MODELING AND CONTROL OF THE THREE DEGREES-OF-FREEDOM PARALLEL MANIPULATED ROBOTIC SENSOR HEAD

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY
BY

MUHAMMED ORKUN ÖĞÜCÜ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

OCTOBER 2014

Approval of the thesis:

MODELING AND CONTROL OF THE THREE DEGREES-OF-FREEDOM PARALLEL MANIPULATED ROBOTIC SENSOR HEAD

submitted by **MUHAMMED ORKUN ÖĞÜCÜ** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Gülbin Dural Ünver
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Gönül Turhan Sayan
Head of Department, **Electrical and Electronics Engineering** _____

Assoc. Prof. Dr. Afşar Saranlı
Supervisor, **Electrical and Electronics Engineering** _____

Examining Committee Members:

Prof. Dr. Ömer Morgül
Electrical and Electronics Engineering Dept., Bilkent University _____

Assoc. Prof. Dr. Afşar Saranlı
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. Aydan Erkmen
Electrical and Electronics Engineering Dept., METU _____

Prof. Dr. M. Kemal Leblebicioğlu
Electrical and Electronics Engineering Dept., METU _____

Assist. Prof. Dr. A. Buğra Koku
Mechanical Engineering Dept., METU _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: MUHAMMED ORKUN ÖĞÜCÜ

Signature :

ABSTRACT

MODELING AND CONTROL OF THE THREE DEGREES-OF-FREEDOM PARALLEL MANIPULATED ROBOTIC SENSOR HEAD

ÖĞÜCÜ, Muhammed Orkun

Ph.D., Department of Electrical and Electronics Engineering

Supervisor : Assoc. Prof. Dr. Afşar Saranlı

October 2014, 155 pages

Legged robot platforms have distinct advantages over wheels in rough terrain and provide better mobility. Also in many applications, including military reconnaissance, disaster relief, hazardous site inspection, search and rescue applications benefit from the legged robots which is capable of moving safely at high speeds through rough natural terrain. However, with the increasing speed, fundamental difficulties like dynamic and mechanical limitations as well as control and computational limitations arise. Also sensor uncertainty and limited terrain data can result in unexpected and dangerous situations. At such situations, the data obtained by a single sensor may not be reliable. To provide certain and robust information, it is conventional that to integrate multiple sensor data which is known as sensor fusion. Gyroscopes and accelerometers are frequently used motion sensors at robotic applications. However, these sensors can only provide an accurate solution for a short period of time because of integration drift. To correct these bias errors, cameras are frequently used in robotics applications with integration of accelerometer and gyro data. However, vision sensors are considerably sensitive to shocks and instantaneous movements especially at high speed and accelerations resulted with motion blur effect. Due to the limited performance of deblurring methods, it is crucial to avoid the captured images affecting from visual disturbances.

In this thesis, head stabilization problem of SensoRHex mobile robot platform which is dynamically stable autonomous hexapod robot with six compliant legs will be examined. Thesis works can be categorized into two groups; Sensing and Control. Accelerometer and gyro-

scope data will be integrated with camera data according to the sensor fusion algorithms. These sensory data will be used to estimate state space vector partially. At the final stage of the sensing works is disturbance and signal characterization. Control strategy will be determined after the examination of disturbance characteristics. But there are intuitive ideas about the disturbances which presence of predictable cyclic characteristic due to the compliant legs and passive filters which the camera will be mounted on. In that case, model predictive control can be used as a controller structure. Anyhow, it would be a good challenge to compare the different control strategies according to the performance evaluation by considering the effects of passive filter parameters.

After selecting the control strategy, it can be attended to solve the main problem of thesis works; head stabilization. For this purpose, the primary objective is to maintain the state vector of the camera which attached to the head, as close as possible to the origin.

Keywords: Parallel Platform, Head Stabilization, Gun-Turret Stabilization, Line-of-Sight Stabilization, Gaze Stabilization, Legged Locomotion, RHex

ÖZ

ROBOTİK UYGULAMALARINDA ÜÇ SERBESTLİK DERECELİ ALGILAYICI STABİLİZASYON ÜNİTESİNİN MODELLENMESİ VE KONTROLÜ

ÖĞÜCÜ, Muhammed Orkun
Doktora, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi : Doç. Dr. Afşar Saranlı

Ekim 2014 , 155 sayfa

Bacaklı robotlar, arazi yapısının düzgün olmadığı koşullarda hareket kabiliyeti açısından tekerlekli robotlara göre daha üstün özellikler göstermektedir. Askeri uygulamalar, arama kurtarma çalışmaları gibi birçok alanda hızlı hareket edebilme kabiliyeti olan bacaklı robotlardan yararlanılmaktadır. Fakat robot platformalarının hareket hızlarının arttması, mekanik yapı ve kontrolcü tasarımı açısından önemli zorlukları beraberinde getirmektedir. Özellikle de robot platformu üzerinde bulunan algılayıcılar yüksek hızlarda önemli bozucu etkilerine maruz kalmakta ve bu durum algılayıcılardan alınan verilerin güvenilirliğini azaltmaktadır. Bu gibi durumlarda birden fazla kaynaktan alınan sensör verilerinin bir arada kullanılması ile daha güvenilir sonuçlar elde edilmektedir. Özellikle artımlı hata problemi bulunan doğrusal ve açısal ivme ölçerlerden alınan verilerin kamera verileri ile birleştirildiği birçok çalışma bulunmaktadır.

Algılayıcı verilerinin birleştirilmesi ile daha kesin sonuçların eldesi için kamera verileri sıklıkla tercih edilmesine rağmen, bu verilerin kamera hareketlerine oldukça duyarlı olduğu bilinmektedir. Bozulan kamera verileri, maruz kalınan kamera hareketinin tersi ile süzgeçlenerek belirli oranlarda düzeltilebilmektedir. Fakat bu yöntemlerin sahip olduğu çeşitli kısıtlar mevcuttur.

Bu tez çalışmasında, kamera verilerinin henüz toplanma aşamasında, görüntünün bozulmasına sebep olan kamera hareketlerinin asgari düzeye indirgenmesi için kullanılması planlanan

kafa stabilizasyonu problemi incelenecektir. Alınan sonuçlar ile yapılacak deneysel çalışmalar sırasında, SensoRHex esnek bacaklı robot platformunun kullanılması planlanmaktadır.

Bu tez çalışması, algılama ve kontrolcü tasarımı şeklinde iki alt başlıkta incelenebilir. Algılama kısmında, doğrusal ve açısal ivme ölçerlerden alınan veriler, kameradan alınan veriler ile birleştirilecek ve kısmi durum kestirimi problemi ele alınacaktır. Robot platformunun maruz kaldığı bozucu sinyal karakteristikleri incelenecektir.

Kontrolcü tasarımı kısmında ise, algılama kısmında edinilen bilgilerden yararlanarak, bozucu etkilerini enazlayacak, kameranın durum vektörünü, durum uzayının istenilen noktasına getirecek bir yapı önerilecektir. Tasarlanan kontrolcü, performans açısından, deneysel uygulamalarda sıkça kullanılan diğer kontrolcü yapıları ile karşılaştırılacaktır. Kontrolcü tasarımı sırasında dikkate alınan zaman ve frekans tanım bölgesi kriterleri ile, alınan kamera verisinin bozulum seviyesinin değerlendirilmesi için kullanılan hareket bulanıklığı arasındaki ilişkiler ortaya konacaktır.

Anahtar Kelimeler: Paralel Platform, Mekanik Kafa Stabilizasyonu, Bakış Açısı Stabilizasyonu, Bacaklı Lokomosyon, RHex

To my parents and little brother

Ali Öğücü, Özlem Öğücü and Mesut Öğücü

ACKNOWLEDGMENTS

I would like to express my special appreciation and thanks to my supervisor Assoc. Prof. Dr. Afşar Saranlı for his continuous support, guidance and friendship. I am very grateful for his motivation, enthusiasm and patience. It was a great honor to work with him for the last six years and our cooperation influenced my academical and world view highly.

A lot of people influenced and supported this work scientifically and their contribution were most valuable for me. I would like to thank Assist. Prof. Dr. Yiğit Yazıcıoğlu for his support and guidance. Assoc. Prof. Dr. Uluç Saranlı personally supplied a lot of important ideas for the real kick off of this work. Members of my thesis committee Professor Kemal Leblebicioğlu and Assist. Prof. Dr. Buğra Koku always gave valuable feedback for the progress of this work, and were not hesitant to warn me of the shortcomings or risks of my work. Professor Aydan Erkmen and Professor Ömer Morgül also provided valuable feedback for the future of this research, which I very appreciate.

This work is also supported by TÜBİTAK-BİDEB PhD scholarship (2211) and TÜBİTAK-ARDEB Scientific and Technological Research Project Program (1001) (Project No: #110E120).

There are a lot of people that were with me in these six years. I would like to thank to all ROLAB members.

And my family also provided invaluable support for this work. I would like to thank specially to my brother Mesut Öğücü. He always make me feel loved and cared. I am also thankful for all the love and support by Ali Öğücü and Özlem Öğücü.

TABLE OF CONTENTS

ABSTRACT v

ÖZ vii

ACKNOWLEDGMENTS x

TABLE OF CONTENTS xi

LIST OF TABLES xvii

LIST OF FIGURES xix

LIST OF ABBREVIATIONS xxviii

CHAPTERS

1 INTRODUCTION 1

1.1 Motivation 1

1.2 Literature Survey 4

1.3 Contributions 6

2 CAMERA STABILIZATION MECHANISM & CONSTITUENTS 9

2.1 PC104 & Data Acquisition Card 9

2.2 Linear DC Servomotor & Motor Driver 9

2.2.1 Design of Experiments 12

2.2.2 Position Control Experiments at No-Load Operation . . . 13

2.2.2.1 Processing and Communicating Delays 13

2.2.2.2 Normal and Analog Modes 15
2.2.2.3 Data Request Frequency 17
2.2.2.4 Motion Driver Internal Parameters 18
2.2.3 Velocity Control Experiments at No-Load Operation . . . 21
2.3 Hall Effect Sensor 22
2.4 Incremental Rotary Encoder 25
2.5 Inertial Measurement Unit 26
2.6 Vision Sensor 27
3 MODELING EFFORTS OF ACTUATING MECHANISM 29
3.1 Basic Concepts 29
3.2 Three-Phase Brushless Linear DC Motor Model & One-Phase Equivalent Circuit 31
3.3 System Identification for Linear DC Servomotor 36
3.3.1 Experimental Measurements & Frequency Spectrum Analysis 36
3.3.2 Model Based Linear Identification 38
3.3.2.1 Transfer Function 38
3.3.3 Data Driven Linear Identification 40
3.3.3.1 Transfer Function 40
3.3.3.2 Polynomial Model 41
3.3.4 Data Driven Nonlinear Identification 42
3.3.4.1 Polynomial Model 42
3.3.4.2 Hammerstein-Wiener Model 44
3.3.5 Model Based Nonlinear Identification 45

3.3.5.1 Stiction Effect and Piecewise Linear Coulomb & Viscous Friction Coefficients 45

3.3.5.2 Static Input Nonlinearity 46

3.4 System Identification for Attached Motor Driver & Linear DC Servomotor 48

4 MODELING EFFORTS OF MECHANICAL STRUCTURE 49

4.1 Planar Layout Equipped with Revolute Joints 50

4.2 Parallel Layout Equipped with Spherical Joint 51

4.3 Symmetric Layout Equipped with Spherical Joint 53

4.4 Parallel Layout Equipped with Universal Joint 54

4.5 Symmetric Layout Equipped with Universal Joint 55

5 MODELING EFFORTS OF VISION SYSTEM 57

5.1 Motion Blur 58

5.2 Mathematical Model of Imaging System 59

5.2.1 Pin-Hole Geometry 59

5.2.2 Intrinsic & Extrinsic Parameters 61

5.2.3 Lens Imperfectness: Distortion 62

5.2.4 Realistic Parameters: Calibration Results 63

5.3 Simulation Model of Imaging System: Virtual Camera 64

5.3.1 Design of 3D Scene 64

5.3.2 Implementation of Camera Model 65

5.3.3 Calculation of Motion Blur 65

5.3.4 Formation of Ground Truth & Motion Blur Imposed Images 66

5.4 Simulation Results 67

5.4.1 Comparison of Motion Blur and Other Blur Metrics 68

5.4.2 Effect of Distance under Pure Translational and Pure Rotational Motion 69

5.4.3 Effect of Decoupled & Coupled Motions 69

5.4.4 Effect of Shutter Speed 70

5.4.5 Effect of Intrinsic Camera Parameters 71

6 CONTROL APPROACHES FOR CAMERA STABILIZATION PLATFORM 73

6.1 Existing Controllers 76

6.1.1 Model-Free Controllers 76

6.1.1.1 Proportional-Integral-Derivative Controller . . 76

6.1.1.2 Sliding Mode Controller 80

6.1.1.3 Proxy-Based Sliding Mode Controller 83

6.1.1.4 Fuzzy Logic Controller 85

6.1.1.5 Artificial Neural Network Controller 87

6.1.2 Model-Based Controllers 89

6.1.2.1 Linear Quadratic Tracker 89

6.1.2.2 Model Predictive Controller 93

6.2 Existing Predictors 97

6.2.1 Smith Predictor 97

6.2.2 Polynomial-Based Predictor 98

6.3 Proposed Approach: ARX-Based Variable Phase Reference Predictor Combined with Operational Controller 100

7 EXPERIMENTAL STUDIES FOR CAMERA STABILIZATION PLATFORM 101

7.1 Simulation Experiments-I: Pulse Responses of Idealized Model Controllers 101

7.1.1 Single-Loop PID Controller 102
7.1.2 Double-Loop PD+PI Controller 104
7.1.3 Sliding Mode Controller 106
7.1.4 Proxy-Based Sliding Mode Controller 108
7.1.5 Fuzzy Logic Controller 110
7.1.6 Artificial Neural Network Controller 112
7.1.7 Linear Quadratic Tracker 114
7.1.8 Model Predictive Controller 116
7.1.9 Summary 118
7.2 Simulation Experiments-II: Pulse Responses of Realistic Model Controllers 118
7.2.1 Double-Loop PD+PI Controller 119
7.2.2 Double-Loop SMC+PI Controller 120
7.2.3 Double-Loop LQT+I+PI Controller 122
7.2.4 Double-Loop MPC+PI Controller 123
7.3 Physical Experiments: Tracking Performance of Proposed Predictor & Selected Controller 125
REFERENCES 127
APPENDICES
A TECHNICAL SPECIFICATIONS OF THE ACTUATING MECHANISM . . 133
A.1 Faulhaber Quickshaft LM1247 Series Linear DC Servomotor 134
A.2 Faulhaber MCLM 3006S Series External Motion Driver 136
B MECHANICAL COMPONENTS OF THE CAMERA STABILIZATION PLATFORM 143

B.1 3D Computer-Aided Design (CAD) Model of the Components . . . 143

B.1.1 Base Platform . . . 144

B.1.2 Top Platform . . . 145

B.1.3 Vertical Support Column . . . 146

B.1.4 Bearing Housing . . . 148

B.1.5 Cylindrical Housing . . . 150

B.1.6 Motor Connector Support . . . 151

B.2 Spherical Rod End Joint . . . 152

B.3 Universal Joint . . . 153

CURRICULUM VITAE . . . 155

LIST OF TABLES

TABLES

Table 2.1 Query and programmable parameters of the MCLM 3006S external motion driver 11

Table 2.2 Performance Specifications at Standalone, Non-Real Time and Real-Time OS 17

Table 2.3 Default and modified motion driver internal parameters 21

Table 3.1 Factory specified and optimized parameters for Faulhaber LM1247 DC servomotor 40

Table 4.1 The top platform's range of motion about roll, yaw and pitch axes for different configurations 49

Table 4.2 The relationship between rotational motion of the top platform and linear motion of the motor shafts for parallel layout 52

Table 4.3 The relationship between rotational motion of the top platform and linear motion of the motor shafts for symmetric layout 53

Table 5.1 The computed intrinsic parameters by using Caltech Camera Calibration Toolbox for various types of camera models 64

Table 5.2 The computed intrinsic parameters by using Caltech Camera Calibration Toolbox for various types of camera models 71

Table 7.1 Time-domain performance specifications obtained by using various controller structures implemented for idealized model of camera stabilization platform 118

Table B.1 Dimension, mass and inertia properties of the base platform 144

Table B.2 Dimension, mass and inertia properties of the top platform 145

Table B.3 Dimension, mass and inertia properties of the support column for single-axis stabilization platform 146

Table B.4 Dimension, mass and inertia properties of the support column for three-axis stabilization platform 147

Table B.5 Dimension, mass and inertia properties of the bearing housing for single-axis stabilization platform 148

Table B.6 Dimension, mass and inertia properties of the bearing housing for three-axis stabilization platform 149

Table B.7 Dimension, mass and inertia properties of the cylindrical housing 150

Table B.8 Dimension, mass and inertia properties of the motor connector support . . . 151

LIST OF FIGURES

FIGURES

Figure 1.1 A selection of the vision sensor application areas 2

Figure 1.2 Two frames of the video sequence from the camera attached to the SensoRHex a) Relatively sharp image at stationary state b) Motion blur imposed image at walking [69] 3

Figure 1.3 Categorization of thesis works: sensing and control 7

Figure 2.1 Diamond System Athena II PC104 & Data Acquisition Card 9

Figure 2.2 a) LM1247 linear DC servomotor b) MCLM 3006S external motion driver 10

Figure 2.3 Schematic diagram of the external motion driver, MCLM 3006S, connected to the three-phase linear servomotor, LM1247, through analog Hall sensors 10

Figure 2.4 Experimental setup for no-load operation of actuating mechansim 12

Figure 2.5 Schematic diagram of experimental setup 12

Figure 2.6 Desired and measured shaft positions at stand alone operating condition. Reference shaft position is a pulse input with amplitude of 10 [mm] and frequency of 0.85, 1.30, 2.67 and 5.48 [Hz], respectively 14

Figure 2.7 Desired and measured shaft positions under non-real time operating system. Reference shaft position is a pulse input with amplitude of 10 [mm] and frequency of 0.85, 1.30, 2.67 and 5.48 [Hz], respectively 15

Figure 2.8 Desired and measured shaft positions under real-time operating system. Reference shaft position is a pulse input with amplitude of 10 [mm] and frequency of 0.85, 1.30, 2.67 and 5.48 [Hz], respectively 16

Figure 2.9 Desired and measured shaft positions for normal and analog operation modes. Reference shaft position is a pulse input with amplitude of 10 [mm] and frequency of 0.85, 1.30, 2.67 and 5.48 [Hz], respectively 18

Figure 2.10 Desired and measured shaft positions for various data requesting rates. a) Reference Period = 1 [s], b) Reference Period = 0.5 [s], c) Reference Period = 0.2 [s] 19

Figure 2.11 The variation of RMSE, defined between desired and measured shaft positions, with respect to the proportional (PP) and derivative (PD) gains of PD-type position controller 20

Figure 2.12 The variation of RMSE, defined between desired and measured shaft positions, with respect to the proportional (POR) and derivative (I) gains of PI-type velocity controller 20

Figure 2.13 The variation of RMSE with respect to the a) maximum acceleration limit (ACC), b) maximum deceleration limit (DEC), c) maximum velocity limit (SP) . . 21

Figure 2.14 Variation of measured velocity, defined as the maximum value of each experiment, with respect to the desired velocity 22

Figure 2.15 Schematic diagram of Faulhaber LM1247-020-10 linear DC servomotor equipped with three analog hall effect sensors 23

Figure 2.16 The normalized Hall effect sensor outputs in [V] with respect to the motor shaft position. At each step, 50 measurements of the HallA, HallB and HallC are plotted as red, green and blue, respectively. Also, the averages of them are indicated by black. 24

Figure 2.17 The desired, measured and estimated shaft positions. The measured position is indicated with blue and read through serial port interface at 50 [Hz]. The estimated position is indicated with red and calculated by using our proposed model at 1 [kHz] 25

Figure 2.18 Autonics E50S8-5000 Incremental Rotary Encoder 26

Figure 2.19 Inertial Measurement Units a) MicroStrain 3DM-GX1 b) IMU3000 Combo 26

Figure 2.20 The attitude estimation for MicroStrain 3DM-GX1 and IMU3000 Combo by using a) acceleration, b) angular rate data 27

Figure 2.21 Logitech C310 Series Webcam 27

Figure 3.1 Exact simulation model for electronically commutated three-phase brushless linear DC servomotor with Hall effect sensors 33

Figure 3.2 Single-phase equivalent circuit simulation model for electronically commutated three-phase brushless linear DC servomotor 34

Figure 3.3 Simulated shaft position according to the various voltage signals for three-phase exact model and single-phase equivalent circuit model 35

Figure 3.4 Classification of system identification techniques depending on whether linear or nonlinear, as well as model based or data driven 36

Figure 3.5 Training set, which consist of input voltage (left) and output shaft position (right) gathered from Faulhaber LM1247 brushless linear DC servomotor. The data, which has similar characteristics grouped together. 37

Figure 3.6 Validation set, which consist of input voltage (left) and output shaft position (right) gathered from Faulhaber LM1247 brushless linear DC servomotor. The data, which has similar characteristics grouped together. 37

Figure 3.7 Single-sided power spectrum of training and validation datasets 38

Figure 3.8 Measured and simulated shaft positions. The simulated results are obtained by substituting factory specified motor parameters into derived transfer function. . 39

Figure 3.9 Measured and simulated shaft positions. The simulated results are obtained by substituting optimized motor parameters into derived transfer function. 39

Figure 3.10 Measured and simulated shaft positions. The simulated results are obtained by using data driven 7^{th} order transfer function. 41

Figure 3.11 Measured and simulated shaft positions. The simulated results are obtained by using the identified polynomial model. 42

Figure 3.12 Structure of nonlinear autoregresive with external input (NARX) model . . 43

Figure 3.13 Measured and simulated shaft positions. The simulated results are obtained by using the identified nonlinear ARX model. 43

Figure 3.14 Structure of nonlinear Hammerstein-Wiener model 44

Figure 3.15 Measured and simulated shaft positions. The simulated results are obtained by using the identified hammerstein-wiener model. 44

Figure 3.16 Structure of multifaceted model with stiction effect and piecewise linear coulomb & viscous friction coefficients 45

Figure 3.17 Measured and simulated shaft positions. The simulated results are obtained by using multifaceted friction model. 46

Figure 3.18 Structure of static input nonlinearity and single-phase equivalent circuit model with factory specified parameters 47

Figure 3.19 Measured and simulated shaft positions. The simulated results are obtained by using the static input nonlinearity and single-phase equivalent circuit model. . 47

Figure 3.20 Measured and simulated shaft positions. The simulated results are obtained by using the static input nonlinearity and single-phase equivalent circuit model. . 48

Figure 3.21 Measured and simulated shaft positions. The simulated results are obtained by using the static input nonlinearity and single-phase equivalent circuit model. . 48

Figure 4.1 SimMechanics model of the planar layout equipped with revolute joints . . 50

Figure 4.2 SimMechanics model of the parallel positioned links with spherical joints . 52

Figure 4.3 SimMechanics model of the symmetric positioned links with spherical joints 53

Figure 4.4 SimMechanics model of the parallel positioned links with universal joints . 54

Figure 4.5 SimMechanics model of the symmetric positioned links with universal joints 55

Figure 5.1 Motion blur caused by a) object movement b) rotational camera movement 58

Figure 5.2 3D to 2D one-to-multiple mapping due to the non-infinitesimal integration time 58

Figure 5.3 a) Original image. Simulated motion blur effect by imposing b) circular motion c) horizontal motion d) vertical motion [52] 59

Figure 5.4 Perspective projection [18] 60

Figure 5.5 Projected (ideal) and distorted image points [16] 62

Figure 5.6 a) Calibration images taken by Point Grey Flea 2 CCD camera for chessboard-type grid pattern b) Reproduced calibration grids [8] 63

Figure 5.7 Designed 3D scene includes a static object and mobile camera 64

Figure 5.8 a) The duration of frame and exposure instants are shown by red and blue lines. b) The magnitude of the centroid velocity and sampled points are given in [pixel/s]. 66

Figure 5.9 The scene layout and generated ground truth synthetic images 67

Figure 5.10 Motion blur imposed images that are calculated by using linear combination of the ground truth images generated within the same exposure time interval. Motion is from right to left, and from top to bottom 67

Figure 5.11 Comparison of perceptual blur [12] and anisotropic quality index [19] calculated by using synthetic images and motion blur calculations obtained by virtual camera model 68

Figure 5.12 The variation of motion blur at various distance between stationary object and mobile virtual camera a) under pure translational motion b) under pure rotational motion 69

Figure 5.13 The amount of blur imposed by a) only translation motion, b) only rotational motion c) translational and rotational coupled motion 70

Figure 5.14 The variation of motion blur for different shutter speed values given in [1/s] 71

Figure 5.15 The variation of motion blur for different camera models under coupled movements of virtual camera along x-axis and about x-y-z axes 72

Figure 6.1 Physical system includes a plant, a single-board computer and other peripherals like serial port and data acquisition board 73

Figure 6.2 Idealized plant model and designed controller structures 75

Figure 6.3 Realistic plant model and designed controller structures 75

Figure 6.4 Block diagrams of single-loop a) fixed gain, b) variable gain PID controllers 77

Figure 6.5 Block diagram of double-loop decoupled PD+PI controller 78

Figure 6.6 Block diagram of double-loop coupled PD+PI controller 78

Figure 6.7 Block diagram of discrete PD controller implemented for realistic model . 79

Figure 6.8 Phase portraits of an unstable second order linear system compensated with a) positive feedback b) negative feedback and c) variable structure control [68] . . 80

Figure 6.9 Block diagram of sliding mode controller implemented for idealized model 82

Figure 6.10 Block diagram of sliding mode controller implemented for realistic model . 82

Figure 6.11 The physical interpretation of the proxy-based sliding mode control [36] . 83

Figure 6.12 Block diagram of proxy-based sliding mode controller $(M > 0)$ 84

Figure 6.13 Block diagram of proxy-based sliding mode controller $(M = 0)$ 84

Figure 6.14 Block diagram of fuzzy controller within a closed loop application [55] . . 85

Figure 6.15 Block diagram of fuzzy logic controller 86

Figure 6.16 Triangular shaped optimized membership functions for a) position, b) derivative of position, c) force, as well as d) control surface 86

Figure 6.17 Structure of a single abstract neuron with n inputs 87

Figure 6.18 Architecture of 3-inputs-2-outputs multilayer artificial neural network . . . 87

Figure 6.19 Block diagram of neural network controller implemented for idealized model 88

Figure 6.20 Block diagram of linear quadratic regulator (LQR) operated in a closed-loop system illustrated with state (A), input (B), output (C) and feedthrough (D) matrices 89

Figure 6.21 Block diagram of linear quadratic tracker implemented for idealized model 90

Figure 6.22 Block diagram of modified linear quadratic tracker supplemented with integral action and designed over augmented model 91

Figure 6.23 Block diagram of linear quadratic tracker with integral action and attached state observer implemented for realistic model 93

Figure 6.24 Block diagram of model predictive controller 94

Figure 6.25 Predicted plant output and optimized sequence of control input a) at time t, b) at time $t + 1$ [7] 95

Figure 6.26 Block diagram of model predictive controller implemented for idealized model 96

Figure 6.27 Block diagram of Smith predictor embedded in a closed-loop control application with a generic compensator and process with known delay 97

Figure 6.28 Equivalent block diagram with operational Smith predictor 98

Figure 6.29 Time history of input and output signals for a hydraulic actuator [29] . . . 98

Figure 6.30 Time history of input and output signals for a hydraulic actuator [29] . . . 99

Figure 6.31 Block diagram of ARX-Based variable phase reference predictor 100

Figure 7.1 Idealized plant model and designed controller structures 101

Figure 7.2 Fixed-gain and variable-gain single loop PID controllers simulated at roll axis 102

Figure 7.3 Fixed-gain and variable-gain single loop PID controllers simulated at yaw axis 103

Figure 7.4 Fixed-gain and variable-gain single loop PID controllers simulated at pitch axis 103

Figure 7.5 Double-loop decoupled and coupled PID controllers simulated at roll axis . 104

Figure 7.6 Double-loop decoupled and coupled PID controllers simulated at yaw axis 105

Figure 7.7 Double-loop decoupled and coupled PID controllers simulated at pitch axis 105

Figure 7.8 Sliding mode controller with sign & saturation functions simulated at roll axis 106

Figure 7.9 Sliding mode controller with sign & saturation functions simulated at yaw axis 107

Figure 7.10 Sliding mode controller with sign & saturation functions simulated at pitch axis 107

Figure 7.11 Proxy-based sliding mode controller for different masses simulated at roll axis 108

Figure 7.12 Proxy-based sliding mode controller for different masses simulated at yaw axis 109

Figure 7.13 Proxy-based sliding mode controller for different masses simulated at pitch axis 109

Figure 7.14 Fuzzy logic controller for two different numbers of MF simulated at roll axis 110

Figure 7.15 Fuzzy logic controller for two different numbers of MF simulated at yaw axis 111

Figure 7.16 Fuzzy logic controller for two different numbers of MF simulated at pitch axis 111

Figure 7.17 Artificial neural network controller simulated at roll axis 112

Figure 7.18 Artificial neural network controller simulated at yaw axis 113

Figure 7.19 Artificial neural network controller simulated at pitch axis 113

Figure 7.20 Linear quadratic tracker without and with integral action simulated at roll axis 114

Figure 7.21 Linear quadratic tracker without and with integral action simulated at yaw axis 115

Figure 7.22 Linear quadratic tracker without and with integral action simulated at pitch axis 115

Figure 7.23 Model predictive controller simulated at roll axis 116

Figure 7.24 Model predictive controller simulated at yaw axis 117

Figure 7.25 Model predictive controller simulated at pitch axis 117

Figure 7.26 Realistic plant model and designed controller structures 118

Figure 7.27 Proportional integral derivative controller & Smith predictor based variant simulated at roll axis 119

Figure 7.28 Proportional integral derivative controller & Smith predictor based variant simulated at yaw axis 119

Figure 7.29 Proportional integral derivative controller & Smith predictor based variant simulated at pitch axis 120

Figure 7.30 Sliding mode controller & Smith-predictor based variant simulated at roll axis 120

Figure 7.31 Sliding mode controller & Smith-predictor based variant simulated at yaw axis 121

Figure 7.32 Sliding mode controller & Smith-predictor based variant simulated at pitch axis 121

Figure 7.33 Discrete linear quadratic gaussian tracker & Smith-predictor based variant simulated at roll axis 122

Figure 7.34 Discrete linear quadratic gaussian tracker & Smith-predictor based variant simulated at yaw axis 122

Figure 7.35 Discrete linear quadratic gaussian tracker & Smith-predictor based variant simulated at pitch axis 123

Figure 7.36 Model predictive controller simulated at roll axis 123

Figure 7.37 Model predictive controller simulated at yaw axis 124

Figure 7.38 Model predictive controller simulated at pitch axis 124

Figure 7.39 Sliding mode controller roll axis movement 125

Figure 7.40 Sliding mode controller yaw axis movement 125

Figure 7.41 Sliding mode controller pitch axis movement 126

Figure B.1 The CAD model of the base platform 144

Figure B.2 The CAD model of the top platform 145
Figure B.3 The CAD model of the support column for single-axis stabilization platform 146
Figure B.4 The CAD model of the support column for three-axis stabilization platform 147
Figure B.5 The CAD model of the bearing housing for single-axis stabilization platform 148
Figure B.6 The CAD model of the bearing housing for three-axis stabilization platform 149
Figure B.7 The CAD model of the cylindrical housing 150
Figure B.8 The CAD model of the motor connector support 151

LIST OF ABBREVIATIONS

PP	Position Controller Proportional Gain
POR	Velocity Controller Proportional Gain
I	Velocity Controller Integral Gain
CI	Current Controller Integral Gain
LPC	Peak Current Limit
LCC	Continuous Current Limit
ACC	Maximum Acceleration Limit
DEC	Maximum Deceleration Limit
SP	Maximum Speed Limit
AUV	Autonomous Underwater Vehicles
ROV	Remotely Operated Vehicles
UAV	Unmanned Aerial Vehicles
ISP	Inertially Stabilized Platform
LOS	Line of Sight
FOV	Field of View
IOPI	Integer Order Proportional Integral
FOPI	Fractional Order Proportional Integral
LIM	Linear Induction Motor
LSM	Linear Synchronous Motor
PMLSM	Permanent Magnet Linear Synchronous Motor
EMF	Electromotive Force
PSF	Point-Spread Function
DLT	Direct Linear Transform
IMU	Inertial Measurement Unit
PID	Proportional-Integral-Derivative
VSC	Variable Structure Control
SMC	Sliding Mode Control
PBSMC	Proxy-Based Sliding Mode Control
FLC	Fuzzy Logic Control
MF	Membership Function
ANNC	Artificial Neural Network Control
LQR	Linear Quadratic Regulator

LQT	Linear Quadratic Tracker
LQT+I	Linear Quadratic Tracker with Integral Action
LQG	Linear Quadratic Gaussian
MPC	Model Predictictive Control

CHAPTER 1

INTRODUCTION

1.1 Motivation

In the age of supercomputers, as machines are humanized, the metals and plastics are evolving into the bones and skin, softwares are evolving into the artificial nervous systems. Nowadays, many of capabilities that human have, can be mimicked by machines, as is vision. The lexical meaning of the visual system in humans is defined as the ability to interpret the surrounding environment by processing information that is contained in visible light, which is more or less true for machines also. Merely, eyes are replaced by cameras and sights are replaced by images. Eventually, it is not surprising to see a mechanical eye instead of us at where we want to see something. From small scale on board cameras that placed on the head of the humanoid robots to large scale military purposed cameras placed on unmanned aerial vehicles (UAV), vision sensors are at everywhere. As illustrated in Figure 1.1, a few examples can be given;

- Aviation: Iris control for plots, aerial photographs for media, landscape images for surveillance, inspection of bridges, topographic modeling, recon mission and target monitoring for UAV's with live image transmission to control station
- Industrial Microscopy: Quality control, surface inspection, object verification, completeness check or error detection.
- Marine: Inspection of underwater pipelines, geological recording of ocean floors, mine detection, monitoring of ports, piers, shipping lanes, moorings and offshore areas.
- Robotics: Disaster relief, hazardous site inspection, search and rescue, object recognition, localisation, navigation, visual servoing, pose estimation and sensor fusion.
- Security: Vicinity surveillance at particularly poorly guarded or dangerous fields.
- Spectroscopy: Light measurement, LED characterization, quality control for foodstuffs, pharmaceutical and cosmetics inspection, recycling and precision farming.
- Traffic: Red-light surveillance, speed surveillance, tunnel surveillance, and driver assistance.



Figure 1.1: A selection of the vision sensor application areas

Undoubtedly, each application has its own challenges. Even in a fully controlled indoor application with exactly known lighting conditions, small differences in illumination can cause problems. Moreover, equipments may become inoperable at unmanageable environmental conditions such as glistening sunlight, dusk, foggy, misty, rainy or snowy weather conditions, turbid water with low visibility distance, as well as uncertain conditions such as dark or limited terrain data. Besides, process may become even more complicated depending on whether the camera and/or the object of interest is stationary or not. Also, mobility requirement at high speed is another factor that fuzzifies the features and reduces the information that is contained in the image. Sensor uncertainty and limited environment/terrain data arising from these conditions may cause unexpected and dangerous situations [32]. In the circumstances, the data obtained by a single sensor may not be reliable.

To provide certain and robust information, it is conventional that to integrate multiple sensor data which is known as sensor fusion. Sensor fusion is the integration of information from different types of sensor into a unified interpretation [26]. The resulting information obtained by sensor fusion process is more accurate, more complete and more dependable than would be possible when these sources were used individually. For example, motion sensors (accelerometers) and rotation sensors (gyroscopes) continuously calculate via dead reckoning the position, orientation, and velocity of a moving object without the need for external references. However, these sensors can only provide an accurate solution for a short period of time because of integration drift. Small errors in the measurement of acceleration and angular

velocity are integrated into progressively larger errors in velocity, which are compounded into still greater errors in position. Since the new position is calculated from the previous calculated position and the measured acceleration and angular velocity, these errors cumulatively increase at a rate roughly proportional to the time. Therefore the position must be periodically corrected by input from other type of sensors. In literature, this topic is examined in details and there are a lot of works related with correcting accelerometer and gyro outputs by other type of sensor data [51, 4]. Furthermore, sensor fusion studies based on integration of accelerometer and gyro data with vision sensors is attended to obtain more robust and accurate results [31, 64, 30, 21, 41].

As seen from the related literature, camera is frequently used with the integration of accelerometer and gyro data. However, vision sensors are considerably sensitive to shocks and instantaneous movements especially at high speed and accelerations. At our laboratory, as shown in Figure 1.2, an experimental study conducted on the highly mobile, dynamically stable six-legged RHex robot platform [37] revealed that the performance of the feature detectors (i.e., Canny edge detection [9] and Harris corner detection [23, 63] algorithms) degrades with increasing velocity values. Obtained results showed that at various normalized velocity values which measured as 0.1, 0.4, 0.8 and 1.0, respectively, every 2, 16, 28 and 60 of 100 frames can not be included to feature detection algorithms due to the intolerable motion blur effect caused by visual disturbances [69].



Figure 1.2: Two frames of the video sequence from the camera attached to the SensoRHex a) Relatively sharp image at stationary state b) Motion blur imposed image at walking [69]

Motion blur is the result of the relative motion between the camera and the scene over the period of exposure determined by the shutter speed. In an image, projection of any point in the scene moving with respect to the camera will be a certain curve on the sensor rather than a point. Thus it will look blurred or smeared along the direction of relative motion. In literature, there are a lot of recovering techniques for the images that lose information due to the degradation. For example, video stabilization studied at [47, 49] deals with repositioning the object of interest, so that it looks like fixed within the frame. Actually, this method does not increase the sharpness of the scene or does not try to enhance the degraded image, instead it maximizes the traceability of the objects in an video sequence viewed by the human eye.

Indeed it is possible to grade the quality of the image at some extent by applying the inverse of the blur effect imposed by the camera motion. For example, Blind Deconvolution method studied by Richardson-Lucy [58, 43], Kundur and Hatzinakos [38], as well as Gonzalez and Woods [20], and Hansen [22] apply this technique to blurred images. However, these studies only deal with a single frame at each instant which is not sufficient to calculate the deconvolution function exactly and uniquely.

The truth of the matter is that to completely determine the blur effect, thusly inverse filter for deconvolution operation, the relative motion between camera and scene have to be known. The studies [6, 52] based on this idea, estimate the motion by using optical flow acquired from successive video frames. Once the motion is determined, the blur effect, as well as inverse filter is obtained and deconvolution operation is carried out. However, they do not deal with loss of information which is the basic problem of inverse-filtering. For eliminating the zeros of inverse-filter, Agrawal and Raskar [1] use variable exposure time for successive frames and achieve to calculate the deconvolution function exactly and uniquely. However, in this study, a body attached fixed camera is used and the relative motion between camera and scene is estimated by using captured images only.

Even though deblurring methods are examined by many researchers and studied at various fields of engineering, the performance of these methods is limited and results may not satisfy the requirements of all kind of applications. Particularly, the proposed methods are not suitable for a mobile platform moving in an absolutely uncontrollable environment. Motion blur correction approaches are good for complementary efforts, but cannot be used to reveal the features in an image which is heavily imposed to camera shake and motion. However, due to their vital importance, visual sensors cannot be discarded. At that point, an important question arises. Is it possible to degrade the effects of visual disturbances during capturing process, instead of after capturing process? Fortunately, this question leads us to the stabilization problem which is the subject of this thesis. Minimizing the effects of disturbances on-the-fly is considered to be a more robust approach than deblurring the motion imposed images by postprocessing after they are captured. This prescience is the basis of our motivation source for the camera stabilization platform studies.

1.2 Literature Survey

Inertially stabilized platforms (ISPs) are used to stabilize and point a broad array of sensors, cameras, telescopes, and weapon systems. They have been used in a wide range of application areas; on every type of moving vehicle, from satellites to submarines. These applications include surveillance, target tracking, missile guidance, gun-turret control, communications, astronomical telescopes, and handheld cameras. Although the requirements for ISPs vary widely depending on the application, they all have a common goal, which is to hold or control the line of sight (LOS) of one object relative to another object or inertial space. This is accomplished by sensing angular movements of the vehicle or angular disturbances acting on

the vehicle and generating the necessary counter-rotations.

The LOS can be the aimpoint of a beam or weapon, the center of the field of view (FOV) of a telescope, or the direction a sensor is pointed. For example, vision sensors are routinely pointed and held stable by inertially stabilized platforms (ISPs) on ground vehicles, ships, aircraft, and spacecraft for diverse missions such as scrutinizing military targets, mapping, and providing high-resolution imagery for environmental surveys [28].

Perhaps the most straightforward and most common approach is to stabilize the main mechanism that carries all parts, which is called *Mass Stabilization* method [28]. However, to reduce the size and weight, *Steering Stabilization* or *Mirror Stabilization* method may be preferred as studied in [13, 59, 40, 27]. At this method, camera or other structures, which is intended to stabilize, is fixed to the base of the moving vehicle. A gimballed mirror is placed in the optical path and the LOS is stabilized by proper movements of this gimballed mirror.

Another distinction between stabilization methods is the location of sensing device. At indirect approach as studied by Wicker [72], the sensing device is fixed to the vehicle body. At such a configuration (i.e., off-gimbal), stabilization can be done by calculating body rotations and forcing the imaging system to rotate by counter-rotations of body angles. This method is less efficient, since the disturbances are not measured in the LOS coordinate frame [35]. However, at some situation it is implemented on a mandatory basis, i.e., if there is no physical place to mount a feedback gyro. On the other hand, direct approach, places the sensing device to the stabilized object. At that configuration (i.e., on-gimbal), sensing device measures the rotations of the movable platform and forces it to remain stationary according to the world coordinate frame. Mounting the gyroscope directly on the LOS axes is often preferred for precision pointing applications [60, 61].

Depending on the desired sensitivity and performance requirements, various design considerations can be applied. For example, mechanism that is used in [53], consists of two nested double-axes gimbals and mechanical vibration dampers. The outer structure is motor controlled and gyro-stabilized, which allows the pan (continuous 360°) and tilt (+10° / -90°) movements. The vision system is mounted in a two-axis inner gimbal, which is essentially frictionless, connected to the outer structure with a freedom to move of a few degrees.

As addressed in [54], physical modeling of such systems are non-trivial tasks. According to the desired precision, various structures can be obtained between simple linear models and extremely complex nonlinear models that incorporate dynamic effects such as structural flexibility, stiction, backlash, and cogging. It discusses the modeling problem of the double gimballed system in terms of geometrical and topological properties of torus.

From the perspective of controller design, it may not sufficient to simply prevent the sensor from rotating in inertial space, but rather, the sensor must be controlled in a specific manner depending on what the sensor system is attempting to accomplish. If the primary purpose of the system is to obtain a clear image of the target, it may suffice to hold the image steady within the sensor field of view (FOV) without concern for the specific aimpoint. If, however,

a beam, such as a laser rangefinder, is to be directed at the target, then holding the target precisely at the center of the FOV may be critical, while small amounts of high-frequency rotation or jitter might be inconsequential. For other systems, both the aimpoint and jitter may be required to control. Thus, the requirements for an ISP system depend on many factors but are primarily determined by the characteristics of the device being stabilized. For example, Masten [46] uses double looped P, PI and PID type controller structures with high-bandwidth inner stabilization loop and outer low-bandwidth track loop. Besides, Osborne [54] figures out an augmented PD controller design on Riemannian Manifolds after a rigorous analysis. Moreover, Wang [71] applies conventional integer-order PI (IOPI) and fractional-order PI (FOPI) type controllers to the stabilization loop of the system with nonlinearity and uncertainty. According to the results of the simulation works, he claims that FOPI type controller outperforms the conventional PI type controller.

As noted before, due to the nonlinearities and uncertainties originated from various disturbing phenomena and kinematic couplings between gimbal axes, obtaining an exact model is a challenging issue. Therefore, in many studies, non-classical controller methods have been used that do not require the exact model or is able to eliminate the modelling errors, i.e., fuzzy controller [33], adaptive controller [39, 40]. Disparately, in [48] $\mathcal{H}_\infty$ control methodology is used to obtain a robust control law for a linear model, which is acquired with experimental frequency response data.

All of the studies examined above used either position-control mode where the angular orientation is regulated or velocity-control mode where the angular rate is controlled. Certainly, the latter one is preferable due to the higher system bandwidth. Moreover, Algrin [3, 2] takes one step forward in his study and operates in an acceleration-control mode, which is even more desirable because this further increases the stabilization bandwidth. For this purpose, he replaces the gyroscopes by linear accelerometers, which are placed at particular locations to sense the angular accelerations.

1.3 Contributions

Our main purpose is to develop a stabilization platform for eliminating the disturbances which affect adversely and degrade the image quality in term of post-processing for feature extraction operations.

Works can be categorized into two groups; Sensing and Control as depicted in Figure 1.3. At the sensing part, sensory data is used to estimate state space vector partially.

The primary objective is to maintain the state vector of the camera which attached to the head, as close as possible to the origin. The state vector of the camera can be defined as follows;

$$\vec{x} = [\alpha \ \beta \ \gamma \ \dot{\alpha} \ \dot{\beta} \ \dot{\gamma}]^T = \vec{0} \tag{1.1}$$

where, α, β and γ are Euler angles.

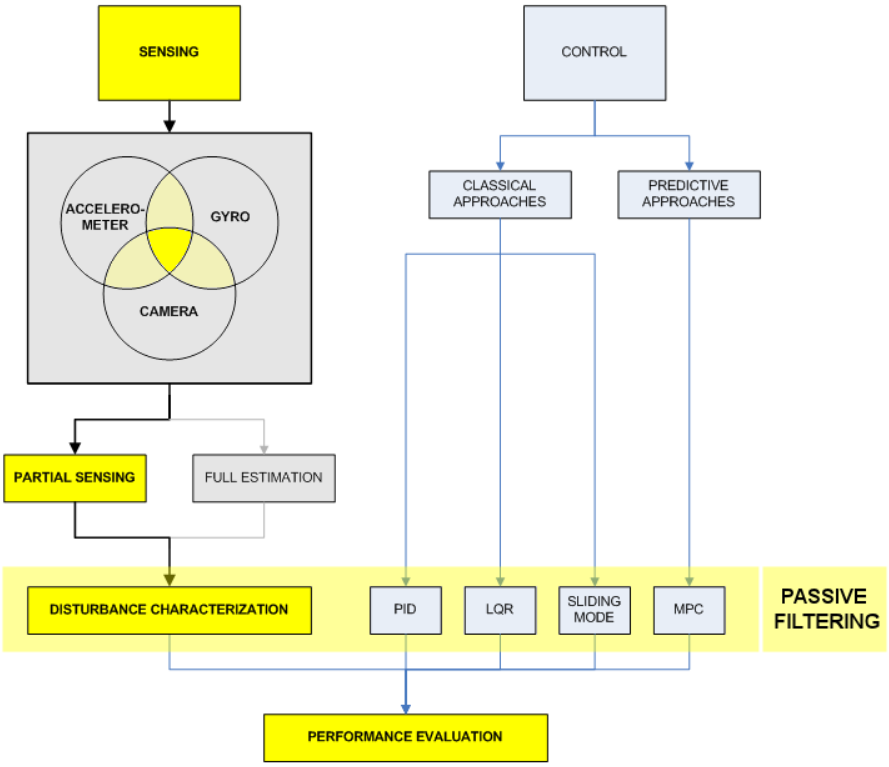


Figure 1.3: Categorization of thesis works: sensing and control

However, this ultimate aim will be reached step by step. At first stage, instead of 3-DOF head stabilization problem, the simplified planar case will be examined. The state vector of the camera now reduced to;

$$\vec{x} = \begin{pmatrix} \theta \\ \dot{\theta} \end{pmatrix} = \vec{0} \tag{1.2}$$

Also some simplification can be done even further. The occurrence of motion blur effect is originated from non-zero angular rate. Error in angular position does not cause to such a visual disturbance. So at preliminary stages of thesis working, non-zero angular position of camera is not the main topic and can be tolerable at some extent. Therefore, it will be focused on angular rate of camera.

The following topics is researched during my thesis works.

- Exploring the disturbances by IMU data which affect the captured image by camera
- Modeling of the disturbances and actuator structures
- According to the results obtained from the above-mentioned topics; Determining the controller structure; PID, LQR, Sliding Mode Control, Model Predictive Control
- Discovering the performance relationship between controller structure and vision system

CHAPTER 2

CAMERA STABILIZATION MECHANISM & CONSTITUENTS

2.1 PC104 & Data Acquisition Card

Diamond System ATHENA II, as shown in Figure 2.1, series processing hardware is based on PC104 form factor. It runs a MATLAB based real time operating system, xPC Target, to gather all the sensor and encoder data from the peripherals, processes these data and to controls the actuators to stabilize the camera platform.

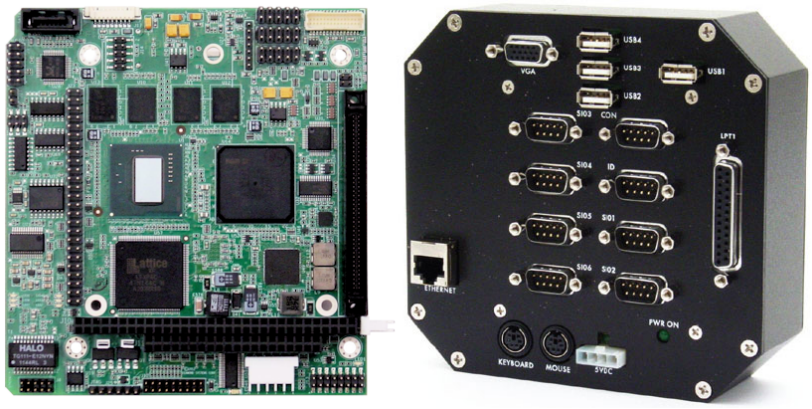


Figure 2.1: Diamond System Athena II PC104 & Data Acquisition Card

2.2 Linear DC Servomotor & Motor Driver

Standard rotary servomotors spin leadscrews or pull belt drives to produce linear motion. Direct-drive linear motors are more likely to use self-supporting coils to position a linear shaft filled with permanent magnets. The servomotor design lets the controller drive the shaft to the new position with almost no residual static force, which makes these types of motors good candidates for micro-positioning. Faulhaber Quickshaft LM1247, which is shown in Figure 2.2.a, comprises an electromechanical part, a metal housing made from a non-magnetic

alloy, a sliding metal shaft equipped with permanent magnets, delta connected three-phase coil windings, as well as three analogue Hall sensors. It has 20 [mm] maximum stroke and 2000 [mm/s] maximum shaft velocity. More detailed technical specifications, as well as dimensions and geometric properties about linear servomotor are supplied in Appendix A.1.

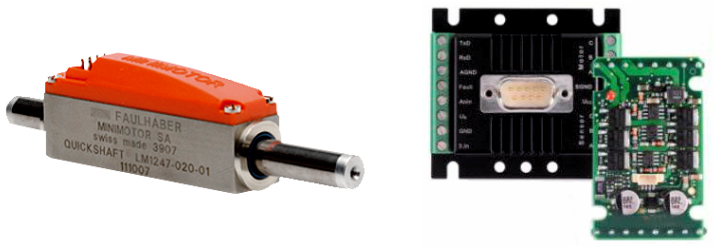


Figure 2.2: a) LM1247 linear DC servomotor b) MCLM 3006S external motion driver

Faulhaber MCLM 3006S, which is shown in Figure 2.2.b, is an external motion driver for LM1247 series linear DC servomotor and can be operated without additional encoders. It has multiple control loops which consist of PD-type position controller and PI-type velocity controller as shown in Figure 2.3. More detailed technical specifications about motor driver are supplied in Appendix A.2.

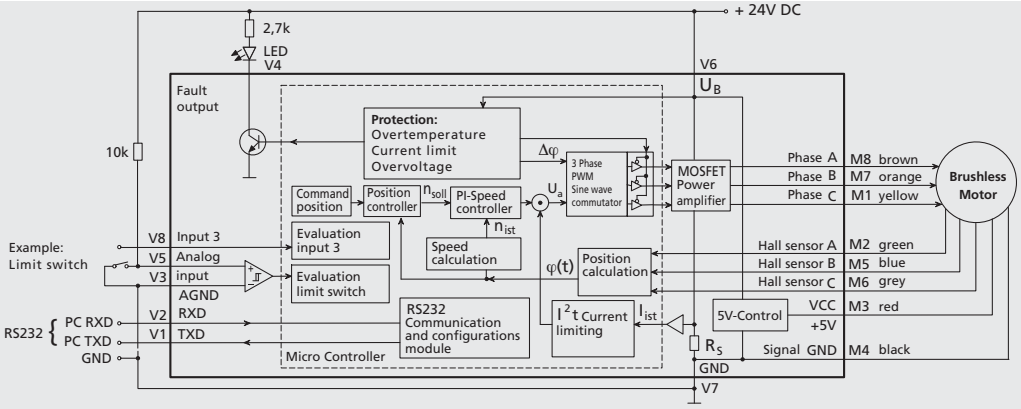


Figure 2.3: Schematic diagram of the external motion driver, MCLM 3006S, connected to the three-phase linear servomotor, LM1247, through analog Hall sensors

The MCLM 3006S has various operation modes which can be itemized as follows;

- Normal Operation Mode (Position & Velocity Control through RS232 Interface)
- Analog Operation Mode (Position Control with Analog Input)
- Stepper Operation Mode (Velocity Control with Digital Input)

The difference between various operation modes are the method of sending reference signals. For example, at normal mode, the position or velocity reference is sent to the motor by using RS232 serial communication interface with related ASCII commands. Whereas, at analog mode, the position reference is set by changing the voltage level at analog input pin. Besides, at stepper mode, the velocity reference is set by using digital input, while the period of the reference signal determines the shaft velocity and the number of pulses determines the step number to be jumped.

At position control modes, the motion driver accepts position reference and activates both outer PD-type position and inner PI-type velocity control loops. At this mode, the position error is used as a reference for the velocity controller. However, at velocity control modes, the motion driver accepts velocity reference and the outer PD-type position control loop is deactivated. At this mode, it only operates inner PI-type velocity control loop.

Independently from the operating mode, through RS232 communication interface, PD and PI controller gains, maximum shaft limit, speed limit, and acceleration/deceleration limits can be set and altered in run-time. Also, position, velocity, voltage or current information can be queried by using related ASCII commands. Some of query and programmable parameters are summarized in Table 2.1.

Table2.1: Query and programmable parameters of the MCLM 3006S external motion driver

Command	**Argument**	**Description**	**Unit**
CONTMOD	-	Set Normal Mode	[-]
APCMOD	-	Set Analog Mode	[-]
STEPMOD	-	Set Stepper Mode	[-]
PP	1 ... 255	Set Position Controller Proportional Gain	[-]
PD	1 ... 255	Set Position Controller Derivative Gain	[-]
POR	1 ... 255	Set Velocity Controller Proportional Gain	[-]
I	1 ... 255	Set Velocity Controller Integral Gain	[-]
CI	1 ... 255	Set Current Controller Integral Gain	[-]
LPC	0 ... 12000	Set Peak Current Limit	[mA]
LCC	0 ... 12000	Set Continuous Current Limit	[mA]
LL	-10 ... 10	Set Position Range Limits	[mm]
CORR	0 ... 10	Set Window Around the Target Position	[mm]
SP	0 ... 10000	Set Maximum Speed	[mm/s]
DEV	0 ... 10000	Set Maximum Deviation of Target Velocity	[mm/s]
ACC	0 ... 30000	Set Maximum Acceleration	[mm/s^2]
DEC	0 ... 30000	Set Maximum Deceleration	[mm/s^2]
POS	-	Get Shaft Position	[mm]
GN	-	Get Shaft Velocity	[mm/s]
GNU	-	Get Motor Voltage	[mV]
GRC	-	Get Motor Current	[mA]

2.2.1 Design of Experiments

In this section, it is aimed to examine the characteristic of the actuating mechanism which consist of Faulhaber Quickshaft LM1247 series linear DC servomotor and Faulhaber MCLM 3006S series external motion driver. For this purpose, actuating mechanism is disassembled from the camera stabilization platform and operated under no load conditions as shown in Figure 2.4. In this way, it is intended to get rid of disturbances originated from the load and to test the performance of the linear motor and motion driver at various operating modes.

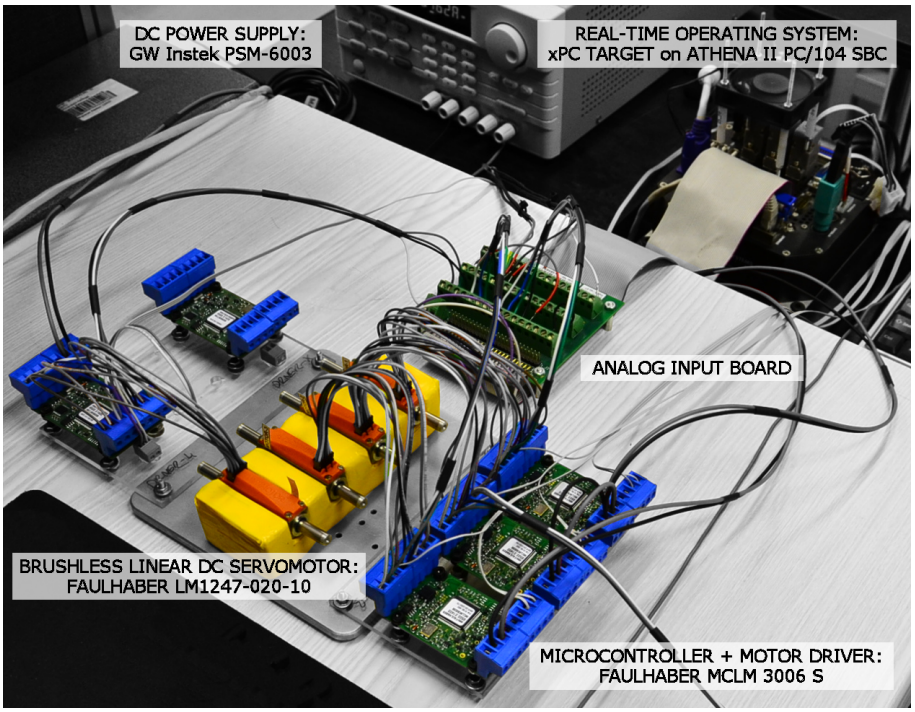


Figure 2.4: Experimental setup for no-load operation of actuating mechansim

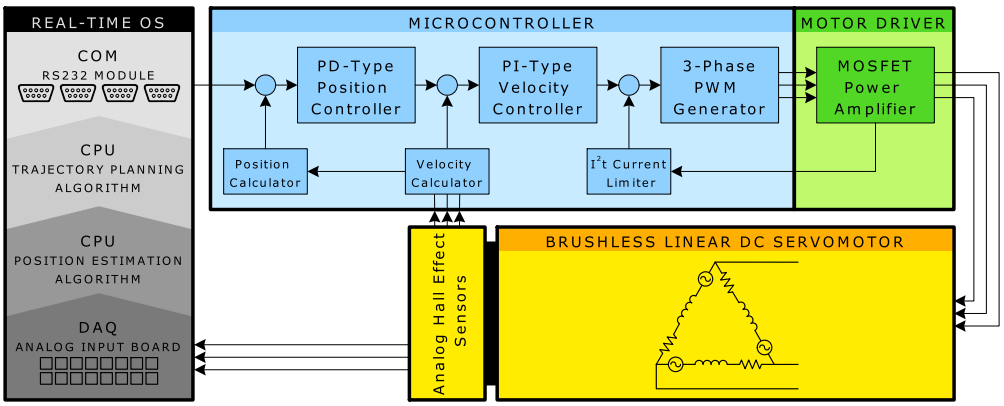


Figure 2.5: Schematic diagram of experimental setup

2.2.2 Position Control Experiments at No-Load Operation

2.2.2.1 Processing and Communicating Delays

The MCLM 3006S external motion driver is capable of transmitting the reference commands to the linear DC servomotor by using ASCII codes through RS232/CAN interfaces, or as analog/digital inputs. In either case, reference commands are sent to the linear DC servomotor step-by-step. Furthermore, it is possible to send all reference commands at once (in-one-go) by using sequential programming syntax and write these programs to the flash memory of external motion controller for standalone operating condition.

At this section, firstly, to get rid of delays originated from either operating system interrupts or communication interface, the motor is operated at standalone mode by using sequential programming syntax. For this purpose, reference shaft position, which is a pulse input with amplitude of 10 [mm] and period of 1.1725, 0.7725, 0.3750 and 0.1825 [s] is written to the flash memory of external motion controller. Then the communication between driver and host computer is disabled and linear DC servomotor is operated at stand alone (in-one-go). For each cases, reference (desired) and queried (measured) shaft positions are given in Figure 2.6. Because there is no prohibitive effect of operating system or communication interface at stand alone operation, the results indicate the maximum performance of linear motor and motor driver for default parameter set.

At stand alone operation condition, we get rid of all prohibitive effects which are originated either from operating system interrupts or communication interface delays. However, pre-programming the linear motor and working at stand alone operation is not suitable for our purposes, which require the rejection of unknown disturbances. So, at second experiment, related ASCII commands are sent to the motor driver through RS232 serial communication interface to set the reference shaft position step-by-step, i.e., the next position command is sent to the motor after a returned notification which indicates the execution of the preceding command. For this purpose, MATLAB numerical computing environment is used under Windows XP operating system. The same experiments are conducted as shown in Figure 2.7, at which the reference (desired) and queried (measured) shaft positions are given.

Previous experiments, conducted at standalone and under non-real-time operating system, show that the overall performance is affected adversely by controlling the motor step-by-step instead of in-one-go approach. Actually, the reason of the performance degrading can be originated from both operating system interrupts and RS232 serial communication delays. Hence, to understand the source of this problem, we migrate to the xPC Target, which is a real-time software environment. Note that, at xPC Target, the operating system interrupts are not applicable, and only communication delays are in effect. The reference (desired) and queried (measured) shaft positions are given in Figure 2.8.

