TRANSMISSION LINE MODELLING APPLIED TO NON-LINEAR CONTROL SYSTEMS

VEIMAR YOBANY MORENO CASTANEDA

A thesis submitted to the University of Huddersfield in partial fulfilment of the requirements for the degree of Doctor of Philosophy

The University of Huddersfield

September 2006

ABSTRACT

This study presents a novel application of the Transmission Line Matrix Method (TLM) for the modelling of the dynamic behaviour of non-linear hybrid systems; and the application of a novel Wavelet algorithm for the determination of natural frequencies and damping coefficients for the CNC machine tools feed drives. The considered feed drives are non-linear hybrid systems where the controller commands the movement of a worktable linked to a motor through a ball-screw.

The application of the TLM technique to the modelling of hybrid systems implies the dividing of the screw shaft into a number of identical elements in order to achieve the synchronisation of events in the simulation, and to produce acceptable resolution according to the maximum frequency of interest. This entails considerable computing effort when small time steps are used in the simulation.

This research presents the extension of that work to the development of a new TLM modelling approach denominated *The Modified Transmission Line Method*, which inherits the modelling advantages of the TLM technique without compromising the model response by the sample time.

Generally, the analysis of torsional and axial dynamic effects on a shaft implies the development of torsional and axial models simulated independently. This study presents a new approach for the modelling of the screw shaft including the axial and torsional dynamics in the same model. In this regard, a procedure for the synchronisation of both axial and torsional effects is presented.

TLM models for single and two-axis models have been built. Simulation results show the accuracy of the models when comparing with measurements from the real systems.

ACKNOWLEDGMENTS

I would like to express my gratitude to my Director of Studies, Dr. Andrew Longstaff, whose expertise and assistance, added considerably to my graduate experience.

I would like to thank the members of the Engineering Control and Machine Performance Research group at The University of Huddersfield, especially Dr. Crinela Pislaru for the assistance she provided at all levels of the research project.

A very special thanks goes out to Professor Derek Ford, without whose motivation and encouragement, I would not have finished this thesis. He provided me with direction, technical support and became more of a mentor than a professor. I doubt that I will ever be able to convey my appreciation fully, but I owe him my eternal gratitude.

I would also like to thank my parents, Eliceo and Adelia, for the support they provide me through my entire life.

I specially acknowledge my wife and best friend, Orcil Gehovanna, without whose love, encouragement and support I would not have reached this goal.

CONTENTS

ABSTRACT ii
ACKNOWLEDGMENTS iii
CONTENTS iv
LIST OF FIGURES xii
LIST OF TABLES xxiii
LIST OF ABBREVIATIONS xxvi
NOMENCLATURE xxvii

CHAPTER 1 INTRODUCTION 1
1.1 Aim and Objectives 5
1.2 Thesis Outline 5

CHAPTER 2 LITERATURE REVIEW 7
2.1 Modelling and Simulation of Feed Drives from CNC Machine Tools 7
2.2 Transmission Line Modelling Techniques 9
2.4 Summary 15

CHAPTER 3 TRANSMISSION LINE MODELLING TECHNIQUES 17
3.1 The Analogue Transform Technique 17
3.2 The Transmission Line Matrix Method 19
3.2.1 The TLM Stub 20
3.2.2 TLM Link 22
3.2.3 Modelling Non-Linear Elements 25
3.3. Comparison Between TLM and ATT Modelling Techniques 26
3.3.1 Model Using the Analogue Transform Technique 27
3.3.2 Model Using the Transmission Line Matrix Method 30
3.3.2.1 Distributed Parameter Elements 30
3.3.2.2 Lumped Parameter Elements 31
3.3.2.3 Simulation Results 32
3.4 The Modified TLM Stub 33
3.5 Conclusions 37
3.6 Project Plan 39

CHAPTER 4 MODELLING THE ELEMENTS OF CNC MACHINE TOOL FEED DRIVES USING THE MODIFIED TRANSMISSION LINE MATRIX METHOD 40

4.1 Introduction to CNC Machine Tool Digital Drives 40

4.2 Permanent Magnet Synchronous Motor 43

4.2.1 Electrical Equations 44

4.2.2 Mechanical Equations 47

4.3 Controller Model 48

4.3.1 Interpolator 49

4.3.2 Position Controller with Velocity Feed Forward 52

4.3.3 Velocity Controller with Acceleration Feed Forward 53

4.3.4 Current Controller and PWM Generation 55

4.4 Dynamic Models for Mechanical Transmission Elements 61

4.4.1 Non-Linearities (Backlash, Friction) 61

4.4.1.1 Friction Model 62

4.4.1.2 Backlash Model 66

4.4.2 Stiffness 68

4.4.3 Bearings 69

4.4.4 Guideways and Slides 71

4.4.5 Coupling 73

4.4.6 Worktable 75

4.4.7 Nut 76

4.4.8 Interrelation Between Nut and Screw Shaft 77

4.4.9 Screw Shaft with Moving Nut 78

4.5 Transducers 81

4.6. Summary 85

CHAPTER 5 TLM MODELS FOR CNC MACHINE TOOL FEED DRIVES 86

5.1 TLM Model of the Single-Axis CNC Feed Drive 86

5.1.1 Motion Controller 87

5.1.2 Inverter and Motor (Electrical) 91

5.1.3 Motor (Mechanical) and Mechanical Transmission Elements 91

5.1.3.1 Motor Mechanical Equations and Coupling 94

5.1.3.2 Screw Shaft Torsional Model 95
5.1.3.3 Screw Shaft Axial Model 97
5.1.3.4 Screw Shaft, Nut and Table 98
5.2 Single-Axis TLM Model for a CNC Machine Tool Feed Drive 102
5.3 Implementation of Two-Axis TLM Models 104
5.4 Review 105

CHAPTER 6 MEASUREMENT TECHNIQUES APPLIED TO POSITION CONTROLLED MECHANISMS WITHIN CNC MACHINE TOOLS 106
6.1 Geometric and Load Error Measurements 107
6.1.1 Equipment Used for the Measurement of the Geometric Errors 109
6.1.2 X-Axis Geometric Errors 111
6.2 Ballbar Measurements 114
6.3 Step and Jerk Limited Response Measurements 116
6.3.1 Single- Axis Test Rig 116
6.3.2 Arrow 500 Cincinnati CNC Machine 119

CHAPTER 7 SIMULATION OF PROPOSED ONE-AXIS AND TWO-AXIS TLM MODELS 122
7.1 Implementation of the Single-Axis Model for the Test Rig in SIMULINK 123
7.1.1 Digital Controller Model 124
7.1.2 Dynamic Model of the Ball-Screw System 125
7.1.2.1 Implementation of the Pulse Propagation 126
7.1.2.2 Torsional Loop Subsystem 134
7.1.2.3 Axial Loop Subsystem 138
7.2 Validation of the Single-axis Model for the Test rig 141
7.2.1 Step Velocity Response 141
7.2.2 Jerk-limited Response 144
7.3 Implementation of Single-Axis TLM Model for Arrow 500 Machine in SIMULINK 146
7.4 Implementation and Validation of Two-Axis TLM Models 148

CHAPTER 8 IDENTIFICATION OF RESONANT FACTORS USING THE WAVELET TRANSFORM 154
8.1 Identification Methods for Modal Parameters of CNC Machine Tools 154

8.2 Transfer Function Identification for Control System Loops (Arrow 500) 158
8.3 Modal Parameters Identification Using Wavelets 160
8.4 Summary 161

CHAPTER 9 CONCLUSIONS AND FURTHER WORK 162
9.1 Contribution to Knowledge 164
9.2 Suggestions for Further work 164

REFERENCES 165

APPENDIX A: TRANSMISSION LINE MODEL 177
A.1 Differential Equation for a Transmission Line 177
A.2 Analytical Solution of the Telegrapher's Equation 178

APPENDIX B: MODELLING EXAMPLE 181
B.1 Model Using the Analogue Transform Technique 181
B.2 Model Using the Transmission Line Matrix Method 182
B.3 Analysis of the TLM Model for the Differential Term 186
B.4 Calculation of the TLM and MTLM Model Errors 187
B.5 Modified TLM Transform for Equation (3.35) 189

APPENDIX C SYNCHRONISATION BETWEEN THE TLM AXIAL AND TORSIONAL MODELS 190

APPENDIX D TORSIONAL AND AXIAL TLM MODELS 193
D.1 TLM Torsional Model 193
D.2 TLM Axial Model (Fixed/fixed Bearing Configuration Case) 201
D.3 TLM Axial Model (Fixed/supported Bearing Configuration Case) 203

APPENDIX E TEST RIG SPECIFICATIONS 204

APPENDIX F THE VMC –500 MACHINE SPECIFICATIONS 211

APPENDIX G MEASUREMENTS 219
G.1 Y-Axis Geometric Errors (Arrow 500) 219
G.2 Polynomial Coefficients Calculated from the Geometric Error Measurements (Arrow 500) 221

G.3 MATLAB Program for the Calculation of the Polynomial Coefficients 223
G.4 Step Velocity Response Measurements for the Test Rig 225
G.5 Measured Jerk-limited Motion Profiles for the Test Rig 226
G.6 Jerk-limited Velocity Response Measurements for the Test Rig 229
G.7 Jerk-limited Position Response Measurements for the Test Rig 230
G.8 Jerk-limited Velocity Response Measurements for the x-axis of the Arrow 500 232
G.9 Jerk-limited Velocity Response Measurements for the y-axis of the Arrow 500 234
G.10 Jerk-limited Position Response Measurements for the x-axis of the Arrow 500 236
G.11 Jerk-limited Position Response Measurements for the y-axis of the Arrow 500 238
G.12 MATLAB Program Used to Plot Position Measurements in Ball-Bar Format 239

APPENDIX H TEST RIG TLM MODEL IN MATLAB/ SIMULINK 240
H.1 Testrig_profile.m Program 240
H.2 Reference Signal Profiles 240
H.2.1 Jerk Limited Profile Generation 242
H.2.2 Sinusoidal Profile Generation 245
H.2.3 White Noise Profile Generation 245
H.2.4 Swept Sine Profile 246
H.2.5 Linear and Circular Interpolation 246
H.2.5.1 Linear Interpolation Routine 247
H.2.5.2 Circular Interpolation Routine 248
H.3 Y-Axis Block Parameters and Initialisation Code 249
H.4 Velocity Controller Block Initialisation Code 250

APPENDIX I STRUCTURE OF DATA FOR THE PULSE PROPAGATION ON THE AXIAL MODEL 252

APPENDIX J PARAMETERS AND INITIALISATION CODE FOR THE TORSIONAL LOOP SUBSYSTEM 257
J.1 Initialisation Code for the Torsional Loop Block 257
J.2 The PWM Generating Function 257
J.3 Initialisation Code for the wm Calculation Block 259
J.4 wht+1 Calculation and Nut Monitoring Blocks 260

APPENDIX K INITIALISATION CODE FOR THE AXIAL LOOP SUBSYSTEM 262

APPENDIX L ARROW 500 TLM MODEL IN SIMULINK 268

APPENDIX M XY GEOMETRIC ERROR CALCULATION SIMULINK BLOCK 272

APPENDIX N MEASURED FREQUENCY RESPONSE Y-AXIS ARROW 500 274

APPENDIX O IDENTIFICATION OF MODAL PARAMETERS USING THE MODIFIED MORLET WAVELET 275

O.1 The Morlet Wavelet 276

O.2 Implementation of the Wavelet Transform 277

O.3 Modal Parameters Identification Using Wavelets 279

O.3.1 Implementation of the Identification Algorithm 283

O.4 MATLAB Program for the Calculation of the CWT (Morlet Wavelet) 286

O.5 MATLAB Program Used for the Example of CWT Calculation 287

O.6 MATLAB Program Used for the Example of Modal Parameters (ζ, ω_d) Identification Using the CWT 288

O.7 MATLAB Program Used for the Automatic Modal Parameters (ζ, ω_d) Identification Using the CWT 289

O.8 MATLAB Program Used for the Calculation of the Dilatation Parameters 290

O.9 MATLAB Program Used for the identification of damping factors and resonant frequencies, x-axis mechanical frequency response 292

O.10 MATLAB Program Used for the identification of damping factors and resonant frequencies, x-axis control system frequency response 292

PAGES

MISSING

IN

ORIGINAL

LIST OF FIGURES

Figure 3.1 General series representation of a distributed-lumped parameter system [19] 17
Figure 3.2 General modules for the termination element [55] 18
Figure 3.3 Impedance module for a distributed parameter element [60] 19
Figure 3.4 Admittance module for a distributed parameter element [60] 19
Figure 3.5 TLM stub [25] 20
Figure 3.6 The TLM scattering algorithm [30] 21
Figure 3.7 TLM Stub block diagram 21
Figure 3.7a Time domain 21
Figure 3.7b Z-domain 21
Figure 3.8 TLM link [25] 23
Figure 3.9 TLM link block diagram (time domain) 23
Figure 3.10 TLM link block diagram (z-domain) 24
Figure 3.11 Capacitor and inductor coupled circuits 24
Figure 3.12 TLM Model for a non-linear inductor [35] 26
Figure 3.13 TLM Model for a non-linear capacitor [36] 26
Figure 3.14 Rotor shell arrangement 27
Figure 3.15 Set of variables considered for the rotor shell assembly 27
Figure 3.16 ATT Model for the rotor shell assembly 27
Figure 3.17 Results for the established angular velocities (ATT Model) 29
Figure 3.17a ω_1 (blue line) and ω_2 (green line) 29
Figure 3.17b ω_3 (blue line) and ω_4 (green line) 29
Figure 3.18 Results for the established torques (ATT Model) 29
Figure 3.18a T_1 (blue line) and T_2 (green line) 29
Figure 3.18b T_3 (blue line) and T_4 (green line) 29
Figure 3.19 Simulation results for a slight change in the propagation time 29
Figure 3.19a $\tau_2 = 5x10^{-3}$ ms 29
Figure 3.19b $\tau_2 = 4.996x10^{-3}$ ms 29
Figure 3.20 Rotor arrangement divided into sections/segments 31
Figure 3.21 TLM model for the rotor arrangement 31
Figure 3.22 TLM models for the lumped parameter elements 32
Figure 3.23 Simulation results for the rotor shell assembly TLM model 32
Figure 3.24 Modelling error at various sample rates for the TLM model of equation (3.35) 33

Figure 3.25 Modelling error for the modified TLM model of equation (3.35) 35
Figure 3.26 TLM and modified TLM modelling error comparison 36
Figure 3.27 Simulation results for the rotor shell assembly (modified TLM stub model) 37
Figure 4.1 Conventional CNC feed drive [63] 41
Figure 4.2 Block diagram of a CNC feed drive [64] 41
Figure 4.3 Typical analogue drive [65] 42
Figure 4.4 Typical digital drive 42
Figure 4.5 Two-pole stator winding [67] 43
Figure 4.6 Three-phase motor with four magnet poles (Two pole pair) [67] 44
Figure 4.7 Stator current vector 45
Figure 4.8 Clarke transformation ($a, b, c \rightarrow \alpha, \beta$) 45
Figure 4.9 Inverse Park transformation ($\alpha, \beta \rightarrow d, q$) 46
Figure 4.10 Block diagram of the electrical equations of the motor 47
Figure 4.11 TLM model of the mechanical dynamics of a motor 48
Figure 4.12 Cascade control with velocity and acceleration feed forward [71] 48
Figure 4.13 Jerk-limited velocity profile generation [73] 49
Figure 4.14 Block diagram of the interpolator [74] 51
Figure 4.15 Two-axis linear movement [75] 51
Figure 4.16 Two-axis circular movement [76] 52
Figure 4.17 Block diagram of the position controller with velocity feed forward [77] 53
Figure 4.18 Block diagram of the velocity controller with acceleration feed forward [77] 54
Figure 4.19 Space vector current control structure [79] 56
Figure 4.20 i_q current controller 56
Figure 4.21 Power bridge (Inverter) [79] 57
Figure 4.22 Voltage base vectors 58
Figure 4.23 Projection of the reference voltage vector 58
Figure 4.24 PWM signal for a reference vector in sector 3 59
Figure 4.25 PWM generator model 61
Figure 4.26 Inverter model 61
Figure 4.27 Stick-slip friction laws [82] 62
Figure 4.27a Stribeck curve 62
Figure 4.27b Friction curve including static friction 62
Figure 4.28 Karnopp model of stick–slip friction [86] 63
Figure 4.29 Linear movement of a rigid body 63
Figure 4.30 Block diagram for the calculation of the stick-slip friction [86] 64
Figure 4.31 TLM representation of frictional force 66

Figure 4.32 Input signal, hysteric backlash and output waveform [89] 66
Figure 4.33 Input–output characteristic of backlash [90] 67
Figure 4.34 Four possible situations for the backlash model [90] 67
Figure 4.35 Fixed/supported bearing configuration [91] 68
Figure 4.36 Fixed/fixed bearing configuration [91] 68
Figure 4.37 Spring contact representation of the torsional stiffness of a feed drive [91] 69
Figure 4.38 Supporting bearing- mounting [92] 69
Figure 4.39 Stiffness representation for a bearing mounting 70
Figure 4.40 TLM model for the bearing mounting stiffness 71
Figure 4.41 Rolling contact guideway [95] 72
Figure 4.42 Relationship between imposed load ratio and friction coefficient [96] 72
Figure 4.43 Forces acting on the carriage block of a guideway [97] 73
Figure 4.44 Coupling [98] 73
Figure 4.45 Representation of the coupling stiffness 74
Figure 4.46 TLM model for coupling 74
Figure 4.47 Linear movement of the worktable 75
Figure 4.48 TLM model of the worktable movement 75
Figure 4.49 Representation of the nut rigidity 76
Figure 4.50 TLM model for the nut stiffness 76
Figure 4.51 TLM model for the nut with pretension 77
Figure 4.52 Ball screw arrangement 78
Figure 4.53 Screw shaft divided into h_t sections 79
Figure 4.54 Torsional and axial TLM models for a ball screw 82
Figure 4.54a TLM model of the torsional effect 82
Figure 4.54b TLM model of the axial effect (fixed/fixed bearing configuration) 82
Figure 4.54c TLM model of the axial effect (fixed/supported bearing configuration 82
Figure 4.55 Sin/cos incremental encoders [102] 83
Figure 4.55a Rotary encoder 83
Figure 4.55b Linear encoder 83
Figure 4.55c Signal generation 83
Figure 5.1 Bridgeport test rig 86
Figure 5.2 Test rig block diagram 87
Figure 5.3 TNC426PB Block Diagram [77] 88
Figure 5.4 Motion controller block diagram 89
Figure 5.5 Transposed direct-form II structure [104] 90
Figure 5.6 PWM generator, inverter and motor block diagram 92

Figure 5.7 PWM generator, inverter and motor reduced block diagram 93
Figure 5.8 TLM model for the motor (mechanical) and the coupling 94
Figure 5.9 Reduced TLM model for the motor (Mechanical) and coupling 95
Figure 5.9a Reduced model 95
Figure 5.9b Thevenin equivalent circuit 95
Figure 5.10 Pulses propagation model for the screw shaft torsional model with moving nut 96
Figure 5.10a Without nut 96
Figure 5.10b *Zone 2* including the nut 96
Figure 5.11 Section f_b of the torsional model 97
Figure 5.12 Section h_t of the torsional model 97
Figure 5.13 Pulses propagation model for the screw shaft axial model with moving nut 98
Figure 5.13a Without nut 98
Figure 5.13b Including the nut 98
Figure 5.14 TLM model of the connection between nut and screw shaft 99
Figure 5.14a Torsional model connection 99
Figure 5.14b Axial model connection 99
Figure 5.15 Equivalent model for the connection between nut and screw shaft 99
Figure 5.16 Backlash model 101
Figure 5.16a Non-contact 101
Figure 5.16b In contact 101
Figure 5.17 TLM model when the screw shaft is not in contact with the nut 101
Figure 5.18 Cincinnati Machine Arrow Series 2 VMC-500 102
Figure 5.19 SINUMERIK 840D Configuration [105] 103
Figure 5.20 Block diagram Siemens controller [103] 104
Figure 5.21 Two-axis feed drive 105
Figure 5.21a X-Y axis feed drive 105
Figure 5.21b Block diagram 105
Figure 6.1 Geometric errors for a machine tool slide [88] 107
Figure 6.2 Laser interferometer measurement [111] 109
Figure 6.3 Granite artefacts [107] 110
Figure 6.4 Precision electronic level [107] 110
Figure 6.5 Ballbar system (Renishaw) [112] 111
Figure 6.6 Ball-bar tests for 360 and 180 degrees [112] 111
Figure 6.7 Origin of the Cartesian coordinate system for the machine's workspace 112
Figure 6.8 X-axis linear positioning error 112
Figure 6.9 X-axis straightness error in the y-axis direction 113

Figure 6.10 X-axis straightness error in the z-axis direction 113
Figure 6.11 X-axis rotation about the z-axis 113
Figure 6.12 Measured ball bar plot 115
Figure 6.13 Test rig velocity step response measurements (500 mm/min) 117
Figure 6.14 Jerk-limited profiles (displacement = 10mm, feedrate =500 mm/min) 118
Figure 6.15 Jerk-limited axis response (displacement = 10mm, feed rate =500 mm/min) 118
Figure 6.16 Position control loop signals (displacement = 10mm, feed rate =500 mm/min) 119
Figure 6.17 Jerk-limited velocity response (x-axis Arrow 500) 120
Figure 6.18 Jerk-limited position response (y-axis Arrow 500) 120
Figure 6.19 Signals measured for the ball-bar test (run1) 121
Figure 6.20 Signals measured for the ball-bar test (run2) 121
Figure 7.1 Block diagram for the test rig single axis model 123
Figure 7.2 Implementation in SIMULINK of the test rig single axis model 124
Figure 7.3 Block diagram of the position controller model in SIMULINK 124
Figure 7.4 Block diagram of the velocity controller model in SIMULINK 125
Figure 7.5 Block diagram of the PID controller model in SIMULINK 125
Figure 7.6 Block diagram of the current controller model in SIMULINK 125
Figure 7.7 Array used to simulate the first zone of the torsional model 126
Figure 7.8 First zone array after a pulse propagation (torsional model) 127
Figure 7.9 Array used to simulate the second zone of the torsional model 128
Figure 7.10 Second zone array including the moving nut (torsional model) 130
Figure 7.10a Circular list divided in two loops 130
Figure 7.10b Mapping of the two loops on the matrix *listM* 130
Figure 7.11 Mapping of the two loops on the matrix *listM* after two pulse propagations 130
Figure 7.12 Second zone array including the moving nut (nut moves to the right) 131
Figure 7.12a Circular list divided in two loops 131
Figure 7.12b Mapping of the two loops on the matrix *listM* 131
Figure 7.13 Second zone array including the moving nut (nut moves to the left) 133
Figure 7.13a Circular list divided in two loops 133
Figure 7.13b Mapping of the two loops on the matrix *listM* 133
Figure 7.14 Block model for the subsystem Torsional loop 135
Figure 7.15 Block model for the inverter and motor in SIMULINK 135
Figure 7.16 Torsional model block in SIMULINK 136
Figure 7.17 Coupling and motor mechanical model block diagram in SIMULINK 136
Figure 7.18 w_{fb+1} calculation block in SIMULINK 137
Figure 7.19 w_{n+1} calculation block in SIMULINK 137

Figure 7.20 Nut position monitoring block in SIMULINK (torsional loop) 138
Figure 7.21 The axial loop subsystem block model in SIMULINK 138
Figure 7.22 v_{1a} calculation block in SIMULINK 139
Figure 7.23 v_{ha+1} calculation block in SIMULINK 139
Figure 7.24 The v_{na+1} calculation block in SIMULINK 140
Figure 7.25 Nut position monitoring block in SIMULINK (axial loop) 140
Figure 7.26 Case 1 block for the nut position monitoring in the axial loop subsystem 140
Figure 7.27 The F0 calculation block in SIMULINK 141
Figure 7.28 The If/Else action blocks in the v_{na+1} calculation model 141
Figure 7.29 Set up for the validation of the step velocity response 142
Figure 7.30 Step velocity response validation for the TLM test rig model (500 mm/min) 142
Figure 7.31 Step velocity response validation for the TLM test rig model (1000mm/min) 142
Figure 7.32 Position control loop validation set-up for jerk limited response 143
Figure 7.33 Comparison between measured and simulated actual position 143
Figure 7.33a Actual position 143
Figure 7.33b Position error 143
Figure 7.34 Velocity control loop validation set-up for jerk limited demand 144
Figure 7.35 Comparison between measured and simulated actual velocity 145
Figure 7.35a Reference velocity 145
Figure 7.35b Zoom acceleration zone 145
Figure 7.36 Experimental and simulated reference current (5000 mm/min) 145
Figure 7.37 Model error for the velocity controller 146
Figure 7.37a Error at 5000 mm/min 146
Figure 7.37b Error at 40000 mm/min 146
Figure 7.38 Validation position response - arrow (fr = 500 mm/min, d= 10 mm) 146
Figure 7.39 X-axis position model response to a sinusoidal demand 147
Figure 7.40 Y-axis position model response to a sinusoidal demand 147
Figure 7.41 Arrow 500 two-axis TLM model 148
Figure 7.42 The block *input profiles* (sample time: 4 ms) 148
Figure 7.43 Two-axis TLM model position response 149
Figure 7.44 Two-axis simulated actual position signals in ball-bar format 149
Figure 7.45 Two-axis model including the effect of geometric and load errors 150
Figure 7.46 SIMULINK implementation of the two-axis geometric error equations 150
Figure 7.47 Block diagram for the x-axis geometric error calculation 150
Figure 7.48 X-axis geometric error calculation (case 1: forward) 151
Figure 7.49 The scaling block 151

Figure 7.50 Two-axis TLM model position response with geometric errors 152
Figure 7.51 Simulated Ball-bar plot 152
Figure 8.1 X-axis controlled frequency response 159
Figure 8.2 X-axis mechanical frequency response 160
Figure A.1. A transmission line [139] 177
Figure A.2. Element of the transmission line of length Δx [140] 177
Figure B.1 ATT Rotor shell model in SIMULINK 181
Figure B.2 Motor & front Bearing block (ATT Model) 182
Figure B.3 Shaft 1 and shaft 2 block (ATT Model) 182
Figure B.4 Rotor shell block (ATT Model) 182
Figure B.5 Rear bearing block (ATT Model) 183
Figure B.6 Rotor shell TLM model 184
Figure B.7 TLM Rotor shell model in SIMULINK 184
Figure B.8 ω_1 calculation block (TLM model) 185
Figure B.9 ω_{21} calculation block (TLM model) 185
Figure B.10 Rotor shell propagation block (TLM model) 186
Figure B.11 ω_3 calculation block (TLM model) 186
Figure B.12 ω_4 calculation block (TLM model) 186
Figure B.13 Motor block diagram model in SIMULINK 187
Figure B.14 Subsystem TLM model 187
Figure B.15 error_cal.mdl SIMULINK model 188
Figure B.16 ω_1 calculation block (modified TLM model) 189
Figure D.1 TLM model of a shaft divided into *h* sections (case *a*) 194
Figure D.2 Graphic representation of the pulses propagation for the case *a* 195
Figure D.3 TLM model of a shaft including the bearings friction (case *b*) 196
Figure D.4 Graphic representation of the pulses propagation for case *b* 197
Figure D.5 TLM model of a shaft divided into eight sections including the bearings friction and moving nut (case *c*) 198
Figure D.6 Graphic representation of the pulse propagation for case *c* (nut on section four) 199
Figure D.7 TLM model for case *c* (nut on section five) 199
Figure D.8 TLM model for the screw shaft axial dynamics (fixed/fixed bearing configuration) 200
Figure D.9 TLM model for the front bearing mounting (Fixed case) 201
Figure D.10 TLM reduce model for the front bearing mounting (Fixed case) 202
Figure D.11 TLM model for the rear bearing mounting (Fixed case) 202
Figure D.12 TLM model for the rear bearing mounting (Supported case) 203
Figure E.1 Block diagram of the TNC 426PB [87] 206

Figure F.1 Velocity and current control block diagram (SIMODRIVE 611) [122] 213
Figure G.1 X about X angular error 219
Figure G.2 X about Y angular error 219
Figure G.3 Y-axis linear error 219
Figure G.4 Y in X straightness error 220
Figure G.5 Y in Z straightness error 220
Figure G.6 Y in X angular error 220
Figure G.7 Z-axis linear error 220
Figure G.8 Z in X straightness error 221
Figure G.9 Z in Y straightness error 221
Figure G.10 Step velocity response (1000 mm/min) 225
Figure G.11 Step velocity response (5000 mm/min) 225
Figure G.12 Step velocity response (10000 mm/min) 225
Figure G.13 Step velocity response (20000 mm/min) 226
Figure G.14 Step velocity response (30000 mm/min) 226
Figure G.15 Displacement = 20mm, feedrate = 1000 mm/min 226
Figure G.16 Displacement = 100mm, feedrate = 5000 mm/min 227
Figure G.17 Displacement = 200mm, feedrate = 10000 mm/min 227
Figure G.18 Displacement = 400mm, feedrate = 20000 mm/min 228
Figure G.19 Displacement = 400mm, feedrate = 40000 mm/min 228
Figure G.20 Displacement = 20mm, feedrate = 1000 mm/min 229
Figure G.21 Displacement = 100mm, feedrate = 5000 mm/min 229
Figure G.22 Displacement = 200mm, feedrate = 10000 mm/min 229
Figure G.23 Displacement = 400mm, feedrate = 20000 mm/min 230
Figure G.24 Displacement = 400mm, feedrate = 40000 mm/min 230
Figure G.25 Displacement = 20mm, feedrate = 1000 mm/min 230
Figure G.26 Displacement = 100mm, feedrate = 5000 mm/min 231
Figure G.27 Displacement = 200mm, feedrate = 10000 mm/min 231
Figure G.28 Displacement = 400mm, feedrate = 20000 mm/min 231
Figure G.29 Displacement = 400mm, feedrate = 40000 mm/min 232
Figure G.30 Feedrate = 1000 mm/min (x-axis) 232
Figure G.31 Feedrate = 5000 mm/min (x-axis) 232
Figure G.32 Feedrate = 10000 mm/min (x-axis) 233
Figure G.33 Feedrate = 20000 mm/min (x-axis) 233
Figure G.34 Feedrate = 30000 mm/min (x-axis) 233
Figure G.35 Feedrate = 500 mm/min (y-axis) 234

Figure G.36 Feedrate = 1000 mm/min (y-axis) 234
Figure G.37 Feedrate = 5000 mm/min (y-axis) 234
Figure G.38 Feedrate = 10000 mm/min (y-axis) 235
Figure G.39 Feedrate = 20000 mm/min (y-axis) 235
Figure G.40 Feedrate = 30000 mm/min (y-axis) 235
Figure G.41 Displacement = 10mm, feedrate = 500 mm/min (x-axis) 236
Figure G.42 Displacement = 20mm, feedrate = 1000 mm/min (x-axis) 236
Figure G.43 Displacement = 100mm, feedrate = 5000 mm/min (x-axis) 236
Figure G.44 Displacement = 200 mm, feedrate = 10000 mm/min (x-axis) 237
Figure G.45 Displacement = 400 mm, feedrate = 20000 mm/min (x-axis) 237
Figure G.46 Displacement = 400 mm, feedrate = 30000 mm/min (x-axis) 237
Figure G.47 Displacement = 20 mm, feedrate = 1000 mm/min (y-axis) 238
Figure G.48 Displacement = 100 mm, feedrate = 5000 mm/min (y-axis) 238
Figure G.49 Displacement = 200 mm, feedrate = 10000 mm/min (y-axis) 238
Figure G.50 Displacement = 400 mm, feedrate = 20000 mm/min (y-axis) 239
Figure G.51 Displacement = 400 mm, feedrate = 30000 mm/min (y-axis) 239
Figure H.1 Jerk-limited position and velocity profiles 240
Figure H.1a Position profile 240
Figure H.1b Velocity profile 240
Figure H.1c Acceleration profile 240
Figure H.1d Jerk profile 240
Figure H.2 Step profile 241
Figure H.3 Sinusoidal profile 241
Figure H.4 White noise profile 242
Figure H.4a White noise signal 242
Figure H.4b Frequency spectrum 242
Figure H.5 Swept sine profile 242
Figure H.5a Generated swept sine signal 242
Figure H.5b Frequency spectrum 242
Figure H.6 Linear interpolation profiles 246
Figure H.6a xy_reference_position 246
Figure H.6b xy_velocity_profile 246
Figure H.6c xy_acceleration_profile 246
Figure H.7 Circular interpolation profiles 247
Figure H.7a xy_reference_position 247
Figure H.7b xy_velocity_profile 247

Figure H.7c xy_acceleration_profile 247
Figure H.8 Y-axis block parameters 249
Figure H.9 Velocity controller block parameters 251
Figure I.1 Array used to simulate the second zone of the axial model 252
Figure I.2 Second zone array including the moving nut (axial model) 254
Figure I.2a Mapping of the two loops on the matrix *listA* 254
Figure I.2b Circular list divided in two loops 254
Figure I.3 Mapping of the two loops on the matrix *listA* after two pulse propagations 254
Figure J.1 w_{ht+1} Calculation block 260
Figure J.2 Case 1 block (nut position monitoring) 260
Figure J.3 Case –1 block (nut position monitoring) 260
Figure K.1 Validation position response - test rig (fr = 1000 mm/min, d= 20 mm) 262
Figure K.2 Validation velocity response - test rig (fr = 1000 mm/min, d= 20 mm) 262
Figure K.3 Frequency response velocity loop - test rig (fr = 1000 mm/min, d= 20 mm) 263
Figure K.4 Validation position response - test rig (fr = 10000 mm/min, d= 200 mm) 263
Figure K.5 Validation velocity response - test rig (fr = 10000 mm/min, d= 200 mm) 264
Figure K.6 Frequency response velocity loop - test rig (fr = 10000 mm/min, d= 200m) 264
Figure K.7 Validation position response - test rig (fr = 20000 mm/min, d= 200 mm) 265
Figure K.8 Validation velocity response - test rig (fr = 20000 mm/min, d= 200 mm) 265
Figure K.9 Frequency response velocity loop - test rig (fr = 20000 mm/min, d= 200 mm) 266
Figure K.10 Validation position response - test rig (fr = 40000 mm/min, d= 200 mm) 266
Figure K.11 Validation velocity response - test rig (fr = 40000 mm/min, d= 200 mm) 267
Figure K.12 Frequency response velocity loop - test rig (fr = 40000 mm/min, d= 200 mm) 267
Figure L.1 Velocity controller block (Arrow 500) 262
Figure L.2 PI Controller block (Arrow 500) 268
Figure L.3 Current controller block (Arrow 500) 268
Figure L.4 w_{ht+1} calculation block 268
Figure L.5 Rear bearing mounting stiffness block (Arrow 500) 268
Figure L.6 Validation position response - arrow (fr = 1000 mm/min, d= 20 mm) 269
Figure L.7 Validation position response - arrow (fr = 5000 mm/min, d= 100 mm) 269
Figure L.8 Validation position response - arrow (fr = 10000 mm/min, d= 200 mm) 270
Figure L.9 Validation position response - arrow (fr = 20000 mm/min, d= 400 mm) 270
Figure L.10 Validation position response - arrow (fr = 30000 mm/min, d= 400 mm) 271
Figure M.1 X-axis geometric error calculation (Case –1 Reverse) 272
Figure M.2 Y-axis geometric error calculation 272
Figure M.3 Y-axis geometric error calculation (Case 1: Forward) 273

Figure M.4 Y-axis geometric error calculation (Case -1: Reverse) 273
Figure N.1 Y-axis controlled frequency response 274
Figure N.2 Y-axis mechanical frequency response 274
Figure O.1 $f_1(t)$ and $f_2(t)$ CWT (Morlet: $N = 2$) 277
Figure O.2 $f(t)$ CWT (Morlet: N=2) 278
Figure O.3 CWT of $f(t)$ for N=128 (Modified Morlet wavelet) 279
Figure O.4 Impulse response second-order system (equation (O.34)) 281
Figure O.5 CWT transform of the impulse response (equation (O.32)) 281
Figure O.6 Slope of the phase of the CWT transform (scale=28) 282
Figure O.7 Slope of the magnitude of the CWT transform (scale=28) 282
Figure O.8 Maximum CWT magnitude for each a row 283
Figure O.9 CWT map fields to analyse for peak verification at the scale a_i 284
Figure O.10 CWT of the x-axis controlled frequency response ($N = 8$) 285
Figure O.11 CWT of the x-axis mechanical frequency response ($N = 8$) 286

LIST OF TABLES

Table 3.1 TLM and Z transforms of integral, differential and proportional terms 22
Table 3.2 Model parameters for the system shafts 28
Table 3.3 TLM model parameters 31
Table 3.4 The modified TLM transform 35
Table 3.5 Extended TLM transform table 38
Table 4.1 Inverter output voltages 57
Table 4.2 Possible stator voltages 58
Table 4.3 Switching-state stator voltages 59
Table 4.4 Duration times of the switching-states 60
Table 6.1 Geometric error components associated with two-axis CNC Machine 109
Table 6.2 Measurements of the geometric and non-rigid errors 114
Table 6.3 Signals that can be accessed by the oscilloscope in the TNC 426PB 116
Table 6.4 Set of movements established for the validation of the test rig TLM model 117
Table 6.5 Signals that can be switched to the DAC channels 119
Table 6.6 Position loop signals 119
Table 8.1 Classical damping measurement methods [120] 155
Table 8.2 Parameters to be identified and stimuli used for this purpose 161
Table B.1 Rotor shell parameters 181
Table C.1 n_a/n_t ratio for variations of 1% in the values of G_{ss} and E_{ss} 190
Table D.1 Calculation of angular velocities and incident pulses for the TLM model of Figure D.1 194
Table D.2 Calculation of angular velocities and incident pulses for the TLM model of Figure D.3 196
Table D.3 Calculation of angular velocities and incident pulses for the TLM model of Figure D.5 198
Table E.1 Test rig specifications 204
Table E.2 Y-axis motion controller parameters [123] 205
Table E.3 Inverter technical data [122] 205
Table E.4 Motor technical data [123] 207
Table E.5 Guideways technical data [114] 207

Table E.6 Bearings technical data (Front and rear) [92] 208
Table E.7 Bearings load factors [93] 208
Table E.8 Coupling technical data [111] 209
Table E.9 Rotary encoder technical data [112] 209
Table E.10 Linear encoder technical data [112] 210
Table E.11 Ball screw technical data [109] 210
Table F.1 X-axis specifications 211
Table F.2 Y-axis specifications 211
Table F.3 Z-axis specifications 212
Table F.4 Motion controller parameters (SINUMERIK 850D) 212
Table F.5 Technical data SIMODRIVE 611D 6SN1123-1AA00-0CA1 (two axis) 212
Table F.6 Axis Parameters (SIMODRIVE 611) 214
Table F.7 Technical data x and y-axis motor 215
Table F.8 Guideways technical data 215
Table F.9 Technical data front bearings 216
Table F.10 Technical data rear bearings x and z-axis 216
Table F.11 Technical data rear bearings y-axis 216
Table F.12 Coupling technical data 217
Table F.13 Ball screw technical data 217
Table F.14 Technical data rotary encoders 217
Table F.15 Technical data linear encoders 218
Table G.1 X-axis linear positioning (Forward) 221
Table G.2 X-axis linear positioning (Reverse) 221
Table G.3 Y-axis straightness in x direction polynomial (Forward) 222
Table G.4 Y-axis straightness in x direction polynomial (Reverse) 222
Table G.5 Y-axis rotation about z-axis (Forward) 222
Table G.6 Y-axis rotation about z-axis (Reverse) 222
Table G.7 Y-axis linear positioning (Forward) 222
Table G.8 Y-axis linear positioning (Reverse) 222
Table G.9 X-axis straightness in y direction (Forward) 223
Table G.10 X-axis straightness in y direction (Reverse) 223
Table G.11 X-axis rotation about z-axis (Forward) 223
Table G.12 X-axis rotation about z-axis (Reverse) 223

Table O.1 $f(t)$ Modal parameters identified by the CWT algorithm ($N = 128$) 284
Table O.2 Identified values for resonant frequencies and damping factors using Bode diagrams and wavelet analysis (x-axis controlled frequency response) 285
Table O.3 Identified values for resonant frequencies and damping factors using Bode diagrams and wavelet analysis (x-axis mechanical frequency response) 285

LIST OF ABBREVIATIONS

3D	Three Dimensional
ASME	American Society of Mechanical Engineers
ATT	Analogue Transform Technique
C	C-Programming Language
CNC	Computer Numerical Controlled
CWT	Continuous Wavelet Transform
DC	Direct Current
DOF	Degree of Freedom
DSM	Data Stored Memory
DSP	Digital Signal Processor
DWT	Discrete Wavelet Transform
FDM	Finite Difference Method
FEM	Finite Element Method
FFT	Fast Fourier Transform
FRF	Frequency Response Function
GUI	Graphical User Interface
IFFT	Inverse Fast Fourier Transform
ISO	International Organisation for Standardisation
JIS	Japanese International Standards
LTI	Linear Time Invariant
MATLAB	Matrix Laboratory
MCU	Motion Control Unit
MDOF	Multiple Degree of Freedom
MODELICA	Modelling Case Language
NC	Numerical Control
PDE	Partial Differential Equation
PI	Proportional Integral Controller
PID	Proportional Integral Differential Controller
PMSM	Permanent Magnet Synchronous Motor
PRBS	Pseudo-Random Binary Signal
PT_1	First-Order Lag Element
PT_2	Second-Order Lag Element
PWM	Pulse Width Modulation
RPM	Revolutions Per Minute
RTI	Real Time Interface
RTW	Real Time Workshop
SDOF	Single Degree of Freedom
SIMULINK	Simulation Linking Interface
SV	Space Vector
TFDTLM	Transient Frequency Domain Transmission Line Matrix
TLM	Transmission Line Matrix
WT	Wavelet Transform

NOMENCLATURE

a	Acceleration [m/s^2]
a	Dilatation or scale parameter (chapter 8)
a_a	Acceleration of the screw shaft point a [m/s^2]
a_m	Maximum possible acceleration [m/s^2]
a_{max}	Maximum acceleration [m/s$^{2]}$
b	Damping (Friction coefficient)
b	Translation parameter (chapter 8)
b_b	Bearing coefficient of friction [N-m-s/rad
b_{gw}	Guideway friction coefficient [N-s/m]
b_m	Coefficient of friction (motor bearings) [N-m-s/rad]
d	Diameter [m] (chapter 3)
d	Displacement [m]
d_a	Position of point a on the shaft towards the nut [mm]
d_{act}	actual position value (rotary or linear encoder) [mm]
d_b	Displacement of the end b of the bearing [m]
d_{bh}	Displacement of the end bh (bearing housing) [m]
d_d	Displacement of the end d of the nut [m]
d_e	Position error [mm]
den	Denominator filter coefficients vector
$difSec$	Position of the nut on the propagation list [sections]
$difSecA$	Position of the nut (axial propagation list) [sections]
d_l	Actual table position (from linear encoder) [mm]
d_{mb}	Relative displacement between bearing ends [m]
d_{min}	Minimum travel distance m]
d_n	Relative displacement between the nut ends [m]
d_{nom}	Reference position signal (before position filter)
d_p	pitch circle diameter of the bearing [mm]
d_{prof}	Axis-drive position profile
d_{ref}	Reference position (position demand) [mm]
d_{ss}	Screw shaft diameter [m]
d_{xref}	Reference position x-axis (position demand) [mm]
d_{yref}	Reference position y-axis (position demand) [mm]
e	Voltage [V]
e_β	β motor voltage [V]
e_α	α motor voltage [V]
$e_{\alpha ref}$	Reference voltage α-component [V]
$e_{\beta ref}$	Reference voltage β-component [V]
$e_{\alpha\beta ref}$	Vector of α-β voltages [V]
e_a, e_b, e_c	Motor line voltages [V]
e_{AN}, e_{BN}, e_{CN}	Motor phase voltages [V]
e_{Cn}	Voltage non-linear capacitor [V]
e_d	Direct motor voltage [V]
e_{dref}	Reference voltage d-component [V]
e_{Ln}	Voltage non-linear inductor [V]
e_m	Motor inertia effort
e_q	Quadrature motor voltage [V]
e_{qref}	Reference voltage q-component [V]
$f(t)$	Time domain signal
f_l	Bearing load coefficient
f_o	Bearing lubrication method
f_r	Feed rate [mm/min]
f_s	Frequency of the PWM signal [Hz]
g	Gravitational constant [m/s^2]
h	Number of sections
h_a	Number of sections of the axial model
h_t	Number of sections torsional model
i	Electric current [A]
i_β	β motor current [A]
i_α	Motor current α-component [A]
i_a, i_b, i_c	Motor phase currents [A]
i_d	Direct motor current [A]
i_{dact}	Actual current d-component [A]
i_{de}	Current error d-component [A]
i_{der}	Derivative component of i_{ref} [A]
i_{dref}	Reference current d-component [A]
i_{ht}	Holding current [A]
n_{la}	Numer of sections on the left (second zone axial model)
n_m	Numer of sections second zone (torsional model)
n_{ma}	Numer of sections second zone (axial model)
i_{int}	Integral component of i_{ref} [A]
$incr$	Encoder timer count or incremental count
i_p	Proportional component of i_{ref} [A]
i_q	Quadrature motor current [A]
i_{qact}	Actual current q-component [A]
i_{qe}	Current error q-component [A]
i_{qref}	Reference current q-component [A]
i_{ref}	Reference current (current demand) [A]
i_s	Stator current vector [A]
j	Jerk [m/s^3]
j_{max}	Maximum jerk [m/s^3]
k	Average slope of the hysteresis loop (chapter 1)
k	Mode number (chapter 8)
k	step number
k_{aff}	Acceleration feed forward gain [A-s^2/rad]
k_b	Ball screw force to torque constant
k_{ci}	Integral gain current controller [V/A-s]
k_{cp}	Proportional gain current controller [V/A]
k_{cs}	Torsional stiffness of the coupling [N-m/rad]
k_d	Velocity controller derivative gain [A-s^2/rad]
k_e	Electric constant of the motor [V-s/rad]
k_{eq}	Bearing mounting stiffness
k_{ff}	Feed forward gain
k_i	Velocity controller integral gain [A/rad]
k_n	Nut rigidity [N/m]
k_p	Velocity controller proportional gain [A-s/rad]
k_{rb}	Bearing stiffness
k_{rbh}	Bearing housing stiffness
k_{rn}	Resulting rigidity of the preloaded nut with mounting bracket [N/m]
k_T	Torque constant of the motor [N-m/A]
k_v	Gain of the position controller [m/min-mm]
k_{vff}	Feed forward velocity gain [rad/mm]
l	Length [m]
$lastSec$	Last nut position [sections]
$lastSecA$	Last nut position (axial model) [sections]
l_{axial}	Length of each section in the axial model [m]
l_d	Lead (pitch) of the ballscrew [m]
len	Number of filter coefficients
$len0$	Filter order
l_{end}	Length end porsion of the screw shaft [m]
l_f	Positions of the front bearing [m]
l_{front}	Length front porsion of the screw shaft [m]
$listF$	First zone propagation list (torsional model)
$listFa$	First zone propagation list (axial model)
$listM$	Second zone propagation list (torsional model)
$listMa$	Second zone propagation list (axial model)
l_n	Positions of the nut [m]
l_o	Absolute reference point for the nut movement [m]
l_r	Positions of the rear bearing [m]
l_s	Ball screw stroke length [m]
l_{ss}	Screw shaft length [m]
l_{tor}	Length of each section in the torsional model [m]
m	FFT number ofsamples (chapter 8)
m	Mass [kg]
m_{end}	Mass end porsion of the screw shaft [kg]
m_{fb}	Mass acting on the front bearing [kg]
m_{front}	Mass front porsion of the screw shaft [kg]
m_{le}	Mass of the linear encoder [kg]
m_{rb}	Mass acting on the rear bearing [kg]
n	Section where the nut is on $listM$
n_a	Axial sample times per torsional sample time
n_a	Section where the nut is on $listMaa$
n_f	Numer of sections first zone (torsional model)
n_{fa}	Numer of sections first zone (axial model)
n_l	Numer of sections on the left (second zone torsional model)
E^i	Incident pulse (stub)
E^i_{aff}	Incident pulse associated to Z_{aff}
E^i_c	Incident pulse associated to Z_c
E^i_{cs}	Incident pulse associated to Z_{cs}

nn	Reduction ratio of the ballscrew
n_n	Position of pulse B^i_{n+1} on *listM*
n_{na}	Position of pulse B^i_{na+1} on *listMa*
n_t	Number of axial sections per torsional section
num	Numerator fileter coefficients vector
p	Magnetic pole pairs
p_{abc}	Sector array
pA_i	Position of pulse A^i_i on a list
pA_{ia}	Position of pulse A^i_{ia} on a list (axial model)
pB_i	Position of pulse
pB_{ia}	Position of pulse B^i_{ia} on a list (axial model)
q^{-1}	Unit delay [s]
r_c	Circle radius [mm]
s	Laplace transform operator
t_a	Propagation time of axial waves [s]
t_c	Current control loop cycle time [s]
t_i	Time [s]
t_{pwm}	PWM slot duration [s]
t_s	Simulation time step
t_t	Propagation time for torsional waves [s]
t_v	Velocity control loop cycle time [s]
t_{xyz}	PWM possible duration times [s]
u	Propagation velocity [m/s]
u_a	Velocity of propagation of axial waves [m/s]
u_t	Velocity of propagation of torsional waves [m/s]
v	Velocity [m/s]
v_a	Velocity of the screw shaft point a [m/s]
v_{act}	Actual velocity [rad/s]
v_b	Displacement of the end b of the bearing [m/s]
v_{bh}	Displacement end bh of the bearing housing [m/s]
v_{bmb}	Relative velocity between bearing ends [m/s]
v_d	Velocity of the end d of the nut [m/s]
v_e	Velocity error [rad/s]
v_{fb}	Front bearing velocity [m/s]
v_{ff}	Velocity feed forward [rad/s]
v_{ha}	Rear bearing velocity [m/s]
v_l	Load velocity [m/s]
v_m	Maximum possible velocity [m/s]
v_n	Nut velocity (relative velocity between ends) [m/s]
v_{na}	Velocity of the nut contact point [m/s]
v_{ref}	Reference velocity value (velocity demand) [rad/s]
x	Distance [m]
x	x-coordinate [mm] (chapter 6)
x_0	Maximum displacement hysteresis loop (chapter 1)
y	y-coordinate [mm] (chapter 6)
z	Z- transform operator
zz_i	Filter delay output

A_i	First significant amplitude (chapter 1)
A^i, B^i	Incident pulse (link)
A_{i+r}	Amplitude after r cycles (chapter 1)
A^i_j, B^i_j	Incident voltages associated to Z_t
A^i_{ja}, B^i_{ja}	Incident voltages associated to Z_a
A^r, B^r	Reflected pulse (link)
A_{ss}	Screw shaft cross-sectional area [m^2]
B	Amplitude (chapter 8)
B_{CD}	Nut ball circle diameter [m]
C	Capacitance [Farad]
C_d	Capacitance per unit length [Farad/m]
C_{gw}	Guideway dynamic load rating [N]
C_n	Dynamic load rating of the nut [N]
D	Backlash [mm]
DP	Velocity limit for the zero slip force [m/s]
DV	Momentum limit for the zero velocity interval [N-s]
D_x	x-axis position [mm]
D_y	y-axis position [mm]
D_z	z-axis position [mm]
Q	Magnitude of FRF (chapter 1)
R	Resistance [ohm]
R_d	Resistance per unit length [ohms/m]
R_f	Shape factor (chapter 8)
R_{pwm}	PWM resolution
T	Torque [N-m]
T_a	Torque due to external load [N-m]
T_b	Bearing frictional torque [N-m]

E^i_d	Incident pulse associated to Z_L (d-component)
E^i_{de}	Incident pulse associated to Z_{ci} (d-component)
E^i_{dl}	Incident pulse for d_l calculation
E^i_{end}	Incident pulse associated with Z_{end}
E^i_{fbh}	Incident pulse associated with Z_{fbh}
E^i_i	Incident pulse associated to Z_i
E^i_l	Incident pulse associated to Z_l
E^i_m	Incident pulse associated to Z_m
E^i_{mfb}	Incident pulse associated with Z_{mfb}
E^i_{mrb}	Incident pulse associated to Z_{mrb}
E^i_{ns}	Incident pulse associated to Z_{ns}
E^i_q	Incident pulse associated to Z_L (q-component)
E^i_{qe}	Incident pulse associated to Z_{ci} (q-component)
E^i_{rb}	Incident pulse associated to Z_{rb}
E^i_{rbh}	Incident pulse associated to Z_{rbh}
E^i_{thm}	Incident pulse for θ_m calculation
E^i_{vff}	Incident pulse associated to Z_{vff}
E^r	Reflected pulse (stub)
E_{ss}	Screw shaft Young's modulus
E_x	Actual error movement of the x-axis [μm]
E_y	Actual error movement of the y-axis [μm]
E_z	Actual error movement of the z-axis [μm]
F	Force [N]
$F(\omega)$	Fourier transform of the signal $f(t)$
F_0	Friction load-component [N]
F_1	Friction velocity-component [N]
F_A	Axial load [N]
F_a	Axial load [N]
F_{ao}	Preloading force applied to the nut [N]
F_c	Cutting force [N]
F_d	Nut axial force [N]
F_E	Equivalent force acting [N]
F_f	Frictional force [N]
F_{gw}	Guideway frictional force [N]
F_{gw0}	Guideway frictional force under no-load [N]
FH	Magnitude of the static friction [N]
F_l	Force acting on the load [N]
F_{lat}	Lateral load [N]
F_{le}	Required force to move the linear encoder [N]
F_o	Coulomb friction [N]
F_p	Nut pre-loading force [N]
F_r	Radial load [N]
F_R	Resulting bearing load [N]
F_{rad}	Radial load [N]
F_{slip}	Slip force [N]
F_{stick}	Stick force [N]
G	Shear modulus
G_{ss}	Screw shaft shear modulus
I_o	Mass polar moment of inertia per unit length
J	Polar moment of inertia [Kg-m^2]
J_c	Mass moment of inertia of a rotor hub
J_{end}	Inertia acting on the rear bearing
J_m	Motor inertia [kg/m^2]
J_{re}	Rotor inertia of the rotary encoder
J_{ss}	Screw shaft polar moment of inertia
L	Inductance [Henry]
L_d	Inductance per unit length [Henry/m] (chapter 3)
L_d	Inductance phase d [Henry]
L_q	Inductance phase q [Henry]
M_{abc}	*abc* to α-β matrix conversion
M_{gw}	Guideway imposed load [kgf]
M_p	Peak value of response (chapter 1)
M_{xyz}	Matrix for the calculation of possible duration times
N	Encoder line count
N	Resolution parameter (chapter 8)
P	Momemtum [N-s]
PO	Percentage overshoot (chapter 1)
Q	Electric charge [Coulomb]
$\phi_y(z)$	z-axis rotation about y-axis [μm/mm]
$\phi_z(x)$	x-axis rotation about z-axis [μm/mm]
$\phi_z(y)$	y-axis rotation about z-axis [μm/mm]
$\phi_z(z)$	z-axis rotation about z-axis [μm/mm]
γ	Propagation function
φ	Phase [rad]
κ	Constant
λ	Induced flux [Webber]

T_c Input torque first section (torsional model) [N-m]
T_d Counter balance torque [N-m]
T_e Generated electromagnetic torque [N-m]
T_{f0} T_b velocity-dependent component [N-m]
T_{f1} T_b load-dependent component [N-m]
T_{fb} Front bearing frictional torque [N-m]
T_i Period [s]
T_{ld} Load torque [N-m]
T_m Motor load torque [N-m]
T_p Nut pre-loading torque [N-m]
t_p Position control loop cycle time [s]
T_{pwm} Period of the PWM signal [s]
T_r Nut reference torque [N-m]
T_{rb} Rear bearing frictional torque [N-m]
V_{DC} DC link voltage [V]
$W_\psi(a, b)$ Wavelet coefficients
X_b Radial load factor
X_{rad} Radial factor
Y_b Axial load factor
Y_d Conductance per unit length [1/ohms-m]
Y_{lat} Lateral factor
Z_a Characteristic impedance axial model
Z_{aff} Characteristic impedance associated to k_{aff}
Z_c Characteristic impedance associated to J_c
Z_{ci} Characteristic impedance associated to k_{ci}
Z_{cs} Characteristic impedance associated to k_{cs}
Z_d Characteristic impedance associated to k_d
Z_{end} Characteristic impedance associated with J_{end}
Z_{fbh} Characteristic impedance associated with k_{fbh}
Z_i Characteristic impedance associated to k_i
Z_L Characteristic impedance associated to L
Z_l Characteristic impedance associated to m
Z_m Characteristic impedance associated to J_m
Z_{mfb} Characteristic impedance associated with m_{fb}
Z_{mrb} Characteristic impedance associated to m_{rb}
Z_{ns} Characteristic impedance associated to k_n
Z_o Characteristic impedance
Z_{rb} Characteristic impedance associated to k_{rb}
Z_{rbh} Characteristic impedance associated to k_{rbh}
Z_t Characteristic impedance torsional model
Z_{vff} Characteristic impedance associated to k_{vff}

β_j ATT gain
β_{ss} Screw shaft lead angle [rad]
χ, δ, η Distributed parameter element constants
δ_f Stick force factor
ε Ball screw efficiency
ϕ Incremental position for a rotary encoder
ϕ_0 Encoder zero position
$\phi_x(x)$ x-axis rotation about x-axis [μm/mm]
$\phi_x(y)$ y-axis rotation about x-axis [μm/mm]
$\phi_x(z)$ z-axis rotation about x-axis [μm/mm]
$\phi_y(x)$ x-axis rotation about y-axis [μm/mm]
$\phi_y(y)$ y-axis rotation about y-axis [μm/mm]

λ Induced flux [Webber]
μ Friction coefficient [N-s/m]
μ_{re} Coefficient of friction of the rotary encoder bearings
θ Angular displacement [rad]
θ_a Angle at the contact point with the nut [rad]
θ_c Relative displacement of coupling ends [rad]
θ_e Electrical position [rad]
θ_l Coupling displacement at the screw shaft side [rad]
θ_m Actual [rad]
θ_m Mechanical position (from rotary encoder) [rad]
$\theta_{xy}(x, y)$ Squareness in the XY plane [μm/mm]
$\theta_{xz}(x, z)$ Squareness in the XZ plane [μm/mm]
$\theta_{yz}(y, z)$ Squareness in the YZ plane [μm/mm]
ρ Density [kg/m^3]
ρ_{ss} Screw shaft density [kg/m^{3}]
τ_c External torque acting on the shaft [N-m/m]
τ_j Propagation time [s]
υ operational viscosity of lubricant
ω_l Coupling velocity at the screw shaft side [rad/s]
ω_m Mechanical velocity (motor angular velocity) [rad/s]
ω_n Undamped natural frequency (chapter 8)
ω_0 Central wavelet frequency (chapter 8)
ω Angular frequency (chapter 8)
ω Angular velocity [rad]
ω_a Angular velocity of the screw shaft point a [rad/s]
ω_c Coupling angular velocity [rad/s]
ω_d Damped natural frequency (chapter 8)
ω_e Electrical velocity [rad/s]
ω_{fb} Front bearing angular velocity [rad/s]
ω_h Rear bearing angular velocity [rad/s]
ω_n Angular velocity of the nut contact point [rad/s]
ω_r Resonant frequency (chapter 1)
ξ_j Characteristic impedance
$\psi(t)$ Mother wavelet fucntion
$\psi_{a,b}(t)$ Son wavelets
$\psi(\omega)$ Fourier transformof the mother wavelet
ζ Damping factor (chapter 1)
X Impedance transfer function
X^I Admittance transfer function
Δt Transmission line propagation time [s]
Δx Transmission line length [m]
ΔU Area of displacement-force hysteresis loop (chapter 1)
ΔX Distance the x-axis is going to move [mm]
ΔY Distance the y-axis is going to move [mm]
$\Delta\omega$ Bandwidth (chapter 1)
Γ Reflection coefficient
$\Lambda_x(x)$ x-axis linear positioning error [μm]
$\Lambda_x(y)$ y-axis straightness in the x-axis direction [μm]
$\Lambda_x(z)$ z-axis straightness in the x-axis direction [μm]
$\Lambda_y(x)$ x-axis straightness in the y-axis direction [μm]
$\Lambda_y(y)$ y-axis linear positioning error [μm]
$\Lambda_y(z)$ z-axis straightness in the y-axis direction [μm]
$\Lambda_z(x)$ x-axis straightness in the z-axis direction [μm]
$\Lambda_z(y)$ y-axis straightness in the z-axis direction [μm]
$\Lambda_z(z)$ z-axis linear positioning error [μm]
Ψ Flux linkage [Webber]

1. INTRODUCTION

Nowadays the necessity of developing innovative and cost effective methods has become an imperative matter for many industries on the path to success in the global economy. This trend is forcing manufacturers to focus on higher levels of productivity and greater accuracy and reliability of products. Computer Numerical Controlled (CNC) machine tools are an integral part of the manufacturing process and the major contributors to workpiece accuracy. Consequently, an accuracy improvement of a machine is directly related to the quality of the parts produced. For that reason it is not surprising the attention that previous and present research efforts on various fields (system dynamics identification, control engineering, advanced motion controls, etc.) have dedicated to the area of machines for precision manufacturing.

A good understanding of the dynamic interaction of all machine components and their respective geometric and non-linear distortions is needed to improve the machine tool performance and motion control accuracy. This requires the development of detailed mathematical models of feed drives which must be optimally tuned to the measured static and dynamic behaviour of the machine tool. The information obtained from simulation results can be used to achieve a variety of benefits: to increase high-speed performance and robustness, reduce costs, improve design strategies, identify machine errors, early detection of wear, etc. Various types of models for feed drives (lumped parameter models, modular approach, hybrid models) have been developed by industrial and academic researchers, but the simulated responses did not reflect entirely the overall dynamic behaviour of the machine tools; generally only analogue drives were modelled and the stiffness calculations were made for only one position of the worktable. The model-system correspondence has to be improved in order to reflect the pointwise and the distributed features of the CNC machine tool feed drives.

Machine tools as non-linear hybrid systems can be characterised as time-dependent or transient systems and numerical methods such as Finite Element (FEM), Finite Difference (FDM), and Transmission Line Matrix (TLM) could be used to find a solution.

Generally, the dynamics involved in a CNC machine tool are governed by sets of *partial differential equations* (PDEs) where the independent variables are time and space coordinates. The problem is characterised in these circumstances as time-dependent and the *transmission line matrix method* can be applied. This technique introduced by Johns and Beurle [1] offered accurate and quick solutions for applications in various scientific fields: electromagnetics, wave propagation, hydraulics, acoustics, mechanics (not much emphasis on digital controllers), etc. The analogy between electrical circuits and physical systems allows the elements of physical systems to be represented by capacitors, inductors and resistances considering the wave propagation through a variety of mediums. Other important advantages of TLM method are:

- Discrete nature of the method is ideal for direct application via a digital computer algorithm – the models and algorithms condense and compute together all the variables without the need of mathematical operations like derivation and factorisation;
- Minimum requirement of data storage and the possibility of fast solutions by the reduction of initial errors;
- Relatively simple procedures - enabling both continuous and discrete models to be accommodated;
- High speed of processing – making this modelling technique very suitable for on-line condition-monitoring methods.
- Time-domain transient analysis is performed when broad band frequency responses are requested;
- Ability to handle complex structures with arbitrary geometries where no analytical solutions have been found yet.

The TLM method is included in the category of *unified methods* [2], which are based on dynamic analogies between equations of motion for systems of different disciplines: mechanical, electrical, fluid and thermal. The system's dynamic behaviour is governed by the energy exchange patterns between the system components. The series of system dynamic elements can be treated as a series of separated parameter elements. This formalised technique provides a basis for analysing the dynamic behaviour of each component and gives an intuitive interpretation of energy flow and storage into the system.

The notions of system state, energy and power (in the form of effort and flow variables) do not depend on the physical domain. They form the basis for defining a set of mathematical equations that govern any physical system behaviour [3]. Dynamics involved in components that are spatially distributed (shafts, beams, pipelines, etc.) are generally governed by sets of partial differential equations. The components that are concentrated and relatively pointwise (motors, couplings, valves, etc.) are appropriately modelled by algebraic or ordinary differential equations. The models are classified in three categories depending on the nature of components:

- Lumped parameter models – contain ordinary differential equations, where time is the only independent variable;
- Distributed parameter models - consist of partial differential equations, where the independent variables are time and the space co-ordinates;
- Hybrid models - include both types of differential equations.

A novel application of TLM method for modelling the dynamic behaviour of CNC machine tool feed drives for various running conditions is presented in this report. The feed drives are non-linear hybrid systems where a controller commands the movement of a worktable linked to a motor through a ball-screw. The interaction between the position loop controller, electrical drive and the worktable mechanism is described by differential equations and corresponding TLM models are derived. All digital feed drives components (starting from the set value generation in the motion controller to the positioning of the workpiece) are considered in the modelling process.

The non-linearities present in machine tools produce distortion of the feed drives response by introducing signal components at frequencies higher than the basic forcing frequency. A comprehensive analysis of machine tool non-linearities is essential for the development of effective TLM models. The dynamic TLM model of the ball-screw with moving nut also includes the distributed inertia of the screw, the effect of moving mass, the axial and torsional forces applied on the nut during its linear movement and the restraints applied by the bearings.

The development for the first time of a TLM model for a digital controller represents an important contribution to knowledge of modelled and simulated motion control systems for CNC machine tools.

The single axis simulation results for various stimuli conditions (step, and jerk-limited inputs) compare well against measurements for the same stimuli conditions on the machine. In this way, the single-axis model for CNC machine tool feed drive is validated on the basis of practical results.

Modern high-speed machining processes require higher machine accuracy at higher operating speeds. The machine tool producers can select any balance between speed and accuracy by taking into account that accuracy is an inverse factor of axes speed. This balance is used by advanced CNC machine tools to optimise the cutting path because the workpiece precision is mainly influenced by machine accuracy.

The machine tool accuracy is influenced by the errors due to geometric, load, thermal and dynamic effects. The methods for error measurement and correction are studied and a two-axis TLM model of a CNC machine tool is built to include geometric (rigid body) error components measured by laser interferometer. A circular interpolation algorithm is implemented in MATLAB and the simulated circular position error traces compare well with the machine error traces measured by a ball bar. The measurements are performed under the conditions established by ISO standards [4-6].

The TLM models containing the geometric and load errors reflect more accurately the dynamic behaviour of the real CNC machine tools. Therefore, a quick and easy characterisation of machine tool elements for a wide range of machine tool feed drives is enabled. However further research is considered necessary in order to reflect the complex interactions within the versatile hybrid multi-body systems which are CNC machine tool drive systems.

The two-axis model for CNC machine tool feed drives contributes to a full investigation into the dynamic state where valid structural resonances other than geometric errors (such as dynamic errors, load errors and thermally induced errors) should be introduced together with measured data. The comprehensive analysis of including the two-axis model of feed drives into the cutting process model ought to be performed if the structural dynamic effects are to be more deeply understood.

The development of a complete parametric model that integrates all the components of complex systems (like CNC machine tools) is a combination of theoretical analysis and experimental testing. This requires the identification of parameters using a range of techniques including experimental set-up to isolate individual parameters and the application of numerical techniques to analyse measured behaviour.

Modern time-frequency methods are intended to deal with a variety of non-stationary signals generated by diverse causes (vibration of rotating machines, transient behaviour, discontinuities, etc.).

Wavelets offer efficient and robust representation of such signals based on time-frequency localisation. At the basis of wavelet representation is the concept of approximating an arbitrary non-linear function in terms of dilates and translates of a single function (usually known as a mother wavelet function).

The wavelet basis functions have the special property of being *localised both in space and frequency*. The crucial problem is to select among many possible wavelet representations available, the most appropriate one to suit the identification of the studied non-linear systems, which are CNC machine tool feed, drives.

An attempt of using the *Continuous Wavelet Transform (CWT)* for the identification of resonant states and damping factors of machine elements is included. The technique showed to be effective for the identification of some resonant states but it could not achieve accurate results on the identification of damping factors.

The original contribution to knowledge consists of:

- The compilation of TLM modelling principles (derived in applications to the modelling of systems of different disciplines) and their extension to the development of mathematical models that can reflect the pointwise and the distributed features of CNC machine tool feed drives (including the moving nut effect). It represents the basis for the development of a universal mathematical model for modern CNC machine tools with digital drives.
- The development of a new TLM model for lumped dynamic behaviour denominated *the modified TLM stub*. This new model improves the convergence and computational processing speed of the original stub algorithm.

1.1 Aim and Objectives

The aim of this investigation is to develop TLM models for machine tool feed drives including for geometric, load and non-linear effects. The systems under investigation are:

- Bridgeport single-axis CNC machine tool;
- Cincinnati Arrow 500 CNC machine tool.

The following objectives were set in order to achieve this aim:

- To develop TLM models for Cartesian CNC machine tool feed drives including for geometric, load and non-linear effects;
- To identify the control loop coefficients and non-linear parameters of the TLM models;
- To implement the TLM models in the MATLAB environment and simulate the feed drive behaviour;
- To validate TLM models by comparing the simulated results with measured data when the same stimuli are applied;

1.2 Thesis Outline

The work presented in this thesis is structured into nine chapters as follows:

- *Chapter 2* presents a critical appraisal of literature regarding methods for the modelling and simulation of CNC machine tool feed drives, transmission line modelling techniques, and identification methods for modal parameters of CNC machine tools.

- *Chapter 3* depicts an overview of TLM techniques including the description of the various specific elements. Also the modelling of non-linear elements using TLM method is analysed. The chapter contains a comparison between TLM and the analogue transform technique, the development of the modified TLM Stub and a project plan is derived from the conclusions.
- *Chapter 4* describes the development of TLM models for the elements of CNC machine tool feed drives. A special emphasis is put on building an accurate TLM model for the open architecture controller comprised into the digitally controlled drive. Dynamic models for mechanical transmission components (bearings, guide ways, slides, ball-screw with moving nut and pre-load effects) are also created.
- *Chapter 5* illustrates how single-axis and two-axis TLM models are constructed from the models for various elements described in Chapter 4. The single-axis TLM model for the Bridgeport machine tool is used as the basis for the modelling approach. Then single- axis and two-axis TLM models (including the effect of geometric errors and moving mass) of the Cincinnati Arrow 500 vertical machining centre are created. The algorithms for linear and circular interpolation are included in the model for digital controller.
- *Chapter 6* describes the measurement techniques used for determining the geometric and load errors within CNC machine tool feed drives. In addition, the response of the closed-loop position control system to step and jerk-limited stimuli is measured.
- *Chapter 7* presents the TLM models implementation into MATLAB/SIMULINK. Simulation results for step and jerk-limited inputs (single axis models) and sine/cosine inputs (two-axis model) are compared with experimental ones.
- *Chapter 8* contains a review of methods for determining resonant frequencies and damping factors from data measured on machine tools. The emphasis is on wavelet transform techniques.
- *Chapter 9* summarises the results and conclusions, and recommends future work which should be carried out in order to amplify the benefits offered by TLM models of digital feed drives and wavelet techniques applied to modal parameter identification.

The next chapter contains a critical appraisal on scientific fields (TLM techniques, modelling, simulation and modal parameter identification of feed drives) relevant to the subject of this thesis, underlining strengths and weaknesses of previous work, the latest state-of-the-art and suggesting possible ways to progress.

2. LITERATURE REVIEW

This chapter presents an informed evaluation of publications relevant to the studied topics. The information is organised according to the research objectives presented in Chapter 1 underlining what is known, unbiased and valid and what remains to be explored in the future.

The main relevant topics are methods for modelling and simulation of CNC machine tool feed drives and transmission line modelling techniques. A summary of the essential theoretical frameworks and practical perspectives makes the link between published papers and this investigation.

2.1 Modelling and Simulation of Feed Drives from CNC Machine Tools

The *lumped-parameter models with load inertia reflected to the motor* [7-9] have been generally used as traditional methods for modelling and simulation of CNC machine tool feed drives. Ford [7] showed that a single-axis feed drive could be considered to be equivalent with a second order element and the resulting Bode diagrams did not contain any resonant frequencies that occurred in the machine response. The interaction and behaviour of individual elements could not be examined and the models had to be altered when any system component changed. Also the effect of load components on the system response was removed because of the "lumping" technique.

Pislaru et al [10] applied a *modular approach* to the modelling of CNC machine tool feed drives in order to overcome the above-mentioned shortcomings. The feed drive elements were defined as modules by using the approach suggested by Fu et al [11] when building a Newton-Euler model of a robot. The kinematic motion was transmitted forward through the model and resistive force flew back through the model. Based on this principle, the single axis feed drive model contained the reaction forces (due to friction and components inertia), which were applied as inputs to precedent modules. The analogue feed drive had a DC motor whose torque had to overcome the load element inertia, the friction forces within bearings, between worktable / saddle and guide ways and between nut and screw.

Pislaru also developed two-axis models [12, 13] and three-axis models [14] using the same modular approach. The machine geometric errors (measured by laser interferometer) were integrated into the two-axis model and a mathematical procedure to calculate the ball bar predicted values was established. The modular approach offered greater flexibility in model construction (various components could be included /removed without altering the whole system model) and the requirements for the control part (controller, pre-amplifier, power amplifier and motor) could be evaluated due to reaction force computation. The single axis simulation results for trapezoidal rate demand [15] compared well with the measured data. Simulated Bode diagrams were produced using Linear Time Invariant (LTI) viewer from MATLAB and models for timing belt and ball-screw considering non-linear behaviour [16] were built. The authors supposed that it was possible to simulate the effect of resonant states of feed drive components without including the measured values of damping factors into the models. The simulation results were similar to measured Bode diagrams and ball bar plots, but the simulated dynamic performance had to be improved because the effect of system resonant states was not present in the simulation results.

Holroyd et al [17] investigated the dynamic characteristics of a CNC machine tool feed drive and modelled its elements as point inertias connected by springs and dampers. An eigenvalue approach was used for determining undamped natural frequencies of the drive. It was evident that more research should be performed regarding modelling non-linearities such as belt tension, friction between belt and pulley, etc.

Pislaru [18] performed a comparison between lumped parameter models and modular approach and developed a *hybrid model* of CNC machine tool feed drive with distributed load, explicit damping coefficients, backlash and friction. The model was a combination of distributed and lumped elements described by partial differential equations and ordinary differential equations as suggested by Bartlett and Whalley [19]. The ball-screw was modelled with distributed parameters (seven SIMULINK modules were produced), while other components (bearings, belt and pulleys, motor, etc.) had lumped parameter models.

The non-linearities and modal parameters (resonant frequencies, damping factors) were measured by specialised equipment (laser interferometer, accelerometers, signal analyser). Novel measurement practices for decoding signals generated by encoders (rotary encoders situated on DC motor, ball-screw end and linear encoders) were defined. The influence of time constants and gains of closed loops for velocity and position control was considered. Also a novel application of continuous wavelet transform for modal parameter identification of machine tool feed drives was elaborated.

The non-linearities included into the hybrid model described in [20] were defined by ordinary differential equation (friction) and partial differential equation (backlash). Although the hybrid model had several disadvantages (worktable positioned at half of travel length and swept sine / random white noise was applied to the pre-amplifier), the simulation results when the nut oscillated at the middle of the screw shaft were similar to the machine responses. The values of simulated resonant states however were still slightly different than the experimental ones therefore more research had to be done in order to optimise the hybrid model.

Holroyd et al [21] developed a generalised eigenvalue method to estimate the undamped and viscous damped natural frequencies, damping coefficients and mode shapes of an analogue feed drive. A study of the influence of stiffness and damping coefficients within the hybrid model on the resonant states was performed. The results could be useful in optimising the hybrid model parameters so the simulation results are in accordance with the real data.

The dynamic behaviour of a ball-screw with moving nut was modelled by Holroyd et al [22] in C language using a finite element approach. The ball-screw was divided into a large number of elements and contact and boundary conditions for each element and adjacent ones were studied. An important conclusion was that the natural frequencies of the ball-screw system vary in time due to two causes: The lateral restraint produced by the nut when the screw transversally vibrated and the relation between screw torsional and axial motion and worktable/saddle tilting.

The models previously developed were implemented in SIMULINK (hybrid models) and C language (dynamic model of a ball-screw). Simulation times were of the order of hours due to the great number of model elements, therefore further research should be performed in order to reduce the simulation times and to improve the accuracy of simulated results.

2.2 Transmission Line Modelling Techniques

Transmission line modelling techniques are based on the extension of the modelling theory of two-wire transmission lines to the modelling of dynamic systems. Sadiku and Agba [23] used the systems perspective (considered as series of components interconnected for energy transfer) in modelling processes. Then the mathematical equivalence between component equations and the equations containing voltages and currents for a transmission line was made. Applications of this concept go back to Auslander [24] who presented the bilateral delay principle for fluid systems modelling.

Two different techniques developed mathematical models describing the dynamics of system components: *Transmission Line Matrix Method* and *Analogue Transform Technique.*

Johns and Beurle [1] presented the transmission line matrix method as a numerical method for solving efficiently lumped-parameter networks and field problems. The technique's flexibility for modelling two and three-dimensional field problems was also addressed. Numerous improvements and developments of this method have been reported for applications to the modelling and simulation of electromagnetic propagation and electromagnetic compatibility [25] and other subjects.

Boucher and Kitsios [26] applied TLM principles to fluid systems modelling demonstrating the feasibility of representing all dynamic elements in a fluid circuit (excluding resistance) as distributed pure time delay elements. Fluid transmission lines were divided into a number of identical time delay lengths and equivalent open-end and closed-end TLM stubs modelled inertance and capacitance. Resistance was lumped at the junctions where all wave transformation by scattering or attenuation was concentrated. Computations on simple circuits showed good agreement with lumped parameter modelling.

The same authors applied the TLM method to the modelling of a hydraulic position control system comprising a hydraulic motor driving a flywheel attached to the motor shaft and coupled to a lead screw mechanism [27]. The motor shaft and screw shaft were treated as distributed (transmission line) elements, conveying torsional stress waves. The hydraulic motor, pump and flywheel were considered to be lumped elements so TLM stub could be used to model them. Comparison between theoretical and experimental results showed good agreement, although friction in the moving parts and inertia of the feedback component were neglected. Comparisons between the TLM model and a traditional lumped one showed the superiority of the distributed approach for the prediction of the transient oscillation frequencies.

Beck et al [28] employed the TLM method for drill strings modelling. The study targeted an arbitrary fluid network including pipes with different lengths and acoustic delay times. The events synchronisation was achieved by setting a common length (delay time) for pipe segmentation. This common length was made small enough such that the shortest pipe could be assumed to become an integer multiple of the segment length. The segment length was chosen in order to produce acceptable resolution according to the maximum frequency of simulation. Simulation analysis and test results validated the computational efficiency and the TLM ability for modelling a wide range of topologies.

Partidge et al [29] treated the shaft torsion effect as a direct analogy to electrical networks. A shaft and turntable with linear and non-linear friction was used as an example. TLM stubs represented lumped elements (turntable inertia), and distributed elements (shaft) were modelled by TLM links. In contrast with Boucher et al [26], the shaft was not divided into equal lengths because of the example simplicity. The study proved the TLM flexibility or usefulness for the modelling of mechanical problems with non-linear friction dynamics.

A series of articles by Hui and Christopoulos [30, 31, 35-36] present the TLM application to the numerical simulation of electronic power circuits and linear and non-linear circuits. Diverse characteristics and developments of TLM are included only for lumped parameter elements.

A new TLM method to model mutual inductance was developed in [30]. This is an important added feature providing a more realistic approach to electrical element modelling. The TLM method showed an efficient treatment of non-linearities (such as switching elements) eliminating the need for time-consuming inversion of system matrices.

The use of a *TLM-based discrete transform* as a solution for electrical networks and general systems of integral-differential equations was discussed in [31]. The implementation procedures were described and a discrete conversion table was constructed. The proposed method was used to simulate an electrical circuit and simulation results were compared using Runge-Kutta fourth order and Gear third order numerical methods. The simulated results generated by the TLM method were close to those using the Gear third-order method. However the TLM method exhibited some advantages: minimum requirement of data storage, the possibility of a fast solution by the reduction of initial errors, the interpretation of calculus equations as an electrical circuit and the handling of both integration and differentiation in the same equation. Further extensions of the discrete transform include the TLM model derived by Stubbs et al [32] including voltage dependent sources and the TLM models built by Murtonen and Lowery [33] for multi-port devices, such as transistors.

The TLM-based discrete transform was applied by Hui and Christopoulos [34] to the modelling of an industrial inverter driving a 4 kW DC motor. Models for the DC motor and the three-phase thyristor converter (inverter) were developed. The inductance of the motor was modelled by a short-circuit transmission line (TLM stub), and each of the inverter's switching devices was represented by a small capacitive TLM stub with a switch at one end to control the pulse direction. The simulated results accurately predicted the system behaviour because they were validated by comparison with the measured data.

The same authors [35] used TLM-based discrete transform for systems with varying coefficients. Each non-linear differential or integral component was represented by a transmission line segment (unity value component). Non-linearities were treated as parts of a forcing function affecting the unity value component, thus avoiding complications with energy conservation following changes of component value. There was no restriction on the nature of the non-linearities as long as the non-linear functions were known. Models of non-linear inductance with and without hysteresis were presented. The proposed transform was tested for numerical and practical problems including non-linearities of real systems.

Hui and Christopoulos [36] applied the discrete transform to develop a TLM model for a high frequency switch mode power supply circuit. The circuit consisted of two stages with widely separated frequencies (a 50Hz rectifier and a 25kHz converter) so a constant time step of 0.01 μs was used in order to include the dynamics of the high frequency stage.

Comparison between simulation results and experimental ones reported by Davis and Ray [37] showed the TLM method reliability and potential for the simulation of various power electronic circuits containing non-linearities. Hui & Zhu [38] applied the non-linear discrete transform to model and simulate the hysteresis effects of ferro and ferrite magnetic materials.

The transmission line equations introduce physically motivated time delays between components due to the wave propagation speed in a transmission line as shown by Krus [39]. The component models could be simulated independent of each other offering the following advantages:

- <u>Implement the system model for simulation using parallel processors</u>- Fung et al [40] decoupled a multistage electronic power circuit into various sub circuits. Each sub circuit was modelled by a small system matrix and simulated in one program module. The TLM link algorithm connected all program modules together ensuring that the parallel simulation was possible. The same authors [41] showed a 70 % reduction of computing time in comparison with the conventional non-decoupled sequential approach. Issues associated with parallel processing such as granularity, synchronisation and load balancing were also discussed.
- <u>Modelling systems with widely spread time constants using a variable time step</u> - Hui et al [42] confirmed that transient effects could be modelled with small time steps and steady-state effects with large time steps. The applicability of the method was verified by comparing simulation results against data from known analytical solutions of coupled electrical circuits. The overall simulation time was substantially reduced while the transient and steady states could be simultaneously observed. Tenorio de Carvalho et al [43] extended this method to perform bi-dimensional electromagnetic analysis of microelectronic circuits. Advantages of this approach over conventional TLM models were confirmed.
- <u>Modelling systems comprising components with different operational frequencies</u> - The traditional time domain simulation approach usually modelled an entire system as a single network and sequentially executed the model algorithms. The smallest time constant and/or the highest switching frequency component limited the time step used in simulation. This restriction was relaxed by Fung and Hui [44] who developed a conversion technique. The system was divided into subsystems that could use simultaneously different time steps suitable for their operational frequencies. The subsystems were linked at regular intervals for energy transfer by an improved TLM link

algorithm [45] and a derived stub/link TLM conversion algorithm. The combination of the two new techniques reduced two thirds of the computing time in a simulation of a three-stage switched-mode power supply system.

Deml and Turkes [46] combined the improved TLM approach with the advantages of two previous methods (the state space averaging method [47] and the envelope following method [48]) obtaining a new link model for fast simulation of transients in power electronic circuits. A circuit was partitioned in sub-circuits with typical periods. Then every sub-circuit was simulated separately (like the improved TLM link algorithm). Finally, the processes were connected by the new link model. Analysis of simulation results showed significant simulation speed-up with simulation errors below 4%.

Johansson et al [49] proved the numeric robustness of TLM method by building a distributed simulation environment in MODELICA (objected-oriented modelling language). Large and complex multi-domain models could be developed in this way.

The TLM principles have been also extended to systems modelling in the *frequency domain*. Jin and Vahldieck [50] combined the flexibility of the conventional TLM method with the computational efficiency of frequency-domain methods. A succession of impulses with sinusoidal envelope excited a TLM mesh so the magnitude of the output waveform (envelope) contained the transfer characteristic of the simulated system at the excitation frequency. A steady-state analysis in the time domain was performed in this way.

Johns and Christopoulos [51] formulated a set of complex frequency dependent simultaneous equations involving the incident voltage at each node and the source of excitation nodes. The set of equations were solved at each frequency for the incident voltages using the Jacobi method or the conjugate gradient method.

The two approaches have different criteria to satisfy but both methods repeat the simulation at every frequency point to compute the response over a frequency band of interest. Salama and Riad [52] presented an approach that combined the features of TLM methods in the time domain and the frequency domain. It was based on a steady-state analysis in the frequency domain using transient analysis techniques and it was referred to as the *Transient Frequency Domain Transmission Line Matrix (TFDTLM)*. The method was able to extract the frequency domain information from only one simulation. The main conclusions of this study showed that:

- A first-order approximation filter can perfectly model lossless inhomogeneous media.
- A second-order approximation filter can provide acceptable order accuracy in the case of a

lossy inhomogeneous medium.

- The TFDTLM can easily be interfaced with any time domain TLM method.

Whalley and Bartlett [53] derived the *analogue transform technique* which is an analytical method based on a distributed-lumped (hybrid) representation of a system. The method was centred on partial differential equation representations for spatially dispersed components (e.g. pipelines, beams, shaft drives) and ordinary differential or algebraic equations for concentrated and relatively point wise components. The Laplace transform converted the differential equations from the time domain to the s-domain. These s-domain equations were represented in state-space form and then converted into a discrete model via the z- and w-transformation process.

Bartlett and Whalley [19] presented the method in 1988 as a "natural" procedure that exposes the correspondence between theoretical assumptions used in the modelling exercise and the physical composition of a system. The modelling of the gas flow through two long pipelines connected by valves and reaction chambers was presented. Distributed parameter elements were modelled using the solution of the equation for a segment of a lossless transmission line and lumped parameter components were represented by their transfer function. Distributed and lumped impedance matrices were parts of a distributed-lumped system matrix in impedance form. However, the inversion of distributed-lumped matrix was necessary to complete the process.

Whalley et al [54] showed that the method generated expressions with multiple combinations of irrational functions and matrix descriptions with an order greater than three. The Smith normal form of the distributed-lumped system impedance matrix was employed in order to speed up the generation of the matrix inverse. The resultant admittance matrix still contained irrational functions for complex systems.

Bartlett and Whalley [55] improved the method by relaxing its restrictions and representing each component by correspondent impedance/admittance modules that can be simulated independently. The combinations of lumped and distributed components were analysed and simulated for two examples: the ventilation of long tunnels and the torsional oscillations of a rotor shell used in many industrial applications. Results demonstrated the effect of distributed mass/inertia and stiffness of the system response that a totally lumped, pointwise model representation could not reproduce.

The technique was also applied for modelling long drive shaft arrangements [56] and marine propulsion systems [57]. The feasibility of the method for investigating distributed-lumped configurations with varying geometry was also studied by Bartlett and Whalley [58] (modelling of long shaft rotors comprising three different cross-sectional areas), and Farshidianfar [59] (modelling of shafts in automotive driveline systems).

Abdul-Ameer [60] extended the method to include additional terms enabling the analysis of more complex hybrid systems such as vehicle dynamometer (comprising an armature controlled DC motor, a roll/drive shaft/roll arrangement, bearings) and a hydraulic pipeline arrangement under unsteady laminar flow conditions. Results obtained from simulations illustrated the method capability and viability for dynamic behaviour analysis of real systems.

2.3 Summary

Although the development of models of feed drives have made an important contribution in the area of machines for precision manufacturing, the majority of the literature refers to lumped parameter models. This is because this type of model is simple to construct and analyse. Lumped parameter models have a good performance in representing the dynamics of interest for design purposes, and simulations do not require a great amount of computational resources. A major draw back of this type of model comes from the fact that stiffness calculations are made for only one position of the worktable, and the effect of load components on the system response is removed during the lumping process. As a result, the model cannot reflect some resonant frequencies contained in the machine response.

Pislaru [10] showed that feed drive components could be defined as modules where kinematical motion is transmitted forward and resistive forces flow back through the model. This *modular approach* offers flexibility in model construction (various components can be included/removed without altering the whole system model) and gives the possibility to evaluate the controller requirements due to reaction force computation. However, the modules are still a lumped representation of the feed drive components and the effect of system resonant states is not present in the simulation results.

Generally, the accuracy with which a model resembles a real system depends on the complexity of the chosen mathematical model. As suggested by various authors ([19], [27]-[29], [49]), a more detail model of the system can be obtained when lumped parameter modules represent components with localised dynamic effects (e.g. bearings, couplings and motors) and distributed parameter modules represent components distributed on space (screw shafts). This principle was used by Pislaru [18] to develop a hybrid model of a CNC machine tool feed drive reporting that the simulation results were similar to the machine responses. A disadvantage of this model is that the dynamic effect of the moving nut is approximated by consideration of the dynamic response only for the screw middle travel position.

A recent study reported by Holroyd et al [22] presented a model of a ballscrew with moving nut using a finite element approach. Results from this study showed an improvement of the accuracy of simulated results; however simulation times were of the order of hours due to the complexity of the model. The need for further research in order to reduce simulation times and to improve the accuracy of simulation results is thus envisaged.

In electromagnetics, the transmission line matrix method is considered a general scheme to solve transient problems. The advantage of the technique is not just because it allows time-domain transient analysis (where broad band frequency responses can be obtained) but also because it has the ability to handle complex structures with arbitrary geometries where no analytical solutions have been found yet. Another advantage of TLM is that it provides a conceptual model that can be simulated exactly on a digital computer and that it can lead to models and algorithms, which condense and compute together all the variables without the need of mathematical operations like differencing and factorisation.

The application of TLM to the modelling of fluid and mechanical systems ([26]-[29], [49]) implies the same representation used by the hybrid approach: lumped parameter modules represent components with localised dynamic effects and distributed parameter modules represent components distributed on space. The difference resides on the fact that transmission line equations introduce natural time delays between components due to the wave propagation speed in a transmission line. Therefore, distributed components must be divided into a number of identical elements in order to: achieve the synchronisation of events in the simulation, and to produce acceptable resolution according to the maximum frequency of the simulation. This characteristic gives the possibility to include the effect of the movement nut like in Holroyd's et al [22] approach.

3. TRANSMISSION LINE MODELLING TECHNIQUES

In the use of transmission line modelling techniques, a series of elements are interconnected to simulate energy transfer throughout a system. The system is represented as a mesh of transmission lines providing a mathematical equivalence between the system equations and the equations for voltages and currents in the ordered mesh.

The modelling principle describes the laws and relations of elements by mathematical models in the form of sets of differential equations. The analytical or numeric solutions can be obtained according to the selected approach. Two model approaches may be identified: the Analogue Transform Technique (ATT) and the Transmission Line Matrix Method.

In the analogue transform technique - the Laplace transform is used to convert the differential equations from the time domain to the s-domain. These s-domain equations are then represented in a state-space form that is converted into a discrete model before determining the solution.

In the transmission line matrix method - a discrete model is provided by a time stepping technique in the discrete time domain. Consequently, the method is ideal for direct implementation via a digital computer algorithm. The discrete model is derived directly, without the intermediate step of the Laplace and Z-domain transformations, in contrast to the state-space modelling approach.

3.1 The Analogue Transform Technique

The analogue transform technique is a modelling technique based on a general matrix description for systems comprising a series of distributed-lumped elements. The realisation arises from consecutively connected distributed parameter elements followed by lumped parameter elements in series ending with a final lumped, termination, element as shown in Figure 3.1.

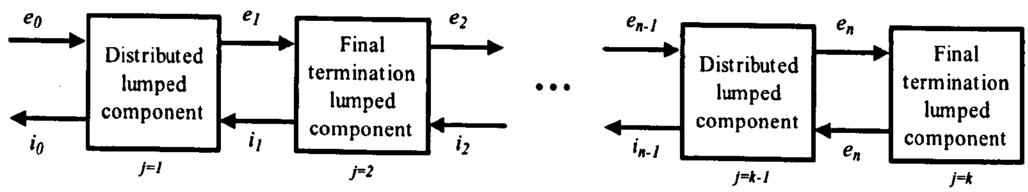


Figure 3.1 General series representation of a distributed-lumped parameter system [19]

For a given j^{th} element: e_{j-1} and i_j represent the effort (voltage) and flux (current) inputs; and e_j and i_{j-1} represent the effort and flux outputs of the element.

Lumped parameter elements are represented by impedance/admittance representations of its transfer function in the z-domain. The Laplace transform is used to convert the differential equations that represent the element from the time domain to the *s*-domain in order to obtain the transfer function. The transfer function is then converted to the z-domain via the z transform. The resultant transfer function in the z-domain is regarded as the admittance representation of the element. The impedance representation ($X(z_{j-1})$) is the inverse of the admittance representation ($X^{-1}(z_{j-1})$). For example, admittance and impedance modules for the final termination element (the j element) described by equation (3.1) are shown in Figure 3.2.

$$e_j(z_{j-1}) = X(z_{j-1})i_j(z_{j-1}) \tag{3.1}$$

Where,
$$z_{j-1} = \exp^{\tau_{j-1}s} \tag{3.2}$$

τ_{j-1} in equation (3.2) represents the propagation time calculated for the j-1 element in the system.

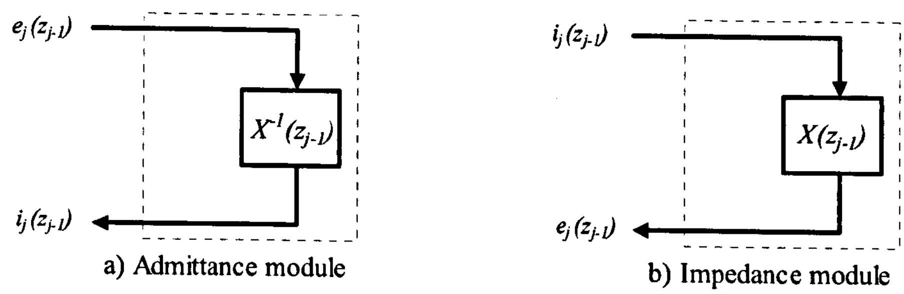


Figure 3.2 General modules for the termination element [55]

Distributed parameter elements are modelled by general impedance/admittance modules derived from (the representation in the z-domain) of two particular cases of the analytical solution of the Telegrapher's Equation, as presented in Appendix A. Figure 3.3 and Figure 3.4 illustrate the corresponding impedance and admittance models.

The characteristic impedance ξ_j and the parameter β_j are defined according to the equivalence between the equations of the element to be simulated (j element) and the differential equation that describes a segment of transmission line – with length l_j, resistance per unit length R_d, inductance per unit length L_d, conductance per unit length Y_d, and capacitance per unit length C_d (for details see Appendix A). Then,

$$\xi_d = \sqrt{L_d / C_d} \tag{3.3}$$

$$\beta_j = \exp^{\tau_j \delta} \tag{3.4}$$

Where,
$$\tau_j = 2l_j\sqrt{L_d C_d} \tag{3.5}$$

$$\delta = R_d / L_d \tag{3.6}$$

It must be noted that for a loss-less line $R_d = 0$, therefore $\beta_j = 1$.

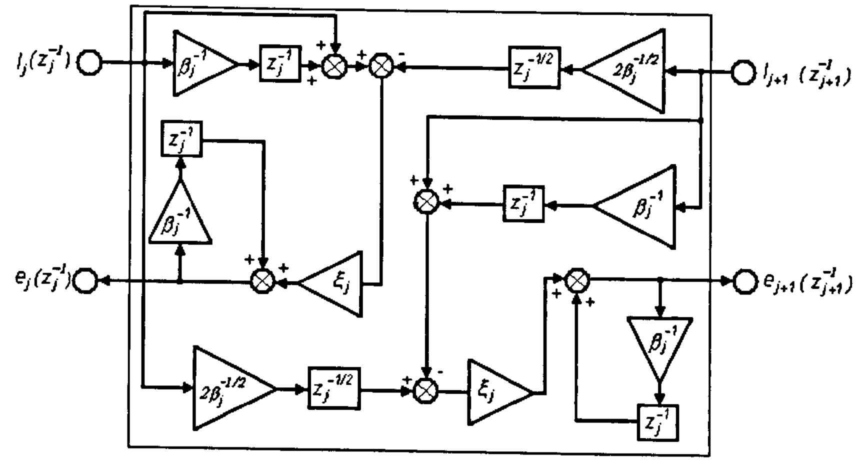


Figure 3.3 Impedance module for a distributed parameter element [60]

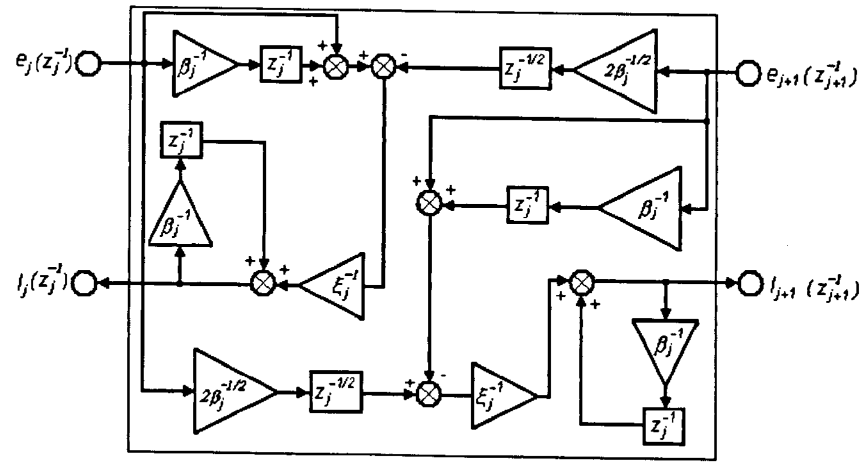


Figure 3.4 Admittance module for a distributed parameter element [60]

3.2 The Transmission Line Matrix Method

The transmission line matrix method belongs to the general class of differential time-domain numerical modelling methods. It is used to solve time-dependent or transient problems, thus involving ordinary and partial differential equations. The method approximates to continuous space representing a system as a mesh of transmission lines. They represent the mathematical equivalence between the system equations and the equations for voltages and currents on the transmission line mesh. Depending on the process being modelled, this can be in one, two or three dimensions.

Two equivalent circuits are used in the TLM technique: **the stub** and **the link** circuits. Those circuits are called the basic TLM units as described by Johns & Beurle [1]. TLM links are two-port one-dimensional building blocks that can be used for one-, two- or three-dimensional modelling. On the other hand, TLM stubs are one-port units, which can be used for solving circuits and equations, and are used in multi-dimensional modelling to complement TLM links. Generally, TLM links are used for modelling distributed parameter elements and TLM stubs can be used to represent lumped parameter elements.

A transmission line segment representing a unity value element is used to model a non-linear element when dealing with problems including non-linearities. Thus, a non-linear variation in the element value is treated as part of the forcing function. This procedure makes the TLM technique a very useful tool for the modelling and simulation of linear and non-linear systems.

3.2.1 The TLM Stub

Christopoulus [25] stated that any electrical circuit could be represented as a network of transmission line sections by simply replacing the reactive elements with corresponding stubs. Variables such as voltage and current are regarded as discrete pulses bouncing at a velocity 'u' to and from the nodes of these stubs at each time step. The voltage and current in each element is determined from the incident and reflected pulses in each stub (Figure 3.5).

