VOL. 10, NO. 19, OCTOBER 2015 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

# MODEL BASED DESIGN OF PID CONTROLLER FOR BLDC MOTOR WITH IMPLEMENTATION OF EMBEDDED ARDUINO MEGA CONTROLLER

M. K. Hat[1], B. S. K. K. Ibrahim[1], T. A. T. Mohd[2] and M. K. Hassan[2]
[1]Department of Mechatronic and Robotic Engineering, Faculty of Electrical &Electronic Engineering, University Tun Hussein Onn Malaysia, Batu Pahat, Johor, Malaysia
[2]Department of Electrical and Electronic, Faculty of Engineering, Universiti Putra Malaysia, UPM Serdang, Malaysia
E-Mail: mohdkamalhat@gmail.com

**ABSTRACT**

Brushless DC motors are the most widely used electrical drive in the industry. The development process of the drive is costly and time-consuming. However, various methods can be used to reduce the development time of the drive. This paper presents the Model-Based Design technique of Brushless DC Motor using MATLAB/Simulink with Arduino support block set. The model of BLDC motor is developed using black-box modeling approach; simulations are performed based on real-time data and processed using MATLAB System Identification tool box. The PID Controller is then designed and tuned within the simulations to attain the drive performance. For real-time application, the controller code is generated and uploaded into Arduino Mega embedded controller. The results obtained from simulation and experiment is discussed and compared. The performed works concludes that Model-based design technique can be applied in any control design application using low cost controller such as Arduino embedded controller.

**Keywords:** model based design, brushless dc motor, pid controller, MATLAB/simulink, arduino.

## INTRODUCTION

Model-Based Design (MBD) has been discussed for a decade's but only in recent years it has evolved into a complete design flow from model creation to complete end product. With advanced simulation tools software nowadays, the introduction of graphical control schematic entry tools and control design tools that greatly simplified the task of complex control design and evaluation (Frederiksen 2013).

The approach used in model based design is to build the abstract representation (model) of application in a modeling language and then obtain the implementation in a programming language automatically using code generator (Wakankar *et al*. 2010). Model-Based Design is a process that enables fast and cost-effective development of dynamic systems, including control systems, signal processing, and communications systems. In Model-Based Design, a system model is at the center of the development process, from requirements development through design, implementation, and testing (Matlab 2012).The simulation-based approach offers a better understanding of design alternatives and trade-offs than traditional hardware prototype-based design methodologies, enabling to optimize the design to meet predefined performance criteria. Rather than using complex structures and extensive software code, designers can define models with advanced functional characteristics using continuous time and discrete time building blocks. Existing C code can be integrated with standard control library blocks to maximize design efficiency.

The purpose of this paper is to implement control strategies of BLDC motor through Model-Based Design technique. Figure-1 below shows the design flow of Model-Based Design implemented in this paper.

