the flux.

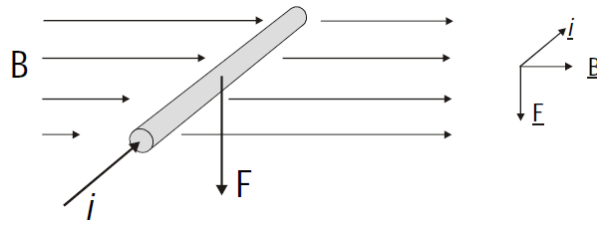


Figure 2.3: Production of torque.

Ampere's law tells us that a force F will be produced whose direction is orthogonal to both the current and flux and whose magnitude is given by :

$$F = Bli \quad (2.1)$$

where :

l is the length of wire within the field.

i is the current flowing in the wire.

In a DC motor the field B is usually produced by means of a strong permanent magnet. The flux is 'guided' by means of a steel magnetic circuit to two pole faces as shown in Figure 2.4. This part of motor is known as the stator.

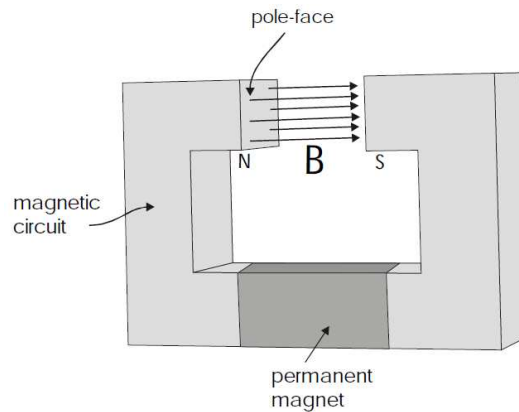


Figure 2.4: Production of magnetic field by permanent magnet.

Instead of having a single current-carrying wire, it is shaped into a coil so that the first conductor lies under the North pole face and the return conductor lies under the South pole face, as shown in Figure 2.5. The coil is supported in the middle on a shaft which rotates in bearings at either end of the motor. This part of the motor is known as the rotor or armature.

The net effect of the two forces - one upward and one downward - is to exert a turning moment or torque of value

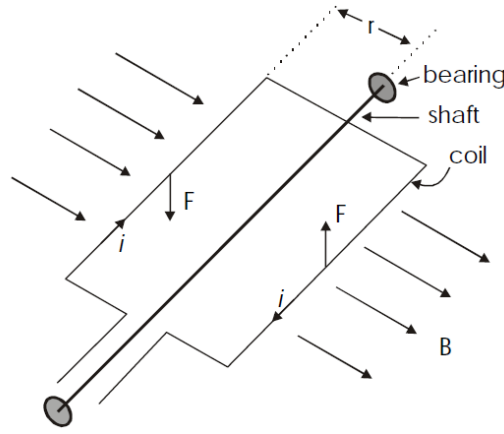


Figure 2.5: Coil carrying current.

$$T = 2Fr \quad (2.2)$$

On the coil, where r is the distance between the centre line of the shaft and the conductor. This causes the coil/shaft assembly to turn - anti-clockwise in the case of Figure 2.5, but this can be reversed by changing the sense of the current.

Two difficulties now arise :

- How do we pass current from a (stationary!) power supply into the coil while it is spinning?
- Once the coil has rotated through 180 from the position shown in Figure 2.5, the relative directions of B and I are reversed, so it would now produce clockwise torque and tend to rotate back again.

These problems are solved by means of a commutator which is really a mechanical switching device synchronized to the rotation of the rotor. It is shown in its simplest form in Figure 2.6 and consists of a split copper ring fixed to the shaft on which rest two carbon brushes to make a sliding electrical contact. Each segment of the ring is connected to one end of the coil. For half of the shaft's rotation, the coil current is in one direction but as soon as the brushes move over the gap onto the other segment, the current is reversed. This is arranged to happen just as the coil moves from being under one pole-face to the other. The result is a continuous rotation in one direction.

There is a considerable step from this simple description to a practical motor but the principle is as described above.

Supposing the coil has N turns instead of just one, the motor torque can be calculated from equ.(2.1) and equ.(2.2) as :

$$T_m = 2NBlri \quad (2.3)$$

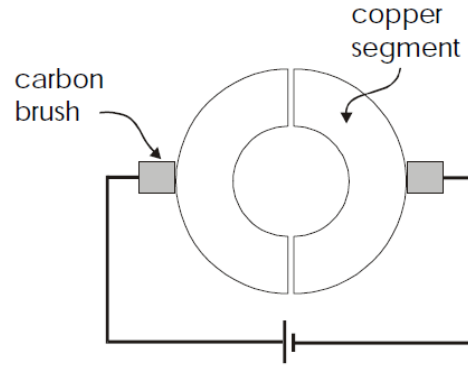


Figure 2.6: Split copper ring.

Note here that N , B , l , and r are constant for any particular motor so we can say :

$$T_m = K_t i \quad (2.4)$$

where K_t is known as the motor's torque constant, with units NmA^{-1} .

Equation 2.5 is the first of the equations which form the d.c. motor model.

2.4.2 Generated E.M.F.

According to Faraday's law, a coil of length l rotating in an uniform magnetic field of flux density B will generate an E.M.F. given by :

$$e = -\frac{d\lambda}{dt} \quad (2.5)$$

where λ is the flux linking the coil.

Consider the coil of length l and radius r to be at an angle θ_m to the magnetic field between the pole-faces, as shown in cross-section in Figure 2.7.

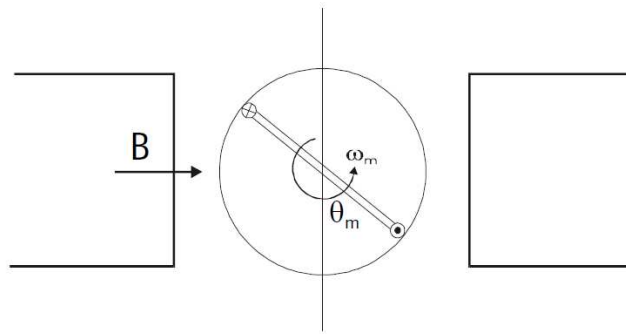


Figure 2.7: Cross section of dc motor

The magnetic flux linking the coil is given by:

$$\phi = Bl2r \cos \theta_m \quad (2.6)$$

and the magnetic flux linkage is:

$$\lambda = N\phi = 2NBlr \cos \theta_m \quad (2.7)$$

If the coil is rotating at a constant angular speed ω_m then its angle at any time t is $\omega_m t$. From equ.(2.5) we then have:

$$e = -\frac{d\lambda}{dt} = -2NBlr \frac{d(\cos \omega_m t)}{dt} = -2NBlr(-\omega_m \sin(\omega_m t)) = 2NBlr\omega_m \sin(\omega_m t) \quad (2.8)$$

When the coil is at $\theta_m = 90^\circ$, the term $\sin(\omega_m t) = 1$ and:

$$e = 2NBlr\omega_m = K_t\omega_m \quad (2.9)$$

Note that the same constant K_t appears in equ.(2.9) as in equ.(2.3) although it is conventional here to call it the back emf constant, with units of (volts per radian per second).

Equation (2.9) is the second of the equations which form the dc motor model.

2.4.3 Motor Modeling and Simulation

A common actuator in control systems is the DC motor [20, 21]. It directly provides rotary motion and, coupled with wheels or drums and cables, can provide translational motion. The electric circuit of the armature and the free body diagram of the rotor are shown in the following Figure 2.8.

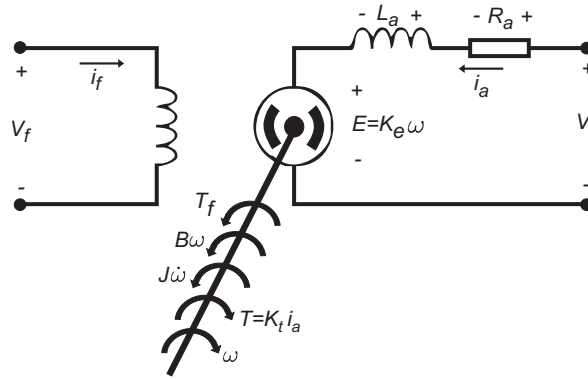


Figure 2.8: Complete equivalent model for dc motor

The motor torque, T , is related to the armature current, i_a , by a constant factor K_t . The back emf, E , is related to the rotational velocity by the constant factor K_e as in the following equations:

$$T = K_t i_a \quad (2.10)$$

$$E = K_e \omega \quad (2.11)$$

The dynamic of the dc motor may be expressed by the following equations:

$$V = R_a i_a + L_a \frac{di_a}{dt} + K_e \omega \quad (2.12)$$

$$K_t i_a = J \frac{d\omega}{dt} + B\omega + T_f \quad (2.13)$$

In the state-space form, the equations above can be expressed by choosing the rotational speed and electric current as the state variables and the voltage as an input. The output is chosen to be the rotational speed or current, so by representing equ.(2.12) and equ.(2.13) in a model of state space form provides:

$$\begin{bmatrix} \dot{i}_a \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} -\frac{R_a}{L_a} & -\frac{K_e}{L_a} \\ \frac{K_t}{J} & -\frac{B}{J} \end{bmatrix} \begin{bmatrix} i_a \\ \omega \end{bmatrix} + \begin{bmatrix} \frac{1}{L_a} & 0 \\ 0 & -\frac{1}{J} \end{bmatrix} \begin{bmatrix} V \\ T_f \end{bmatrix} \quad (2.14)$$

Using Laplace Transforms, the above modeling equations can be expressed in terms of s.

Armature current can be expressed using laplace transform as following:

$$I_a(s) = \left[\frac{1}{L_a s + R_a} \right] [V(s) - K_e \omega(s)] \quad (2.15)$$

Rotational speed also expressed as following:

$$I_a(s) = \left[\frac{1}{sJ + B} \right] [K_t I_a(s) - T_f] \quad (2.16)$$

Now the block diagram of dc motor can be derived from the above equations as shown in Figure (2.9).

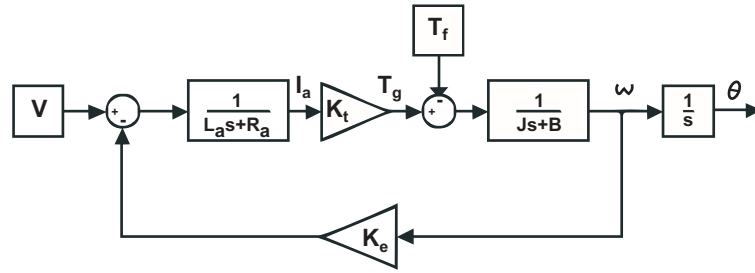


Figure 2.9: The Block diagram of dc motor

The transfer function from the input armature voltage, $V(s)$, to the output armature current, $I_a(s)$, directly follows:

$$\frac{I_a(s)}{V(s)} = \frac{\frac{1}{L_a} (s + \frac{B}{J})}{s^2 + s(\frac{R_a}{L_a} + \frac{B}{J}) + (\frac{R_a B}{L_a J} + \frac{K_e K_t}{L_a J})} \quad (2.17)$$

The transfer function in equ.(4.2) may be expressed as following:

$$\frac{I_a(s)}{V(s)} = \frac{s + \frac{B}{J}}{L_a s^2 + s(R_a + \frac{L_a B}{J}) + (\frac{R_a B}{J} + \frac{K_e K_t}{J})} \quad (2.18)$$

The transfer function from the input armature voltage, $V(s)$, to the rotational speed in (rad/sec), $W(s)$, directly follows:

$$\frac{W(s)}{V(s)} = \frac{\frac{K_t}{L_a J}}{s^2 + s(\frac{R_a}{L_a} + \frac{B}{J}) + (\frac{R_a B}{L_a J} + \frac{K_e K_t}{L_a J})} \quad (2.19)$$

or by the following equation:

$$\frac{W(s)}{V(s)} = \frac{\frac{K_t}{J}}{L_a s^2 + s(R_a + \frac{L_a B}{J}) + (\frac{R_a B}{J} + \frac{K_e K_t}{J})} \quad (2.20)$$

In metric (SI) units K_t ($\frac{N.m}{A}$) is equal to K_e ($\frac{Volts}{rad/sec}$) as justified in the literature [22].

2.5 Parameters Identification

Once we installed the experimental system and acquired the required signals needed, the stage of parameters identification of dc motor is now ready. The parameters needed for identification is summarized in table 2.1:

There are various possible approaches used for parameters identification, these approaches are studied in details and trying to apply them on the experimental system, these approaches are:

- Pseudo inverse technique
- Nonlinear least-square method
- Pattern search algorithm

2.5.1 Pseudo Inverse Technique

The pseudo inverse technique is a method capable of identifying an equivalent discrete transfer function of simple systems like motor, thermal system, mass damper systems etc with no zeros (only poles).

The transfer function of the system (speed over armature voltage) in z domain is of the following form:

$$\frac{W(z)}{V(z)} = \frac{b_0 z^2}{z^2 + a_1 z + a_2} \quad (2.21)$$

Table 2.1: DC motor parameters.

Parameter	Definition	Comment
K_e	Back-EMF constant	Dominate factor in determining motor's steady state speed for a given voltage
K_t	Torque Constant	Determines motor's required current for a given torque output
R_a	Terminal Resistance	Determines how much power will be dissipated in the motor for a given current level
L_a	Armature Inductance	Determines how fast current to the motor can be turned on
J	moment of inertia of the rotor	A measure of an object's resistance to changes in its rotation rate
B	Viscous (Damping) Friction	A measure of dynamic friction

From the above relation we can get the following equation

$$W_k = b_0 V_k + a_1 W_{k-1} + a_2 W_{k-2} \quad (2.22)$$

From the recorded values of input (V_k) and output (W_k) we can form the following equations.

$$W_{k-1} = b_0 V_{k-1} + a_1 W_{k-2} + a_2 W_{k-3} \quad (2.23)$$

...

$$W_{k-m} = b_0 V_{k-m} + a_1 W_{k-m-1} + a_2 W_{k-m-2} \quad (2.24)$$

The above set of equations can be represented as a matrix equation. This matrix equation is given below.

$$\begin{bmatrix} W_k \\ W_{k-1} \\ W_{k-2} \\ \vdots \\ \vdots \\ W_{k-m} \end{bmatrix} = \begin{bmatrix} W_{k-1} & W_{k-2} & \dots & W_{k-n} & V_k \\ W_{k-2} & W_{k-3} & \dots & W_{k-n-1} & V_{k-1} \\ \vdots & \vdots & & \vdots & \vdots \\ W_{k-m-1} & W_{k-m-2} & \dots & W_{k-m-n} & V_{k-m} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \\ b_0 \end{bmatrix} \quad (2.25)$$

The above equation can be represented as $W = [Q \ V_k] A$

Let $X = [Q \ V_k]$

Q : Matrix built using W_k values as shown in the above matrix equation.

X : Matrix built using $[Q \ V]$

$W = X A$

Size of the matrices in the above matrix equation is given below.

W : $(m+1) \times 1$

X : $(m+1) \times (n+1)$

A : $(n+1) \times 1$

Premultiply Matrix equation M1 by X^T on both sides

$X^T W = X^T X A$

Premultiply by $[X^T X]^{-1}$ on both sides of the Matrix Eqn

$[X^T X]^{-1} \cdot X^T \cdot W = [X^T X]^{-1} [X^T X] A$

$[X^T X]^{-1} \cdot X^T \cdot W = A$

$A = [[X^T X]^{-1} \cdot X^T] W$

Where $[X^T X]^{-1} \cdot X^T$ is called as the pseudo inverse of X

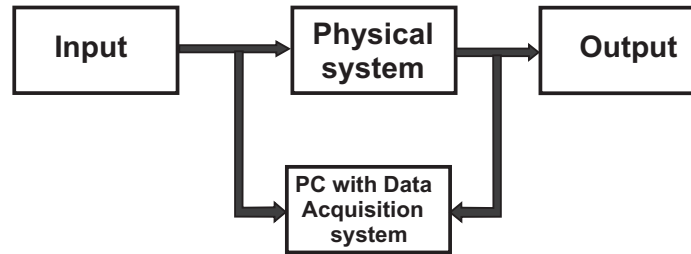


Figure 2.10: Experimental setup requirement

Figure (2.10) shows the experimental setup requirement prior to the parameter identification. This is the recording phase.