The performance specifications for obtained results are summarized in Table 2.2. Note that, the velocity values are almost identical for three different operating conditions. Actually it

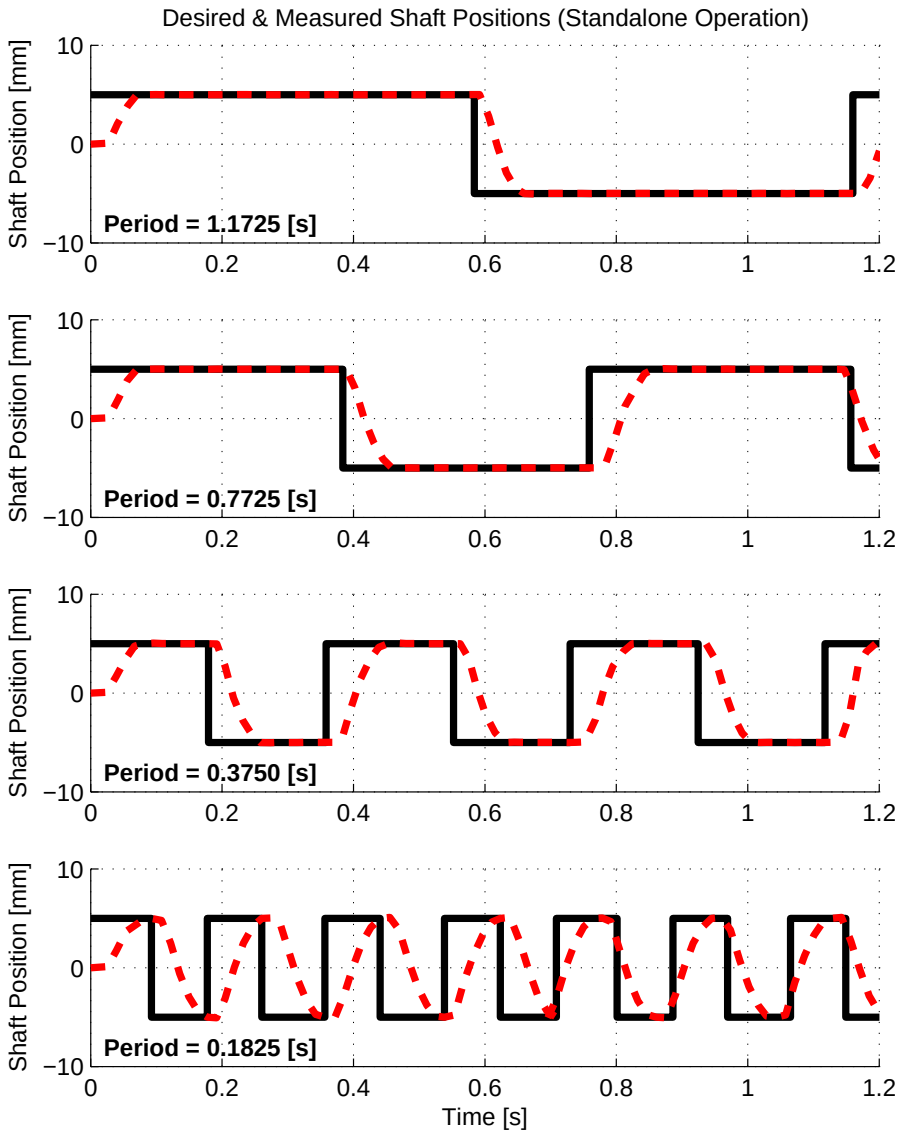


Figure 2.6: Desired and measured shaft positions at stand alone operating condition. Reference shaft position is a pulse input with amplitude of 10 [mm] and frequency of 0.85, 1.30, 2.67 and 5.48 [Hz], respectively

is an expected situation, because the velocity is computed between initial and settling time of motor movements. Once the motor begins to the motion, it behaves nearly the same. The interesting point comes up shortly before the beginning of motion. At standalone mode, there is a ~15 [ms] time delay to get the motor into motion from the stationary state. Because there is neither communication delay nor operating system interrupts at standalone mode, it is understood that, there is a processing time originated from motion driver itself. Under the non-real time and real-time operating systems, the delay is almost identical and ~50 [ms] for relatively slower references (i.e., signals with frequency of 0.85 and 1.30 [Hz]). However, for faster references, the effects of operating system interrupts become visible and add extra delay to the overall process. Even for the last case, settling event never occurs under non-

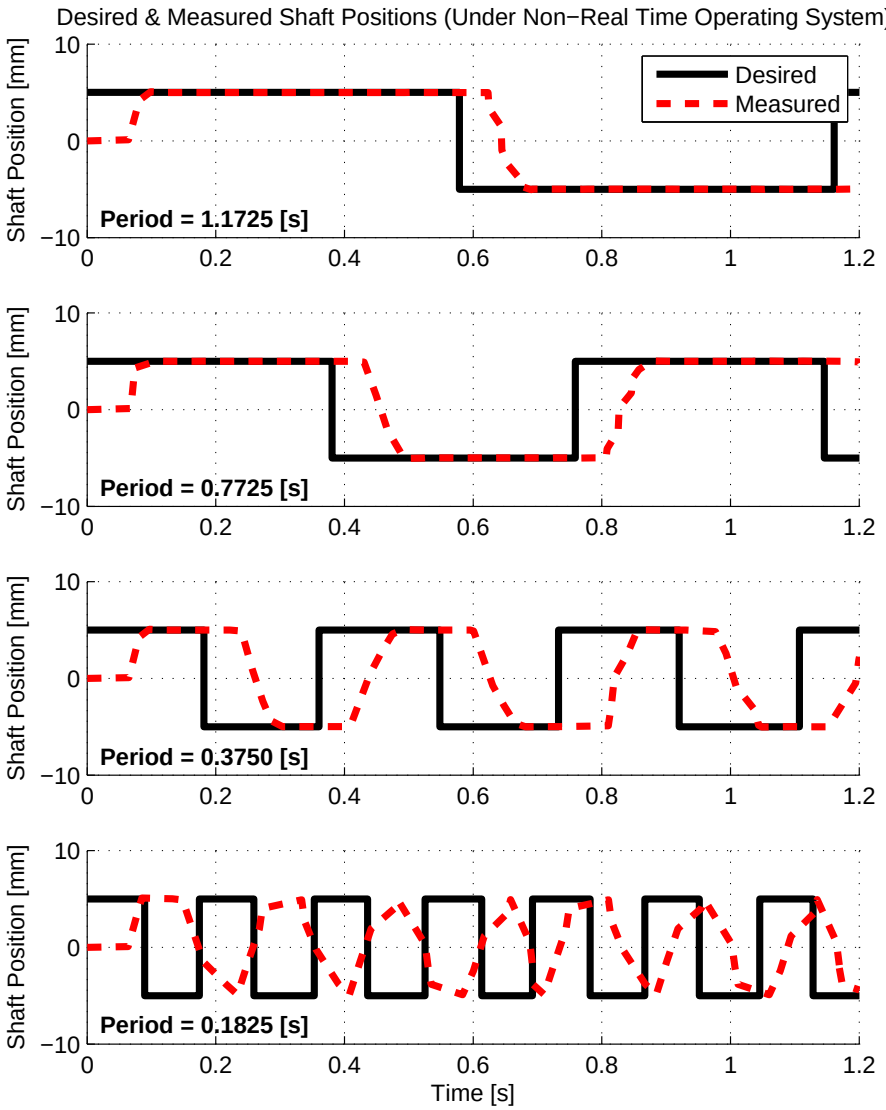


Figure 2.7: Desired and measured shaft positions under non-real time operating system. Reference shaft position is a pulse input with amplitude of 10 [mm] and frequency of 0.85, 1.30, 2.67 and 5.48 [Hz], respectively

real-time operating system, so performance specifications can not be given. In either case, RSR232 communication interface adds approximately ~35 [ms] delay to the motion driver's processing time.

2.2.2.2 Normal and Analog Modes

At this section, normal and analog modes are examined and compared according to the experimentally obtained maximum velocity and time delay values. Remember that, at normal mode, the reference shaft position is sent to the motor by using ASCII commands through RS232 serial communication interface. On the other hand, at analog mode, the reference

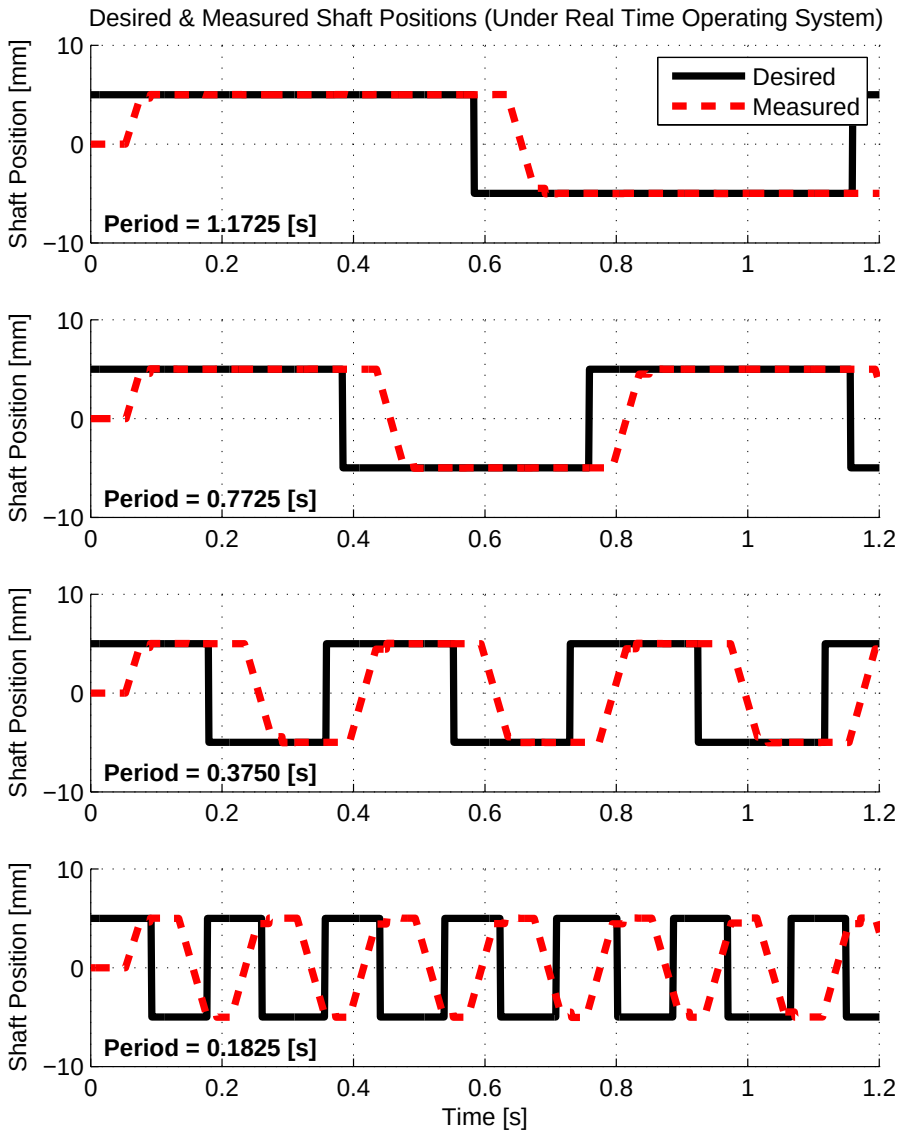


Figure 2.8: Desired and measured shaft positions under real-time operating system. Reference shaft position is a pulse input with amplitude of 10 [mm] and frequency of 0.85, 1.30, 2.67 and 5.48 [Hz], respectively

position is set by changing the analog input between [-10, 10] [V], which corresponds to the [-10, 10] [mm] shaft stroke. The experiments are conducted for the same scenarios and the obtained results are shown in Figure 2.9. Note that, the time delay calculated for analog mode experiments are approximately ~15 [ms], which indicates that there is no extra delay apart from motion driver's own processing time. However, the maximum velocity value is calculated approximately as ~118 [mm/s], which is quite less than normal mode counterpart.

As a consequence, there is a trade-off between normal and analog modes usage. At normal mode, it is possible to reach ~230 [mm/s] shaft velocity for default parameter set, but there is ~35 [ms] extra delay to the driver's processing time. On the other hand, there is no extra

Table2.2: Performance Specifications at Standalone, Non-Real Time and Real-Time OS

Mode	Amp. [mm]	Freq. [Hz]	Max Velocity [mm/s]	Delay Time [ms]
Standalone	10	0.85	233.86	15.90
		1.30	235.74	14.40
		2.67	235.27	13.30
		5.48	231.22	14.70
Non-Real Time	10	0.85	230.13	44.80
		1.30	228.89	49.60
		2.67	232.04	55.80
		5.48	-	-
Real-Time	10	0.85	229.82	48.50
		1.30	232.57	42.70
		2.67	231.33	44.60
		5.48	232.30	46.00

delay for analog mode, but shaft velocity hardly reaches to the ~118 [mm/s].

2.2.2.3 Data Request Frequency

Due to the lack of an external encoder, the only way to get the shaft position or velocity information is requesting it from the motion driver by using ASCII commands through RS232 serial communication interface. Also, as indicated before, at normal mode, the reference command is sent to the motor by using serial interface. However, the bandwidth of the serial communication is limited even at 115.2 [kBaud], which is the maximum allowable communication speed obliged by the hardware constraints. Also, sending or requesting excessive amount of data may cause malfunction of motion driver, even if the limits of the serial interface is not exceeded. So, there is a trade-off between the amount of send data (i.e., position reference) and requested data (i.e., position measurement), to not prevent the proper operation. To decide the best data sending-requesting rate combination, various experiments are performed both at normal mode, several of which are shown in Figure 2.10. The requesting rate is started from 1 [kHz], and reduced one-by-one until 1 [Hz], which are shown by blue lines. Also, the experiment lead to the minimum RMSE value, defined between desired and measured shaft positions, is indicated by red line. The results show that, the optimal requesting rate is 250 [Hz] for reference signal with period of 1 [s]. However, if the period of reference signal is reduced to 0.5 and 0.2 [s], the optimal requesting rates are calculated as 200 [Hz] and 167 [Hz], respectively.

It is not difficult to guess, if the period of the reference signal further decreased, the data requesting rate also have to be reduced. It is obvious that, for faster movements, the external motion driver process more operation, which causes either to prevent the execution of control action properly or to send the requested data at right instants. Unfortunately, due to the lack

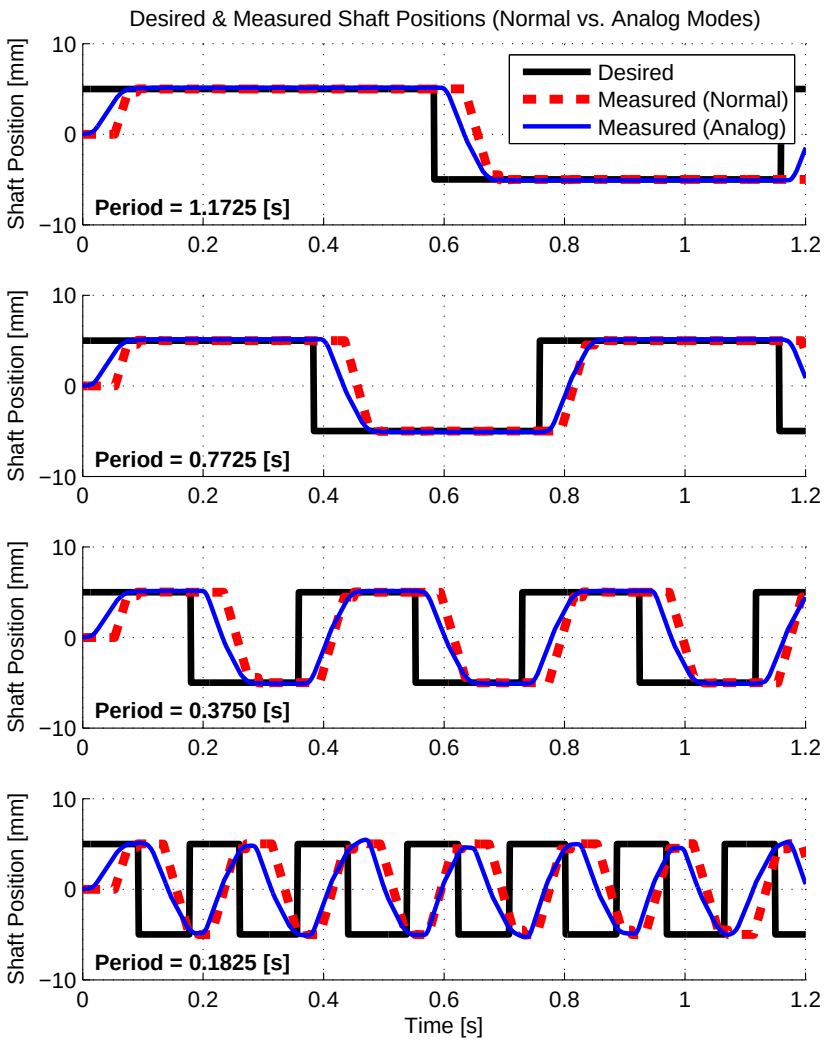


Figure 2.9: Desired and measured shaft positions for normal and analog operation modes. Reference shaft position is a pulse input with amplitude of 10 [mm] and frequency of 0.85, 1.30, 2.67 and 5.48 [Hz], respectively

of an external encoder, we are not sure about which portion of the process is problematic. In other words, control action may be executed properly, but position information may not be sent at right instants, or vice versa. However, dependence to the serial interface or external encoder to obtain the shaft position is resolved at subsequent studies by estimating shaft position directly from Hall Effect Sensor measurements at high rates (i.e., up to 2 [kHz]).

2.2.2.4 Motion Driver Internal Parameters

At this section, the effects of motion driver internal parameters are examined in terms of the position control performance at normal operation mode. For this purpose, the same reference signal, which is shown in Figure 2.10.b, is used. At every set of experiments, one of the pa-

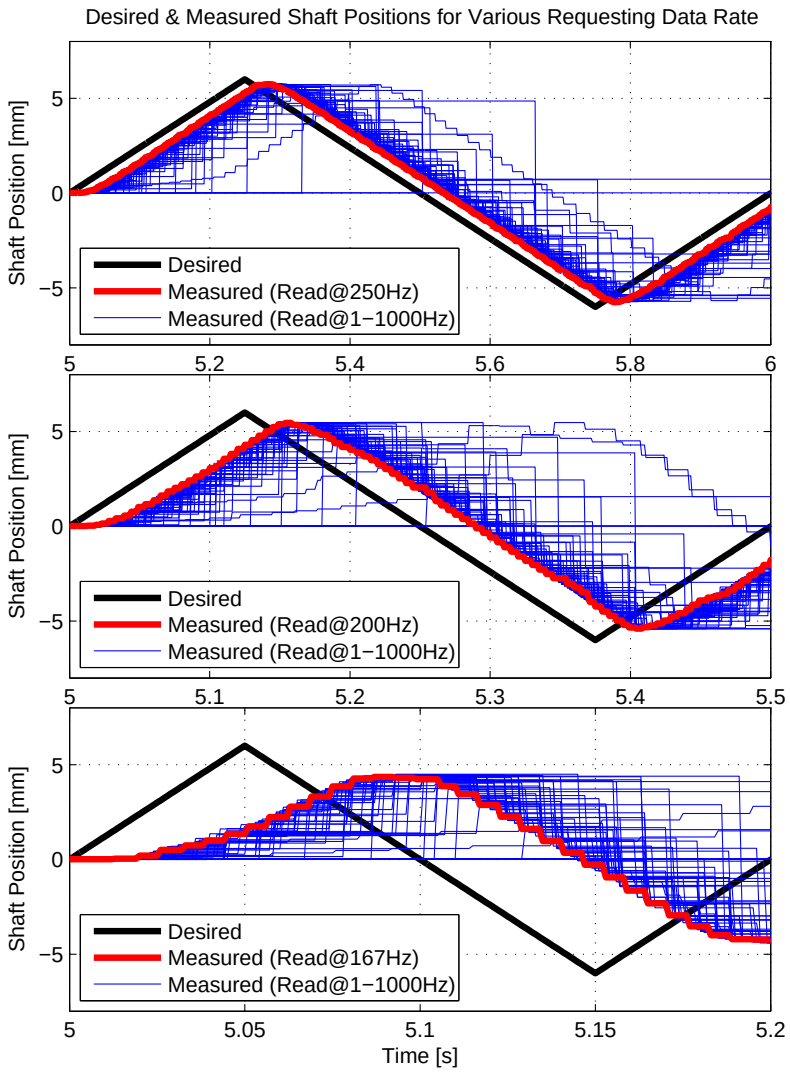


Figure 2.10: Desired and measured shaft positions for various data requesting rates.
a) Reference Period = 1 [s], b) Reference Period = 0.5 [s], c) Reference Period = 0.2 [s]

rameter is altered between its lower and upper limits, while the others are fixed at their default values. Remember that, external motion driver has two cascaded controllers; PD-type position controller at outer loop and PI-type velocity controller at inner loop. The first experiment set is obtained by changing the proportional and derivative gains of PD-type position controller as shown in Figure 2.11. Here, the variation of RMSE value, which is defined between desired and measured shaft positions, is given with respect to the controller gains. In each figure, related controller gain, which causes the minimum error, is indicated by a marker. Notwithstanding, the aim of these experiments are to find the stability region for further optimization efforts, rather than to find the optimum values.

The second experiment set is obtained by changing the proportional and integral gains of PI-type velocity controller as shown in Figure 2.12.

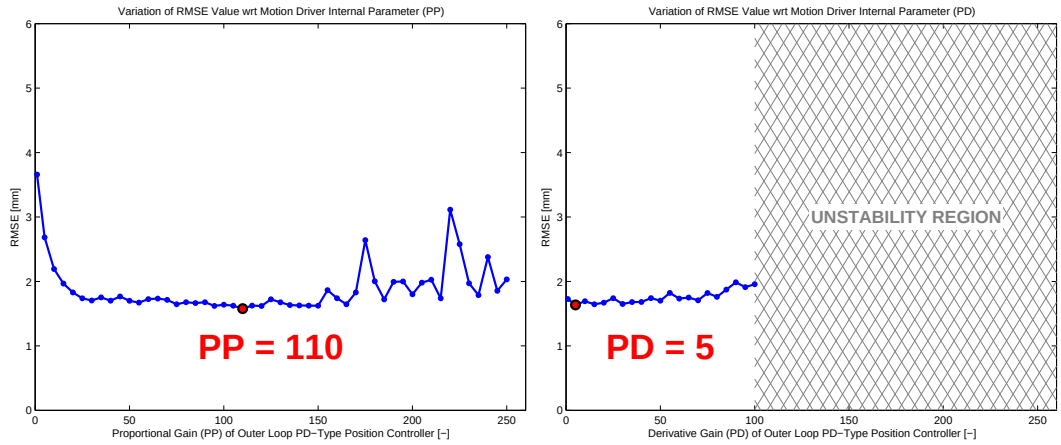


Figure 2.11: The variation of RMSE, defined between desired and measured shaft positions, with respect to the proportional (PP) and derivative (PD) gains of PD-type position controller

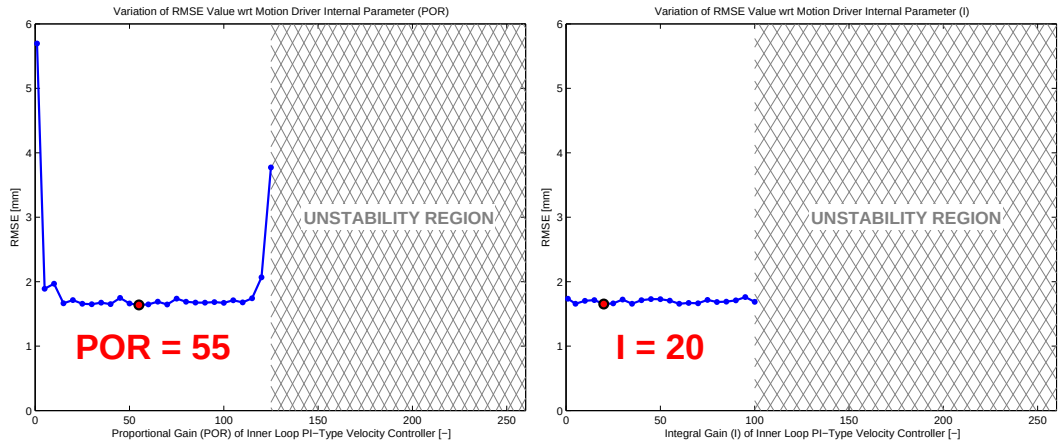


Figure 2.12: The variation of RMSE, defined between desired and measured shaft positions, with respect to the proportional (POR) and derivative (I) gains of PI-type velocity controller

Finally at this section, the effects of maximum acceleration limit, maximum deceleration limit and maximum velocity limit are investigated as shown in Figure 2.13. Note that, maximum acceleration limit does not improve or worsen the overall performance except for very low values, however, the same inference is not valid for maximum deceleration limit. As shown in Figure 2.13.b, the performance degrades for relatively low values smaller than the 7000 [mm/s^2], as well as 30000 [mm/s^2] which is the highest value of the maximum deceleration limit. As a special situation, at the highest value of maximum deceleration limit, the external motion driver disables the velocity profiler, which accepts the output of the position controller and generates a reshaped velocity reference signal for the velocity controller. More expressly, the velocity profiler reforms the reference input of the velocity controller to minimize the possible overshoots. Therefore, due to the occurred overshoots and high frequency oscillations around reference point, the RMSE value increases for the highest value of maximum deceleration limit, which deactivates the velocity profiler. Furthermore, surprisingly as

shown in Figure 2.13.c, the values of the maximum velocity limit, which is greater than 8000 [mm/s], affects the response, while the theoretically achievable maximum velocity value is 2000 [mm/s] for this motor according to the factory specifications.

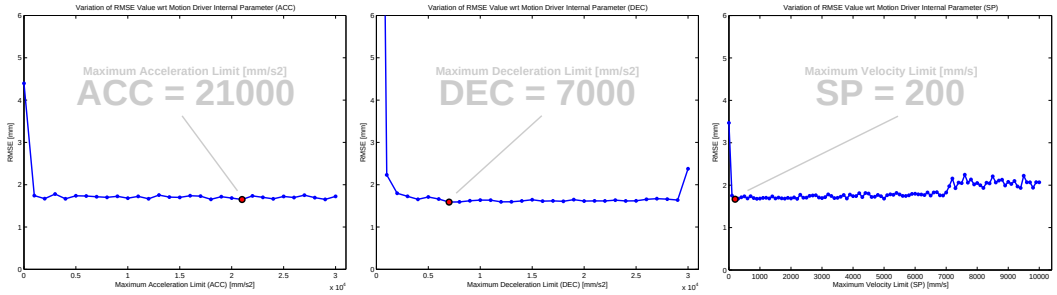


Figure 2.13: The variation of RMSE with respect to the a) maximum acceleration limit (ACC), b) maximum deceleration limit (DEC), c) maximum velocity limit (SP)

Table2.3: Default and modified motion driver internal parameters

Parameter	**Range**	**Default Value**	**Modified Value**	**Unit**
PP	1 ... 255	80	NOT DECIDED	[-]
PD	1 ... 255	10	NOT DECIDED	[-]
POR	1 ... 255	20	NOT DECIDED	[-]
I	1 ... 255	10	NOT DECIDED	[-]
CI	1 ... 255	40	3	[-]
LPC	0 ... 1660	1440	1650	[mA]
LCC	0 ... 550	480	550	[mA]
ACC	0 ... 30000	30000	30000	[mm/s^2]
DEC	0 ... 30000	4000	30000	[mm/s^2]
SP	0 ... 10000	500	10000	[mm/s]

According to the results of these experiments and recommended operation instructions, the default values of the motion driver internal parameters are modified in compliance with necessities of our application as given in Table 2.3. Note that, the values of the controller gains are decided after an optimization process at subsequent studies. The values of the safety current limiter integral gain (CI), peak current limit (LPC) and continuous current limit (LCC) are determined according to the operation instructions to obtain high dynamic response. The highest values of maximum acceleration (ACC), deceleration (DEC) and velocity (SP) limits are selected by taking the consequences of mentioned drawbacks above. By designing our own controller, it is intended to eliminate the adverse impacts of disabled velocity profiler.

2.2.3 Velocity Control Experiments at No-Load Operation

At this section, unlike the previous experiments, velocity reference is sent to the motion driver, instead of position reference. At this operation mode, the motion driver deactivates outer

position control loop and only operates inner velocity control loop. It also disables the velocity profiler.

To perform these experiments, the reference shaft velocity is increased from 50 [mm/s] to 2000 [mm/s], which is the theoretical maximum velocity, by 50 [mm/s] steps. In each case, the shaft is driven back and forth between its mechanical limits, repeatedly. In Figure 2.14, the variation of measured velocity, which is the maximum value of each experiment, is given with respect to the reference velocity. Note that, these experiments are performed by using modified internal parameters given in Table 2.3.

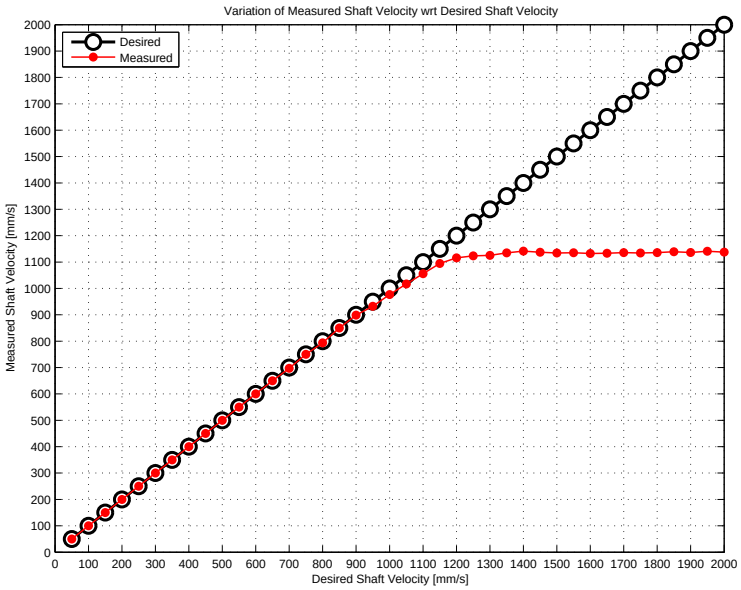


Figure 2.14: Variation of measured velocity, defined as the maximum value of each experiment, with respect to the desired velocity

If the motor shaft is able to track the reference velocity, two lines given in Figure 2.14 have to overlap with one another. However, due to the limitations of used hardware and technical problems originated from non-ideal cases, the experimentally reached maximum velocity value (~1100 [mm/s]) remains lower than the theoretical value (2000 [mm/s]).

2.3 Hall Effect Sensor

The motion driver, MCLM 3003/06 S, provides various query commands to request and to get information about motor states by using RS232 serial communication interface (i.e., position, velocity, current or voltage). However, experiments showed that requesting these information through serial interface affects adversely the performance of the motion driver as pointed out at the previous chapter. So, to measure the motor position at higher rates without degrading the motion driver performance and to resolve the dependence to the serial interface, it is aimed to estimate the motor position by using analog Hall effect sensor outputs. As shown in

Figure 2.15, the linear DC servomotor has three analog Hall effect sensors. While the shaft is sliding from one direction to other, these sensors generate sinusoidal signals in the range of 1-4 [V] according to the magnetic poles. Space between them is a motor design parameter and carefully selected to provide phase shift between generated signals.

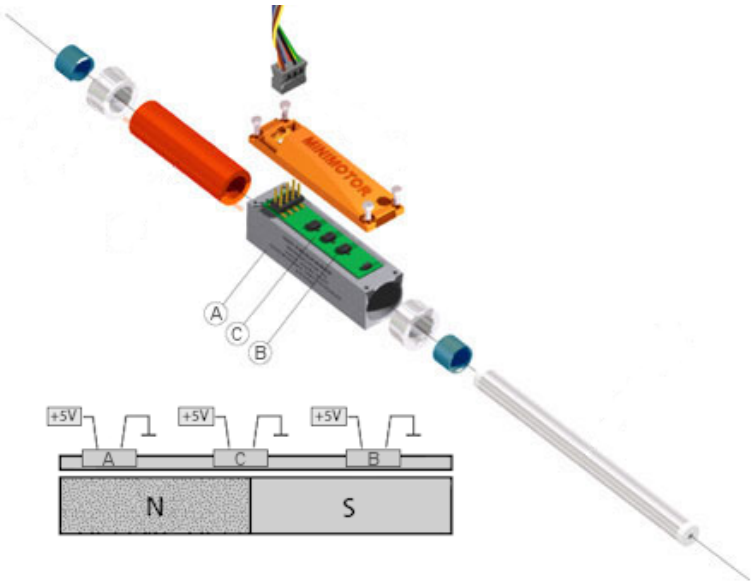


Figure 2.15: Schematic diagram of Faulhaber LM1247-020-10 linear DC servomotor equipped with three analog hall effect sensors

Actually, the Hall effect sensor model is nothing but a simple mathematical equation, which maps the three sensor measurements to the shaft position. Therefore, to reveal the input-output relationship and to propose a mathematical model that represents the system, an experiment is conducted by driving the motor from one direction to other step-by-step between its maximum stroke values. At each step, 50 measurements are taken for Hall-A, Hall-B and Hall-C, which is given in Figure 2.16 as normalized voltage values.

In a rotary motor, a full mechanical cycle of the rotor corresponds to the one electrical period of the generated sinusoidal signals, so the position of the rotor can be uniquely determined by using three sensor outputs. However, in our situation, one period of the sinusoidal signal does not correspond to the "one-cycle" of the linear motor. Here, "one-cycle" is used for motor's full stroke while it is sliding from minimum to maximum position. As a result, the same sensor outputs are encountered multiple times in the range of motor movement.

For example, in Figure 2.16, Region-1 and Region-13 have completely the same signal characteristics and there is no distinctive feature between them. Similarly, the properties of the sinusoidal signals are identical at Region-2 & Region-14, Region-3 & Region-15, ..., Region-10 & Region-22. So, it is a typical one-to-multiple problem. In other words, the same sensor output values have to generate shaft position between [-15.60,-1.79] [mm] or [1.19, 15.60] [mm], depending on from which region they are gathered. Nevertheless, signals at Region-

11 and Region-12 are matchless and the shaft position can be uniquely determined by using sensor measurements at those regions.

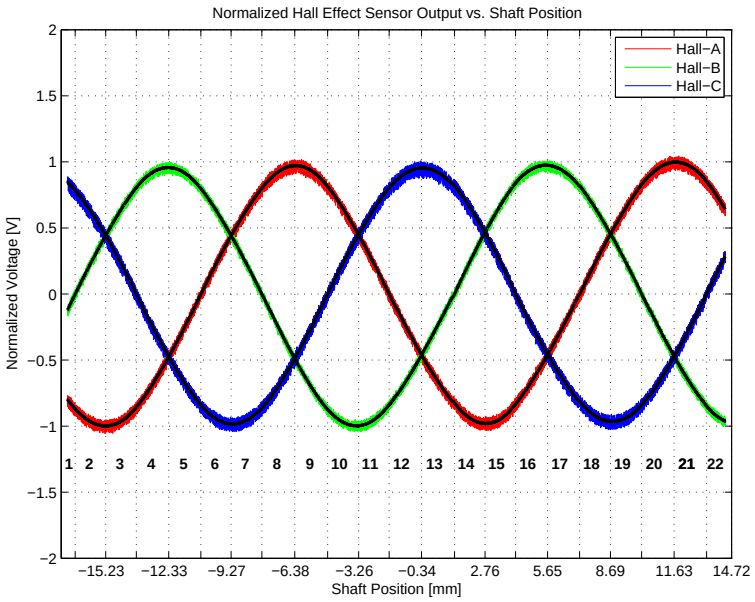


Figure 2.16: The normalized Hall effect sensor outputs in [V] with respect to the motor shaft position. At each step, 50 measurements of the HallA, HallB and HallC are plotted as red, green and blue, respectively. Also, the averages of them are indicated by black.

Actually, the frequencies of these "sinusoidal" signals are not constant and consequently, the summation of them varies. So, the first idea is to determine a threshold voltage for distinguishing of those regions. It works in simulation environment, however, due to the measurement noise it does not work at real application. The second idea is to estimate the shaft position by taking into account the former estimations. In other words, the estimated shaft position is calculated between [-15.60,-1.79] [mm], if the former estimation is also negative. Similarly, it is calculated between [1.19, 15.60] [mm], if the former estimation is positive. Fortunately, as indicated before, Region-11 and Region-12 are matchless, so switching from negative side to positive side or vice versa can be determined undoubtedly.

$$P = K_A \arcsin(A) + K_B \arcsin(B) + K_C \arcsin(C) + L \tag{2.1}$$

To map the three sensor outputs to the shaft position, a mathematical relationship is proposed as given in Equation 2.1, where P is the shaft position in [mm], A, B and C are the Hall effect sensor measurements in [V], K_A, K_B, K_C and L are the tunable parameters of the estimation model, which are determined by system identification. As indicated before, the frequencies of the sinusoidal signals generated by Hall effect sensors are not constant. So, the entire signal is more like a sweeping sine, instead of a pure sine, which makes difficult to find a unique parameter set and a unique estimation model. Therefore, the entire signal is divided into small regions as shown in Figure 2.16. At each division, the signal segments substantially behave

like a pure sine wave. The estimation model parameters, K_A, K_B, K_C and L, are optimized at those small regions for mapping sensor outputs to the shaft position.

By using optimized model, the shaft position is estimated with the rate of 1 [kHz] in real time as shown in Figure 2.17. Remember that, motion driver is affected adversely by the increasing rate of serial port measurements, so is limited by 50 [Hz] to prevent the performance degrading. Due to the low serial communication rate, the position reference command can not be sent at desired instants. More importantly, the measured position by using serial port communication does not reveal the absolute and overall characteristic of the whole process. It is obvious that the overshoot and oscillatory behaviors captured by our estimation model, but missed in serial port measurement.

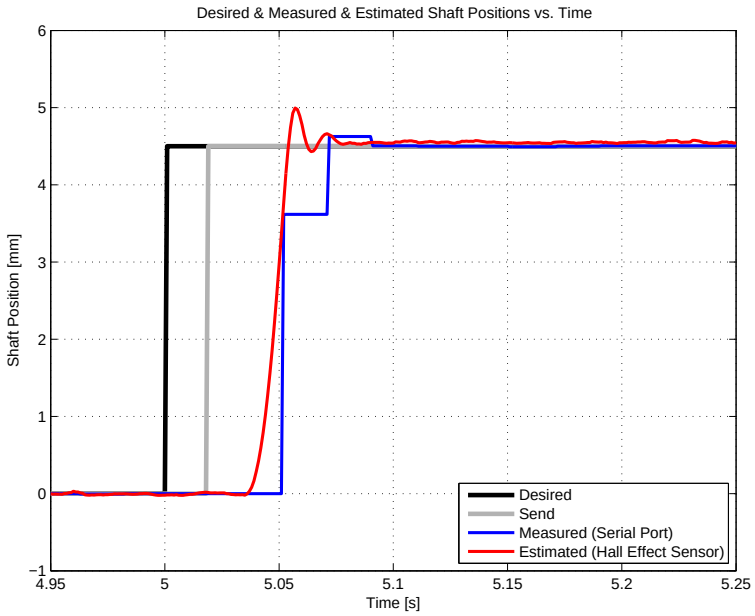


Figure 2.17: The desired, measured and estimated shaft positions. The measured position is indicated with blue and read through serial port interface at 50 [Hz]. The estimated position is indicated with red and calculated by using our proposed model at 1 [kHz]

2.4 Incremental Rotary Encoder

At similar applications such as line-of-sight or turret stabilization, according to the location of sensing device, two different approaches namely indirect and direct methods can be used. At indirect method, the sensing devices are fixed to the main body. At that configuration, the rotations of the main body are measured and the stabilized object is forced to rotate by counter-angles of main body. To realize this scenario and perform such experiments, two-degrees-of-freedom gimballed platform is used for generating disturbance signals. By using Autonics E50S8-5000 incremental rotary encoders as shown in Figure 2.18, roll and pitch angles of the base plate can be measured at resolution of 5000 [pulse/rev].



Figure 2.18: Autonics E50S8-5000 Incremental Rotary Encoder

2.5 Inertial Measurement Unit

On the contrary to the indirect method, at direct method sensing devices are mounted to the stabilized object. At that configuration, the rotations of the movable platform are measured and the stabilized object is forced to remain stationary according to the world coordinate frame. Also, apart from being a feedback signal for controller structure, the amount of deviation of top platform's angles are needed as a success criterion during quantification process. So, to estimate orientation of the stabilized object, two inertial measurement units, MicroStrain 3DM-GX1 and IMU3000 Combo as shown in Figure 2.19, are mounted to the upper plate. By using STM32F4 series microcontroller for gathering and synchronizing sensor measurements, three-axes acceleration and three-axes angular rate data are measured at 200 [Hz] from each sensor.

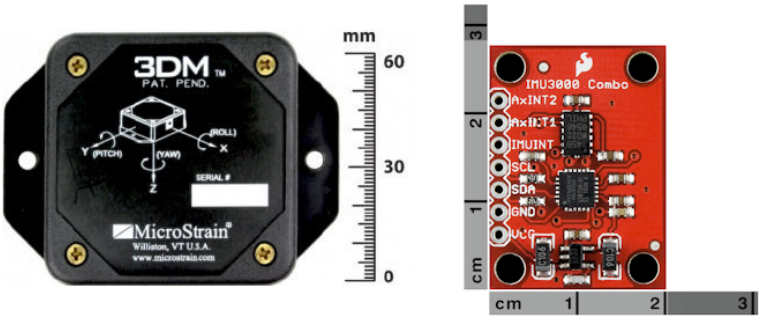


Figure 2.19: Inertial Measurement Units a) MicroStrain 3DM-GX1 b) IMU3000 Combo

If the only force acting on the system is gravity, then it is possible to calculate the orientation with trigonometric quantities in roll and pitch axes just by using acceleration data. However, vibration and other external forces influence the calculated results making them very noisy and, in many cases, unusable by themselves. So orientation estimation by accelerometer data is reliable only at stationary conditions. On the other hand, unlike accelerometers, rate gyros are nominally unaffected by acceleration, making gyro-based angle estimates immune to external forces. However, due to the noisy and imperfect measurements, small errors in the angular velocity integrated and accumulated into progressively larger errors in angular

position. So, gyro-based angle estimate tends to drift over time. In Figure 2.20.a and 2.20.b, the results for the attitude estimation just by using accelerometer or gyroscope data are shown, respectively.

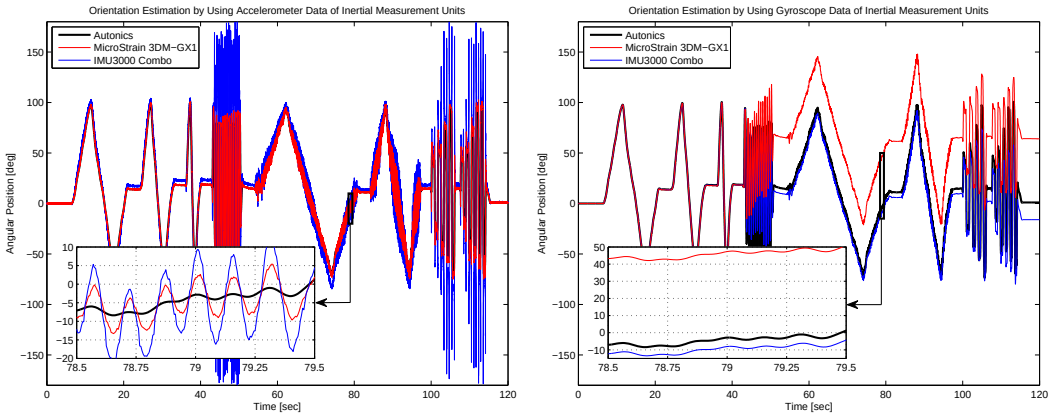


Figure 2.20: The attitude estimation for MicroStrain 3DM-GX1 and IMU3000 Combo by using a) acceleration, b) angular rate data

2.6 Vision Sensor

Hall effect sensor, inertial measurement unit and rotary encoders allow to measure shaft position, as well as base and top plates' angles. In this way, they provide necessary information to define a system error between desired and measured signals, also time-domain specifications like rising and settling time. However, it should be noted that, the main purpose of developing a stabilization platform is to eliminate the disturbances which cause to the unwanted motion blur and affect the image quality adversely. The system error defined between desired and measured signals or time domain specifications does not give an idea about how much an image is degraded due to the imposed motion blur. So, defining a performance metric calculated by using vision data can be an alternative option for quantification process. Therefore, a vision sensor, Logitech C310 series webcam as shown in Figure 2.21, is mounted to the upper plate.



Figure 2.21: Logitech C310 Series Webcam

CHAPTER 3

MODELING EFFORTS OF ACTUATING MECHANISM

One method for obtaining the optimal controller parameters is to assemble the actuating mechanism and the stabilization platform, then to perform hardware-in-the-loop optimizations. However this process may lead to the objectionable experiments which could cause hazardous situation lead to hardware damages due to the mechanical limits of camera platform, especially, while the controller parameters varies between its minimum and maximum limits. Therefore, our primary objective is to derive a realistic simulation model which includes both actuating mechanism and stabilization platform. Thereby, the optimization works that could damage the system, is harmlessly carried out on this simulation model.

In this chapter, the modeling efforts of linear DC servomotor and motor driver are explained in details. Simulation model of actuating mechanism is derived in three phases. Firstly, linear DC servomotor is detached from motor driver and driven directly by using input voltage. Thus, a simulation model covers only linear DC servomotor, which maps the input voltage to the output shaft position, is derived as detailed in Section 3.3. At the second phase, the obtained motor model is attached to the motor driver at velocity control mode. At this operating condition, only inner loop PI-type velocity controller is enabled, whereas, outer loop PD-type position controller and velocity profiler are at non-functioning status. In this way, a simulation model which maps desired shaft velocity to output shaft position is obtained as detailed in Section 3.4. Lastly, at the third phase, motor driver is operated at position control mode, which enables inner and outer loop controllers as well as velocity profiler. In this way, a complete simulation model for actuating mechanism is accomplished to map the desired shaft position to the output shaft position as detailed in Section 3.4.

3.1 Basic Concepts

A linear motor is functionally the same as a rotary motor with the rotor and stator circular magnetic field components laid out in a straight line. Actually, a linear motor is an "unrolled" version of a rotary motor, in which the angular quantities are replaced by translational counterparts and the torque quantity is replaced by force.

As pointed out by Faraday's law, if an external magnetic field is applied across a current-

carrying conductor, then the conductor experiences the Lorentz force in a direction perpendicular to both the directions of the electric current and the magnetic field. Either brushed or brushless, rotary or linear, all kinds of DC motors operate according to this physical fact. They are designed to harness the magnetic interaction between the external magnetic field and the current-carrying conductor to generate motion.

When the motor is energized, the motion is initiated as a result of the repulsive and attractive forces between like and unlike poles of the magnetic fields that are created on the stator and rotor. To maintain the continuance of the movement, the polarity of the dynamic magnetic field have to be changed at the right instance that is shortly after the alignment of unlike poles. In the single-phase motors, this swapping operation is concluded by changing the direction of electric current. On the other hand, in three-phase motors, it is achieved by routing the electric current to a different set of rotor windings. This process is called commutation. DC motors can be classified as brushed and brushless according to the their commutation mechanisms.

At brushed DC (BDC) motors, the static magnetic field that is created on the stator, is formed by permanent magnets or electromagnets with field windings (separately excited, series-wound, shunt-wound, and compound-wound). Moreover, the dynamic magnetic field that is created on the rotor is energized with mechanical structure consisting of brushes and commutator. Brushes are fixed to the stator, whereas the commutator is attached to the rotor and electrically connected to the rotor windings. Brushes, commutator contacts and rotor windings are arranged in such a way that as the rotor moves to align the dynamic magnetic field to the stator's static magnetic field, the brushes come into contact with the different segments of the rotating commutator to provide an electrical connection between DC power source and various armature windings installed on rotor side [10]. Since, the incoming current is commuted mechanically, brushed DC motors can be powered by direct current source. Their control is relatively simple, and is even, no control is required for constant speed operation. However, higher rotor inertia originated from armature windings limits the dynamic characteristics and internal construction reduces the heat dissipation properties. Also, brush friction degrades the speed/torque characteristics and brush arcing generates noise that radiates as Electromagnetic Interference (EMI).

On the other hand, at brushless DC (BLDC) motors, the static magnetic field is created on rotor by permanent magnets and dynamic magnetic field is created on stator by armature windings. Installing the armature windings to the stator, instead of rotor side, eliminates the need for brushes. However, to maintain the continuance of the movement, as in brushed DC motors, the armature windings of the BLDC motors should be energized in a sequence. For that purpose, the commutation of a BLDC motor is controlled electronically.

3.2 Three-Phase Brushless Linear DC Motor Model & One-Phase Equivalent Circuit

For three-phase electronically commutated brushless DC servomotor, it can be shown that voltage equation can be expressed as given in the Equation 3.1 at each phase.

$$u(t) = R \cdot i(t) + (L - M)\frac{di(t)}{dt} + e(t) \tag{3.1}$$

So, the three-phase voltage equation can be expressed in the matrix form as given in Equation 3.2, where, $u_{a,b,c}$ are phase voltages, $i_{a,b,c}$ are phase currents, $e_{a,b,c}$ are induces phase electromotive forces, $R_{a,b,c}$ are phase resistances, $L_{a,b,c}$ are self-inductances of each phase and M is mutual inductance.

$$\begin{bmatrix} u_a \\ u_b \\ u_c \end{bmatrix} = \begin{bmatrix} R_a & 0 & 0 \\ 0 & R_b & 0 \\ 0 & 0 & R_c \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \begin{bmatrix} L_a - M & 0 & 0 \\ 0 & L_b - M & 0 \\ 0 & 0 & L_c - M \end{bmatrix} \frac{d}{dt} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \begin{bmatrix} e_a \\ e_b \\ e_c \end{bmatrix} \tag{3.2}$$

Here, back-EMF is shaft position dependent and can be expressed as given in Equation 3.3, where ν is electrical speed of rotor, ψ_m is maximum value of flux linkage, $f_{a,b,c}(x)$ is back-EMF waveform function, and x is the shaft position.

$$e_k = \nu \psi_m f_k(x), \quad where, k = a, b, c \tag{3.3}$$

The power transferred to the rotor, which is called the electromagnetic power, equals the sum of the product of current and back-EMF of the three phases as given in Equation 3.4.

$$P_e = e_a i_a + e_b i_b + e_c i_c \tag{3.4}$$

Ignoring the mechanical loss and stray loss, the electromagnetic power is totally turned into kinetic energy as given in Equation 3.5, where F_e is electromagnetic force and υ is shaft velocity.

$$P_e = F_e \upsilon \tag{3.5}$$

Hence, by using Equation 3.4 and 3.5, electromagnetic force can be expressed as given in Equation 3.6.

$$F_e = \frac{e_a i_a + e_b i_b + e_c i_c}{\upsilon} \tag{3.6}$$

Substituting Equation 3.3 into Equation 3.6 another form of the force equation can be represented as given in Equation 3.7, where p is the pole number.

$$F_e = p\left[\psi_m f_a(x) i_a + \psi_m f_b(x) i_b + \psi_m f_c(x) i_c\right] \tag{3.7}$$

It can be shown that, if the BLDC motor runs in the 120 conduction mode and the corresponding transient commutation process is ignored, then Equation 3.7 can be further simplified as expressed in Equation 3.8, where K_f is force constant and i is steady-state current [73].

$$F_e = K_f i \tag{3.8}$$

The force generated by the motor accelerates the motor shaft, as well as any additional load attached. Some of the force also goes towards overcoming the friction. For a mechanical system undergoing pure translational motion, Newton's first law can be applied as given in Equation 3.9, where F_l load force, M is rotor mass, and B is viscous friction coefficient.

$$F_e - F_l - B\upsilon = M\frac{d\upsilon}{dt} \tag{3.9}$$

Substituting Equation 3.8 into Equation 3.9 yields that;

$$K_f i - F_l = M\frac{d\upsilon}{dt} + B\upsilon \tag{3.10}$$

Consequentially, Equation 3.2 togethers with Equation 3.10, gives complete mathematical model of the three-phase brushless linear DC servomotor. By using these differential equations, simulation model for the BLDC motor is implemented as shown in Figure 3.1. However, simulation speed of this complex model is relatively slow, hence it does not suitable for performing optimization works. So, it is decided to obtain a simpler model, which generates hopefully the same results at higher simulation speed.

Actually, the BLDC motor is usually designed to be three-phase or multiphase. However, for each conducted phase winding, the mechanisms of back-EMF and electromagnetic torque are all the same with those of the traditional brushed DC motor. Thus similar analysis methods can be adopted and mathematical equations can be derived for single-phase equivalent circuit same as in brushed DC motor.

The line voltage equation can be obtained by subtracting the phase voltage equations as given in Equation 3.11.

$$\begin{bmatrix} u_{ab} \\ u_{bc} \\ u_{ca} \end{bmatrix} = \begin{bmatrix} R & -R & 0 \\ 0 & R & -R \\ -R & 0 & R \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \begin{bmatrix} L-M & M-L & 0 \\ 0 & L-M & M-L \\ M-L & 0 & L-M \end{bmatrix} \frac{d}{dt} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \begin{bmatrix} e_a - e_b \\ e_b - e_c \\ e_c - e_a \end{bmatrix} \tag{3.11}$$

Suppose that the windings of phase A and B are conducted, then according to the Kirchhoff's Law, it can be simply written that;

$$\begin{aligned} i_a &= -i_b = i \\ \frac{di_a}{dt} &= -\frac{di_b}{dt} = \frac{di}{dt} \end{aligned} \tag{3.12}$$

Thus the line voltage in Equation 3.11 can be rewritten as;

$$u_{ab} = 2Ri + 2\left(L - M\right)\frac{di}{dt} + \left(e_a - e_b\right) \tag{3.13}$$

If the transient response are ignored, then the steady-state value of e_a and e_b are equal in amplitude, and opposite in direction. So, Equation 3.13 can be expressed as;

$$u_{ab} = U_d = 2Ri + 2\left(L - M\right)\frac{di}{dt} + 2e_a = \hat{R}i + \hat{L}\frac{di}{dt} + K_e\upsilon \tag{3.14}$$

where, $\hat{R}$ is equivalent line resistance, $\hat{L}$ is equivalent line inductance, K_e back-EMF constant [73]. By assuming motor operates at no-load, Equation 3.10 can be written as;

$$i = \frac{M}{K_f}\frac{d\upsilon}{dt} + \frac{B}{K_f}\upsilon \tag{3.15}$$

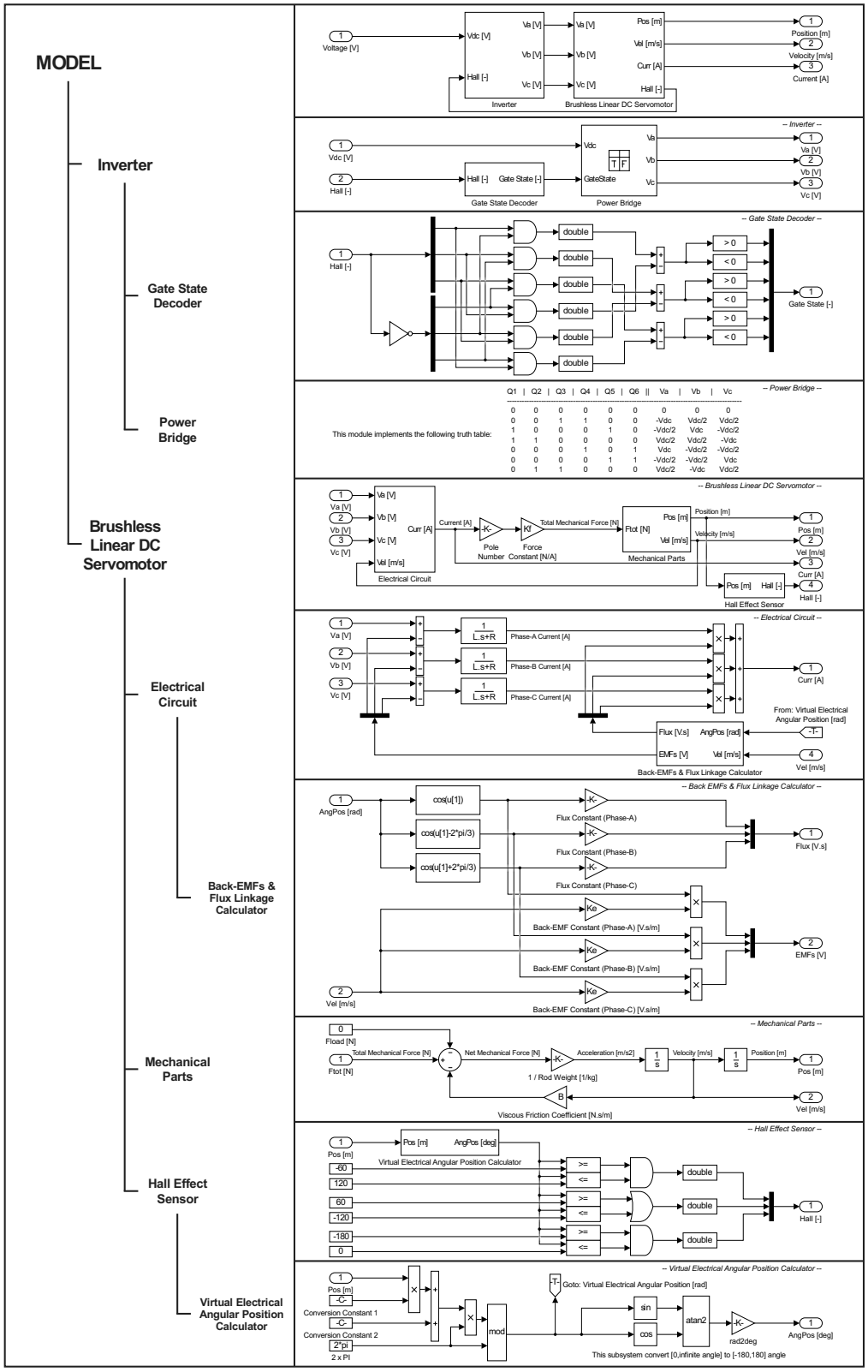


Figure 3.1: Exact simulation model for electronically commutated three-phase brushless linear DC servomotor with Hall effect sensors

Substituting Equation 3.15 into Equation 3.14 yields that;

$$\begin{aligned} U_d &= \hat{R}\left(\frac{M}{K_f}\frac{dv}{dt} + \frac{B}{K_f}v\right) + \hat{L}\frac{d}{dt}\left(\frac{M}{K_f}\frac{dv}{dt} + \frac{B}{K_f}v\right) + K_e v \\ &= \frac{\hat{L}M}{K_f}\frac{d^2v}{dt^2} + \frac{\hat{R}M + \hat{L}B}{K_f}\frac{dv}{dt} + \frac{\hat{R}B + K_e K_f}{K_f}v \end{aligned} \tag{3.16}$$

By assuming states are initially zero, the Laplace transformation of Equation 3.16 gives the transfer function between bus voltage and shaft velocity as represented in Equation 3.17.

$$G_v = \frac{V(s)}{U(s)} = \frac{K_f}{\hat{L}Ms^2 + \left(\hat{R}M + \hat{L}B\right)s + \hat{R}B + K_e K_f} \tag{3.17}$$

Also, transfer function between bus voltage and shaft position can be given as represented in Equation 3.18.

$$G_x(s) = \frac{X(s)}{U(s)} = \frac{K_f}{\hat{L}Ms^3 + \left(\hat{R}M + \hat{L}B\right)s^2 + \left(\hat{R}B + K_e K_f\right)s} \tag{3.18}$$

According to the obtained transfer function between bus voltage and shaft position, the simulation model is implemented as shown in Figure 3.2.

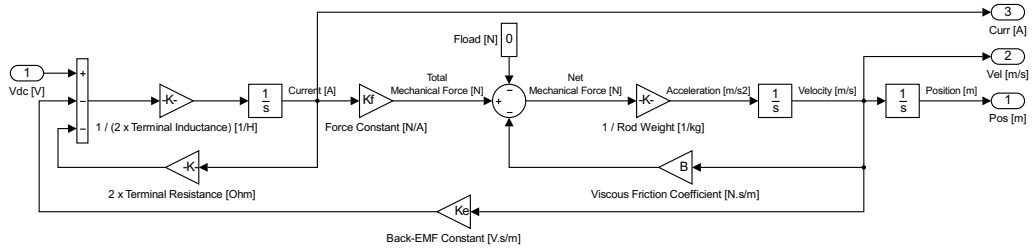


Figure 3.2: Single-phase equivalent circuit simulation model for electronically commutated three-phase brushless linear DC servomotor

For comparison purposes, various voltage signals are applied to three-phase exact model and single-phase equivalent circuit model, then the shaft positions are examined as shown in Figure 3.3. The results indicate that the generated outcomes of these models are almost identical. So, the single-phase equivalent circuit model, which runs much faster, can be used without any difference for optimization works.

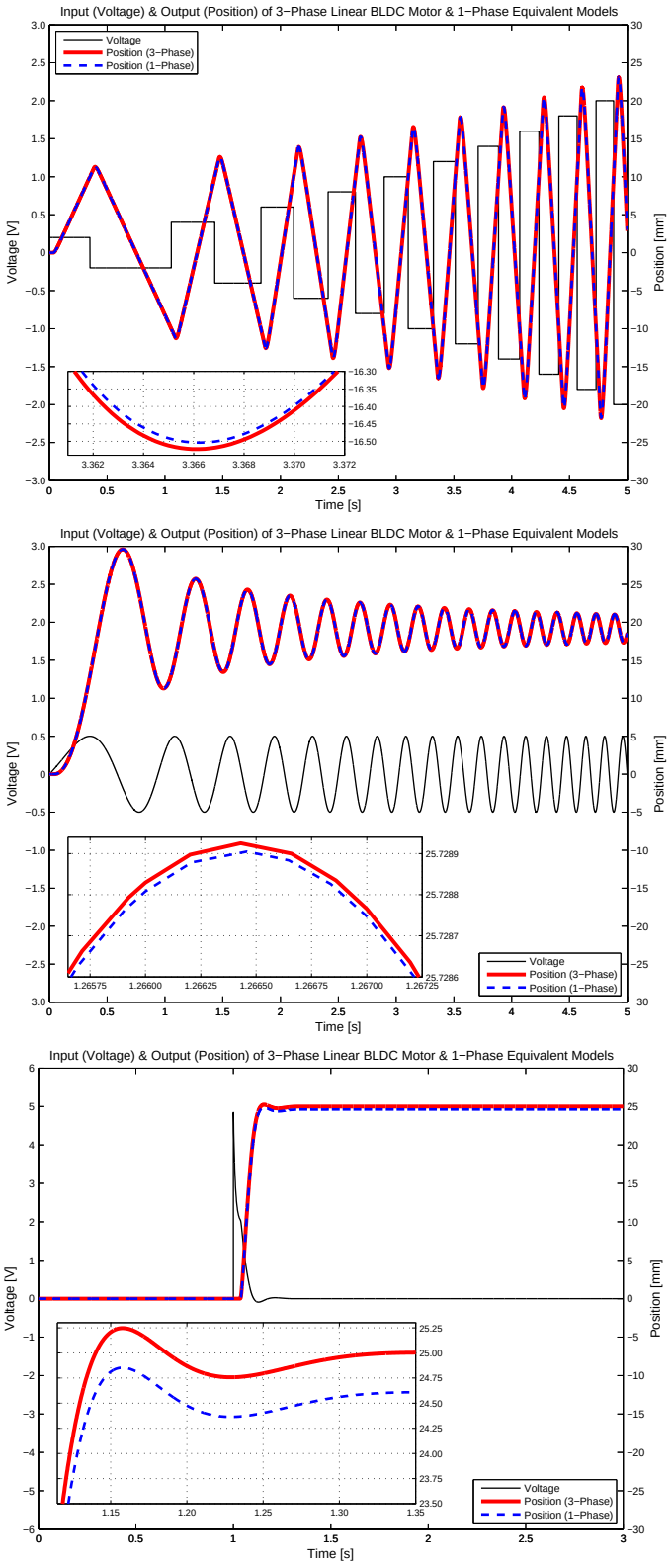


Figure 3.3: Simulated shaft position according to the various voltage signals for three-phase exact model and single-phase equivalent circuit model

3.3 System Identification for Linear DC Servomotor

Although, a detailed simulation model is obtained, still there may exist unmodeled dynamics or unknown physical interactions. Also, some assumptions, approximations or simplifications can be done during modeling works. Even factory specified motor parameters may not reflect the exact values. So, in this section, various system identification techniques are applied to find the exact motor parameters and, if any, to reveal the unmodelled dynamics. To find the most accurate and most appropriate model, linear model based and linear data-driven identification techniques are studied as well as nonlinear counterparts as represented in Figure 3.4.