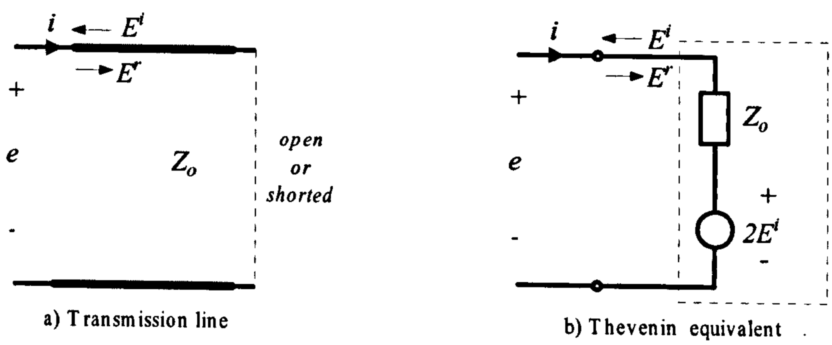


Figure 3.5 TLM stub [25]

The operation begins with an incident pulse (E^i) representing the initial conditions being injected into a stub with characteristic impedance Z_0. The incident pulse in the stub (transmission line of length Δx) takes a time step (Δt) to travel a round-trip ($2\Delta x$) to the other end and back. If the far end of the stub is short-circuited (i.e. inductive), the pulse will be reflected and inverted. If it is open-circuited (i.e. capacitive), the pulse will be reflected without inversion. The reflected pulse (E^r) thus becomes the incident pulse in the next time step. The pulse will interact with other parts of the circuit on incidence to the node. The pulses velocity of propagation (u) is calculated as:

$$u = \frac{\Delta x}{\Delta t / 2} \tag{3.7}$$

This discrete process is governed by the denominated scattering algorithm, which is illustrated in Figure 3.6.

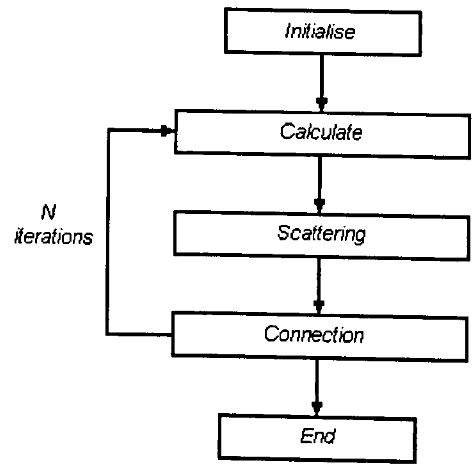


Figure 3.6 The TLM scattering algorithm [30]

Initialise - Initialise the problem space and apply boundary conditions.

Calculate - Calculate the problem elements at every node, output if required: If $E^i(k)$ is known at step k, the voltages and current in Figure 3.5 may be calculated. Taken $e(k)$ as the discrete stimulus applied to the transmission line, from the Thevenin equivalent branch (Figure 3.5.b) formula gives:

$$i(k) = \left(e(k) - 2E^i(k)\right) / Z_0 \tag{3.8}$$

Scattering - Scatter each incident voltage pulse off each node to generate reflected pulses according to the value of the reflection coefficient Γ:

$$E^r(k) = \Gamma(e(k) - E^i(k)) \tag{3.9}$$

Connection - Connect each reflected voltage pulse from each arm of each TLM node to its adjacent neighbour: The reflected pulse becomes the next incident pulse, hence

$$E^i(k+1) = E^r(k) \tag{3.10}$$

N Iterations - repeat until problem has been simulated: With $E^i(k+1)$ obtained from equation (3.9), $i(k+1)$ may be obtained from equation (3.7). Then the process is repeated for as long as desired.

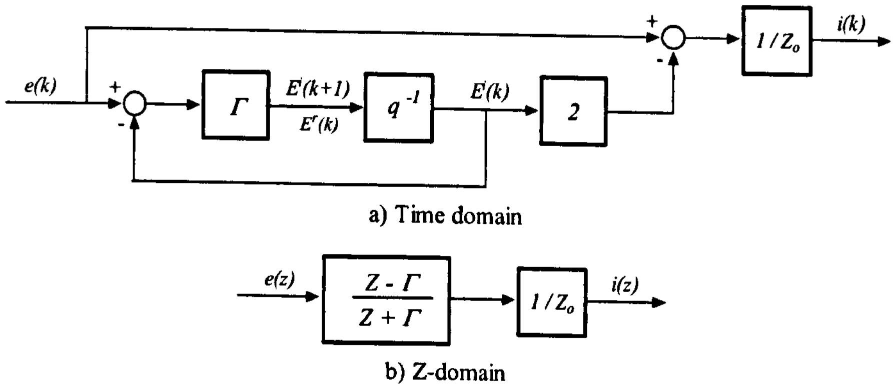


Figure 3.7 TLM Stub block diagram

Assuming q^{-1} the delay (Δt) for a pulse, the equations stated in the TLM algorithm for the stub can be represented as illustrated in Figure 3.7(a). Figure 3.7(b) shows the equivalent block diagram in z-domain ($z^{-1} = q^{-1}$).

Hui & Christopoulos [30] underlined that the weighting of the characteristic elements in the stub can be chosen accordingly to the nature of the represented element. Then, the characteristic impedance of an inductive stub is $Z_o=L/(\Delta t/2)$ and the reflection coefficient is $\Gamma= -1$, where L is the inductance. Similarly, the characteristic impedance of a capacitive stub is $Z_o=(\Delta t/2)/C$ and the reflection coefficient is $\Gamma= 1$, where C is the capacitance.

Hui & Christopoulos [31] used this property to extend the application of TLM stubs to the solution of integral-differential equations. The coefficient of a differential term can be represented by an inductance; it can then be modelled as a short-circuited transmission line. Likewise, the voltage on a capacitor (which in turn can be modelled as an open circuited transmission line) can represent an integral term. Thus, the propagation time Δt is equivalent to the time step used in numerical integration methods. Proportional terms are simply modelled by a resistance. For comparison purposes, Table 3.1 describes the TLM and the equivalent Z transform of the integral, differential and proportional terms. Note that the TLM transform for an integral term is equivalent to the trapezoidal integration method when $\kappa= 1$.

Continuous model	TLM transform (Discrete model)	Equivalent Z-domain transform
$e(t) = \kappa i(t)$	$e(k) = \kappa i(k)$	$e(z) = \kappa i(z)$
$e(t) = \kappa \frac{d}{dt} i(t)$	$Z_0 = \kappa /(\Delta t/2)$ $e(k) = Z_0 i(k) + 2E^i(k)$ $E^i(k+1) = E^i(k) - e(k)$	$e(z) = Z_o\left(\frac{Z-1}{Z+1}\right)i(z)$
$e(t) = \frac{1}{\kappa}\int i(t)dt$	$Z_0 = (\Delta t/2)/\kappa$ $e(k) = Z_0 i(k) + 2E^i(k)$ $E^i(k+1) = e(k) - E^i(k)$	$e(z) = Z_o\left(\frac{Z+1}{Z-1}\right)i(z)$

Table 3.1 TLM and Z transforms of integral, differential and proportional terms

It can be seen that the TLM transform can be used like the Z transform to find a solution. This is implemented substituting a calculus model for a respective TLM model. Then, a discrete model in the discrete time domain is obtained to achieve a solution in a stepping routine.

3.2.2 TLM Link

A TLM link is the discrete representation of a loss-less transmission line as shown in Figure 3.8. Pulses incident at each port are reflected and propagated to the other port where they become the incident pulses in the next iteration.

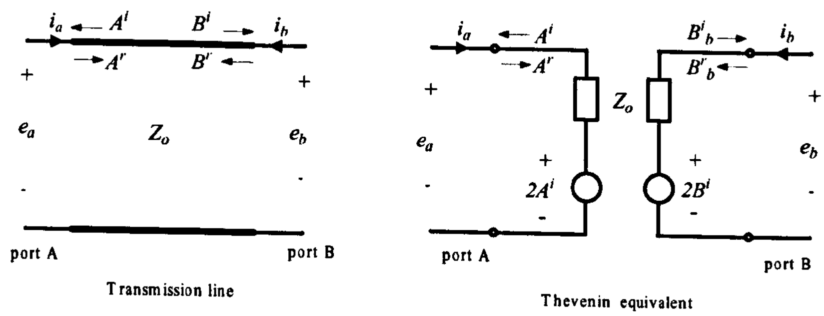


Figure 3.8 TLM link [25]

According to the Thevenin equivalent branch at each port in Figure 3.8, the scattering algorithm is represented by the following equations:

Calculate - If $E^i_a(k)$ and $E^i_b(k)$ are known at time step k, $i_a(k)$ and $i_b(k)$ can be calculated from the boundary conditions, then:

$$i_a(k) = (e_a(k) - 2A^i(k)) / Z_0 \tag{3.11}$$

$$i_b(k) = (e_b(k) - 2B^i(k)) / Z_0 \tag{3.12}$$

Scattering - Scatter each incident voltage pulse off each node to generate reflected pulses:

$$A^r(k) = e_a(k) - A^i(k) \tag{3.13}$$

$$B^r(k) = e_b(k) - B^i(k) \tag{3.14}$$

Substituting equations (3.11) and (3.12) into equation (3.13) and (3.14) gives:

$$A^r(k) = A^i(k) + i_a(k)Z_0 \tag{3.15}$$

$$B^r(k) = B^i(k) - i_b(k)Z_0 \tag{3.16}$$

Connection - The reflected pulse becomes the next incident pulse, hence

$$B^i(k+1) = A^r(k) \tag{3.17}$$

$$A^i(k+1) = B^r(k) \tag{3.18}$$

N Iterations - Now with $A^i(k{+}1)$ and $B^i(k{+}1)$ obtained from equations (3.17) and (3.18), the process can be repeated for as long as desired. Figure 3.9 shows the block diagram for the TLM link. The equivalent block diagram in z-domain is illustrated in Figure 3.10.

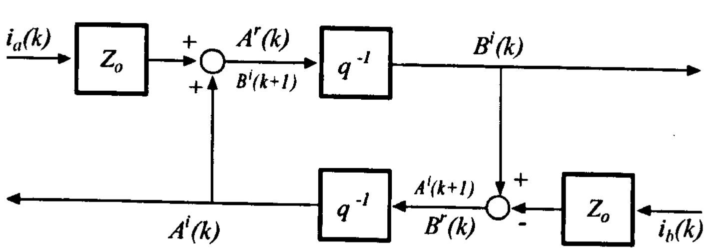


Figure 3.9 TLM link block diagram (time domain)

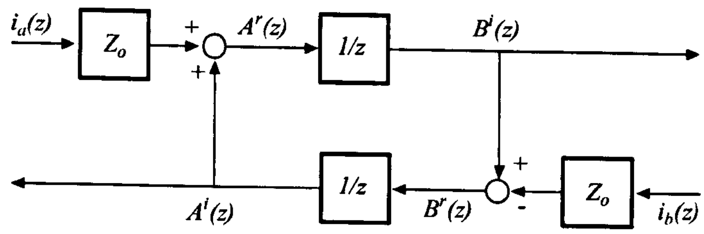


Figure 3.10 TLM link block diagram (z-domain)

Mathematically, the dynamics involve in a loss-less transmission line are governed by equation (3.19).

$$L\frac{\partial^2}{dx^2}y(x,t)=\frac{1}{C}\frac{\partial^2}{dt^2}y(x,t) \tag{3.19}$$

Where, L and C are the respective inductance and capacitance of the line; x is the direction of propagation of energy; and $y(x,t)$ represents either the voltage (e) or the current (i) at a distance x from the port A of the line. The parameters of a TLM link representing a transmission line of length Δx are:

- Speed of propagation $u=\sqrt{LC}$
- Characteristic impedance $Z_0=\sqrt{L/C}$
- Propagation time $\Delta t=\Delta x/u$

Capacitors and inductors are often used as coupling elements between electric circuits (Figure 3.11). In these cases, the TLM link can be used to represent inductive or capacitive properties by choosing appropriate weighting of the elements. For an inductor L, the characteristic impedance of the inductive link is $Z_0=L/\Delta t$, where Δt is the time taken for a pulse to travel a single-trip from one end to the other end. For a capacitor the characteristic impedance of the TLM capacitive link is $Z_0=\Delta t/C$.

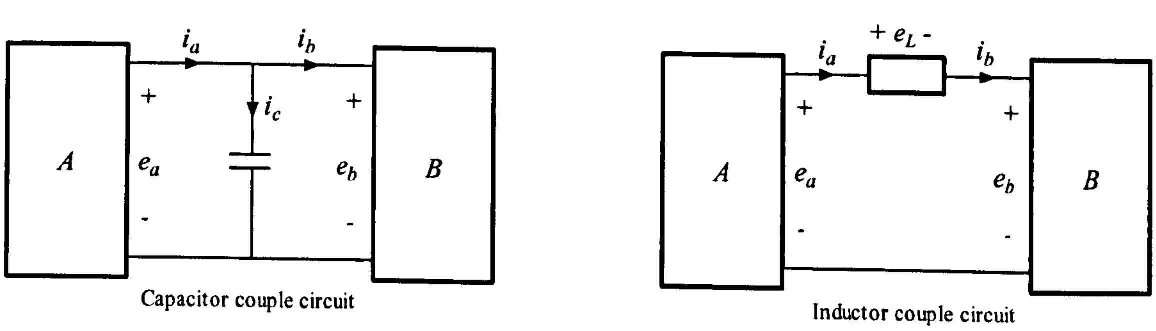


Figure 3.11 Capacitor and inductor coupled circuits

Fung and Hui [44] defined an improved TLM link model based on the fact that fluctuations in the two-port voltages are induced when using TLM links to model capacitors and inductors as coupling elements. In this model the voltage across a capacitor is the average of the two port voltages. The current flowing through the capacitor is the sum of the two port currents, then:

$$e_c = (e_a + e_b)/2 \tag{3.20}$$

$$i_c = i_a - i_b \tag{3.21}$$

Equations (3.13) and (3.14) are replaced by equations (3.22) and (3.23) for the improved TLM link model of a capacitor.

$$B^r(k) = e_c(k) - B^i(k) \tag{3.22}$$

$$A^r(k) = e_c(k) - A^i(k) \tag{3.23}$$

For an inductor model, the voltage across the inductor is the difference between the port-voltages while the current flowing though the inductor is the average of the two port currents. Equations (3.11) and (3.12) are changed by equation (3.26) and (3.27) for the improved TLM link model of an inductor.

$$e_L = e_a - e_b \tag{3.24}$$

$$i_L = (i_a + i_b)/2 \tag{3.25}$$

$$e_a(k) = 2A^i(k) + i_L(k)Z_0 \tag{3.26}$$

$$e_b(k) = 2B^i(k) - i_L(k)Z_0 \tag{3.27}$$

3.2.3 Modelling Non-Linear Elements

A non-linear resistor is simply represented by its own characteristic resistance as in the linear case. The resulting equations with varying coefficients cannot be solved directly with the linear transform in the case when L and C are not constant. This section presents the general TLM formulation by Hui and Christopoulos [35, 36] to deal with such problems as follows: consider the equation $e_{Ln} = d\lambda/dy = d(Li/dt)$ for an inductor where λ is the flux and L is the inductance as a function of current, $L(i)$. When the coefficient or inductance is non-linear, it yields:

$$\frac{d\lambda}{dt} = L\frac{di}{dt} + i\frac{dL}{dt} \tag{3.28}$$

The right-hand terms of the equation cannot be solved easily with the linear transforms. For example, the function of L may be multi-valued and therefore not differentiable. Instead of solving the differential term in that form, $d\lambda/dt$ may be expressed as:

$$\frac{d\lambda}{dt} = L(i)\frac{di}{dt} \quad \text{where } L(i) = \frac{d\lambda}{di} \tag{3.29}$$

If a non-linear inductance is considered, $L(i)$ will be effectively the differential or incremental inductance. $L(i)$ can be determined knowing the non-linear properties of such inductor. Calculating di/dt as the voltage across an inductor of one Henry ($e_L=2di/dt$ and $Z_L=1/(\Delta t/2)$) the discrete transformation can be applied to determine e_{Ln} in equation (3.30). Although $L(i)$ is considered to be current-dependent, it could be any non-linear function. The TLM model of a non-linear inductor is illustrated in Figure 3.11.

$$e_{Ln} = L(i)\frac{di}{dt} = L(i).e_L = L(i)(\frac{2}{\Delta t}i + 2e_L^i) \tag{3.30}$$

Similarly, a voltage-dependent capacitor can be described by a non-linear function $C(e)$. The voltage of this non-linear capacitor can be represented as a non-linear voltage source (Figure 3.12). The capacitor voltage e_{Ch} is given by:

$$e_{Cn}(e) = \frac{Q}{C(e)} = \frac{Z_c * i_c + 2e_c^i}{C(e)} \tag{3.31}$$

where Q represents the electric charge stored in the non-linear capacitor, Z_c is the characteristic impedance of an arbitrary capacitor of one Farad ($Z_c=\Delta t/2$), E^i_c is the incident pulse in the arbitrary capacitor, and i_c is the current of the equivalent branch.

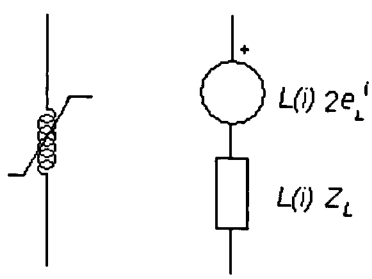


Figure 3.12 TLM Model for a non-linear inductor [35]

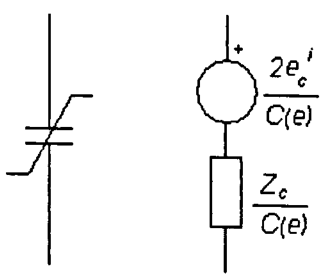


Figure 3.13 TLM Model for a non-linear capacitor [36]

3.3. Comparison Between TLM and ATT Modelling Techniques

The system used by Bartlett and Whalley [55, 58] to demonstrate the ATT method is modelled in this section using both ATT and TLM techniques. The purpose of this exercise is to provide a consistent grade of certainty when comparing results between the transmission line matrix and the analogue transform technique.

Figure 3.14 illustrates a rotor shell assembly commonly used in paper manufacturing units. The rotor assembly refers to the motor drive, including the armature and bearing friction of the motor, front bearing, first rotor shaft, rotor shell, the second rotor drive and the rear bearing. Typically parameter values for the assembly are given in Table B.1

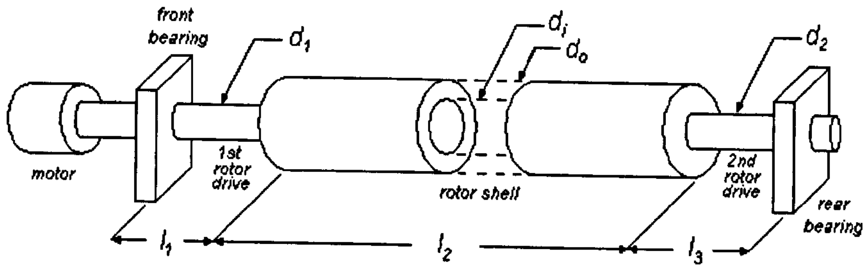


Figure 3.14 Rotor shell arrangement

The motor and supporting bearings are analysed as lumped parameter elements and the rotors as distributed parameter elements. The set of angular velocities and torques shown in Figure 3.15 is specified to evaluate the interaction between elements.

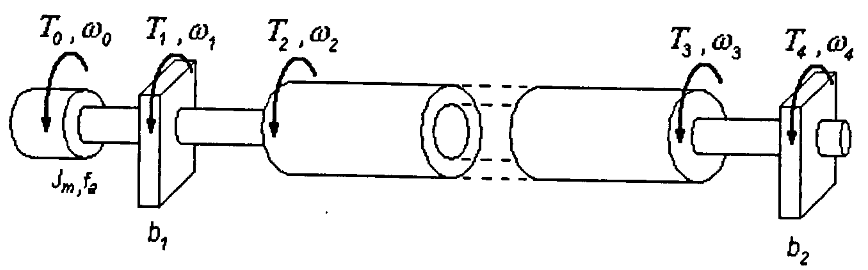


Figure 3.15 Set of variables considered for the rotor shell assembly

3.3.1 Model using the Analogue Transform Technique

Following the procedure summarised in section 3.1, the system can be represented as series of distributed/lumped admittance and impedance modules as shown in Figure 3.16.

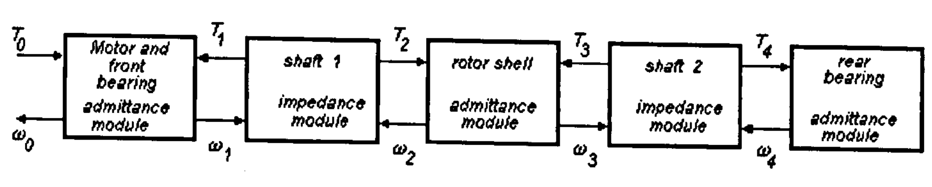


Figure 3.16 ATT Model for the rotor shell assembly

According with ATT theory, model parameters for the shafts (characteristic impedance, ξ_j, and propagation time delay, τ_j) are respectively calculated using equations (3.3) and (3.5), where [58]:

$$L_j = J_j \rho_j \tag{3.32}$$

$$C_j = 1/(G_j J_j) \tag{3.33}$$

Calculated ξ_j and τ_j values for the first shaft, the rotor shell and shaft 2 are contained in table 3.2.

Element	ξ_j	τ_j [ms]
Shaft 1	15.328	0.6245
Rotor	2316.62	4.996
Shaft 2	15.328	0.6245

Table 3.2 Model parameters for the system shafts

The form of the lumped parameter elements is arranged in concordance with that proposed in section 3.1. The final termination (rear bearing) is represented by a respective admittance lumped parameter module. Thus,

$$T_4 = b_2 \omega_4 \tag{3.34}$$

A lumped parameter module represents the motor and front bearing. Thus, its dynamic behaviour is governed by

$$T_0 - T_1 - \omega_0 (f_a + b_1) = J_m \frac{d}{dt} \omega_0 \tag{3.35}$$

The Laplace transform of equations (3.34) and (3.35) for initial conditions equal to zero gives

$$\frac{\omega_4(s)}{T_4(s)} = \frac{1}{b_2} = \frac{1}{0.25} = 4 \tag{3.36}$$

$$\frac{\omega_0(s)}{T_0(s) - T_1(s)} = \frac{1}{J_m s + (f_a + b_1)} = \frac{1}{0.49s + 1} \tag{3.37}$$

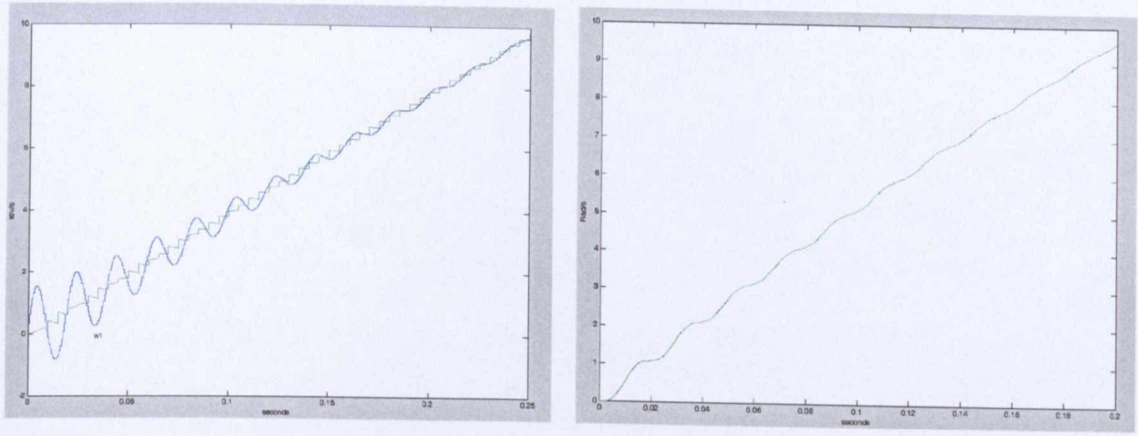
The z-transform is applied to equations (3.36) and (3.37) for a sampling time equal to the lowest propagation time of the system (τ_l=0.6245x10^{-3}), hence in delay representation takes the form

$$\frac{\omega_4(z)}{T_4(z)} = 4 \tag{3.38}$$

$$\frac{\omega_0(z)}{T_0(z) - T_1(z)} = \frac{0.001274}{z - 0.9987} \tag{3.39}$$

Appendix B.2 contains the resultant ATT model for the system in MATLAB/SIMULINK. Simulation results for the established angular velocities and torques are shown in Figures 3.17 and 3.18. The propagation time delay for the rotor shell, τ_2, was approximated from 4.9996 ms to 5 ms, as reported by Bartlett and Whalley [55, 58].

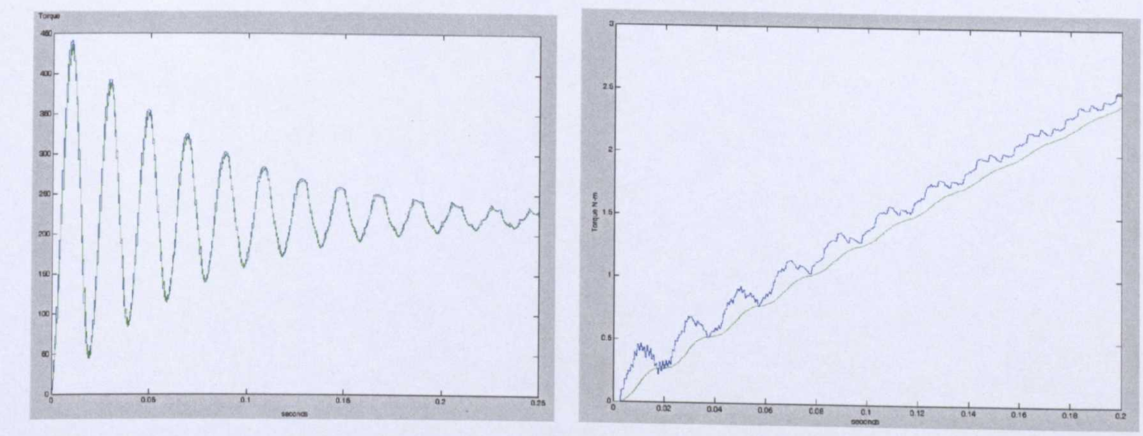
As can be seen in Figure 3.17a, the oscillating behaviour of ω_l around ω_2 illustrates the effect of the distributed parameter characteristics of shaft 1 during the transient period. The torque curves presented in Figure 3.18a also corroborate this similar effect. These results match those presented in [55] and [58].



a) ω_1 (blue line) and ω_2 (green line)

b) ω_3 (blue line) and ω_4 (green line)

Figure 3.17 Results for the established angular velocities (ATT Model)

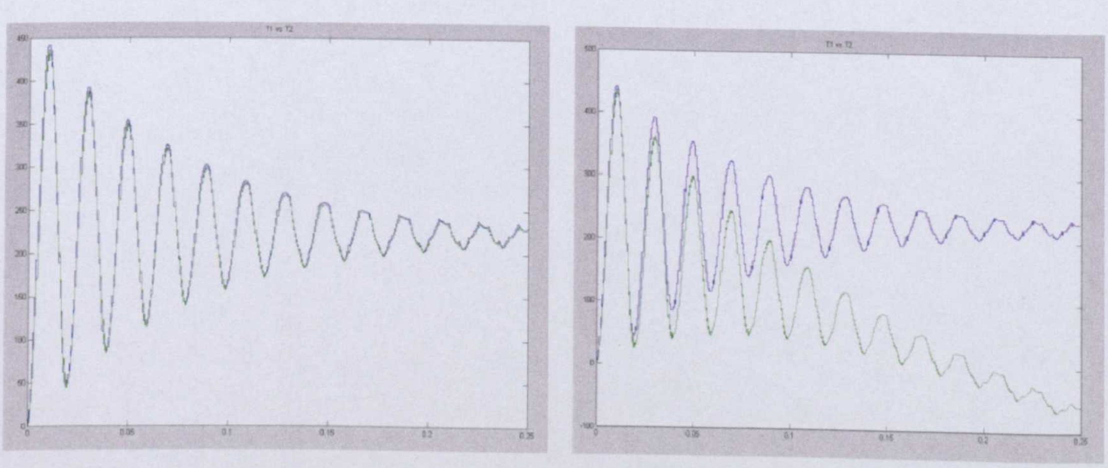


a) T_1 (blue line) and T_2 (green line)

b) T_3 (blue line) and T_4 (green line)

Figure 3.18 Results for the established torques (ATT Model)

Figure 3.19 shows the simulation results when the calculated value for τ_2 is used. Notice the effect of propagation time change from 0.005 to 0.004996 seconds on the results for torque two, T_2, (green line). This effect evidences the sensitivity that the ATT model shows to small changes in the propagation time.



a) $\tau_2 = 5x10^{-3} s$

b) $\tau_2 = 4.996x10^{-3}\ s$

Figure 3.19 Simulation results for a slight change in the propagation time

This sensitivity is attributed to the algebraic loops established when the impedance/admittance modules are connected. Including a unit delay on the feedback signals for each module could eliminate this effect, but it entails an alteration of the ATT modelling equations and therefore a change in the converging speed of the model.

3.3.2 Model using the transmission line matrix method

This section presents the TLM representation of the differential equations that characterise the behaviour of each of the elements presented in section 3.1.

3.3.2.1 Distributed parameter elements

As Partridge, et al demonstrated [29], the behaviour of a shaft subjected to torque about its longitudinal axis can be represented by the following differential equations for angular displacement (θ) and torque (T):

$$\frac{\partial^2 \theta}{\partial x^2} = \frac{1}{u^2} \frac{\partial^2 \theta}{\partial t^2} \tag{3.40}$$

$$\frac{\partial^2 T}{\partial x^2} = \frac{1}{u^2} \frac{\partial^2 T}{\partial t^2} \tag{3.41}$$

Equations (3.40) and (3.41) are wave equations and can be represented by a TLM link as presented in section 3.2. Thus, the characteristic impedance is given by

$$Z_0 = J\sqrt{\rho G} \tag{3.42}$$

The velocity of propagation of the torsional waves will be:

$$u = \sqrt{G / \rho} \tag{3.43}$$

And the time taken for a pulse to travel the length l of the shaft is given by

$$\Delta t = l / u = l\sqrt{G / \rho} \tag{3.44}$$

Where ρ, J, G are the density, polar second moment of area, and the shear modulus of the shaft; d and l the diameter and length of the shaft.

Although each section of the rotor assembly could be represented by its equivalent TLM link, the TLM method needs the same propagation time on each element of the system. A solution is to divide the rotor shell into segments assuring the same propagation time.

Equations (3.42) and (3.43) show for a given material that: Z_0 depends on the geometry of the segment while Δt is dependent on the length. As the three shafts are made from the same material, the solution is to divide the shafts into segments of the same length. Accordingly, the rotor shell is divided in eight segments as illustrated in Figure 3.20. Table 3.3 resumes the TLM parameters for the two shafts and the rotor shell; see equations (B.9) to (B.12) in the Appendix B.2 for more details. Figure 3.21 shows the resultant TLM model.

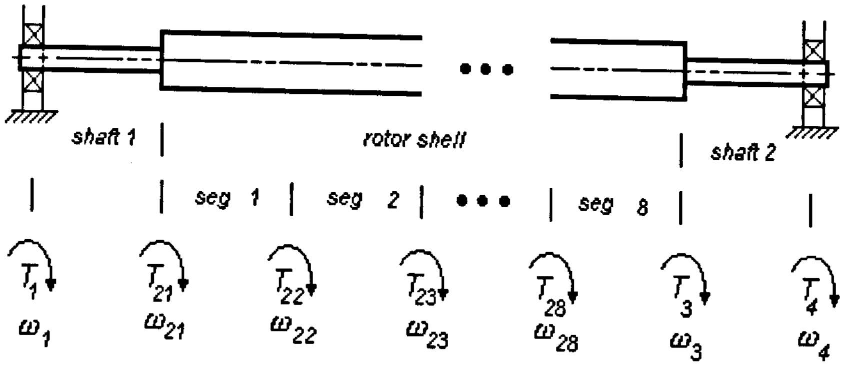


Figure 3.20 Rotor arrangement divided into sections/segments

Element	Z_o	Δt [ms]
Shaft 1	15.3274	0.31225
Rotor Shell segments 1	2316.62	0.31225
Shaft 2	15.3274	0.31225

Table 3.3. TLM model parameters

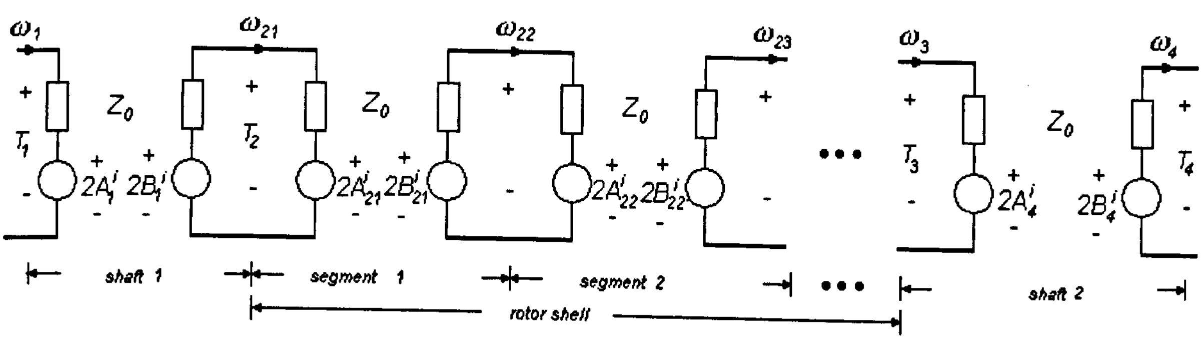


Figure 3.21 TLM model for the rotor arrangement

3.3.2.2 Lumped parameter elements

The dynamic behaviour of the rear bearing (equation (3.34)) has the form

$$f(t) = ax(t) \tag{3.45}$$

According to Table 3.1, the TLM model of this type of equation gives:

$$T_4(k) = b_2\omega_4(k) \tag{3.46}$$

Using the same procedure, the TLM transform of equation (3.35) gives:

$$T_0(k) - T_1(k) - \omega_0(k)(f_a + b_1) = \omega_0(k)Z_m + 2E_m^i(k) \tag{3.47}$$

Where

$$Z_m = J_m /(\Delta t/2) \tag{3.48}$$

$$E_m^i(k+1) = -(\omega_0(k)Z_m + E_m^i(k)) \tag{3.49}$$

The TLM model for equations (3.46) and (3.47) is illustrated in Figure 3.22. Finally the TLM model for the system is completed including the model of the rotor arrangement as illustrated in Appendix B.3.

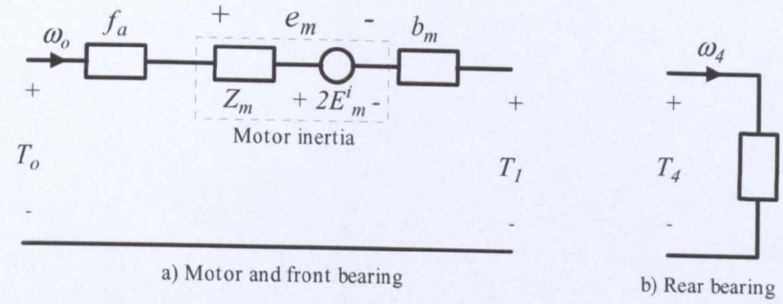


Figure 3.22 TLM models for the lumped parameter elements

3.3.2.3 Simulation Results

A program was built in SIMULINK in order to simulate the TLM model of the system. As was done with the ATT model, slightly changes of the propagation time were performed to see the effect on the response. Simulation results showed that the TLM model was not affected, however the signal oscillations on the transient period seems to be less damped for the TLM model (see Figure 3.23).

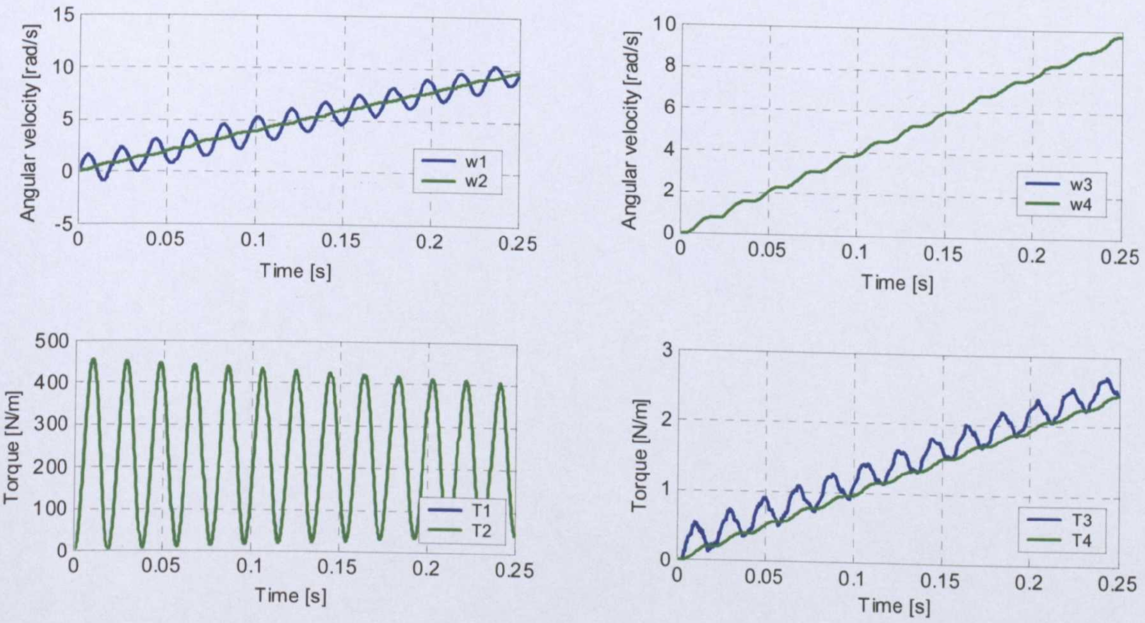


Figure 3.23 Simulation results for the rotor shell assembly TLM model

A detailed analysis of the model showed that this behaviour is caused by the modelling error inherent to the TLM stub unit used to model the differential term in equation (3.35). The error magnitude is dependent on the sample time as illustrated in Figure 3.24 (See Appendix B.4).

As observed in Figure 3.24, the TLM model response lags the response of the transfer function of the system. This effect influences the response of the TLM rotor shell model reducing the overall damping factor; and therefore, increasing the oscillation of the model response as shown in Figure 3.23.

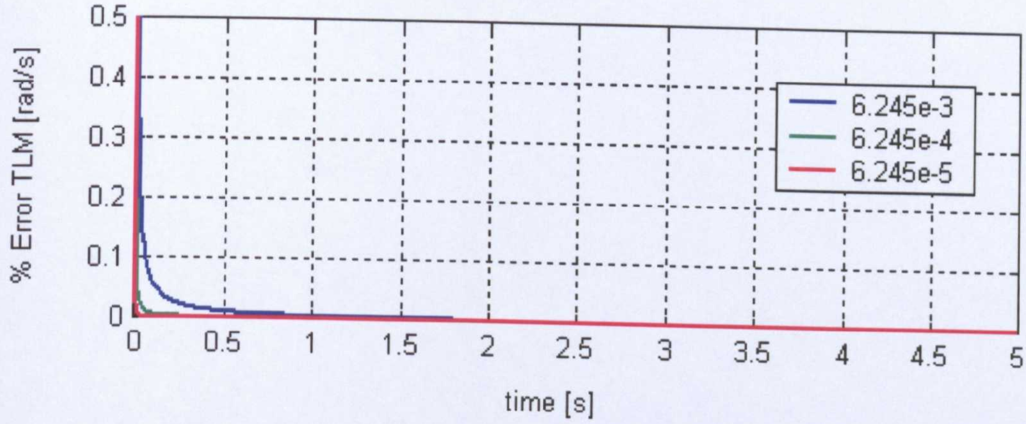


Figure 3.24 Modelling error at various sample rates for the TLM model of equation (3.35)

Accordingly, the selection of the sample time becomes more crucial when the dynamic behaviour of the system comprises various differential terms as could be the case for models of complex systems like axis feed drives. An alternative model for the differential term had to be devised in such way that all the modelling advantages of the TLM technique could be used without compromising the model response by the sample time. That is the subject of the next section.

3.4 The Modified TLM Stub

The first thought for the reduction of the error on a TLM stub unit response is to reduce the sample time. This is feasible when modelling systems composed by lumped-parameter elements, like electric circuits. In fact, the TLM theory was addressed in the first instance to the modelling of that type of systems. The magnitude of the sample time is dependent on a desired computation speed or the available memory in this case.

Conditions change when modelling hybrid systems (a mix of lumped-parameter and distributed-parameter elements). The sample time is driven by the physical properties of the considered distributed-parameter elements. Taking the case of the shaft subject to torsion: the velocity of propagation (u_t) depends on the material and the characteristic impedance (Z_0) depends on the geometry. Therefore, the sample time (propagation time) is dependent on the length of the segment. The number of segments in which the shaft is divided has to be increased to match a reasonable sample time. This action becomes complicated if the shaft is built with various segments featuring different geometries, lengths and materials.

An ideal solution implies the specification of a TLM stub more robust to changes of the sample time. The denominated "*modified TLM stub*" is derived taking into account this precept. Consider the TLM transform for a differential term (Table 3.1):

$$e(t) = a\frac{d}{dt}i(t) \tag{3.50}$$

$$Z_0 = a/(\Delta t/2) \tag{3.51}$$

$$e(k) = Z_0 i(k) + 2E^i(k) \tag{3.52}$$

$$E^i(k+1) = E^i(k) - e(k) = -Z_0 i(k) - E^i(k) \tag{3.53}$$

One way to reduce the oscillations on the response (equation 3.52) is to average the actual output with the previous one:

$$e(k) = \frac{e(k) - e(k-1)}{2} \tag{3.54}$$

Rearranging and substituting equation (3.52) into (3.54) gives:

$$2e(k) = Z_0 i(k) + 2E^i(k) + Z_0 i(k-1) + 2E^i(k-1) \tag{3.55}$$

Substituting equation (3.53) into (3.55)

$$e(k) = \frac{Z_0}{2}(i(k) - i(k-1)) \tag{3.56}$$

And then, by substituting equation (3.51) into (3.56) gives

$$e(k) = \frac{a}{\Delta t}(i(k) - i(k-1)) \tag{3.57}$$

Or, by redefining Z_o as

$$e(k) = Z_0 i(k) + E^i(k) \tag{3.58}$$

Where, $$Z_o = \frac{a}{\Delta t} \tag{3.59}$$

$$E^i(k+1) = -Z_0 i(k) \tag{3.60}$$

Equations (3.58 – 3.60) comprise the scattering algorithm that governs the discrete process of the *modified TLM stub* for differential terms.

Assuming q^{-1} the delay (Δt) for a pulse, the modified TLM transform for the differential term can be expressed in z-domain ($z^{-1} = q^{-1}$) by

$$e(z) = \frac{a}{\Delta t}\left(\frac{z-1}{z}\right)i(z) \tag{3.61}$$

The modified TLM transform for the integral term in z-domain is defined as the inverse of equation (3.61), hence

$$e(z) = \frac{\Delta t}{a}\left(\frac{z}{z-1}\right)i(z) \tag{3.62}$$

Note that the modified TLM transform for an integral term is equivalent to the backward

Euler integration method when $a = 1$.

The equations in time domain for the integral term are calculated applying the z-inverse transform to equation (3.62), thus

$$e(k) = Z_0 i(k) + E^i(k) \tag{3.63}$$

Where,
$$Z_o = \frac{\Delta t}{a} \tag{3.64}$$

$$E^i(k+1) = e(k) \tag{3.65}$$

Table 3.4 describes the modified TLM and the equivalent Z transform for integral, differential and proportional terms. Figure 3.25 shows the modelling error at various sample rates when the modified TLM transform is applied to equation (3.35).

Continuous model	TLM transform (Discrete model)	Equivalent Z-domain transform
$e(t) = ai(t)$	$e(k) = ai(k)$	$e(z) = ai(z)$
$e(t) = a\frac{d}{dt}i(t)$	$Z_0 = a/\Delta t$ $e(k) = Z_0 i(k) + E^i(k)$ $E^i(k+1) = -Z_0 i(k)$	$e(z) = Z_o\left(\frac{z-1}{z}\right)i(z)$
$e(t) = \frac{1}{a}\int i(t)dt$	$Z_0 = \Delta t / a$ $e(k) = Z_0 i(k) + E^i(k)$ $E^i(k+1) = e(k)$	$e(z) = Z_o\left(\frac{z}{z-1}\right)i(z)$

Table 3.4 The modified TLM transform

The modified TLM transform was applied to equation (3.35) as was performed with the TLM model. Figure 3.25 shows the calculated percentage error for various time steps.

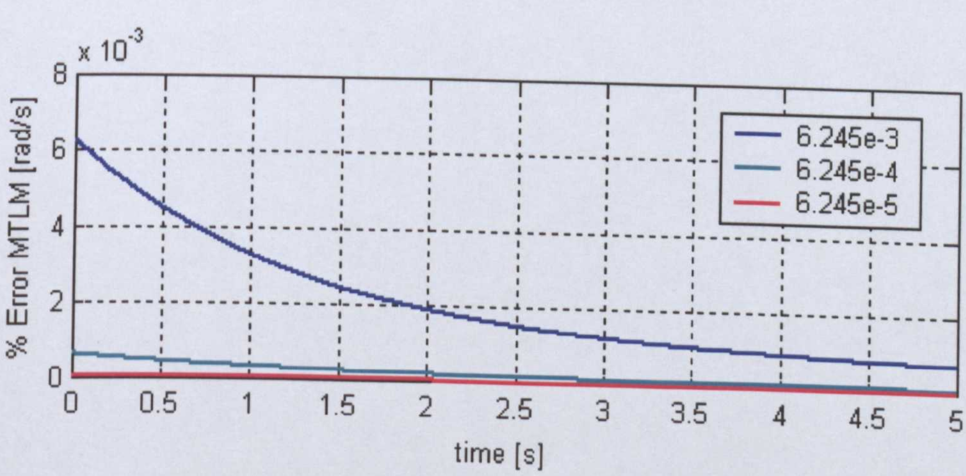


Figure 3.25 Modelling error for the modified TLM model of equation (3.35)

A comparison between the results for the TLM stub and the modified TLM stub is illustrated in Figure 3.26 (sample time = 6.245 e-3 s). The following features can be highlighted:

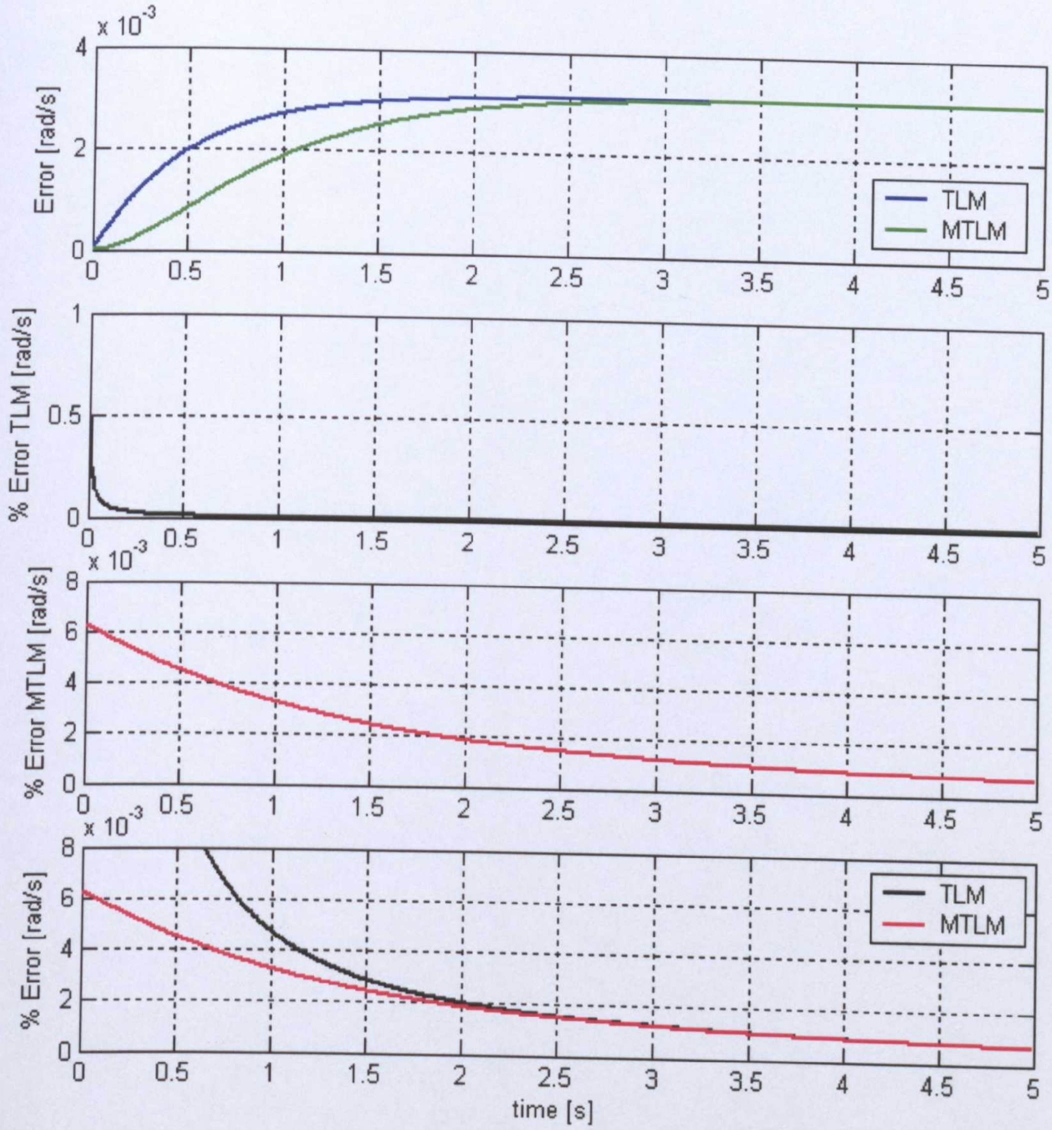


Figure 3.26 TLM and modified TLM modelling error comparison

- The maximum percentage error for the modified TLM stub (0.006% on the red line) is approximately 1.2% of the maximum percentage error for the TLM stub (0.5% on the black line). As a result a reduction of 35% on the mean square error is achieved using the modified TLM stub algorithm.
- The number of operations for the scattering algorithm has been reduced by to 40% using the modified TLM stub.

The modified TLM transform was applied to the modelling of the rotor shaft arrangement as presented in Appendix B.5. Results for the stated variables match those presented in [55] and [58] as shown in Figure 3.27. Slight changes of the propagation time were performed to see the effect on the response. Simulation results showed that the modified TLM model was not affected.

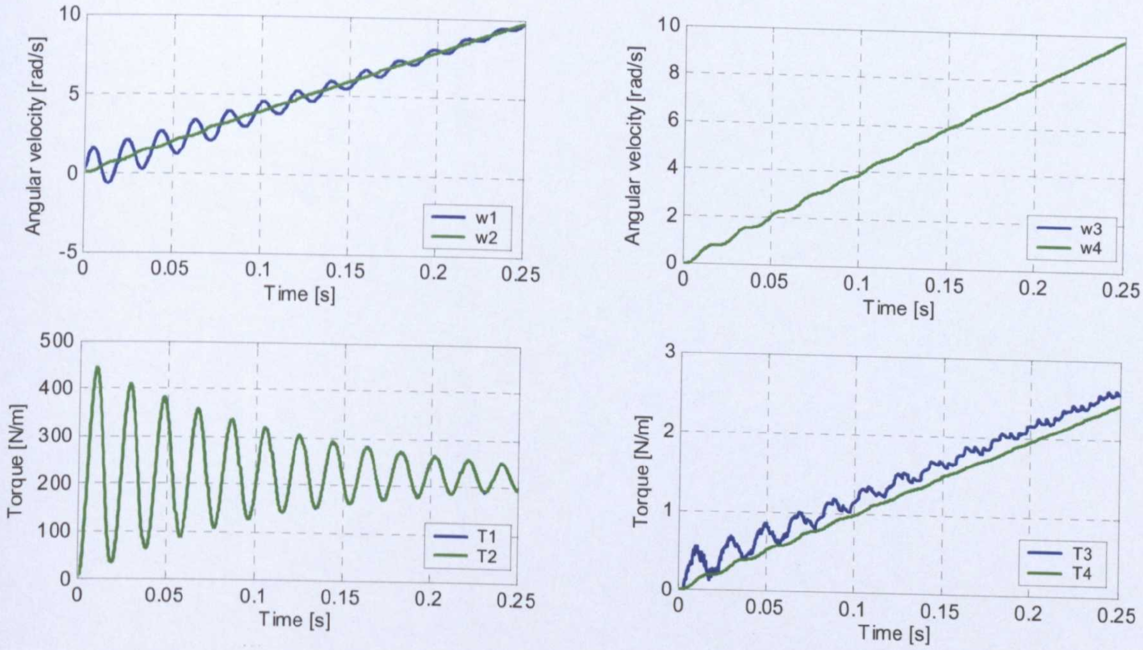


Figure 3.27 Simulation results for the rotor shell assembly (modified TLM stub model)

3.5 Conclusions

In the modelling of systems with transmission line techniques, a hybrid system is seen as a series of distributed and lumped parameter elements. The modelling process begins by defining the differential equations that govern the dynamics of each element. Then, analytical or numerical transform techniques are applied to find a solution.

In the analogue transform technique distributed parameter elements are modelled by general impedance/admittance discrete representations derived from the solution of two particular cases of the equation for a transmission line. For the modelling of lumped parameter elements, the Laplace transform is used to convert the differential equations that represent the element from the time domain to the s-domain in order to obtain the transfer function. The transfer function is then converted to the z-domain via the z transform.

The TLM transform can be utilised like the Laplace or Z transformation to substitute integral and differential equations with the appropriate transform used in each case. A discrete model in time-domain is obtained directly to reach a solution in a stepping routine. Then, the discrete transform described in Table 3.4 (The modified TLM stub) can be employed to model lumped-parameter elements.

A TLM link is the discrete representation of a transmission line governed by a hyperbolic differential equation (the wave equation). Thus, Table 3.4 can be extended including this TLM unit to model distributed-parameter elements governed by PDEs of the same type (see Table 3.5).

Continuous model	TLM transform (Discrete model)	Equivalent Z-domain transform
$e(t) = ai(t)$	$e(k) = ai(k)$	$e(z) = ai(z)$
$e(t) = a\frac{d}{dt}i(t)$	$Z_0 = a / \Delta t$ $e(k) = Z_0 i(k) + E'(k)$ $E'(k+1) = -Z_0 i(k)$	$e(z) = Z_o\left(\frac{z-1}{z}\right)i(z)$
$e(t) = \frac{1}{a}\int i(t)dt$	$Z_0 = \Delta t / a$ $e(k) = Z_0 i(k) + E'(k)$ $E'(k+1) = e(k)$	$e(z) = Z_o\left(\frac{z}{z-1}\right)i(z)$
$a\frac{\partial^2}{\partial x^2}f(x,t) = \frac{1}{b}\frac{\partial^2}{\partial t^2}f(x,t)$ $f(x,t) = e(x,t)$ or $f(x,t) = i(x,t)$	Link of length Δl $u = \sqrt{ab}$ $Z_o = u / b$ $\Delta t = \Delta l / u$	$E_a^i(z) = \frac{1}{z}\left(E_b^i(z) - Z_o i_b(z)\right)$ $E_b^i(z) = \frac{1}{z}\left(E_a^i(z) + Z_o i_a(z)\right)$

Table 3.5 Extended TLM transform table

A TLM model is considered an equivalent electric representation of a system. In TLM models of mechanical systems, for example, voltage sources represent forces/torques, and electric currents represent velocities.

A transmission line segment representing a unity value element is used to model a non-linear element. Thus, a non-linear variation in the element value is treated as part of the forcing function. This procedure makes TLM technique a very useful tool for the modelling and simulation of linear and non-linear systems.

The possibility offered to decouple a system model using the improved link approach makes TLM a suitable method for the implementation of models for simulation on parallel processors. It also offers the possibility of implementations in real time by reducing the complexity of simulation algorithms.

A rotor shell assembly commonly used in paper manufacturing was modelled using both ATT and TLM techniques in order to compare results between the approaches. Data from simulations show that the two techniques give the same results. However, ATT showed sensitivity to small changes in the propagation times. It was also found that the selection of the sample time for the TLM technique becomes crucial when looking for accurate results in models of complex systems. For comparison purposes, it should be noted that for a system model:

- The parameter ξ of the ATT model is equivalent to the Z_o parameter in the TLM model.
- The propagation time to be used for the ATT model must be twice the value of Δt calculated for the correspondent TLM model.

As a result from this exercise, a new approach for the modelling of complex shafts was derived using the TLM (Partridge et al [29]) representation for a simple shaft, and the approximation of continuous space feature of TLM technique.

A new model for the TLM stub unit was derived in order to overcome the restrictions highlighted by the results of the rotor shell modelling exercise. A model for the rotor shell was built using the new model denominated the "*modified TLM stub*". Simulation results verified the robustness and accuracy of the model. The main improvements over the original TLM stub model are:

- A reduction of 40% on the number of mathematical operations, and
- A reduction of almost the 35 % on the mean square modelling error.

3.6 Project Plan

The modified TLM was selected as the modelling technique due to advantages of the method over ATT. The following project plan was established to accomplish the objectives of this research:

- To derive a CNC machine tool TLM model for a single-axis feed drive (including for torsional and axial mechanical vibrations, and non-linearities such as backlash, dynamic and static frictional forces).
- To validate the model against experimental data collected from a real system considering various displacements at different feed rates and positions of the feed drive.
- To build a two-axis model on the basis of the single-axis model (including linear and circular interpolation).
- To include in the model the effect of geometric errors (measured with specialised equipment such as laser, ball-bar and/or electronic precision level) and compare simulated results with measured ones.
- To carry out measurement trials at the machine to identify modal parameters (damping factors and resonant frequencies).
- To use Wavelet Transform techniques to detect resonant states and damping factors of machine elements from measured data.

In the next chapter, the derivation of the adequate TLM models for the elements of a CNC machine tool feed drive is presented.

4. MODELLING THE ELEMENTS OF CNC MACHINE TOOL FEED DRIVE USING THE MODIFIED TRANSMISSION LINE MATRIX METHOD

This chapter presents the development of transmission line models for the elements of a typical arrangement of a CNC feed drive. The models are built by developing the equations that characterise the behaviour of each element, and then by representing those equations with corresponding TLM transformations. Descriptions of closed loop control principles, space vector control, and selected models for backlash and friction are included.

4.1. Introduction to CNC Machine Tool Digital Drives

The basic function of CNC in a machine is the automatic, precise and consistent control of its directions of motion, which are classified as linear (drive along a straight line) and rotary (drive along a circular path). The set of elements performing a linear direction of motion is named *feed drive*. It consists of mechanical, power electronic, and CNC units.

The mechanical elements of a feed drive usually converts angular motions of a motor to linear transverse velocity of a table supported on guide ways. However, recent developments in motor technology have resulted in the implementation of direct actuators that supply force and velocity to the table directly without the need for a mechanical transmission. That is, typical transmissions such as gearboxes, belts and pulleys, ball screws, and rack and pinions are replaced by linear motors. The power electronic units supply the voltage to the motor and signals from the limit switches of the system. A computer unit, and position and velocity sensors for the drive mechanism make up the CNC as defined by Lyang et al. [61].

Although machine tool designs vary immensely, the mechanical configuration of a feed drive is largely standardised. According to Braasch [62], the most common configuration used is a *ballscrew* coupled to a servomotor. A ballscrew assembly consists of a precision ground screw shaft, a nut (the outer race) with an internal groove, and a circuit of precision steel balls that recirculate in the grooves between the screw and nut as defined by Degenova [63]. Figure 4.1 shows the scheme of a conventional CNC feed drive. The motion control of a feed drive can be summarised in the following actions:

- A motion command in a NC program executed within the Motion Control Unit (MCU) signals the motor (through a drive) to rotate at a defined velocity.
- The rotation of the motor in turn rotates the screw shaft, which interacts with the nut causing linear motion of the table.
- A feedback device attached to the rotor of the motor allows the control to confirm that the commanded rate of rotations has taken place.

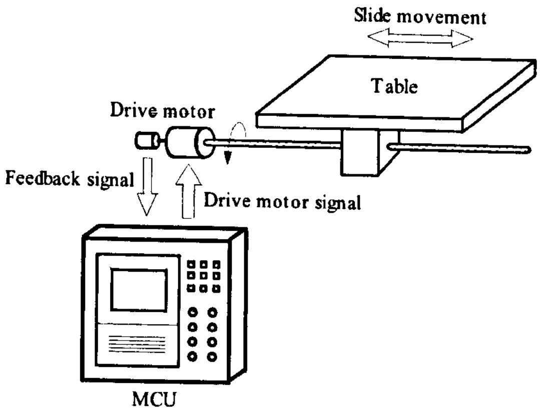


Figure 4.1 Conventional CNC feed drive [63]

A motion control system of a feed drive can include an interpolator, a position loop, a velocity loop, current loop, and the motor commutation and power stages (inverter). An *electrical drive* performs several of these functions and a *motion controller* performs the remainder. Different modes of operation are provided according to the functions the electrical drive develops, the three more frequent configurations are: velocity, current, and power block modes. In velocity drives, the motion controller provides the velocity command to the drive; in current drives, the command is for current. In power blocks, the drive is just dedicated to provide the power stage output to the motor and the feedback to the current controller [64]. Figure 4.2 shows the block diagram of a feed drive configured in current mode.

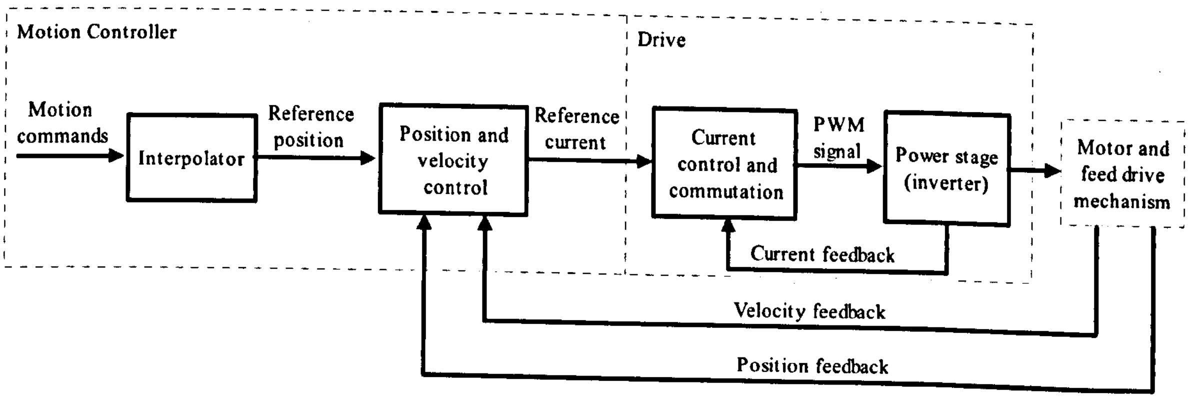


Figure 4.2 Block diagram of a CNC feed drive [64]

Industrial drive technology has advanced considerably over the past several years. The largest advances have been in the area of digital control algorithms. In the past, drives were predominately analogue. Today, *digital drives* provide numerous opportunities to improve the performance of a system without requiring expensive mechanical solutions. In an *analogue drive* the velocity and current loops are closed using analogue components (such as operational amplifiers). The gains are set using passive components (such as resistors, potentiometers, and capacitors). Figure 4.3 shows the diagram of a classical *analogue drive* in current mode.

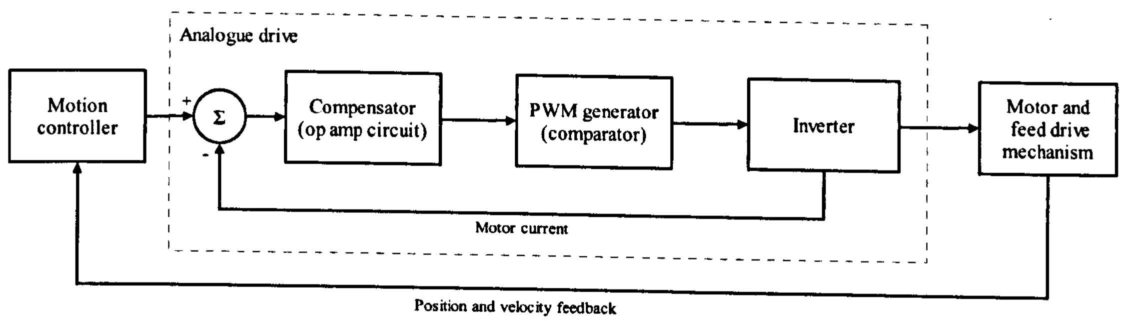


Figure 4.3 Typical analogue drive [65]

The motion controller generates the current command signal. The drive monitors the motor winding current and the angular displacement of the motor shaft. A compensator takes the difference between the commanded and monitored current signals and generates a voltage command signal. Next, a Pulse Width Modulated (PWM) signal is applied to an inverter, which switches the correct voltages to the motor. In contrast, in a *digital drive,* the functions of most of the analogue circuitry (summing amplifier, compensator and PWM generator) are replaced by software running on a microprocessor or a Digital Signal Processor (DSP) as illustrated in Figure 4.4; the power inverter remains the same [65].

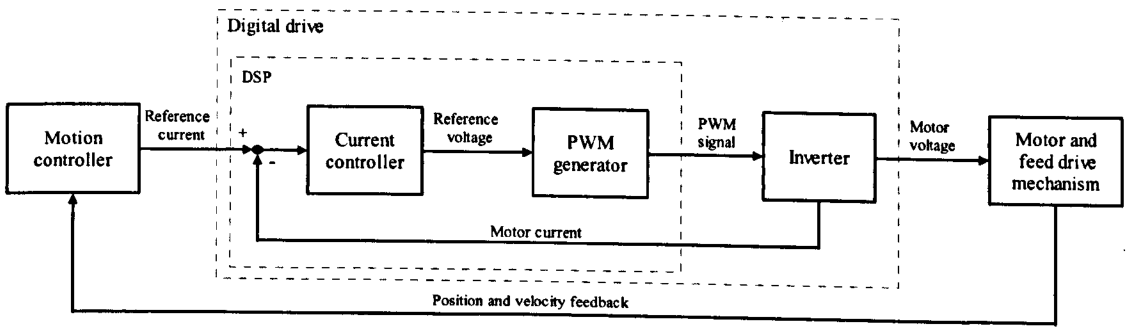


Figure 4.4 Typical digital drive

Digital drives provide several advantages over analogue drives [66]:

- *Flexibility* - Digital drives can be reconfigured digitally. In most cases, there is no need to change a "personality card" or adjust a potentiometer with a digital drive. Many parameters can be changed during operation, which simplifies configuration of the motion system.
- *Gains are set in the software* then parameters can be transferred precisely from one drive to another. Analogue drives require passive component changes or, more commonly, a potentiometer adjustment. Unfortunately, it is not always practical to adjust potentiometers with accuracy.
- *Supports digital communications*- eliminating most noise problems in servo systems. In addition, digital communication greatly reduces the number of wiring interconnections of servo systems.

In contrast, the key advantages of analogue drives are simplicity to configure and less expensive than digital drives, especially in low power (under 500 Watts) applications.

4.2 Permanent Magnet Synchronous Motor (PMSM)

In a synchronous motor, the wire is coiled into loops (stator windings) and placed into slots in the motor housing. Each loop is identified as a phase stator, and the number of poles is determined by how many times a phase winding appears. The mechanical angle between phase stators is the resultant of dividing 360 degrees by the number of phases of the motor. Figure 4.5 illustrates a stator with three-phase windings and two poles. Phases a, b and c are placed 120 degrees apart.

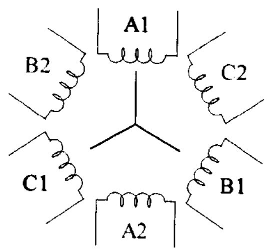


Figure 4.5 Two-pole stator winding [67]

In a permanent magnet synchronous motor, permanent rare-earth magnets are glued onto the rotor. This design leads to a low rotor inertia thus providing fast acceleration and high overload torque ratings. The motor operation relies on the generation of a magnetic rotating field by applying sinusoidal voltages to the stator phases of the motor. A resulting sinusoidal current flows in the coils and generates the rotating stator flux. The rotation of the rotor shaft is then created by attraction of the permanent rotor flux with the stator flux. The motor is called "synchronous" because the rotor operates in synchronism with the rotating magnetic field. To establish the correct polarity of the stator's magnetic field, the position of the permanent magnet rotor with respect to the rotating magnetic field of the stator must be monitored. A feedback device known as *rotary encoder* provides this information. On PMSM, the encoder gives the absolute position of the rotor within one revolution.

In electric motor theory, two measures of position and velocity are generally defined: *mechanical* and *electrical*. The mechanical position, θ_m, is related to the rotation of the rotor shaft. When the rotor shaft has accomplished 360 mechanical degrees, the rotor is back in the same position where it started. The electrical position, θ_e, of the rotor is related to the rotation of the rotor magnetic field. In Figure 4.6, the rotor needs only to move 180 mechanical degrees to obtain an identical magnetic configuration as when it started. The electrical position of the rotor is then related to the number of magnetic pole pairs 'p' on it.

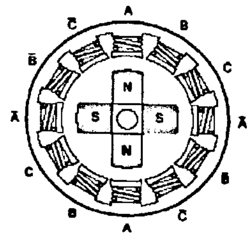


Figure 4.6 Three-phase motor with four magnet poles (Two pole pair) [67]

Thus, the electrical position of the rotor is linked to the mechanical position of the rotor by the expression

$$\theta_e = p\theta_m \tag{4.1}$$

Then, a similar relationship also exists towards electrical velocity ω_e and mechanical velocity ω_m.

$$\omega_e = p\omega_m \tag{4.2}$$

4.2.1 Electrical Equations

The voltage relations of a synchronous machine are given by the following general equation presented by Moreton [68].

$$[e] = \frac{d}{dt}[\psi] + [R][i] \tag{4.3}$$

Where R is the winding resistance of the stator which is assumed to be equal for all stator windings, Ψ, the flux linkage, is given by $[L][i]$. The matrix $[L]$ contains the winding self-inductances. The phase voltages and currents are contained in the column vectors $[e]$ and $[i]$. Most of the coefficients of this set of differential equations are periodic functions of the rotor angle θ_e and therefore complex functions of time. To avoid the time dependence of the coefficients a coordinate system attached to the rotor is used (d-q). This coordinate system rotates at the velocity of the electrical velocity of the rotor and the d axis is aligned with the electrical position of the rotor flux. In this coordinate system, the electrical expression of the torque becomes independent from θ_e . An α-β coordinate system on the stator, is used to illustrate the conversion between the a-b-c and the d-q coordinate systems, as presented by Simon [69]: The three sinusoidal currents created by the 120 degrees phase shifted voltages (e_a, e_b, e_c) applied to the motor are also 120 degrees phase shifted one from another. The stator current vector i_s is represented in Figure 4.7 in the 3-phase static coordinate system (a, b, c).

$$i_s = i_a + i_b \exp^{j2\pi/3} + i_c \exp^{j4\pi/3} \tag{4.4}$$

$$i_a + i_b + i_c = 0 \tag{4.5}$$

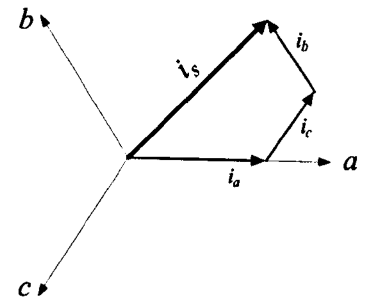


Figure 4.7 Stator current vector

The stator current vector i_s can also be generated by a bi-phase system placed on a fixed α-β coordinate system as shown in Figure 4.8. This coordinate transformation is called *Clarke Transformation.*

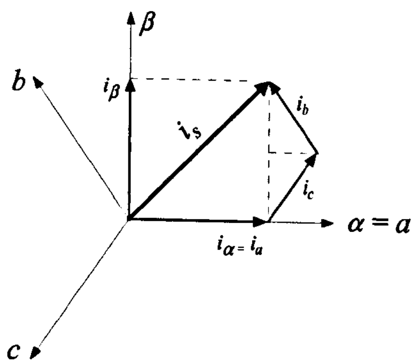


Figure 4.8 Clarke transformation ($a, b, c \rightarrow \alpha, \beta$)

The projection of the stator current in the α, β frame gives

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1/\sqrt{3} & 1/\sqrt{3} \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} \tag{4.6}$$

The conversion from α, β to a, b, c system is denominated the inverse Clark transformation. This transformation is given by equation (4.7).

$$\begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -0.5 & -\sqrt{3}/2 \\ -0.5 & \sqrt{3}/2 \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} \tag{4.7}$$

In the α, β frame, the expression of the torque is still dependent on the position of the rotor flux (θ_e). To remove this dependence, the α, β vectors are projected to the d, q system, which rotates at the electrical velocity of the rotor (ω_e). This transformation is known as inverse *Park Transformation* (Figure 4.9). The equation corresponding to this transformation is:

$$\begin{bmatrix} i_d \\ i_q \end{bmatrix} = \begin{bmatrix} \cos\theta_e & \sin\theta_e \\ -\sin\theta_e & \cos\theta_e \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} \tag{4.8}$$

The Park transformation ($d, q \rightarrow \alpha,\beta$) is given by

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \begin{bmatrix} \cos\theta_e & -\sin\theta_e \\ \sin\theta_e & \cos\theta_e \end{bmatrix} \begin{bmatrix} i_d \\ i_q \end{bmatrix} \tag{4.9}$$

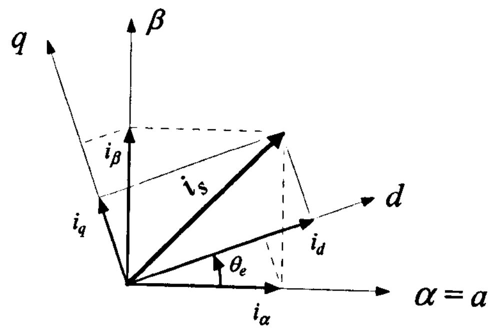


Figure 4.9 Inverse Park transformation ($\alpha,\beta \rightarrow d, q$)

The equations of performance for a PMSM in the *d-q coordinate system* representation are:

$$L_d \frac{d}{dt} i_d = e_d - Ri_d + L_q \omega_e i_q \tag{4.10}$$

$$L_q \frac{d}{dt} i_q = e_q - Ri_q - L_d \omega_e i_d - \lambda\omega_e \tag{4.11}$$

In which, λ signifies the amplitude of the flux induced by the permanent magnets of the rotor in the stator phases; L_d and L_q are the d and q axis inductances. Subscripts d and q are used to identify the direct and quadrature currents/voltages respectively. The generated electromagnetic torque T_e, is defined as

$$T_e = 1.5p(\lambda i_q + (L_d - L_q) i_d i_q) \tag{4.12}$$

If the current $i_d \rightarrow 0$, $T_e \rightarrow 1.5p\lambda i_q$; hence the generated electromagnetic torque can be approximated using the torque constant of the motor, k_T:

$$T_e = k_T i_q \tag{4.13}$$

For a motor built with a uniform air gap (round rotor) $L = L_d = L_q$, then

$$T_e = 1.5p\lambda i_q \tag{4.14}$$

The factor $p\lambda$ in equations (4.12) and (4.14) is known as the electric constant of the motor (k_e). The model for the motor is presented in Figure 4.10.

Considering a motor with round rotor and a simulation time step t_s, the TLM transform of equations (4.10), (4.11) and (4.13) gives (see Table 3.2):

$$Z_L i_d(k) + E_d^i(k) = e_d(k) - Ri_d(k) + L\omega_e(k) i_q(k) \tag{4.15}$$

Where,

$$Z_L = L / t_s \tag{4.16}$$

$$E_d^i(k+1) = -i_d(k) Z_L \tag{4.17}$$

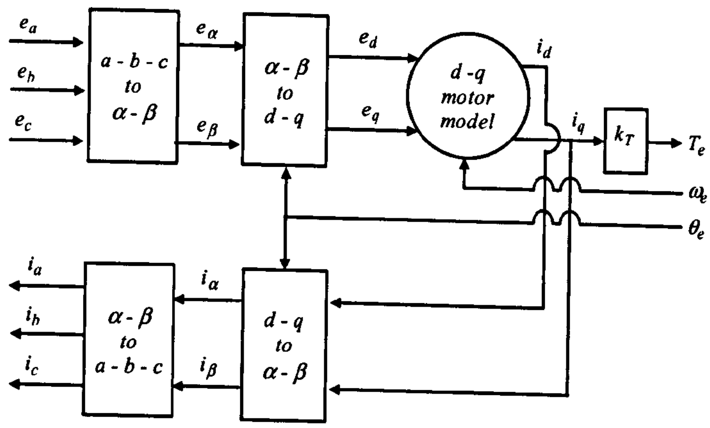


Figure 4.10 Block diagram of the electrical equations of the motor

$$Z_L i_q(k) + E_q^i(k) = e_q(k) - Ri_q(k) - \omega_e(k)(Li_d(k) + \lambda) \tag{4.18}$$

$$E_q^i(k+1) = -i_q(k) Z_L \tag{4.19}$$

$$T_e(k) = k_T i_q(k) \tag{4.20}$$

Rearranging equations (4.16) and (4.18) provides:

$$e_d(k) - E_d^i(k) - (Z_L + R)i_d(k) = -\omega_e(k) Li_q(k) \tag{4.21}$$

$$e_q(k) - E_q^i(k) - (Z_L + R)i_q(k) = \omega_e(k)(Li_d(k) + \lambda) \tag{4.22}$$

For simulation purposes, it can be assumed that the terms on the left side of equations (4.21) and (4.22) are the resultant of the operations on the right side of the equations. Thus, these equations are rewritten as:

$$i_d(k) = M_{dq}(e_d(k) - E_d^i(k) + \omega_e(k-1) Li_q(k-1)) \tag{4.23}$$

$$i_q(k) = M_{dq}(e_q(k) - E_q^i(k) - \omega_e(k-1)(Li_d(k-1) + \lambda)) \tag{4.24}$$

Where,

$$M_{dq} = 1/(Z_L + R) \tag{4.25}$$

4.2.2 Mechanical Equations

Given the mass moment of inertia of the rotor (J_m), the load torque (T_m), the frictional torque at the bearings (b_m) and the mechanical velocity of the motor, the energy conversion process in the motor is governed by the equation

$$T_e - T_m = b_m \omega_m + J_m \frac{d}{dt}\omega_m \tag{4.26}$$

The TLM transform of equation (4.26) for the considered propagation time t_s gives:

$$T_e(k) - T_m(k) = b_m \omega_m(k) + Z_m \omega_m(k) + E_m^i(k) \tag{4.27}$$

Where,

$$Z_m = J_m / t_s \tag{4.28}$$

$$E_m^i(k+1) = -Z_m \omega_m(k) \tag{4.29}$$

The corresponding TLM model is presented in Figure 4.11.

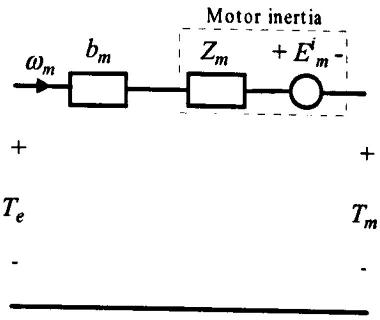


Figure 4.11 TLM model of the mechanical dynamics of a motor

4.3 Controller Model

The control for an axis feed drive is generally implemented with an algorithm following the principle of cascade control in conjunction with feed forward control. Most applications require position control in addition to velocity control [70] The most common way to provide position control is to add a position loop "outside" the velocity loop (as shown in Figure 4.12), which is known as *cascade control.* Machine tools normally operate based on this principle because it offers significant advantages. The subsequent controllers in cascade control compensate directly for the disturbances allowing the primary control to function without the effect of these disturbances. Then, the inner control loop is protected because the outer control loop is limiting its input value. *Feed forward* control improves trajectory tracking compensating for the effect of disturbances before they affect the response of the controlled system as Seborg et al [71].

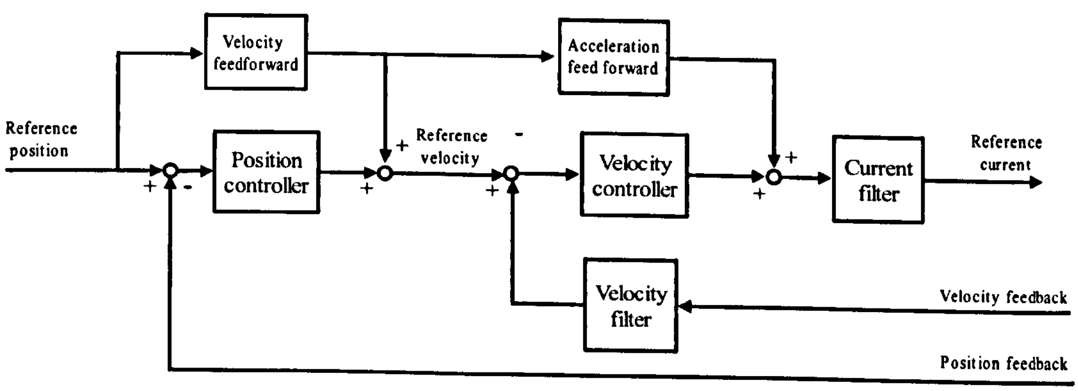


Figure 4.12 Cascade control with velocity and acceleration feed forward [71]

The velocity feed forward path connects the reference position to the velocity loop through the gain, k_{ff}. When the position profile changes, the velocity feed forward transfers immediately that change to the velocity command. This speeds up the system response relative to relying solely on the position loop. The primary shortcoming of velocity feed forward is that it induces overshoot when changes on the velocity direction occur.

Acceleration feed forward eliminates the overshoot caused by velocity feed forward without reducing loop gains. As illustrated in Figure 4.12, the basic idea of feed forward control is to inject the position and velocity set points (reference) and their correspondent first derivatives at appropriate points in the control loops. The following errors are minimised thus achieving a significant increase in the dynamic response to position and velocity set point changes as demonstrated by Heinemann & Papiernik [72].

4.3.1 Interpolator

The interpolator calculates a velocity profile $v(t)$, according to defined motion commands: feed rate (f_r), maximum acceleration (a_{max}), maximum jerk (j_{max}), and desired displacement (d). This velocity profile is then used to generate the set of reference positions. To achieve a smooth velocity transition, the velocity profile is divided into seven phases as shown in Figure 4.13.

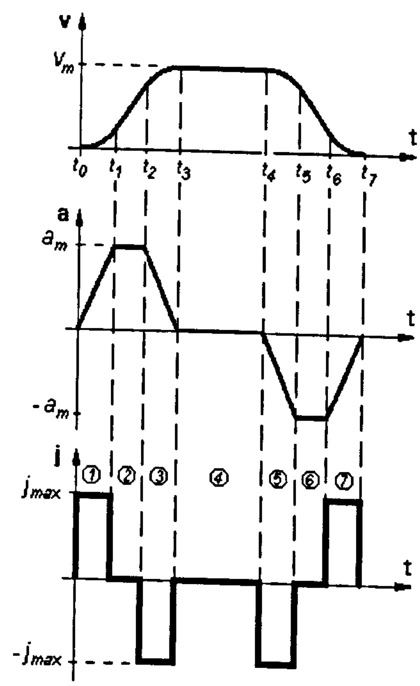


Figure 4.13 Jerk-limited velocity profile generation [73]

The calculation of jerk, velocity and acceleration for each phase was presented by Altintas [73] as follows:

- To ensure proper operation of an axis, the following two conditions must be fulfilled:
  - A minimum distance d_{min} must be traversed in order to attain the programmed feed rate. If $d < d_{min}$ the feed rate must be reduced to its maximum possible (v_m), thus the velocity profile must include at least phases 1, 3, 5 and 7:

$$\frac{f_r}{2} = 0.5j(\frac{t_3}{2})^2 \tag{4.30}$$

$$d_{\min} = f_r t_3 \tag{4.31}$$

Substituting equation (4.35) into equation (4.36) gives:

$$d_{\min} = 2 f_r \sqrt{f_r / j_{\max}} \tag{4.32}$$

$$v_m = \begin{cases} \sqrt[3]{\dfrac{j_{\max} d^2}{4}} & d < d_{\min} \\ f_r & d \geq d_{\min} \end{cases} \tag{4.33}$$

- The maximum velocity and maximum jerk result in an acceleration a_m (see if phases 2 and 6 exist).

$$a_m = \begin{cases} \sqrt{v_m j_{\max}} & a_m \leq a_{\max} \\ a_{\max} & a_m > a_{\max} \end{cases} \tag{4.34}$$

- The duration time of each phase (T_i for $i = 1,2\ldots7$) is given by:

$$T_i = t_i - t_{i-1} \qquad for \quad i = 1,2\ldots7 \tag{4.35}$$

$$T_1 = T_3 = T_5 = T_7 = a_m / j_{\max} \tag{4.36}$$

$$T_2 = T_6 = v_m / a_m - a_m / j_{\max} \tag{4.37}$$

$$T_4 = \frac{d}{v_m} - \sum_{i=1}^{3} T_i \tag{4.38}$$

- The values for jerk, acceleration and velocity profiles for each phase are calculated according to the following equations:

$$j(t) = \begin{cases} j_{\max} & 0 < t \leq t_1 \\ 0 & t_1 < t \leq t_2 \\ -j_{\max} & t_2 < t \leq t_3 \\ 0 & t_3 < t \leq t_4 \\ -j_{\max} & t_4 < t \leq t_5 \\ 0 & t_5 < t \leq t_6 \\ j_{\max} & t_6 < t \leq t_7 \end{cases} \tag{4.39}$$

$$a(t) = \begin{cases} j_{\max} t & 0 < t \leq t_1 \\ a_m & t_1 < t \leq t_2 \\ a_m - j_{\max}(t - t_2) & t_2 < t \leq t_3 \\ 0 & t_3 < t \leq t_4 \\ -j_{\max}(t - t_4) & t_4 < t \leq t_5 \\ -a_m & t_5 < t \leq t_6 \\ -a_m + j_{\max}(t - t_6) & t_6 < t \leq t_7 \end{cases} \tag{4.40}$$

$$v(t)=\begin{cases}0.5j_{\max}t^2 & 0<t\le t_1\\ v(t_1)+a_m(t-t_1) & t_1<t\le t_2\\ v(t_2)+a_m(t-t_2)-0.5j_{\max}(t-t_2)^2 & t_2<t\le t_3\\ v_m & t_3<t\le t_4\\ v_m-0.5j_{\max}(t-t_4)^2 & t_4<t\le t_5\\ v(t_5)-a_m(t-t_5) & t_5<t\le t_6\\ v(t_6)-a_m(t-t_6)+0.5j_{\max}(t-t_6)^2 & t_6<t\le t_7\end{cases} \tag{4.41}$$

The interpolator generates a reference position (d_{ref}) value every t_p seconds by the following relation:

$$d_{ref}(t)=d_{ref}(t-t_p)+v(t)t_p \tag{4.42}$$

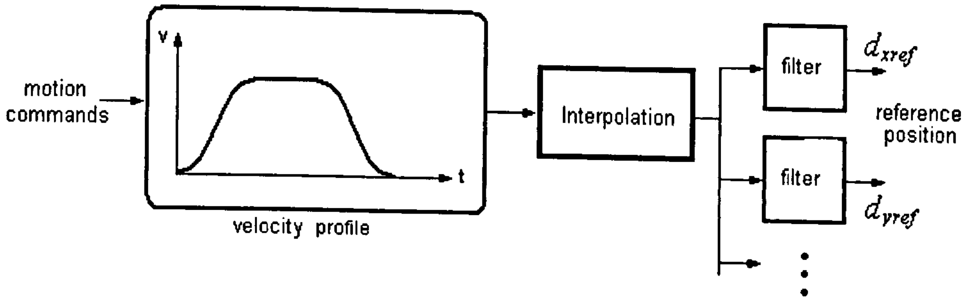


Figure 4.14 Block diagram of the interpolator [74]

If the system has multiple axes, the motions of individual axes are co-ordinated. When the individual axes are not co-ordinated, each of the axes will start moving at the same time, but finish at separate times producing *slew motion* as defined by Hugh [74]. The interpolator overcomes this effect using an interpolation technique. Position values from the interpolator can be filtered in order to obtain a smoother motion profile. The filtered values constitute the reference position signals for the position controllers. A block diagram of the interpolator is illustrated in Figure 4.14.

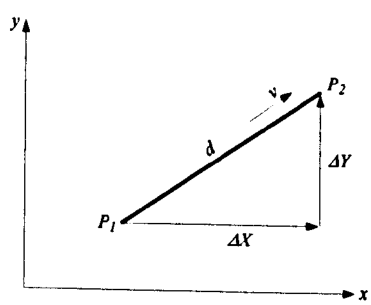


Figure 4.15 Two-axis linear movement [75]

Linear interpolation is performed to keep the tool path velocity constant at the given feed rate along a straight line in a plane of motion. For a xy-plane movement (Figure 4.15), the reference position values for x-axis ($d_{xref}(t)$) and y-axis ($d_{yref}(t)$) are derived using the equations presented by Koren [75]:

$$d_{xref}(t) = d_{ref}(t)(\Delta X / d) \quad (4.43)$$

$$d_{yref}(t) = d_{ref}(t)(\Delta Y / d) \quad (4.44)$$

$$d = \sqrt{\Delta X^2 + \Delta Y^2} \quad (4.45)$$

Where, ΔX is the distance the x-axis is going to move, and ΔY is the distance the y-axis is going to move.

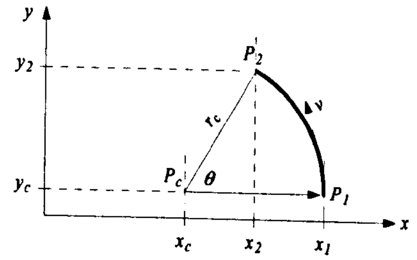


Figure 4.16 Two-axis circular movement [76]

In *circular interpolation,* the velocities of two axes on a plane of motion are varied to perform a semicircle movement at a specified feed rate. As presented by Bullock [76], two commands are needed in a NC program to describe a semicircle: the radius r_c and the end point $P_2(x_2, y_2)$ – the starting point $P_1(x_1, y_1)$ is defined by the coordinates of the actual position of the axes (Figure 4.16). In this case, the velocity profile v is calculated for a displacement d of magnitude:

$$d = r_c\theta \quad (4.46)$$

Where, the angle of displacement θ is defined by

$$\theta = \tan^{-1}\left(\frac{y_2 - y_c}{x_2 - x_c}\right) \quad (4.47)$$

$$x_c = x_1 - r_c \quad (4.48)$$

$$y_c = y_1 \quad (4.49)$$

The interpolator generates the reference position (d_{ref}) according to equation (4.47). Then, the reference position values for the x-axis and y-axis are:

$$d_{xref}(t) = x_1 - r_c \sin(d_{ref}(t) / r_c) \quad (4.50)$$

$$d_{yref}(t) = y_1 + r_c \cos(d_{ref}(t) / r_c) \quad (4.51)$$

4.3.2 Position Controller with Velocity Feed Forward

The position controller evaluates the difference between the reference and actual position value d_{act} (from the rotary or linear encoder) to calculate the position error d_e. A reference velocity value (v_{ref}) is then generated every t_s seconds according to the following relation:

$$v_{ref} = d_e k_v + v_{ff} \tag{4.52}$$

Where,
$$d_e = d_{ref} - d_{act} \tag{4.53}$$

$$v_{ff} = k_{ff} \frac{d}{dt} d_{ref} \tag{4.54}$$

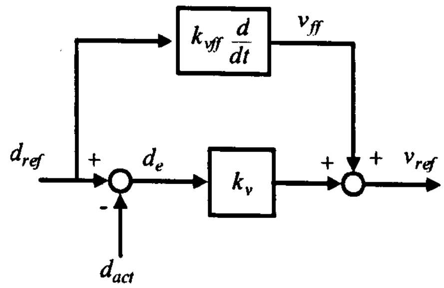


Figure 4.17 Block diagram of the position controller with velocity feed forward [77]

k_v is the gain of the position controller, k_{vff} the feed forward velocity gain and v_{ff} is the velocity feed forward signal. The TLM transform of equations (4.52) to (4.54) gives:

$$v_{ff}(k) = d_{ref}(k) Z_{vff} + E^i_{vff}(k) \tag{4.55}$$

$$E^i_{vff}(k+1) = -d_{ref}(k) Z_{vff} \tag{4.56}$$

$$v_{ref}(k) = k_v d_e(k) + v_{ff}(k) \tag{4.57}$$

$$d_e(k) = d_{ref}(k) - d_{act}(k) \tag{4.58}$$

Where
$$Z_{vff} = k_{vff} / t_s \tag{4.59}$$

Z_{vff} is the characteristic impedance of the stub and $E^i_{vff}(k)$ is the incident pulse. The signal d_{act} is calculated by equation (4.60) if the linear encoder is used as a position feedback system. Equation (4.61) gives the value for d_{act} when the rotary encoder is used instead.

$$d_{act}(k) = d_l(k) \tag{4.60}$$

$$d_{act}(k) = k_b \theta_m(k) \tag{4.61}$$

Where k_b is the force to torque constant of the ball screw.

4.3.3 Velocity Controller with Acceleration Feed Forward

The velocity controller (Figure 4.18) evaluates the difference between the reference and actual velocity value (v_{act}) measured with a rotary encoder attached to the motor. A reference current value (i_{ref}) is generated every t_s seconds by a Proportional-Integral-Differential (PID) strategy. Acceleration feed-forward is used in parallel with the velocity controller in order to minimise the spikes caused by changes in velocity direction. A signal i_{ht} (holding current) is added to the acceleration feed forward to counterbalance the axis load when the axis in the vertical position.

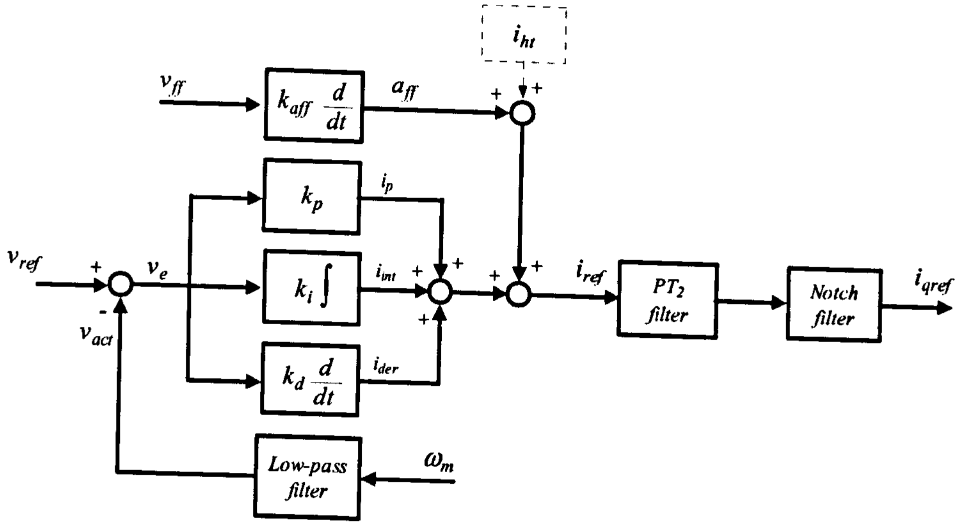


Figure 4.18 Block diagram of the velocity controller with acceleration feed forward [77]

The following relation calculates the reference current signal:

$$i_{ref} = i_p + i_{\text{int}} + i_{der} + a_{ff} + i_{ht} \tag{4.62}$$

Where,

$$i_p = k_p v_e \tag{4.63}$$

$$i_{\text{int}} = k_i \int v_e dt \tag{4.64}$$

$$i_{der} = k_d \frac{dv_e}{dt} \tag{4.65}$$

$$a_{ff} = k_{aff} \frac{dv_{ff}}{dt} \tag{4.66}$$

$$v_e = v_{ref} - v_{act} \tag{4.67}$$

Here k_p, k_i, k_d are the proportional, integral, and derivative gains of the velocity error v_e, respectively. k_{aff} is the acceleration feed forward gain. The TLM transform of equations (4.70 - 4.75) for the cycle time t_s gives:

$$i_{ref}(k) = i_p(k) + i_{\text{int}}(k) + i_{der}(k) + a_{ff}(k) \tag{4.68}$$

$$i_p(k) = k_p v_e(k) \tag{4.69}$$

$$i_{\text{int}}(k) = Z_i v_e(k) + E_i^i(k) \tag{4.70}$$

$$E_i^i(k+1) = i_{\text{int}}(k) \tag{4.71}$$

$$i_{der}(k) = Z_d v_e(k) + E_d^i(k) \tag{4.72}$$

$$E_d^i(k+1) = -Z_d v_e(k) \tag{4.73}$$

$$a_{ffi}(k) = Z_{aff} v_{ff}(k) + E_{aff}^i(k) \tag{4.74}$$

$$E_{aff}^i(k+1) = -Z_{aff} v_{ff}(k) \tag{4.75}$$

$$v_e(k) = v_{ref}(k) - v_{act}(k) \tag{4.76}$$

Where,
$$Z_i = k_i t_s \tag{4.77}$$

$$Z_d = k_d / t_s \tag{4.78}$$

$$Z_{aff} = k_{aff} / t_s \tag{4.79}$$

A filter is generally used in the velocity feedback to damp the fundamental frequency of the control system, when it is higher than 500 Hz. A 1st-order low-pass filter is used when the oscillation frequency is between 500 and 700 Hz. A 2nd-order low-pass filter is used if the oscillation frequency is higher than 700 Hz [77].

When the controlled system is insufficiently damped (e.g. direct motor coupling or roller bearings), it will be impossible to attain a sufficiently short settling time without inducing oscillations in the step response of the velocity controller. The step response will oscillate even with a low proportional factor. A 2nd-order lag element (PT_2) is used to include a delay in the reference current (i_{ref}) to damp the frequency interference oscillations.

A band-rejection filter is included in series with the PT_2 element to damp oscillations that cannot be compensated by the differential factor of the velocity controller, the PT_2 element, or the low-pass filter.

These digital filters are modelled as implementations of the standard difference equation:

$$\begin{aligned} a(1)\mathrm{i}_{\mathrm{qref}}(k) &= b(1)\mathrm{i}_{\mathrm{ref}}(k) + b(2)\mathrm{i}_{\mathrm{ref}}(k-1) + \ldots + b(2N+1)i_{\mathrm{ref}}(k-2N) \\ &\quad - a(2)\mathrm{i}_{\mathrm{qref}}(k-1) - \ldots - a(2N+1)\mathrm{i}_{\mathrm{qref}}(k-2N) \end{aligned} \tag{4.80}$$

Where, b and a represent the numerator and denominator coefficients of the filter. N is the order of the filter [78].

4.3.4 Current Controller and PWM Generation

The control of an AC motor is performed by controlling independently the currents in the d and q axes. The component of the current in the d direction generates a field component on the d-axis, corresponding to the magnetising field in the stator of a DC motor. Similarly, the component of the current on the q direction generates a field component on the q-axis inducing rotor current to produce the motor torque. In motors employing permanent magnet rotors, the q-axis current determines the magnitude and direction of the torque. The control strategy is achieved by regulating the q-axis current to follow the reference current command from the velocity control. The d-axis current is controlled to be zero. Figure 4.19 illustrates the *Space Vector* (SV) PWM approach for implementing this type of current control, as presented by Prokop and Grasblum [79].

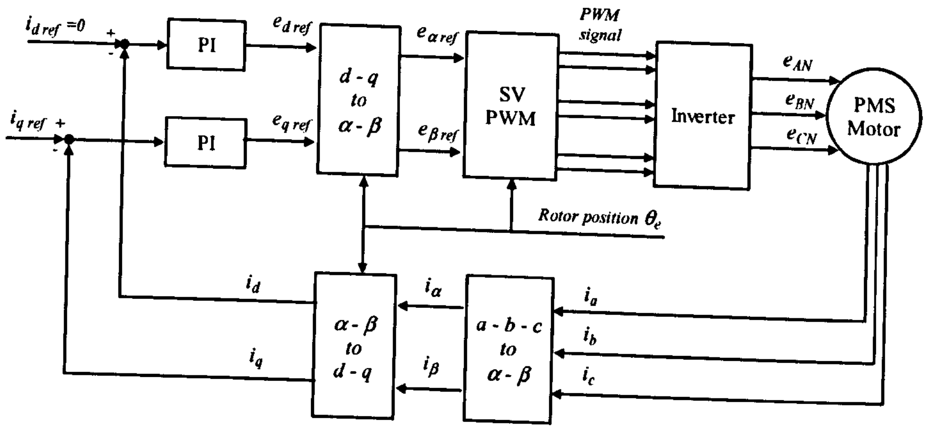


Figure 4.19 Space vector current control structure [79]

The current controllers evaluate the difference between the reference currents i_{dref}, i_{qref} and the actual currents i_d, i_q calculated from the i_a, i_b, i_c currents measured on the motor. A reference voltage values e_{dref}, e_{qref} are generated every t_s seconds by a Proportional-Integral (PI) strategy. Figure 4.20 illustrates the i_q current controller.