1. Deploy a data acquisition system, which can record the input and output at the required sampling frequency (according to the system dynamics).
2. Feed the system with rich inputs. (Inputs must change with time).
3. Record the inputs and corresponding outputs simultaneously using this data acquisition system

Now we have all the information required for identifying the mathematical model of the system. This method is programmed in MATLAB and is applied to our dc motor signals and give us these the transfer function of DC motor in Z-domain but it's difficult to extract the parameters from this transfer function. So we will look at another technique which may help us to find these parameters.

2.5.2 Nonlinear Least-Square Method

Nonlinear least squares regression extends linear least squares regression for use with a much larger and more general class of functions. Almost any function that can be written in closed form can be incorporated in a nonlinear regression model. Unlike linear regression, there are very few limitations on the way parameters can be used in the functional part of a nonlinear regression model. The way in which the unknown parameters in the function are estimated, however, is conceptually the same as it is in linear least squares regression.

As the name suggests, a nonlinear model is any model of the basic form:

$$y = f(x; \beta) + \epsilon \quad (2.26)$$

in which:

1. the functional part of the model is not linear with respect to the unknown parameters, β_0, β_1, \dots
2. the method of least squares is used to estimate the values of the unknown parameters.

Due to the way in which the unknown parameters of the function are usually estimated, however, it is often much easier to work with models that meet two additional criteria:

3. the function is smooth with respect to the unknown parameters.
4. the least squares criterion that is used to obtain the parameter estimates has a unique solution.

These last two criteria are not essential parts of the definition of a nonlinear least squares model, but are of practical importance.

The biggest advantage of nonlinear least squares regression over many other techniques is the broad range of functions that can be fit. Although many scientific and engineering processes can be described well using linear models, or other relatively simple types of models, there are many other processes that are inherently nonlinear. For example, the strengthening of concrete as it cures is a nonlinear process. Research on concrete strength shows that the strength increases quickly at first and then levels off, or approaches an asymptote in mathematical terms, over time. Linear models do not describe processes that asymptote very well because for all linear functions the function value can't increase or decrease at a declining rate as the explanatory variables go to the extremes. There are many types of nonlinear models, on the other hand, that describe the asymptotic behavior of a process well. Like the asymptotic behavior of some processes, other features of physical processes can often be expressed more easily using nonlinear models than with simpler model types.

Being a "least squares" procedure, nonlinear least squares has some of the same advantages (and disadvantages) that linear least squares regression has over other

methods. One common advantage is efficient use of data. Nonlinear regression can produce good estimates of the unknown parameters in the model with relatively small data sets. Another advantage that nonlinear least squares shares with linear least squares is a fairly well-developed theory for computing confidence, prediction and calibration intervals to answer scientific and engineering questions. In most cases the probabilistic interpretation of the intervals produced by nonlinear regression are only approximately correct, but these intervals still work very well in practice.

The major cost of moving to nonlinear least squares regression from simpler modeling techniques like linear least squares is the need to use iterative optimization procedures to compute the parameter estimates. With functions that are linear in the parameters, the least squares estimates of the parameters can always be obtained analytically, while that is generally not the case with nonlinear models. The use of iterative procedures requires the user to provide starting values for the unknown parameters before the software can begin the optimization. The starting values must be reasonably close to the as yet unknown parameter estimates or the optimization procedure may not converge. Bad starting values can also cause the software to converge to a local minimum rather than the global minimum that defines the least squares estimates.

Disadvantages shared with the linear least squares procedure includes a strong sensitivity to outliers. Just as in a linear least squares analysis, the presence of one or two outliers in the data can seriously affect the results of a nonlinear analysis. In addition there are unfortunately fewer model validation tools for the detection of outliers in nonlinear regression than there are for linear regression

2.5.3 Pattern Search Algorithm

Pattern search is an attractive alternative to the genetic algorithm, as it is often computationally less expensive and can minimize the same types of functions. Additionally, the Genetic Algorithm and Direct Search Toolbox includes a pattern search method that can solve problems with linear constraints [23].

Pattern search operates by searching a set of points called a pattern, which expands or shrinks depending on whether any point within the pattern has a lower objective function value than the current point. The search stops after a minimum pattern size is reached. Like the genetic algorithm, the pattern search algorithm does not use derivatives to determine descent, and so it works well on nondifferentiable, stochastic, and discontinuous objective functions. Pattern search is also effective at finding a global minimum because of the nature of its search.

A pattern is a set of vectors \mathbf{v}_i that the pattern search algorithm uses to determine which points to search at each iteration. The set \mathbf{v}_i is defined by the number of independent variables in the objective function, N , and the positive basis set. Two commonly used positive basis sets in pattern search algorithms are the maximal basis, with $2N$ vectors, and the minimal basis, with $N+1$ vectors.

With Generalized Pattern Search Algorithm (GPS) , the collection of vectors that form the pattern are fixed-direction vectors. For example, if there are three independent variables in the optimization problem, the default for a $2N$ positive basis consists of the following pattern vectors:

$$\begin{aligned} v1 &= [1 \ 0 \ 0], v2 = [0 \ 1 \ 0], v3 = [0 \ 0 \ 1] \\ v4 &= [-1 \ 0 \ 0], v5 = [0 \ -1 \ 0], v6 = [0 \ 0 \ -1] \end{aligned}$$

An $N+1$ positive basis consists of the following default pattern vectors.

$$\begin{aligned} v1 &= [1 \ 0 \ 0], v2 = [0 \ 1 \ 0], v3 = [0 \ 0 \ 1] \\ v4 &= [-1 \ -1 \ -1] \end{aligned}$$

with Mesh Adaptive Search Algorithm (MADS), the collection of vectors that form the pattern are randomly selected by the algorithm. Depending on the poll method choice, the number of vectors selected will be $2N$ or $N+1$. As in GPS, $2N$ vectors consist of N vectors and their N negatives, while $N+1$ vectors consist of N vectors and one that is the negative of the sum of the others.

At each step, the pattern search algorithm searches a set of points, called a mesh, for a point that improves the objective function. The GPS and MADS algorithms form the mesh by

- Generating a set of vectors d_i by multiplying each pattern vector v_i by a scalar Δ^m . Δ^m is called the mesh size.
- Adding the d_i to the current point the point with the best objective function value found at the previous step.

The pattern vector that produces a mesh point is called its direction.

At each step, the algorithm polls the points in the current mesh by computing their objective function values. When the Complete poll option has the (default) setting Off, the algorithm stops polling the mesh points as soon as it finds a point whose objective function value is less than that of the current point. If this occurs, the poll is called successful and the point it finds becomes the current point at the next iteration.

The algorithm only computes the mesh points and their objective function values up to the point at which it stops the poll. If the algorithm fails to find a point that improves the objective function, the poll is called unsuccessful and the current point stays the same at the next iteration.

When the Complete poll option has the setting On, the algorithm computes the objective function values at all mesh points. The algorithm then compares the mesh point with the smallest objective function value to the current point. If that mesh point has a smaller value than the current point, the poll is successful.

After polling, the algorithm changes the value of the mesh size Δ^m . The default is to multiply Δ^m by 2 after a successful poll, and by 0.5 after an unsuccessful poll.

The pattern search algorithms find a sequence of points, x_0, x_1, x_2, \dots , that approaches an optimal point. The value of the objective function either decreases or remains the same from each point in the sequence to the next

The algorithm stops when any of the following conditions occurs:

- The mesh size is less than Mesh tolerance.
- The number of iterations performed by the algorithm reaches the value of Max iteration.
- The total number of objective function evaluations performed by the algorithm reaches the value of Max function evaluations.
- The time, in seconds, the algorithm runs until it reaches the value of Time limit.
- The distance between the point found in two consecutive iterations and the mesh size are both less than X tolerance.
- The change in the objective function in two consecutive iterations and the mesh size are both less than Function tolerance.

Nonlinear constraint tolerance is not used as stopping criterion. It determines the feasibility with respect to nonlinear constraints.

Chapter 3

System Hardware Development

3.1 Introduction

System Hardware development begin with searching for DC motor that may be contain on parameters information, unfortunately we can't find this kind of motor. Finally, we find a DC motor to perform our experiments. The DC motor is the backbone of our system. This motor has the nominal characteristics printed on its nameplate as shown in Table (3.1).

Table 3.1: Specification of experimental dc motor.

Rated power	200 W
Rated speed	1500 rpm
Armature rated voltage	110 V
Armature rated current	3 A
Field voltage	2.5 V
Field current	1.3 A
Magnetic break	24 V

This DC motor is shown in Figure 3.1.
The DC motor have the following wires:

- Two wires for the armature terminals.
- Two wires for the field terminals.
- Two wires for the magnetic break (24VDC).
- Two wires for overheat protection.

From the figure drawn on its nameplate it's clearly that it's series wound motor, but its characteristics is different than permanent magnet dc motor characteristics that will studied in out thesis, so we try to change to use it as a permanent magnet DC motor by applying a constant voltage to its field terminals. This is done by

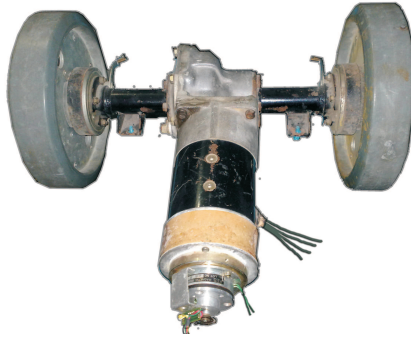


Figure 3.1: DC motor used in our experiment.

connecting it as series connection then measure the current and voltage field and apply it separately. Finally we success on using it in our study, because permanent magnet is the same characteristics as separately excited.

In the next section we begin to design our system to use it in our experiment.

3.2 Hardware Design

The experimental system consists of the following components as shown in Figure (3.2):

1. Separately excited DC motor
2. Power supply unit
3. Field driver circuit
4. Armature driver circuit
5. Interfacing circuit
6. Current and voltage panel meters
7. NI-DAQ USB-6008

The design is consisting of driving circuits and sensing circuits connected to DC motor to establish the goal of our thesis. We used the power MOSFET as the main switching element for driving the motor's field and armature with the specific voltage with the helping of PWM generator circuit to achieve a variable voltage needed for testing and identification process.

The next section will focus on the MOSFET as a main driving element for DC motor.

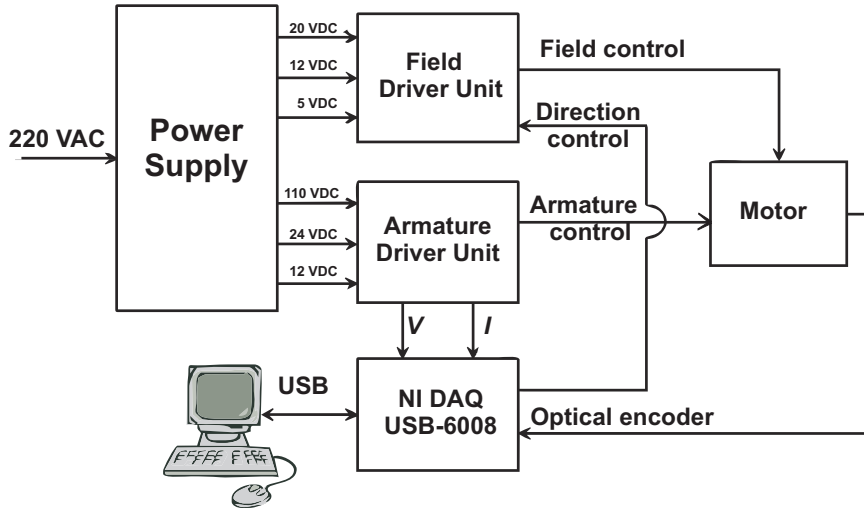


Figure 3.2: Architecture of the system

3.3 The MOSFET as a Switch

3.3.1 Introduction

Metal oxide semiconductor field-effect transistors (MOSFETs) are incredibly popular transistors that in some ways resemble JFETs [24]. For instance, when a small voltage is applied at its gate lead, the current flow through its drain-source channel is altered. However, unlike JFETs, MOSFETs have larger gate lead input impedances ($\geq 10^{14}\Omega$, as compared with $\sim 10^9\Omega$ for JFETs), which means that they draw almost no gate current whatsoever. This increased input impedance is made possible by placing a metal oxide insulator between the gate-drain/source channel. There is a price to pay for this increased amount of input impedance, which amounts to a very low gate to channel capacitance (a few pF). If too much static electricity builds up on the gate of certain types of MOSFETs during handling, the accumulated charge may break through the gate and destroy the MOSFET. (Some MOSFETs are designed with safe guards against this breakdownbut not all.)

The two major kinds of MOSFETs include enhancement-type MOSFETs and depletion type MOSFETs. A depletion-type MOSFET is normally on (maximum current flows from drain to source) when no difference in voltage exists between the gate and source terminals ($V_{GS} = V_G - V_S = 0V$). However, if a voltage is applied to its gate lead, the drain-source channel becomes more resistivea behavior similar to a JFET. An enhancement-type MOSFET is normally off (minimum current flows from drain to source) when $V_{GS} = 0V$. However, if a voltage is applied to its gate lead, the drain-source channel becomes less resistive.

Both enhancement-type and depletion-type MOSFETs come in either n-channel or p-channel forms. For an n-channel depletion-type MOSFET, a negative gate-source voltage ($V_G < V_S$) increases the drain-source channel resistance, whereas

for a p-channel depletion-type MOSFET, a positive gate-source voltage ($V_G > V_S$) increases the channel resistance. For an n-channel enhancement-type MOSFET, a positive gate-source voltage ($V_G > V_S$) decreases the drain-source channel resistance, whereas for a p-channel enhancement-type MOSFET, a negative gate-source voltage ($V_G < V_S$) decreases the channel resistance.

MOSFETs are perhaps the most popular transistors used today; they draw very little input current, are easy to make (require few ingredients), can be made extremely small, and consume very little power. In terms of applications, MOSFETs are used in ultrahigh input impedance amplifier circuits, voltage-controlled resistor circuits, switching circuits, and found with large-scale integrated digital ICs.

Like JFETs, MOSFETs have small transconductance values when compared with bipolar transistors. In terms of amplifier applications, this can lead to decreased gain values. For this reason, you will rarely see MOSFETs in simple amplifier circuits, unless there is a need for ultrahigh input impedance and low input current features.

As the motor load is inductive, a simple "Free-wheeling" diode is connected across the load to dissipate any back emf generated by the motor when the MOSFET turns it "OFF". The Zener diode is used to prevent excessive gate-source input voltages.

3.3.2 Choosing a MOSFET

Choosing the right power MOSFET that will be used to drive the sufficient current to DC motor is an important part. This choice depends on load specifications, the maximum voltage and maximum current of the load, because MOSFETs are generally advertised by their maximum drain current, maximum drain-source voltage and static drain-source on resistance. To examine the parameters of MOSFETs, it is useful to have a sample datasheet to hand. If you cannot find a single MOSFET with a high enough maximum drain current, then you can connect more than one in parallel.

Selecting the right MOSFET driver for the application requires a thorough understanding of power dissipation in relation to the MOSFET's gate charge and operating frequencies. For example, charging and discharging a MOSFET's gate requires the same amount of energy, regardless of how fast or slow the gate voltage transitions are [25].

A MOSFET driver's power dissipation capabilities are determined by three key elements:

- Power dissipation due to charging and discharging the MOSFET's gate capacitance.
- Power dissipation due to the MOSFET driver's quiescent-current draw.

- Power dissipation due to cross-conduction (shoot-through) current in the MOSFET driver.

Of these three elements, power dissipation due to the charging and discharging of the MOSFET's gate capacitance is most important, especially at lower switching frequencies. This is given by:

$$P_c = C_g \times V_{dd}^2 \times F \quad (3.1)$$

where C_g = MOSFET gate capacitance, V_{dd} = supply voltage of MOSFET driver (V), and F = switching frequency.

In addition to power dissipation, designers must understand the peak drive current required from the MOSFET driver and the associated turn-on and -off times. Matching the MOSFET driver to the MOSFET in an application depends on how fast the application requires the power MOSFET to be switched on and off.

The optimum rise or fall time in any application is based on many requirements, such as EMI, switching losses, lead/circuit inductance, and switching frequency. The relationship between gate capacitance, transition times, and the MOSFET driver current rating is given by:

$$dT = [dV \times C]/I \quad (3.2)$$

where dT = turn-on/turn-off time, dV = gate voltage, C = gate capacitance, and I = MOSFET peak drive current.