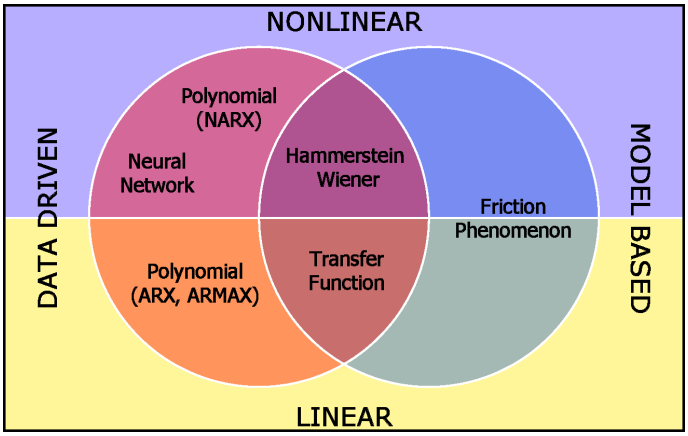


Figure 3.4: Classification of system identification techniques depending on whether linear or nonlinear, as well as model based or data driven

3.3.1 Experimental Measurements & Frequency Spectrum Analysis

Independently from the controller model, it is aimed to obtain a self-directed motor model, which accepts voltage as input and generates shaft position as output. If controllers are in functioning and operational, even for different motor parameter sets, they generate a *"suitable"* control action, which lead the motor states to the same point in the state-space.

For this purpose, PD and PI type position and velocity controllers are disabled and motor driver is operated at voltage inverter mode, which generates a PWM signal for given DC voltage level.

In this way, Faulhaber LM1247 linear DC servomotor is driven directly by input voltage and corresponding shaft position is examined. The measurements, as shown in Figure 3.6, divided into training and validation datasets such that to perform the identification works at the widest possible frequency spectrum with the smallest possible number of sets. At these figures, the axes at the left side represents the input voltage, while the axes at the right side represents the output shaft position.

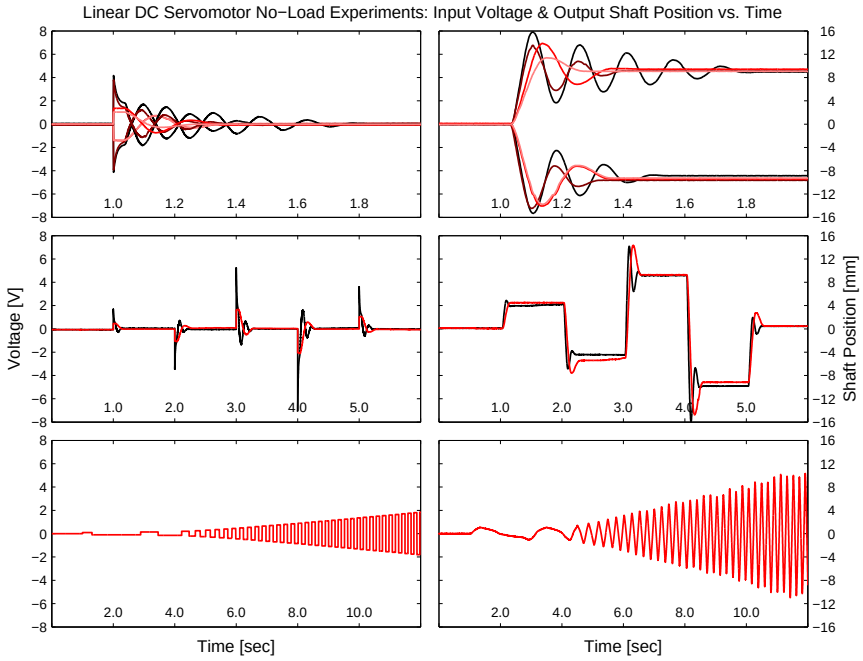


Figure 3.5: Training set, which consist of input voltage (left) and output shaft position (right) gathered from Faulhaber LM1247 brushless linear DC servomotor. The data, which has similar characteristics grouped together.

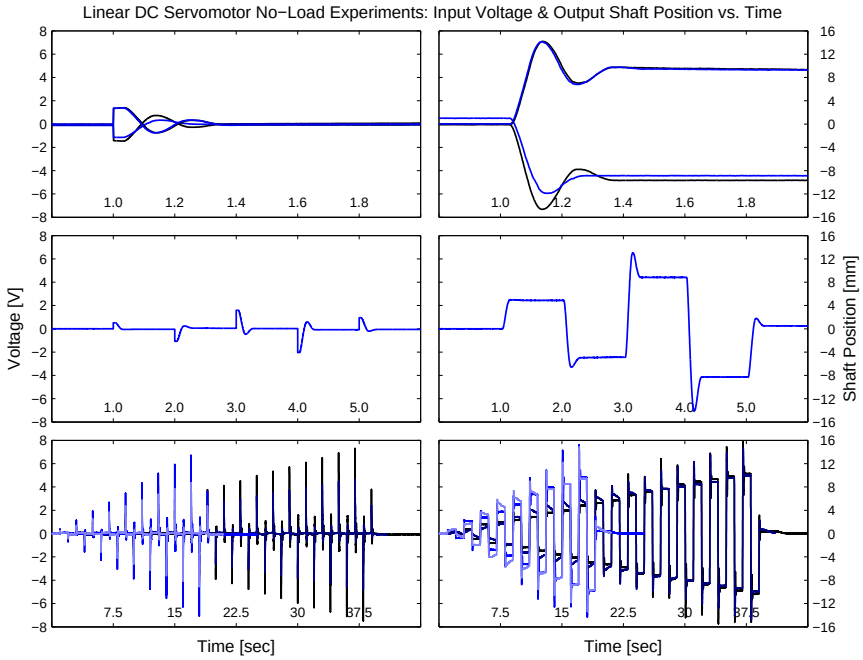


Figure 3.6: Validation set, which consist of input voltage (left) and output shaft position (right) gathered from Faulhaber LM1247 brushless linear DC servomotor. The data, which has similar characteristics grouped together.

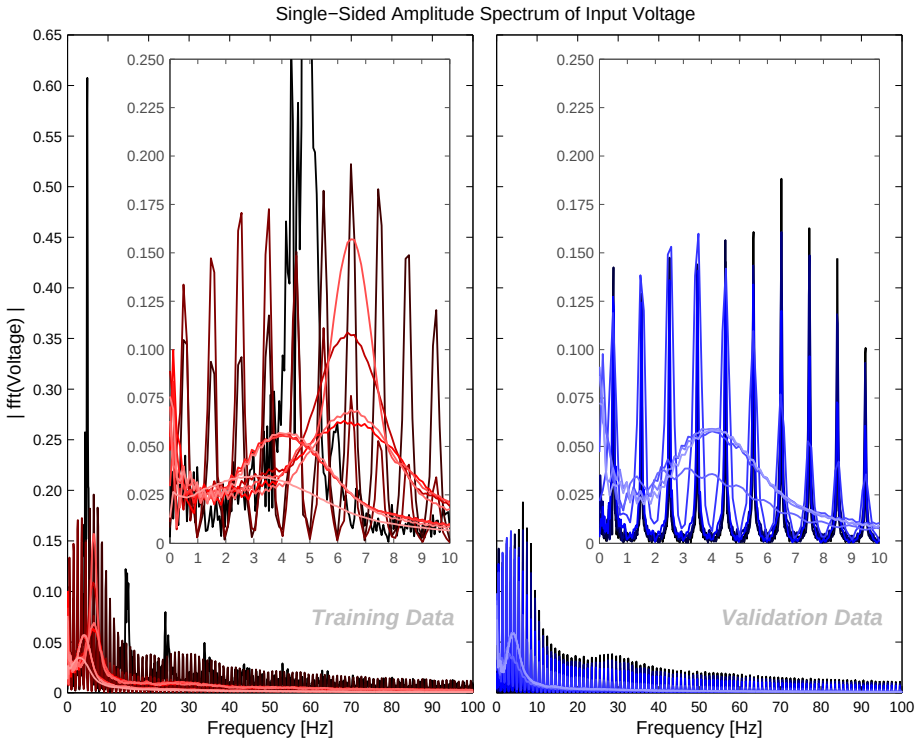


Figure 3.7: Single-sided power spectrum of training and validation datasets

3.3.2 Model Based Linear Identification

3.3.2.1 Transfer Function

By substituting factory specified parameters into Equation 3.18 and by integrating known system delay of approximately ~40 [ms], the transfer function is obtained for Faulhaber LM1247 brushless linear DC servomotor as given in Equation 3.19. The measured and simulated shaft positions generated by using the given transfer function is shown in Figure 3.8.

$$\frac{X(s)}{U_d(s)} = e^{-0.040s} \cdot \frac{2.1782 \cdot 10^5}{s^3 + 1.6072 \cdot 10^4 s^2 + 1.3220 \cdot 10^6 s} \tag{3.19}$$

It is obvious that, the simulation model does not behave as expected. The potential problems can be listed as follows;

- Linearity assumption and model order may be sufficient, but parameters are inaccurate
- Linearity assumption may be sufficient, but model order have to be increased due to the ignored and unmodeled relationships (i.e., ripple forces, frictional forces, physical imperfections)
- Linearity assumption may not be sufficient to represents the actual plant

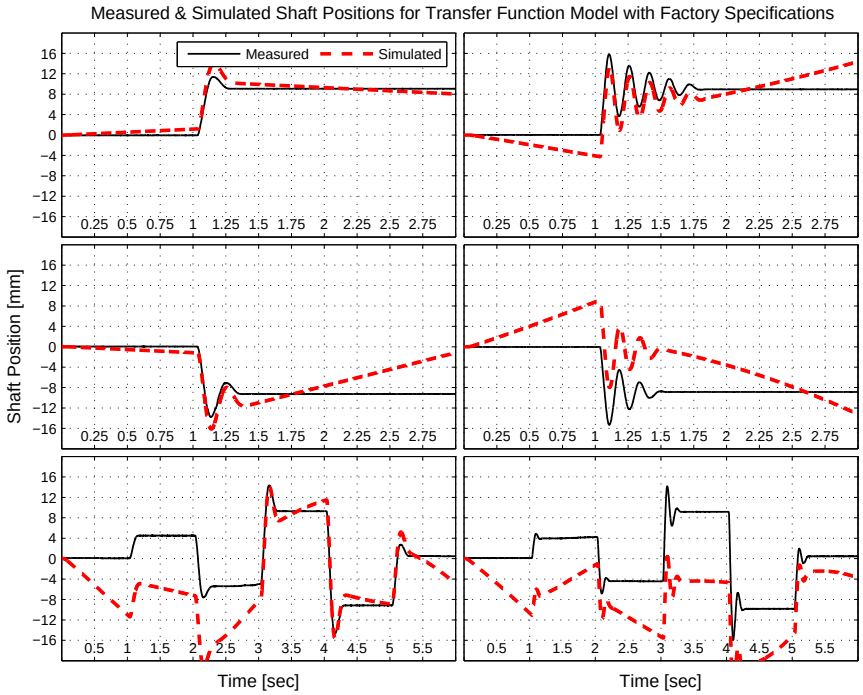


Figure 3.8: Measured and simulated shaft positions. The simulated results are obtained by substituting factory specified motor parameters into derived transfer function.

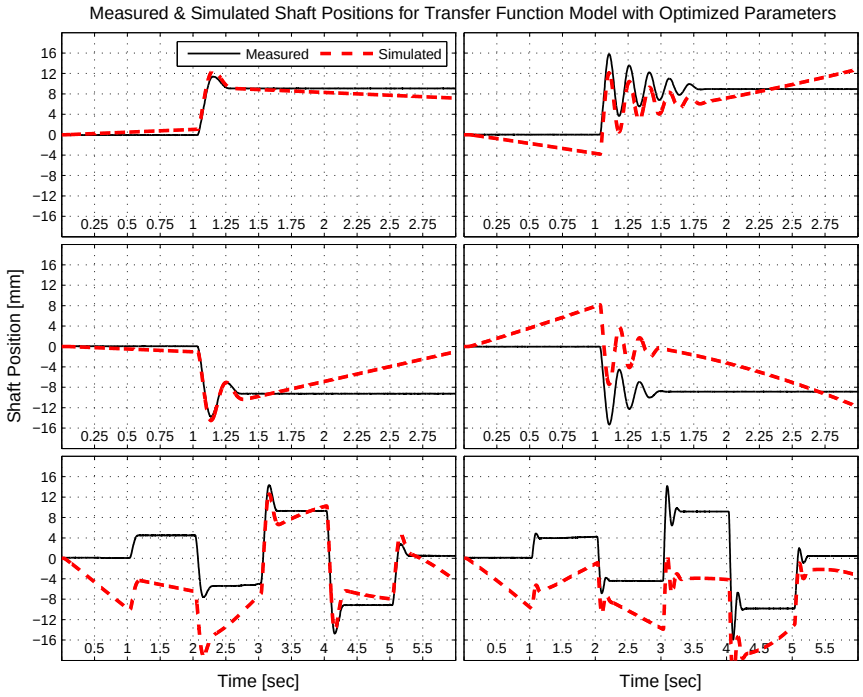


Figure 3.9: Measured and simulated shaft positions. The simulated results are obtained by substituting optimized motor parameters into derived transfer function.

In order to understand whether the source of the problem is related to the inaccurate motor parameters or not, let optimize them. For this purpose, the third order transfer function, which is given in Equation 3.18 is subjected to the parameter optimization. Due to the knowing of delay exactly and undoubtedly, the value of the dead time is excluded from optimization process. The parameters are searched within $\pm 75\%$ of each value, except viscous friction coefficient that is relaxed and totally free. After the optimization process, the motor parameters are calculated as given in Table 3.1.

Table3.1: Factory specified and optimized parameters for Faulhaber LM1247 DC servomotor

Parameter	**Default Value**	**Optimized Value**	**Unit**
Armature Resistance	13.17	14.01	$[\Omega]$
Armature Inductance	8.20×10^{-4}	1.06×10^{-3}	$[H]$
Force Constant	6.43	9.12	$[N/A]$
Back$-$EMF Constant	5.25	5.94	$[V/m/s]$
Viscous Friction Coefficient	0.20	0.28	$[N/m/s]$
Shaft Mass	18.00×10^{-3}	20.01×10^{-3}	$[kg]$

By substituting optimized parameters into Equation 3.18, the transfer function is obtained as given in Equation 3.20. The measured and simulated shaft positions generated by using the given transfer function is shown in Figure 3.9. The simulation results for optimized parameters show that, the problem can not be solved with a simple parameter optimization process.

$$\frac{X(s)}{U_d(s)} = e^{-0.040s} \cdot \frac{2.1605 \cdot 10^5}{s^3 + 1.3284 \cdot 10^4 s^2 + 1.4696 \cdot 10^6 s} \tag{3.20}$$

3.3.3 Data Driven Linear Identification

3.3.3.1 Transfer Function

At this section, in order to understand whether the source of the problem is related with model order or not, let derive higher order data driven transfer functions by using System Identification Toolbox of MATLAB computation environment. To try the various combination of pole-zero configurations, the number of poles is varied between 2 and 12, while the number of zeros is varied between 0 and pole number. The smallest modeling error is achieved for 7^{th} order transfer function which is given in Equation 3.21. The measured and simulated shaft positions generated by using the given transfer function is shown in Figure 3.10. The simulation results show that, the modeling error can not be solved by increasing the order.

$$\frac{X(s)}{U_d(s)} = e^{-0.040s} \cdot \frac{1.1785 \cdot 10^6 s^3 + 2.8627 \cdot 10^6 s^2 + 3.4203 \cdot 10^6 s + 4.6633 \cdot 10^5}{s^7 + 552.6043 s^6 + 3.9613 \cdot 10^4 s^5 + 9.9533 \cdot 10^6 s^4 + ...}$$
$$... + 2.1264 \cdot 10^7 s^3 + 4.2395 \cdot 10^7 s^2 + 8.8905 \cdot 10^5 s + 1.7553 \cdot 10^6 \tag{3.21}$$

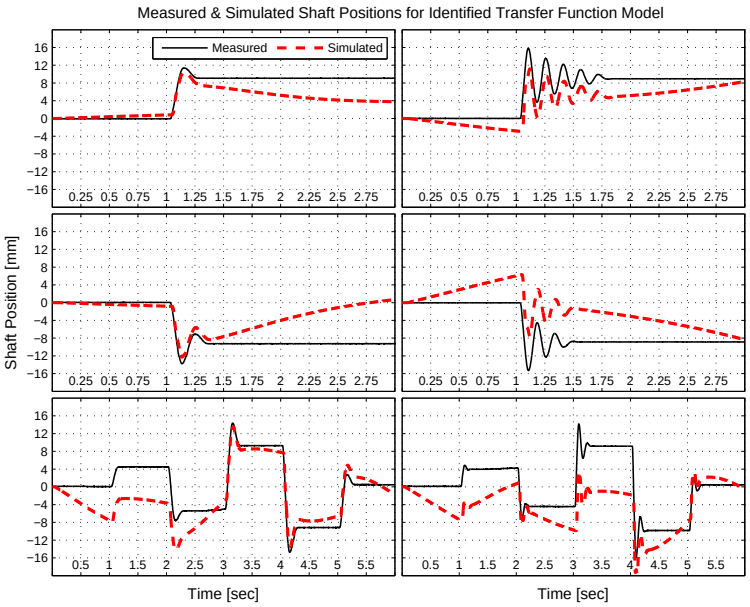


Figure 3.10: Measured and simulated shaft positions. The simulated results are obtained by using data driven 7^{th} order transfer function.

3.3.3.2 Polynomial Model

A polynomial model uses a generalized notion of transfer functions to express the relationship between the input, $u(t)$, the output $y(t)$, and the noise $e(t)$ using the Equation 3.22, where the variables A, B, C, D, and F are polynomials expressed in the time-shift operator q^{-1}, u_i is the i^{th} input, nu is the total number of inputs, k_i is the i^{th} input delay. The identified model is given in Equation 3.23, which is the generalized case of *ARX*, *ARMAX* and *Box-Jenkins* models.

$$A(q)y(t) = \sum_{i=1}^{nu} \frac{B_i(q)}{F_i(q)} u_i(t - k_i) + \frac{C(q)}{D(q)} e(t) \tag{3.22}$$

$$\begin{aligned}
A(q) &= 1 - 1.132q^{-1} + 0.09594q^{-2} - 0.7971q^{-3} + 0.8327q^{-4} \\
B(q) &= 1.882 \cdot 10^{-5}q^{-36} + 1.917 \cdot 10^{-5}q^{-37} + 2.941 \cdot 10^{-6}q^{-38} - 1.041 \cdot 10^{-5}q^{-39} \\
&\quad - 1.786 \cdot 10^{-5}q^{-40} + 5.235 \cdot 10^{-6}q^{-41} + 8.719 \cdot 10^{-6}q^{-42} \\
&\quad - 1.435 \cdot 10^{-6}q^{-43} - 1.185 \cdot 10^{-5}q^{-44} - 1.332 \cdot 10^{-5}q^{-45} \\
C(q) &= 1 + 0.2482q^{-1} - 0.1543q^{-2} - 0.8241q^{-3} - 0.2653q^{-4} \\
D(q) &= 1 - 2.04q^{-1} + 0.1458q^{-2} + 1.828q^{-3} - 0.9342q^{-4} \\
F(q) &= 1 + 0.4695q^{-1} - 0.5032q^{-2} + 0.02539q^{-3} - 0.2918q^{-4} + 0.4424q^{-5} \\
&\quad + 0.3469q^{-6} - 0.6805q^{-7} - 0.2535q^{-8} - 0.003253q^{-9}
\end{aligned} \tag{3.23}$$

The measured and simulated shaft positions generated by using the identified polynomial model is shown in Figure 3.11. The simulation results show that, the identified model is insufficient to represent the actual plant dynamics.

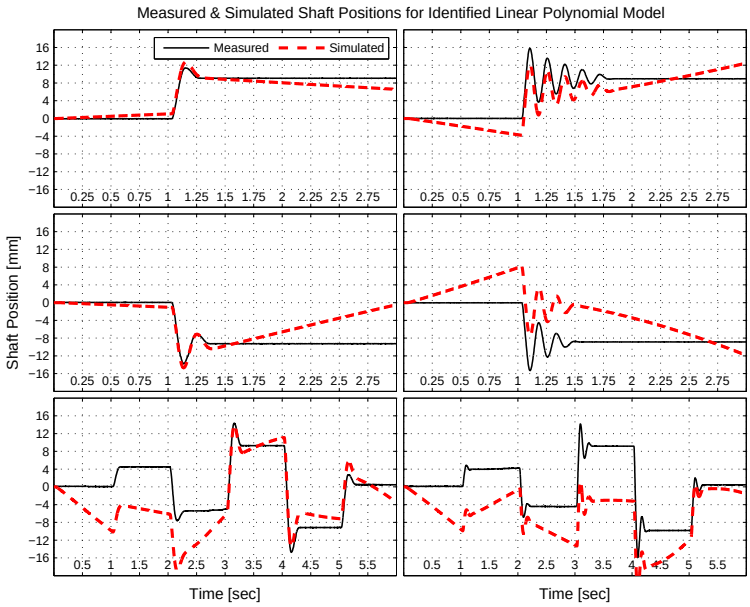


Figure 3.11: Measured and simulated shaft positions. The simulated results are obtained by using the identified polynomial model.

3.3.4 Data Driven Nonlinear Identification

3.3.4.1 Polynomial Model

At this section, a nonlinear autoregresive with external input (NARX) model is derived by using identification techniques. A nonlinear ARX model can be best understood by examining the linear counterpart, which is given in Equation 3.24.

$$y(t) + a_1 y(t-1) + a_2 y(t-2) + ... + a_{na} y(t-na) = \\ b_1 u(t) + b_2 u(t-2) + ... + b_{nb} u(t-nb+1) + e(t) \tag{3.24}$$

This structure implies that the current output, $y(t)$, is predicted as a weighted sum of past output values, current input value, and past input values. Rewriting the equation as product of vectors yields;

$$y(t) = [-a_1, -a_2, ..., -a_{na}, b_1, b_2, ..., b_{nb}] \cdot \\ [y(t-1), y(t-2), ..., y(t-na), u(t), u(t-1), ..., u(t-nb-1)]^T \tag{3.25}$$

where, $y(t-1)$, $y(t-2)$, ..., $y(t-na)$, $u(t)$, $u(t-1)$, $u(t-2)$, ..., $u(t-nb+1)$ are delayed output and input variables, which are called regressors. The linear ARX model thus predicts the current output $y(t)$ as a weighted sum of its regressors. The nonlinear ARX model extends this structure and has a more flexible nonlinear mapping function as given in Equation 3.26, where f is a nonlinear function like wavelet, sigmoid or neural network type estimators. Also,

in nonlinear ARX model, new regressors can be defined by using the past output and input variables, like $y(t-1)^2$, $y(t-2) \cdot u(t-1)$, $tan\,(u(t-2))$, etc.

$$y(t) = f\,(y(t-1), y(t-2), y(t-3), ...u(t), u(t-1), .u(t-2), ..) \tag{3.26}$$

The structure of a nonlinear ARX model is depicted in the block diagram given in Figure 3.12.

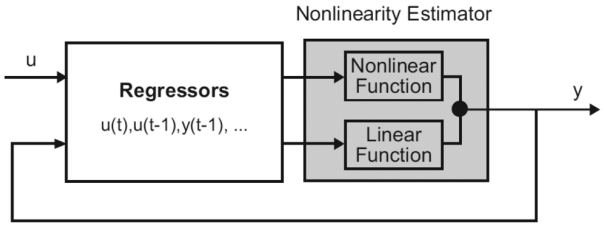


Figure 3.12: Structure of nonlinear autoregresive with external input (NARX) model

The measured and simulated shaft positions generated by using the identified nonlinear ARX model is shown in Figure 3.13. The simulation results show that, the identified nonlinear ARX model represents the system dynamics in a better way compared to linear models. Consequently, it can be said that without any doubt, there is an unknown nonlinearty in the actual plant.

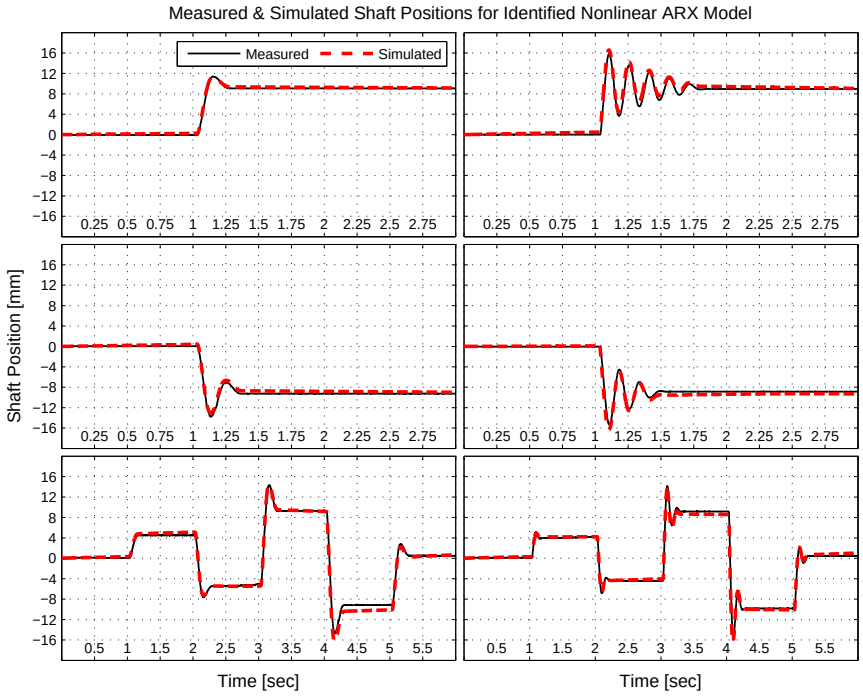


Figure 3.13: Measured and simulated shaft positions. The simulated results are obtained by using the identified nonlinear ARX model.

3.3.4.2 Hammerstein-Wiener Model

Although, the composition of some systems has nonlinear nature, they can be separated into static nonlinearities and linear dynamics. Especially, electro-mechanical systems have nonlinearities due to the actuator limits, despite the fact that having linear dynamics. The Hammerstein-Wiener model, which is illustrated in Figure 3.14, achieves this configuration as a series connection of static nonlinear structures with a dynamic linear model. Modeling the system by making this kind of decomposition might be more appropriate than obtaining a nonlinear ARX model, which has no physical meaning.

Hammerstein-Wiener structure is convenient for both grey-box and black-box identification approaches. At grey-box modeling, linear dynamics can be derived by using first principle. In this case, only input and/or output nonlinearities are subjected to the identification studies. On the other hand, at black-box approach, both linear dynamics and static nonlinearities are included for identification. In addition, according to the existence of input or output static nonlinearities, this method has two special cases; It is named as Hammerstein model or Wiener model, if only input or output nonlinearity is operative, respectively.

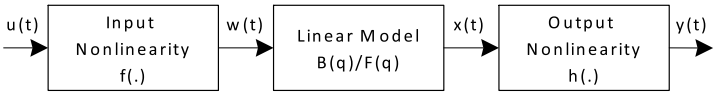


Figure 3.14: Structure of nonlinear Hammerstein-Wiener model

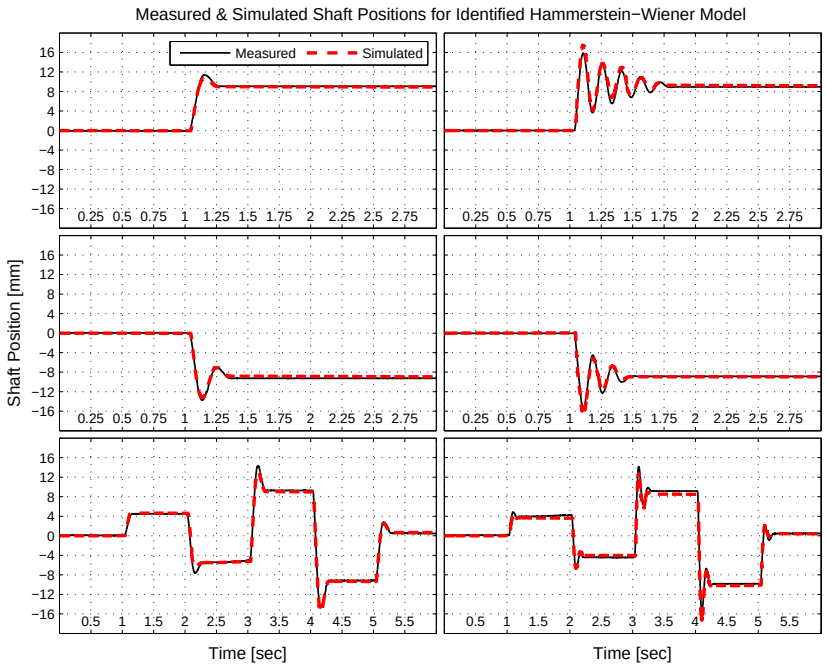


Figure 3.15: Measured and simulated shaft positions. The simulated results are obtained by using the identified hammerstein-wiener model.

Note that, during system identification of Hammerstein-Wiener model, various structures are examined to find the best goodness of fit value. Piecewise-linear, wavelet, sigmoid, saturation, dead-zone and polynomial functions, as well as unit gain are tried for both input and output nonlinearities. Among them, the minimum modeling error is obtained for a Hammerstein model with piecewise-linear function for input nonlinearity, 4^{th} order dynamics for linear structure and unit gain for output nonlinearity. The measured and simulated shaft positions generated by using the identified Hammerstein model is shown in Figure 3.15. The simulation results are quite similar to one, which are obtained for nonlinear ARX model. However, Hammerstein model gives a better insight about the actual plant. By interpreting the structure of this model, it can be able to understand that, why the linear models are insufficient to represent the physical system dynamics.

3.3.5 Model Based Nonlinear Identification

3.3.5.1 Stiction Effect and Piecewise Linear Coulomb & Viscous Friction Coefficients

At previous sections for the case of transfer function with viscous friction model, it is observed that the simulated shaft position accelerates, as shown in Figure 3.8, Figure 3.9, Figure 3.10, although the actual system remains at rest. Inconsistency between actual plant and simulation model outputs might be due to the stiction effect and more complex friction phenomenon exist on physical system. So in this section, multifaceted friction model as shown in Figure 3.16, is subjected to the identification process. At this model, stiction coefficient is a scalar, while the viscous and coulomb friction coefficients are piecewise linear functions of shaft velocity.

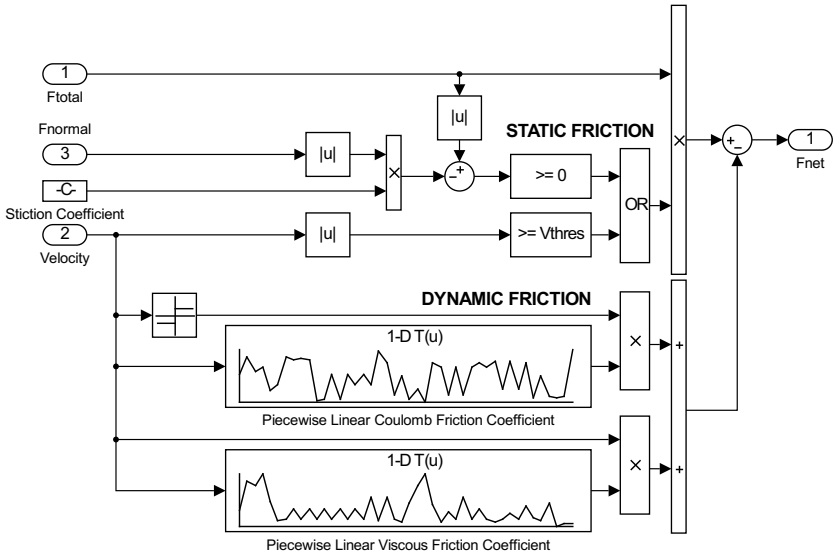


Figure 3.16: Structure of multifaceted model with stiction effect and piecewise linear coulomb & viscous friction coefficients

The measured and simulated shaft positions generated by using the multifaceted friction model is shown in Figure 3.17. Eventhough, the results are not exactly matching, it is obvious that identified complex friction model performs better and reduces the modeling error compared to the viscous friction model.

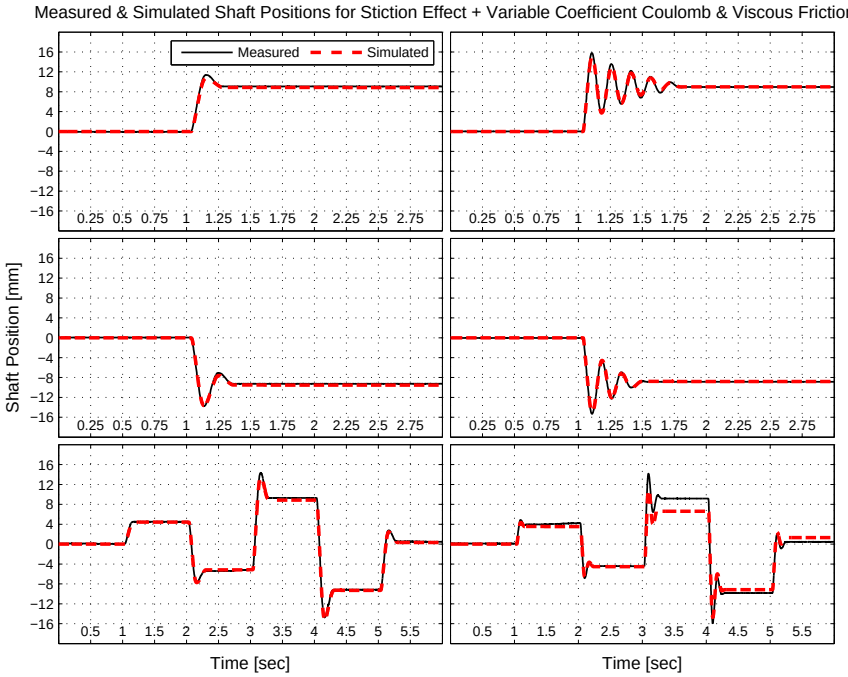


Figure 3.17: Measured and simulated shaft positions. The simulated results are obtained by using multifaceted friction model.

3.3.5.2 Static Input Nonlinearity

As will be remembered, in section 3.3.4.2, a Hammerstein model is obtained with a piecewise linear function for input nonlinearity and 4^{th} order dynamics for linear structure, as well as unit gain for output nonlinearity. While acquiring this model, it is assumed that there is no a priori knowledge about system and so, black-box approach is applied. Although, there is an unmodeled dynamics at the plant, it is not completely unknown. Still, derived model by using first principle gives an insight about system nature. Even if, inspiring by Hammerstein approach and applying grey-box technique, unknown dynamics of the system can be revealed. For this purpose, in this section, single-phase equivalent circuit model is used by substituting factory specified parameters and it is assumed that whole uncertainties, unknowns and unmodeled dynamics are incorporated at the input nonlinearity, as shown in Figure 3.18. A piecewise constant function that is dependent to the input voltage is subjected to the system identification process. The domain of this function is defined as $[-30, 30]$, which is the rated motor voltage. Also, the range of this function is defined as $[0, 1]$, so it is only allowed to degradation of input voltage.

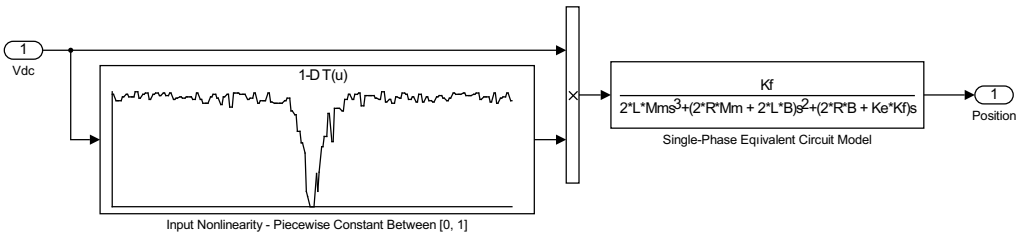


Figure 3.18: Structure of static input nonlinearity and single-phase equivalent circuit model with factory specified parameters

The optimized function converges to such values that for small input voltages, $[-0.1, 0.1]$, it behaves like a dead-zone nonlinearity, as well as, for relatively higher input voltages, it behaves like a saturation with value of 1. These observations coincide with the nonlinearities that may exist in actual plant. The measured and simulated shaft positions generated by using the identified static input nonlinearity is shown in Figure 3.19.

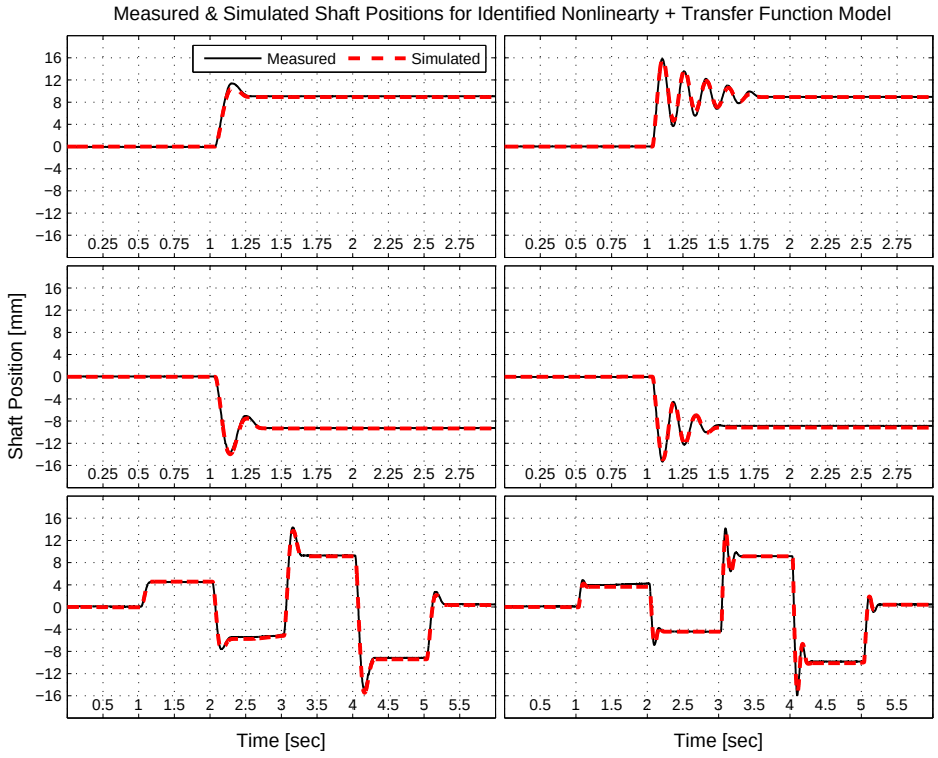


Figure 3.19: Measured and simulated shaft positions. The simulated results are obtained by using the static input nonlinearity and single-phase equivalent circuit model.

3.4 System Identification for Attached Motor Driver & Linear DC Servomotor

Due to the better performance and relatively simpler structure, single-phase equivalent circuit model with static input nonlinearity is selected among others for the remaining of this study. Once the detached motor model is obtained, it can be now connected to the controller structures to accomplish the modeling works of the entire actuating mechanism. For this purpose, the motor model is firstly attached to the inner loop PI-type velocity controller. After validating the unified model, it is connected to the PD-type position controller to form the double-loop structure running on motor driver.

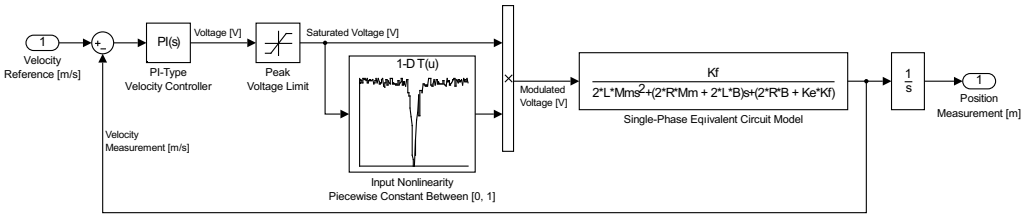


Figure 3.20: Measured and simulated shaft positions. The simulated results are obtained by using the static input nonlinearity and single-phase equivalent circuit model.

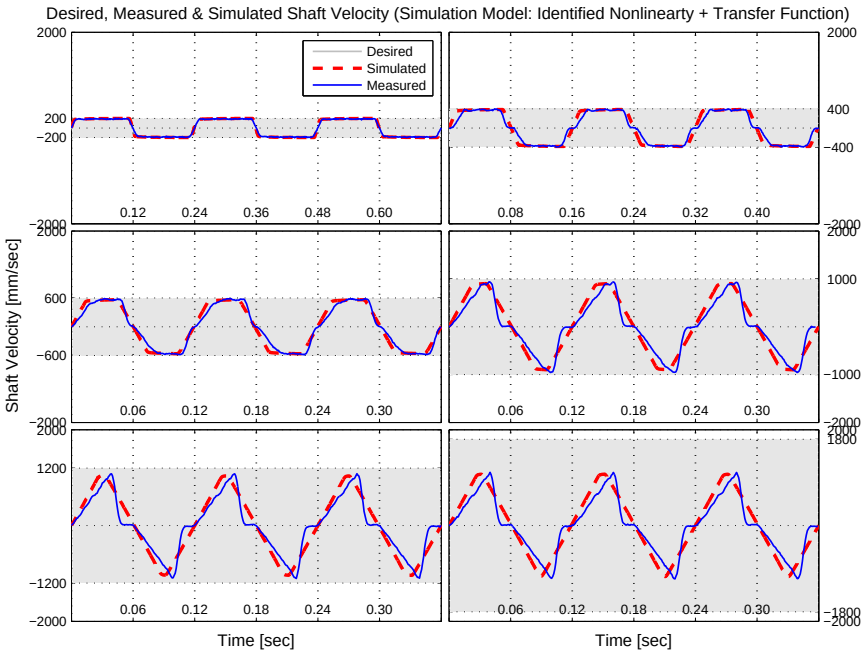


Figure 3.21: Measured and simulated shaft positions. The simulated results are obtained by using the static input nonlinearity and single-phase equivalent circuit model.

CHAPTER 4

MODELING EFFORTS OF MECHANICAL STRUCTURE

The camera stabilization mechanism, which is inspired by well-known Stewart Platform, can be roughly described as a closed chain parallel manipulator. The translational degrees-of-freedom are excluded from the 6-DOF Stewart Platform by removing the three actuators, that leads to the reduced 3-DOF structure with rotational degrees-of-freedom.

It is designed and constructed in a modular fashion which can be constituted two different layouts, namely parallel and symmetric. Both of these configurations have three links, which consist of linear motors, joints and other auxiliary structural elements. Parallel and symmetric layouts can be configured by changing the connection points of these links between base and top platforms. At parallel layout, as the name suggests, the links are placed in a parallel form. However, at symmetric one, they are adapted to a more compact and intricate way. Moreover, in each link that connects the base and top plates, spherical or universal type joints can be used as connecting elements. Consequently, four different layouts can be formed by changing the connection points and included joints of the links. Also, a simplified planar version of the mechanism is designed by using only a single link, which consist of revolute joints instead of spherical or universal joints. In this configuration, the top plate, which has only one rotational degree-of-freedom about pitch axis, is actuated with a single linear motor. For different configurations, the top platform's range of motion about roll, yaw and pitch axes are given in Table 4.1. These layouts are explained and compared in details at the following subsections.

Table4.1: The top platform's range of motion about roll, yaw and pitch axes for different configurations

Configuration	**Roll**	**Yaw**	**Pitch**	**Unit**
Planar Layout with Revolute Joint	-	-	[-14, +14]	[deg]
Parallel Layout with Spherical Joint	[-13, +13]	[-9.5, +9.5]	[-13, +13]	[deg]
Symmetric Layout with Spherical Joint	[-13, +13]	[-2, +16]	[-13, +13]	[deg]
Parallel Layout with Universal Joint	[-13, +13]	[-8.5, +8.5]	[-13, +13]	[deg]
Symmetric Layout with Universal Joint	[-13, +13]	[-11, +11]	[-13, +13]	[deg]

It is not easy to obtain an analytical model of such a complex system successfully without making simplifications and approximations. In such situations, multibody modeling tools can be used to obtain realistic models for dynamical systems. MATLAB's multi-body modeling toolbox, SimMechanics, is used for modeling and simulation of complex dynamical systems with great success as understood from related literature. For example, in [15], a SimMechanics model was developed for a modular truck semi-trailer and validated using a wide range of measurement data obtained from measurements on a real air sprung tractor semi-trailer. Therefore, instead of attempting to derive dynamic equations of the system analytically, it is decided to obtain SimMechanics model as detailed in the subsequent sections. The first step of the implementation is to obtain the dimensions, mass and inertia properties. For this purpose, the mass properties of the produced parts are measured by a precision weighing of accuracy 1 [g]. Also, for dimensions and inertia properties, Solidworks CAD models of are used as summarized in Appendix B.1.

4.1 Planar Layout Equipped with Revolute Joints

It is considered that, at the initial steps, instead of dealing with the complex three-dimensional control and stabilization problem, it would be more instructive to deal with a simplified case.

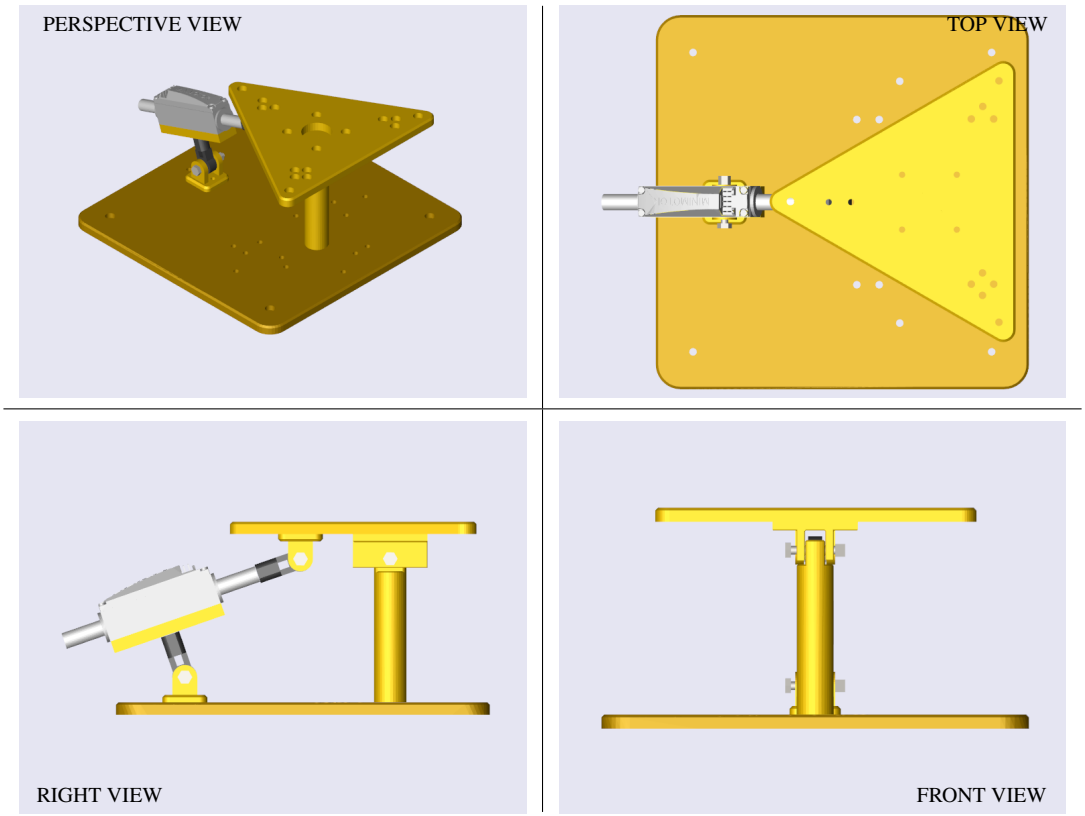


Figure 4.1: SimMechanics model of the planar layout equipped with revolute joints

So, a modified version of the stabilization mechanism is designed and constructed by limiting the axis of movement and by reducing the number of actuators. At this layout, the top plate, which is connected to the vertical support column by using a revolute joint, has one degree-of-freedom about pitch axis only. Also, linear motor is connected to the top and base plates by using revolute joints as shown in Figure 4.1. At this configuration, there are three revolute joints (1R) and one prismatic joint (1P). Also, there are four linkages which connect the Base-1R (from base to cylindrical housing), 1R-1P (from cylindrical housing to motor shaft), 1P-1R (from motor shaft to cylindrical housing) and Base-1R (from base to bearing housing). Note that, the linkage term is used for connection between two joints. Accordingly, the degree-of-freedom can be calculated as given in Equation 4.1, where η is the degree-of-freedom of the space, ℓ is the number of linkages, j is the number of joints and f_i is the freedom of each joint, $i = 1, 2, ..., j$.

$$\begin{aligned} DoF &= \eta \times (\ell - j - 1) + \sum_{i=1}^{j} f_i \\ &= 3 \times (4 - 4 - 1) + (3 \times 1R) + (1 \times 1P) \\ &= 1 \end{aligned} \tag{4.1}$$

4.2 Parallel Layout Equipped with Spherical Joint

At this layout, the spherical joint, which is used to connect the top platform to the vertical support column, allows roll-yaw-pitch movements of the top plate at three axes. As shown in Figure 4.2, the links, which consist of a linear motor, two spherical joints and other auxiliary structural elements, are positioned in parallel form. The technical specifications about spherical joints, which are used to connect the linear motor to the base and top platforms are supplied in Appendix B.2. At this configuration, there are seven spherical joints (3R) and three cylindrical joints (2PR). Also, there are eight linkages which connect the 3R-2PR (from cylindrical housing to motor shaft), 2PR-3R (from motor shaft to cylindrical housing) at each link, as well as Base-3R (from base to cylindrical housing) and Base-3R (from base to bearing housing). Hence, the degree-of-freedom can be calculated as given in Equation 4.2, where η is the degree-of-freedom of the space, ℓ is the number of linkages, j is the number of joints and f_i is the freedom of each joint, $i = 1, 2, ..., j$.

$$\begin{aligned} DoF &= \eta \times (\ell - j - 1) + \sum_{i=1}^{j} f_i \\ &= 6 \times (8 - 10 - 1) + (7 \times 3R) + (3 \times 2PR) \\ &= 9 \end{aligned} \tag{4.2}$$

The whole mechanism involves excessive degrees of freedom as calculated in the above equation, because of the spherical degrees of freedom at each end of the actuator. In other words, the redundant six degrees-of-freedom are originated from the twist of the motor body and the

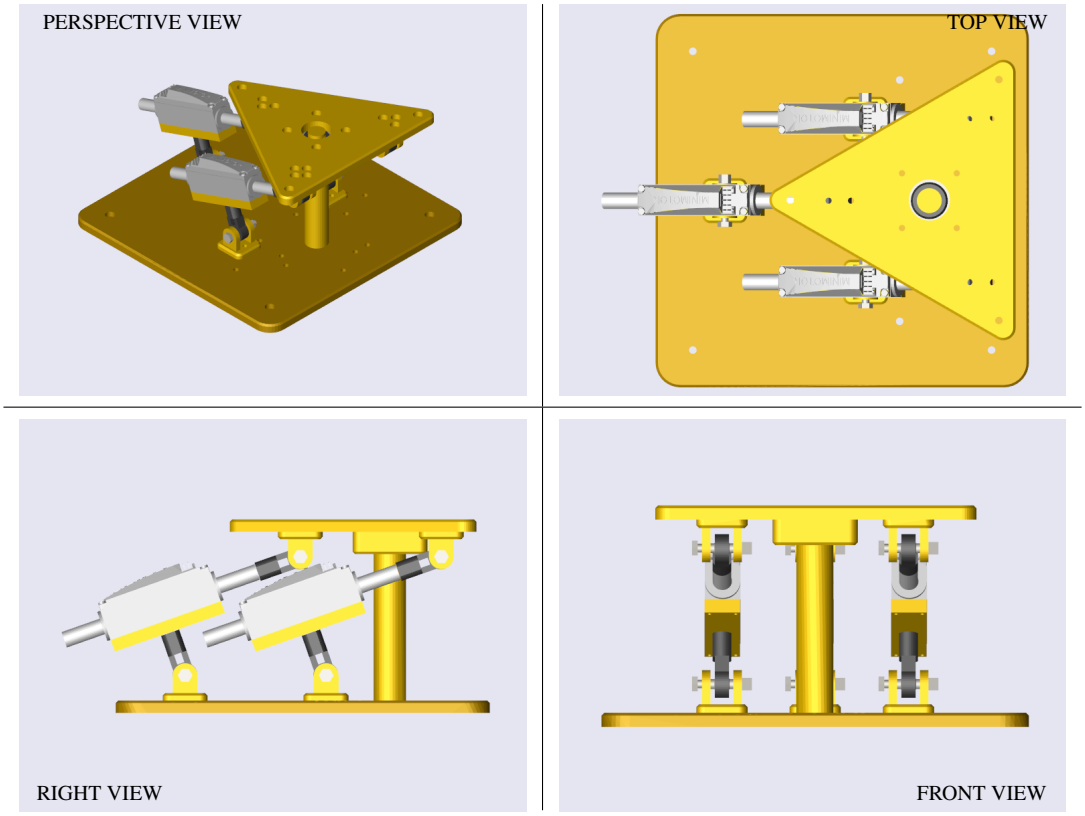


Figure 4.2: SimMechanics model of the parallel positioned links with spherical joints

motor shaft about the centerline axis of itself. However, redundant degrees-of-freedom are not constrained, because it does not affect the attitude of the top platform, since the effective length between the base and top platform connection points does not change with the free twist of the actuators. At this configuration, three axes rotational motion of the top platform is generated by simultaneous displacements of the motor shafts as summarized in Table 4.2. In this table, the shaft displacements are given for pure rotational motion of the top platform on single axis. However, the spatial rotational motion of it can be easily obtained by the combination of these shaft movements.

Table4.2: The relationship between rotational motion of the top platform and linear motion of the motor shafts for parallel layout

Top Platform	**Fore-Left Motor**	**Fore-Right Motor**	**Aft Motor**
Pure Roll to Left	Shorten	Lengthen	Rest
Pure Roll to Right	Lengthen	Shorten	Rest
Pure Yaw to Left	Shorten	Lengthen	Lengthen
Pure Yaw to Right	Lengthen	Shorten	Lengthen
Pure Pitch Down	Shorten	Shorten	Lengthen
Pure Pitch Up	Lengthen	Lengthen	Shorten

4.3 Symmetric Layout Equipped with Spherical Joint

At this layout, the same structural components are used with the previous configuration. However, the connection points of the cylindrical housings are changed and consequently, the links are positioned in more compact and intricate way as shown in Figure 4.3.

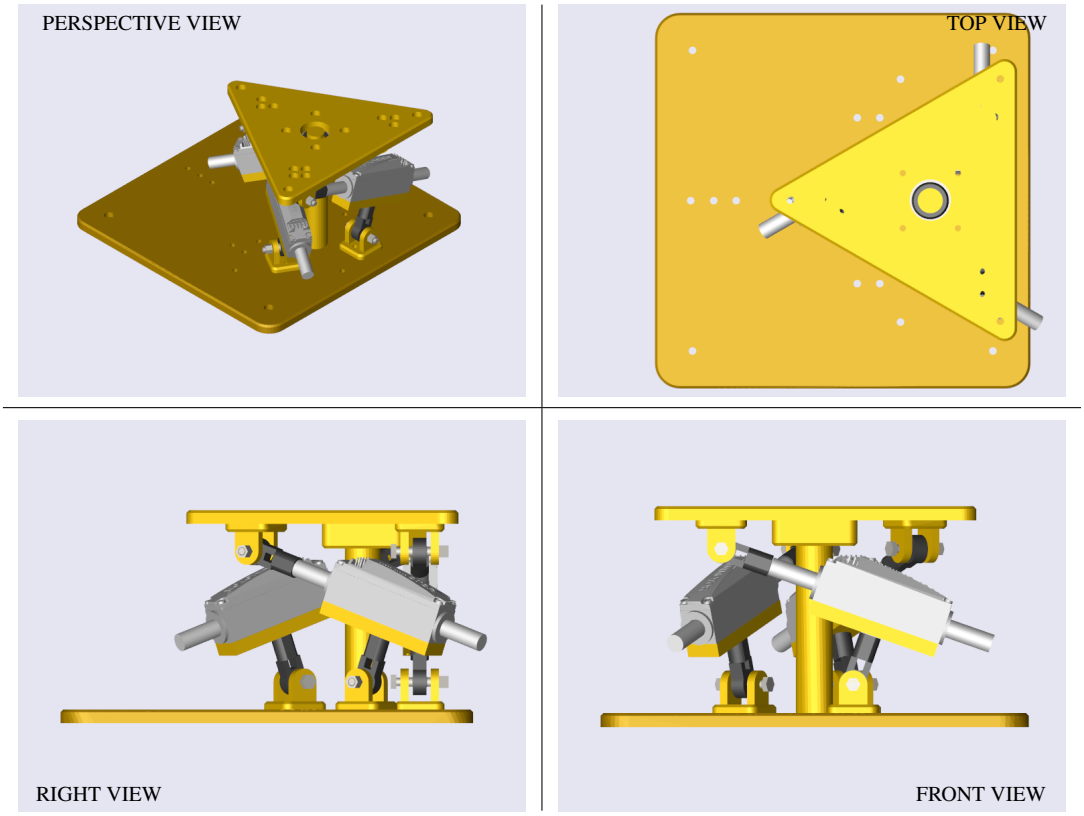


Figure 4.3: SimMechanics model of the symmetric positioned links with spherical joints

Table4.3: The relationship between rotational motion of the top platform and linear motion of the motor shafts for symmetric layout

Top Platform	**Fore-Left Motor**	**Fore-Right Motor**	**Aft Motor**
Pure Roll to Left	Shorten	Rest	Lengthen
Pure Roll to Right	Lengthen	Rest	Shorten
Pure Yaw to Left	Shorten	Shorten	Shorten
Pure Yaw to Right	Lengthen	Lengthen	Lengthen
Pure Pitch Down	Shorten	Lengthen	Shorten
Pure Pitch Up	Lengthen	Shorten	Lengthen

Due to the symmetric conformation, the linear motors can be programmed more easily as summarized in Table 4.3. Also, the kinematic stability of this layout is better than the parallel configuration. For instance, a nonzero actuator force is needed to hold the parallel layout at

its home position, however, the symmetric layout stands in balance by itself because of its geometric structure.

4.4 Parallel Layout Equipped with Universal Joint

At this configuration, the miniature universal joints are used instead of the spherical rod-end joints to connect the linear motors to the base and top platforms as shown in Figure 4.4. Therefore, the cylindrical housing parts, which are the holders for spherical rod-end joints, are omitted that leads to simpler and reliable design. The detailed technical information about miniature universal joints can be found in Appendix B.3.

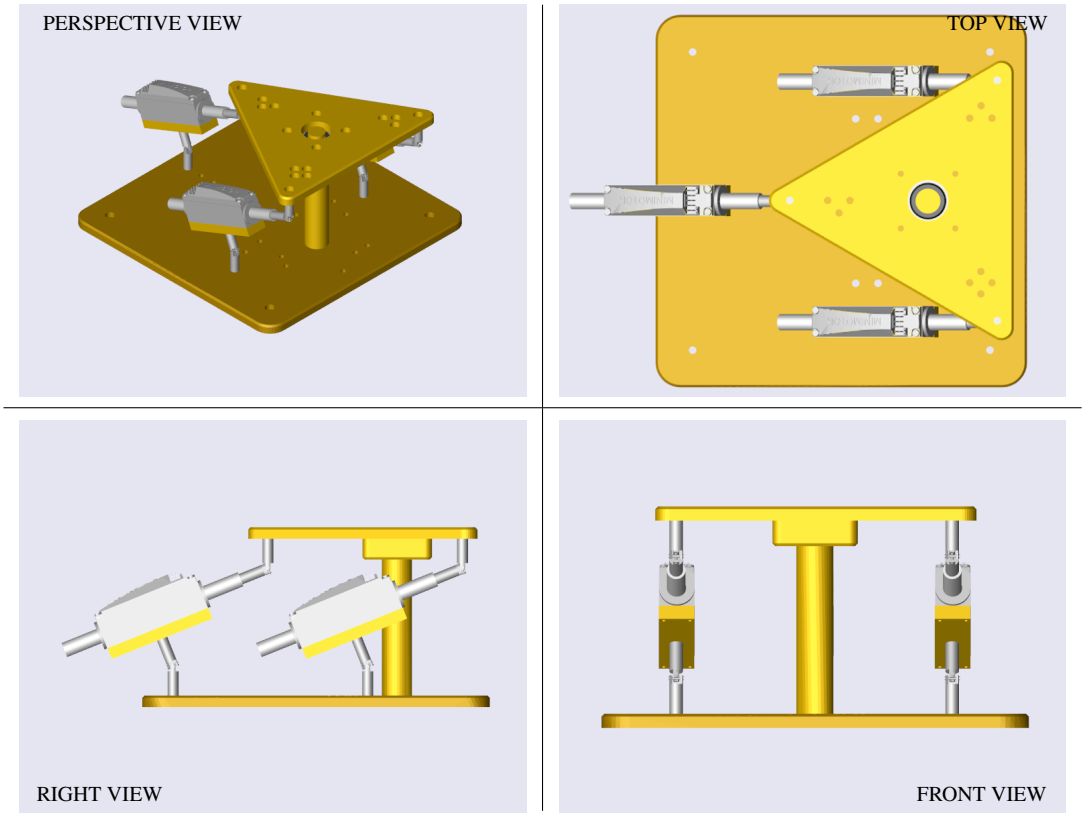


Figure 4.4: SimMechanics model of the parallel positioned links with universal joints

In this case, there are six universal joints (2R), one spherical joint (3R) and three cylindrical joints (2PR). Also, there are eight linkages which connect the 2R-2PR (from universal joint to motor shaft), 2PR-2R (from motor shaft to universal joint) at each link, as well as Base-2R (from base to universal joint) and Base-3R (from base to bearing housing). Hence, the degree-of-freedom can be calculated as given in Equation 4.3, where η is the degree-of-freedom of the space, ℓ is the number of linkages, j is the number of joints and f_i is the freedom of each joint, $i = 1, 2, ..., j$. By using universal joint, the excessive degrees of freedom are eliminated

and the kinematic stability is also satisfied.

$$\begin{aligned} DoF &= \eta \times (\ell - j - 1) + \sum_{i=1}^{j} f_i \\ &= 6 \times (8 - 10 - 1) + (6 \times 2R) + (1 \times 3R) + (3 \times 2PR) \\ &= 3 \end{aligned} \tag{4.3}$$

4.5 Symmetric Layout Equipped with Universal Joint

At this layout, the same structural components are used with the previous configuration. However, the connection points of the universal joints are changed and consequently, the links are positioned in more compact and intricate way as shown in Figure 4.5.