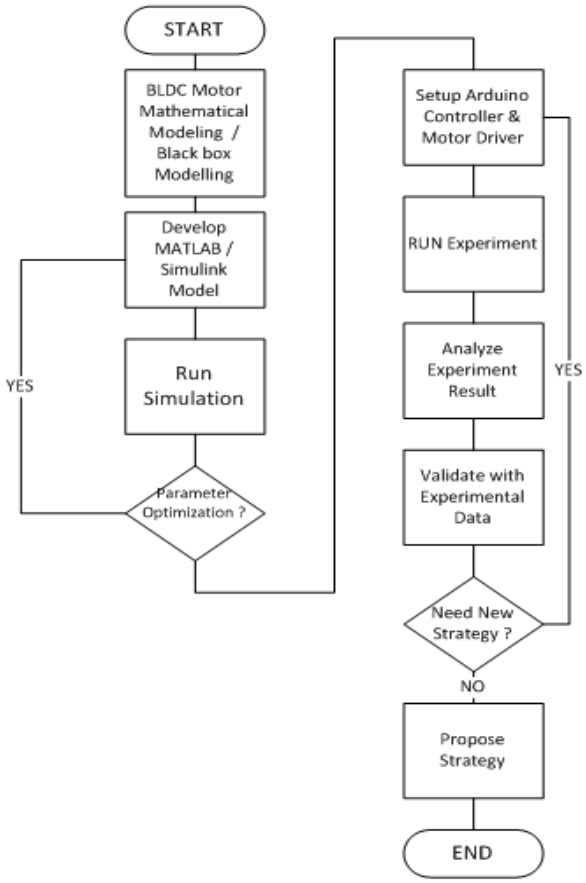


**Figure-1.** Design flow of model-based design.

## BRUSHEDLESS DIRECT CURRECT (BLDC) MOTOR

### Introduction of BLDC motor

Due to high efficiency, reliability, low maintenance and compact size, BLDC motors have become a more popular research area for researchers worldwide. BLDC motor has permanent magnet as a rotor. Hence, there is no excitation loss, simple construction is realized and operation is very reliable. As such advantages, BLDC motor can provide fast dynamic response with high efficiency (T.V.Mahendiran 2013). The position of the rotor can be sensed by using hall sensors or optical encoders. Due to their advantages, BLDC motors are suitable for electrical vehicles and hybrid electrical vehicles.

### BLDC motor modeling

The development process of BLDC motor drive is usually very time-intensive (Vonkomer & Žalman 2012). In practice, the design of BLDC motor drive involves a complex process such as mathematical modeling, simulation, hardware implementation and validation (Faieghi & Azimi 2010).Simulation can reduce the time-consume in design state, but it takes a lot of time to transform the model to the final code. Modeling and simulation have been widely used both at the designer and user stage for predicting and studying steady state and transient behavior (Motor *et al*. 1998)

Tuning a controller on a physical prototype or plant hardware can lead to unsafe operating conditions and damage the hardware. A more reliable approach is to build a plant model and simulate it to verify the controller at diverse operating conditions so as to run what-if scenarios without risk (Vinnakota 2012.).

Two common approaches to modeling dynamic systems are the White-box and the Black-box modeling or known as data-driven techniques. In the White-box technique, physics laws are used to derive the mathematical equations that describe the dynamics of the system. In the Black-box technique, the input and output signals of the actual system are measured and an approximate mathematical model is found to fit the collected I/O data (Pourboghrat 2011). Simulink can be used for modeling dynamic systems when their mathematical models are already known. When first-principle modeling is not feasible, an alternative is to develop models from input-output measurements of the plant. A low-order, linear model might be sufficient for designing a basic controller. Detailed analysis and design of a higher-performance controller requires a higher-fidelity and possibly nonlinear model. Using a System Identification toolbox in MATLAB, the BLDC motor model is developed. This paper presents how to identify a plant model from input-output data, use the identified model to design a controller, and implement it. The workflow includes the following steps: acquiring data, identifying linear and nonlinear plant models, designing and simulating feedback controllers, and implementing these controllers on an embedded microprocessor for real-time testing.

### Plant description

The BLDC motor plant is set to perform real-time system identification. The BLDC motor is mounted on a small aluminum platform, in parallel with one dynamic load and one quadrature rotary encoder as in Figure-2. The BLDC motor is powered by +24 volt DC switching power supply. Motor rpm varies from 0 to 3000 rpm. Motor shaft is attached to +24 volt dual stepper motor (shaft-typed), act as dynamic load. Rotary encoder is attached to the other end of dynamic load shaft to capture speed. Motor speed captured by Arduino MEGA controller, is then sent to the host computer through RS232 communication port.


**Figure-2.** BLDC motor plant.

### Data collection

The first step in system identification is to collect data from the BLDC motor. The task is done by embedded controller (Arduino Mega) interfaced through RS232 serial communication to the motor and host computer. Figure-3 show the Arduino MEGA communicates between host computer and BLDC motor drive. Arduino MEGA sends the PWM signal to BLDC motor; captures the speeds and send them to the host computer. Simulink model with Arduino hardware support block set is used to perform the PWM switching signal and data collection.