The total MOSFET gate capacitance can be properly determined by looking at the total gate charge (Q_G). Gate charge Q_G is given by:

$$Q_G = C \times V \quad (3.3)$$

Then $I = Q_G/dT$.

This method assumes a constant current. A good rule of thumb is that the average value found is half of the MOSFET driver's peak current rating. MOSFET drivers are rated by the driver output peak current drive capability.

The peak current rating is typically stated for the part's maximum bias voltage. This means that, if the MOSFET driver is being used with a lower bias voltage, its peak current drive capability will be reduced.

Another method designers can use to selecting the appropriate MOSFET driver is to use a time-constant approach. In this approach, the MOSFET driver resistance, any external gate resistance, and the lumped capacitance are used.

$$T_{charge} = ((R_{driver} + R_{gate}) \times C_{total}) \times TC \quad (3.4)$$

where $R_{driver} = R_{DS(on)}$ of the output driver stage, R_{gate} = any external gate resistance between the driver and MOSFET gate, C_{total} = total gate capacitance, and TC = number of time constants.

3.3.3 MOSFET Gate Driver

To turn a power MOSFET on, the gate terminal must be set to a voltage at least 10 volts greater than the source terminal (about 4 volts for logic level MOSFETs). This is comfortably above the $V_{gs_{th}}$ parameter.

One feature of power MOSFETs is that they have a large stray capacitance between the gate and the other terminals, C_{iss} . The effect of this is that when the pulse to the gate terminal arrives, it must first charge this capacitance up before the gate voltage can reach the 10 volts required. The gate terminal then effectively does take current. Therefore the circuit that drives the gate terminal should be capable of supplying a reasonable current so the stray capacitance can be charged up as quickly as possible. The best way to do this is to use a dedicated MOSFET driver chip.

There are a lot of MOSFET driver chips available from several companies. Some are shown with links to the datasheets in the table below. Some require the MOSFET source terminal to be grounded (for the lower 2 MOSFETs in a full bridge or just a simple switching circuit). Some can drive a MOSFET with the source at a higher voltage. These have an on-chip charge pump, which means they can generate the 22 volts required to turn the upper MOSFET in a full bridge on. The TDA340 even controls the swicthing sequence for you. Some can supply as much as 6 Amps current as a very short pulse to charge up the stray gate capacitance.

Often you will see a low value resistor between the MOSFET driver and the MOSFET gate terminal. This is to dampen down any ringing oscillations caused by the lead inductance and gate capacitance which can otherwise exceed the maximum voltage allowed on the gate terminal. It also slows down the rate at which the MOSFET turns on and off. This can be useful if the intrinsic diodes in the MOSFET do not turn on fast enough.

Table 3.2: MOSFET Drivers.

Manufacturer	IC	Features
Maxim and others	ICL 7667	Dual inverting driver
Maxim	MAX622/MAX1614	High side drivers
Maxim	MAX626/MAX627/MAX628	Low side drivers
International Rectifier	IR2110	High and low side driver
Harris / Intersil	HIP4080/4081/4082	Full bridge drivers
SGS Thomson (ST)	TD340	New full bridge driver with analogue or PWM speed demand input

3.4 Pulse Width Modulation (PWM)

PWM, or Pulse Width Modulation is a powerful way of controlling analog circuits and systems, using the digital outputs of microprocessors. Defining the term, we can say that PWM is the way we control a digital signal simulating an analog one, by means of altering it's state and frequency of this.

The PWM is actually a square wave modulated. This modulation infects on the frequency (clock cycle) and the duty cycle of the signal. PWM signal is characterized from the duty clock and the duty cycle. The amplitude of the signal remains stable during time (except of course from the rising and falling ramps). The clock cycle is measured in Hz and the duty cycle is measured in hundred percent (These are the basic parameters that characterizes a PWM signal).

One of the most popular usages of PWM is the control of voltage delivered to loads. Those loads could be for example an LED which would utilize a LED dimmer, or a motor that could be a simple DC motor and would be converted into a controlled speed DC motor, as used for example in the modern PC motherboard fans. A PWM signal with 100% duty cycle would deliver 100% of the voltage. It would be like a DC power supply. But by altering the duty cycle, the result is to reduce the area of the power delivered to the load. The total power delivered to the connected load each time, is the area under the positive state of the PWM. It is clearly seen that by altering the duty cycle, we can alter the power delivered by the supply. And because the wave form is a square wave, the power supplied each time is calculated by equ.(3.5).

$$P_{Delivered} = P_{Supplied} \times DutyCycle \quad (3.5)$$

Due to the efficiency and simplicity of PWM, as well as the flexibility of this modulation type, there are numbers of applications that it can take place. Using PWM someone can modulate transmit and/or storage analog signals like audio/voice telecommunications and music. Switching power supplies that uses this technology are much more energy efficient than the classic power supplies, reaching up to 60% power saving! The output power and/or voltage can be controlled either digitally using a micro-controller or with the old classic potentiometer. Stepper motors can be easily controlled using PWM, as well as DC motors. The torque and speed of a DC motor can be controlled by altering the voltage or the duty cycle of the PWM easily. PWM is widely used in dimmer circuits that could control a variety of lamps including LEDs.

In our thesis a Pulse width modulation (PWM) signal at a frequency of 250 Hz is used to adjust the magnitude of excitation voltage applied on the field terminals. Usually the frequency is set to be at least 5 times higher than the rotation speed of the motor. In our case the max speed of motor is 1500 rpm (25 rps) the frequency is better be higher than $5 \times 25Hz = 125Hz$. A frequency of 250 Hz will be a suitable value. Reversing the rotation direction of the motor is easily done by reversing the field excitation. Therefore an additional relay is employed to facilitate user control

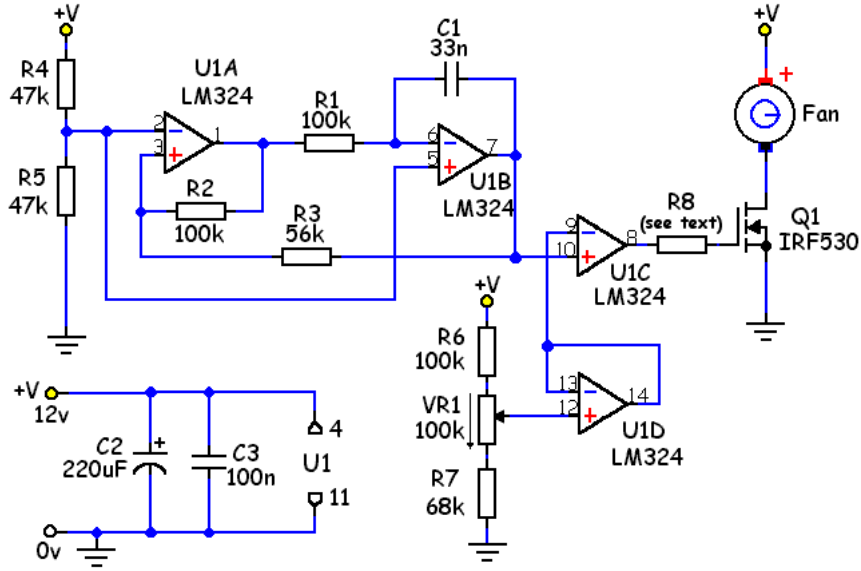


Figure 3.3: A Practical PWM Circuit.

of motor direction. Field driver circuit is shown in Figure (3.4), Figure (B.1) and Figure (B.2).

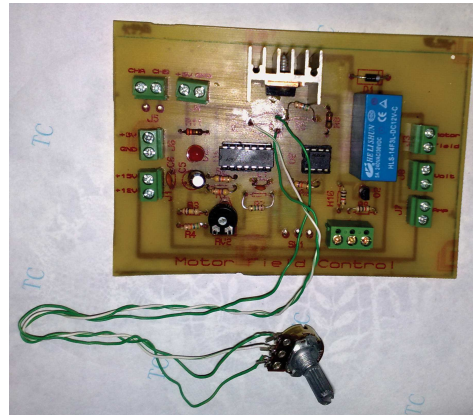


Figure 3.4: Field driver circuit.

Armature driver circuit is the same version of filed driver circuit. Voltage and current sensing components are added in this circuit. Moreover, a 100nf capacitor is connected across the motor armature in order to reduce electromagnetic interference (EMI). Both field and armature driver circuits are featured by the ability to be controlled locally or remotely via the data acquisition card. Two digital panel meters are used to monitor the current and voltage of the armature winding and a third one is used to monitor field current. Armature driver circuit is shown in Figure (3.5), Figure (B.3) and Figure (B.4).

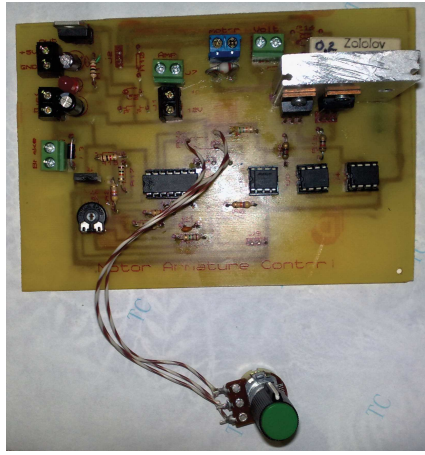


Figure 3.5: Armature driver circuit.

3.5 Power Supply

The power supply unit consists of a traditional components which supply all necessary voltage levels. Separate voltage levels are required for the field driver unit and for the armature driver unit. The first unit requires voltage supplies of +20VDC, +12VDC and +5VDC. On the other hand, the second unit requires +110VDC, +24VDC and +12VDC voltage levels. Figure 3.6 show the final design of the power supply box. The schematics for the power circuits are in Figure (B.5) and Figure (B.6).

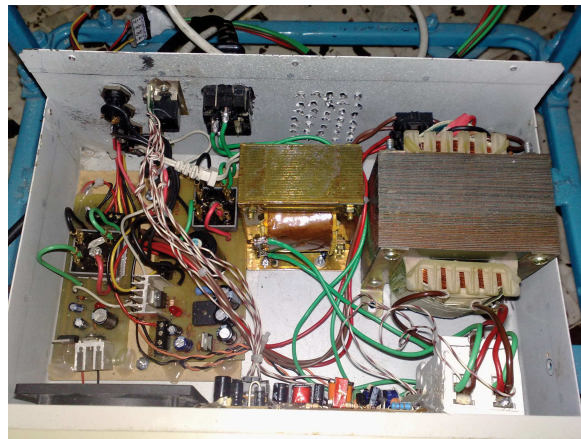


Figure 3.6: Power Box.

Important thing in power supply is to use an isolated transformers, because if the transformer is not isolated so it's easy for interference to cause problems and damage to electronic components.

3.6 Voltage Doubler

A voltage doubler is an electric circuit with an AC input and a DC output of roughly twice the peak input voltage. They are a variety of voltage multiplier circuit and are often, but not always, a single stage of a general form of such circuits. The term is usually applied to circuits consisting of rectifying diodes and capacitors only, other means of doubling voltages are not included [31].

The main goal for using this circuit is to double the power source of 12V to get +24VDC suitable to drive the magnetic break of dc motor. The schematic of the circuit used is shown in Figure (3.7). The circuits used two diodes and two capacitors beside the transformer and with a +24VDC regulator.

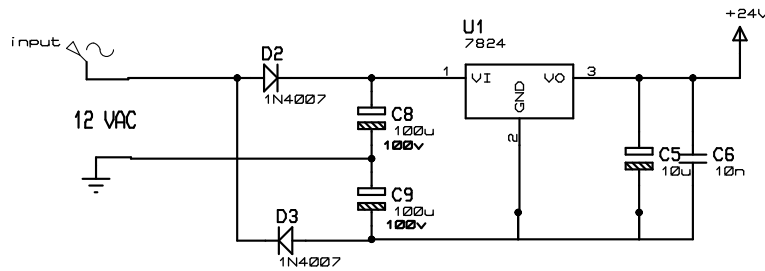


Figure 3.7: Voltage Doubler circuit.

3.7 Sensors and Feedback Elements

3.7.1 Introduction

Sensors are a critical component in a motor control system [26]. They are used to sense the current, position, speed and direction of the rotating motor. Recent advancements in sensor technology have improved the accuracy and reliability of sensors, while reducing the cost. Many sensors are now available that integrate the sensor and signal-conditioning circuitry into a single package. In most motor control systems, several sensors are used to provide feedback information on the motor. These sensors are used in the control loop and to improve the reliability by detecting fault conditions that may damage the motor. As an example, Figure(3.8) provides a block diagram of a DC motor control system to show the sensor feedback provided for a typical motor control.

A list of the sensors that can be used to feedback information to a DAQ or microcontroller are listed below:

- Current sensors

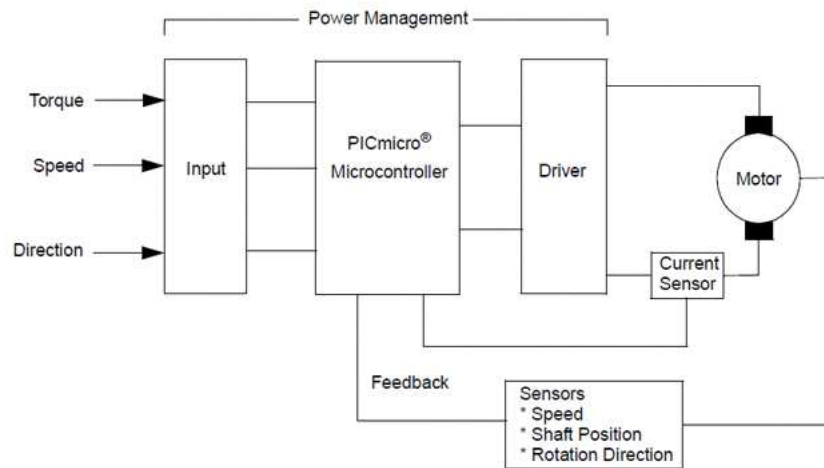


Figure 3.8: Typical DC Motor Block Diagram.

- Shunt resistor
- Current-sensing transformer
- Hall effect current sensor
- Voltage sensors
 - Voltage Divider
- Speed/position sensors
 - Quadrature encoder
 - Hall effect tachometer
- Back EMF/Sensorless control method

3.7.2 Current Sensors

The three most popular current sensors in motor control applications are:

- Shunt resistors
- Hall effect sensors
- Current transformers

Shunt resistors are popular current sensors because they provide an accurate measurement at a low cost. Hall effect current sensors are widely used because they provide a non-intrusive measurement and are available in a small IC package that combines the sensor and signal-conditioning circuit. Current-sensing transformers are also a popular sensor technology, especially in high-current or AC line-monitoring applications. A summary of the advantages and disadvantages of each of the current sensors is provided in Table (3.3).

Table 3.3: Current sensing methods.

Current Sensing Method	Shunt Resistor	Hall Effect	Current Sensing Transformer
Accuracy	Good	Good	Medium
Accuracy vs. Temperature	Good	Poor	Good
Cost	Low	High	Medium
Isolation	No	Yes	Yes
High Current-Measuring Capability	Poor	Good	Good
DC Offset Problem	Yes	No	No
Saturation or Hysteresis Problem	No	Yes	Yes
Power Consumption	High	Low	Low
Intrusive Measurement	Yes	No	No
AC/DC Measurements	Both	Both	Only AC

3.7.3 Shunt Resistors as a Current Sensor

Shunt resistors are a popular current-sensing sensor because of their low cost and good accuracy. The voltage drop across a known low value resistor is monitored in order to determine the current flowing through the load. If the resistor is small in magnitude, the ‘voltage drop will be small and the measurement will not have a major effect on the motor circuit. The power dissipation of the resistance makes current shunts impractical for measurements of more than approximately 20 amperes. The selection criteria of a shunt current resistor requires the evaluation of several trade-offs, including:

- Increasing R_{SENSE} increases the V_{SENSE} voltage, which makes the voltage offset (VOS) and input bias current offset (IOS) amplifier errors less significant.
- A large R_{SENSE} value causes a voltage loss and a reduction in the power efficiency due to the $I^2 \times R$ loss of the resistor.
- A large R_{SENSE} value will cause a voltage offset to the load in a low-side measurement that may impact the EMI characteristics and noise sensitivity of the system.
- Special-purpose, low inductance resistors are required if the current has a high-frequency content.
- The power rating of R_{SENSE} must be evaluated because the $I^2 \times R$ power dissipation can produce self heating and a change in the nominal resistance of the shunt.