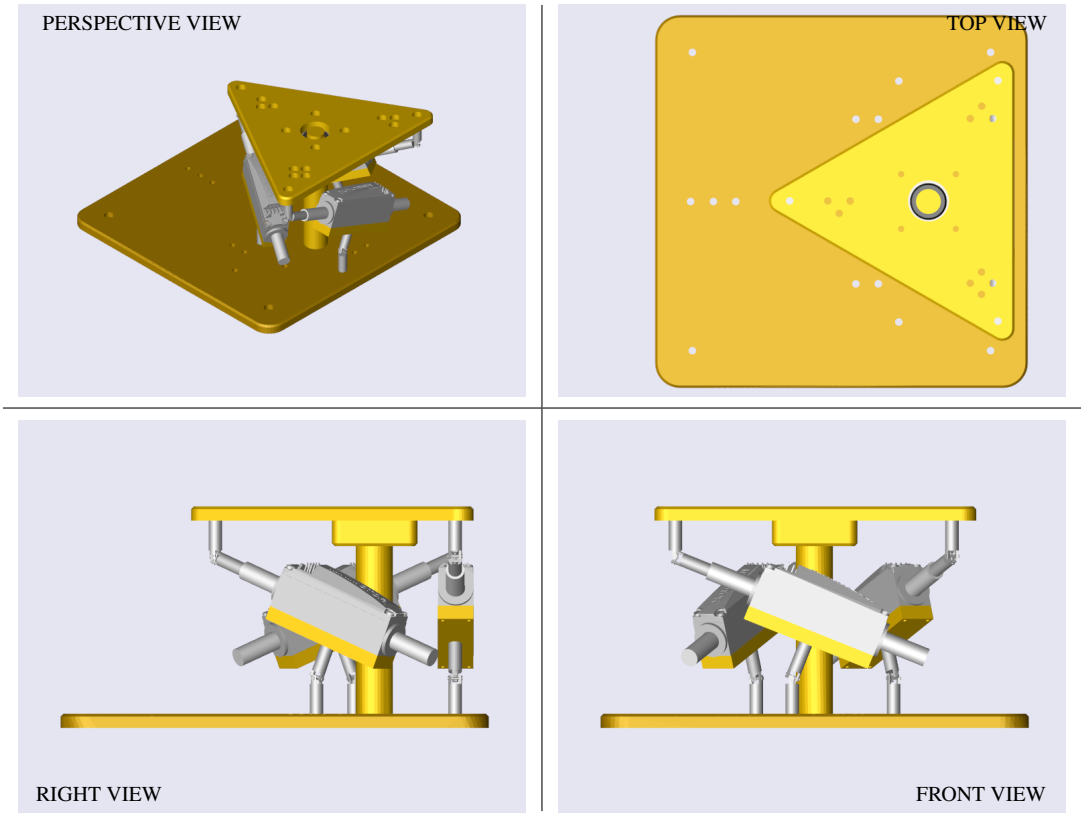


Figure 4.5: SimMechanics model of the symmetric positioned links with universal joints

CHAPTER 5

MODELING EFFORTS OF VISION SYSTEM

In the next chapter, various controllers are designed and implemented for the head structure to track a desired reference signal or to stabilize the top platform under various disturbances. In either case, it is necessary to define some criteria or a metric to evaluate the performance of these controllers and to make some comparisons between them.

For this purpose, a system error may be defined between reference input and the measured output. Suppose that, the system error is calculated between reference and measured shaft positions. Also consider that, an experiment shows that calculated position error remains between $[-1, +1]$ [mm]. Is it sufficient or not? Another performance criterion can be defined between desired and measured angular velocity of camera platform. Now assume that, the angular velocity deviates from zero between $[-10, +10]$ [deg/s]. Does the controller work well enough?

Our main purpose is to develop a stabilization platform for eliminating the disturbances which cause to the unwanted motion blur and affect the image quality adversely. However, the system errors, that are suggested above, may not give an idea about imposed motion blur. Therefore, it is more convenient that using the camera data and the amount of blur to evaluate the performance of the designed controllers. Actually, to degrade an image, in other words to add blur to a sharp image, is a matter of calculating the suitable point-spread function (PSF) and applying it to the image. However, to compute the motion blur from a degraded image is a little problematic, due to the lack of ground truth image (sharp image).

Therefore, a virtual camera model is implemented on MATLAB environment in order to define a performance metric related with motion blur. This simulation model calculates the motion blur without needing to use a ground truth or blur imposed images. Instead, it takes into account how much camera movement occurs during a given expose time as detailed in the subsequent sections.

5.1 Motion Blur

Motion blur is the result of the relative motion between the camera and the scene over the period of exposure determined by the shutter speed. Relative motion could cause either the camera or scene object movements as shown in Figure 5.1, however, in general both of them move.

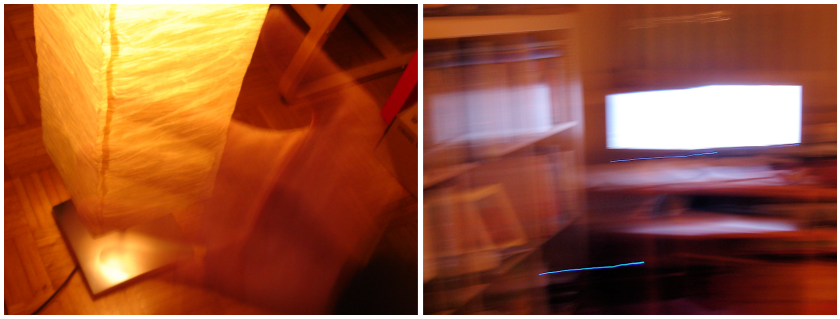


Figure 5.1: Motion blur caused by a) object movement b) rotational camera movement

The projection of any point in the scene moving with respect to the camera is a certain curve on the sensor rather than a point. Thus it will look blurred or smeared along the direction of relative motion as shown in Figure 5.2. The amount of motion blur increases as either the exposure or the relative motion between the camera and the scene increases, and it becomes more apparent at higher resolutions.

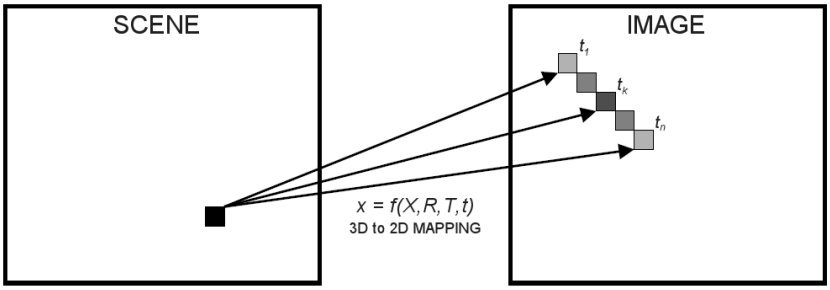


Figure 5.2: 3D to 2D one-to-multiple mapping due to the non-infinitesimal integration time

Motion blur has a wide range of application area. For example, it is used to obtain structure of a scene and motion information. In that field, a velocity vector is assigned to each image points to obtain a 3D velocity map. Moreover, it looks natural in a film or television image because the human eye behaves in much the same way. This is why a video game without motion blur effect will seem staggered, while natural motion filmed at the same frame rate appears rather more continuous. It has also been used in computer graphics to create more realistic images because appearance of motion blur is a strong perceptual cue for the illusion of object motion. Also, it can be used for aesthetic purposes such as emphasizing the dynamic

nature of a scene.

Despite its usefulness to human viewers, motion blur is an extensive image distortion and undesired effect in a large number of applications. In photography, a perfectly fixed camera and a static scene is necessary to capture the sharpest, most detailed photo so motion blur is considered to be an effect that can significantly degrade image quality. Figure 5.3 demonstrates simulated examples of images that are blurred by simple horizontal and vertical motions. [52] To measure the amount of blur imposed, there are some quality assessment techniques, which



Figure 5.3: a) Original image. Simulated motion blur effect by imposing b) circular motion c) horizontal motion d) vertical motion [52]

can be divided into three main categories; full-reference, reduced-reference and no-reference [11]. In full-reference techniques, both the original undegraded image and the distorted images must be available. In reduced-reference methods, only a few features of the original image are necessary. However, in most cases, especially in robotics applications, the original image is not obtainable. So, these techniques are generally used in applications of photogrammetry. On the other hand, in no-reference techniques [44, 65, 45, 17], neither the reference image nor any of its features is needed.

5.2 Mathematical Model of Imaging System

5.2.1 Pin-Hole Geometry

The pinhole camera model describes the mathematical relationship between the coordinates of a 3D point and its projection onto the 2D image plane. In this model, the camera aperture is described as a infinitesimal hole or a point. So, only one ray of light passes through this hole and project onto the single point, which leads to the one-to-one correspondence assumption. However, in the non-ideal case, the pinhole has a finite size, thus, each image point collects

light from a cone of rays. Moreover, this model does not include geometric distortions or blurring of unfocused objects caused by lenses. It also does not take into account that most practical cameras have only discrete image coordinates. Consequently, the pinhole camera model can only be used as a first order approximation of the mapping from a 3D scene to a 2D image. However, this model can be used to derive fundamental properties in a mathematical sense that describe the imaging process. Furthermore, these properties can be extended by using more realistic relationships to represent the real imaging devices. [66]

Despite its simplicity, the pinhole perspective projection model often provides an acceptable approximation of the imaging process. Let consider a coordinate system (O, i, j, k) attached to a pinhole camera as shown in Figure 5.4.

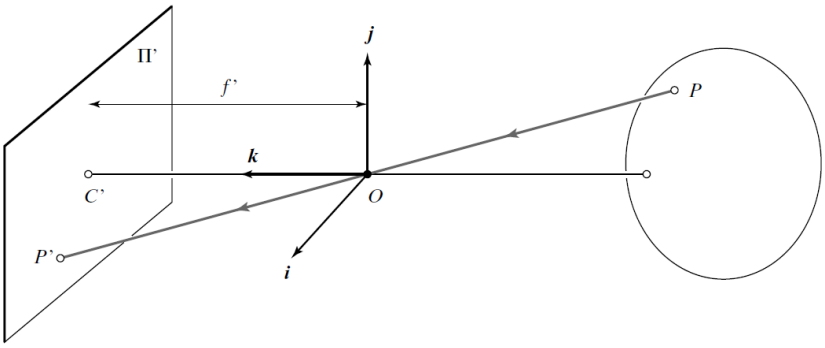


Figure 5.4: Perspective projection [18]

The camera center O, also denoted as optical center or center of projection, is the origin of a 3D coordinate system with the axis i, j and k. This 3D coordinate system is denoted as camera reference frame or simply camera frame. The image plane $\sqcap'$ is defined to be parallel to the $i - j$ plane. The point C' where the k axis intersects the image plane is referred to as image center. The the line through O and C' is denoted as optical axis. Let P denote a scene point with coordinates (x, y, z) and P' denote its image with coordinates (x', y', z'). Since P' lies in the image plane, we have $z' = f'$. Since the three points P, O, and P' are collinear, we have OP' = λOP for some number λ, where f' is the focal length of the camera. [18]

$$\begin{array}{l} x' = \lambda x \\ y' = \lambda y \\ f' = \lambda z \end{array} \quad \Rightarrow \quad \lambda = \frac{x'}{x} = \frac{y'}{y} = \frac{f'}{z} \quad \Rightarrow \quad \begin{array}{l} x' = f'\frac{x}{z} \\ y' = f'\frac{y}{z} \end{array} \tag{5.1}$$

The pinhole perspective projection can be further simplified by introducing some assumptions. These simpler models can be grouped under affine projection, namely, weak-perspective, orthographic and paraperspective projections [18].

5.2.2 Intrinsic & Extrinsic Parameters

Until now, all relationships between 3D scene points and 2D image points have been defined with respect to a common reference frame, i.e., camera frame. Usually, the location of a point in the world is not known in camera coordinates. Thus, if it is desired to relate the world coordinates to the image coordinates, or vice versa, it has to be considered the physical parameters of the camera. The intrinsic parameters describe the projection of a point in the camera frame onto the image plane, i.e., the transformation of camera coordinates into image coordinates. This transformation extends the ideal perspective camera model that is introduced in the previous section, to the real imaging devices such as CCD cameras. The projection matrix M_i can be derived as follows;

$$M_i = \begin{bmatrix} -f/s_x & k & o_x \\ 0 & -f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \tag{5.2}$$

where, f represents the focal length, s_x and s_y are the effective pixel size in x and y directions, respectively, k is the skew coefficient, and (o_x, o_y) are the coordinates of the image center. The skew coefficient k determines the angle between the pixel axis and is usually zero, i.e., the $x-$ and $y-$ axes are perpendicular. (o_x, o_y) can be seen as an offset that translates the projection of the camera origin onto the image origin in pixel dimensions. The aspect ratio of the camera is defined as; s_x/s_y. If $s_x/s_y = 1$, then, the sensors of the CCD array are ideally square. If $s_x = s_y = 1$ and $o_x = o_y = k = 0$, then, M_i represents an ideal pinhole perspective camera.

The extrinsic parameters take the transformation between a fixed world coordinate system and the camera coordinate system into account. This includes the translation and rotation of the coordinate axis as follows;

$$M_e = \begin{bmatrix} r_{11} & r_{12} & r_{13} & -R_1^T T \\ r_{21} & r_{22} & r_{23} & -R_2^T T \\ r_{31} & r_{32} & r_{33} & -R_3^T T \end{bmatrix} \tag{5.3}$$

where, $\mathbf{T} = (Tx, Ty, Tz)^T$ is a $[3 \times 1]$ translation vector and $\mathbf{R}$ is a $[3 \times 3]$ rotation matrix. Also, r_{ij} $(i, j \in 1, 2, 3)$ are the matrix elements of $\mathbf{R}$ at i^{th} row and j^{th} column, as well as, R_i indicates the i^{th} row of the $\mathbf{R}$. Thus, the relationship between world and image coordinates can be written in terms of two matrix multiplications as follows;

$$\begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} = M_i M_e \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad \Rightarrow \quad \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} p_1/p_3 \\ p_2/p_3 \end{pmatrix} \tag{5.4}$$

where, $(X, Y, Z, 1)^T$ represents a 3D world point in homogeneous coordinates, and $(p1, p2, p3)^T$ indicates camera coordinates. $\mathbf{M} = \mathbf{M_i M_e}$ is denoted as projection matrix. The desired projected points (x, y) can be computed in image coordinates as given in Equation 5.4. [18]

5.2.3 Lens Imperfectness: Distortion

Due to the manufacturing defect and structure of the lenses, the linear projection can not be guaranteed always. So, the resulting image coordinates may be distorted as shown in Figure 5.5. If high accuracy and precision is required, the simple mathematical relationships

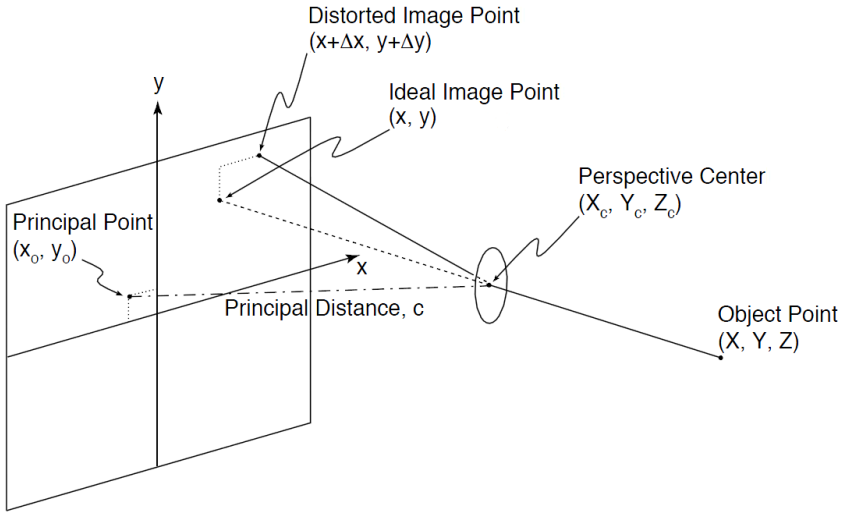


Figure 5.5: Projected (ideal) and distorted image points [16]

introduced before are not sufficient. To take into consideration this effect, a distortion model has to be defined. A common radial distortion model, as given in [24], can be written as;

$$\begin{pmatrix} x_d \\ y_d \end{pmatrix} = L(r) \begin{pmatrix} x \\ y \end{pmatrix}, \quad r = \sqrt{x^2 + y^2} \tag{5.5}$$

where, $(x, y)^T$ is the undistorted and $(x_d, y_d)^T$ the corresponding distorted image positions. The function $L(r)$ determines the amount of distortion depending on the radial distance from the center of the radial distortion. An arbitrary distortion factor $L(r)$ can be approximated as follows, which is defined for $r > 0$ and $L(0) = 1$.

$$L(r) = 1 + \sum_{i}^{m} \kappa_i r^i \tag{5.6}$$

The center of radial distortion, as well as the distortion coefficients, κ_i, can be seen as additional intrinsic parameters of the camera model. The number of coefficients, m, depends on the required accuracy and the available computation time. Usually less than the first three or four coefficients are considered. In common calibration procedures only the even coefficients are taken into account while odd coefficients are set to zero. In this case, one or two coefficients are sufficient to compensate for the distortion in most cases. Beside the radial distortion model, there are several other models including tangential, linear, and thin prism distortion. Usually a radial distortion model is combined with a tangential model as proposed in [25].

In this thesis, the distortion model, given in Equation 5.7, is used, where, κ_1, κ_2 are the radial and κ_3, κ_4 are the tangential distortion coefficients.

$$\begin{aligned} x_d &= \left(1 + \kappa_1 r^2 + \kappa_2 r^4\right) x + 2\kappa_3 xy + \kappa_4 \left(r^2 + 2x^2\right) \\ y_d &= \left(1 + \kappa_1 r^2 + \kappa_2 r^4\right) y + \kappa_3 \left(r^2 + 2y^2\right) + 2\kappa_4 xy \end{aligned} \tag{5.7}$$

5.2.4 Realistic Parameters: Calibration Results

There are several approaches to compute the unknown intrinsic and extrinsic parameters of a camera. The most common methods are based on known correspondences between real world points and image coordinates. A chessboard-like calibration grid has become quite common as a calibration pattern.

In a captured image of the calibration pattern, the corners can be extracted at pixel (or sub-pixel) level, and mapped to world coordinates. If there is a sufficient number of correspondences, one can try to solve a homogeneous linear system of equations based on the projection matrix, $\mathbf{M}$. There are linear and nonlinear methods to solve for the projection matrix. Linear methods assume an ideal pinhole camera and ignore distortion effects. Thus, these methods can be solved in closed-form as is direct linear transform (DLT) to compute the parameters in a non-iterative algorithm. If higher accuracy is needed, nonlinear optimization techniques must be investigated to accomplish for distortion models. Usually, the parameters are estimated by minimizing the pixel error between a measured point correspondence and the reprojected position of the world point using the projection matrix in least-square sense. This is an iterative process that may end up with a bad solution unless a good initial guess is available. Therefore, linear and nonlinear methods are combined as the DLT can be used for initialization of the nonlinear optimization.

To obtain intrinsic and extrinsic parameters for various types of camera models, the Caltech Camera Calibration Toolbox [8] is used. The calibration images taken by Point Grey Flea 2 CCD camera and reproduced calibration grids are shown in Figure 5.6 for chessboard-type grid pattern.

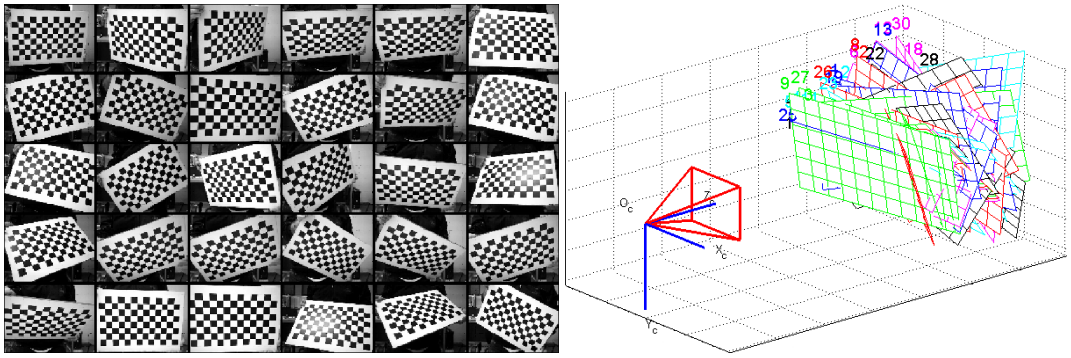


Figure 5.6: a) Calibration images taken by Point Grey Flea 2 CCD camera for chessboard-type grid pattern b) Reproduced calibration grids [8]

By using this toolbox, three different camera models are subjected to the calibration procedure. The obtained intrinsic parameters are summarized in Table 5.1.

Table5.1: The computed intrinsic parameters by using Caltech Camera Calibration Toolbox for various types of camera models

Intrinsic Parameter	**Point Grey Flea2**	**Logitech C310**	**Sony Cybershot W580**
Focal Length [pix]	466.39, 465.69	886.05, 881.72	1160.45, 1155.78
Principle Point [pix]	306.31, 235.51	425.55, 289.92	807.68, 600.49
Skew Coefficient	0	0	0
Radial Distortion	-0.3513, 0.1250	0.0145, -0.0926	-0.0056, 0.0092
Tangential Distortion	0.0013, 0.0023	-0.0012, 0.0044	-0.0020, 0.0040

5.3 Simulation Model of Imaging System: Virtual Camera

5.3.1 Design of 3D Scene

First of all, a 3D scene is created which includes a static object and a mobile camera as shown in Figure 5.7. The object, embodied by a rectangular prism, is colored in RGB/XYZ standard, i.e., red-green-blue colors indicate x-, y-, z-axis, respectively. The black colored camera moves during the simulation to indicate the position and orientation of the imaging device. Also, the optical axis of the camera is indicated by a dashed yellow line. The camera

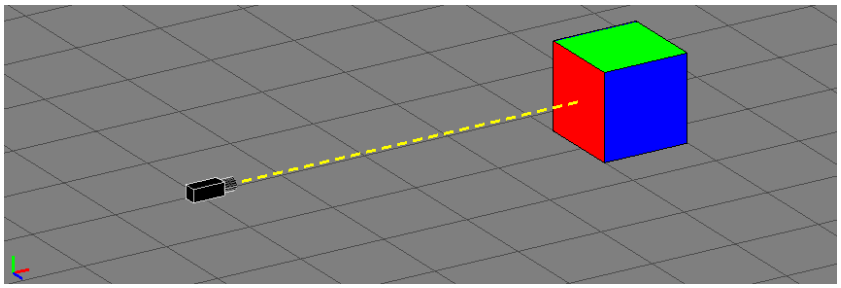


Figure 5.7: Designed 3D scene includes a static object and mobile camera

can be driven at six-axes independently by generated constant, linear, sinusoidal, sawtooth signals or IMU measurements. Also, it is possible to add white, random and laplacian noise to the driving signals.

5.3.2 Implementation of Camera Model

The transformation between 3D scene points and 2D image plane points are carried out by using pin-hole camera model with perspective projection scheme as detailed in section 5.2.1 and section 5.2.2. Also, a lens distortion model is added to incorporate the effects of lens imperfectness as explained in section 5.2.3. Moreover, three different parameter sets obtained by calibration process as indicated in section 5.2.4, are available to examine the effects of different properties of imaging system.

5.3.3 Calculation of Motion Blur

At the next step, the eight corner points of the object are projected to the image plane according to the implemented camera model and camera parameters. So, at each instant of the simulation, in other words for each camera pose, we have eight projected 2D points in pixel coordinates.

$$\begin{pmatrix} x_i & y_i \end{pmatrix}^T = F\begin{pmatrix} X_i & Y_i & Z_i \end{pmatrix}^T, \quad where \quad i \in \{1, 2, ...8\} \tag{5.8}$$

The centroid of these projected points is calculated by taking averages of them at each axis. In this way, scene points are reduced to single point which moves on the image plane according to the camera pose. Note that, due to the distortion effect, the model is no longer linear. So, first projecting the scene points and then calculating centroid on the image plane is different from first calculating the centroid of scene points and then projecting this single point to the image plane.

$$\bar{x} = \frac{1}{m}\sum_{i}^{m} x_i \quad and \quad \bar{y} = \frac{1}{m}\sum_{i}^{m} y_i \quad where \quad m = 8 \tag{5.9}$$

The velocity of this virtual point is computed by simply taking difference.

$$v_x = \frac{\bar{x}_{k+1} - \bar{x}_k}{\Delta k} \quad and \quad v_y = \frac{\bar{y}_{k+1} - \bar{y}_k}{\Delta k} \tag{5.10}$$

The magnitude of the velocity vector of the centroid point can be calculated in [pixel/s].

$$v = \sqrt{v_x^2 + v_y^2} \tag{5.11}$$

Suppose that, sketch of the frame and expose instants, as well as the magnitude of the centroid velocity are given as in Figure 5.8. The first sketch, indicated by letter f, shows the duration of frame and exposure instants by red and blue lines, respectively. The second sketch, indicated by letter v, shows the magnitude of the centroid velocity and sampled points in [pixel/s]. Once the magnitude of the centroid velocity is obtained, then the motion blur is simply calculated by multiplying centroid velocity in [pixel/s] and expose time in [s]. Thereby, the value of the motion blur can be calculated in [pixel], in computationally efficient way, without finding a point-spread function or applying inverse filters. According to the frame-per-second and shutter speed (1/expose time) parameters of the imaging system, and sampling frequency of the velocity data, there may be a necessity to take the average of the multiple velocity samples

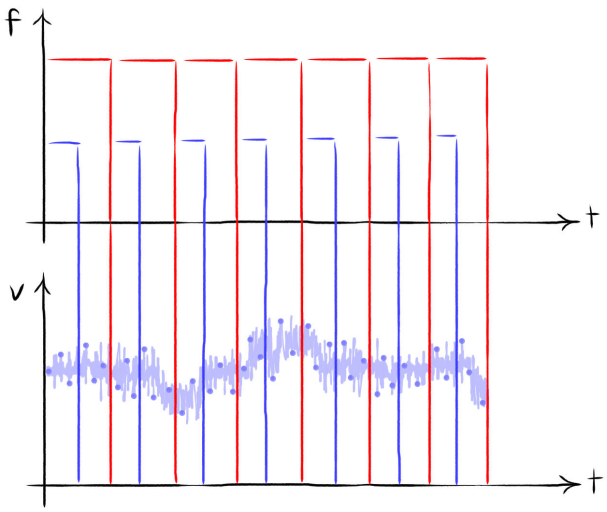


Figure 5.8: a) The duration of frame and exposure instants are shown by red and blue lines. b) The magnitude of the centroid velocity and sampled points are given in [pixel/s].

or interpolate between closest two samples to get a single velocity value within that interval. As noted that, by using this approach, motion blur can be calculated by using imaging system specific parameters without needing to the sharp and/or motion blur imposed images.

5.3.4 Formation of Ground Truth & Motion Blur Imposed Images

For only illustrative purposes, ground truth and motion blur imposed images can be synthetically generated. According to the position and orientation of the camera and scene object, some portion of the object may or may not have line-of-sight with the imaging system. To form the synthetic image correctly, the visibility of the faces have to be determined first. For this purpose, each vertex of the object and center of the camera are connected by a virtual line. Then, this line is checked for intersection with all of the object faces. If it does not intersect with any of them, this vertex has a line-of-sight with the imaging system and it is classified as a visible vertex. Conversely, if the virtual line and an object face have intersection, then this vertex can not be seen by the camera and it is classified as a non-visible vertex. Actually, it is not necessary to check all of the faces, because a vertex could not be obstructed by its own face. So for the rectangular prism, checking only three faces is sufficient to determine whether this vertex is visible or non-visible.

Once all of the vertices of the object are classified, then the visible faces can be determined by doing this observation; a face is visible, if and only if, all of its vertices are visible. If at least, one of the vertices is not visible, then this face could not be seen by the imaging system. Fortunately, this is true for all objects with convex structure. The scene layout and generated ground truth synthetic images are shown in Figure 5.9.

The motion blur imposed images can be generated by filtering the ground truth images with

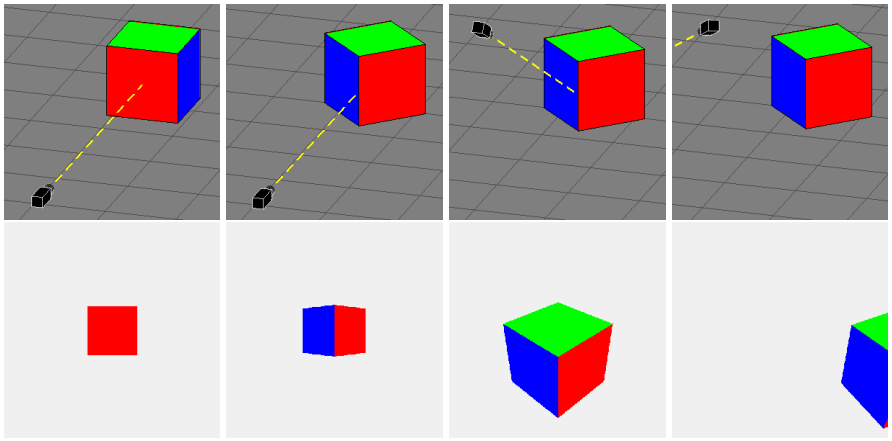


Figure 5.9: The scene layout and generated ground truth synthetic images

a suitable point-spread-function calculated from the motion information of the camera. However, this method is computationally inefficient. Instead we approximate to the calculation of the degraded image by using linear combination of the multiple ground truth images generated during expose interval, in other words, we simply average them as shown in Figure 5.10.

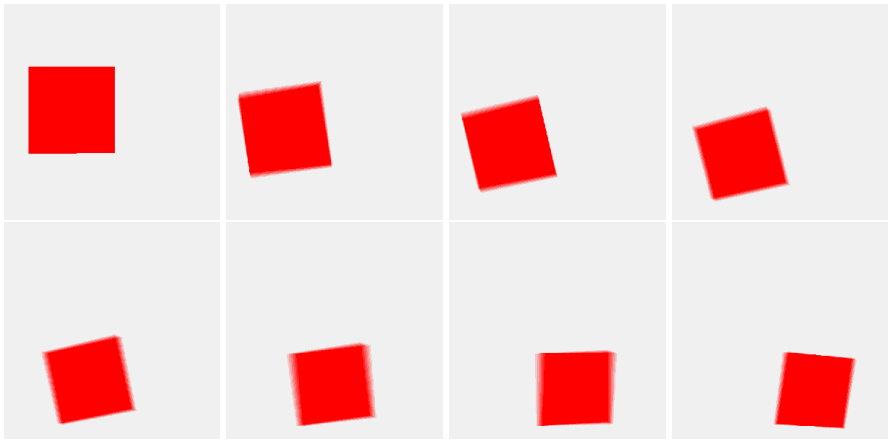


Figure 5.10: Motion blur imposed images that are calculated by using linear combination of the ground truth images generated within the same exposure time interval. Motion is from right to left, and from top to bottom

5.4 Simulation Results

At this section, the virtual camera model is validated by comparing with blur metrics borrowed from the related literature. Also, the effects of distance on motion blur under pure translation and pure rotational movements are investigated. Moreover, the effects of shutter speed and intrinsic camera parameters are examined. Note that, x-y-z axes of the camera co-

ordinate frame correspond to the roll-yaw-pitch axes, respectively. In other words, the depth of the scene coincidences with the x-axis of the camera coordinate frame. Also, vertical and horizontal axes of the image plane coincidence with the y- and z-axes of the camera coordinate frame, respectively. Unless otherwise specified, the calibration result of the Point Grey Flea2 is used for intrinsic camera parameters. Also, the dimension of the object is set to 2 x 2 x 2 [m] (width x height x depth).

5.4.1 Comparison of Motion Blur and Other Blur Metrics

For validation purposes, motion blur imposed images are generated by using virtual camera for arbitrary camera movements. Then, these synthetic images are used to calculate some blur metrics, which are already available as an MATLAB implementation. The corresponding blur metrics; perceptual blur [12] and anisotropic quality index [19] are compared with our motion blur calculations as shown in Figure 5.11. Actually, the main trends of the motion

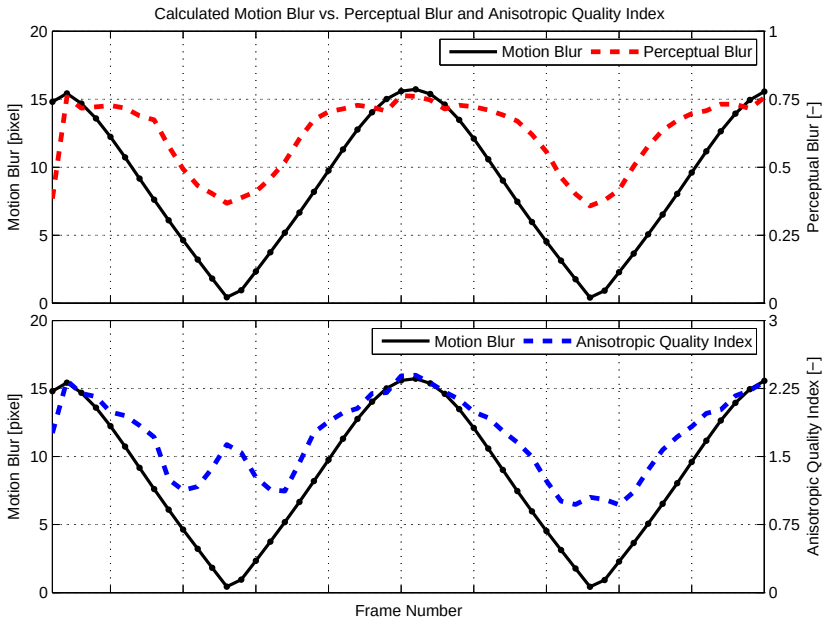


Figure 5.11: Comparison of perceptual blur [12] and anisotropic quality index [19] calculated by using synthetic images and motion blur calculations obtained by virtual camera model

blur and other blur metrics are convenient. Indeed, it is not expected to overlap exactly, but the difference between minimum values are higher than considered. At this simulation, the distance between camera and object is fixed and the results are observed under pure translation motion along y- and z-axes. The sinusoidal motion is applied with amplitude of 2 and 5 [m], as well as period of 1 and 2 [s], respectively. In briefly, due to the sinusoidal motion, it would not be strange to expect that blur metric has to take small values at somewhere the velocity is zero and the motion changes its direction. From this point, the motion blur values calculated by virtual camera model make more sense.

5.4.2 Effect of Distance under Pure Translational and Pure Rotational Motion

The first simulation is conducted to examine the effect of distance between the imaging system and the scene object, on amount of blur under pure translational motion. For this purpose, the angular position of the camera is fixed to its initial condition about x-y-z axes (roll-yaw-pitch). The camera is driven by a sinusoidal signal with amplitude 0.2 [m] and period 1 [s] along y- and z- axes, which leads to the maximum velocity of 1.22 [m/s]. The distance between the camera and object is varied from 10 [m] to 2[m]. Expectedly, for the same camera motion, the closer object cause higher motion blur as shown in Figure 5.12.a.

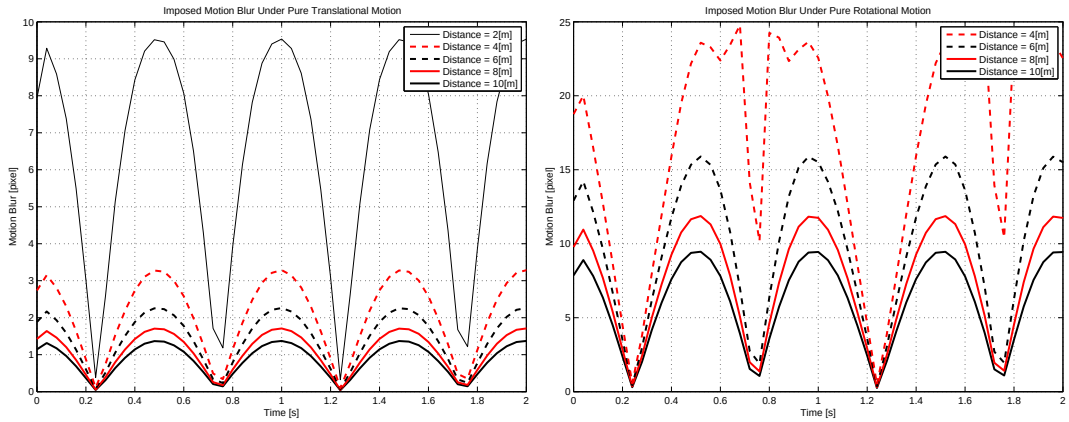


Figure 5.12: The variation of motion blur at various distance between stationary object and mobile virtual camera a) under pure translational motion b) under pure rotational motion

The second simulation is conducted to examine the effect of distance between the imaging system and the scene object, but this time under pure rotational motion. For this purpose, the linear position of the camera is fixed to its initial condition along y-z axes (vertical-horizontal). It is driven by a sinusoidal signal with amplitude 10 [deg] and period 1 [s] about x-y-z-axes (roll-yaw-pitch), which leads to the maximum velocity of 60 [deg/s]. The distance between the camera and object is varied from 10 [m] to 4[m]. Again, as expectedly, for the same camera motion, the closer object cause higher motion blur values in [pixel] as shown in Figure 5.12.b.

5.4.3 Effect of Decoupled & Coupled Motions

In the third simulation, the camera is fixed along y-z axes and driven by a sawtooth signal along x-axis to become closer to and move away from the object by constant velocity. Meanwhile, it is driven by sinusoidal signals about x-y-z axes (roll-yaw-pitch) with amplitude 10 [deg] and period 1 [s], which leads to the maximum velocity of 60 [deg/s]. Actually, the camera becomes closer to the object and then moves away from it, while performing some angular rotations.

Then three kind of motion blur are calculated as shown in Figure 5.13. In (a), it is supposed that, the camera is fixed to its initial angular position and tracks only the linear motion signal. Similarly, in (b), it is supposed that, the camera is fixed to its initial linear position and tracks only the angular motion signal. Finally, in (c), the whole motion is taken into account and calculated the "actual" motion blur.

It is already known and shown that the closer objects lead to higher amount of motion blur. However, it is interesting to observe that, "actual" motion blur, which incorporates the whole motion signals defined on 6-axes, is not a simple summation of pure translational and pure rotational motion blurs, instead it is the modulation of them. Note that, we can obtain the actual motion blur signal by modulating the pure rotational motion blur with the envelope of the pure translational motion blur.

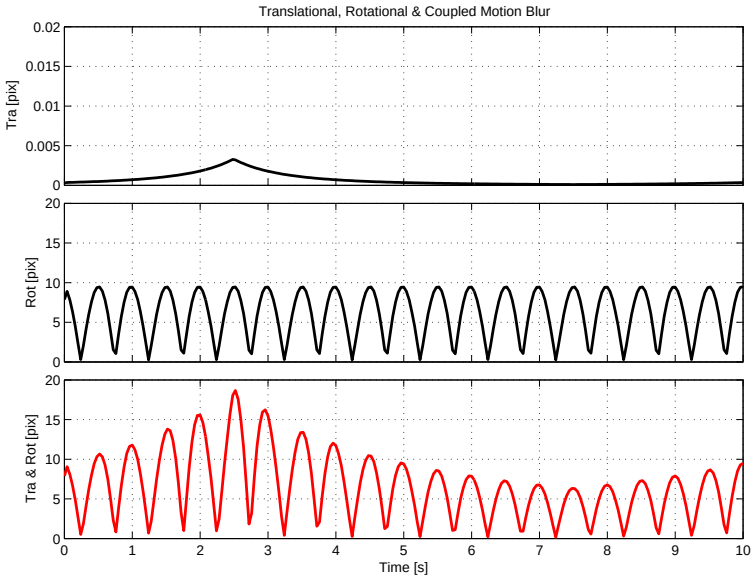


Figure 5.13: The amount of blur imposed by a) only translation motion, b) only rotational motion c) translational and rotational coupled motion

5.4.4 Effect of Shutter Speed

In the fourth simulation, the effect of shutter speed is examined. For this purpose, the camera is fixed along y-z axes and driven by a sawtooth signal along x-axis to become closer to and move away from the object by constant velocity. Meanwhile, it is driven by sinusoidal signals about x-y-z axes (roll-yaw-pitch) with amplitude 10 [deg] and period 1 [s].

Shutter speed is the effective length of time a camera's shutter is open. Note that, shutter speed values are given in [1/s]. It is a common term used to discuss the exposure time. Remember that, motion blur is caused by relative motion between camera and scene object during exposure time. So, it is expected that, for smaller shutter speed (or shorter expose

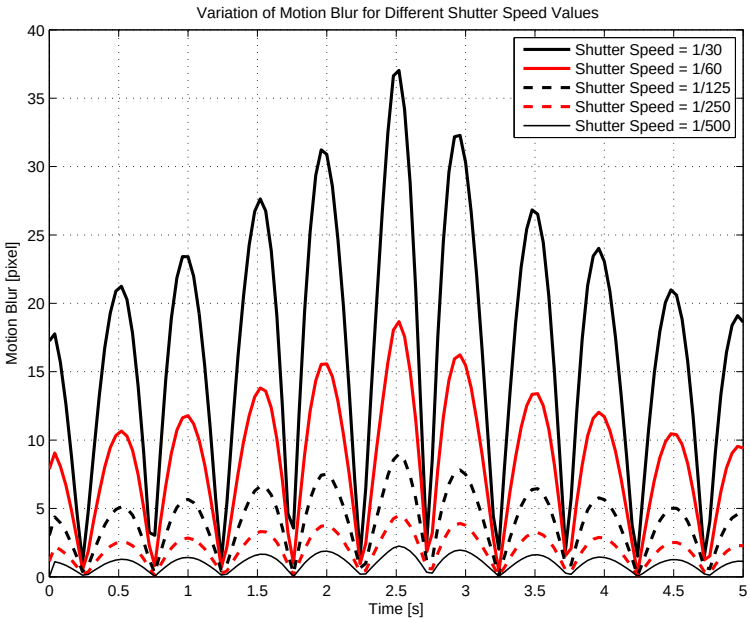


Figure 5.14: The variation of motion blur for different shutter speed values given in [1/s]

time), the same motion causes to the smaller amount of motion blur. From this point, the simulation results are consistent with our expectations as shown in Figure 5.14.

5.4.5 Effect of Intrinsic Camera Parameters

In the last simulation, the effect of camera model on amount of motion blur is examined. Remember that, various camera model parameters are already obtained in section 5.2.4 by using Caltech Camera Calibration Toolbox. These models and related intrinsic parameters are summarized in Table 5.2.

Table5.2: The computed intrinsic parameters by using Caltech Camera Calibration Toolbox for various types of camera models

Intrinsic Parameter	**Point Grey Flea2**	**Logitech C310**	**Sony Cybershot W580**
Focal Length [pix]	466.39, 465.69	886.05, 881.72	1160.45, 1155.78
Principle Point [pix]	306.31, 235.51	425.55, 289.92	807.68, 600.49
Skew Coefficient	0	0	0
Radial Distortion	-0.3513, 0.1250	0.0145, -0.0926	-0.0056, 0.0092
Tangential Distortion	0.0013, 0.0023	-0.0012, 0.0044	-0.0020, 0.0040

The same motion signal with the previous simulation is applied to the virtual camera model and the results are graphed in Figure 5.15. We already know that, motion blur becomes more apparent at higher resolutions, which is validated by the simulation results that is the case for Sony Cybershoot W580.

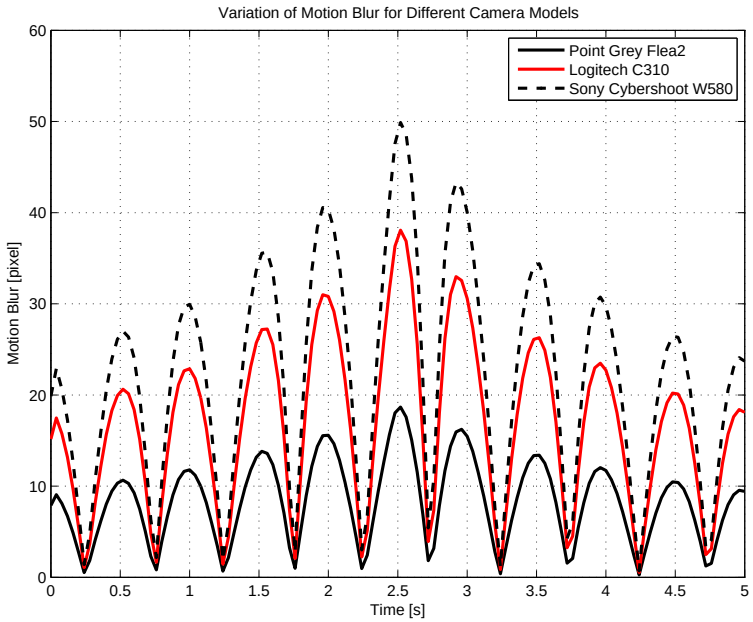


Figure 5.15: The variation of motion blur for different camera models under coupled movements of virtual camera along x-axis and about x-y-z axes

CHAPTER 6

CONTROL APPROACHES FOR CAMERA STABILIZATION PLATFORM

In this thesis, as schematically shown in Figure 6.1, the physical system of interest consists of a plant that moves the attached camera in 3-DOFs, a single-board computer that runs the real-time operating system to generate necessary control actions and other peripherals (serial port, data acquisition board) that allows data transfer. The motor driver, motor and the mechanical components are the main elements that make up the plant. Additionally, Hall effect sensor and inertial measurement unit (IMU) are used to calculate or to estimate the linear position of the motor shaft and the angular position of the top platform. Furthermore, the real-time operating system running on the single-board computer is used to execute the kinematics equations, as well as the position, state and orientation estimators. It also calculates a control action according to the preferred controller configuration which uses position reference as well as the position measurement or full-state feedback. The controller output that is algorithmically computed on the software environment, is sent to the plant through serial port. Similarly, the angular velocity and linear acceleration of the camera stabilization platform that is measured by attached IMU, are read by using serial communication. Also, the output of the Hall effect sensor, which generates an analog voltage depending on the position of the motor shaft, is measured by analog input of the data acquisition board connected to the single-board computer.

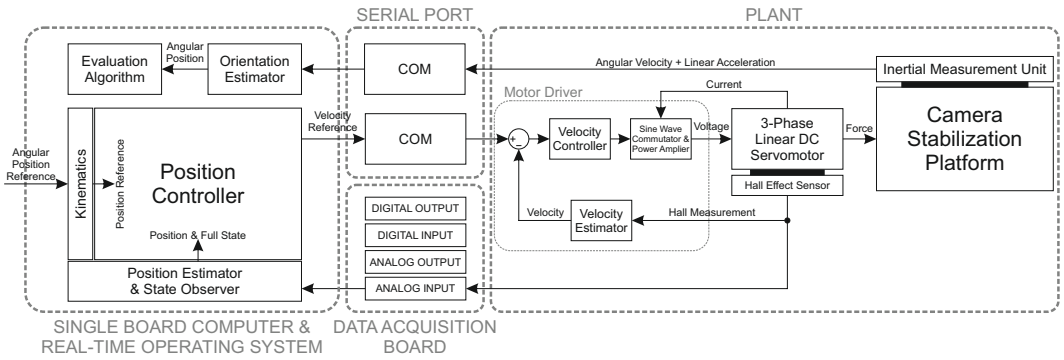


Figure 6.1: Physical system includes a plant, a single-board computer and other peripherals like serial port and data acquisition board

Thus far, in Section 3.3 and in Section 3.4, a transfer function model with identified static nonlinearity is obtained for motor driver and motor by using various system identification techniques. Also, in Chapter 4 a nonlinear model is obtained for mechanical components of the camera stabilization platform by using SimMechanics multi-body modeling tool. Additionally, actuator model and SimMechanics model are integrated to attain overall nonlinear system model and a linearized state-space model. In this chapter, various controller structures are implemented to examine the tracking performance for a given reference signal and stabilizing performance under disturbances of the camera stabilization mechanism. The implementations can be grouped under two main categories;

1. Model-Free Controllers
    - Proportional-Integral-Derivative Controller (PID)
    - Sliding Mode Controller (SMC)
    - Proxy-Based Sliding Mode Controller (PBSMC)
    - Fuzzy-Logic Controller (FLC)
    - Artificial Neural Network Controller (ANNC)
2. Model-Based Controllers
    - Linear Quadratic Tracker (LQT)
    - Model Predictive Controller (MPC)

Our main goal is to realize these controllers on the physical system, perform tracking / stabilizing experiments, and compare the results in terms of time domain specifications (i.e., rise time, maximum overshoot) and vision system degradation metrics (i.e., motion blur). However, there are many factors arising from the imperfectness and limitations of the physical systems that affect the performance of a controller. For example, the PID controllers use the error signal itself, it's integral and it's derivative to calculate the control signal, whereas the LQTs use state values only. Due to the derivative term, increasing amount of measurement noise more adversely affects the performance of the PID controller than LQT. On the other hand, some states may not always be measurable which makes necessary to use LQT with state observer. For whatever reason it might be, impossibleness or physical limitations, the increasing number of unmeasurable states more adversely affects the performance of the LQT than PID controller.

It is obvious that, the performance of a controller is closely related to the restrictions and limitations of a physical system. Therefore, to evaluate the performance of the controllers in a fair and equitable way, firstly, an idealized case is conceptualized by ignoring all imperfectness, limitations and restrictions as itemized below and as shown in Figure 6.2.

- Case-A: Idealized Configuration
    - Ideal force source, instead of an driver+actuator model

– Perfectly executable control action (no serial communication without any delay)

– Measurable states without any measurement noise

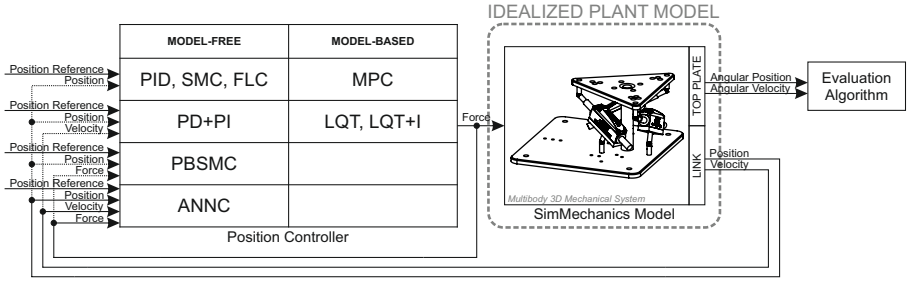


Figure 6.2: Idealized plant model and designed controller structures

At Case-A, it is assumed that, the output of the position controller is calculated in terms of force unit and can be applied to the plant without any quantization process and delay. Also, it is assumed that the resulting link position and link velocity can be measured perfectly without any noise. Undoubtedly, the results obtained for this idealized configuration show the best possible controller performances and a suitable test bed for comparison purposes. However, the designed controllers for this conceptualized case can not be applied to the physical system. Due to the existence of motor and motor driver, the input of the real plant is at the unit of velocity instead of force. So, selected controllers are modified according to the requirements of physical system and a realistic configuration is implemented as shown in Figure 6.3.

- Case-B: Realistic Configuration

  – Incorporated actuator model

  – Operative velocity controller on motor driver

  – Serial communication at very low sampling frequency with delay

  – Measurable shaft position with measurement noise

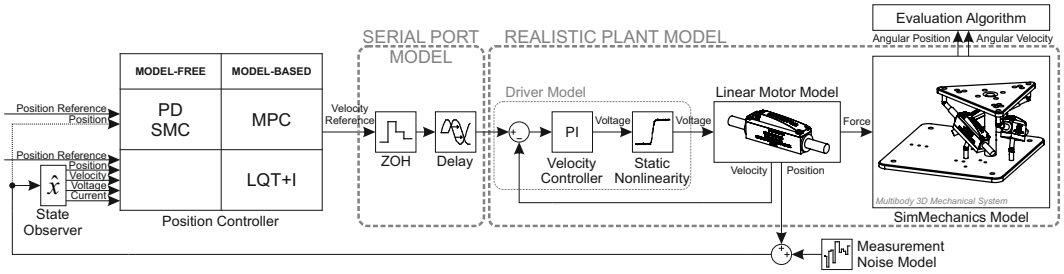


Figure 6.3: Realistic plant model and designed controller structures

6.1 Existing Controllers

In this section, the implementation details of the model-free and model-based controllers are given that is studied for both idealized and realistic configurations. At each situation, the design parameters of the controllers are subjected to the optimization process, while the top platform is driven by using pulse input at three-axes separately and independently. To minimize the motion-related image degradation by smoothing the angular velocity variation, and to avoid the exceeding of mechanical constraints by decreasing the overshoots, where possible, the design parameters are optimized to get an overdamped system behaviour. Note that, for roll, yaw and pitch axes movements of the top platform, the operating range of the three links are almost identical. So, the parameters of the three controllers are determined to be identical to each others as well. During optimization, it is aimed to control the related nonlinear idealized or realistic plant models. For model-based controllers, where necessary, linear plant model is obtained by linearizing the overall nonlinear model at the origin of the orientation space.

6.1.1 Model-Free Controllers

6.1.1.1 Proportional-Integral-Derivative Controller

The first effectuation is a single-loop Proportional-Integral-Derivative (PID) controller that is implemented to control the idealized configuration according to the algorithm given in Equation 6.1, where $F(t)$ is controller output in the unit of force, $e(t)$ is controller input in the unit of position error defined between position reference and position measurement, as well as K_P, K_I and K_D are proportional, integral and derivative gains, respectively.

$$F(t) = K_P e(t) + K_I \int_0^t e(\tau) d\tau + K_D \frac{de(t)}{dt} \tag{6.1}$$

In the result of optimization process, the fixed PID gains are computed as given below;

$$K_P = 237.8128, \quad K_I = 717.0607, \quad K_D = 5.4546$$

By preserving the structure of this algorithm, and modifying the conformation of the coefficients, a variation is obtained that can be called single-loop variable-gain PID controller. At this variant, the coefficients themselves are the exponential function of position error as detailed below;

$$\begin{aligned} K_P &= 861.3098 \times \exp\left(-416.5796 \times |e(t)|\right) + 62.6308 \\ K_I &= 1.7925 \times 10^3 \times \exp\left(-2.3489 \times 10^3 \times |e(t)|\right) + 166.2963 \\ K_D &= 32.9469 \times \exp\left(-1.9641 \times 10^3 \times |e(t)|\right) + 6.2981 \end{aligned}$$

The block diagrams of the single-loop fixed gain and single-loop variable gain PID controllers are shown in Figure 6.4.a and Figure 6.4.b, respectively.

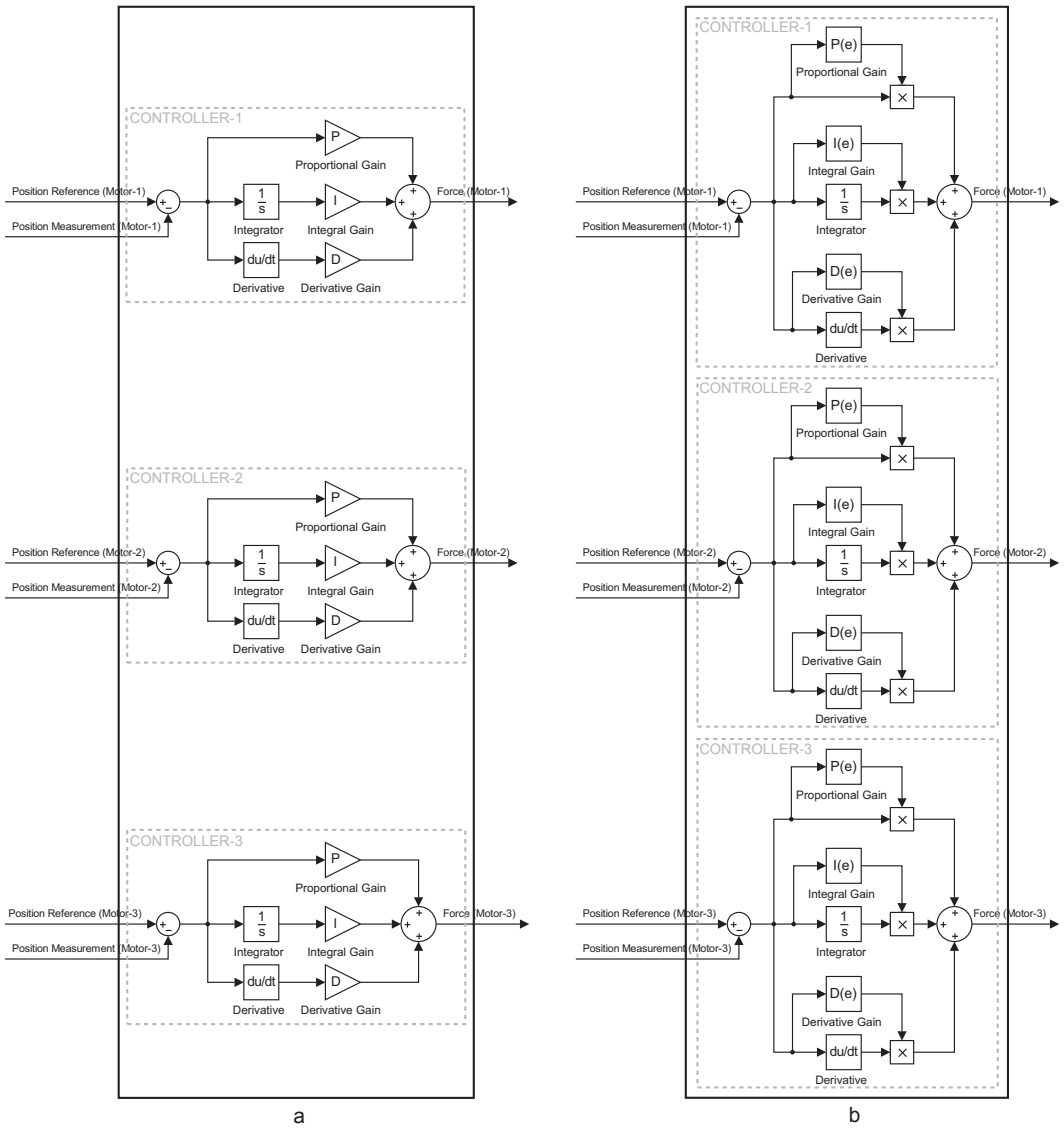


Figure 6.4: Block diagrams of single-loop a) fixed gain, b) variable gain PID controllers

Apart from the single-loop PID controllers described above, double-loop PD+PI controllers are also implemented for idealized plant. This kind of configuration is particularly preferred in the servomechanism control applications. In this arrangement, PD structure that is operated as a position controller is placed at the outer loop, whereas, PI structure that is operated as a velocity controller is placed at the inner loop as shown in Figure 6.5. The PD position controller accepts position error as input and calculates velocity reference as output. Correlatively, PI velocity controller accepts velocity error as input and calculates force as output. The optimized design parameters for the double-loop PD+PI controller are given below;

$$Position\ Controller : K_P = 25.9171, \quad K_D = 1.6942$$
$$Velocity\ Controller : K_P = 18.1255, \quad K_I = 192.2034$$

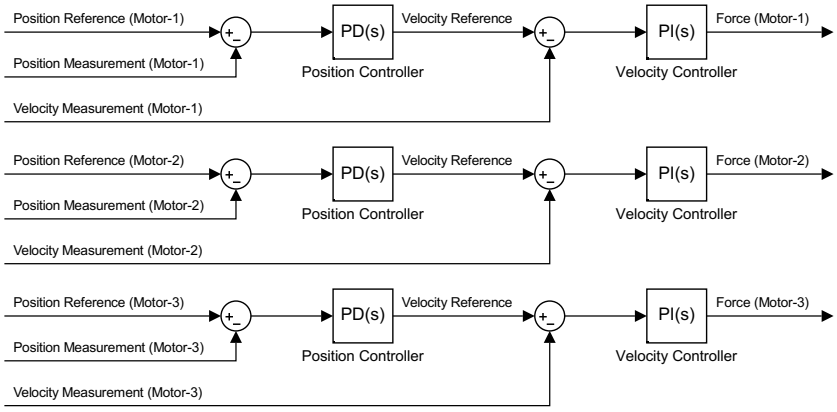


Figure 6.5: Block diagram of double-loop decoupled PD+PI controller

The controller shown in Figure 6.5 is designed under the assumption of independent links. In other words, it is supposed that the movement of a link does not impel the other links. However, they are connected to the each others through the base and top platforms. Therefore, the motion of a link results with a disturbance which expels the other links from the set point. To reject or at least to decrease these kind of effects originated from mechanism itself, previous controller is modified to obtain double-loop coupled PD+PI configuration as shown in Figure 6.6. At this variant, the inputs of the position and velocity controllers are determined by using the error values of all three links according to the coupling matrices.