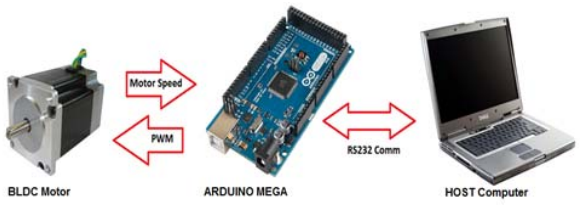


**Figure-3.** Arduino communication BLDC motor and computer.

The general procedure to determine a motor dynamic of the system involved 4 steps through experimental approach (Hussin 2011).

1. Experimental design

To capture input and output data under experiment procedure.

2. Select and define model structure or complexity
   To determine whether it is linear or non-linear system. A suitable model structure is chosen using prior knowledge and trial and error.
3. Parameter estimation
   The model fitting used to measure the percentage of acceptance and select the best fit of the model
4. Model validation
   Verification of the identified model and to investigate whether it meets the model requirement

Figure-4 below shows the flow chart of the system identification experiment conducted.

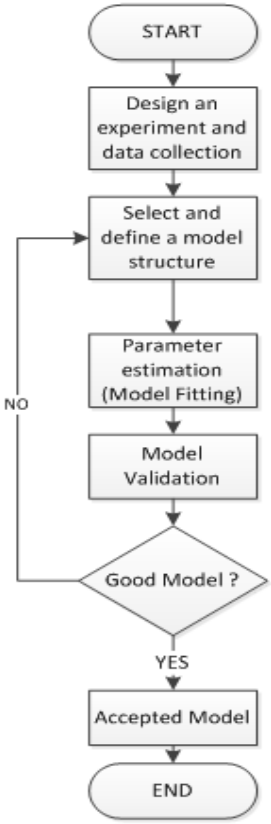


**Figure-4.** System identification steps.

**Experimental setup**

System Identification toolbox is used to find the time-domain model of the BLDC motor. The host computer is interfaced to the BLDC motor and motor speed is captured through Arduino Mega microcontroller. In order to collect the actual input/output data from the physical system, the Arduino Mega is programmed using Arduino block-set developed by MATLAB to send a test signal to the BLDC motor and receive the motor speed measurement from the motor encoder.

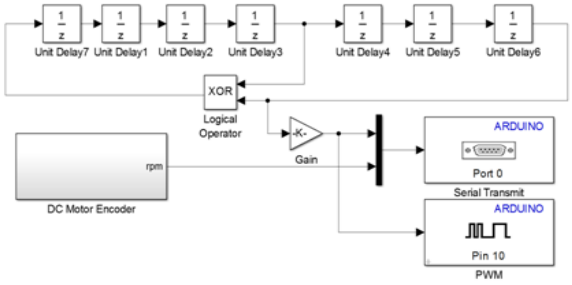


**Figure-5.** PRBS generator simulink model.

The Pseudo-Random Binary Signal (PRBS) signal is used as a simulink model of Arduino Mega program and +24 volt DC applied to the Motor as in figure 5 to capture the actual input and output data, with a sampling time of T=0.01s and motor is set to 3000 rpm, speed and actual Performance of motor at no load condition are analyzed. A Longs BLDC motor model 57BLF03 and motor driver rated current 20A is used. The List of parameters of motor is given in Table-1.

**Table-1.** 3 Phase BLDC motor specifications.

| Motor Model : 57BLF03 | |
|---|---|
| Output Power Watts | 188 Watt |
| Number of Poles | 8 |
| Number of phase | 3 |
| Stator Resistance | 0.2 Ohm |
| Stator Inductance | 0.31 Ohm |
| Rated Voltage | +24V |
| Rated Speed | 3000 RPM |
| Hording Torque | 0.6 N-m |
| Peak Current | 27.8 Amps |
| Torque Constant | 0.066 N-m/Amps |

The captured input and output data for PBRS input is transferred to the host computer for processing and system modeling. Two data consist of the armature voltage are applied through Pulse Wave Modulation (PWM) to motor stator and the motor speed are transmitted to the host computer in real-time using Arduino Serial Transmit block set. The data then is captured in MATLAB and plotted as in Figure-6.