Special-purpose, shunt current measurement resistors are available from a number of vendors. If standard resistors are used, it is recommended that metal-film resistors be used rather than wire-wound resistors that have a relatively large inductance.

3.7.4 Speed/Position Sensor

A quadrature encoder can be used to provide the speed, direction and shaft position of a rotating motor. A simplified block diagram of an optical quadrature encoder is shown in Figure (3.9).

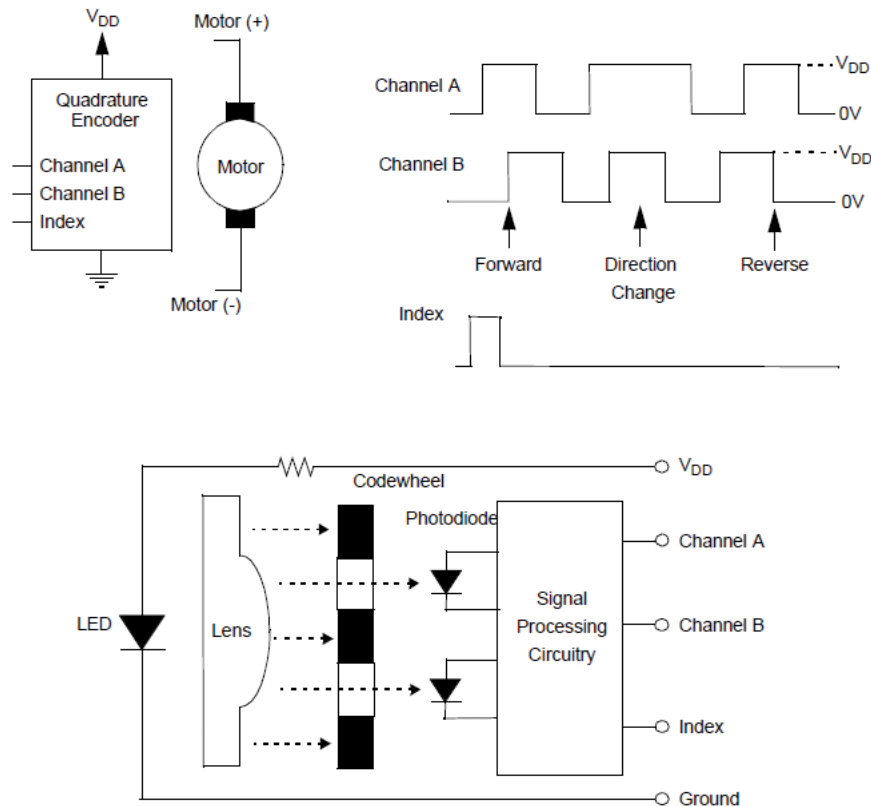


Figure 3.9: Quadrature Encoder.

The typical quadrature encoder is packaged inside the motor assembly and provides three logic-level signals that can be directly connected to the microcontroller. Motor speed is determined by the frequency of the Channel A and B signals. Note that the counts-per-revolution (CPR) depends on the location of the encoder and whether motor-gearing is used. The phase relationship between Channel A and B can be used to determine if the motor is turning in either a forward or reverse direction. The Index signal provides the position of the motor and, typically, a single pulse is generated for every 360 degrees of shaft rotation. The quadrature encoders speed and direction information can be determined either with discrete logic, a quadrature encoder logic IC or a PICmicro microcontroller. Vendors, such as LSI Computer Systems, offer an IC that converts the three encoder signals to

a signal that represents the velocity, position and distance that the motor has moved. Alternatively, the encoder information can be obtained from the hardware registers and software logic inside a PICmicro microcontroller. For example, the PIC18FXX31 dsPIC MCUs have a Quadrature Encoder Interface logic integrated into the processor.

3.7.5 Voltage Sensing

The voltage of any test point can be sensed using voltage divider. In electronics, a voltage divider (also known as a potential divider) is a simple linear circuit that produces an output voltage (V_{out}) that is a fraction of its input voltage (V_{in}). Voltage division refers to the partitioning of a voltage among the components of the divider.

The formula governing a voltage divider is similar to that for a current divider, but the ratio describing voltage division places the selected impedance in the numerator, unlike current division where it is the unselected components that enter the numerator.

A simple example of a voltage divider consists of two resistors in series or a potentiometer. It is commonly used to create a reference voltage, and may also be used as a signal attenuator at low frequencies.

voltage divider referenced to ground is created by connecting two electrical impedances in series, as shown in Figure (3.10). The input voltage is applied across the series impedances Z_1 and Z_2 and the output is the voltage across Z_2 . Z_1 and Z_2 may be composed of any combination of elements such as resistors, inductors and capacitors.

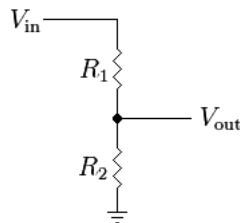


Figure 3.10: Voltage Divider Circuit.

Chapter 4

Experimentation, Testing and Results

4.1 Experimental System

Figure 4.1 show the experimental system which is designed to meet the requirement for the interfacing with DAQ and for the process of parameters identification. The experimental setup is designed for the theoretical study and practical investigation of basic and advanced electrical machines and control engineering principles. This can include system dynamics modelling, identification, analysis and various controllers design by classical and modern methods. The experimental system is equipped with terminals for ease of connection and interfacing with many controllers such as DAQ, PC, Microcontrollers, DSP Kits and FPGA kits to perform the controllers needed.



Figure 4.1: General view of the experimental system

The next section illustrated the steps used in programming the system in MATLAB for measuring signals which it's next used in the parameters identification process.

4.2 DC Motor Speed

Speed of dc motor is measured using optical encoder (GP1A30R) which is mounted in the shaft of motor and not at the gear output shaft. The disk of optical encoder has 120 slits. The dc motor speed is measured using the NI DAQ with a suitable code written in MATLAB.

Although many data acquisition devices have counters built into them (in our case USB-6008 have 32-bit counter), sometimes the built-in counters are not sufficient. For example, our signal might be too noisy to be reliably counted using a standard counter, or our signal might not conform to the voltage requirements of the counter.

In this thesis we use an analog input channel from a data acquisition device to acquire the signal and then postprocess it in MATLAB to count the number of pulses. This method uses an analog input channel from the NI USB-6008. The hardware for this demo is a mechanical push button switch that is switching a 5-volt signal on and off. It is difficult to count the number of times that a mechanical push button has been pressed because of bouncing.

This method uses the following steps:

1. Create the Analog Input Object: Before acquiring any data, we must create an analog input object and specify the channels from which to acquire data. In our case we use the AI2 channel in differential mode.
2. Configure the Analog Input Object: Now that we have created the analog input object, it must be configured for this particular acquisition. For this acquisition, configure the hardware with:
 - The SampleRate property set to 3300 samples per second
 - The SamplesPerTrigger property set to Inf samples, and we run the acquisition for 10 seconds so we collected a total of 33000 samples.
3. Acquire and View the Data: To acquire the data, we issue the start command to the analog input object. we can then use the pause command to wait for the acquisition to be completed, and the getdata command to return the data to MATLAB. The getdata command can return the time that each sample was taken, in addition to the actual sample. Some samples of the acquiring pulses is show in Figure (4.2).

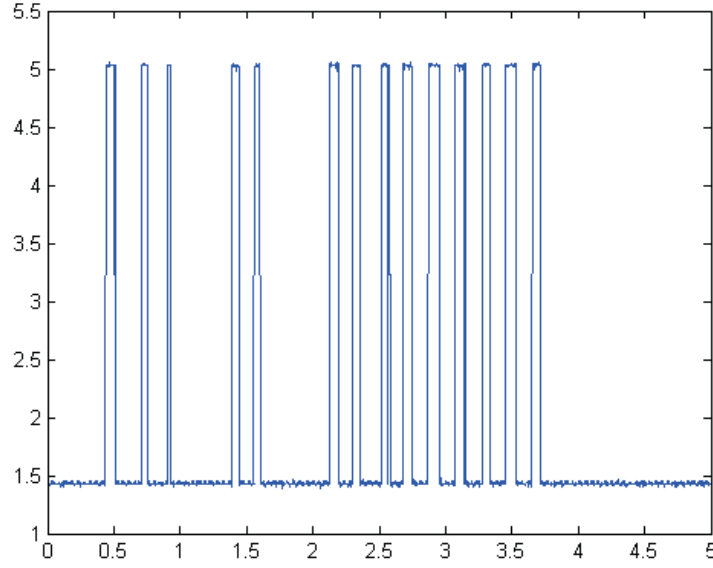


Figure 4.2: Samples of acquiring pulses.

4. Find the Pulse Edges: Finding the edges of the pulses is easy in MATLAB. we simply set a threshold about 2.4V and then look for consecutive samples that are on opposite sides of the threshold. The easiest way to do this to create a duplicate vector of data that is offset from the original data by one sample. Because the pulse time is important, we need to find the rising and falling edges of each pulse. The output of this step is a plot where rising and falling edges are shown with overlapping x's and o's as illustrated in Figure (4.3).
5. Count the Pulses: Now it is possible to count the number of pulses in the data signal. We only need to count the number of edges (rising or falling).
6. Represent speed in rpm: The collected pulses from optical encoder at specific time 10ms at analog input of DAQ is used to measure the rotational speed of dc motor in *rpm* using the relation in equ. (4.1)

$$Speed(rpm) = \frac{60}{10ms} \times \left[\frac{\#of counting pulses}{\#of slits} \right] \quad (4.1)$$

4.3 Panel Meters

Current and voltage digital panel meters are used to monitor the current and voltage of the armature winding of DC motor, while the other current panel meter is monitoring the current of motor field. These panels give us indication and help for the response of DC motor on controlling. Each panel meter has a +5VDC power separated from the sensing circuit. The panel meters are shown in Figure (4.4).

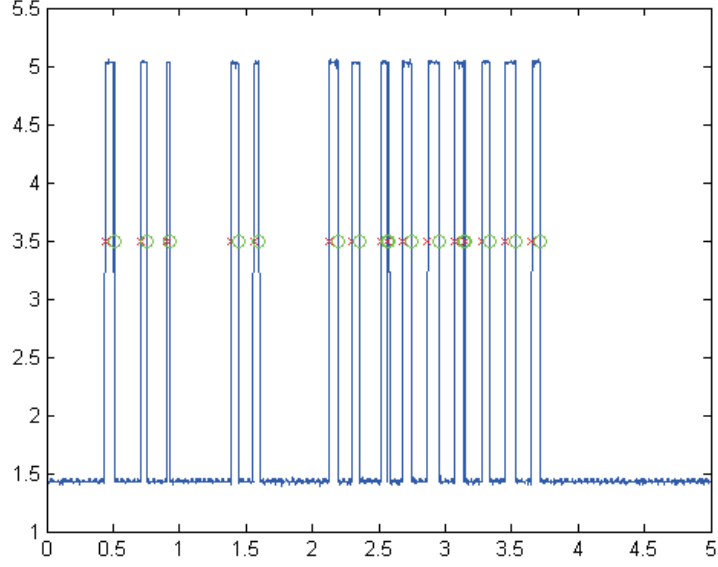


Figure 4.3: Pulse Edges.



Figure 4.4: Digital panel meters.

4.4 Sensing Circuit

NIDAQ USB-6008 from national instruments is used to acquiring test signals from DC motor and to send control signals to the motor [32]. We used it to acquiring armature voltage, armature current and speed of motor and made some analysis for parameter identification. To prepare the suitable signals needed to be sensed by NIDAQ which are armature current, armature voltage and speed of dc motor we used a voltage divider to sense the armature voltage. The armature current is sensed using a small resistance (0.2Ω) connected in series with the motor. The sensing for both armature voltage and armature current is shown in Figure (4.5).

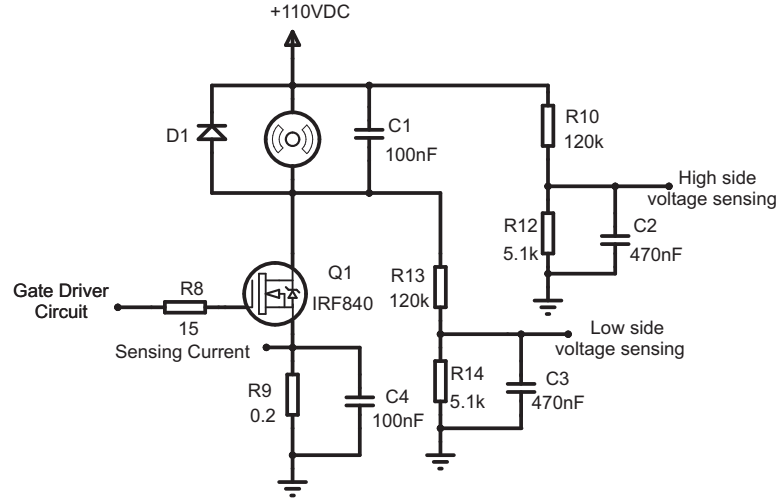


Figure 4.5: Sensing circuit of the system.

4.5 Interfacing Circuit

The interfacing circuit is designed for the flexibility of connection between driver circuits and NIDAQ. We used a DB9 connection which each pin has a specific purpose. The specification for each pin is summarized in Table (4.1).

Table 4.1: Interfacing circuit DB9 pinout .

Pin	Purpose
1	GND of field circuit
2	Control of motor direction
3	Sensing armature current
4	Low side armature voltage sensing
5	High side armature voltage sensing
6	GND of armature circuit
7	Control speed of motor using PWM
8	Magnetic break of motor
9	CHA of optical encoder

The interfacing circuit consists of a voltage controlled PWM generator to generate a PWM signal for controlling speed of motor. This circuit also consists of an optocoupler to isolate NIDAQ from driver circuits, so we can control direction and break of motor.

The finishing PCB for the interfacing circuit is shown in Figure (4.6). Which is ready for the connection between drivers through DB9 connector and NIDAQ.

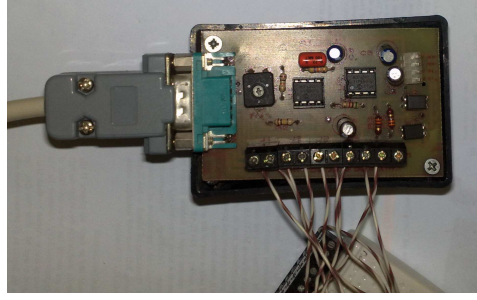


Figure 4.6: PCB of interfacing circuit

4.6 Data Acquisition

The data acquisition used in the thesis is the national instruments USB-6008. The National Instruments USB-6008 is a low-cost, multifunction data acquisition device (DAQ). It has 8 analog inputs, 2 analog outputs, and 12 digital input/outputs. The digital channels are divided into two ports. When one or more channels on each port is set to either input or output, the port is locked into that particular mode.

The USB-6008 uses the NI-DAQmx driver software and is compatible with LabVIEW, Measurement Studio for Visual Studio, .NET, LabWindows/CVI, and MATLAB (version R2006a and newer). The image of this device is shown in Figure (4.7).



Figure 4.7: The National Instruments USB-6008

NI DAQ USB-6008 which is available in control laboratories is used to apply the parameters identification analysis in MATLAB using DAQ toolbox. The specifications for connections to NI DAQ was summarized in table 4.2.

For analysis, processing data and perform parameters identification on signals we used Data Acquisition Toolbox (DAT) of MATLAB. Data Acquisition Toolbox software provides a complete set of tools for analog input, analog output, and digital I/O from a variety of PC-compatible data acquisition hardware. The toolbox lets you configure your external hardware devices, read data into MATLAB and Simulink environments for immediate analysis, and send out data.

Table 4.2: NI DAQ inputs and outputs.

Channel	Mode	Purpose
Analog 0 (AI0)	Differential input	measuring voltage at motor terminals
Analog 1 (AI1)	Differential input	measuring current flowing in motor
Analog 2 (AI2)	Differential input	count # of pluses from optical encoder
Analog 0 (AO0)	Differential output	controlling speed of motor
Digital pin P0.0	Digital output mode	controlling direction of motor
Digital pin P0.1	Digital output mode	controlling break of motor

Data Acquisition Toolbox enables you to customize your acquisitions, access the built-in features of hardware devices, and incorporate the analysis and visualization features of MATLAB and related toolboxes into your design. You can analyze or visualize your data, save it for post-processing, and make iterative updates to your test setup based on your analysis results. Data Acquisition Toolbox allows you to use MATLAB as a single, integrated environment to support the entire data acquisition, data analysis, and application development process.