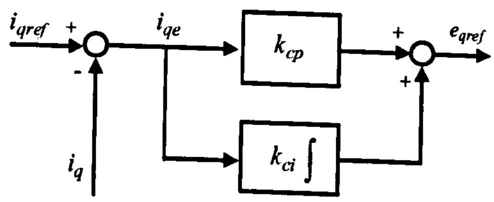


Figure 4.20 i_q current controller

The equations for both controllers are:

$$e_{dref} = k_{cp} i_{de} + k_{ci} \int i_{de} dt \tag{4.81}$$

$$e_{qref} = k_{cp} i_{qe} + k_{ci} \int i_{qe} dt \tag{4.82}$$

Where

$$i_{de} = i_{dref} - i_d \tag{4.83}$$

$$i_{qe} = i_{qref} - i_q \tag{4.84}$$

Here k_{cp} and k_{ci} are the proportional and integral gains of the current errors i_{de} and i_{qe}, respectively. The TLM transform of equations (4.93) to (4.96) for a cycle time t_s gives

$$i_{de}(k) = i_{dref}(k) - i_d(k) \tag{4.85}$$

$$e_{dref}(k) = k_{cp} i_{de}(k) + i_{de}(k) Z_{ci} + E^i_{de}(k) \tag{4.86}$$

$$E^i_{de}(k+1) = i_{de}(k) Z_{ci} + E^i_{de}(k) \tag{4.87}$$

$$i_{qe}(k) = i_{qref}(k) - i_q(k) \tag{4.88}$$

$$e_{qref}(k) = k_{cp} i_{qe}(k) + i_{qe}(k) Z_{ci} + E^i_{qe}(k) \tag{4.89}$$

$$E^i_{qe}(k+1) = i_{qe}(k) Z_{ci} + E^i_{qe}(k) \tag{4.90}$$

Where $$Z_{ci} = k_{ci} t_c \tag{4.91}$$

In the SV-PWM approach, the sinusoidal 3-phase voltages e_a, e_b, e_c (equivalent to the desired stator reference voltages $e_{\alpha ref}$, $e_{\beta ref}$) are approximated by the voltages (e_{AN}, e_{BN}, e_{CN}) resultant from the switching of the power bridge illustrated in Figure 4.21.

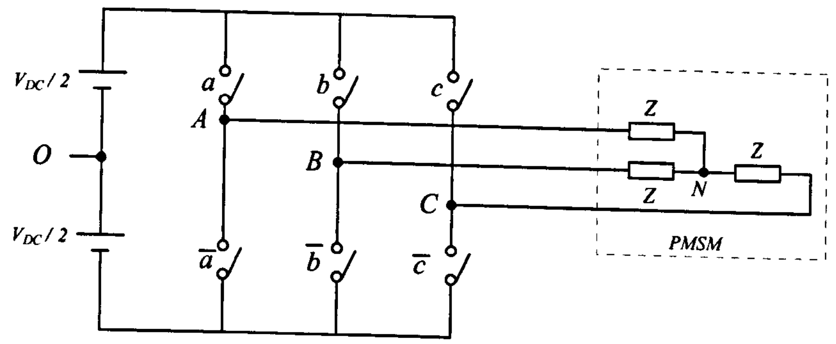


Figure 4.21 Power bridge (Inverter) [79]

The six power transistors in Figure 4.21 are activated by *a, b, c* signals and their respective complement PWM signals. The switching of the switches representing the power transistors leads to the eight possible combinations contained in Table 4.1. The resultant voltages applied to the motor are referenced to the virtual middle point of the link voltage in the inverter (V_{DC}). Equation (4.92) expresses the conversion of each motor phase to their neutral voltages.

a	b	c	e_{AO}	e_{BO}	e_{CO}	e_{AN}	e_{BN}	e_{CN}
0	0	0	$-V_{DC}/2$	$-V_{DC}/2$	$-V_{DC}/2$	0	0	0
0	0	1	$-V_{DC}/2$	$-V_{DC}/2$	$V_{DC}/2$	$-V_{DC}/3$	$-V_{DC}/3$	$2V_{DC}/3$
0	1	0	$-V_{DC}/2$	$V_{DC}/2$	$-V_{DC}/2$	$-V_{DC}/3$	$2V_{DC}/3$	$-V_{DC}/3$
0	1	1	$-V_{DC}/2$	$V_{DC}/2$	$V_{DC}/2$	$-2V_{DC}/3$	$V_{DC}/3$	$V_{DC}/3$
1	0	0	$V_{DC}/2$	$-V_{DC}/2$	$-V_{DC}/2$	$2V_{DC}/3$	$-V_{DC}/3$	$-V_{DC}/3$
1	0	1	$V_{DC}/2$	$-V_{DC}/2$	$V_{DC}/2$	$V_{DC}/3$	$-2V_{DC}/3$	$V_{DC}/3$
1	1	0	$V_{DC}/2$	$V_{DC}/2$	$-V_{DC}/2$	$V_{DC}/3$	$V_{DC}/3$	$-2V_{DC}/3$
1	1	1	$V_{DC}/2$	$V_{DC}/2$	$V_{DC}/2$	0	0	0

Table 4.1 Inverter output voltages

$$\begin{bmatrix} e_{AN} \\ e_{BN} \\ e_{CN} \end{bmatrix} = \frac{1}{3} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix} \begin{bmatrix} e_{AO} \\ e_{BO} \\ e_{CO} \end{bmatrix} \tag{4.92}$$

The projection of the eight possible e_{AN}, e_{BN}, e_{CN} voltages in the α, β coordinate (equation (4.93) system gives the eight combinations that e_α, e_β voltages can take according to the status of the PWM signals (see Table 4.2). Figure 4.22 presents the representation of the eight base vectors in the α, β coordinate system.

$$\begin{bmatrix} e_\alpha \\ e_\beta \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1/\sqrt{3} & 1/\sqrt{3} \end{bmatrix} \begin{bmatrix} e_{AN} \\ e_{BN} \\ e_{CN} \end{bmatrix} \tag{4.93}$$

a	b	c	e_α	e_β	
0	0	0	0	0	$\vec{e}_0$
0	0	1	$-V_{DC}/3$	$-V_{DC}/\sqrt{3}$	$\vec{e}_1$
0	1	0	$-V_{DC}/3$	$V_{DC}/\sqrt{3}$	$\vec{e}_2$
0	1	1	$-2V_{DC}/3$	0	$\vec{e}_3$
1	0	0	$2V_{DC}/3$	0	$\vec{e}_4$
1	0	1	$V_{DC}/3$	$-V_{DC}/\sqrt{3}$	$\vec{e}_5$
1	1	0	$V_{DC}/3$	$V_{DC}/\sqrt{3}$	$\vec{e}_6$
1	1	1	0	0	$\vec{e}_7$

Table 4.2 Possible stator voltages

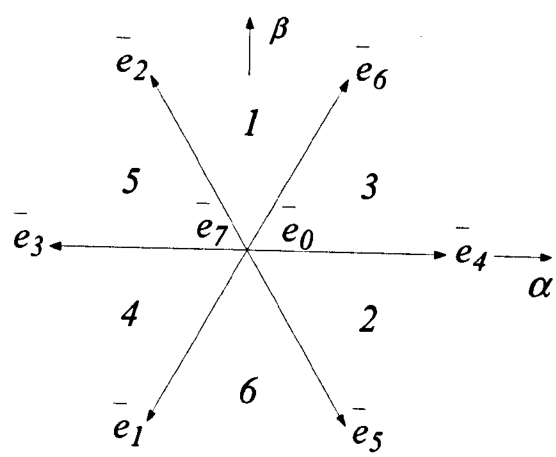


Figure 4.22 Voltage base vectors

The method used to approximate the desired stator reference voltage with the eight possible states of the switches is to combine the adjacent vectors of the reference voltage and to modulate the time of application of each adjacent vector. Figure 4.23 shows an example of a reference vector in the third sector.

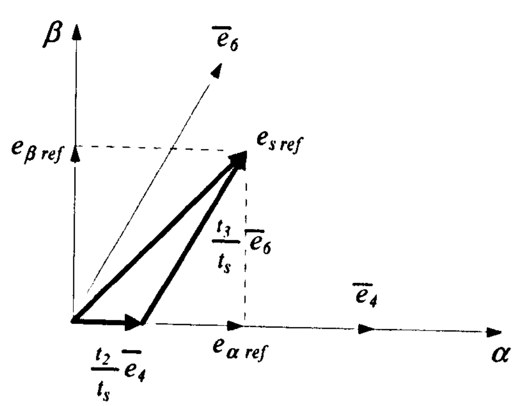


Figure 4.23 Projection of the reference voltage vector

The PWM signal is then conformed for each sector by the application of seven stator voltages as presented in Table 4.3. The application of each stator voltage is denominated switching-state. The corresponding PWM signal for sector three is illustrated in Figure 4.24.

Switching-state sector	1	2	3	4	5	6	7
1	$\vec{e}_0$	$\vec{e}_6$	$\vec{e}_2$	$\vec{e}_7$	$\vec{e}_2$	$\vec{e}_6$	$\vec{e}_0$
2	$\vec{e}_0$	$\vec{e}_5$	$\vec{e}_4$	$\vec{e}_7$	$\vec{e}_4$	$\vec{e}_5$	$\vec{e}_0$
3	$\vec{e}_0$	$\vec{e}_4$	$\vec{e}_6$	$\vec{e}_7$	$\vec{e}_6$	$\vec{e}_4$	$\vec{e}_0$
4	$\vec{e}_0$	$\vec{e}_3$	$\vec{e}_1$	$\vec{e}_7$	$\vec{e}_1$	$\vec{e}_3$	$\vec{e}_0$
5	$\vec{e}_0$	$\vec{e}_2$	$\vec{e}_3$	$\vec{e}_7$	$\vec{e}_3$	$\vec{e}_2$	$\vec{e}_0$
6	$\vec{e}_0$	$\vec{e}_1$	$\vec{e}_5$	$\vec{e}_7$	$\vec{e}_5$	$\vec{e}_1$	$\vec{e}_0$

Table 4.3 Switching-state stator voltages

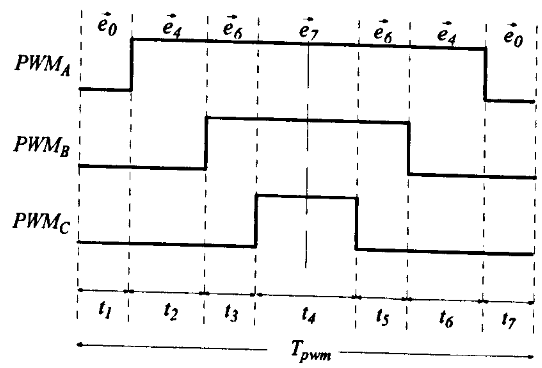


Figure 4.24 PWM signal for a reference vector in sector 3

The durations the switching-state vectors are specified by:

$$2f_s\left(t_2\vec{e}_4 + t_3\vec{e}_6\right) = \vec{e}_{sref} \tag{4.94}$$

$$t_1 = \frac{1}{2f_s} - t_2 - t_3 \tag{4.95}$$

$$t_4 = 2t_1 \tag{4.96}$$

$$t_5 = t_3 \tag{4.97}$$

$$t_6 = t_2 \tag{4.98}$$

$$t_7 = t_1 \tag{4.99}$$

Where, f_s is the switching frequency. The application of equation (4.100) in each sector leads to three possible values (t_{xyz}) for t_2 and t_3:

$$t_{xyz} = M_{xyz} e_{\alpha\beta ref} \tag{4.100}$$

Where
$$t_{xyz} = [t_x \quad t_y \quad t_z] \tag{4.101}$$

$$M_{xyz} = \frac{T_{pwm}}{V_{DC}} \begin{bmatrix} 0 & \sqrt{3} \\ 1.5 & \sqrt{3}/2 \\ -1.5 & \sqrt{3}/2 \end{bmatrix} \tag{4.102}$$

T_{pwm} represent the period of the PWM signal. The values t_2 and t_3 for each sector are presented in Table 4.4.

sector	1	2	3	4	5	6
t_2	t_y	$-t_x$	$-t_z$	t_z	t_x	$-t_y$
t_3	t_z	t_y	t_x	$-t_x$	$-t_y$	$-t_z$

Table 4.4 Duration times of the switching-states

The sector in which the reference vector is found is calculated by the following procedure:

- Translation of the $e_{\alpha ref}$, $e_{\beta ref}$ voltages to the abc coordinate system

$$e_{abc} = M_{abc} e_{\alpha\beta ref} \tag{4.103}$$

Where $e_{abc} = \begin{bmatrix} e_a \\ e_b \\ e_c \end{bmatrix}$ $\quad e_{\alpha\beta er} = \begin{bmatrix} e_{\alpha ref} \\ e_{\beta ref} \end{bmatrix}$ $\quad M_{abc} = \begin{bmatrix} 0 & 1 \\ \sqrt{3}/2 & -0.5 \\ -\sqrt{3}/2 & -0.5 \end{bmatrix}$ (4.104)

- Calculation of the sector

$$\begin{array}{ll} \text{if } e_a > 0 & p_{abc}(1) = 1; \\ \quad \text{else} & p_{abc}(1) = 0; \\ \text{if } e_b > 0 & p_{abc}(2) = 2; \\ \quad \text{else} & p_{abc}(2) = 0; \\ \text{if } e_c > 0 & p_{abc}(3) = 4; \\ \quad \text{else} & p_{abc}(3) = 0; \end{array}$$

The sector is the resultant of the sum of the components of the p_{abc} array:

$$sector = p_{abc}(1) + p_{abc}(2) + p_{abc}(3) \tag{4.105}$$

The model of the PWM generator (Figure 4.25) can be summarised in the following actions:

- Project the reference voltages e_{dref}, e_{qref} in the α, β coordinate system ($e_{\alpha ref}$, $e_{\beta ref}$ are calculated according to equation (4.9))
- Calculate the sector in which the reference vector is found (equation (4.105)
- Select the base vectors for the defined sector according to table 4.3.
- Calculate the times t_2 and t_3 (equation (4.102) and table 4.4)
- Calculate times t_1, t_4, t_5, t_6 and t_7 (equations (4.95 – 4.99)

The period of the PWM signal (T_{pwm}) is divided into R_{pwm} slots of duration t_{pwm} seconds in order to implement this procedure on a microprocessor. In this regard, the times t_1 to t_7 are specified in terms of equivalent number of slots. The factor R_{pwm} is known as the resolution of the PWM signal.

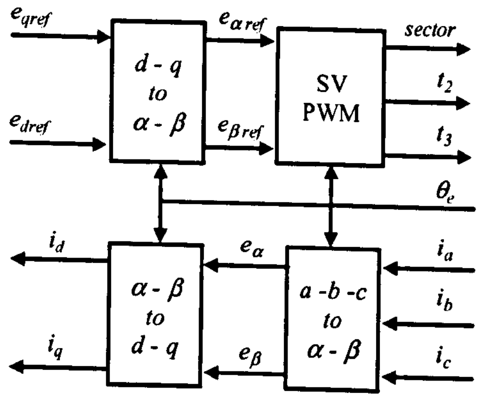


Figure 4.25 PWM generator model

The model for the inverter (Figure 4.26) is described by the following actions:

- Select the series of stator voltages (e_α, e_β) to be applied from Table 4.3.
- Apply the stator voltage *i* during the time t_i (i=1,2,3...7)
- Calculate the e_{AN}, e_{BN}, e_{CN} voltages equivalent to the stator voltage *i* using the inverse Park transformation (equation (4.8)) and inverse Clark transformation (equation (4.7))

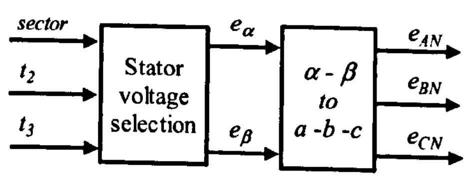


Figure 4.26 Inverter model

4.4 Dynamic Models for Mechanical Transmission Elements

The equations that govern the dynamic behaviour of the mechanical transmission elements are introduced in this section. For modelling purposes, the coupling, bearings, bearing housings, the nut and table-slide are considered lumped parameter elements. The screw shaft is treated as a distributed parameter element.

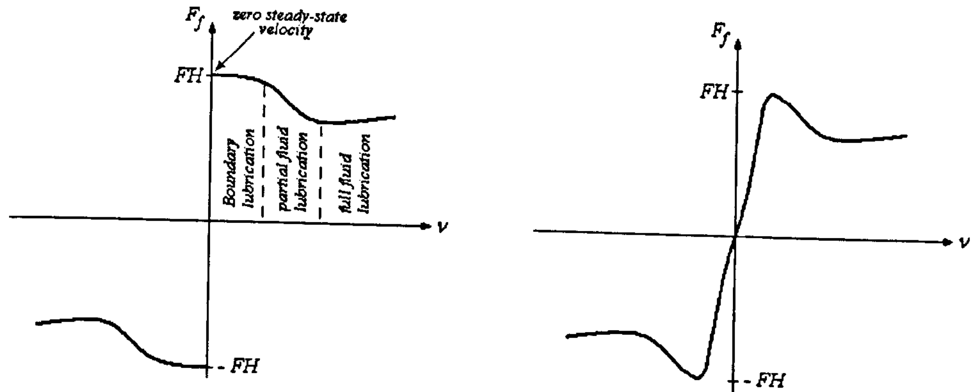
4.4.1 Non-linearities (Backlash, Friction)

Friction and backlash play an important role for applications involving high precision positioning and tracking applications. Understanding of these dynamics is crucial in the modelling exercise because they can deteriorate the performance of positioning systems (like CNC axis feed drives) when moving slowly or at velocity reversals as underlined by Park et al. [80].

Friction modelling has been the object of an ongoing research process and a number of publications can be found in the literature. In contrast, backlash has been mostly referred as a hysteresis loop. This section presents the description of the friction and backlash approaches selected for this study.

4.4.1.1 Friction Model

As stated by Armstrong-Helouvry et al [81], friction models are generally conformed by the sum of Coulomb and viscous friction and four additional components that shape the behaviour of stick-slip motion in machines: Stribeck friction, rising static friction, friction memory and pre-sliding displacement (see Figure 4.27).



a) Stribeck curve

b) Friction curve including static friction

Figure 4.27 Stick-slip friction laws [82]

According to Armstrong-Helouvry [82], four regimes will be observed in oil or grease lubricated contacts as shown in Figure 4.27a:

- Non-sliding: Motion exists as the interface bonding sites deform elastically.
- Boundary lubrication: Sliding occurs with solid-to-solid contact because of not adequate fluid lubrication into the junction.
- Partial fluid lubrication: There is some fluid into the junction but not enough to fully separate the surfaces.
- Full fluid lubrication: The surfaces are fully separated by a fluid film.

The *static friction* is the break away force and the magnitude of the Stribeck friction evaluated with zero velocity (Zero steady state velocity). In lubricated metal-to-metal contacts, the static friction rises from a lower kinetic value to the higher during the time required to expel the fluid lubricant film from the contact interface (Dupont & Dunlap [83]). Then, the Stribeck curve takes the shape illustrated in Figure 4.27(b).

Friction Memory represents the state of adjusting the new sliding conditions in the interface to new values of the frictional force. When velocity changes, the friction does not change instantly, but adjusts to its new value only after some time. This effect is modelled by a simple lag (time delay) model as stated by Dupont et al. [84].

Pre-sliding displacement is a consequence of elastic deformation of the surface asperities where contact and sliding occur (Haessing & Friedland [85]).

Karnopp [86] presented a stick-slip friction law that simplifies the Stribeck curve (including static friction and pre-sliding displacement), to stick-slip behaviour with constant causality as shown in Figure 4.28. This model represents zero velocity sticking without equation reformulation or the introduction of numerical stiffness problems; hence, it has been selected to model friction on the different components of the studied systems.

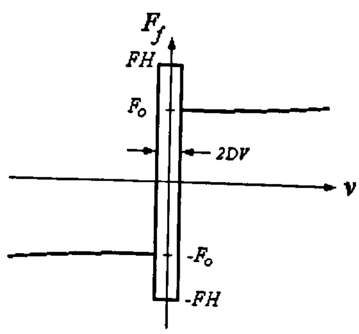


Figure 4.28 Karnopp model of stick–slip friction [86]

In this approach the frictional force, F_f, is always a function of velocity, v. A region of small velocity is defined as $-DV \leq v \leq DV$. Inside this region, v is considered zero. This region is necessary for digital computation time since an exact value of zero will not be computed. The magnitude of the static friction is represented by FH, and F_o is the Coulomb friction. Figure 4.30 shows the block diagram of the algorithm for the computation of the frictional force action on the system illustrated in Figure 4.29. In Figure 4.29, a force F is applied on a body with mass m inducing the momentum P, the velocity v, and the friction force F_{gw} on the system. The coefficient of friction is represented by μ.

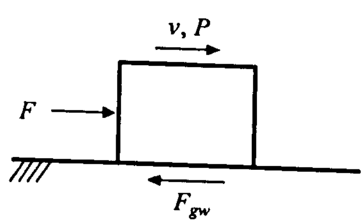


Figure 4.29 Linear movement of a rigid body

The following steps can resume the algorithm for the calculation of the frictional force:

Step 1: Calculation of the impulse

$$P = \int (F - F_{gw})\, dt \tag{4.106}$$

Step 2: Calculation of the velocity

$$v = \begin{cases} 0 & \text{for} \quad |P| < DP \\ P/m & \quad\quad |P| \geq DP \end{cases} \tag{4.107}$$

Where

$$DP = \mu DV \tag{4.108}$$

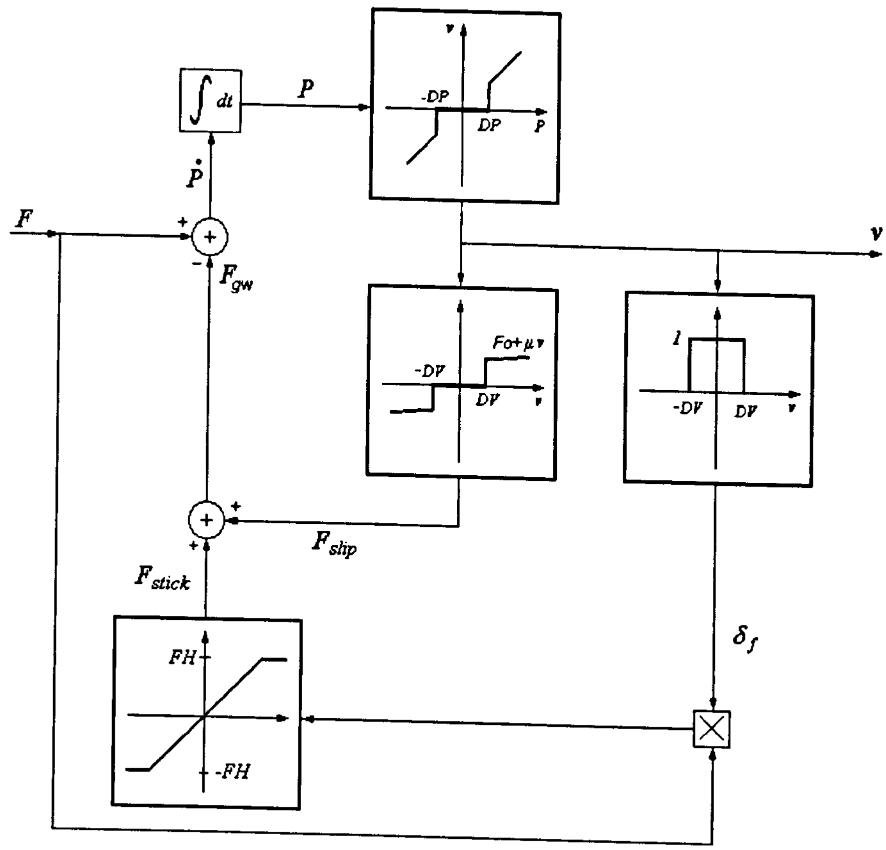


Figure 4.30 Block diagram for the calculation of the Stick-slip friction [86]

Step 3: Calculation of the Slip Force (F_{slip}) and stick force factor (δ_f)

$$F_{slip} = \begin{cases} F_o + \mu v & v > DV \\ 0 \quad for & |v| \le DV \\ -F_o + \mu v & v < -DV \end{cases} \tag{4.109}$$

$$\delta_f = \begin{cases} 1 \\ 0 \end{cases} for \quad \begin{matrix} |v| \le DV \\ |v| > DV \end{matrix} \tag{4.110}$$

Where
$$F_o = 9.81 \mu m \tag{4.111}$$

Step 4: Calculation of the Stick Force (F_{stick})

$$F_{stick} = \begin{cases} 0 \\ sat(F) \end{cases} for \quad \begin{matrix} \delta_f = 0 \\ \delta_f = 1 \end{matrix} \tag{4.112}$$

Where
$$sat(F) = \begin{cases} FH & F > FH \\ F \quad for & |F| \le FH \\ -FH & F < -FH \end{cases} \tag{4.113}$$

Step 5: Calculation of the new frictional force value

$$F_{gw} = F_{slip} + F_{stick} \tag{4.114}$$

F_{slip} and F_{stick} are mutually exclusive, then steps three to five can be combined in one to define the friction force as follows:

$$F_{gw} = \begin{cases} F_{slip} & \\ sat(F) & \end{cases} for \quad \begin{matrix} |v| \geq DV \\ |v| < DV \end{matrix} \tag{4.115}$$

A comparison between the equations of steps one and two and the equation of the TLM model of the system (in Figure 4.29) can be used to specify which changes could be done to the algorithm in order to use it with TLM models. According to Newton's second law of motion:

$$F - F_{gw} = m\frac{dv}{dt} \tag{4.116}$$

Applying the discrete transform to equation (4.116) for a time step t_s gives:

$$F(k) - F_{gw}(k) = v(k)Z_o + E^i(k) \tag{4.117}$$

$$Z_o = m / t_s \tag{4.118}$$

Where Z_o is the characteristic impedance and $E^i(k)$ the incident pulse in the simulation at the simulation step k. Rearranging equation (4.117) gives:

$$v(k) = \frac{F(k) - F_{gw}(k) - E^i(k)}{Z_o} \tag{4.119}$$

By comparing equation (4.119) with equation (4.107) when $|P| > DP$ it can be deduced that the impulse may be expressed in terms of TLM variables, thus:

$$P(k) = F(k) - F_{gw}(k) - E^i(k) \tag{4.120}$$

$$v(k) = P(k) / Z_o \tag{4.121}$$

As a TLM model is considered an equivalent electric representation of a system, forces are represented by voltage sources and velocities by electric currents. Hence, the impulse is characterised by the sum of voltage sources and the mass is represented by the impedance of the electric circuit. Steps for the calculation of the friction force can be then adjusted and expressed as:

Step 1:
$$P(k) = \sum voltages \tag{4.122}$$

Step 2:
$$v(k) = \begin{cases} 0 & \\ P(k)/Z_o & \end{cases} for \quad \begin{matrix} |P(k)| < F_o \\ |P(k)| \geq F_o \end{matrix} \tag{4.123}$$

Step 3:
$$F_{gw}(k+1) = \begin{cases} F_o + \mu v(k) & v(k) > DV \\ sat(F(k)) \quad for & |v(k)| \leq DV \\ -F_o + \mu v(k) & v(k) < -DV \end{cases} \tag{4.124}$$

Where
$$sat(F(k)) = \begin{cases} FH & F(k) > FH \\ F(k) \ for & |F(k)| \leq FH \\ -FH & F(k) < -FH \end{cases} \tag{4.125}$$

Equation (4.124) implies a *friction memory* (equivalent to the simulation time step) in the friction model. This feature can be exploited further to include adequate values of friction memory. This TLM version of Karnopp's friction model (Figure 4.28) will be represented, for the purposes of this study, by a dependant source of voltage as shown in Figure 4.31.

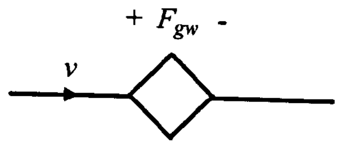


Figure 4.31 TLM representation of frictional force

4.4.1.2 Backlash Model

Backlash is a play between nut and screw shaft that has the effect of temporarily uncoupling and re-coupling them when changing velocity and direction (Oakley [87]). It increases with wear and affects the accuracy of the feed drive (bi-directional repeatability, positioning errors, straightness and others).

Approaches for the modelling of backlash include the formulation of describing functions and the specification of equations related to the series of events that conform this non-linearity. Equation (4.126) represents the describing function for backlash in gearing derived by Stockdale [88] and the meaning of the variables is depicted in Figure 4.32.

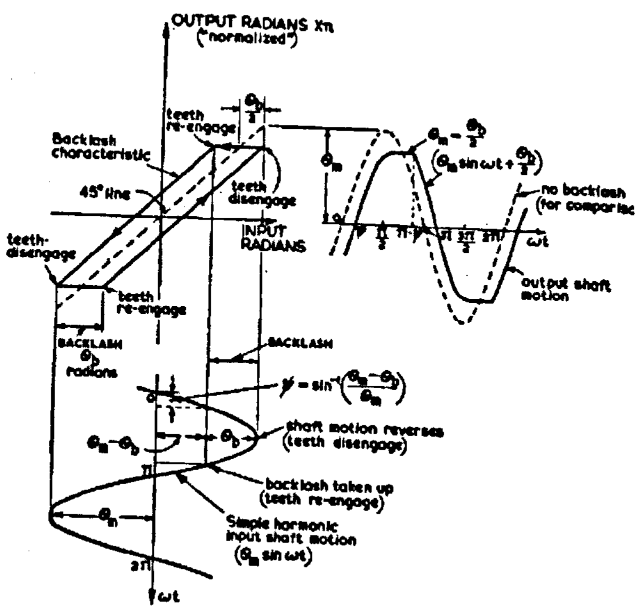


Figure 4.32 Input signal, hysteric backlash and output waveform [89]

As stated by Robertson [89], the disadvantages of describing functions include:

- The prediction of a limit cycle even though it does not exist;
- The determination of values for amplitude and frequency that could be different from the true values;

- A limit cycle may not be forecasted even though it actually exists.

$$N(jw) = \frac{1}{\pi}\left[\left\{\frac{\pi}{2} + \Psi + \left(1 - \frac{\theta_b}{\theta_m}\right)\cos\Psi\right\} + j\frac{\theta_b}{\theta_m}\left\{\frac{\theta_b}{\theta_m} - 2\right\}\right] \tag{4.126}$$

Kao et al. [90] presented the mathematical model of the hysteric backlash representation according to the possible four events presented in it (Figures 4.33 and 4.34).

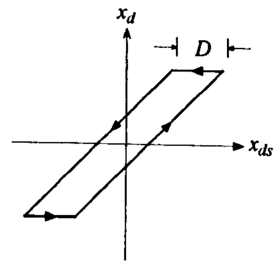


Figure 4.33 Input–output characteristic of backlash [90]

If d_d is the position of the nut; d_a the position of a point a on the shaft towards the nut; D the backlash distance; Δd_a the incremental position feedback at the i^{th} time step; d_a(i-1) the relative distance at the $(i\text{-}1)^{th}$ time step. The possible situations can be expressed as:

- Figure 4.31(a): If $0 < d_a(i) < D$ and $\Delta d_a > 0$ then point d is stationary
- Figure 4.31 (b): If $d_a(i) > D$ then $d_d(i) = d_d(i\text{-}1) + d_a(i) - D$ and $d_a(i) = D$
- Figure 4.31 (c): If $0 < d_a(i) < D$ and $\Delta d_a < 0$ then point d is stationary
- Figure 4.31 (d): If $d_a(i) < D$ then $d_d(i) = d_d(i\text{-}1) + d_a(i)$ and $d_a(i) = 0$

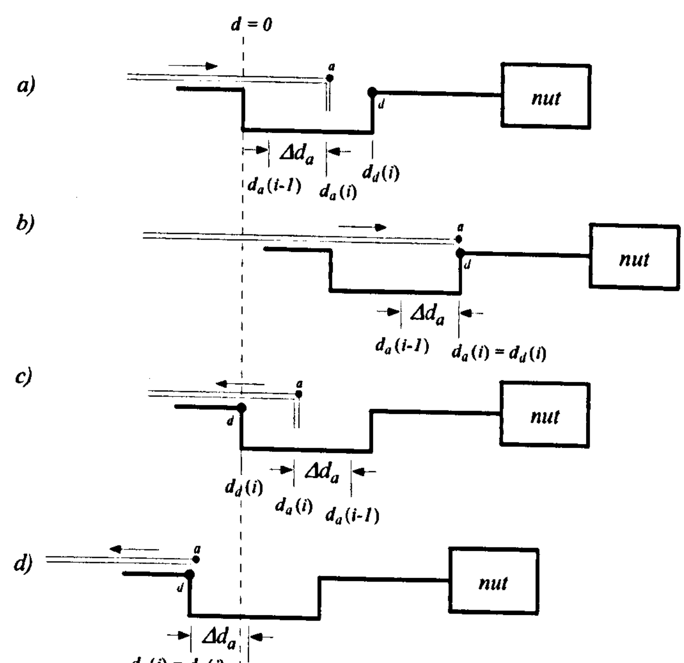


Figure 4.34 Four possible situations for the backlash model [90]

As can be seen, this model of backlash does not include differential or integral terms.

PAGE
MISSING
IN
ORIGINAL

F_R - resulting bearing load [N]

F_r, F_a - radial and axial loads acting on the bearing (Figure 4.38),

X_b, Y_b - load factors

b_b – bearing coefficient of friction

Equation (4.128) can be approximated to the expression in equation (4.131) when the operational velocity range of the bearing is lower than 2000 RPM [93].

$$T_{f0} = b_b \omega \qquad \text{[N-m]} \tag{4.131}$$

The stiffness of the bearing mounting (fixed case) is represented by a spring contact comprised of the bearing and the bearing housing subject to the axial force F_a acting on the bearing as illustrated in Figure 4.39. m_{rb} represents the sum of masses of the bearing inner ring, the tightening nut and the shaft section not subject to axial tension.

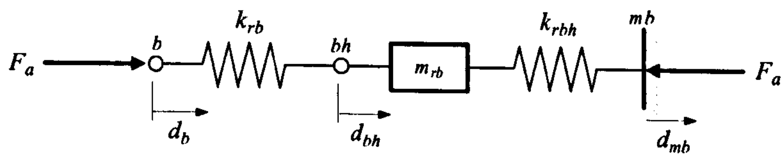


Figure 4.39 Stiffness representation for a bearing mounting

The displacements of the points b, bh and mb are represented by the variables d_b, d_{bh} and d_{mb} respectively. mb is the contact point of the bearing housing with the machine bed. The dynamic equations for this arrangement are:

$$F_a = k_{rb}(d_b - d_{bh}) \tag{4.132}$$

$$F_a - k_{rbh}(d_{bh} - d_{mb}) = m_{rb}\frac{d}{dt}v_{bh} \tag{4.133}$$

Where k_{rb} and k_{rbh} represent the bearing and bearing housing rigidity. Assuming the rigidity of the machine bed to be infinite (d_{mb}=0) gives:

$$F_a - k_{rbh}d_{bh} = m_{rb}\frac{d}{dt}v_{bh} \tag{4.134}$$

The velocities of the bearing mounting ends are:

$$v_b = \frac{d}{dt}d_b \tag{4.135}$$

$$v_{bh} = \frac{d}{dt}d_{bh} \tag{4.136}$$

Substituting equations (4.135) and (4.136) in equation (4.134) gives:

$$F_a = k_{rb}\int(v_b - v_{bh})dt = k_{rb}\int v_{rb}dt \tag{4.137}$$

Substituting equation (4.136) in equation (4.133) gives:

$$F_a = k_{rbh} \int v_{bh} dt + m_{rb} \frac{d}{dt} v_{bh} \tag{4.138}$$

Applying the discrete transform to equations (4.137) and (4.138) for a time step t_s gives:

$$F_a(k) = v_{rb}(k) Z_{rb} + E^i_{rb}(k) \tag{4.139}$$

$$F_a(k) = v_{bh}(k) Z_{rbh} + E^i_{rbh}(k) + v_{bh}(k) Z_{mrb} + E^i_{mrb}(k) \tag{4.140}$$

Where,

$$Z_{rb} = t_s /(1/k_{rb}) \tag{4.141}$$

$$E^i_{rb}(k+1) = E^i_{rb}(k) + v_{bhb}(k) \tag{4.142}$$

$$Z_{rbh} = t_s /(1/k_{rbh}) \tag{4.143}$$

$$E^i_{rbh}(k+1) = E^i_{rbh}(k) + v_{bh}(k) Z_{rbh} \tag{4.144}$$

$$Z_{mrb} = m_{rb} / t_s \tag{4.145}$$

$$E^i_{mrb}(k+1) = -v_{bh}(k) Z_{mrb} \tag{4.146}$$

Figure 4.40 shows the TLM model for the bearing mounting stiffness.

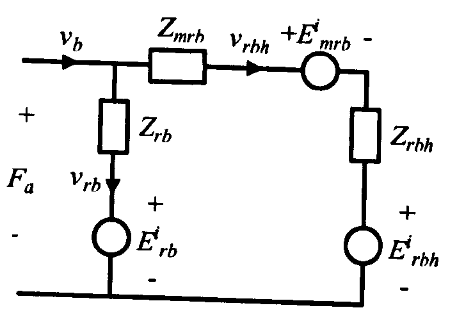


Figure 4.40 TLM model for the bearing mounting stiffness

4.4.4 Guideways and Slides

Linear motion guideways can be classified as: Sliding contact guideways (slides) and rolling contact guideways. In the sliding contact guideways, the relative motion between the elements is sliding, thus giving rise to sliding friction. Rolling guideways consist of a rail with ground ball tracks as well as a block. Continuously rotating balls ensure low friction and connect the block with the rail. The balls are kept in the slide way of the block by a cleat so that the installation of the components is possible without additional auxiliaries. The block is protected against the penetration of dust on every side by scrapers as shown in Figure 4.41 [94].

Advantages of using rolling contact guideways instead of sliding contact guideways include [95]:

- The efficiency will be more than 95% due to the low coefficient of friction; hence, a much more compact motor will be sufficient to run a roller guideways system.

- Absence of 'stick slip' phenomena permits uniform motion at low speeds, which results in very high positioning accuracy.
- In sliding guideways, the quantity of oil film between slides is subjected to variation with speed of travel of slides and cutting forces. Rolling guideways have a preloaded metal to metal contact with negligible oil film and are therefore subjected to negligible change in the position of slides between static and dynamic conditions.
- Efficient lubrication, heat dissipation and other maintenance need not be given much importance in rolling guideways. This will save a lot of maintenance cost and time.

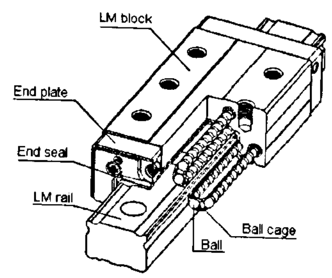


Figure 4.41 Rolling contact guideway [95]

Frictional resistance in rolling guideways varies with the magnitude of the preload, the viscosity resistance of the lubricant used, the load exerted on the system, and other factors. An example of a graph of the friction coefficient as a function of the imposed load ratio is shown in Figure 4.42 [96]. The load ratio (M_{gw} / C_{gw}) in Figure 4.42 is defined as the ratio between the imposed load (M_{gw}) and the basic dynamic load rating of the guideway (C_{gw}), where:

$$M_{gw} = mg \tag{4.147}$$

Parameters m and g in equation (4.161) represent the mass of the load and the gravitational acceleration respectively.

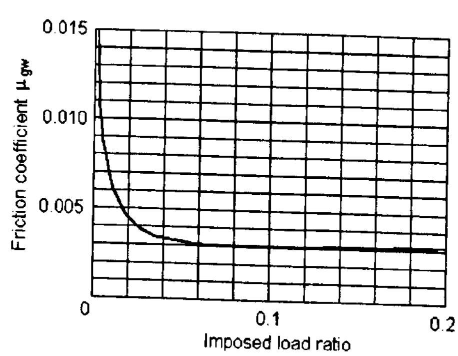


Figure 4.42 Relationship between imposed load ratio and friction coefficient [96]

When radial (F_{rad}) and lateral (F_{lat}) loads are exerted on the block simultaneously (Figure 4.43), an equivalent load (F_E) is calculated using the following equation:

$$F_E = X_{rad}F_{rad} + Y_{lat}F_{lat} \tag{4.148}$$

Where X_{rad} and Y_{lat} represent the equivalent factors according to the configuration of the guideway.

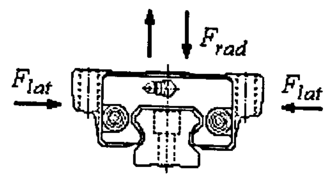


Figure 4.43 Forces acting on the carriage block of a guideway [97]

Accordingly, the force of friction acting on the guideway (F_{gw}) is defined as [97]

$$F_{gw} = F_{gw0} + b_{gw}F_E + b_{gw}v_l \tag{4.149}$$

Where b_{gw} is the friction coefficient, F_{gw0} is the frictional force of the guideways under no-load and v_l is the velocity of the load. Doing:

$$F_0 = F_{gw0} + b_{gw}F_E \tag{4.150}$$

$$F_1 = b_{gw}v_l \tag{4.151}$$

Equation (4.149) becomes

$$F_{gw} = F_0 + F_1 \tag{4.152}$$

4.4.5 Coupling

The coupling consists of two hubs and one flexible intermediate ring in the form of a star (Figure 4.44a). The ring is pressed under a slight pretension into the claws to achieve backlash-free torque transmission, as shown in Figure 4.44b [98]. The coupling is used to accomplish the following functions:

- To transmit the torque induced in the motor to the screw shaft.
- To compensate radial, axial and angular shaft misalignments.
- To isolate the motor against axial vibrations experienced by the ball screw arrangement.

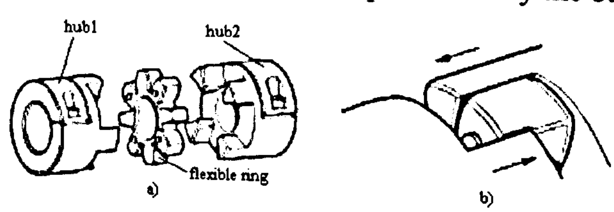


Figure 4.44 Coupling [98]

Given the mass moment of inertia of a rotor hub (J_c), and the angular velocity of the hub (ω), the energy conversion process in the hub is governed by the equation

$$T = J_c(d\omega / dt) \tag{4.153}$$

The coupling torsional stiffness is considered a linear spring subject to a torque T, as shown in Figure 4.45.

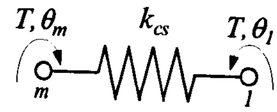


Figure 4.45 Representation of the coupling stiffness

$$T = k_{cs}(\theta_m - \theta_1) = k_{cs}\theta_c \tag{4.154}$$

Where: k_{cs} – Torsional stiffness of the coupling [N-m/rad]

θ_m – Displacement of the end *m* of the coupling [rad] (in contact with the motor)

θ_l – Displacement of the end *1* of the coupling [rad] (in contact with the screw)

θ_c – Relative displacement of coupling ends [rad]

The velocities of the coupling ends will be:

$$\omega_m = d\theta_m / dt \tag{4.155}$$

$$\omega_1 = d\theta_1 / dt \tag{4.156}$$

Rearranging equations (4.155) and (4.156) and replacing them in equation (4.154) gives:

$$T = k_{cs}\int(\omega_m - \omega_1)dt = k_{cs}\int\omega_c dt \tag{4.157}$$

Where ω_m represents the angular velocity of the end *m*, ω_l is the angular velocity of the end *1* and ω_c the angular velocity of the coupling. TLM stubs model the hub inertia and coupling stiffness, as shown in Figure 4.46.

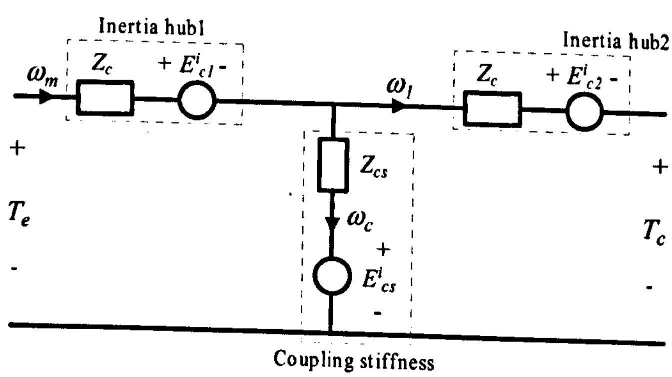


Figure 4.46 TLM model for coupling

The TLM transform of equation (4.153) for the considered propagation time t_s is:

$$T(k) = Z_c\omega(k) + E_c^i(k) \tag{4.158}$$

$$E_c^i(k+1) = -Z_c\omega_c(k) \tag{4.159}$$

$$Z_c = J_c / t_s \tag{4.160}$$

Applying the discrete transform to equation (4.157) for the time step t_s gives:

$$T(k) = \omega_c(k)Z_{cs} + E_{cs}^i(k) \tag{4.161}$$

$$E_{cs}^i(k+1) = T(k) \tag{4.162}$$

$$Z_{cs} = t_s /(1/k_{cs}) \tag{4.163}$$

The discrete transform for equation (4.155) gives:

$$\theta_m(k) = t_s\omega_m(k) + E^i_{thm}(k) \tag{4.164}$$

$$E^i_{thm}(k+1) = \theta_m(k) \tag{4.165}$$

4.4.6 Worktable

The movement of the worktable on the guideways is represented by the system illustrated in Figure 4.47

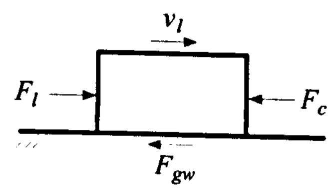


Figure 4.47 Linear movement of the worktable

In Figure 4.47, a force F_l and the horizontal component of the cutting force F_c are applied on a rigid body with mass m inducing the velocity v_l, the displacement d_l and the frictional force F_{gw} on the system. The dynamics of the worktable are given by:

$$F_l(t) - F_c(t) - F_{gw}(t) = m\frac{d}{dt}v_l(t) \tag{4.166}$$

$$d_l(t) = \int v_l(t)dt \tag{4.167}$$

The discrete transform of equations (4.166) and (4.167) for a time step t_s gives:

$$F_l(k) - F_c(k) - F_{gw}(k) = Z_l v_l(k) + E^i_l(k) \tag{4.168}$$

$$E^i_l(k+1) = -Z_l v_l(k) \tag{4.169}$$

$$d_l(k) = t_s v_l(k) + E^i_{dl}(k) \tag{4.170}$$

$$E^i_{dl}(k+1) = d_l(k) \tag{4.171}$$

$$Z_l = m/t_s \tag{4.172}$$

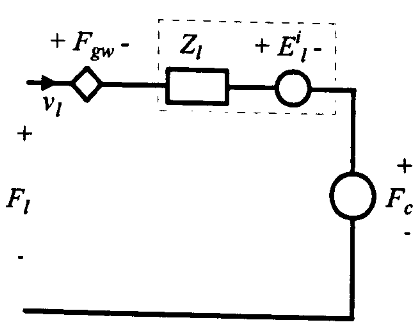


Figure 4.48 TLM model of the worktable movement

4.4.7 Nut

The nut is considered a linear spring subject to an axial force F_a, as shown in Figure 4.49.

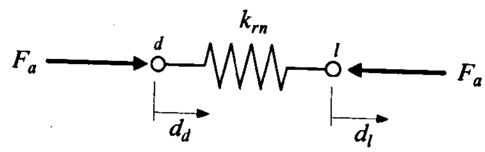


Figure 4.49 Representation of the nut rigidity

$$F_a(t) = k_{rn}(d_d(t) - d_l(t)) = k_{rn}d_n(t) \qquad (4.173)$$

$$k_{rn} = 0.8k_n(F_{ao}/(0.1C_n))^{1/3} \qquad (4.174)$$

Where k_n and k_{rn} represent the nut rigidity and the resulting rigidity of the preloaded nut with mounting bracket respectively. d_d and d_l are the displacement of points d and l. F_{ao} is the preloading force applied to the nut and C_n the dynamic load rating of the nut. The velocities of the nut ends (v_d and v_l) will be:

$$v_d(t) = \frac{d}{dt}d_d(t) \qquad\qquad v_l(t) = \frac{d}{dt}d_l(t) \qquad (4.175)$$

Rearranging equations (4.175) and replacing into equation (4.173) gives:

$$F_a = k_n\int(v_d - v_l)dt = k_n\int v_n dt \qquad (4.176)$$

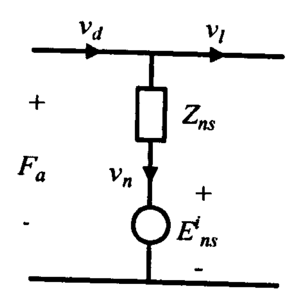


Figure 4.50 TLM model for the nut stiffness

The nut is the coupling element between the ballscrew and the table, then a stub, as shown in Figure 4.50, models its stiffness. Applying the discrete transform to equation (4.176) for a time step t_s gives:

$$Z_{ns} = (t_s)/(1/k_n) \qquad (4.177)$$

$$F_a(k) = v_d(k)Z_{ns} + E^i_{ns}(k) \qquad (4.178)$$

$$E^i_{ns}(k+1) = F_a(k) \qquad (4.179)$$

The nut is pre-loaded to make its axial clearance (backlash) zero and reduce the displacement with respect to the axial load [99]. This pre-loading induces the resistance torque T_p on the screw shaft, which can be expressed by:

$$T_p = T_r n \tag{4.180}$$

$$T_r = 0.005(\tan\beta_{ss})^{-0.5} F_{ao} l_d /(2\pi) \tag{4.181}$$

$$\tan(\beta_{ss}) = l_d /(\pi B_{CD}) \tag{4.182}$$

Where T_r is the reference torque, n the reduction ratio of the ballscrew, β_{ss} the screw shaft lead angle, F_{ao} the nut pre-loading load, l_d the lead (pitch) of the ballscrew and B_{CD} the nut ball circle diameter.

4.4.8 Interrelation Between Nut and Screw Shaft

The rotational torque (T_d) required to counter balance the external load (T_a and the pre-loading of the nut is calculated in concordance with [99]:

$$T_d = T_a + T_p \tag{4.183}$$

The rotational torque required to counter balance the external load is given by:

$$T_a = k_b F_a \tag{4.184}$$

$$k_b = l_d /(n2\pi)) \tag{4.185}$$

Where k_b is the force to torque conversion and ε represents the ballscrew efficiency. The following equations are valid for the transformation of rotary movement to linear movement:

$$d_a(t) = k_b \theta_a(t) \tag{4.186}$$

$$v_a(t) = k_b \omega_a(t) \tag{4.187}$$

$$a_a(t) = k_b \frac{d}{dt} \omega_a(t) \tag{4.188}$$

Where θ_a, ω_a, v_a and a_a are the angle, the angular velocity, the velocity and the acceleration evaluated at the contact point between the screw shaft and the nut. Figure 4.51 illustrates the TLM model for the interrelation between the nut and the screw shaft.

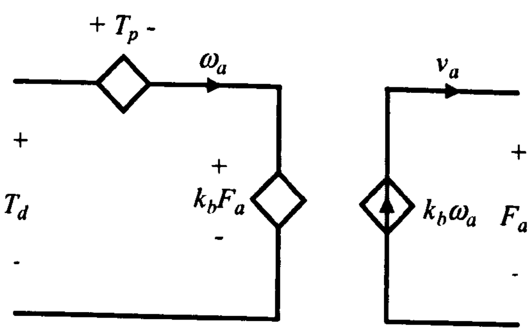


Figure 4.51 TLM model for the nut with pretension

The dynamic effect of the torque T_p acting on the screw shaft is similar to the effect produced by the load component of a frictional force, thus

$$\omega_a = 0 \quad for \qquad |P| < F_p \tag{4.189}$$

Where
$$P = T_d - k_b F_a \tag{4.190}$$

4.4.9 Screw Shaft with Moving Nut

The screw shaft is considered an elastic shaft of length l_{ss} with diameter d_{ss}, mass polar moment of inertia per unit length I_o and polar moment of inertia of the cross section J_{ss}; made of a material with Young's modulus E_{ss}, shear modulus G_{ss} , and mass density ρ_{ss}. The shaft is mounted on two bearings as shown in Figure (4.52).

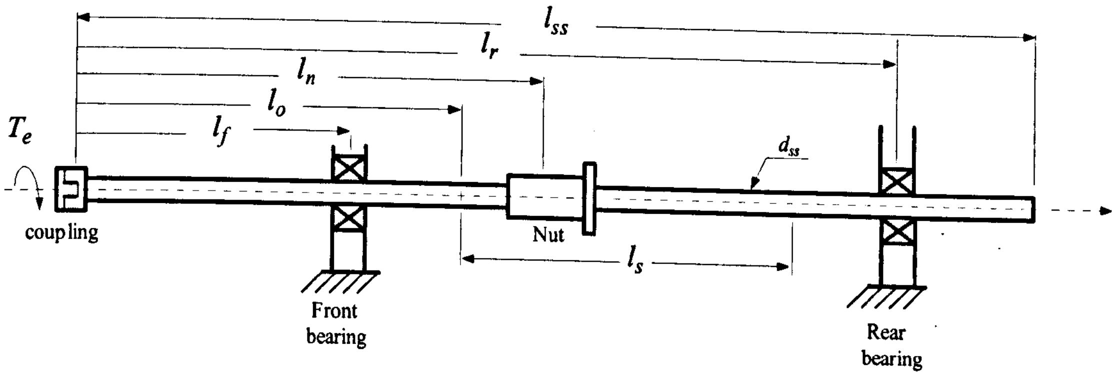


Figure 4.52 Ball screw arrangement

The positions of the front bearing (l_f), rear bearing (l_r) and nut (l_n) are defined taking as a reference the screw end attached to the coupling. l_o is the absolute position of the reference point for the movement of the nut. The ball screw stroke length is denoted l_s. It is also considered that a counter clockwise rotation of the shaft (looking down on the axis) will cause the nut to move towards the rear bearing (positive direction of motion).

As Rao [100] presented, the dynamic behaviour of a shaft subjected to torque about its longitudinal axis is represented by:

$$G_{ss} J_{ss} \frac{\delta^2 y(x,t)}{\delta x^2} + \tau_c(x,t) = I_o \frac{\delta^2 y(x,t)}{\delta t^2} \tag{4.191}$$

Where $y(x,t)$ represents either the angle of twist $\theta(x,t)$ of the cross section or the torque $T(x,t)$; and $\tau_c(x)$ is the external torque acting on the shaft per unit length. If $\tau_c(x) = 0$ (free vibration) and $I_o=\rho_{ss}J_{ss}$ (uniform cross section), equation (4.191) reduces to

$$\frac{\delta^2 y(x,t)}{\delta x^2} = \frac{1}{u_t} \frac{\delta^2 y(x,t)}{\delta t^2} \tag{4.192}$$

Where the velocity of propagation of torsional waves on the material is

$$u_t = \sqrt{G_{ss} / \rho_{ss}} \tag{4.193}$$

Equation (4.193) is modelled by a TLM link (See Table 3.5 pp. 42) with the following characteristics:

$$Z_t = I_o u_t \tag{4.194}$$

$$t_t = l_{ss} / u_t \tag{4.195}$$

u_t represents the speed of propagation of the torsional waves, Z_t is the characteristic impedance, and t_t is the propagation time of torsional waves on the material. As can be seen, for a given shaft the speed of propagation depends on the material characteristics, the characteristic impedance is dependent on the geometry of the shaft (because the inertia is defined by the geometry), and the propagation time depends on the length of the shaft.

To include the dynamic effect of the moving nut, the shaft is divided into h_t equal sections as shown in Figure 4.53. This approach assumes that the dynamic behaviour of the shaft is approximately the same between the limits of each section. This procedure will also help with the synchronisation of the simulation when including other distributed components. A negative effect of that discretisation of space is that the number of natural frequencies of the system is limited by the number of sections. The limiting case will be an infinite number of sections, each infinitesimally small, which is precisely the distributed-parameter model. As more sections imply the necessity of more computational resources, a general solution is to limit the number of sections according to the frequencies of interest. This is a normal practice when using other numerical methods as FEM and FDM, as stated by Doebelin [101].

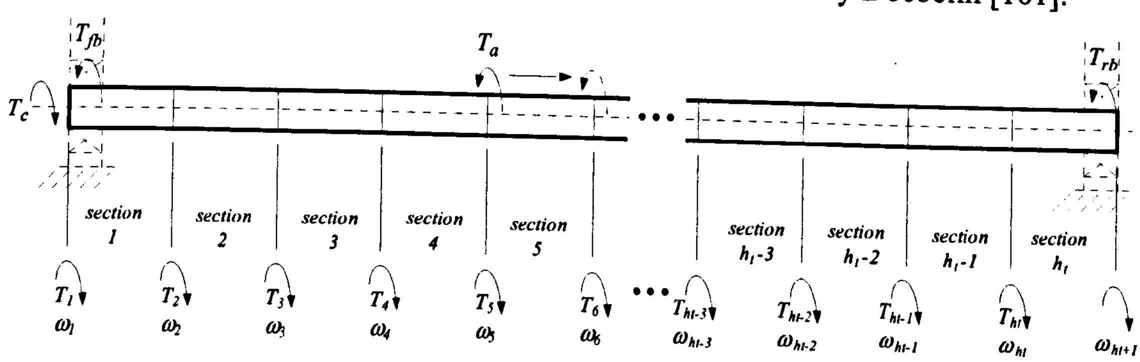


Figure 4.53 Screw shaft divided into h_t sections

In Figure 4.53, the point where the dynamic effect of the nut (torque T_a) affects the shaft changes as the nut moves (like in the real system), but jumping from section to section. As a result, a model that changes with time is obtained. T_i and ω_j, for j = 1,2,3 … h_t+1, represent the torques and angular velocities at the boundaries of the sections.

The same approach is applied for the equation of motion for the longitudinal vibration – equation (4.196). The function $y(x,t)$ represents either the force F acting on the shaft or the axial displacement of the shaft d_a and u_a the velocity of propagation of axial waves.

$$u_a = \sqrt{E_{ss} / \rho_{ss}} \tag{4.196}$$

$$\frac{\delta^2 y(x,t)}{\delta x^2} = \frac{1}{u_a}\frac{\delta^2 y(x,t)}{\delta t^2} \tag{4.197}$$

The model for the equation (4.196) is a TLM link with the following characteristics:

$$Z_a = \rho_{ss} A_{ss} u_a \tag{4.198}$$

$$t_a = l_{ss} / u_a \tag{4.199}$$

Where, Z_a is the characteristic impedance and t_a is the propagation time of axial waves on the material. The constant cross-sectional area of the shaft (A_{ss}) is defined as

$$A_{ss} = \pi (d_{ss}/2)^2 \tag{4.200}$$

Each section in the torsional model is modelled by a TLM link with characteristic impedance Z_t and incident voltages A^i_j and B^i_j. A TLM link with characteristic impedance Z_a and incident voltages A^i_{ja} and B^i_{ja} model each section of the axial model. As a result, the model parameters of each section in the torsional model are: the velocity of propagation u_t, the impedance Z_t, and the propagation time t_t, were:

$$t_t = l_{ss} / (h_t u_t) \tag{4.201}$$

The velocity of propagation u_a, the impedance Z_a, and the propagation time t_a are the parameters of each section in the axial model:

$$t_a = l_{ss} / (h_a u_a) \tag{4.202}$$

As can be seen, the torsional and axial propagation velocities are different (equation (4.193) and (4.197). This leads to different torsional and axial propagation times for the same section length. A synchronisation method must be implemented to assure that torsional and axial waves arrive to the same point at the same time.

A solution to this modelling restriction is achieved by setting up the equations for the torsional and axial models according to the procedure presented in Appendix C thus:

- The screw shaft is divided into h_t sections for the torsional model and the torsional model is synchronised with the motor and coupling models by setting the length of each section (l_{tor}) such as the propagation time becomes a specified t_{pwm} sampling time, thus:

$$t_t = t_{pwm} \tag{4.203}$$

$$h_t = floor\left(\frac{l_{ss}}{t_{pwm}\sqrt{G_{ss}/\rho_{ss}}}\right) \tag{4.204}$$

$$l_{tor} = l_{ss} / h_t \tag{4.205}$$

floor means to round the value between parentheses to the nearest integer towards minus infinity.

- Then, u_t and Z_t become: $$u_t = l_{tor} / t_{pwm} \quad (4.206)$$

$$Z_t = I_o u_t \quad (4.207)$$

- As a result, the material density of the shaft is estimated as:

$$\rho_{ss} = G_{ss} / u_t^2 \quad (4.208)$$

- Each torsional section is divided into n_t axial sections to assure that axial and torsional pulses arrive to the same point at the same time. Subsequently the number of sections of the axial model (h_a) will be n_t times the number of sections in the torsional model, (n_a= 8 and n_t=5)

$$h_a = n_t h_t \quad (4.209)$$

- The length of each section in the axial model (l_{axial}) will be

$$l_{axial} = l_{ss} / h_a \quad (4.210)$$

- The propagation time and the velocity of propagation for the axial model are:

$$u_a = u_t n_a / n_t \quad (4.211)$$

$$t_a = l_{axial} / u_a \quad (4.212)$$

- Z_a is calculated using equation (4.198).

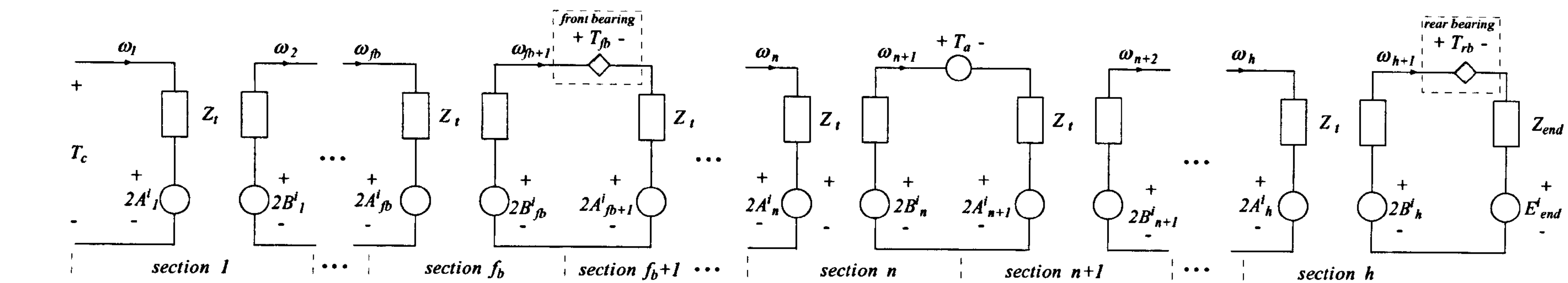
Appendix D presents the derivation of the TLM torsional and axial models for a ball screw. Figure 4.54a shows the derived TLM torsional model of the screw shaft. The front bearing is on the f_b section, the nut is on the n section, and the rear bearing is on the h_t section. The TLM axial model for the screw shaft on a fixed/fixed configuration is shown in Figure 4.54b. Figure 4.54c shows the TLM axial model for a fixed/supported configuration.

4.5 Transducers

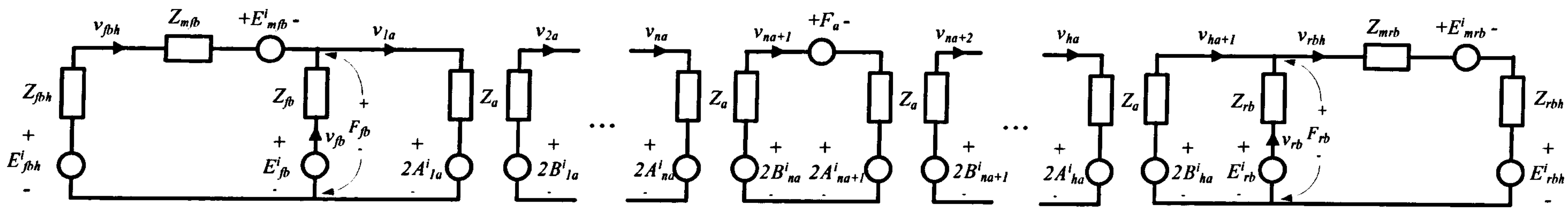
Two types of encoders are used in applications for digital feed drives: Rotary and linear encoders. The rotary encoder is mounted on the motor and performs the following roles:

- Tachometer for speed actual value sensing.
- Rotor position encoder for inverter control.
- Indirect measuring system for the position control loop.

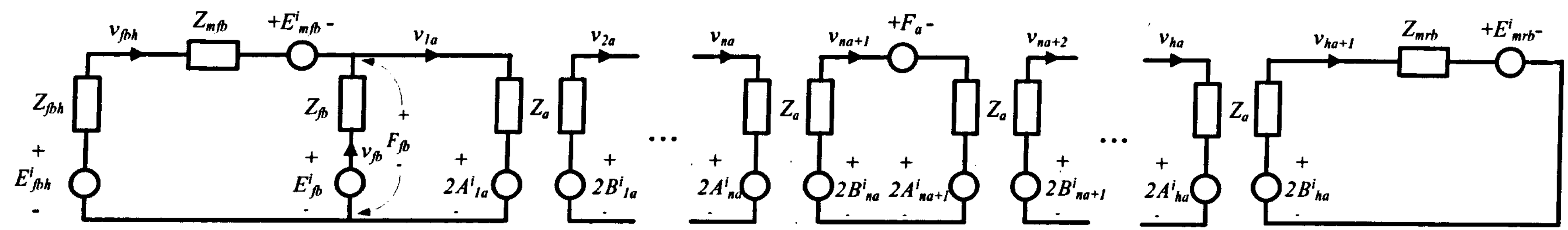
The linear encoder is used as a direct measuring system for the position control loop. These encoders (sin/cos type) operate on the principle of photo-electrical scanning of a very fine grating. Two scanning principles can be used depending on the fineness of the grating [102]:



a) TLM model of the torsional effect

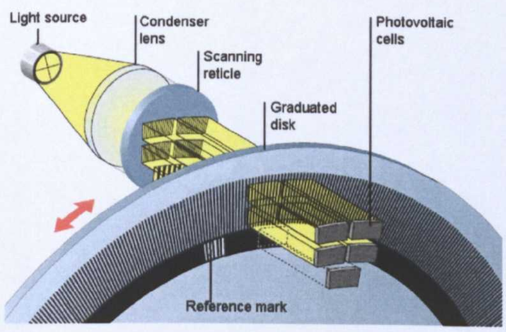


b) TLM model of the axial effect (fixed/fixed bearing configuration)

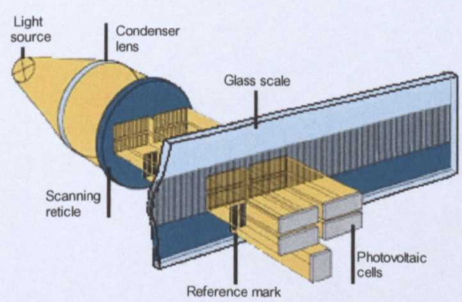


c) TLM model of the axial effect (fixed/supported bearing configuration)

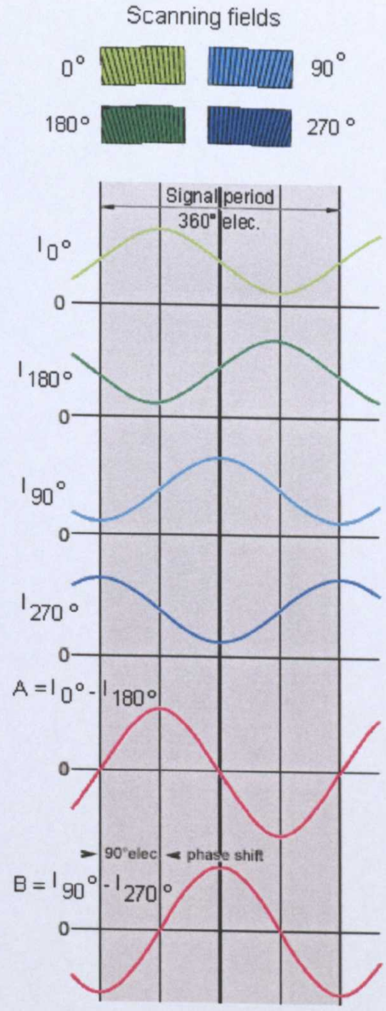
Figure 4.54 Torsional and axial TLM models for a ball screw



a) Rotary encoder



b) Linear encoder



c) Signal generation

Figure 4.55 Sin/cos incremental encoders [102]

- *The imaging principle* for rotary encoders, angle encoders and linear encoders with grating distances of *20 μm* to *100 μm*.
- *The interferential principle* for linear encoders with grating distances of *8 μm* and *4 μm*.

In the imaging principle, a scale with a line grating (glass graduation carrier) is moved relative to another grating with the same structure (the scanning reticle) modulating a beam of light whose intensity is sensed by photoelectrical cells. Figure 4.55 shows this principle for rotary and linear encoders. The scanning reticle has four line gratings, which are offset to each other by one-fourth of a grating. The photocells for the incremental track generate four sinusoidal current signals as shown in Figure 4.55c. These current signals are added to produce two 90° phase-shifted (electrical) sinusoidal signals (A and B).

A second track carries a reference mark that modulates a reference mark signal R at a maximum once per (mechanical) revolution. This signal often serves to locate a specific position during the shaft rotation. Movement direction is determined by detecting which one of the two quadrature-encoded signals (A or B) is the leading sequence. The incremental count and hence the incremental position is determined by a timer that counts up when A is the leading sequence and counts down when B is the leading sequence.

When digitised, both edges of A and B are counted, thus one incremental step is equivalent to a 90° phase shift of the signals, A and B. The incremental position for a rotary encoder, ϕ, is given by [103].

$$\phi = (360/4N)incr + \phi_0 \tag{4.213}$$

Where *incr* is the timer count or incremental count, N is the line count of the encoder and ϕ_o is the zero position. The line count of a rotary encoder is the number of periods of signals A and B over one mechanical revolution. The incremental position for a linear encoder (S) is given by equation (4.214). S_o represents the zero position and T_s is the output signal period.

$$S = (T_s/4)incr + S_o \tag{4.214}$$

One of the major advantages of the sine encoder is the ability to "interpolate" each complete sine wave, which greatly increases the system's resolution. The phase φ of the sinusoidal signals A and B can be used to interpolate the position between two consecutive line counts or four incremental steps, which are equivalent to each other.

As an example, a sin/cos encoder with a resolution of 2048 line per revolution (line count) used with an amplifier that has an interpolation factor of 256, provides an encoder output resolution of $2048 * 256 * 4 = 2097152$.

$$\varphi = \begin{cases} 90 + \arctan(B/A) & A \geq 0 \\ 270 + \arctan(B/A) & A < 0 \end{cases} \tag{4.215}$$

sin/cos encoders with Z track include two auxiliary sinusoidal channels called C & D, whose specifications are the same as for incremental signals A & B. Signals C & D are used to provide absolute positioning within one revolution.

Rotary and linear encoders are regarded as lumped parameter elements. Therefore, the following actions are considered for modelling purposes:

- Rotor inertia (J_{re}) of the rotary encoder will be added to the motor inertia.
- Coefficient of friction of the rotary encoder bearings (μ_{re}) will be added to the motor bearings coefficient of friction.
- Mass of the linear encoder (m_{le}) will be added to the table mass.
- The required moving force (F_{le}) for the linear encoder will be added to the static force of friction of the guideways.
- These are considered as feedback elements with transfer function equal to one.

4.6 Summary

The development of a transmission line model for the elements of a typical arrangement of a CNC feed drive has been presented in this chapter. TLM models for the torsional and axial dynamics of the screw shaft were derived. In this regard a synchronisation approach between the axial and torsional models was depicted.

A modelling example of a shaft divided into eight sections was undertaken in order to derive the general equations for the TLM model. It was concluded from this exercise that pulses are propagated throughout the shaft until a disturbance is present in the system (torque or force). At those points incident pulses are reflected according to the boundary conditions. Therefore the equations of the model are reduced to calculate the velocities and incident pulses at the sections affected by the perturbations; and the propagation of incident pulses on the other sections.

These TLM models of the feed drive elements will be taken as the basis for the modelling of a single-axis and a two-axis feed drive of a Cartesian CNC machine tool in the next chapter.

5. TLM MODELS FOR CNC MACHINE TOOL FEED DRIVES

This chapter presents the development of TLM models for single-axis and two-axis CNC feed drives. The TLM model for a single-axis test rig is presented in the first section (the test rig is representative of the y-axis of a Bridgeport Vertical Machining Centre). This model is taken as the basis for the modelling of one-axis and two-axis feed drive of a Cartesian CNC machine tool as described in the second and third sections (5.2 and 5.3) respectively. The effects of geometric errors and the displacement of masses in a two-axis machine tool are also considered.

5.1 TLM Model of the Single-Axis CNC Feed Drive

The test rig (Figure 5.1) is fitted with a TNC-426PB Heidenhain motion controller, a SIMODRIVE-611 Siemens inverter, and a ball screw arrangement directly coupled to a permanent magnet synchronous motor.

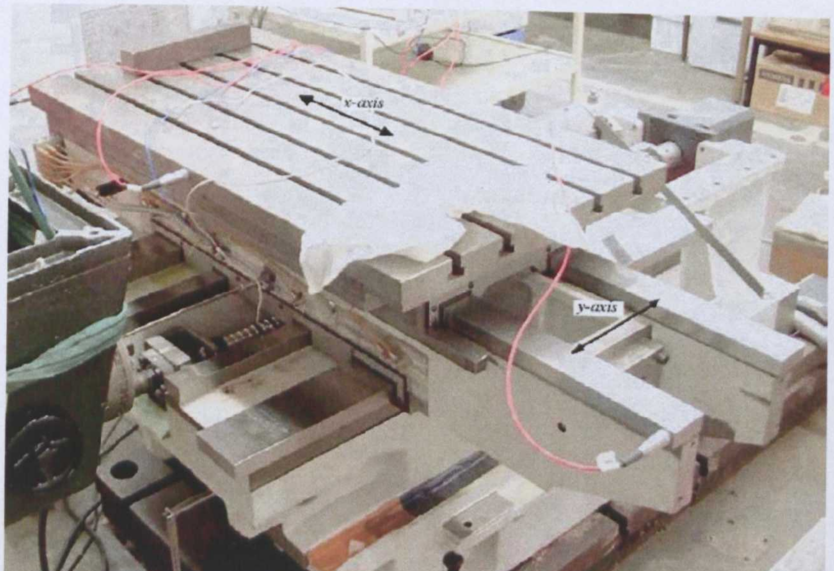


Figure 5.1 Bridgeport test rig

The TNC-426PB motion controller offers digital control for up to five-axis machining centre. Functions of interpolation, position control, speed control, current control and PWM generation are combined into one unit; therefore, the motion controller manages each inverter unit by means of PWM signals. The control of each axis is implemented with an algorithm following the principle of cascade control in conjunction with velocity and acceleration feed-forward (see Figure E.1 pp 216). Functions for compensation of errors resulting from mechanical imperfections (backlash and position axis error) are also included. Backlash errors are compensated by subtracting a predefined value from the position encoder signal after a reversal in direction. The position errors caused by errors in the machine geometry are compensated by subtracting predetermined values from the position encoder signals according to values held in a looking up table [70].

The SIMODRIVE-611 consists of a common feed module that provides the DC voltage link from the power supply mains and a set of drive modules that activate each motor. In the case of the test rig, the drive module consists of a power module (inverter) and an interface card that communicates the TNC-426PB motion controller with the power module.

The nut of the ball screw system is preloaded and the screw shaft is mounted on preloaded bearings on a fixed-fixed configuration.

The model of the axis feed drive is built by interconnecting the TLM models of the rig elements according to the TLM models presented in Chapter 4. In this regard, the dynamics acting on the axis feed drive are defined by the interrelation of three blocks: The motion controller, the inverter & motor electrical equations, and motor mechanical equations & mechanical transmission elements as illustrated in Figure 5.2. Appendix E contains the technical data of the test rig.

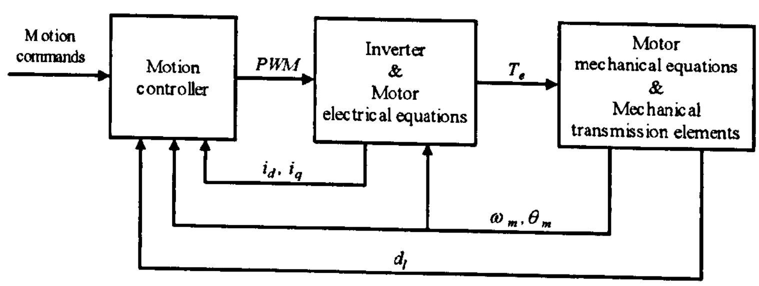


Figure 5.2 Test rig block diagram

5.1.1 Motion controller

The motion controller (Figure 5.3) is implemented in software featuring the algorithms for position, velocity and current control at different sampling rates [70]. That is:

- The interpolator generates a reference position value d_e every 3 ms.
- The position controller generates a reference velocity value v_{ref} every 3 ms.
- The velocity controller generates a reference current value i_{qref} every 0.6 ms.
- The current controller gives a reference voltage value e_{dqref} to the PWM generator at a rate of 0.2 ms.

The dynamic response-matching filter (1st order delay filter) is used to delay the position profile signal according to the transient response during acceleration and deceleration (the equivalent position time constant of the closed position control loop. Delay values can be set in the interval 1 to 255 ms.

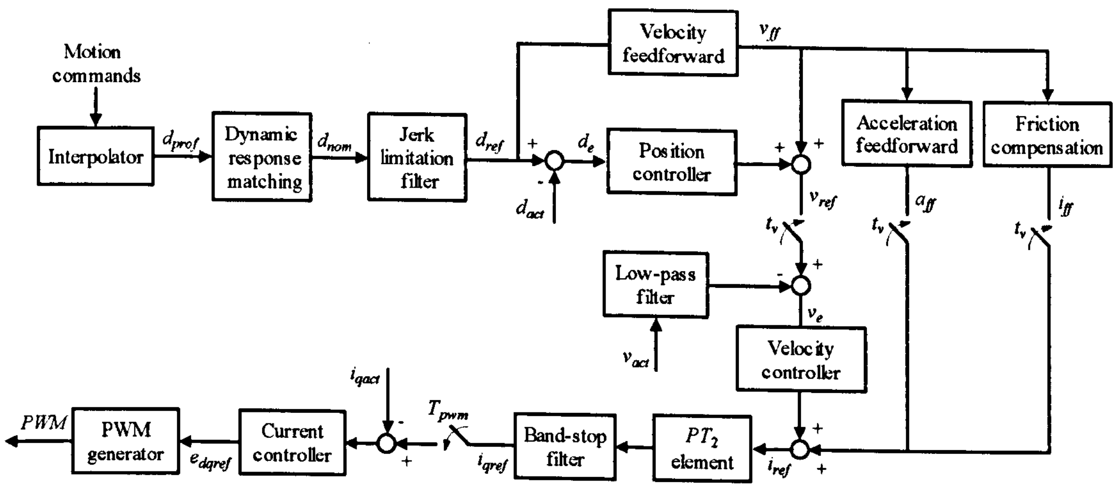


Figure 5.3 TNC426PB Block Diagram [77]

The jerk limitation filter is used to adapt the position profile to the machine dynamics in order to attain high machining velocity. The coefficients of the filter are calculated according to the minimum order of the filter and the tolerance for contour transitions defined by the user.

The low-pass filter is used to damp the fundamental frequency of the TNC when it is higher than 600 Hz. A 1st-order low-pass filter is used when the oscillation frequency is between 600 and 700 Hz. A 2nd-order low-pass filter is used if the oscillation frequency is higher than 700 Hz.

The PT_2 element is used to include a delay in the reference current (i_{ref}) to damp the frequency interference oscillations. Normal values are in the interval: 0.3 to 2 ms.

The Band-rejection filter is used to damp oscillations that cannot be compensated with the differential factor of the velocity controller, the *PT_2* element, or the low-pass filter.

Sliding friction is compensated within the range of the velocity controller by compensating the sliding friction at low velocity and at the rated velocity of the motor. The compensation at low velocity is achieved by feeding forward the reference current value (measured at approximately 10 rpm) at every change in direction. The compensation at the rated velocity is done feeding forward the current i_{ff_rv} according to the value of the reference velocity (equation (5.2). A delay filter is included to prevent overcompensation when the traverse direction is reversed at high feed rates. In a circular interpolation test, such overcompensation appears in the form of reversal spikes that jut inward.

$$i_{ff}(k) = i_{ff\_l} + i_{ff\_rv}(k) \tag{5.1}$$

$$i_{ff\_rv}(k) = \begin{cases} i_{ffm} & v_{ref}(k) \geq i_{ffm} \\ v_{ref}(k)k_{iff} \;\; for & \left|v_{ref}(k)\right| < i_{ffm} \\ -i_{ffm} & v_{ref}(k) \leq -i_{ffm} \end{cases} \tag{5.2}$$

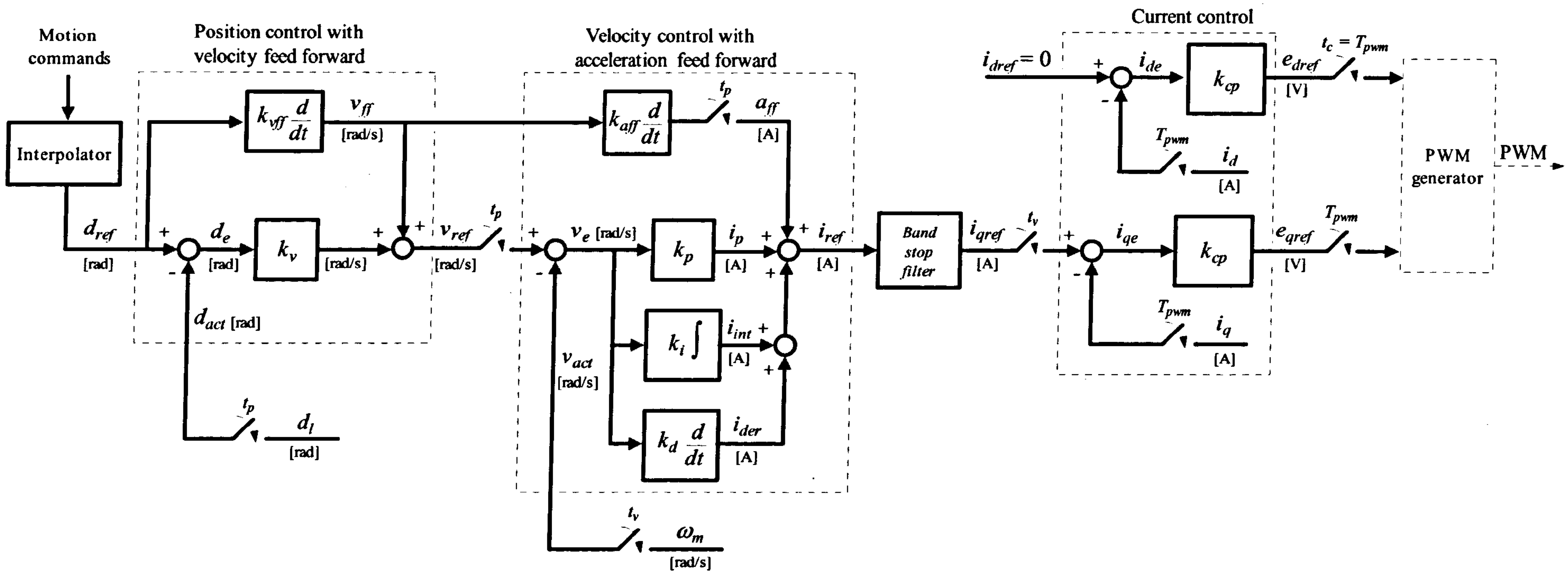


Figure 5.4 Motion controller block diagram

Where i_{ffm} is the reference current measured at the rated velocity of the motor, i_{ff_l} is the reference current at 10 rpm and k_{iff} is the scaling factor.

The jerk-limitation filter, the low-pass filter, the PT_2 element and the friction compensation are not included in the motion controller model because these modules are not active in the actual configuration of the controller. Therefore the motion controller model is reduced to the block diagram shown in Figure 5.4.

The *interpolator* generates the position profile (d_{ref}) for the axis according to the procedure presented in section 4.3.1.

The *position controller with velocity feed forward* has the structure presented in section 4.3.2. The TLM model for this module is represented by equations (4.60) to (4.63) for a sample time t_s=3 ms.

The *velocity controller with acceleration feed forward* has the structure presented in section 4.3.3. Equations (4.55 – 4.61) represent the TLM model for this module (t_s= 0.6 ms)

The *band-stop filter* is implemented as the transposed direct-form II structure (Figure 5.5) of equation (4.80), where n-1 is the filter order. This is a canonical form that has the minimum number of delay elements [104]. At sample k, the routine computes the difference equations:

$$i_{qref}(k) = num(1)i_{ref}(k) + zz_1(k-1) \tag{5.3}$$

$$zz_1(k) = num(2)i_{ref}(k) + zz_2(k-1) - den(2)i_{qref}(k) \tag{5.4}$$

$$zz_{len-3}(k) = num(len-2)i_{ref}(k) + zz_{len-2}(k-1) - den(len-2)i_{qref}(k) \tag{5.5}$$

$$zz_{len-2}(k) = num(len0)i_{ref}(k) + zz_{len0}(k-1) - den(len0)i_{qref}(k) \tag{5.6}$$

$$zz_{len0}(k) = num(len)i_{ref}(k) - den(len)i_{qref}(k) \tag{5.7}$$

$$len = len0{+}1 \tag{5.8}$$

Where, *len0* is the filter order, and *num* and *den* represent the numerator and denominator filter coefficients. The delay outputs $zz_i(1)$, i = 1, ..., *len0* are initialised to 0. This is equivalent to assuming both past inputs and outputs are zero.

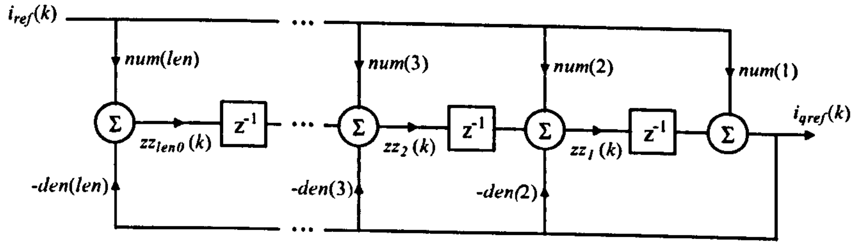


Figure 5.5 Transposed direct-form II structure [104]

The current controller is an implementation of the model presented in section 4.3.4 when integral term k_{ci} is equal to zero. Equations (4.85 – 4.91) are then reduced to:

$$i_{de}(k) = i_{dref}(k) - i_{dact}(k) \tag{5.9}$$

$$e_{dref}(k) = k_{cp} i_{de}(k) \tag{5.10}$$

$$i_{qe}(k) = i_{qref}(k) - i_{qact}(k) \tag{5.11}$$

$$e_{qref}(k) = k_{cp} i_{qe}(k) \tag{5.12}$$

The PWM generator is modelled according to section 4.3.4.

5.1.2 Inverter and Motor (Electrical)

The TLM model for the motor has the structure presented in section 4.2.1. Equations (4.16, 4.17, 4.19, 4.20 and 4.23 - 4.25) conform the TLM motor model when $t_s = t_{pwm}$. The inverter is modelled according to section 4.3.4. Figure 5.6 shows the block diagram for the interconnection of the current controller, PWM generator, inverter and motor. Some blocks in this figure can be removed to speed up the simulation of the model (The reduced block diagram is illustrated in Figure 5.7):

- Block 1 is reading the signal e_α, e_β and block 2 is giving the same signal back.
- Block 4 is reading the signal i_d, i_q and block 6 is giving it back.

5.1.3 Motor (Mechanical) and Mechanical Transmission Elements

As was established in section 4.4.8, the screw shaft is considered a distributed parameter element, which is divided into various sections in order to include the dynamic effect of the moving nut.

Two models were defined to analyse the dynamic behaviour of the ballscrew: a torsional model and an axial model. The application of the TLM transform to both models lead to different torsional and axial propagation velocities and therefore, to different torsional and axial propagation times for the same section length.

The synchronisation of the axial and torsional models was achieved by setting up the parameters of the torsional and axial models (Z_t, t_t, u_t, Z_a, t_a, u_a) according to the procedure presented in Appendix C.

Under these circumstances, the motor and the mechanical transmission elements are modelled for two sampling times as follows:

- The motor, the coupling and the torsional model of the screw shaft are modelled at the torsional sampling time t_t.
- The nut, the table and the axial model of the screw shaft are modelled at the axial propagation time t_a.

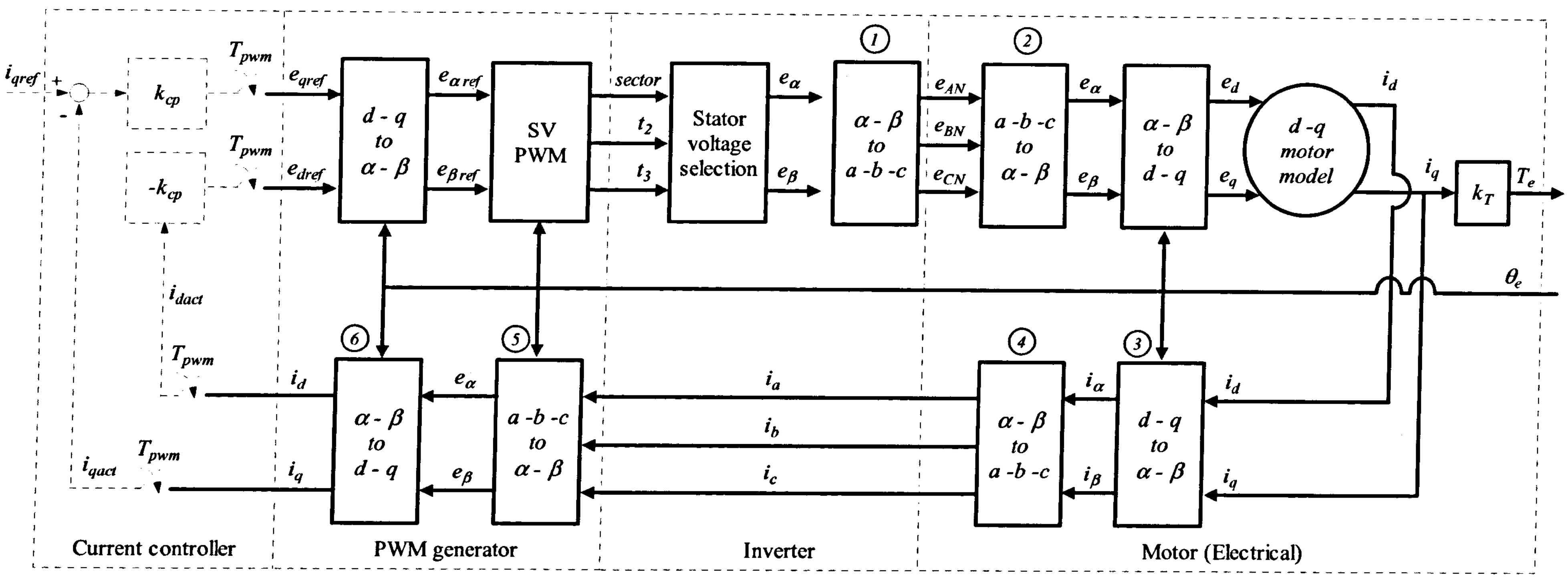


Figure 5.6 PWM generator, inverter and motor block diagram

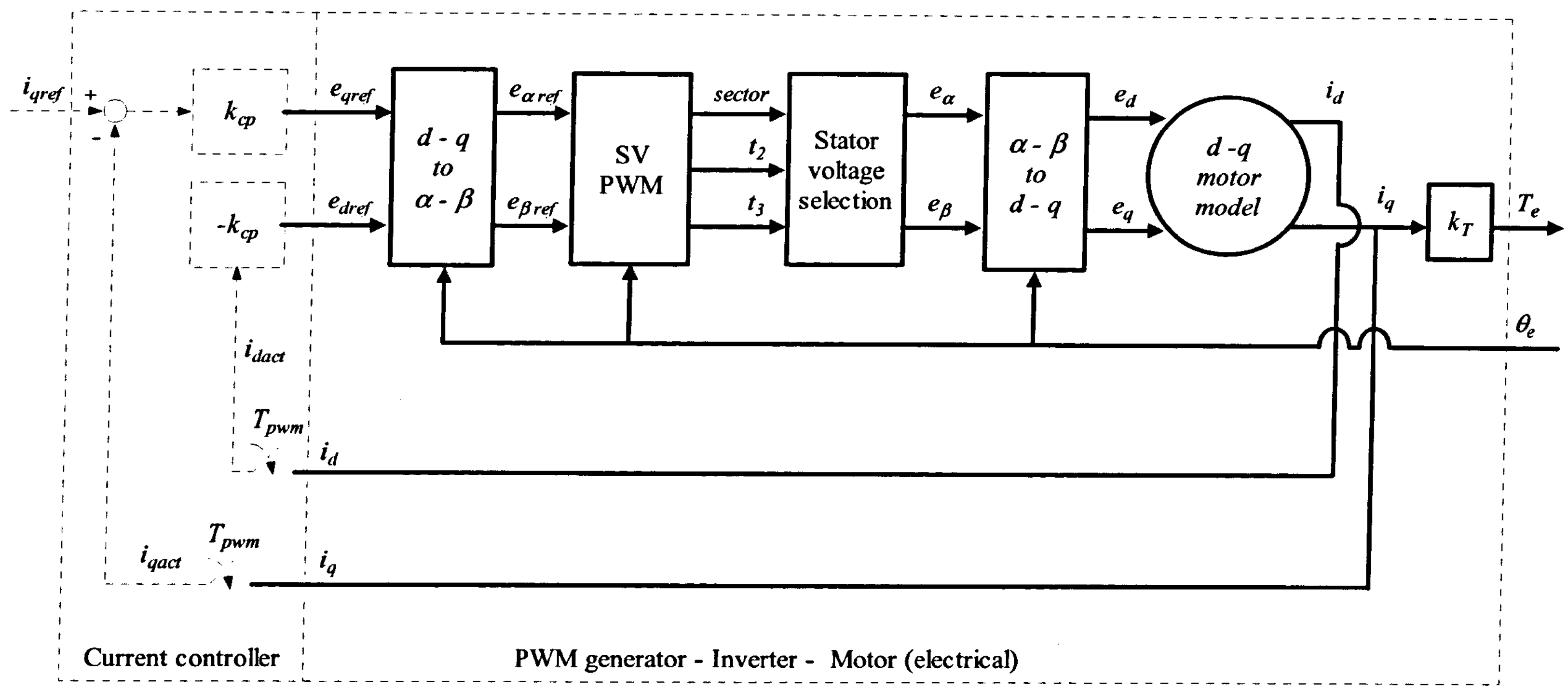


Figure 5.7 PWM generator, inverter and motor reduced block diagram

5.1.3.1 Motor Mechanical Equations and Coupling

Figure 5.8 shows the TLM model of the motor (mechanical equations) and the coupling according to sections 4.2.2 and 4.4.5.

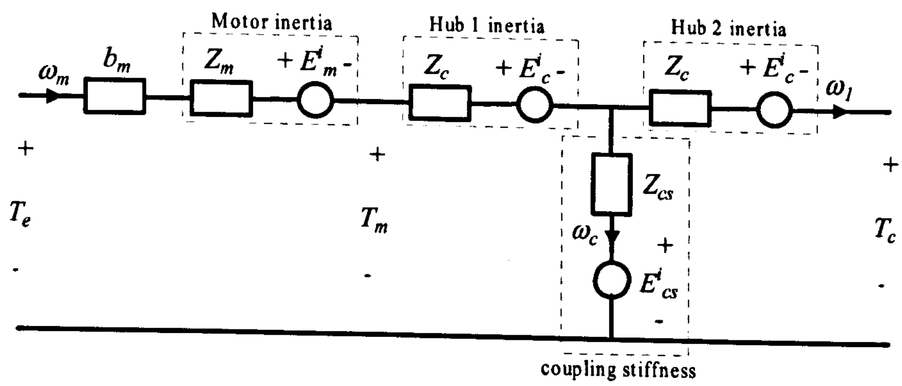


Figure 5.8 TLM model for the motor (mechanical) and the coupling

The inertias of the rotary encoder and the hub 1 can be added to the inertia of the motor to simplify the calculations. Hence,

$$J_{mc} = J_m + J_c + J_{re} \tag{5.13}$$

This reduction of the model is possible because those inertias are modelled as lumped parameter elements. The resultant TLM model is illustrated in Figure 5.9a. This electric circuit is solved finding the Thevenin equivalent with respect to T_{cs} (Figure 5.9b), thus:

$$E_{eq}(k) = Z_{Ecs}E^i_{cs}(k) + Z_{Ect}E_{ct}(k) \tag{5.14}$$

$$Z_{eq} = \frac{Z_{cs}Z_{ct}}{Z_{cs} + Z_{ct}} \tag{5.15}$$

Where,

$$Z_{Ecs} = Z_{ct}/(Z_{cs} + Z_{ct}) \tag{5.16}$$

$$Z_{Ect} = Z_{cs}/(Z_{cs} + Z_{ct}) \tag{5.17}$$

$$Z_{ct} = Z_c + Z_t \tag{5.18}$$

$$E_{ct}(k) = E^i_c(k) + 2A^i_1(k) \tag{5.19}$$

$$\omega_m(k) = M_{\omega m}(T_e(k) - E^i_{mc}(k) - E_{eq}(k)) \tag{5.20}$$

$$T_{cs}(k) = \omega_m(k)Z_{eq} + E_{eq}(k) \tag{5.21}$$

$$\omega_1(k) = M_{\omega 1}(T_{cs}(k) - E_{ct}(k)) \tag{5.22}$$

$$E^i_{cs}(k+1) = T_{cs}(k) \tag{5.23}$$

$$E^i_c(k+1) = -Z_c\omega_1(k) \tag{5.24}$$

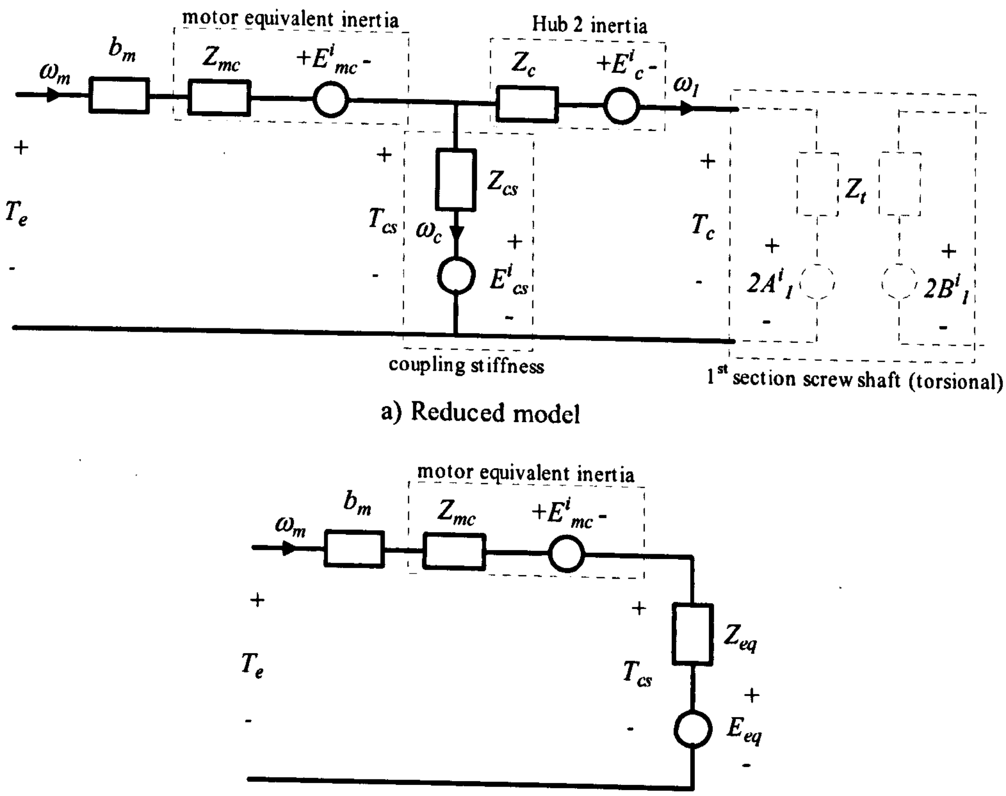
$$E^i_{mc}(k+1) = -Z_{mc}\omega_m(k) \tag{5.25}$$

$$B^i_1(k+1) = \omega_1(k)Z_t + A^i_1(k) \tag{5.26}$$

Where, $$M_{\omega m} = 1/(b_m + Z_{mc} + Z_{eq}) \tag{5.27}$$

$$M_{\omega 1} = 1/Z_{ct} \tag{5.28}$$

$$Z_{mc} = J_{mc} / t_t \tag{5.29}$$



a) Reduced model

b) Thevenin equivalent circuit

Figure 5.9 Reduced TLM model for the motor (Mechanical) and coupling

Equations (5.14 –5.28) represent the TLM model of the motor mechanical equations & coupling.