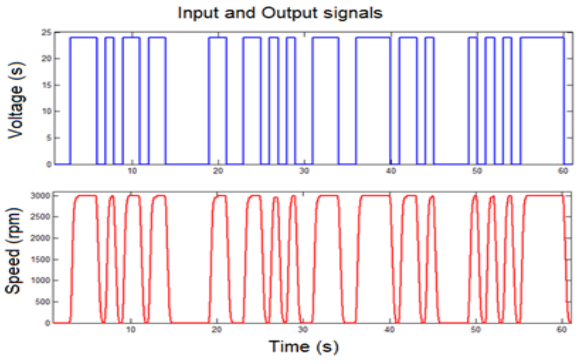

**Figure-6.** PRBS and motor speed signal captured in MATLAB.

A second order discrete transfer-function model as in equation (1), with sampling-time of Ts = 0.01s, is found for the system, with the model output of 91.44% best fit after the input and output signal processed by system identification tools box in MATLAB as shown in Figure-7.

$$RMSE = \sqrt{\sum_{K=1}^{N} \frac{(\hat{\omega}_m - \omega_m(K))^2}{N}} \quad (1)$$

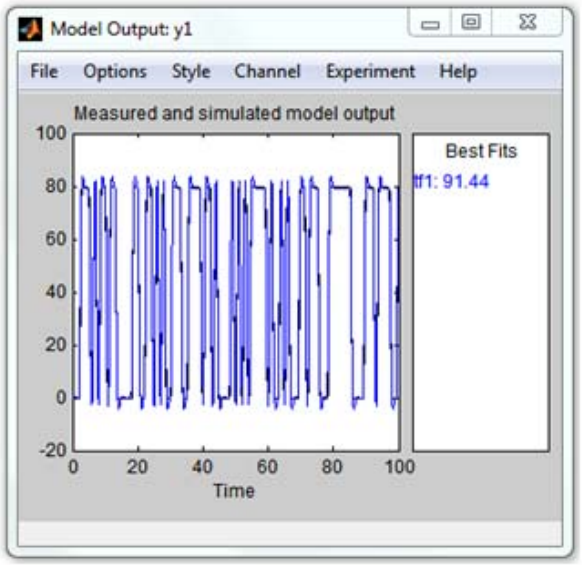


**Figure-7.** Model output best fits.

In Table-2, the model response parameter obtained from the system identification process, shows the physical nature of the BLDC motor performance. This parameter is very important while tuning the PID controller. The Rise Time (Tr) value must be equal or greater than the obtained Tr as in Table-1 while tuning the controller to ensure the PID controller work adequately when it is implemented in real plant. However, this condition is inapplicable during simulation works.

**Table-2.** Performance indicator of obtained BLDC motor model.

| Parameter | Value |
|---|---|
| Rise time | 0.535 sec |
| Settling time | 1.52 sec |
| Overshoot (%) | 4.87 % |
| Steady state value | 1.05 |

**Model validation**

The identified Transfer function model has been validated through comparison between the actual BLDC motor speed response and the model response. Model validation implemented in MATLAB/ Simulink software. The validation process consists of applying the measured PBRS signal as inputs to the model and comparing the simulated speed with the measured actual speed. Figure-8 verifies that the speed response of the transfer function model fit the actual speed from the BLDC motor.