Data Acquisition Toolbox also supports Simulink with blocks that enable you to incorporate live data or hardware configuration directly into Simulink models [33]. You can then verify and validate your model against live, measured data as part of the system development process.

The measured voltage, current and speed of dc motor is plotted using MATLAB but there are some ripple noise due to switching of MOSFET and EMI effects, so we design a digital average filter which it helps to reduce the effects of these noise.

4.7 Parameters Identification

This method is built in Simulink Parameter Estimation software in MATLAB for the parameters identification of the dc motor. Simulink Parameter Estimation software is a simulink-based product for estimating and calibrating model parameters from experimental data. This product supports the following types of estimation:

- Transient Estimation : Estimate parameters by comparing model output to the experimental data for a given input.

- Initial Condition Estimation : Estimate the initial conditions of states using experimental data.
- Adaptive Lookup Tables : Estimate the table values at the prescribed break-points by using measurements from the physical system.

Simulink Parameter Estimation software provides the tools used to:

1. Set up the problem.
2. Specify which model parameters to estimate.
3. Import and prepare the experimental data for parameter estimation (or pre-process).
4. View the estimation progress.
5. Validate the estimation results based on plots of measured versus. simulated data and residuals.

How Simulink Parameter Estimation Software Works

Simulink Parameter Estimation software compares empirical data with data generated by a Simulink model. Using optimization techniques, the software estimates the parameter and (optionally) initial conditions of states such that a user-selected cost function is minimized. The cost function typically calculates a least-square error between the empirical and model data signals.

After you build a Simulink model, follow these steps to configure and run a parameter estimation:

1. Select Tools → Parameter Estimation in your Simulink model window.
This opens the Control and Estimation Tools Manager, creates a new project, and adds an Estimation node to the workspace directory tree.
2. Import the input and output data set for estimating parameters of your Simulink model.
3. Select the parameters and initial conditions you want to estimate.
4. Configure the estimation itself, including cost functions and data views.
5. Run the estimation.
6. Check the results by examining either the cost-function values, plots, or parameter values.

Description of the DC Motor System

We developed the model of the DC motor using simulink as discussed in chapter 2. The model consists of input port for acquiring input signal of armature voltage and the output port for acquiring the output signals of armature current and speed of motor's rotor. The model is shown in Figure (4.8)

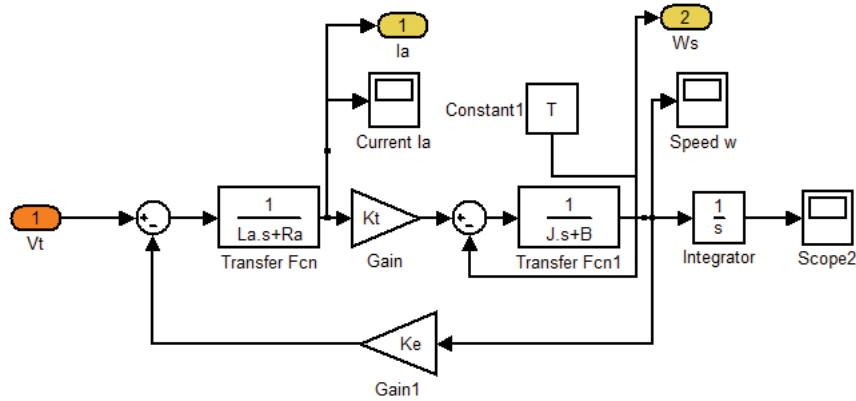


Figure 4.8: Block diagram of DC motor

While manufacturers may provide values for some of these quantities, they are only estimates. We want to estimate these parameters as precisely as possible for our model to ascertain whether it is an accurate representation of the actual DC motor system.

When we apply a step voltage to the motor input, the motor shaft turns in response. However, if the model parameters do not match those of the physical system, the model response will not match that of the actual system, either. Figure 4.10 shows the current response of our model using the initial parameter values in the model with the armature input voltage shown in Figure 4.9. It is obvious that we need to estimate our parameters. This is where Simulink Parameter Estimation plays a pivotal role.

Estimating Parameters of the DC Motor Model

parameter estimation process consists of a number of well-defined steps:

- Collect test data from your system (experiment).
- Specify the parameters to estimate (including initial guesses, parameter bounds, etc.).

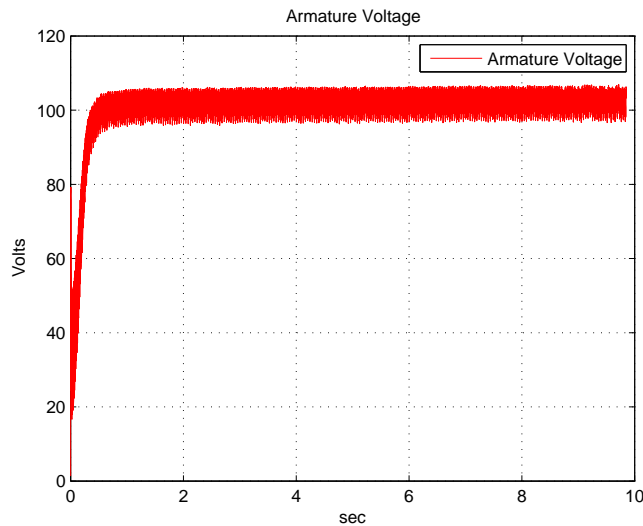


Figure 4.9: Armature voltage

- Configure our estimation and run a suitable estimation algorithm.
- Validate the results against other test data sets and repeat above steps if necessary.

Simulink Parameter Estimation provides a graphical user interface (GUI) to help you follow these steps, neatly organize your estimation project, and save it for future work. You can launch this product from the Tools menu of your Simulink model. The tool automatically creates a default estimation project for you:

1. **Importing Experimental Data:** The first step in our estimation project is to import the experimental data that we have collected from the actual DC motor system. We imported two data sets corresponding to various experimental conditions.

Simulink Parameter Estimation provides preprocessing tools such as noise filtering, detrending, splitting, and so forth to prepare data sets for estimation and/or validation. You can import experimental data sets from various sources including MATLAB variables, MAT files, Excel files, or comma-separated-value files. Once you import the data, you can also plot them to confirm that you have the right data sets in your estimation project.

2. **Selecting Parameters for Estimation:** Next, we select the model parameters whose value we want to estimate. The tool automatically recognizes all the parameters that we use in the model and lets us select a subset of them for estimation. Once we select the estimation parameters, we can set initial guesses for the parameter values, as well as the minimum and maximum bounds on these values, as shown below.

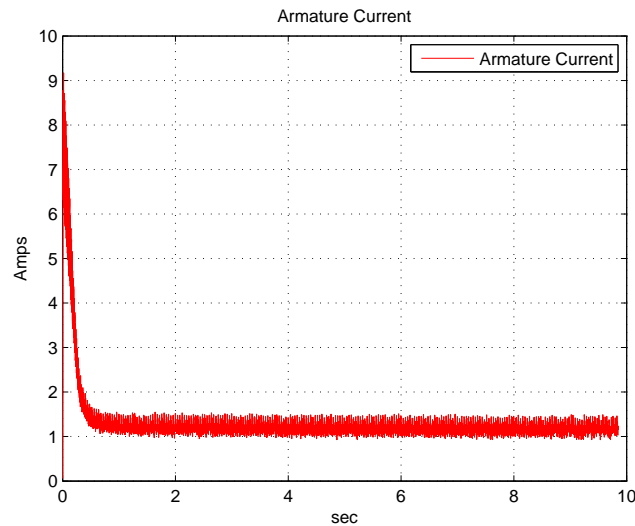


Figure 4.10: Armature current response

Simulink Parameter Estimation lets you estimate some or all of these parameters in a manner that best suits your application. For our DC motor, we will estimate all six parameters of the motor model: B , J , K_e , K_t , L_a , and R_a (these parameters are summarized in Table 2.1.

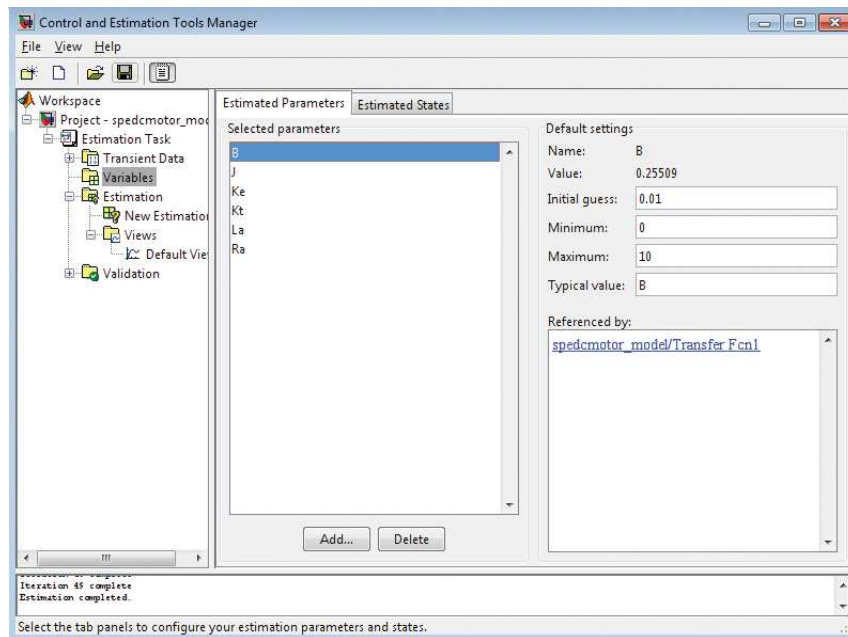


Figure 4.11: Selecting Parameters for Estimation

3. Defining an Estimation: Once we import our data and select the parameters to estimate, the next step is to create an estimation node in the GUI tree and configure various options. Each estimation uses your experimental data

sets to estimate the parameter that we have selected. However, we have the option of using only some of the data sets for estimation while keeping the others for validation. You can also select to estimate only a subset of the selected parameters.

Simulink Parameter Estimation also provides various state-of-the-art estimation algorithms. The most common selections include the nonlinear least-squares and Nelder-Mead optimization methods.

In our thesis we used the nonlinear least-square method and pattern search algorithm. Because it gives us a suitable results may be identical to the actual response and to compare them.

4. Optimization Options: Specify termination options in the Optimization options area is a vital role in identification results. These options are shown in Figure 4.12.

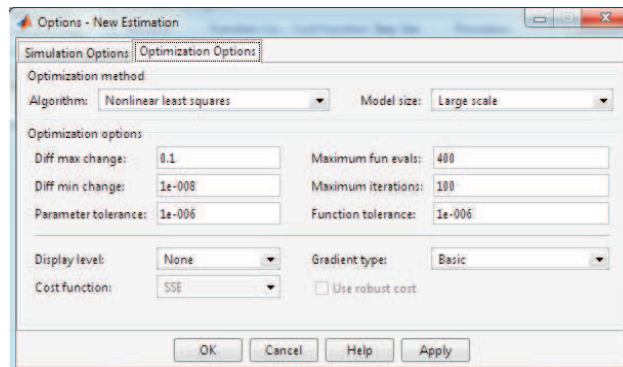


Figure 4.12: Optimization options

These options define when the optimization terminates, these options are:

- Diff max change: The maximum allowable change in variables for finite-difference derivatives, we use the default value (0.1).
- Diff min change: The minimum allowable change in variables for finite-difference derivatives, we use the default value (1e-8).
- Parameter tolerance: Optimization terminates when successive parameter values change by less than this number, we use the default value (1e-6).
- Maximum fun evals: The maximum number of cost function evaluations allowed. The optimization terminates when the number of function evaluations exceeds this value, we use the default value (400).
- Maximum iterations: The maximum number of iterations allowed. The optimization terminates when the number of iterations exceeds this value, we use the default value (100).

- Function tolerance: The optimization terminates when successive function values are less than this value, we use the default value (1e-4).

By varying these parameters, you can force the optimization to continue searching for a solution or to continue searching for a more accurate solution.

5. Running the Estimation: Once we finish configuring our estimation, we can run it and see how Simulink Parameter Estimation adjusts our model parameter values in order to match the model's response with our experimental data sets. Figure 4.13 shows the GUI figure while the estimation is running.

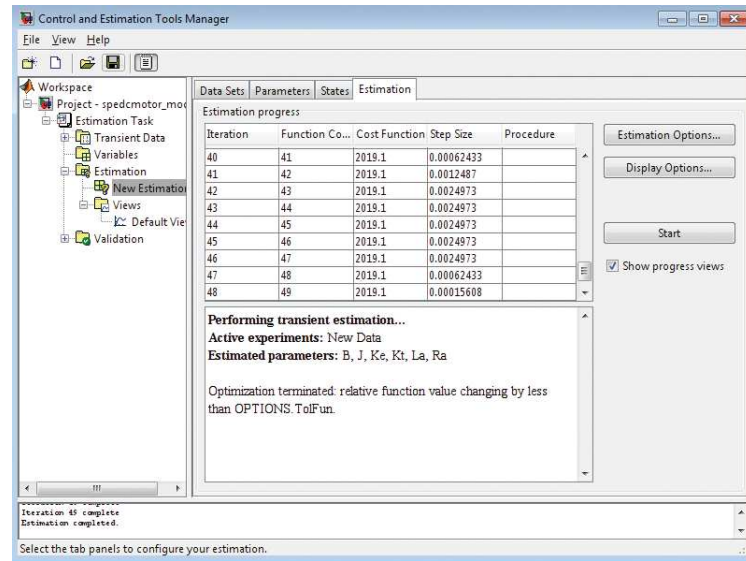


Figure 4.13: Estimation process tab

The result for the two different methods as following :

- (a) **Nonlinear least square method :**

As the parameter values improve, the simulation curves gets closer to the experimental data curves. Figure 4.14 and Figure 4.15 shows a plot of current response and speed response respectively after 50 iterations. It is clear from these figures that simulation results based on identified parameters are consistent with actual results.

The trajectories of estimated parameters versus number of iterations are illustrated in Figure 4.16.

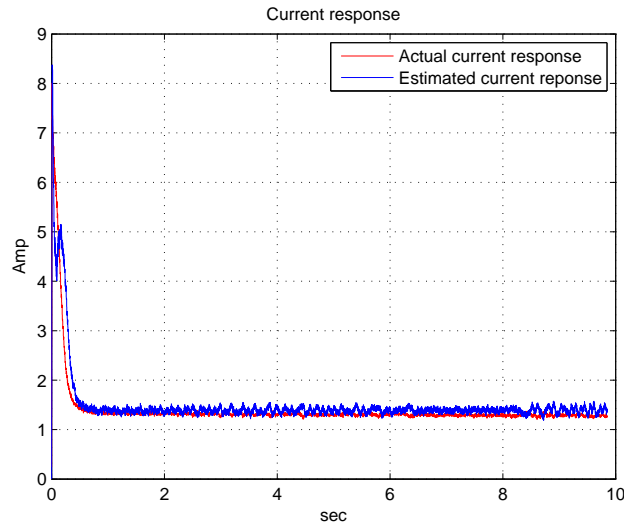


Figure 4.14: Current response for the actual and estimated models

Comparing the response of the system before and after the estimation process clearly shows that the estimation successfully identified the model parameters and the simulated response accurately matches the experimental data. Also the speed response for the actual model of the DC motor is nearly identical to the estimated speed response as shown below.

Once the estimation is complete, we can inspect the estimated parameter values. The Value column below displays the new parameter values at the end of the estimation as illustrated in Figure 4.17.

When the algorithm is begin then the MATLAB workspace is begin to show us the iteration process through identification. Below is a sample of the estimation process. Table 4.3 summarized the estimation process.

A summary of the parameters estimation results are given in the table below.

It's clear from the table that nonlinear-least squares algorithm is powerful and accurate because there are a very good match between the actual response and the estimated response for both current response and speed response for repeating the algorithm many times.

