MODELING OF BRUSHLESS DRIVE MOTOR CONTROLLER FOR GMRT

Submitted in Partial Fulfillment of the Requirements for the Degree of

Bachelor of Technology
in
Avionics

by

VIVEK KUMAR
(SC08B080)

Department of Avionics
Indian Institute of Space Science and Technology
Thiruvananthapuram
May 2012

BONAFIDE CERTIFICATE

This is to certify that this project report entitled **“Modeling of Brushless Drive Motor Controller for GMRT”** submitted to **Indian Institute of Space Science and Technology, Thiruvananthapuram**, is a bonafide record of work done by “**VIVEK KUMAR”** under my supervision from **“January 9, 2012 to April 27, 2012”.**

Mr. SURESH SABHAPATHY
Engineer-‘F’, GMRT
NCRA-TIFR
PUNE

Dr. S.K. GHOSH
Centre Director, GMRT
NCRA-TIFR

Place
Date

Declaration by Author

This is to declare that this report has been written by me. No part of the report is plagiarized from other sources. All information included from other sources has been duly acknowledged. I aver that if any part of the report is found to be plagiarized, I shall take full responsibility for it.

VIVEK KUMAR
SC08B080

Place
Date

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of people who made it possible and whose constant encouragement and guidance has been a source of inspiration throughout the course of this project.

I express my sincere thanks to our Director Dr. K.S.Dasgupta, Mrs. Sheebarani Ph.D., Assistant Professor, Avionics at IIST, Mrs. Anandamayee Tej, Ph.D., Assistant Professor, Physical Sciences at IIST for providing such a great opportunity to work on the project.

I thank Mr. Suresh Sabhapathy, Engineer-'F' , GMRT, Khodad, Pune for the excellent guidance he has provided in completing this project.

I also express our profound gratitude to Dr. Selvaganeshan, Ph.D., Assistant Professor, Avionics at IIST for his sincere supervision and encouragement.

Finally and most importantly I thank the almighty for all this success.

ABSTRACT

This project discusses the modeling of the PMAC (Programmable Multi Axis Controller) as a controller for the brushless drive servo motor, which is replacing the brushed motors in most of the industries and facilities including GMRT (Giant Metrewave Radio Telescope). It gives details of the mathematical model of a Permanent Magnet DC Motor, and then that of a Brushless Motor. The software tool used for modeling and analysis is Matlab/Simulink. This project also discusses the advantages of the Brushless Motor over conventional motors. The project gives details of the PID servo filter which is sufficient to control the system. It is this PID servo filter that is modeled as the controller (PMAC). Finally this controller model is implemented in the system with the motor model, and the overall system response is obtained and analyzed.

Contents

LIST OF FIGURES

1 INTRODUCTION

1.1 BRIEF ABOUT GMRT

Giant Metrewave Radio Telescope (GMRT) is located at Khodad, 80 km north from Pune in India. It is the world's largest array of radio telescope at meter wavelength. It is operated by National Centre for Radio Astrophysics (NCRA), a part of Tata Institute of Fundamental Research (TIFR), Mumbai. A nearby town is Narayangaon on Pune – Nasik highway, 15 km from GMRT. GMRT consists of 30 fully steerable giant parabolic dishes of 45m diameter each spread over distance of up to 25km. This is a unique setup for astronomical research using metrewave length range of radio spectrum. At high frequencies the study of universe can easily be done. But in so high frequency RF noise is also high in other countries, but in India this RF noise level is comparatively low.

There are 14 antennas randomly arranged in the central square, with a further 16 arranged in 3 arms of the nearly "Y" shaped array giving an informating baseline of about 25km. GMRT is an interferometer, uses a technique known as aperture synthesis to make image of radio sources. Each antenna is of 45m diameter and consists of a solid surface like many radio telescopes; the reflector is made of wire rope stretched between metal struts in parabolic configuration. Each antenna has 4 different receivers mounted at the focus. Each individual receiver can rotate so that the user can select the frequency at which to observe. The array operated in six frequency bands centered on 50, 153, 233, 325, 610 and 1420 MHz. All these feeds provide dual polarization output. The construction of 30 large dishes at a relatively small cost has been possible due to an important technological breakthrough achieved by Indian Scientists and Engineers in the design of lightweight, low cost dishes. The design is based on what is being called the `SMART' concept for Stretch Mesh Attached to Rope Trusses.

Figure 1-1 GMRT Antenna Array

1.2 IMPORTANCE OF THE PROBLEM

In GMRT there are total 30 antennas and each antenna consists of four servo motors, two for azimuth operation and two for elevation operation. Thus there are total 120 servomotors. Earlier, these were all brushed motors. Brushes wear out due to constant contact during rotation between stator and rotor of these motors. Carbon powder gathers in between stator and rotor. Thus commutator segments might get shorted due to the burned carbon participated in between which might result in unbalancing of motor operation. Thus DC motor requires a periodic maintenance to prevent it from permanent damage. Thus in case of GMRT where a large number of these servo motors are used and they work constantly as GMRT is 24×7 observatory, it is a huge problem for maintenance all these 120 antennas periodically. This is costly as well as it takes a large time. GMRT antennas have 45m diameter and they observes the universe like Jupiter, Sun, Pulsar, and X rays sources. It also observes the near Galaxies, Supernova and cluster of galaxies at very high Radio Frequency ranges. These antennas specify for the ranges 40 – 1700 MHz. But in such high frequency noise level also creates problem for these brushed motors. In case of brushless motor the RFI can be eliminated.

Due to these reasons, GMRT has been replacing the brushed motors with the brushless servo motors. A model for these motors can therefore be used to easily see the improvements achieved with this decision, as well as to study and analyze several aspects that need an immediate attention. For accurate and efficient positioning and tracking of the GMRT antennas, a high-performance servo motion controller capable of commanding multiple axes of motion with a high level of sophistication is used. Such a device is PMAC. For an overall understanding of the control action of the PMAC, and to see how the controller improves the attributes of the motor, a model based approach is appropriate.

1.3 SCOPE OF THE PROJECT

The project takes a model based approach for the understanding of a problem. There is more emphasis on the mathematical modeling of the Motor for obtaining the transfer function rather than going into deep details of its construction and working. Mathematical models for a DC Brushed Motor and a Brushless Motor is achieved and implemented in Matlab/Simulink for study. The brushless motor used at GMRT is implemented in the model obtained. It is required to model the controller that PMAC provides for position control, and implement it in Matlab/Simulink with the motor model using the experimental data provided. Finally, the system response is obtained and analyzed.

2 WORK DESCRIPTION: APPROACH USED

2.1 SOFTWARE TOOL USED

MATLAB and SIMULINK

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include:

• Math and computation

• Algorithm development

• Data acquisition

• Modeling, simulation, and prototyping

• Data analysis, exploration, and visualization

• Scientific and engineering graphics

• Application development, including graphical user interface building

The name MATLAB stands for Matrix Laboratory.

SIMULINK is software for modeling, simulating, and analyzing dynamic systems. Simulink enables you to pose a question about a system, model it, and see what happens.

2.2 SCIENTIFIC MODELING

- Modeling as a substitute for direct measurement and experimentation
- Simulation: Implementation of a model
- Structure: Different elements and entities
- Systems: Construct or collection of different elements and entities
- Process of generating a model: As a conceptual representation of some phenomenon
- Process of evaluating a model: Ability to explain past observations, predict future observations, and simplicity

- Visualization: any technique of creating images, diagrams or animations to communicate a message

2.3 MOTOR

An electric motor is a machine which converts electric energy into mechanical energy. Its action is based on the current carrying conductor placed in magnetic field.

The BLDC motor used in the project for modeling purposes is:

KOLLMORGEN, ID.

BL MOTOR – 3-PHASE PM SERVOMOTOR

MODEL: AKM73M – KSC2R – 02

Performance Data - AKM7x Frame

AKM7x - Up to 640 VDC

See system data beginning on page 8 for typical torque/speed performance.

	PARAMETER	Tol	SYMBOL	UNITS	AKM72 K	AKM72 M	AKM72 P	AKM73 M	AKM73 P	AKM74 L	AKM74 P
	Max Rated DC Bus Voltage	Max	Vbus	Vdc	640	640	640	640	640	640	640
	Continuous Torque (Stall) for ΔT winding = 100°C	Nom	T_{cs}	N-m	29.7	30.0	29.4	42.0	41.6	53.0	52.5
				lb-in	263	266	260	372	368	469	465
	Continuous Current (Stall) for ΔT winding = 100°C	Nom	I_{cs}	A_{rms}	9.3	13.0	18.7	13.6	19.5	12.9	18.5
	Continuous Torque (Stall) for ΔT winding = 60°C	Nom	T_{cs}	N-m	23.8	24.0	23.5	33.6	33.3	42.4	42.0
				lb-in	211	212	208	297	295	375	372
	Max Mechanical Speed	Nom	N_{max}	rpm	6000	6000	6000	6000	6000	6000	6000
	Peak Torque	Nom	T_p	Nm	79.2	79.7	78.5	113	111	143	142
				lb-in	701	705	695	997	985	1269	1253
	Peak Current	Nom	I_p	A_{rms}	27.8	38.9	56.1	40.8	58.6	38.7	55.5
75VDC	Rated Torque (speed)		T_{rtd}	N-m	-	-	-	-	-	-	-
				lb-in	-	-	-	-	-	-	-
	Rated Speed		N_{rtd}	rpm	-	-	-	-	-	-	-
	Rated Power (speed)		P_{rtd}	kW	-	-	-	-	-	-	-
				Hp	-	-	-	-	-	-	-
160VDC	Rated Torque (speed)		T_{rtd}	N-m	-	-	-	-	-	-	-
				lb-in	-	-	-	-	-	-	-
	Rated Speed		N_{rtd}	rpm	-	-	-	-	-	-	-
	Rated Power (speed)		P_{rtd}	kW	-	-	-	-	-	-	-
				Hp	-	-	-	-	-	-	-
320VDC	Rated Torque (speed)		T_{rtd}	N-m	-	-	23.8	-	34.7	-	-
				lb-in	-	-	211	-	307	-	-
	Rated Speed		N_{rtd}	rpm	-	-	1800	-	1300	-	-
	Rated Power (speed)		P_{rtd}	kW	-	-	4.49	-	4.72	-	-
				Hp	-	-	6.01	-	6.33	-	-
560VDC	Rated Torque (speed)		T_{rtd}	N-m	25.1	23.6	20.1	33.8	28.5	43.5	39.6
				lb-in	222	209	178	299	252	385	350
	Rated Speed		N_{rtd}	rpm	1500	2000	3000	1500	2400	1200	1800
	Rated Power (speed)		P_{rtd}	kW	3.94	4.94	6.31	5.31	7.16	5.47	7.46
				Hp	5.29	6.63	8.46	7.12	9.60	7.33	10.01
640VDC	Rated Torque (speed)		T_{rtd}	N-m	24.0	22.1	18.2	32.1	26.3	41.5	35.9
				lb-in	212	196	161	284	233	367	318
	Rated Speed		N_{rtd}	rpm	1800	2500	3500	1800	2800	1400	2000
	Rated Power (speed)		P_{rtd}	kW	4.52	5.79	6.67	6.05	7.71	6.08	7.52
				Hp	6.06	7.76	8.94	8.11	10.34	8.16	10.08
	Torque Constant	±10%	K_t	N-m/A_{rms}	3.23	2.33	1.58	3.10	2.13	4.14	2.84
				lb-in/A_{rms}	28.6	20.6	14.0	27.4	18.9	36.6	25.1
	Back EMF constant	±10%	K_e	V/k_{rpm}	208	150	102	200	137	266	183
	Resistance (line-line)	±10%	R_m		1.22	0.64	0.33	0.68	0.35	0.85	0.43
	Inductance (line-line)		L	mH	20.7	10.8	5.0	12.4	5.9	16.4	7.7
	Inertia (includes Resolver feedback)		J_m	kg-cm²	65			92		120	
				lb-in-s²	0.057			0.082		0.11	
	Optional Brake Inertia (additional)		J_m	kg-cm²	1.64			1.64		1.64	
				lb-in-s²	1.46×10^{-3}			1.46×10^{-3}		1.46×10^{-3}	
	Weight		W	kg	19.7			26.7		33.6	
				lb	43.4			58.8		74.0	
	Static Friction		T_f	N-m	0.16			0.24		0.33	
				lb-in	1.4			2.1		2.9	
	Viscous Damping		K_{dv}	N-m/k_{rpm}	0.06			0.13		0.2	
				lb-in/k_{rpm}	0.5			1.2		1.8	
	Thermal Time Constant		TCT	minutes	46			53		60	
	Thermal Resistance		R_{thw-a}	°C/W	0.43			0.37		0.33	
	Pole Pairs				5			5		5	
	Heatsink Size				10"x10"x½" Aluminum Plate			10"x10"x½" Alum. Plate		10"x10"x½" Alum. Plate	