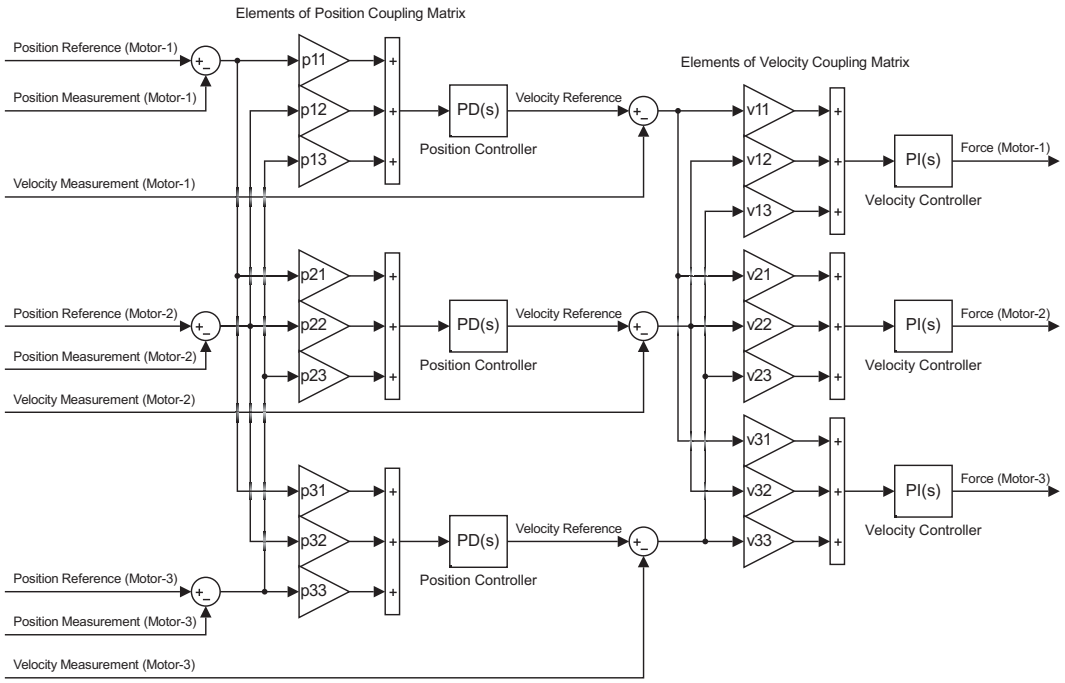


Figure 6.6: Block diagram of double-loop coupled PD+PI controller

For example, the input of the position controller at link-1 is calculated as;

$$p_{11} \cdot e_1(t) + p_{12} \cdot e_2(t) + p_{13} \cdot e_3(t)$$

where $e_1(t)$, $e_2(t)$, and $e_3(t)$ are the position error for link-1, link-2 and link-3, respectively. Similar computations can be done for velocity controller and for other links as well. The optimized design parameters and coupling matrices for the double-loop coupled PD+PI controller are given below;

$$Position\ Controller : K_P = 26.2343, \quad K_D = 1.6953$$
$$Velocity\ Controller : K_P = 3.1835, \quad K_I = 72.1661$$

$$CM_P = \begin{bmatrix} 1.0000 & 0 & 0 \\ 0 & 1.0000 & 0 \\ 0.0260 & 0.0470 & 0.9815 \end{bmatrix} \quad CM_V = \begin{bmatrix} 0.9479 & 0.2674 & 0.3988 \\ 0.3303 & 1.0000 & 0.2350 \\ 0.3257 & 0.3231 & 0.9390 \end{bmatrix}$$

Thus far, mentioned controllers are implemented for idealized plant model that is designed at continuous-time domain and outputted in the unit of force. However, these controllers can not be applied to the physical system, so a single-loop PD position controller is destinated for the realistic plant model as shown in Figure 6.7. It is designed to be a single-loop and devised to output in the unit of velocity reference, because there is an operational PI-type velocity controller on the motor driver of physical system. Thus, the position controller algorithmically implemented on the single-board computer and the velocity controller operated on motor driver form a double-loop controller structure. Note that, the serial communication at very low sampling frequency obligates the realization of this controller in discrete-time domain. The optimized design parameters for the PD-type position and PI-type velocity controllers are given below;

$$Position\ Controller : K_P = 13.6080, \quad K_D = 0.0192$$
$$Velocity\ Controller : K_P = 30.0000, \quad K_I = 90.0000$$

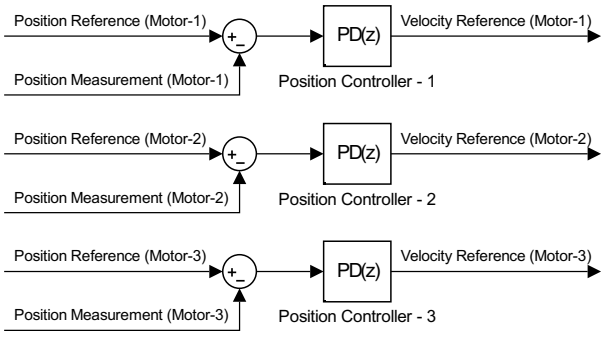


Figure 6.7: Block diagram of discrete PD controller implemented for realistic model

6.1.1.2 Sliding Mode Controller

Sliding mode control (SMC) is an extension of variable structure control (VSC) approach that belongs to the family of discontinuous nonlinear automatic control systems. At this method, the framework of the control law deliberately switched between some control rules according to the predefined criterions on system states. The multiple control structures are destinated so that the states started from an arbitrary initial condition are steered toward and kept on a well-designed surface in state-space.

For example, phase portraits of an unstable second order linear system that is compensated with positive or negative feedback are shown in Figure 6.8.a and Figure 6.8.b, respectively. Although, separately applied positive or negative feedback approach could not stabilize the system, the desired dynamic behaviour can be achieved by using variable structure control with a proper switching logic between these two control rules as shown in Figure 6.8.c. The differential equation describing the system dynamics and positive/negative feedback control rules are given in Equation 6.2.

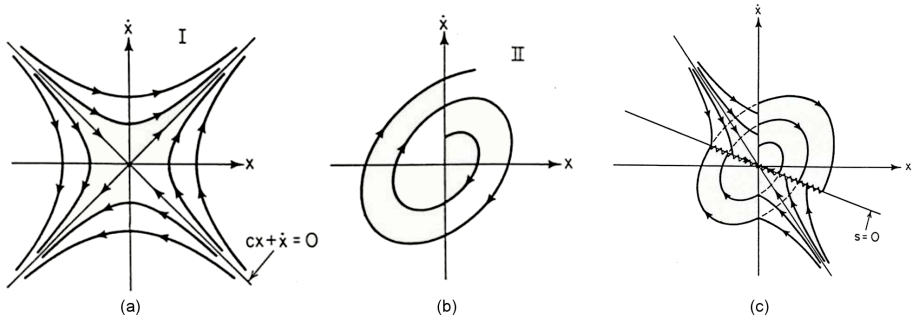


Figure 6.8: Phase portraits of an unstable second order linear system compensated with a) positive feedback b) negative feedback and c) variable structure control [68]

$$\ddot{x} + a_2\dot{x} + a_1 x = u, \quad where \quad u = k\,|x| \quad or \quad u = -k\,|x| \tag{6.2}$$

The dynamics and state trajectories of the system that is compensated with sliding mode control can be examined in two phases. The period necessary for starting from a point in state-space and attaining to a point on the well-designed surface is defined as reaching phase. Once the states hit to the surface, the sliding phase begins and the states move along the boundaries of the surface. This is why the surface is described as sliding manifold or sliding (hyper)surface. Due to the nature of the sliding mode control, the design procedure involves two issues; 1) selection of a control law that makes the sliding surface attractive to the system state, 2) selection of a manifold so that the sliding motion satisfies design specifications. Note that during sliding phase, the state trajectories are bounded to remain on the manifold, the describing functions are reduced to the surface equations and the system dynamics are restricted to the sliding dynamics. Hence it is possible to meet the design specifications and to satisfy the desired dynamics by properly selected surface equation.

Often, the sliding surface depends on the tracking error, e, together with a certain number of its derivatives.

$$\sigma = \sigma\left(e, \dot{e}, ..., e^k\right), \quad where \quad e = y_d(t) - y(t) \tag{6.3}$$

Here y_d and y indicate desired and measured outputs, respectively. Note that function σ defines a surface in the error space and should be selected in such a way that the tracking error, which is the solution of the differential equation, tends to zero eventually during the sliding phase ($\sigma = 0$). The most typical choice for the sliding manifold is a linear combination of the following type;

$$\sigma = e^{(k)} + \sum_{i=0}^{k-1} c_i e^{(i)} \tag{6.4}$$

For more information about sliding surface derivation refer to [62]. The number of derivatives to be included should be $k+1$, where k is the relative degree between number of input(s) and output(s). In this thesis, the camera stabilization platform has 3-inputs and 3-outputs, so the relative degree between them is zero. Therefore only tracking error and its first derivative are included to obtain the sliding surface. In Equation 6.5, the selected manifold and the solution of the differential equation are given. By selecting $c_0 > 0$, it is guaranteed that the tracking error gradually decreases and eventually tends to zero over time. Moreover, the value of this constant determines the convergence rate, so it is a critical design parameter to meet the desired specifications.

$$\begin{gathered} \sigma = \dot{e} + c_0 e = 0 \quad where \quad c_0 > 0 \\ e(t) = e(0)\exp(-c_0 t) \quad and \quad \dot{e}(t) = -c_0 e(0)\exp(-c_0 t) \end{gathered} \tag{6.5}$$

Apart from choosing the surface, the control law, which steers the state trajectories onto the sliding manifold, has to be determined to accomplish the compensator design. There are several approaches offering different control laws; first-order standard algorithm as given in Equation 6.6, second-order super-twist algorithm as given in Equation 6.7, and higher-order advanced algorithms [5], [42], as well as their variants. By using first order standard algorithm, the sliding mode controller is implemented for idealized model as shown in Figure 6.9.

$$u(t) = U \cdot \mathrm{sgn}(\sigma) \quad where \quad \mathrm{sgn}(x) = \begin{cases} +1 & x > 0 \\ -1 & x < 0 \end{cases} \tag{6.6}$$

$$\begin{aligned} u(t) &= \lambda \cdot \sqrt{|\sigma|} \cdot \mathrm{sgn}(\sigma) + w(t) \\ \dot{w}(t) &= W \cdot \mathrm{sgn}(\sigma) \end{aligned} \tag{6.7}$$

The design parameters subjected to the optimization process are given as follows;

$$c_1 = 0.4857, \;\; c_2 = 51.2980, \;\; U = 9.3461$$

Besides remarkable properties of accuracy, robustness, easy tuning and easy implementation, there are two main advantages of sliding mode control. Firstly, at least theoretically, the closed loop response becomes insensitive to model inaccuracies, uncertainties and disturbances. Since, at sliding phase, the system dynamics are reduced to the sliding dynamics,

which is determined by differential equation of hypersurface. Secondly, the value of the controller output is a design parameter, so appropriate selection of SMC gain provides to get rid of actuator saturation.

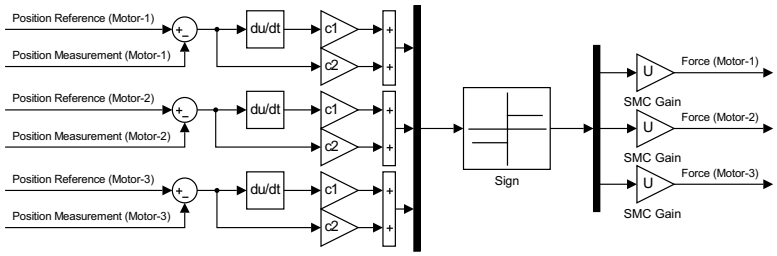


Figure 6.9: Block diagram of sliding mode controller implemented for idealized model

In steady state $(\sigma = 0)$, the controller output commutes between extreme values of SMC gain $(u(t) = +U \;\; or \;\; u(t) = -U)$ at very high frequency. Mentioned phenomenon that is defined as chattering issue leads to the emergence of significant problems especially at control applications of mechanical systems as in our case. One of the solution of chattering problem is replacing the discontinuous $\mathrm{sgn}(x)$ function with the smoothed continuous approximation. The functions given in Equation 6.8 are two of many examples.

$$\begin{aligned} Saturation &: u(t) = U \cdot \frac{\sigma}{|\sigma| + \varepsilon} \qquad \varepsilon > 0 \quad \varepsilon \approx 0 \\ Tangent\ Hyperbolic &: u(t) = U \cdot \tanh\left(\frac{\sigma}{\varepsilon}\right) \quad \varepsilon > 0 \quad \varepsilon \approx 0 \end{aligned} \tag{6.8}$$

By incorporating the saturation function instead of signum function, the modified sliding mode controller is implemented for realistic model as shown in Figure 6.10. Here, the parameter k is also a design parameter that adjusts the slope of the saturation function. It gradually begins to resemble to the signum function with decreasing value of parameter k. The design parameters subjected to the optimization process are given as follows;

$$Position\ Controller : c_1 = 0.0792, \;\; c_2 = 21.2874, \;\; V = 0.3427, \;\; k = 0.4378$$
$$Velocity\ Controller : K_P = 30.0000, \;\; K_I = 90.0000$$

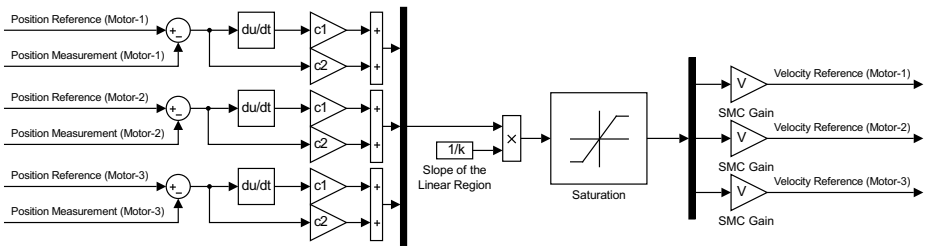


Figure 6.10: Block diagram of sliding mode controller implemented for realistic model

6.1.1.3 Proxy-Based Sliding Mode Controller

In [36], the proposed method that is called proxy-based sliding mode control (PBSMC), is described as an "alternative approximation of sliding mode control (SMC), and extension of proportional-integral-derivative (PID) control". The authors claim that the behaviour of the compensated system to small and large positional errors can be designed independently. In other words, the proposed method leads to the responsive tracking for small positional errors, while it avoids the excessive overshoots for large positional errors, which constitutes the main advantage of PBSMC over PID method. Also, as proven by the authors, signum function contained in a closed loop without latency or delay can be invariably replaced by saturation function. Hence, the problems arising from the discontinuous signum function, which is the source of chattering phenomenon, are solved consequently. The physical interpretation of this method is conceptualized at the following illustration as shown in Figure 6.11. Here, the desired position block and controlled object block can be thought as a reference generator and plant, respectively. According to the given desired position, the SMC portion of the compensator controls the virtual object, which is indicated as proxy block. Additionally, the PID portion of the compensator establishes a virtual spring-like coupling element between proxy and controlled object.

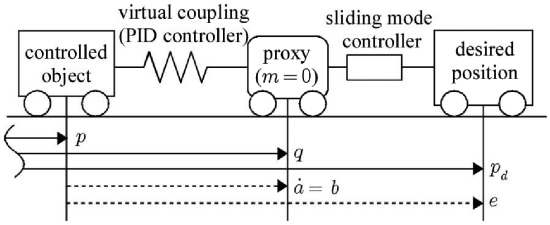


Figure 6.11: The physical interpretation of the proxy-based sliding mode control [36]

Let p_d and p denote the desired and measured position of the controlled object, as well as q indicates the measured position of the proxy. Then, the force produced by the PID portion of the compensator can be written as given in Equation 6.9, where K_P, K_I and K_D represent the proportional, integral and derivative gains.

$$F_{PID} = K_I \cdot a + K_P \cdot \dot{a} + K_D \cdot \ddot{a} \quad where, \quad a \triangleq \int (q - p)\, dt \tag{6.9}$$

Additionally, the force generated by the SMC portion of the compensator can be written as given in Equation 6.10.

$$F_{SMC} = U \mathrm{sgn}\left(c_1 \left(\dot{p}_d - \dot{q}\right) + c_2 \left(p_d - q\right)\right) \quad where, \quad U > 0 \tag{6.10}$$

Describing function of the system dynamics can be written as given in Equation 6.11, where M is the proxy mass.

$$M\ddot{q} = F_{SMC} - F_{PID} \tag{6.11}$$

In related literature, the proxy mass is assumed to be zero for avoiding undesired delays and unresponsive dynamics. However, in this thesis, both nonzero-mass and zero-mass cases are

examined for comparative purposes. Actually, it is obvious that increasing proxy mass leads to the smoother system dynamics. Hence our main purpose is using proxy mass as a design parameter to reduce the chattering problems. According to the Equation 6.9, Equation 6.10 and Equation 6.11, the PBSMC is implemented for nonzero and zero proxy mass cases as shown in Figure 6.12 and Figure 6.13, respectively. The design parameters subjected to the optimization process are given as follows;

$$
\begin{aligned}
0 < M \leq 1 : c_1 = 7.0531,\ \ c_2 = 169.3091,\ \ U = 6.4306,\ \ M = 0.9938 \\
: P = 1.0018 \cdot 10^4,\ \ I = 5.2783 \cdot 10^3,\ \ D = 95.9995 \\
M > 0,\ \ M \approx 0 : c_1 = 38.6647,\ \ c_2 = 1.7418 \cdot 10^3,\ \ U = 5.1996,\ \ M = 0.0995 \\
: P = 1.1085 \cdot 10^3,\ \ I = 6.5933 \cdot 10^3,\ \ D = 406.3069 \\
M = 0 : c_1 = 5.6692,\ \ c_2 = 435.9590,\ \ U = 3.0767,\ \ M = 0 \\
: P = 2.0369 \cdot 10^3,\ \ I = 285.1313,\ \ D = 436.6933
\end{aligned}
$$

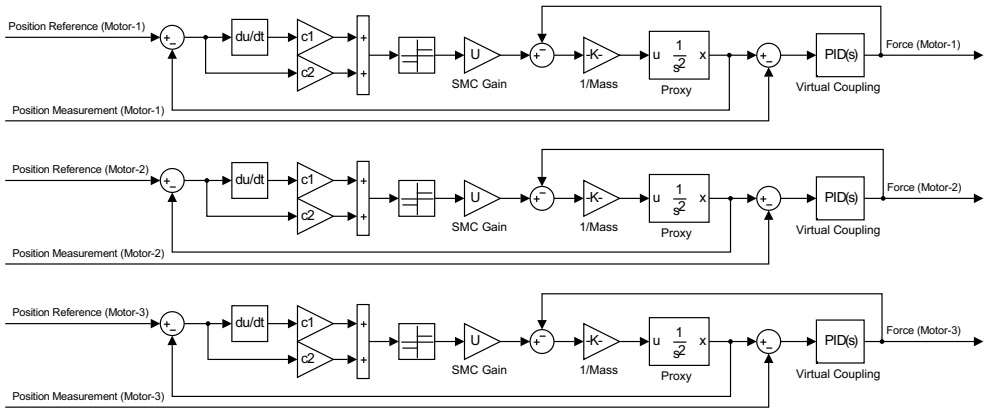


Figure 6.12: Block diagram of proxy-based sliding mode controller $(M > 0)$

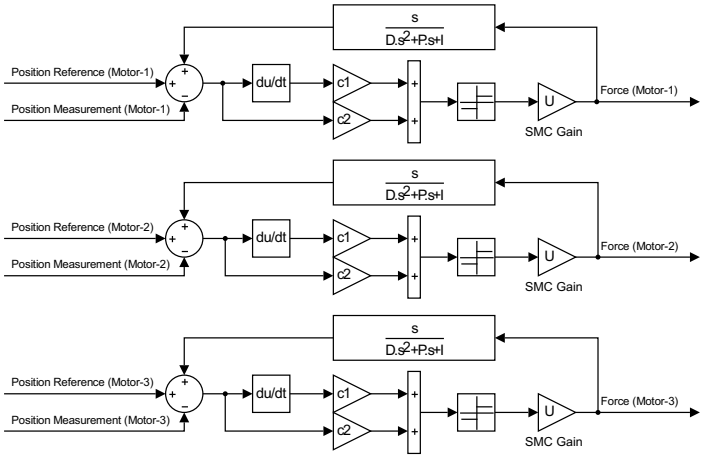


Figure 6.13: Block diagram of proxy-based sliding mode controller $(M = 0)$

The PBSMC can not be materialized and actualized for real system due to the limitations of physical facilities. Necessity to force feedback makes it impossible to perform such experiments in our application. Therefore, it is implemented only for idealized model.

6.1.1.4 Fuzzy Logic Controller

Fuzzy logic control (FLC) provides a formal methodology for representing, manipulating, and implementing the heuristic knowledge about how to control a system [55]. Basically, a fuzzy logic controller is an artificial decision maker that operates in a closed-loop application. It is composed of the following four elements;

1. A rule-base, which holds the knowledge in the form of if-then rules and contains quantification of the expert's linguistic description of how to achieve a good control.
2. An inference mechanism, which emulates the expert's decision making in interpreting and applying knowledge.
3. A fuzzification interface, which converts controller inputs into information that the inference mechanism can easily use to activate and apply rules.
4. A defuzzification interface, which converts the conclusions of the inference mechanism into actual inputs for the process.

Schematic diagram of the fuzzy logic controller operated within a closed-loop system is illustrated in Figure 6.14 with rule base, inference mechanism, fuzzification and defuzzification structures.

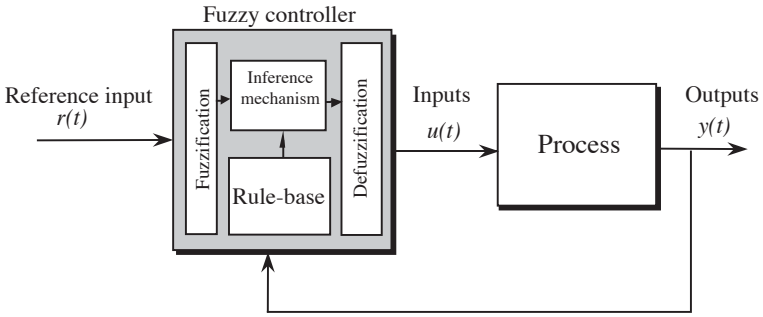


Figure 6.14: Block diagram of fuzzy controller within a closed loop application [55]

In this thesis, a fuzzy logic controller is implemented for idealized model as shown in Figure 6.15. It accepts position error and derivative of position error as inputs and generates link forces as outputs. The fuzzification and defuzzification interfaces have very simple structures for this application. Actually, they are normalization constants and scaling factors which map the physical quantities to membership functions definition range or vice versa. The fuzzification constant for position error, K_e, the fuzzification constant for derivative of position error,

K_{de}, and the defuzzification constant for link force, K_f, are design parameters of the FLC and subjected to the optimization process resulted as follows;

$$K_e = 0.0298, \quad K_{de} = 18.0662 \quad K_f = 8.0788$$

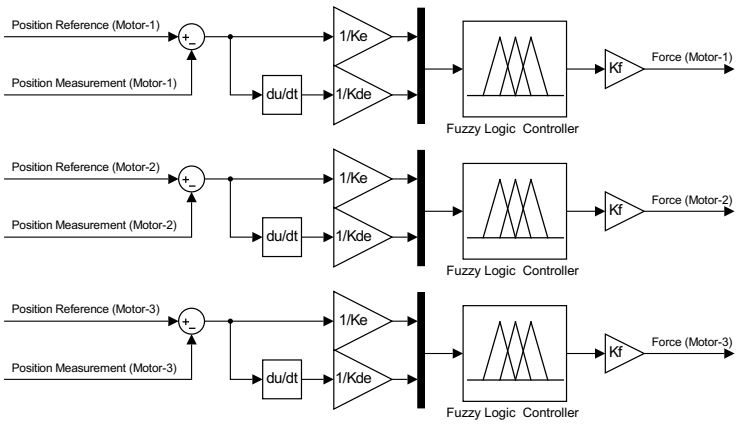


Figure 6.15: Block diagram of fuzzy logic controller

The triangular shaped membership functions, which are also design parameters of the controller, are optimized by using genetic algorithm routines. The resulted membership functions for inputs and output, as well as control surface are shown in Figure 6.16.

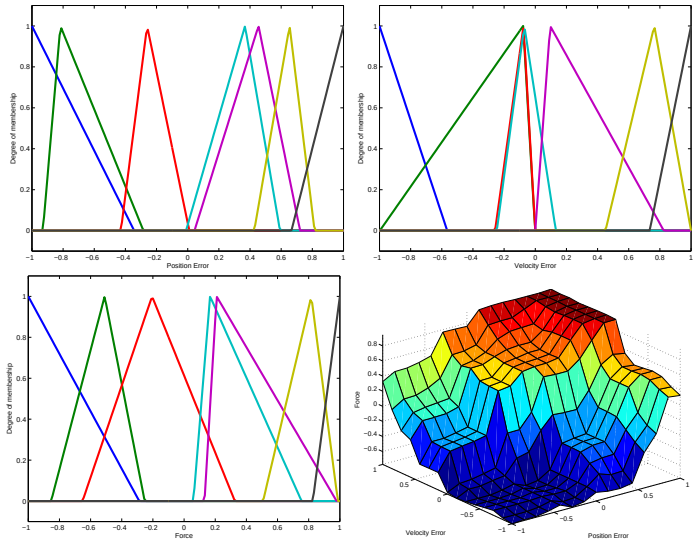


Figure 6.16: Triangular shaped optimized membership functions for a) position, b) derivative of position, c) force, as well as d) control surface

In [57], a detailed study on stability of fuzzy control systems is examined in terms of industrial regulations. Another reference specialized on predictive fuzzy control is [14]. Additionally, for more advanced fuzzy techniques refer to the [74].

6.1.1.5 Artificial Neural Network Controller

Artificial neural networks (ANNs) are developed by morphologically and computationally simulating a human brain [56]. They are generally represented as systems of interconnected artificial nodes, which are called neurons. After being weighted and transformed by an activation function, the outputs of these neurons are passed to the other neurons. The weights are conceptually connection strengths between neurons, that are recalculated during training process. The activation function might be any kind of linear or nonlinear mapping, yet differentiable ones have some advantages. For illustrative purposes, a single abstract neuron is shown in Figure 6.17 with its inputs, x_1, x_2, ..., x_n, its input weights, w_1, w_2, ..., w_n, and its activation function, $f(.)$. Furthermore, 3-inputs-2-outputs multilayer neural network architecture formed by bringing together in layers of these neurons are shown in Figure 6.18.

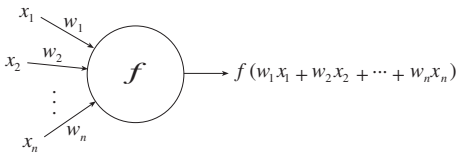


Figure 6.17: Structure of a single abstract neuron with n inputs

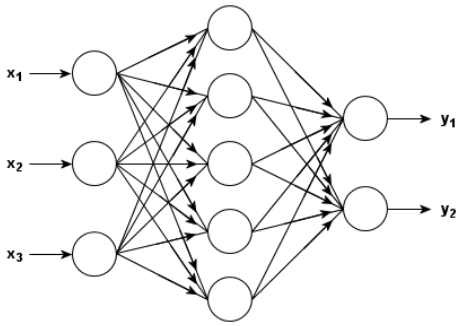


Figure 6.18: Architecture of 3-inputs-2-outputs multilayer artificial neural network

The properties of the ANNs include function approximation, learning, classification, etc. As a matter of course, they are used as closed-loop controllers in this thesis, but in a slightly different form. The training process of a network, which is determining the weights between neurons, are generally accomplished by using backpropogation algorithm. However in this study, only the network architecture is borrowed whereas the weights are optimized directly by using genetic algorithm instead of conventional training processes. As shown in Figure 6.19, the position reference, $r(k)$, the position measurement and its two past values, $p(k)$, $p(k-1)$, $p(k-2)$, the velocity measurement and its two past values, $v(k)$, $v(k-1)$, $v(k-2)$, and three past values of force measurement, $f(k-1)$, $f(k-2)$, $f(k-3)$ are defined as inputs of the network. There are two hidden layers and one output layer, where hidden

layer-1 and hidden layer-2 have 10 and 6 hidden neurons, respectively. Also, activation functions are selected as tangent hyperbolic for hidden layers and linear for output layer. The design parameters, which are normalization constants for position, S_p, velocity, S_v, force, S_f, and scaling factor for force, U, as well as weight matrices, W_1, W_2, W_3 and bias vectors, B_1, B_2, B_3, are subjected to the optimization process as given below;

$$S_p = 0.0078, \quad S_v = 2.2110 \quad S_f = 6.2908 \quad U = 1.2548$$

$$B_1 = 0.0051, \quad B_2 = 0.0429, \quad B_3 = 0.0381$$

$$W_1 = \begin{bmatrix} -0.0969 & 0.0858 & 1.3318 & -0.8003 & 0.3216 & 1.9388 & 1.8190 & 0.5573 & 0.4409 & -1.1938 \\ -0.4736 & 0.9764 & 0.3877 & -1.4525 & -0.5714 & -1.8143 & 1.6107 & 0.2087 & 1.9674 & 1.2669 \\ 1.8611 & -1.6130 & 1.0197 & -0.5234 & -1.8197 & -0.5024 & -0.7785 & 1.9698 & 1.8614 & 1.3916 \\ -0.7283 & 0.5119 & -0.1352 & 0.3327 & 1.2084 & 1.7486 & -0.4403 & -0.3852 & 0.3707 & -1.2364 \\ -0.0382 & 0.1580 & -1.4544 & -0.8475 & -0.8176 & 1.2065 & 2.1370 & -1.4299 & 1.2044 & 0.4614 \\ 0.3828 & -1.1603 & 1.5429 & 2.4521 & -1.4968 & 0.8494 & 0.9487 & 1.6080 & -0.2657 & -1.6223 \\ -0.0968 & -1.6225 & 0.5196 & -1.4161 & -1.2711 & -0.3486 & -0.2733 & 0.9210 & -0.9544 & 1.0149 \\ -1.2074 & -0.7002 & -1.1656 & 0.7367 & 1.5778 & -0.0327 & 1.9191 & 0.3422 & 0.0661 & 1.3514 \\ 0.4737 & -0.4481 & -0.4054 & -0.2825 & -0.3449 & 0.7817 & -2.9976 & -0.0857 & -1.7639 & -1.8310 \\ 0.9237 & -0.8499 & 2.0109 & 0.0975 & 0.9242 & -0.1624 & 0.9541 & 0.9700 & 2.1835 & -0.6140 \end{bmatrix}$$

$$W_2 = \begin{bmatrix} -0.4976 & -0.0709 & -1.5526 & 0.0543 & -1.0931 & -1.9524 & -1.1862 & -1.1417 & -0.0286 & 1.9587 \\ -0.1530 & 0.3741 & -1.4979 & -1.9957 & 0.3842 & -0.8861 & 1.1320 & -0.4033 & 1.4336 & 0.2755 \\ -1.1130 & -0.8671 & 1.2731 & 2.0368 & 0.2713 & 0.9004 & 0.8351 & -1.6668 & -0.3460 & -0.8825 \\ 2.1684 & -0.6166 & 1.4006 & 1.1330 & -0.1414 & 2.4540 & 0.3185 & 1.7575 & 0.5407 & 0.4290 \\ -1.9863 & 0.4659 & -0.0086 & -1.5246 & 0.2600 & 0.1514 & -0.4621 & 0.4348 & 1.4096 & 0.4062 \\ 0.3503 & 0.7218 & 1.5055 & -0.6651 & -0.4130 & -1.2922 & -1.4037 & 1.0073 & 0.9692 & 1.8387 \end{bmatrix}$$

$$W_3 = \begin{bmatrix} -0.7995 & 1.2296 & 2.1528 & 0.5313 & 1.3360 & 0.8792 \end{bmatrix}$$

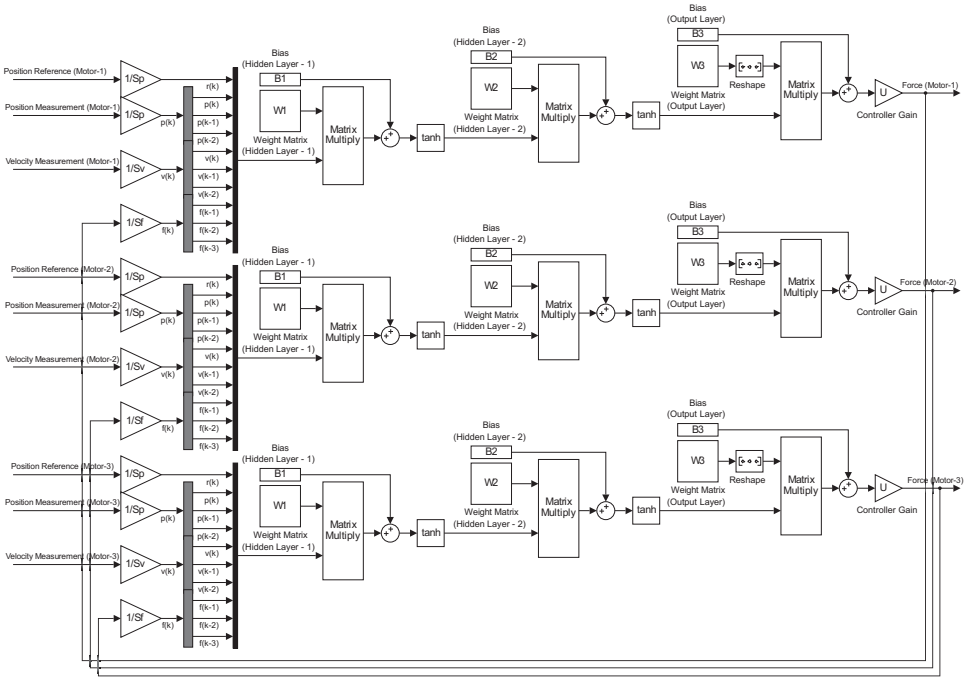


Figure 6.19: Block diagram of neural network controller implemented for idealized model

6.1.2 Model-Based Controllers

6.1.2.1 Linear Quadratic Tracker

Optimal control theory is a mathematical optimization method for deriving control policies that operates the dynamic systems at minimum cost. The general optimal control problem reduces to the linear quadratic (LQ) problem at the case where the system dynamics are described by a set of linear differential equations and the cost is described by a quadratic function. Linear quadratic regulator (LQR), a particular form of the LQ, deals with finding an optimal control law;

$$u\left(t\right) = -Kx\left(t\right) \tag{6.12}$$

$$K = R^{-1}B^{T}S \tag{6.13}$$

$$0 = -SA - A^{T}S + SBR^{-1}B^{T}S - Q \tag{6.14}$$

by minimizing the infinite horizon quadratic continuous-time cost functional;

$$J = \frac{1}{2}\int_{0}^{\infty}\left[x(t)^{T}Qx\left(t\right) + u(t)^{T}Ru\left(t\right)\right]dt \tag{6.15}$$

subject to the linear time-invariant first-order dynamic constraints and the initial condition;

$$\dot{x}\left(t\right) = Ax\left(t\right) + Bu\left(t\right), \quad x\left(t_{0}\right) = x_{0} \tag{6.16}$$

where, A and B are system matrices, $x\left(t\right)$ and $u\left(t\right)$ are state and input vectors, Q and R are positive semi-definite weighting matrices for states and inputs, respectively, K is feedback gain matrix, S is the solution of algebraic Riccati equation given in 6.14. The block diagram of the LQR operated in a closed-loop application is illustrated as shown in Figure 6.20.

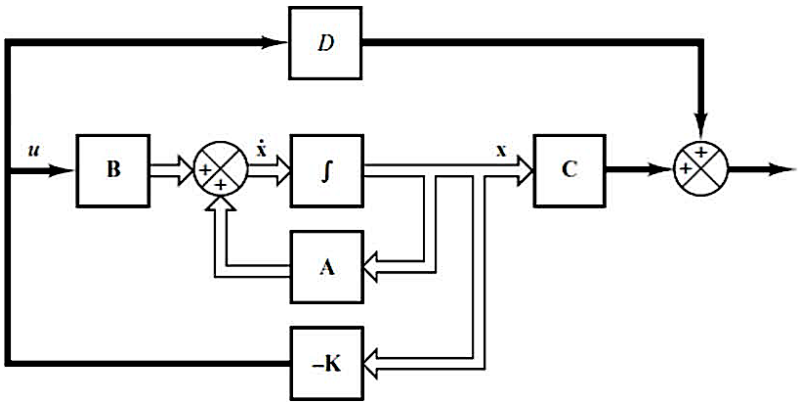


Figure 6.20: Block diagram of linear quadratic regulator (LQR) operated in a closed-loop system illustrated with state (A), input (B), output (C) and feedthrough (D) matrices

As its name implies, LQR is used to regulate the closed-loop system which can be explained as moving the internal states to the origin from an arbitrary point in the state space. However,

if feedback gain acts on error values instead of internal states, then the regulator structure can be directly used for tracking applications as shown in Figure 6.21. Looked at from this perspective, the new configuration that is called linear quadratic tracker (LQT), strives to equate the error values to zero. The state space model obtained by linearization of the camera stabilization platform at zero point of the orientation space is given in Equation 6.17, where, A and B are linearized system matrices, $\vec{x}$ internal state vector of link positions, p_1, p_2, p_3, and link velocities, v_1, v_2, v_3, as well as $\vec{u}$ input vector of link forces, f_1, f_2, f_3. The design parameters of the implemented controller are subjected to the optimization process as given in Equation 6.18. Also, the calculated LQT gain, which is obtained by using optimized weight matrices, is given in Equation 6.19.

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 52.6427 & 0 & 255.6256 & 0 & 264.0721 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 260.8788 & 0 & 47.3127 & 0 & 258.5088 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 262.5837 & 0 & 258.5487 & 0 & 57.3743 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 19.1946 & -7.3438 & -7.4777 \\ 0 & 0 & 0 \\ -7.3438 & 19.4614 & -7.3635 \\ 0 & 0 & 0 \\ -7.4777 & -7.3635 & 19.1263 \end{bmatrix} \quad \vec{x} = \begin{bmatrix} p_1 \\ v_1 \\ p_2 \\ v_2 \\ p_3 \\ v_3 \end{bmatrix} \quad \vec{u} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \end{bmatrix} \tag{6.17}$$

$$\begin{aligned} Q &= diag \begin{bmatrix} q_1 & q_2 & q_3 & q_4 & q_5 & q_6 \end{bmatrix} \\ R &= diag \begin{bmatrix} r_1 & r_2 & r_3 \end{bmatrix} \\ q_1 &= q_3 = q_5 = 4.9972 \cdot 10^5 \\ q_2 &= q_4 = q_6 = 5.8462 \\ r_1 &= r_2 = r_3 = 1.5281 \end{aligned} \tag{6.18}$$

$$K = \begin{bmatrix} 614.5115 & 10.5564 & 49.4546 & 3.6997 & 50.7187 & 3.7767 \\ 49.6636 & 3.6986 & 612.8433 & 10.4260 & 49.7668 & 3.7121 \\ 50.6648 & 3.7777 & 49.7581 & 3.7142 & 615.0726 & 10.5890 \end{bmatrix} \tag{6.19}$$

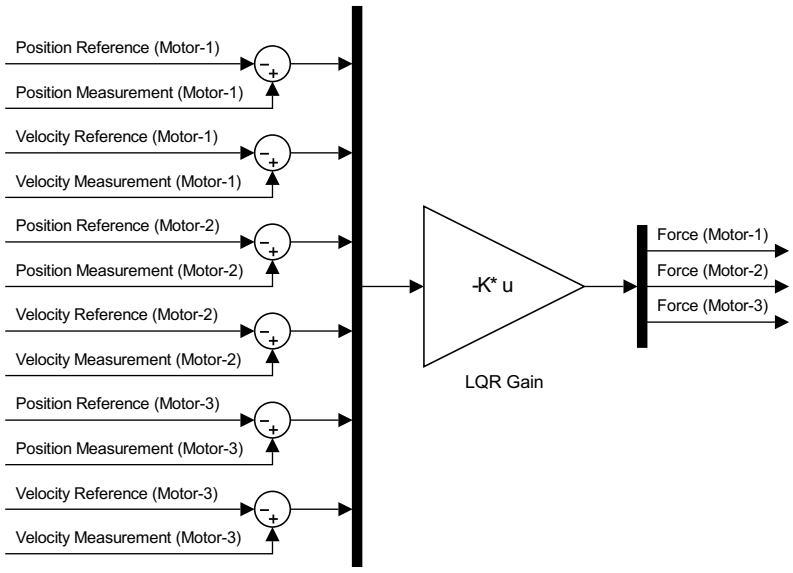


Figure 6.21: Block diagram of linear quadratic tracker implemented for idealized model

In some applications where the model inaccuracies exist originated from unknowns or approximations, the desired results may not be obtained by using LQT. In such situations, this approach is modified by incorporating the integral action as shown in Figure 6.22. Note that the feedback gain acts on internal system states, as well as integral of the position error. At this modified version, LQT+I, the determination of weight matrices and calculation of feedback gain are carried out by using augmented system model, which is given in Equation 6.20, where A_{aug} and B_{aug} are augmented system matrices, $\vec{x}_{aug}$ is augmented state vector of link positions, p_1, p_2, p_3, link velocities, v_1, v_2, v_3 and newly defined states, $\int (r_1 - p_1)\, dt$, $\int (r_2 - p_2)\, dt$, $\int (r_3 - p_3)\, dt$. The design parameters of the implemented controller are subjected to the optimization process as given in Equation 6.21. Also, the calculated LQT+I gain, which is obtained by using optimized weight matrices, is given in Equation 6.22.

$$A_{aug} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 52.6427 & 0 & 255.6256 & 0 & 264.0721 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 260.8788 & 0 & 47.3127 & 0 & 258.5088 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 262.5837 & 0 & 258.5487 & 0 & 57.3743 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \end{bmatrix} \quad B_{aug} = \begin{bmatrix} 0 & 0 & 0 \\ 19.1946 & -7.3438 & -7.4777 \\ 0 & 0 & 0 \\ -7.3438 & 19.4614 & -7.3635 \\ 0 & 0 & 0 \\ -7.4777 & -7.3635 & 19.1263 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \vec{x}_{aug} = \begin{bmatrix} p_1 \\ v_1 \\ p_2 \\ v_2 \\ p_3 \\ v_3 \\ \int (r_1 - p_1)\, dt \\ \int (r_2 - p_2)\, dt \\ \int (r_3 - p_3)\, dt \end{bmatrix} \tag{6.20}$$

$$\begin{aligned} Q_{aug} &= diag \begin{bmatrix} q_1 & q_2 & q_3 & q_4 & q_5 & q_6 & q_7 & q_8 & q_9 \end{bmatrix} \\ R_{aug} &= diag \begin{bmatrix} r_1 & r_2 & r_3 \end{bmatrix} \\ q_1 &= q_3 = q_5 = 3.7356 \cdot 10^3 \\ q_2 &= q_4 = q_6 = 0.0491 \\ q_7 &= q_8 = q_9 = 5.4211 \cdot 10^7 \\ r_1 &= r_2 = r_3 = 2.9024 \cdot 10^{-4} \end{aligned} \tag{6.21}$$

$$K = \begin{bmatrix} 6.6649 \cdot 10^3 & 37.3081 & 813.3208 & 12.4545 & 830.2945 & 12.7125 & -4.3218 \cdot 10^5 & -2.1164 & 1.8916 \\ 813.5247 & 12.4542 & 6.6366 \cdot 10^3 & 36.8735 & 816.3875 & 12.4999 & 2.1164 & -4.3218 \cdot 10^5 & 3.9864 \\ 830.2494 & 12.7128 & 816.3926 & 12.5006 & 6.6719 \cdot 10^3 & 37.4126 & -1.8916 & -3.9864 & -4.3218 \cdot 10^5 \end{bmatrix} \tag{6.22}$$

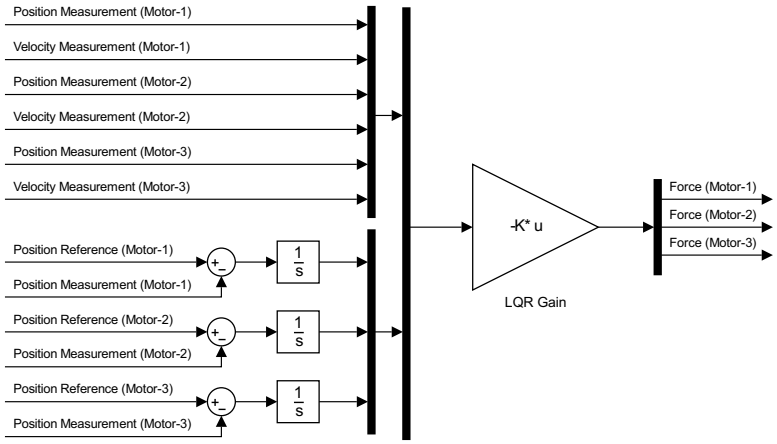


Figure 6.22: Block diagram of modified linear quadratic tracker supplemented with integral action and designed over augmented model

Thus far, mentioned controllers are implemented for idealized plant model that is designed at continuous-time domain and outputted in the unit of force. However, the serial communication at very low sampling rate obligates that the controller design has to be done in discrete-time domain. Also, due to the operational velocity controller on the motor driver, the output of the position controller must be in the unit of velocity reference.

Additionally, it is assumed that all of the states are measurable without any measurement noise. But, the limitations of physical system restrict us to measure only link positions. Hence, these controllers can not be applied to the physical system directly.

For this purpose, a more appropriate controller structure, which is in the form of LQT+I incorporated with state observer, is implemented for realistic model as shown in Figure 6.23. If a Kalman filter is used as a state observer, then it is possible to combine observer and compensator parts, which is called linear quadratic gaussian (LQG) in the related literature.

The system matrices and state vector of the linearized realistic model are given in Equation 6.23, where A_{aug} and B_{aug} are augmented system matrices, $\vec{x}_{aug}$ is augmented state vector of link positions, p_1, p_2, p_3, link velocities, v_1, v_2, v_3, motor voltages, v_1, v_2, v_3, motor currents, i_1, i_2, i_3, and newly defined states, $\int (r_1 - p_1)\, dt$, $\int (r_2 - p_2)\, dt$, $\int (r_3 - p_3)\, dt$. Remember that, the motor driver and motor models are excluded from the idealized case, however included to realistic one. Therefore, dissimilarly from the idealized case, the state vector of the realistic model have link positions and link velocities, as well as motor voltages and motor currents.

The design parameters of the implemented controller are subjected to the optimization process as given in Equation 6.24. Also, the calculated LQT+I gain, which is obtained by using optimized weight matrices, is given in Equation 6.25.

$$
A_{aug} = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
140.3284 & -34.3521 & 17.7692 & -11.2814 & 18.5499 & -10.4499 & 2.1515 \cdot 10^4 & 0 & -5.0404 \cdot 10^3 & 0 & -6.4023 \cdot 10^3 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
23.4342 & -12.7580 & 116.8332 & -35.2714 & 17.9466 & -12.6214 & -5.0404 \cdot 10^3 & 0 & 2.1654 \cdot 10^4 & 0 & -5.2862 \cdot 10^3 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-14.7042 & -6.7134 & 4.8965 & -7.4702 & 119.4793 & -28.8508 & -6.4023 \cdot 10^3 & 0 & -5.2862 \cdot 10^3 & 0 & 1.9296 \cdot 10^4 & 0 & 0 & 0 & 0 \\
0 & -45.4985 & 0 & 0 & 0 & 0 & -1.6061 \cdot 10^4 & 40.2485 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -569.3302 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -45.4985 & 0 & 0 & 0 & 0 & -1.6061 \cdot 10^4 & 40.2485 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -569.3302 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -45.4985 & 0 & 0 & 0 & 0 & -1.6061 \cdot 10^4 & 40.2485 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -569.3302 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

$$
B_{aug} = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
40.2485 & 0 & 0 \\
569.3302 & 0 & 0 \\
0 & 40.2485 & 0 \\
0 & 569.3302 & 0 \\
0 & 0 & 40.2485 \\
0 & 0 & 569.3302 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\quad
\vec{x}_{aug} = \begin{bmatrix}
p_1 \\ v_1 \\ p_2 \\ v_2 \\ p_3 \\ v_3 \\ v_1 \\ i_1 \\ v_2 \\ i_2 \\ v_3 \\ i_3 \\ \int (r_1 - p_1)\, dt \\ \int (r_2 - p_2)\, dt \\ \int (r_3 - p_3)\, dt
\end{bmatrix}
\tag{6.23}
$$

$$
\begin{aligned}
Q_{aug} &= diag\left[\begin{array}{cccccccccccccccc} q_1 & q_2 & q_3 & q_4 & q_5 & q_6 & q_7 & q_8 & q_9 & q_{10} & q_{11} & q_{12} & q_{13} & q_{14} & q_{15} \end{array}\right] \\
R_{aug} &= diag\left[\begin{array}{ccc} r_1 & r_2 & r_3 \end{array}\right] \\
q_1 &= q_3 = q_5 = 4.5885 \cdot 10^3 \\
q_2 &= q_4 = q_6 = 64.7307 \\
q_7 &= q_9 = q_{11} = 37.4485 \\
q_8 &= q_{10} = q_{12} = 16.7699 \\
q_{13} &= q_{14} = q_{15} = 1.6573 \cdot 10^7 \\
r_1 &= r_2 = r_3 = 870.2922
\end{aligned} \tag{6.24}
$$

$$
K = \left[\begin{array}{ccccccccccccccc}
18.0212 & 0.0054 & 0.5207 & -0.0190 & 0.4685 & -0.0238 & 0.0258 & 0.1001 & -0.0203 & 0.0020 & -0.0253 & 0.0039 & -137.9951 & -0.3001 & 0.7294 \\
0.4535 & -0.0227 & 17.9478 & 0.0055 & 0.3942 & -0.0263 & -0.0224 & 0.0021 & 0.0264 & 0.0998 & -0.0251 & 0.0027 & 0.3052 & -137.9936 & 0.9746 \\
0.6034 & -0.0153 & 0.6220 & -0.0141 & 18.0982 & 0.0082 & -0.0202 & 0.0035 & -0.0177 & 0.0022 & 0.0235 & 0.1027 & -0.7273 & -0.9762 & -137.9920
\end{array}\right] \tag{6.25}
$$

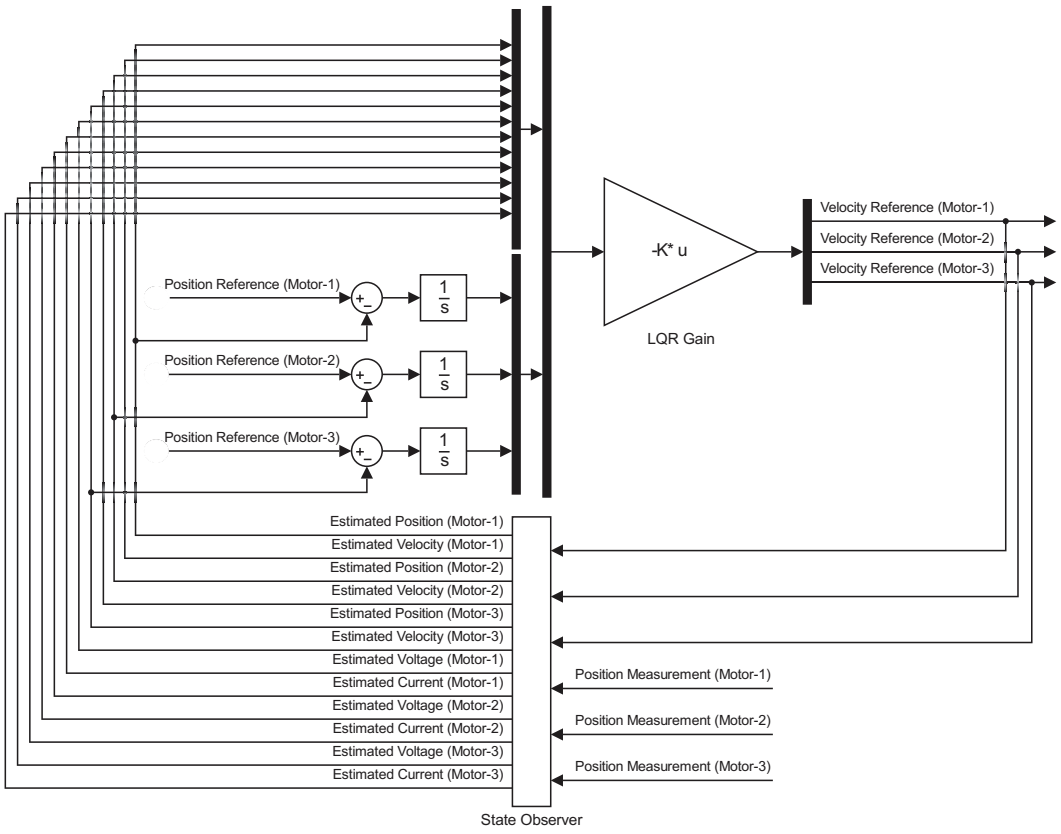


Figure 6.23: Block diagram of linear quadratic tracker with integral action and attached state observer implemented for realistic model

6.1.2.2 Model Predictive Controller

Model Predictive Control (MPC) is an optimization based multivariable strategy that attempts to find an optimal control sequence in discrete-time domain by taking into account input and output constraints of the plant. The MPC predicts and anticipates future responses by using system dynamics and measurements to optimize the plant inputs at each sampling time.

Through predictive features, it successfully manages to control processes with large time delays and high-order dynamics that are difficult for conventional controllers. The block diagram of the MPC embedded in a closed-loop application, is shown in Figure 6.24 and related definitions are given as follows;

- Plant Inputs
  - Manipulated Variables: The controller adjusts these in order to achieve its goals.
  - Measured Disturbances: The controller can't adjust them, but uses them for feed-forward compensation.
  - Unmeasured Disturbances: These are independent inputs of which the controller has no direct knowledge, and for which it must compensate.
- Plant Outputs
  - Measured Outputs: The controller uses these to estimate unmeasured quantities and as feedback on the success of its adjustments.
  - Unmeasured Outputs: The controller estimates these based on available measurements and the plant model.

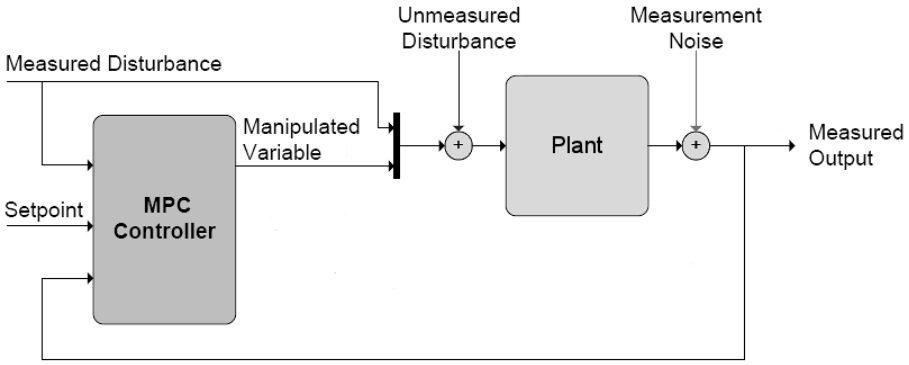


Figure 6.24: Block diagram of model predictive controller

In a MPC framework the control goals, such as the tracking of a reference or the satisfaction of constraints, are formulated as a numerical optimization problem. In most cases this problem is represented as a Quadratic programming or QP problem. For such an optimization problem, the cost function is the additive sum of individual terms that express various control requirements. These terms are multiplied by weighting factors defining the relative importance of each individual control goal. [50] The MPC deals with finding an optimal control sequence;

$$u^* = \left[\begin{array}{cccc} u^*(t) & u^*(t+1) & \ldots & u^*(t+N-1) \end{array} \right] \tag{6.26}$$

by minimizing the quadratic discrete-time cost functional at time t over a N steps horizon;

$$J = \sum_{k=0}^{N-1} w_k^x (r_k - x_k)^2 + \sum_{k=0}^{N-1} w_k^u \Delta u_k^2 \tag{6.27}$$

subject to the linear time-invariant first-order discrete-time system, the input/output constraints and the initial conditions;

$$\begin{aligned} x_{t+k+1} &= Ax_{t+k} + Bu_{t+k} \\ y_{t+k} &= Cx_{t+k} + Du_{t+k} \\ u_{\min} &\leq u_{t+k} \leq u_{\max} \\ y_{\min} &\leq y_{t+k} \leq y_{\max} \\ x\,(t) &= x_t, \quad k = 0, ..., N-1 \end{aligned} \tag{6.28}$$

An infinite horizon suboptimal controller can be designed by repeatedly solving finite time optimal control problems in a receding horizon fashion. [7] This means that the resulting optimization problem needs to be solved at each sampling interval based on actual measurements. [50] At each sampling time, starting at the current state, an open-loop optimal control problem is solved over a finite horizon as shown in Figure 6.25.a. The computed optimal manipulated input signal is applied to the process only during the following sampling interval $[t, t+1]$. At the next time step, $t+1$, a new optimal control problem based on new measurements of the state is solved over a shifted horizon as shown in Figure 6.25.b. The resulting controller is referred to as a Receding Horizon Controller (RHC). [7]

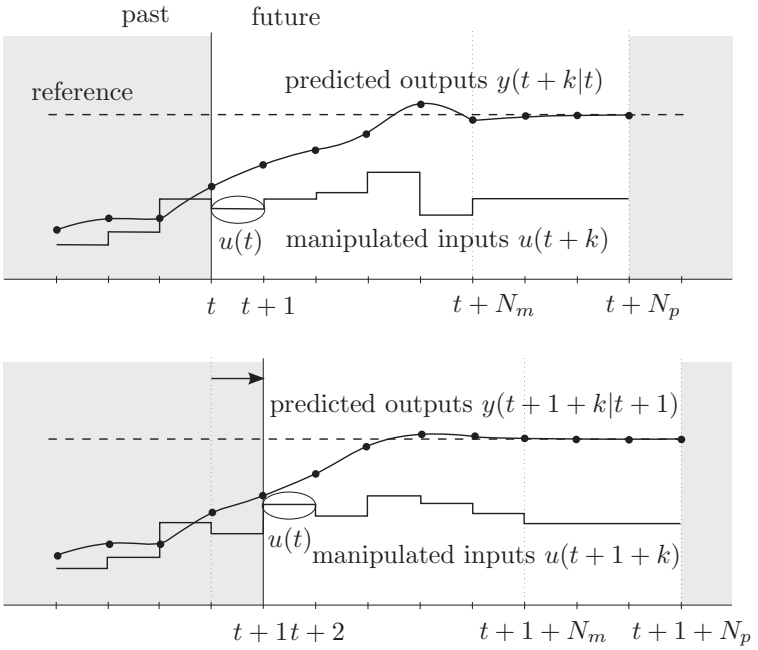


Figure 6.25: Predicted plant output and optimized sequence of control input
a) at time t, b) at time $t+1$ [7]

A RCH where the finite time optimal control law is computed by solving an optimization problem on-line is usually referred to as MPC. To reduce the computational complexity, in some situations, the MPC problems can be reformulated to multi-parametric quadratic programming or mp-QP problems. A problem formulated in this manner can be solved off-line

once only for a given range of parameters; then at each sampling time the MPC controller is effectively reduced to a look-up table process. This represents a very fast controller implementation solution. It should be noted that this approach is not an approximation but gives exactly the same results as if the optimization problem would have been solved at each sampling interval. [50]

The controllers are implemented by using Model Predictive Toolbox routines of MATLAB / Simulink environment for both idealized model, as shown in Figure 6.26, and realistic model. At each cases, weight matrices for manipulated variable, W_u, manipulated variable rate, $W_{\Delta u}$, and measured output, W_y, as well as prediction horizon, H_p, and control horizon, H_c, parameters are subjected to the optimization process. Also, some improvements related with estimation model, are accomplished by tuning the Kalman filter gain. The optimized weight matrices are given below;

$$W_u = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$W_{\Delta u} = \begin{bmatrix} 8.8876 \cdot 10^{-4} & 0 & 0 \\ 0 & 8.8876 \cdot 10^{-4} & 0 \\ 0 & 0 & 8.8876 \cdot 10^{-4} \end{bmatrix}$$

$$W_y = \begin{bmatrix} 1.5343 \cdot 10^{6} & 0 & 0 \\ 0 & 1.5343 \cdot 10^{6} & 0 \\ 0 & 0 & 1.5343 \cdot 10^{6} \end{bmatrix}$$

$$H_p = 11, \quad H_c = 2, \quad T_s = 0.01\,[s]$$

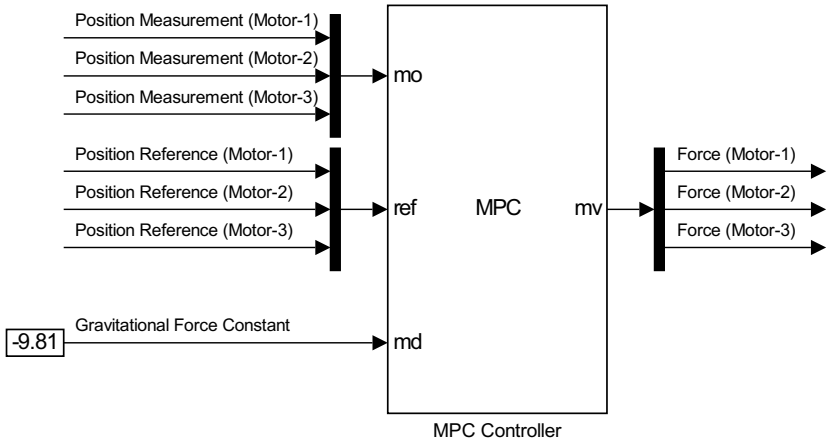


Figure 6.26: Block diagram of model predictive controller implemented for idealized model

6.2 Existing Predictors

It is difficult to obtain satisfactory performances of control systems with time delay, which is a well recognized problem in many control processes. Time delay, also called dead time, is mostly aroused by transportation lags, measurement lags, analysis times, computation and communication lags. And it exists in a lot of systems such as industrial process control systems, engineering systems, economical, and biological systems. [34] In our application, there exist known time delay originated from serial port communication interface, as well as motor driver processing issues. In this section, widely-used delay compensators are examined to understand the applicability into our system.

6.2.1 Smith Predictor

Assume that it is desired to control a process with time delay as shown in Figure 6.27.

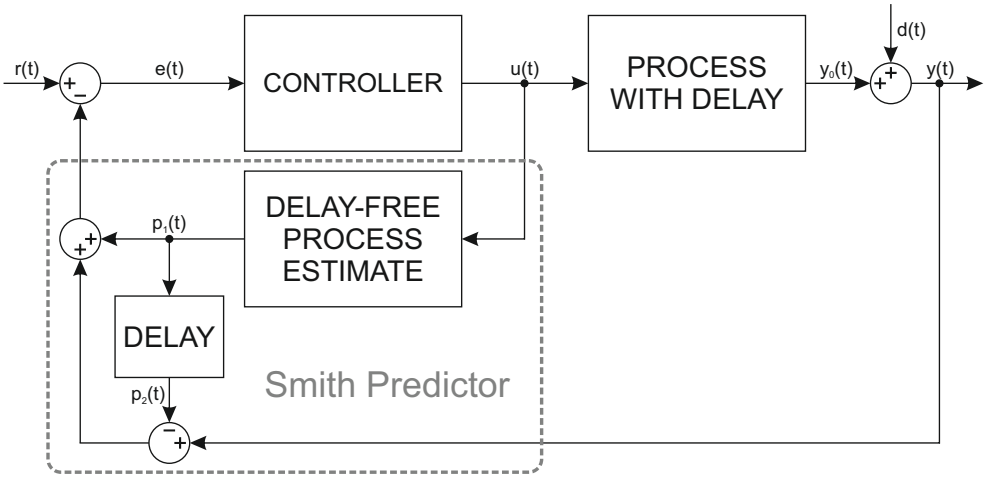


Figure 6.27: Block diagram of Smith predictor embedded in a closed-loop control application with a generic compensator and process with known delay

Let the transfer function of the controller is $G_c(s)$, and the transfer function of the process is $G_p(s) \cdot e^{-\tau s}$, where τ is known time delay. Then, the closed-loop transfer function without Smith predictor would be;

$$\frac{Y(s)}{R(s)} = \frac{G_c(s) G_p(s) \cdot e^{-\tau s}}{1 + G_c(s) G_p(s) \cdot e^{-\tau s}} \tag{6.29}$$

And the closed loop characteristic equation is;

$$1 + G_c(s) G_p(s) \cdot e^{-\tau s} = 0 \tag{6.30}$$

Note that the time delay term, $e^{-\tau s}$, in characteristic equation produces phase lag that makes it difficult to control, even makes the system unstable. On the other hand, the closed loop

transfer function with Smith predictor would be;

$$\frac{Y(s)}{R(s)} = \frac{G_c(s)\,G_p(s)\cdot e^{-\tau s}}{1 + G_c(s)\,G_p(s)} \tag{6.31}$$

And the corresponding characteristic equation is;

$$1 + G_c(s)\,G_p(s) = 0 \tag{6.32}$$

Note that, by using Smith predictor, the time delay is eliminated from characteristic equation which improves the control performance significantly. It can be easily shown that, the block diagram given in Figure 6.27 is equivalent to the block diagram given in Figure 6.28. It is obvious that the equivalent transformation of the Smith predictor puts the time delay term to the outside of the loop which indicates that system dynamics do not affected. [34]

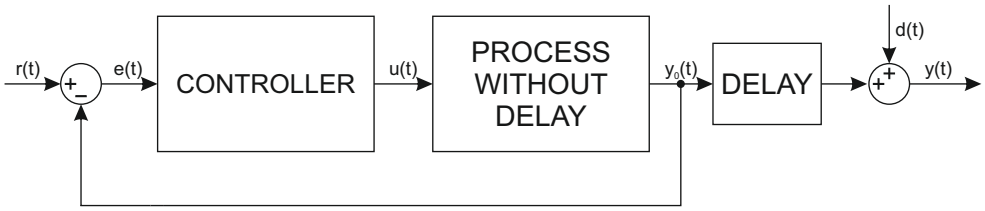


Figure 6.28: Equivalent block diagram with operational Smith predictor

6.2.2 Polynomial-Based Predictor

In [29], a new seismic experimental method for structural systems is examined that combines the output of the actuator-excited vibration experiment and response calculation method in real time. However, as shown in Figure 6.29, there is an inevitable response delay originated from the dynamic characteristics of the hydraulic actuators which makes the proposed method impracticable. [29]

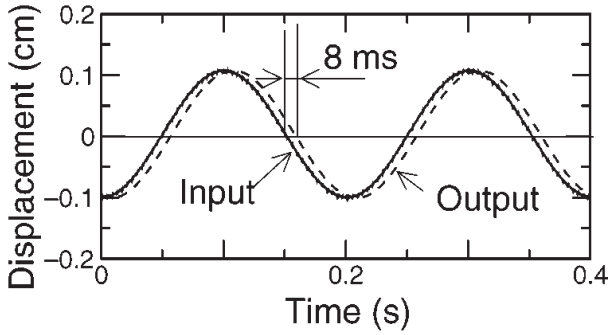


Figure 6.29: Time history of input and output signals for a hydraulic actuator [29]

To overcome this problem, a polynomial-based forward prediction method is proposed which compensates the response delay of the actuator. Assume that, x be the target signal, which

is desired to be followed by the hydraulic actuator as shown in Figure 6.30.a. If the future values of the target signal can be predicted, then by inputting the predicted values, x', as a control signal to the actuator, the resulting displacement, x'', becomes almost identical to the target values. As a prediction model the following equation is used;

$$x' = \sum_{i=0}^{N} a_i x_i \tag{6.33}$$

where, n is the order of prediction, x_0 is the present calculated displacement, x_i is the calculated displacement at δt x i units of time ago, and a_i are the design parameters. In this equation, the predicted value, x', is obtained by extrapolating an nth-order polynomial function based on the present and n previous calculated values, as shown in Figure 6.30.b.