5.1.3.2 Screw Shaft Torsional Model

The presence of the supporting bearings in the TLM model of the shaft generates the reflection of pulses arriving to the sections where they are placed (see Appendix D Figure D.3). In that case, the propagation of pulses in the TLM model takes place on two specific zones (loops) as it is graphically represented in Figure 5.10a. The front bearing is placed on section f_b, the nut is on section n_t and the rear bearing is on section h_t, where:

$$h_t = round(l_r / l_{tor}) \tag{5.30}$$

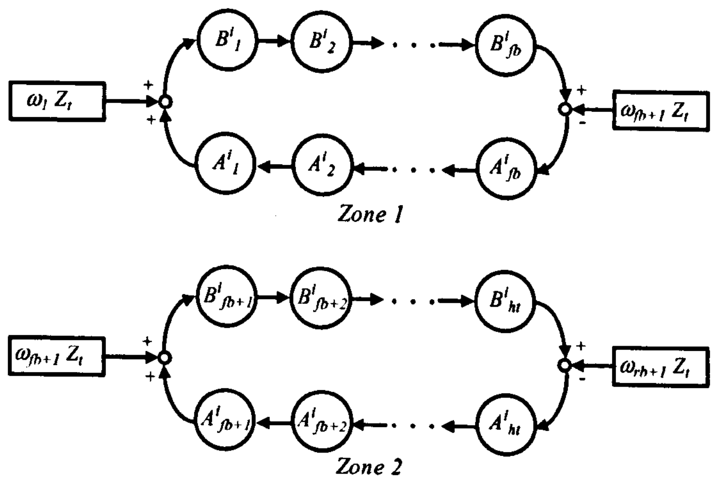
$$l_{end} = l_{ss} - h_t l_{tor} \tag{5.31}$$

$$J_{end} = l_{end} I_o \tag{5.32}$$

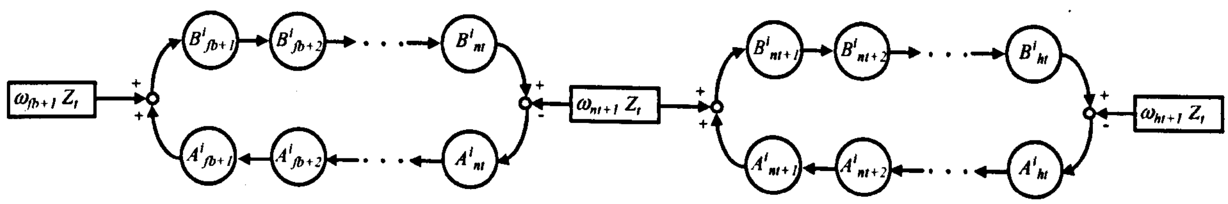
$$f_b = round(l_f / l_{tor}) \tag{5.33}$$

$$n_t = ceil(l_n / l_{tor}) \tag{5.34}$$

The inclusion of the nut in the model will cause the reflection of pulses arriving to section n_t, and therefore splitting the *zone 2* in two loops (case *c* in Appendix D, Figures D.6 and D.7) as shown in Figure 5.10b. The model is then reduced to the calculation of the angular velocity on sections one, f_b, n_t and h_t; and the propagation of pulses on the other sections.



a) Without nut



b) *Zone 2* including the nut

Figure 5.10 Pulses propagation model for the screw shaft torsional model with moving nut

The velocity of the front bearing (ω_{fb+1}) is calculated including the TLM model derived for the bearing's friction in Section 4.4.3 (see Figure 5.11), thus

$$T = 2(B^i_{fb}(k) - A^i_{fb+1}(k)) \tag{5.35}$$

$$\omega_{fb+1}(k) = \begin{cases} 0 & \text{for} \quad |T| < T_{fb1} \\ (T - sign(T)T_{fb1})/Z_{fbeq} & \quad |T| \geq T_{fb1} \end{cases} \tag{5.36}$$

Where
$$Z_{fbeq} = 2Z_t + b_{fb} \tag{5.37}$$

Next pulses:
$$A^i_{fb}(k+1) = B^i_{fb}(k) - \omega_{fb+1}(k)Z_t \tag{5.38}$$

$$B^i_{fb+1}(k+1) = A^i_{fb+1}(k) + \omega_{fb+1}(k)Z_t \tag{5.39}$$

The angular velocity ω_{nt+1} is calculated according to the procedure specified in section 5.1.3.4 The pulse propagation is defined by

$$B^i_{nt+1}(k+1) = A^i_{nt+1}(k) + \omega_{nt+1}(k)Z_t \tag{5.40}$$

$$A^i_{nt}(k+1) = B^i_{nt}(k) - \omega_{nt+1}(k)Z_t \tag{5.41}$$

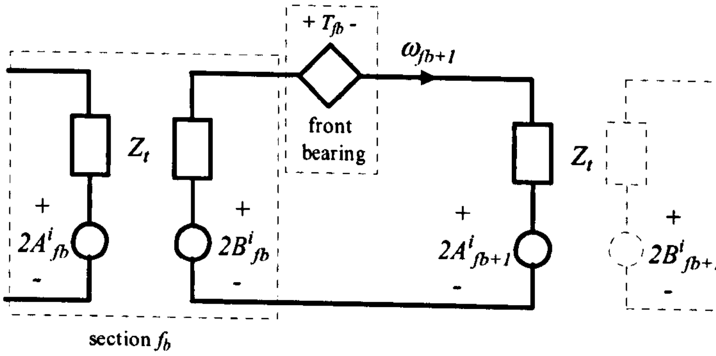


Figure 5.11 Section f_b of the torsional model

The velocity of the rear bearing (ω_{ht+1}) is calculated using the procedure applied to the front bearing (see Figure 5.12).

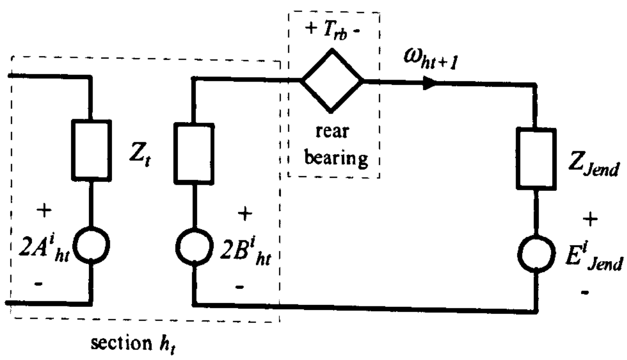


Figure 5.12 Section h_t of the torsional model

$$T = 2B^i_{ht}(k) - E^i_{Jend}(k) \tag{5.42}$$

$$\omega_{ht+1}(k) = \begin{cases} 0 & \text{for } |T| < T_{rb1} \\ (T - sign(T)T_{rb1})/Z_{rbeq} & \text{for } |T| \geq T_{rb1} \end{cases} \tag{5.43}$$

$$Z_{rbeq} = Z_t + b_{rb} + Z_{Jend} \tag{5.44}$$

Next pulses: $$A^i_{ht}(k+1) = B^i_{ht}(k) - \omega_{ht+1}(k)Z_t \tag{5.45}$$

$$E^i_{Jend}(k+1) = -\omega_{ht+1}(k)Z_{Jend} \tag{5.46}$$

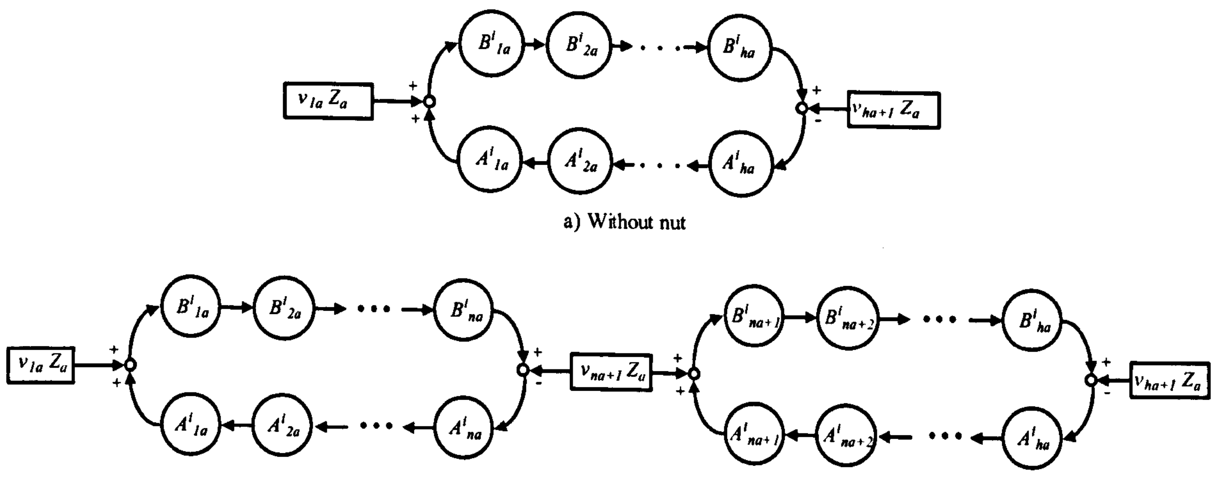
The propagation of A^i and B^i pulses on the other sections is given by:

$$B^i_j(k+1) = B^i_{j-1}(k) \quad \text{for } j = 2,\ldots, h_t \qquad j \neq f_b+1,\ n_t+1 \tag{5.47}$$

$$A^i_j(k+1) = A^i_{j+1}(k) \quad \text{for } j = 1,\ldots, h_t-1 \quad j \neq f_b,\ n_t,\ h_t \tag{5.48}$$

5.1.3.3 Screw Shaft Axial Model

As was established in Appendix D.2, the presence of the supporting bearings and the nut generates the reflection of pulses arriving to the sections where they are placed. This leads to the propagation of pulses on one zone (loops) in the axial model as it is graphically represented in Figure 5.13a.



a) Without nut

b) Including the nut

Figure 5.13 Pulses propagation model for the screw shaft axial model with moving nut

The inclusion of the nut in the model will cause the reflection of pulses arriving to section na, and therefore splits the model in two loops as shown in Figure 5.13b. The front bearing is placed on the first section, the nut is on section n_a and the rear bearing is on section h_a, where:

$$h = round(l_r / l_{axial}) \tag{5.49}$$

$$l_{end} = l_{ss} - l_{axial}h \tag{5.50}$$

$$f_{ba} = round(l_f / l_{axial}) \tag{5.51}$$

$$l_{front} = f_{ba} l_{axial} \tag{5.52}$$

$$h_a = h - f_{ba}) \tag{5.53}$$

$$n_a = ceil((l_n - l_{front}) / l_{tor}) \tag{5.54}$$

The model is reduced to the calculation of the velocities v_{1a}, v_{na+1} and v_{ha+1}; and the pulse propagation on the other sections as defined by the procedure presented in Appendix D.2:

- Equations (D.33 – D.47) for the calculation of velocity v_{1a}
- Equations (D.48 – D.62) for the calculation of velocity v_{ha+1}
- Equations (D.63 – D.64) for the pulse propagation.

5.1.3.4 Screw Shaft, Nut and Table

Figure 5.14 illustrates the connection of the axial and torsional TLM screw shaft models with the nut and table models according to sections 4.4.6 to 4.4.9, thus

$$\omega_{nt+1}(k) = \begin{cases} 0 & \\ M_{wnt1}(T - sign(T)T_p) \end{cases} for \begin{matrix} |T| \le T_p \\ |T| > T_p \end{matrix} \tag{5.55}$$

$$v_{nt+1}(k) = k_b \omega_{nt+1}(k) \tag{5.56}$$

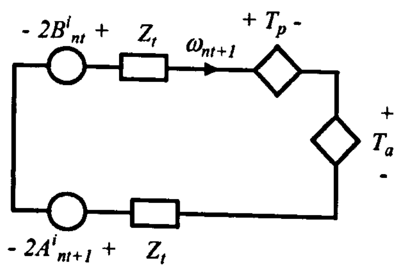
where

$$T_a(k) = k_b F_a(k-1) \tag{5.57}$$

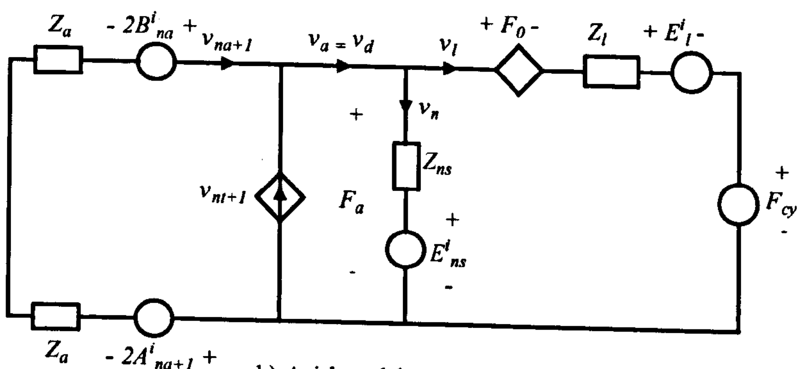
$$T = 2BA - T_a(k) \tag{5.58}$$

$$BA = B^i_{nt}(k) - A^i_{nt+1}(k) \tag{5.59}$$

$$M_{wnt1}(k) = 1/(2Z_t) \tag{5.60}$$



a) Torsional model connection



b) Axial model connection

Figure 5.14 TLM model of the connection between nut and screw shaft

The components of the frictional force F_{gw} are calculated according to section 4.4.4, as follows:

$$F_{Ry} = m_{ly} + F_{cz} \tag{5.61}$$

$$F_{Ly} = |F_{cx}| \tag{5.62}$$

$$F_E = X_y F_{Ry} + Y_y F_{Ly} \tag{5.63}$$

$$F_0 = F_{gw0} + b_{gw} F_E \tag{5.64}$$

$$F_1 = b_{gw} v_l \tag{5.65}$$

The velocity of the table (v_l) is calculated finding the Thevenin equivalent with respect to F_a (Figure 5.15).

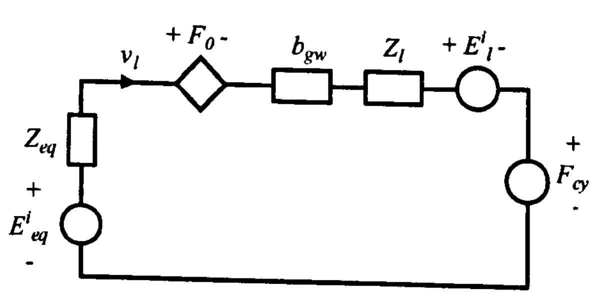


Figure 5.15 Equivalent model for the connection between nut and screw shaft

$$E_{eq} = v_{n+1}(k)Z_{eq} + Z_{CD}CD + Z_{Ens}E^i_{ns}(k) \tag{5.66}$$

where,

$$CD = B^i_{na}(k) - A^i_{na+1}(k) \tag{5.67}$$

$$Z_{eq} = 2Z_aZ_{ns}/(2Z_a + Z_{ns}) \tag{5.68}$$

$$Z_{CD} = 2Z_{ns}/(2Z_a + Z_{ns}) \tag{5.69}$$

$$Z_{Ens} = 2Z_a/(2Z_a + Z_{ns}) \tag{5.70}$$

$$v_l(k) = \begin{cases} 0 & \text{for} \quad |F| \le F_0 \\ M_{vl}(F - sign(F)F_0) & \quad\quad |F| > F_0 \end{cases} \tag{5.71}$$

where

$$F = E_{eq} - E^i_l(k) - F_{cy}(k) \tag{5.72}$$

$$M_{vl} = 1/(Z_{eq} + b_{gw} + Z_l) \tag{5.73}$$

$$F_a(k) = E_{eq} - v_l(k)Z_{eq} \tag{5.74}$$

$$E^i_l(k+1) = -v_l(k)Z_l \tag{5.75}$$

$$E^i_{ns}(k+1) = F_a(k) \tag{5.76}$$

$$v_{na+1}(k) = M_{vna1}(2CD - F_a(k)) \tag{5.77}$$

$$M_{vna1} = 1/(2Z_a) \tag{5.78}$$

$$d_l(k) = v_l(k)t_a + E^i_{dl}(k) \tag{5.79}$$

$$E^i_{dl}(k+1) = d_l(k) \tag{5.80}$$

It must be noted that a ballscrew with preload is assumed to have no or minimal backlash. However, a model for the ball screw with backlash is included in order to make the model applicable to both cases:

- Ballscrew with pretension in the nut (Figure 5.14) $Backlash = 0$.
- Ballscrew without pretension in the nut (Figure 5.16) $T_p = 0$ and $Backlash \neq 0$.

The backlash model presented in section 4.4.1.2 has been reduced to the following two possible states:

- When the screw shaft is not in contact with the nut (Figure 5.16a).
- When the screw shaft is in contact with the nut (Figure 5.16b).

The state in which the axis will start at the beginning of the simulation depends on the following conditions:

- Non-contact if:
  - $d_d \neq d_a$

- $d_d = d_a$ and the direction of motion is negative (nut moving towards the motor).
- $d_d = d_a + Backlash$ and the direction of motion is positive.

- Contact:
  - $d_d = d_a$ and the direction of motion is positive.
  - $d_d = d_a + Backlash$ and the direction of motion is negative.

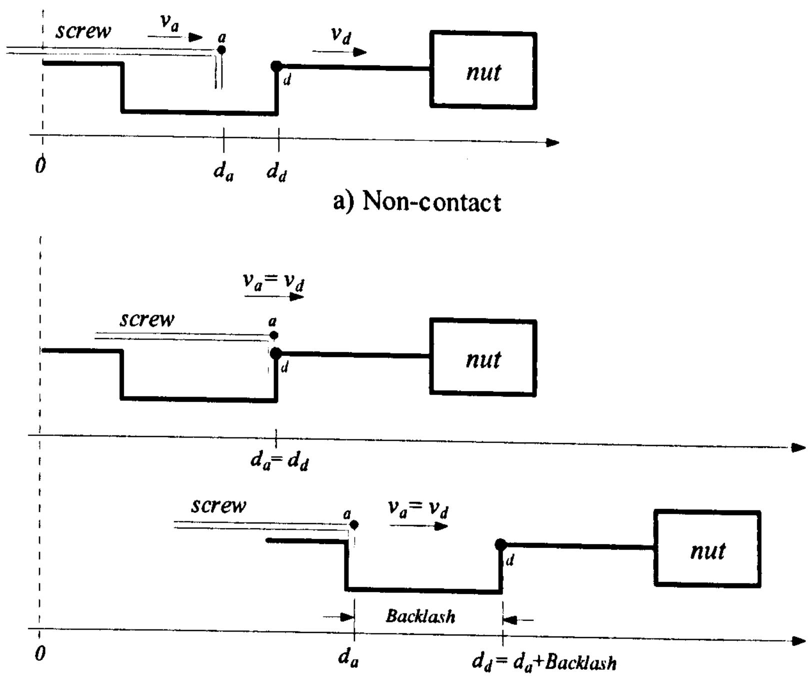


Figure 5.16 Backlash model

Variables T_a and F_a are zero if the screw shaft is not in contact with the nut. Thus the model in Figure 5.14 is reduced to the model illustrated in Figure 5.17.

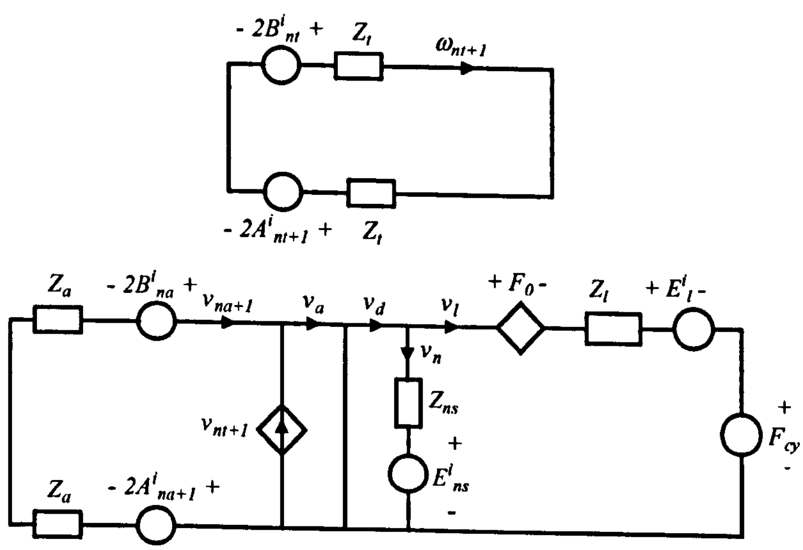


Figure 5.17 TLM model when the screw shaft is not in contact with the nut

Values for the velocities are given by the following equations:

$$\omega_{nt+1}(k) = BA / Z_t \tag{5.81}$$

$$v_{nt+1}(k) = k_b \omega_{nt+1}(k) \tag{5.82}$$

$$v_{na+1}(k) = CD / Z_a \tag{5.83}$$

$$v_a(k) = v_{nt+1}(k) + v_{na+1}(k) \tag{5.84}$$

$$v_n(k) = -E_{ns}^i(k) / Z_{ns} \tag{5.85}$$

$$v_l(k) = \begin{cases} 0 & \text{for } |F| \le F_0 \\ M_{vl}(F - sign(F)F_0) & |F| > F_0 \end{cases} \tag{5.86}$$

$$v_d(k) = v_l(k) + v_n(k) \tag{5.87}$$

where

$$F = -E_l^i(k) - F_{cy}(k) \tag{5.88}$$

$$M_{vl} = 1/(b_{gw} + Z_l) \tag{5.89}$$

$$E_{ns}^i(k+1) = 0 \tag{5.90}$$

$$F_a(k) = 0 \tag{5.91}$$

$$E_l^i(k+1) = -v_l(k) Z_l \tag{5.92}$$

The positions d_a, d_d are calculated integrating the velocities for the sampling time t_d(equations (5.84) and (5.87). d_l is calculated as in the case for preloaded nut. The model remains in this state if $0 < d_d(k) - d_a(k) < Backlash$, otherwise the model changes to the contact state (Figure 5.14 with $T_p = 0$). The model will switch to the non-contact state when the sign of velocity v_{nt+1} changes.

5.2 Single-Axis TLM Model for a CNC Machine Tool Feed Drive

This section describes the extension of the TLM model presented in section 5.1 to the modelling of the x and y axes of a Cincinnati Machine Arrow Series 2 VMC-500 vertical machining centre (Figure 5.18). This machining centre is representative of a three-axis Cartesian CNC machine tool where the X-axis carries the table and the workpiece, the Y-axis carries the X-axis, and the Z-axis is the vertical axis.

Figure 5.18 Cincinnati machine Arrow series 2 VMC-500

The VMC-500 is fitted with a SINUMERIK 840D SIEMENS motion controller, which commands the CNC kernel functions for interpolation and position control. The motion controller is connected to the drives and I/O units via a PROFIBUS-DP interface as shown in Figure 5.19 [105]. Each axis integrates a SIMODRIVE-611 Siemens inverter and a ball screw arrangement directly coupled to a permanent magnet synchronous motor. The ball screw systems incorporate a preloaded nut and the screw shaft mounted on a fixed-supported bearing configuration.

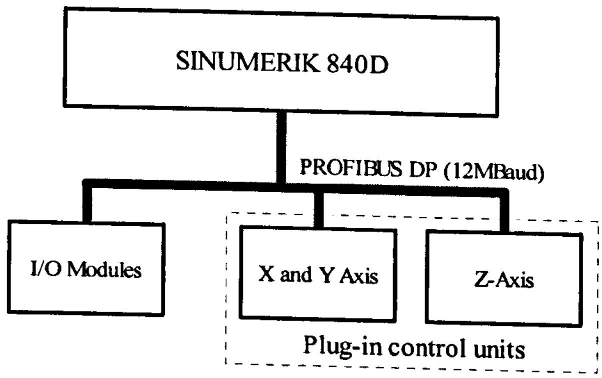


Figure 5.19 SINUMERIK 840D configuration [105]

The SIMODRIVE-611 unit consists of a common feed module that provides the DC voltage link from the power supply mains and a set of drive modules that activate each motor. Every drive module consists of a power module (inverter) and a closed-loop plug-in unit. The closed-loop plug-in unit is dedicated to velocity control, current control and PWM generation functions. Appendix F contains the technical data for the VMC-500 Machine centre.

Figure 5.20 shows the block diagram for the control approach performed by the SINUMERIK 840 D and the plug-in control units. The main differences between the TNC 426PB and the SINUMERIK 840 D are:

- The SINUMERIK 840D includes a velocity response matching filter (1^{st}-order delay-filter) used to delay the velocity feed forward signal according to the equivalent position time constant of the closed velocity control loop.
- The SINUMERIK 840D configuration established for the Cincinnati machining centre does not include acceleration (torque) feed forward.
- A velocity filter is included to damp the resonant frequencies in the closed position loop.
- A velocity limitation in the form of saturation is included in the position loop.
- Torque and current limitations in the form of saturation are included in the velocity loop.
- The velocity controller does not include differential term (PI control).
- Two additional filters are included in the velocity loop in order to get a filtering process with better time/frequency response. For example, Filter 1 can be configured as the PT2

filter in the TNC 426 PB and Filters 2 to 4 can be combined to get a band-rejection filter with better damping and frequency properties than the band-rejection filter in the TNC 426 PB.

- The current controllers include integral term (see PI controller model in section 4.3.4).
- The sample time for the interpolator and position controller is 4 ms.
- The sample time for the velocity controller is 0.125 ms.
- The sample time for the current controller is 0.125 ms.

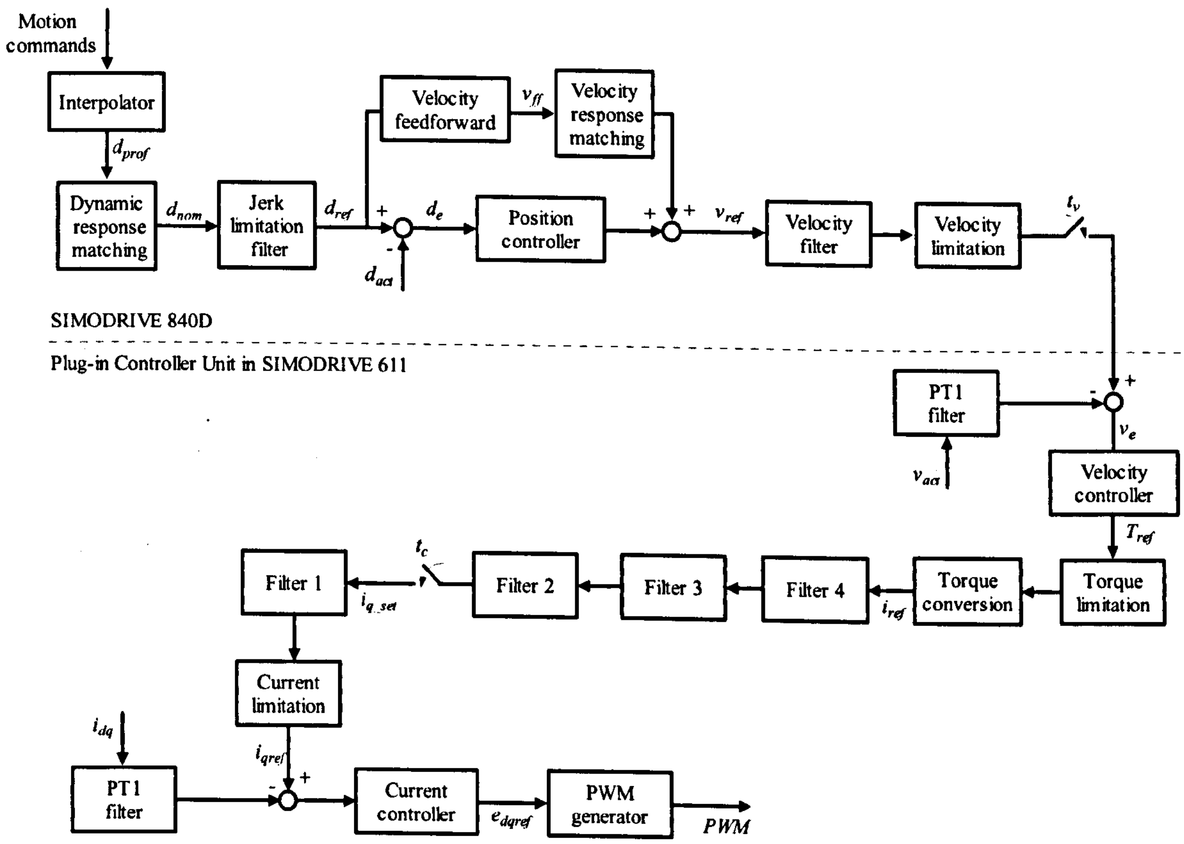


Figure 5.20 Block diagram Siemens controller [106]

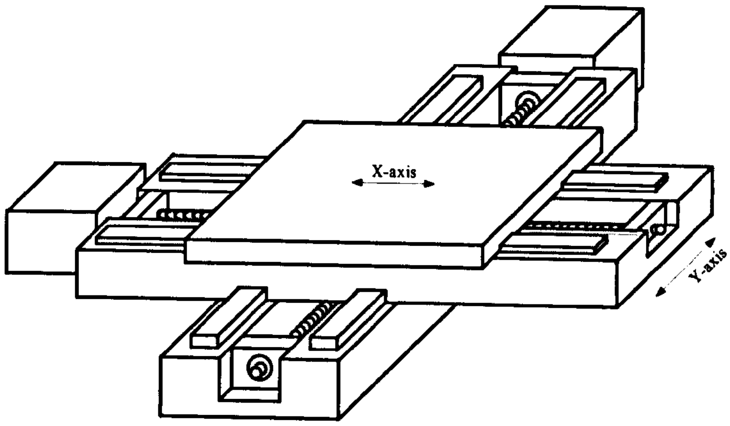
Although the control algorithm is distributed in two different units (SINUMERIK 480D and the plug-in control units) the dynamics acting on the x and y axes are modelled as in the single-axis test rig case (Figure 5.2).

The blocks for the inverter & motor electrical equations and motor mechanical equations & mechanical transmission elements are modelled as presented in sections 5.1.2 and 5.1.3. The model for the rear bearing mounting has been updated as presented in Appendix D.3 to reproduce the fixed-supported bearing configuration.

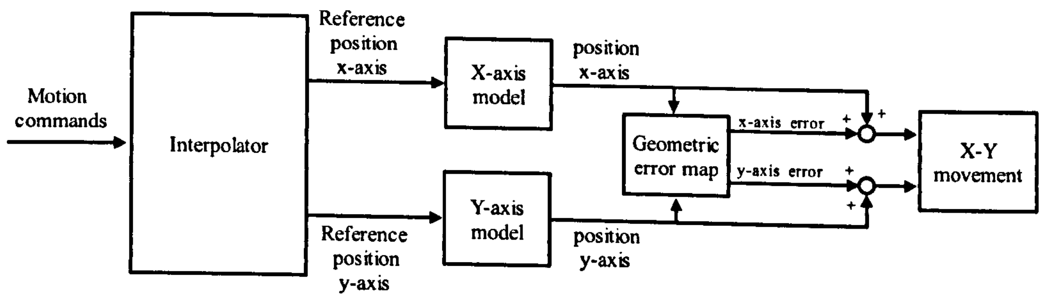
5.3 Implementation of Two-Axis TLM Models

This section presents the TLM model for two-axis feed of a Cartesian CNC machine tool. The two-axis system is configured with the y-axis carrying the x-axis, and the x-axis carrying the worktable as shown in Figure 5.21a. In this regard the TLM model described in section 5.2 is used to model the Y-axis. Linear and circular interpolation methods described in section 4.3.1 are included in the interpolator to coordinate the movement of the axes.

Pre-calibrated geometric errors are included in the form of an error map that is used to correct the control movements over the working zone. Figure 5.21b represents the block diagram for the two- axis model.



a) x-y axis feed drive



b) Block diagram

Figure 5.21 Two-axis feed drive

The equations for the calculation of the geometric errors are presented in Chapter 6.

5.4 Review

The TLM model for a Bridgeport test rig single-axis has been developed according to the TLM models for various elements described in Chapter 4. This Single-axis TLM model will be used to validate the modelling approach in the next Chapter. The single-axis TLM model has been extended to the modelling of a single-axis and two-axis TLM models of a Cincinnati Arrow 500 vertical machining centre (including the effect of geometric errors and moving mass). The purpose of this modelling exercise is to validate the two-axis model against measured data obtained from ball bar tests on the Arrow 500, once the single-axis had been validated.

6. MEASUREMENT TECHNIQUES APPLIED TO POSITION CONTROLLED MECHANISMS WITHIN CNC MACHINE TOOLS

The accuracy of a machine tool is an assessment of the machine's ability to accurately position each one of its axes according to the manufacturing specifications established for a given work-piece. The main factors affecting this accuracy are geometric errors, non-rigid errors, thermal errors and wear [107].

Geometric errors are caused by mechanical imperfections of the machine tool structure and misalignments of the machine tool elements, which are inherent in the production and build of a machine or wear during the lifetime of the machine. These geometrical inaccuracies produce errors in the squareness and parallelism between the machine moving elements. If the machine is a rigid body, the geometric errors can be measured at any point on the machine and will give the same results. If the machine is non-rigid, the error will be different depending upon axes position and load. These errors are often negligible but may have an effect on some machines.

Non-rigid errors (load errors) occur due to loading of the machine structural elements. This could be in the form of a new weight distribution on the machine structure due to the movement of the machine axes, the movement of a heavy work-piece that could induce larger deformations than the axes weight alone, and the forces induced during the cutting process.

Thermal errors are induced by the machine structural elements causing deformation due to temperature changes. Friction in bearings, drive motors and transmission systems (gearbox, ballscrew), draughts through doors and the cutting process are typical sources of temperature gradients. These errors are characterised by a slow time response and have not been considered part of this research.

Wear errors are caused by the contact between moving parts in the machine and increase with time. Wear in the nut; ballscrew and guide-ways can reduce the repeatability of the machine as well as affect the geometric errors. Wear in the cutting tool reduces the size of the tool causing errors in the workpiece and surface finishing. This is a broad field of research, which is outside the scope of this study.

The following sections recount the measurement techniques used for determining the geometric and load errors for the two-axis feed drive system; and the step and jerk-limited responses for the x and y-axis to be evaluated in the next Chapter.

6.1 Geometric and Load Error Measurements

Geometric errors are referred as rigid body errors and therefore are measured without specific consideration of load. Geometric errors can be classified into linear positioning errors, straightness errors, rotational errors and squareness errors [108].

Linear positioning errors are mainly originated by the ball screw pitch error and backlash between the mechanical components of the axis drive. Straightness errors are guide way profile errors due to improper assembling of the guide-way rails or the support bearing interfaces. Rotational errors are produced when a second axis moves. Errors of this type are the roll error (about the axis of travel) and the pitch and yaw errors (about axes perpendicular to the axis of travel). Squareness errors reflect the out-of-squareness of two nominally orthogonal axes. Geometric errors produced for a machine tool slide are shown in Figure 6.1.

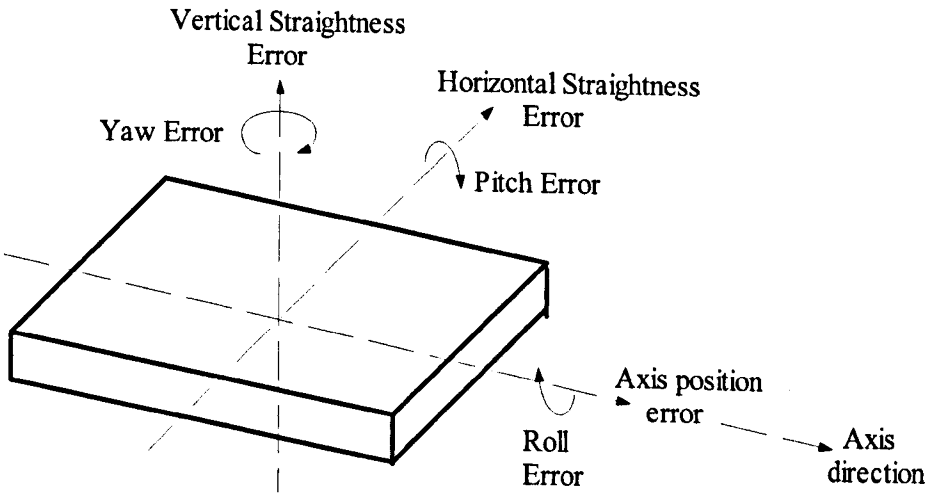


Figure 6.1 Geometric errors for a machine tool slide [108]

The geometric errors for a three-axis Cartesian machine where the X-axis travels on top of the Y-axis saddle are defined by the following equations [109]:

$$E_x = \Lambda_x(x) + \Lambda_x(y) + \Lambda_x(z) + \phi_y(x)D_z + \phi_y(y)D_z + \phi_z(y)D_y + \theta_{xy}(x,y)D_y + \theta_{xz}(x,z)D_z \quad (6.1)$$

$$E_y = \Lambda_y(x) + \Lambda_y(y) + \Lambda_y(z) + \phi_x(x)D_z + \phi_x(y)D_z - \phi_z(x)D_x + \theta_{yz}(y,z)D_z \quad (6.2)$$

$$E_z = \Lambda_z(x) + \Lambda_z(y) + \Lambda_z(z) - \phi_x(x)D_y - \phi_y(x)D_x - \phi_y(y)D_x \quad (6.3)$$

Where, E_x, E_y, E_z : actual error movement of the x, y and z-axis [μm]

D_x, D_y, D_z : x, y and z-axis position [mm]

$\Lambda_x(x)$, $\Lambda_y(y)$, $\Lambda_z(z)$: x, y and z-axis linear positioning error [μm]

$\Lambda_x(y)$: y-axis straightness in the x-axis direction [μm]

$\Lambda_y(x)$: x-axis straightness in the y-axis direction [μm]

$\Lambda_z(x)$: x-axis straightness in the z-axis direction [μm]

$\Lambda_z(y)$: y-axis straightness in the z-axis direction [μm]

$\Lambda_x(z)$: z-axis straightness in the x-axis direction [μm]

$\Lambda_y(z)$: z-axis straightness in the y-axis direction [μm]

$\theta_{xy}(x, y)$: Squareness in the XY plane [μm/mm]

$\theta_{xz}(x, z)$: Squareness in the XZ plane [μm/mm]

$\theta_{yz}(y, z)$: Squareness in the YZ plane [μm/mm]

$\phi_x(x)$: x-axis rotation about x-axis [μm/mm]

$\phi_y(x)$: x-axis rotation about y-axis [μm/mm]

$\phi_z(x)$: x-axis rotation about z-axis [μm/mm]

$\phi_x(y)$: y-axis rotation about x-axis [μm/mm]

$\phi_y(y)$: y-axis rotation about y-axis [μm/mm]

$\phi_z(y)$: y-axis rotation about z-axis [μm/mm]

$\phi_x(z)$: z-axis rotation about x-axis [μm/mm]

$\phi_y(z)$: z-axis rotation about y-axis [μm/mm]

$\phi_z(z)$: z-axis rotation about z-axis [μm/mm]

The geometric error components can be changed by the deformation of the machine structure due to the movement of the machine axes and work-piece weight. A technique for identifying the presence of load or non-rigid errors was presented by Ford et al. [110]. The study showed that:

- The main geometric error components exhibiting a non-rigid effect were the angular error components.
- There was a definite correlation between the change in the angular errors produced by the non-rigid effects and the measured change in the axis linear positioning errors.
- In any compensation or correction strategy it may be adequate to concentrate on the angular error components in order to eliminate non-rigid effects.

In this regard, the non-rigid error components can be inserted in the angular parameters $\phi_x(x)$, $\phi_y(x)$, $\phi_z(x)$, $\phi_x(y)$, $\phi_y(y)$, $\phi_z(y)$, $\phi_x(z)$, $\phi_y(z)$ and $\phi_z(z)$, by relating those error components as a function of the x, y coordinates. Equations (6.1 - 6.3) become:

$$\begin{aligned} E_x = {} & \Lambda_x(x)+\Lambda_x(y)+\Lambda_x(z)+\phi_y(x,y)D_z+\phi_y(y,x)D_z+\phi_z(y,x)D_y \\ & +\theta_{xy}(x,y)D_y+\theta_{xz}(x,z)D_z \end{aligned} \tag{6.4}$$

$$E_y=\Lambda_y(x)+\Lambda_y(y)+\Lambda_y(z)+\phi_x(x,y)D_z+\phi_x(y,x)D_z-\phi_z(x,y)D_x+\theta_{yz}(y,z)D_z \tag{6.5}$$

$$E_z=\Lambda_z(x)+\Lambda_z(y)+\Lambda_z(z)-\phi_x(x,y)D_y-\phi_y(x,y)D_x-\phi_y(y,x)D_x \tag{6.6}$$

The two-axis equations for the X-Y plane are derived considering only the geometric error components associated with the x and y-axis, thus:

$$E_x = \Lambda_x(x) + \Lambda_x(y) + \Lambda_x(z) + \phi_y(x,y)D_z + \phi_y(y,x)D_z + \phi_z(y,x)D_y + \theta_{xy}(x,y)D_y \quad (6.7)$$

$$E_y = \Lambda_y(x) + \Lambda_y(y) + \Lambda_y(z) + \phi_x(x,y)D_z + \phi_x(y,x)D_z - \phi_z(x,y)D_x \quad (6.8)$$

Table 6.1 shows the geometric error components associated with a two-axis machine centre.

Error type	Number of error
Linear positioning errors	2
Straightness errors	4
Rotational errors	6
Orthogonality between axes	1
Total number of errors	13

Table 6.1 Geometric error components associated with two-axis CNC Machine

6.1.1 Equipment Used for the Measurement of the Geometric Errors

Four types of equipment are specified for the measurement of the geometric errors: laser interferometer systems, artefacts (straight edge and precision squares), electronic precision levels and ball-bar systems. Types of equipment specified for the measurement of the geometric errors are as follows:

- Linear positioning: Laser interferometer.
- Straightness measurement: Laser and straight edge.
- Angular measurement: Laser, Talyvel electronic level and two dial gauges.
- Squareness: Granite square artefact and dial gauge, Ballbar and laser with optical square.

The laser interferometer measures distance by analysing the wave interference of two beams: one reflected at fixed distance and the other reflected from a changeable position as shown in Figure 6.2. Linear, angular (pitch and yaw) or straightness measurements, between table and spindle, can then each be made with the appropriate choice of interferometer optics [111].

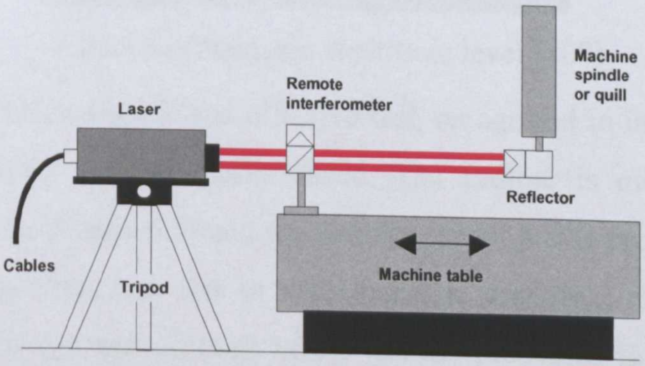


Figure 6.2 Laser interferometer measurement [111]

The straight edge and square are precision artefacts constructed out of granite to provide a great deal of rigidity, thermal stability and hard surface that is smooth and resists damage (Figure 6.3).

Figure 6.3 Granite artefacts [107]

These artefacts are used in conjunction with a dial test indicator that is set-up to run along particular edges of the square and straight edge enabling measurement of squareness between two machine axes. One edge forms the reference and the other is used to measure the perpendicularity.

A Precision Electronic Level (Figure 6.4) is a device used for measuring angular error. It is conformed by a pendulum suspended in oil (for damping) that is affected by change in inclination and an encoder, which measures that change. Two units are used to give a differential reading with a resolution of 0.1 arc-seconds. This is required for measuring machine tools to isolate the angular error of an axis from the movement of the entire machine.

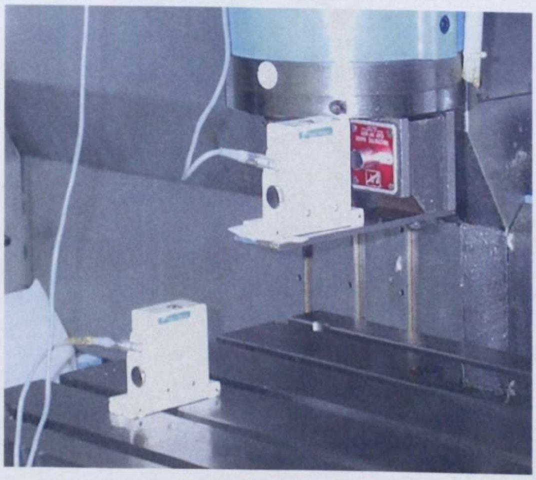


Figure 6.4 Precision electronic level [107]

The Ballbar system provides a quick and effective test, recognised in international standards (e.g. ISO 230.4) to verify machine performance. The Ballbar is mounted between two repeatable magnetic joints (Figure 6.5) and the machine under test is programmed to perform two consecutive circular arcs, one test in the clockwise direction, the other in the anti-clockwise direction. First and last portions of the test are processed to remove acceleration and deceleration effects [112].

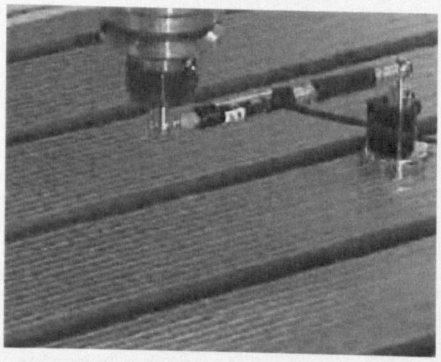


Figure 6.5 Ballbar system (Renishaw) [112]

The tests can be achieved in one of two ways, depending upon the constraints of the setup:

- Over a 360° circle either dynamically or statically.
- Over a 180° arc statically in 45° increments.

The differing tests apply because measurement of a full circle is not practical in the vertical planes, but is easily achievable in the horizontal plane (See figure 6.6). Analysis software extracts useful information from the circular data such as reversal, backlash, squareness, servo mismatch and straightness. Ballbar tests are rapid to execute and can be performed in all three Cartesian planes; however it only gives a snapshot of a region of the machine.

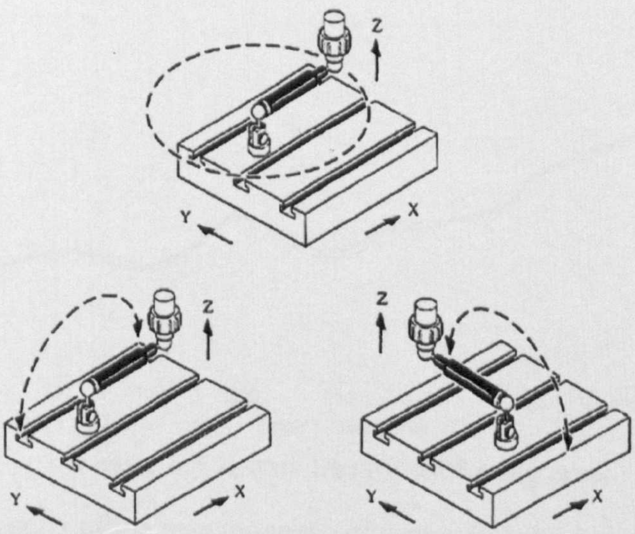


Figure 6.6 Ball-bar tests for 360 and 180 degrees [112]

6.1.2 X-axis Geometric Errors

The geometric error components presented in this section were measured over the full axis travel (500mm) of the axes feed drive in the Arrow 500 Machine tool. The reference point for all the measurements was the origin of the coordinate system specified in the controller (point P in Figure 6.7). Each geometric error was measured bi-directionally, using a step size of 25mm. The process was repeated for a number of runs in accordance with the ISO standard 230-2 [113].

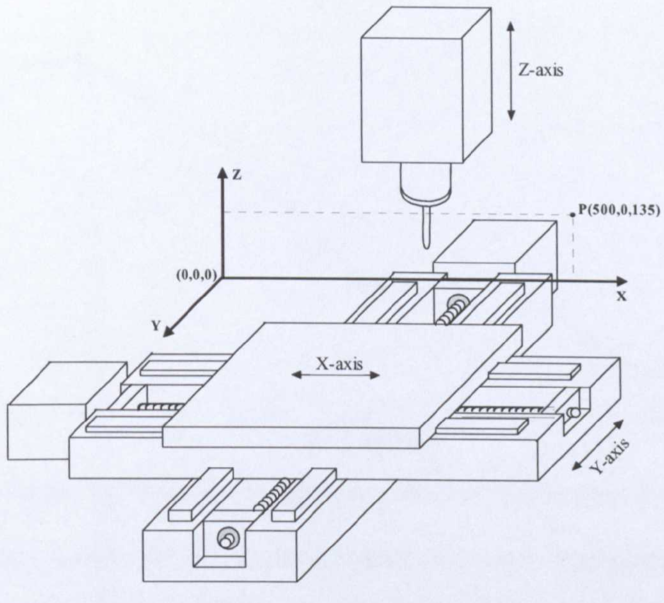


Figure 6.7 Origin of the Cartesian coordinate system for the machine's workspace

A dual electronic Talyvel was used for the measurement of the rotation of the x-axis about the x-axis and the rotation of the y-axis about the y-axis. Each squareness value was calculated from a Ballbar test and a laser interferometer was used for the measurement of all the other geometric errors. The coordinates for the centre of the Ballbar test were (250,250,135).

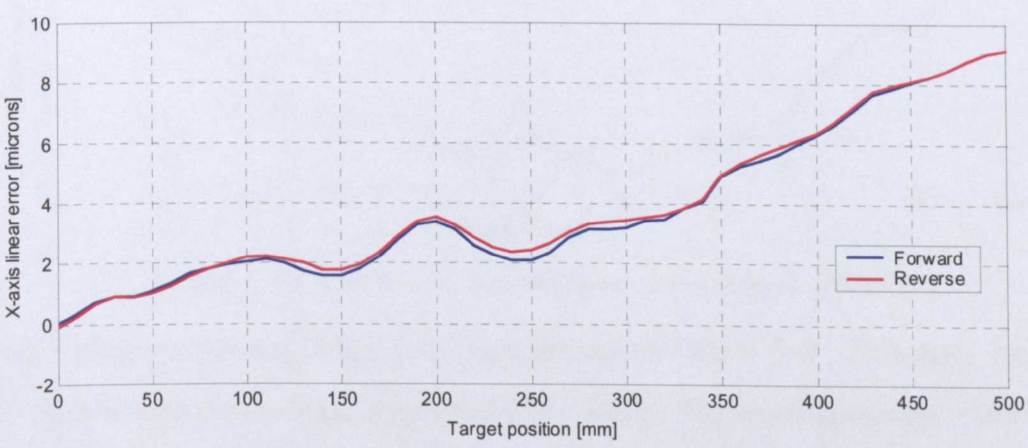


Figure 6.8 X-axis linear positioning error

Figure 6.8 shows the results of the measurement of x-axis linear positioning error ($e_x(x)$). This error has a total range between –0.05µm and 9µm. Although the error trend is irregular in the interval [100, 425] mm, the slope of the error tends to be linear. The progressive error is at its greatest when the x-axis is at the positive extreme of travel (9µm at 500 mm). The axis reversal is negligible. The unidirectional repeatability of the axis was measured as one micron for both the forward and reverse directions.

Figure 6.9 shows the results of the measurement of the x-axis straightness error in the y-axis direction ($e_y(x)$). The straightness error has a total range of 3.5µm to –3.46µm. The reversal for this error component is one micron at most.

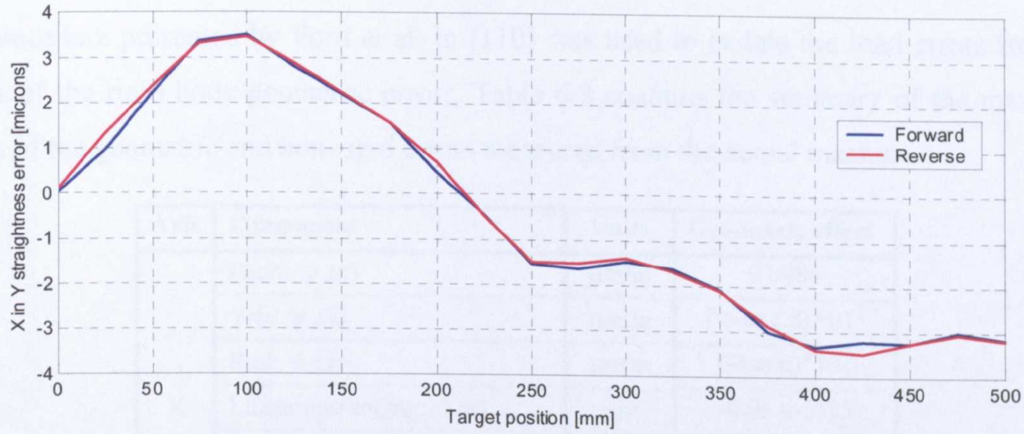


Figure 6.9 X-axis straightness error in the y-axis direction

Figure 6.10 shows the results of the measurement of x-axis straightness error in the z-axis direction ($e_z(x)$). The straightness error has a total range of –0.64μm to –8.84μm. The reversal for this error component was one micron at most.

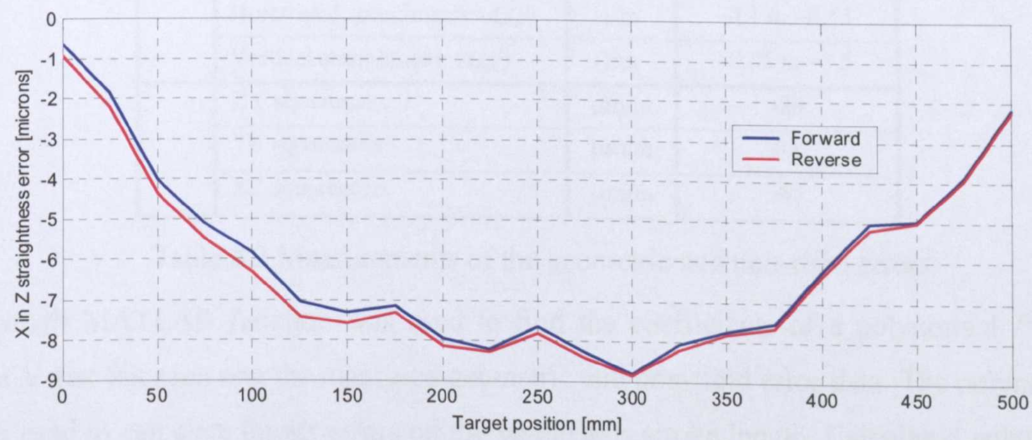


Figure 6.10 X-axis straightness error in the z-axis direction

The x-axis rotation about the z-axis ($\phi_z(x)$) is shown in Figure 6.11. This error has a total range of –5μm/mm to 2.8μm/mm. Appendix G.1 contains the set of geometric errors for the y-axis of the Arrow 500 machine tool.

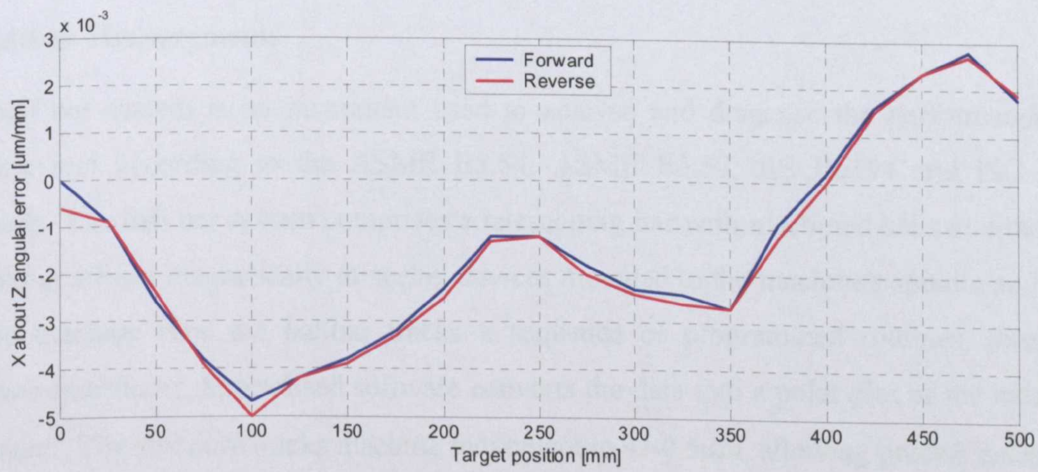


Figure 6.11 X-axis rotation about the z-axis

The procedure presented by Ford et al. in [110] was used to isolate the load errors from the effects of the rigid body geometric errors. Table 6.2 contains the summary of the maximum values of the geometric and non-rigid errors measured from the actual machine.

Axis	Component	Units	Geometric effect
X	Pitch: $\phi_y(x)$	µm/m	0.0484
	Yaw: $\phi_z(x)$	µm/m	(-5 to 2.8)*10^{-3}
	Roll: $\phi_x(x)$	µm/m	(-7 to 6)*10^{-3}
	Linear positioning: $\Lambda_x(x)$	µm	-0.05 to 9.15
	Horizontal straightness: $\Lambda_y(x)$	µm	-3.46 to 3.5
	Vertical straightness: $\Lambda_z(x)$	µm	-8.84 to –0.64
Y	Pitch: $\phi_x(y)$	µm/m	–0.018.5
	Yaw: $\phi_z(y)$	µm/m	(-5 to 2.8)*10^{-3}
	Roll: $\phi_y(y)$	µm/m	0 to 0.028
	Linear positioning: $\Lambda_y(y)$	µm	-0.14 to –13.92
	Horizontal straightness: $\Lambda_x(y)$	µm	-0.1 to –6.53
	Vertical straightness: $\Lambda_z(y)$	µm	-0.05 to –3.5
	XY squareness	µm/m	-26
	YZ squareness	µm/m	64
	XZ squareness	µm/m	-83

Table 6.2 Measurements of the geometric and non-rigid errors

The *polyfit* MATLAB function was used to find the coefficients of a polynomial $P(X)$ of degree N that fits each one the measured geometric and non-rigid error data. The polynomials can be used to calculate theses errors on the whole axis stroke length. Calculated polynomial coefficients and the MATLAB program employed to obtain the coefficients are included in Appendix G.2. The polynomial has the form:

$$P(X) = P(1)X^N + P(2)X^{N-1} + ... + P(N)X + P(N+1) \tag{6.9}$$

Where X is the actual position of the axis feed drive.

6.2 Ballbar Measurements

The ball bar system is an instrument used to analyse and diagnose the performance of a machine tool according to the ASME B5.54, ASME B5.57, JIS B6194 and ISO 230-4 standards. The ball bar system comprises a telescoping bar with machined balls at either end, the ballbar affixes magnetically to socket devices mounted to the machine's spindle and table. As the machine runs the ballbar tracks a sequence of programmed routines, through a precision transducer. Specialised software converts the data into a polar plot of the machine's movement. The software tracks machine movement to +/-0.5µm, allowing precise assessment of machine geometry, circularity and stick/slip error, servo gain mismatch, backlash, repeatability and scale mismatch [112].

The circularity error is the difference between the maximum outward deviation and maximum inward deviation from the best circle through captured data, as defined by the ISO 230-1 standard.

Steady state following error mismatch occurs when the gains of the position and velocity controllers are not properly set. Negative sign indicates that the x-axis leads the y-axis in the XY plane and that the x-axis gain should be reduced. The resultant plot will have the shape of two ellipses in different contouring direction.

The backlash is mainly caused by the elastic deformation of the ball screw arrangement and play between the nut and the screw shaft. This error is characterised by spikes occurring at the zones where the velocity direction changes.

The repeatability is calculated according to the ISO 230-2 standard. The scale mismatch indicates that one of the axes is over travelling or under travelling. The higher the feedrate, the lower the mismatch error.

The ballbar measurements (Figure 6.12) were undertaken for the nominal length of 150 mm (circle of 300mm diameter) at a feedrate of 1000 mm/min. Angular overshoot of 180° before and after data capture for a two cycle 360° data capture was utilised.

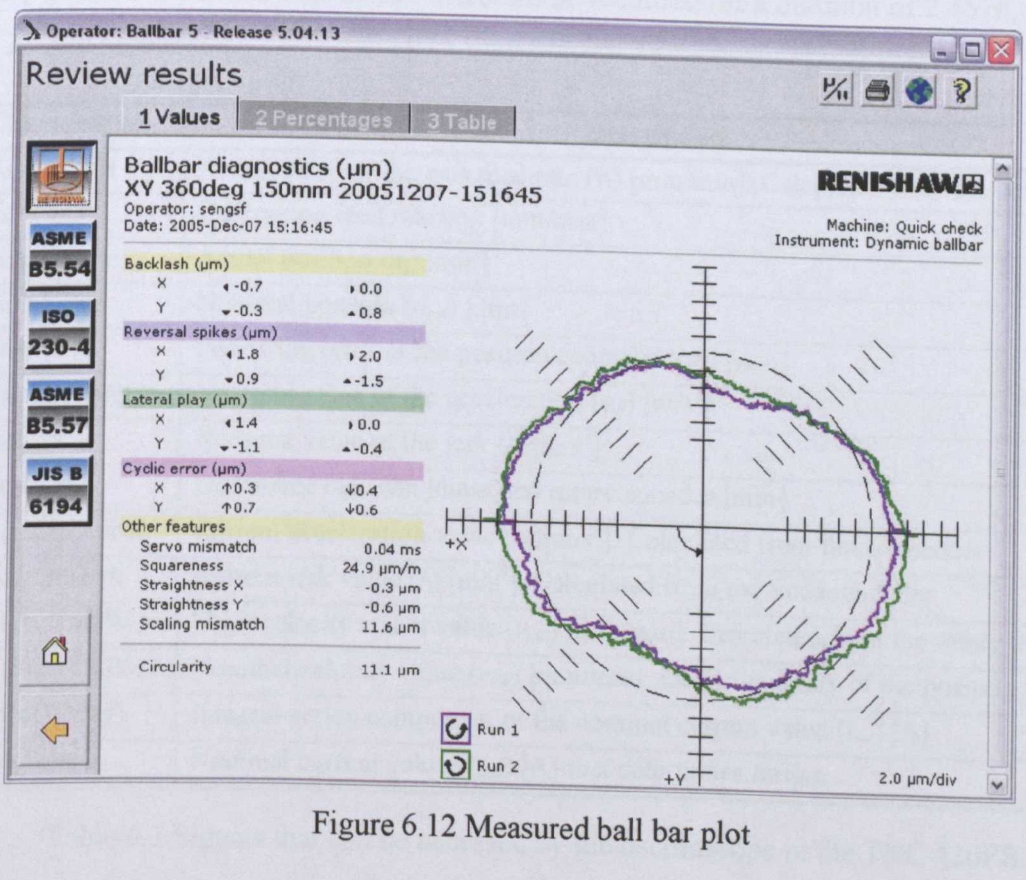


Figure 6.12 Measured ball bar plot

6.3 Step and Jerk Limited Response Measurements

This section contains the set of measurements undertaken on the single-axis test rig and on the x and y-axis of the Arrow 500 CNC machine Tool.

6.3.1 Single- Axis Test Rig

The motion controller for the test rig (Heidenhain TNC 426PB) features an integrated oscilloscope, which is used for monitoring and commissioning the control loops [114]. The integrated oscilloscope can record the characteristics contained in Table 6.3 in up to four channels. Three parameters are specified for a measurement:

- Output – To select whether the nominal speed value is to be issued as a step or ramp. The programmed feed rate, the position controller gain, and acceleration values specified with the machine parameters go into effect when ramp output is selected. If step output is selected, a step will be output as nominal velocity value when the axis direction buttons in the manual-operating mode are pressured (the position control loop is opened during this output).
- Feedrate – to specify the height of the step for the nominal velocity value (in mm/min). This parameter has no effect for ramp output.
- Sample time – To set the time interval for recording the signals: 0.6, 3 or 6 ms. 4096 samples are stored. The signals can therefore be recorded for a duration of 2.4576, 12.288 or 24.576 seconds.

Characteristic	Description
Actl. speed	Actual value of the axis feed rate (v_l) [mm/min]. Calculated from the position
Feed rate	Contouring feed rate (v_{ff}) [mm/min]
Actual pos	Actual position (d_l) [mm]
Noml. pos	Nominal position (d_{ref}) [mm]
Lag	Following error of the position controller (d_e) [mm]
Acceleration	Nominal value of the acceleration (a_{ff}) [m/s^2]
Jerk	Nominal value of the jerk (j) [m/s^3]
Pos. Diff.	Difference between linear and rotary encoder [mm]
Current Accel	Current acceleration value (a_l)[m/s^2]. Calculated from linear encoder
Current Jerk	Current jerk value (j_l) [m/s^3]. Calculated from the linear encoder
V(ACT RPM)	Shaft velocity actual value (v_{act}) [mm/min]. Calculated from the rotary encoder
V(NOM RPM)	Nominal velocity value (v_{ref}) [mm/min]. Output quantity of the position
I(INT RPM)	Integral-action component of the nominal current value (i_{int}) [A]
I-nominal	Nominal current value (i_{qref}) [A] that determines torque

Table 6.3 Signals that can be accessed by the oscilloscope in the TNC 426PB

The following procedure was established for the measurement of the signals needed to validate the model for this axis drive:

- A set of movements for various feed rates and displacements was defined as described in Table 6.4. The magnitude of the displacements was chosen so as to describe a jerk-limited movement within the recording duration for the highest sample rate (2.4576 s).

Displacement [mm]	10	20	100	200	400	400
Feed rate [mm/min]	500	1000	5000	10000	20000	40000

Table 6.4 Set of movements established for the validation of the test rig TLM model

- To record the signals needed for the validation of the step response of the velocity loop . This action was carried out by recording the *V(NOM RPM)*, *V(ACT RPM)* and *I-nominal* signals for each one of the feed rates specified in Table 6.4. An example of the measurements is shown in Figure 6.13. Appendix G.4 (Figures G.10 – G.14) contains the measurements for the other feed rates. The set of parameters selected in the oscilloscope were:
  - Output: step (step response and position control loop open).
  - Feed rate: each one of the values in Table 6.4.
  - Sample time: 0.6 ms (the velocity control loop cycle time).

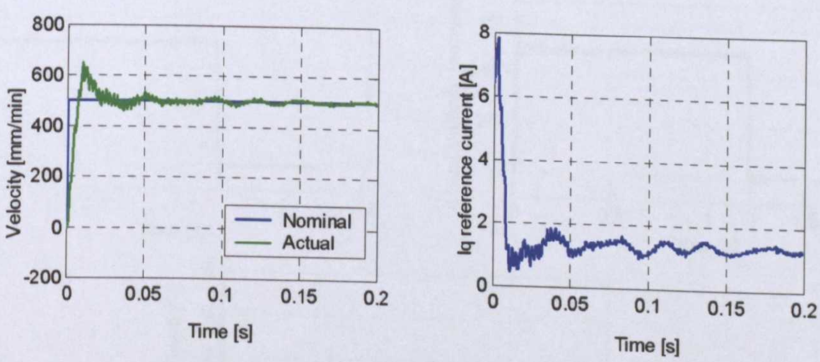


Figure 6.13 Test rig velocity step response measurements (500 mm/min)

- To record the signals needed for the validation of the jerk-limited response (ramp output in the oscilloscope) of the position and velocity loops. This action was undertaken by:
  - Recording the *Noml. pos*, *Feed rate* and *Acceleration* signals for a sample time of 3ms (position control loop cycle time). These signals are used, respectively, as the position reference (d_{ref}) , velocity feed forward (v_{ff}) and acceleration feed forward (a_{ff}) input signals for the axis model. Figure 6.14 shows the set of signals recorded for a displacement of 10 mm at 500 mm/min. Appendix G.5 (Figures G.15 –G.19) contains the measurements taken for the remaining feed rates.

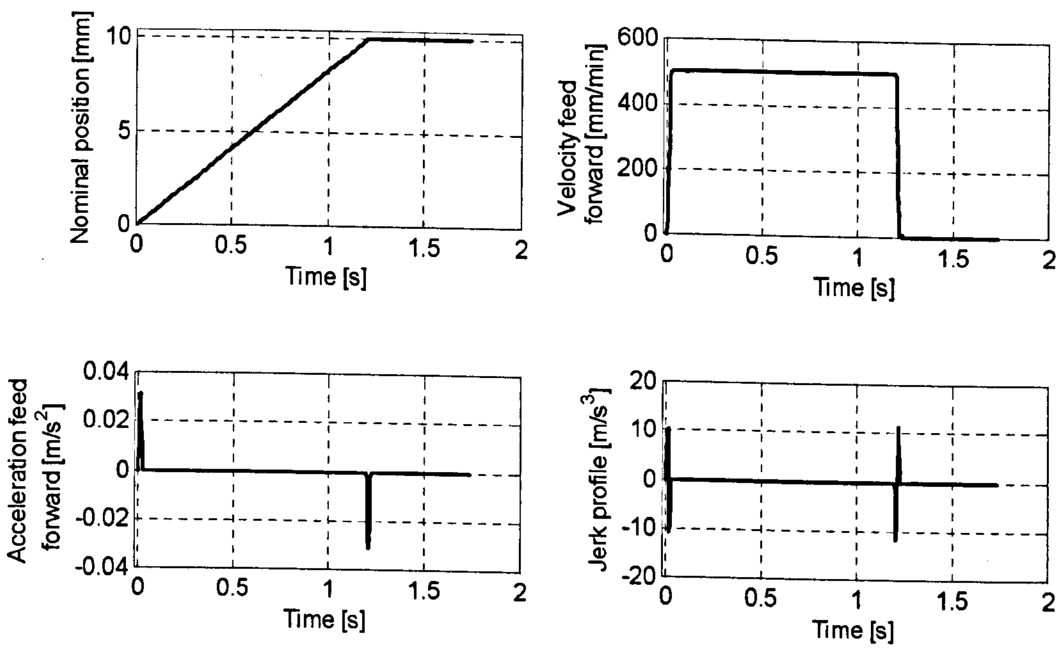


Figure 6.14 Jerk-limited profiles (displacement = 10mm, feedrate =500 mm/min)

- Recording the *V(NOM RPM)*, *V(ACT RPM)* and *I-nominal* signals to validate the velocity control loop model (Sample time = 0.6 ms). Figure 6.15 shows the set of signals recorded for a feed rate of 500 mm/min. Appendix G.6 (Figures G.20 – G.24) contains the measurements taken for the remaining feed rates.

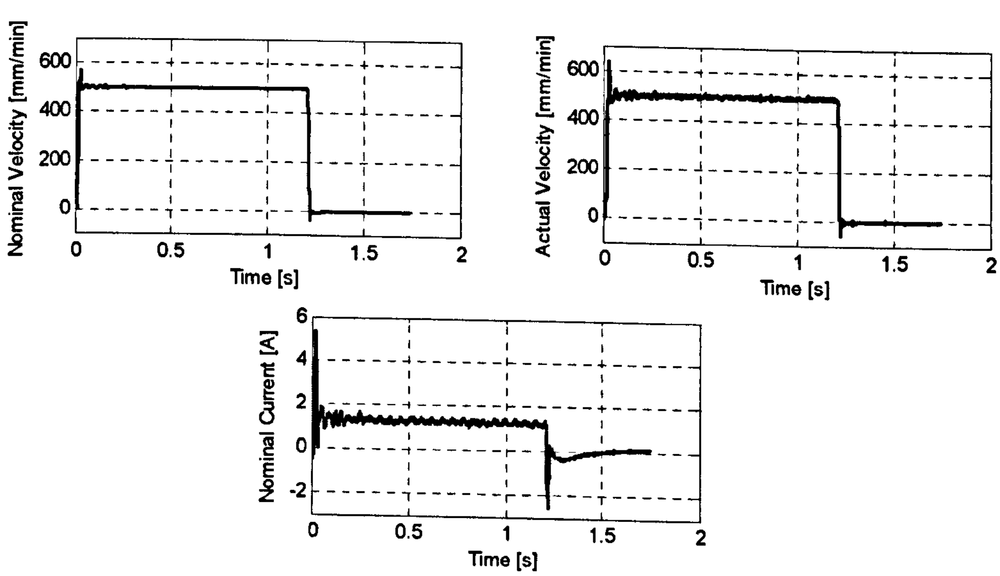


Figure 6.15 Jerk-limited axis response (displacement = 10mm, feed rate =500 mm/min)

- To record the *Actual pos*, *Lag* and *Pos. Diff* signals to validate the position control loop model. (Sample time = 3ms). Figure 6.16 shows the set of signals recorded for a displacement of 10 mm at 500 mm/min. Appendix G.7 (Figures G.25 – G.29) contains the measurements taken for the remaining feed rates.

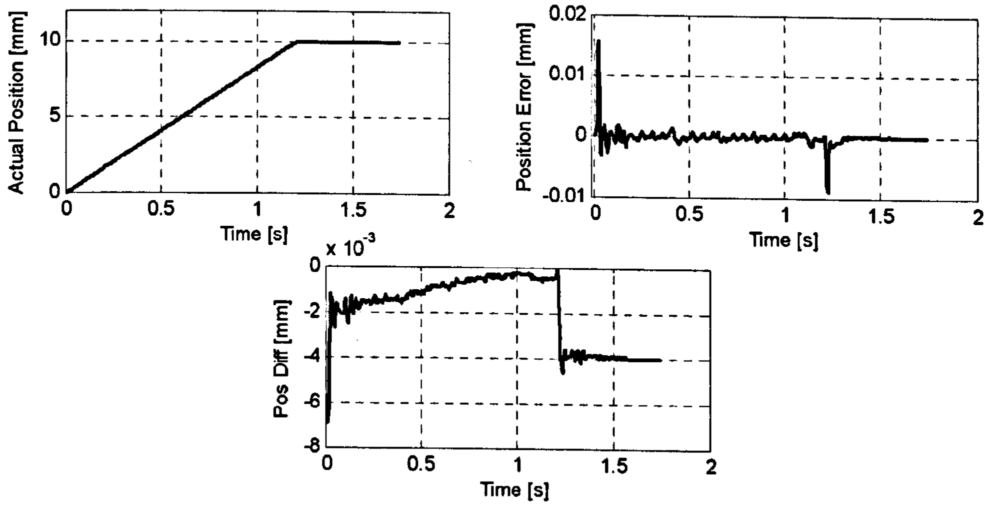


Figure 6.16 Position control loop signals (displacement = 10mm, feed rate =500 mm/min)

6.3.2 Arrow 500 Cincinnati CNC Machine

The SINUMERIK 840D SIEMENS motion controller features a servo-trace interface, which allows the time and/or frequency response of drives and closed-loop controls both to be recorded in the hard drive or to be displayed in graphic form on the screen. This interface is also used to set and activate the three digital analog converter (DAC) channels available on the SINUMERIK 810D and at each 611D closed-loop control module [115]. Some of the signals that can be monitored by the servo-trace are listed in tables 6.5 and 6.6.

Designation	**Unit**
Velocity set-point (v_{ref})	rpm
Velocity actual value (motor) (v_{act})	rpm
Absolute Velocity actual value (v_l)	rpm
Torque set-point (limited) T_{ref}	N-m
Torque set-point (Velocity controller output) T_{ref}	N-m
Current set-point i_q (limited after the filter)	A
Current set-point i_q (before the filter)	A
Absolute current actual value (i_q)	A

Table 6.5 Signals that can be switched to the DAC channels

Designation	**Unit**
Velocity feed forward set-point (v_{ff})	rpm
Set-point position (d_{ref})	mm
Actual position (d_{act})	mm
Following error (d_e)	mm

Table 6.6 Position loop signals

The servo-trace can record these signals in up to four channels and the sample time is calculated automatically according to the length of the selected recording time. The procedure described in the preceding section was used as the basis for the measurement of the signals for the x-axis and y-axis validation. Only data for the jerk-limited response was measured, as the servo-trace can not force the drives to respond to a variable step demand in time domain. This action was undertaken by:

- Recording the *Actual position*, *Velocity set-point*, *Velocity actual value* (motor), and *Torque set-point* (limited after the filter) signals to validate the velocity control loop model (Sample time = 125 μs). Figure 6.17 shows the set of signals recorded for an x-axis displacement of 10 mm at 500 mm/min. Appendix G.8 (Figures G.30 – G.34) contains the measurements taken for the remaining feed rates. Measurements for the y-axis are presented in Appendix G.9 (Figures G.35 – G.40).

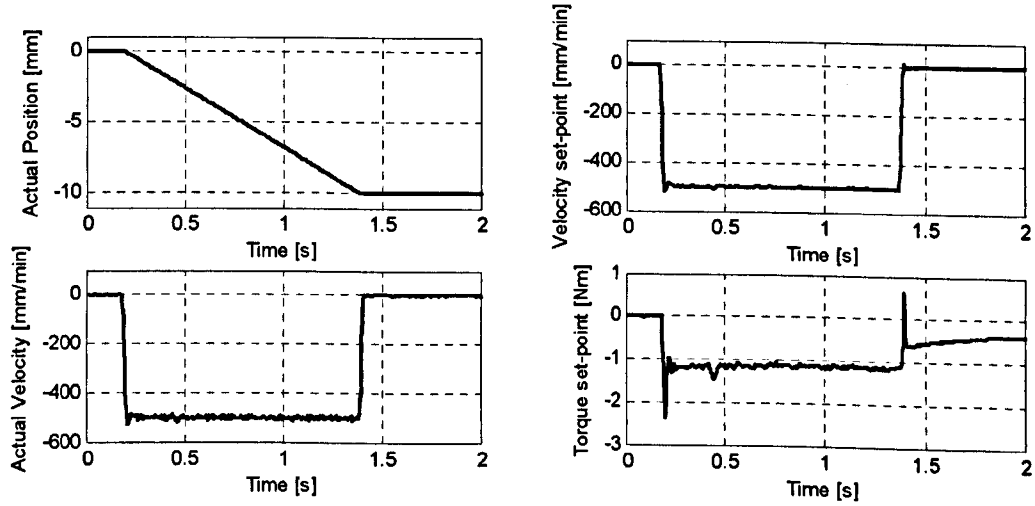


Figure 6.17 Jerk-limited velocity response (x-axis Arrow 500)

- Recording the *Set-point position*, *Actual position*, *Following error* and *Velocity feed forward set-point* signals to validate the position control loop model. (time-step = 4ms).

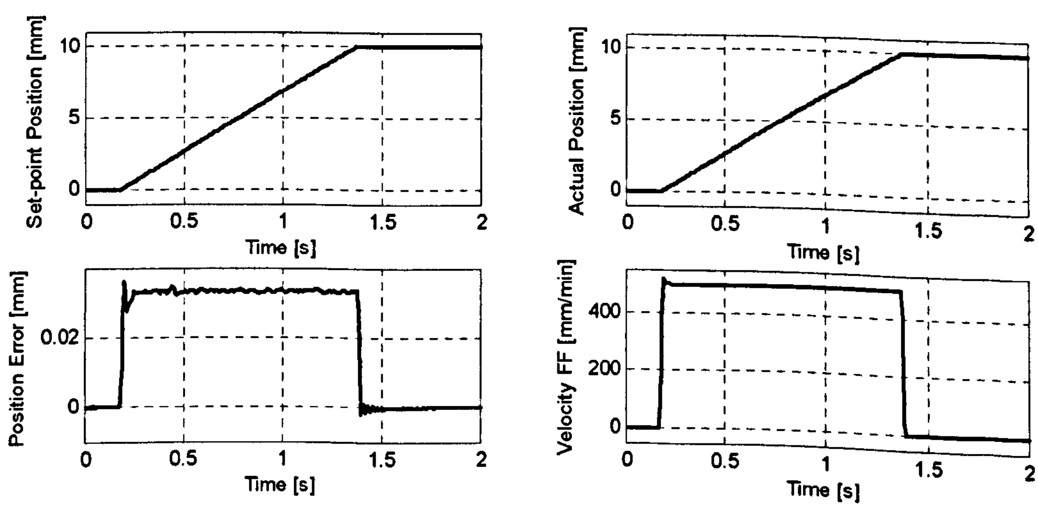


Figure 6.18 Jerk-limited position response (y-axis Arrow 500)

Figure 6.18 shows the set of signals recorded for a y-axis displacement of 10 mm at 500 mm/min. Appendix G.10 (Figures G.41 – G.46) contains the measurements taken for the x-axis. Measurements for the y-axis are presented in Appendix G.11 (Figures G.46 – G.51).

- Recording the *Set-point position*, *Velocity feed forward* and *Actual position* signals for a ballbar test (radius: 150 mm, feed rate: 1000 mm/min). Figures 6.19 and 6.20 show the set of signals recorded (run1: counter clockwise, run2: clockwise).

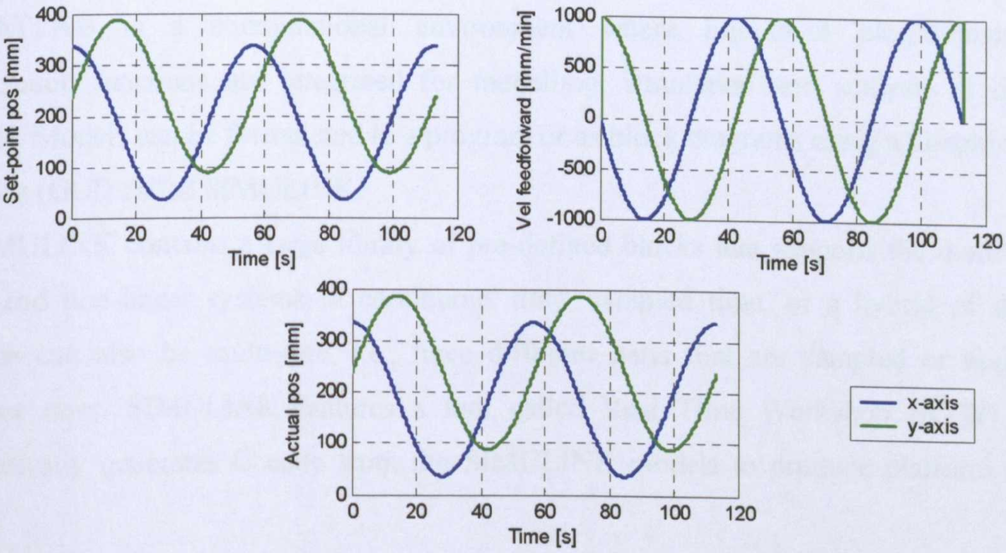


Figure 6.19 Signals measured for the ball-bar test (run1)

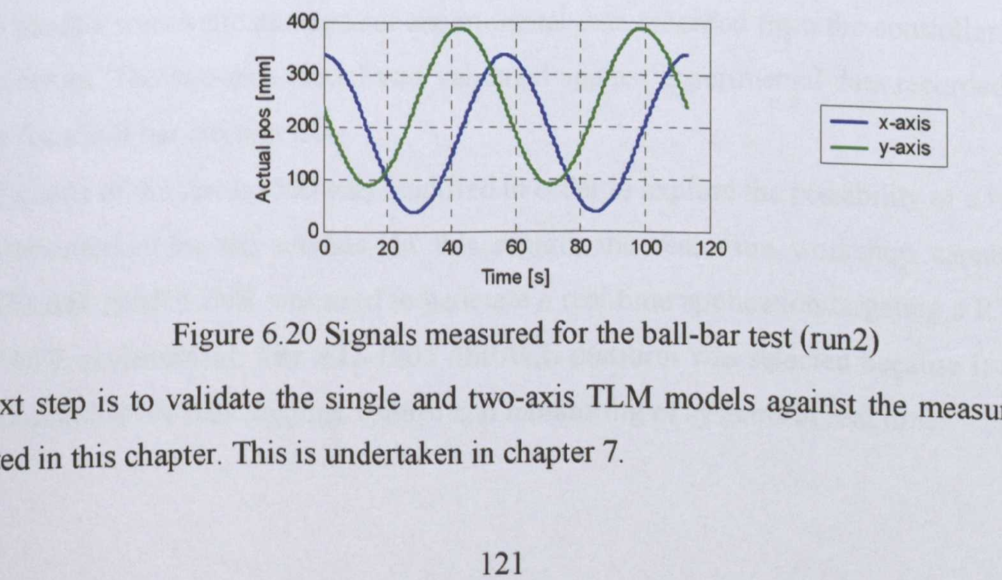


Figure 6.20 Signals measured for the ball-bar test (run2)

The next step is to validate the single and two-axis TLM models against the measurements presented in this chapter. This is undertaken in chapter 7.

7. SIMULATION OF PROPOSED ONE-AXIS AND TWO-AXIS TLM MODELS

One of the aspects that lead to the study of the TLM method for the modelling of feed drives was the possibility of formulation of comprehensive models, which could be implemented in real time. This chapter presents the implementation in MATLAB of the feed drive models derived in chapter 5 and discusses the model parameters to be taken into account for a real time version of the models. Specific attention has been devoted to the x-axis of the Arrow 500 CNC machine tool.

MATLAB is a computational environment where high-level programming and visualisation functions are integrated for modelling, simulation and analysis of dynamic systems. Models can be formulated in a program or as block diagrams using a Graphical User Interface (GUI) called SIMULINK.

SIMULINK contains a large library of pre-defined blocks that supports the modelling of linear and non-linear systems in continuous time, sampled time, or a hybrid of the two. Systems can also be multi-rate, i.e., have different parts that are sampled or updated at different rates. SIMULINK features a tool called Real Time Workshop (RTW), which automatically generates C code from the SIMULINK models to produce platform-specific code.

The simulation and validation of the models were performed according to the following methodology:

- The model for the single-axis test rig was built in SIMULINK in order to validate the modelling approach. The validation of the model was achieved by comparing simulated results with experimental data recorded from the controller.
- Following this step, the single and two-axis models for the Arrow 500 machine tool were implemented in SIMULINK taking as a basis the TLM model for the test rig. The x and y-axis models were validated against experimental data recorded from the controller and the axis drives. The two-axis model was validated against experimental data recorded in real time for a ball bar circular test.
- The x-axis of the Arrow 500 was modified in order to explore the possibility of a real time implementation for the models. In this regard, the real time workshop capability of MATLAB/ SIMULINK was used to generate a real time application targeting a RTI-1005 dSPACE environment. The RTI-1005 dSPACE platform was selected because it features the possibility for data logging, control and monitoring of systems in real time.

7.1 Implementation of the Single-Axis Model for the Test Rig in SIMULINK

The following considerations were taken into account for the implementation in SIMULINK of the test rig model presented in section 5.1 (pp 90-106):

- The position controller generates a reference velocity value v_{ref} at a rate $t_p = 3$ ms.
- The velocity controller generates a reference current value i_{qref} at a rate $t_v = 0.6$ ms.
- The current controller gives a reference voltage value e_{dqref} to the PWM generator at a rate $t_c = 0.2$ ms.
- Each PWM signal is composed of seven e_{dq} voltages (switching states) calculated according to the expected currents to be induced in the motor. The duration (t_dc) of each e_{dq} voltage is specified in multiples of the propagation time for the torsional model (t_t). To accomplish this, t_t is made equal to the sampling period on the PWM signal (t_{pwm}), then:

$$t_t = t_{pwm} = t_c / R_{pwm} \tag{7.1}$$

- The propagation time on the axial model is a sub-multiple of the torsional propagation time as defined by the method proposed in Appendix C (synchronisation of the torsional and axial models).

These actions represent five multi-rate subsystems, which are implemented in software by the block diagram illustrated in Figure 7.1.

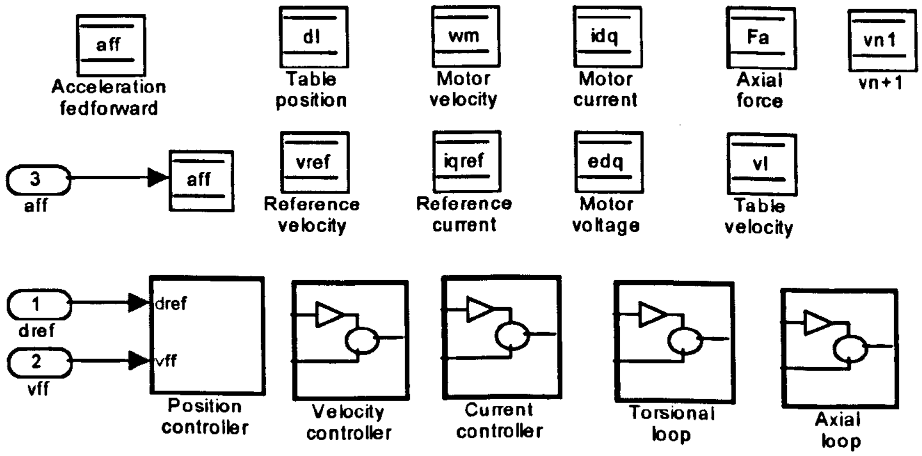


Figure 7.1 Block diagram for the test rig single axis model

Variables interfacing the multi-rate subsystems (*aff*, *dl*, *vref*, *wm*, *iqref*, *idq*, *edq*, *Fa*, *vl*, *vn*+1) are implemented in Data Stored Memory blocks (DSM). A DSM defines a memory region for use by the *data store read* and *data store write* blocks. This feature gives access of the memory region to the different sub-systems in order to read from or write to a designed variable at predetermined sample rates.

As discussed in section 5.1.1, the interpolator generates the reference position (*dref*), the velocity feed forward (*vff*) and the acceleration feed forward (*aff*) signals at the sample rate t_p. These signals are applied to the model in the form of variables proceeding from the MATLAB workspace as shown in Figure 7.2.

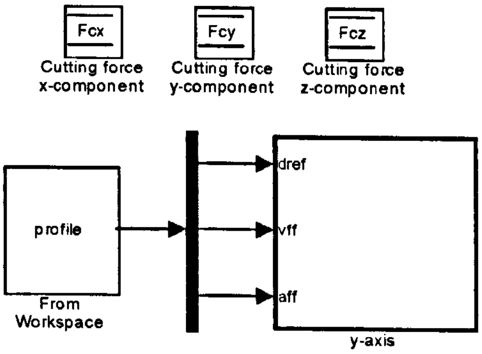


Figure 7.2 Implementation in SIMULINK of the test rig single axis model

The block *profile* defines the name and sample rate of the workspace variable containing the *dref*, *vff* and *aff* variables. The variable *profile* is composed by the program *testrig_profile.m* included in Appendix H.1. This program can either read a file containing the experimental values measured from the test rig controller or call one of five different routines built to generate the reference position signal. See Appendix H.2 for the jerk-limited profile, step profile, sinusoidal profile, white noise profile and swept sine profile. Two-axis linear and circular interpolations have been also included.

The three components of the cutting force (*Fcx*, *Fcy* and *Fcz*) have been included in the model, although the analysis of cutting forces is out of the scope of this study.

The block *y-axis* contains the block diagram illustrated in figure 7.1. The block parameters and initialisation code are included in Appendix H.3.

7.1.1 Digital Controller Model

The subsystem *Position controller* has been built on the basis of equations (4.57 – 4.59) and its constituent elements are shown in Figure 7.3. See Appendix H.4 for the block parameters and the initialisation code.

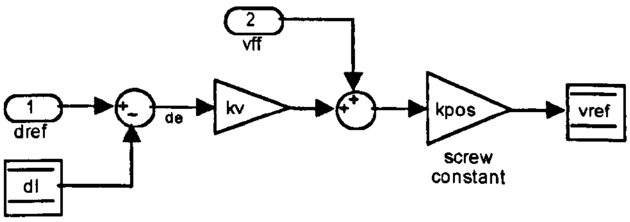


Figure 7.3 Block diagram of the position controller model in SIMULINK

The velocity controller model (equations 4.55 –4.61) has been implemented in the subsystem *Velocity controller*. Figure 7.4 shows the block constituent elements.

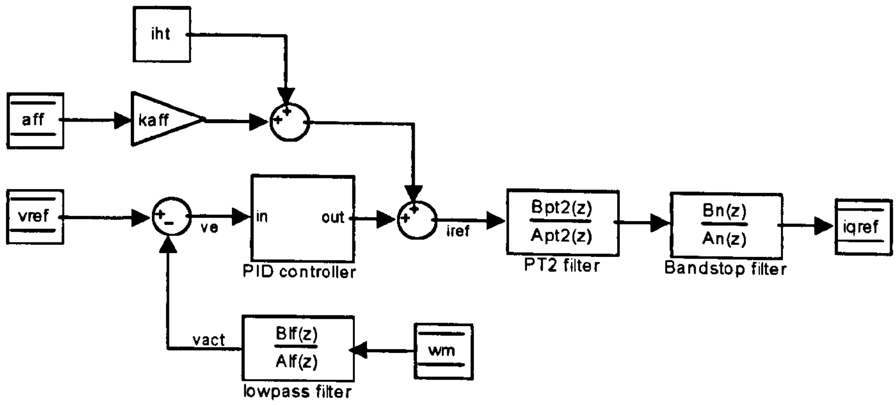


Figure 7.4 Block diagram of the velocity controller model in SIMULINK

The *PID controller* block implements the control strategy in terms of the TLM transform, as illustrated in Figure 7.5.

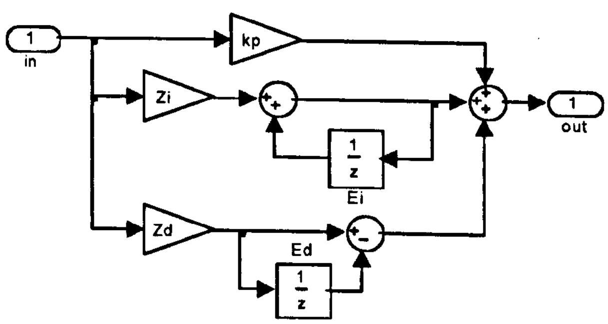


Figure 7.5 Block diagram of the PID controller model in SIMULINK

The subsystem *Current controller* has been built on the basis of equations (5.9 – 5.12) and its constituent elements are shown in Figure 7.6.

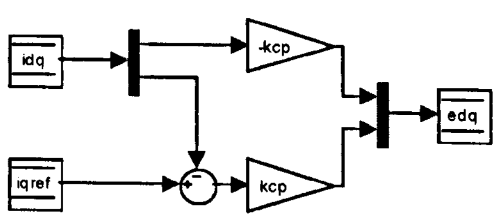


Figure 7.6 Block diagram of the current controller model in SIMULINK

7.1.2 Dynamic Model of the Ball-Screw System

As presented in Appendix D, the torsional and axial models for the screw shaft are reduced to the calculation of velocities and the incident pulses affected by the perturbations and the propagation of pulses on the other sections. Accordingly, the implementation of the model for the ball screw system has been structured into two subsystems (see Figure 7.1):

- Torsional loop.
- Axial loop.

These subsystems are the implementation of the TLM models described in section 5.1.3. Special attention has been taken on the structure of data for the simulation of the pulse propagation in the axial and torsional models. This topic is treated first, as it defines the central data structure of the ball-screw system model.

7.1.2.1 Implementation of the Pulse Propagation

The pulse propagation for the torsional model is given by equations (D.8 – D.28) in Appendix D. As presented in section 5.1.3.4, the propagation of pulses on each zone resembles a circular linked list where the pulses magnitude is stored and modified at defined positions (The first, f_b and h_t sections).

column	*1*	*2*	*3*		nf-2	nf-1	nf	nf+1	nf+2	nf+3		2*nf-1	2*nf
pulse	A_1	A_2	A_3		A_{fb-2}	A_{fb-1}	A_{fb}	B_{fb}	B_{fb-1}	B_{fb-2}		B_2	B_1
next	2	3	4		nf-1	nf	nf+1	nf+2	nf+3	nf+4		2*nf	1
previous	2*nf	1	2		nf-3	nf-2	nf-1	nf	nf+1	nf+2		2*nf-2	2*nf-1
	↑ pA_1							↑ pB_{fb}					

Figure 7.7 Array used to simulate the first zone of the torsional model

The circular list for the first zone is implemented on a 3 x n_f matrix (Figure 7.7) where:

The number of sections on the list (n_f) is equal to

$$n_f = f_b \tag{7.2}$$

- The first row holds the magnitude of the pulses.
- The second row holds the position (column number) of the next pulse on the array.
- The third row holds the position of the previous pulse on the array.