Notes:
1. Motor winding temperature rise, ΔT=100°C, at 40°C ambient.
2. All data referenced to sinusoidal commutation.
3. Add parking brake if applicable for total inertia.
4. Motor with standard heatsink.
5. May be limited at some values of Vbus.
6. Measured at 25°C.
7. Brake motor option reduces continuous torque ratings by 1 N-m.
8. Non-Resolver feedback options reduce continuous torque ratings by:
AKM72 = 2.0 N-m AKM73 = 2.7 N-m
AKM74 = 3.4 N-m
9. Motors with non-Resolver feedback and Brake option, reduce continuous torque by:
AKM72 = 3.9 N-mAKM73 = 5.1 N-m
AKM74 = 6.2 N-m
10. For motors with optional shaft seal, reduce torque shown by 0.25 N-m (2.21 lb-in), and increase T_f by the same amount.

Figure 2-1 AKM MOTORS DATASHEET

2.4 MOTOR MODEL

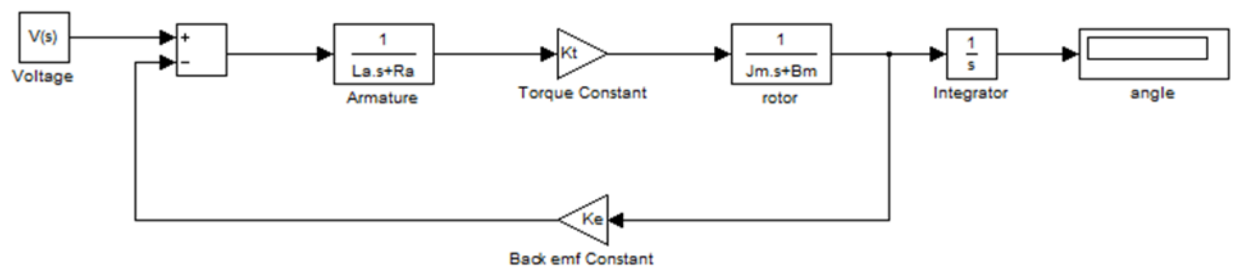


Figure 2-2 DC Motor Block Diagram

2.4.1 BRUSH DC MOTOR

Mathematical Model:

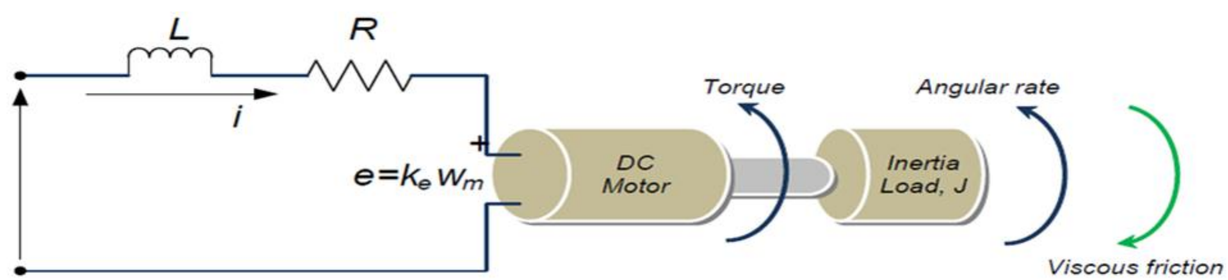


Figure 2-3 DC Motor Electromechanical Arrangement

Table 2-1 Electromechanical Parameters

L	Armature Inductance
R	Armature Resistance
B_m	Viscous Friction Constant
J	Rotor Inertia
K_e	Back emf Constant
K_t	Torque Constant
T	Electric Torque
ω	Angular Rate

Applying KVL:

$$V = Ri + L\frac{di}{dt} + e \qquad \text{Eq. 2-1}$$

$$e = K_e\omega \qquad \text{Eq. 2-2}$$

Therefore,

$$V = Ri + L\frac{di}{dt} + K_e\omega \qquad \text{Eq. 2-3}$$

Also, for no load, Electric Torque can be expressed as

$$T = B_m\omega + J\frac{d\omega}{dt} \qquad \text{Eq. 2-4}$$

This gives,

$$\frac{di}{dt} = \frac{V}{L} - \frac{iR}{L} - \frac{K_e}{L}\omega \qquad \text{Eq. 2-5}$$

Also,

$$\frac{d\omega}{dt} = \frac{K_t i}{J} - \frac{B_m\omega}{J} \qquad \text{Eq. 2-6}$$

Applying Laplace Transform to the differential equations, we get

$$si = \frac{V}{L} - \frac{iR}{L} - \frac{K_e\omega}{L} \qquad \text{Eq. 2-7}$$

$$s\omega = \frac{K_t i}{J} - \frac{B_m\omega}{J} \qquad \text{Eq. 2-8}$$

This gives,

$$i = \frac{s\omega + B_m\omega}{\frac{K_t}{J}} \qquad \text{Eq. 2-9}$$

Putting this value of current in Eq. 2-7, we get

$$\left(\frac{s\omega + \frac{B_m\omega}{J}}{\frac{K_t}{J}}\right)\left(s + \frac{R}{L}\right) = \frac{-K_e\omega}{L} + \frac{V}{L} \qquad \text{Eq. 2-10}$$

Or, solving this results to

$$\frac{\omega}{V} = \frac{K_t}{Js^2L+B_mLs+RJs+B_mR+K_tK_e} \qquad \text{Eq. 2-11}$$

This is the required transfer function of a conventional DC Motor.

Ideal Condition Analysis and Approximations: Time Constants

Consider a DC Motor with no Load torque and negligible viscous damping when the rotor rotates. Therefore,

$$B_m = 0 \qquad \text{Eq. 2-12}$$

Thus, the transfer function becomes:

$$\frac{\omega}{V} = \frac{K_t}{Js^2L+RJs+K_tK_e} \qquad \text{Eq. 2-13}$$

Now, by multiplying the numerator and denominator by $\frac{R}{RK_eK_t}$ we get

$$\frac{\omega}{V} = \frac{1/K_e}{\frac{RJ}{K_eK_t}\frac{Ls^2}{R}+\frac{RJ}{K_eK_t}s+1} \qquad \text{Eq. 2-14}$$

Now, define:

$$\tau_m = \frac{RJ}{K_eK_t}$$

$$\tau_e = \frac{L}{R}$$

Where τ_m and τ_e are Mechanical Time Constant and Electrical Time Constant respectively.

Thus the transfer function of DC Motor is

$$\boldsymbol{\frac{\omega}{V} = \frac{1/K_e}{\tau_m\tau_e s^2 + \tau_m s + 1}}$$

2.4.2 BRUSHLESS DRIVE MOTOR

2.4.2.1 MAIN FEATURES

- It can be envisioned as a brush DC motor turned inside out, where the permanent magnets are on the rotor, and windings on the stator.
- As a result, there are no mechanical brushes and commutators in this motor.
- There is no physical contact between the rotor and the stator.
- The motor is referred to as a “DC motor” because its coils are driven by a DC power source which is applied to the various stator coils in a predetermined sequential pattern: process known as commutation.
- The name “BLDC” is actually a misnomer, motor being effectively an AC motor: current in each coil alternates during each electrical cycle.
- How it is driven: in a BLDC, which stator coil is driven is determined by the rotor position. So, the knowledge of the rotor position is required to determine which stator coils to energize.

2.4.2.2 ADVANTAGES OVER CONVENTIONAL MOTORS

- More efficient, much cooler.
- There is no mechanical commutation to wear out.
- There is very less chance of sparking, unlike brushed motors, making them better suited to environments with volatile chemicals and fuels.
- Acoustically very quiet motors.
- Long operating life.
- High dynamic response.
- Better speed vs. torque characteristics.
- Higher torque – weight ratio.
- Higher speed range.

2.4.2.3 MATHEMATICAL MODELING

Typically, the mathematical model of a brushless motor is not totally different from the conventional DC motor. The major thing added is the phases involved which affects the overall results of the BLDC motor. The phases peculiarly affect the resistive and the inductive of the BLDC arrangement. For example, a simple arrangement with a symmetrical 3-phase and "wye" internal connection could give a brief illustration of the whole phase concept.

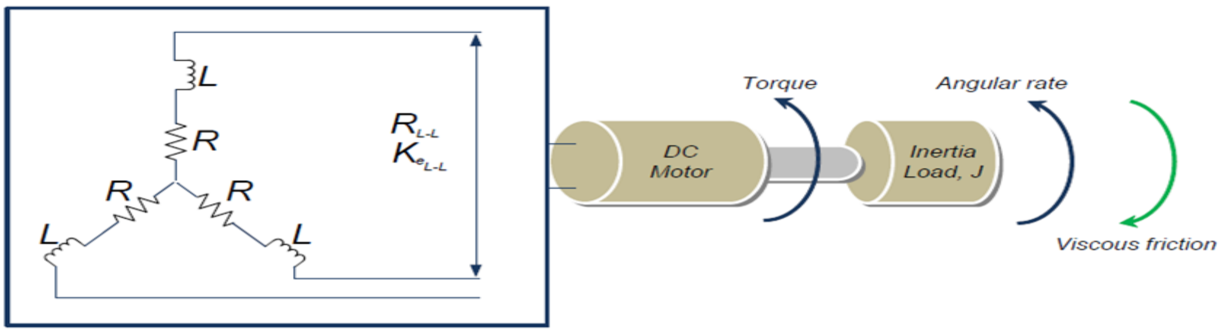


Figure 2-4 A Brushless Motor Electromechanical Arrangement

The mechanical and electrical time constants for a brushless motor have the same basic equations as the conventional DC motors with some variations.