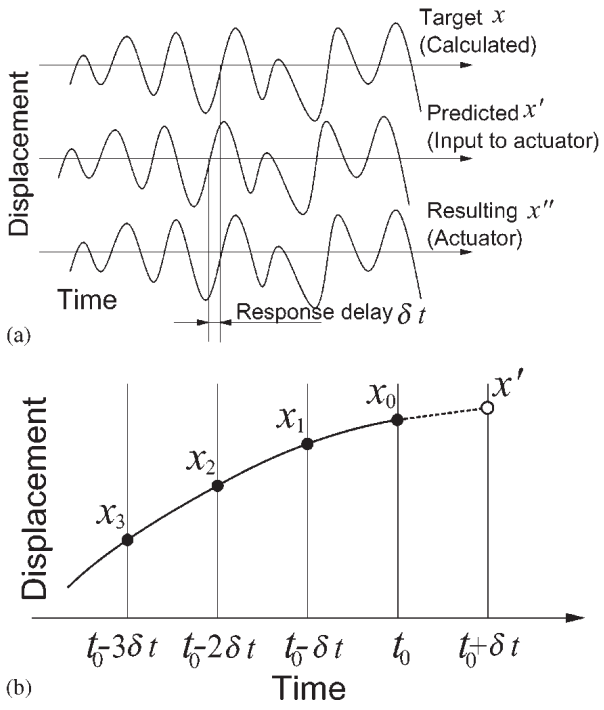


Figure 6.30: Time history of input and output signals for a hydraulic actuator [29]

Note that, this method is only applicable the problems with constant delay time. To remove this restriction, in [34], [67], and [70], various polynomial-based adaptive algorithms are proposed that tunes the prediction model coefficients online. However, these approaches have some problems as well. For example, low-order prediction models are insufficient to characterize the complex signals. On the other hand, higher-order prediction models are computationally inefficient for modifying online.

6.3 Proposed Approach: ARX-Based Variable Phase Reference Predictor Combined with Operational Controller

As mentioned before, our system has $0.040[sec]$ time delay originated from serial port communication interface and motor driver processing issues.

As understood from the related literature, the Smith predictor approach is a widely-used method for systems with known time delay. However, for the success of this method, the delay has to be long enough. For our system, the results obtained by using Smith predictor implementation is not satisfactory, due to the $0.040[sec]$ delay time, which corresponds 5 sampling time. It is required that the camera stabilization platform tracks signals, which have frequency of $5[Hz]$ and magnitude of $\pm 10[deg]$. So, polynomial based reference prediction algorithms are insufficient to characterize these complex signals.

For this purpose, an ARX-based variable phase reference prediction structure is prosed to operate with any type of controller as shown in Figure 6.31. In this structure, multiple ARX models are obtained by using system identification techniques to model a desired process based on measurements. During the operation, according to the desired and measured signals, the prediction step calculater computes that how many step ahead has to be predicted. Then by a simple switch, the related ARX model is selected to predict the future values of the reference signals. By using such a structure, the computational complexity is reduced, because the optimization or identification is performed offline. Also, by means of powerful modeling techniques, ARX, even complex signals can be predicted with great success. However, one major drawback of this approach is poor performance of the prediction step calculater with noisy signals.

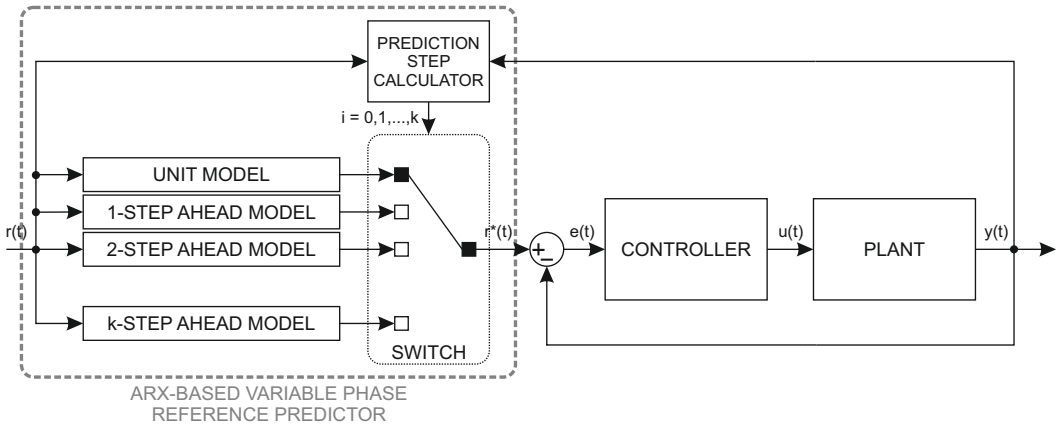


Figure 6.31: Block diagram of ARX-Based variable phase reference predictor

CHAPTER 7

EXPERIMENTAL STUDIES FOR CAMERA STABILIZATION PLATFORM

In Chapter 6, the model-free and model-based controllers are implemented for idealized and realistic cases and are subjected to the parameter optimization processes. Also, an ARX-based reference predictor is proposed to eliminate the phase lags, as well as communication and processing delays. In this chapter, simulation and physical experiments are performed by using controllers and predictors obtained in the previous chapter. In Section 7.1, the controllers implemented for idealized case are compared in terms of time-domain specifications, like rising time, settling time, maximum overshoot, etc. The ones, which are inadequate to control the idealized model or conduces to the poor control performance, are eliminated. In Section 7.2, the realistic model versions of the selected candidates are compared for a second time to choose the best possible structure for controlling the camera stabilization platform. In Section 7.3 and Section **??**, the tracking and stabilizing performances of the selected controller are examined with and without proposed ARX-based reference predictor.

7.1 Simulation Experiments-I: Pulse Responses of Idealized Model Controllers

In this section, three different simulation results are obtained for each controllers shown in Figure 7.1. At each simulation, top platform is desired to remain stationary in two axes of roll-yaw-pitch (RYP), while to track a pulse signal at the other axis. Thus, the tracking performance can be evaluated at the axis of movement, while disturbance rejection performance can be examined at the other axes.

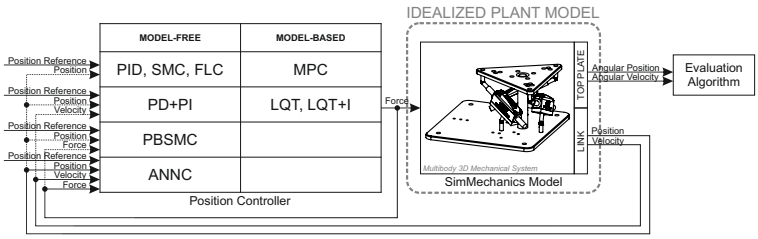


Figure 7.1: Idealized plant model and designed controller structures

7.1.1 Single-Loop PID Controller

As detailed in Section 6.1.1.1, the single-loop fixed-gain and variable-gain PID controllers are implemented for the idealized model of camera stabilization mechanism. For the fixed-gain PID controller, various tuning methods are applied to find an appropriate parameter set without any success. Additionally, these parameters are subjected to the optimization process, but obtained controller fails to satisfy the requirements. It can be said that without any doubt single-loop fixed-gain PID controller structure is insufficient to stabilize this mechanism. On the contrary, the single-loop variable-gain PID controller is able to stabilize the system, but the results are still not satisfactory as shown in Figure 7.2 to Figure 7.4.

At each graph, the major axes are arranged in the leftmost column, where the entire response of the system is plotted between $[0, 2.5]$ (sec). The remaining axes numbered between $1 - 7$, spans a specific range of the major axes to pick up a further thought about significant instants. For example, in Figure 7.2, the axes numbered by 1 and 3 show an insight about rising time, settling time and maximum overshoot of the transient response at the negative and positive movement directions, respectively. Additionally, the axes numbered by 2 gives an idea about steady-state error. Also, the axes numbered by 4, 5, 6 and 7 reveal the maximum deviation from zero position and disturbance rejection time. Note that, due to the poor control performances, the time domain specifications could not be listed for these compensators.

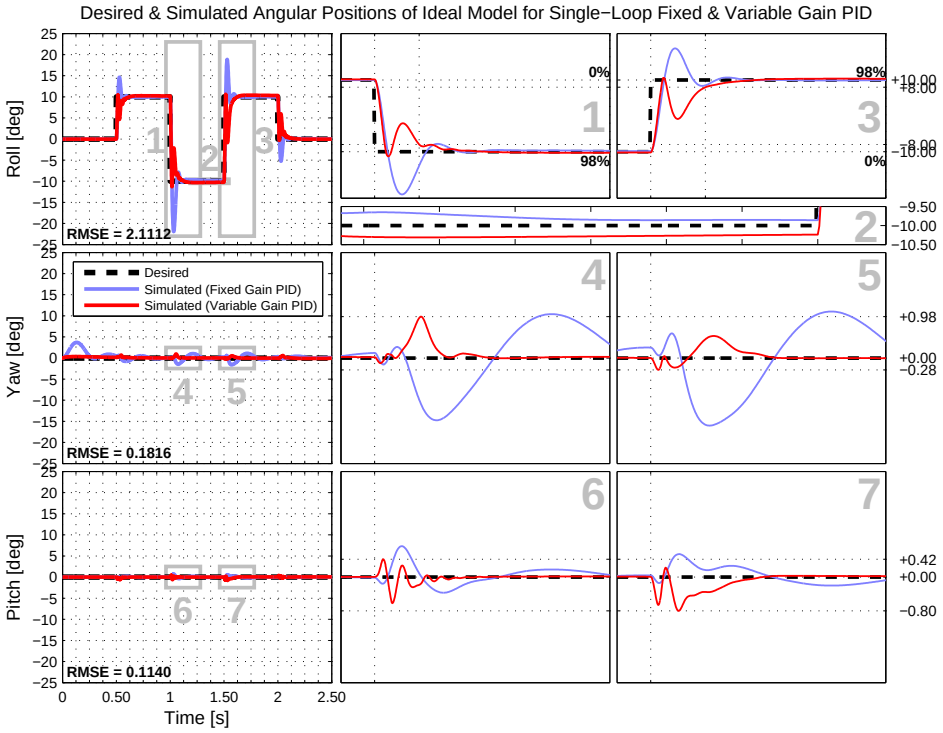


Figure 7.2: Fixed-gain and variable-gain single loop PID controllers simulated at roll axis

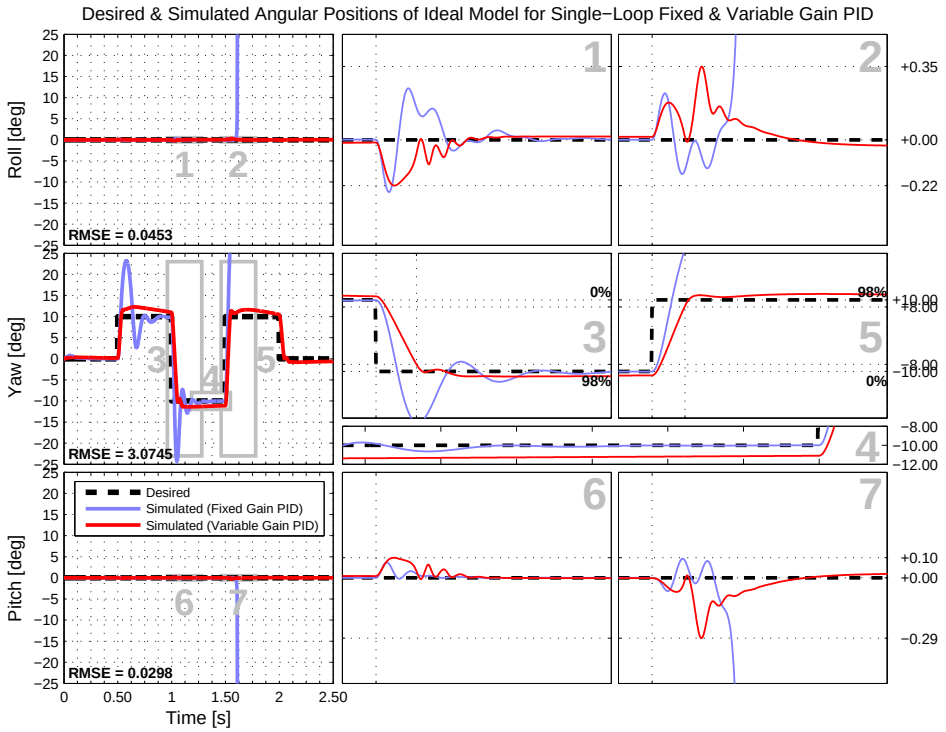


Figure 7.3: Fixed-gain and variable-gain single loop PID controllers simulated at yaw axis

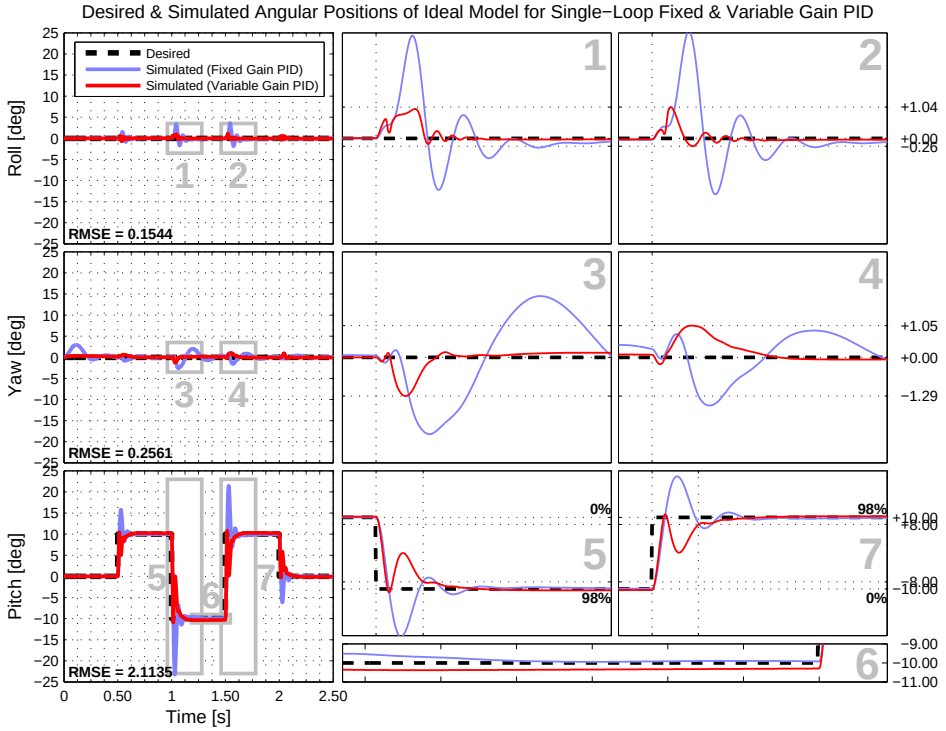


Figure 7.4: Fixed-gain and variable-gain single loop PID controllers simulated at pitch axis

7.1.2 Double-Loop PD+PI Controller

As detailed in Section 6.1.1.1, double-loop decoupled and coupled type of PD+PI controllers are implemented such that the PD structure that is operated as a position controller is placed at the outer loop, whereas, the PI structure that is operated as a velocity controller is placed at the inner loop. Note that, as a characteristic property of the mechanism that is independent from any controller performance, the roll and pitch axes dynamics are very similar to each others. However while doing a yaw axis movement, the center of gravity of the link swings to right and left which generate a self-disturbance due to link mass. The results as shown in Figure 7.5 to Figure 7.7 indicate that the performance of these controllers are very similar at roll and pitch axis movements. However, at yaw axis movement, decoupled PD+PI could not obviate the swinging, on the other hand, coupled PD+PI leads to perform a smoother movement. By extracting from these graphs, the time-domain performance specifications like, rising time, T_r, settling time, T_s, disturbance rejection time, T_{dr}, and maximum devation, M_d, are given for the double-loop coupled PD+PI controller as follows;

	T_r [sec]	T_s [sec]	T_{dr} [sec]	M_d [deg]
$Roll$	0.078	0.169	0.240	0.370
Yaw	0.066	0.165	0.160	0.170
$Pitch$	0.078	0.173	0.240	0.470

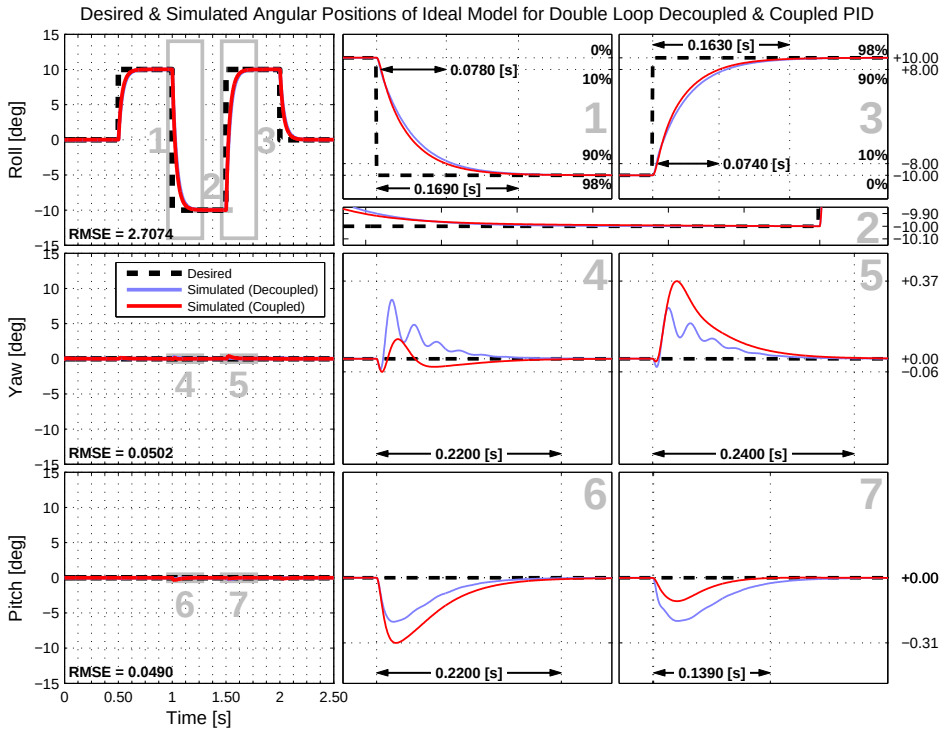


Figure 7.5: Double-loop decoupled and coupled PID controllers simulated at roll axis

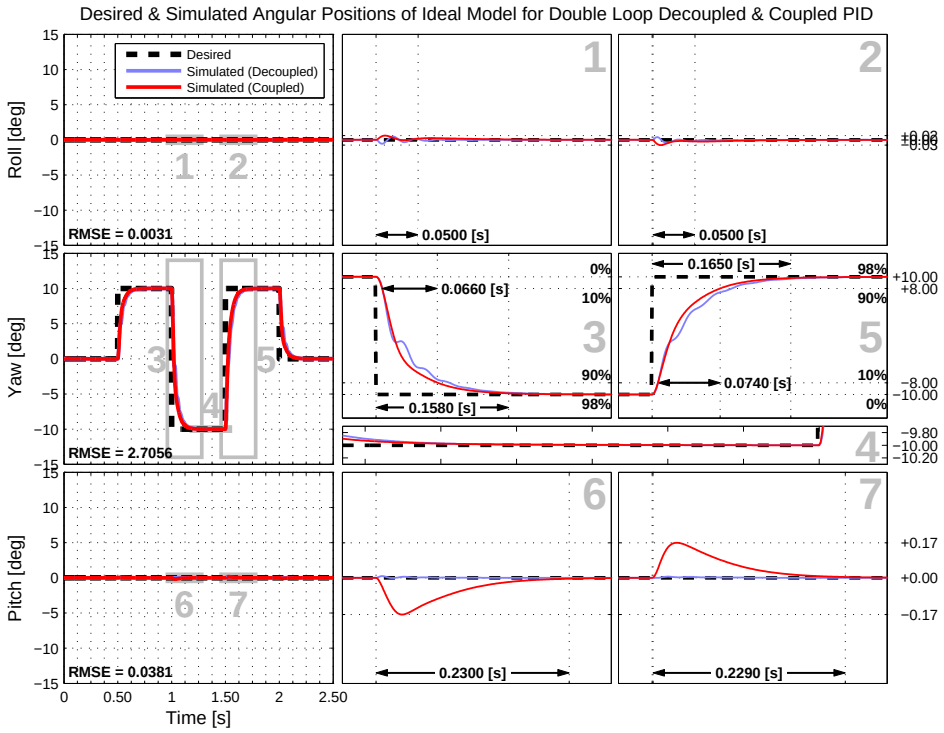


Figure 7.6: Double-loop decoupled and coupled PID controllers simulated at yaw axis

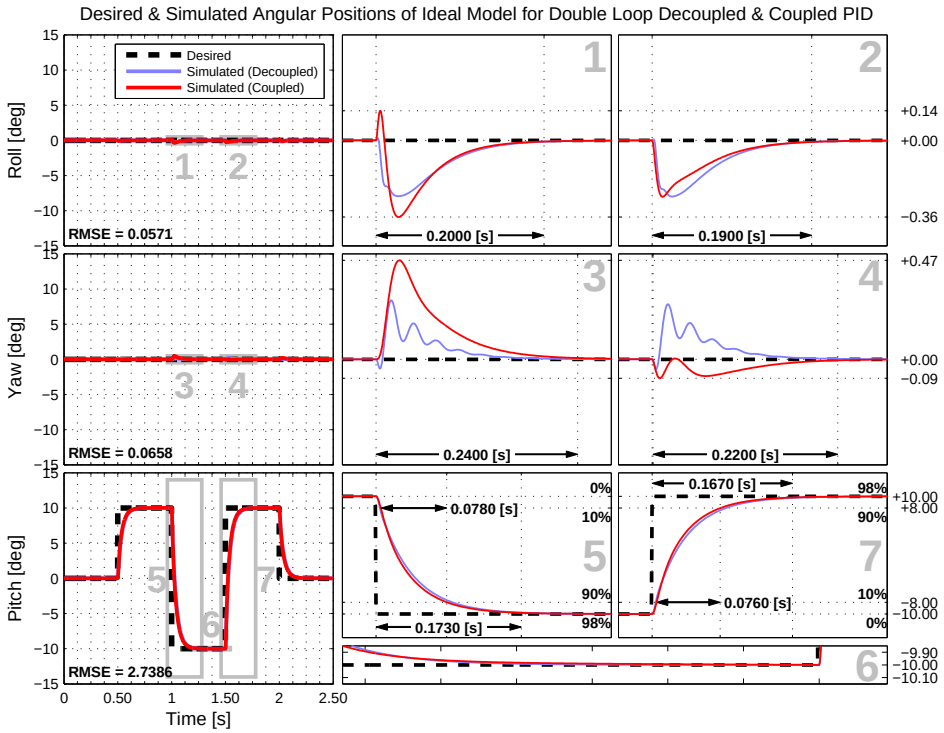


Figure 7.7: Double-loop decoupled and coupled PID controllers simulated at pitch axis

7.1.3 Sliding Mode Controller

As detailed in Section 6.1.1.2, a standard version of the first-order sliding mode controller is implemented by using the signum function. In addition, to prevent the chattering problem of the conventional sliding mode controller, the signum function is replaced by the saturation function. Moreover, the saturation function is subjected to the optimization process for tuning the slope of it. This replacement avoids the chattering problem, but cause to steady-state error. Actually, there is a trade-off between these two phenomenons. Decreasing slope of the saturation function increases the "width" of the sliding manifold, which reduce the chattering problem, but increase the steady-state error. The results as shown in Figure 7.8 to Figure 7.10 indicate that even in a simulation environment without any measurement noise, the standard version of the sliding mode controller that uses signum function suffers from chattering issues. This problem becomes more visible at the axes of the immobility, where the plotting range is smaller. It is obvious that, the modified sliding mode controller performs better than the conventional one. By extracting from these graphs, the time-domain performance specifications like, rising time, T_r, settling time, T_s, disturbance rejection time, T_{dr}, and maximum devation, M_d, are given for the sliding mode controller with saturation function as follows;

	T_r [sec]	T_s [sec]	T_{dr} [sec]	M_d [deg]
$Roll$	0.022	0.045	0.060	0.330
Yaw	0.021	0.045	0.050	0.170
$Pitch$	0.021	0.044	0.060	0.620

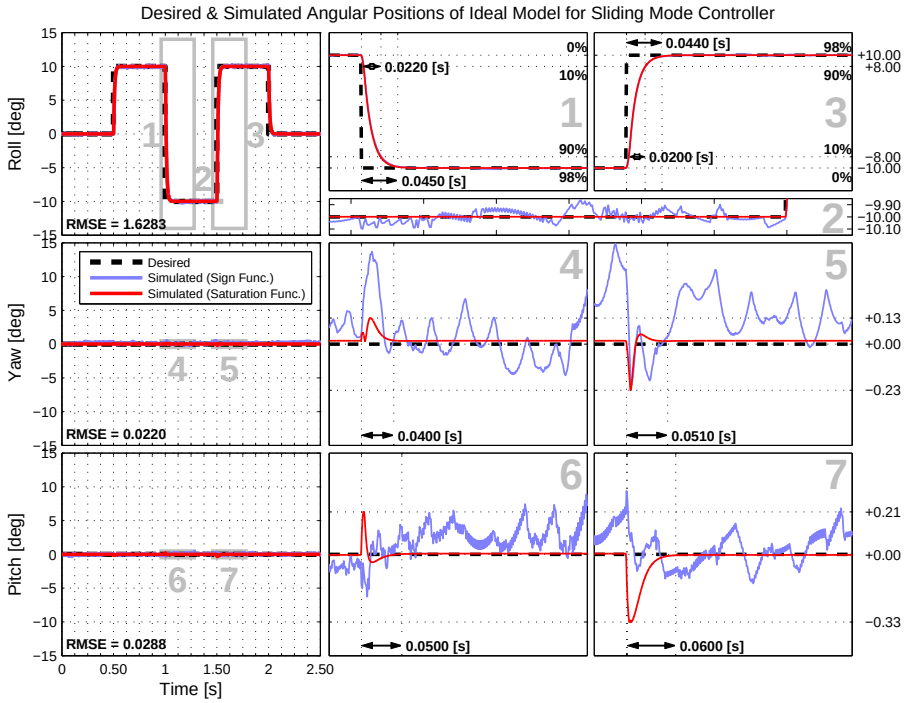


Figure 7.8: Sliding mode controller with sign & saturation functions simulated at roll axis

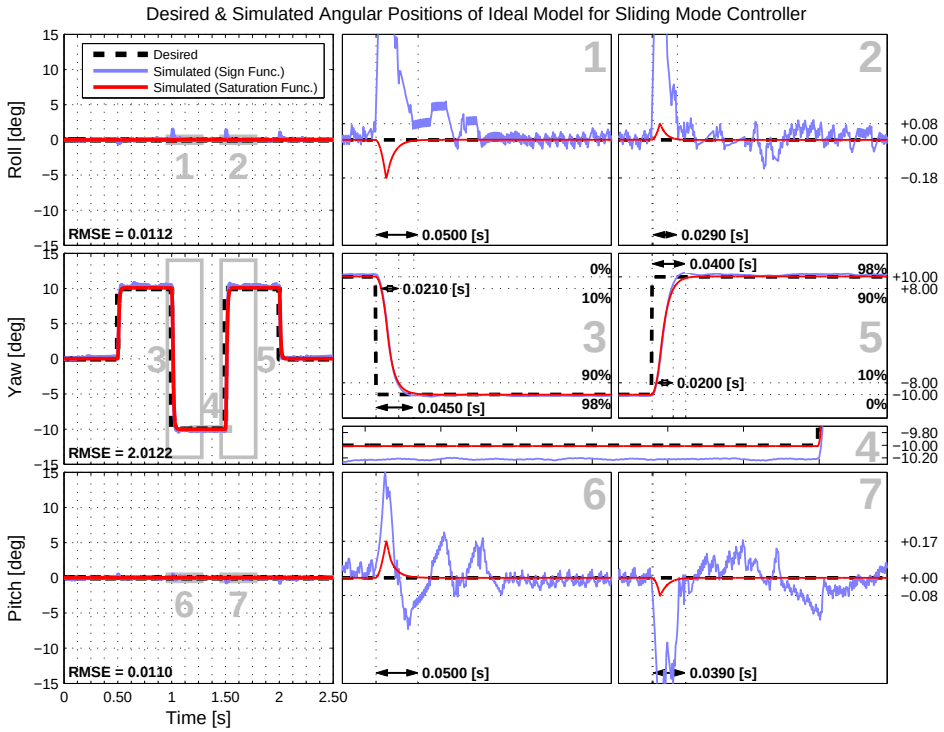


Figure 7.9: Sliding mode controller with sign & saturation functions simulated at yaw axis

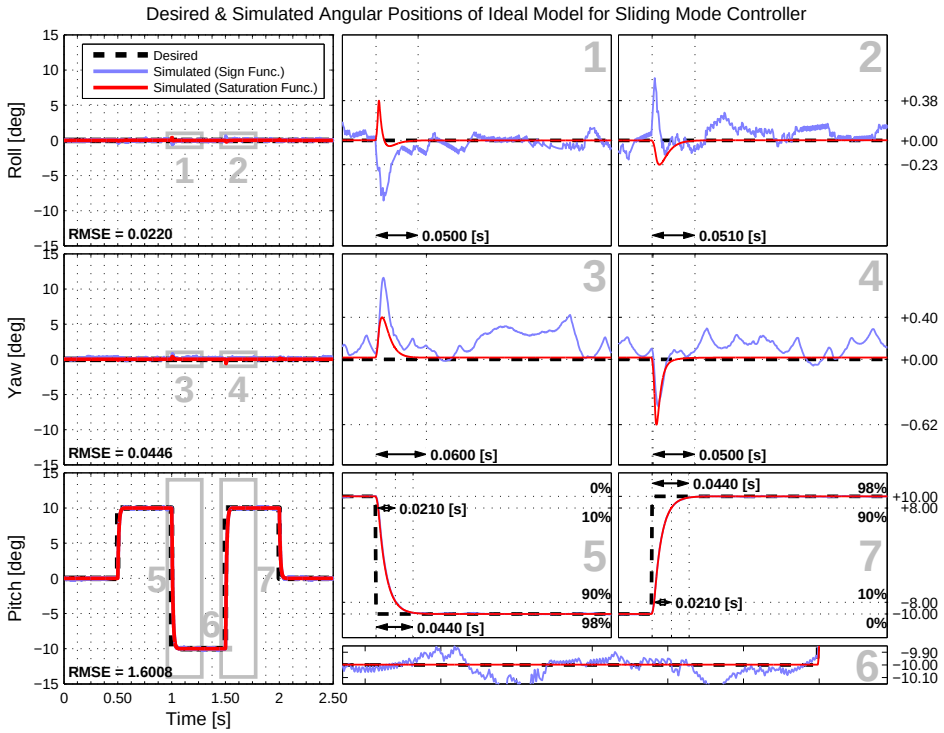


Figure 7.10: Sliding mode controller with sign & saturation functions simulated at pitch axis

7.1.4 Proxy-Based Sliding Mode Controller

As detailed in Section 6.1.1.3, the behaviour of the proxy-based sliding mode controller to small and large positional errors can be designed independently. In other words, it leads to the responsive tracking for small positional errors, while it avoids the excessive overshoots for large positional errors. At this method, the sliding mode controller acts on virtual mass, which is called proxy, and the actual plant is forced to track the proxy by the PID-type controllers. In related literature, the proxy mass is assumed to be zero for avoiding undesired delays and unresponsive dynamics. However, for zero mass chattering issues are observed in our application. So, in order to understand the effect of the proxy mass, and to use the proxy as a smoothing element, three different variants are implemented for $M_{proxy} = 0[kg]$, $M_{proxy} \approx 0.1[kg]$ and $M_{proxy} \approx 1[kg]$. The results as shown in Figure 7.11 to Figure 7.13 reveal that the system dynamics become more and more unresponsive by the increasing amount of the proxy mass. However, this attitude leads to more robust and imperturbable dynamics especially at the axes of immobility and improves the disturbance rejection and maximum deviation values. By extracting from these graphs, the time-domain performance specifications like, rising time, T_r, settling time, T_s, disturbance rejection time, T_{dr}, and maximum devation, M_d, are given for the proxy-based sliding mode controller with zero proxy mass as follows;

	T_r [sec]	T_s [sec]	T_{dr} [sec]	M_d [deg]
$Roll$	0.030	0.074	0.097	0.480
Yaw	0.027	0.080	0.074	1.080
$Pitch$	0.030	0.073	0.075	0.830

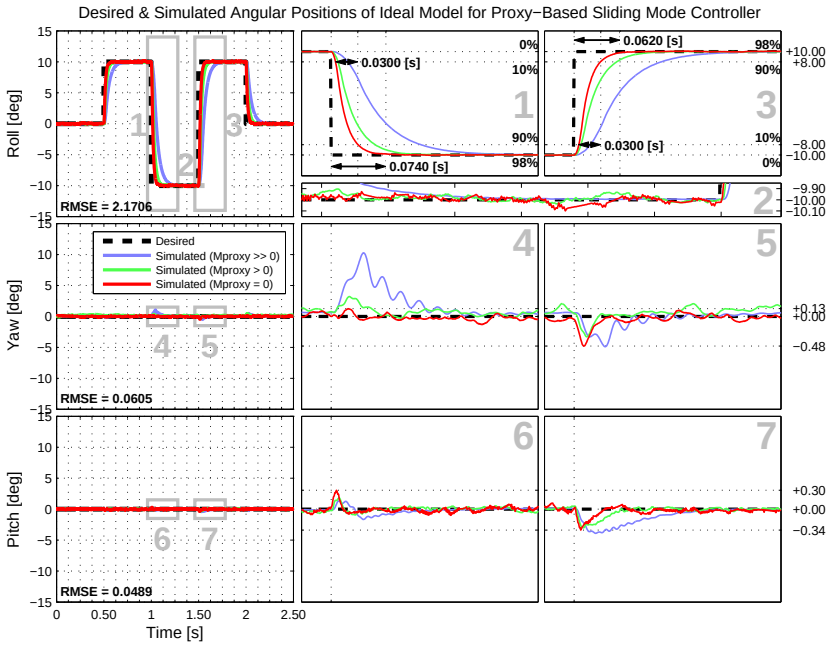


Figure 7.11: Proxy-based sliding mode controller for different masses simulated at roll axis

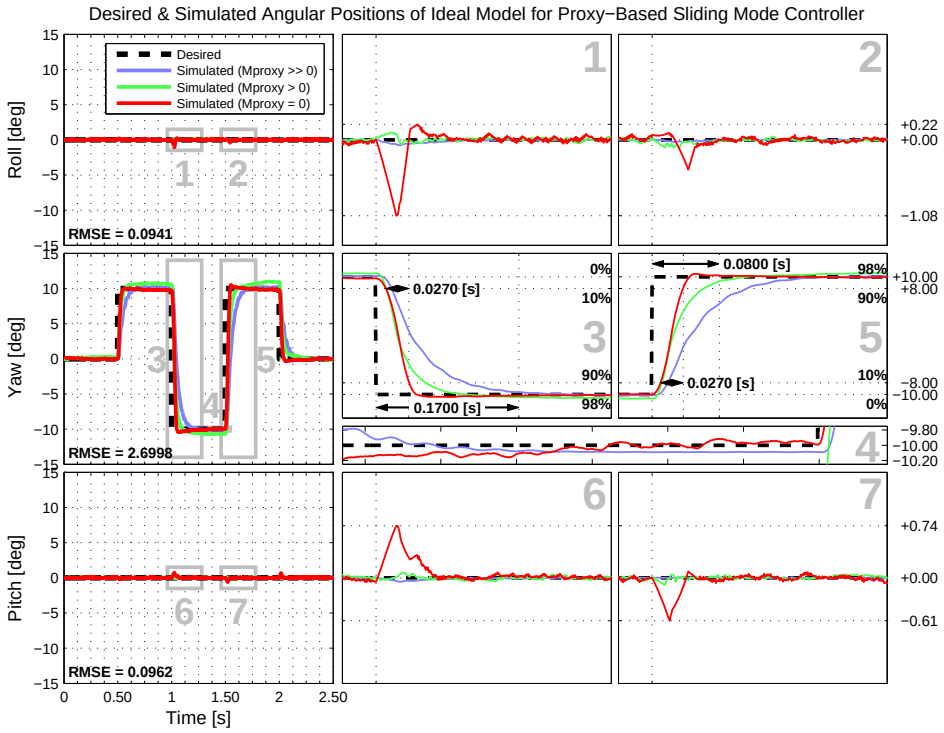


Figure 7.12: Proxy-based sliding mode controller for different masses simulated at yaw axis

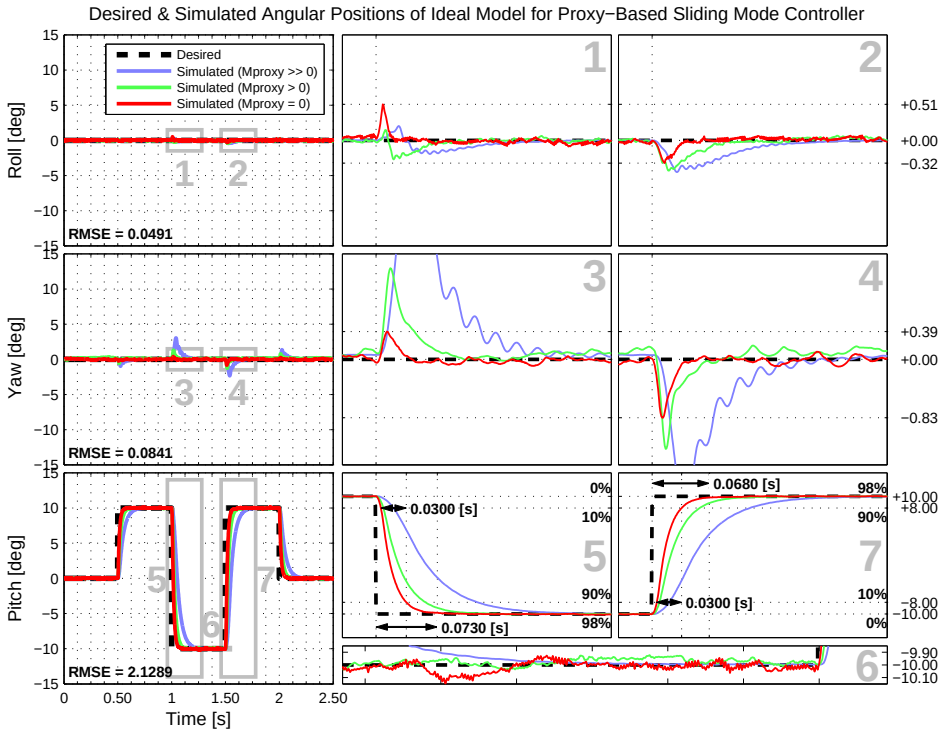


Figure 7.13: Proxy-based sliding mode controller for different masses simulated at pitch axis

7.1.5 Fuzzy Logic Controller

As detailed in Section 6.1.1.4, a fuzzy logic controller is implemented for idealized model that accepts position error and derivative of position error as inputs and generates link forces as outputs. The fuzzification and defuzzification constants, as well as triangular shaped membership functions (MF) are optimized by using genetic algorithm routines. For comparison purposes, as shown in Figure 7.14 to Figure 7.16, the results are obtained for two different controllers, which uses 5 and 7 number of membership functions for each inputs and outputs. Due to the mechanical constraints and hard stops of the mechanism, as well as concerns about degradation of vision data originated from rapid changes in the platform velocity, our main purpose is to design the controllers lead to the overdamped system. However, this objective can not be achieved by using fuzzy logic controllers. The computational complexity of the membership function optimization is very high, even for the relatively small number of rules. For that matter, the fuzzy logic controllers are implemented with 9 and 11 number of membership functions, but the optimization process can not be finalized within a reasonable period of time. By extracting from these graphs, the time-domain performance specifications like, rising time, T_r, settling time, T_s, disturbance rejection time, T_{dr}, and maximum devation, M_d, are given for the fuzzy logic controller with 7 number of membership functions as follows;

	T_r [sec]	T_s [sec]	T_{dr} [sec]	M_d [deg]
$Roll$	0.020	0.052	0.135	1.900
Yaw	0.029	0.131	0.130	0.320
$Pitch$	0.020	0.051	0.130	1.830

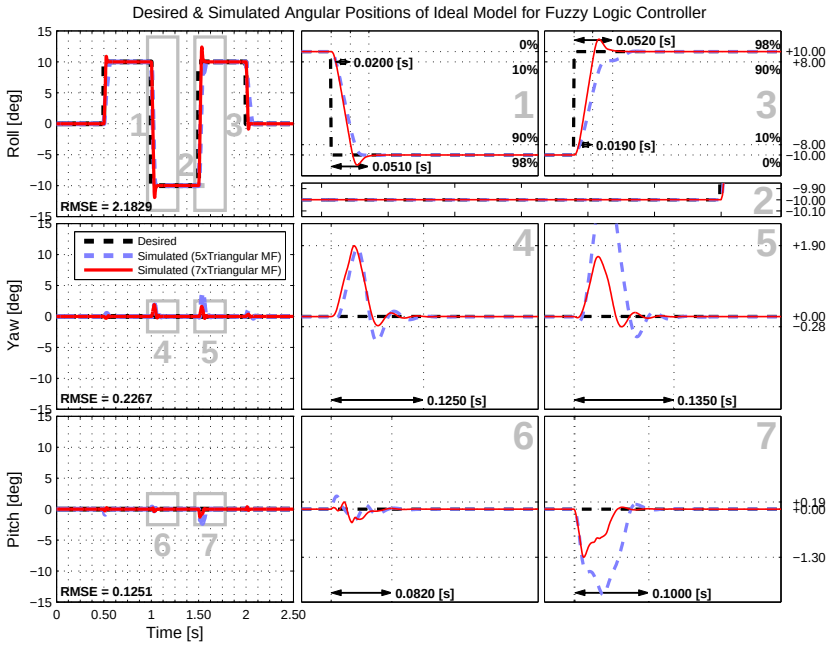


Figure 7.14: Fuzzy logic controller for two different numbers of MF simulated at roll axis

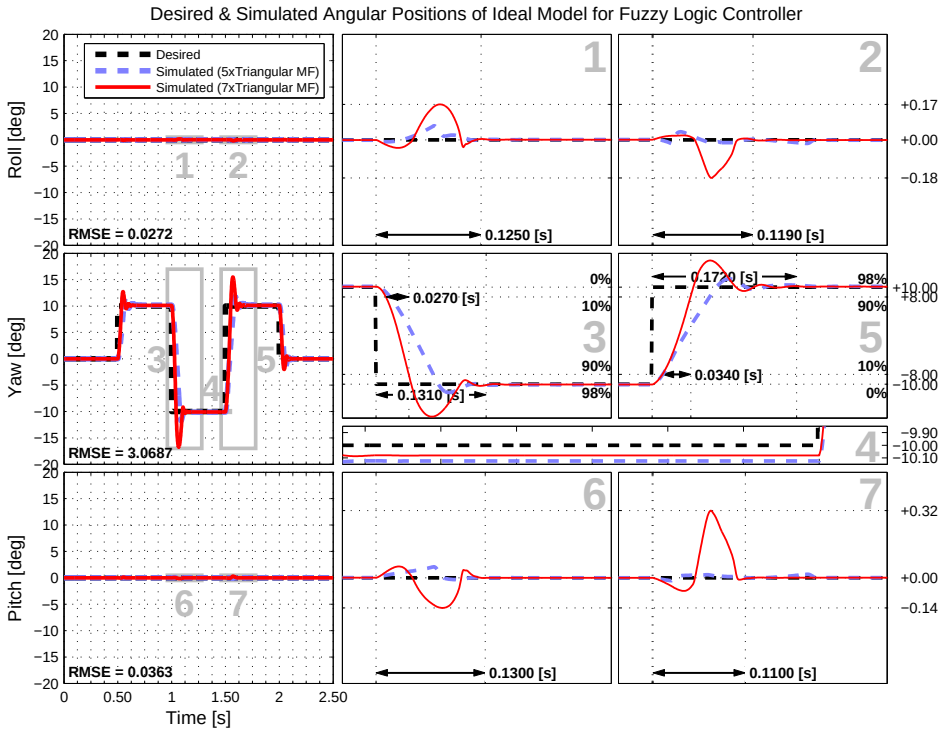


Figure 7.15: Fuzzy logic controller for two different numbers of MF simulated at yaw axis

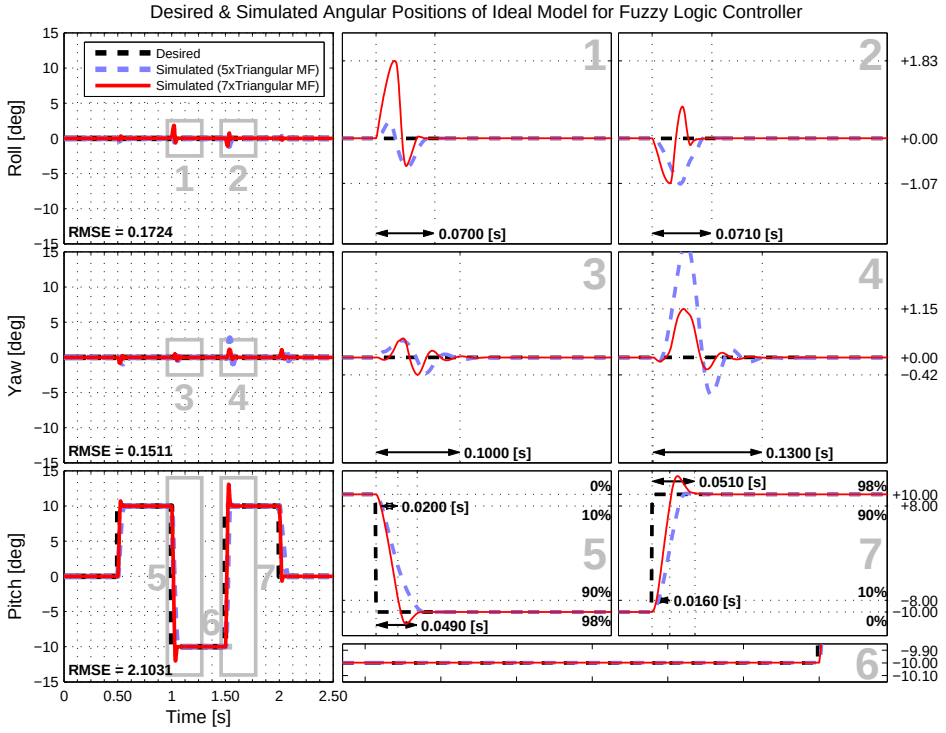


Figure 7.16: Fuzzy logic controller for two different numbers of MF simulated at pitch axis

7.1.6 Artificial Neural Network Controller

The training process of a artificial neural network are generally accomplished by using back-propogation algorithm. However in this study, as detailed in Section 6.1.1.5, only the network architecture is borrowed whereas the weights are optimized directly by using genetic algorithm instead of conventional training processes. For comparative purposes, two different networks are implemented to control the idealized model. They have two hidden layers with tangent hyperbolic activation function and one output layer with linear activation function. The difference between them is the number of hidden neurons. The first implementation has a $10x6$ formation, which can be interpreted as there are 10 and 6 hidden neurons at the hidden layer-1 and hidden layer-2, respectively. In similar fashion, the second implementation has a $15x10$ formation. For sure, to expand the network up to a certain extent improves the ability of mapping the inputs to the outputs. However, due to the computational complexity, the optimization process can not be finalized within a reasonable period of time for greater networks. As shown in Figure 7.17 to Figure 7.19, the results indicate that the implemented and optimized compensators are inadequate to control the idealized model and fail to satisfy the requirements. By extracting from these graphs, the time-domain performance specifications like, rising time, T_r, settling time, T_s, disturbance rejection time, T_{dr}, and maximum devation, M_d, are given for the neural network controller with $15x10$ formation as follows;

	T_r [sec]	T_s [sec]	T_{dr} [sec]	M_d [deg]
$Roll$	0.033	0.244	NA	1.940
Yaw	NA	NA	NA	NA
$Pitch$	0.038	0.280	NA	2.120

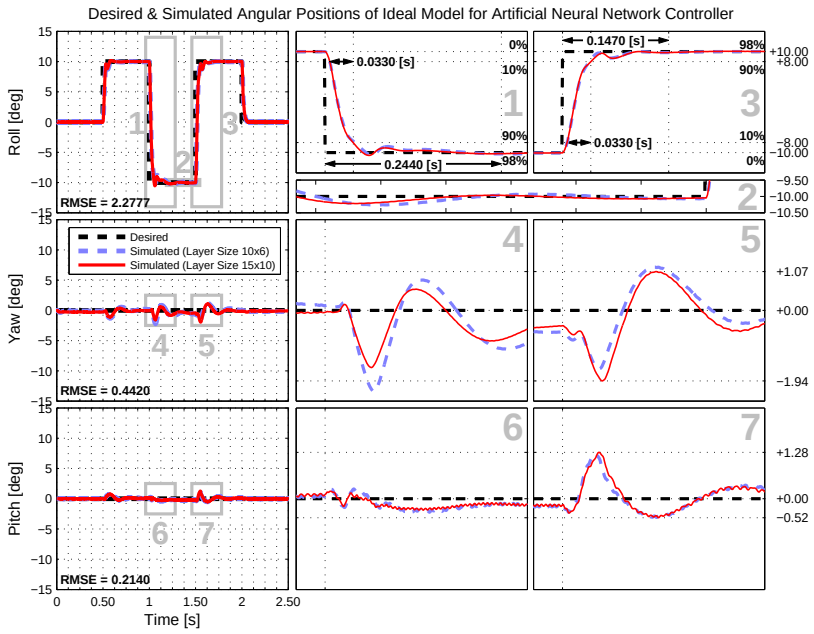


Figure 7.17: Artificial neural network controller simulated at roll axis

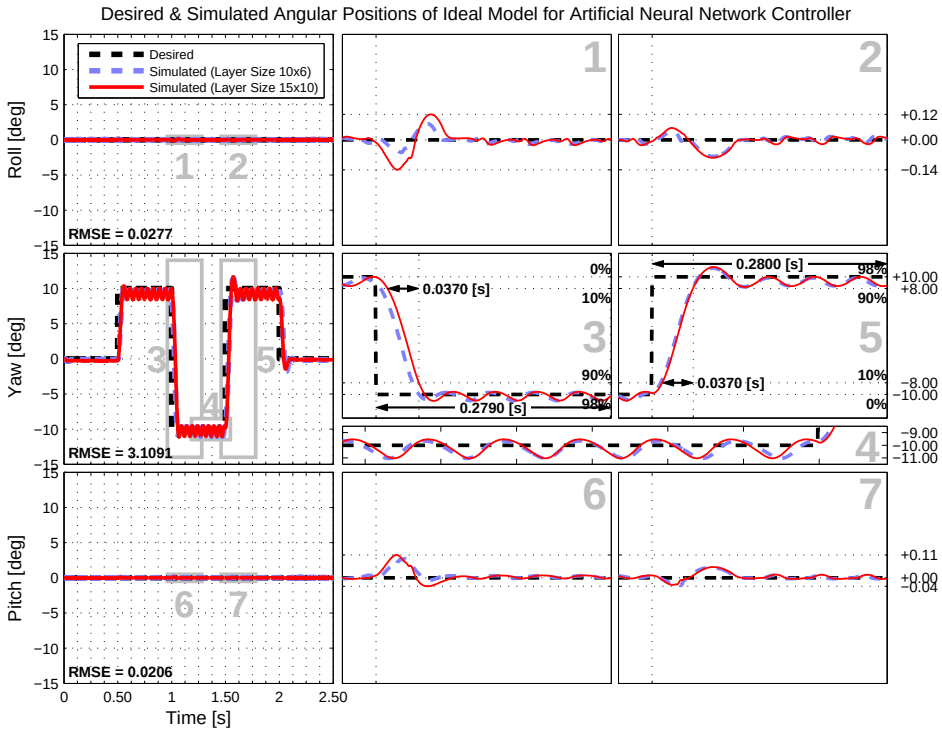


Figure 7.18: Artificial neural network controller simulated at yaw axis

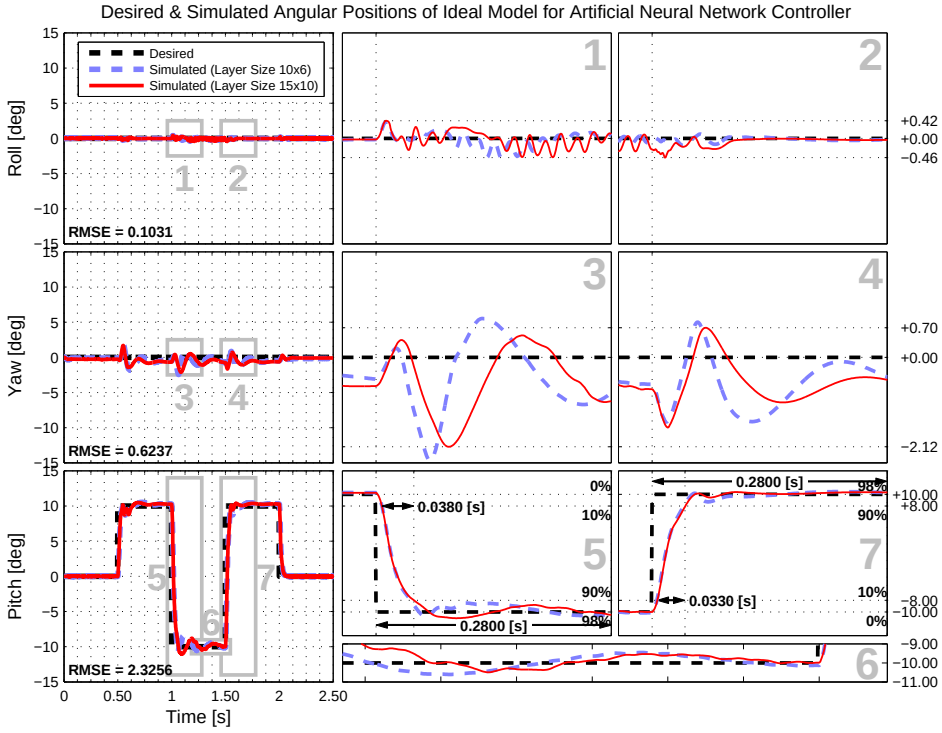


Figure 7.19: Artificial neural network controller simulated at pitch axis

7.1.7 Linear Quadratic Tracker

As detailed in Section 6.1.2.1, the linear quadratic tracking is a variation of the linear quadratic regulator problem which provides a way to follow a reference trajectory. However, in some applications where the model inaccuracies exist originated from unknowns or approximations, the desired results may not be obtained by using LQT. In such situations, this approach is modified by incorporating the integral action. For comparative purposes, linear quadratic tracker and it's variant augmented with integral action are implemented and subjected to the optimization process. As shown in Figure 7.20 to Figure 7.22, the results indicate that the standard version of the linear quadratic tracker is inadequate to control the idealized model and fails to satisfy the requirements. On the contrary, the modified version augmented with integral action performs well. However, remember that all of the state variables are assumed to be measurable without any measurement noise. So, it is important to reveal that how the control performance is affected from the unmeasurable states. By extracting from these graphs, the time-domain performance specifications like, rising time, T_r, settling time, T_s, disturbance rejection time, T_{dr}, and maximum devation, M_d, are given for the linear quadratic tracker with integral action as follows;

	T_r [sec]	T_s [sec]	T_{dr} [sec]	M_d [deg]
$Roll$	0.020	0.041	0.071	0.240
Yaw	0.024	0.077	0.050	0.060
$Pitch$	0.020	0.041	0.075	0.312

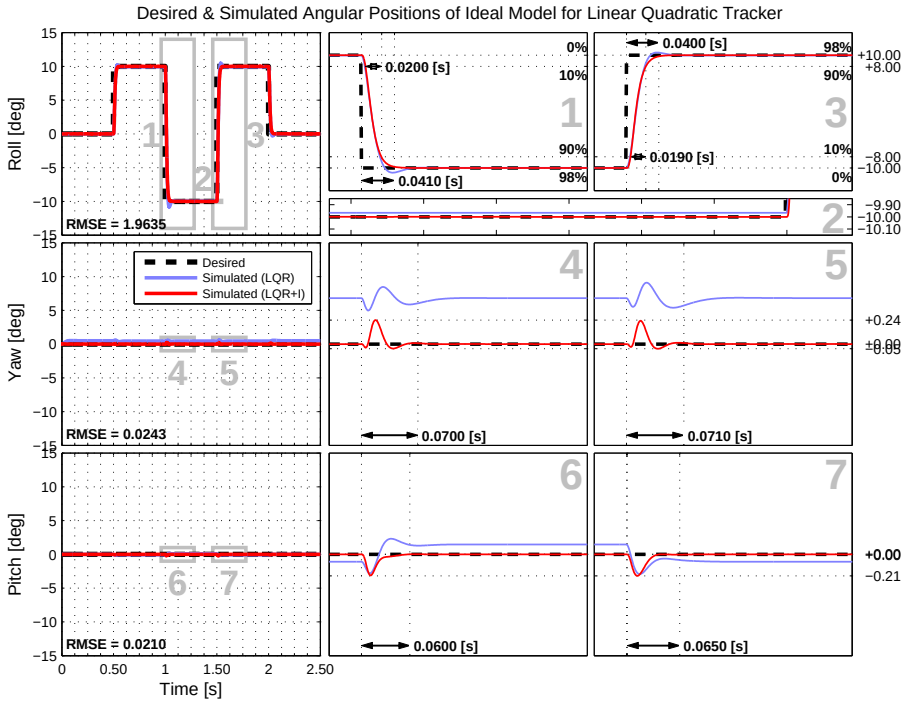


Figure 7.20: Linear quadratic tracker without and with integral action simulated at roll axis

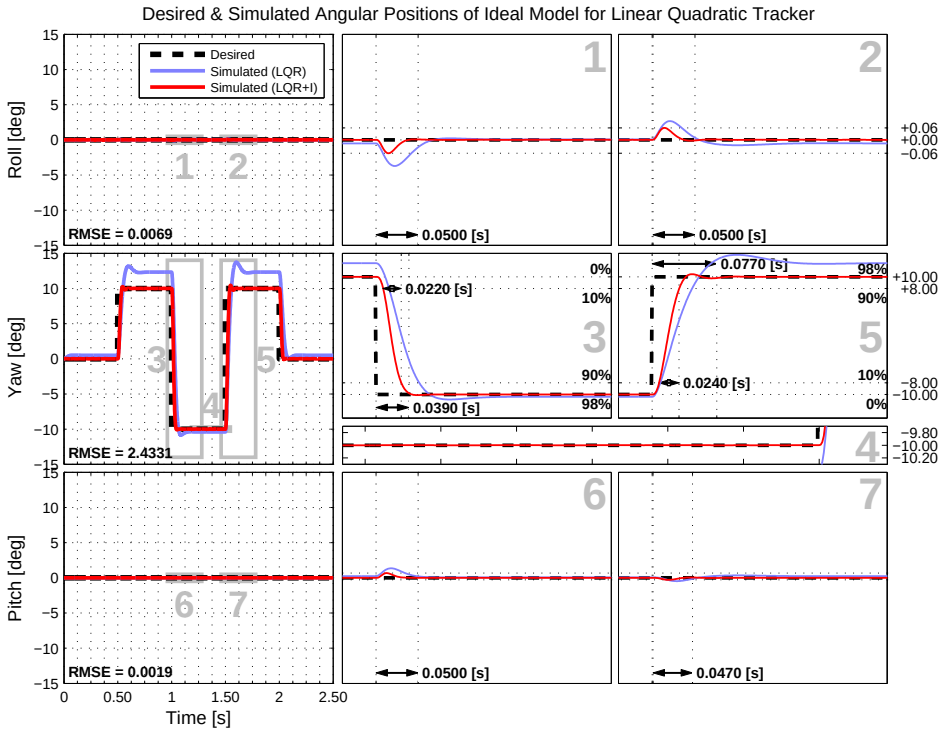


Figure 7.21: Linear quadratic tracker without and with integral action simulated at yaw axis

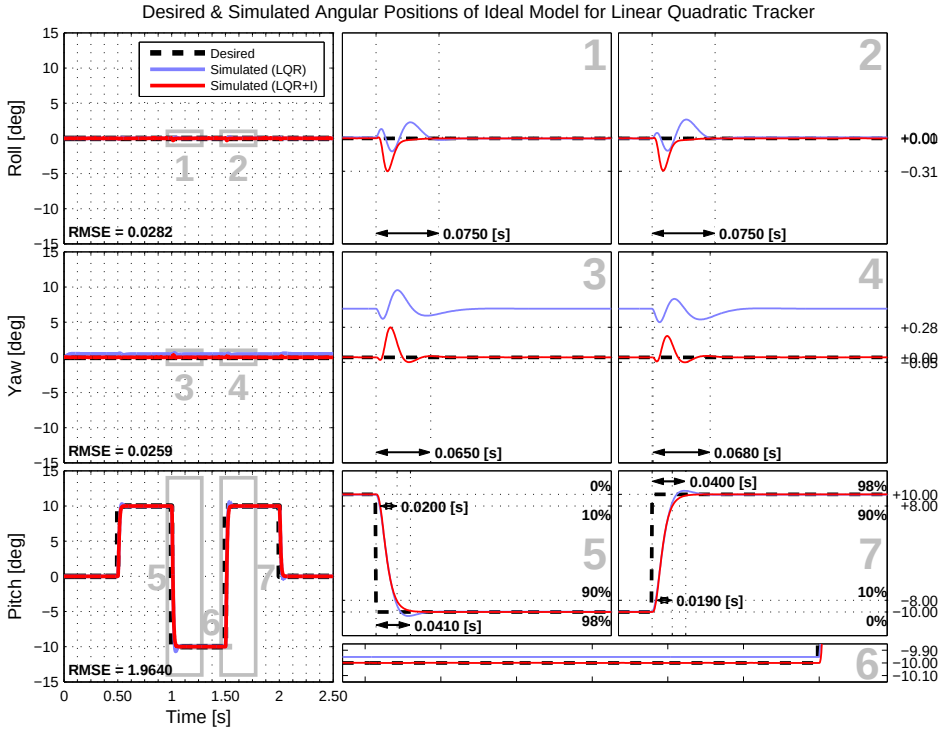


Figure 7.22: Linear quadratic tracker without and with integral action simulated at pitch axis

7.1.8 Model Predictive Controller

As detailed in Section 6.1.2.2, model predictive control is an optimization based multivariable strategy that attempts to find an optimal control sequence in discrete-time domain by taking into account input and output constraints of the plant. This type of controller is implemented by using Model Predictive Control Toolbox routines of MATLAB / Simulink environment for idealized model.