Two pointers are included to register the position of the pulses needed for the calculation of the angular velocities of interest (ω_1, ω_{fb+1}): pA_1 registers the position of the pulse A^i_1, and pB_{fb} registers the position of the pulse B^i_{fb}. A particular element in the matrix is referenced by specifying its row and column number using the syntax: $listF(row,column)$, where $listF$ is the matrix variable. The magnitude of the pulse A^i_1 is held in the matrix element $listF(1,1)$, the matrix element $listF(2,1)$ holds the number of the column where the magnitude of the A^i_2 pulse is held, and the matrix element $listF(3,1)$ hold the number of the column where the magnitude of the $B^i{}_1$ pulse is held. Then,

$$listF(1,1) = A_1^i \tag{7.3}$$

$$listF(2,1) = 2 \tag{7.4}$$

$$listF(3,1) = 2n_f \tag{7.5}$$

$$pA_1 = 1 \tag{7.6}$$

$$pB_{fb} = n_f + 1 \tag{7.7}$$

Moving the pointers to the next position on the array simulates the pulses propagation. Then pA_1 and pB_{fb} become:

$$pA_1 = listF(2,1) = 2 \tag{7.8}$$

$$pB_{fb} = listF(2, n_f + 1) = n_f + 2 \tag{7.9}$$

The new arrangement of pulses is illustrated in Figure 7.8. Note that the execution of equations (D.15) and (D.16) in Appendix D is replaced by using this approach. Thus, a significant reduction of computing time is achieved.

column	*1*	*2*	*3*		*nf*-2	*nf*-1	*nf*	*nf*+	*nf*+	*nf*+		2*nf*-1	2*nf*
	A_1	A_2	A_3	...	A_{fb-2}	A_{fb-1}	A_{fb}	B_{fb}	B_{fb-1}	B_{fb-2}	...	B_2	B_1
pulse	$\mathbf{B_1}$	$\mathbf{A_1}$	$\mathbf{A_2}$		$\mathbf{A_{fb-3}}$	$\mathbf{A_{fb-2}}$	$\mathbf{A_{fb-1}}$	$\mathbf{A_{fb}}$	$\mathbf{B_{fb}}$	$\mathbf{B_{fb-1}}$		$\mathbf{B_3}$	$\mathbf{B_2}$
	2	3	4		*nf*-1	*nf*	*nf*+	*nf*+	*nf*+	*nf*+		2*nf*	1
	2nf	1	2		*nf*-3	*nf*-2	*nf*-1	*nf*	*nf*+	*nf*+		2*nf*-2	2*nf*-1
		↑ pA_1							↑ pB_{fb}				

Figure 7.8 First zone array after a pulse propagation (torsional model)

If the angular velocities ω_1 and ω_{fb+1} are known, the pulses propagation can be simulated by the following procedure:

- Calculation of the value for the next $B^i{}_1$ pulse ($B^i{}_1(k+1)$) according to equation (D.13) in Appendix D: pA_1 is used as a reference to $A^i{}_1$ and $B^i{}_1(k+1)$ on the list due to the fact that $B^i{}_1$ will take the position of $A^i{}_1$ after the pointers are move.

$$listF(1, pA_1) = \omega_1 Z_t + listF(1, pA_1) \tag{7.10}$$

- Calculation of the value for the next $A^i{}_{fb}$ pulse ($A^i{}_{fb}(k+1)$) according to equation (d.14): pB_{fb} is used as a reference to $B^i{}_{fb}$ and $A^i{}_{fb}(k+1)$ on the list due to the fact that A^i_{fb} will take the position of $B^i{}_{fb}$ after the pointers are move.

$$listF(1, pB_{fb}) = listF(1, pB_{fb}) - \omega_{fb+1} Z_t \tag{7.11}$$

- Move the pointers to the next position.

$$pA_1 = listF(2, pA_1) \tag{7.12}$$

$$pB_{fb} = listF(2, pB_{fb}) \tag{7.13}$$

column	*1*	2	3		*nm*-2	*nm*-1	*nm*	*nm*+1	*nm*+2	*nm*+3		2*nm*-1	2*nm*
pulse	A_{fb+1}	A_{fb+2}	A_{fb+3}	...	A_{rb-2}	A_{rb-1}	A_{rb}	B_{rb}	B_{rb-1}	B_{rb-2}	...	B_{fb+2}	B_{fb+1}
next	2	3	4		*nm*-1	*nm*	*nm*+1	*nm*+2	*nm*+3	*nm*+4		2*nm*	1
previous	2*nm*	1	2		*nm*-3	*nm*-2	*nm*-1	*nm*	*nm*+1	*nm*+2		2*nm*-2	2*nm*-1
	↑ pA_{fb1}							↑ pB_{rb}					

Figure 7.9 Array used to simulate the second zone of the torsional model

The implementation of the circular list for zone two gives:

- The circular list is implemented on a 3x n_m matrix called *listM* (Figure 7.9). The number of sections on the list (n_m) is equal to

$$n_m = h_t - f_b \tag{7.14}$$

- pA_{fb1} and pB_{rb} register the position of the pulses A^i_{fb+1} and B^i_{rb} respectively:

$$pA_{fb1} = 1 \tag{7.15}$$

$$pB_{rb} = n_m + 1 \tag{7.16}$$

If the angular velocities ω_{fb+1} and ω_{rb+1} are known, the pulses propagation is simulated by the following equations (equations D.18 – D.21):

$$listM(1, pA_{fb1}) = \omega_{fb+1} Z_t + listM(1, pA_{fb1}) \tag{7.17}$$

$$listM(1, pB_{rb}) = listM(1, pB_{rb}) - \omega_{rb+1} Z_t \tag{7.18}$$

$$pA_{fb1} = listM(2, pA_{fb1}) \tag{7.19}$$

$$pB_{rb} = listM(2, pB_{rb}) \tag{7.20}$$

The inclusion of the nut in the model will cause the reflection of pulses arriving to section *n*, and therefore splitting the *listM* in two as shown in Figure 7.10a. The following variables are added in order to complete the model for the moving nut:

- n_l: The number of sections in the left loop on the Figure 7.10b

$$n_l = n - f_b \tag{7.21}$$

- n_n: The position of the pulse B^i_{n+1} on *listM*

$$n_n = 2h_t - n_l \tag{7.22}$$

- pA_{n1} and pB_n register the position of pulses A^i_{n+1} and B^i_n respectively:

$$pA_{n1} = n_l + 1 \tag{7.23}$$

$$pB_n = n_n + 1 \tag{7.24}$$

- Pulse A^i_n is connected with pulse B^i_n:

$$pA_n = n_l \tag{7.25}$$

$$listM(2, pA_n) = pB_n \tag{7.26}$$

$$listM(3, pB_n) = pA_n \tag{7.27}$$

- Pulse B^i_{n+1} is connected with pulse A^i_{n+1}:

$$pB_{n1} = n_n \tag{7.28}$$

$$listM(2, pB_{n1}) = pA_{n1} \tag{7.29}$$

$$listM(3, pA_{n1}) = pB_{n1} \tag{7.30}$$

If the angular velocity ω_{n+1} is known, the pulse propagations on *section n* (equations D22 to D.26 in Appendix D) is simulated by the following equations:

$$listM(1, pA_{n1}) = \omega_{n+1}Z_t + listM(1, pA_{n1}) \tag{7.31}$$

$$listM(1, pB_n) = listM(1, pB_n) - \omega_{n+1}Z_t \tag{7.32}$$

$$pA_{n1} = listM(2, pA_{n1}) \tag{7.33}$$

$$pB_n = listM(2, pB_n) \tag{7.34}$$

Figure 7.11 shows the status of the matrix *listM* after two pulse propagations. The number of sections on the two loops in zone two changes when the nut moves to an adjacent section. Figure 7.12 shows the changes on the two loops when the nut moves to the next section on the right (from section n to section $n+1$). In this case, the connections of the pulses A^i_n, A^i_{n+1}, A^i_{n+2}, B^i_n, B^i_{n+1} and B^i_{n+2} change. The mapping of those changes on the matrix *listM* are carried out by the following procedure:

- The position of the pulses A^i_n, A^i_{n+2}, B^i_{n+1} and B^i_{n+2} is held in the variables pA_n, pA_{n2}, pB_{n1} and pB_{n2} respectively:

$$pA_n = listM(3, pB_n) \tag{7.35}$$

$$pA_{n2} = listM(2, pA_{n1}) \tag{7.36}$$

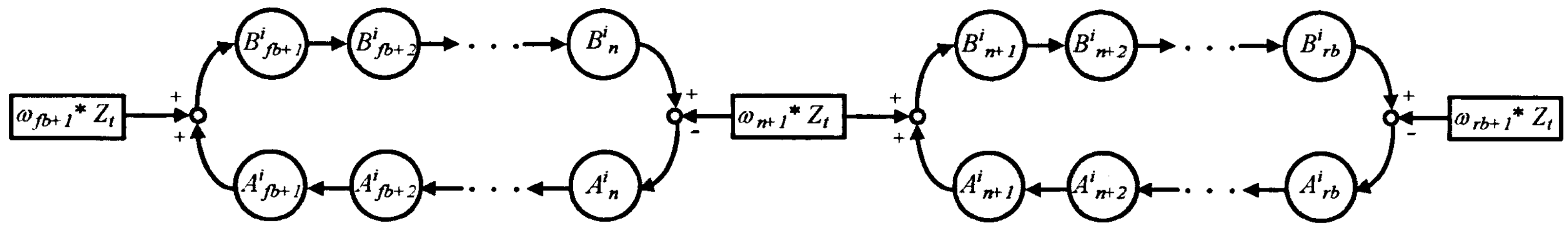
$$pB_{n1} = listM(3, pA_{n1}) \tag{7.37}$$

$$pB_{n2} = listM(3, pB_{n1}) \tag{7.38}$$

- Pulse A^i_n is connected with pulse A^i_{n+1}:

$$listM(2, pA_n) = pA_{n1} \tag{7.39}$$

$$listM(3, pA_{n1}) = pA_n \tag{7.40}$$



a) Circular list divided in two loops

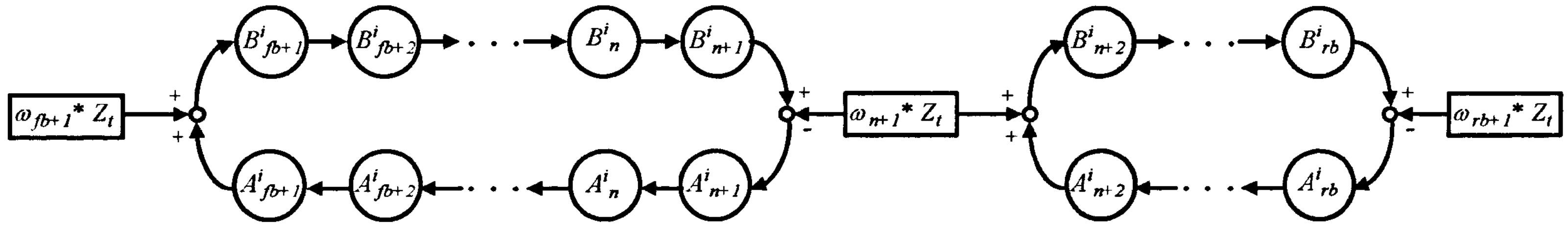
column	*1*	*2*	*3*		*nl*	*nl+1*	*nl+2*	*nl+3*	*nl+4*		*nm*	*nm+1*	*nm+2*	*nm+3*		*nn*	*nn+1*	*nn+2*	*nn+3*		*2*nm*
pulse	A_{fb+1}	A_{fb+2}	A_{fb+3}	...	A_n	A_{n+1}	A_{n+2}	A_{n+3}	A_{n+4}		A_{rb}	B_{rb}	B_{rb-1}	B_{rb-2}		B_{n+1}	B_n	B_{n-1}	B_{n-2}	..	B_{fb+1}
next	2	3	4		**nn+1**	nl+2	nl+3	nl+4	nl+5		nm+1	nm+2	nm+3	nm+4		**nl+1**	nn+2	nn+3	nn+4		1
previous	2*nm	1	2		nl-1	**nn**	nl+1	nl+2	nl+3		nm-1	nm	nm+1	nm+2		nn-1	**nl**	nn+1	nn+2		2*nm-1
	↑ pA_{fb1}					↑ pA_{nl}						↑ pB_{rb}					↑ pB_n				

b) Mapping of the two loops on the matrix *listM*

Figure 7.10 Second zone array including the moving nut (torsional model)

column	*1*	*2*	*3*		*nl*	*nl+1*	*nl+2*	*nl+3*	*nl+4*		*nm*	*nm+1*	*nm+2*	*nm+3*		*nn*	*nn+1*	*nn+2*	*nn+3*		*2*nm*
pulse	B_{fb+2}	B_{fb+1}	A_{fb+1}	...	A_{n-2}	B_{n+2}	B_{n+1}	A_{n+1}	A_{n+2}	.	A_{rb-2}	A_{rb-1}	A_{rb}	B_{rb}		B_{n+3}	A_{n-1}	A_n	B_n	..	B_{fb+3}
next	2	3	4		nn+1	nl+2	nl+3	nl+4	nl+5		nm+1	nm+2	nm+3	nm+4		nl+1	nn+2	nn+3	nn+4		1
previous	2*nm	1	2		nl-1	nn	nl+1	nl+2	nl+3		nm-1	nm	nm+1	nm+2		nn-1	nl	nn+1	nn+2		2*nm-1
			↑ pA_{fb1}					↑ pA_{nl}						↑ pB_{rb}					↑ pB_n		

Figure 7.11 Mapping of the two loops on the matrix *listM* after two pulse propagations



a) Circular list divided in two loops

column	1	2	3		nl	nl+1	nl+2	nl+3	nl+4		nm	nm+1	nm+2	nm+3		nn	nn+1	nn+2	nn+3		2*nm
pulse	B_{fb+2}	B_{fb+1}	A_{fb+1}	...	A_{n-2}	B_{n+2}	B_{n+1}	A_{n+1}	A_{n+2}		A_{rb-2}	A_{rb-1}	A_{rb}	B_{rb}		B_{n+3}	A_{n-1}	A_n	B_n	..	B_{fb+3}
next	2	3	4		nn+1	**nl+4**	**nn+3**	**nl+2**	nl+5		nm+1	nm+2	nm+3	nm+4		nl+1	nn+2	**nl+3**	nn+4		1
previous	2*nm	1	2		nl-1	nn	**nl+3**	**2*nn+2**	**nl+1**		nm-1	nm	nm+1	nm+2		2*nn-1	nl	nn+1	**nl+2**		2*nm-1
			↑ pA_{fb1}				↑ pB_n		↑ pA_{n1}					↑ pB_{rb}							

b) Mapping of the two loops on the matrix *listM*

Figure 7.12 Second zone array including the moving nut (nut moves to the right)

- Pulse A^i_{n+1} is connected with pulse B^i_{n+1}:

$$listM(2, pA_{n1}) = pB_{n1} \tag{7.41}$$

$$listM(3, pB_{n1}) = pA_{n1} \tag{7.42}$$

- Pulse B^i_{n+1} is connected with pulse B^i_n:

$$listM(2, pB_{n1}) = pB_n \tag{7.43}$$

$$listM(3, pB_n) = pB_{n1} \tag{7.44}$$

- Pulse B^i_{n+2} is connected with pulse A^i_{n+2}:

$$listM(2, pB_{n2}) = pA_{n2} \tag{7.45}$$

$$listM(3, pA_{n2}) = pB_{n2} \tag{7.46}$$

- Pointers pB_n and pA_{n1} are set to their new values:

$$pB_n = pB_{n1} \tag{7.47}$$

$$pA_{n1} = pA_{n2} \tag{7.48}$$

A similar procedure is applied when the nut moves to the next section on the left (from section n to section n-1). Pulses affected by this movement are: A^i_{n-1}, A^i_n, A^i_{n+1}, B^i_{n-1}, B^i_n and B^i_{n+1} (Figure 7.13). The mapping of the changes on the matrix *listM* are carried out in this case by the following procedure:

- The position of the pulses A^i_n, A^i_{n-1}, B^i_{n+1} and B^i_{n-1} is held in the variables pA_n, pA_{1n}, pB_{n1} and pB_{1n} respectively:

$$pA_n = listM(3, pB_n) \tag{7.49}$$

$$pA_{1n} = listM(3, pA_n) \tag{7.50}$$

$$pB_{n1} = listM(3, pA_{n1}) \tag{7.51}$$

$$pB_{1n} = listM(2, pB_n) \tag{7.52}$$

- Pulse A^i_n is connected with pulse A^i_{n+1}:

$$listM(2, pA_n) = pA_{n1} \tag{7.53}$$

$$listM(3, pA_{n1}) = pA_n \tag{7.54}$$

- Pulse A^i_{n-1} is connected with pulse B^i_{n-1}:

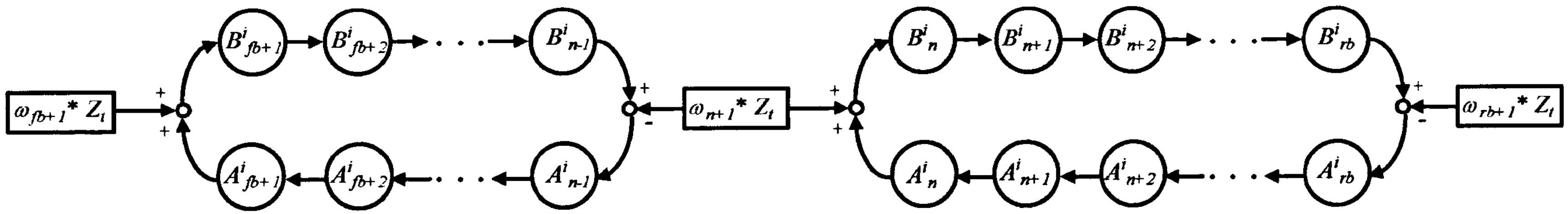
$$listM(2, pA_{1n}) = pB_{1n} \tag{7.55}$$

$$listM(3, pB_{1n}) = pA_{1n} \tag{7.56}$$

- Pulse B^i_{n+1} is connected with pulse B^i_n:

$$listM(2, pB_{n1}) = pB_n \tag{7.57}$$

$$listM(3, pB_n) = pB_{n1} \tag{7.58}$$



a) Circular list divided in two loops

column	1	2	3		nl	nl+1	nl+2	nl+3	nl+4		nm	nm+1	nm+2	nm+3		nn+1	nn+2	nn+3	nn+4		2*nm
pulse	B_{fb+2}	B_{fb+1}	A_{fb+1}	...	A_{n-2}	B_{n+2}	B_{n+1}	A_{n+1}	A_{n+2}		A_{rb-2}	A_{rb-1}	A_{rb}	B_{rb}		A_{n-1}	A_n	B_n	B_{n-1}	..	B_{fb+3}
next	2	3	4		nn+1	nl+2	**nn+3**	nl+4	nl+5		nm+1	nm+2	nm+3	nm+4		**nn+4**	**nl+3**	**nn+2**	2*nn+5		1
previous	2*nm	1	2		nl-1	nn	nl+1	**nn+2**	nl+3		nm-1	nm	nm+1	nm+2		nl	**nn+3**	**nl+2**	**nn+1**		2*nm-1
			↑ pA_{fb1}											↑ pB_{rb}			↑ pA_{n1}		↑ pB_n		

b) Mapping of the two loops on the matrix *listM*

Figure 7.13 Second zone array including the moving nut (nut moves to the left)

- Pulse B^i_n is connected with pulse A^i_n:

$$listM(2, pB_n) = pA_n \tag{7.59}$$

$$listM(3, pA_n) = pB_n \tag{7.60}$$

- Pointers pB_n and pA_{n1} are set to their new values:

$$pB_n = pB_{1n} \tag{7.61}$$

$$pA_{n1} = pA_n \tag{7.62}$$

Variables *difSec* and *lastSec* are included to verify if the nut has moved to an adjacent section and therefore decided which part of the code will be executed (nut is on the same section, nut has moved to the left or nut has moved to the right). The choice is taken according to the following procedure:

- Calculate the section where the nut is on

$$n = floor(l_n / l_{tor}) \tag{7.63}$$

- Calculate the difference between the new and the last section

$$difSec = n - lastSec \tag{7.64}$$

- Switch between the two cases based on the value for *difSec*

```
switch difSec
    case 1
        run code when the nut has moved to the right
    case -1
        run code when the nut has moved to the left
end
```

- Assign the value of n to *lastSec:*

$$lastSec = n \tag{7.65}$$

This approach is used to implement the code for the axial model as presented in Appendix I.

7.1.2.2 Torsional Loop Subsystem

This subsystem contains the models for the permanent magnet motor, the coupling and the screw shaft torsional dynamics. It comprises three blocks: PSM motor, Torsional model and the Nut position monitoring, as shown in Figure 7.14. The subsystem initialisation code is included in Appendix J.1

It must be noted that the variables defined in the last section to simulate the pulse propagation have been implemented in Data Stored Memory blocks (*listF*, pA_1, pB_{fb}, *listM*, pA_{fb+1}, pB_n, pA_{n+1}, pB_{ht}, *lastSec*, pB_{n-1}, pB_{n+1}, pB_{n+2}, pA_{n-1}, pA_n and pA_{n+2})

The block *PMS motor* contains the model established for the inverter and the motor in section 5.1.2.

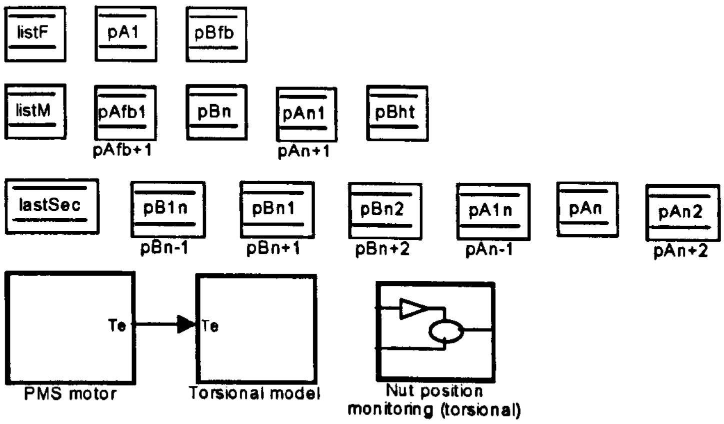


Figure 7.14 Block model for the subsystem Torsional loop

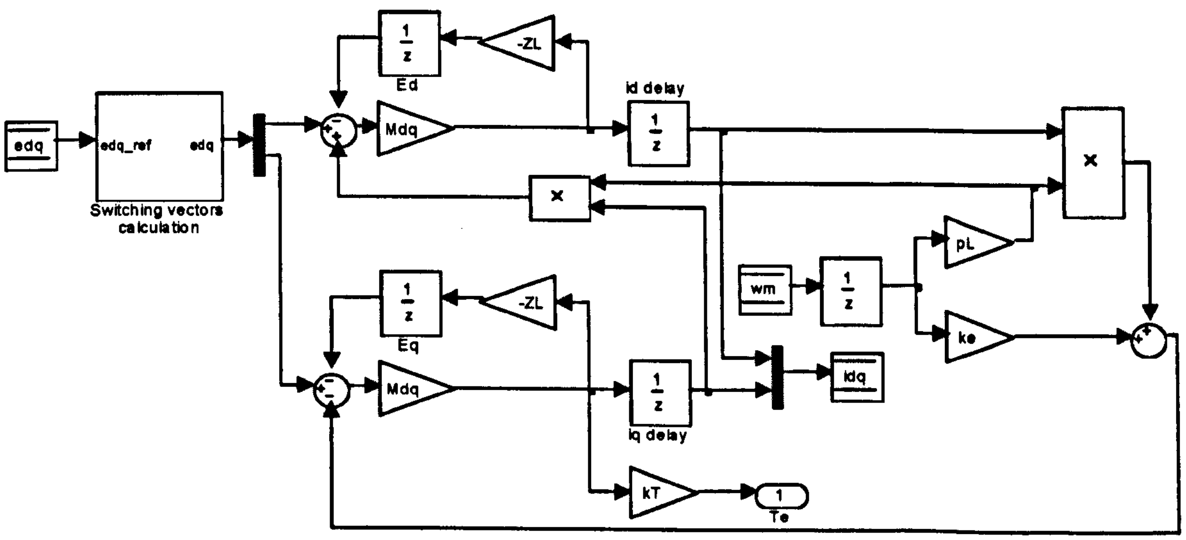


Figure 7.15 Block model for the inverter and motor in SIMULINK

The block *Switching vectors* in Figure 7.15 features the generation of the e_{dq} voltages to be applied to the motor according to the PWM strategy described in section 4.3.4. Appendix J.2 contains the code for the MATLAB function called by this block.

The *Torsional model* block contains the calculation of velocities ω_m, ω_{fb+1}, ω_{n+1} and ω_{ht+1} and the pulse propagation of torsional waves as described in section 5.1.3.1 and 5.1.3.2. See Figure 7.16.

The block *wm calculation*, comprises the motor mechanical model and the coupling - equations (5.19 – 5.26). Its constituent elements are shown in Figure 7.17.

The model for the calculation of the front bearing angular velocity (ω_{fb+1}) is implemented in the block *wfb+1 calculation*. The conditions derived from the bearing friction model, Equation (5.36), are modelled by *if condition* and *if action* blocks as shown in Figure 7.18.

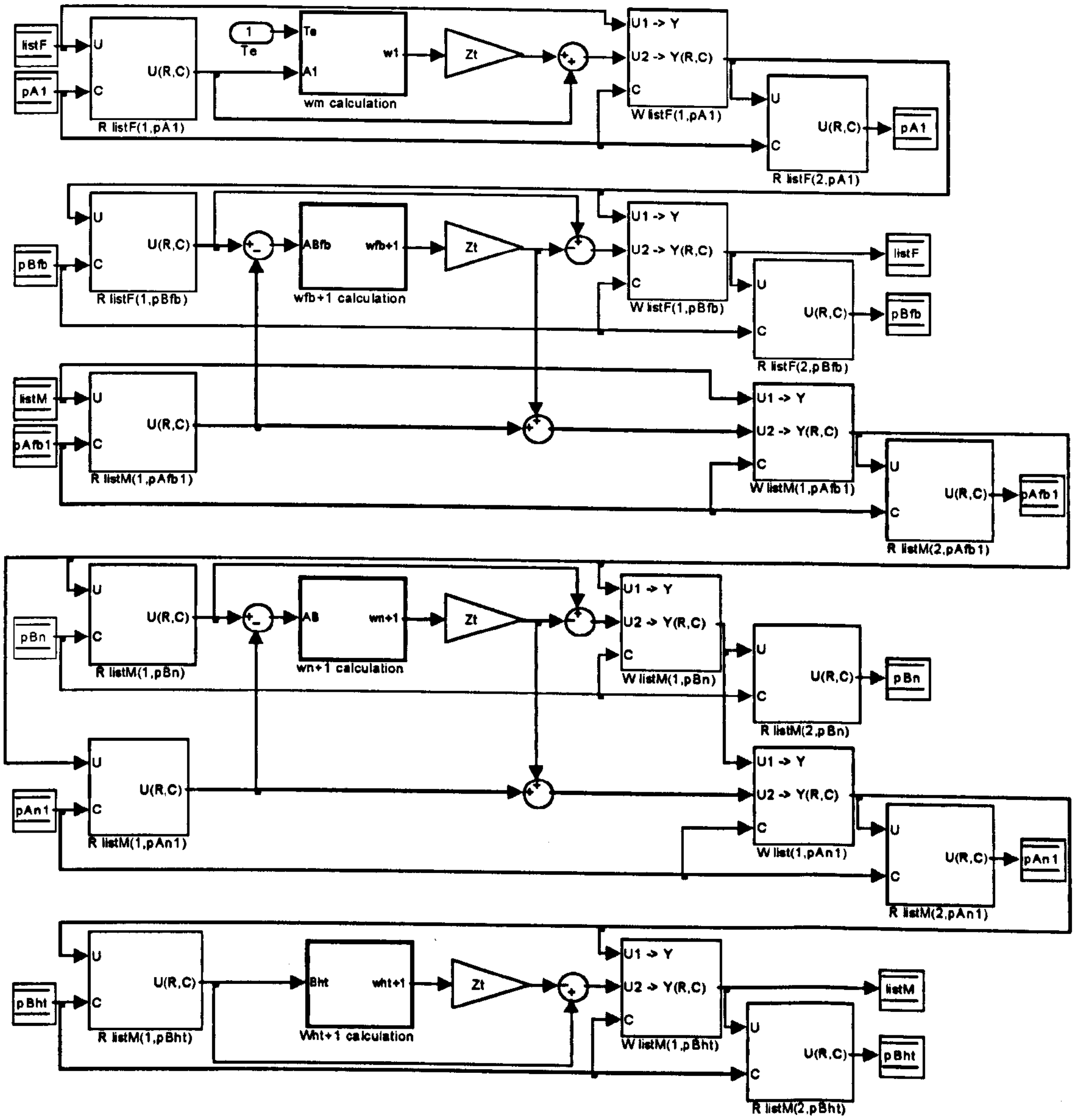


Figure 7.16 Torsional model block in SIMULINK

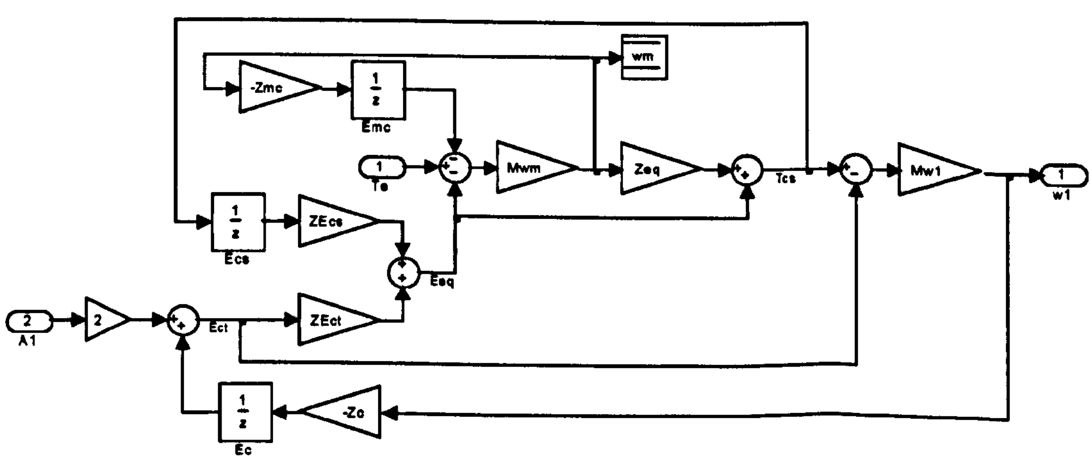


Figure 7.17 Coupling and motor mechanical model block diagram in SIMULINK

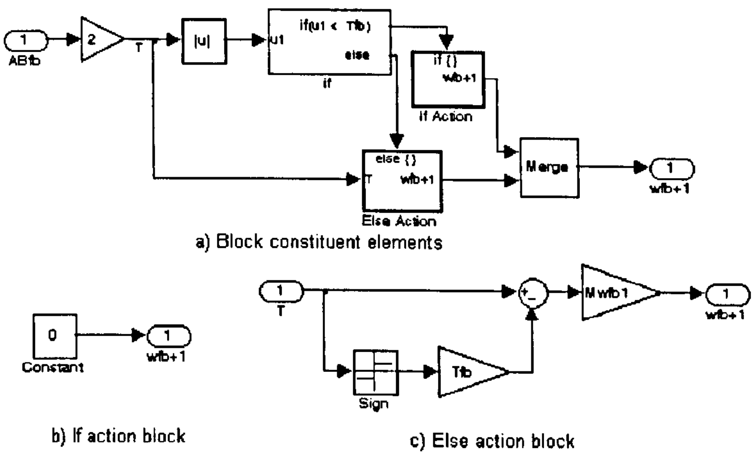


Figure 7.18 w_{fb+1} calculation block in SIMULINK

The block *wn+1 calculation*, contains the torsional part of the model for the connection between nut and screw shaft as defined in section 5.1.3.3. *If condition* and *if action* blocks are used to model the effect of the nut pretension torque T_p as shown in Figure 7.19 (Equation 5.55).

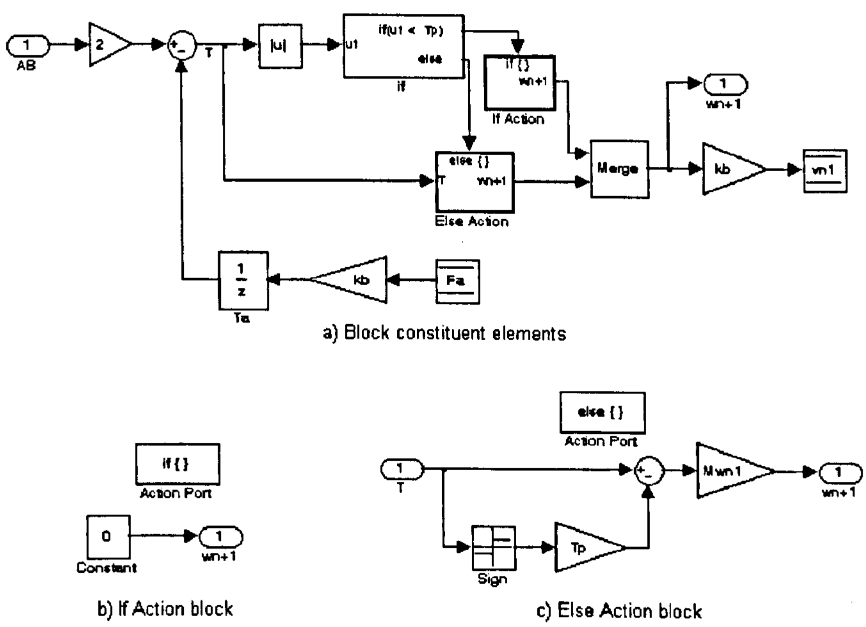


Figure 7.19 w_{n+1} calculation block in SIMULINK

The block w_{ht+1} *calculation*, features the calculation of the rear bearing angular velocity. This block has the same structure of that presented for the w_{fb+1} *calculation* block. Figure J.1 in Appendix J contains the constituent elements for this block.

The *nut position monitoring* block updates the pointers to the *listF* and *listM* variables according to the displacement of the nut. The model resembles the algorithm described in section 7.1.2.1 for the pulse propagation on the screw shaft torsional model. A particular sorted order has been set for the block execution as two different blocks access the variable *lastSec* at the same time (See the step numbers on each block in Figure 7.20). The constituent elements of the *case 1* and *case -1* blocks are illustrated in Figures J.2 and J.3 respectively in Appendix J.

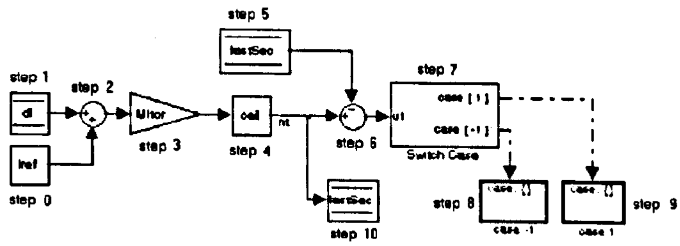


Figure 7.20 Nut position monitoring block in SIMULINK (torsional loop)

7.1.2.3 Axial Loop Subsystem

This subsystem contains the calculation of velocities v_{1a}, v_{na+1} and v_{ha+1} and the pulse propagation of axial waves as described in sections 5.1.3.3 and 5.1.3.4 (See Figure 7.21). This subsystem initialisation code is included in Appendix K.

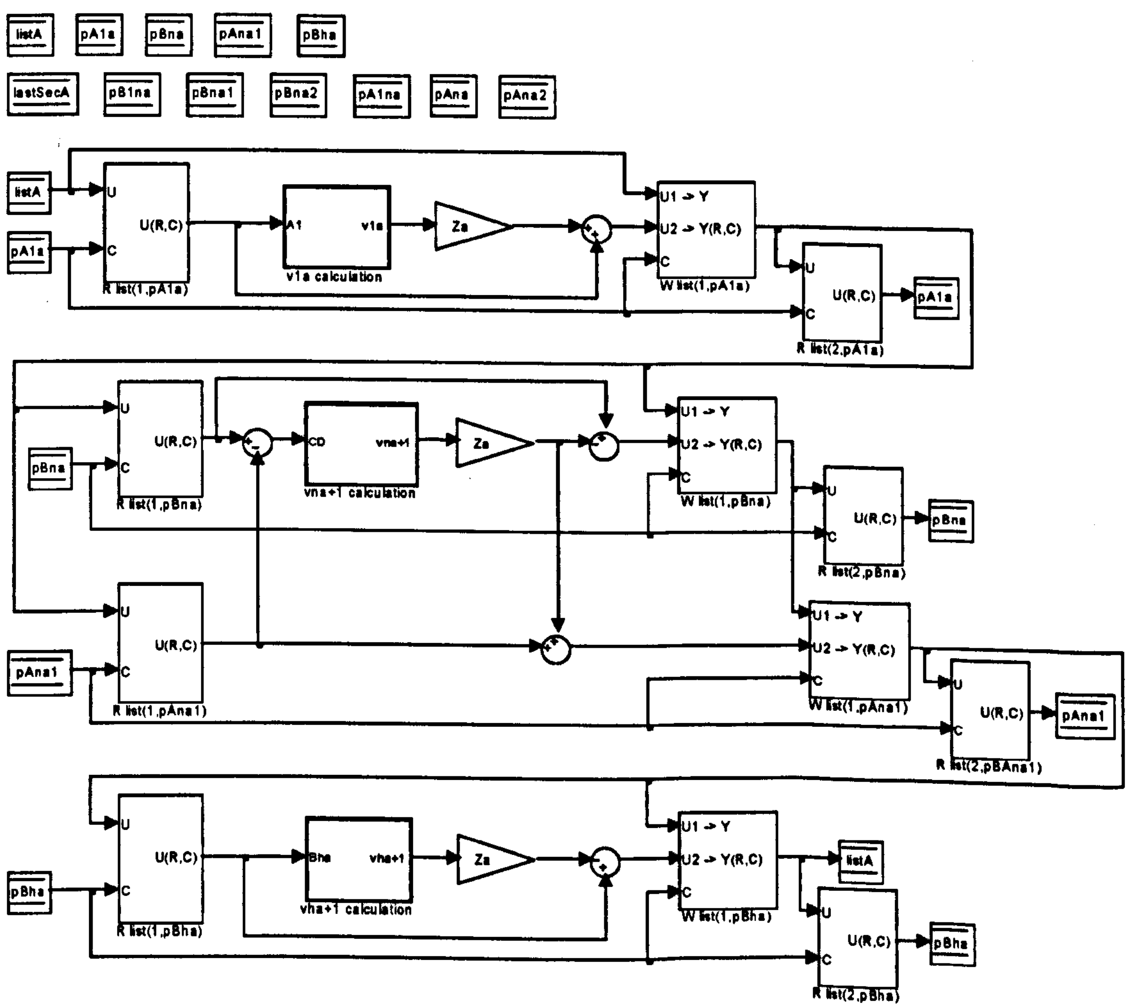


Figure 7.21 The axial loop subsystem block model in SIMULINK

The variables defined in Appendix I to simulate the axial pulse propagation have been implemented in Data Stored Memory blocks (*listA*, pA_{1a}, pB_{na}, pA_{na+1}, pB_{ha}, *lastSecA*, pB_{na-1}, pB_{na+1}, pB_{na+2}, pA_{na-1}, pA_{na} and pA_{na+2}).

The block *v1a calculation*, comprises the front bearing mounting stiffness model - equations (D33 – D.47) in Appendix D. Its constituent elements are shown in Figure 7.22.

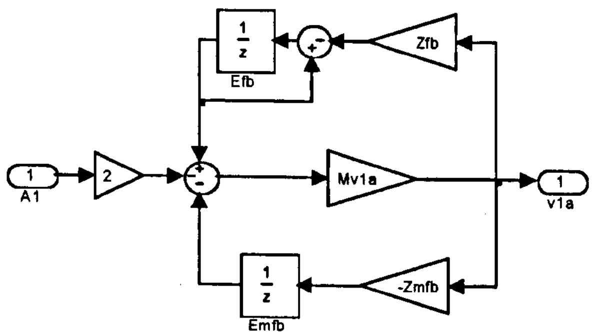


Figure 7.22 v_{1a} calculation block in SIMULINK

The rear bearing mounting stiffness model (Appendix D equations (D48 – D.62)) is implemented in the block *vha+1 calculation*. Its constituent elements are shown in Figure 7.23.

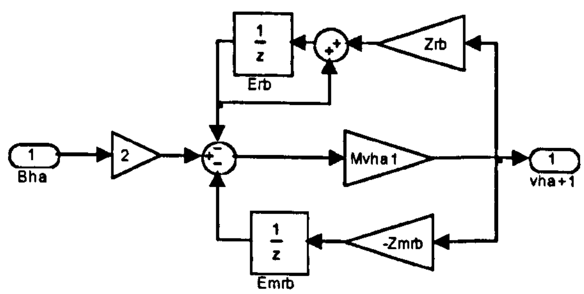


Figure 7.23 v_{ha+1} calculation block in SIMULINK

The block *vna+1 calculation*, comprises the nut and table models as presented in section 5.1.3.4 - equations (5.61 – 5.80). Its constituent elements are shown in Figure 7.24.

The *nut position monitoring* block updates the pointers to the *listA* variables according to the displacement of the nut. The model resembles the algorithm described in Appendix I for the pulse propagation on the screw shaft axial model. A particular sorted order has been set for the block execution as two different blocks access the variable *lastSecA* at the same time (See the numbers on each block in Figure 7.25). The constituent elements of the *case 1* and *case -1* blocks are illustrated in Figure 7.26 and Figure K.1 respectively.

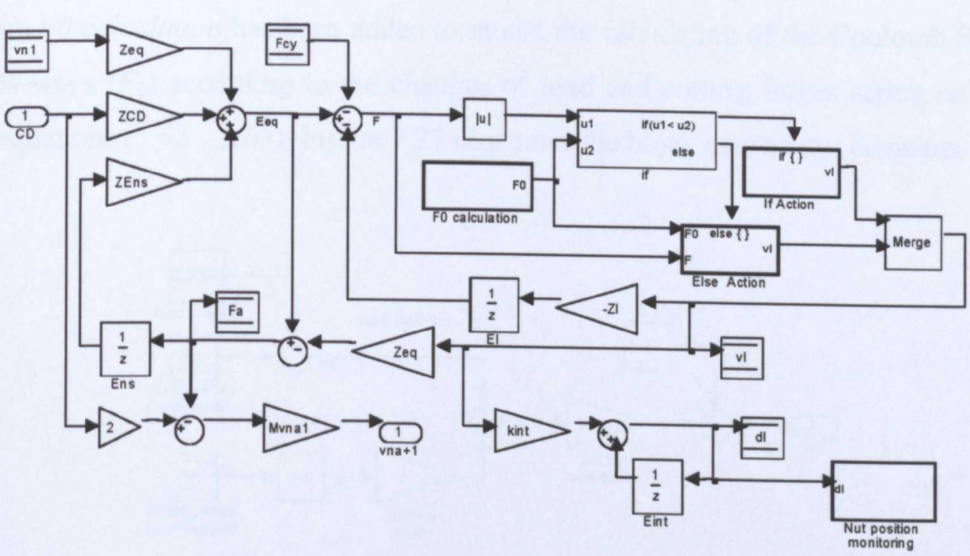


Figure 7.24 The v_{na+1} calculation block in SIMULINK

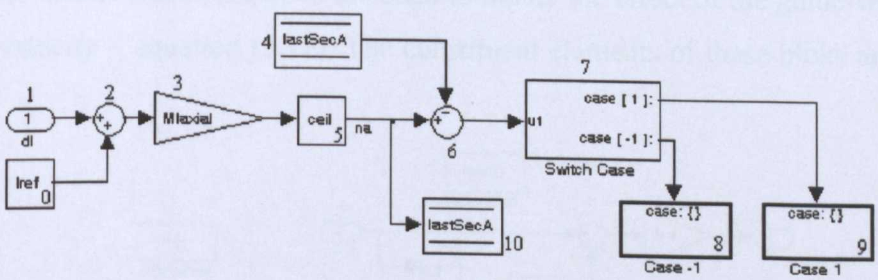


Figure 7.25 Nut position monitoring block in SIMULINK (axial loop)

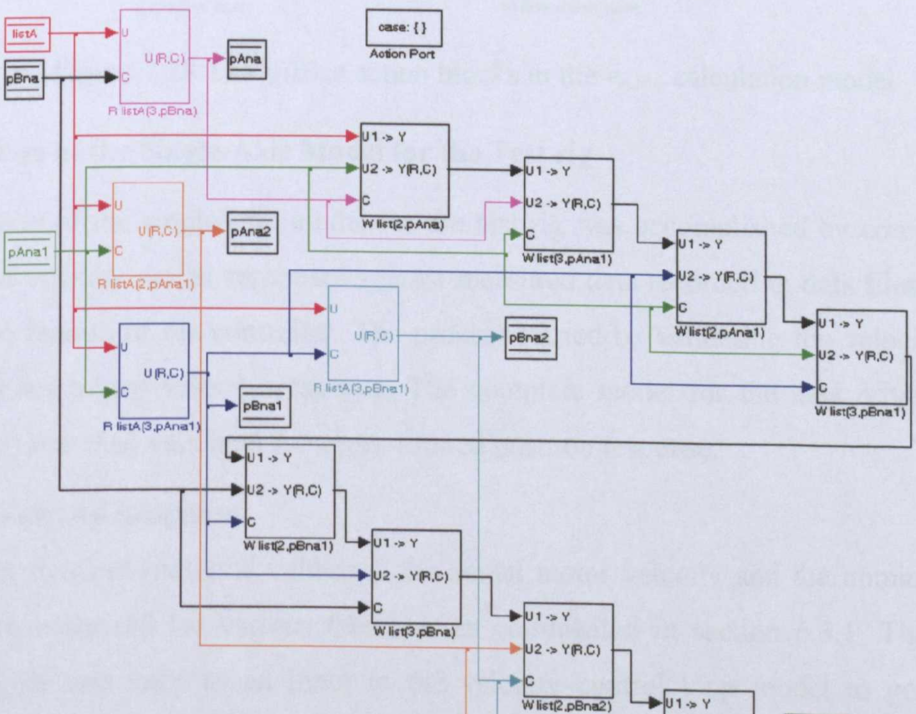


Figure 7.26 Case 1 block for the nut position monitoring in the axial loop subsystem

The block *F0 calculation* has been added to model the calculation of the Coulomb friction in the guide-ways (F_0) according to the changes of load and cutting forces acting on the axis drive – equations (5.62 – 5.65). Figure 7.27 illustrates the block constituent elements.

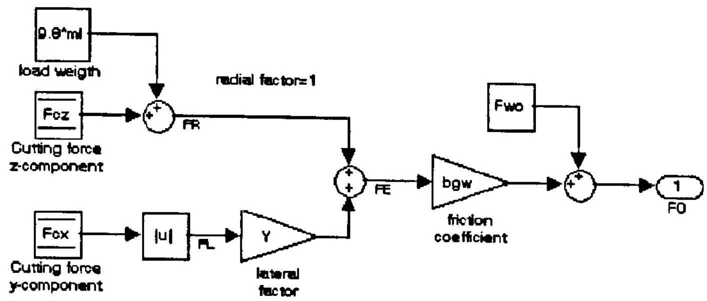


Figure 7.27 The F0 calculation block in SIMULINK

A set of If/Else action blocks has been included to model the effect of the guide-ways friction on the table velocity – equation (5.71). The constituent elements of these bloks are shown in Figure 7.28.

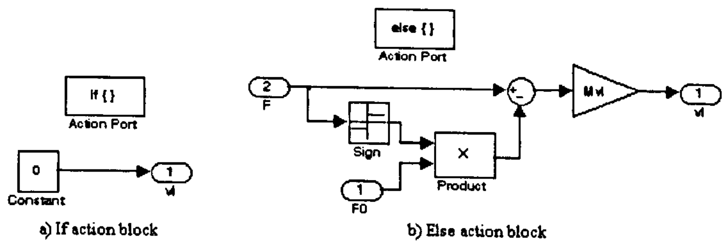


Figure 7.28 The If/Else action blocks in the v_{na+1} calculation model

7.2 Validation of the Single-Axis Model for the Test rig

The validation of the single-axis model for the test rig was accomplished by comparing the position and velocity model responses against measured data recorded in data files using the oscilloscope feature of the controller. The process started by validating the velocity control loop model for a step velocity response. The complete model for the axis drive (position control loop) was then validated for a jerk-limited position response.

7.2.1 Step Velocity Response

The velocity demand (nominal velocity), the actual motor velocity and the nominal current signals were measured for various feedrates as commented in section 6.3.1. The nominal velocity signal was used as an input to the velocity control loop model to generate the simulated actual velocity and nominal current as shown in Figure 7.29. These signals were then compared with the measured ones. Figure 7.30 and 7.31 show the results for step demands of 500 and 1000 mm/min respectively.

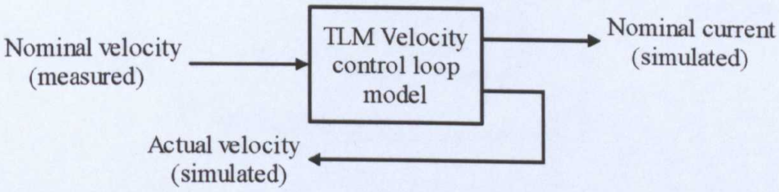


Figure 7.29 Set up for the validation of the step velocity response

The maximum value of the i_q reference current appears as the drive attempts to accelerate the load in response to the nominal velocity step signal. The transient settles and the current reduces in magnitude after the motor has reached the requested value. The motor is now at constant velocity and is overcoming the effects of friction and pretension.

Simulated velocity and current responses (red line) match closely the measured values (green line). A 3% error is visualised on the transient area (0 – 0.02 seconds). The percentage error increases to about 5% when the motor reaches constant velocity. This error is mainly caused by a frequency oscillation of about 150 Hz that the model is damping as illustrated in Figure K.3 A deeper frequency analysis of the axis-drive is needed in order to improve the accuracy of the model and therefore identify the model parameters subject to modification.

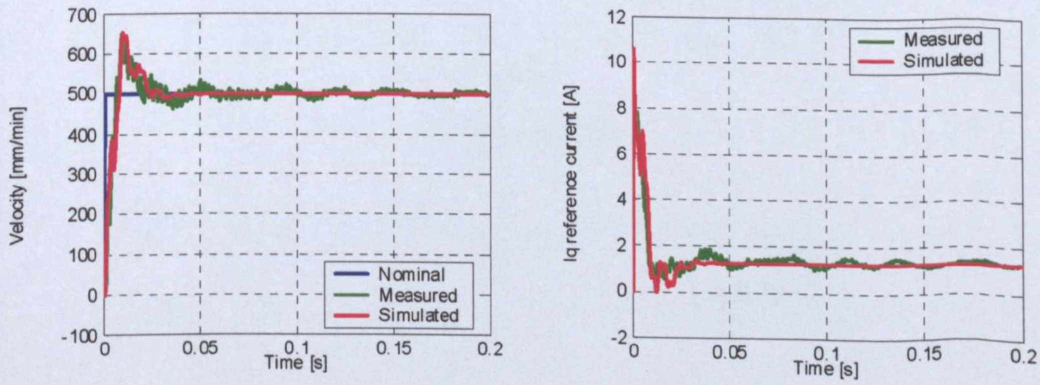


Figure 7.30 Step velocity response validation for the TLM test rig model (500 mm/min)

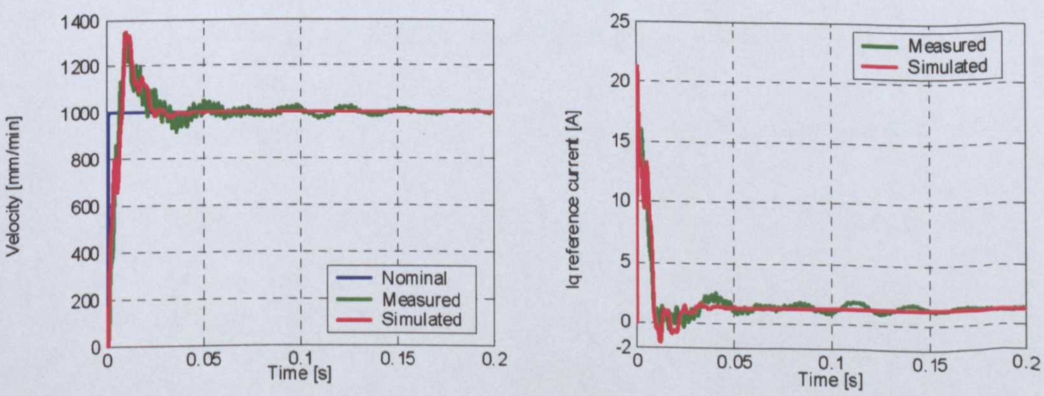


Figure 7.31 Step velocity response validation for the TLM test rig model (1000mm/min)

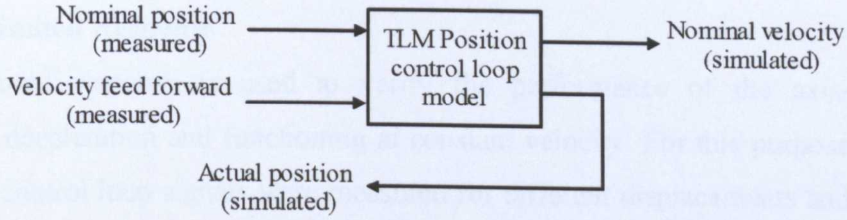
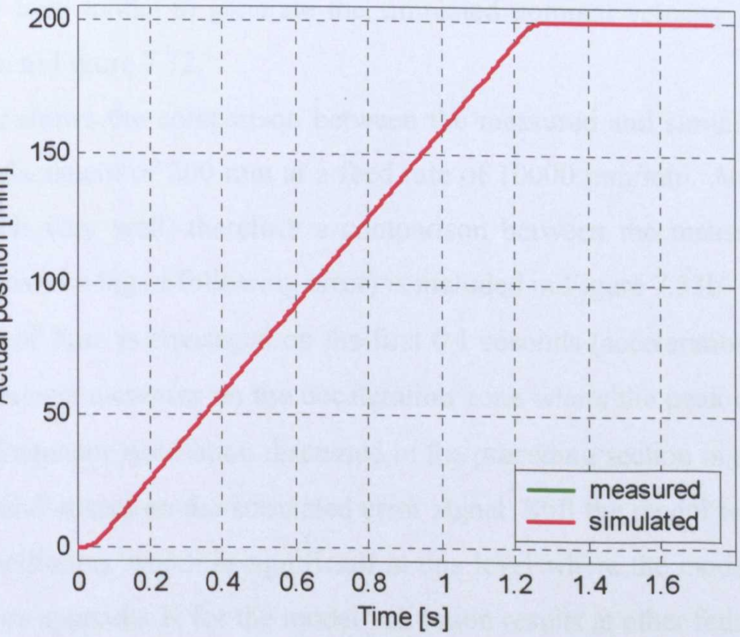
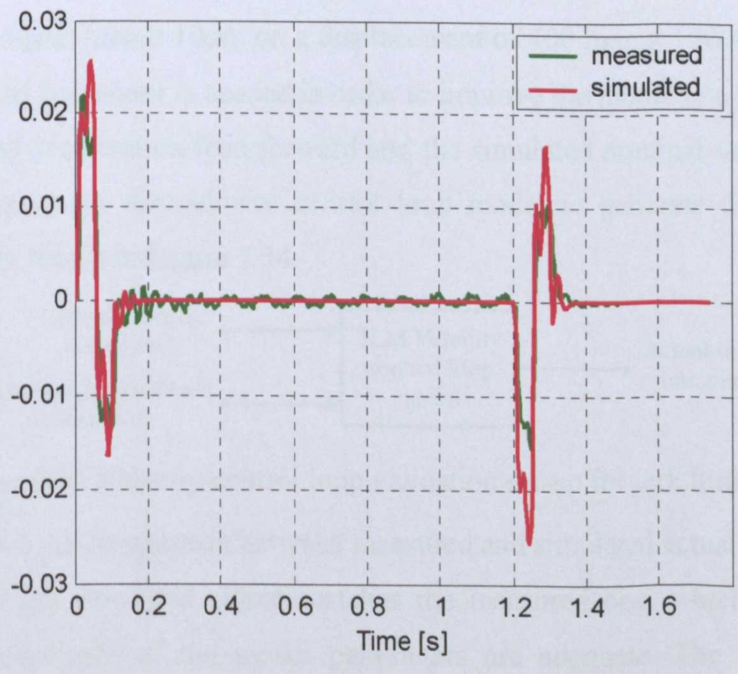


Figure 7.32 Position control loop validation set-up for jerk limited response



a) Actual position



b) Position error

Figure 7.33 Comparison between measured and simulated actual position

7.2.2 Jerk-Limited Response

The jerk-limited response is used to verify the performance of the axis-drive during acceleration, deceleration and functioning at constant velocity. For this purpose, the position and velocity control loop signals were measured for different displacements and feed rates as described in section 6.3.1.

The nominal position and the velocity feed forward signals were used as inputs for the position control loop model to generate the simulated nominal velocity and actual position signals as shown in Figure 7.32.

Figure 7.33a shows the comparison between the measured and simulated actual position signal for a displacement of 200 mm at a feed rate of 10000 mm/min. As seen in the figure, the signals match very well, therefore a comparison between the measured and simulated position error (position lag or following error) is included in Figure 7.33b.

A difference of 2μm is envisaged on the first 0.1 seconds (acceleration zone) of the error signal. The difference increases on the deceleration zone where the peak difference is almost 8μm. The low-frequency oscillation discussed in the preceding section is present again on the measured error and absent on the simulated error signal. Still the model response seems to be damping that oscillation, which is significant at this level where the model response must be very accurate. See appendix K for the model validation results at other federates.

The results are encouraging because the model is responding closely to the real system; however the data is showing a 20% error at maximum on the deceleration zone of the following error signal (about 10μm on a displacement of 400 mm at 10000 mm/min). A deep study of the model behaviour is needed in order to improve the model to a higher accuracy.

The measured acceleration feed forward and the simulated nominal velocity signals were then used as inputs for the velocity control loop model to generate the simulated actual velocity signal as shown in Figure 7.34.

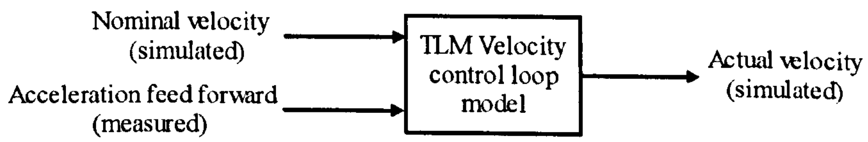
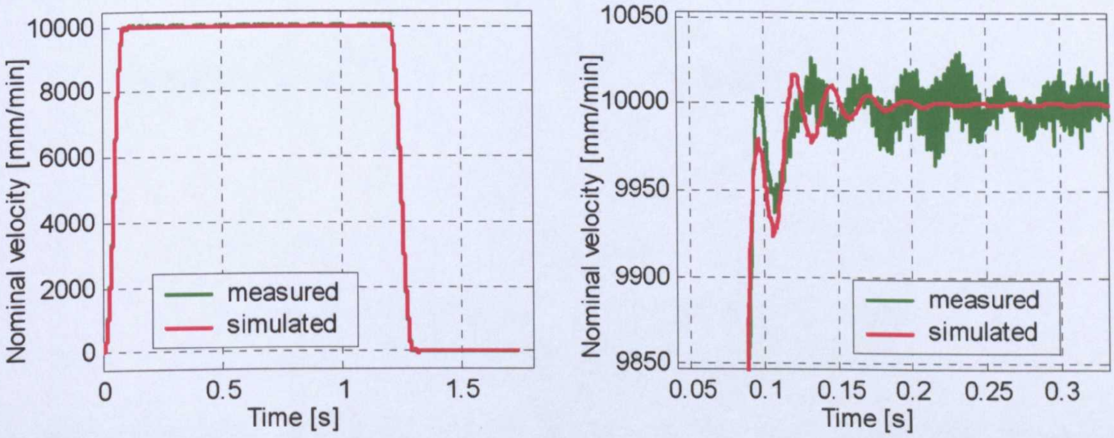


Figure 7.34 Velocity control loop validation set-up for jerk limited demand

Figure 7.35 shows the comparison between measured and simulated actual velocity signals. As can be seen the simulated signal matches the measured one, which indicates that the structure and magnitude of the model parameters are adequate. The difference between measured and the simulated signal is the reflection of the differences in the position error signals amplified by the position controller gain plus the modelling error inherent in to the application of the TLM transform.



a) Reference velocity

b) Zoom acceleration zone

Figure 7.35 Comparison between measured and simulated actual velocity

Figure 7.36 shows the measured and simulated current demand signal (i_{qref}) for a feedrate of 5000 mm/min. As can be seen the simulated and measured signals match reasonably closely, which indicates that the structure and magnitude of the gain parameters for the PID controller and the notch filter are adequate. Figure 7.37a shows the difference between the measured and simulated reference current signals for the established feed rate. Figure 7.37b illustrates the model error for a feed rate of 40000 mm/min. Although the model response closely matches the measured signal, it is evident that there is a difference between both signals, with a maximum error of 0.15 A at the highest possible feed rate for the system (40000 mm/min). This effect can be attributed to the difference between the velocity filter parameters and the ones used in the model due to the fact that the type of filter, its order and coefficients are not accessible to the public domain so are estimated.

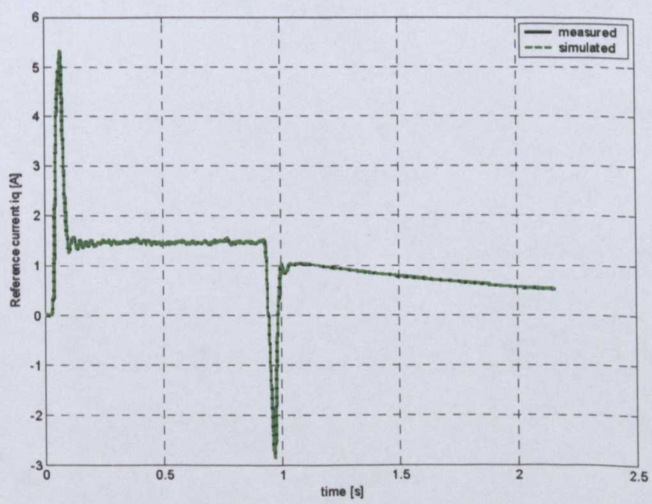
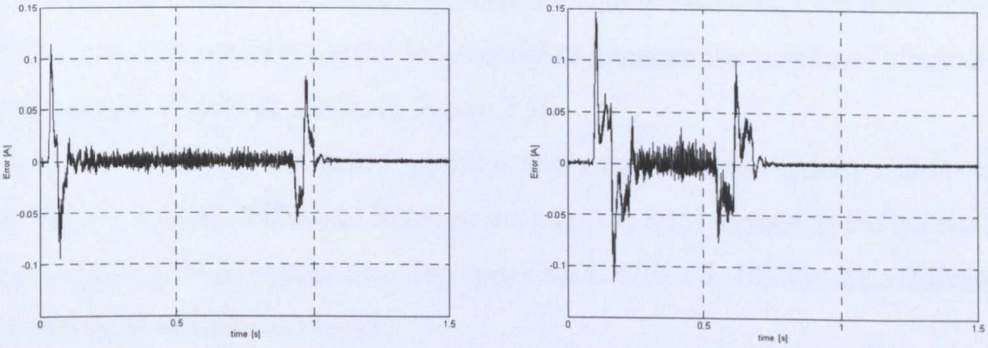


Figure 7.36 Experimental and simulated reference current (5000 mm/min)



a) Error at 5000 mm/min b) Error at 40000 mm/min

Figure 7.37 Model error for the velocity controller

7.3 Implementation of Single-Axis TLM Model for the Arrow 500 Machine in SIMULINK

As discussed in chapter five, the main differences between the Arrow 500 axis drive and the test rig single-axis drive are the control algorithm and the type of the rear bearing mounting used. Some blocks of the test rig model (the velocity controller block, the current controller block, w_{ht+1} calculation block and the v_{ha+1} calculation block) were modified consequently to account for the configuration of the Arrow 500 feed drives. Appendix L contains the implementation of these blocks in SIMULINK.

The updated model was then used to model the x and y-axis of the Arrow 500 CNC machine tool. The approach used for the validation of the test rig TLM model was used for the validation of the x and y-axis of the Arrow 500.

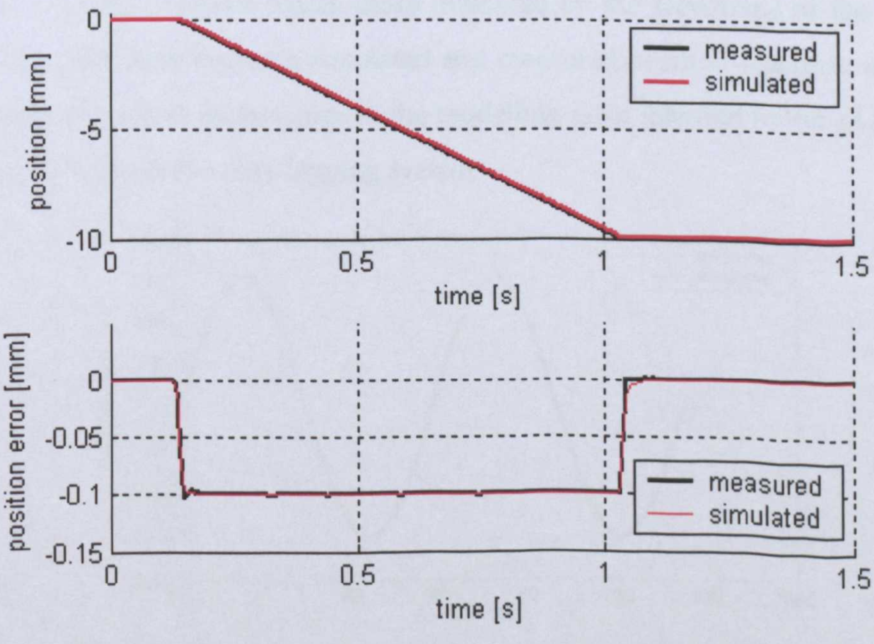


Figure 7.38 Validation position response - arrow (fr = 500 mm/min, d= 10 mm)

The nominal position and the velocity feed forward signals measured from the controller were used as inputs for the position control loop model to generate the simulated nominal velocity and actual position signals as shown in Figure 7.38.

As seen in the figure, the actual position and error position signals match very well; however there is a small difference between the position error signals in the acceleration and deceleration picks (3% at maximum). See appendix L (L.6 – L.10) for the validation results for other federates and displacements.

The reference position and velocity feed-forward signals, measured for a circular movement on the x-y plane (radius 150mm and feedrate of 1000 mm/min) were used as testing inputs for the validation of the x and y-axis TLM model response to a sinusoidal position demand. The response for the x-axis model is shown in Figure 7.39. Figure 7.40 illustrates the y-axis model response.

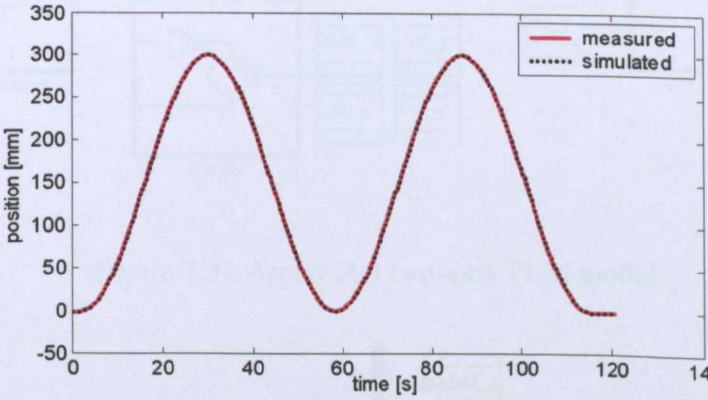


Figure 7.39 X-axis position model response to a sinusoidal demand

As observed, simulation results match those measured by the servotrace of the Arrow 500's controller. The difference between simulated and measured position response is about 0.1% and is the result of various factors such as the modelling error inherent to the TLM algorithms and quantisation noise in the data-logging system.

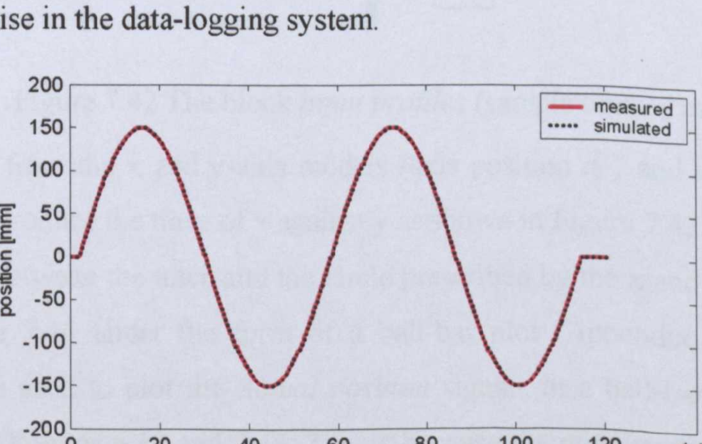


Figure 7.40 Y-axis position model response to a sinusoidal demand

7.4 Implementation and Validation of Two-Axis TLM Models

The two-axis model contains the separate models for the x-axis and y-axis discussed previously (Figure 7.41). Sine and cosine position demand signals measured to prescribe a circular movement are introduced from the MATLAB workspace to the x and y-axis models as depicted in Figure 7.42.

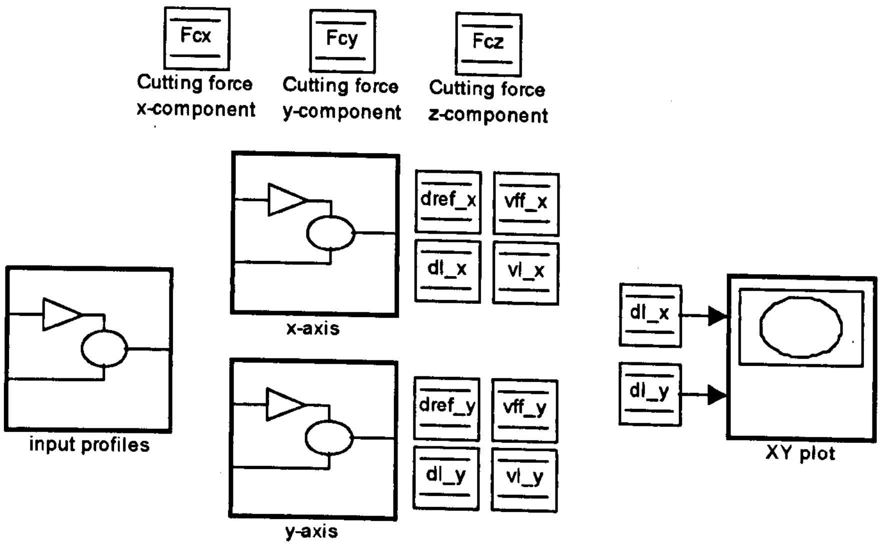


Figure 7.41 Arrow 500 two-axis TLM model

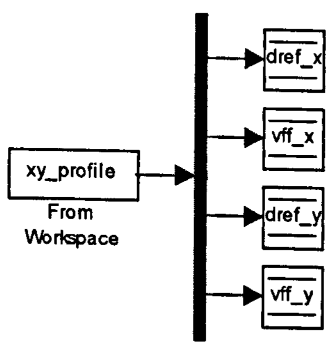


Figure 7.42 The block *input profiles* (sample time: 4 ms)

The output signals from the x and y-axis models (axis position d_{l_x} and d_{l_y}) are introduced into a XY block to produce the trace of x against y as shown in Figure 7.43.

The difference between the trace and the circle prescribed by the x and y position demand is drawn in Figure 7.44 under the form of a ball-bar plot (Appendix G.12 contains the MATLAB program used to plot the *Actual position* signals in a ball-bar format). Note the difference between Figures 6.12 and 7.44. This is because the measurements from the linear encoders (Figure 6.21) do not reflect the geometric and load errors of the two-axis system.

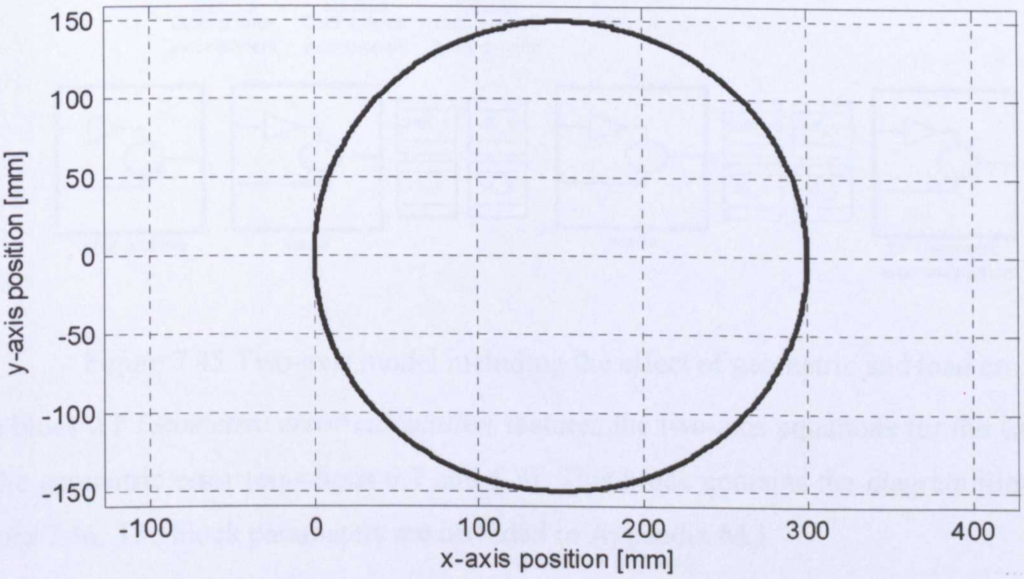


Figure 7.43 Two-axis TLM model position response

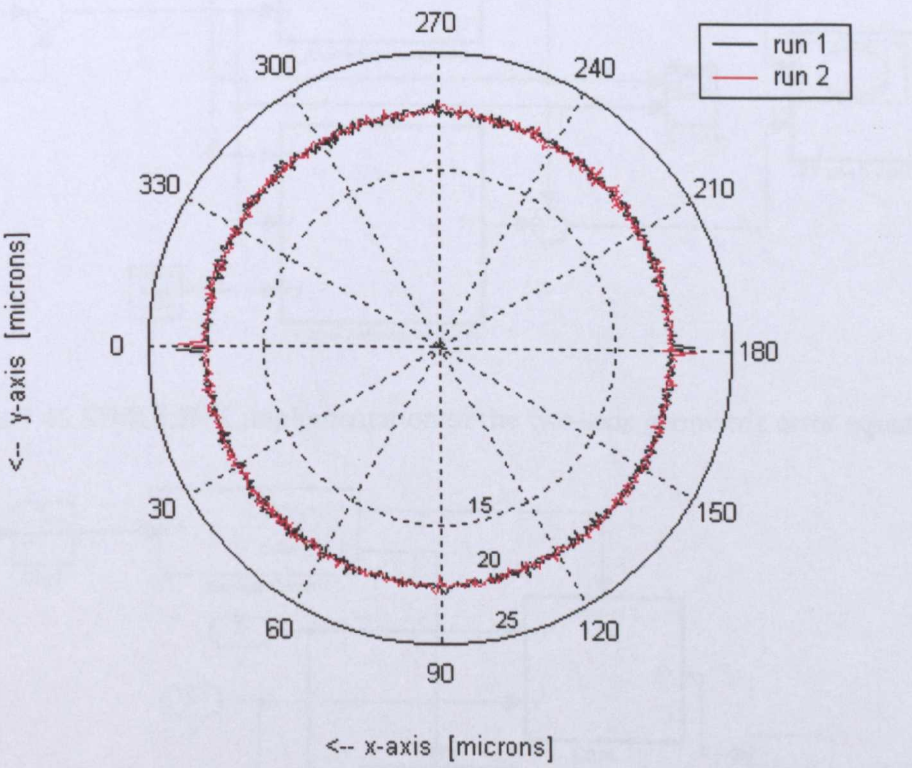


Figure 7.44 Two-axis simulated actual position in ball-bar format

To provide a model that conforms more closely to an actual machine, it is necessary to model the geometric and load error components for each axis and incorporate them into the simulation as described in section 5.3. Figure 7.45 illustrates the two-axis model including the geometric and load error calculation.

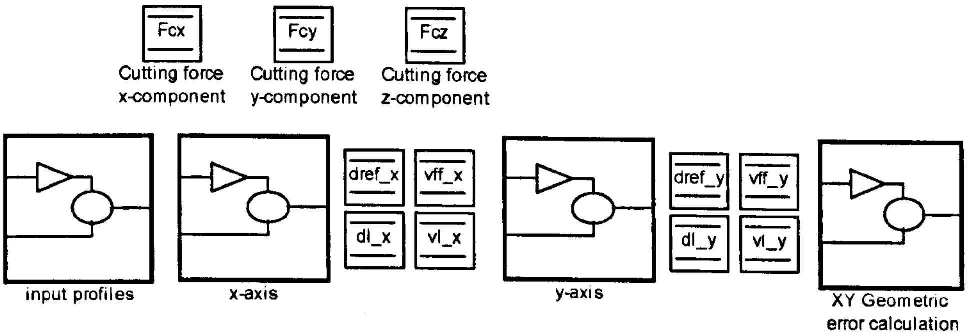


Figure 7.45 Two-axis model including the effect of geometric and load errors

The block *XY Geometric error calculation* features the two-axis equations for the calculation of the geometric error (equations 6.7 and 6.8). This block contains the diagram illustrated in Figure 7.46. The block parameters are included in Appendix M.1.

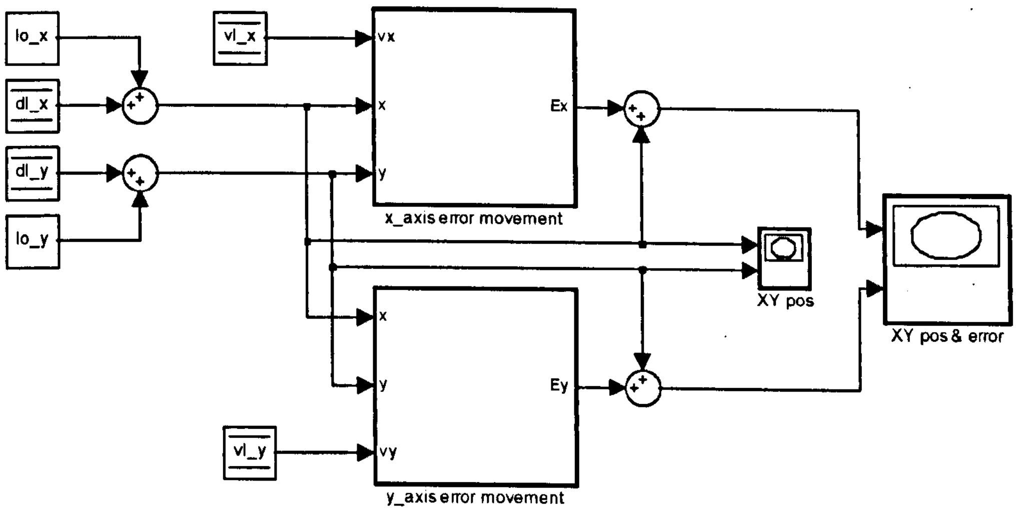


Figure 7.46 SIMULINK implementation of the two-axis geometric error equations

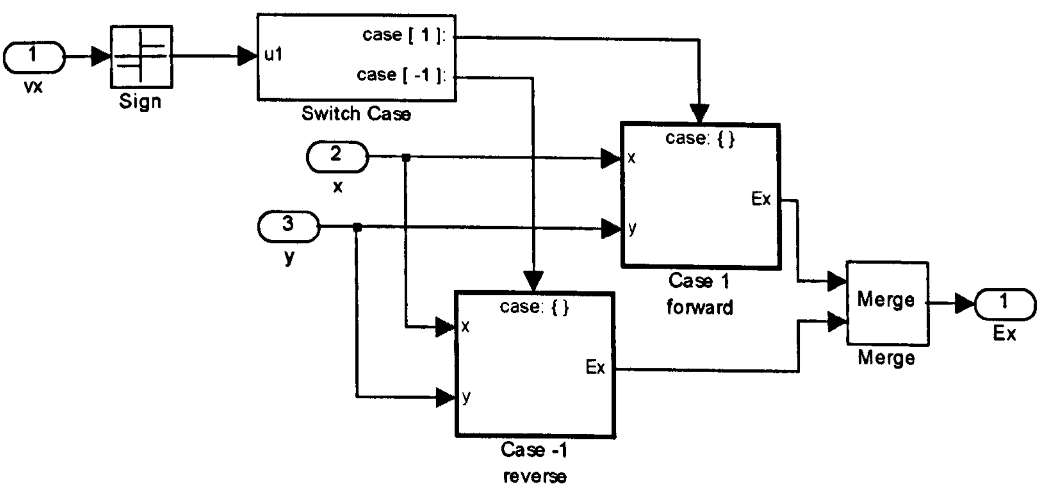


Figure 7.47 Block diagram for the x-axis geometric error calculation

The DSM *vl_x* and *vl_y* blocks have been included to monitor the movement direction of each axis and to select the appropriate set of geometric errors (forward or reverse). The block *x-axis error movement* contains the block diagram for equation (6.7) as shown in Figure 7.47.

Two cases are considered: axis moving forward (moving away from origin of the coordinate system) or axis moving in reverse (moving towards the origin of the coordinate system).

Figure 7.48 illustrates the block structure for *case 1* (moving forward). The polynomials calculated from the geometric error components (see Appendix G.2) are implemented in *polynomial evaluation* SIMULINK blocks. The scaling and centring of each polynomial is executed by the *scaling* blocks, which have the structure presented in Figure 7.49. Figure M.1 contains the block diagram for *case –1* (reverse).

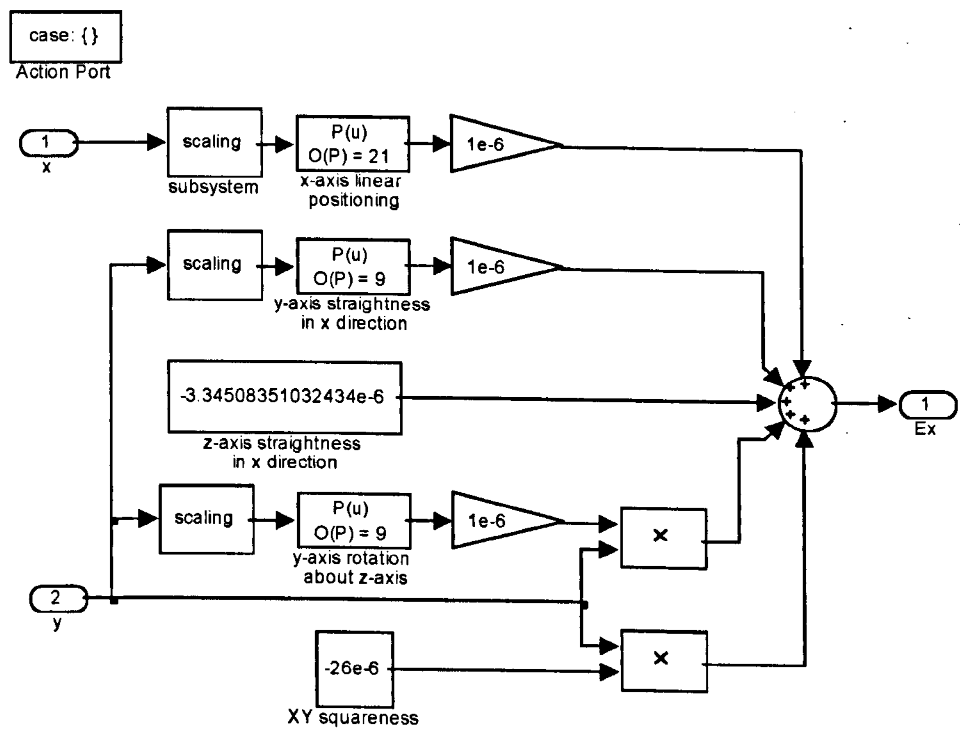


Figure 7.48 X-axis geometric error calculation (case 1: forward)

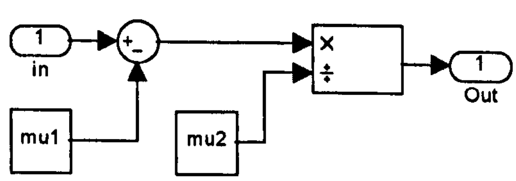


Figure 7.49 The scaling block

Figures M.3 and M.4 contain the block diagrams for the subsystem *y-axis error movement* .

The two-axis model including the geometric error model was simulated and the model position response (axis position d_l) was introduced into a XY block to produce the trace of x against y as shown in Figure 7.50. The difference between the trace and the circle conformed by the x and y position demand is drawn in Figure 7.51 under the form of a ball-bar plot.

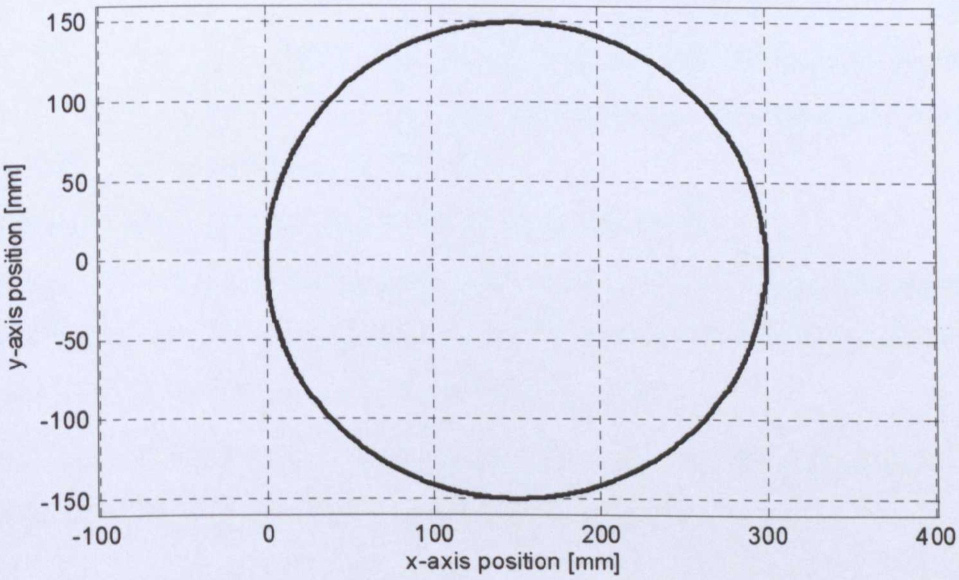


Figure 7.50 Two-axis TLM model position response with geometric errors

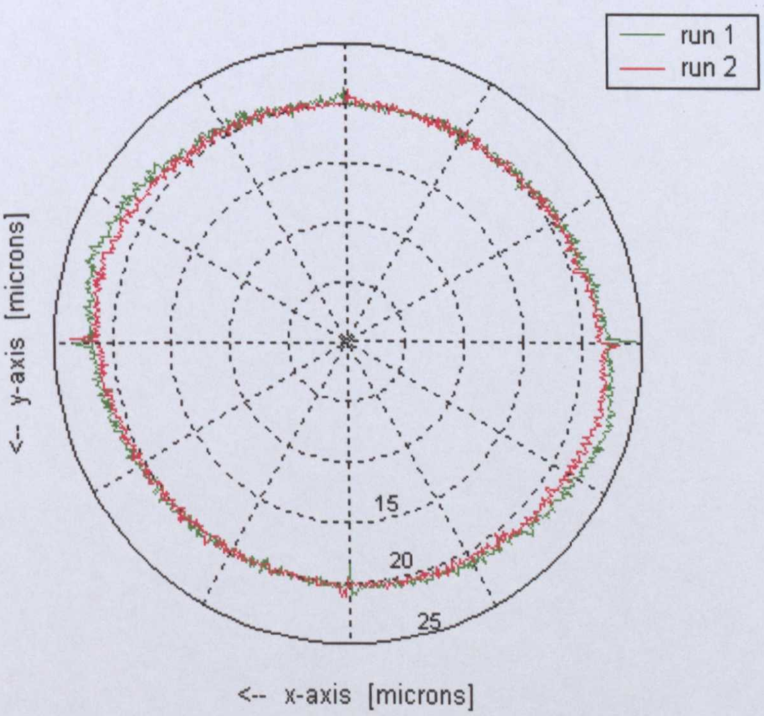


Figure 7.51 Simulated Ball-bar plot with polar coordinates

A comparison of this plot (Figure 7.51) against the results from the ballbar test performed on the machine (Figure 6.12) gives the following results:

- The plot has an oval shape as a result of the squareness error (-26 μm/m) which is consistent with the same feature illustrated in Figure 6.12.
- The two axis model shows that the combination of the geometric errors generates a progressive error deviation when the machine worktable prescribes a circle. However the simulated results do not match the ballbar measurements on the arcs described in the intervals [45° - 90°] and [135° - 270°]. This difference may be attributed to divergences on the straightness measurements for the x-axis.
- The reversal spikes match closely on both x and y directions
- Although the model including geometric errors gives an approximation of the real movement of the table taking as a reference the cutting tool hub, more analysis is needed in order to improve the model to high standard response.

The next chapter presents the techniques generally used for the identification of natural frequencies and damping ratios from measured bode diagrams.

8 IDENTIFICATION OF RESONANT FACTORS IN CNC MACHINE TOOLS

The application of the derived TLM models to real systems such as CNC machine tools implies a combination of theoretical and experimental analysis. The dynamic behaviour of the system under study is in general complex with various aspects unknown or not studied. Non-linearities, noise and element tolerances produce distortion on the system response that the model will struggle to replicate without future work. The complete dynamic behaviour of the system could be derived from identified modal parameters (natural frequencies of vibration, damping coefficients and mode shapes) and adequate corrections can be undertaken on the model.

This Chapter deals with methods generally used for system identification and some considerations about the use of the Continuous Wavelet Transform (CWT) for the detection of resonant frequencies and damping factors in CNC machine tools.

8.1 Identification Methods for Modal Parameters of CNC Machine Tools

The modal parameters estimation of real systems is an essential step in the modelling process because it provides important information on inherent dynamic properties of the structure. Since dynamic properties are directly related to mass, stiffness, damping and boundary conditions, modal parameters can be regarded as a function of these properties. Kullaa [116] showed that modal parameters could be used to improve analytical models, enhance system design or for condition monitoring purposes.

Andersen [117] stated that modal parameters can be extracted via parametric and non-parametric system identification methods through a process known as *modal analysis:*

- <u>The parametric methods</u> consist of building a mathematical model from a set of assumed parameters. These parameters are estimated from an iteration process during the system identification, and modal parameters are then derived using direct mathematical relationships with the estimated model parameters.
- <u>The non-parametric methods</u> apply different curve fitting procedures in order to match defined curves, functional relationships or tables to measured system response and /or excitation signals.

The fundamental modal parameters are *natural frequencies* (the resonant frequencies), *damping ratios* (the degree to which the structure itself is able to damp out vibrations), *mode shapes* (the way the structure moves at a certain resonant frequency) and *modal participation factors* (masses and residues). Drexel & Ginsberg [118] showed that mode shapes characterise the so-called modal vectors and modal participation factors characterise modal scaling.

The modal analysis involves a quantifiable input that is applied to the system and the output is measured. A modal model (set of modal parameters) is obtained from the measurements via a non-parametric or parametric system identification method. In some cases, this process is truncated by the difficulty to apply a measurable input or the impossibility to measure an ambient excitation. Therefore, the outputs are the only information available for the identification algorithm. The assumption in this case is that the input is the realisation of a stochastic process (white stationary noise). Peeters & De Roeck [119] demonstrated that the dominant frequencies could not be separated from the eigenfrequencies of the system when the assumption was violated by the presence of dominant frequency components.

A detailed survey of classical methods for modal parameter identification has been performed by De Silva [120]. As a result of this thorough overview, the author presented five most used methods for modal parameter identification of single-degree-of-freedom (SDOF) and multiple-degree-of-freedom (MDOF) systems (see Table 8.1).

Method	Measurements	Formulas
Logarithmic decrement method	A_i – first significant amplitude A_{i+r} – amplitude after r cycles ζ - damping factor	Logarithmic decrement $\delta = \frac{1}{r}\ln\frac{A_i}{A_{i+r}} = \frac{2\pi\zeta}{\sqrt{1-\zeta^2}}$
Step-response method	M_p – peak value of response PO – percentage overshoot (over steady-state value)	$M_p = 1 + \exp\left[\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}\right]$ $PO = 100\exp\left[\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}\right]$
Hysteresis loop method	ΔU – area of displacement-force hysteresis loop x_0 – maximum displacement of the hysteresis loop k – average slope of the hysteresis loop	Hysterectic damping constant $h = \frac{\Delta U}{\pi x^2}$ Equivalent damping ratio $\zeta = \frac{h}{2k}$
Magnification-factor method	Q – magnitude of FRF (Frequency Response Function) at resonance frequency	$Q = \frac{1}{2\pi\zeta\sqrt{1-\zeta^2}}$
Bandwidth method	$\Delta\omega$ – bandwidth at 0.707 of resonant peak ω_r – resonant frequency	$\zeta = \frac{\Delta\omega}{2\omega_r}$

Table 8.1 Classical damping measurement methods [120]

The methods use measured Bode diagrams and determine the resonant frequencies and damping factors from the magnitude of the gain plot when the modal frequencies are not too closely spaced and the system is slightly damped. The damping coefficient depends on the peak width and the resonant frequency is the central value for the frequency interval where the peak occurs.

Zhang et al [121] underlined that these methods were not able to estimate the modal parameters from signals containing non-linearities and variable frequencies (non-stationary signals). Time-frequency analysis has become a solution to this problem and various methods have been proposed in the last two decades. Examples of time-frequency analysis are: Wavelet transform and applications of the short-time Fourier Transform (Gabor transform and Wigner-Ville distribution).

Wavelet analysis employs adaptive windows in order to achieve the best time resolution. Cohen [122] observed that the frequency resolution is different between the lower and the higher frequency band so signals with very high and very low components can be studied giving the opportunity to track changes in oscillation frequencies and amplitudes.

Robertson et al [123] applied the Discrete Wavelet Transform (DWT) for the calculation of impulse response for a system with four degrees of freedom. The forward DWT used the input signal and generated an input/output relation matrix. The impulse response resulted from applying the inverse DWT to this matrix. This method was similar to the Fast Fourier Transform (FFT) based extraction procedure, but the data is handled only in the time-domain. The main conclusions of this study were:

- The wavelet's suitability for a system excited by harmonic oscillations (narrow-frequency band) where FFT-based methods performed poorly, unless a sine sweep technique was utilised;
- The same difficulties were experienced by the wavelet method as for the FFT methods using arbitrary random excitations;
- The wavelet method was as good as the correlated FFT method for detecting high-frequency signals in the case of ideal excitations (laboratory-generated burst random signals);
- The extracted impulse response functions could be used by system identification algorithms to extract modal parameters [124].

Ruzzene et al. [125] estimated the natural frequencies and damping factors for a system free response by employing the wavelet analysis. The Hilbert Transform Method was applied to MDOF systems assuming that the excitation signals had zero mean (stationary random processes). The Random Decrement Technique converted the system random responses to free decay responses. Then the Wavelet Transform (WT) estimated the natural frequencies and the mean values of the instantaneous frequency-time histories. Damping ratios were identified from the decay rate of the linear interpolation performed on the WT modulus.

Kijewsky and Kareem [126] studied the application of multi-resolution Morlet wavelet to non-linear systems. A complete modal separation and stability of damping estimates required tripling the classical mean square bandwidth. Also the instantaneous frequency (identified from the wavelet phase or the ridges of the amplitude) was relatively insensitive to end-effects.

Staszewski [127] presented three methods for damping identification based on time-scale decomposition of the continuous wavelet transform: The wavelet transform cross-section procedure, the impulse response recovery procedure based on wavelet domain filtering, and the ridge detection procedure.

The methods did not depend on the choice of the analysing wavelet function, and the *wavelet ridge detection technique* gave the best accuracy (especially for noisy data). The studied numerical examples were systems with two DOF closed and well-separated modes, respectively.

Based on these results, Staszewski [128] presented a procedure for non-linear system identification. The time-dependent amplitude and phase functions of the system impulse response were obtained from the ridges and skeletons of the Morlet wavelet transform. It was also highlighted that the usual method of *local maxima of the transform amplitude* used for the ridge extraction was valid only for linear ridges (linear systems). Identification methods based on the wavelet amplitude (parameterised ridge and combinatorial optimisation procedure, Carmona et al [129]) and methods based on the phase function (Delprat et al. [130]) were used for non-linear systems analysis. Applications of the proposed method to simulated SDOF and MDOF systems gave satisfactory identification results including cases where the noise in the data was up to a relatively high level (20%).

Pislaru et al [131] developed a novel application of CWT for modal parameter identification to CNC machine tool feed drives. Random white noise was applied as excitation signal to estimate the impulse response of the system. The CWT of the impulse response was calculated and the local maxima of the wavelet amplitude were used to specify the resonant frequencies. The linear regression was applied to a damped sine envelope therefore the damping factors were calculated from the logarithmic decrement. The modal parameters identified by CWT of the impulse response were compared with those estimated from measured Bode diagrams. CWT enabled the accurate detection of amplitude variations for weak signals combined with relatively high noise and non-stationary signals. The study demonstrated the superiority of the CWT over classical methods for modal parameter identification based on FFT. The first peak of wavelet amplitude was automatically detected, but the positions of the next ones had to be estimated by human intervention.

The same authors [132] further investigated the construction of an algorithm for the automatic detection of the peaks within the three-dimensional (3D) graph generated by CWT. One modal response was removed at a time from the impulse response of the simulated data containing modes close to each other in frequency or with high damping ratios. The modal parameters could not be detected by the FFT method, but CWT generated accurate results even in extreme cases. However, the algorithm was applied only for simulated data.

8.2 Transfer Function Identification for Control System Loops (Arrow 500)

Built in functions on the motion controller for the Arrow 500 perform spectral analysis for the optimisation of the position, velocity and current closed control loops for each axis [133]. The control frequency characteristics are calculated by entering a Pseudo-Random Binary Signal (PRBS) at the set point of each control loop. Four frequency responses can be measured:

- Closed position controller loop (d_{act} / d_{ref}).
- Closed velocity controller loop (v_{act} / v_{ref}).
- Controlled system: (v_{act} / i_{qact}).
- Mechanical frequency response (v_{act} rotary encoder / v_{act} linear encoder).

The closed position and velocity controller loop responses are used for the optimisation of the position and velocity controller parameters. The control system and mechanical frequency responses are used to set up the parameters for the filters used to damp the system resonant frequencies. These two particular responses are of interest for the identification of the modal parameters of the axis drive.

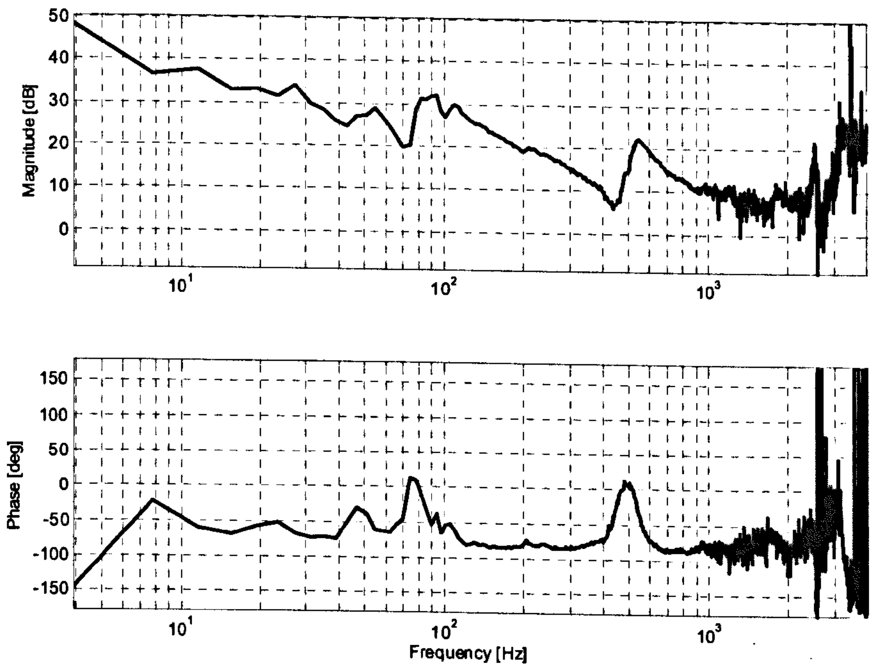


Figure 8.1 X-axis controlled frequency response

The frequency response of the controlled system is used to provide better estimation of the poles and zeros of the control system without any influence of the controller. The transfer function is calculated as:

$$\frac{actual\ \ velocity\ \ motor}{actual\ \ current\ \ motor} = \frac{\omega_m(\omega)}{i_{qact}(\omega)} \tag{8.1}$$

A low gain k_p and a high integrator time T_i (e.g. k_p= 0.1, T_i = 500 ms) are used at the velocity controller to obtain an improved frequency response at low frequencies. Figure 8.1 show a resonant frequency of 555 Hz with a damping ratio of 0.1.

The mechanical frequency response is used for a comprehensive analysis of the performance of an axis. The first pole on this test represents the natural frequency of the axis (table frequency or locked rotor frequency). The higher this pole is, the higher is the performance of the axis.

Figure 8.2 illustrates the mechanical frequency response for the x-axis. The transfer function is calculated as:

$$\frac{actual\ \ velocity\ \ linear\ \ encoder}{actual\ \ velocity\ \ rotary\ \ encoder} = \frac{v_l(\omega)}{\omega_m(\omega)} \tag{8.2}$$

The first pole of the mechanical frequency response limits the dynamics of the axis, normally the maximum reachable k_v factor, the maximum acceleration (a_{max}) and the maximum acceptable jerk (j_{max}).

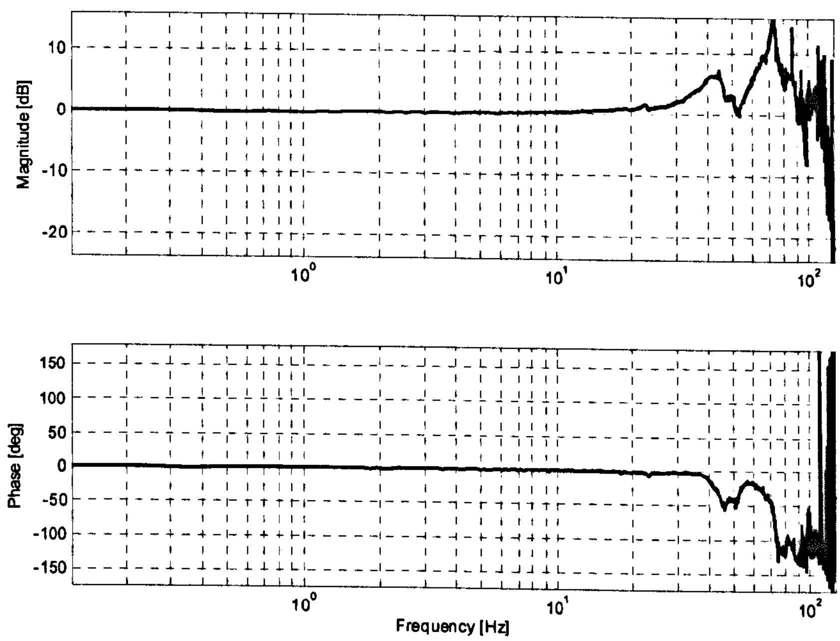


Figure 8.2 X-axis mechanical frequency response

Figure 8.2 shows a natural frequency of 42 Hz with a damping ratio of 0.25 on the x-axis. The controlled and mechanical frequency response measured for the y-axis is presented in Appendix N.