The mechanical time constant is

$$\tau_{m_{BLDC}} = \frac{\sum R_{phase} J}{K_{e_{phase}} K_t}$$

The electrical time constant is

$$\tau_{e_{BLDC}} = \frac{L_{L-L}}{\sum R_{L-L}}$$

Now,

$$K_{e_{phase}} = \frac{K_{e_{L-L}}}{\sqrt{3}} = K_e \qquad \text{Eq. 2-15}$$

Since, this is a symmetrical arrangement, i.e. R1=R2=R3=R

Therefore,

$$\sum R_{L-L} = 3R \qquad \text{Eq. 2-16}$$

Hence, the mechanical time constant for this example of brushless motor is

$$\tau_{m_{BLDC}} = \frac{3RJ}{K_e K_t} = 3\tau_m$$

And the electrical time constant is

$$\tau_{e_{BLDC}} = \frac{L}{3R} = \frac{\tau_e}{3}$$

Putting this in the transfer function derived for a DC motor:

$$\boldsymbol{\frac{\omega}{V} = \frac{1/K_e}{\tau_m \tau_e s^2 + \tau_m s + 1}}$$

Eq. 2-17

We get the transfer function of a brushless motor:

$$\boldsymbol{\frac{\omega}{V} = \frac{1/K_e}{\tau_{m_{BLDC}} \tau_{e_{BLDC}} s^2 + \tau_{m_{BLDC}} s + 1}}$$

Eq. 2-18

This result obtained is implemented in Matlab to study the brushless motor open loop characteristics, and the open loop transient and frequency responses. In practical applications or for the modeling purposes, the viscous friction can't be neglected. Hence, the motor model is implemented in Matlab/Simulink with added viscous friction dynamics, and the response and characteristics are analyzed, and compared to the ideal scenario.

Further, the motor is modeled and improved with the use of PID controller toolbox, a Simulink Toolbox and GUI application to study and understand the effects of P, I and D gains before implementing it with the PMAC PID filter controller model.

2.5 CONTROL

A Controller is a device which monitors and affects the operational conditions of a dynamic system. It serves to govern in some predetermined manner the performance of an entity or a device, such as an electric motor. When one or more output variables of a system need to follow a certain reference over time, the controller manipulates the inputs to a system to obtain the desired effect on the output of the system. Control has four functions: Measure, Compare, Compute and Correct.

At GMRT, there are 30 antennas and each antenna consists of 4 servomotors. This project aims at modeling the controller that can be used for position control for at least one motor to position the antenna in a desired direction. The following figure gives a general idea of the feedback control system where the controller positions the antenna by giving a controlled signal to the motor:

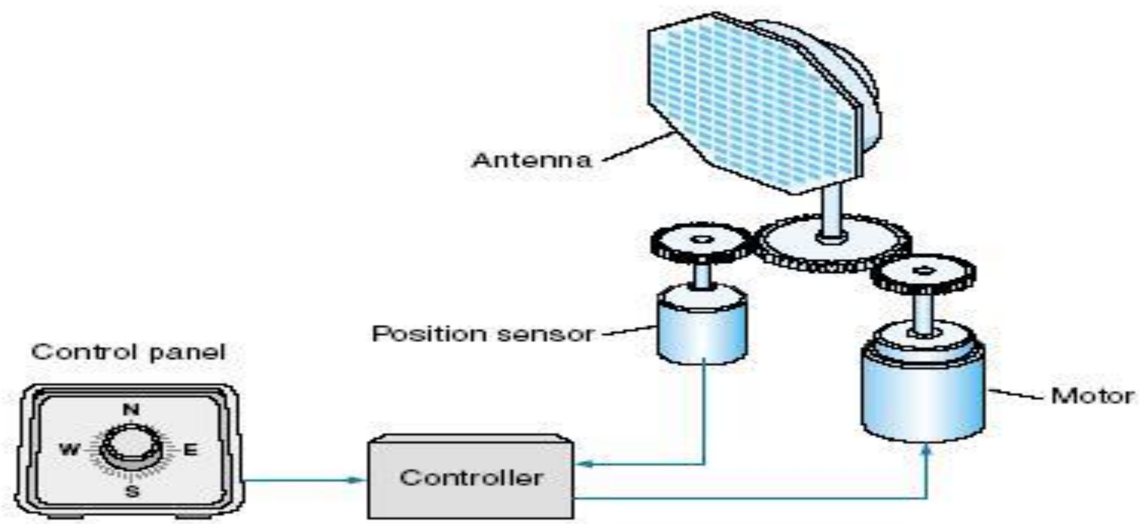


Figure 2-5 A simple Control System Involving Antenna Positioning

The antennas at GMRT need to be regularly synchronized with each other, and they need to be accurately positioned and to be able to track an object in the sky. This sort of positioning and tracking requires controller with a high level of accuracy and sophistication, and the motion control need to be done for several axes. Such a controller is PMAC.

2.5.1 PMAC

PMAC stands for Programmable Multi-Axis Controller. The Delta Tau Data Systems, Inc. Programmable Multi-Axis Controller (PMAC) is a family of high performance servo motion controllers capable of commanding up to eight axes of motion simultaneously with a high level of sophistication. Through the power of a Digital Signal Processor (DSP), PMAC offers a price-performance ratio for multi-axis control that was not previously available. Motorola's DSP56001 is the CPU for PMAC, and it handles all the calculations for all eight axes. There are four hardware versions of PMAC: the PMAC-PC, the PMAC-Lite, the PMAC-VME, and the PMAC-STD. These cards differ from each other in their form factor, the nature of the bus interface, and in the availability of certain I/O ports. All versions of the card have identical on-board firmware, so PMAC programs written for one version will run on any other version. The PMAC-STD has a different memory mapping of some I/O. Any version of PMAC may run as a standalone controller, or it may be commanded by a host computer, either over a serial port or over a bus port.

The Universal PMAC-Lite board, member of the PMAC family, is a 4-axis motion controller. The term "Lite" stands to indicate a maximum of four on-board axes of motion control. The term "Universal" indicates that this motion controller can have different types of on-board backup memory, either battery based type or flash type. Each axis is controlled by an independent channel circuitry, which in turn is composed of the following features:

- A single differential 16-bits DAC output
- Amplifier enable output
- One quadrature incremental encoder input
- Four dedicated flag inputs: two end-of-travel limits, one home input and one amplifier fault input

The Universal PMAC-Lite can be programmed to control the motion of up to four motors in any coordinated fashion, either independently of each other or coordinated with, for example, linear or circular interpolation. The Universal PMAC-Lite is not only a very sophisticated motion controller but it is also a PLC, Programmable Logic Controller device. PLC programs in PMAC run conveniently independently of each other and of motion programs and can be very tightly synchronized to the motion sequence. PMAC has its own on-board memory. Programs and motion parameters can be kept in memory without the need to reprogram each time PMAC is power up.

2.5.1.1 PMAC SOFTWARE SETUP

- I – variables (Initialization Parameters)
- P – variables
- M – variables
- Q – variables

We are mostly concerned with some of the I-variables which are required in the controller model.

2.5.1.2 PMAC FEATURES: TASKS IN PRIORITY

- Executing Motion Programs: The most obvious task of PMAC is executing sequences of motions given to it in a motion program. When told to execute a motion program, PMAC works through the program one move at a time, performing all the calculations up to that move command (including non-motion tasks) to prepare for actual execution of the move. PMAC is always working ahead of the actual move in progress, so it can blend properly into the upcoming move, if required.
- Executing PLC Programs: The sequential nature of motion program suits it well for commanding a series of moves and other coordinated actions; however these programs are not good at performing actions that are not directly coordinated with the sequence of motions. For these types of tasks, PMAC provides the capability for users to write PLC programs. These are named after Programmable Logic Controllers because they operate in a similar manner, continually scanning through their operations as fast as processor time allows. These programs are very useful for any task that is asynchronous to the motion sequences.
- Servo Loop Update: In an automatic task that is essentially invisible to the PMAC user, PMAC performs a servo update for each motor at a fixed frequency (usually around 2 kHz). The servo update for a motor consists of incrementing the commanded position (if necessary) according the equations generated by the motion program or other motion

command, comparing this to the actual position as read from the feedback sensor, and computing a command output based on the difference. This task occurs automatically without the need for any explicit commands.

- Commutation Update: If PMAC is requested to perform the commutation for a multiphase motor, it will perform commutation updates automatically at a fixed frequency (usually around 9 kHz). The commutation, or phasing, update for a motor consists of measuring and/or estimating the rotor magnetic field orientation, then apportioning the command that was calculated by the servo update among the different phases of the motor. This task occurs automatically without the need for any explicit commands.
- Housekeeping: PMAC regularly and automatically performs housekeeping tasks that make sure the system is in good working order. These tasks include the safety checks, such as following error limits, hardware overtravel limits, software overtravel limits, and amplifier faults. They also include the update of the watchdog timer. If any problem in hardware or software keeps these tasks from executing, the watchdog timer will trip, and the card will shut down.
- Communicating With the Host: PMAC can communicate with the host at any time, even in the middle of a sequence of motions. PMAC will accept a command, and take the appropriate action (putting the command in a program buffer for later execution), providing a data response to the host, starting a motor move, etc. If the command is illegal, it will report an error to the host.

2.5.1.3 CONTROLLER

The PID Controller:

The PID controller is the most common form of feedback controller. It was an essential element of early governors and it became the standard tool when process control emerged in the 1940s. In process control today, more than 95% of the control loops are of PID type, most loops are actually PI control. PID controllers are today found in all areas where control is used. The controllers come in many different forms. There are standalone systems in boxes for one or a few loops, which are manufactured by the hundred thousand yearly. PID control is an important ingredient of a distributed control system. The controllers are also embedded in many special purpose control systems. PID control is often combined with logic, sequential functions, selectors, and simple function blocks to build the complicated automation systems used for energy production, transportation, and manufacturing. Many sophisticated control strategies, such as model predictive control, are also organized hierarchically. PID control is used at the lowest level; the multivariable controller gives the set points to the controllers at the lower level. The PID controller can thus be said to be the “bread and butter’’ of control engineering. It is an important component in every control engineer’s tool box.

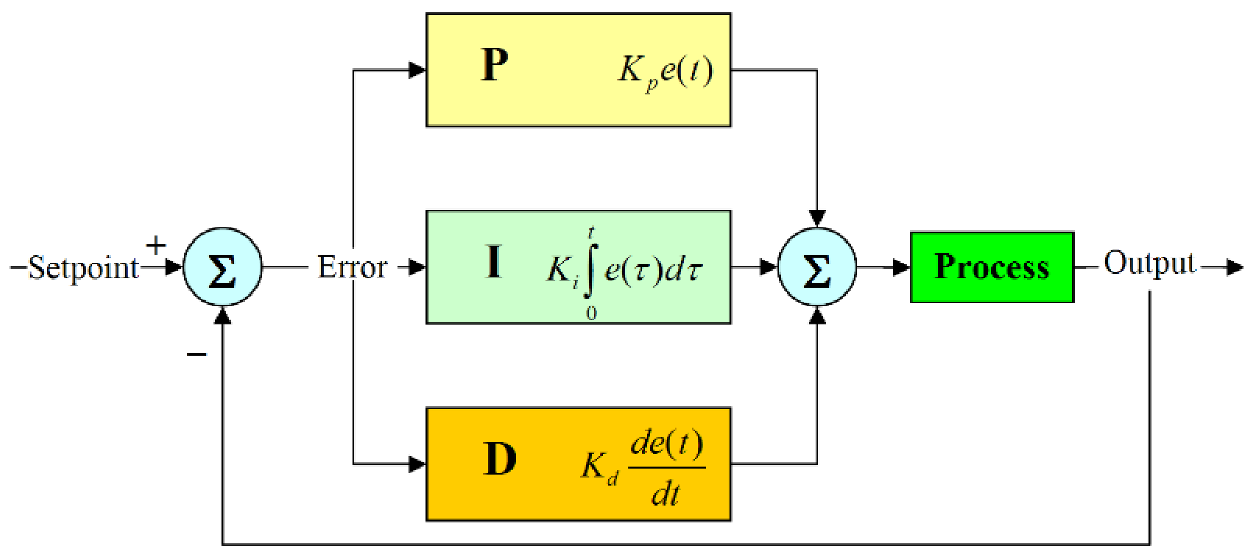


Figure 2-6 A System with PID Control

Some Characteristic Effects:

The proportional gain Kp will reduce the rise time and might reduce the steady state error of the system. The integral gain Ki will eliminate the steady state error but it might have a negative effect on the transient response. And the derivative gain Kd will tend to increase the stability of the system, reducing overshoot percentage, and improving the transient response of the system. In all, the following table gives the comprehensive effects of each of the controllers on a typical closed loop system:

Table 2-2 PID Controller Parameter Characteristics on a Typical System

Parameter	Rise Time	Overshoot	Settling Time	Steady State Error
Kp	decreases	increases	Small change	decreases
Ki	decreases	increases	increases	eliminate
Kd	Small change	decreases	decreases	Small change

The PMAC Controller to be modeled:

The controller is a multi-axis PID controller with feed-forward and feedback capabilities. In order to work within the available memory of the PMAC, and correctly deal with the different resolutions of the encoders, scale factors are used throughout the control loop. The PMAC command output is governed by Equation below where the n represents the time step. The output servo command is commutated and sent to a linear differential amplifier that is tuned for each axis. The PMAC takes approximately 0.443ms per servo cycle which is about 2257 samples per second which represents a loop closure rate of 2.257 kHz for control purposes. The command output of the PMAC shown in Equation below is essentially a PID filter with feed-forward terms; the variables are listed in the following table. The command output is given in encoder counts and limited to 32,767 encoder counts with a range of ±10V volts:

PMAC eqn. 2-1:

$$CMDout(n) = 2^{-19}.Ix30\left[\left\{Ix08.FE(n) + \frac{Ix32.CV(n)+Ix35.IE(n)}{2^7} + \frac{Ix33.IE(n)}{2^{23}}\right\} - \frac{Ix31.Ix09.AV(n)}{2^7}\right]$$

Table 2-3 Variables Listed and Their Description

Variable Name	Description
Ix30, Ix31, Ix33	PID respectively
Ix08, Ix09	Position and Velocity loop scale factors
Ix32	Feed-forward velocity gain
Ix35	Feed-forward acceleration gain

CA(n), CV(n)	Command Acceleration and Velocity
FE(n), IE(n)	Following error and Integration error
AV(n)	Actual Velocity

2.5.2 PMAC PID FILTER

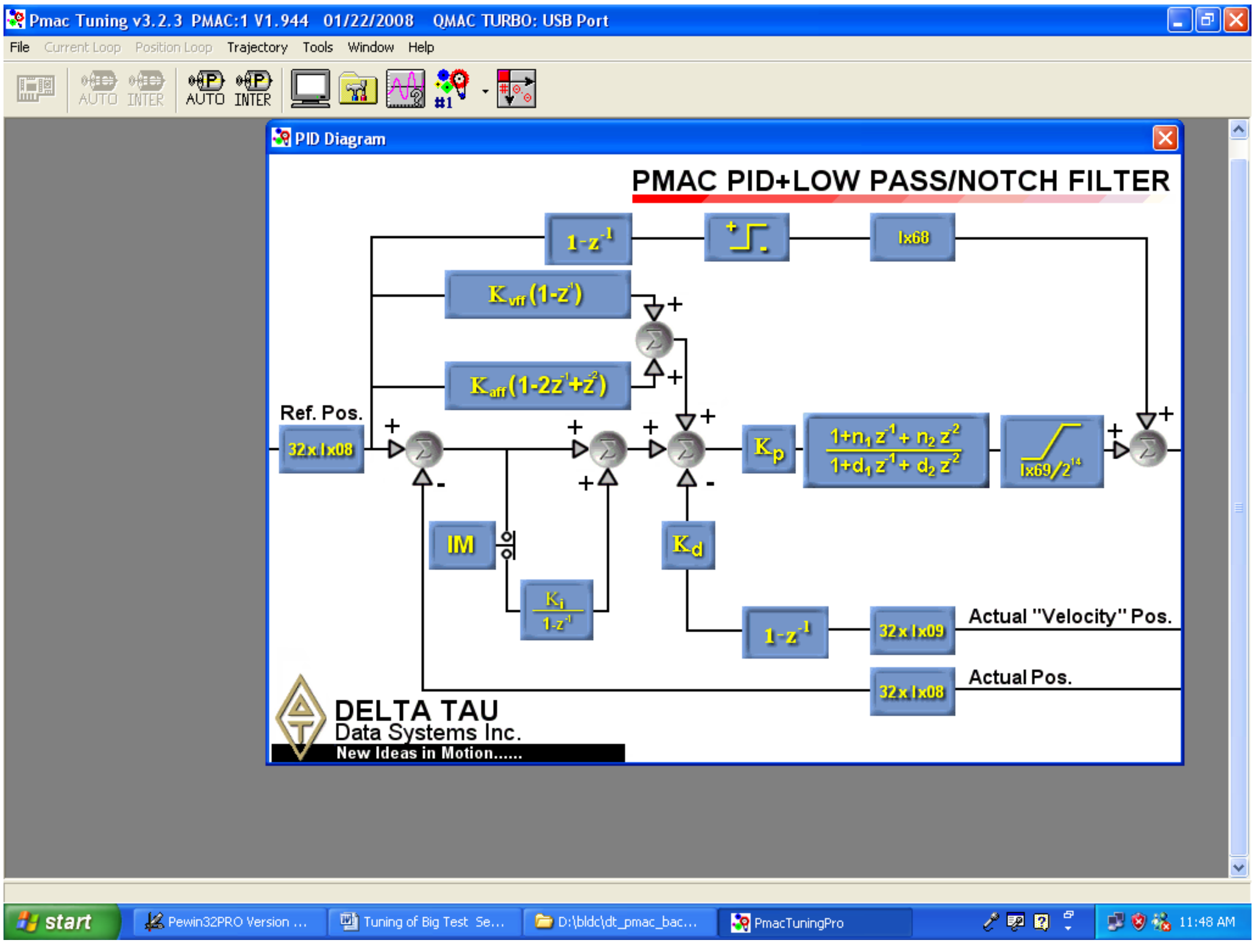


Figure 2-7 Diagram for PMAC PID Servo Loop Filter

The standard PMAC controller provides a PID position loop servo filter. Usually, this filter is sufficient to control the system, and easily understandable as well, even for non-control specialists. The filter is tuned by setting the appropriate I-variables for each motor.

The proportional gain (P — Ix30) provides the stiffness of the system; the differential gain (D — Ix31) provides the damping for stability; the integral gain (I — Ix33) eliminates steady-state errors. Ix34 determines whether the integral gain is active all the time, or just during periods when the commanded velocity is zero.

In addition, velocity feed forward gain (Ix32) reduces following errors introduced by damping (which are proportional to velocity), and acceleration feed forward gain (Ix35) reduces or eliminates following errors due to system inertia (which are proportional to acceleration).

Present Terms:

Ixx30 (Kp) - Proportional Gain - Increasing proportional gain stiffens the servo loop and increases the natural frequency of the closed loop system. Theoretically, increasing the proportional gain will result in improved positioning and tracking. However, often for real systems increasing the proportional gain increases their sensitivity to the noise and disturbances. If skipping the auto tuning, begin tuning with the low proportional gain setting. The default value is a very conservative value for most systems and is a good starting point.

Ixx31 (Kd) - Derivative Gain - Derivative gain works like damper. Higher the derivative gain, higher the damping action. This gain prevents overshoot but makes the system sluggish. Also, in digital system the quantization noise is amplified when derivative gain is applied, and in slow moving (low counts per second) system this noise might contribute significantly to the error. If skipping the auto tuning, it is always wise to begin the tuning with the low derivative gain setting. Default value is conservative value for most system and is a good starting points.

Ixx32 (Kvff) - Velocity FF (feed forward) Gain - Velocity feed forward gain will help the system with steady state error reduction. But whereas PID is feedback gain (causal), velocity feed forward gain is feed forward gain (non-causal). Setting it to an unreasonable value will destabilize the system. Using the derivative gain is recommended. Often the optimal result is obtained by setting this value equal to the derivative gain value.

Ixx33 (Ki) - Integral Gain - Integral gain acts to correct the system according to the accumulated following error of the system. It is effective particularly to counter the steady state error caused by friction. However, for numerical reasons (too high integral gain will saturate servo loop numerically) and servo stability concerns, an excessively high value of the integral gain is discouraged. Begin tuning with a lower value and observe the improvements. If no integral action is desired, set it to 0.

Ixx34 (IM) - Integral Mode - Setting this value to 0 enables the integrator all the time and setting it to 1 enables the integrator only when commend velocity is zero.

Ixx35 (Kaff) - Acceleration FF (feed forward) Gain - This gain helps the tracking effort of the system. Determination of this gain involves somewhat complex calculations, but there are intuitive ways to apply this gain. If your velocity plot (try parabolic move for checking velocity following) of the system shows bad tracking at initial acceleration or deceleration, applying this gain will help.

Ixx29 - DAC Offset - If for some reason the amplifier or PMAC has DAC offset, it is recommended for the first step that the DAC offset in hardware is zeroed manually; enter ‘o0’ command in the terminal window and the adjust the voltage of the amplifier to zero volts by manipulating the potentiometer on the amplifier. For the second step, whatever offset cannot be corrected by hardware adjustment may be corrected in software level by setting the DAC offset.

Ixx69 - DAC Limit - If the amplifier has more power than the motor can handle, make sure that the DAC limit is set so that excessive current will not burn the motor.

Ixx68 - Friction FF (feed forward) Gain - This gain acts when the position of the servo is not within in position limits at zero velocity state (steady state), according to the direction needed to be compensated. Use moderation in setting this gain.

3 MODELING AND SIMULATIONS

3.1 IMPORTANT FEATURES AND STEPS

- PMAC has tuning software that can determine the PID gains.
- The PMAC tuning algorithm is proprietary but likely a variant of Ziegler-Nichols.
- Model based tuning is much more efficient.
- The motor is in an axis of the plant. This axis is actuated by an input command that comes from the PMAC in the form of DACcounts.
- This value is then converted by a scale factor, K_{DAC} of 20/65536 V/DACcounts and sent to the amplifier as a command voltage across the motor.
- The amplifier then turns this value into a command current proportional to its transconductance, K_{TC}.
- The output position is converted from counts to degrees by a rotary encoder and then feedback to the PMAC.
- Amplifier dynamics is much faster than the motor dynamics. A reasonable assumption is they are negligible. The amplifier is included in the model so that the variables associated with it can be adjusted to see their effect on the overall system.
- The servo command is converted into voltage by resolution of the DAC.
- Actual position is converted from encoder counts to degrees by the 4551 multiplier.
- The amplifier dynamics are much faster than the motor dynamics. A reasonable assumption is that they are negligible. However, the amplifier is included in the model so that the variables associated with it can be adjusted to see their effect on the overall system. The amplifier also acts as a low-pass filter of 523Hz.