(b) **Pattern search algorithm :**

In this method we need to make so modifications on Matlab code in order to apply this method. We change the following line

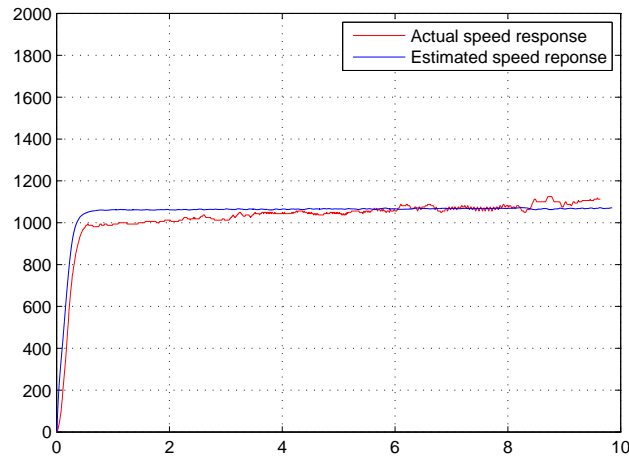


Figure 4.15: Speed response for the actual and estimated models

```
hEst.OptimOptions.Algorithm = 'lsqnonlin';
```

with this one

```
hEst.OptimOptions.Algorithm= 'Pattern';
```

When the algorithm is begin then the MATLAB workspace is begin to show us the iteration process through identification. Below is a sample of the estimation process. Table 4.5 summarized the estimation process.

After the iteration is finished we have a summary of the parameters estimation results which are given in Table 4.6.

It can be seen from Table 4.5 that we achieve the same results when we apply the method three times. This conclude that this method give us the same results for the same acquiring signals.

Figure 4.18 and Figure 4.19 shows a plot of current response and speed response respectively. It is clear from these figures that simulation results based on identified parameters are consistent with actual results.

The transfer function of the dc motor model is needed for studying the state of our system and applying the controlling techniques on it. The transfer function for current to voltage and speed to voltage is derived by using a code in MATLAB. These transfer functions are described as below from the nonlinear least square method:

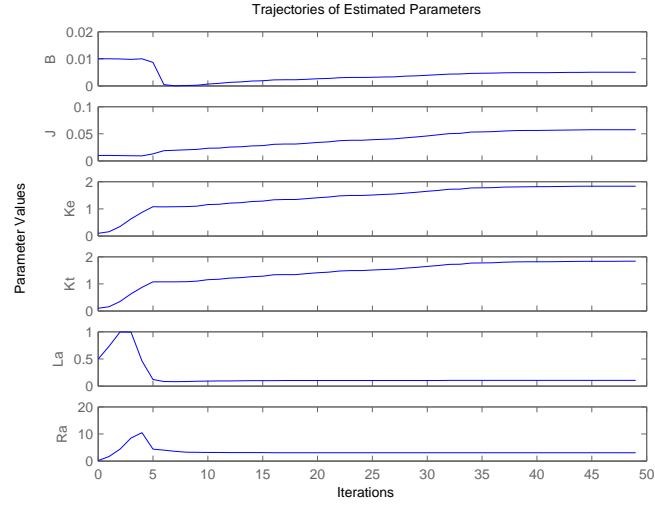


Figure 4.16: Trajectories of estimated parameters

$$\frac{I_a(s)}{V(s)} = \frac{s + 1.039}{0.01324s^2 + 3.278s + 80.75} \quad (4.2)$$

$$\frac{W(s)}{V(s)} = \frac{65.04}{0.01324s^2 + 3.278s + 80.75} \quad (4.3)$$

4.7.1 GUI Interface

GUI interface is implemented to automate the identification process. We build it using matlab guide as illustrated in Fig. 4.20. It's featured with all functions needed for the identification process. These functions are:

1. Acquiring data: The function of this button is to apply a constant voltage about 110V to the motor and acquires the armature voltage, armature current and motor speed signals.
2. Parameters identification: The function of this button is to apply the suitable algorithm (nonlinear least square) which is described above in simulink parameter estimation but as matlab code.
3. Validation: The function of this button is to acquire a new data for validation then comparing of estimated response to the actual repousse is plotted to show how accuracy and efficiency of the estimated parameters. The main parameters need for identification process is also displayed in the figure to parameters name.

The GUI interface is built for user friendly and to get all parameters at a minimum time. It providing control functions for speed controlling, direction controlling

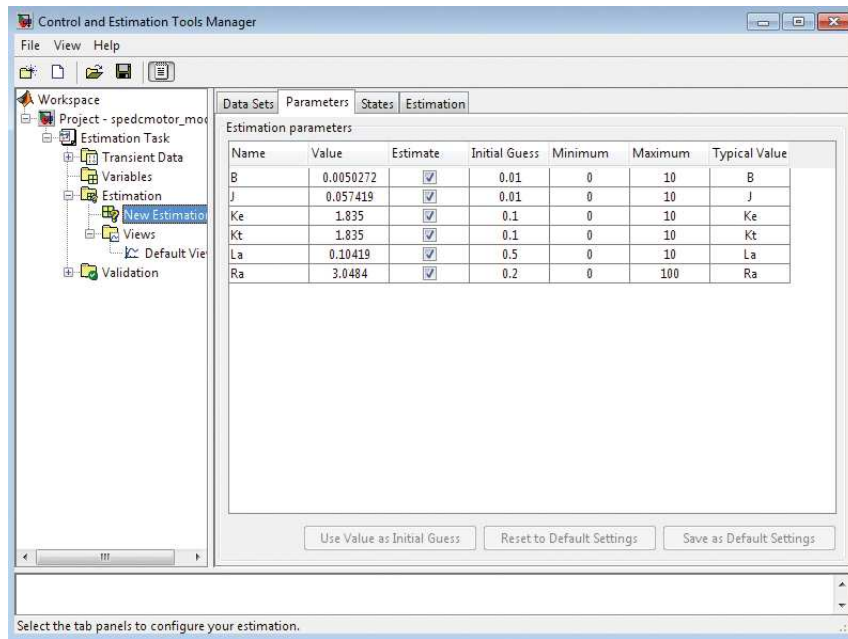


Figure 4.17: Parameters estimation results

and stopping the motion of motor. An indication in the figure help us to know the direction of motor is running.

Table 4.3: Summary of parameter estimation process(NLS method)

Iter	Func-count	f(x)	Norm of step	First-order optimality	CG-iterations
0	1	3043.05		3.59e+6	
1	2	1377.92	0.0887085	3.36e+4	0
2	3	1195.33	0.456647	2.38e+5	0
3	4	1175.86	0.0234237	1.32e+4	0
4	5	1169.95	0.0888268	4.85e+4	0
5	6	1168.02	0.0105985	4.71e+3	0
6	7	1167.34	0.0289562	1.62e+4	0
7	8	1167.1	0.00419732	1.79e+3	0
...
...
26	27	1166.96	5.82921e-6	28.3	0
27	28	1166.96	5.82921e-6	12.6	0
28	29	1166.96	1.4573e-6	1.25	0
29	30	1166.96	1.4573e-6	33.3	0

Table 4.4: Summary of parameter estimation results(NLS method)

Parameters	1 st case:	2 nd case:	3 rd case:
$B(Nms)$	0.019	0.019	0.019
$J(kg.m^2)$	0.01829	0.01829	0.01829
$K_e(V/rad/sec)$	1.1895	1.1895	1.1895
$K_t(N.m/A)$	1.1895	1.1895	1.1895
$L_a(H)$	0.013242	0.013242	0.013242
$R_a(\Omega)$	3.2645	3.2645	3.2645

Table 4.5: Summary of parameter estimation process(Pattern method)

Iter	f-count	f(x)	MeshSize	Method
0	1	3043.05	1	
1	6	3043.05	0.5	Refine Mesh
2	12	3043.05	0.25	Refine Mesh
3	18	3043.05	0.125	Refine Mesh
4	26	3043.05	0.0625	Refine Mesh
5	33	2867.84	0.125	Successful Poll
6	40	2867.84	0.0625	Refine Mesh
...
...
48	387	1182.95	0.0002441	Refine Mesh
49	399	1182.95	0.0001221	Refine Mesh
50	400	1182.95	6.104e-005	Refine Mesh

Table 4.6: Summary of parameter estimation results(Pattern method)

Parameters	1 st case:	2 nd case:	3 rd case:
$B(Nms)$	0.011	0.011	0.011
$J(kg.m^2)$	0.01	0.01	0.01
$K_e(V/rad/sec)$	0.54531	0.54531	1.8674
$K_t(N.m/A)$	1.4594	1.4594	4594
$L_a(H)$	0.013906	0.013906	0.013906
$R_a(\Omega)$	3.2313	3.2313	3.2313

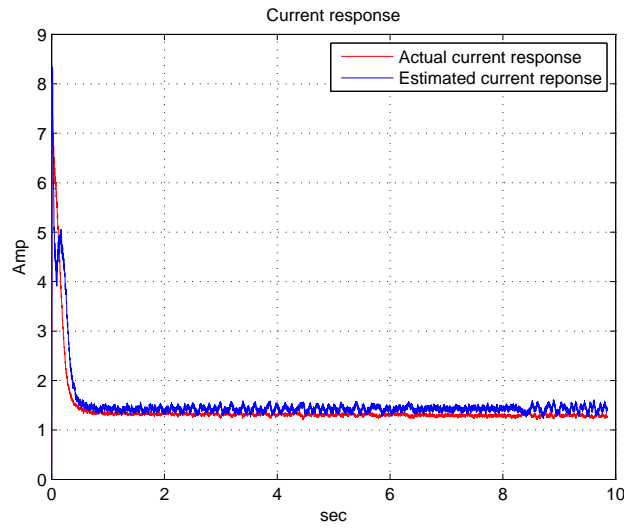


Figure 4.18: Current response for the actual and estimated models



Figure 4.19: Speed response for the actual and estimated models

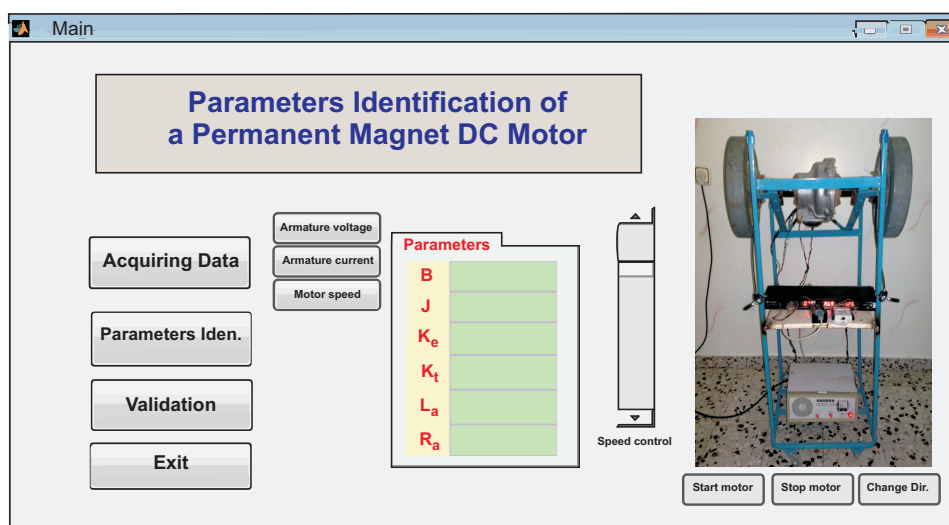


Figure 4.20: GUI interface

Chapter 5

Conclusions and Future Work

5.1 Conclusions

This thesis has investigated the issues involved in applying different methods in parameters identification of DC motor models. These issues have been considered both theoretically and experimentally. The experimental work was performed on a DC motor with an optical encoder as a position feedback device. All necessary drivers and sensor circuits are implemented to constitute a compact experimental platform. Computer interface is based on a low-cost data acquisition module from National Instruments. A MATLAB program, based on the Simulink Parameter Estimation toolbox, is created to call the nonlinear least-square algorithm or the pattern search method and estimate the motor parameters. The estimated parameters are used to generate simulation curves for current and speed step responses. These curves are found to be in agreement with actual curves within the precision level of our experiment.

Finally, I suggest building a number of this experimental system that could be used as a good kit in colleges and universities in control and power courses.

5.2 Future Work

In this thesis we used a data acquisition module from national instruments to acquire the required experimental signals. Interfacing and identification process is built and done using the MATLAB in PC. Good future work may be if we can apply the acquiring and the identification process using microcontrollers or embedded controllers such as FPGA or DSP controllers, this will serve the cost of DAQ and PC used in pervious process. Also if we can design a portable and small size system in a nice finishing will be powerful design.

Finally an important future work is tuning the system to be compatible with any voltage level of the motor will be identified.

5.3 Summary of Contributions

This research has contributed to the field of system modeling and system identification. This is a relatively new area that has growing importance in control problems. Since accurate mathematical models and their parameters are essential when designing controllers because they allow the designer to predict the closed loop behavior of the plant. Errors in parameter values can lead to poor control and instability. Therefore, accuracy and adequacy of parameters identification are too major modeling issues that always have to be dealt with.

In summary, the work described in this thesis made the following contributions: A comparative study for methods used in parameters identification of DC motor and try to apply these method practically on our experimental system. The summary of these methods are :

- Pasek's method: This method of motor testing is based on the starting current, peak current, peak current time of occurrence, current at twice the time of the peak current occurrence, steady state current and steady state velocity. Pasek's method measures a few specific points on the current time response curve making it very sensitive to current commutation noise, which renders the method inaccurate for low-cost DC motors. Pasek's method is in fact only useable on special high quality servo motors. Because of the noise sensitivity problem, this method can't be applied to our experiment.
- Frequency response method: determines the parameters of high performance dc motors by treating a second order motor model as an electrical impedance (RLC circuit), and by tuning the values of RLC circuit elements to match the response of dc motor and by some relations the parameters of dc motor are calculated. This method uses an ac signal with specific frequency about 1kHz. Unfortunately, this method is not suitable to the power drivers used in our experimental platform. Also, some instrumentation devices needed for measurement such as spectrum analyzer is another problem. Sensitivity to noise is another reason that discouraged us to utilized this method.
- The pseudo inverse technique : is a method capable of identifying an equivalent discrete transfer function of simple systems like motor, thermal system, mass damper systems etc with no zeros (only poles). This method assumes a single input single output system and unable to find the dc motor parameters and doesn't fit our objective such our object is to find the motor parameters not the coefficients of the transfer function. Also the transfer function is a complex on and it's not easy to solve and find the parameters which need another study.
- Nonlinear least square method : is a general approach in identification seeks to define an objective function that would reach its minimum. The physical system to be investigated is described in terms of parameters, and then, the objective function is minimized with respect to the parameters by an iterative procedure. At the minimum of the objective function, the values of the

parameters describe the real structure of the physical system. This method is utilized in MATLAB and we success to find the motor parameters as our objective[41]. The feature of this method is that you can apply it to any type of system you need to identify it after you determine which parameters are needed to identify.

- Pattern search method : Pattern search is an attractive alternative to the genetic algorithm, as it is often computationally less expensive and can minimize the same types of functions to find the optimal parameters.

Table 5.1 show us the results for the different methods were used which they are the nonlinear least square method and pattern search method. It can be seen that the values of these methods is approximately the same with minimum error.

Table 5.1: Parameter estimation results

Parameters	Nonlinear least square	Pattern search
$B(Nms)$	0.011	0.019
$J(kg.m^2)$	0.01	0.01829
$K_e(V/rad/sec)$	0.54531	1.1895
$K_t(N.m/A)$	1.4594	1.1895
$L_a(H)$	0.013906	0.013242
$R_a(\Omega)$	3.2313	3.2645

Finally, the nonlinear least square method and pattern search method are the best methods among the methods are studied because of the performance and the accurate in results and can be automated in MATLAB.

We also used a low cost data acquisition module for the acquiring process. We succeed in installing and interfacing NI DAQ with MATLAB after we face a some problems in interfacing. A GUI also is automated in MATLAB for user friendly, low time and flexibility in identification. A non-linear least square algorithm is used and applied to identification process which got a good results of identification.

These contributions extend the understanding of the DC motor response in assisting in control fields and designing robust controllers.