As shown in Figure 7.23 to Figure 7.25, the results are satisfactory, but it is obvious that the model predictive controller is an underperformer according to the linear quadratic tracker with integral action. By extracting from these graphs, the time-domain performance specifications like, rising time, T_r, settling time, T_s, disturbance rejection time, T_{dr}, and maximum devation, M_d, are given for the model predictive controller as follows;

	T_r [sec]	T_s [sec]	T_{dr} [sec]	M_d [deg]
$Roll$	0.056	0.118	0.175	0.390
Yaw	0.041	0.087	0.220	0.090
$Pitch$	0.055	0.116	0.200	0.470

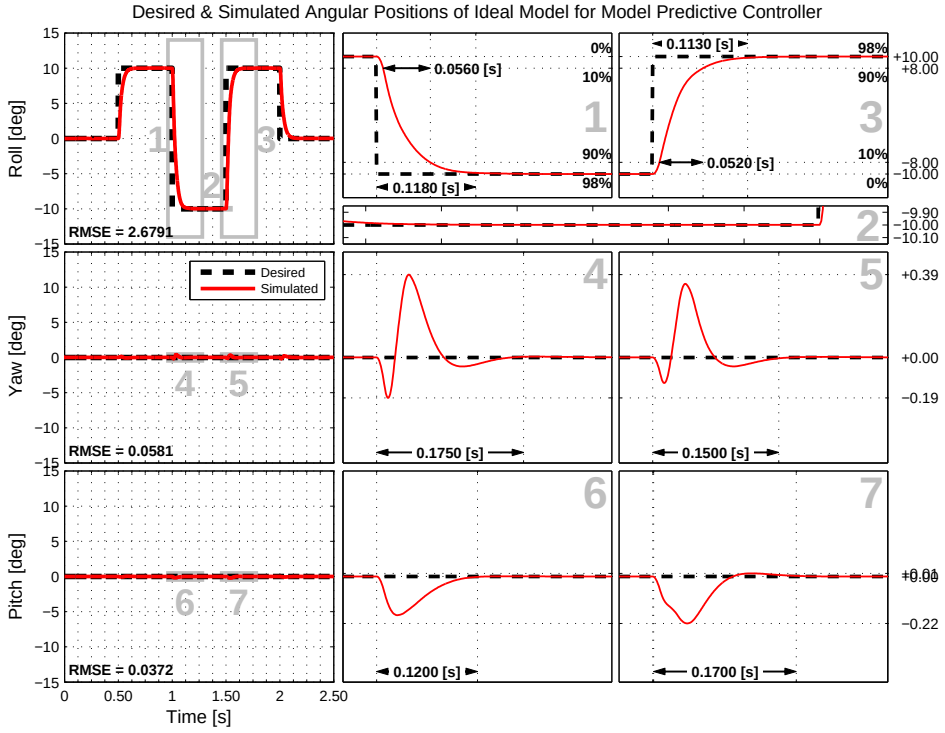


Figure 7.23: Model predictive controller simulated at roll axis

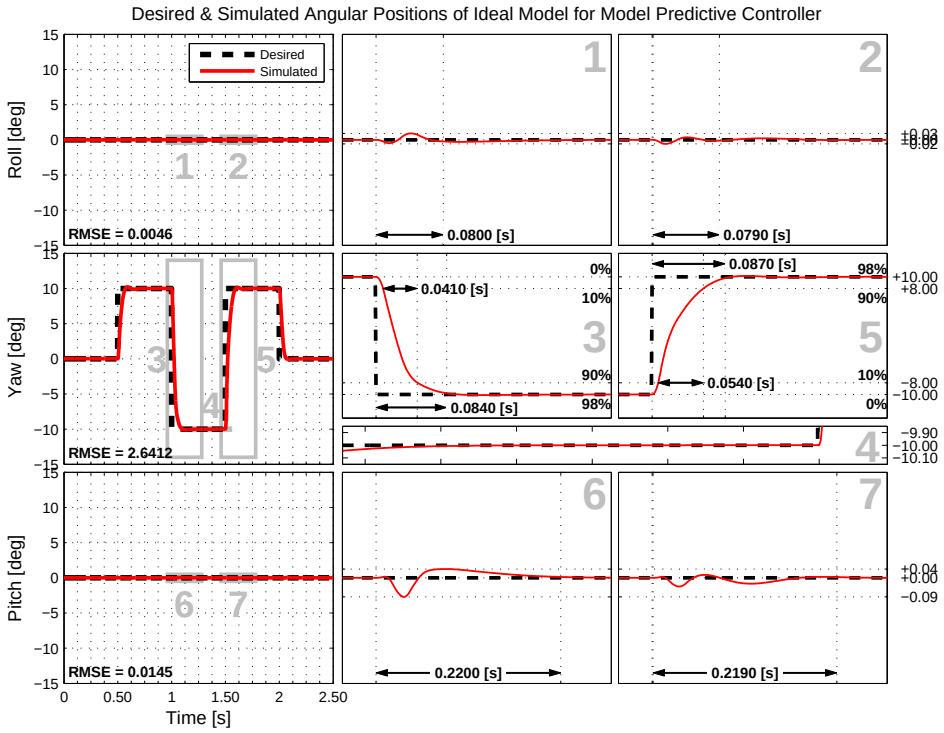


Figure 7.24: Model predictive controller simulated at yaw axis

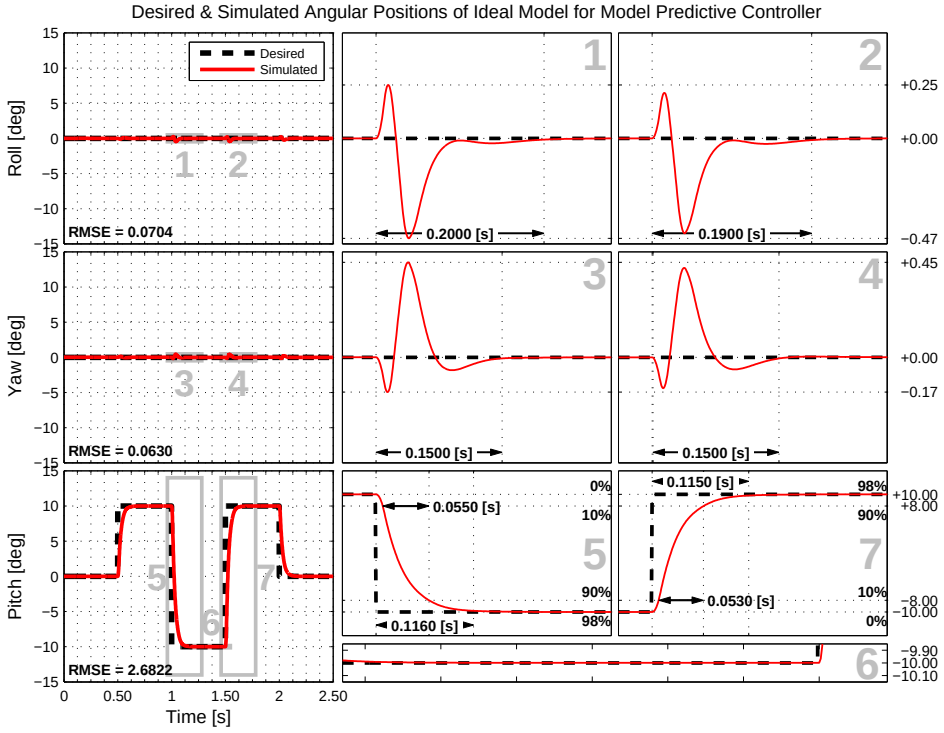


Figure 7.25: Model predictive controller simulated at pitch axis

7.1.9 Summary

Table7.1: Time-domain performance specifications obtained by using various controller structures implemented for idealized model of camera stabilization platform

	CONTROLLER TYPE *	AXIS OF MOVEMENT														
		ROLL					YAW					PITCH				
		T_r	T_s	PO	T_{dr}	M_d	T_r	T_s	PO	T_{dr}	M_d	T_r	T_s	PO	T_{dr}	M_d
MODEL FREE	**SL.FG.PID**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	SL.VG.PID	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
	DL.D.PD+PI	0.092	0.190	NA	0.243	0.279	0.088	0.201	NA	0.163	0.012	0.080	0.176	NA	0.243	**0.291**
	DL.C.PD+PI	0.078	0.169	NA	0.240	0.370	0.066	0.165	NA	0.160	0.170	0.078	0.173	NA	0.240	0.470
	SMC	**0.022**	0.045	NA	**0.060**	0.330	**0.021**	**0.045**	NA	**0.050**	0.170	0.021	0.044	NA	**0.060**	0.620
	PBSMC	0.030	0.074	NA	0.097	0.480	0.027	0.080	1.0	0.074	1.080	0.030	0.073	NA	0.075	0.830
	FLC	0.020	0.052	2.0	0.135	1.900	0.029	0.131	8.5	0.130	0.320	0.020	0.051	6.2	0.130	1.830
	ANNC	0.033	0.244	2.5	NA	1.940	NA	NA	NA	NA	NA	0.038	0.280	7.4	NA	2.120
MODEL BASED	**LQT**	0.023	0.070	4.7	NA	NA	NA	NA	NA	NA	NA	0.023	0.087	3.8	NA	0.628
	LQT+I	0.020	**0.041**	NA	0.071	**0.240**	0.024	0.077	1.0	**0.050**	**0.060**	**0.020**	**0.041**	NA	0.075	0.312
	MPC	0.056	0.118	NA	0.175	0.390	0.041	0.087	NA	0.220	0.090	0.055	0.116	NA	0.200	0.470

SL.FG.PID : Single-Loop Fixed-Gain Proportional-Integral-Derivative
SL.VG.PID : Single-Loop Variable-Gain Proportional-Integral-Derivative
DL.D.PD+PI : Doble-Loop Decoupled Proportional-Derivative + Proportional-Integral
DL.C.PD+PI : Doble-Loop Coupled Proportional-Derivative + Proportional-Integral
SMC : Sliding Mode Controller
PBSMC : Proxy-Based Sliding Mode Controller
FLC : Fuzzy Logic Controller
ANNC : Artificial Neural Network Controller
LQT : Linear Quadratic Tracker
LQT+I : Linear Quadratic Tracker with Integral Action
MPC : Model Predictive Controller

7.2 Simulation Experiments-II: Pulse Responses of Realistic Model Controllers

In this section, three different simulation results are obtained for each controllers shown in Figure 7.26. At each simulation, top platform is desired to remain stationary in two axes of roll-yaw-pitch (RYP), while to track a pulse signal at the other axis. Thus, the tracking performance can be evaluated at the axis of movement, while disturbance rejection performance can be examined at the other axes. Note that, for the realistic model case the PI-type velocity controller is operational on the motor driver, so with the position controller implementation, they necessarily constitutes double-loop formation.

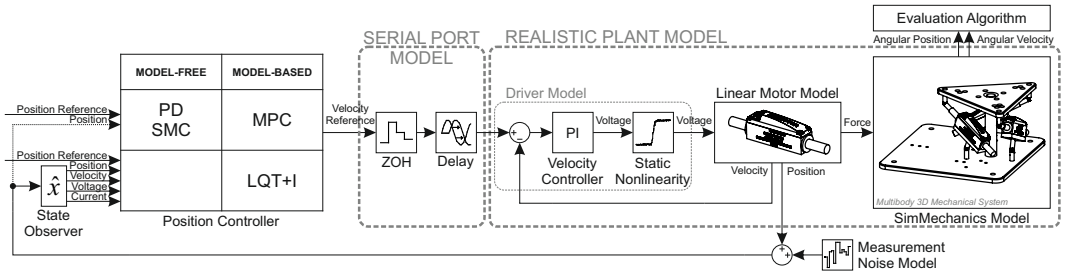


Figure 7.26: Realistic plant model and designed controller structures

7.2.1 Double-Loop PD+PI Controller

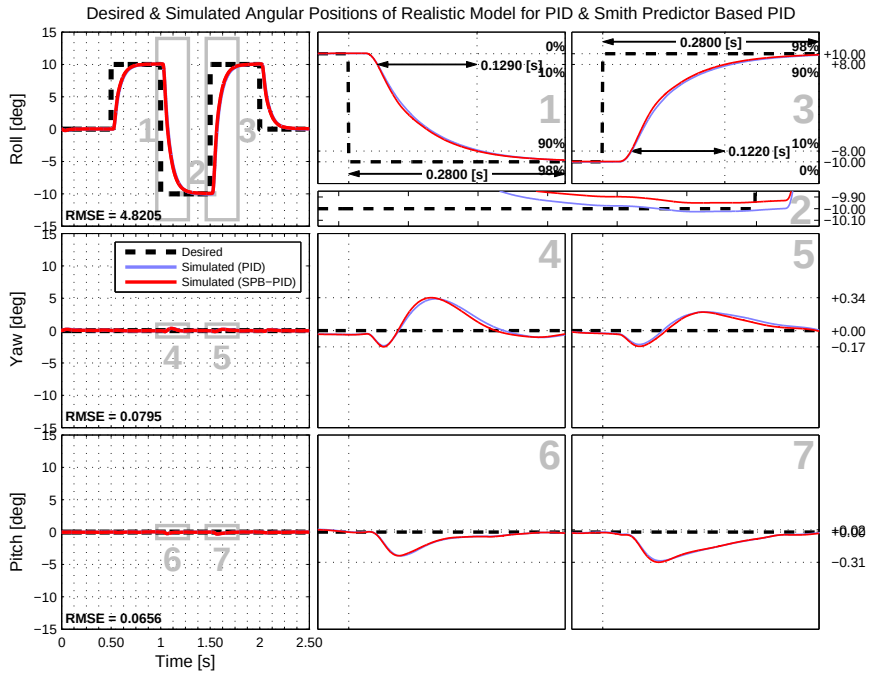


Figure 7.27: Proportional integral derivative controller & Smith predictor based variant simulated at roll axis

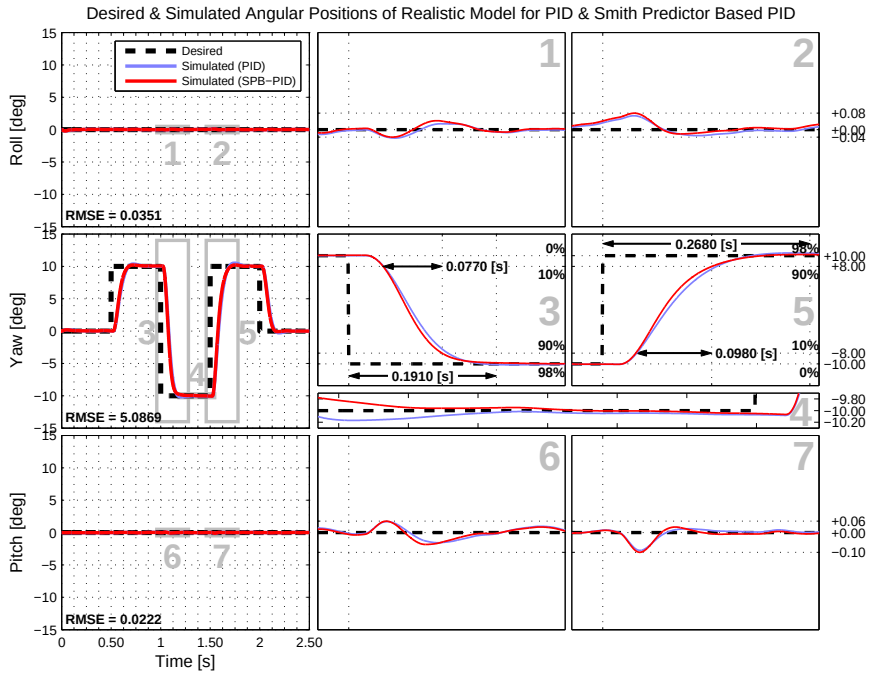


Figure 7.28: Proportional integral derivative controller & Smith predictor based variant simulated at yaw axis

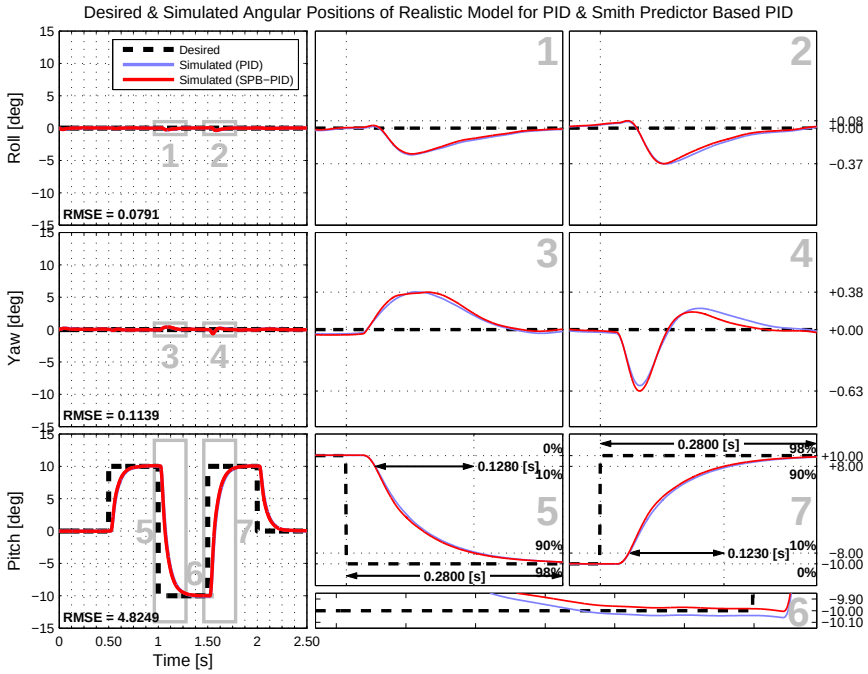


Figure 7.29: Proportional integral derivative controller & Smith predictor based variant simulated at pitch axis

7.2.2 Double-Loop SMC+PI Controller

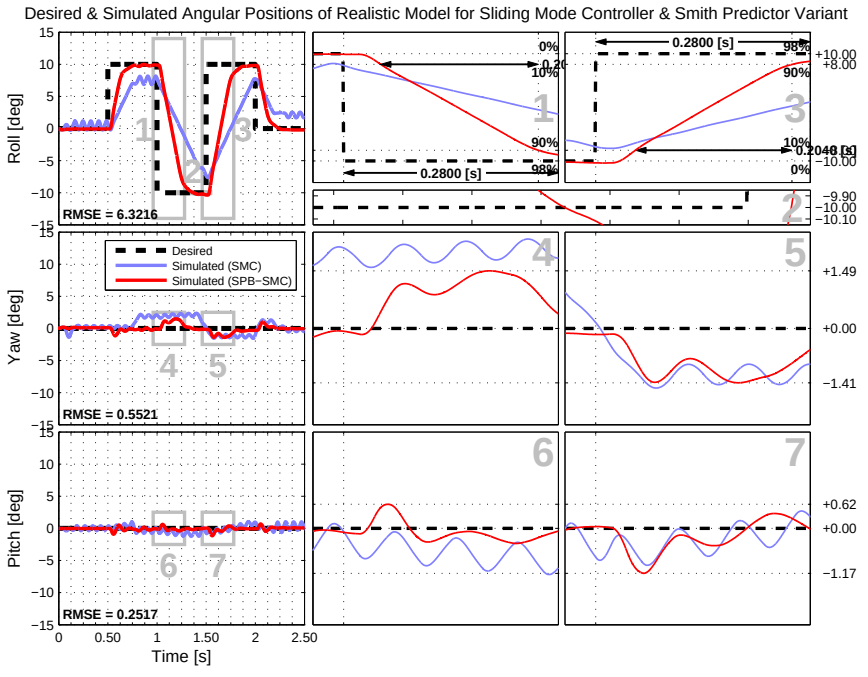


Figure 7.30: Sliding mode controller & Smith-predictor based variant simulated at roll axis

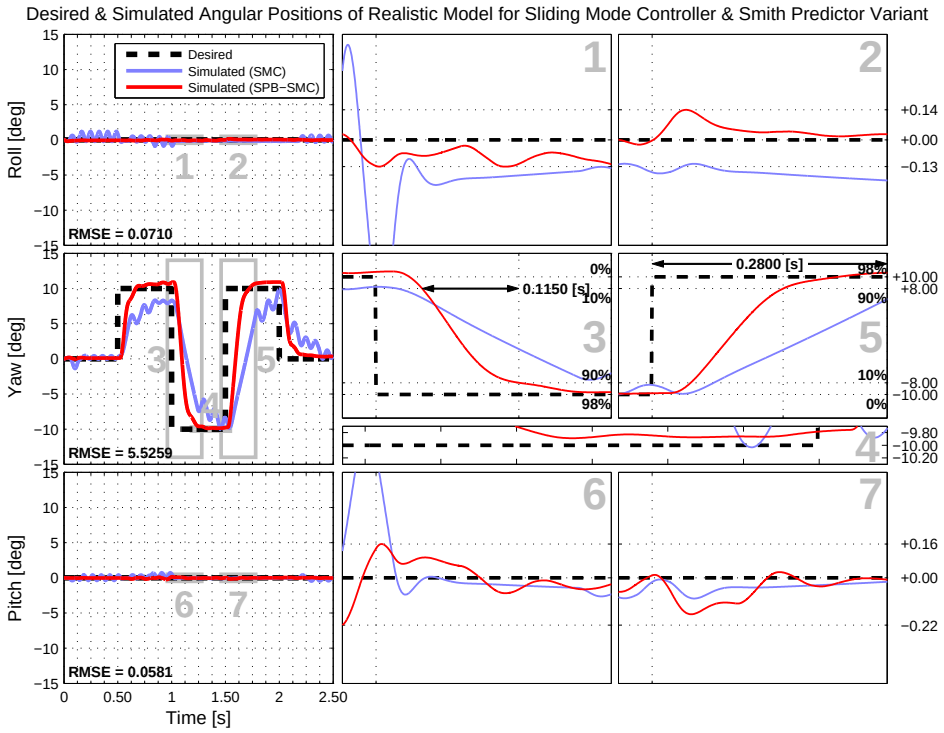


Figure 7.31: Sliding mode controller & Smith-predictor based variant simulated at yaw axis

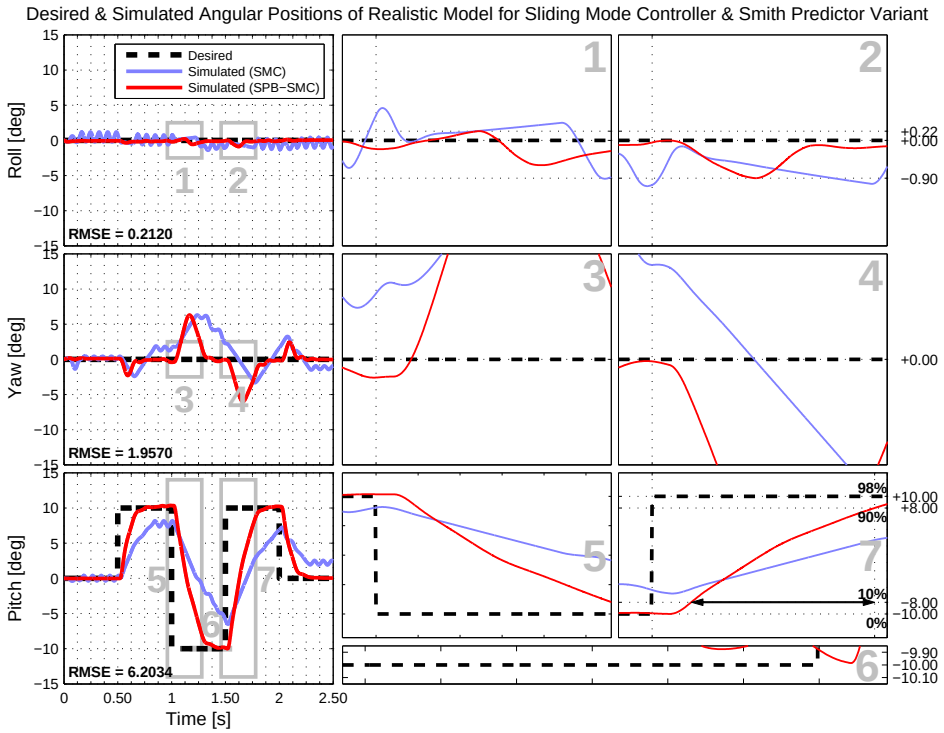


Figure 7.32: Sliding mode controller & Smith-predictor based variant simulated at pitch axis

7.2.3 Double-Loop LQT+I+PI Controller

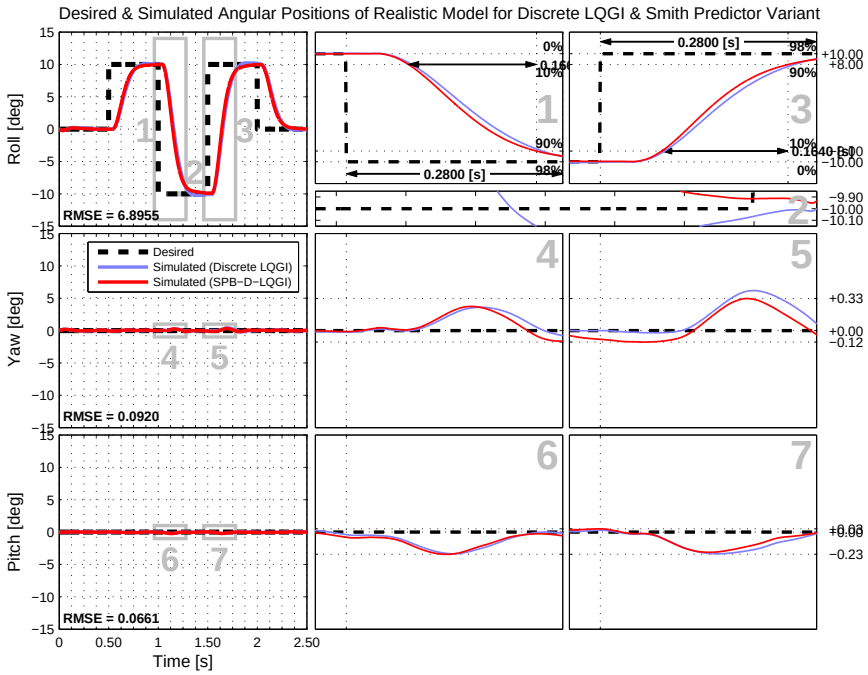


Figure 7.33: Discrete linear quadratic gaussian tracker & Smith-predictor based variant simulated at roll axis

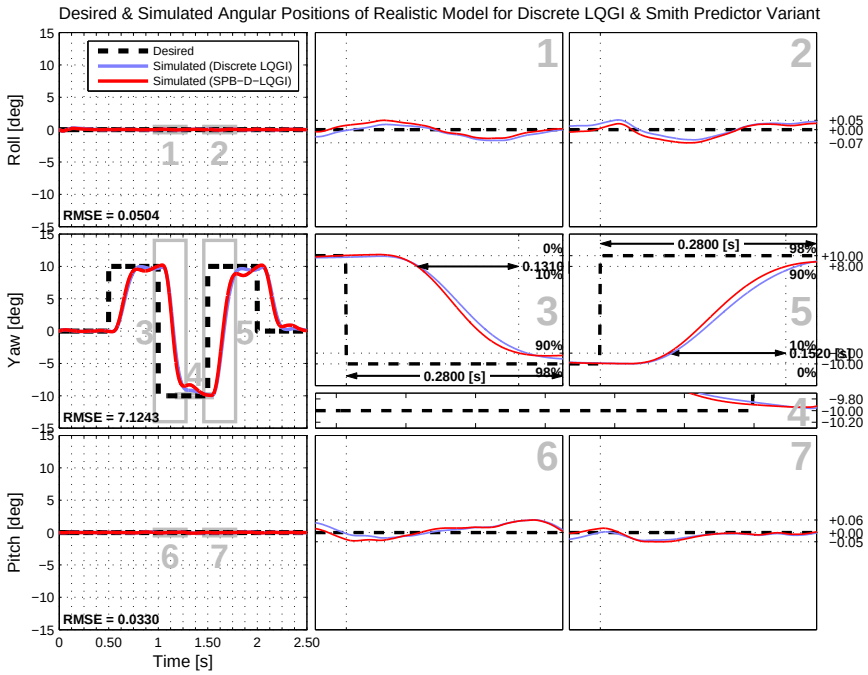


Figure 7.34: Discrete linear quadratic gaussian tracker & Smith-predictor based variant simulated at yaw axis

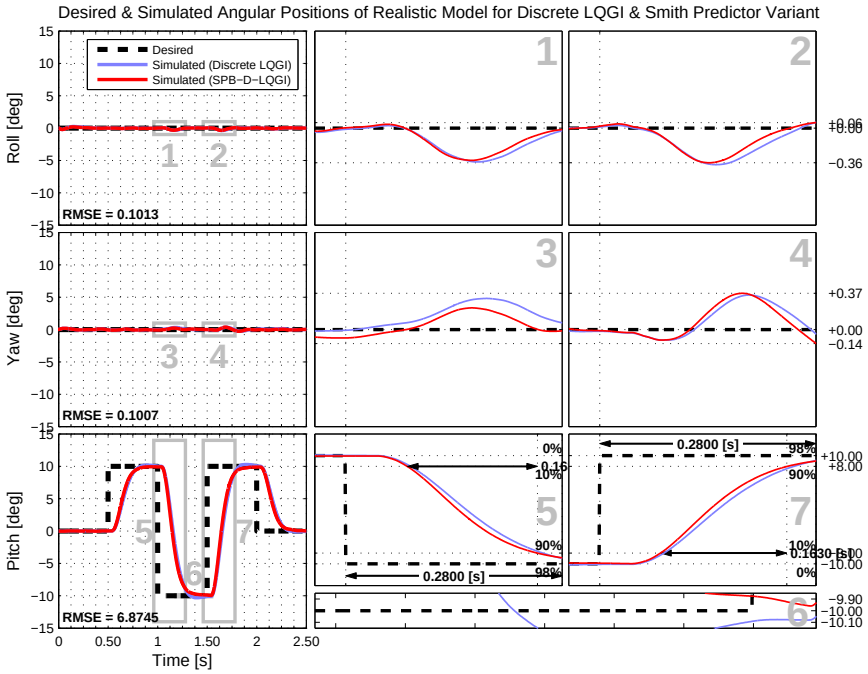


Figure 7.35: Discrete linear quadratic gaussian tracker & Smith-predictor based variant simulated at pitch axis

7.2.4 Double-Loop MPC+PI Controller

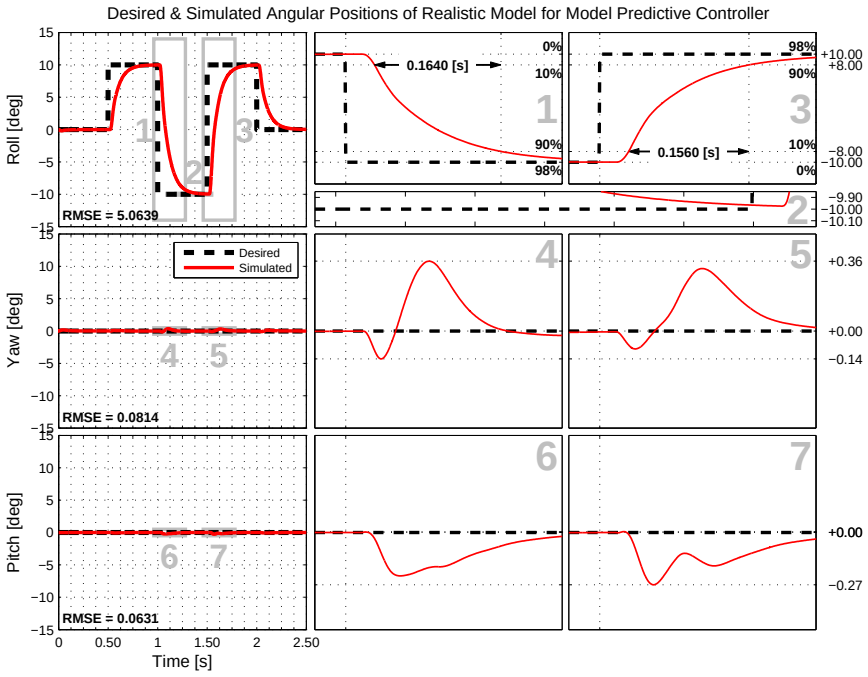


Figure 7.36: Model predictive controller simulated at roll axis

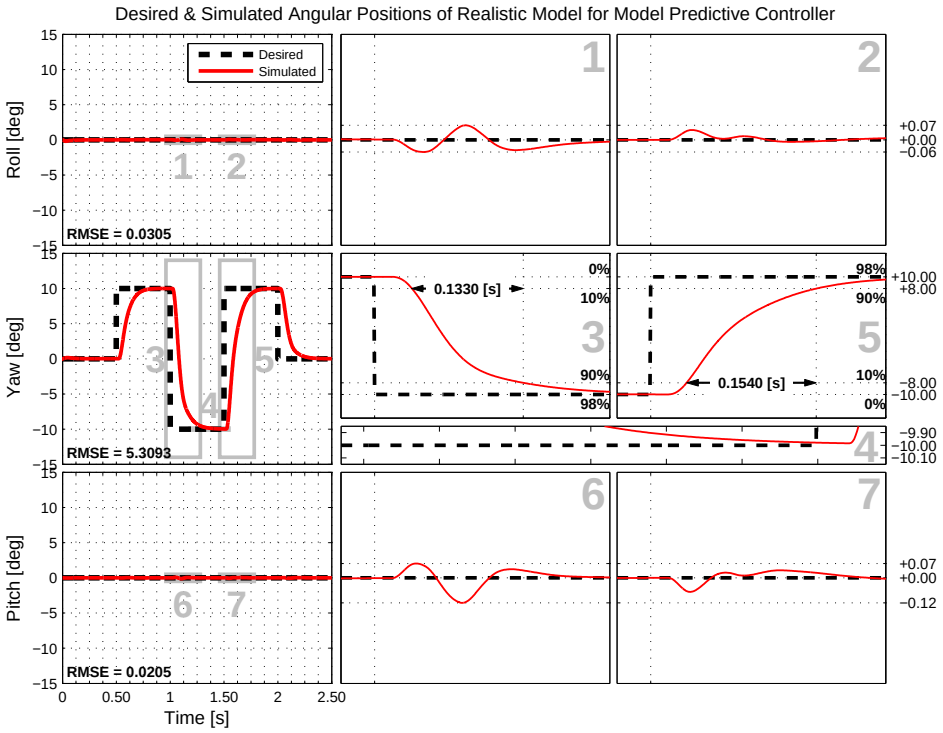


Figure 7.37: Model predictive controller simulated at yaw axis

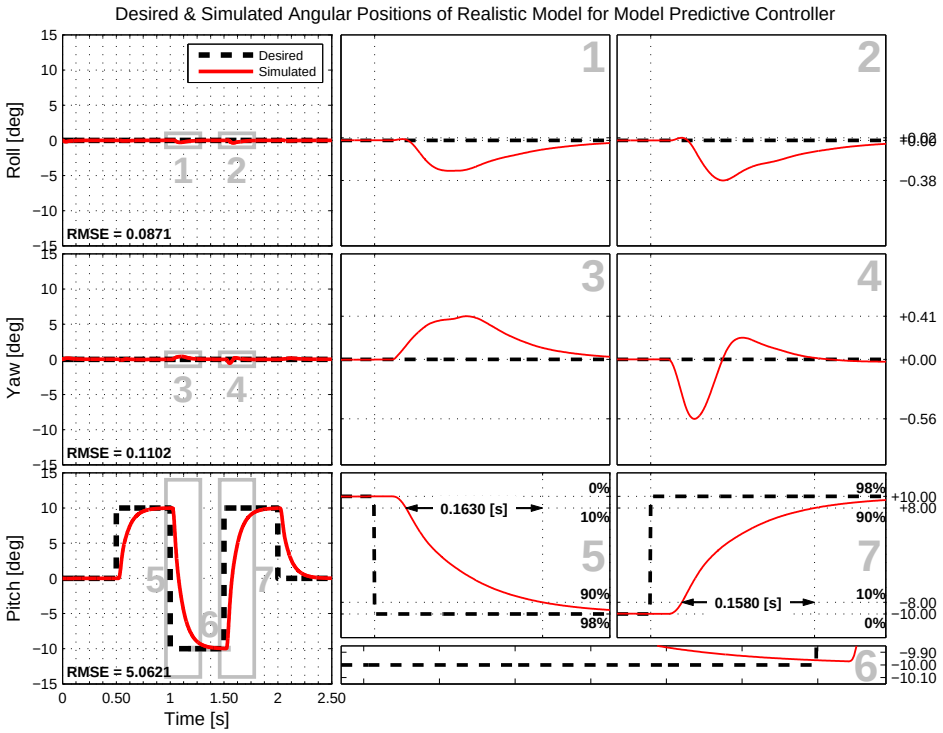


Figure 7.38: Model predictive controller simulated at pitch axis

7.3 Physical Experiments: Tracking Performance of Proposed Predictor & Selected Controller

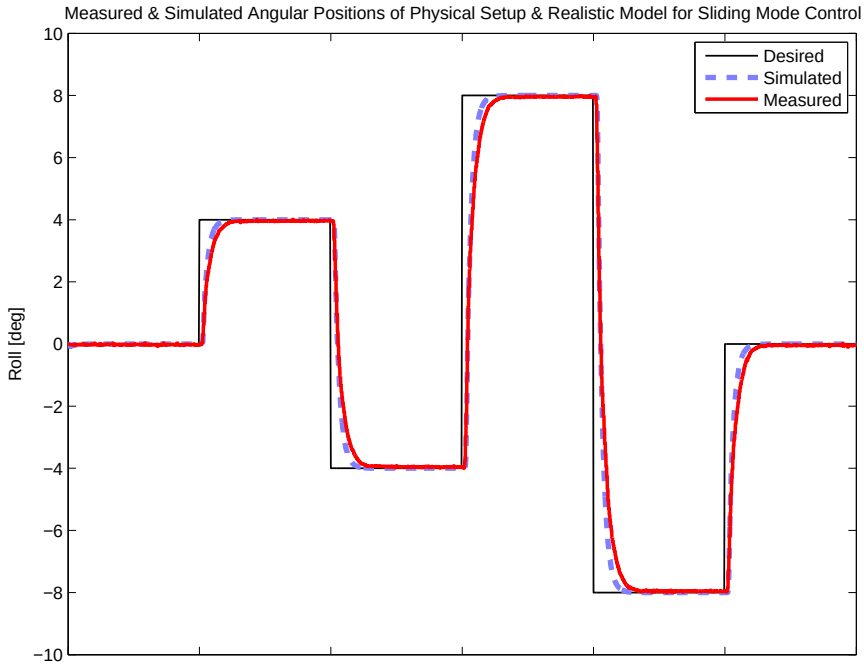


Figure 7.39: Sliding mode controller roll axis movement

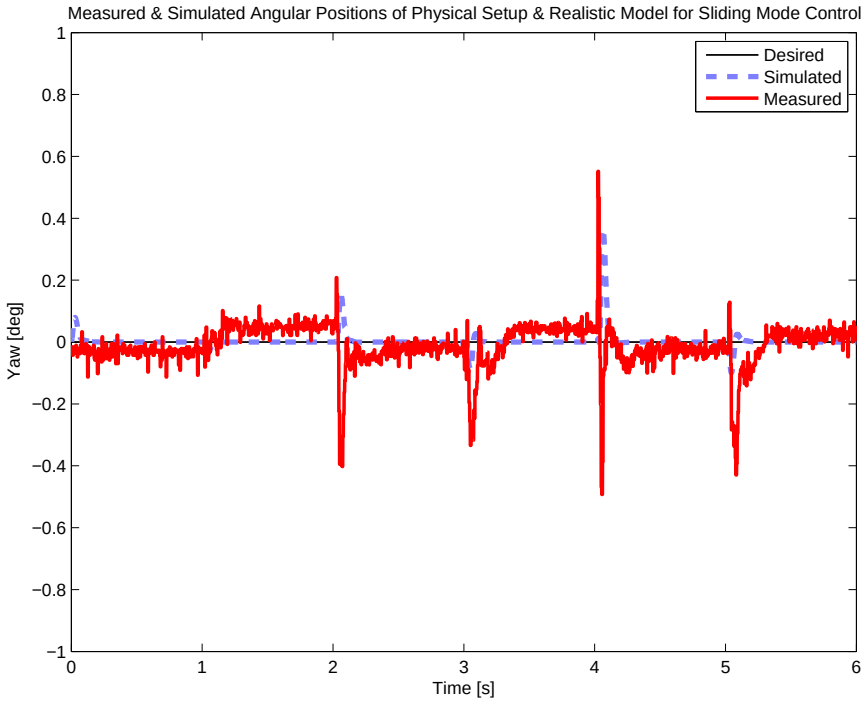


Figure 7.40: Sliding mode controller yaw axis movement

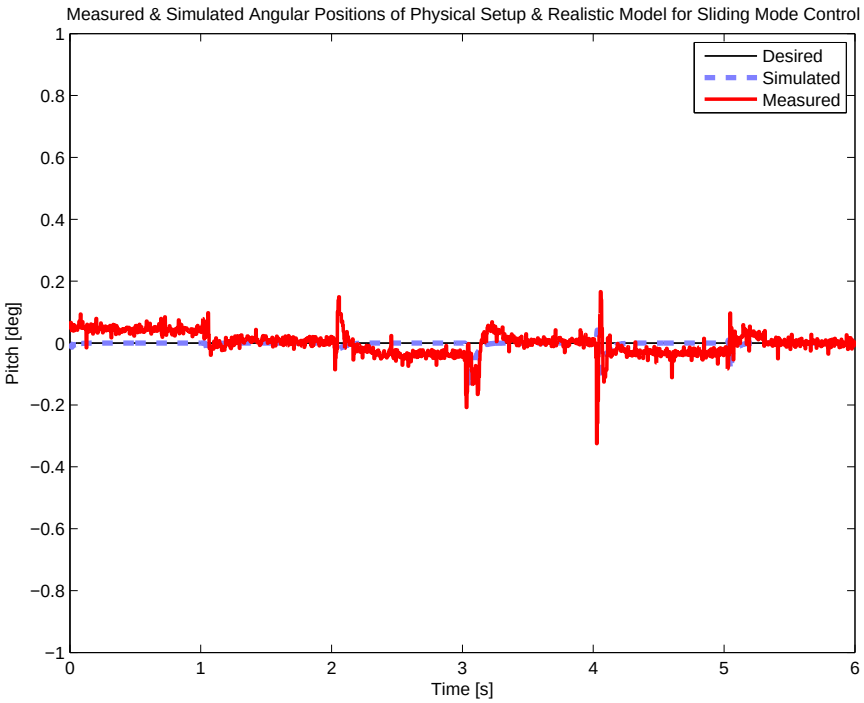


Figure 7.41: Sliding mode controller pitch axis movement

REFERENCES

[1] A. Agrawal, Y. Xu, and R. Raskar. Invertible motion blur in video. In *ACM SIGGRAPH 2009 papers*, SIGGRAPH '09, pages 95:1–95:8, New York, NY, USA, 2009. ACM.

[2] M. Algrain and J. Quinn. Accelerometer based line-of-sight stabilization approach for pointing and tracking systems. In *Control Applications, 1993., Second IEEE Conference on*, pages 159–163 vol.1, 1993.

[3] M. C. Algrain. Gyroless line-of-sight stabilization for pointing and tracking systems. *Optical Engineering*, 33(4):1255–1260, 1994.

[4] K. Ali, C. Vanelli, J. Biesiadecki, M. Maimone, Y. Cheng, A. San Martin, and J. Alexander. Attitude and position estimation on the mars exploration rovers. In *Systems, Man and Cybernetics, 2005 IEEE International Conference on*, volume 1, pages 20 – 27 Vol. 1, 2005.

[5] B. Bandyopadhyay, F. Deepak, and K.-S. Kim. *Sliding Mode Control Using Novel Sliding Surfaces*. Springer Berlin Heidelberg, 2009.

[6] M. Ben-Ezra and S. Nayar. Motion deblurring using hybrid imaging. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–657–I–664 vol.1, 2003.

[7] F. Borrelli, A. Bemporad, and M. Morari. *Predictive Control for Linear and Hybrid Systems*. Cambridge University Press, 2014.

[8] J. Bouguet. A caltech camera calibration toolbox for matlab. `http://www.vision.caltech.edu/bouguetj/calib_doc/index.html`, 2007.

[9] J. Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-8(6):679–698, 1986.

[10] S. J. Chapman. Dc machinery fundamentals. In *Electric Machinery Fundamentals, 5th Edition*, pages 404–463. McGraw Hill, 2012.

[11] S. Chikkerur, V. Sundaram, M. Reisslein, and L. Karam. Objective video quality assessment methods: A classification, review, and performance comparison. *Broadcasting, IEEE Transactions on*, 57(2):165–182, 2011.

[12] F. Crete, T. Dolmiere, P. Ladret, and M. Nicolas. The blur effect: perception and estimation with a new no-reference perceptual blur metric. In *Human Vision and Electronic Imaging XII*, volume 6492, pages 64920I–11, 2007.

[13] J. C. Debruin. Derivation of line-of-sight stabilization equations for gimbaled-mirror optical systems. In M. A. Ealey, editor, *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 1543 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, pages 236–247, jan 1992.

[14] J. Espinosa, J. Vandewalle, and V. Wertz. *Fuzzy Logic, Identification and Predictive Control*. Springer London, 2005.

[15] W. Evers, I. Besselink, H. Nijmeijer, and A. V. d. Knaap. Development and validation of a modular simulation model for commercial vehicles. *International Journal of Heavy Vehicle Systems*, 16(1-1):132–153, 2009-03-17T00:00:00.

[16] C. Fang-Jenq. Application of least-squares adjustment technique to geometric camera calibration and photogrammetric flow visualization. Technical report, 1997.

[17] R. Ferzli and L. Karam. Human visual system based no-reference objective image sharpness metric. In *Image Processing, 2006 IEEE International Conference on*, pages 2949–2952, 2006.

[18] D. A. Forsyth and J. Ponce. *Computer Vision: A Modern Approach*. Prentice Hall Professional Technical Reference, 2002.

[19] S. Gabarda and G. Cristóbal. Blind image quality assessment through anisotropy. *J. Opt. Soc. Am. A*, 24(12):B42–B51, Dec 2007.

[20] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, 1992.

[21] J. Goulding. Biologically-inspired image-based sensor fusion approach to compensate gyro sensor drift in mobile robot systems that balance. In *Multisensor Fusion and Integration for Intelligent Systems (MFI), 2010 IEEE Conference on*, pages 102 –108, 2010.

[22] P. C. Hansen, J. G. Nagy, and D. P. O'Leary. *Deblurring Images: Matrices, Spectra, and Filtering (Fundamentals of Algorithms 3) (Fundamentals of Algorithms)*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2006.

[23] C. Harris and M. Stephens. A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151, 1988.

[24] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

[25] J. Heikkila and O. Silven. A four-step camera calibration procedure with implicit image correction. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 1106–1112, 1997.

[26] T. Henderson, M. Dekhil, R. Kessler, and M. Griss. Sensor fusion. In B. Siciliano and K. Valavanis, editors, *Control Problems in Robotics and Automation*, volume 230 of *Lecture Notes in Control and Information Sciences*, pages 193–207. Springer Berlin / Heidelberg, 1998. 10.1007/BFb0015084.

[27] J. M. Hilkert. A comparison of inertial line-of-sight stabilization techniques using mirrors. In M. K. Masten and L. A. Stockum, editors, *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 5430 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, pages 13–22, jul 2004.

[28] J. M. Hilkert. Inertially stabilized platform technology concepts and principles. *Control Systems, IEEE*, 28(1):26–46, 2008.

[29] T. Horiuchi, M. Inoue, T. Konno, and Y. Namita. Real-time hybrid experimental system with actuator delay compensation and its application to a piping system with energy absorber. *Earthquake Engineering & Structural Dynamics*, 28(10):1121–1141, 1999.

[30] A. Huster, E. Frew, and S. Rock. Relative position estimation for auvs by fusing bearing and inertial rate sensor measurements. In *OCEANS '02 MTS/IEEE*, volume 3, pages 1863 – 1870 vol.3, 2002.

[31] A. Huster and S. Rock. Relative position estimation for manipulation tasks by fusing vision and inertial measurements. In *OCEANS, 2001. MTS/IEEE Conference and Exhibition*, volume 2, pages 1025 –1031 vol.2, 2001.

[32] K. Iagnemma, D. Golda, M. Spenko, and S. Dubowsky. Experimental study of high-speed rough-terrain mobile robot models for reactive behaviors. In B. Siciliano and P. Dario, editors, *Experimental Robotics VIII*, volume 5 of *Springer Tracts in Advanced Robotics*, pages 654–663. Springer Berlin / Heidelberg, 2003.

[33] W. Ji, Q. Li, and B. Xu. Design study of adaptive fuzzy pid controller for los stabilized system. In *Intelligent Systems Design and Applications, 2006. ISDA '06. Sixth International Conference on*, volume 1, pages 336–341, 2006.

[34] M. Jianhong. Smith predictor for controlling systems with time delay. `http://urrg.eng.usm.my/index.php`, May 2012.

[35] P. Kennedy and R. Kennedy. Direct versus indirect line of sight (los) stabilization. *Control Systems Technology, IEEE Transactions on*, 11(1):3 – 15, Jan. 2003.

[36] R. Kikuuwe, S. Yasukouchi, H. Fujimoto, and M. Yamamoto. Proxy-based sliding mode control: A safer extension of pid position control. *Robotics, IEEE Transactions on*, 26(4):670–683, Aug 2010.

[37] H. Komsuoglu. Dynamic legged mobility—an overview. In *Proceedings of International Joint Robotics Conference and Workshop*, 2009.

[38] D. Kundur and D. Hatzinakos. Blind image deconvolution. *Signal Processing Magazine, IEEE*, 13(3):43 –64, May 1996.

[39] T. Lee, E. K. Koh, and M. K. Loh. Stable adaptive control of multivariable servomechanisms, with application to a passive line-of-sight stabilization system. *Industrial Electronics, IEEE Transactions on*, 43(1):98–105, 1996.

[40] T. Lee, K. Tan, and M. Lee. A variable structure-augmented adaptive controller for a gyro-mirror line-of-sight stabilization platform. *Mechatronics*, 8(1):47 – 64, 1998.

[41] P.-C. Lin, H. Komsuoglu, and D. Koditschek. Sensor data fusion for body state estimation in a hexapod robot with dynamical gaits. *Robotics, IEEE Transactions on*, 22(5):932 –943, 2006.

[42] J. Liu and X. Wang. *Advanced Sliding Mode Control for Mechanical Systems*. Springer Berlin Heidelberg, 2011.

[43] L. B. Lucy. An iterative technique for the rectification of observed distributions. *The Astronomical Journal*, 79:745+, June 1974.

[44] X. Marichal, W.-Y. Ma, and H. Zhang. Blur determination in the compressed domain using dct information. In *Image Processing, 1999. ICIP 99. Proceedings. 1999 International Conference on*, volume 2, pages 386–390 vol.2, 1999.

[45] P. Marziliano, F. Dufaux, S. Winkler, and T. Ebrahimi. Perceptual blur and ringing metrics: application to jpeg2000. *Signal Processing: Image Communication*, 19(2):163 – 172, 2004.

[46] M. K. Masten. Inertially stabilized platforms for optical imaging systems. *Control Systems, IEEE*, 28(1):47–64, 2008.

[47] Y. Matsushita, E. Ofek, X. Tang, and H.-Y. Shum. Full-frame video stabilization. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 50 – 57 vol. 1, 2005.

[48] J. A. R. K. Moorty, R. Marathe, and V. R. Sule'. $\mathcal{H}_\infty$ control law for line-of-sight stabilization for mobile land vehicles. *Optical Engineering*, 41(11):2935–2944, 2002.

[49] C. Morimoto and R. Chellappa. Fast electronic digital image stabilization. In *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, volume 3, pages 284–288 vol.3, 1996.

[50] D. Morrison. Model predictive control, honeywell international inc. `http://www.honeywellonramp.com/TechnicalResources/Pages/Model-Predictive-Control.aspx`, June 2014.

[51] K. Mostov, A. Soloviev, and T.-K. Koo. Initial attitude determination and correction of gyro-free ins angular orientation on the basis of gps linear navigation parameters. In *Intelligent Transportation System, 1997. ITSC '97., IEEE Conference on*, pages 1034 –1039, Nov. 1997.

[52] S. Nayar and M. Ben-Ezra. Motion-based motion deblurring. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(6):689 –698, 2004.

[53] J. Nygards, P. Skoglar, M. Ulvklo, and T. Hogstrom. Navigation aided image processing in uav surveillance: Preliminary results and design of an airborne experimental system. *Journal of Robotic Systems*, 21(2):63–72, 2004.

[54] J. Osborne, G. Hicks, and R. Fuentes. Global analysis of the double-gimbal mechanism. *Control Systems, IEEE*, 28(4):44–64, 2008.

[55] M. K. Passino and S. Yurkovich. *Fuzzy Control*. Addison Wesley Publishing Company, 2008.

[56] D. T. Pham and X. Liu. *Neural Networks for Identification, Prediction and Control*. Springer London, 1995.

[57] A. Piegat. *Fuzzy Modeling and Control*. Physica-Verlag HD, 2001.

[58] H. W. Richardson. Bayesian-based iterative method of image restoration. *Journal of the Optical Society of America*, 62(1):55–59, January 1972.

[59] J. M. Royalty. Development of kinematics for gimballed mirror systems. In S. Gowrinathan, editor, *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 1304 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, pages 262–274, sep 1990.

[60] A. Rue. Stabilization of precision electrooptical pointing and tracking systems. *Aerospace and Electronic Systems, IEEE Transactions on*, AES-5(5):805–819, 1969.

[61] A. Rue. Precision stabilization systems. *Aerospace and Electronic Systems, IEEE Transactions on*, AES-10(1):34–42, 1974.

[62] Y. Shtessel, C. Edwards, L. Fridman, and A. Levant. *Sliding Mode Control and Observation*. Springer New York, 2014.

[63] M. Stephens and C. Harris. 3d wire-frame integration from image sequences. *Image and Vision Computing*, 7(1):24 – 30, 1989.

[64] D. Strelow and S. Singh. Optimal motion estimation from visual and inertial measurements. In *Applications of Computer Vision, 2002. (WACV 2002). Proceedings. Sixth IEEE Workshop on*, pages 314 – 319, 2002.

[65] H. Tong, M. Li, H. Zhang, and C. Zhang. Blur detection for digital images using wavelet transform. In *Multimedia and Expo, 2004. ICME '04. 2004 IEEE International Conference on*, volume 1, pages 17–20 Vol.1, 2004.

[66] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.

[67] J.-Y. Tu, W.-D. Hsiao, and C.-Y. Chen. Modelling and control issues of dynamically substructured systems: adaptive forward prediction taken as an example. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 470(2168), Jun 2014.

[68] V. I. Utkin. First stage of vss: People and events. In X. Yu and J.-X. Xu, editors, *Variable Structure Systems: Towards the 21st Century*, volume 274 of *Lecture Notes in Control and Information Sciences*, pages 1–32. Springer Berlin Heidelberg, 2002.

[69] F. Uzer. Camera motion blur and its effects on feature detectors. Master's thesis, 2010.

[70] M. Wallace, D. Wagg, and S. Neild. An adaptive polynomial based forward prediction algorithm for multi-actuator real-time dynamic substructuring. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 461(2064):3807–3826, 2005.

[71] L. Wang, M. Ling, D. Wang, Y. Chen, and C. Wang. Line-of-sight stabilization system based on fractional-order control. In *Systems and Control in Aerospace and Astronautics, 2008. ISSCAA 2008. 2nd International Symposium on*, pages 1–4, 2008.

[72] J. M. Wicker, T. Yang, J. Ly, R. L. Wong, P. Dahl, and R. O. Rogers. Low cost spacecraft disturbance rejection via sensor matching for off-gimbal precision pointing system. In *AIAA Guidance Navigation and Control Conference and Exhibit*, pages 5827–5833, 2003.

[73] C.-L. Xia. Mathematical model and characteristics analysis of the bldc motor. In *Permanent Magnet Brushless DC Motor Drives and Controls*, pages 25–62. John Wiley & Sons Singapore Pte. Ltd., 2012.

[74] H. Zhang and D. Liu. *Fuzzy Modeling and Fuzzy Control*. Birkhäuser Boston, 2006.

APPENDIX A

TECHNICAL SPECIFICATIONS OF THE ACTUATING MECHANISM

The technical specifications of the Faulhaber Quickshaft LM1247-020-01 series linear DC servomotor and Faulhaber MCLM 3006S series external motion driver are given in Appendix A.1 and Appendix A.2, respectively.

A.1 Faulhaber Quickshaft LM1247 Series Linear DC Servomotor

FAULHABER

Linear DC-Servomotors 3,6 N

with Analog Hall Sensors
QUICKSHAFT® Technology

For combination with
Motion Controllers:
MCLM 3003/06 S, MCLM 3003/06 C

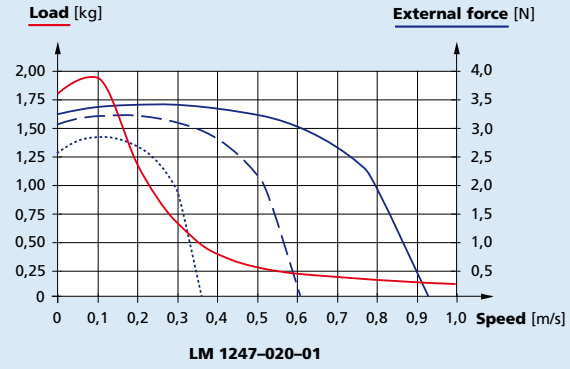
Series LM 1247 ... 01

		LM 1247–	020–01	040–01	060–01	080–01	100–01	120–01	
1	Continuous force [1]	$F_{e\,max.}$	3,6						N
2	Peak force [1] [2]	$F_{p\,max.}$	10,7						N
3	Continuous current [1]	$I_{e\,max.}$	0,55						A
4	Peak current [1] [2]	$I_{p\,max.}$	1,66						A
5	Back-EMF constant	k_E	5,25						V/m/s
6	Force constant [3]	k_F	6,43						N/A
7	Terminal resistance, phase-phase	R	13,17						Ω
8	Terminal inductance, phase-phase	L	820						µH
9	Stroke length	$s_{max.}$	20	40	60	80	100	120	mm
10	Repeatability [4]		40	40	40	40	40	40	µm
11	Precision [4]		120	140	160	180	200	220	µm
12	Acceleration [5]	$a_{e\,max.}$	198,0	148,5	127,3	101,8	91,4	82,9	m/s^2
13	Speed [5] [6]	$v_{e\,max.}$	2,0	2,4	2,8	2,9	3,0	3,2	m/s
14	Thermal resistance	$R_{th\,1}$ / $R_{th\,2}$	3,2 / 20,0						K/W
15	Thermal time constant	τ_{w1} / τ_{w2}	11 / 624						s
16	Operating temperature range		– 20 ... +125						°C
17	Rod weight [7]	m_m	18	24	28	35	39	43	g
18	Total weight [7]	m_t	57	63	67	74	78	82	g
19	Magnetic pitch	τ_m	18						mm
20	Rod bearings		polymer sleeves						
21	Housing material		metal, non-magnetic						
22	Direction of movement		electronically reversible						

[1] thermal resistance $R_{th\,2}$ by 55% reduced
[2] for max. 1 second with a duty cycle of 20%
[3] with sine wave commutation
[4] typical values with integrated linear Hall sensors and Motion Controller MCLM 3003/06 S/C.
The values depend on conditions of use
[5] theorical value, referring only to the motor
[6] with a triangular speed profile and the max. stroke
[7] rounded value, for reference only

Notes: These motors are for operation with DC-voltage < 75 V DC.
The given values are for free standing motors.
The mounting with magnetic conductive metal can influence the characteristics of the motor.

Caution: Presence of strong magnetic fields. Static sensitive device.



LM 1247–020–01

Trapezoidal motion profile (t1 = t2 = t3)

Displacement distance:	20 mm
Friction coefficient:	0,2
Slope angle:	0°
Rest time:	0,1 s

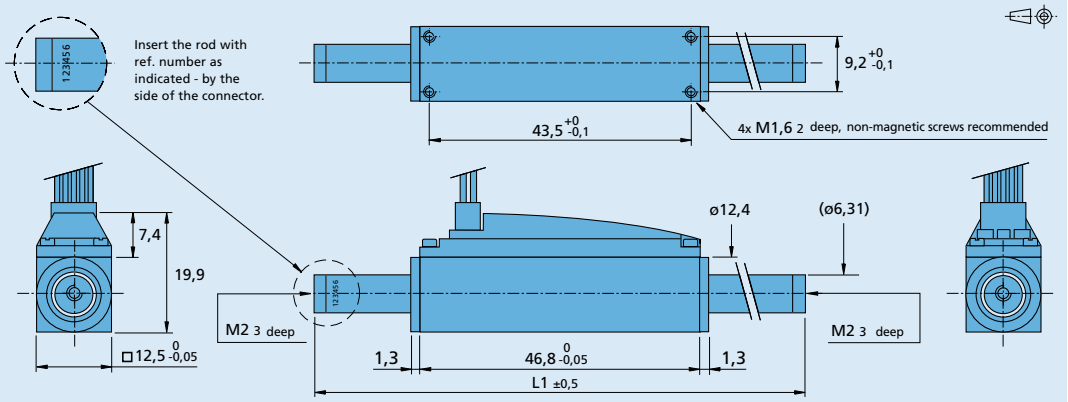
Load: The max. permissible load at a given speed with an external force of 0 N

External force: The max. permissible external force at a given speed with a load of:
- 0,1 Kg
- 0,2 Kg
- 0,5 Kg

For notes on technical data and lifetime performance refer to "Technical Information".
Edition 2010 Jun. 14

© DR. FRITZ FAULHABER GMBH & CO. KG
Specifications subject to change without notice.