3.2 SYSTEM MODELING: TRANSFER FUNCTION

The model for the brushless motor where input is voltage and output is position read by encoder, from eqn. 2-11 is

$$\frac{\theta}{V} = \left(\frac{K_t}{Js^2L+B_mLs+RJs+B_mR+K_tK_e}\right)\left(\frac{1}{s}\right) \qquad \text{Eq. 3-1}$$

The amplifier is treated a simple low pass filter with a gain. The transfer function for the combined motor and amplifier is

$$\frac{AP}{CMDout} = Res.DAC\left(\frac{K_A.2\pi f}{s+2\pi f}\right)\left(\frac{K_t}{Js^2L+B_mLs+RJs+B_mR+K_tK_e}\right)\left(\frac{1}{s}\right) \qquad \text{Eq. 3-2}$$

Where, Res is the digital to analog converter resolution, DAC is the conversion factor of the digital to analog converter, f is the cut-off frequency of the low pass filter, and K_A is the amplifier gain.

By putting the value of CMDout from PMAC eqn. 2-1, we get

$$AP = 2^{-19}.Ix30\left[\left\{Ix08.FE(n) + \frac{Ix32.CV(n) + Ix35.IE(n)}{2^7} + \frac{Ix33.IE(n)}{2^{23}}\right\} - \frac{Ix31.Ix09.AV(n)}{2^7}\right] Res.DAC\left(\frac{K_A.2\pi f}{s+2\pi f}\right)\left(\frac{K_t}{Js^2L+B_mLs+RJs+B_mR+K_tK_e}\right)\left(\frac{1}{s}\right)$$

Eq. 3-3

This equation gives the actual position of the rotor.

Now,

$$FE(n) = CP(n) - AP(n) \qquad \text{Eq. 3-4}$$

$$CV(n) = CP(n) - CP(n-1) = CP(1 - z^{-1}) \qquad \text{Eq. 3-5}$$

$$CA(n) = CV(n) - CV(n-1) = CP(1 - 2z^{-1} + z^{-2} \qquad \text{Eq. 3-6}$$

$$IE(n) = \sum_{i=1}^{n} FE(n)_i = \frac{1}{1-z^{-1}}(CP - AP) \qquad \text{Eq. 3-7}$$

For simplicity, define:

$$K_{ps} = 2^{-19} Ix30$$

$$K_{is} = 2^{-23} Ix33$$

$$K_{ds} = 2^{-7} Ix31$$

$$K_{vffs} = 2^{-7} Ix32$$

$$K_{affs} = 2^{-7} Ix35$$

Using the above definitions yields:

$$CMDout = \left[K_{ps} Ix08 \left(1 + K_{vffs}(1 - z^{-1}) + K_{affs}(1 - 2z^{-1} + z^{-2}) + \frac{K_{is}}{1-z^{-1}}\right) CP - K_{ps}\left(Ix08 + \frac{Ix08 K_i}{1-z^{-1}} + K_{ds} Ix09(1 - z^{-1})\right) AP\right] \qquad \text{Eq. 3-8}$$

Also, from eqn. 3-2, we have

$$CMDout = \frac{AP}{Res.DAC\left(\frac{K_A \cdot 2\pi f}{s+2\pi f}\right)\left(\frac{K_t}{Js^2 L + B_m Ls + RJs + B_m R + K_t K_e}\right)\left(\frac{1}{s}\right)} \qquad \text{Eq. 3-9}$$

Thus, by equating equations 3-8 and 3-9, we get the transfer function $\frac{AP}{CP}$.

(The terms in s-domain are converted to z-domain by bilinear transformation for further analyzing).

These values and equations are used in modeling the PMAC + Motor system in Matlab/Simulink environment.

3.3 SIMULATIONS

3.3.1 MOTOR MODELS

3.3.1.1 DC MOTOR MODEL

The simple DC motor model is implemented in SIMULINK. The responses are plotted and compared using LTI viewer in control design-linearize model GUI tool.

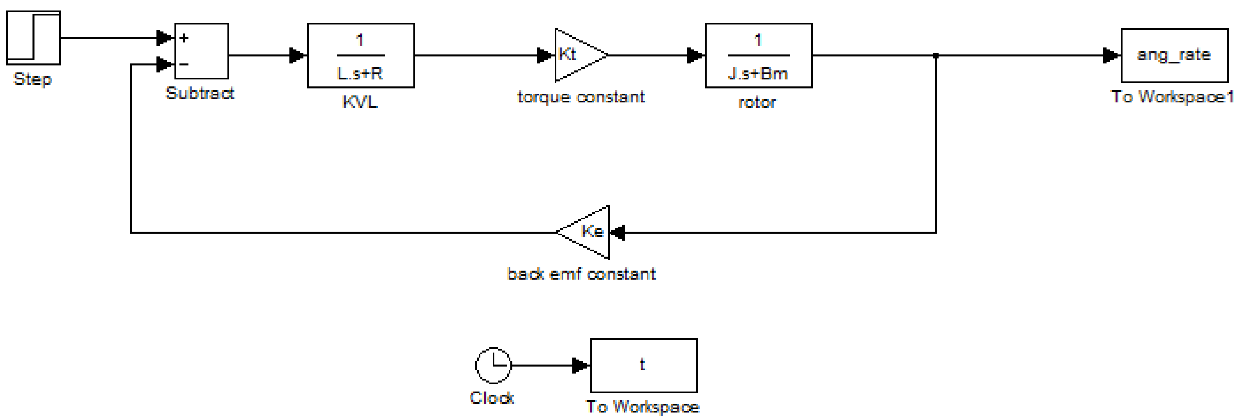


Figure 3-1DC Motor Simulink Model

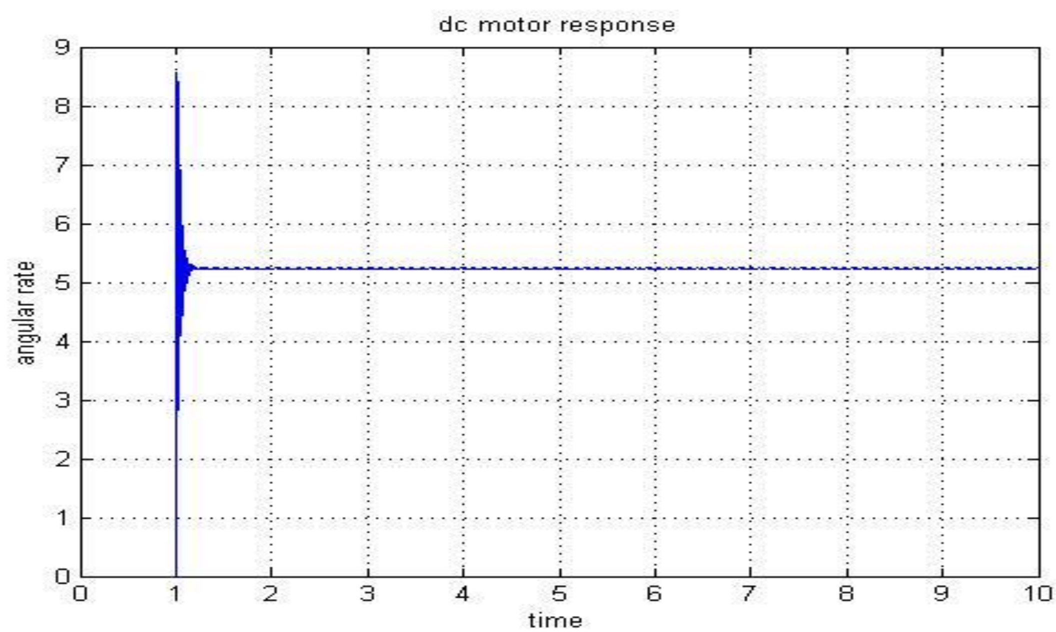
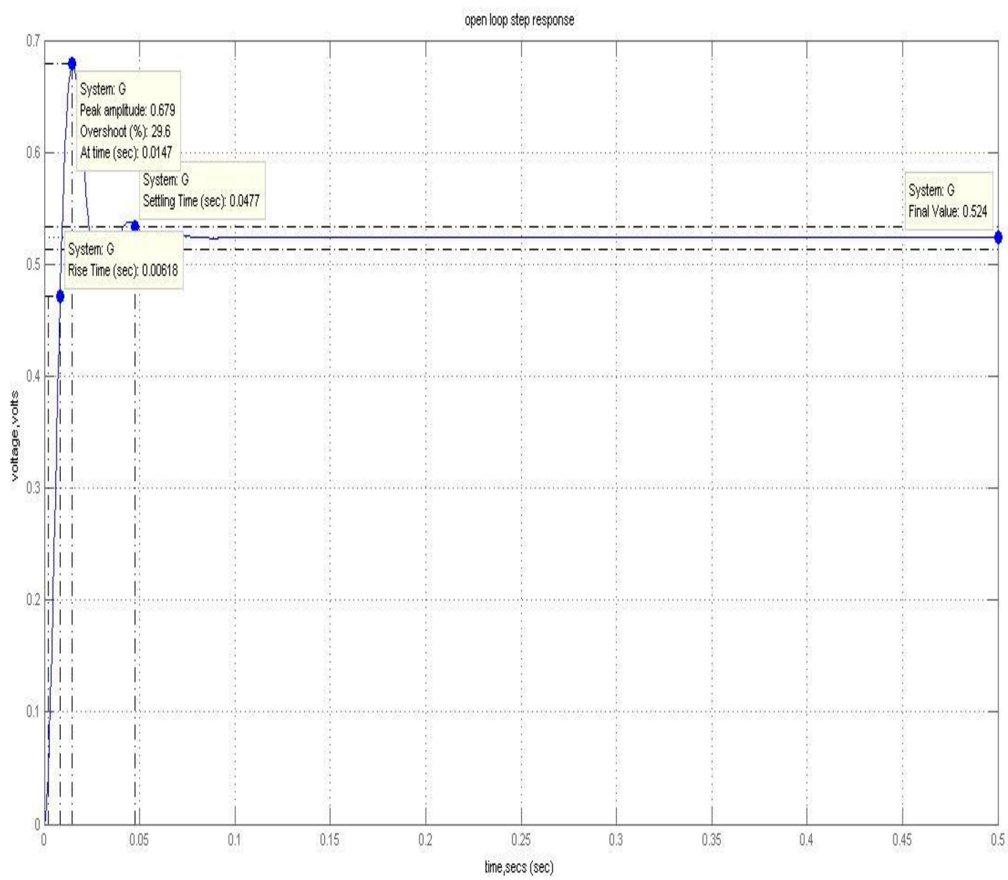


Figure 3-2 a DC Motor Angular Rate: Step Voltage Applied

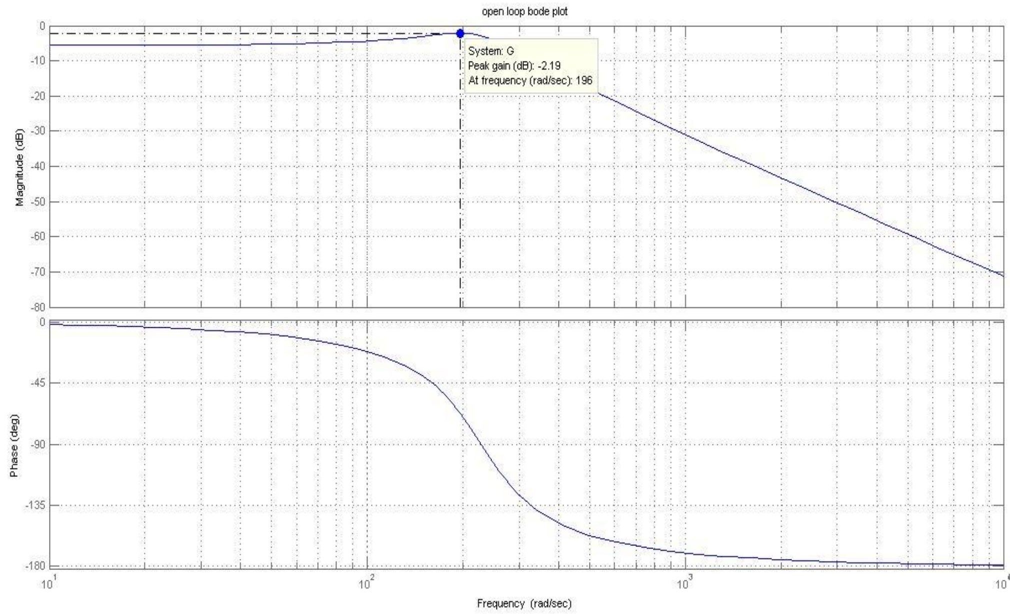
3.3.1.2 BRUSHLESS MOTOR MODEL

First, an open loop analysis of brushless motor dynamics is done using the provided datasheet values. The transfer function expression obtained earlier (refer eqn. 2-18) is calculated using matlab and the responses are plotted. Refer Appendix A for the matlab codes used.