8.3 Modal Parameters Identification Using Wavelets

Wavelet theory has exhibited good results when dealing with problems that involve representation of non-stationary signals generated by diverse causes (vibration of rotating machines, transient behaviour, discontinuities, etc.) [134]. This theory has been applied to a diverse set of general applications in systems theory, and has shown important results in practical applications of systems identification as Pawlak/ Hasiewicz [135] and Liu et al. [136] demonstrated. Special attention has been dedicated to the Morlet wavelet because of various advantages [137]:

- Natural robustness against shifting a feature in time because little or no special precautions are needed to ensure that the feature will make itself known in the same way no matter when it occurs.
- The best filter at simultaneously locating a feature in terms of its period and when it appears.
- Specially convenient for analysing signals with a wide range of dominant frequencies which are localised in different time intervals or amplitude and frequency modulated spectral components.

Appendix O contains the application of a wavelet based algorithm for the identification of modal parameters to simulated data and to bode diagrams measured from the x-axis feed drive of the Arrow 500 CNC machine.

Results showed that the algorithm performed well for the identification of simulated data, but it lacked of precision when it was applied to the data measured from the Arrow 500 CNC machine.

8.4 Summary

Generally, certain types of signals are considered to identify the parameters reflecting the effects of non-linearities and disturbances in machine tools (see Table 8.2) [138]. These signals are chosen because they have been developed under certain design criteria, and for reasons of analytical simplicity. They can include instantaneous change, sinusoidal changes, or one change at a constant rate.

Parameter	Stimuli
Velocity bandwidth Overshooting check	Step
Backlash Dead Band	Triangular
Position Gain Coulomb Friction	Trapezoidal
Resonance frequencies Damping Coefficients	Random white noise Swept sine

Table 8.2 Parameters to be identified and stimuli used for this purpose

The transfer function of the control loops can be identified using frequency response analysis. White noise or swept sine signals are used in this case to excite the system over the bandwidth of interest. Spectral analysis of the bode diagrams is then used to estimate the terms of the transfer function.

Applications of CWT for modal parameter identification have shown that an accurate detection of amplitude variations for weak signals combined with high level of noise and non-stationary signals can be achieved [132]. However, the detection methods used perform well only for linear systems and the algorithms were applied only to simulated data.

An algorithm for the identification of non-linear systems based on the wavelet amplitude (Carmona et al [129]) and Delprat et al [130]) was considered. The technique showed to be effective for the identification of some resonant states but it could not achieve accurate results on the identification of damping factors.

9 CONCLUSIONS AND FURTHER WORK

The aim of the investigation was to focus on the modelling of non-linear control systems associated with the single-axis Bridgeport and the three-axis Arrow 500 CNC machine tools.

The aim and objectives outlined in section 1.1 have been completed and the following conclusions reached:

a) The TLM modelling principles (derived in applications to the modelling of systems of different disciplines) and their extension to the development of mathematical models that can reflect the pointwise and the distributed features of non-linear control systems has been compiled. It was shown that the TLM transform can be used like the Laplace or Z transforms to solve differential equations. The method substitutes a calculus model for a respective TLM model. Then, a discrete model in the discrete time domain is obtained to achieve a solution in a stepping routine.

b) A comparison between the TLM method and the ATT modelling method showed that the two techniques give the same results. However, ATT showed sensitivity to small changes in the propagation times. It was also found that the selection of the sample time for the TLM technique becomes crucial when looking for accurate results in models of complex systems. For comparison purposes, it was also noted that for a system model:

   - The parameter ξ of the ATT model is equivalent to the Z_o parameter in the TLM model.
   - The propagation time to be used for the ATT model must be twice the value of the one calculated for the correspondent TLM model.

c) A new TLM model for lumped dynamic behaviour denominated *the modified TLM stub* was developed. This new model improves the convergence and computational processing speed of the original stub algorithm. In this regard, a table that describes the modified TLM transform for integral, differential and partial differential equations was elaborated. The main improvements over the original TLM stub model include a reduction of 40% on the number of mathematical operations, and a reduction of almost 35 % on the mean square modelling method error.

d) Comprehensive transmission line models for the elements of a typical arrangement of a CNC feed drive have been described. All known non-linear functions including geometric and load errors have been included as calculated and identified by the measurements undertaken during the research. Specialised equipment such as laser interferometer, ball bar, electronic levels, artefacts, signal acquisition systems and others were used to obtain parameter data.

e) Generally torsional and axial dynamic behaviours of a shaft are modelled and simulated separately. A model that simulates the torsional and axial dynamics of the screw shaft including the moving nut was derived. In this regard a synchronisation approach between the axial and torsional models was depicted.

f) A TLM model for a CNC single-axis feed drive including a digital controller has been developed. The model was extended to the modelling of two-axis drive of a machining centre including geometric and load errors. This model constitutes the basis for a universal model for the modelling of CNC machine tools including digital drives.

g) The simulation of the single-axis and two-axis models to various feed rates and displacements, including linear and circular interpolation, match well in comparison with the measured response at the machines under study. A maximum percentage error of 2% was estimated for the velocity and current control loop responses.

h) Although simulated results for the position control loop showed a 20% error at maximum on the following error signal (about 10μm on a displacement of 400 mm at 10000 mm/min), the models are considered to produce data useful for the prediction of performance, accuracy and stability associated with the studied drive systems. Nevertheless, a deeper study of the model behaviour is needed to be undertaken in order to improve the model to a higher accuracy.

i) The application of the modified TLM transform and the torsional and axial model's synchronisation approach to the modelling of the single and two-axis feed drives led to a real time implementation of the feed drive models. Results from performance analysis of single, and two-axis real time models was carried out at the end of the project and the results are not included in this study, however initial results shown the feasibility of the developed modelling technique for the implementation of two and three-axis models running on real time.

j) An algorithm for the identification of non-linear systems based on the wavelet amplitude was considered. The technique showed to be effective for the identification of some resonant states but it could not achieve accurate results on the identification of damping factors.

9.1 Contribution to Knowledge

Several areas have been identified as giving a significant contribution to knowledge under the scope of this study:

a) The shortcomings of TLM models for lumped parameter elements have been identified. A new TLM model for lumped dynamic behaviour has been derived as a result of this analysis (see section 3.4).

b) A novel method for the modelling of shafts including the torsional and axial dynamics in the same model has been stated (see section 4.4.9).

c) Geometric errors measured by specialised metrology equipment were clearly demonstrated to be essential for inclusion in two-axis models if realistic contouring accuracy was to be achieved (see sections 5.3 and 7.4).

9.2 Suggestions for Further Work

a) To improve the realism of the TLM model for the single-axis feed drive by inserting the spectral density of the noise measured from the machine.

b) To extend the single and two axis models to the development of multi-axis models including geometric and load errors.

c) To optimise the SIMULINK model of the single-axis drive as a first step towards the development of real time models including parameter identification and auto tuning.

d) To include the cutting forces on the single-axis model to analyse the dynamic behaviour of the system under cutting conditions.

e) To run the optimised model in parallel with the real system in order to detect the cutting force element.

f) A direction to the study of CWT to modal parameter identification of feed drives should be performed in order to develop an algorithm to detect automatically the damping parameters from experimental data.

The main goal of future work is the derivation of algorithms that track fast variations in the optimal parameters despite noise and modelling uncertainties present in most real systems.

REFERENCES

[1] Johns, P.B. & Beurle, R.L. (1971) – *Numerical Solution of 2-Dimensional Scattering Problems Using A Transmission Line Matrix*, Proceedings of IEEE, Vol. 118, pp. 1203-1208.

[2] Layton, R. A. (1998) - *Principles of Analytical Systems Dynamics*. First edition, New York, Springer Verlag.

[3] Mosterman, P.J., Biswas, G. (2000) – *A Comprehensive Methodology for Building Hybrid Models of Physical Systems*, Artificial Intelligence, Vol. 121, pp. 171 – 209.

[4] ISO 230 - 1: *Geometric Accuracy Of Machines Operating Under No-Load Or Finishing Conditions*. ISO Standards 1996.

[5] ISO 230 - 2: *Determination Of Accuracy And Repeatability Of Positioning Of Numerically Controlled Machine Tools*. ISO Standards 1996.

[6] ISO 230-4: *Circular Test For Numerically Controlled Machine Tools*. ISO standards 1996.

[7] Ford, D.G. (1989) - *A CAD-CAE Aid to Design of a Machine Tool System*, PhD thesis, The University of Huddersfield, U.K.

[8] Erkorkmaz, K. & Altintas, Y. (2001) - *High Speed CNC System Design: Part II: Modelling And Identification Of Feed Drives*, International Journal of Machine Tools & Manufacture, Vol. 41, pp. 1487–1509

[9] Ebrahimi, M. & Whalley, R. (2000) – *Analysis, Modelling And Simulation Of Stiffness In Machine Tool Drives*, Computers & Industrial Engineering, Vol. 38, pp. 93-105.

[10] Pislaru, C., Ford, D.G, Freeman J. M. (1999) - *A New Approach to the Modelling and Simulation of a CNC Machine Tool Axis Drive*. Proceedings of International Conference on Laser Metrology and Machine Tool, CMM and Robot Performance LAMDAMAP'99, Newcastle upon Tyne, U.K., Vol. 1, pp. 335 - 343.

[11] Fu, K., Gonzalez, R. S., Lee, C. S. G. (1988) - *Robotics, Sensing, Vision and Intelligence*. McGraw-Hill International Ltd.

[12] Pislaru, C., Ford, D. G., Freeman, J. M. (1999) - *Dynamic Simulation of CNC Machine Tool Axis Drives*. Proceedings of International Conference PCIM '99, Intelligent Motion Section, Nuremberg, Germany, pp. 259 - 264.

[13] Pislaru, C., Ford, D. G., Freeman J. M. (2001) - *Improving CNC Machine Tools Performance by Using Modular Approach*. Proceedings of International Conference on Laser Metrology and Machine Tool, CMM and Robot Performance LAMDAMAP 2001, Birmingham, U.K., pp. 301 - 313.

[14] Pislaru, C., Ford, D. G., Freeman J. M. (1999) - *A New 3D Model for Evaluating the Performance of CNC Machine Tool Axis Drives.* Proceedings of First International Conference of the European Society for Precision Engineering and Nanotechnology EUSPEN'99, Bremen, Germany, Vol. 1, pp. 72 – 75.

[15] Pislaru, C., Ford, D. G., Freeman, J. M. (2002) – *On Modular Modelling of CNC Machine Tool Drives: A Comparison between Simulation and Experimentation.* Proceedings of International Conference on Optimisation of Electrical and Electronic Equipment OPTIM 2002, Brasov, Romania, Vol. II, pp. 559 - 564.

[16] Pislaru, C., Ford, D. G., Holroyd, G. (2002) – *Modelling Non-linearities from CNC Machine Tool Drives.* Proceedings of International Conference on Optimisation of Electrical and Electronic Equipment OPTIM 2002, Brasov, Romania, Vol. II, pp. 553 - 558.

[17] Holroyd, G., Pislaru, C., Ford, D. G. (2001) - *Identification of Damping Elements in a CNC Machine Tool Drive.* Proceedings of International Conference on Laser Metrology and Machine Tool, CMM and Robot Performance LAMDAMAP 2001, Birmingham, U.K., pp. 289 - 301.

[18] Pislaru, C. (2001) – *Parameter Identification and Hybrid Mathematical Modelling Techniques Applied to Non-Linear Control Systems.* Ph.D. Thesis, University of Huddersfield, U.K.

[19] Bartlett, H & Whalley, R. (1988) - *The Response of Distributed-Lumped Parameter Systems.* Proceedings of Institution of Mechanical Engineers, Part C, Journal of Mechanical Engineering Science, Part C, Vol. 202, No. 6, pp. 421-429.

[20] Pislaru, C., Ford, D. G., Holroyd, G. (2004) – *Hybrid Modelling and Simulation of a CNC Machine Tool Axis Drive.* Proceedings of the Institution of Mechanical Engineers , Part I, Journal of Systems and Control Engineering, Vol. 218, No. 2, pp. 111-120.

[21] Holroyd, G., Pislaru, C., Ford D. G. (2003) – *Determination of Stiffness and Damping Sensitivity for Computer Numerically Controlled Machine Tool Drives.* Proceedings of Institution of Mechanical Engineers, Part C, Journal of Mechanical Engineering Science, Vol. 217, pp. 1165 – 1177.

[22] Holroyd, G., Pislaru, C., Ford, D. G. (2003) – *Modelling the Dynamic Behaviour of a Ball-screw System Taking into Account the Changing Position of the Ball-screw Nut.* Proceedings of International Conference on Laser Metrology and Machine Tool, CMM and Robot Performance LAMDAMAP 2003, Huddersfield, U.K., pp. 337-348.

[23] Sadiku, M.N.O., Agba, L.C. (1990) - *A Simple Introduction to the Transmission Line Modeling,* IEEE Transactions on Circuits and Systems, Vol. 31, No. 8, pp 991-999

[24] Auslander, D.M. (1968) - *Distributed Systems Simulation with Bilateral Delay Line Models,* Transactions ASME, Vol. 80, pp 195-200.

[25] Christopoulos, C. (1995). *The Transmission Line Modeling Method (TLM).* 1st edition. New York. The IEEE/OUP Series on Electromagnetic Wave Theory.

[26] Boucher, R.F., Kitsios, E.E. (1986) - *Simulation of Fluid Network Dynamics by Transmission Line Modelling,* Proceedings of Institution of Mechanical Engineers, Part C, Journal of Mechanical Engineering Science, Vol. 200, No.1, pp. 21-29.

[27] Kitsios, E.E., Boucher, R.F. (1986) - *Transmission Line Modelling of a Hydraulic Position Control System,* Proceedings of Institution of Mechanical Engineers, Part B, Vol. 200, No. 4, pp. 229-236.

[28] Beck, S.M., Haider, H, Boucher, R.F. (1995) - *Transmission Line Modelling Of Simulated Drill Strings Undergoing Water Hammer,* Proceedings of Institution of Mechanical Engineers, Part C, Journal of Mechanical Engineering Science Vol. 209, pp. 419-427.

[29] Partridge, G J, Christopoulos, C, and Johns, P B (1987) - *Transmission Line Modelling of Shaft System Dynamics,* Proceedings of Institution of Mechanical Engineers, Part C, Journal of Mechanical Engineering Science, Vol. 210, No. 4, pp. 271-278.

[30] Hui, S.Y.R., Christopoulos, C. (1990) - *Numerical simulation of power circuits using transmission line modelling,* IEE Proceedings A, Vol. 137, No. 6, pp. 379-384.

[31] Hui, S.Y.R., Christopoulos, C. (1991) - Discrete Transform Technique For Solving Coupled Integro-Differential Equations In Digital Computers, IEE Proceedings-A, Vol. 138, No. 5, pp. 273–280.

[32] Stubbs, D.M., Pulko, S.H, Wilson, B. (1995) - *Extension Of The Transmission Line Matrix (TLM) Method In Its Applications To Lumped Networks,* IEE Electronic letters, Vol. 31, No. 21, pp. 1849-1851

[33] Murtonen, C.N. & Lowery, A.J. (1995) - *Multiport Devices In Transmission Line Models Of Lumped Electronic Circuits,* IEE Electronic letters, Vol. 31, No. 14, pp. 1120-1121.

[34] Hui, S.Y.R., Christopoulos, C. (1991) - *Computer Simulation Of Converted-Fed DC Drive Using The Transmission Line Modelling Technique,* IEEE Transactions on Power Electronics, Vol. 10, No. 1, pp. 48-54.

[35] Hui, S.Y.R., Christopoulos, C. (1992) - *Discrete Transform Technique For Solving Non-Linear Circuits And Equations*, IEE Proceedings Part A, Vol. 139, No. 6, pp. 321–328.

[36] Hui, S.Y.R., Christopoulos, C. (1995) - *Modeling Non-Linear Power Electronic Circuits With The Transmission Line Modeling Technique*, IEEE Transactions on Power Electronics, Vol. 10, No. 1, pp. 48-54.

[37] Davis, R.M, Ray, W.F., Jeffers, R. (1989) - *A Practical Evaluation Of Non-Linear Snubbers Using Multi-Layer Ceramic Capacitors*, Proceedings of the third EPE Conference, Vol. 2, pp. 621-625.

[38] Hui, S.Y.R., Zhu, J. (1995) - Numerical Modelling And Simulation Of Hysteresis Effects In Magnetic Cores Using Transmission-Line Modelling And Preisach Theory, IEE Proceedings on Electric Power Applications, Vol. 142, No.1, pp. 57-62.

[39] Krus, P. (1999) - *Modelling Of Mechanical Systems Using Rigid Bodies And Transmission Line Joints*, ASME Journal of Dynamic Systems, Measurement and Control, Vol. 121, No. 4, pp. 606-611.

[40] Fung, K K, Hui, S Y R, Christopoulos, C. (1996) - *Concurrent Programming and Simulation of Decoupled Power Electronic-Circuits*, IEEE Transactions on Energy Conversion. Vol. 11, No 2, pp. 287-297.

[41] Fung, K.K., Hui, S.Y.R., Christopoulos, C. (1996) - *Concurrent Programming And Simulation Of Decoupled Power Electronic Circuits*, IEE Proceedings of Science Measurements Technologies, Vol. 43, No. 2, pp 131-136.

[42] Hui, S.Y.R., Fung, K.K., Zhang, M.Q., Christopoulos, C. (1993) - *Variable Time Step Technique For Transmission Line Modelling*, IEE Proceedings A, Vol. 140, No. 4, pp. 299-302.

[43] Tenorio de Carvalho, C.A., de Menezes, L.R.A.X., Melo, V.H.C. (2003) - *Accelerating Bidimensional TLM Using Adjustable Time Steps*, IEE Electronic Letters on line, Vol.39, No.25, pp. 1825-1826.

[44] Fung K.K., and Hui, S.Y.R. (1996) - *Improved TLM Link Model For Reactive Circuit Components*, IEE Proceedings-A, Vol. 143, No. 5, pp. 341-344.

[45] Fung K.K., Hui, S.Y.R. (1996) - *Fast Simulation Multistage Power Electronics Systems With Widely Separate Operating Frequencies*, IEEE Transactions on Power Electronics, Vol. 11, No. 3, pp. 405-411.

[46] Deml, C., Turkes, P. (1997) - *Fast Simulation Technique For Power Electronic Circuits With Widely Different Time Constants*, Proceedings of the annual meeting of the IEEE Industrial Applications Society, ISSN:0160-8592, pp. 1204-1210

[47] Sun, J. and Grostollen, H. (1996) - *Symbolic Analysis Of Switching Power Converters Based On General Averaging Method*, Proceedings of the IEEE Power Electronics Specialists Conference, pp 543-549.

[48] White, J and Leeb, S.B. (1991) - *An Envelope-Following Approach To Switching Power Converter Simulation*, IEEE Transactions on Power Electronics, Vol. 6, No. 2, pp. 303-307.

[49] Johansson, B., Krus, P., Palmberg, J.O (2000) - *Distributed Modelling: Object-Oriented Implementation With Modelica And Transmission Lines.* Proceedings of Power Transmission and Motion Control Conference. PTMC 2000, Bath, UK.

[50] Jin, H., Vahldieck, R. (1992) - *The Frequency Domain Transmission Line Matrix – A New Concept*, IEEE Transactions on Microwave Theory and Techniques, Vol. 40, No. , pp 2207-2218.

[51] Johns, D., Christopoulos, C. (1994) - *New Frequency-Domain TLM Method For The Numerical Solution Of Steady-State Electromagnetic Problems*, IEE Proceedings of Science Measurements Technologies, Vol. 141, No. 4, pp. 310-316.

[52] Salama, I., Riad, S. M. (2000) - TFDTLM – *A New Computationally Efficient Frequency-Domain Transmission Line Matrix Method*, IEEE Transactions on Microwave Theory and Techniques, Vol. 48, No. 7, pp. 1089-1097.

[53] Whalley, R., Bartlett, H. (1995) - *Gas Flow In Pipes And Tunnels*, Proceedings of Institution of Mechanical Engineers, Part I, Vol. 209, pp. 41-52.

[54] Whalley, R., Bartlett, H., Ebrahimi, M. (1997) - *Analytical Solution Of Distributed-Lumped Parameter Networks Models,* Proceedings of Institution of Mechanical Engineers, Part I, Vol. 221, pp. 203-218.

[55] Bartlett, H., Whalley, R. (1998) - *Analogue Solution To The Modelling And Simulation Of Distributed-Lumped Parameter Systems,* Proceedings of Institution of Mechanical Engineers, Part I, Vol. 212, pp. 99-114.

[56] Bartlett, H., Whalley, R. (1998) - *Power Transmission System Modelling*, Proceedings of Institution of Mechanical Engineers, Vol. 212, Part C, pp. 497-508.

[57] Bartlett, H., Whalley, R., Rizvi, S. S. I. (1999) - *Hybrid Modelling, Simulation And Torque Control Of A Marine Propulsion System,* Proceedings of Institution of Mechanical Engineers, Part I, Vol. 213, pp. 1-10.

[58] Bartlett, H., Whalley, R. (1998) - *The Rotor Dynamics Of Reeling Machinery*, Applied Mathematics Modelling, No. 23, pp. 757-777.

[59] Farshidianfar, A. (2001) *Optimisation Of Torsional Vibrations In Drive-Line Systems.* PhD Thesis. University of Bradford (UK).

[60] Abdul-Ameer, A.A. (2001). *Mathematical Modelling And Simulation Of Hybrid Mechanical Systems*, PhD Thesis. University of Bradford (UK).

[61] Liang, S.Y., Hecker R.L., Landers L.G. (2004) - *Machining process monitoring and control: the state–of–the–art*, Journal of Manufacturing Science and Engineering, Volume 126, Issue 2, pp. 297-310

[62] Braasch, J. - *Position measurements on machine tools by linear encoder or ball screw and rotary encoder.* [On-line]: Available: www.heidenhain.com/posmeas.html Accessed: 24 August 2003.

[63] Degenova, J. (1999) - *What you ought to know about ball screws*, Machine Design Magazine, Penton Media publishers, pp 57-60.

[64] Danaher Motion – *Servo handbook: Drive control modes.* [On-line]: Available: www.motionvillage.com/training/handbook/drive/servocontrol/drivecontrolmodes.html Accessed: 28 August 2003.

[65] Galil Motion Control Inc. (2000) - *Motion Control*, Machine Design Magazine, Penton Media publishers, pp 121-136.

[66] Danaher Motion – *Digital versus analogue control.* [On-line]: Available: www.motionvillage.com/training/handbook/drive/servocontrol/digitalvsanalog.html Accessed: 3 September 2003.

[67] Siemens (2001) - *Basics of General Motion Control - Servomotor construction*, SiemensTechnical Education Program (STEP), [On-line]: Available: www.siemens.com Accessed: 10 September 2003.

[68] Moreton, P. (2000) - *Industrial Brushless Servomotors*, 1st edition, Newnes, Oxford (UK). p.p. 101-115. ISBN: 0 7506 3931 8

[69] Simon, N. (1999) - *Implementation of a speed field oriented control of 3-phase PMSM motor using TMS320F240*, Texas Instruments- Digital control systems, application report SPRA588.

[70] Heidenhain (2000) - *Technical manual TNC 426PB – NC Software 280 472*, Section 4.8: The control loop, pp 166 - 189.

[71] Seborg, D. E, Edgar T.F, and Mellichamp D.A. (1989) - *Process Dynamics and Control*, John Wiley& Sons, New York

[72] Heinemann, G. & Papiernik,W. (1997) - *Direct linear servo drives for machine tool applications*, Intelligent Motion Proceedings. pp. 93-102.

[73] Altintas, Y (2000) – *Manufacturing Automation*, 1st ed. Cambridge University Press, New York , pp 179-203.

[74] Hugh, J. (2002) - *Integration and automation of manufacturing systems - Chapter 13: Motion control*, Grand Valley state University, Michigan.

[75] Koren, Y. (1983) - *Computer control of manufacturing systems*, McGraw Hill, Singapore.

[76] Bullock, T. (2000) - Command generation. [On-line]: Available: www.control.com/ *control_com/Papers/CommandGeneration_html* Accessed: 24 October 2003.

[77] Heidenhain (2002) - *Technical manual TNC 426/CB/PB/N – NC Software 280 476*, Section 6.8: The control loop, pp 109 - 149.

[78] The Math Works Inc. (2002) – *Filter design toolbox for use in MATLAB- Chapter 10: Quantized fitering analysis examples*, Users's Guide version 2. pp 3-11.

[79] Prokop, L. & Grasblum, P. (2002), *3-phase PM synchronous motor vector control using DSP56F80x*, Motorola Literature Distribution, AN1931/D, Denver (USA)

[80] Park, E; Lim, H. and Chong-Ho, C. (2003) - *Position control of X-Y table at velocity reversal using pre-sliding friction Characteristics*, IEEE Transactions on Control Systems Technology, Vol. 11, No. 1: pp 24-31

[81] Armstrong-Helouvry, B., Dupont, P, and Canudas de Wit, C. (1994) - *A survey of models, analysis tools and compensation methods for the control of machines with friction*, Automatica, Vol. 30, No. 7: pp 1083-1138.

[82] Armstrong-Helouvry, B. (1993) - *Stick-slip and control in low-speed motion*, IEEE Transactions on Automatic Control, Vol 38, No. 10: pp 1483-1496.

[83] Dupont, P.E. and Dunlap E.P. (1995) - *Friction modeling and PD compensation at very low velocities*, Transactions of the ASME: Journal of Dynamic Systems, measurement and Control, Vol 117: pp 8-14.

[84] Dupont, P., Hayward, V., and Armstrong, B. (2002) - *Single state elasto-plastic friction models*, IEEE Transactions on Automatic Control, Vol 47, No. 5: pp 787-792.

[85] Haessing, D.A. and Friendland, B. (1991) - *On the modelling and simulation of friction*, Transactions of ASME: Journal of Dynamic Systems, Measurements and Control. Vol 113, No. 9: pp 354-362.

[86] Karnopp, D. (1985) - *Computer simulation of stick-slip friction in mechanical dynamic systems*, Journal of Dynamic Systems, Measurement and Control, *ASME*, Vol. 107, No.1: pp 100-103

[87] Oakley, M. (2001) – *Coupling high performance servos to mechanical loads.* [On-line]: Available: www.ormec.com/motion/csedocs/coupling.htm Accessed: 12 Decemeber 2003.

[88] Stockdale, L.A. (1962) – *Servomechanisms*, Pitman paperbacks. ISBN: 0-273-40490-3 / 0273404903

[89] Robertson, A. – *Non-linear control and servo systems. Lecture 8, March 2004.* [On-line]: Available: Lund Institute of Technology, Sweden: www.control.lth.se/~kursolin/ lectures01/lec08.pdf, Accessed: 16 December 2003.

[90] Kao, J.Y., Yeh, Z.M., Tarng, Y.S. & Lin, Y.S. (1996), - *A study of backlash on the motion accuracy of CNC lathes*, International Journal of Mach. Tools Manufacturing, Vol. 36, No. 5: pp 539-550

[91] INA - *The Tensioning of Lead Screws with Rolling Bearing Supports.* [On-line]: Available: www.ina.com, Accessed: 10 January 2004.

[92] INA (1999) - *Bearings for Screw Drives*, ZAE publications, Herzogenaurach, pp. 11-20

[93] INA - *Bearings basic calculations*, [On-line]: Available: medias.uk.ina.com/medias/ en!hp.tg.cat/tg_rot*CHEBHCFE;aK8borjuC0zf Accessed: 14 January 2004.

[94] HIWIN - *Linear guideways – structure*, [On-line]: Available: www.hiwin.de /internet_uk/start.htm Accessed: 16 January 2004.

[95] NRDC - *Linear Motion Roller Guideways*,[On-line]: Available: www.nrdcindia.com/ pages/linear.htm, Accessed 25 January 2004

[96] INA - *Linear Guideways*, [On-line]: Available: medias.uk.ina.com/medias/ en!hp.tg/15;aK8borjuC0zf ,Accessed: 4 February 2004.

[97] THK - *Caged Ball LM Guide*, [On-line]: Available: www.lmsystem.com/en/products/ index.php Accessed: 10 February 2004.

[98] Mayr - *ROBA-ES*, [On-line]: Catalogue K_940_05_GB_27_11_2003.pdf, Available: mayr.de/english/p_old/sh_coupl/default.htm, Accessed: 20 February 2004.

[99] THK (1999)- *LM System – Ball Screws*, Linear system editorial group, Japan, pp 17-56

[100] Rao, S.S. (1990) – *Mechanical vibrations*, 2nd edition, Addison Wesley, New York, pp. 383-393. ISBN: 0 201 50156 2.

[101] Doebelin, E.O. (1998) – *System Dynamics: Modelling, Analysis, Simulation, Design*, Marcel Dekker, Inc. New York, pp. . ISBN:

[102] Heidenhain, (2003) - *Selection Guide for Incremental Rotary and linear Encoders*, [On-line]: Available: www.heidenhain.com Accessed: 14 March 2004

[103] Staebler, M (2002) – *TMS320F240 DSP-Solution for High-Resolution Position with Sin/Cos-Encoders*, Digital Signal Processing Solutions - Texas Instruments Application Report SPRA496.

[104] The MathWorks Inc. (2002) – *Signal processing toolbox for use in MATLAB*, Users's Guide version 3.

[105] Siemens - Simodrive 611 6SN1123-1AA00-0CA1[On-line]: Available: mall.ad.siemens .com/WW/guest/index.asp?aktPrim=0&nodeID=10008015&mlfb=6SN1123%2D1AA0 0%2D0CA1&aktTab=4&lang=en, Accessed: 20 March 2004.

[106] Siemens - *1FT6082-1AF7-1AG1*, [On-line]: Available: mall.ad.siemens.com/WW/ guest/index.asp?aktPrim=0&nodeID=10023401&mlfb=1FT6082%2D1AF71%2D%2E A%2E1&aktTab=4&lang=en, Accessed: 25 March 2004.

[107] Longstaff, A. (2002) - *Volumetric Accuracy Assessment of Cincinnati Arrow 500 At The University of Huddersfield*, Ultra Precision Engineering Centre, Internal report ESPRC CAPAM project.

[108] Longstaff, A and Fletcher, S. (2003) - *Compensation of Cincinnati Arrow 500 At The University of Huddersfield*, Ultra Precision Engineering Centre, Internal report ESPRC CAPAM project.

[109] Postlethwaite, S.S. (1997) - *Dynamic Calibration of CNC Machine Tools*, International Journal of Machine Tools Manufacturing. Vol. 37, No.3, pp 287 -294

[110] Ford, D.G., Postlethwaite, S.S. and Blake M.D. (1999) - *The identification of non-rigid errors in a vertical machining center*, Proceedings of Institution of Mechanical Engineers, Part B, Vol. 213, pp. 555-566.

[111] Chapman, M - *The benefits of laser systems that use remote interferometer optics for linear, angular and straightness measurements,* [On-line]: Available: www.renishaw.com Accessed: 18 April 2004.

[112] Renishaw – Ballbar system. Available: http://www.renishaw.com/client/product/ UKEnglish/PGP-10.shtml, Accessed: 25 April 2004.

[113] ISO standards (1996) - *ISO 230-2: Determination of accuracy and repeatability of positioning of numerically controlled machine tools.*

[114] HEIDENHAIN (2002), *Technical Manual TNC 426/430*, Section 6.23 Integrated Oscilloscope, pp 781 –786

[115] Siemens – 810-D_Servo_Optimzation.doc: Available: http://mediaibox.siemens.com.br /upfiles/1040.pdf, Accessed: 11 May 2004.

[116] Kullaa, J. (2003) - *Damage Detection Of The Z24 Bridge Using Control Charts*, Journal of Mechanical Systems and Signal Processing, Vol. 17, No. 1, pp.163-170

[117] Andersen, P. (1997) - *Identification of Civil Engineering Structures using Vector ARMA Models*, PhD Thesis, Aalbog University (Denmark).

[118] Drexel, M & Ginsberg, J (2000) - *Modal Parameter Extraction From Structural Diagnostic Signals.* [On-line]: Available: www.me.gatech.edu/diagnostics/C5.PDF Accessed: 18 July 2003

[119] Peeters, B. & De Roeck, G. (1999) - *Reference-Based Stochastic Subspace Identification For Ouput-Only Modal Analysis*, Mechanical Systems and Signal Processing, Vol. 13, No. 6. pp. 855-878.

[120] De Silva, C. W. (2000) - *Vibration – Fundamentals And Practice*, Taylor-Francis, CRC Press, Boca Raton, FL.

[121] Zhang, Z.Y., Hua H.X. and Xu X.Z.; Huang Z (2003) - *Modal Parameter Identification through Gabor Expansion Of Response Signals*, Journal of Sound and Vibration, Vol. 266, pp. 943-955.

[122] Cohen, L (1995) - *Time Frequency Analysis*, Prentice-Hall, Englewood Cliffs, NJ.

[123] Robertson, A.N., Park, K.C. and Alvin, K.F. (1998) - *Extraction Of Impulse Response Data Via Wavelet Transform For Structural System Identification,* Transactions of ASME, Vol. 120, pp. 252-260.

[124] Roberson, A.N., Park, K.C. and Alvin, K.F. (1998) - *Identification Of Structural Dynamics Models Using Wavelet-Generated Impulse Response Data,* Transactions of ASME, Vol. 120, pp. 261-266.

[125] Ruzzene, A, Fasana, L, Garibaldi, L. and Piombo, B. (1997) - *Natural Frequencies and Dampings Identification Using Wavelet Transform: Application To Real Data,* Mechanical Systems and Signal Processing, Vol. 11, No. 11, pp. 207-218.

[126] Kijewski, T., Kareem, A. (2003) - *Wavelet Transforms For System Identification In Civil Engineering,* Computer-Aided Civil and Infrastructure Engineering, Vol. 18, pp. 339-355.

[127] Staszewski, W.J. (1997) - *Identification Of Damping In MDOF Systems Using Time-Scale Descomposition*, Journal of Sound and Vibration, Vol. 203, No. 2, pp 283-305.

[128] Staszewski, W.J. (1998) - *Identification Of Non-Linear Systems Using Multi-Scale Ridges And Skeletons Of The Wavelet Transform*, Journal of Sound and Vibration, Vol. 214, No. 4, pp. 639-658.

[129] Carmona, R.A, Hwang, W.L., Torresani, B. (1997) - *Characterization of Signals by the Ridges of Their Wavelet Transforms*, IEEE Transactions on Signal Processing, Vol. 45, No.10, pp. 2586 – 2590.

[130] Delprat, N. Escudie, B., Guillemain, P., Kronland-Martinet, R, Tchamitchian, Ph., and Torrésani, B (1992) - *Asymptotic Wavelet and Gabor Analysis: Extraction of InstantaneousFrequencies,* IEEE Transactions on Information Theory, Vol. 38, No. 2, pp. 644-664.

[131] Pislaru, C, Freeman, J.M, Ford, D.G. (2003) - *Modal Parameter Identification for CNC Machine Tools Using Wavelet Transform*', International Journal of Machine Tools & Manufacture, Vol. 43, pp. 987-993.

[132] Pislaru, C., Ford, D. G., Freeman, J. M. (2003) *Identification of Modal Parameters Affecting the Dynamic Performance of a CNC Machine Tool.* Proceedings of International Conference on Laser Metrology and Machine Tool, CMM and Robot Performance LAMDAMAP 2003, Huddersfield, U.K., pp. 161-170.

[133] Barlow, J. (2002) *SIMODRIVE 611D / SINUMERIK 840D (Software version 6.2.06 for 840D)Training course.* Siemens U.K. pp. 1- 22

[134] Bruce, A., Dohono, D, and Ye Gao, H. (1996), *Wavelet Analysis*, IEEE Spectrum, Vol. 33, No. 10: pp 26-35

[135] Pawlak, M. and Hasiewicz, Z. (1998). *Non-linear System Identification by the Haar Multiresolution Analysis*, IEEE Transactions on Circuits and Systems-I: Fundamental Theory and Applications, Vol. 45, No. 9: pp 945-961

[136] Liu, G.P., Billings, S.A., and Kadirkamanathan, V. (1998), *Non-linear System Identification Using Wavelet Networks*, UKACC International Conference on Control, No. 455, Vol. 2, No. 455: pp 1248-1253

[137] Rioul, O. and Duhamel, P. (1992), *Fast Algorithms for Discrete and Continuous Wavelet Transforms*. IEEE Trans. On Information Theory, Vol. 38, No. 2, pp. 569-586

[138] Ford, D.G. (1995), *Machining to microns: Error avoidance or compensation?* Proceedings of the International Conference on Laser Metrology and Machine Performance (LAMDAMAP 95) pp. 277 – 286

[139] Breitholtz, C., Molander, M., and Navarro-Adlemo, R. (1991) - *Space and time continuous lumped transmission line model*, IEE Proceedings G, Vol. 138, No. 6: pp 661-670.

[140] Singer, S. and Shmilovitz, D. (1995) - *Transformer matrix of some transmission lines topologies*, IEE Proceedings Circuits, Devices and Systems, Vol. 142, No. 1: pp 21-26.

[141] Jeffrey, A. (2003) - *Applied Partial Differential Equations- An Introduction*. 1st edition. Academic Press. Massachusetts, pp. 30-31.

[142] Mallat, S. G. (1999), *A wavelet tour of signal processing*. 2nd edition. Academic Press, San Diego, USA. pp 79 -102

[143] Fasana, L., Garibaldi, L., Giorcclli, E. and Marchcsiollo, S. (1998), *A road bridge dynamic response analysis by wavelet and other estimation techniques*. Third International Conference on Acoustical and Surveillance Methods, pp. 1-9

[144] Lardies, J. and Gouttebroze, S. (2002), *Identification of modal parameters using the wavelet transform*. Intern. Journal of Mechanical Sciences. Vol. 44. pp 2263-2283.

APPENDIX A TRANSMISSION LINE MODEL

A.1 Differential Equation for a Transmission Line

A transmission line is an arrangement of a pair of parallel wires on which electric energy is transmitted. This process is studied looking at the voltage difference between the wires, $e(x,t)$, and the current, $i(x,t)$, of the transmission line at an arbitrary distance, x, from the source terminal, $e_s(t)$, at the time $t > 0$, as shown in Figure A.1.

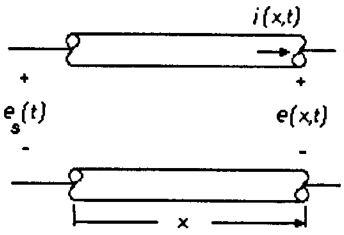


Figure A.1. A transmission line [139]

This transmission line is analysed by an element of transmission line of length Δx. Its equivalent electrical circuit is presented in Figure A.2.

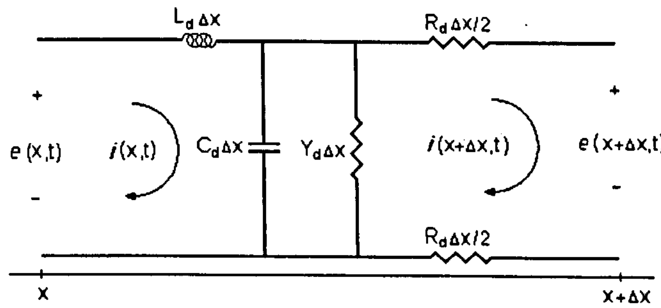


Figure A.2. Element of the transmission line of length Δx [140]

Here, R_d, L_d, Y_d, and C_d are the characteristic resistance, inductance, conductance, and capacitance per unit length of the line (These parameters are considered to be constant).
Applying Kirchhoff's laws at a time t to the element of transmission line shown in Figure A.2 situated at a position x, followed by the cancellation of Δx, shows that in the limit as $\Delta x \to 0$

$$\frac{\partial}{\partial t} L_d i(x,t) + R_d i(x,t) + \frac{\partial}{\partial x} e(x,t) = 0 \qquad \text{(A.1)}$$

$$C_d \frac{\partial}{\partial t} e(x,t) + Y_d * e(x,t) + \frac{\partial}{\partial x} i(x,t) = 0 \qquad \text{(A.2)}$$

Eliminating either i or e, the following second-order constant coefficient PDE is obtained

$$\frac{\partial^2}{\partial x^2} y(x,t) = L_d C_d \frac{\partial^2}{\partial t^2} y(x,t) + (R_d C_d + Y_d L_d) \frac{\partial}{\partial t} y(x,t) + R_d Y_d y(x,t) \qquad \text{(A.3)}$$

Setting $\chi = \frac{1}{\sqrt{L_d C_d}}$, $\delta = \frac{R_d}{L_d}$, and $\eta = \frac{Y_d}{C_d}$, and rearranging equation (A.3) gives

$$\chi^2 \frac{\partial^2}{\partial x^2} y(x,t) = \frac{\partial^2}{\partial t^2} y(x,t) + (\delta + \eta)\frac{\partial}{\partial t} y(x,t) + (\delta\eta) y(x,t) \qquad \text{(A.4)}$$

Where *y(x,t)* is replaced by either *e(x,t)* or *i(x,t)*. This PDE is known as the **Telegrapher's equation,** because it first arose when determining the current and voltage distribution along telegraph landlines. Ignoring certain parameters in equation (A.3), leads to the following special cases [23]:

Elliptic partial differential equation (Poisson's equation)

$L_d = C_d = 0$ and $\kappa_1 = R_d Y_d$ yields

$$\frac{\partial^2}{\partial x^2} y(x,t) = \kappa_1 y(x,t) \qquad \text{(A.5)}$$

- Parabolic partial differential equation (the diffusion equation)

  $R_d = C_d R_d$ or $Y_d = L_d = 0$, and $\kappa_2 = Y_d L_d$ or $R_d C_d$, then

$$\frac{\partial^2}{\partial x^2} y(x,t) = \kappa_2 \frac{\partial}{\partial t} y(x,t) \qquad \text{(A.6)}$$

- Hyperbolic partial differential equation (Helmholtz equation, or simply the wave equation) - $R_d = Y_d = 0$ (Loss-less line) and $\kappa_3 = L_d C_d$ yields

$$\frac{\partial^2}{\partial x^2} y(x,t) = \kappa_3 \frac{\partial^2}{\partial t^2} y(x,t) \qquad \text{(A.7)}$$

Thus, the element of transmission line can be used, under certain conditions, to model problems involving an elliptic, parabolic, or hyperbolic partial differential equation.

A.2 Analytical Solution of the Telegrapher's Equation

Assuming that equation (A.4) possesses a Laplace transform with respect to time, and the initial conditions are zero

$$\chi^2 \frac{\partial^2}{\partial x^2} y(x,s) = s^2 y(x,s) + s(\delta + \eta) y(x,s) + (\delta\eta) y(x,s) \qquad \text{(A.8)}$$

Rearranging equation (A.8) gives

$$\frac{\partial^2}{\partial x^2} y(x,s) = \frac{1}{\chi^2}(s^2 + s(\delta + \eta) + \delta\eta) y(x,s) \qquad \text{(A.9)}$$

If $\quad \gamma^2 = \frac{1}{\chi^2}(s^2 + s(\delta+\eta) + \delta\eta)$ or $\gamma^2 = \frac{1}{\chi^2}(s+\delta)(s+\eta) \qquad$ (A.10)

Hence, $$\frac{d^2}{dx^2}y(x,s)=\gamma^2 y(x,s) \tag{A.11}$$

Applying the boundary conditions, the solutions of (A.11) can be expressed in matrix form as [141]

$$\begin{bmatrix} e(x,s) \\ i(x,s) \end{bmatrix} = \begin{bmatrix} \cosh(\gamma x) & -Z_0 \sinh(\gamma x) \\ -\dfrac{\sinh(\gamma x)}{Z_0} & \cosh(\gamma x) \end{bmatrix} \begin{bmatrix} e(0,s) \\ i(0,s) \end{bmatrix} \tag{A.12}$$

Where γ and Z_0 are named, respectively, the *propagation function* and the *characteristic impedance* of the transmission line

$$\gamma=\sqrt{L_d C_d (s+\delta)(s+\eta)} \tag{A.13}$$

$$Z_o=\sqrt{\frac{L_d}{C_d}}\sqrt{\frac{s+\delta}{s+\eta}} \tag{A.14}$$

Two special cases, leading to a real value of Z_0, can be directly analysed:

- Case 1: $R_d = G_d = 0$ (Whalley et al approach [55]), thus

$$\gamma=s\sqrt{L_d C_d} \qquad Z_0=\sqrt{L_d / C_d} \tag{A.15}$$

- Case 2: $\delta = \eta$ (Abdul-Ameer approach [60]), thus

$$\gamma=(s+b)\sqrt{L_d C_d} \qquad Z_0=\sqrt{L_d / C_d} \tag{A.16}$$

As it can be seen, case 1 is equivalent to case 2 when $\delta = \eta = 0$. For illustration purposes, only case 2 is presented following Whalley approach ($\xi = Z_0$).

If the output of the j component is the input to the next component and considering x the length of the j component ($x = l_j$), then

$$e_j(l_j,s)=e_{j+1}(0,s)=e_{j+1}(s) \qquad i_j(l_j,s)=i_{j+1}(0,s)=i_{j+1}(s) \tag{A.17}$$

$$\begin{bmatrix} e_{j+1}(s) \\ e_j(s) \end{bmatrix} = \begin{bmatrix} -\xi_j ctnh(\gamma_j l_j) & \xi_j \operatorname{csc}h(\gamma_j l_j) \\ -\xi_j \operatorname{csc}h(\gamma_j l_j) & \xi_j ctnh(\gamma_j l_j) \end{bmatrix} \begin{bmatrix} i_{j+1}(s) \\ i_j(s) \end{bmatrix} \tag{A.18}$$

The propagation frequency for each section of line is expressed in terms of the line with the greatest propagation frequency in either the discrete time or fictitious frequency domain. So, the round trip time for the waves on the transmission line segment j may be assumed to be

$$\Delta t_j = 4 l_j \sqrt{L_j C_j} \tag{A.19}$$

Whalley [19] showed that the 'basic' line $\exp^{s\Delta t_j/2} = z_j$ generates frequencies high enough to construct every other wave from integer multiples of it. Hence, for a delay representation the propagation time is expressed as

$$\tau_j = \Delta t_j / 2 \tag{A.20}$$

τ_j and hence $z_j = exp^{g^{*}s}$ are independent variables, then

$$\beta_j = \exp^{\tau_j \delta} \tag{A.21}$$

$$ctnh(\gamma_j l_j) = w_j = (\beta_j z_j + 1)/(\beta_j z_j - 1) \tag{A.22}$$

$$\operatorname{csc}h(\gamma_j l_j) = \sqrt{(ctnh^2(\gamma_j l_j) - 1} \tag{A.23}$$

$$\operatorname{csc}h(\gamma_j l_j) = \sqrt{w^2 - 1} \tag{A.24}$$

Equation (A.18) becomes

$$\begin{bmatrix} e_j(w_j) \\ e_{j+1}(w_{j+1}) \end{bmatrix} = \begin{bmatrix} \xi_j w_j & -\xi_j\sqrt{w_j^2 - 1} \\ \xi_j\sqrt{w_j^2 - 1} & -\xi_j w_j \end{bmatrix} \begin{bmatrix} i_j(w_j) \\ i_{j+1}(w_{j+1}) \end{bmatrix} \tag{A.25}$$

And in a transfer matrix or impedance form:

$$\begin{bmatrix} i_j(w_j) \\ i_{j+1}(w_{j+1}) \end{bmatrix} = \begin{bmatrix} \xi_j^{-1} w_j & -\xi_j^{-1}\sqrt{w_j^2 - 1} \\ \xi_j^{-1}\sqrt{w_j^2 - 1} & -\xi_j^{-1} w_j \end{bmatrix} \begin{bmatrix} e_j(w_j) \\ e_{j+1}(w_{j+1}) \end{bmatrix} \tag{A.26}$$

The Equation (A.25) could be written in z domain:

$$e_j(z_j) = \xi_j \left(\frac{\beta_j z_j + 1}{\beta_j z_j - 1} \right) i_j(z_j) - 2\xi_j \left(\frac{\beta_j^{1/2} z_j^{1/2}}{\beta_j z_j - 1} \right) i_{j+1}(z_{j+1}) \tag{A.27}$$

$$e_{j+1}(z_{j+1}) = 2\xi_j \left(\frac{\beta_j^{1/2} z_j^{1/2}}{\beta_j z_j - 1} \right) i_j(z_j) - \xi_j \left(\frac{\beta_j z_j + 1}{\beta_j z_j - 1} \right) i_{j+1}(z_{j+1}) \tag{A.28}$$

In delay form

$$e_j(z_j^{-1}) = \xi_j \left(\frac{1 + \beta_j^{-1} z_j^{-1}}{1 - \beta_j^{-1} z_j^{-1}} \right) i_j(z_j^{-1}) - 2\xi_j \left(\frac{\beta_j^{-1/2} z_j^{-1/2}}{1 - \beta_j^{-1} z_j^{-1}} \right) i_{j+1}(z_{j+1}^{-1}) \tag{A.29}$$

$$e_{j+1}(z_{j+1}^{-1}) = 2\xi_j \left(\frac{\beta_j^{-1/2} z_j^{-1/2}}{1 - \beta_j^{-1} z_j^{-1}} \right) i_j(z_j^{-1}) - \xi_j \left(\frac{1 + \beta_j^{-1} z_j^{-1}}{1 - \beta_j^{-1} z_j^{-1}} \right) i_{j+1}(z_{j+1}^{-1}) \tag{A.30}$$

By applying the same treatment to the admittance from Equation (A.26), it yields:

$$i_j(z_j^{-1}) = \xi_j^{-1} \left(\frac{1 + \beta_j^{-1} z_j^{-1}}{1 - \beta_j^{-1} z_j^{-1}} \right) e_j(z_j^{-1}) - 2\xi_j^{-1} \left(\frac{\beta_j^{-1/2} z_j^{-1/2}}{1 - \beta_j^{-1} z_j^{-1}} \right) e_{j+1}(z_{j+1}^{-1}) \tag{A.31}$$

$$i_{j+1}(z_{j+1}^{-1}) = 2\xi_j^{-1} \left(\frac{\beta_j^{-1/2} z_j^{-1/2}}{1 - \beta_j^{-1} z_j^{-1}} \right) e_j(z_j^{-1}) - \xi_j^{-1} \left(\frac{1 + \beta_j^{-1} z_j^{-1}}{1 - \beta_j^{-1} z_j^{-1}} \right) e_{j+1}(z_{j+1}^{-1}) \tag{A.32}$$

It is to be noted that $\beta = 1$ when $\delta = 0$ (case 1, loss-less line).

APPENDIX B MODELLING EXAMPLE

B.1 Model Using the Analogue Transform Technique

Motor		
J_m	Inertia	0.49 kg m^2
b_m	Motor damping (windage and bearing friction)	0.75 N m s/rad
T_0	Input torque	250 N m
Front bearing		
b_1	Damping (bearing friction)	0.25 N m s/rad
Shaft 1 (1st rotor drive) and shaft 2 (2nd rotor drive)		
l_1	Length	1 m
d_1	Outside diameter	0.05 m
J_1	Inertia	$6.1359x10^{-7}$ Kg m^2
Rotor Shell		
l_2	Length	8 m
d_o	Outside diameter	0.2 m
d_i	Inside diameter	0.16 m
J_2	Inertia	$9.2739x10^{-5}$ Kg m^2
Rear bearing (final termination)		
b_2	Damping (bearing friction)	0.25 N m s/rad
General		
ρ	Density	7800 kg/m^3
G	Modulus of Rigidity	$80x10^9$ N/m^2

Table B.1 Rotor shell parameters

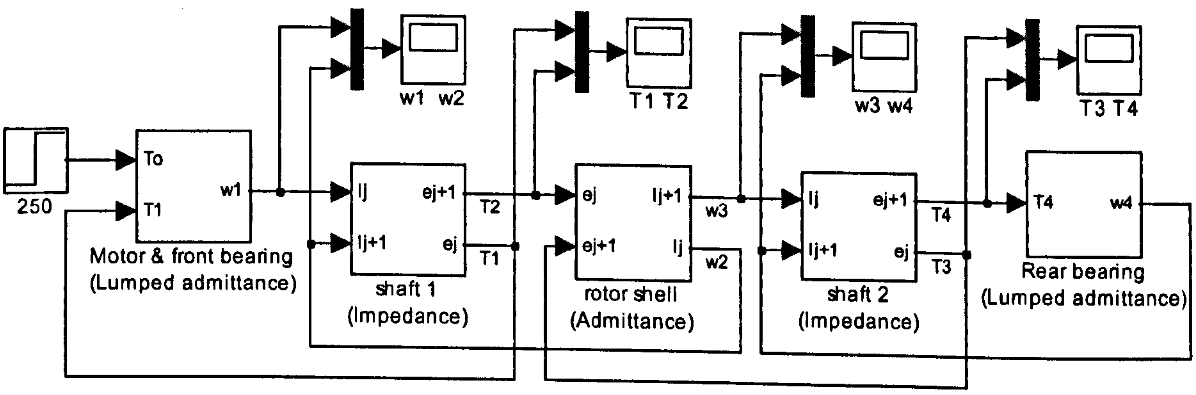


Figure B.1 ATT rotor shell model in SIMULINK

Model parameters for the first and second shaft are:

$$L_1 = J_1\rho = 6.1359x10^{-7} * 7800 = 4.786x10^{-3} \quad (B.1)$$

$$C_1 = \frac{1}{GJ_1} = \frac{1}{80x10^9 * 6.1359x10^{-7}} = 20.372x10^{-6} \quad (B.2)$$

$$\xi_1 = \sqrt{L_1 / C_1} = \sqrt{4.786x10^{-3} / 20.372x10^{-6}} = 15.328 \quad (B.3)$$

$$\tau_1 = 2l_1\sqrt{L_1C_1} = 2*1*\sqrt{4.786x10^{-3}*20.372x10^{-6}} = 0.6245x10^{-3} \tag{B.4}$$

Model parameters for the rotor shell are:

$$L_2 = J_2\rho = 9.2739x10^{-5}*7800 = 0.7233642 \tag{B.5}$$

$$C_2 = \frac{1}{GJ_2} = \frac{1}{80x10^9*9.2739x10^{-5}} = 1.3478x10^{-7} \tag{B.6}$$

$$\xi_2 = \sqrt{L_2/C_2} = \sqrt{0.7233642/1.3478x10^{-7}} = 2316.6194 \tag{B.7}$$

$$\tau_2 = 2l_2\sqrt{L_2C_2} = 2*8*\sqrt{0.7233642*1.3478x10^{-7}} = 4.996 \tag{B.8}$$

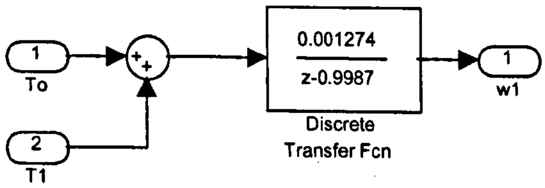


Figure B.2 Motor & front bearing block (ATT Model)

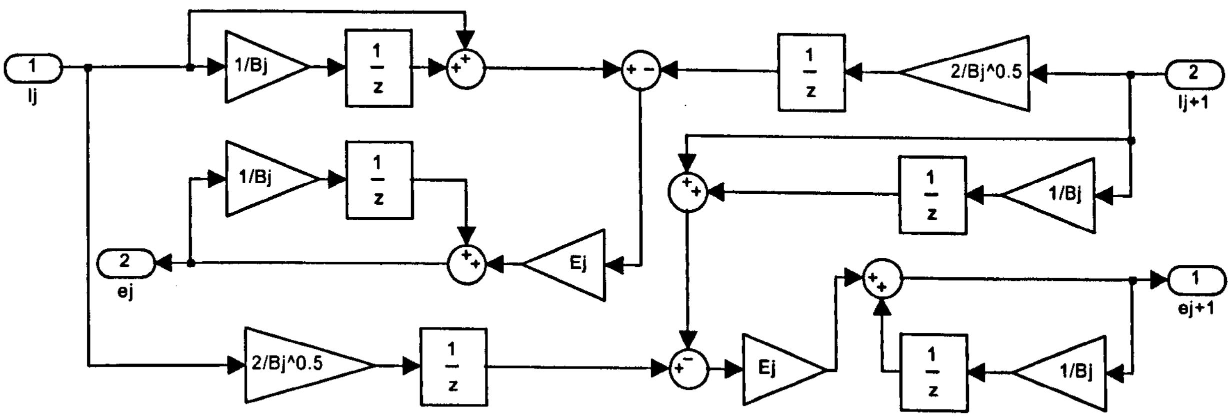


Figure B.3 Shaft 1 and shaft 2 block (ATT Model)

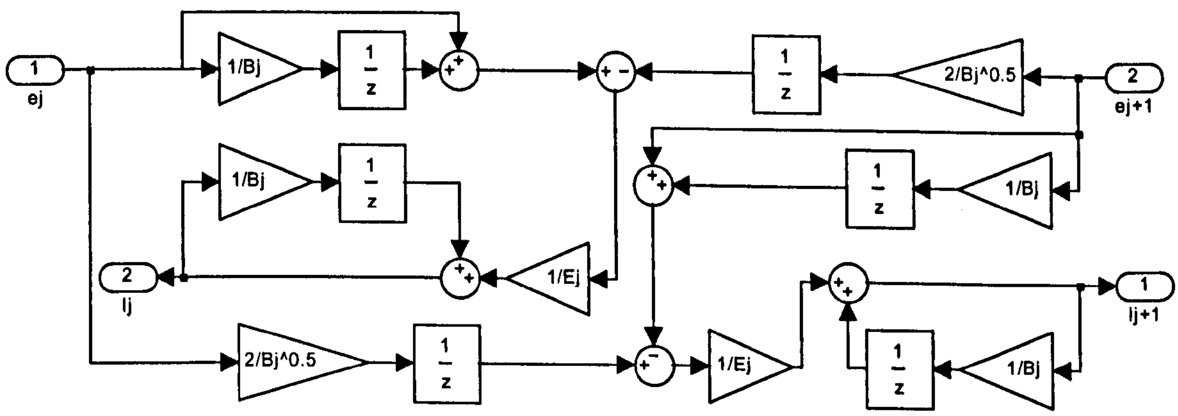


Figure B.4 Rotor shell block (ATT Model)

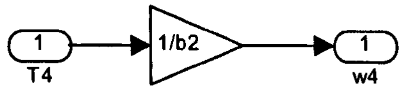


Figure B.5 Rear bearing block (ATT Model)

B.2 Model Using the Transmission Line Matrix Method

The TLM model for the rotor shell is presented in Figure B.6. Figure B.7 shows the implementation of the TLM model in SIMULINK. The TLM parameters for each segment will be:

$$Z_1 = J_1\sqrt{\rho G} = 6.1359x10^{-7} * \sqrt{7800*80x10^9} = 15.3274 \quad \text{(B.9)}$$

$$Z_2 = J_2\sqrt{\rho G} = 9.2739x10^{-5} * \sqrt{7800*80x10^9} = 2316.64 \quad \text{(B.10)}$$

$$\Delta t = l_{seg}\sqrt{G/\rho} = 1*\sqrt{7800/80x10^9} = 0.31225x10^{-3} \text{ [s]} \quad \text{(B.11)}$$

$$Z_m = J_m/(\Delta t/2) = 0.49/(0.31225x10^{-3}/2) = 3138.51 \quad \text{(B.12)}$$

The equations for the TLM model are:

$$\omega_1(k) = M_{w1}(T_o(k) - 2E_m^i(k) - 2A_1^i(k)) \quad \text{(B.13)}$$

where, $$M_{w1} = \frac{1}{b_m + Z_m + b_1 + Z_1} = \frac{1}{0.75 + 3138.51 + 0.25 + 15.3274} = 3.1697x10^{-4} \quad \text{(B.14)}$$

$$e_m(k) = \omega_1(k)Z_m + 2E_m^i(k) \quad \text{(B.15)}$$

$$E_m^i(k+1) = -(e_m(k) - E_m^i(k)) = -\omega_1(k)Z_m - E_m^i(k) \quad \text{(B.16)}$$

$$B_1^i(k+1) = A_1^i(k) + \omega_1(k)Z_1 \quad \text{(B.17)}$$

$$T_1(k) = 2A_1^i(k) + \omega_1(k)Z_1 \quad \text{(B.18)}$$

$$\omega_{21}(k) = M_{w21}(B_1^i(k) - A_{21}^i(k)) \quad \text{(B.19)}$$

where, $$M_{w21} = \frac{2}{Z_1 + Z_2} = \frac{2}{15.3274 + 2316.62} = 8.57652*10^{-4} \quad \text{(B.20)}$$

$$A_1^i(k+1) = B_1^i(k) - \omega_{21}(k)Z_1 \quad \text{(B.21)}$$

$$B_{21}^i(k+1) = A_{21}^i(k) + \omega_{21}(k)Z_2 \quad \text{(B.22)}$$

$$T_2(k) = 2A_{21}^i(k) + \omega_{21}(k)Z_2 \quad \text{(B.23)}$$

$$B_j^i(k+1) = B_{j-1}^i(k) \quad \text{for } j = 22,23,\ldots, 28 \quad \text{(B.24)}$$

$$A_j^i(k+1) = A_{j+1}^i(k) \quad \text{for } j=21,22,\ldots, 27 \quad \text{(B.25)}$$

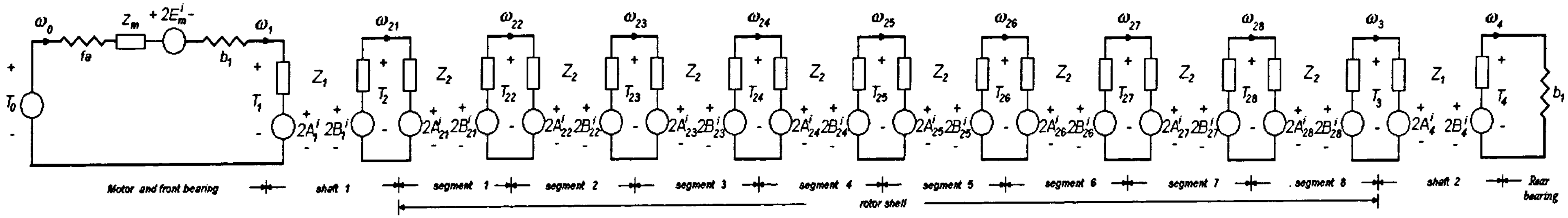


Figure B.6 Rotor shell TLM model

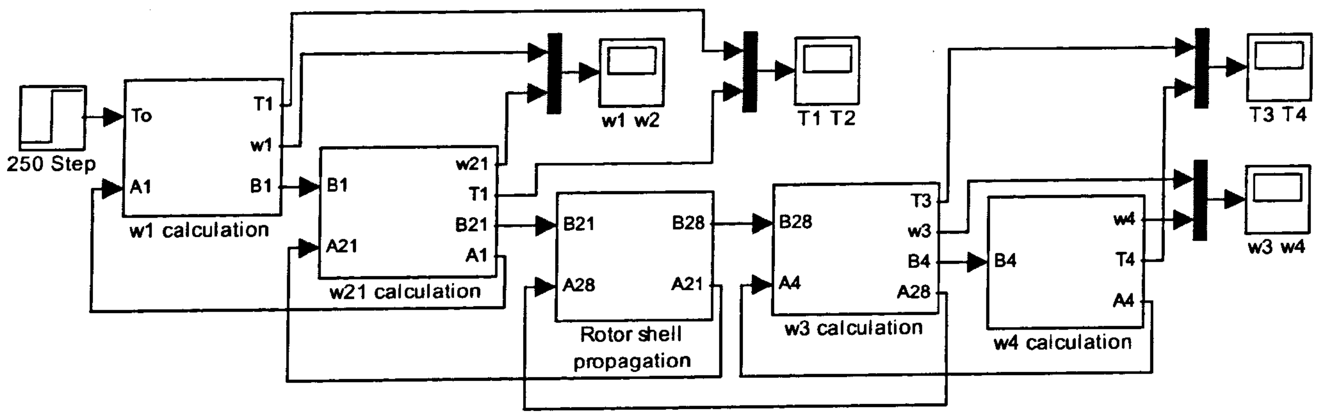


Figure B.7 TLM Rotor shell model in SIMULINK

$$\omega_3(k) = M_{w3}(B_{28}^i(k) - A_4^i(k)) \tag{B.26}$$

where,
$$M_{w3} = \frac{2}{Z_2 + Z_1} = \frac{2}{2316.62 + 15.3274} = 8.57652 * 10^{-4} \tag{B.27}$$

$$A_{28}^i(k+1) = B_{28}^i(k) - \omega_3(k)Z_2 \tag{B.28}$$

$$B_4^i(k+1) = A_4^i(k) + \omega_3(k)Z_1 \tag{B.29}$$

$$T_3(k) = 2A_4^i(k) + \omega_3(k)Z_1 \tag{B.30}$$

$$\omega_4(k) = M_{w4}B_4^i(k) \tag{B.31}$$

where,
$$M_{w4} = \frac{2}{Z_1 + b_2} = \frac{2}{15.3274 + 0.25} = 0.12839 \tag{B.32}$$

$$A_4^i(k+1) = B_4^i(k) - \omega_4(k)Z_1 \tag{B.33}$$

$$T_4(k) = \omega_1(k)b_2 \tag{B.34}$$

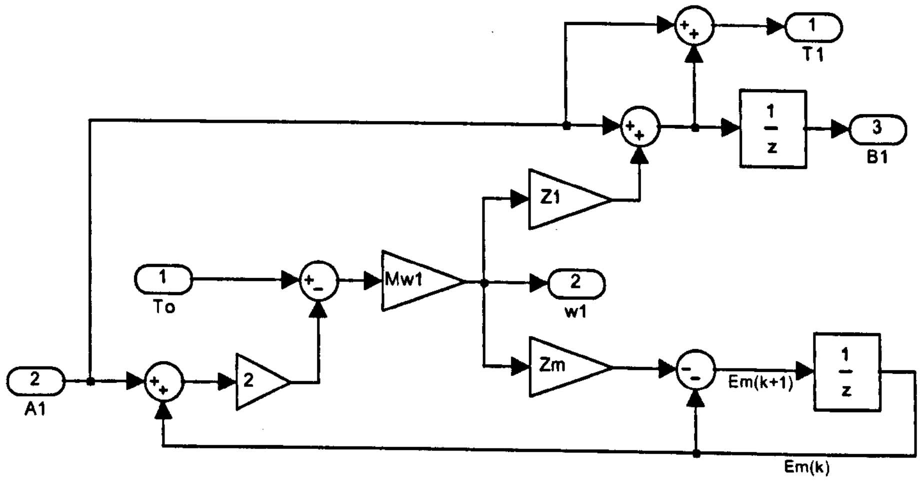


Figure B.8 ω_1 calculation block (TLM model)

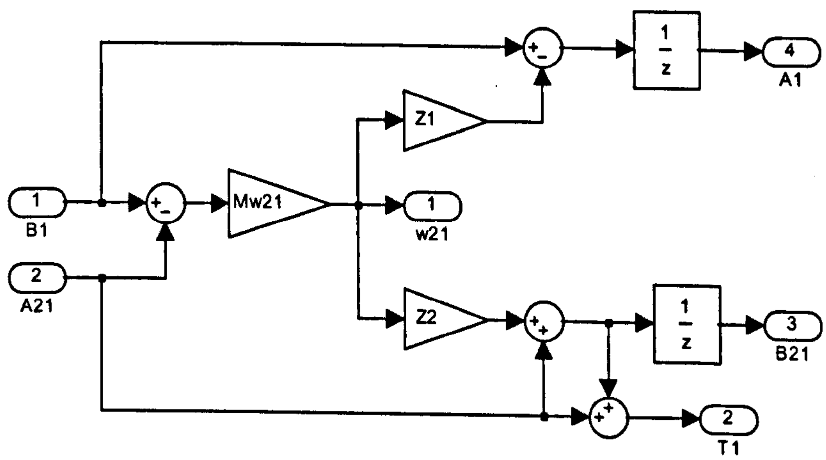


Figure B.9 ω_{21} calculation block (TLM model)

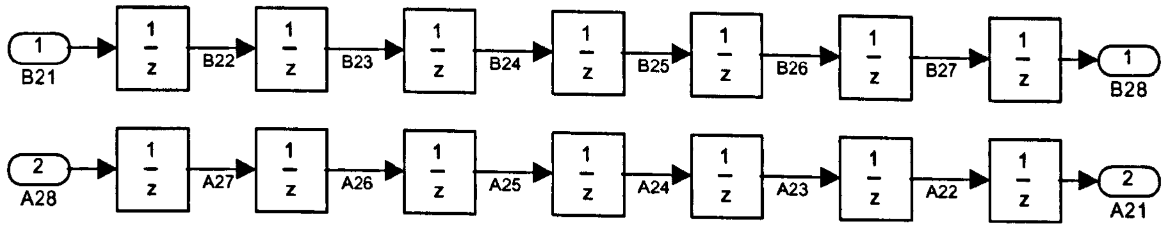


Figure B.10 Rotor shell propagation block (TLM model)

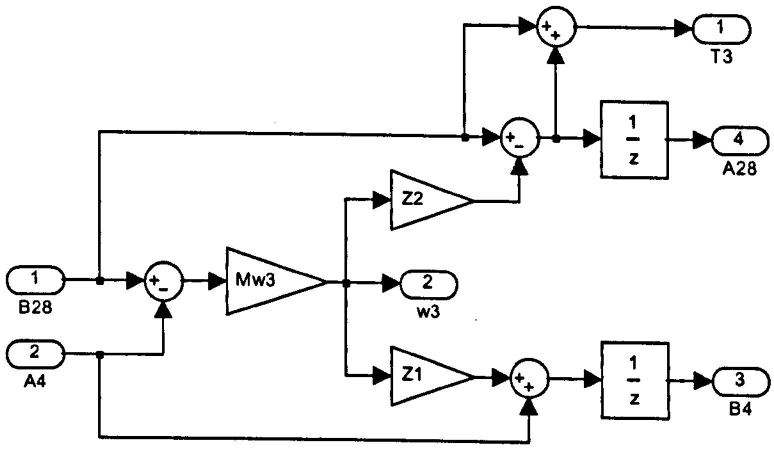


Figure B.11 ω_3 calculation block (TLM model)

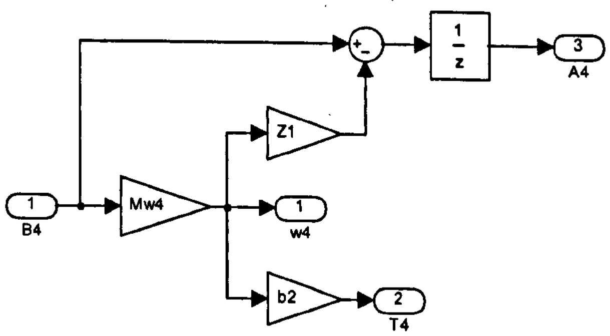


Figure B.12 ω_4 calculation block (TLM model)

B.3 Analysis of the TLM Model for the Differential Term

T_1 in Figure 3.22a is assumed zero for simplicity of the analysis. The equation for the mechanical dynamics of the motor becomes:

$$T_0(k) - \omega_0(k)(b_m + b_1) = e_m(k) \tag{B.35}$$

where,

$$e_m(k) = \omega_0(k) Z_m + 2E_m^i(k) \tag{B.36}$$

$$Z_m = J_m / (\Delta t / 2) \tag{B.37}$$

$$E_m^i(k+1) = -(e_m(k) - E_m^i(k)) \tag{B.38}$$

The angular velocity ω_0 is calculated from equations (B.35 and B.36) as

$$\omega_0(k) = (T_0(k) - 2E_m^i(k))/(Z_m + b_m + b_1) \qquad \text{(B.39)}$$

Replacing parameters f_a and b_m in equation (B.40) by the amounts specified in Appendix B.1 gives:

$$\omega_0(k) = (T_0(k) - 2E_m^i(k))/(Z_m + 1) \qquad \text{(B.40)}$$

The implementation of this TLM model (equations B.36 to 3.40) was done in MATLAB in graphic mode using SIMULINK. See Figures B.37- B40

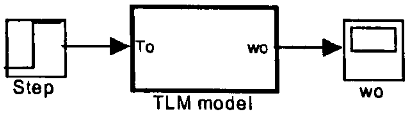


Figure B.13 Motor block diagram model in SIMULINK

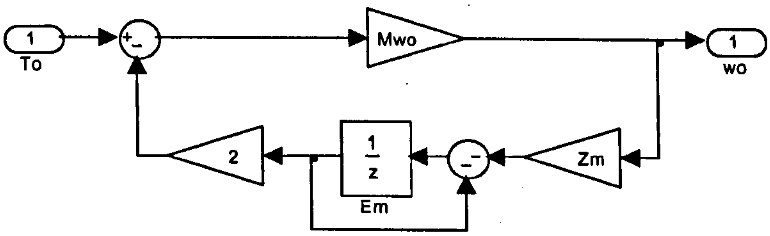


Figure B.14 Subsystem TLM model

The transfer function of equation 3.35 (equation B.41) was used to calculate the step response of the system and the TLM model error for three sample times: 6.245e-3, 6.245e-4 and 6.245e-5 seconds. The MATLAB code for the program used to calculate the error is included in Appendix B.5

$$\frac{\omega_o(s)}{T_o(s)} = \frac{1}{J_m s + (b_m + b_1)} = \frac{1}{0.49s + 1} \qquad \text{(B.41)}$$

B.4 Calculation of the TLM and MTLM Model Errors

```
% error_TLM.m
% This program simulates the step response of the circuit in Figure 3.22a. The error for the TLM and MTL
% models is calculated and plotted
clc, clear all
st=6.245e-3; % sample time [s]
t=0:st:5; % Vector of time [s]
t=t';
len=length(t); % Number of samples to be simulated
% Declaration of the input torque To [N-m]
To=t; % ramp response
tmax=max(t)+st;
simin=[t To];
Jm=0.49; % Motor inertia form Appendix B.1 [kg-m^2]
fa=0.75;
```

```
bm=0.25;
dd=fa+bm;
Tl=0;
 % "Ideal response" from the transfer function in s-domain
% Pre-allocation of variables in memory
num=1;
den=[Jm dd];
% now run model error_cal.mdl to get the "ideal response" (from the transfer
% function)
% after that run the program error_tlm_resp.m to get the errors
```

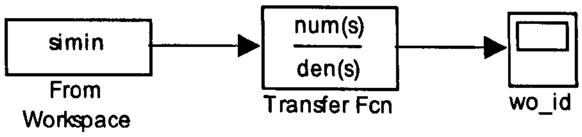


Figure B.15 error_cal.mdl SIMULINK model

```
% error_tlm_resp.m
% Model using the TLM stub
Zm=Jm/(st/2); % Characteristic impedance associated to Jm - equation (B.37)
% Pre-allocation of variables in memory
w_id=wo_id.signals.values; % response from the error_calc.mdl model
% Model using the TLM stub
Zm=Jm/(st/2); % Characteristic impedance associated to Jm - equation (B.37)
wo=zeros(len,1);   % motor angular velocity [rad/s]
em=zeros(len,1);   % Torque almacenated by the inertia [N-m]
Em=zeros(len+1,1);   % Incident pulse associated to Jm [N-m]
for k=1:len
   % Velocity calculation:
   wo(k)=(To(k)-2*Em(k))/(Zm+dd);
   em(k)=wo(k)*Zm+2*Em(k);
   % Calculation next incident pulse:
   Em(k+1)=Em(k)-em(k);
end
w_e=w_id-wo; % error TLM model
% Model using the modified TLM stub:
Zm=Jm/st; % Characteristic impedance associated to Jm - equation (B.37)
% Pre-allocation of variables in memory
wo_m=zeros(len,1);   % motor angular velocity [rad/s]
em=zeros(len,1);   % Torque almacenated by the inertia [N-m]
Em=zeros(len+1,1);   % Incident pulse associated to Jm [N-m]
for k=1:len
   % Velocity calculation:
   wo_m(k)=(To(k)-Em(k))/(Zm+dd);
   em(k)=wo_m(k)*Zm+Em(k);
   % Calculation next incident pulse:
   Em(k+1)=-wo_m(k)*Zm;
end
w_em=w_id-wo_m; % error modified TLM model
plot(t,w_e),grid, xlabel('time [s]'), ylabel('% Error TLM [rad/s]')
pause
plot(t,w_em),grid, xlabel('time [s]'), ylabel('% Error MTLM [rad/s]')
pause
% TLM and MTLM error comparison
subplot(4,1,1)
plot(t,w_e,t,w_em),grid, xlabel('time [s]'), ylabel('Error [rad/s]')
legend('TLM','MTLM',0)
```

```
subplot(4,1,2)
plot(t,w_e./w_id),grid, xlabel('time [s]'), ylabel('% Error TLM [rad/s]')
%legend('TLM','MTLM',0)
subplot(4,1,3)
plot(t,w_em./w_id),grid, xlabel('time [s]'), ylabel('% Error MTLM [rad/s]')
%legend('TLM','MTLM',0)
subplot(4,1,4)
plot(t,w_e./w_id,t,w_em./w_id),grid, xlabel('time [s]'), ylabel('% Error [rad/s]')
axis([0 5 0 8e-4]);
legend('TLM','MTLM')
subplot(111)
```

B.5 Modified TLM Transform for Equation (3.35)

Applying the modified TLM transform (See table 3.2) to the differential term in equation (3.35) gives:

$$T_0(k) - T_1(k) - \omega_0(k)(b_m + b_1) = \omega_0(k)Z_m + E_m^i(k) \tag{B.42}$$

where $$Z_m = J_m / \Delta t = 0.49 / 0.31225x10^{-3} = 1569.255 \tag{B.43}$$

Equation (B.13) becomes:

$$\omega_1(k) = M_{w1}(T_o(k) - E_m^i(k) - 2A_1^i(k)) \tag{B.44}$$

$$\text{where, } M_{w1} = \frac{1}{b_m + Z_m + b_1 + Z_1} = \frac{1}{0.75 + 1569.255 + 0.25 + 15.3274} = 6.3068x10^{-4} \tag{B.45}$$

$$E_m^i(k+1) = -\omega_0(k)Z_m \tag{B.46}$$

Figures B.41 and B.42 show the block diagram for the ω_l calculation subsystem and its initialisation code.

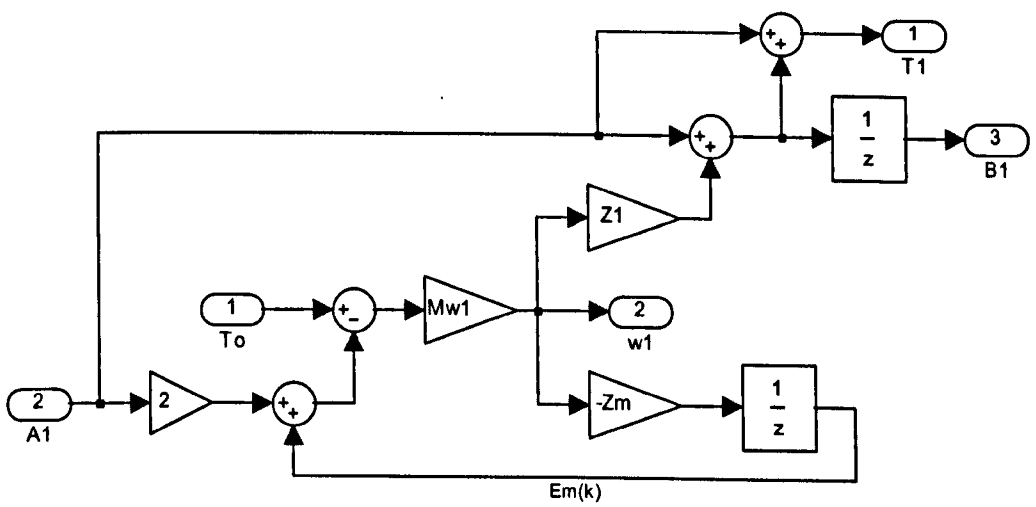


Figure B.16 ω_l calculation block (modified TLM model)

APPENDIX C SYNCHRONISATION BETWEEN THE TLM AXIAL AND TORSIONAL MODELS

The synchronisation between the TLM axial and torsional models for the screw shaft can be achieved using the ratio between axial and torsional propagation speeds as follows:

$$\frac{u_a}{u_t} = \sqrt{\frac{E_{ss}}{G_{ss}}} = \frac{n_a}{n_t} \tag{C.1}$$

This value means that the time spent by an axial wave travelling n_a sections is the time spent by a torsional wave travelling on n_t sections (where n_a and n_t are integers). To model this effect, each torsional section is dived into n_t axial sections to assure that axial and torsional pulses are arriving to the same point at the same timC. Subsequently the number of sections of the axial model (h_a) will be n_t times the number of sections in the torsional model (h) , hence

$$h_a = n_t h \tag{C.2}$$

The application of this procedure using the values for G_{ss} and E_{ss} specified for steel gives a ratio of

$$\frac{n_a}{n_t} = \sqrt{\frac{206x10^9}{79.6x10^9}} = \frac{1027}{638} = 1.6097 \tag{C.3}$$

This ratio implies to evaluate the axial model 1027 times per every simulation of the torsional model, if each section of the torsional model contains 638 sections of the axial model. This value can be reduced to speed up the simulation by analysing a variation of 1% in the values of the parameters G_{ss} and E_{ss} as shown in Table C.1.

	1.01 E_{ss}	0.99 E_{ss}
1.01 G_{ss}	1.6087	1.5927
0.99 G_{ss}	1.6249	1.6087

Table C.1 n_a/n_t ratio for variations of 1% in the values of G_{ss} and E_{ss}

It can be assumed from Table C.1 that a ratio between 1.5929 and 1.6246 is valid taking into account the variations the screw shaft material may have due to the fabrication process. Therefore, the minimum rational number found into this interval (8/5) is selected for the modelling process (n_a= 8 and n_t = 5). In these conditions, E_{ss} is approximated to $204.8x10^9$ N/m^2 for a given value of $G_{ss} = 80x10^9$ N/m^3.

The events synchronisation between the torsional model of the screw shaft and the models for the motor and coupling is achieved by setting the length of each section such as the propagation time becomes a desired t_{pwm} sampling time, thus:

$$t_t = t_{pwm} \tag{C.4}$$

The length of each section in the torsional model (l_{tor}) will be

$$l_{tor} = t_{pwm} u_t \tag{C.5}$$

The number of sections (h) in which the screw shaft is dived will be

$$h = l_{ss} / l_{tor} \tag{C.6}$$

If m is not an integer number, it is rounded to the nearest integer. This implies to change the length of the screw shaft by certain quantity. Applying this procedure to a screw shaft made of steel (density = 7850 kg/m^3) for a length of 1.346 metres gives:

$$u_t = \sqrt{80x10^9 / 7850} = 3192.3 \text{ [m/s]} \tag{C.7}$$

$$l_{tor} = 1x10^{-6} * 3192.3 = 3.1923x10^{-3} \text{ [m]} \tag{C.8}$$

$$h = 1.346 / 3.1923x10^{-3} = 421.63 \approx 422 \text{ sections} \tag{C.9}$$

This means, an increase in the length of the screw shaft (l_{ss}) of:

$$(422 - 421.03) * 3.1923x10^{-3} = 117.06 \mu m \tag{C.10}$$

This error model could be present in the real system due to the tolerances in the machining process of the shaft and changes in the values of the physical properties of the material. For example, if the density value is changed the 0.63% to 7800, the number of sections will be 420 and the length of the screw shaft will be reduced 92.35 μm.

An approach to cope with this limitation of the modelling technique is to assume that the density of the material could vary 1% its nominal value. In consequence, a valid number of sections can be defined as the round value of h (equation (C.6)) towards minus infinity. Then:

$$h = 1346 / 3.1923 = 421.63 \approx 421 \text{ sections} \tag{C.11}$$

The number of sections of the axial model (equation (C.2) will be

$$h_a = 5 * 421 = 2105 \text{ sections} \tag{C.12}$$

Rearranging equation (C.6) gives:

$$l_{tor} = l_{ss} / h = 1346 / 421 = 3.1971 \text{ [mm]} \tag{C.13}$$

u_t can be calculated from equation (C.5) as:

$$u_t = l_{tor} / t_{pwm} = 3.1971x10^{-3} / 1x10^{-6} = 3197.15 \text{ [m/s]} \tag{C.14}$$

The density ρ_{ss} is estimated as:

$$\rho_{ss} = \frac{G_{ss}}{u_t^2} = \frac{80x10^9}{(3197.15)^2} = 7826.43 \text{ [kg/m}^3\text{]} \tag{C.15}$$

The torsional impedance is calculated using equation (4.191):

$$Z_t = 1.97x10^{-3} * 3197.15 = 6.29 \tag{C.16}$$

The length of each axial section is given by

$$l_{axial} = l_{tor} / n_t = 3.1971/5 = 0.63942 \text{ [mm]} \tag{C.17}$$

u_a can be calculated from equation (C.1) as:

$$u_a = \frac{n_a}{n_t} u_t = \frac{5}{8} * 3197.15 = 5115.44 \text{ [m/s]} \tag{C.18}$$

The propagation time for the axial model is

$$t_a = l_{axial} / u_a = 0.6392x10^{-3} / 5115.44 = 125x10^{-9} \text{ [s]} \tag{C.19}$$

The axial impedance is calculated using equation (4.194):

$$Z_a = \rho_{ss} A_{ss} u_a = 7826.43 * 9.13x10^{-4} * 5115.44 = 36563.32 \tag{C.20}$$

APPENDIX D TORSIONAL AND AXIAL TLM MODELS

This Appendix presents the derivation of the TLM model for the torsional and axial dynamics of a ball screw with moving nut. Two bearing configurations are considered: fixed/fixed and fixed/supported. The torsional model remains the same for both bearing configurations due to the fact that a fixed or supported rear bearing will induce a frictional torque T_{rb} on the section where it is placed. In contrast, the axial model changes for the fixed/supported case because the rear bearing mounting is not imposing the restriction that it does in the fixed/fixed case.

D.1 TLM Torsional Model

Three cases are considered for the modelling of the torsional dynamics:

a) Screw shaft subject to an input torque on one end and a load torque on the other end.
b) Screw shaft subject to an input torque including bearings.
c) Case *b* including the nut.

Figure D.1 illustrates the TLM model for case *a*. Torques T_c and T_{ld} are applied to each end of the screw shaft respectively. The shaft is divided into *h* sections. Table D.1 contains the equation for each section of the shaft. It can be seen that pulses are propagated through out the shaft until a disturbance is present in the system - torque T_c in the first section and torque T_{ld} in section *h*. Incident pulses are reflected at those points according to the boundary conditions. Thus, the model is reduced to:

The calculation of the velocities ω_l, ω_{j+1}:

$$\omega_1(k) = (T_c(k) - 2A_1^i(k)) / Z_t \tag{D.1}$$

$$\omega_{h+1} = (2B_h^i - T_{ld}) / Z_t \tag{D.2}$$

- The calculation of the incident pulses affected by the perturbations at the first and last sections:

$$B_1^i(k+1) = A_1^i(k) + \omega_1 Z_t \tag{D.3}$$

$$A_h^i(k+1) = B_h^i(k) - \omega_{h+1} Z_t \tag{D.4}$$

- The propagation of the other A^i and B^i pulses:

$$B_j^i(k+1) = B_{j-1}^i(k) \quad \text{for } j = 2,3,\ldots,h \tag{D.5}$$

$$A_j^i(k+1) = A_{j+1}^i(k) \quad \text{for } j = 1,2,\ldots,h\text{-}1 \tag{D.6}$$

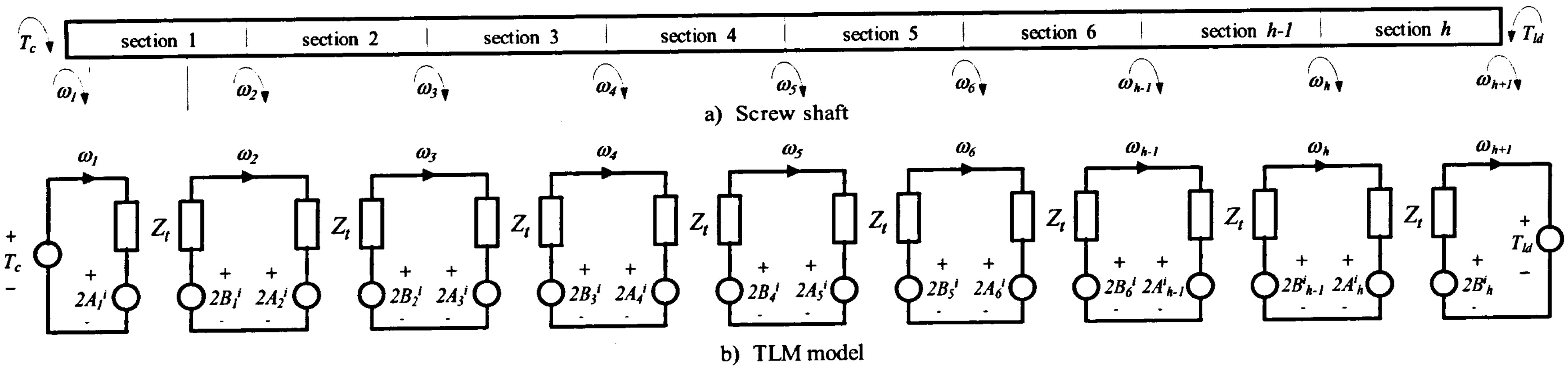


Figure D.1 TLM model of a shaft divided into h sections (case a)

j	1	2	3	4	5	6	h-1	h	h+1
ω_j	$\boxed{\dfrac{T_c - 2A_1^i}{Z_t}}$	$\dfrac{2(B_1^i - A_2^i)}{2Z_t}$	$\dfrac{B_2^i - A_3^i}{Z_t}$	$\dfrac{B_3^i - A_4^i}{Z_t}$	$\dfrac{B_4^i - A_5^i}{Z_t}$	$\dfrac{B_5^i - A_6^i}{Z_t}$	$\dfrac{B_6^i - A_{h-1}^i}{Z_t}$	$\dfrac{B_{h-1}^i - A_h^i}{Z_t}$	$\boxed{\dfrac{2B_h^i - T_{ld}}{Z_t}}$
A_j^r	$\omega_1 Z_t + A_i^i$	$\omega_2 Z_t + A_2^i$	$\omega_3 Z_t + A_3^i$	$\omega_4 Z_t + A_4^i$	$\omega_5 Z_t + A_5^i$	$\omega_6 Z_t + A_6^i$	$\omega_{h-1} Z_t + A_{h-1}^i$	$\omega_h Z_t + A_h^i$	
B_j^r	$B_1^i - \omega_2 Z_t$	$B_2^i - \omega_3 Z_t$	$B_3^i - \omega_4 Z_t$	$B_4^i - \omega_5 Z_t$	$B_5^i - \omega_6 Z_t$	$B_6^i - \omega_{h-1} Z_t$	$B_{h-1}^i - \omega_h Z_t$	$B_h^i - \omega_{h+1} Z_t$	

$B_j^i(k+1) = A_j^r$ and $A_j^i(k+1) = B_j^r$, then when substituting ω_j into the equation for A_j^r gives

$B_j^i(k+1)$	$\boxed{\omega_1 Z_t + A_i^i}$	B_1^i	B_2^i	B_3^i	B_4^i	B_5^i	B_6^i	B_{h-1}^i	
$A_j^i(k+1)$	A_2^i	A_3^i	A_4^i	A_5^i	A_6^i	A_{h-1}^i	A_h^i	$\boxed{B_h^i - \omega_{h+1} Z_t}$	

Table D.1 Calculation of angular velocities and incident pulses for the TLM model of Figure D.1

This dynamic behaviour resembles a circular (linked) list where information is stored to be analysed and modified at designated positions as shown in Figure D.2

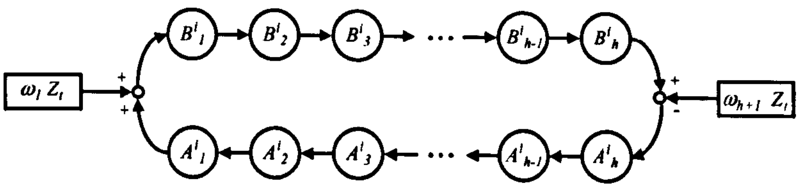


Figure D.2 Graphic representation of the pulses propagation for the case *a*

Figure D.3 illustrates the TLM model for case *b*. The presence of the bearings will generate the perturbations T_{fb} and T_{rb} on the shaft, as shown in Figure D.3a. The perturbations T_c, T_{fb} and T_{rb} split the shaft into two propagation zones: the first one between perturbations T_c and T_{fb}, and the second one between perturbations T_{fb} and T_{rb}. The remaining segment of the shaft will not transmit any torque; therefore it can be modelled as a lumped inertia J_{end}. Where,

$$Z_{end} = J_{end} /(t_t / 2) \tag{D.7}$$

Each one of the propagation zones can be seen as a shaft subject to an input torque on one end and a load torque on the other end, as analysed for the case *a*. The portion of shaft between the front end (to be connected to the coupling) and the rear bearing is divided into h_t sections, the front bearing is placed on the f_b section and the rear bearing is on the h_t section. Figure D.3b shows the resultant TLM model for the system when $h_t = 6$ and $f_b = 2$. As observed in Table D.2 pulses A^i_2 and B^i_3 are reflected due to the perturbation T_{fb} on section 2. The same effect is observed for the pulse A^i_6 due to the perturbation T_{rb} on section 6. Applying equations (D.1 – D.6) to each zone gives:

Calculation of velocities ω_1, ω_3 and ω_7 (ω_1, ω_{fb+1} and ω_{ht+1}):

$$\omega_1(k) = (T_c(k) - 2A^i_1(k)) / Z_t \tag{D.8}$$

$$\omega_{fb+1}(k) = \frac{2(B^i_{fb} - A^i_{fb+1}) - T_{fb}}{2Z_t} \tag{D.9}$$

$$\omega_{ht+1}(k) = \frac{2(B^i_{ht} - E^i_{end}) - T_{rb})}{Z_t + Z_{end}} \tag{D.10}$$

$$E^i_{end}(k+1) = -\omega_{ht+1}(k) Z_{end} - E^i_{end}(k) \tag{D.11}$$

Calculation of the incident pulses affected on zone one (sections one and two)

$$h = f_b \tag{D.12}$$

$$B^i_1(k+1) = A^i_1(k) + \omega_1(k) Z_t \tag{D.13}$$

$$A^i_{fb}(k+1) = B^i_{fb}(k) - \omega_{fb+1}(k) Z_t \tag{D.14}$$

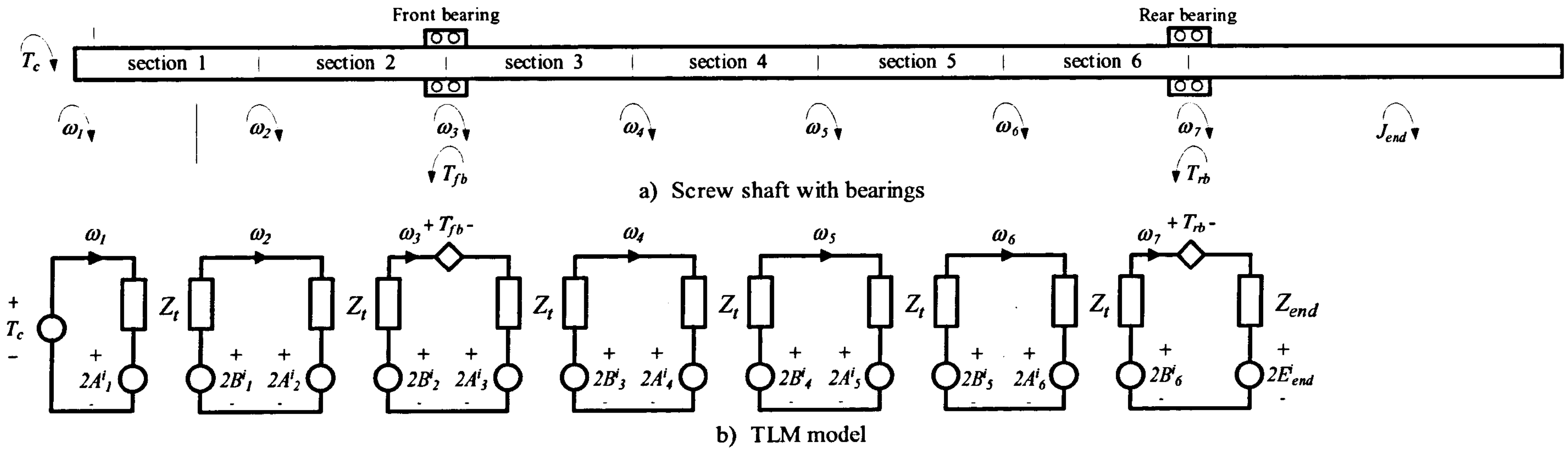


Figure D.3 TLM model of a shaft including the bearings friction (case *b*)

j	1	2	3	4	5	6	7
ω_j	$\dfrac{T_c - 2A_1^i}{Z_t}$	$\dfrac{2(B_1^i - A_2^i)}{2Z_t}$	$\dfrac{2(B_2^i - A_3^i) - T_{fb}}{2Z_t}$	$\dfrac{B_3^i - A_4^i}{Z_t}$	$\dfrac{B_4^i - A_5^i}{Z_t}$	$\dfrac{B_5^i - A_6^i}{Z_t}$	$\dfrac{2(B_6^i - E_{end}^i) - T_{rb}}{Z_t + Z_{end}}$
A_j^r	$\omega_1 Z_t + A_i^i$	$\omega_2 Z_t + A_2^i$	$\omega_3 Z_t + A_3^i$	$\omega_4 Z_t + A_4^i$	$\omega_5 Z_t + A_5^i$	$\omega_6 Z_t + A_6^i$	
B_j^r	$B_1^i - \omega_2 Z_t$	$B_2^i - \omega_3 Z_t$	$B_3^i - \omega_4 Z_t$	$B_4^i - \omega_5 Z_t$	$B_5^i - \omega_6 Z_t$	$B_6^i - \omega_7 Z_t$	

$B_j^i(k+1) = A_j^r$ and $A_j^i(k+1) = B_j^r$, then when substituting ω_j into the equation for A_j^r gives

$B_j^i(k+1)$	$\omega_1 Z_t + A_i^i$	B_1^i	$\omega_3 Z_t + A_3^i$	B_3^i	B_4^i	B_5^i	
$A_j^i(k+1)$	A_2^i	$B_2^i - \omega_3 Z_t$	A_4^i	A_5^i	A_6^i	$B_6^i - \omega_7 Z_t$	

Table D.2 Calculation of angular velocities and incident pulses for the TLM model of Figure D.3

- Propagation of the other A^i and B^i pulses on zone one:

$$B_j^i(k+1) = B_{j-1}^i(k) \quad \text{for } j = 2,\dots,f_b \tag{D.15}$$

$$A_j^i(k+1) = A_{j+1}^i(k) \quad \text{for } j = 1,\dots,f_b - 1 \tag{D.16}$$

- Calculation of the incident pulses affected on zone two (sections three to six)

$$h = h_t - f_b \tag{D.17}$$

$$B_{fb+1}^i(k+1) = A_{fb+1}^i(k) + \omega_{fb+1}(k)Z_t \tag{D.18}$$

$$A_{ht}^i(k+1) = B_{ht}^i(k) - \omega_{ht+1}(k)Z_t \tag{D.19}$$

- Propagation of the other A^i and B^i pulses on zone two:

$$B_j^i(k+1) = B_{j-1}^i(k) \quad \text{for } j = f_b + 2,\dots, h_t \tag{D.20}$$

$$A_j^i(k+1) = A_{j+1}^i(k) \quad \text{for } j = f_b + 1,\dots, h_t - 1 \tag{D.21}$$

Figure D.4 shows the graphical representation of the pulses propagation.

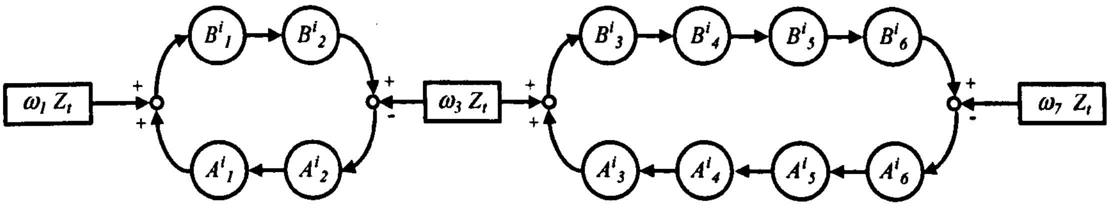


Figure D.4 Graphic representation of the pulses propagation for case *b*

Figure D.5 shows the TLM model for the ball screw with the moving nut (case *c*). Table D.3 contains the equations for the model.