Bibliography

- [1] N. Sinha, C. Diczko and B. Szabados, “ Modeling of DC motors for control applications ”. *IEEE Trans. Industrial Electronics And Control Instrumentation*, vol. IECI-21, pp. 84-88, August 1974.
- [2] W. Lord and J. Hwang, “ DC servomotors modeling and parameter determination ”. *IEEE Trans. Industrial Applications*, vol. IA-3, pp. 234-243, May/June 1973.
- [3] W. Lord and J. Hwang, “ DC motor model parameters ”. *IEEE Trans. on Industrial Electronics and Control Instrumentation*, vol. IECI-23, No.3, August 1975.
- [4] R. Schulz, “A frequency for determining the parameters of high performance DC motors”. *IEEE Trans. on Industrial Electronics*, vol. IE-30, No.1, February 1983.
- [5] A. Rajkumar and R. Somanahtham, “ Determination of DC motor transients using a microcomputer ”. *IEEE Trans, on Instrumentation and Measurement*, vol. 37, No.4, Dec. 1988.
- [6] S. Weerasooriya and M. El-Sharkawi, “ Identification and control of a DC motor using back-propagation neural networks ”. *IEEE Trans. on Energy Conversion*, vol. 6, No.4, December 1991.
- [7] Y. G. Jung, K. Cho, Y. Lim, J. Park and Y. H. Change, “ Time domain identification of brushless DC motor parameters ”. *IEEE Trans. on Industrial Electronics*, vol. 2, 593-597, February 1992.
- [8] D. M. Gillard and K. E. Bollinger, “ Neural network identification of power system transfer functions ”. *IEEE Transactions on Energy Conversion*, Vol. 11, No. 1, March 1996.
- [9] D. M. Gillard and K. E. Bollinger, “ Online identification and control of a DC motor using learning adaptation of neural networks ”. *IEEE Trans. on Industrial Applications*, VOL. 36, NO. 3, MAY/JUNE 2000.
- [10] Hanamoto, T. Tanaka, Y. Karube, I. Mochizuk and T. Xu, Z., “ A parameter estimation method using block pulse functions for DC servo motor

- systems". *Industrial Electronics, Control, Instrumentation, and Automation, 1992. Power Electronics and Motion Control*, proceedings of the 1992 International Conference on vol., no., pp.1288-1293 vol.3, 9-13 Nov 1992.
- [11] S. Saab and R. Abi Kaed-Bey, "Parameter identification of a DC motor: an experimental approach". *IEEE Trans. on Electronics, Circuits and Systems*, vol. 2, 981-984, 2001.
 - [12] A. Dupuis, M. Ghribi and A. Kaddouri, "Multiobjective genetic estimation of DC motor parameters and load torque". *IEEE International Conference on Industrial Technology*, vol. 3, 1511-1514, Dec. 2004.
 - [13] Radojka Krneta, Sanja Antic and Danilo Stojanovic, "Recursive least squares method in parameters identification of DC motors models". *ELEC. ENERG*, vol. 18, no. 3, 467-478, December 2005.
 - [14] R. Garrido and R. Miranda, "Closed loop identification of a DC servomechanism". *Power Electronics Congress, 10th IEEE*, vol., no., pp.1-5, 16-18 Oct. 2006.
 - [15] W. P. Aung, "Analysis on modeling and simulink of DC motor and its driving system used for wheeled mobile robot". *International Journal of Computer, Information, and Systems Science, and Engineering*, Volume 2 Number 1, Nov. 2007.
 - [16] M. Hadeif and R. Mekideche, "Parameter identification of a separately excited DC motor via inverse problem methodology". *Turk J Elec Eng and Comp Sci*, vol. 17, No.2, 2009.
 - [17] Adrian V. Bos, Parameter Estimation for Scientists and Engineers, First Edition, Wiley, Inc., 2007.
 - [18] Rik Pintelon and Johan Schoukens, System identification: a frequency domain approach, First Edition, IEEE Press, Inc., 2001.
 - [19] Wikipedia, The free encyclopedia http://en.wikipedia.org/wiki/Electric_motor
 - [20] Ogata K., Modern Control Engineering, 2nd ed., Prentice Hall, NJ, 1990.
 - [21] Kuo B.C., Automatic Control Systems, 6th ed., Prentice Hall, NJ, 1991.
 - [22] DC Servomotors Tutorial, Callon Technology, URL: http://www.callantechnology.com/callan_technology_resources/
 - [23] Genetic Algorithm and Direct Search Toolbox, Mathworks, URL: <http://www.mathworks.com/access/helpdesk/help/toolbox/gads/f7805.html>
 - [24] Paul Scherz, Practical Electronics For Inventors, First Edition, McGraw-Hill, Inc., 2000.

- [25] Selecting the right MOSFET driver, Hearst Electronic Products, URL:http://www2.electronicproducts.com/Selecting_the_right_MOSFET_driver-article-microchip-mar2007-html.aspx.
- [26] Motor Control Sensor Feedback Circuits, Microchip, Inc. URL:<http://ww1.microchip.com/downloads/en/AppNotes/00894a.pdf>.
- [27] Kuo B., Digital Control Systems, Holt Saunders, Tokyo, 1980.
- [28] Ogata K., Discrete-Time Control Systems, Prentice Hall, N.J, 1987.
- [29] Astrom K.J., Wittenmark B., Computer Controlled Systems - Theory and Design. 3rd edition, Prentice-Hall, Englewood Cliffs. N.J., 1997.
- [30] Franklin G.F., Powell J.D., Workman M.L., Digital Control of Dynamic Systems, 3rd edition, Addison Wesley, Reading, Mass, 1998.
- [31] Thomas L. Floyd, Electronics Fundamentals: Circuits, Devices and Applications, Prentice Hall, 7th Ed., 2006.
- [32] NI USB-6008, National Instruments Corporation, URL:<http://sine.ni.com/nips/cds/view/p/lang/en/nid/14604>.
- [33] Data Acquisition Toolbox, The MathWorks, Inc. URL:<http://www.mathworks.com/products/daq/>.
- [34] Franklin G., Powell J.D., Feedback Control of Dynamic Systems, Addison Wesley, Reading, Mass, 1986.
- [35] Shinskey F.G., Process Control Systems, McGraw-Hill, N.Y., 1979.
- [36] Wirk G.S., Digital Computer Systems, MacMillan, London, 1991.
- [37] Loan D. Landau, Gianluca Zito, Digital Control Systems - design, identification and implementation, Springer Verlag, London, 2006.
- [38] Phillips C.L., Nagle H.T., Digital Control Systems Analysis and Design, 3rd edition. Prentice Hall, N.J., 1995.
- [39] Richard C. Dorf, Robert H. Bishop, Modern Control Systems, Addison Wesley, 2002.
- [40] Gene F. Franklin, J. David Powell, Abbas Emami-Naeini, Feedback Control Of Dynamic Systems, 5th ed., New Jersey, Prentice Hall, 2006.
- [41] M. Salah and M. Abdelati, " Parameters identification of a permanent magnet DC motor ". *Modelling, Identification and Control MIC*, 675-085 , February 2010.

Appendix A

MALTLAB Codes

A.1 Acquiring Signals

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% File: acq_but.m
% Usage: Interfacing with NI-DAQ6008 to send as analog output signal to run
% DC motor for acquiring analog signals for 10 sec's of time.
% Author: Eng. Mohammed S. Salah
% http://www.ucas.edu.ps
% Department of electrical engineering
% The Islamic University of Gaza
% May 2009
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
global arm_volt_avg
global sen_cur_avg
global time

clc
scalevolt=132.1/5.1; scalecurrent=22.4; Vt=110; Vpwm=Vt/25;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%delete open daqs
openDAQ = daqfind;
for i = 1:length(openDAQ)
    delete(openDAQ(i));
end

% open the daq
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%configure analog inputs
ai = analoginput('Nidaq','Dev1');
```

```

chan = addchannel(ai,0:2);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% configure analog outputs
dio = digitalio('nidaq','Dev1');
addline(dio,0,0,'out');

ao = analogoutput('nidaq','Dev1');
ch = addchannel(ao, 0);
putsample(ao,0); %%% stop motor at default

warning('off', 'all')
button = questdlg('Turn on the power and press OK button to begin acquiring',
'Acquiring data','Ok','Cancel','No');

switch button
case 'Ok',
    set(handles.ind1,'String','Begin')
    set(handles.ind1,'Visible','on')

    Fs=3300 %%% sampling rate 1000 samples/sec
    set(ai,'samplespertrigger',Inf);% set sample rate

    set(ai,'samplerate',Fs); %Fastest sampling by USB-6008

    set(ai,'loggingmode','disk'); % set to log to file

    %%%%%%%%%

    %%%%%
    start(ai)% start data collection

    putsample(ao,Vpwm); %%% start motor

    message = 'begin data aquisition for 10 secs'% report when finished

    pause(10)

    stop(ai)% stop data collection

    putsample(ao,0); %%% stop motor at default
    message = 'data aquisition finished'% report when finished

    %%%%%%%%%%%%%%%

clear message

```

```

% get data from log file

file = get(ai, 'logfile');

[data,time] = daqread(file);

arm_volt=data(:,1);
sen_cur=data(:,2);
opti_enc=data(:,3);

arm_volt=arm_volt.*scalevolt;
sen_cur=sen_cur.*scalecurrent;

fs=Fs;
Duration=.01; % seconds .01
totalrpm=[];
totaltt=[];
% Start the acquisition.

% [orgdata time] = getdata(ai,ai.SamplesAcquired);
orgdata=opti_enc;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Set the threshold to 3.5 V.
threshold = 2.4;

numsamples=length(orgdata);
numofloops=round(numsamples/(fs*Duration))-1;

for k=0:numofloops-1

    data=orgdata(1+k*fs*Duration:fs*Duration*(1+k));

    % Create the offset data. Need to append a NaN to the final sample
since
    % both vectors need to have the same length.
    offsetData = [data(2:end); NaN];

    % Find the rising edge.
    risingEdge = find(data < threshold & offsetData > threshold);

    % Find the falling edge.
    fallingEdge = find(data > threshold & offsetData < threshold);

    if length(time(risingEdge))==0

```



```

        numPulses=0;
        %disp('no pulses')
    else
        %plot(time(risingEdge), threshold, 'rx');

        numPulses = length(risingEdge);
        %disp(['numPulse = ',num2str(numPulses)])

    end
    %%%%%%%%%%% Compute the speed rpm %%%%%%%%%%%
    rpm=(60/Duration)*numPulses/120*2.5; % 120 is num of slits of disk

    totalrpm=[totalrpm rpm];
    totaltt=[totaltt k*Duration];
end
%%%%%%%%%%

helpdlg('Acquiring data is finished, turn off motor','Acquiring data');

%%%%%%%%%% processing data for identification
%%%%%%%%%% average filter fro
%%%%%%%%%% armature voltage
M=30;
a=[1 zeros(1,length(M))];
b=(1/M)*[ones(1,M)];
arm_volt_avg=filter(b,a,arm_volt);

%%%%%%%%%% armature current
M=30;
a=[1 zeros(1,length(M))];
b=(1/M)*[ones(1,M)];
sen_cur_avg=filter(b,a,sen_cur);
%%% speed
M=20;
a=[1 zeros(1,length(M))];
b=(1/M)*[ones(1,M)];
totalrpm_avg=filter(b,a,totalrpm);

totalrpm_avg_sim=[0 totalrpm(21:end)];
totaltt_sim=totaltt(1:end-19);

M=20;
a=[1 zeros(1,length(M))];
b=(1/M)*[ones(1,M)];

```

```

totalrpm_avg_sim=filter(b,a,totalrpm_avg_sim);
%%%%%%%%%%
%%%%%%%%%% Voltage === current model

signals = struct('values',{totalrpm_avg_sim},'dimesions',{1},'label',"")
Ws = struct('time',{totaltt_sim},'signals',signals,'blockName',{'Motor2/To
Workspace1'})

signals = struct('values',{arm_volt_avg},'dimesions',{1},'label',"");
Vs = struct('time',{time},'signals',signals,'blockName',{'Motor2/To Workspace'});

signals = struct('values',{sen_cur_avg},'dimesions',{1},'label',"");
Is = struct('time',{time},'signals',signals,'blockName',{'Motor2/To Workspace1'});

save mydata.mat arm_volt arm_volt_avg sen_cur sen_cur_avg time to-
taltt_sim totalrpm_avg_sim Ws Vs Is

set(handles.pushbutton2,'Enable','on')
set(handles.pushbutton3,'Enable','on')

set(handles.pushbutton6,'Enable','on')
set(handles.pushbutton7,'Enable','on')
set(handles.pushbutton8,'Enable','on')

set(handles.ind1,'String','Ready')
set(handles.ind1,'Visible','on')
end
% collect data until a 'stop' is issued
warning('on', 'all')

```

A.2 Parameters Identification

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% File: parm_ident.m
% Usage: Applying parameters identification which we used the nonlinear
% least square method to find DC Motor parameters.
% Author: Eng. Mohammed S. Salah
% http://www.ucas.edu.ps
% Department of electrical engineering
% The Islamic University of Gaza
% June 2009
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
global arm_volt_avg
global sen_cur_avg
global time

clc
evalin('base','load mydata.mat');

Ws = evalin('base', 'Ws');
Vs = evalin('base', 'Vs');
Is = evalin('base', 'Is');

J=0.01;B=0.01;Ke=0.1;Kt=0.1;Ra=.2;La=0.5;T=0;
Fs=3300 %% sampling rate 1000 samples/sec
Ts=1/Fs;%0.00001;
Ia=[];
W=[];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% data for parameter identification

iodata=[Vs.signals.values Is.signals.values];

save mydata1.mat J B Ke Kt Ra La T Fs Ts;
evalin('base','load mydata1.mat');

model = 'spedcmotor_model';
open_system(model)
set( find_system(model, 'FindAll', 'on', 'BlockType', 'Scope'), 'Open', 'off' )

hExp = ParameterEstimator.TransientExperiment(model);
set(hExp.InputData(1), 'Data', iodata(:,1), 'Time', time);
set(hExp.OutputData(1), 'Data', iodata(:,2), 'Time', time);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
hPar(1) = ParameterEstimator.Parameter('B');
```

```

set(hPar(1), 'Minimum', 0, 'Maximum', 10, 'Estimated', true)

hPar(2) = ParameterEstimator.Parameter('J');
set(hPar(2), 'Minimum', 0, 'Maximum', 10, 'Estimated', true)

hPar(3) = ParameterEstimator.Parameter('Ke');
set(hPar(3), 'Minimum', 0, 'Maximum', 10, 'Estimated', true)

hPar(4) = ParameterEstimator.Parameter('Kt');
set(hPar(4), 'Minimum', 0, 'Maximum', 10, 'Estimated', true)

hPar(5) = ParameterEstimator.Parameter('La'); % Initial value: Zd = -80
set(hPar(5), 'Minimum', 0, 'Maximum', 10, 'Estimated', true)

hPar(6) = ParameterEstimator.Parameter('Ra');
set(hPar(6), 'Minimum', 0, 'Maximum', 100, 'Estimated', true)

%%%%%%%%%%%%%%
hEst = ParameterEstimator.Estimation(model, hPar, hExp);
%%%%%%%%%%%%%%

hEst.OptimOptions.Algorithm= 'lsqnonlin';
hEst.OptimOptions.GradientType= 'basic';
hEst.OptimOptions.Display= 'iter';

estimate(hEst)

find(hEst.Parameters, 'Estimated', true)

B = evalin('base', 'B');
B=0.0201202;
J = evalin('base', 'J');
Ke = evalin('base', 'Ke');
Kt = evalin('base', 'Kt');
La = evalin('base', 'La');
Ra = evalin('base', 'Ra');

set(handles.B_value,'String',num2str(B))
set(handles.J_value,'String',num2str(J))
set(handles.Ke_value,'String',num2str(Ke))
set(handles.Kt_value,'String',num2str(Kt))
set(handles.La_value,'String',num2str(La))
set(handles.Ra_value,'String',num2str(Ra))

```

A.3 Validation of Identification

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% File: validation.m
% Usage: Get new date from DC motor using NI-DAQ6008 and make validation
% between acutual response and estimated response.
% Author: Eng. Mohammed S. Salah
% http://www.ucas.edu.ps
% Department of electrical engineering
% The Islamic University of Gaza
% July 2009
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global arm_volt_avg
global sen_cur_avg
global time

clc
evalin('base','load mydata.mat');

Ws = evalin('base', 'Ws');
Vs = evalin('base', 'Vs');
Is = evalin('base', 'Is');