Linear DC-Servomotor LM 1247

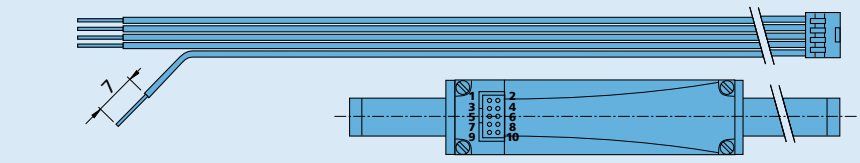


Ordering information

Linear DC-Servomotors Series	Stroke mm	Rod length L1 ±0,5 mm
LM 1247–020–01	–10 … +10	82
LM 1247–040–01	–20 … +20	109
LM 1247–060–01	–30 … +30	127
LM 1247–080–01	–40 … +40	154
LM 1247–100–01	–50 … +50	172
LM 1247–120–01	–60 … +60	190

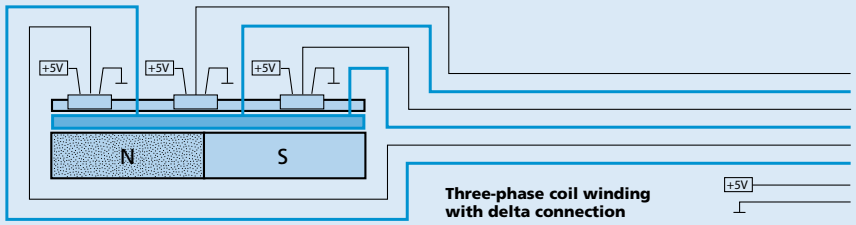
Note: Single rod available on request.

Cable and connection information



Cable
Single wires, material PVC
Length 200 mm ± 10 mm
10 conductors, AWG 28

Recommended connector
Molex - Nr. 51110-1060



Three-phase coil winding with delta connection

Connection

PIN	Function	Colour
10	N.C.	purple
9	N.C.	white
6	Hall sensor C	grey
1	Phase C	yellow
5	Hall sensor B	blue
7	Phase B	orange
2	Hall sensor A	green
8	Phase A	brown
3	Logic supply +5V	red
4	Logic GND	black

© DR. FRITZ FAULHABER GMBH & CO. KG
Specifications subject to change without notice.

A.2 Faulhaber MCLM 3006S Series External Motion Driver

FAULHABER

Motion Controller

4-Quadrant PWM
with RS232 or CAN interface

For combination with:
Linear DC-Servomotors
with Hall sensors

Series MCLM 3003 S/C

		MCLM 3003S/C	
Power supply	U_B	12 ... 30	V DC
PWM switching frequency	f_{PWM}	78,12	kHz
Efficiency	η	95	%
Max. continuous output current [1]	I_{dauer}	3	A
Max. peak output current	I_{max}	10	A
Total standby current	I_{el}	0,06	A
Speed range [2]		2 ... 10 000	mm/s
Scanning rate	N	100	µs
Encoder resolution with Hall Sensors [3]		≤ 3 000	inc./τ_m
Resolution with external encoder [3]		≤ 65 535	inc./mm
Input/output (partially free configurable)		3	
Program memory:			
– memory size		3,3	kWord
– Number of instructions		approx. 1 000	instructions
Operating temperature range		0 ... + 70	°C
Storage temperature		– 25 ... + 85	°C
Housing material		without housing	
Weight		18	g

[1] at 22°C ambient temperature
[2] Speed in the range 1 ... 5 mm/s may have fluctuations due to the motor type, load characteristics and controller parameters
[3] τ_m is the magnetic pitch of the linear motor

Connection information				
Connection communication:				
Interface			RS232 / CAN	
Communication profile			Faulhaber - ASCII / CAN	
Max. transfer speed rate RS232			115 200	baud
Max. transfer speed rate CAN			1	Mbit/s
Connection "AGND":				
– analog ground			analog GND	
– digital input	external encoder		channel B	
		R_{In}	10	kΩ
		f	≤ 400	kHz
Connection "Fault":				
– digital input		R_{In}	100	kΩ
– digital output (open collector)		U	≤ U_B	V
		I	≤ 30	mA
		clear	switched to GND	
		set	high-impedance	
	fault output	no error	switched to GND	
		error	high-impedance	
	signal output	f	≤ 2	kHz
		resolution	1...255	inc./τ_m
Connection "AnIn":			"AGND" as GND	
– analog input	set position value	U_{In}	± 10	V
– digital input	external encoder		channel A	
		f	≤ 400	kHz
	step frequency input	f	≤ 400	kHz
		R_{In}	5	kΩ
Connection "+24V":		U_B	12 ... 30	V DC
Connection "GND":			ground	
Connection "3. In":				
– digital input		R_{In}	22	kΩ
– electronic supply voltage [4]		U_B	12 ... 30	V DC

[4] Separate supply is optionally available (Option no. 2993).

© DR. FRITZ FAULHABER GMBH & CO. KG
Specifications subject to change without notice.

Connection information					
Phase connection “A”, “B”, “C”:					
	A		Phase A	brown [1]	
	B		Phase B	orange [1]	
	C		Phase C	yellow [1]	
		U_{Out}	0 … U_B		V
PWM switching frequency		f_{PWM}	78,12		kHz
Hall Sensor connection “A”, “B”, “C”:					
	A		Hall Sensor A	green [1]	
	B		Hall Sensor B	blue [1]	
	C		Hall Sensor C	grey [1]	
		U_{In}	≤ 5		V
Connection “SGND”:					
Signal GND			Signal ground	black [1]	
Connection “+5V”:					
Output voltage for external use [2]		U_{Out}	5	red [1]	V DC
Load current		I_{out}	≤ 60		mA

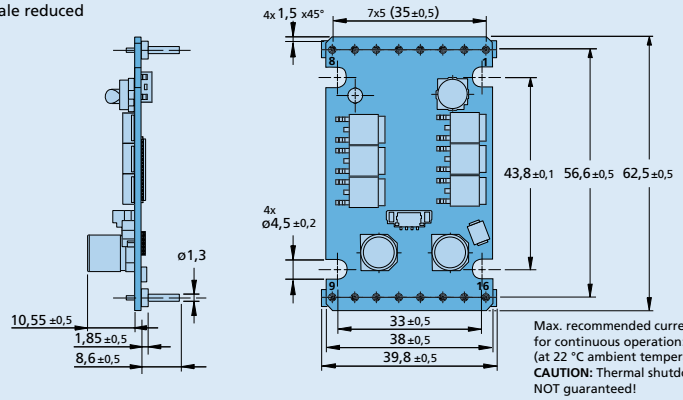
[1] Colour identification for linear DC-Servomotor
[2] E.g. Hall sensor

Digital inputs general information			
- PLC, default	high	12,5 … U_B	V
	low	0 … 7	V
- TTL	high	3,5 … U_B	V
	low	0 … 0,5	V

The signal level (PLC or TTL) of the digital inputs can be set over the interface (see instruction manual).

Dimensional drawing and connection information MCLM 3003 S/C

Scale reduced



Max. recommended current for continuous operation: 3A (at 22 °C ambient temperature)
CAUTION: Thermal shutdown is NOT guaranteed!

Connection

Pin	Function
1	Ph C
2	Hall A
3	+ 5V
4	SGND
5	Hall B
6	Hall C
7	Ph B
8	Ph A
9	TxD / CAN_H
10	RxD / CAN_L
11	AGND
12	Fault
13	AnIn
14	+ 24V
15	GND
16	3. In

© DR. FRITZ FAULHABER GMBH & CO. KG
Specifications subject to change without notice.

MC Function / RS232

General description

FAULHABER Motion Controllers are available as external electronic controls to be used in combination with DC-Micromotors, Brushless DC-Servomotors and Linear DC-Servomotors or, if already integrated, with Brushless DC-Servomotors in the form of motion control systems. Motion control is thus possible for the majority of motors from a diameter of 6 mm upward. The integrated systems reduce the amount of space required. At the same time, their wiring requirements are minimal, which helps to simplify installation.

Operating modes

- **Speed control**
  PI speed control, even for demanding synchronization requirements
- **Positioning**
  For moving to defined positions with a high level of resolution. With a PD Controller, the dynamic response can be adjusted to suit the application. Reference and limit switches are evaluated by means of various homing modes.
- **Speed profiles**
  Acceleration ramps, deceleration ramps and maximum velocity can also be defined for each section. As a result, even complex profiles can be implemented quickly and effectively.
- **Current control**
  Protects the drive by limiting the motor current to the set peak current. The current is limited to the continuous current by the integrated I^2t monitoring if required.
- **Protective functions**
  - Protection against ESD
  - Overload protection for the electronic circuitry and the motor
  - Self-protection against overheating
  - Overvoltage protection in generator mode
- **Extended operating modes**
  - Stepper motor mode
  - Gearing mode
  - Position control to analog set point
  - Operation as servo amplifier in voltage adjuster mode
  - Torque/force controller using variable set current input

Interfaces - Discrete I/O

- **Setpoint input**
  Depending on the operating mode, setpoints can be input via the serial port, via an analog voltage value, a PWM signal or a quadrature signal.
- **Error output** (Open Collector)
  This can also be used as a digital input for the evaluation of reference switches or for specifying direction of rotation.
- **RS232 interface**
  For connection to a PC with a transmission rate of up to 115 kbaud. The information can be stored in the integrated memory (FLASH).

The interface also offers the option of querying online operating data and values. The RS232 interface also allows the operation of several networked drives on one control.

Sensor interfaces (position and speed sensors, depending on motor type)

- **Analog Hall signals**
  Three analog Hall signals, offset by 120°, for motor position and speed in the case of Brushless DC-Motors and Linear DC-Servomotors
- **Incremental encoders**
  Incremental encoders for motor position and speed in the case of DC-Micromotors and as additional sensors for Brushless DC-Motors
- **Absolute encoders**
  Serial SSI port for motor position and speed matching Brushless DC-Servomotors with an AES Encoder

RS232 Programming / Configuration

An extensive ASCII command set is available for programming and operation. This can be preset from the PC, e.g. via any Windows terminal program or via any other control computer.

In addition, even complex processes can be created from these commands and stored in the drive. Once programmed as a stepper motor, electronic gear or as a speed or position controller via the analogue input, the drive can be operated independently of the RS232 interface.

"Faulhaber Motion Manager" software is available for Windows operating systems. It considerably simplifies operation and configuration and also enables graphic online analysis of the operating data.

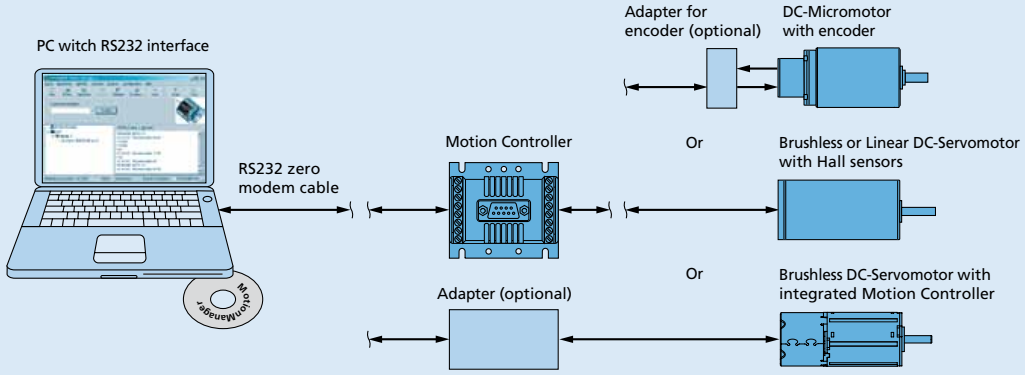
Options

Separate supply of power to the motor and electronic actuator is optional (important for safety-relevant applications), in which case no third input is required. Depending on the controller, additional programming adaptors and connection aids are available. The modes and parameters can be specially preconfigured on request

Notes

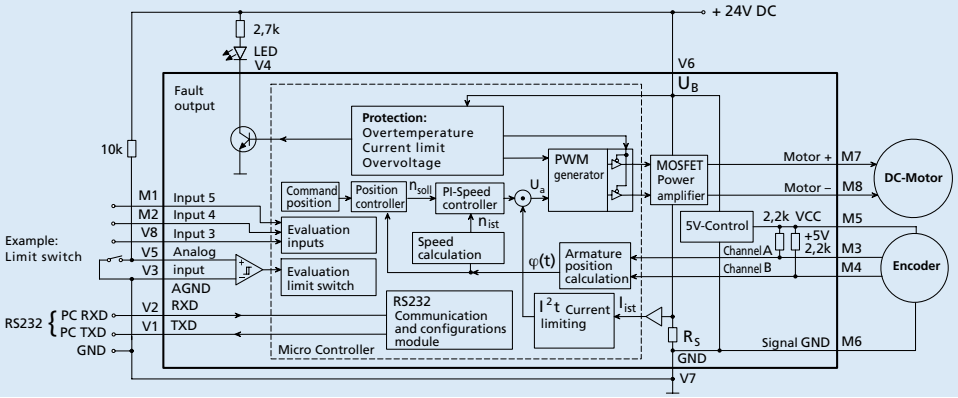
Motion Controllers and Motion Control Systems are accompanied by a **device manual** for installation and putting into operation.
A communication and function manual and the **"Faulhaber Motion Manager"** software are available on request and on the Internet at www.faulhaber.com.

Connection diagram

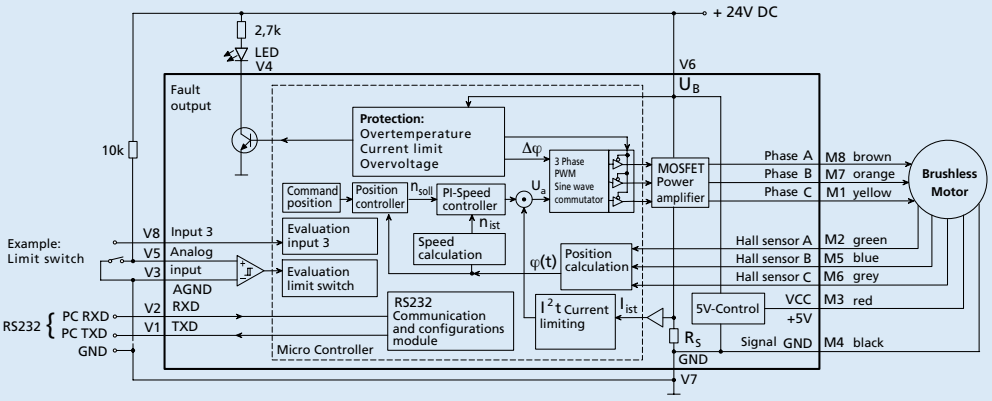


© DR. FRITZ FAULHABER GMBH & CO. KG
Specifications subject to change without notice.

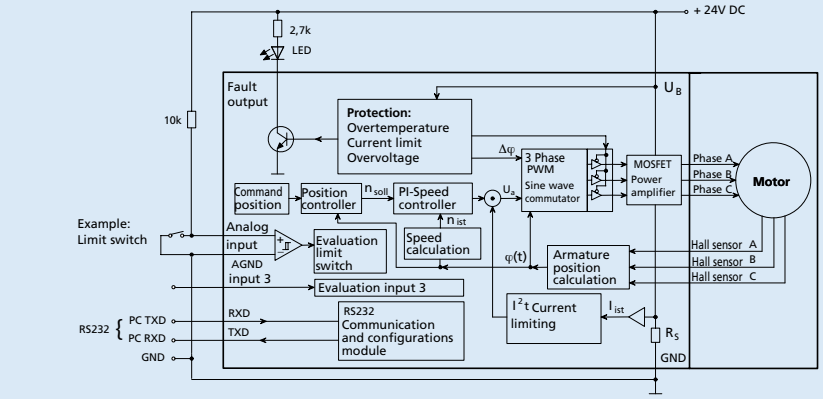
Position control DC-Micromotor with encoder



Position control Brushless DC-Servomotor with Hall sensors



Position control Brushless DC-Servomotor with integrated Motion Controller



© DR. FRITZ FAULHABER GMBH & CO. KG
Specifications subject to change without notice.

MC Function / CAN

General description

FAULHABER Motion Controllers are available as external electronic controls to be used in combination with DC-Micromotors, Brushless DC-Servomotors and Linear DC-Servomotors or, if already integrated, with Brushless DC-Servomotors in the form of motion control systems. Motion control is thus possible for the majority of motors from a diameter of 6 mm upward. The integrated systems reduce the amount of space required. At the same time, their wiring requirements are minimal, which helps to simplify installation.

Operating modes

- **Speed control**
  PI speed control, even for demanding synchronization requirements
- **Positioning**
  For moving to defined positions with a high level of resolution. With a PD Controller, the dynamic response can be adjusted to suit the application. Reference and limit switches are evaluated by means of various homing modes.
- **Speed profiles**
  Acceleration ramps, deceleration ramps and maximum velocity can also be defined for each section. As a result, even complex profiles can be implemented quickly and effectively.
- **Current control**
  Protects the drive by limiting the motor current to the set peak current. The current is limited to the continuous current by the integrated I^2t monitoring if required.
- **Protective functions**
  - Protection against ESD
  - Overload protection for the electronic circuitry and the motor
  - Self-protection against overheating
  - Overvoltage protection in generator mode
- **Extended operating modes**
  - Stepper motor mode
  - Gearing mode
  - Position control to analog set point
  - Operation as servo amplifier in voltage adjuster mode
  - Torque/force controller using variable set current input

Interfaces - Discrete I/O

- **Setpoint input**
  Depending on the operating mode, setpoints can be input via the serial port, via an analog voltage value, a PWM signal or a quadrature signal.
- **Error output** (Open Collector)
  This can also be used as a digital input for the evaluation of reference switches or for specifying direction of rotation.
- **CANopen interface**
  For integration into a CAN network with transfer rates of up to 1 Mbit/s. Via the CAN interface a number of drives can be networked and operated on a higher-level control.

Sensor interfaces (position and speed sensors, depending on motor type)

- **Analog Hall signals**
  Three analog Hall signals, offset by 120°, for motor position and speed in the case of Brushless DC-Motors and Linear DC-Servomotors
- **Incremental encoders**
  Incremental encoders for motor position and speed in the case of DC-Micromotors and as additional sensors for Brushless DC-Motors
- **Absolute encoders**
  Serial SSI port for motor position and speed matching Brushless DC-Servomotors with an AES Encoder

CAN Programming / Configuration

FAULHABER Motion Controllers support the CANopen communication profile under DS301 V4.02 in accordance with the CiA specification for slave devices with the following services:
- 1 Server SDO
- 3 transmit PDOs, 3 receive PDOs
- Static PDO mapping
- NMT with node guarding
- Emergency Object

The transfer rate and node no. are set via the network in accordance with the LSS protocol conforming to DSP305 V1.11, and automatic baud rate detection is also implemented. In addition, all the functions and parameters of the drive unit can be easily activated via a special FAULHABER PDO channel.

As regards the CiA device profile for Motion Controllers (CiA 402), the following are supported:
- Profile Position Mode and Position Control Function
- Homing Mode
- Profile Velocity Mode

Also, for each Faulhaber command there is an appropriate CAN frame available on the PDO channel, with which the CAN unit can be operated in the same way as the serial variant and the extended operating modes can be supported.
"Faulhaber Motion Manager" software is available for Windows operating systems. It considerably simplifies operation and configuration and also enables graphic online analysis of the operating data.

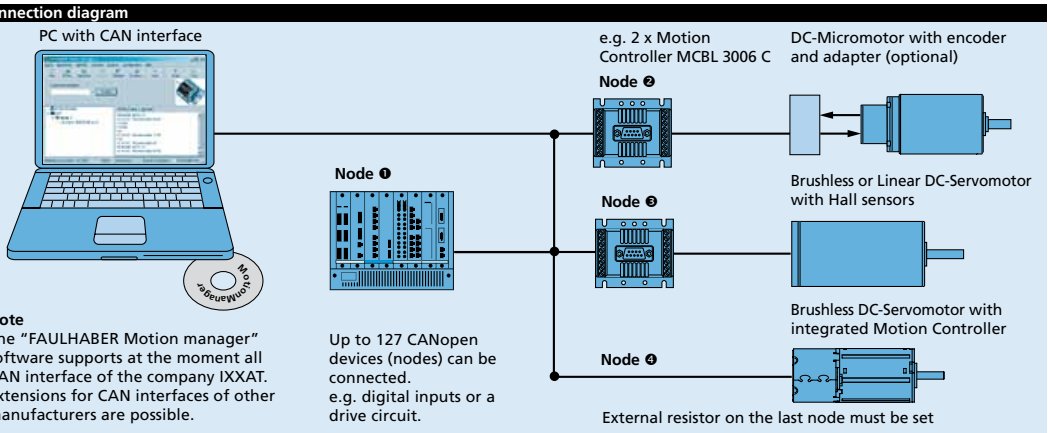
Options

Separate supply of power to the motor and electronic actuator is optional (important for safety-relevant applications), in which case no third input is required. Depending on the controller, additional programming adaptors and connection aids are available. The modes and parameters can be specially preconfigured on request

Notes

Motion Controllers and Motion Control Systems are accompanied by a **device manual** for installation and putting into operation.
A communication and function manual and the **"Faulhaber Motion Manager"** software are available on request and on the Internet at www.faulhaber.com.

Connection diagram



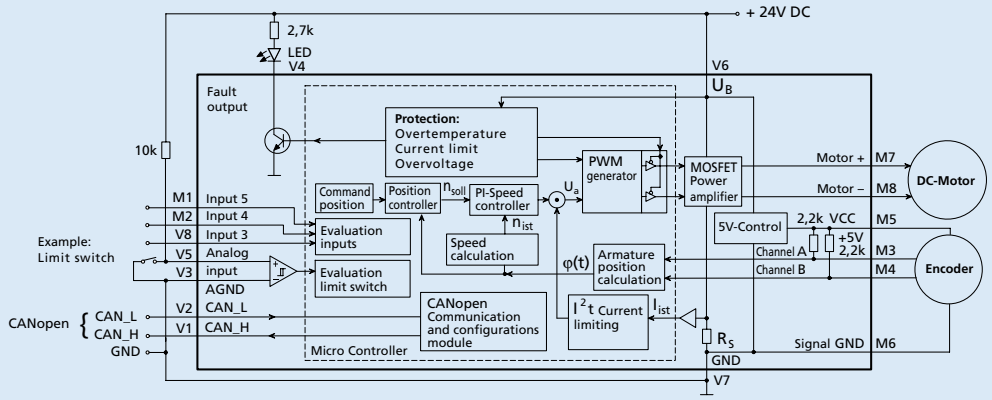
Note
The "FAULHABER Motion manager" software supports at the moment all CAN interface of the company IXXAT. Extensions for CAN interfaces of other manufacturers are possible.

Up to 127 CANopen devices (nodes) can be connected.
e.g. digital inputs or a drive circuit.

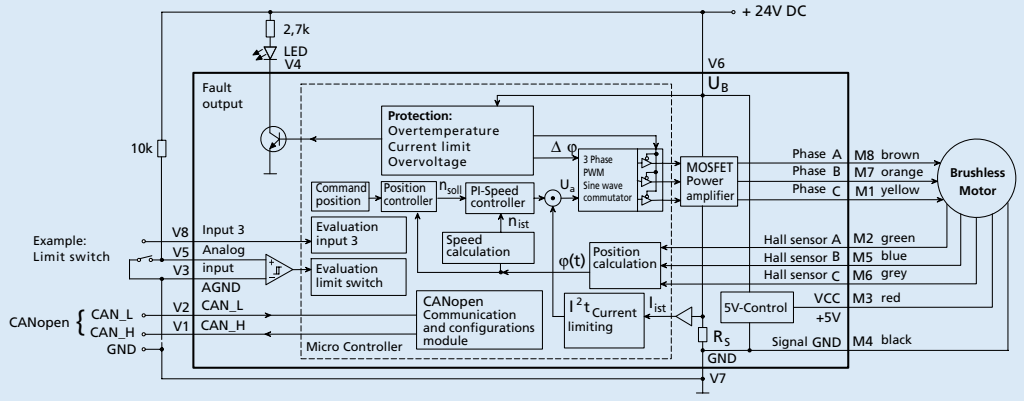
External resistor on the last node must be set

© DR. FRITZ FAULHABER GMBH & CO. KG
Specifications subject to change without notice.

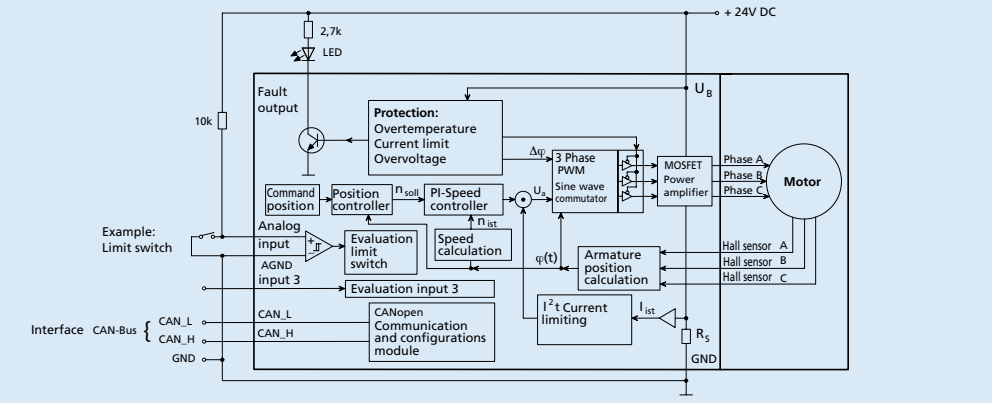
Position control DC-Micromotor with encoder



Position control Brushless DC-Servomotor with Hall sensors



Position control Brushless DC-Servomotor with integrated Motion Controller



© DR. FRITZ FAULHABER GMBH & CO. KG
Specifications subject to change without notice.

APPENDIX B

MECHANICAL COMPONENTS OF THE CAMERA STABILIZATION PLATFORM

In this appendix, construction information about mechanical structure of the camera stabilization platform are given by specifying each component's technical information. In Appendix B.1, the 3D Computer-Aided Design (CAD) models of the purchased and manufactured components are presented individually and they are detailed by summarizing their dimension, mass and inertia properties. Also, the data sheets of the spherical and universal joints, which are used in various mechanical layouts, are supplied in Appendix B.2 and Appendix B.3, respectively.

B.1 3D Computer-Aided Design (CAD) Model of the Components

The Computer-Aided Design (CAD) models of the components, as well as mass and inertia properties are summarized in the following subsections. Note that, the mass property is measured by a precision weighing of accuracy 1 [g] and the inertia property is obtained by using Solidwork software.

B.1.1 Base Platform

The base platform is designed as a carrier, which all of the remaining parts are mounted on. It is a common components used in every configurations. The CAD model of the base platform is shown in Figure B.1. Also, dimension, mass and inertia properties are given in Table B.1.

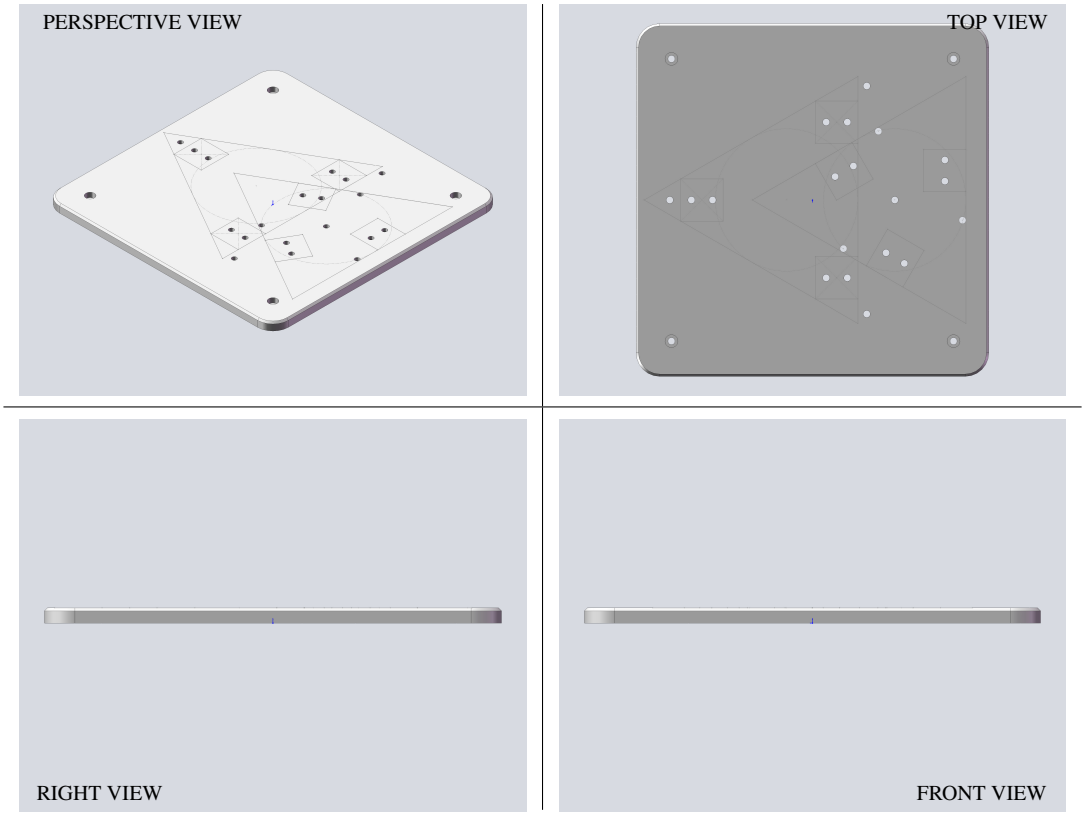


Figure B.1: The CAD model of the base platform

TableB.1: Dimension, mass and inertia properties of the base platform

Name	Value	Unit
Length	149.97×10^{-3}	[m]
Height	4.96×10^{-3}	[m]
Width	149.97×10^{-3}	[m]
Mass	290×10^{-3}	[kg]
Inertia	$\begin{bmatrix} 5.4289 \times 10^{-4} & 3.6771 \times 10^{-8} & -2.7133 \times 10^{-10} \\ 3.6771 \times 10^{-8} & 1.0816 \times 10^{-3} & 2.3957 \times 10^{-12} \\ -2.7133 \times 10^{-10} & 2.3957 \times 10^{-12} & 5.3991 \times 10^{-4} \end{bmatrix}$	[kg.m2]

B.1.2 Top Platform

The top platform is designed as a holder for camera and other sensors, which are desired to stabilize and to isolate from the undesirable movements of the base platform. It is also a common components used in every configurations. The CAD model of the top platform is shown in Figure B.2. The dimension, mass and inertia properties are given in Table B.2.

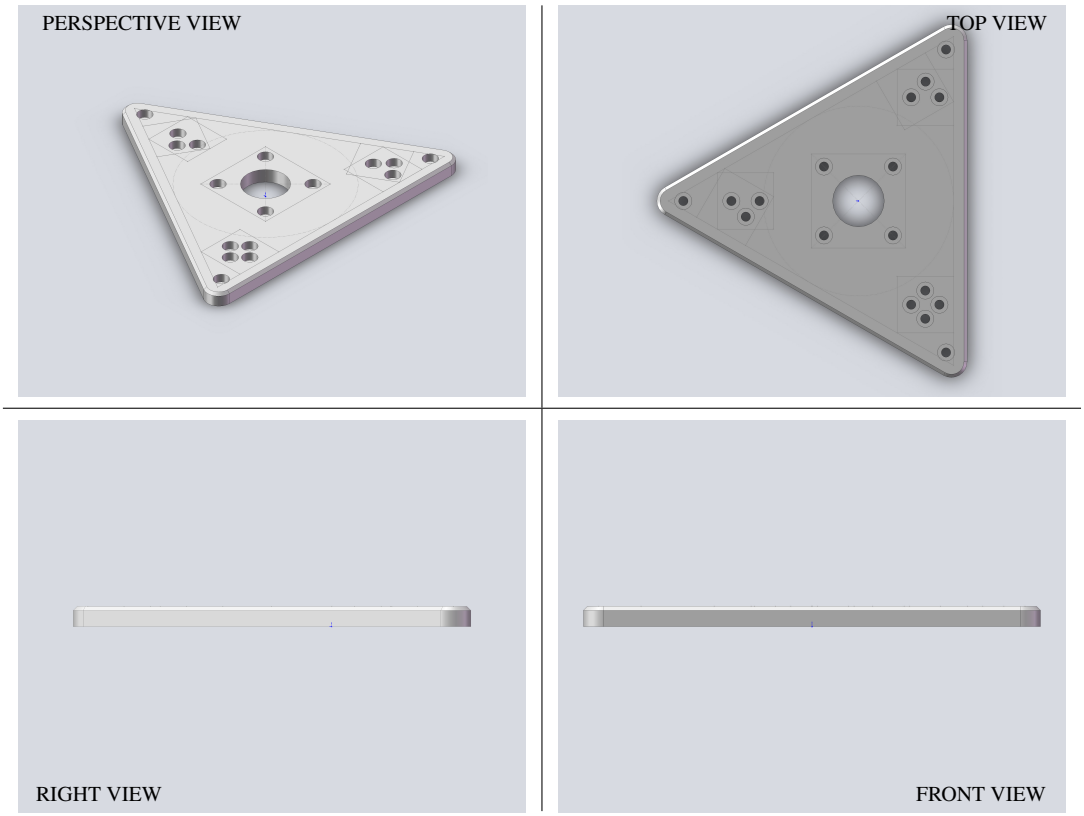


Figure B.2: The CAD model of the top platform

TableB.2: Dimension, mass and inertia properties of the top platform

Name	**Value**	**Unit**
Length	96.91×10^{-3}	[m]
Height	4.96×10^{-3}	[m]
Width	109.81×10^{-3}	[m]
Mass	74×10^{-3}	[kg]
Inertia	$\begin{bmatrix} 4.4407 \times 10^{-5} & 8.0170 \times 10^{-10} & -1.0297 \times 10^{-7} \\ 8.0170 \times 10^{-10} & 8.8316 \times 10^{-5} & 7.2958 \times 10^{-9} \\ -1.0297 \times 10^{-7} & 7.2958 \times 10^{-9} & 4.4216 \times 10^{-5} \end{bmatrix}$	[kg.m2]

B.1.3 Vertical Support Column

The vertical support column is designed to carry the weight of the top platform, as well as the camera and other sensors which are attached to it. Two types of support columns are manufactured for different configurations. The first one, which is shown in Figure B.3, is designed for single-axis stabilization platform.

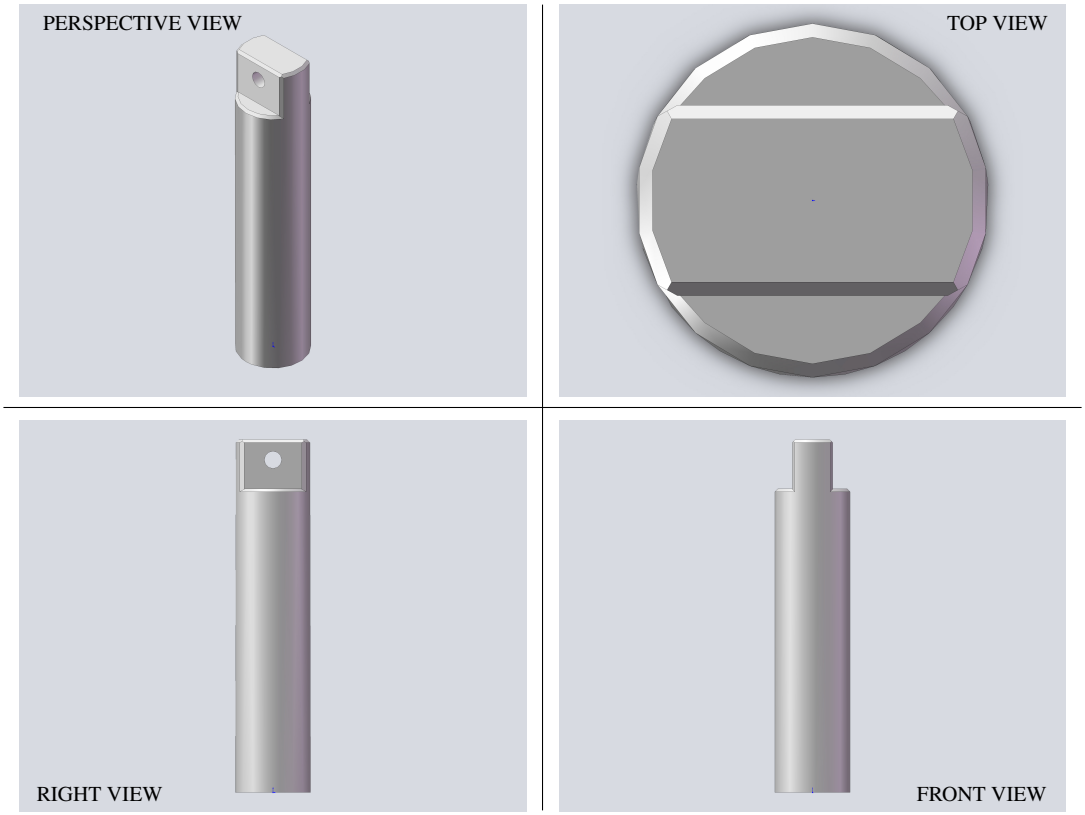


Figure B.3: The CAD model of the support column for single-axis stabilization platform

TableB.3: Dimension, mass and inertia properties of the support column for single-axis stabilization platform

Name	**Value**	**Unit**
Length	13.00×10^{-3}	[m]
Height	61.06×10^{-3}	[m]
Width	13.00×10^{-3}	[m]
Mass	20×10^{-3}	[kg]
Inertia	$\begin{bmatrix} 5.8626 \times 10^{-6} & 0 & 0 \\ 0 & 4.2093 \times 10^{-7} & 0 \\ 0 & 0 & 5.8802 \times 10^{-6} \end{bmatrix}$	[kg.m2]

It allows only pitch axis movements and restricts the motion between certain limits. The second one, which is shown in Figure B.4, is designed for three-axis stabilization platform and allows roll-yaw-pitch axes movements. The dimension, mass and inertia properties of the two types support columns are given in Table B.3 and in Table B.4, respectively.

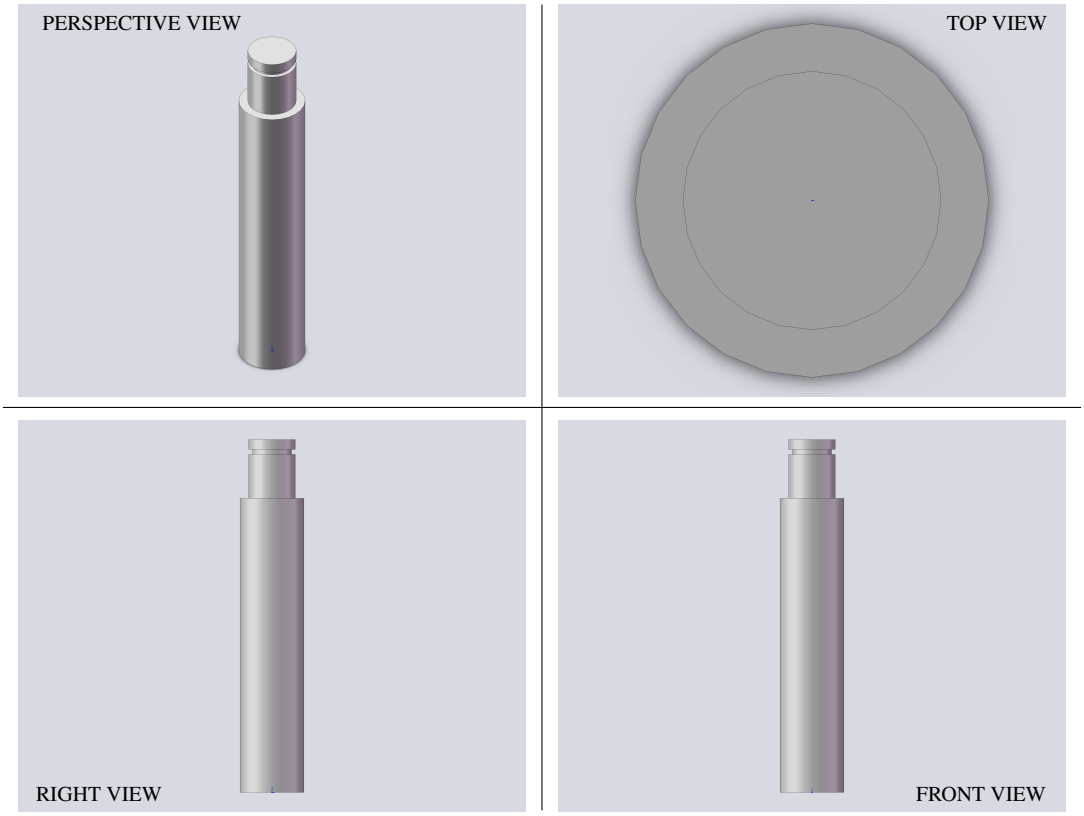


Figure B.4: The CAD model of the support column for three-axis stabilization platform

TableB.4: Dimension, mass and inertia properties of the support column for three-axis stabilization platform

Name	**Value**	**Unit**
Length	13.04×10^{-3}	[m]
Height	72.03×10^{-3}	[m]
Width	13.04×10^{-3}	[m]
Mass	24×10^{-3}	[kg]
Inertia	$\begin{bmatrix} 9.3841 \times 10^{-6} & 0 & 0 \\ 0 & 4.8960 \times 10^{-7} & 0 \\ 0 & 0 & 9.3841 \times 10^{-6} \end{bmatrix}$	[kg.m2]

B.1.4 Bearing Housing

The bearing housing is designed for screwing and fastening the vertical support column and top platform. Two types of bearing housing are manufactured as shown in Figure B.5 and Figure B.6, respectively. Depending on the included joint, it provides the desired degree-of-freedom (DOF), which determines the axis of movement of the top platform.

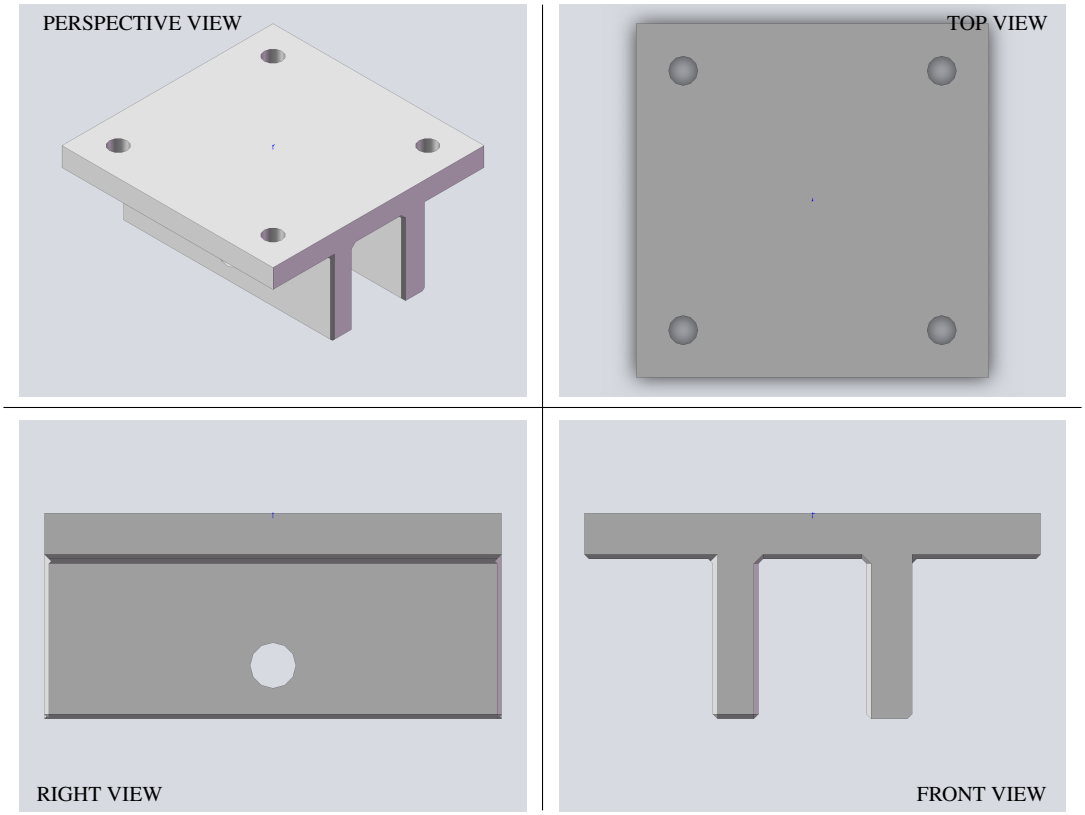


Figure B.5: The CAD model of the bearing housing for single-axis stabilization platform

TableB.5: Dimension, mass and inertia properties of the bearing housing for single-axis stabilization platform

Name	**Value**	**Unit**
Length	30.15×10^{-3}	[m]
Height	13.52×10^{-3}	[m]
Width	30.15×10^{-3}	[m]
Mass	11×10^{-3}	[kg]
Inertia	$\begin{bmatrix} 7.6259 \times 10^{-7} & -6.5820 \times 10^{-11} & -1.1095 \times 10^{-10} \\ -6.5820 \times 10^{-11} & 1.4208 \times 10^{-6} & 5.5035 \times 10^{-11} \\ -1.1095 \times 10^{-10} & 5.5035 \times 10^{-11} & 9.9005 \times 10^{-7} \end{bmatrix}$	[kg.m2]

The first one is designed for single-axis stabilization platform. It connects the support column and top platform with a revolute joint, which allows only single axis movements. The second one, designed for three-axis stabilization platform, makes the connection with a spherical joint and allows three axis roll-yaw-pitch movements. The dimension, mass and inertia properties of the two types bearing housing are given in Table B.5 and in Table B.6, respectively.

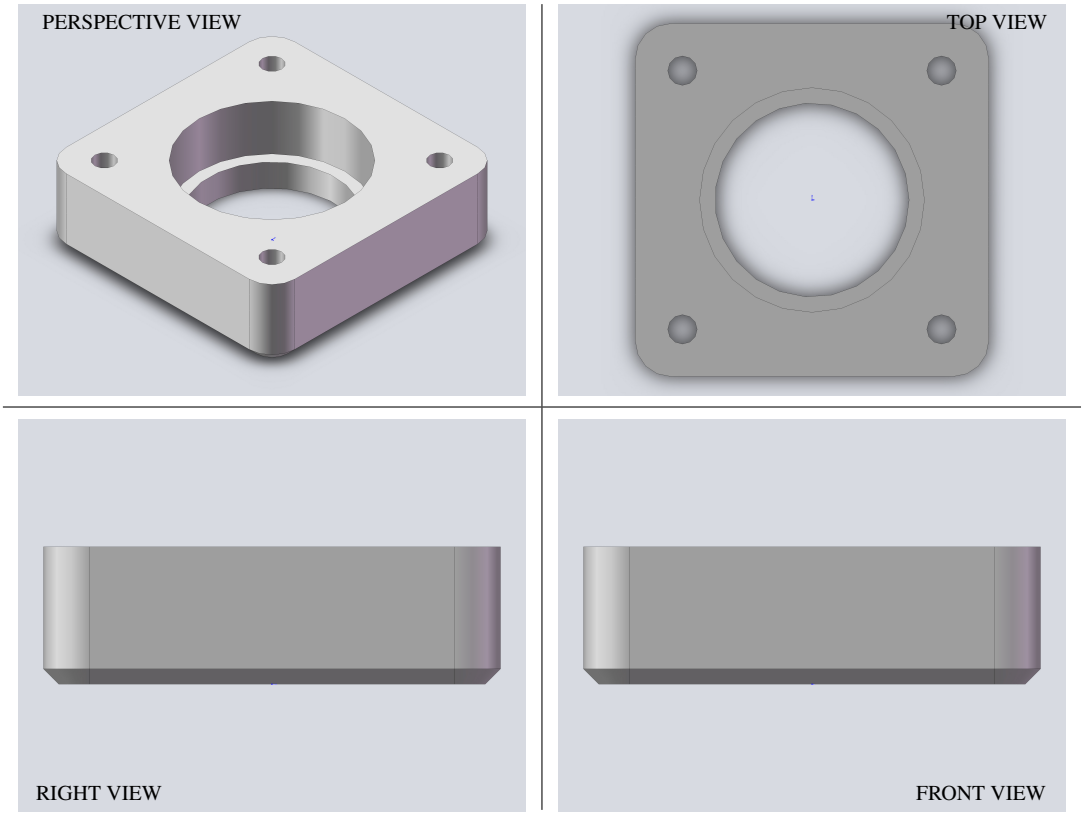


Figure B.6: The CAD model of the bearing housing for three-axis stabilization platform

TableB.6: Dimension, mass and inertia properties of the bearing housing for three-axis stabilization platform

Name	**Value**	**Unit**
Length	30.00×10^{-3}	[m]
Height	8.94×10^{-3}	[m]
Width	30.00×10^{-3}	[m]
Mass	17×10^{-3}	[kg]
Inertia	$\begin{bmatrix} 1.7212 \times 10^{-6} & 0 & 0 \\ 0 & 3.2128 \times 10^{-6} & 0 \\ 0 & 0 & 1.7212 \times 10^{-6} \end{bmatrix}$	[kg.m2]

B.1.5 Cylindrical Housing

The cylindrical housing, shown in Figure B.7, is designed as a holder for the revolute and spherical joints. Hence, it can be used as a part of the single-axis, as well as three-axis stabilization platforms equipped with revolute and spherical joints. The dimension, mass and inertia properties of the cylindrical housing is given in Table B.7.

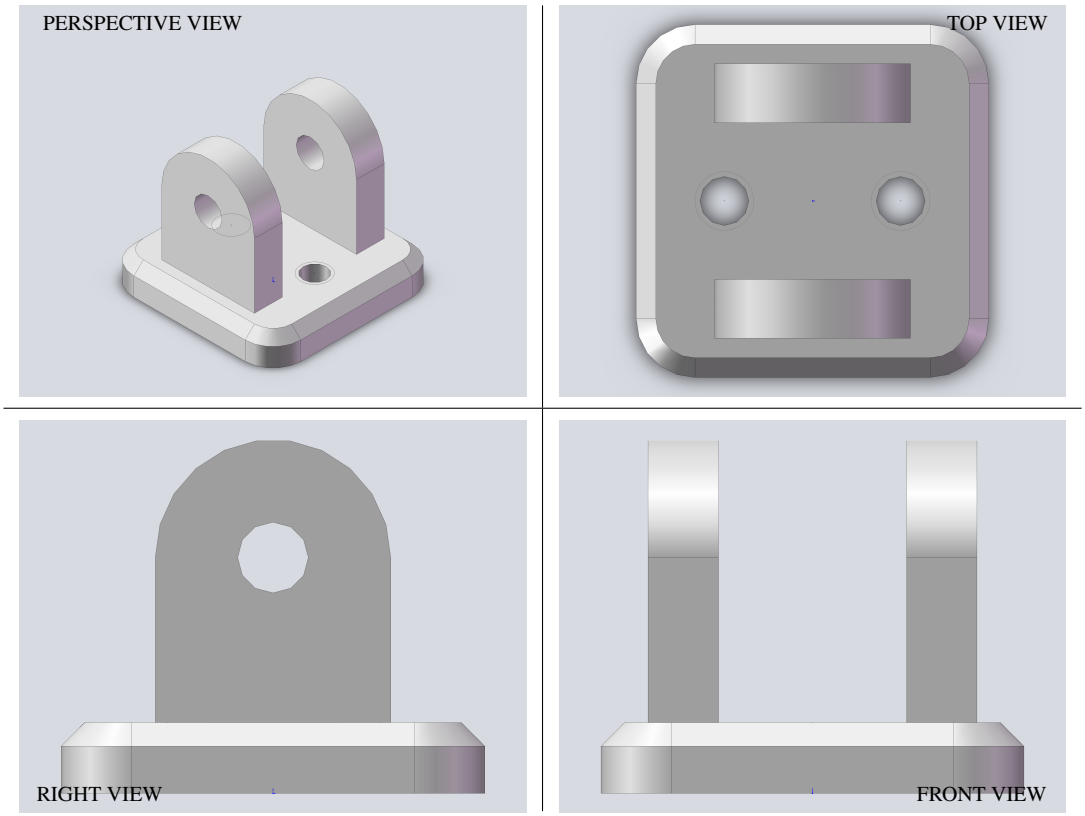


Figure B.7: The CAD model of the cylindrical housing

TableB.7: Dimension, mass and inertia properties of the cylindrical housing

Name	**Value**	**Unit**
Length	18.02×10^{-3}	[m]
Height	14.91×10^{-3}	[m]
Width	18.02×10^{-3}	[m]
Mass	3.9×10^{-3}	[kg]
Inertia	$\begin{bmatrix} 1.7433 \times 10^{-7} & 0 & 0 \\ 0 & 1.8067 \times 10^{-7} & 0 \\ 0 & 0 & 1.3623 \times 10^{-7} \end{bmatrix}$	[kg.m2]

B.1.6 Motor Connector Support

The motor connector support is designed for fastening the motor body to the revolute, spherical or universal joints depending on the layout. It is a common components used in every configurations. The CAD model of the motor connector support is shown in Figure B.8. Also, the dimension, mass and inertia properties are given in Table B.8.

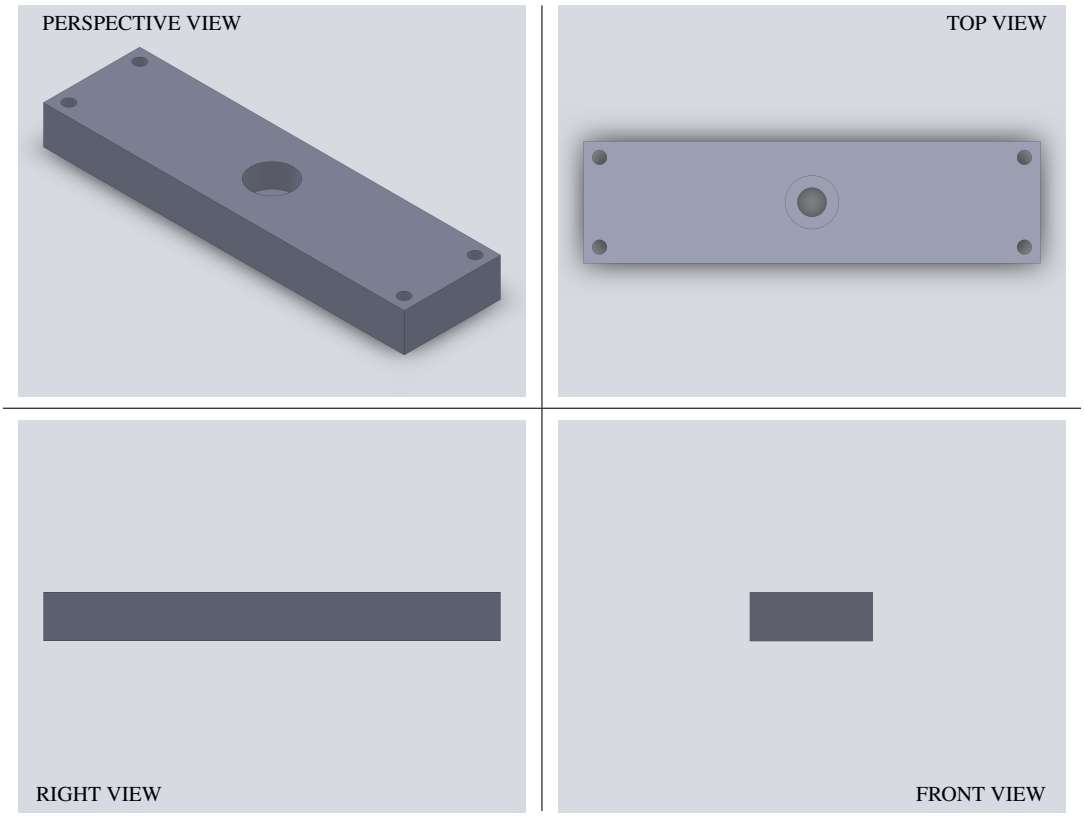


Figure B.8: The CAD model of the motor connector support

TableB.8: Dimension, mass and inertia properties of the motor connector support

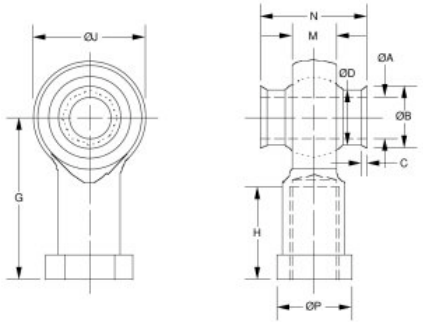
Name	**Value**	**Unit**
Length	46.88×10^{-3}	[m]
Height	4.99×10^{-3}	[m]
Width	12.57×10^{-3}	[m]
Mass	8.00×10^{-3}	[kg]
Inertia	$\begin{bmatrix} 1.2077 \times 10^{-7} & 0 & 0 \\ 0 & 1.5526 \times 10^{-6} & 0 \\ 0 & 0 & 1.4651 \times 10^{-6} \end{bmatrix}$	[kg.m2]

B.2 Spherical Rod End Joint

MWPF (Booted, Nylon Raced, Trunion Rod Ends)

Steel Housing, Plated Zinc Yellow
Nylon Bearing Race



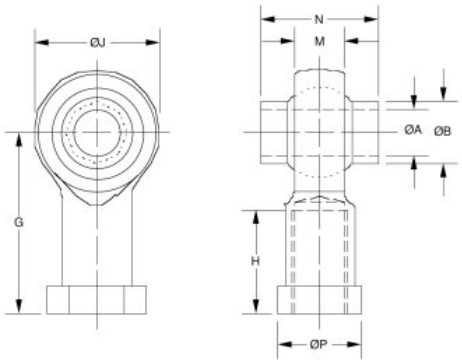


Part Number	Thread	A	B	C	D	G	H	J	M	N	P
MWPF-8T	1/2-20 RH	0.395	0.63	0.06	0.55	1.631	0.94	1.125	0.442	1.062	0.746
MWPFL-8T	1/2-20 LH	0.395	0.63	0.06	0.55	1.631	0.94	1.125	0.442	1.062	0.746

MWSF (Booted, Steel Raced, Trunion Rod Ends)

Steel Housing, Plated Zinc Yellow





Part Number	Thread	A	B	G	H	J	M	N	P
MWSF-6T	3/8-24 RH	0.376	0.53	1.631	0.94	1.125	0.375	1.065	0.632
MWSFL-6T	3/8-24 LH	0.376	0.53	1.631	0.94	1.125	0.375	1.065	0.632

B.3 Universal Joint

S METRIC — Miniature Universal Joints

■ SINGLE & DOUBLE JOINT ■ ZERO BACKLASH

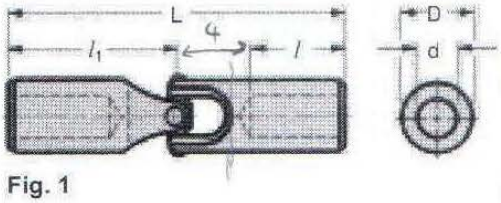
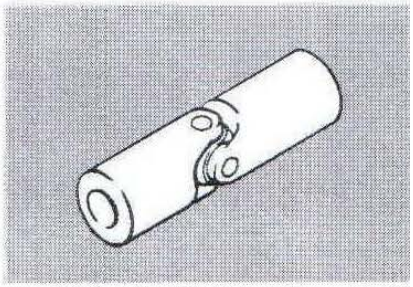


Fig. 1



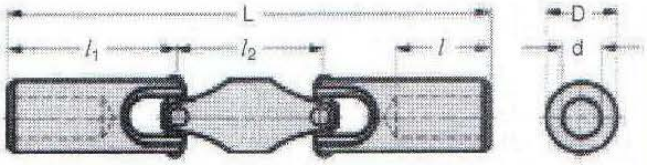
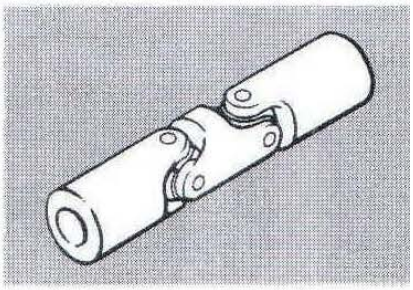


Fig. 2



FEATURES:

- Zero backlash through use of precision balls and burnished sockets
- Sealed-in lubrication
- Lightweight
- Low inertia

SPECIFICATIONS:

Torque Ratings – For static conditions with in-line loading, maximum load being one which causes joint to yield.

Max. Operating Angle: **Fig. 1** – 30° @ 4000 rpm
Fig. 2 – 60° @ 4000 rpm

MATERIAL:

Body – 303 Stainless Steel
Balls – 440 Stainless Steel

LUBRICATION:

Dry Film Plus Oil Suspension

Catalog Number	D	d Bore +0.013	l Bore Depth	l_1 End Unit	l_2 Center	L Length	Torque N•cm
• FIG. 1		SINGLE JOINT					
S57PY5MSU0005 S57PY5MSU0205 S57PY5MSU0305	4.76	Solid 2.5 3	— 8.7 8.7	12.7	—	25.4	11
S57PY5MSU0407 S57PY5MSU0507	7.15	4 5	11.1	17.5	—	34.9	45
S57PY5MSU0510 S57PY5MSU0610	9.53	5 6	12.7	22.2	—	44.5	180
• FIG. 2		DOUBLE JOINT					
S57PY5MDU0205 S57PY5MDU0305	4.76	2.5 3	8.7	12.7	12.7	38.1	11
S57PY5MDU0407 S57PY5MDU0507	7.15	4 5	11.1	17.5	15.9	50.8	45
S57PY5MDU0510 S57PY5MDU0610	9.53	5 6	12.7	22.2	19.1	63.5	180

Set screws available as special order.

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Öğücü, M. Orkun
Nationality: Turkish (TC)
Date and Place of Birth: 22.01.1980, Gaziantep
Marital Status: Single
Phone: 0-532-2607820

EDUCATION

Degree	Institution	Year of Graduation
M.S.	Istanbul Technical University - Control Engineering	2006
B.S.	Istanbul Technical University - Electrical Engineering	2004
High School	Gaziantep Anatolian High School	1998

PROFESSIONAL EXPERIENCE

Year	Place	Enrollment
2011 - 2013	Middle East Technical University - Robotics Laboratory	Project Assistant