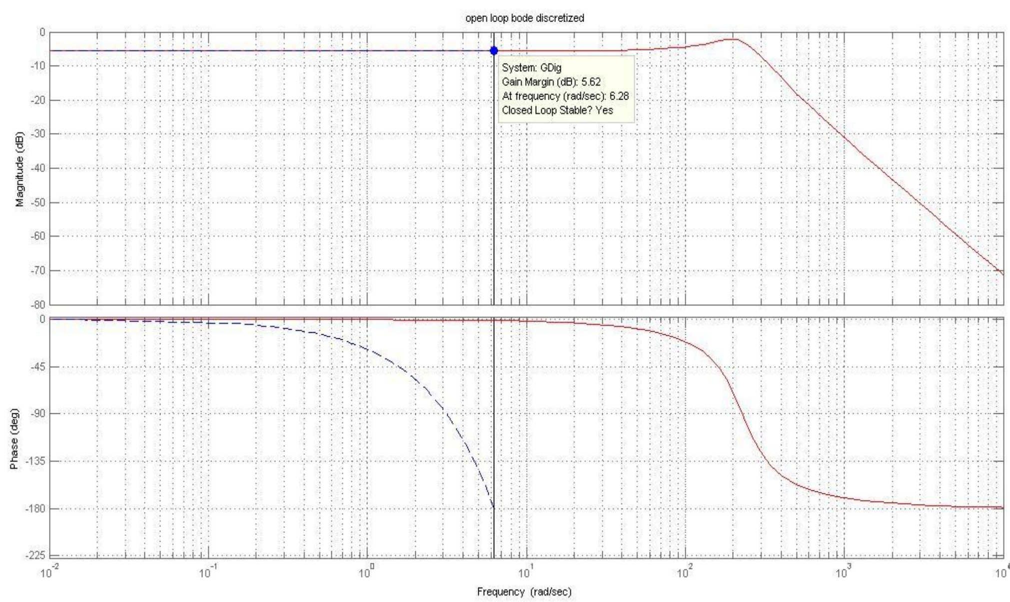
The plots:



Eq. 3-10 BL Motor Model Step Response



Eq. 3-11 Open Loop Bode Plot



Eq. 3-12 Open Loop Bode Discretized

Transfer Function, Mechanical and Electrical Time Constants obtained by Matlab:

```
>> G

Transfer function:
            0.5236
------------------------------
1.927e-005 s^2 + 0.00317 s + 1


>> te

te =

    0.0061

>> tm

tm =

    0.0032
```

This analysis was a time constant approach, and used an assumption by neglecting viscous friction constant. Further, the brushless motor model is implemented with added viscous friction:

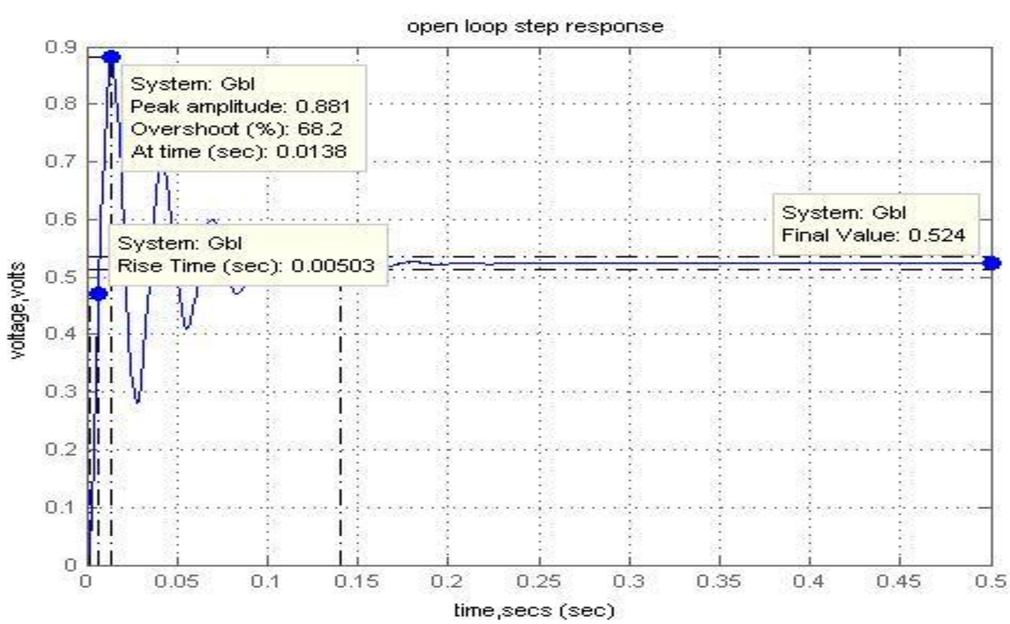


Figure 3-3 brushless motor step response

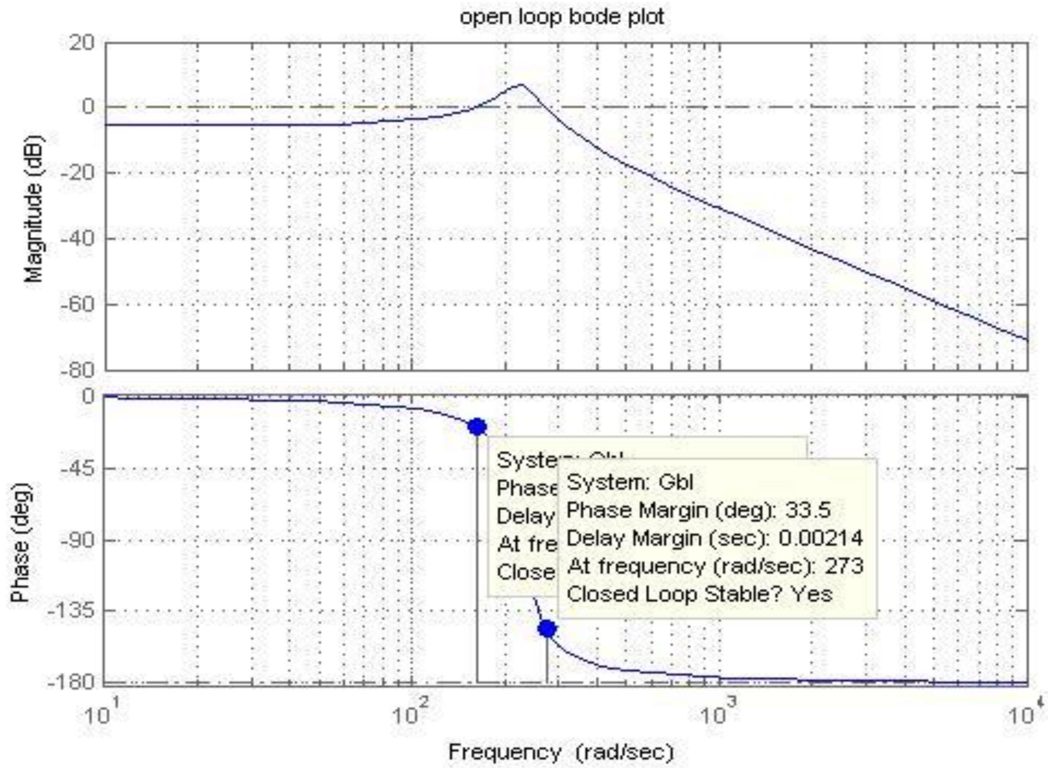


Figure 3-4 Brushless Motor Open Loop Bode

3.3.1.3 EFFECTS OF A PID CONTROLLER ON THE MOTOR RESPONSE

The motor model obtained is implemented in Matlab/Simulink with an inbuilt digital PID controller block. This block provides a GUI to automatically tune the gain values for the desired response. With the help of these system simulations, the theory on PID is verified as well as it gives a general idea of how a PID controller is justified in improving the attributes of a motor, which is very helpful in further modeling of the PMAC controller model.