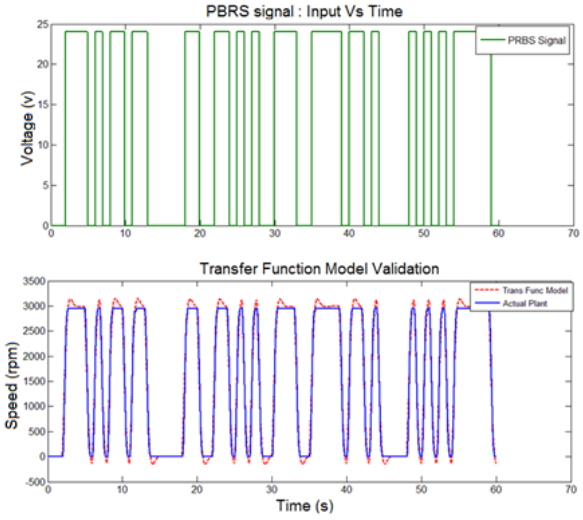


**Figure-8.** Transfer function model validation.

**SIMULATION AND EXPERIMENTAL RESULTS**

The transfer function model obtained from Black-box technique is used to design conventional PID controller. The complete system model is presented as simulink model in Figure-9.The MATLAB Simulink software is used to analyze the controller respond, tune the control system, observe and verify the system performance simultaneously as in Figure-10.

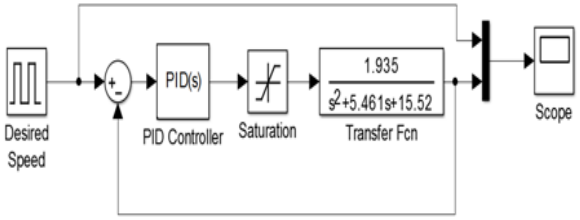


**Figure-9.** System model for BLDC motor control.

MATLAB PID Tuner provides a fast and widely applicable single-loop PID tuning method for the Simulink® PID Controller blocks. As such, PID controller parameters can be easily tuned to achieve a robust design with desired response time (Anon 2015). In this work, the PID controller is tuned to get the desired response from the BLDC motor model obtained from Black-box modeling technique.

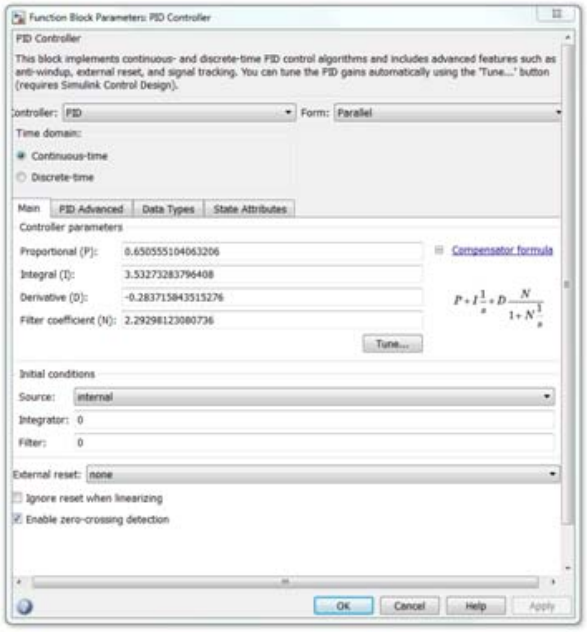


**Figure-10.** PID controller auto tuning GUI.

## Simulation results

By using the obtained tuned PID parameter, the response of the BLDC motor model is validated by simulating the model and analyzing the output plot .The response of auto-tuned model for the PID control parameter and simulated closed-loop motor speed are as shown in Figure-11. The advantage of this simulation is that, any changes in model tuning can be directly applied into the model and the verification of the model can be performed automatically before the code is generated.

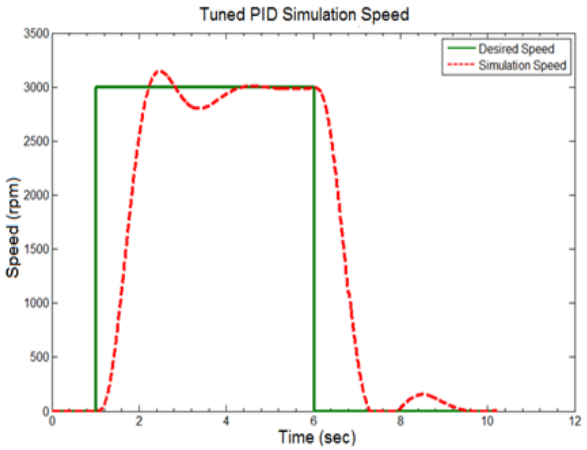


**Figure-11.** Simulation of PID controller model response.

The parameter from tuned model as presented in Table-2 shows the Rise time (Tr = 0.691 sec) exceeds the value of physical nature of BLDC motor and the PID controller works adequately with any input value.