B= eval(get(handles.B_value,'String'));
J= eval(get(handles.J_value,'String'));
Ke= eval(get(handles.Ke_value,'String'));
Kt= eval(get(handles.Kt_value,'String'));
La= eval(get(handles.La_value,'String'));
Ra= eval(get(handles.Ra_value,'String'));
sen_cur_avg = evalin('base', 'sen_cur_avg');

totalrpm_avg_sim = evalin('base', 'totalrpm_avg_sim');
totaltt_sim = evalin('base', 'totaltt_sim');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% current transfer function
numI=[1 B/J];
denI=[La (Ra+La*B/J) (Ra*B/J+Ke*Kt/J)];
%%disp('current tranfer function')
Isys=tf(numI,denI);
[ia1,t1]=lsim(Isys,Vs.signals.values,Vs.time);

sen_cur_mean=mean(sen_cur_avg(5000:end))
current_mean_estimated=mean(ia1(end-20:end))
err_diff=sen_cur_mean-current_mean_estimated;
while err_diff>0.001
```

```

B=B+0.001;
numI=[1 B/J];
denI=[La (Ra+La*B/J) (Ra*B/J+Ke*Kt/J)];
%%disp('current tranfer function')
Isys=tf(numI,denI);
[ia1,t1]=lsim(Isys,Vs.signals.values,Vs.time);

current_mean_estimated=mean(ia1(end-20:end));
err_diff=sen_cur_mean-current_mean_estimated;
end
Isys=tf(numI,denI)
set(handles.B_value,'String',num2str(B))
figure
plot(Is.time,Is.signals.values,'r')

title('Current response')
ylabel('Amp')
xlabel('sec')
grid
hold on
plot(t1,ia1);

legend('Actual current response','Estimated current reponse')

%%%%%%%%%%%% speed transfer function
numW=[Kt/J];
denW=[La (Ra+La*B/J) (Ra*B/J+Ke*Kt/J)];
%%disp('speed tranfer function')
%subplot(2,1,2)
Wsys=tf(numW,denW)
[w2,t2]=lsim(Wsys,Vs.signals.values,Vs.time);
figure
plot(Ws.time,Ws.signals.values,'r')
hold on
Duration=.01; % seconds .01
totalrpm_mean=mean(totalrpm_avg_sim(5/Duration:9/Duration));

speed_mean_estimated=mean(w2(end-20:end));
correction=totalrpm_mean/speed_mean_estimated;
w2=correction*w2;
grid
plot(t2,w2);
legend('Actual speed response','Estimated speed reponse')
ylim([0 2000])

```

Appendix B

Schematics

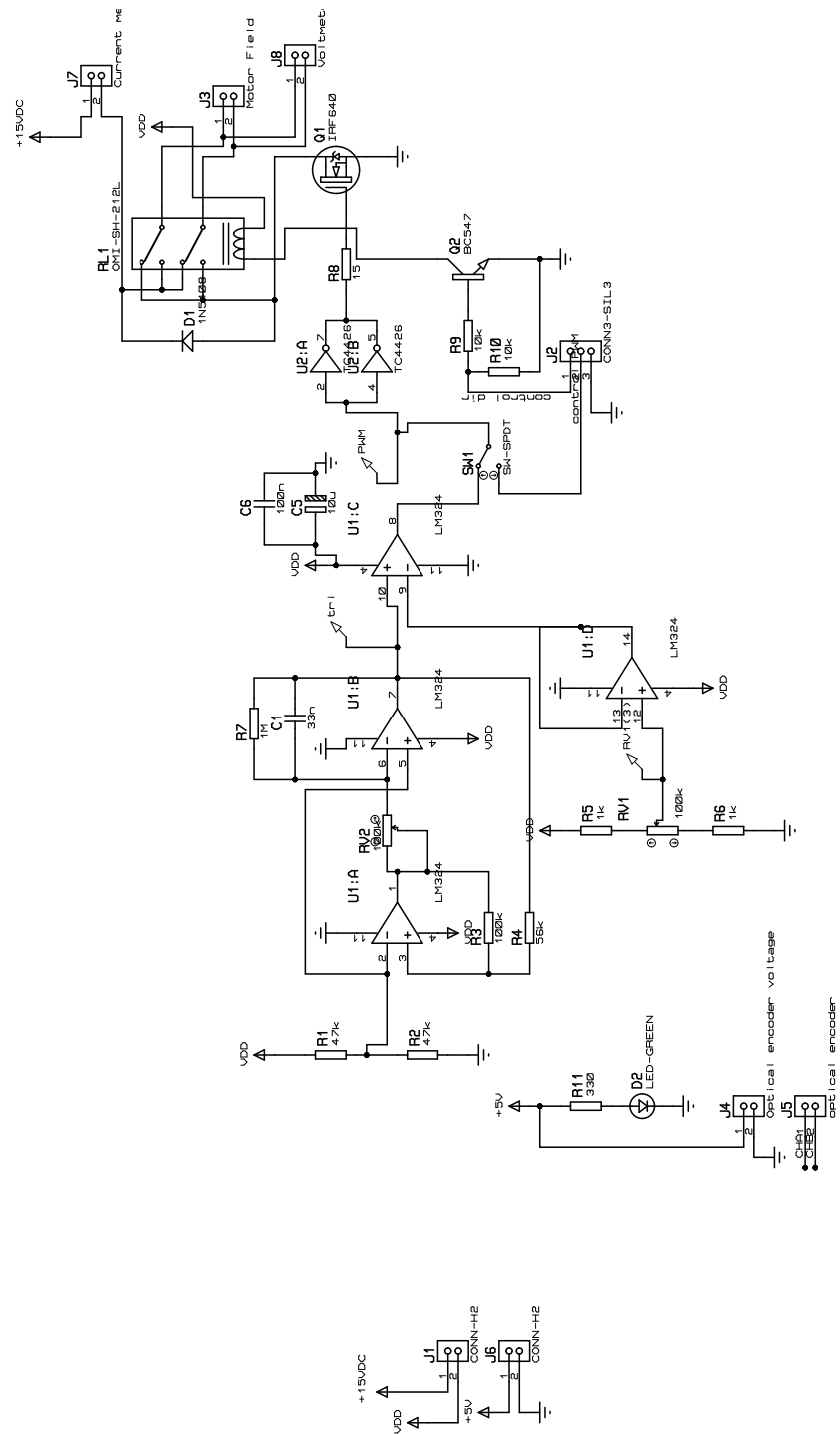


Figure B.1: Field driver unit schematics.

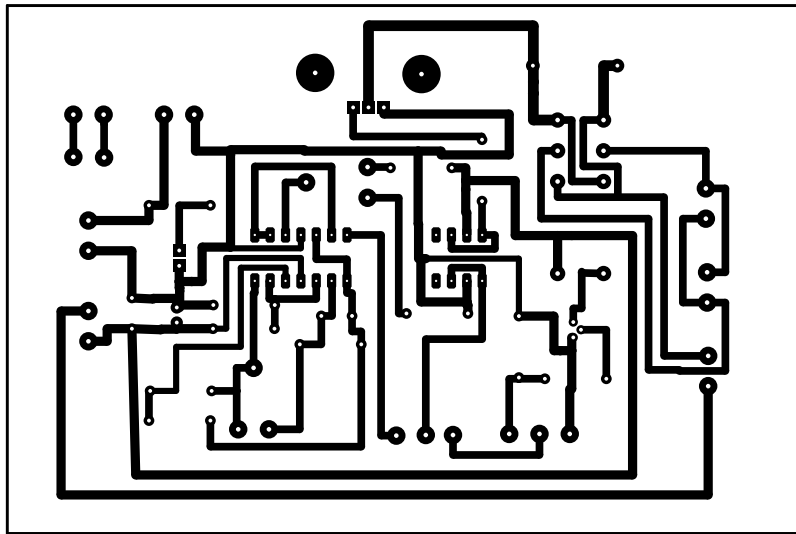


Figure B.2: Field driver unit layout.

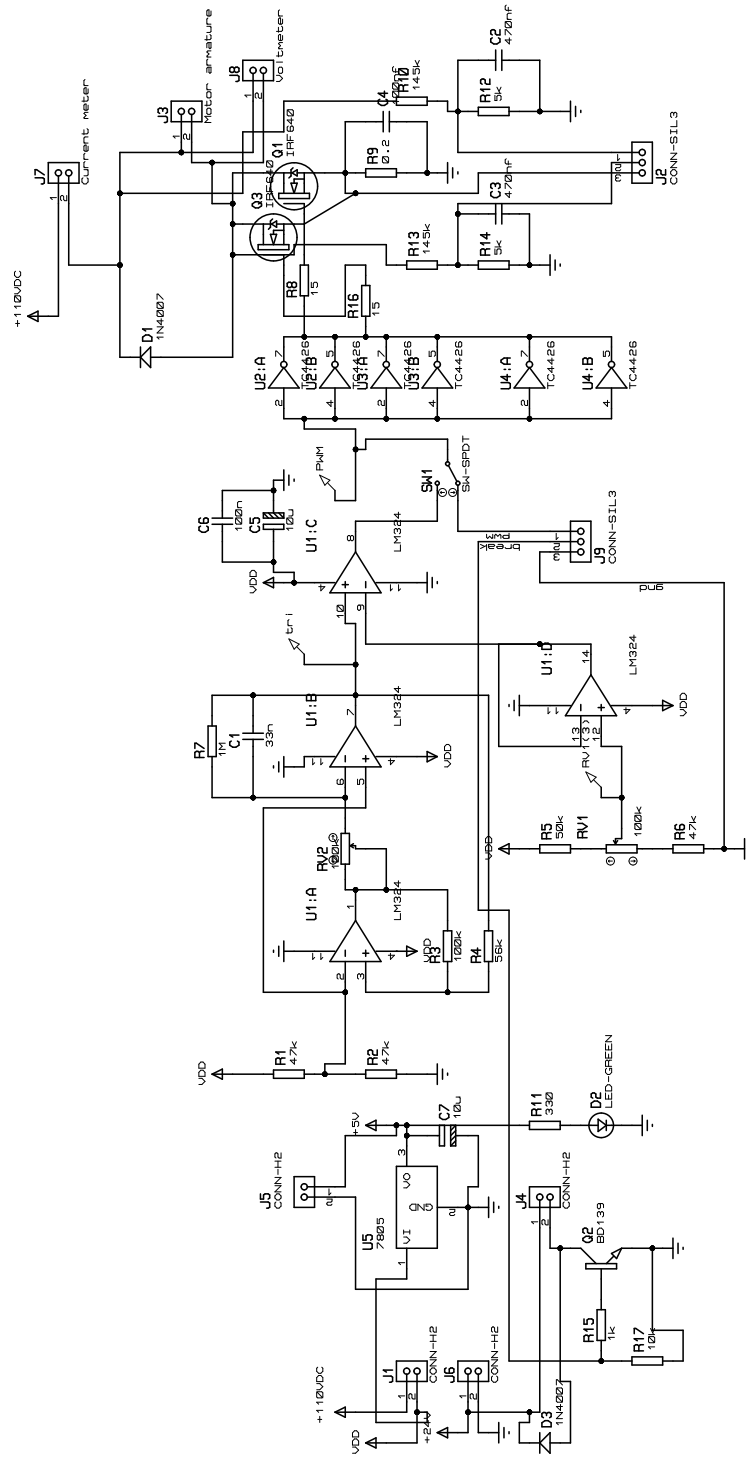


Figure B.3: Armature driver unit.

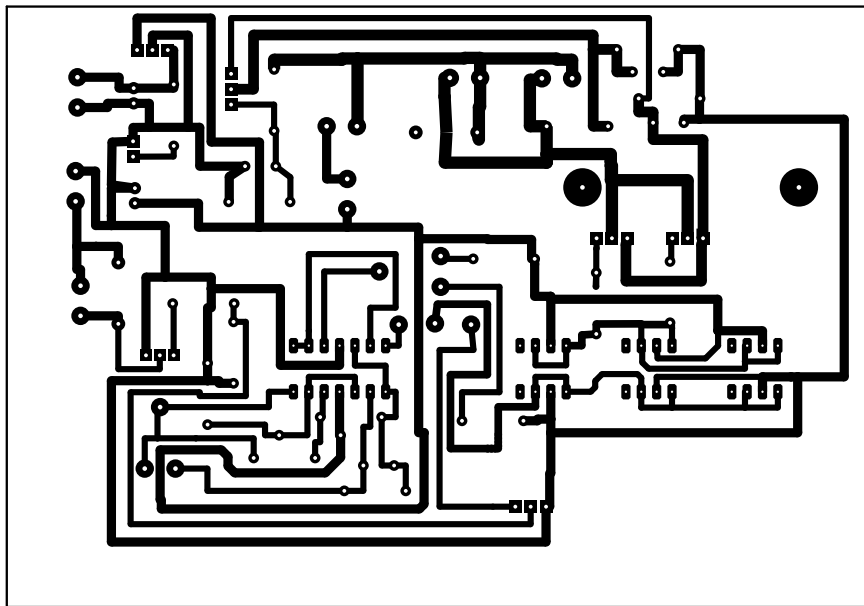


Figure B.4: Armature driver unit layout.

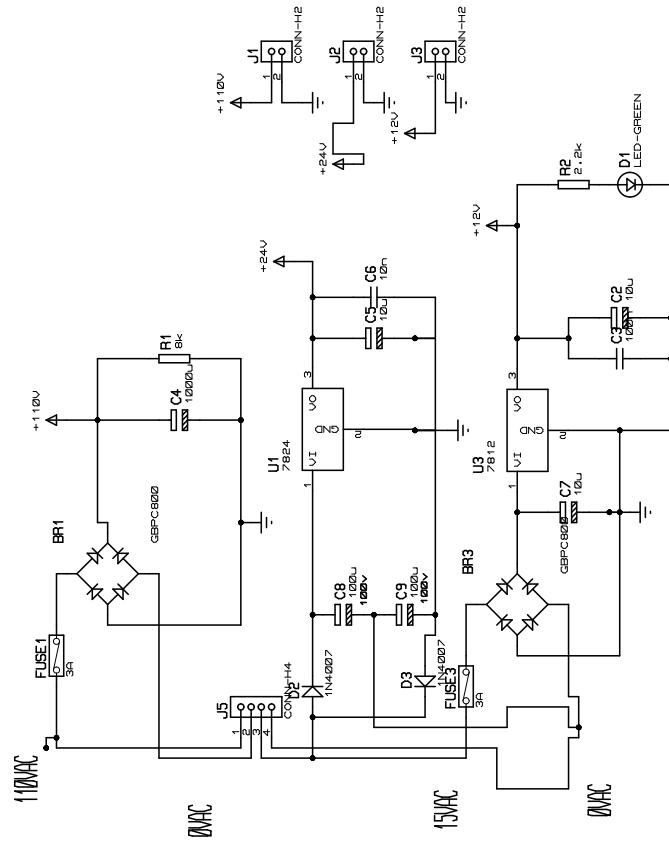


Figure B.5: Armature power circuit.

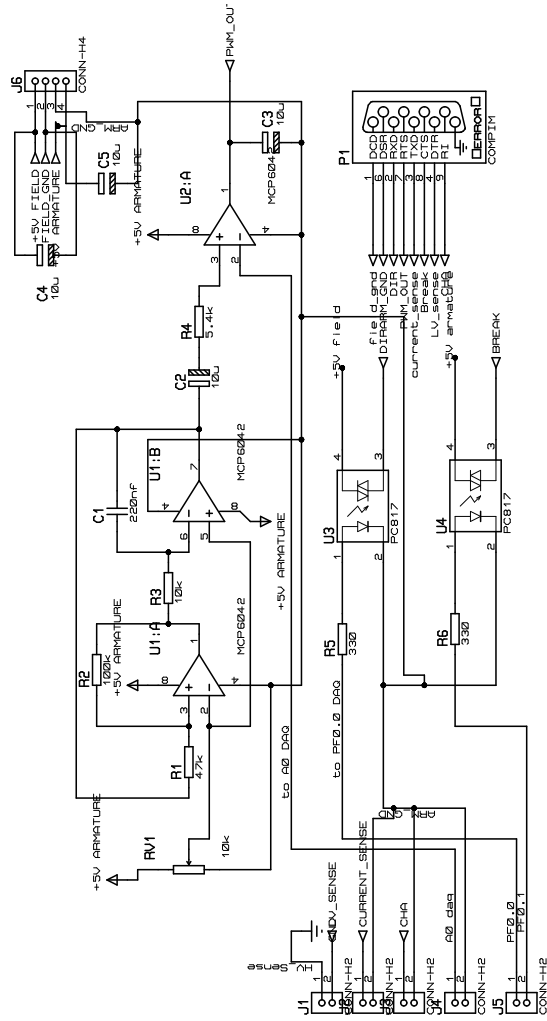


Figure B.7: interfacing circuit.