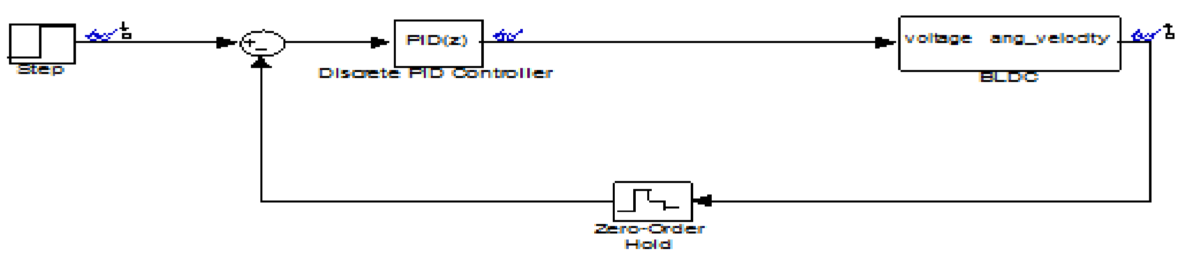


Figure 3-5 Motor with PID controller

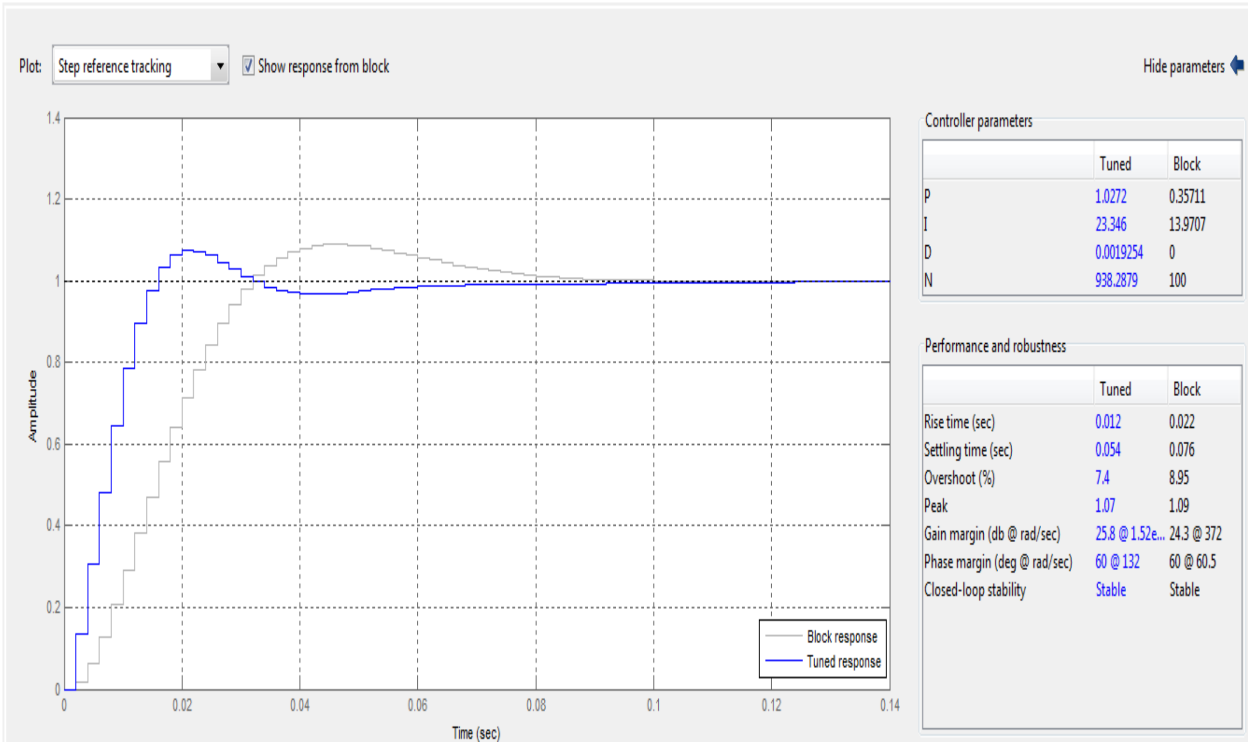


Figure 3-6 Simulink PID Block GUI

3.3.2 SYSTEM MODEL

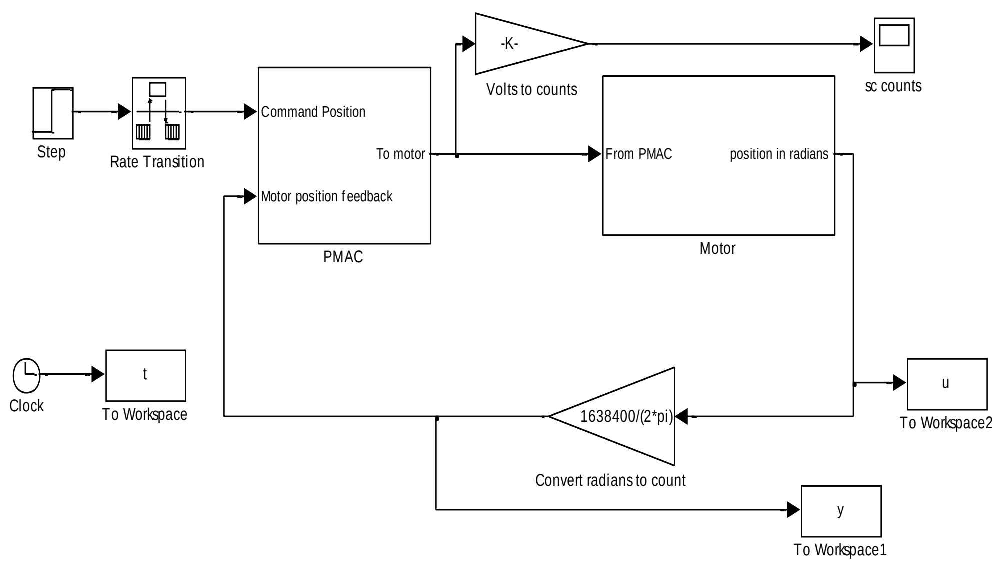


Figure 3-7 Model of the PMAC Controller + Motor

This is the model of the servo loop with PMAC PID Filter and Motor as subsystem blocks.

3.3.2.1 PMAC MODEL

This model is created with the help of the equations 3-1 to 3-9 and is a result of several attempts at modifying the PID Filter Diagram presented in PMAC documentation. Further, the I-variables used in the model were provided by GMRT. These I-variables were obtained by the PMAC tuning software at GMRT during several experiments and test procedures, for minimum positioning errors.

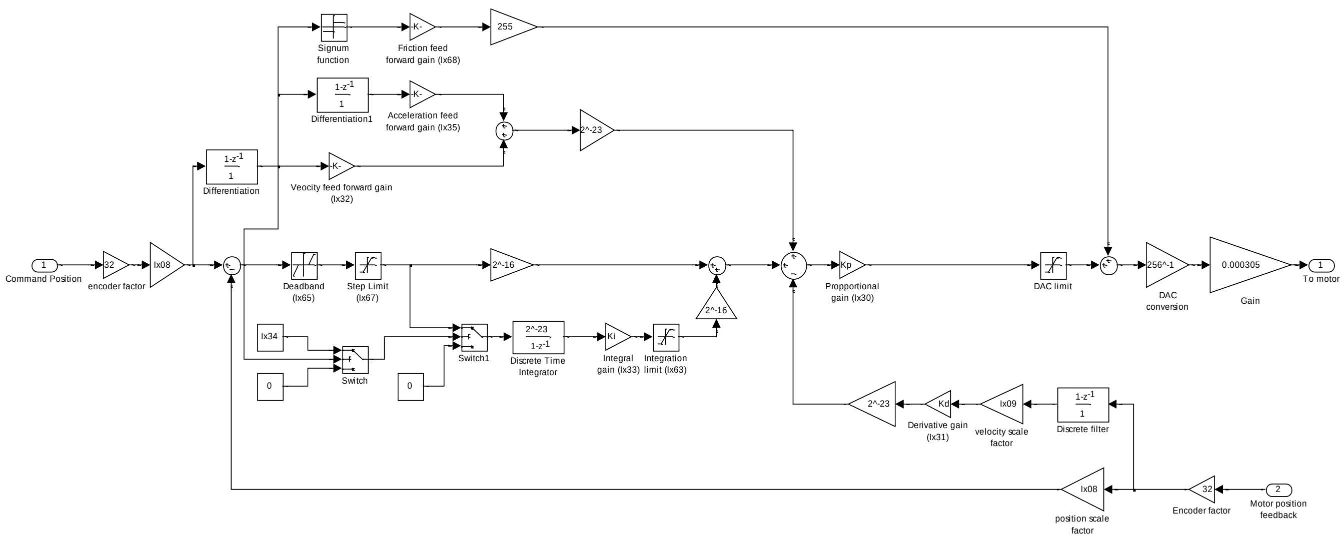


Figure 3-8 PMAC: The Controller Model

3.3.2.2 MOTOR WITH AMPLIFIER

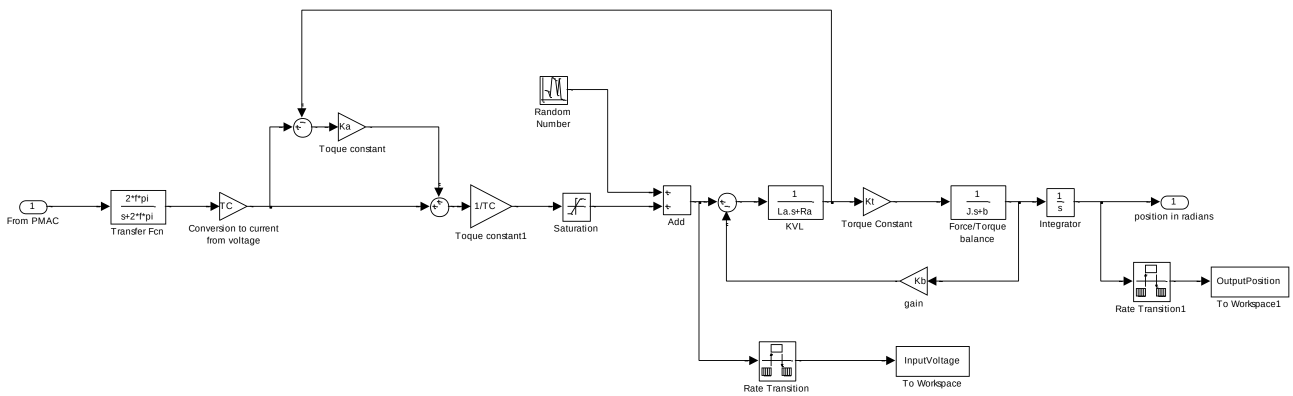


Figure 3-9 Motor and Amplifier Modeled

4 RESULTS AND DISCUSSION

4.1 SYSTEM RESPONSE

The system was simulated using Matlab/Simulink and the response was plotted for a step input of 10000 DAC counts. The 1st result was:

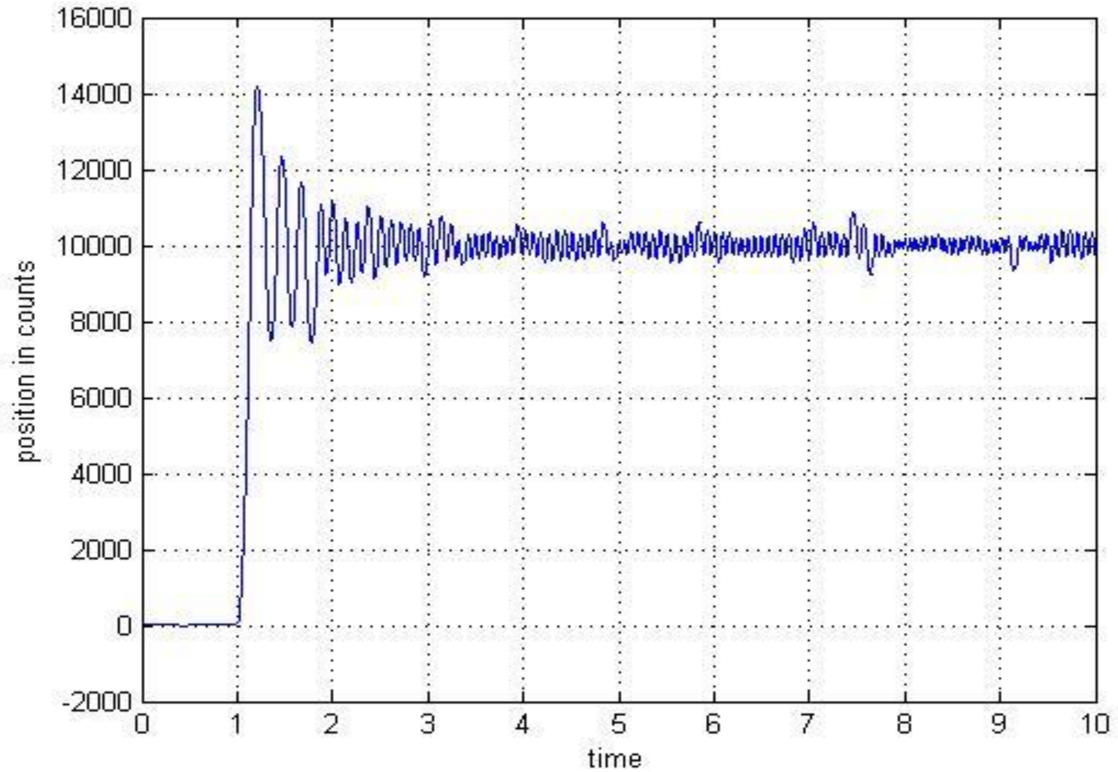


Figure 4-1 Step Response of the system model

This result contains some random oscillations. This response includes the effect of random number input to the motor along with the servo command. This explains the effect of random noise from the amplifier on the overall system.