**Table-3.** Tuned PID controller simulation parameter.

| Parameter | Value |
|---|---|
| Rise time | 0.715 sec |
| Settling time | 2.99 sec |
| Overshoot (%) | 5.07 % |
| Steady state value | 0.992 |

## Experimental results

The tuned (continuous-time) PID controller parameter block is then added to simulink model of Arduino program for real time implementation via Arduino MEGA. The parameters are as shown as in figure 10. The Arduino program is then built and uploaded into the embedded Arduino Mega microcontroller by using Arduino toolbox as in Figure-12 for implementing the tuned PID controller. The uploaded program would run the microcontroller.

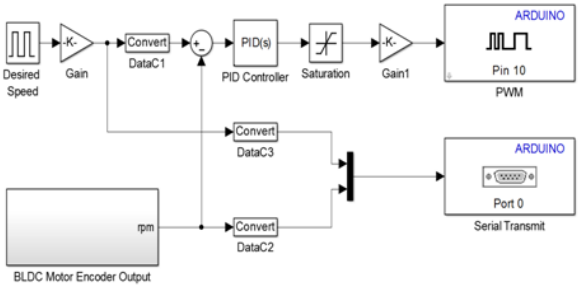


**Figure-12.** Simulink model of arduino mega program for PID controller.

## Validation results

The actual desired signal and motor speed in the closed-loop system are then captured and transmitted to the host computer in real time for the purpose of plotting and verification. Figure-13 shows the actual measurement of the motor speed with the desired command in the closed-loop system. Although the actual motor speed is slightly different than the simulated response as shows in Figure-14, the performance of the Arduino embedded controller is acceptable.

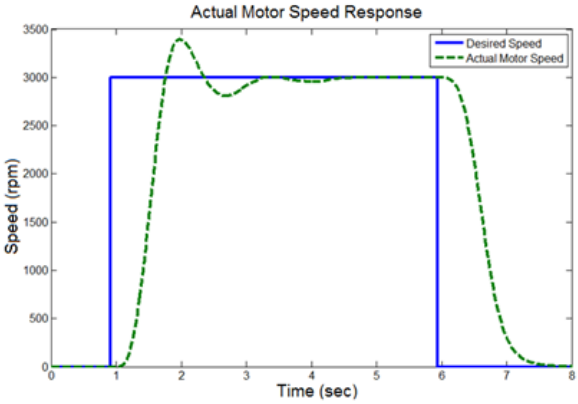


**Figure-13.** Actual motor speed in closed-loop.

Table-3 illustrates the actual BLDC motor speed response parameter. Comparison between the simulated and real-time implementation of PID BLDC motor speed controller, reveals only the insignificant small change of parameter value and therefore can be applied to a simple PID controller.

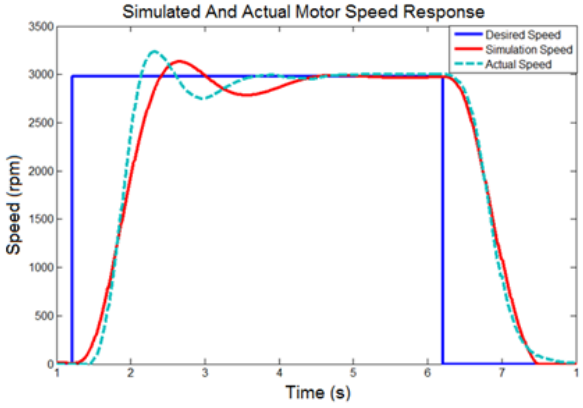


**Figure-14.** Comparison of simulation and actual speed.

In order to evaluate the simulation motor speed fits the Actual measured speed, root mean square error (RMSE) between measured and simulation speed was used. A lower RMSE indicates a higher accuracy of the controller.

$$G(s) = \frac{1.935}{s^2 + 5.461s + 15.52} \quad (2)$$

RMSE used in this calculation as in equation 2 (Rujan & Diaconu 2012) is applied. Where ω^_m Actual motor speed and ω_m (K) sampled valued of simulation speed response. Calculated RMSE obtained value from this work is 8.6.