The inclusion of the perturbations T_a on section four of the shaft generates the reflection of the pulses arriving to sections four and five. Assuming n the section where the nut is on ($n = 4$ for this case) gives:

$$\omega_{n+1}(k) = \frac{2(B_n^i - A_{n+1}^i) - T_a}{2Z_t} \tag{D.22}$$

$$B_{n+1}^i(k+1) = A_{n+1}^i(k) + \omega_{n+1}(k)Z_t \tag{D.23}$$

$$A_n^i(k+1) = B_n^i(k) - \omega_{n+1}(k)Z_t \tag{D.24}$$

Equations (D.20) and (D.21) become

$$B_j^i(k+1) = B_{j-1}^i(k) \quad \text{for } j = f_b + 2,\dots, h_t \qquad j \neq n+1 \tag{D.25}$$

$$A_j^i(k+1) = A_{j+1}^i(k) \quad \text{for } j = f_b + 1,\dots, h_t - 1 \quad j \neq n \tag{D.26}$$

Consequently, the zone two of the shaft is divided into two loops (Figure D.6) that change with time as the nut moves along the screw shaft. For example, the TLM model presented in Figure D.5 changes to the one in Figure D.7 when the nut moves to section five.

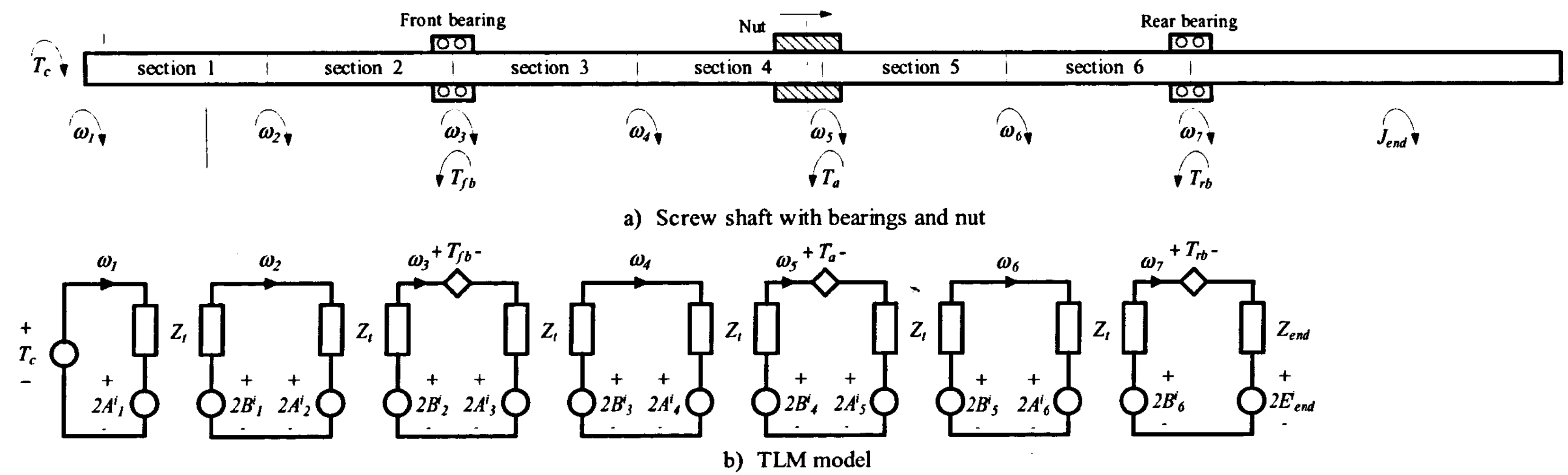


Figure D.5 TLM model of a shaft divided into eight sections including the bearings friction and moving nut (case *c*)

j	1	2	3	4	5	6	7
ω_j	$\frac{T_c - 2A_1^i}{Z_t}$	$\frac{2(B_1^i - A_2^i)}{2Z_t}$	$\frac{2(B_2^i - A_3^i) - T_{fb}}{2Z_t}$	$\frac{B_3^i - A_4^i}{Z_t}$	$\frac{2(B_4^i - A_5^i) - T_a}{2Z_t}$	$\frac{B_5^i - A_6^i}{Z_t}$	$\frac{2(B_6^i - E_{end}^i) - T_{rb}}{Z_t + Z_{end}}$
A_j^r	$\omega_1 Z_t + A_i^i$	$\omega_2 Z_t + A_2^i$	$\omega_3 Z_t + A_3^i$	$\omega_4 Z_t + A_4^i$	$\omega_5 Z_t + A_5^i$	$\omega_6 Z_t + A_6^i$	
B_j^r	$B_1^i - \omega_2 Z_t$	$B_2^i - \omega_3 Z_t$	$B_3^i - \omega_4 Z_t$	$B_4^i - \omega_5 Z_t$	$B_5^i - \omega_6 Z_t$	$B_6^i - \omega_7 Z_t$	

$B_j^i(k+1) = A_j^r$ and $A_j^i(k+1) = B_j^r$, then when substituting ω_j into the equation for A_j^r gives

$B_j^i(k+1)$	$\omega_1 Z_t + A_i^i$	B_1^i	$\omega_3 Z_t + A_3^i$	B_3^i	$\omega_5 Z_t + A_5^i$	B_5^i	
$A_j^i(k+1)$	A_2^i	$B_2^i - \omega_3 Z_t$	A_4^i	$B_4^i - \omega_5 Z_t$	A_6^i	$B_6^i - \omega_7 Z_t$	

Table D.3 Calculation of angular velocities and incident pulses for the TLM model of Figure D.5

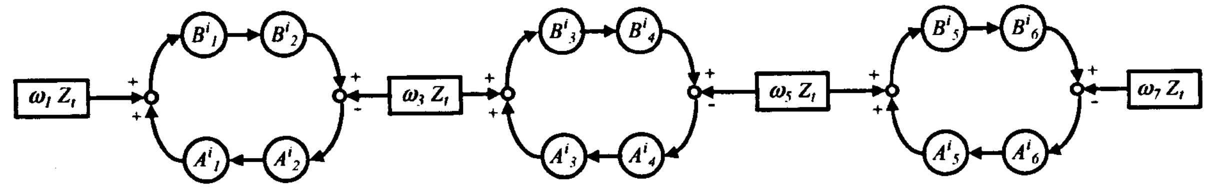
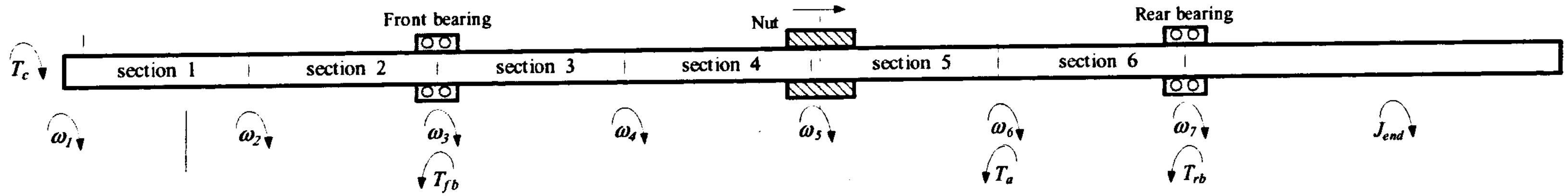
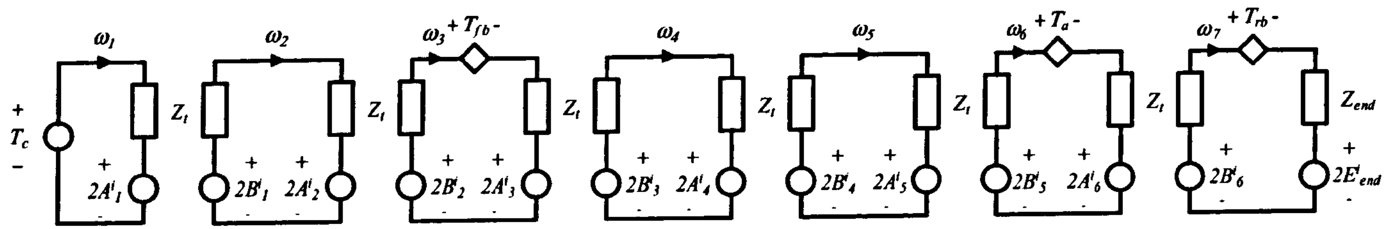


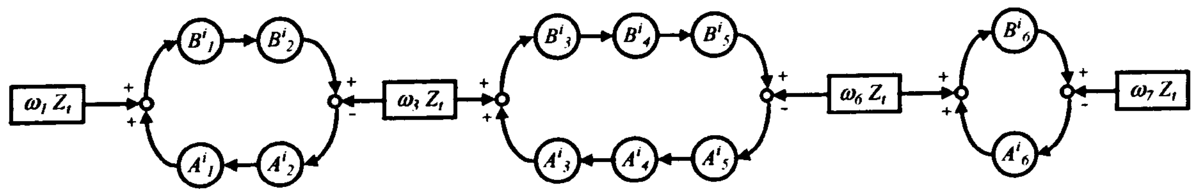
Figure D.6 Graphic representation of the pulse propagation for case c (nut on section four)



a) Screw shaft with bearings and nut

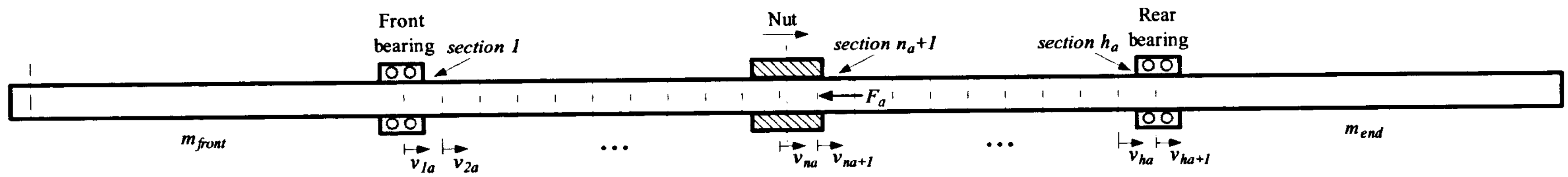


b) TLM model

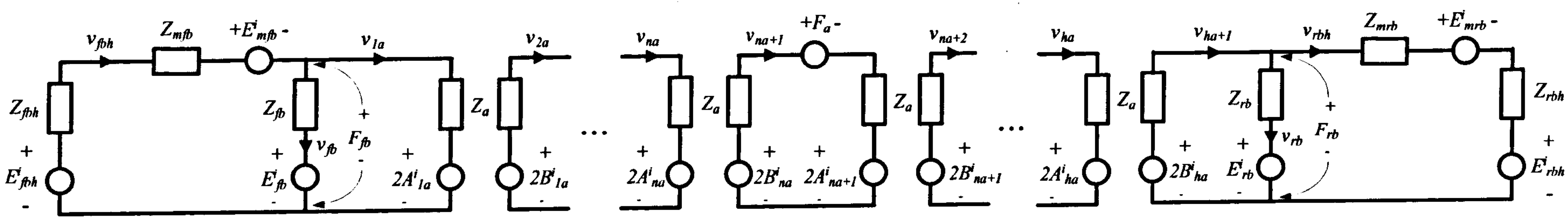


c) Graphic representation of the pulses propagation

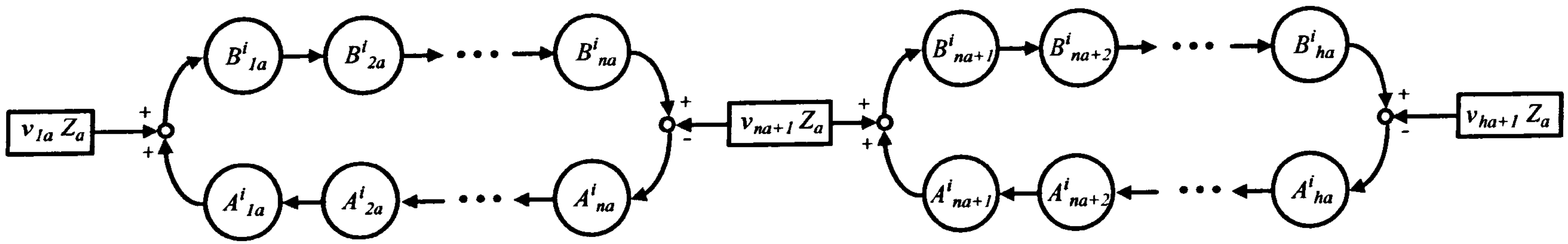
Figure D.7 TLM model for case c (nut on section five)



a) Screw shaft with bearings and nut



b) MTLM model



c) Graphic representation of the pulses propagation

Figure D.8 TLM model for the screw shaft axial dynamics (fixed/fixed bearing configuration)

D.2 TLM Axial Model (Fixed/fixed Bearing Configuration Case)

The TLM model for the axial dynamics is derived applying the procedure used for the case c of the torsional model. The portion of shaft between bearings is divided into h_a sections, the front bearing is placed at the beginning of the *first* section, the nut is on the n_a section, and the rear bearing is on the h_a section as shown in Figure D.8a. The shaft portions at the front and at the end do not affect the axial rigidity as explained in section 4.4.2. These portions are then modelled as lumped masses m_{fb} and m_{rb}.

Masses m_{fb} and m_{rb} are calculated given the density (ρ_{ss}) and the cross sectional area of the shaft (A_{ss}):

$$m_{fb} = l_{front} A_{ss} \rho_{ss} \tag{D.27}$$

$$m_{rb} = l_{end} A_{ss} \rho_{ss} \tag{D.28}$$

Where l_{front} and l_{end} represent the length of the front and the end portions of the shaft. These lengths are calculated for a screw shaft of length l_{ss} as:

$$l_{front} = f_b l_{tor} \tag{D.29}$$

$$l_{end} = l_{ss} - h_t l_{tor} \tag{D.30}$$

The corresponding TLM impedances are

$$Z_{mfb} = m_{fb} / t_a \tag{D.31}$$

$$Z_{mrb} = m_{rb} / t_a \tag{D.32}$$

The restrictions imposed by the bearing mountings on the first and h_a sections generate the reflection of pulses arriving to those sections. The procedure derived in Appendix D.1 can be applied taking into account the appropriated signals representation (linear velocity instead of angular velocity and force instead of torque). The graphical representation of the corresponding pulse propagation is illustrated in Figure D.8c.

The velocities for the front bearing mounting are calculated including the TLM model derived for the bearings stiffness in Section 4.4.3 as shown in Figure D.9.

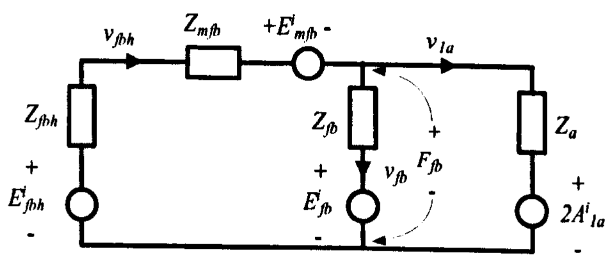


Figure D.9 TLM model for the front bearing mounting (Fixed case)

This electric circuit is solved finding the Thevenin equivalent with respect to F_{fb} (Figure

D.10), thus:

$$E_{eq}(k) = Z_{Emfbh}E_{mfbh}(k) + Z_{Efb}E^{i}_{fb}(k) \tag{D.33}$$

$$Z_{eq} = Z_{mfbh}Z_{fb} /(Z_{mfbh} + Z_{fb}) \tag{D.34}$$

Where,

$$Z_{Emfbh} = Z_{fb} /(Z_{mfbh} + Z_{fb}) \tag{D.35}$$

$$Z_{Efb} = Z_{mfbh} /(Z_{mfbh} + Z_{fb}) \tag{D.36}$$

$$Z_{mfbh} = Z_{fbh} + Z_{mfb} \tag{D.37}$$

$$E_{mfbh}(k) = E^{i}_{fbh}(k) - E^{i}_{mfb}(k) \tag{D.38}$$

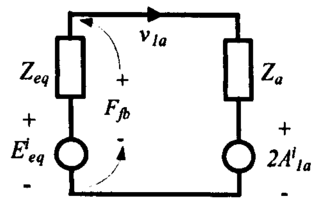


Figure D.10 TLM reduce model for the front bearing mounting (Fixed case)

$$v_{1a}(k) = M_{v1a}(E_{eq}(k) - 2A^{i}_{1}(k)) \tag{D.39}$$

$$F_{fb}(k) = E_{eq}(k) - v_{1a}(k)Z_{eq} \tag{D.40}$$

$$v_{fbh}(k) = M_{vfbh}(E_{mfbh}(k) - F_{fb}(k)) \tag{D.41}$$

$$E^{i}_{fb}(k+1) = F_{fb}(k) \tag{D.42}$$

$$E^{i}_{fbh}(k+1) = E^{i}_{fbh}(k) - Z_{fbh}v_{fbh}(k) \tag{D.43}$$

$$E^{i}_{mfb}(k+1) = -Z_{mfb}v_{fbh}(k) \tag{D.44}$$

$$B^{i}_{1a}(k+1) = A^{i}_{1a}(k) + v_{1a}(k)Z_{a} \tag{D.45}$$

Where,

$$M_{v1a} = 1/(Z_{eq} + Z_{a}) \tag{D.46}$$

$$M_{vfbh} = 1/Z_{mfbh} \tag{D.47}$$

Applying the same procedure to the rear bearing mounting (Figure D.11) gives:

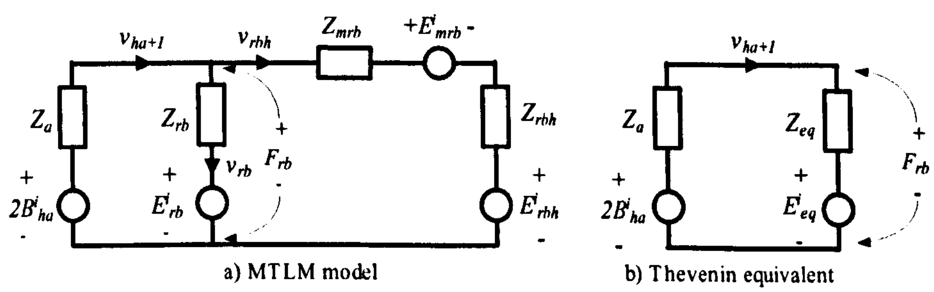


Figure D.11 TLM model for the rear bearing mounting (Fixed case)

$$E_{eq}(k) = Z_{Erb}E^{i}_{rb}(k) + Z_{Emrbh}E_{mrbh}(k) \tag{D.48}$$

$$Z_{eq} = Z_{mrbh}Z_{rb} /(Z_{mrbh} + Z_{rb}) \tag{D.49}$$

Where, $$Z_{Emrbh} = Z_{rb} /(Z_{mrbh} + Z_{rb}) \tag{D.50}$$

$$Z_{Erb} = Z_{mrbh} /(Z_{mrbh} + Z_{rb}) \tag{D.51}$$

$$Z_{mrbh} = Z_{mrb} + Z_{rbh} \tag{D.52}$$

$$E_{mrbh}(k) = E^{i}_{mrb}(k) + E^{i}_{rbh}(k) \tag{D.53}$$

$$v_{ha1}(k) = M_{vha1}(2B^{i}_{ha}(k) - E_{eq}(k)) \tag{D.54}$$

$$F_{rb}(k) = E_{eq}(k) + v_{ha1}(k)Z_{eq} \tag{D.55}$$

$$v_{rbh}(k) = M_{vrbh}(F_{rb}(k) - E_{mrbh}(k)) \tag{D.56}$$

$$B^{i}_{1a}(k+1) = A^{i}_{1a}(k) + v_{1a}(k)Z_{a} \tag{D.57}$$

$$E^{i}_{rb}(k+1) = F_{rb}(k) \tag{D.58}$$

$$E^{i}_{rbh}(k+1) = Z_{rbh}v_{rbh}(k) + E^{i}_{rbh}(k) \tag{D.59}$$

$$E^{i}_{mrb}(k+1) = -Z_{mrb}v_{rbh}(k) \tag{D.60}$$

Where, $$M_{vha1} = 1/(Z_{a} + Z_{eq}) \tag{D.61}$$

$$M_{vrbh} = 1/Z_{mrbh} \tag{D.62}$$

The propagation of A^{i}_{a} and B^{i}_{a} pulses on the other sections is given by:

$$B^{i}_{ja}(k+1) = B^{i}_{ja-1}(k) \quad \text{for } j = 2,\ldots, h_a \;\; j \neq n_a{+}1 \tag{D.63}$$

$$A^{i}_{ja}(k+1) = A^{i}_{ja+1}(k) \quad \text{for } j = 1,\ldots, h_a{-}1 \;\; j \neq n_a \tag{D.64}$$

D.3 TLM Axial Model (Fixed/supported Bearing Configuration Case)

In the fixed/supported-bearing configuration only the front bearing (fixed side) imposes restrictions to the axial displacement of the shaft, as described in section 4.4.3. The model for the rear bearing mounting is reduced to the one presented in Figure D.12. $B^{i}_{1a}(k+1)$ is calculated using equation (D.57)

$$v_{ha+1}(k) = M_{vha1}(2B^{i}_{ha}(k) - E^{i}_{mrb}(k)) \tag{D.65}$$

Where $$M_{vha1} = 1/(Z_{a} + Z_{mrb}) \tag{D.66}$$

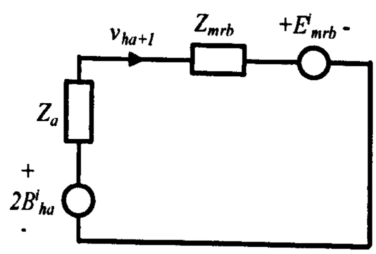


Figure D.12 TLM model for the rear bearing mounting (Supported case)

APPENDIX E TEST RIG SPECIFICATIONS

This appendix contains the specifications of the constitutive elements of the test rig. This rig is a CNC single axis drive representative of the y-axis of a Bridgeport Vertical Machining Centre.

Motion Controller	HEIDENHAIN TNC 426PB (280 476-24)
Electrical Drive	SIEMENS Simodrive 611 (6SN1123-1AA00-0CA1)
Motor	SIEMENS 1FT6082-1AF7-1AG1
Guideways	THK SNS45-LC
Front bearing	Double Row Angular Contact Ball Bearing ZKLF 3080.2RS PE
Rear bearing	Double Row Angular Contact Ball Bearing ZKLF 3080.2RS PE
Ballscrew	THK BIF (BNFN) 4016-5 RRG0S-1346(1065)L-C3-E
Coupling	ROBA – ES 28 940.000
Rotary encoder	HEIDENHAIN ERN 1387
Linear encoder	HEIDENHAIN AE LS486C ML620
Transverse	500 mm
Rapid Traverses	30000 mm/min
Feed Rates	100 – 12000 mm/min
Table Size	115mm x 58 mm
Table mass	312 Kg
Saddle mass	524 Kg
Load mass	853.6 Kg

Table E.1 Test rig specifications

Parameter	Controller code	Symbol	Value	Units
Maximum feed rate (According to rig specifications)	MP 1010	v_{max}	32000	mm/min
Maximum acceleration	MP 1060	a_{max}	6	m/s^2
Maximum jerk	MP 1090	j_{max}	75	m/s^3
Position controller cycle time	MP 7600	t_p	3	ms
Transient response during acceleration and deceleration	MP 1521	t_{ad}	*0*	ms
Position filter: Tolerance for contour transitions	MP 1096	d_{tol}	0	mm
Minimum position filter order	MP 1099	n_{pfo}	5	
Proportional factor position controller	MP 1510	k_v	4	(m/min)/mm
Velocity controller cycle time		t_v	0.6	ms
Velocity feed forward factor	MP 1396	k_{vff}	1	
Transverse per motor revolution (screw shaft lead)	MP 2020	k_b	16	*mm*
Proportional factor – velocity controller	MP 2500	k_p	18	*A-s/rev*
Integral factor- velocity controller	MP 2510	k_i	2500	A/rev
Differential factor-velocity controller	MP 2520	k_d	0	$A\text{-}s^2$/rev
Low-pass filter - actual velocity	MP2560	PT_1	0	
Holding torque factor (vertical axis)	MP 2630	i_{ht}	0	A
Acceleration feed forward factor	MP 2600	k_{aff}	0.039	$A\text{-}s^2$/rev
Friction compensation at low motor speed (10 rpm)	MP 2610	i_{ff_l}	0	A
Delay of the friction compensation	MP 2612	t_{fc}	0	ms
Friction compensation at motor rated speed	MP 2620	i_{ffm}	0	A
Second order delay	MP	PT_2	0	ms
Band-stop filter damping	MP	dB_1	9	dB
Band-stop filter center frequency	MP	f_1	338	Hz
Current controller cycle time		t_c	0.2	ms
Proportional factor at standstill – current controller	MP 2400	k_{cp}	50	V/A
Proportional factor – current controller	MP 2402	k_{cp}	50	V/A
Input voltage (DC link voltage)	MP	V_{DC}	600	V
Switching frequency	MP 2180	f_{pwm}	5	KHz

Table E.2 Y-axis motion controller parameters [106]

MP 2421 and MP 2431 in Figure E.1 are the current controller parameters when the system is controlling induction motors.

SIEMENS Simodrive 611 6SN1123-1AA00-0CA1 (one axis)			
Parameter	Symbol	Value	Units
Input voltage (DC link voltage)	V_{DC}	600	V
Output voltage (3-pase AC)	e_{abc}	0 - 430	V
Maximum current	I_{max}	50	A

Table E.3 Inverter technical data [105]

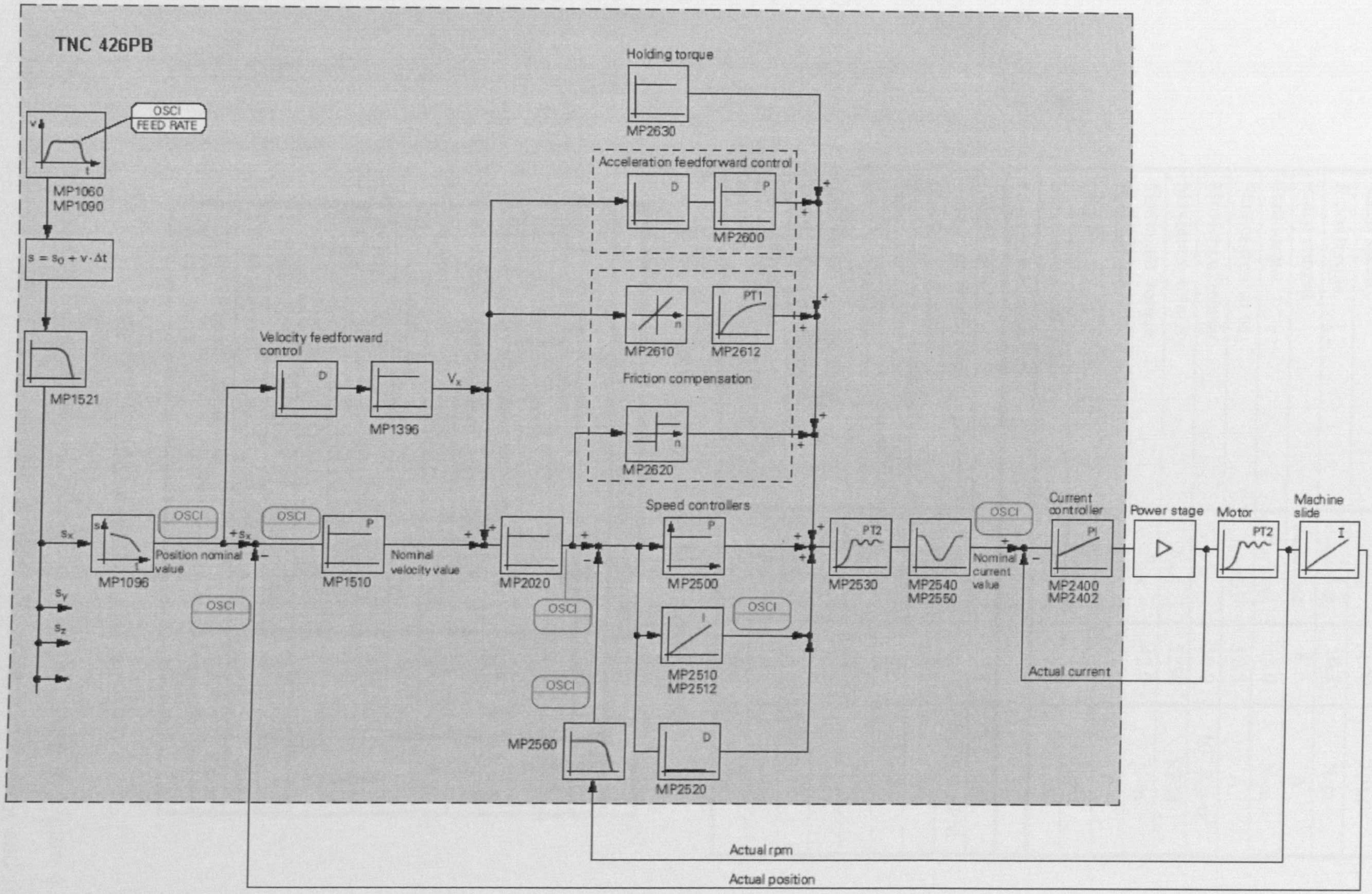


Figure E.1 Block diagram of the TNC 426PB [70]

SIEMENS 1FT6082-1AF7-1AG1 (Permanent magnet synchronous motor)			
Parameter	Symbol	Value	Units
Number of pair poles	p	4	
Rated speed	n_{rated}	3000	RPM
Rated torque	$T_{rated\ (100k)}$	10.3	N-m
Rated current	$I_{rated\ (100K)}$	8.7	A
Stall torque	$T_{0\ (100K)}$	13.0	N-m
Stall current at M_o	$I_{0\ (100K)}$	10.2	A
Moment of inertia	J_m	30.0	10^{-4} Kg-m^2
Maximum velocity	n_{max}	5250	RPM
Maximum. torque	T_{max}	42.0	N-m
Peak current	I_{max}	41.0	A
Limiting torque (600V)	T_{limit}	29.0	N-m
Limiting current (600V)	I_{limit}	29.9	A
Physical constants			
Torque constant	k_T	1.28	N-m/A
Voltage constant (phase to phase)	k_e	80.0	V/1000 RPM
Winding resistance	R	0.68	Ohm
Three-phase inductance	L	6.2	mH
Electric time constant	t_{el}	9.3	ms
Mechanical time constant	t_{mech}	3.7	ms
Thermal time constant	t_{th}	35	min
Thermal resistance	R_{th}	0.15	W/K
Mass	m_m	15.0	Kg

Table E.4 Motor technical data [106]

THK linear guideway SNS45-LC			
Parameter	Symbol	Value	Units
Basic static load rating	C_o	222	KN
Basic dynamic load rating	C_{gw}	123	KN
Radial rigidity (downward/upward)	k_{rl}	1.56/1.15	N/µm
Resistance under no load	F_{gw0}	15	N
Imposed load	M_{gw}	853.6	Kg
Friction coefficient	μ_{gw}	0.003	
Radial load	F_{rad}	8373.816	N
Lateral load	F_{lat}	0	N
Radial factor	X_{rad}	1	
Lateral factor	Y_{lat}	0.935	
LM block mass	m_{bl}	3.4	Kg
LM rail mass per unit length	m_{rl}	9.8	Kg/m
LM rail length	l_{rl}	1.3	m

Table E.5 Guideways technical data [97]

The friction coefficient of the guideway is calculated from the graph in Figure 4.42. In this case the imposed load (M_{gw}) is the weight of load (the table, saddle, nut and the four LM blocks)

$$M_{gw} = m_{table} + m_{saddle} + m_{nut} + 4 * m_{bl} = 312 + 524 + 4 + 4 * 3.4 = 853.6 \ \text{[kg]} \tag{E.1}$$

The radial force of friction is

$$F_{rad} = 9.81 * M = 9.81 * 853.6 = 8373.816 \ \text{[N]} \tag{E.2}$$

According to equation (4.102), the equivalent load is

$$F_E = 1 * 8373.816 + 0.935 * 0 = 8373.816 \ \text{[N]} \tag{E.3}$$

For a load ratio of 0.068 the friction coefficient is $\mu = 0.003$.

$$M_{gw} / C_{gw} = 853.6 * 9.8 / 123000 = 0.068 \tag{E.4}$$

Thus, the frictional force due to the load (equation (4.104)) is

$$F_o = 15 + 0.003 * 8373.816 = 40.121 \ \text{[N]} \tag{E.5}$$

Double Row Angular Contact Ball Bearing ZKLF 3080.2RS PE			
Parameter	Code	Value	Units
Contact angle	α	60	Degrees
Bore diameter	d_b	40	mm
Pitch circle diameter	d_{bl}	45.5	mm
Outer diameter	D_b	80	mm
Basic static load rating	C_o	64	KN
Axial rigidity	k_b	850	N/µm
Moment of inertia (rotating inner ring)	J_b	0.73	Kg-cm^2
Mass	m_b	0.7	Kg
Radial load	F_r	65.727	N
Preloading load	F_{ao}	2180	N
Limiting speed	v_{bmax}	2200	RPM
Bearing housing ridity front/rear	k_{bh}	1.9/1.2	KN-µm
Grease lubricated DIN: K3K-30	υ=100 mm^2/s at 40°C		

Table E.6 Bearings technical data (Front and rear) [92]

From reference [93]:

$$f_0 = 4 \tag{E.6}$$

$$f_1 = 0.001 * (P_1 / C_o)^{1/3} \tag{E.7}$$

Where

$$P_1 = 1.4 * F_{ab} - 0.1 * F_{rb} \tag{E.8}$$

Load factors are chosen from table E.1

α=60°	X_b	Y_b
$F_{ab} / F_{rb} \le 2.17$	1.9	0.55
$F_{ab} / F_{rb} > 2.17$	0.92	1

Table E.7 Bearings load factors [93]

The radial load of the bearing is the force due to the screw shaft weight, thus

$$F_{rb} = m_s g / 2 = 13.4 * 9.81 / 2 = 65.727 \ \text{[N]} \tag{E.9}$$

The axial load is the resultant from the sum of the preloading load and the axial force induce on the ballscrew arrangement, thus

$$F_{ab} = F_{ao} + |F_a| = 2180 + |F_a| \text{ [N]} \tag{E.10}$$

The minimum F_{ab} / F_{rb} ratio is

$$F_{ab} / F_{rb} = 2180 / 65.727 = 162.68 \tag{E.11}$$

Thus from table E.7: X_b=0.92 and Y_b=1. From equation (4.90), the Resultant bearing load will be:

$$F_R = 0.92 * 65.727 + 1 * (2180 + |F_a|) = 2240.47 + |F_a| \text{ [N]} \tag{E.12}$$

The load-dependent component (equation (4.89)):

$$T_1 = 0.001 * \left(\frac{1.4 * (2180 + |F_a|) - 0.1 * 65.727}{64000} \right)^{1/3} (2180 + |F_a|) * 45.5 / 1000 \tag{E.13}$$

$$T_1 = (0.0992 + 4.55x10^{-5} * |F_a|) * (0.3627 + 7.8217 * 10^{-5} * |F_a|) \tag{E.14}$$

Given a maximum axial load of 8kN, values of T_l will vary from 0.0363 to 0.2038 N-m.

The velocity-dependent component (equation (4.88)):

$$T_0 = 4 * 10^{-10} * \left(100 * \omega * \frac{60}{2 * \pi} \right)^{2/3} * 45.5^3 = 0.0037 * \omega^{2/3} \text{ [N-m]} \tag{E.15}$$

Parameter	Symbol	Value	Units
Nominal torque of coupling	T_{KN}	160	Nm
Maximum torque of coupling	$T_{K\,max}$	320	Nm
Static torsional stiffness	$C_{T\,stat.}$	4200	Nm/ rad
Dynamic torsional stiffness	$C_{T\,dyn.}$	10100	Nm/ rad
Static radial stiffness	C_r	3500	N/mm
Maximum speed	n_{max}	8500	rpm
Mass moment of inertia per hub	J_c	$200.3x10^{-6}$	kg-m^2
Mass per hub	m_c	0.253	kg
Hub material aluminium			
Elastomeric element colour red, Shore 98 Sh A			

Table E.8 Coupling technical data [111]

HEIDENHAIN ERN 1387			
Parameter	Symbol	Value	Units
Incremental signal: sinusoidal		1	V_{pp}
Resolution		2048	pulse/revolution
Limiting velocity	v_{bmax}	15000	RPM
Mass	m_{re}	0.25	Kg
Moment of inertia	J_{re}	$2.6x10^{-6}$	Kg-m^2

Table E.9 Rotary encoder technical data [112]

HEIDENHAIN AE LS486C ML620			
Technical data	Symbol	Value	Units
Absolute signal: sinusoidal		1	V_{pp}
Grating period		20	μm
Measuring length		620	mm
Maximum transverse velocity	v_{lemax}	120	m/min
Mass	m_{le}	0.4	Kg
Required moving force	F_{le}	5	N

Table E.10 Linear encoder technical data [112]

Parameter	Symbol	Value	Units
Screw shaft			
Lead	l_d	16	mm/rev
Outer diameter	d_s	40	mm
Thread minor diameter	d_c	34.1	mm
Length	l_{ss}	1346	mm
Moment of Inertia per unit mass	I_o	1.97	10^{-2} Kg-cm^2/mm
Reduction ratio	n	1	
Efficiency	ef	0.9	%
Young's modulus	E_{ss}	206	KN/mm^2
Shear modulus	G_{ss}	79.6	GN/m^3
Density	ρ_{ss}	7850	Kg/m^3
Mass	m_s	13.3	Kg
Material	AISI 4150 H Steel (Standard low alloy)		
Nut			
Pre-loading force	F_{ao}	2250	N
Ball circle diameter	BCD	42	mm
Dynamic load rating	C_a	61.4	KN
Static load rating	C_{oa}	158.8	KN
Axial rigidity	k_n	1500	N-μm
Mass	m_n	4	Kg
Material	SAE 8620 (common carburising steel)		
Mounting dimensions			
Front bearing position	l_f	120	mm
Rear bearing position	l_r	1276	mm
Absolute position reference point (with linear encoder)	l_o	660	mm
Stroke length (with linear encoder)	l_s	415	mm
Absolute position reference point (with rotary encoder)	l_o	610	mm
Stroke length (with rotary encoder)	l_s	462	mm

Table E.11 Ball screw technical data [109]

APPENDIX F THE VMC -500 MACHINE SPECIFICATIONS

Motion Controller	SIEMENS SINUMERIK 840D (S7-315-2DP)
Electrical Drive	SIEMENS SIMODRIVE 611D (6SN1123-1AA00-0CA1)
Motor	SIEMENS 1FK6063-6AF71-1AG2
Guideways	THK HSR35 A2 SS CO QZ + 1090L H II
Front bearing	Double Row Angular Contact Ball Bearing NSK BSB 025062
Rear bearing	Single-row deep groove ball bearing NSK 6305 TB
Ballscrew	THK
Coupling	ROTEX 24 GS spider 98 Shore A
Rotary encoder	HEIDENHAIN ERN 1387
Linear encoder	HEIDENHAIN AE LS186C ML620
Transverse	510 mm
Rapid Traverses	30000 mm/min
Feed Rates	300 – 12000 mm/min
Table Size	700 mm x 520 mm
Table mass	327 Kg
Load mass	Kg
Total mass	Kg

Table F.1 X-axis specifications

Motion Controller	SIEMENS SINUMERIK 840D (S7-315-2DP)
Electrical Drive	SIEMENS SIMODRIVE 611 (6SN1123-1AA00-0CA1)
Motor	SIEMENS 1FK6063-6AF71-1AG2
Guideways	THK HSR35 A2 SS CO QZ + 1090L H II
Front bearing	Double Row Angular Contact Ball Bearing NSK BSB 025062
Rear bearing	Single-row radial ball bearing NSK 6005 – 2RS
Ballscrew	THK
Coupling	ROTEX 24 GS spider 98 Shore A
Rotary encoder	HEIDENHAIN ERN 1387
Linear encoder	HEIDENHAIN AE LS186C ML620
Transverse	510 mm
Rapid Traverses	30000 mm/min
Feed Rates	300 – 12000 mm/min
Saddle mass	160Kg
Load mass	Kg
Total mass	

Table F.2 Y-axis specifications

Motion Controller	SIEMENS SINUMERIK 840D (S7-315-2DP)
Electrical Drive	SIEMENS SIMODRIVE 611 (6SN1123-1AA00-0CA1)
Motor	SIEMENS 1FT6064-6AH71-4AB1
Guideways	THK HSR35 A2 SS CO QZ + 1090L H II
Front bearing	Double Row Angular Contact Ball Bearing NSK BSB 025062
Rear bearing	Single-row deep groove ball bearing NSK 6305 – 2 RS
Ballscrew	THK
Coupling	ROTEX 24 GS spider 98 Shore A
Rotary encoder	HEIDENHAIN ERN 1387
Linear encoder	HEIDENHAIN AE LS186C ML620
Transverse	510 mm
Rapid Traverses	30000 mm/min
Feed Rates	300 – 12000 mm/min
Carrier mass	212 Kg
Motor mass	152 Kg
Load mass	364 Kg

Table F.3 Z-axis specifications

Code	Parameter	Value	Units
10061	Position control cycle time	4	ms
32200	Position control gain	3	(m/min)/mm
32300	Maximum acceleration	6	m/s^2
32430	Maximum jerk	100	m/s^3
32620	Feed-forward mode*	3	Velocity
32630	Feed forward activation mode**	0	Off
36200	Maximum feed rate	34500	mm/min

Table F.4 Motion controller parameters (SINUMERIK 850D)

* 3: velocity feed-forward
4: velocity and torque (current) feed-forward
** 0: disabled
1: to switch on/off in the part-program use FFWON or FFWOF (by default)

Parameter	Code	Value	Units
Input voltage (DC link voltage)	V_{DC}	600	V
Output voltage (3-pase AC)	e_{abc}	0 - 430	V
Maximum current	I_{max}	50	A
Motor Rated current			
1FT5	i_{rated}	25	A
1FT6	i_{rated}	18	A
1PH	i_{rated}	24	A
Induction motors	i_{rated}	24	A

Table F.5 Technical data SIMODRIVE 611D 6SN1123-1AA00-0CA1 (two axis)

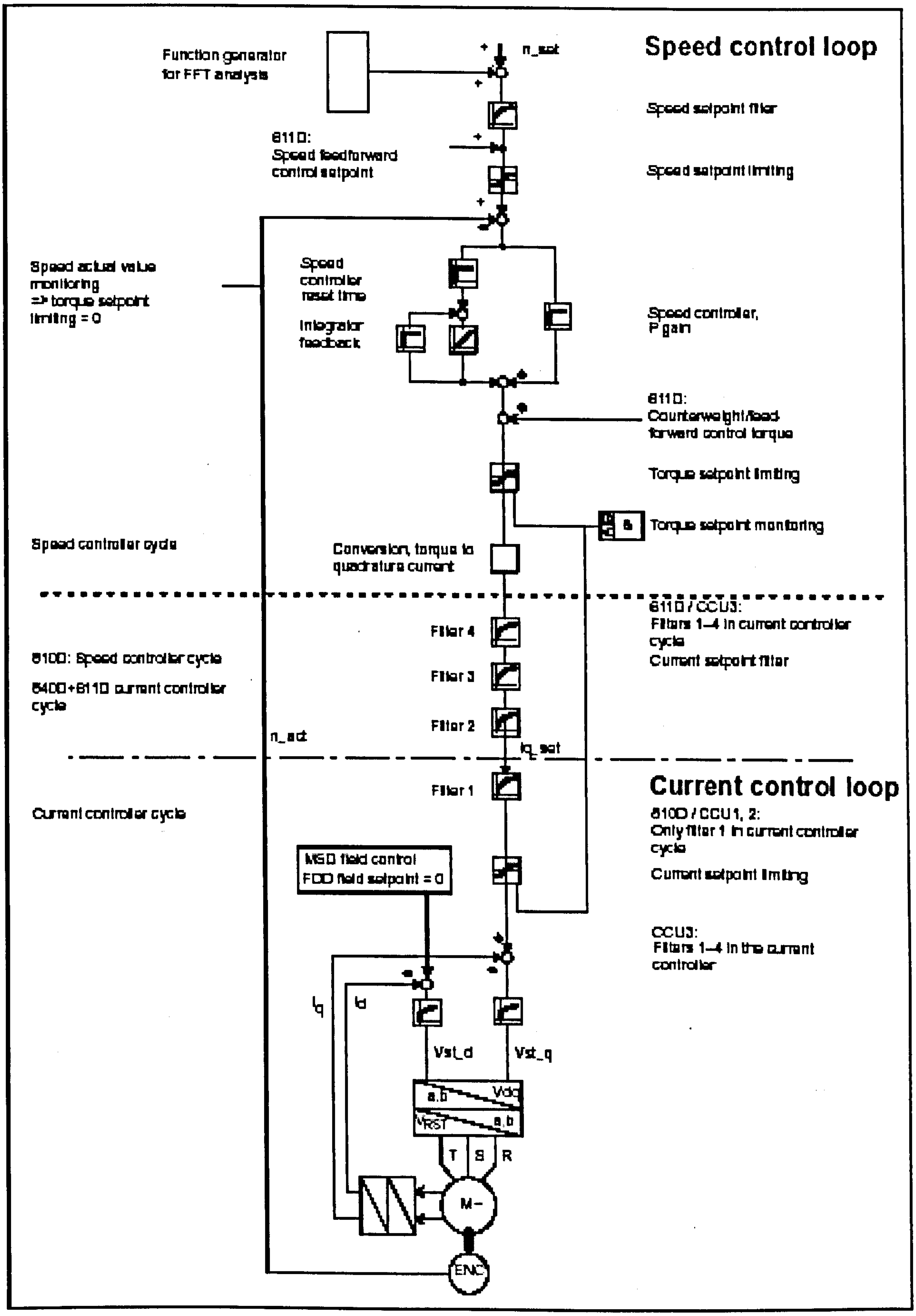


Figure F.1 Velocity and current control block diagram (SIMODRIVE 611) [105]

Machine code	Parameter	x-axis	y-axis	z-axis	Units
1000	Current controller cycle time	0.125	0.125	0.125	ms
1001	Velocity control cycle time	0.125	0.125	0.125	ms
1100	PWM frequency	4000	4000	4000	Hz
1103	Motor nominal current	4.7	4.7	4.9	A
1104	Motor maximum current	28	28	33	A
1112	Number of pole pairs	3	3	3	
1113	Torque current ratio	1.39	1.39	1.56	Nm/A
1114	EMF Voltage	92	92		V/1000 RPM
1115	Armature resistance	0.83	0.83	1.42	Ohms
1116	Armature induction	6.5	6.5	13.5	mH
1117	Motor inertia	0.00161	0.00161	0.0013	Kg/m^2
1118	Motor stand still current	7.9	7.9	6.1	A
1120	Current control gain	15.73109055	18.90908813	32.67226410	V/A
1121	Current control integrator time	2000	2000	2000	µs
1122	Motor limit current	22	22	10.6	A
1200	Number of current filters*	1	1	1	Filter 1
1201	Current filter configuration **	'FH'	'FH'	'FH'	Band-stop
1210	Current filter 1 suppression frequency	554	700	510	Hz
1211	Current filter 1 bandwidth	277	400	255	Hz
1212	Current filter 1 numerator bandwidth	200	20	80	Hz
1250	Actual current filter frequency***	100	100	100	Hz
1407	Velocity control gain	1.8	2	1.8	Nm-s/rad
1409	Velocity control integrator time	7.5	5	6	ms
1421	Velocity control integrator feedback	0	0	0	Disabled
1500	Number of velocity filters****	0	0	0	No filters

Table F.6 Axis parameters (SIMODRIVE 611)

* 0: no current filters

1: Filter 1 active

2: Filter 1 and 2 active

3: Filter 1, 2 and 3 active

4: Filter 1, 2, 3 and 4 active

** 0: Low-pass (PT2)

1: Band-stop

*** f_o frequency of the PT1 low pass filter. The time constant is: $T_I = 1/(2\pi f_o)$

**** 0: no velocity filters

1: Filter 1 active

2: Filter 1 and 2 active

SIEMENS 1FK6063-6AF71-1AG2			
Parameter	Code	Value	Units
Number of pair poles	p	6	
Rated velocity	n_{rated}	3000	RPM
Rated torque	$T_{rated\ (100k)}$	6	N-m
Rated current	$I_{rated\ (100K)}$	4.7	A
Stall torque	$T_{0\ (60K)}$	9.1	N-m
	$T_{0\ (100K)}$	11	N-m
Stall current at M_o	$I_{0\ (60K)}$	6.3	A
	$I_{0\ (100K)}$	7.9	A
Moment of inertia	J_m	16.7	10^{-4} Kg-m^2
Max. velocity	n_{max}	5300	RPM
Max. torque	T_{max}	36	N-m
Peak current	I_{max}	28	A
Limiting torque (600V)	T_{limit}	35	N-m
Limiting current (600V)	I_{limit}	28	A
Physical constants			
Torque constant	k_T	1.39	N-m/A
Voltage constant (phase to phase)	k_e	92	V/1000 RPM
Winding resistance	R	0.83	Ohm
Three-phase inductance	L	6.5	mH
Electric time constant	t_{el}	7.8	ms
Mechanical time constant	t_{mech}	2.1	ms
Thermal time constant	t_{th}	35	min
Thermal resistance	R_{th}	0.15	W/K
Mass	m_m	13.8	Kg

Table F.7 Technical data x and y-axis motor

THK HSR35 A2 SS CO OZ + 1090L H II			
Parameter	Code	Value	Units
Basic static load rating	C_o	222	KN
Basic dynamic load rating	C	123	KN
Radial rigidity (downward/upward)	k_{rl}	1.56/1.15	N/µm
Resistance under no load	F_{gw0}	15	N
Imposed load	M		Kg
Friction coefficient	μ	0.003	
Radial load	F_R	8373.816	N
Lateral load	F_L	0	N
Radial factor	X	1	
Lateral factor	Y	0.935	
LM block mass	m_{bl}	3.4	Kg
LM rail mass per unit length	m_{rl}	9.8	Kg/m
LM rail length	l_{rl}	1.09	m

Table F.8 Guideways technical data

Double Row Angular Contact Ball Bearing: RHP BSB025062DBHP3			
Parameter	Code	Value	Units
Contact angle	α	60	Degrees
Bore diameter	d_b	25	mm
Pitch circle diameter	d_{bl}		mm
Outer diameter	D_b	62	mm
Width	B_b	30	mm
Basic static load rating	C_o	40.5	KN
Axial rigidity	k_b	1000	N/μm
Moment of inertia (rotating inner ring)	J_b		Kg-cm^2
Mass	m_b	0.18	Kg
Preloading load	F_{ao}	4500	N
Limiting velocity	v_{bmax}	6000	RPM
Bearing housing ridity front/rear	k_{bh}	1.9/1.2	KN-μm
Grease lubricated DIN: K3K-30	υ=100 mm^2/s at 40°C		

Table F.9 Technical data front bearings

RHP 6305-2RSJ RE AV2S5			
Parameter	Code	Value	Units
Contact angle	α	0	Degrees
Bore diameter	d_b	25	mm
Outer diameter	D_b	62	mm
Width	B_b	17	mm
Basic static load rating	C_o	20.6	KN
Moment of inertia (rotating inner ring)	J_b		Kg-cm^2
Mass	m_b	0.235	Kg
Limiting velocity	v_{bmax}	8000	RPM
Grease lubricated DIN: K3K-30	υ=100 mm^2/s at 40°C		

Table F.10 Technical data rear bearings x and z-axis

RHP 6005-2RSJ RE AV2S5			
Parameter	Code	Value	Units
Contact angle	α	0	Degrees
Bore diameter	d_b	25	mm
Outer diameter	D_b	47	mm
Width	B_b	12	mm
Basic static load rating	C_o	10.1	KN
Moment of inertia (rotating inner ring)	J_b		Kg-cm^2
Mass	m_b	0.079	Kg
Limiting velocity	v_{bmax}	9500	RPM
Grease lubricated DIN: K3K-30	υ=100 mm^2/s at 40°C		

Table F.11 Technical data rear bearings y-axis

Parameter	Code	Value	Units
Nominal torque of coupling	T_{KN}	60	Nm
Maximum torque of coupling	$T_{K\,max}$	120	Nm
Static torsional stiffness	$C_{T\,stat.}$	2063	Nm/ rad
Dynamic torsional stiffness	$C_{T\,dyn.}$	6189	Nm/ rad
Static radial stiffness	C_r	2560	N/mm
Maximum velocity	n_{max}	6950	rpm
Mass moment of inertia per hub	J_c	200.3x10^{-6}	kg-m^2
Mass per hub	m_c	0.253	kg
Hub material: aluminium. Elastomeric element colour red, Shore 98 Sh A			

Table F.12 Coupling technical data

Parameter	Code	X-axis	Y-axis	Units
Lead	l_d	12	12	mm/rev
Outer diameter	d_s	32	32	mm
Thread minor diameter	d_c	25	25	mm
Length	l_{ss}	813.5	996	mm
Moment of Inertia per unit mass	I_o	1.57	1.57	10^{-2} Kg-cm^2/mm
Reduction ratio	n	1	1	
Efficiency	ef	0.9	0.9	%
Young's modulus	E_{ss}	206	206	KN/mm^2
Shear modulus	G_{ss}	79.6	79.6	GN/m^3
Density	ρ_{ss}	7850	7850	Kg/m^3
Mass	m_s	10.66	10.66	Kg
Material	AISI 4150 H Steel (Standard low alloy)			
Nut				
Pre-loading force	F_{ao}	1560	1560	N
Ball circle diameter	BCD	42	42	mm
Dynamic load rating	C_a	25.95	25.95	KN
Static load rating	C_{oa}	56.49	56.49	KN
Axial rigidity	k_n	677	677	N-μm
Mass	m_n	1.34	1.34	Kg
Material	SAE 8620 (common carburising steel)			
Mounting dimensions				
Front bearing position	l_f	73.5	73.5	mm
Rear bearing position	l_r	792	979	mm
Absolute position reference	l_o	184	357	mm
Stroke length	l_s	510	510	mm

Table F.13 Ball screw technical data

HEIDENHAIN ERN 1387			
Parameter	Code	Value	Units
Incremental signal: sinusoidal		1	V_{pp}
Resolution		2048	pulse/revolution
Limiting velocity	v_{bmax}	15000	RPM
Mass	m_{re}	0.25	Kg
Moment of inertia	J_{re}	2.6x10^{-6}	Kg-m^2

Table F.14 Technical data rotary encoders

HEIDENHAIN AE LS486C ML620			
Parameter	Code	Value	Units
Absolute signal: sinusoidal		1	V_{pp}
Grating period		20	μm
Measuring length		620	mm
Maximum transversing velocity	v_{lemax}	120	m/min
Mass	m_{le}	0.4	Kg
Required moving force	F_{le}	5	N

Table F.15 Technical data linear encoders

APPENDIX G MEASUREMENTS

G.1 Y-Axis Geometric Errors (Arrow 500)

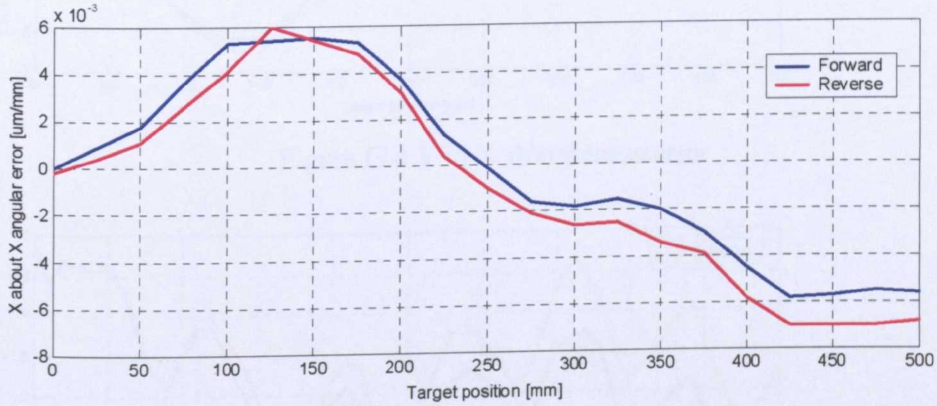


Figure G.1 X about X angular error

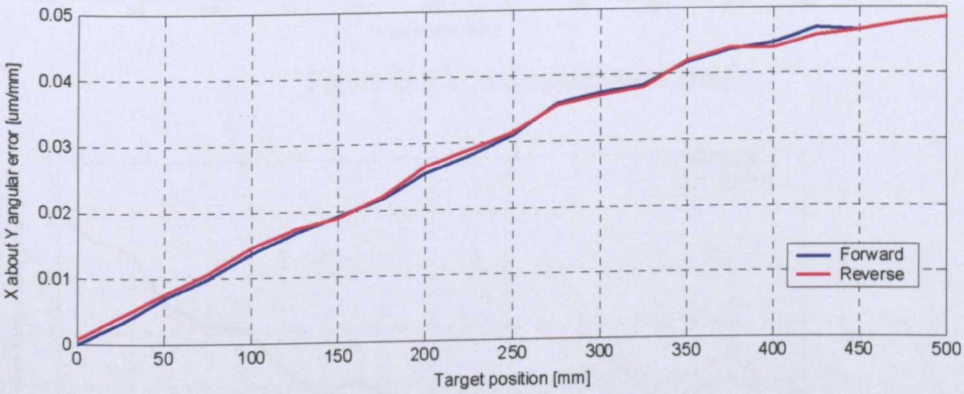


Figure G.2 X about Y angular error

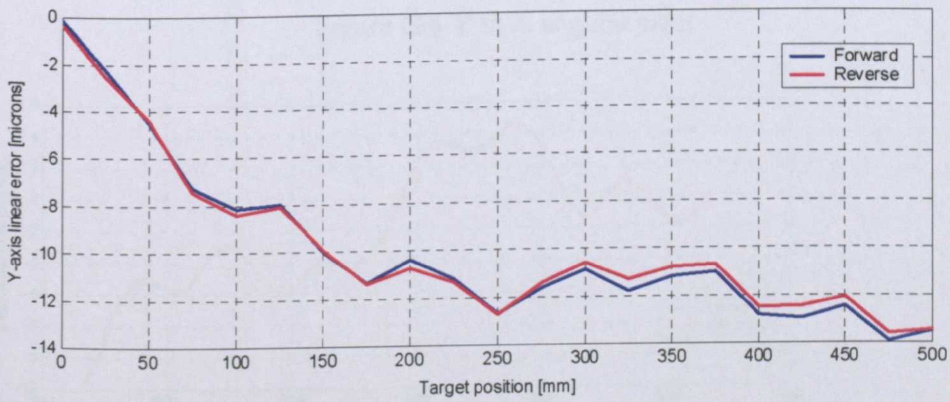


Figure G.3 Y-axis linear error

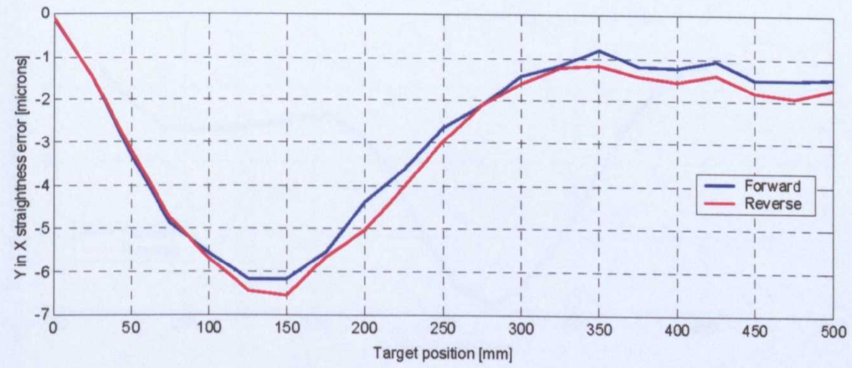


Figure G.4 Y in X straightness error

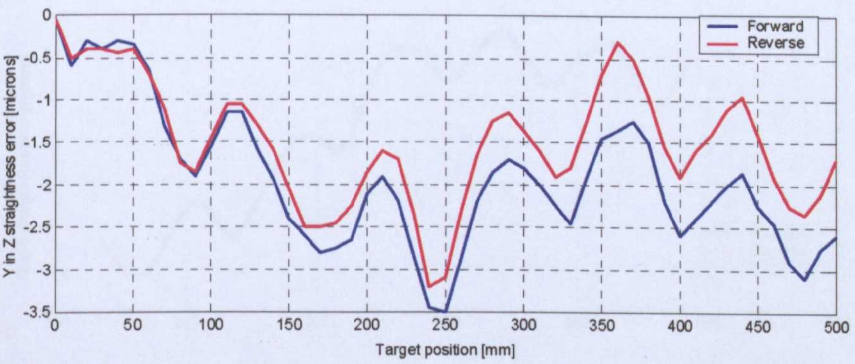


Figure G.5 Y in Z straightness error

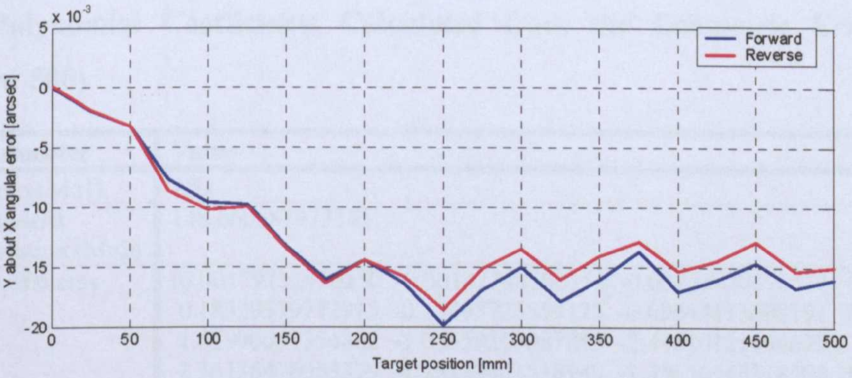


Figure G.6 Y in X angular error

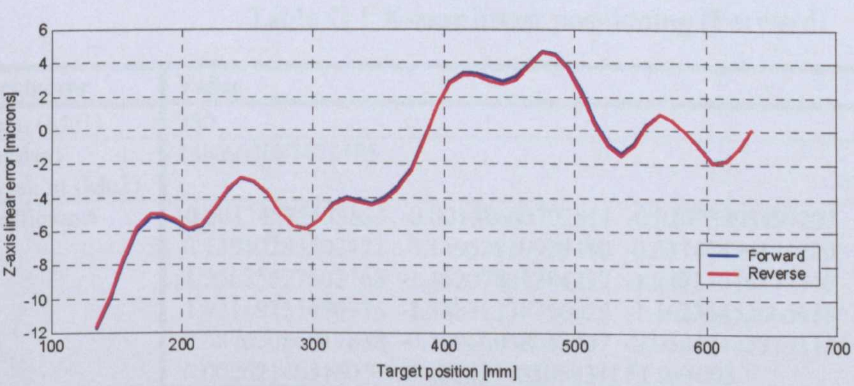


Figure G.7 Z-axis linear error

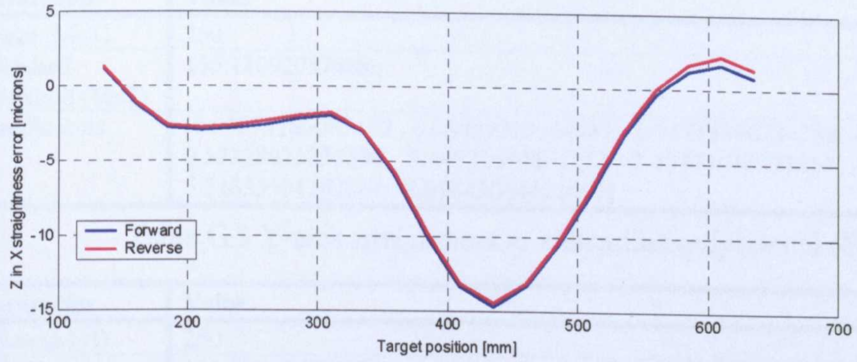


Figure G.8 Z in X straightness error

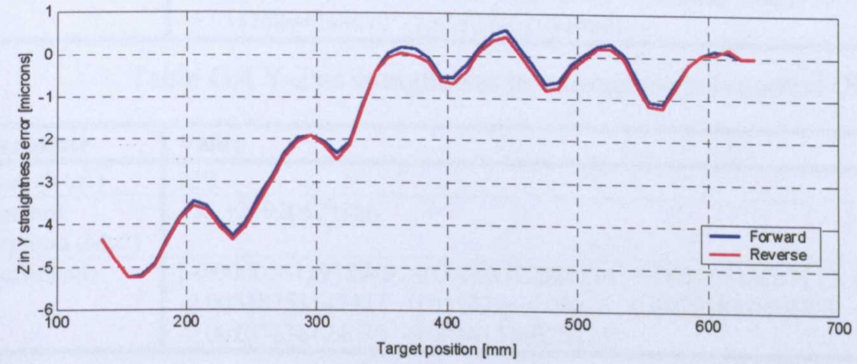


Figure G.9 Z in Y straightness error

G.2 Polynomial Coefficients Calculated from the Geometric Error Measurements (Arrow 500)

Parameter	Value
Mean (Mu1)	250
Standard deviation (Mu2)	148.660687473185
Coefficients	[0.00179732510250 -0.00142748300352 -0.02758470890611 0.02119522726521 0.18329579772915 -0.13495725559125 -0.69044433698191 0.48151465521848 1.61990069354882 -1.05558200087885 -2.44510125444629 1.46623137237084 2.36136476053379 -1.28239441518949 -1.39656548318502 0.67618230713412 0.45549967295162 -0.19418906530361 -0.06297691669556 0.02543826072979 0.00288196289205 0.00213422850003]*1.0e+003

Table G.1 X-axis linear positioning (Forward)

Parameter	Value
Mean (Mu1)	250
Standard deviation (Mu2)	148.660687473185
Coefficients	[0.00134575433833 -0.00158660702811 -0.02078998497203 0.02352618548467 0.13940283502421 -0.14952920928440 -0.53147490131220 0.53214470619456 1.26625527602768 -1.16207801284227 -1.94731516274546 1.60415167414940 1.92119751496976 -1.38832238756029 -1.16236452665438 0.71866124294236 0.38760086917838 -0.19966068652157 -0.05442475979112 0.02451906783458 0.00262146516057 0.00242346505934] *1.0e+003

Table G.2 X-axis linear positioning (Reverse)

Parameter	Value
Mean (Mu1)	250
Standard deviation (Mu2)	155.120920574886
Coefficients	[0.03974186087169 0.08059910944983 -0.11715166784588 -0.94229616554107 0.52328935255338 3.44322694204112 -3.43371614233664 -3.26497247200367 5.21635501182507 -2.69843044422061]

Table G.3 Y-axis straightness in x direction polynomial (Forward)

Parameter	Value
Mean (Mu1)	250
Standard deviation (Mu2)	155.120920574886
Coefficients	[0.17854334961051 0.20359509493929 -1.02889000132959 -1.58242506505894 2.72495368420529 4.45231569842000 -5.74888277080710 -3.66662807702597 5.95470894283670 -2.95183462964080]

Table G.4 Y-axis straightness in x direction polynomial (Reverse)

Parameter	Value
Mean (Mu1)	250
Standard deviation (Mu2)	155.120920574886
Coefficients	[-0.00005811978949 0.00063322587214 0.00110640947717 -0.00472170699316 -0.00518753147447 0.01137940204076 0.00765812889905 -0.00789240049935 -0.00107236124039 -0.00141516022175]

Table G.5 Y-axis rotation about z-axis (Forward)

Parameter	Value
Mean (Mu1)	250
Standard deviation (Mu2)	155.120920574886
Coefficients	[0.00006802545427 0.00055192788308 0.00051999904004 -0.00446329512048 -0.00430539513897 0.01127494228493 0.00718200162211 -0.00800216323090 -0.00103354790038 -0.00151698961665]

Table G.6 Y-axis rotation about z-axis (Reverse)

Parameter	Value
Mean (Mu1)	250
Standard deviation (Mu2)	155.120920574886
Coefficients	[0.59682881931994 -0.73981047189995 -3.33809706675324 4.14459990655644 6.39894895305915 -7.17517388110818 -6.31768217820217 5.53714427861941 0.40000017068817 -11.81409209327226]

Table G.7 Y-axis linear positioning (Forward)

Parameter	Value
Mean (Mu1)	250
Standard deviation (Mu2)	155.120920574886
Coefficients	[0.68943248082590 -0.82435006112284 -3.98348234506237 4.59161305985440 7.97008783102876 -8.03331035389856 -7.96188888747018 6.19288498536962 1.25882090827488 -11.81449599039490]

Table G.8 Y-axis linear positioning (Reverse)

Parameter	Value
Mean (Mu1)	250
Standard deviation (Mu2)	155.120920574886
Coefficients	[-0.09299345120317 -0.31499377398290 -0.17793547816669 2.09929244367061 2.37955078989997 -5.16258537089714 -2.66413689167969 4.58659654079912 -2.78687237713905 -1.10895075063088]

Table G.9 X-axis straightness in y direction (Forward)

Parameter	Value
Mean (Mu1)	250
Standard deviation (Mu2)	155.120920574886
Coefficients	[-0.05357641507915 -0.51224653650652 -0.29835351655399 3.03845440163535 2.37291530376561 -6.43132234604616 -2.50196285985755 5.00057134312902 -2.88022484951574 -1.07239114807921]

Table G.10 X-axis straightness in y direction (Reverse)

Parameter	Value
Mean (Mu1)	250
Standard deviation (Mu2)	155.120920574886
Coefficients	[-0.00005811978949 0.00063322587214 0.00110640947717 -0.00472170699316 -0.00518753147447 0.01137940204076 0.00765812889905 -0.00789240049935 -0.00107236124039 -0.00141516022175]

Table G.11 X-axis rotation about z-axis (Forward)

Parameter	Value
Mean (Mu1)	250
Standard deviation (Mu2)	155.120920574886
Coefficients	[0.00006802545427 0.00055192788308 0.00051999904004 -0.00446329512048 -0.00430539513897 0.01127494228493 0.00718200162211 -0.00800216323090 -0.00103354790038 -0.00151698961665]

Table G.12 X-axis rotation about z-axis (Reverse)

G.3 MATLAB Program for the Calculation of the Polynomial Coefficients

```
% get_poly uses the MATLAB function polyfit to define a polynom that fits the measured error data. The
% polynom will be used to determine the errors on the whole axis stroke
% The program assumes the measured errors are recorded in 13 files, which are loaded one by one
clc
clear all
for n=1:13
switch n
    case 1
      load x_linear_error; y_lab=strcat(name(1),strrep(name(5),'scale:',''));
    case 2
      load x_y_strightness_error; y_lab=strcat(name(1),strrep(name(5),'scale:',''));
    case 3
      load x_z_strightness_error; y_lab=strcat(name(1),strrep(name(5),'scale:',''));
    case 4
      load x_x_angular_error; y_lab=strcat(name(1),strrep(name(5),'scale:',''));
    case 5
      load x_y_angular_error; y_lab=strcat(name(1),strrep(name(5),'scale:',''));
```

```
    case 6
      load x_z_angular_error; y_lab=strcat(name(1),strrep(name(5),'scale:',''));
    case 7
      load y_linear_error; y_lab=strcat(name(1),strrep(name(5),'scale:',''));
    case 8
      load y_x_strightness_error; y_lab=strcat(name(1),strrep(name(5),'scale:',''));
    case 9
      load y_z_strightness_error; y_lab=strcat(name(1),strrep(name(5),'scale:',''));
    case 10
      load y_x_angular_error; y_lab=strcat(name(1),strrep(name(5),'scale:',''));
    case 11
      load z_linear_error; y_lab=strcat(name(1),strrep(name(5),'scale:',''));
    case 12
      load z_x_strightness_error; y_lab=strcat(name(1),strrep(name(5),'scale:',''));
    case 13
      load z_y_strightness_error; y_lab=strcat(name(1),strrep(name(5),'scale:',''));
       end
       len=length(pos);
       for i=1:len
               tp(i,1)=pos(i);
               fwd_mean(i,1)=mean(fwd(i,:));
               rev_mean(i,1)=mean(rev(i,:));
       end
       plot(tp,fwd_mean,'b',tp,rev_mean,'r')
       grid
       xlabel('Target position [mm]')
       ylabel(y_lab)
       [max(fwd_mean) max(rev_mean);min(fwd_mean) min(rev_mean)]
pause
       deg=ceil(length(pos)*.4); % polynom degree
       [poly_fwd, s,miu_f]=polyfit(tp,fwd_mean,deg);
       [poly_rev,s,miu_r]=polyfit(tp,rev_mean,deg);
switch n
    case 1
      save x_linear name tp fwd_mean rev_mean  poly_fwd poly_rev  miu_f miu_r
    case 2
      save x_y_strightness name tp fwd_mean rev_mean  poly_fwd poly_rev  miu_f miu_r
    case 3
      save x_z_strightness name tp fwd_mean rev_mean  poly_fwd poly_rev  miu_f miu_r
    case 4
      save x_x_angular name tp fwd_mean rev_mean  poly_fwd poly_rev  miu_f miu_r
    case 5
      save x_y_angular name tp fwd_mean rev_mean  poly_fwd poly_rev  miu_f miu_r
    case 6
      save x_z_angular name tp fwd_mean rev_mean  poly_fwd poly_rev  miu_f miu_r
    case 7
      save y_linear name tp fwd_mean rev_mean  poly_fwd poly_rev  miu_f miu_r
    case 8
      save y_x_strightness name tp fwd_mean rev_mean  poly_fwd poly_rev  miu_f miu_r
    case 9
      save y_z_strightness name tp fwd_mean rev_mean  poly_fwd poly_rev  miu_f miu_r
    case 10
      save y_x_angular name tp fwd_mean rev_mean  poly_fwd poly_rev miu_f miu_r
    case 11
      save z_linear name tp fwd_mean rev_mean  poly_fwd poly_rev  miu_f miu_r
    case 12
      save z_x_strightness name tp fwd_mean rev_mean  poly_fwd poly_rev  miu_f miu_r
    case 13
      save z_y_strightness name tp fwd_mean rev_mean  poly_fwd poly_rev miu_f miu_r
      end
      hold on
```

```
        plot(tp,polyval(poly_fwd,tp,[],miu_f),'c',tp,polyval(poly_rev,tp,[],miu_r),'m')
    legend('Forward','Reverse','Forward (pol)', 'Reverse(pol)',0)
        hold off
    pause
end
%POLYVAL Evaluate polynomial.
%   Y = POLYVAL(P,X), when P is a vector of length N+1 whose elements are the coefficients of a polynomial,
%   is the value of the polynomial evaluated at X.
%       Y = P(1)*X^N + P(2)*X^(N-1) + ... + P(N)*X + P(N+1)
%   If X is a matrix or vector, the polynomial is evaluated at all points in X.  See also POLYVALM for
%   evaluation in a matrix sense.
%   Y = POLYVAL(P,X,[],MU) uses XHAT = (X-MU(1))/MU(2) in place of X.
%   The centering and scaling parameters MU are optional output computed by POLYFIT.
%   [Y,DELTA] = POLYVAL(P,X,S) or [Y,DELTA] = POLYVAL(P,X,S,MU) uses the optional output
%   structure S provided by POLYFIT to generate  error estimates, Y +/- delta.  If the errors in the data input to
%   POLYFIT are independent normal with constant variance, Y +/- DELTA contains at least 50% of the
%   predictions.
```

G.4 Step Velocity Response Measurements for the Test Rig

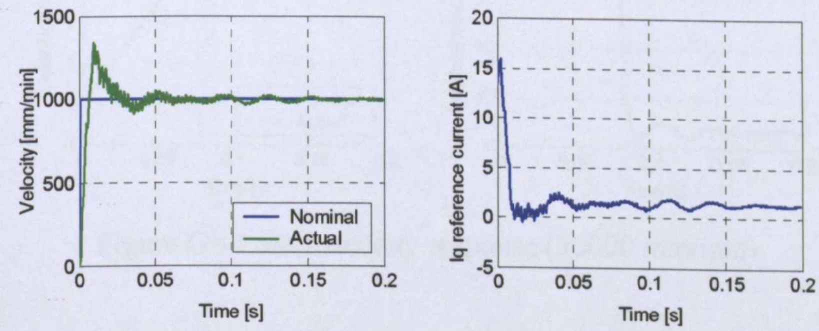


Figure G.10 Step velocity response (1000 mm/min)

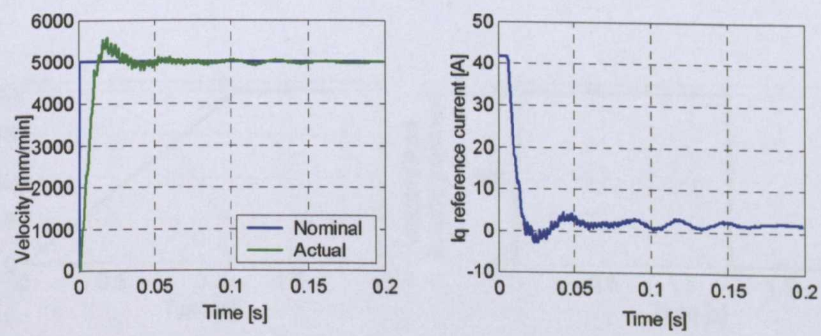


Figure G.11 Step velocity response (5000 mm/min)

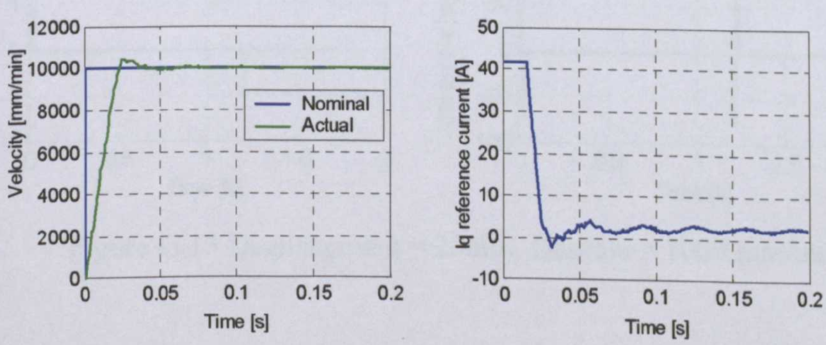


Figure G.12 Step velocity response (10000 mm/min)

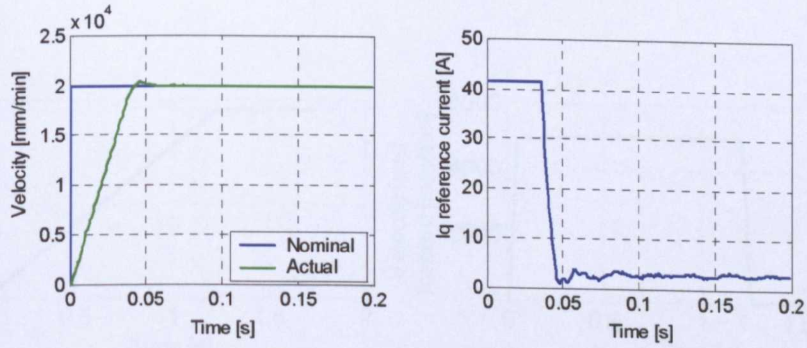


Figure G.13 Step velocity response (20000 mm/min)

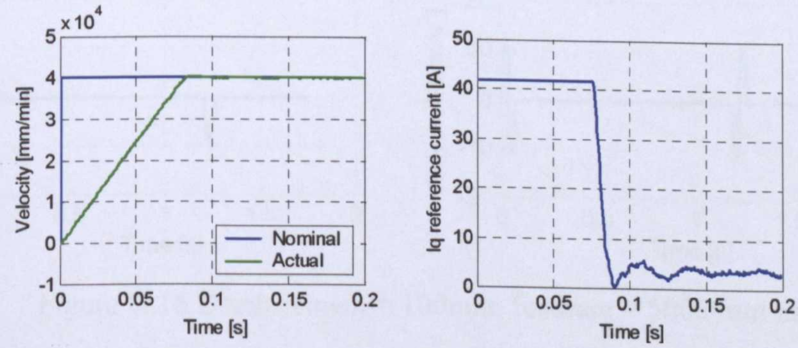


Figure G.14 Step velocity response (30000 mm/min)

G.5 Measured Jerk-limited Motion Profiles for the Test Rig

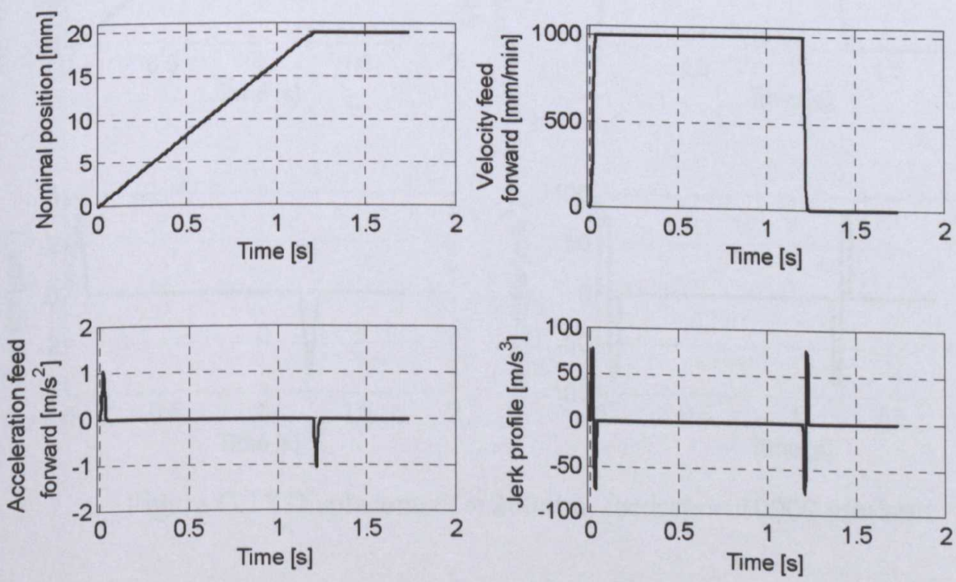


Figure G.15 Displacement = 20mm, feedrate = 1000 mm/min

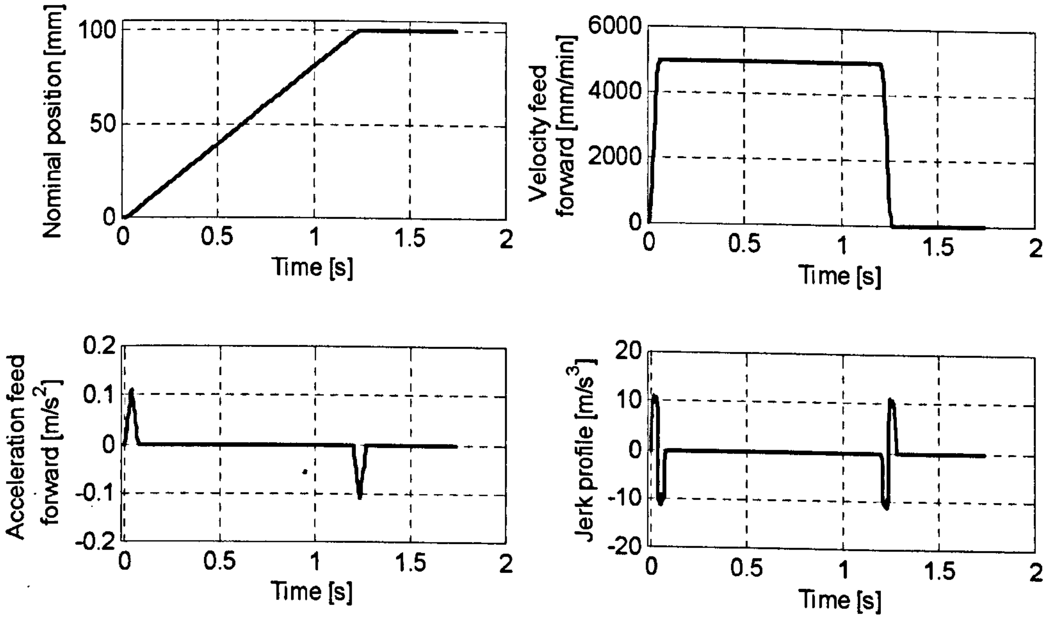


Figure G.16 Displacement = 100mm, feedrate = 5000 mm/min

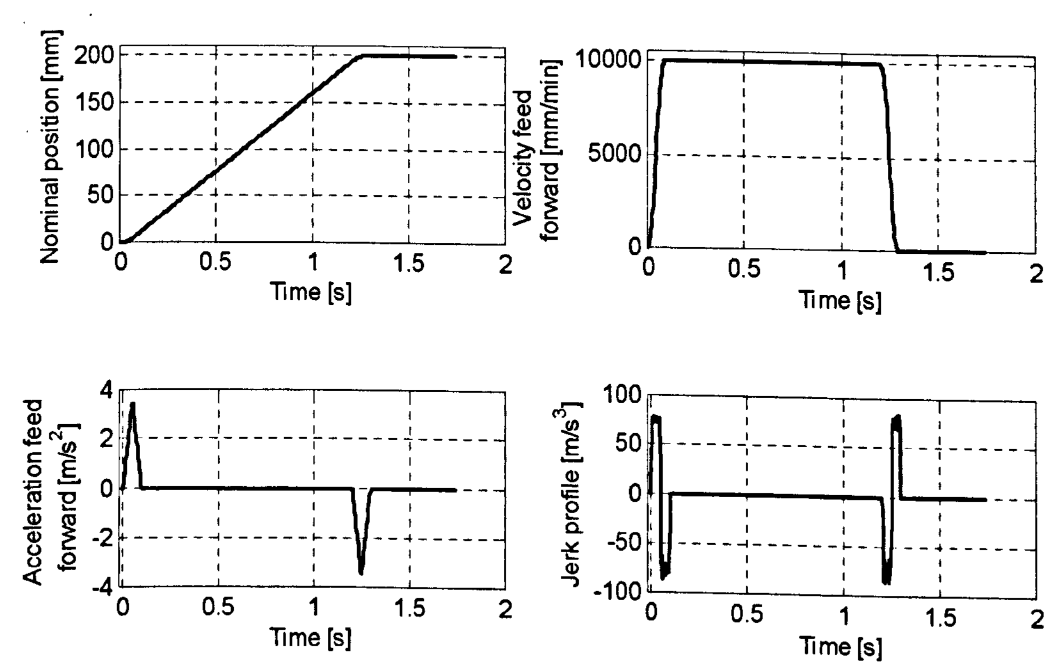


Figure G.17 Displacement = 200mm, feedrate = 10000 mm/min

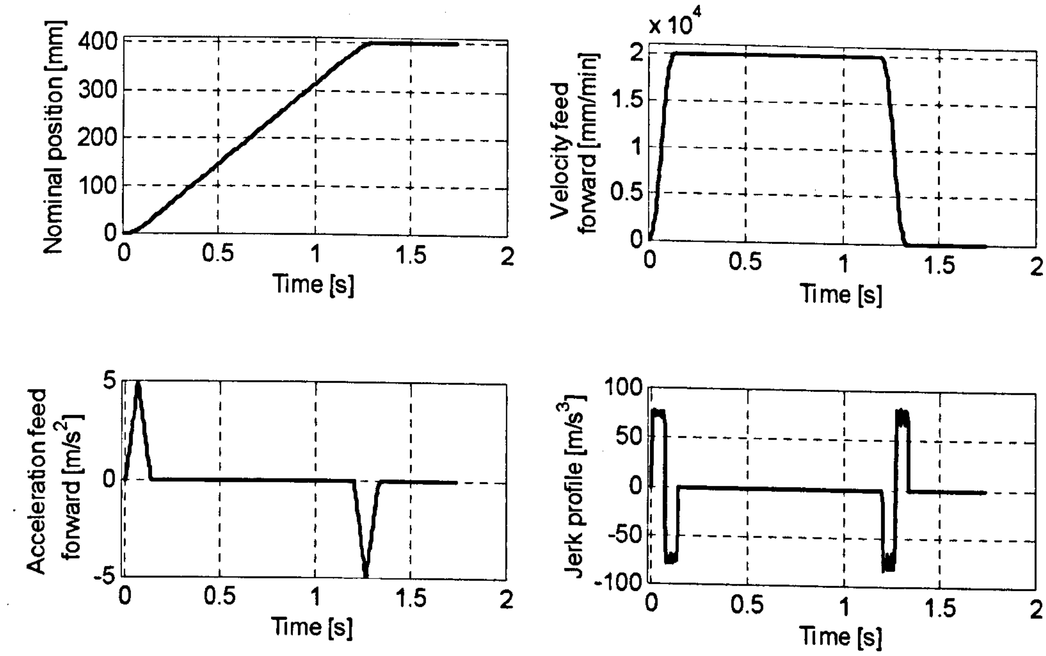


Figure G.18 Displacement = 400mm, feedrate = 20000 mm/min

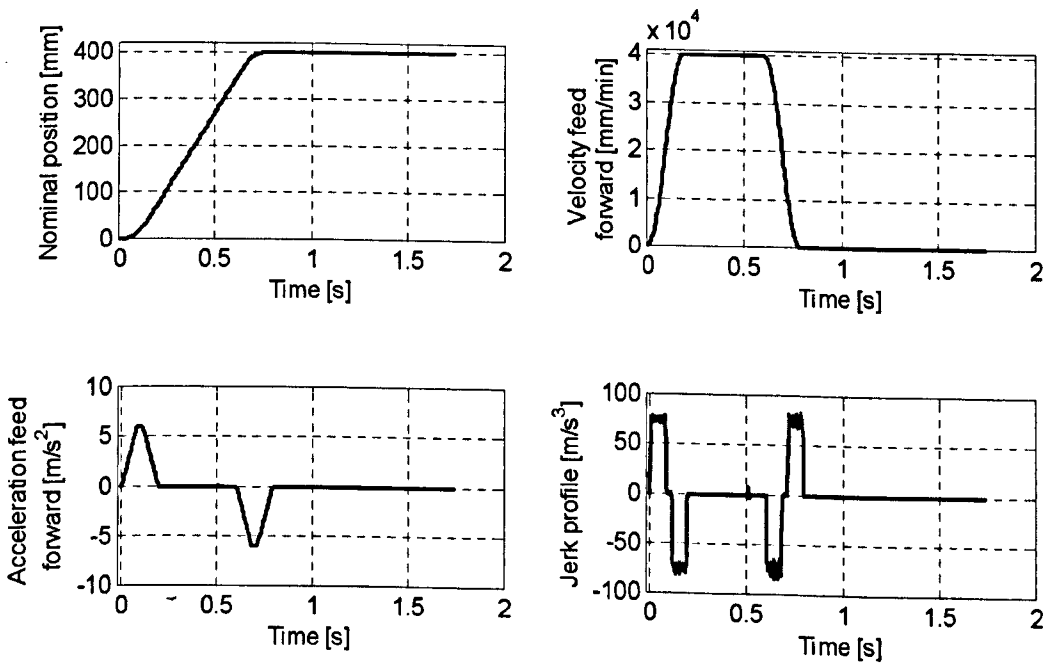


Figure G.19 Displacement = 400mm, feedrate = 40000 mm/min

G.6 Jerk-limited Velocity Response Measurements for the Test Rig

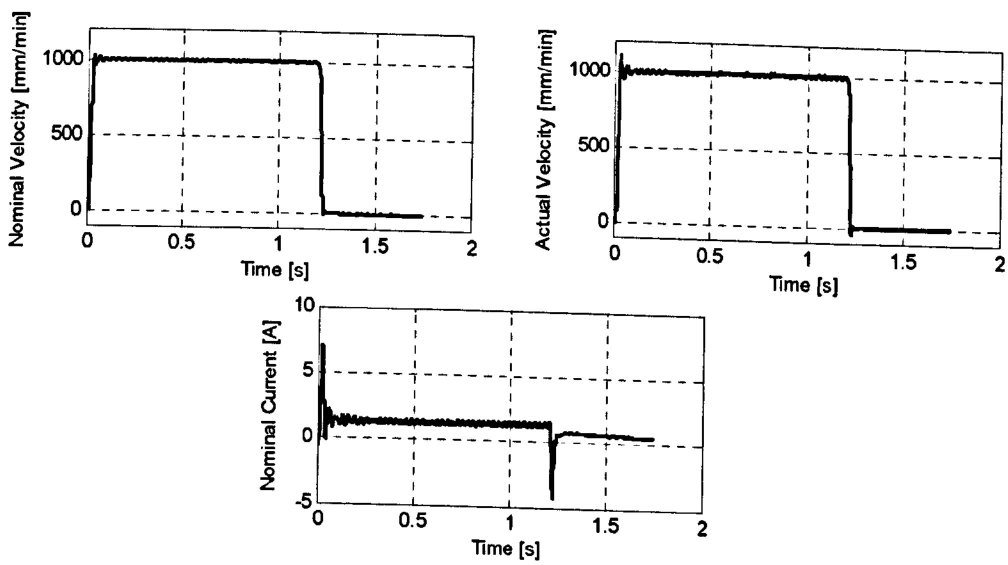


Figure G.20 Displacement = 20mm, feedrate = 1000 mm/min

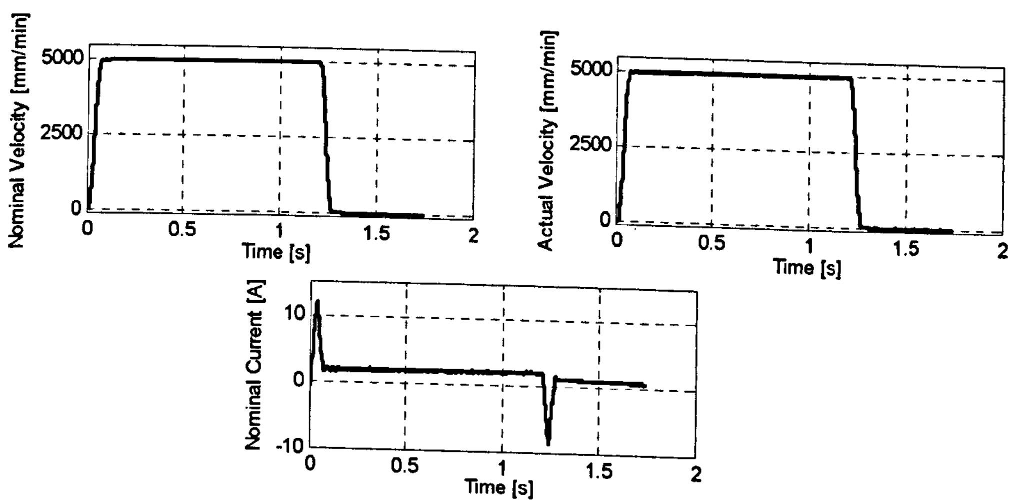


Figure G.21 Displacement = 100mm, feedrate = 5000 mm/min

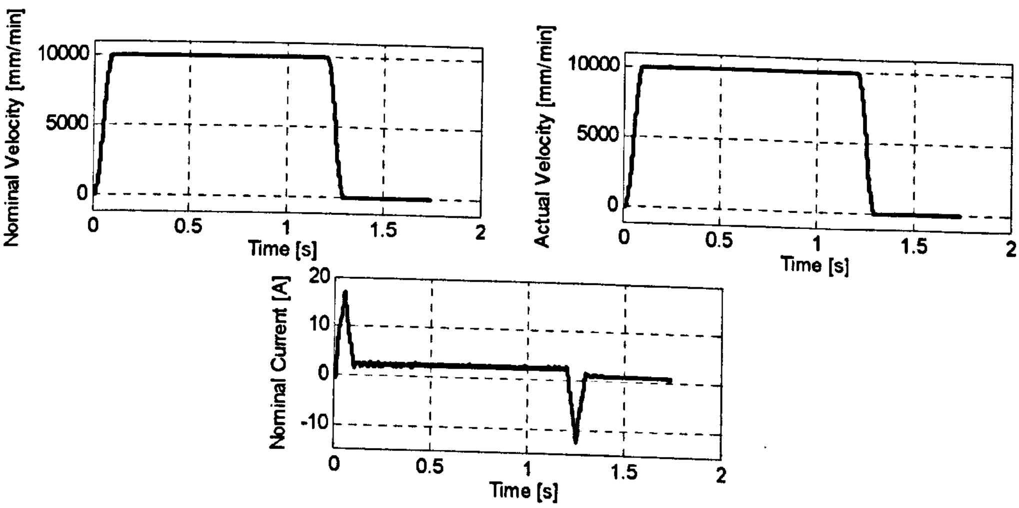


Figure G.22 Displacement = 200mm, feedrate = 10000 mm/min

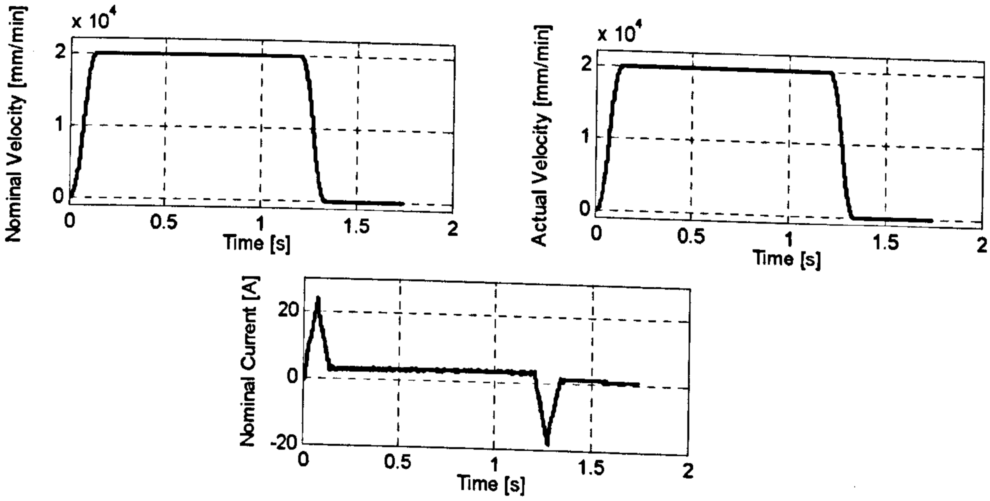


Figure G.23 Displacement = 400mm, feedrate = 20000 mm/min

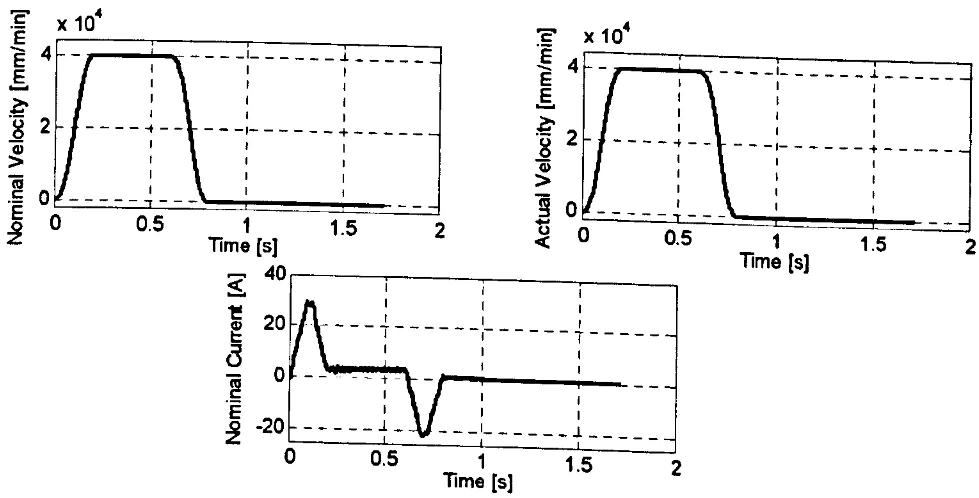


Figure G.24 Displacement = 400mm, feedrate = 40000 mm/min

G.7 Jerk-limited Position Response Measurements for the Test Rig

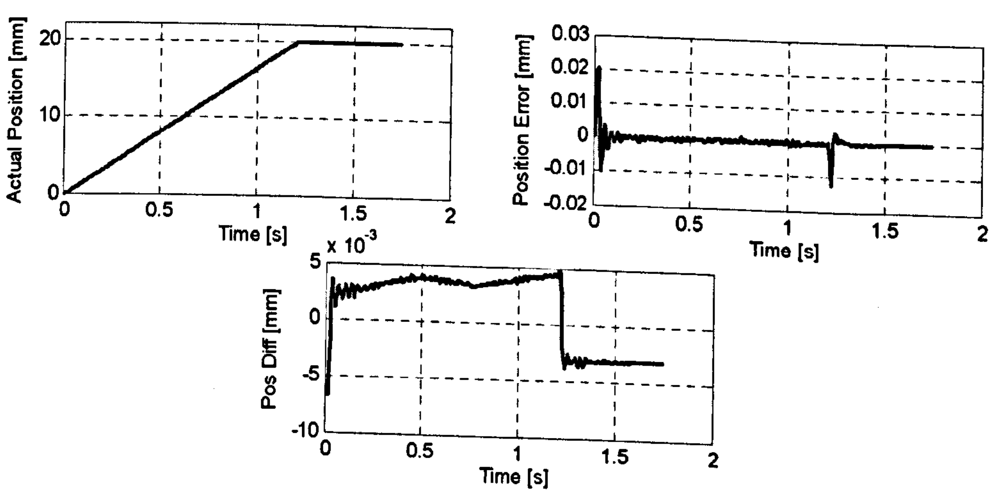


Figure G.25 Displacement = 20mm, feedrate = 1000 mm/min

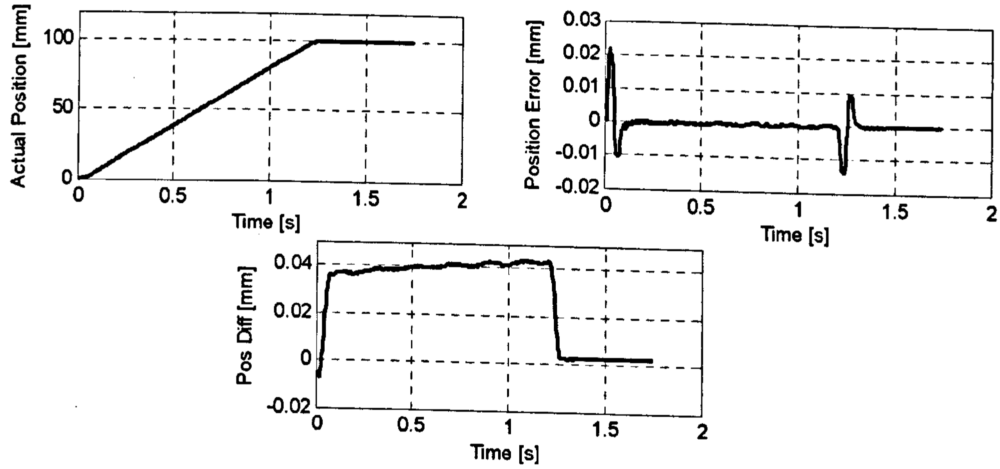


Figure G.26 Displacement = 100mm, feedrate = 5000 mm/min

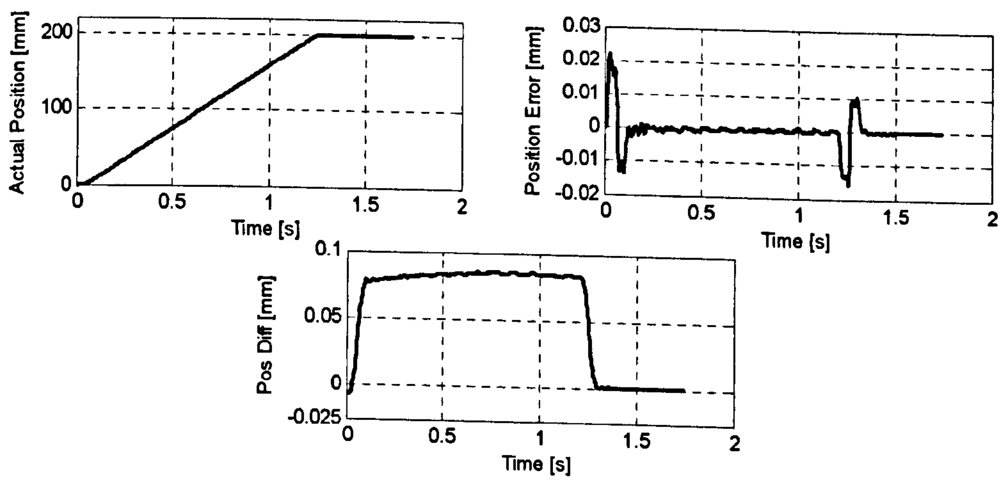


Figure G.27 Displacement = 200mm, feedrate = 10000 mm/min

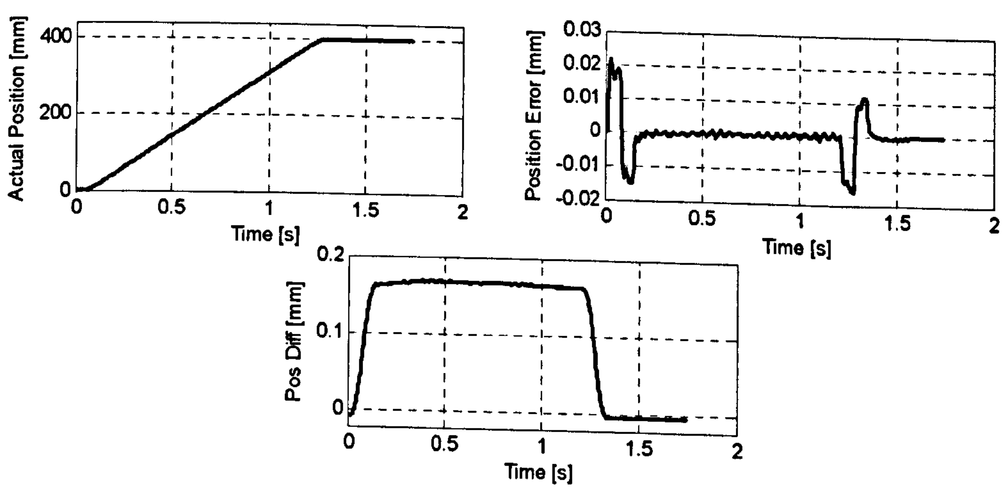


Figure G.28 Displacement = 400mm, feedrate = 20000 mm/min

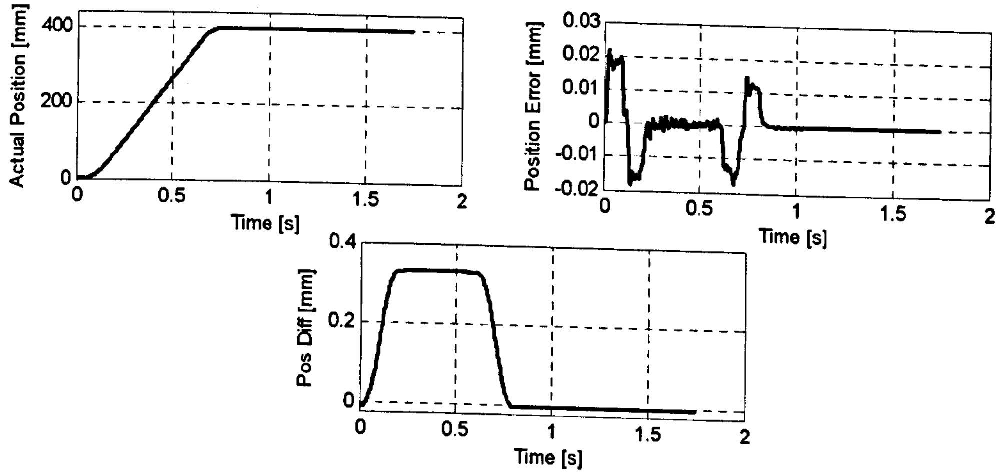


Figure G.29 Displacement = 400mm, feedrate = 40000 mm/min

G.8 Jerk-limited Velocity Response Measurements for the x-axis of the Arrow 500

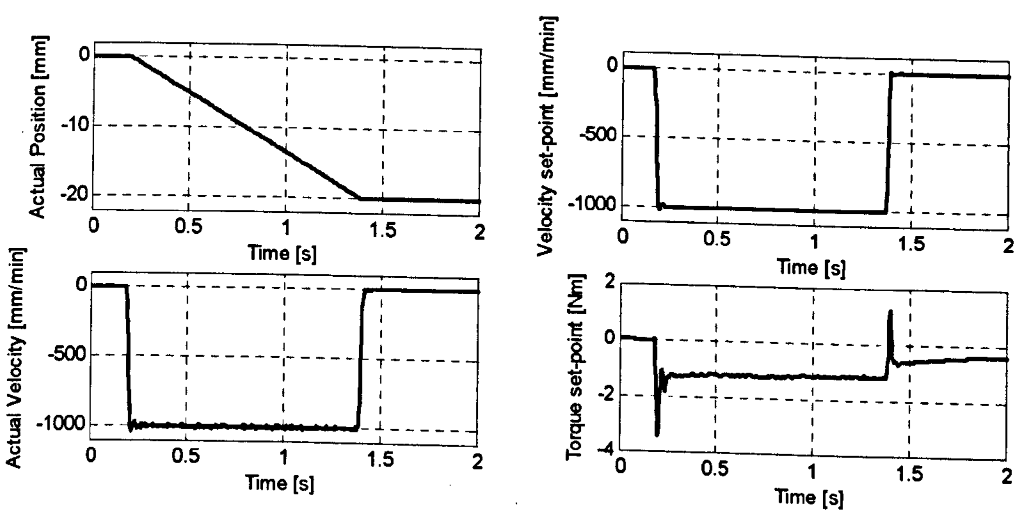


Figure G.30 Feedrate = 1000 mm/min (x-axis)