The system is then simulated without adding the random noise to the motor. It gives the following response:

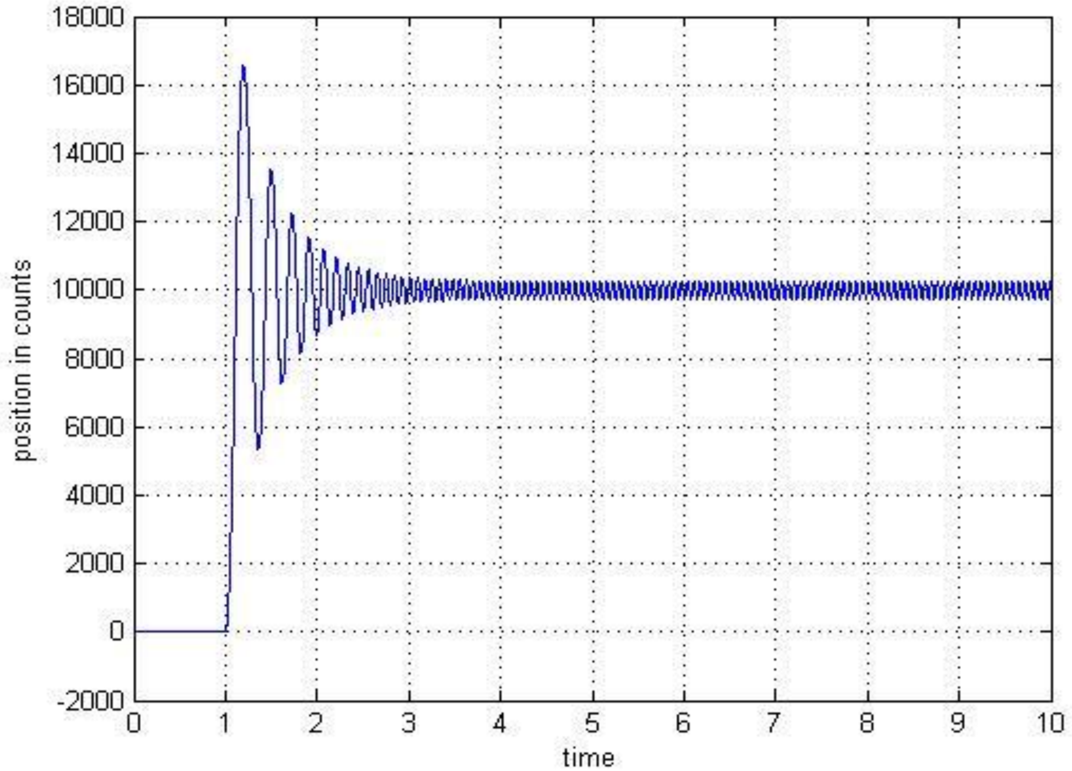


Figure 4-2 Step Response with no Random Noise Added to the Motor

Now, an idea comes in mind that the impulse response of a low pass filter is a sinc function. This could lead to a large overshoot in the transient response, however improving the frequency response. To check, the system is simulated again with the amplifier removed from the model:

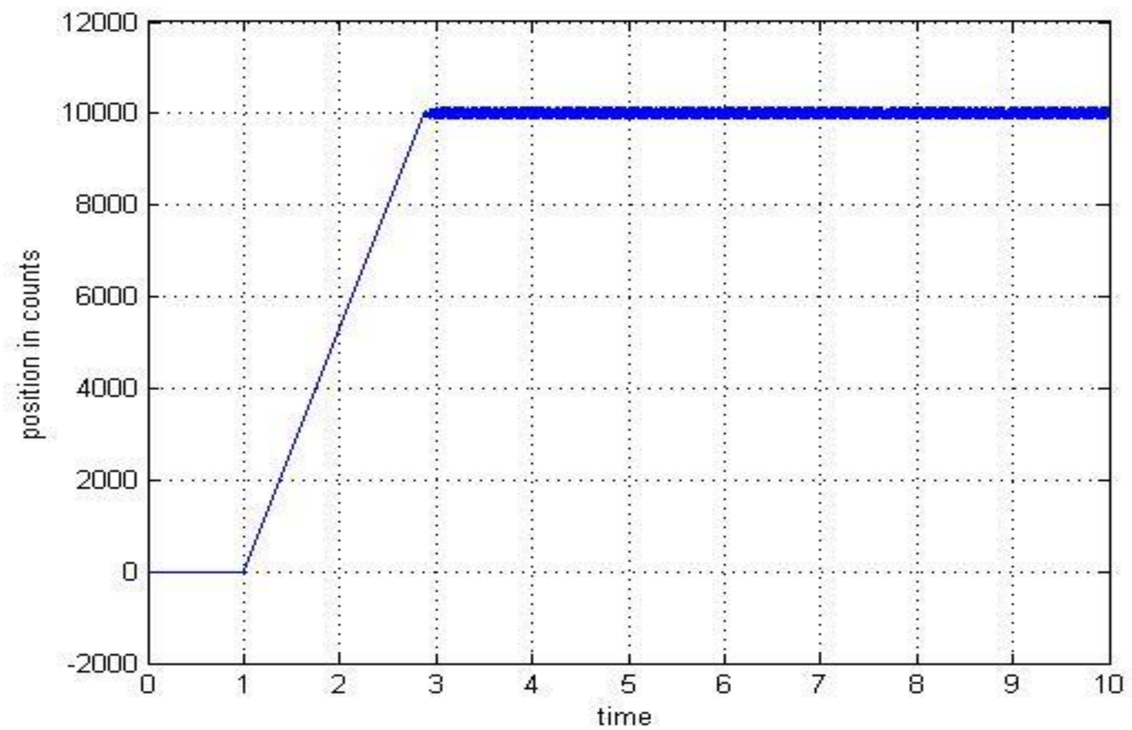


Figure 4-3 Step Response with Amplifier Dynamics Removed

Now, one can see that the idea was right. Now, there is almost no overshoot and steady state error is eliminated. So, no change is needed in Ki and Kd values. Also, we need not change the Kp gain value as we see the system is stiff enough and further increase in proportional gain will increase the sensitivity of the system towards noise and disturbances. Thus, this response is acceptable for the model.

4.2 PID VALUES PUBLISHED

The feedback and feed forward gain values used to get the response in Figure 4-3:

- Kfff=0
- Kaff=0
- Kp=300000
- Kd=7000
- Ki=10000
- Kvff=7000

5 CONCLUSIONS

- The mathematical modeling of the conventional DC motor and the brushless motor provides a good understanding of the overall motor dynamics.
- The motor is modeled as a second order dynamic system, and thus the concepts of a second order system can be conveniently applied for motor design purposes.
- The PMAC is modeled as a PID servo loop filter controller and effectively applied to improve the attributes of a brushless motor. The I-variables values used in the PMAC tuning software environment for practical applications are used in this model, and the results provided by the modeling agree with the experimental data. Also, no further fine tuning of the PID values are needed.
- The step response is studied with and without amplifier and noise. The response without amplifier dynamics in the model produced low overshoot and minimum errors. Also, adding random noise to the motor causes overshoot and random oscillations in the response, which is undesirable for the system.
- The graphs are plotted and studied.

6 RECOMMENDATIONS

The PMAC is a state of the art motion controller with uncountable attributes and functions it can perform. Model based approach is very convenient to fully understand a specific operation of such a class of devices, may it be control, commutations or any other function. It however requires a lot of exploration in the documents provided by the PMAC, and is also very time consuming for a person new to such devices. My experience from the project tells that once a working model is built and introduced, it can be further modified for several axes simultaneously, as everything in PMAC is interconnected and there is a lot of scope for improvements and findings.

A model based approach is convenient to implement when there is not enough experimental data known or when the work requires attention in several areas simultaneously. Regarding this project, the work has a scope for further progress by adding the plant (antenna) dynamics to the system, or by making the controller work with several motors in different axes simultaneously. The current results can of course be further improved with more time and expert knowledge.

7 APPENDICES

APPENDIX A: m-Files for Open Loop Motor Analysis

Constants.m

```
%start of code
% Characteristics parameters for modeling brushless motor AKM73M-KSC2R-02
R=0.68;                    %ohms
L=0.0124;                  %Henry
Kt=3.1;                    %Nm/A, Torque Constant
Ke=6/pi;                   %V/rad/s, Back emf Constant
J=0.0092;                  %kg.m^2, Rotor inertia
p=3;                       %no. of phases
```

Evaluatedconstants.m

```
%start of code
% Characteristics parameters for modeling brushless motor AKM73M-KSC2R-02
R=0.68;                    %ohms
L=0.0124;                  %Henry
Kt=3.1;                    %Nm/A, Torque Constant
Ke=6/pi;                   %V/rad/s, Back emf Constant
J=0.0092;                  %kg.m^2, Rotor inertia
p=3;                       %no. of phases
```

openloop.m

```
%start of code
%include constant parameters
constants
%include evaluated constants
evaluatedconstants
G=tf(1/Ke,[tm*te tm 1]);                    %transfer function
%plots the step response
figure;
step(G,0.5);
title('open loop step response');
xlabel('time,secs');
ylabel('voltage,volts');
grid on;
%plots the bode plot
figure;
bode(G);
title('open loop bode plot');
grid on;
figure;
GDig=c2d(G,0.5);
bode(G,'r',GDig,'b--');
title('open loop bode discretized');
grid on;
```

bldcwithbm.m

```
%include constant parameters
constants
%transfer function evaluation
Bm=0.0012;                                      %Nm/rad/s, viscous friction
constant
Gbl=tf(Kt,[J*L (R*J+Bm*L) (Kt*Ke+Bm*R)]);       %transfer finction
%plots the step response
figure;
step(Gbl,0.5);
title('open loop step response');
xlabel('time,secs');
ylabel('voltage,volts');
grid on;
%plots the bode plot
figure;
bode(Gbl);
title('open loop bode plot');
grid on;
```

APPENDIX B: m-Files for DC Motor Simulink Model

Parameters.m

```
%start of code
% Characteristics parameters for modeling
R=0.68;                  %ohms
p=3;                     %no.of phases
L=0.0124;                %Henry
Kt=3.1;                  %Nm/A, Torque Constant
Ke=6/pi;                 %V/rad/s, Back emf Constant
Bm=0.0012;               %Nm/rad/s, viscous friction constant
J=0.0092;                %kg.m^2, Rotor inertia
sim('dcmotor')           %call simulink model
plot(t,ang_rate)
title('dc motor response');
xlabel('time');
ylabel('angular rate');
grid on;
```

APPENDIX C: System Model m-File

Valuesparameters.m

```
%beginning of code
Ix08=96;                    %position scale factor
Kfff=0;                     %friction feed forward gain
Kaff=0;                     %acceleration feed forward gain
Kp=300000;                  %proportional gain
Kd=7000;                    %derivative gain
Ki=10000;                   %integral gain
Kvff=7000;                  %velocity feed forward gain
Ix34=0;
Ix09=48;                    %velocity scale factor
J=0.0092;
Kt=3.1;
Ke=6/pi;
Ra=0.68;
b=0.0012;
La=0.0124;
TC=0.5;
Ka=100;
f=523;
sim('FinalPMACnMotor');
plot(t,y);
xlabel('time');
ylabel('position in counts');
grid on
%end of code
```

8 REFERENCES

1. Harold Klee, *Simulation of Dynamic Systems with MATLAB and Simulink*, CRC Press, Taylor and Francis Group, 2007

2. User Manuals, *PMAC Users*, DELTA TAU, Delta Systems Inc.

3. Oludayo John Oguntoyinbo, *PID Control of Brushless DC Motor and Robot Trajectory Planning and Simulation with Matlab/Simulink*, Thesis Document,2009.

4. *www.Mathworks.in*

5. HANDBOOK OF PI AND PID CONTROLLER TUNING RULES (3rd Edition), *Aiden O'Dwyer* © Imperial College Press.