**Table-4.** Actual BLDC motor response parameter.

| Parameter | Value |
|---|---|
| Rise time | 0.894 |
| Settling time | 2.49 |
| Overshoot (%) | 4.35% |
| Steady state value | 1.01 |

## CONCLUSIONS

In these works, the Model-Based Design approach in development of BLDC Motor PID controller using low cost embedded Arduino Mega controller has been presented. The comparison between simulated and real-time obtained data shows that the motor speed in actual implementation is not significantly different than the simulation result. However, both response are able to track their given input command and are acceptable. Performance indices using Root Mean Squared Error are evaluated and tuned PID controller compared between simulation and actual motor speed response. This work clearly demonstrates that Model-Based Design method which includes task of modeling, control design and rapid-prototyping of designing control system can be easier performed in MATLAB/ Simulink environment using any supported embedded microcontroller with their simulink block-set. The advantages of using Arduino controller with Simulink Arduino Target is an inexpensive, open-source microcontroller board and allows the creation of applications in the Arduino platform based on a visual programming environment with block diagrams. Furthermore, this method is feasible to the development of complex control system design such as artificial intelligence and controller optimization.

## REFERENCES

[1] Anon. (2005). PID Controller Tuning in Simulink - MATLAB & Simulink. Available at: http://www.mathworks.com/help/slcontrol/gs/automated-tuning-of-simulink-pid-controller-block.html [Accessed April 10, 2015].

[2] Faieghi, M. R. & Azimi, S. M. (2010). Design an optimized PID controller for brushless DC motor by using PSO and based on NARMAX identified model with ANFIS. In UKSim2010 - UKSim 12th International Conference on Computer Modelling and Simulation. pp. 16–21.

[3] Frederiksen, A. (2013). Model-Based Design of MS-2577. Available at: http://www.analog.com/media/en/technical-documentation/technical-articles/Model-Based-Design-of-Advanced-Motor-Control-Systems-MS-2577.pdf.

[4] Hussin, M. (2011). Modeling and validation of brushless dc motor. Modeling, Simulation and …, pp.0–3. Available at:

http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5775620 [Accessed January 14, 2015].

[5] MATLAB. (2012). SIMULINK: Simulation and Model-Based Design. Available at: http://www.mathworks.co.uk/products/simulink/ [Accessed April 3, 2015].

[6] Motor, I., Yaacob, S. & Mohamed, F. (1998). Black-box modelling of the induction motor. SICE'98. Proceedings of the 37th …, pp.883–886. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=742935 [Accessed December 10, 2014].

[7] Pourboghrat, F. (2011). enhancing mechatronics education using model- based techniques and math works tools enhancing mechatronics education using Model-Based Techniques and Mathworks. Journal of Engineering Education.

[8] Rujan, A. & Diaconu, L. (2012). Modified fuzzy controller for direct torque control of induction motor. Optimization of Electrical and Electronic, (2), pp.685–691. Available at: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6231979 [Accessed November 8, 2013].

[9] T. V. Mahendiran, K. T. (2013). A New Improved Algorithm For Speed Control Of Brushless DC Motor. IEEE International Conference on current Trends in Engineering and Technology, ICCTET'13.

[10] Vinnakota, B. P. (2012). Motor Control with Arduino. A Case Study in Data-Driven Modeling and Control Design. , (Ic), pp.1–9.

[11] Vonkomer, J. & Žalman, M. (2012). Model Based Design of Electric drives. , 1(4), pp.130–133.

[12] Wakankar, A. *et al*. (2010). Automatic test case generation in model based software design to achieve higher reliability. 2010 2nd International Conference on Reliability, Safety and Hazard - Risk-Based Technologies and Physics-of-Failure Methods (ICRESH), pp.493–499.