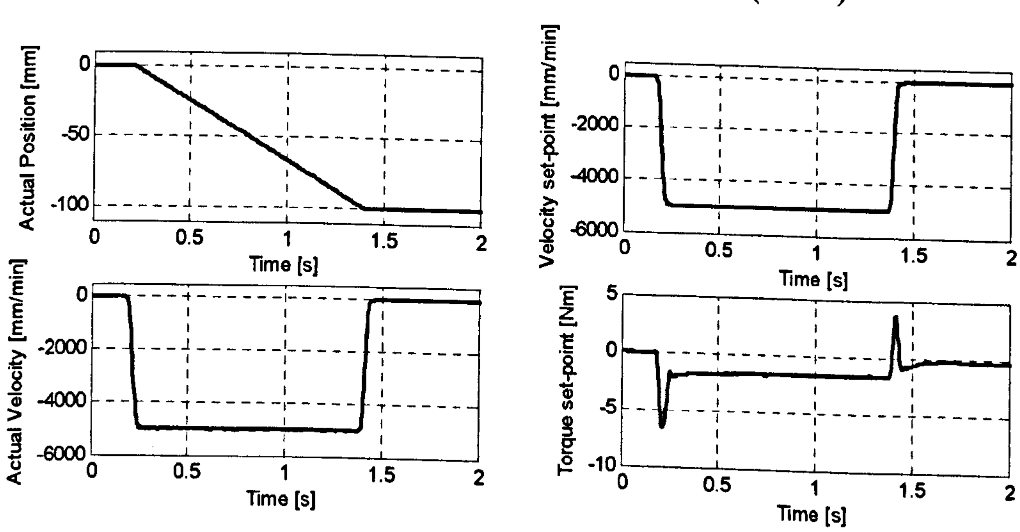


Figure G.31 Feedrate = 5000 mm/min (x-axis)

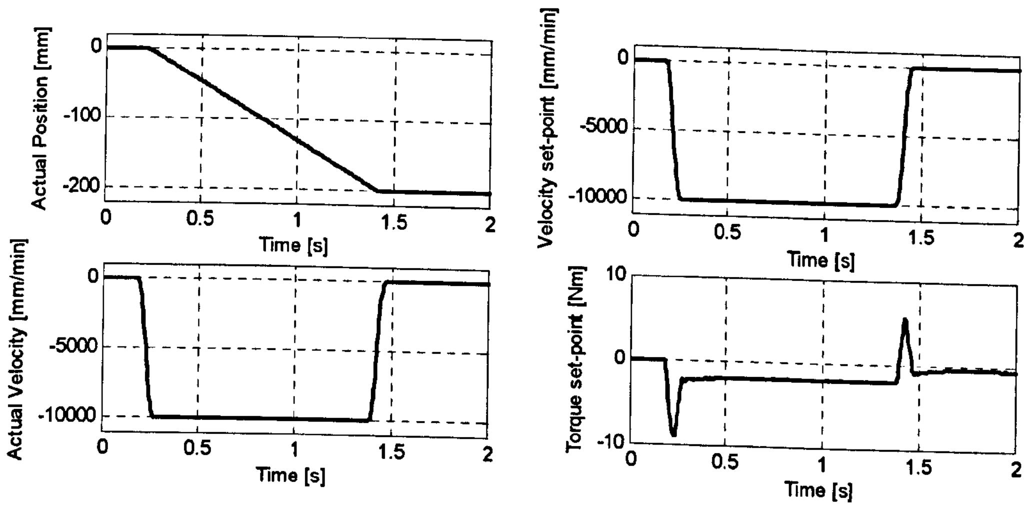


Figure G.32 Feedrate = 10000 mm/min (x-axis)

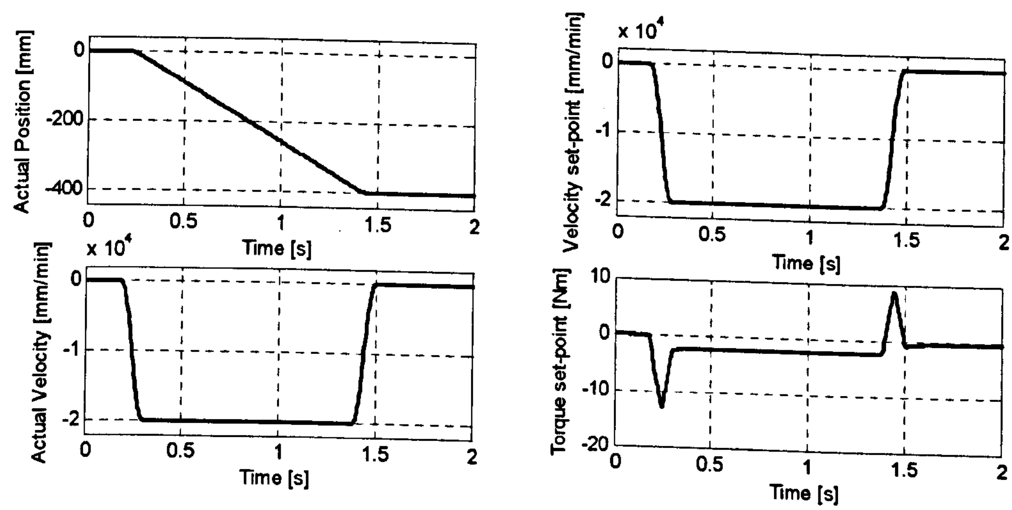


Figure G.33 Feedrate = 20000 mm/min (x-axis)

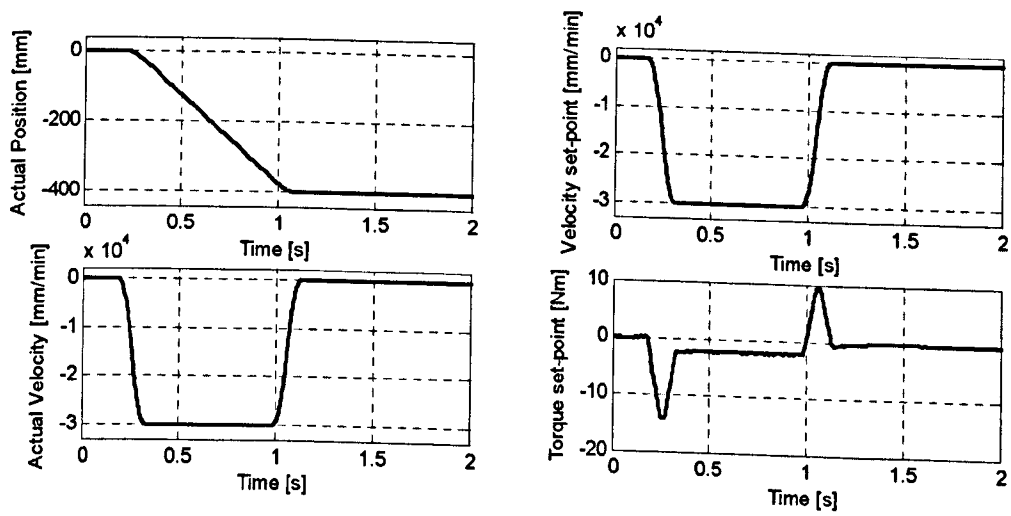


Figure G.34 Feedrate = 30000 mm/min (x-axis)

G.9 Jerk-limited Velocity Response Measurements for the y-axis of the Arrow 500

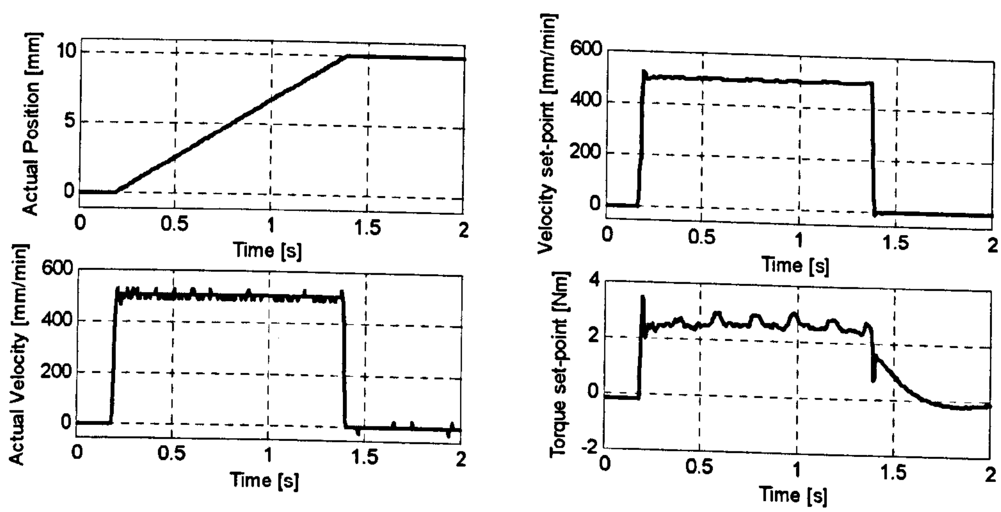


Figure G.35 Feedrate = 500 mm/min (y-axis)

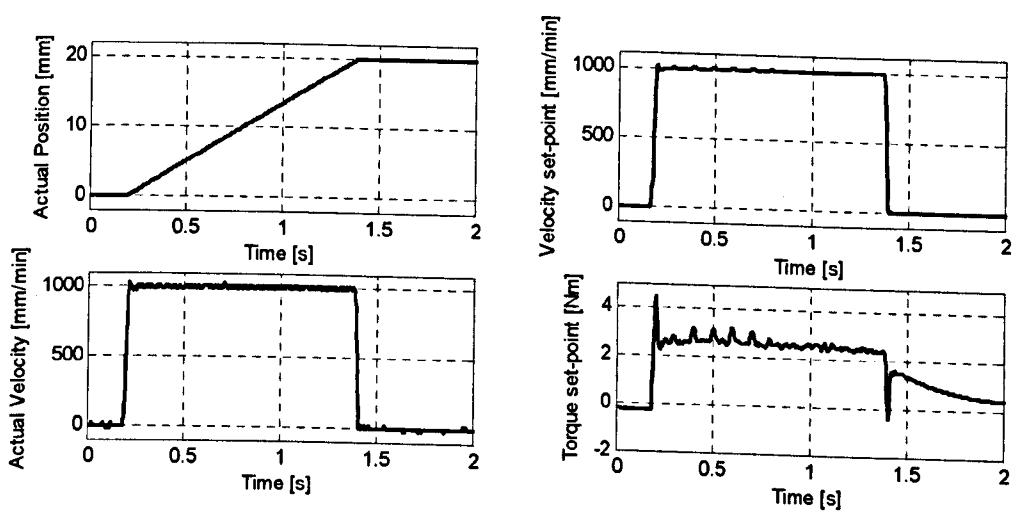


Figure G.36 Feedrate = 1000 mm/min (y-axis)

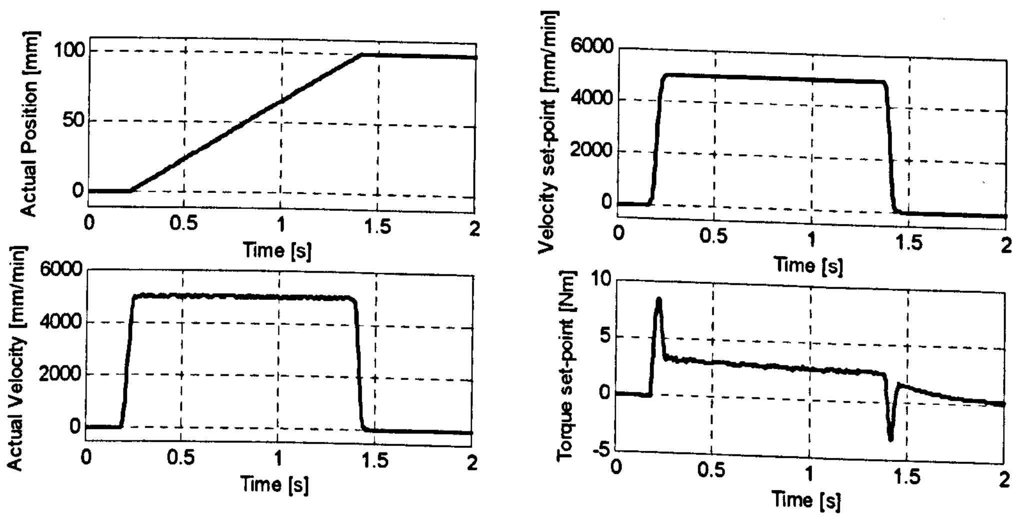


Figure G.37 Feedrate = 5000 mm/min (y-axis)

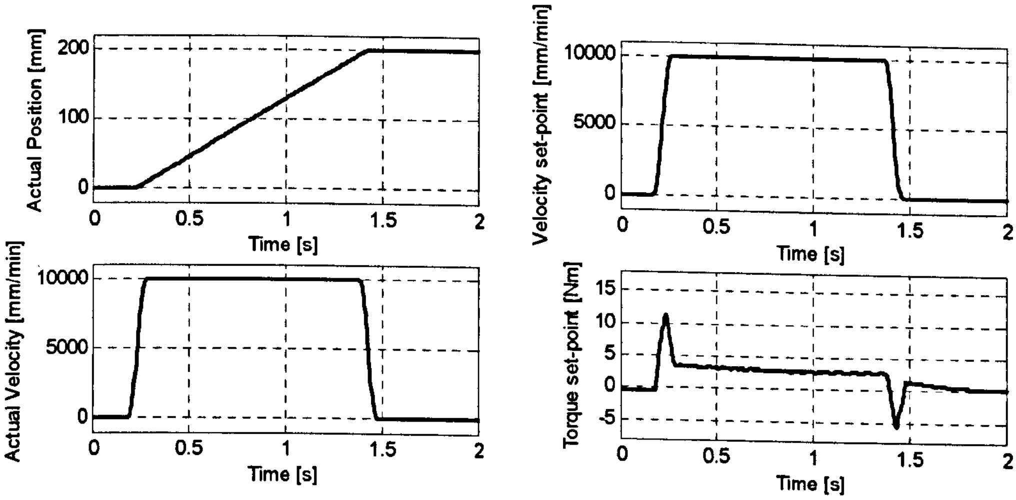


Figure G.38 Feedrate = 10000 mm/min (y-axis)

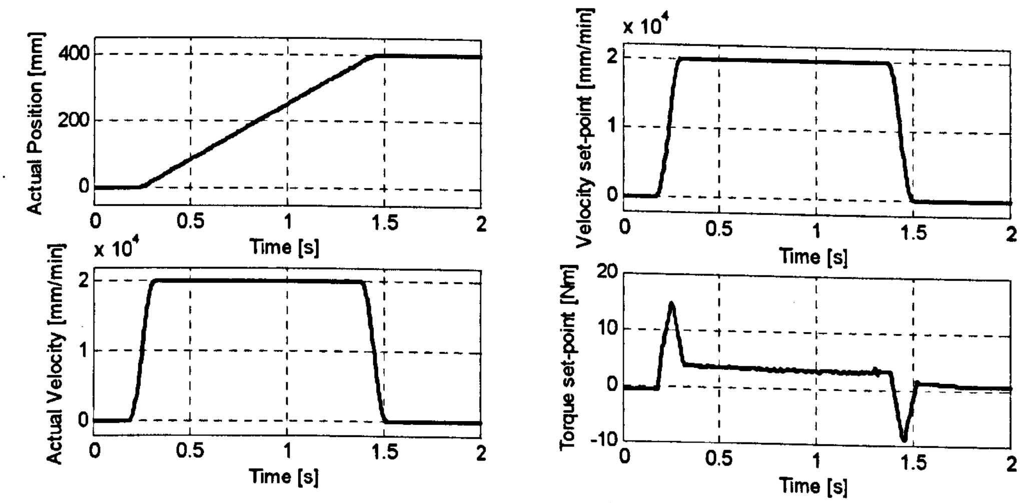


Figure G.39 Feedrate = 20000 mm/min (y-axis)

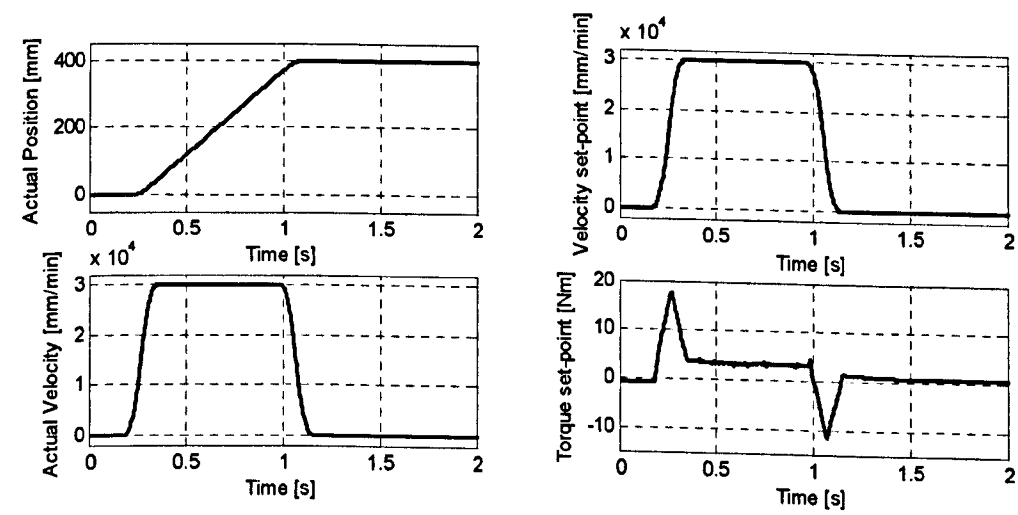


Figure G.40 Feedrate = 30000 mm/min (y-axis)

G.10 Jerk-limited Position Response Measurements for the x-axis of the Arrow 500

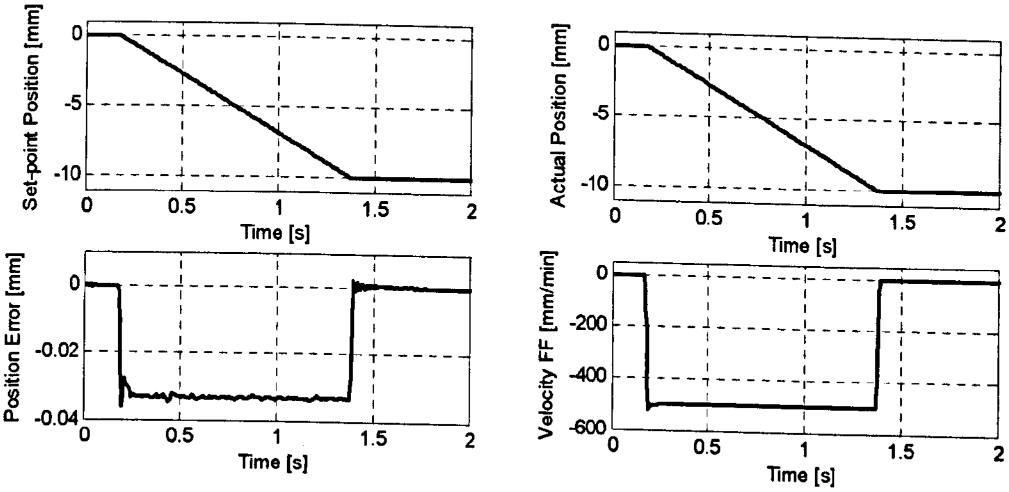


Figure G.41 Displacement = 10mm, feedrate = 500 mm/min (x-axis)

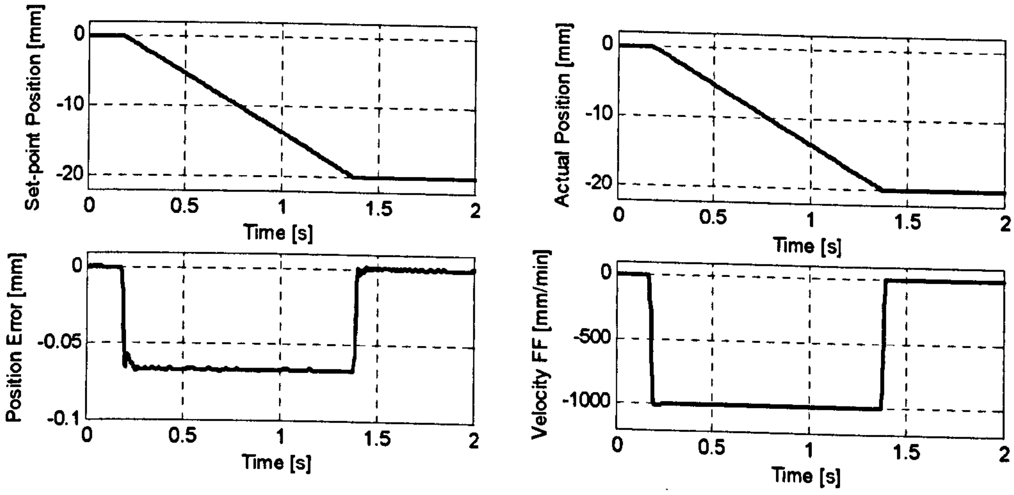


Figure G.42 Displacement = 20mm, feedrate = 1000 mm/min (x-axis)

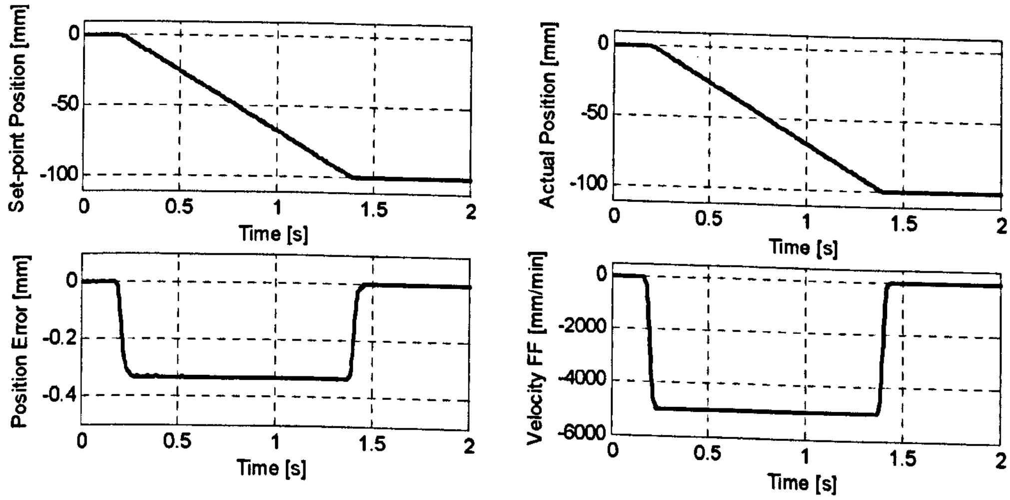


Figure G.43 Displacement = 100mm, feedrate = 5000 mm/min (x-axis)

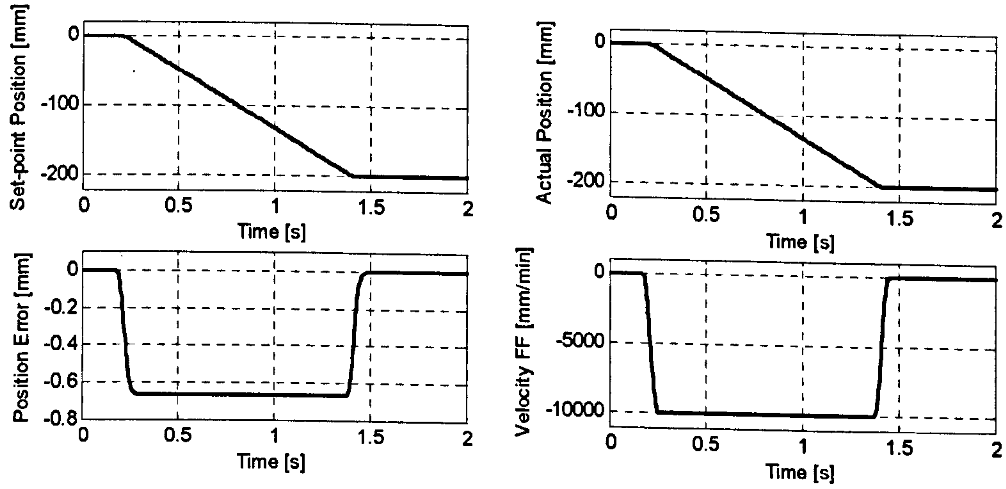


Figure G.44 Displacement = 200 mm, feedrate = 10000 mm/min (x-axis)

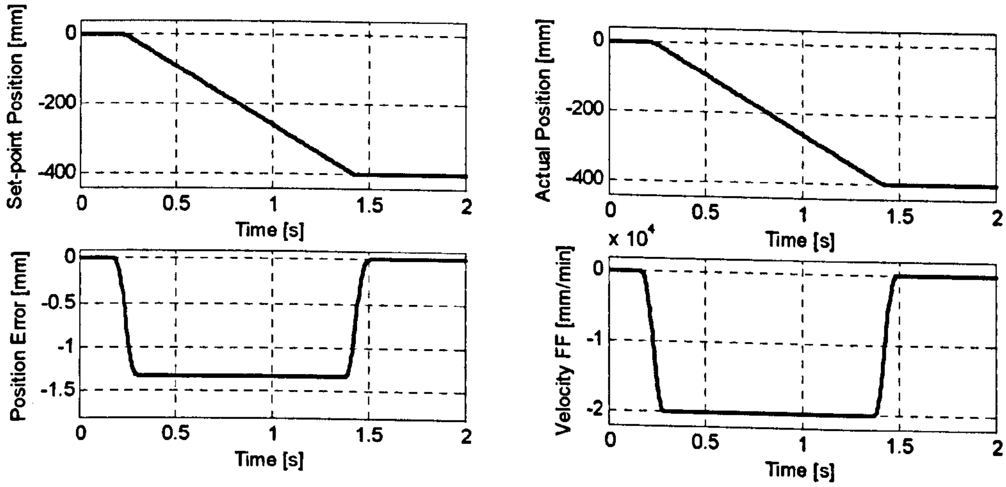


Figure G.45 Displacement = 400 mm, feedrate = 20000 mm/min (x-axis)

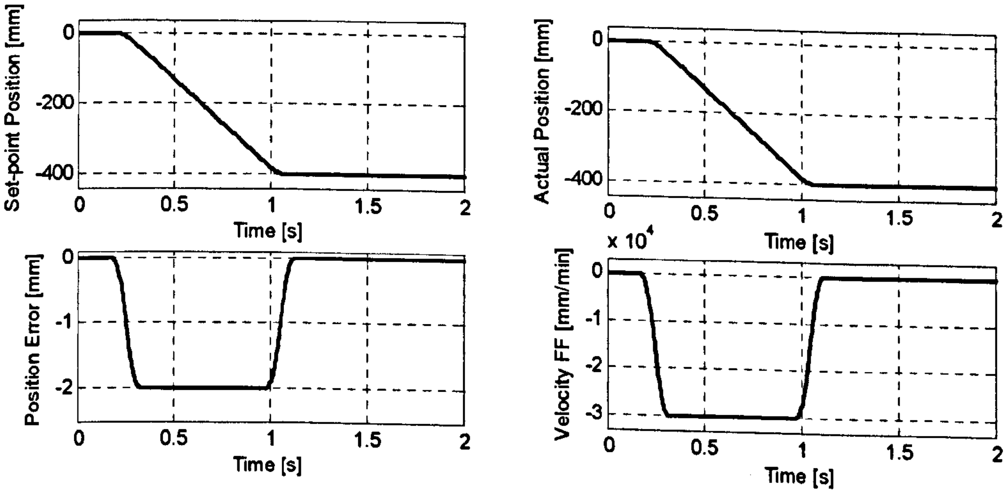


Figure G.46 Displacement = 400 mm, feedrate = 30000 mm/min (x-axis)

G.11 Jerk-limited Position Response Measurements for the y-axis of the Arrow 500

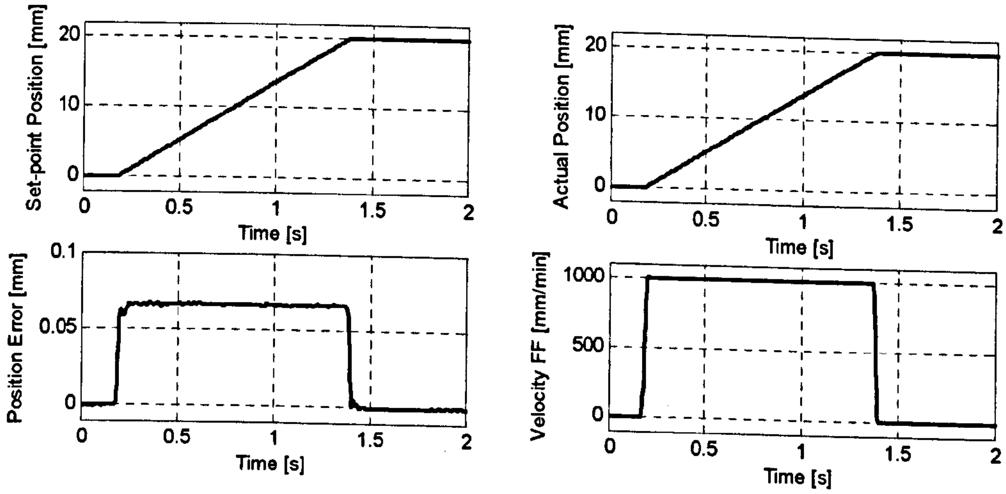


Figure G.47 Displacement = 20 mm, feedrate = 1000 mm/min (y-axis)

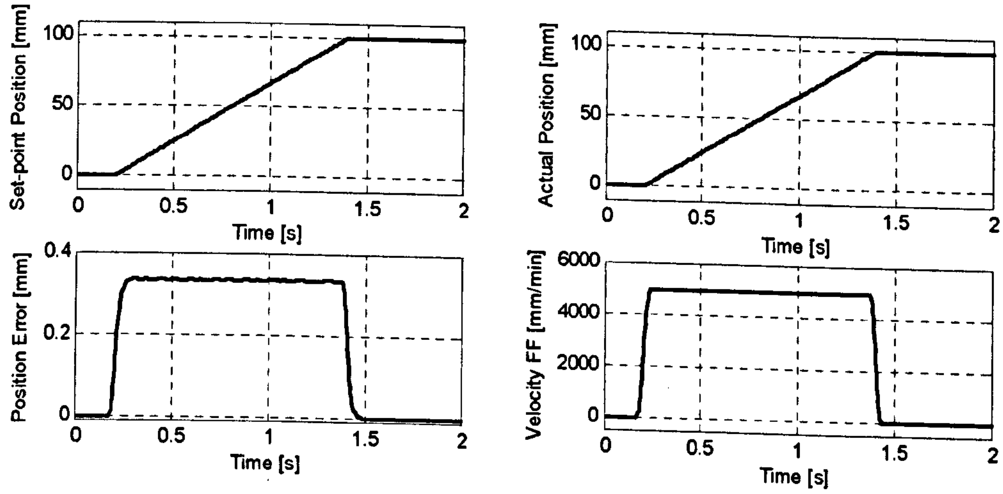


Figure G.48 Displacement = 100 mm, feedrate = 5000 mm/min (y-axis)

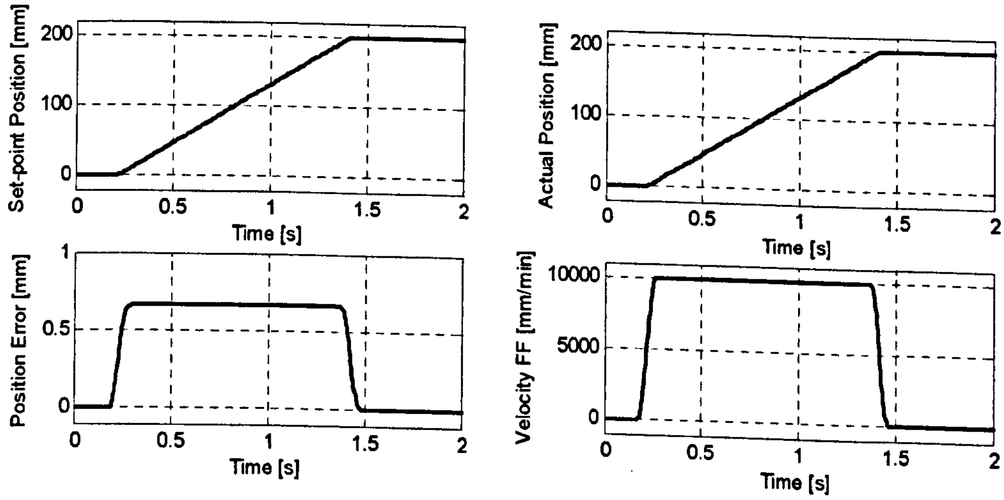


Figure G.49 Displacement = 200 mm, feedrate = 10000 mm/min (y-axis)

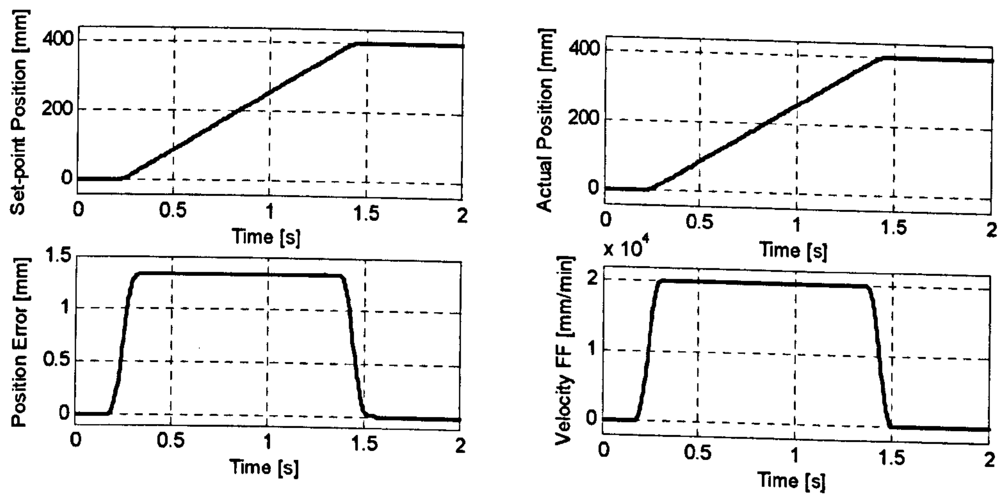


Figure G.50 Displacement = 400 mm, feedrate = 20000 mm/min (y-axis)

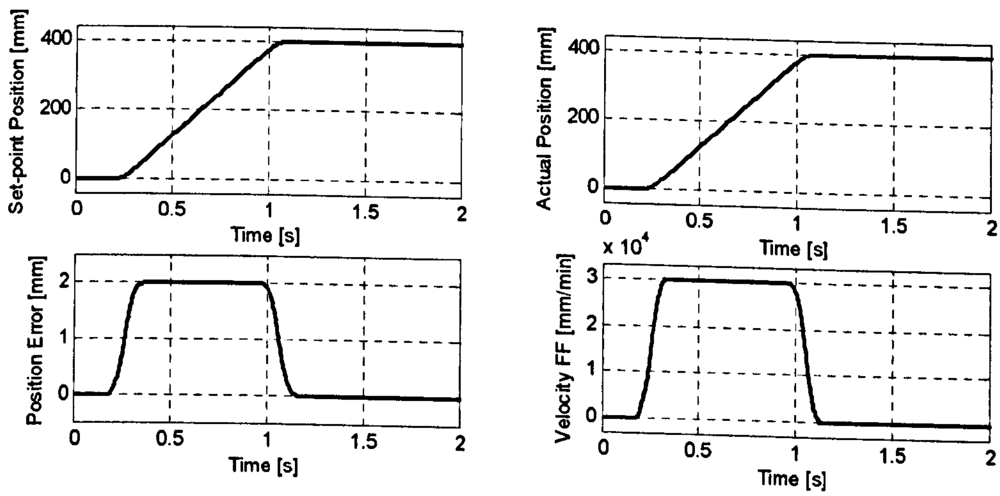


Figure G.51 Displacement = 400 mm, feedrate = 30000 mm/min (y-axis)

G.12 MATLAB Program Used to Plot Position Measurements in Ball-Bar Format

```
% This program plots the ballbar graph from the x and y-axis data in Cartesian coordinates
% It uses the MATLAB functions cart2pol and pol2cart
% Data measured from the linear encoders via dSPACE, sample time = 0.625 ms
% load run1 and run2 data:
load meas_bb
% Move the origin to the point (0,0)
x1=x1+150;
x2=x2+150;
% convert from cartesian to polar coordinates
[theta1,rho1] = cart2pol(x1,y1); % for run1
[theta2,rho2] = cart2pol(x2,y2); % for run2
% substract the radius and add the 20 microns radius to fit the ballbar graph
bb1=rho1-150+20e-3; % [microns] for run1
bb2=rho2-150+20e-3; % [microns] for run2
% convert from polar to catesian coordinates
[x1,y1]=pol2cart(theta1,bb1);
[x2,y2]=pol2cart(theta2,bb2);
plot(-x1,-y1,-x2,-y2)
% the minus is included to get the axis configuration on the machine the axes have to be reversed in the property
% editor of the figure
grid, legend('run1','run2'),  axis equal
```

APPENDIX H TEST RIG TLM MODEL IN MATLAB/ SIMULINK

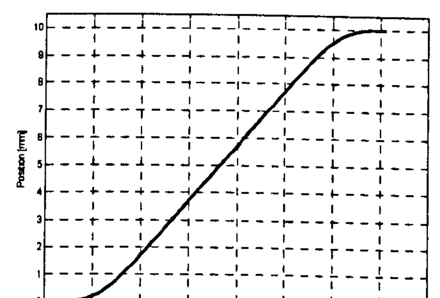
H.1 Testrig_profile.m Program

```
% this program  creates the variable profile as an input for the SIMULINK model of the test rig
st=3e-3; % sample time [s]
 load f002 % 1000 mm/min
len=length(data);
t=0:st:st*(len-1);
dref=data(:,1);    % Reference position  [mm]
vff=data(:,2)/60000;% velocity feed forward [m/s]
aff=data(:,3);     % acceleration feed forward [m/s^2]
profile=[t' dref vff aff];
tsim=max(t);   % simulation time for the SIMULINK model
```

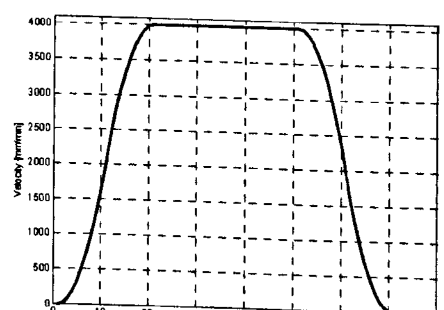
H.2 Reference Signal Profiles

The jerk-limited profile is generated by the interpolator to address the positioning movements commanded by an instruction of a NC program. It is generated according to the procedure presented in section 4.3.1. This profile is also used to represent the movement of the table on single-axis linear path and two-axis linear or circular path. The generation of this profile is implemented in the MATLAB function *jlProfile*. An example of the velocity and position profiles generated for a movement of 10 mm at 4000 mm/min is presented in Figure H.1 (*maximum_acceleration* = 6 m/s^2, *maximum_jerk* = 75 m/s^3, and *sampling_time* = 3 ms).

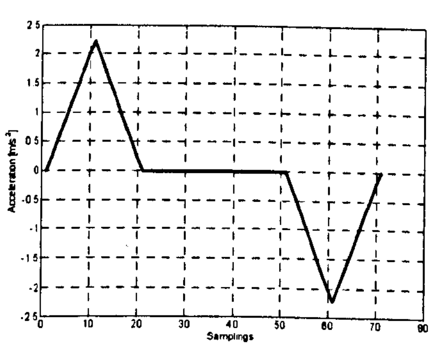
[position_profile,velocity_profile,acceleration_profile,jerk_profile]=jlProfile(4000, 10, 6,75, 3e-3);



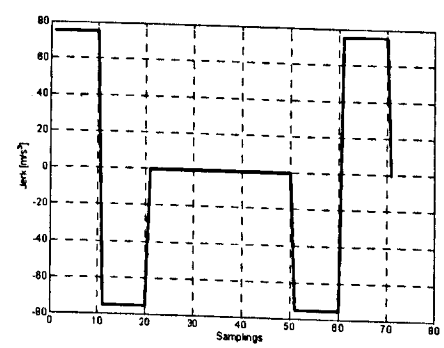
a) Position profile



b) Velocity profile



c) Acceleration profile



d) Jerk profile

Figure H.1 Jerk-limited position and velocity profiles

The step profile is used to simulate the step response of the model. It is conformed by the acceleration and maximum speed zones of the velocity profile generated for a one-axis movement of 10mm at 100 mm/min, as described in the TNC 426PB controller manual.
This profile is implemented in the MATLAB function *step_prof*. Figure H.2 shows the profile calculated for a sampling time of 3ms (*profile=step_profile(6,75,3e-3)*).

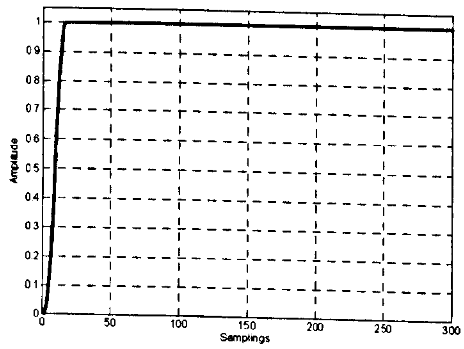


Figure H.2 Step Profile

The sinusoidal profile is used to simulate the model response to a sinusoidal signal. It is implemented in the MATLAB function *sin_profile*. The maximum frequency to be used will be the equivalent to ¼ the sampling time of the system to analyse. Figure H.3 shows the profile calculated for a sampling time of 3ms and a frequency of 10Hz (profile=sin_profile(3e-3,10)).

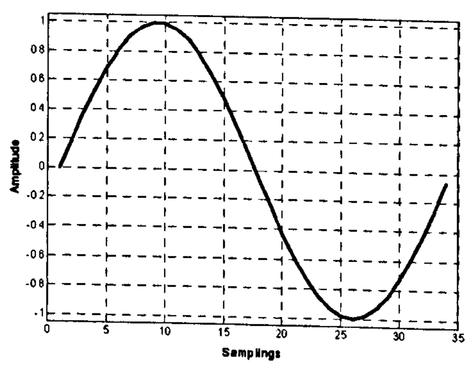
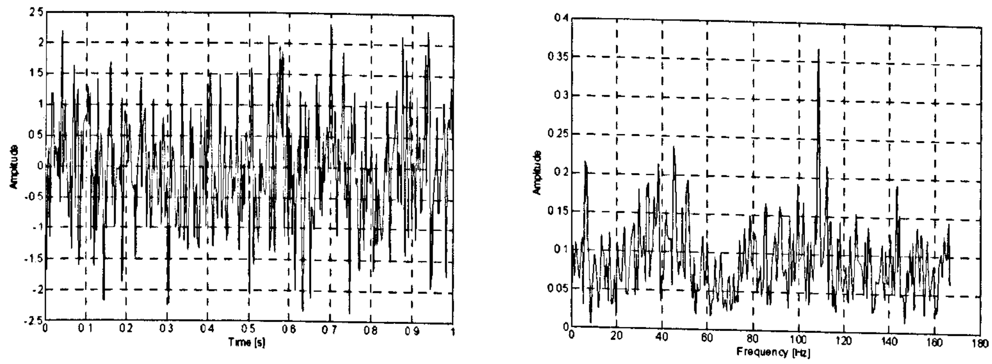


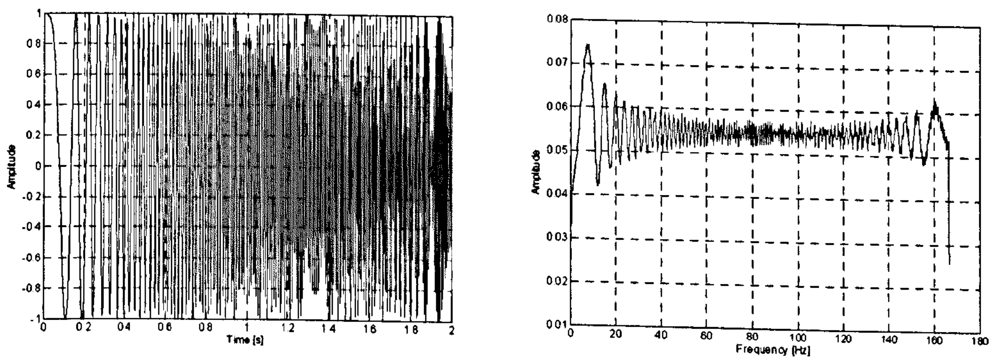
Figure H.3 Sinusoidal profile

The white noise profile will be used to analyse the frequency response of the model. It is implemented in the MATLAB function *white_noise_profile*. This function calls the MATLAB function *rand* to generate a sequence of pseudo-random numbers with duration of one second. Figure H.4 shows the profile calculated for a sampling time of 3ms (*profile=white_noise_profile(3e-3)*).



a) White noise signal b) Frequency spectrum

Figure H.4 White noise profile



a) Generated swept sine signal b) Frequency spectrum

Figure H.5 Swept sine profile

The swept sine profile will be used to analyse the frequency response of the model. It is implemented in the MATLAB function *chirp_profile*. This function calls the MATLAB function *chirp* to generate samples of a linear swept-frequency cosine signal between dc at time zero and ¼ the sampling rate at one second. Figure H.5 shows the profile calculated for a sampling time of 3ms (*profile=chirp_profile(3e-3)*).

H.2.1 Jerk Limited Profile Generation

```
function [position_profile,velocity_profile,acceleration_profile,jerk_profile]=jlProfile(feedrate,displacement,...
   maximum_acceleration,maximum_jerk,sampling_time)

% Generation of the position and velocity profiles according to the jerk-limited approach
%
% Author: Veimar Yobany Moreno Castañeda
% Date:  20 October 2004
% University of Huddersfield (U.H.)
% School of Computing and Engineering
%
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
% [position_profile,velocity_profile,acceleration_profile,jerk_profile]=jlProfile(feedrate,displacement,...
%   maximum_acceleration,maximum_jerk,sampling_time)
%
% INPUTS
%  feedrate              [m/s]
%  displacement          [m]
%  maximum_acceleration  [m/s^2]
```

```
% maximum_jerk       [m/s^3]
% sampling_time      [s]
%
% OUTPUTS
% position_profile      [m]
% velocity_profile      [m/s]
% acceleration_profile  [m/s^2]
% jerk_profile          [m/s^3]

% Pre-allocation of variables in memory
position_profile=zeros(4096,1);   % Maximum number of samples 4096
velocity_profile=zeros(4096,1);
acceleration_profile=zeros(4096,1);
jerk_profile=zeros(4096,1);
minimum_distance=0;     % [m] minimum distance that must be transversed in order to attain the programmed
feedrate
maximum_velocity=0;   % [m/s] Maximum possible value for the velocity profile
acceleration=0;       % [m/s^2] Maximum posible acceleration
T=zeros(7,1);         % [s] Array of duration times: T(i) is the duration time of the phase i  (i=1,2, ...,7)
time_end_phase=T;     % [s] Array of total times: time_end_phase(i) is the time at the end of phase i
velocity_end_phase=T;  % [m/s] Array of velocities: velocity_end_phase(i) is the value of the velocity profile at
                       % the end of phase i
phase4_exists=true;   % flag
k=1;                  % number of the sampling instant
% Verify the minimum distance dmin
minimum_distance=2*feedrate*sqrt(feedrate/maximum_jerk);
if displacement < minimum_distance
   % Reduce feed rate to ist maximum possible
   maximum_velocity=(maximum_jerk*displacement^2/4)^(1/3);
   % Phase 4 does not exist
   phase4_exists=false;
else
   maximum_velocity=feedrate;
end
% maximum_velocity and maximum_jerk results in an acceleration (See if phases 2 and 6 exist)
acceleration=sqrt(maximum_velocity*maximum_jerk);
if acceleration > maximum_acceleration
   acceleration=maximum_acceleration;
   T(2)=maximum_velocity/acceleration-acceleration/maximum_jerk;
else
   T(2)=0; % duration phase 2
end
T(6)=T(2);  % duration phase 6
T(1)=acceleration/maximum_jerk;  % duration phase 1
T(3)=T(1);  % duration phase 3
T(5)=T(1);  % duration phase 5
T(7)=T(1);  % duration phase 7
if phase4_exists == false
   T(4)=0; % duration phase 4
else
   T(4)=(displacement/maximum_velocity)-sum(T(1:3));
end
% Calculate the time at the end of each phase
time_end_phase(1)=T(1);
for i=2:7
   time_end_phase(i)=sum(T(1:i));
end
% Velocity at the end of each phase
velocity_end_phase(1)=0.5*maximum_jerk*T(1)^2;
velocity_end_phase(2)=velocity_end_phase(1)+acceleration*T(2);
velocity_end_phase(3)=maximum_velocity;
```

```
velocity_end_phase(4)=maximum_velocity;
velocity_end_phase(5)=velocity_end_phase(4)-0.5*maximum_jerk*T(5)^2;
velocity_end_phase(6)=velocity_end_phase(5)-acceleration*T(6);
% time_end_phase(7) is the total simulation time
% calculate the jerk, acceleration and velocity profiles at the given sampling_time
for ts=0:sampling_time:time_end_phase(7)
  if ts < time_end_phase(1)
    jerk_profile(k)=maximum_jerk;
    acceleration_profile(k)=maximum_jerk*ts;
    velocity_profile(k)=0.5*maximum_jerk*ts^2;
    k=k+1;
  elseif ts < time_end_phase(2)
    temp=(ts-time_end_phase(1));
    jerk_profile(k)=0;
    acceleration_profile(k)=acceleration;
    velocity_profile(k)=velocity_end_phase(1)+acceleration*(ts-time_end_phase(1));
    k=k+1;
  elseif ts < time_end_phase(3)
    temp=(ts-time_end_phase(2));
    jerk_profile(k)=-maximum_jerk;
    acceleration_profile(k)=acceleration-maximum_jerk*(ts-time_end_phase(2));
    velocity_profile(k)=velocity_end_phase(2)+acceleration*(ts-time_end_phase(2))-0.5*maximum_jerk*...
      (ts-time_end_phase(2))^2;
    k=k+1;
  elseif ts < time_end_phase(4)
    jerk_profile(k)=0;
    acceleration_profile(k)=0;
    velocity_profile(k)=maximum_velocity;
    k=k+1;
  elseif ts < time_end_phase(5)
    temp=ts-time_end_phase(4);
    jerk_profile(k)=-maximum_jerk;
    acceleration_profile(k)=-maximum_jerk*(ts-time_end_phase(4));
    velocity_profile(k)=maximum_velocity-0.5*maximum_jerk*(ts-time_end_phase(4))^2;
    k=k+1;
  elseif ts < time_end_phase(6)
    temp=ts-time_end_phase(5);
    jerk_profile(k)=0;
    acceleration_profile(k)=-acceleration;
    velocity_profile(k)=velocity_end_phase(5)-acceleration*(ts-time_end_phase(5));
    k=k+1;
  elseif ts < time_end_phase(7)
    temp=ts-time_end_phase(6);
    jerk_profile(k)=maximum_jerk;
    acceleration_profile(k)=-acceleration+maximum_jerk*(ts-time_end_phase(6));
    velocity_profile(k)=velocity_end_phase(6)-acceleration*(ts-time_end_phase(6))+0.5*maximum_jerk*...
      (ts-time_end_phase(6))^2;
    k=k+1;
  else
    jerk_profile(k)=0;
    acceleration_profile(k)=0;
    velocity_profile(k)=0;
    k=k+1;
  end
end
% Calculation of the position profile.  k is the number of elements calculated for the profiles
for i=2:k
  position_profile(i)=position_profile(i-1)+velocity_profile(i-1)*sampling_time;
  if position_profile(i) > displacement
    position_profile(i) = displacement;
  end
```

```
end
% Limit the length of the vetor to the number of calculated values
position_profile=position_profile(1:k);
velocity_profile=velocity_profile(1:k,1);
acceleration_profile=acceleration_profile(1:k,1);
jerk_profile=jerk_profile(1:k,1);
```

H.2.2 Sinusoidal Profile Generation

```
function profile=sin_profile(sampling_time,frequency)

% Generation of a sinusoidal signal
%
% Author: Veimar Yobany Moreno Castañeda
% Date:   20 October 2004
% University of Huddersfield (U.H.)
% School of Computing and Engineering
%
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
%        profile=sin_profile(sampling_time)
%
% INPUTS
%   sampling_time   [s] sampling time of the system to analyse
%   frequency       [Hz] frequency of the signal
%
% OUTPUTS
%   profile         signal with amplitud one
%
% The maximum frequency is 1/4 the sampling rate
max_f=(1/sampling_time)/4;
if frequency <= max_f
   t=0:sampling_time:1/frequency;
   profile=sin(2*pi*frequency*t);
else
   maximum_possible_frequency=max_f
end
```

H.2.3 White Noise Profile Generation

```
function profile=white_noise_profile(sampling_time)

% Generation of a white noise signal
%
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
%        profile=white_noise_profile(sampling_time)
%
% INPUTS
%   sampling_time   [s] sampling time of the system to analyse
%
% OUTPUTS
%   profile         signal with amplitud one

% This function uses the MATLAB signal randn.
%          y=randn([M,N])
% produces M-by-N matrices with pseudo-random numbers.  The sequence of numbers generated is determined
% by the state of the generator.
% Since MATLAB resets the state at start-up, the sequence of numbers generated will be the same unless the
% state is changed.
profile=randn(1,200); %ceil(1/sampling_time));
% This profile has a duration time of one second. The number of random numbers is ceil(1/sampling_time)
```

H.2.4 Swept Sine Profile

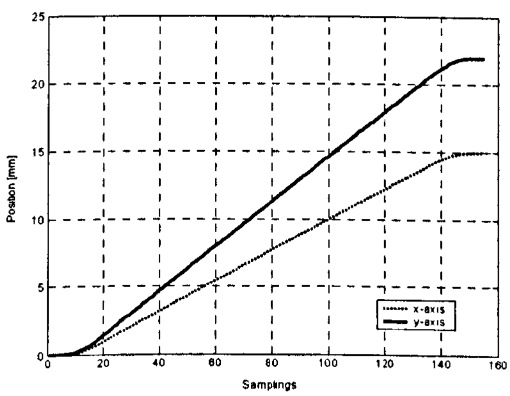
```
function profile=chirp_profile(sampling_time)

% Generation of a linear swept-frequency cosine signal
%
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
%       profile=chirp_profile(sampling_time)
%
% INPUTS
%   sampling_time       [s] sampling time of the system to analyse
% OUTPUTS
%   profile   signal with amplitud one
% This function uses the MATLAB signal chirp.
%         y = chirp(T,F0,T1,F1)
% generates samples of a linear swept-frequency cosine signal at the time instances defined in array T.
% The instantaneous frequency at time 0 is F0 Hertz.  The instantaneous frequency F1 is achieved at time T1
profile=chirp((0:sampling_time:2),0,3e-3/sampling_time,(1/sampling_time)/4);
% This profile starts at DC and cross f/4 Hz at t=3e-3/sampling_time sec (f=1/sampling_time)
```

H.2.5 Linear and Circular Interpolation

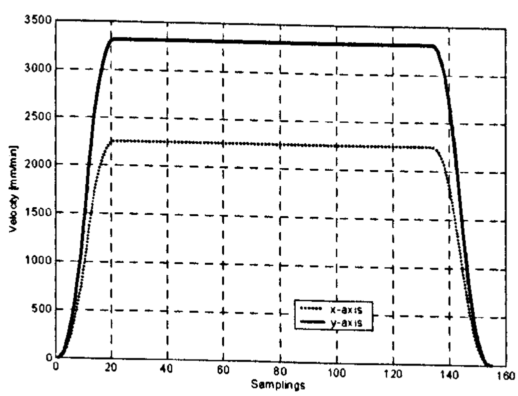
The linear and circular interpolation routines presented in section 4.3.1 are implemented in the MATLAB functions linear_interpolation.m and circular_interpolation.m respectively.

The profiles obtained for a linear movement from the coordinates (0,0) to the point with coordinates (15,22) are illustrated in Figure H.6

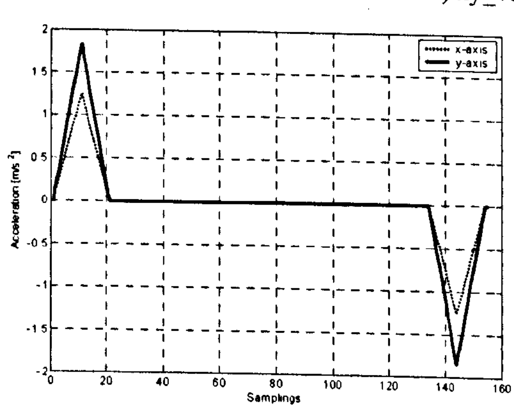
```
[xy_reference_position,xy_velocity_profile,xy_acceleration_profile]=linear_interp([0,0],[15,22],4000,6,75,3e-3)
```



a) xy_reference_position



b) xy_velocity_profile

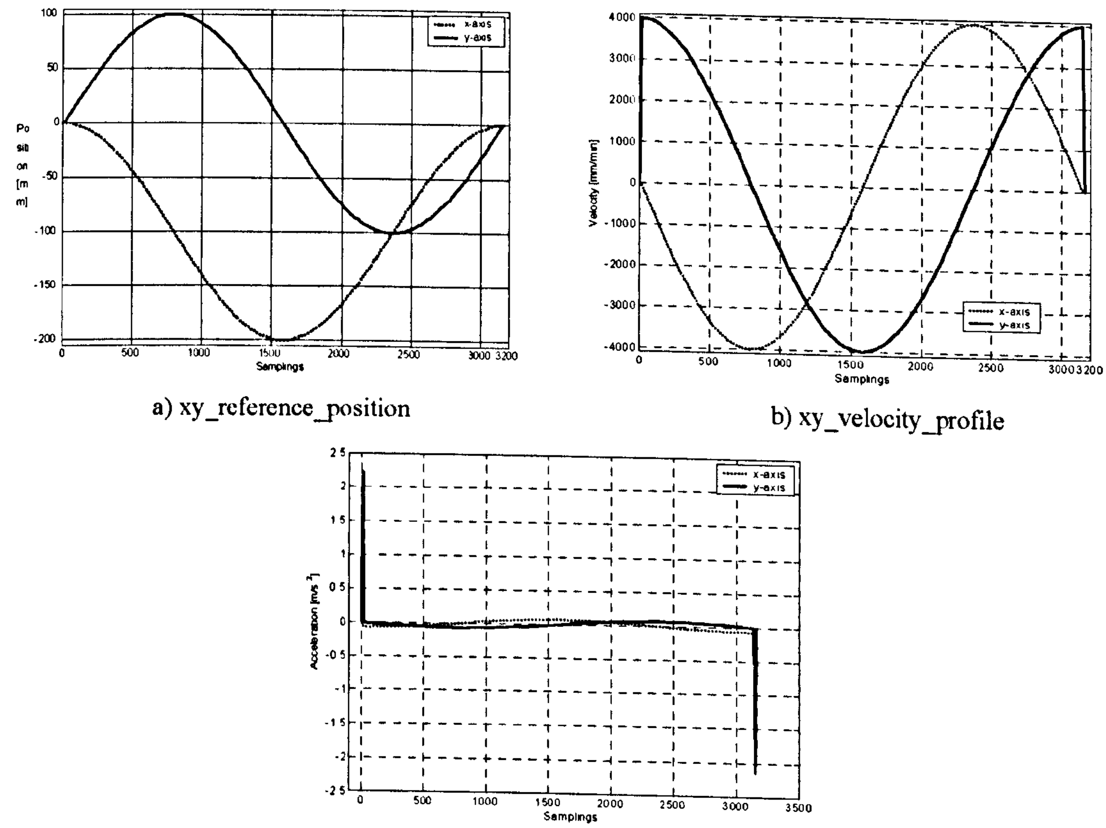


c) xy_acceleration_profile

Figure H.6 Linear interpolation profiles

The profiles obtained for a movement from the start point (0,0) with an arc of 2π radians and a radius of 100 mm are illustrated in Figure H.7 (a counter clockwise rotation towards the left side of the starting point is assumed)

```
[xy_reference_position,xy_velocity_profile,xy_acceleration_profile]=circular_interp([0,0],100,2*pi,1,1,...
    4000,6,75,3e-3);
```



a) xy_reference_position

b) xy_velocity_profile

c) xy_acceleration_profile

Figure H.7 Circular interpolation profiles

H.2.5.1 Linear Interpolation Routine

```
function [xy_reference_position,xy_velocity_profile]=linear_interpolation(start_point,end_point,feedrate,...
  maximum_acceleration,maximum_jerk,sampling_time)

% Performs linear interpolation up to two axis
%
% Author: Veimar Yobany Moreno Castañeda
% Date:  20 October 2004
% University of Huddersfield (U.H.)
% School of Computing and Engineering
%
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
% [xy_reference_position,xy_velocity_profile]=interpolator(start_point,end_point,feedrate,...
%  maximum_acceleration,maximum_jerk,sampling_time)
%
% INPUTS
%  start_point      [m] absolute coordinates of the actual position of the axes as an array [x_start;y_start]
%  end_point        [m] absolute coordinates of the end point as an array [x_end;y_end]
%  feedrate         [m/s]
```

```
% maximum_acceleration [m/s^2]
% maximum_jerk        [m/s^3]
% sampling_time       [s]
%
% OUTPUTS
% xy_reference_position  [m] matrix of the reference positions:
%                       xy_reference_position(:,1) holds the x_axis reference position
%                       xy_reference_position(:,2) holds the y_axis reference position
% xy_velocity_profile   [m/s] matrix of the velocity profiles:
%                       xy_velocity_position(:,1) holds the x_axis velocity profile
%                       xy_velocity_position(:,2) holds the y_axis velocity profile

% Pre-allocation of variables in memory
xy_displacement=[0;0]; % Displacement of each axis
               % xy_displacement(1) for x-axis
               % xy_displacement(2) for y-axis
displacement=0;   % [m] total displacement
xy_displacement=end_point-start_point; % displacement of each axis:
displacement=sqrt(xy_displacement(1)^2+xy_displacement(2)^2); % [m] total displacemnt
% Calculate the path profiles:
[position_profile,velocity_profile,acceleration_profile,jerk_profile]=jlProfile(feedrate,displacement,...
  maximum_acceleration,maximum_jerk,sampling_time);
% Pre-allocate the refrence_position and velocity_profile arrays in memory
xy_reference_position=zeros(length(position_profile),2);
xy_velocity_profile=zeros(length(position_profile),2);
% Calculate the reference position for each axis
xy_reference_position(:,1)=(xy_displacement(1)/displacement)*position_profile; % [m] x-axis
xy_reference_position(:,2)=(xy_displacement(2)/displacement)*position_profile; % [m] y-axis
% convert xy_reference_position to absolute coordinates:
xy_reference_position(:,1)=xy_reference_position(:,1)+start_point(1); % x-axis
xy_reference_position(:,2)=xy_reference_position(:,2)+start_point(2); % y-axis
% Calculate the velocity profile for each axis
xy_velocity_profile(:,1)=(xy_displacement(1)/displacement)*velocity_profile; % [m/s] x-axis
xy_velocity_profile(:,2)=(xy_displacement(2)/displacement)*velocity_profile; % [m/s] y-axis
```

H.2.5.2 Circular Interpolation Routine

```
function [xy_reference_position,xy_velocity_profile]=circular_interpolation(start_point,radius,angle,...
  dir_rotation,dir_movement,feedrate,maximum_acceleration,maximum_jerk,sampling_time)

% Performs circular interpolation for two axis
%
% Author: Veimar Yobany Moreno Castañeda
% Date:  20 October 2004
% University of Huddersfield (U.H.)
% School of Computing and Engineering
%
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
%
% [xy_reference_position,xy_velocity_profile]=circular_interpolation(start_point,radius,angle,dir_rotation,...
%  dir_movement,feedrate,maximum_acceleration,maximum_jerk,sampling_time)
%
% INPUTS
% start_point      [m] absolute coordinates of the actual position of the axes as an array [x_start;y_start]
% radius          [m] radius of the arc
% angle           [rad] angle of the arc
% dir_rotation      clockwise=-1; counterclockwise =1
% dir_movement       to the left of the start_point =1; to the rigth of the start_point= -1
% feedrate         [m/s]
% maximum_acceleration [m/s^2]
% maximum_jerk       [m/s^3]
```

```
% sampling_time       [s]
%
% OUTPUTS
% xy_reference_position  [m] matrix of the reference positions:
% xy_reference_position(:,1) holds the x_axis reference position
% xy_reference_position(:,2) holds the y_axis reference position
% xy_velocity_profile    [m/s] matrix of the velocity profiles:
% xy_velocity_position(:,1) holds the x_axis velocity profile
% xy_velocity_position(:,2) holds the y_axis velocity profile

% Pre-allocation of variables in memory
displacement=0;    % [m] total displacement
%angle_profile        [rad] position profile in radians
displacement=radius*angle;
% Calculate the path profiles:
[position_profile,velocity_profile,acceleration_profile,jerk_profile]=jlProfile(feedrate,displacement,...
   maximum_acceleration,maximum_jerk,sampling_time);
% Pre-allocate the reference_position and velocity_profile arrays in memory
xy_reference_position=zeros(length(position_profile),2);
xy_velocity_profile=zeros(length(position_profile),2);
% Convert the position_profile to an array of angles
angle_profile=position_profile/radius;
% Calculate the reference position for each axis
xy_reference_position(:,1)=dir_movement*radius*(cos(angle_profile)-1); % [m] x-axis
xy_reference_position(:,2)=dir_rotation*radius*sin(angle_profile); % [m] y-axis
% convert xy_reference_position to absolute coordinates:
xy_reference_position(:,1)=xy_reference_position(:,1)+start_point(1); % x-axis
xy_reference_position(:,2)=xy_reference_position(:,2)+start_point(2); % y-axis
% Calculate the velocity profile for each axis
xy_velocity_profile(:,1)=dir_movement*velocity_profile.*-sin(angle_profile); % [m/s] x-axis
xy_velocity_profile(:,2)=dir_rotation*velocity_profile.*cos(angle_profile); % [m/s] y-axis
```

H.3 Y-Axis Block Parameters and Initialisation Code

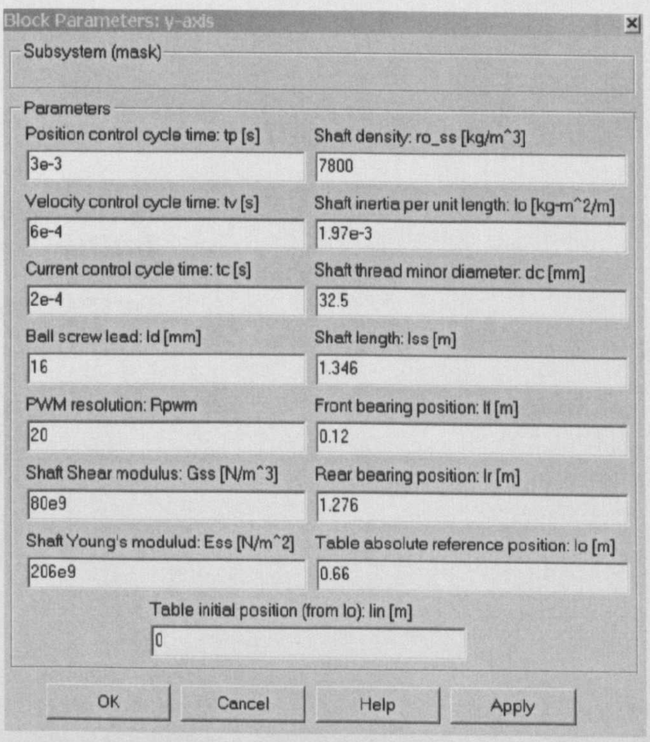


Figure H.8 Y-axis block parameters

The following code was included in the initialisation feature of the block's mask:

```
% Parameters for the synchronisation of torsional and axial models
ct=5;    % number of axial sections per torsional section
ca=8;
% Torsional propagation parameters:
tt=tc/Rpwm;              % torsional propagation time [s]
ut=sqrt(Gss/ro_ss); % torsional propagation velocity [m/s]
Zt=Io*ut;                % Equivalent impedance torsional model
ltor=ut*tt;         % section length torsional model [m]
h=round(lr/ltor); % Number of sections for the torsional model
lend=lss-h*ltor; % Shaft length after rear bearing [m]
Jend=lend*Io;            % Inertia associated to lend [kg-m^2]
fb=round(lf/ltor);       % Number of sections in zone1
switch fb
    case 0
        fb=1; % zone1 must have at least one section
end
ht=h-fb;                 % Number of sections in zone2
lref=lo+lin-ltor*fb;     % Reference for the nut position montoring [m]

% Axial propagation parameters:
ta=tt/ca;                % Axial propagation time [s]
ua=sqrt(Ess/ro_ss);  % axial propagation velocity [m/s]
Ass=pi*((dc/1000)/2)^2; % Screw shaft cross sectional area [m^2]
Za=ro_ss*Ass*ua;           % Equivalent impedance axial model
laxial=ltor/ct;            % Section length (axial model) [m]
ha=ht*ct;                  % number of sections axial model
```

H.4 Velocity Controller Block Initialisation Code

The following code was included in the initialisation feature of the block's mask:

```
kp=kp/(2*pi);            % [A-s/rad]
ki=ki/(2*pi);            % [A/rad]
kd=kd/(2*pi);            % [A-s^2/rad]
kaff=kaff/(2*pi);        % [A-s^2/rad]
% lowpass filter
switch lpf_flag
    case 1 % Disabled
        Blf=1;  Alf=1;
    case 2 % Between 600 and 700 Hz
        % [Blf,Alf]=butter(1,600*2*tv);
    case 3 % > 700Hz
        % [Blf,Alf]=butter(2,700*2*tv);
end
% PT2 filter
if pt2_delay == 0
    Bpt2=1;     Apt2=1;
else
    Bpt2=1;     Apt2=1;
end
% Bandstop filter
if bsf_freq == 0
    Bn=1;  An=1;
else
    %[Bn,An]= ellip(2,0.25,bsf_dam,bsf_freq*2*tv);
Bn=[0.89302559243828 -1.76897528904828 2.66063770787048 -1.76897528904828 0.89302559243828];
An=[1.00000000000000 -1.89654351266864 2.72974439415250 -1.74471708749617 0.84679015853235];
end
```

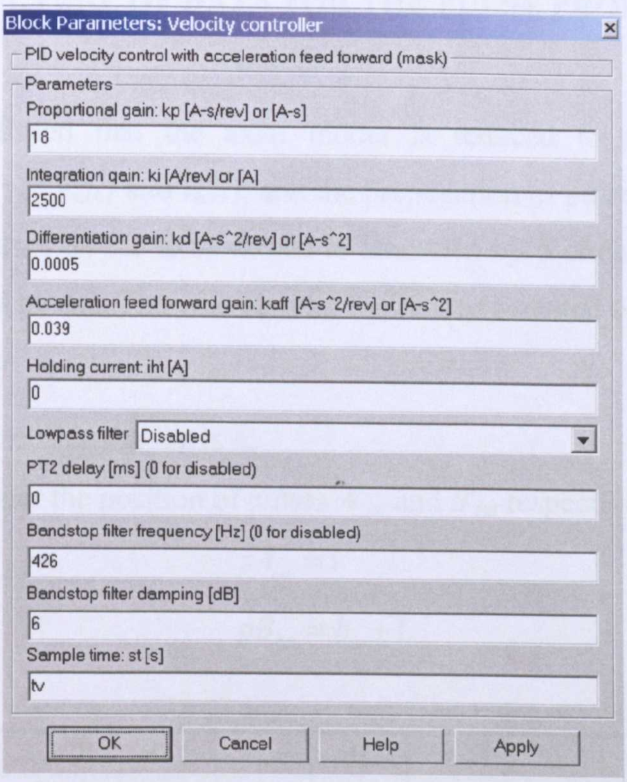


Figure H.9 Velocity controller block parameters

APPENDIX I STRUCTURE OF DATA FOR THE PULSE PROPAGATION ON THE AXIAL MODEL

Section 5.1.3.3 established that the axial model is reduced to the calculation of the longitudinal velocities v_{1a}, v_{na+1} and v_{ha+1}; and the propagation of pulses on the other sections. The propagation of pulses on the axial model is modelled by a circular linked list as in the zone 2 for the torsional model. Thus the circular list is implemented on a 3xn_{ma} matrix called *listA* (Figure I.1) where:

- The number of sections on the list is h_a.
- pA_{1a} and pB_{ha} register the position of pulses A^i_{1a} and B^i_{ha} respectively:

$$pA_{1a} = 1 \tag{I.1}$$

$$pB_{ha} = h_a + 1 \tag{I.2}$$

column	*1*	*2*	*3*		*ha-2*	*ha-1*	*ha*	*ha+1*	*ha+2*	*ha+3*		*2ha-1*	*2ha*
pulse	A_{1a}	A_{2a}	A_{3a}	...	A_{ha-2}	A_{ha-1}	A_{ha}	B_{ha}	B_{ha-1}	B_{ha-2}	...	B_{2a}	B_{1a}
next	2	3	4		*ha-1*	*ha*	*ha+1*	*ha+2*	*ha+3*	*ha+4*		*2ha*	1
previou	*2ha*	1	2		*ha-3*	*ha-2*	*ha-1*	*ha*	*ha+1*	*ha+2*		*2ha-2*	*2ha-1*
	↑ pA_{1a}							↑ pB_{ha}					

Figure I.1 Array used to simulate the second zone of the axial model

If the velocities v_{1a} and v_{ha+1} are known, the pulses propagation is simulated by the following equations:

$$listA(1, pA_{ha1}) = v_{1a}(k)Z_a + listA(1, pA_{1a}) \tag{I.3}$$

$$listA(1, pB_{ha}) = listA(1, pB_{ha}) - v_{ha+1}(k)Z_a \tag{I.4}$$

$$pA_{1a} = listMa(2, pA_{1a}) \tag{I.5}$$

$$pB_{ha} = listMa(2, pB_{ha}) \tag{I.6}$$

The inclusion of the nut in the model will cause the reflection of pulses arriving to section n_a, and therefore splitting the *listA* in two as shown in Figure I.2a. The following variables are added in order to complete the model for the moving nut:

- n_{la}: The number of sections in the left loop on the Figure I.2a. pB_{na} , pA_{na1} and pB_{na} register the position of pulses B^i_{na+1} A^i_{na+1} and B^i_{na} respectively:

$$pB_{na+1} = 2h_a - n_{la} + 1 \tag{I.7}$$

$$pA_{na1} = n_{la} + 1 \tag{I.8}$$

$$pB_{na} = pB_{na+1} + 1 \tag{I.9}$$

- Pulse A^i_{na} is connected with pulse B^i_{na}

$$pA_{na} = n_{la} \tag{I.10}$$

$$listA(2, pA_{na}) = pB_{na} \tag{I.11}$$

$$listA(3, pB_{na}) = pA_{na} \tag{I.12}$$

- Pulse B^i_{na+1} is connected with pulse A^i_{na+1}:

$$listA(2, pB_{na1}) = pA_{na1} \tag{I.13}$$

$$listA(3, pA_{na1}) = pB_{na1} \tag{I.14}$$

If the angular velocity v_{na+1} is known, the pulse propagations on section n_a is simulated by the following equations:

$$listA(1, pA_{na1}) = v_{na+1}(k)Z_a + listA(1, pA_{na1}) \tag{I.15}$$

$$listA(1, pB_{na}) = listA(1, pB_{na}) - v_{na+1}(k)Z_a \tag{I.16}$$

$$pA_{na1} = listA(2, pA_{na1}) \tag{I.17}$$

$$pB_{na} = listA(2, pB_{na}) \tag{I.18}$$

Figure I.3 shows the status of the matrix *listA* after two pulse propagations. The number of sections on the two loops in zone two changes when the nut moves to an adjacent section. As for the torsional model, the connections of the pulses $A^i_{na,}$, A^i_{na+1}, A^i_{na+2}, $B^i_{na,}$, B^i_{na+1} and B^i_{na+2} change when the nut moves to the next section on the right (from section n_a to section n_a+1). The mapping of those changes on the matrix *listA* are carried out by the following procedure:

- The position of the pulses A^i_{na}, A^i_{na+2}, B^i_{na+1} and B^i_{na+2} is held in the variables pA_{na}, pA_{na2}, pB_{na1} and pB_{na2} respectively:

$$pA_{na} = listA(3, pB_{na}) \tag{I.19}$$

$$pA_{na2} = listA(2, pA_{na1}) \tag{I.20}$$

$$pB_{na1} = listA(3, pA_{na1}) \tag{I.21}$$

$$pB_{na2} = listA(3, pB_{na1}) \tag{I.22}$$

- Pulse A^i_{na} is connected with pulse A^i_{na+1}:

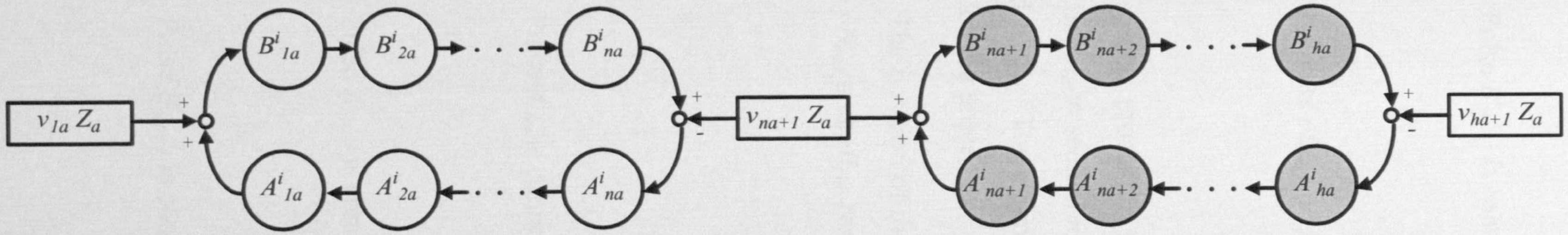
$$listA(2, pA_{na}) = pA_{na1} \tag{I.23}$$

$$listA(3, pA_{na1}) = pA_{na} \tag{I.24}$$

- Pulse A^i_{na+1} is connected with pulse B^i_{na+1}:

$$listA(2, pA_{na1}) = pB_{na1} \tag{I.25}$$

$$listA(3, pB_{na1}) = pA_{na1} \tag{I.26}$$



a) Circular list divided in two loops

column	*1*	*2*	*3*		*nla*	*nla+1*	*nla+2*	*nla+3*	*nla+4*		*ha*	*ha+1*	*ha+2*	*ha+3*		*nna*	*nna+1*	*nna+2*	*nna+3*		*2ha*
pulse	A_{1a}	A_{2a}	A_{3a}	...	A_{na}	A_{na+1}	A_{na+2}	A_{na+3}	A_{na+4}		A_{ha}	B_{ha}	B_{ha-1}	B_{ha-2}		B_{na+1}	B_{na}	B_{na-1}	B_{na-2}	..	B_{1a}
next	*2*	*3*	*4*		***nna+1***	*nla+2*	*nla+3*	*nla+4*	*nla+5*		*ha+1*	*ha+2*	*ha+3*	*ha+4*		***nla+1***	*nna+2*	*nna+3*	*nna+4*		*1*
previous	*2ha*	*1*	*2*		*nla-1*	***nna***	*nla+1*	*nla+2*	*nla+3*		*ha-1*	*ha*	*ha+1*	*ha+2*		*nna-1*	***nla***	*nna+1*	*nna+2*		*2ha-1*
	↑ pA_{1a}					↑ pA_{nal}						↑ pB_{rba}					↑ pB_{na}				

b) Mapping of the two loops on the matrix *listA*

Figure I.2 Second zone array including the moving nut (axial model)

column	*1*	*2*	*3*		*nla*	*nla+1*	*nla+2*	*nla+3*	*nla+4*		*ha*	*ha+1*	*ha+2*	*ha+3*		*nna*	*nna+1*	*nna+2*	*nna+3*		*2ha*
pulse	$\mathbf{B_{2a}}$	$\mathbf{B_{1a}}$	$\mathbf{A_{1a}}$	...	$\mathbf{A_{na-2}}$	$\mathbf{B_{na+2}}$	$\mathbf{B_{na+1}}$	$\mathbf{A_{na+1}}$	$\mathbf{A_{na+2}}$	.	$\mathbf{A_{ha-2}}$	$\mathbf{A_{ha-1}}$	$\mathbf{A_{ha}}$	$\mathbf{B_{ha}}$		$\mathbf{B_{na+3}}$	$\mathbf{A_{na-1}}$	$\mathbf{A_{na}}$	$\mathbf{B_{na}}$	..	$\mathbf{B_{3a}}$
next	*2*	*3*	*4*		*nna+1*	*nla+2*	*nla+3*	*nla+4*	*nla+5*		*ha+1*	*ha+2*	*ha+3*	*ha+4*		*nla+1*	*nna+2*	*nna+3*	*nna+4*		*1*
previous	*2ha*	*1*	*2*		*nla-1*	*nna*	*nla+1*	*nla+2*	*nla+3*		*ha-1*	*ha*	*ha+1*	*ha+2*		*nna-1*	*nla*	*nna+1*	*nna+2*		*2ha-1*
			↑ pA_{1a}					↑ pA_{nal}						↑ pB_{ha}					↑ pB_{na}		

Figure I.3 Mapping of the two loops on the matrix *listA* after two pulse propagations

- Pulse B^i_{na+1} is connected with pulse B^i_{na}:

$$listA(2, pB_{na1}) = pB_{na} \tag{I.27}$$

$$listA(3, pB_{na}) = pB_{na1} \tag{I.28}$$

- Pulse B^i_{na+2} is connected with pulse A^i_{na+2}:

$$listA(2, pB_{na2}) = pA_{na2} \tag{I.29}$$

$$listA(3, pA_{na2}) = pB_{na2} \tag{I.30}$$

- Pointers pB_{na} and pA_{na1} are set to their new values:

$$pB_{na} = pB_{na1} \tag{I.31}$$

$$pA_{na1} = pA_{na2} \tag{I.32}$$

A similar procedure is applied when the nut moves to the next section on the left (from section n_a to section n_a-1). Pulses affected by this movement are: A^i_{na-1}, A^i_{na}, A^i_{na+1}, B^i_{na-1}, B^i_{na} and B^i_{na+1}. The mapping of the changes on the matrix *listA* is carried out in this case by the following procedure:

- The position of the pulses A^i_{na}, A^i_{na-1}, B^i_{na+1} and B^i_{na-1} is held in the variables pA_{na}, pA_{1na}, pB_{na1} and pB_{1na} respectively:

$$pA_{na} = listA(3, pB_{na}) \tag{I.33}$$

$$pA_{1na} = listA(3, pA_{na}) \tag{I.34}$$

$$pB_{na1} = listA(3, pA_{na1}) \tag{I.35}$$

$$pB_{1na} = listA(2, pB_{na}) \tag{I.36}$$

- Pulse A^i_{na} is connected with pulse A^i_{na+1}:

$$listA(2, pA_{na}) = pA_{na1} \tag{I.37}$$

$$listA(3, pA_{na1}) = pA_{na} \tag{I.38}$$

- Pulse A^i_{na-1} is connected with pulse B^i_{na-1}:

$$listA(2, pA_{1na}) = pB_{1na} \tag{I.39}$$

$$listA(3, pB_{1na}) = pA_{1na} \tag{I.40}$$

- Pulse B^i_{na+1} is connected with pulse B^i_{na}:

$$listA(2, pB_{na1}) = pB_{na} \tag{I.41}$$

$$listA(3, pB_{na}) = pB_{na1} \tag{I.42}$$

- Pulse B^i_{na} is connected with pulse A^i_{na}:

$$listA(2, pB_{na}) = pA_{na} \tag{I.43}$$

$$listA(3, pA_{na}) = pB_{na} \tag{I.44}$$

- Pointers pB_{na} and pA_{na1} are set to their new values:

$$pB_{na} = pB_{1na} \tag{I.45}$$

$$pA_{na1} = pA_{na} \tag{I.46}$$

Variables *difSecA* and *lastSecA* are included to verify if the nut has moved to an adjacent section and therefore decide which part of the code will be executed (the nut is on the same section, the nut has moved to the left or the nut has moved to the right). The choice is taken according to the following procedure:

- Calculate the section where the nut is on

$$n_a = ceil(l_n / l_{axial}) \tag{I.47}$$

- Calculate the difference between the new and the last section

$$difSecA = n_a - lastSecA \tag{I.48}$$

- Switch between the two cases based on the value for *difSecA*

```
switch difSecA
    case 1
        run code when the nut has moved to the right
    case -1
        run code when the nut has moved to the left
end
```

- Assign the value of n_a to *lastSecA*

$$lastSecA = n_a \tag{I.49}$$

APPENDIX J PARAMETERS AND INITIALISATION CODE FOR THE TORSIONAL LOOP SUBSYSTEM

J.1 Initialisation Code for the Torsional Loop Block

```
% Zone1 parameters:
lenF=2*fb;
A1=1;                  % position of A1 on the list
Bfb=fb+1;              % position of Bfb on the list
B1=lenF;               % postion of B1 on the list
listF=zeros(2,lenF);
for i=1:lenF-1
        listF(2,i)=i+1;
end
listF(2,B1)=A1;  % next to B1

% Zone2 parameters
nt=ceil(lref/ltor); % Nut position [sections]
lenM=2*ht;             % number of elements on the list
Afb1=1;                % position of Afb+1 on the list
An=nt;                 % position of An on the list
An1=nt+1;              % position of An+1 on the list
Bht=ht+1;              % position of Bht on the list
Bn1=lenM-nt;   % position of Bn+1 on the list
Bn=Bn1+1;              % position of Bn on the list
Bfb1=lenM;             % position of Bfb+1 on the list
listM=zeros(3,lenM);

% Row 2 assignment:
for i=1:lenM-1
        listM(2,i)=i+1;
end

listM(2,Bfb1)=Afb1;    % Next to Bfb+1
listM(2,An)=Bn;        % Next to An
listM(2,Bn1)=An1;      % Next to Bn+1

% Row 3 assignment
for i=2:lenM
        listM(3,i)=i-1;
end

listM(3,Afb1)=Bfb1;    % Before Afb+1
listM(3,Bn)=An;        % Before Bn
listM(3,An1)=Bn1;      % Before An+1
Mltor=1/(ltor*1000);% [1/mm] for the nut movement monitoring
```

J.2 The PWM Generating Function

```
function [e_abc,vec,t_dc]=pwm_inverter(th_e,e_dqr)

% Generation of the switching vectors and their duration time
%
% Author: Veimar Yobany Moreno Castañeda
% Date:  20 October 2004
% University of Huddersfield (U.K.)
% School of Computing and Engineering
%
```

```
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
%
% [e_abc,vec,t_dc]=pwm_inverter(th_e,e_dqr)
%
% INPUTS
%  th_e   [rad]  Electrical position of the motor
%  e_dqr  [V]    d,q reference voltages
%
% OUTPUTS
%  e_abc  [V]    Equivalent three phase voltages to be applied to the motor
%  vec    [V]    Vector of switching voltages. vec(:,i) represents the switching voltages for the switching i
%  t_dc [samples]  Vector of switching voltages duration. t_dc(i) represents the duration of the switching state i

global h % structure that contains the variables of interest

% h.pwm.Mxyz         % Matrix used for the calculation of the t_xyz times
% h.pwm.Tpwm2        % Tpwm/2
% h.pwm.Rpwm         % PWM resolution
% h.pwm.tpwm         % sampling time for the Motor model
% h.pwm.stator_volt  % Array used for the calculation of the switching voltages

% Pre-allocation of variables in memory
% e_alpha_beta=[0;0];
e_abc=[0;0;0];

% t_xyz=[0 0 0];  % xyz times
p_abc=[0 0 0];    % Used to define the sector

% sector=0;       % sector in which the refernce stator voltage is om
% t_23=[0 0];     % duration of switching states 2 and 3
vec=zeros(2,7);
t_dc=zeros(1,7);

% Translate to alpha beta frame system:
e_alpha_beta=[sin(th_e) cos(th_e);
        -cos(th_e) sin(th_e)]*e_dqr;

% Translate to abc frame system:
e_abc=[    0    1.0000;
    0.8660  -0.5000;
   -0.8660  -0.5000]*e_alpha_beta;

% Fin dout the xyz times
t_xyz=h.pwm.Mxyz*e_alpha_beta;

% determine in which sector the e_alpha_beta is found
if e_abc(1)>0
   p_abc(1)=1;
else
   p_abc(1)=0;
end
if e_abc(2)>0
   p_abc(2)=2;
else
   p_abc(2)=0;
end
if e_abc(3)>0
   p_abc(3)=4;
else
   p_abc(3)=0;
end
```

```
sector=sum(p_abc);

% Choose the two sector boundary vectors and calcualte the duration of application for each one
% Values for the possible stator voltages are calculated assuming a DC link voltage of 600 V
switch sector
  case 1
    t_23=[ t_xyz(2)  t_xyz(3)];
    vec(:,2:3)=[h.pwm.stator_volt(:,6) h.pwm.stator_volt(:,2)];
    vec(:,5:6)=[h.pwm.stator_volt(:,2) h.pwm.stator_volt(:,6)];
  case 2
    t_23=[ -t_xyz(1) t_xyz(2)];
    vec(:,2:3)=[h.pwm.stator_volt(:,5) h.pwm.stator_volt(:,4)];
    vec(:,5:6)=[h.pwm.stator_volt(:,4) h.pwm.stator_volt(:,5)];
  case 3
    t_23=[-t_xyz(3)  t_xyz(1)];
    vec(:,2:3)=[h.pwm.stator_volt(:,4) h.pwm.stator_volt(:,6)];
    vec(:,5:6)=[h.pwm.stator_volt(:,6) h.pwm.stator_volt(:,4)];
  case 4
    t_23=[t_xyz(3)  -t_xyz(1)];
    vec(:,2:3)=[h.pwm.stator_volt(:,3) h.pwm.stator_volt(:,1)];
    vec(:,5:6)=[h.pwm.stator_volt(:,1) h.pwm.stator_volt(:,3)];
  case 5
    t_23=[ t_xyz(1) -t_xyz(2)];
    vec(:,2:3)=[h.pwm.stator_volt(:,2) h.pwm.stator_volt(:,3)];
    vec(:,5:6)=[h.pwm.stator_volt(:,3) h.pwm.stator_volt(:,2)];
  case 6
    t_23=[-t_xyz(2) -t_xyz(3)];
    vec(:,2:3)=[h.pwm.stator_volt(:,1) h.pwm.stator_volt(:,5)];
    vec(:,5:6)=[h.pwm.stator_volt(:,5) h.pwm.stator_volt(:,1)];
end

% saturate the duration of the two sector boundary vectors
tem=sum(t_23);
if tem > h.pwm.Tpwm2
  t_23=t_23*h.pwm.Tpwm2/tem;
end

% normalize t_23 to samples
t_23=round(t_23/h.pwm.tpwm);

% Compose the vector of samples (times) for the application of each element of vec  -> sum(vec)=h.pwm.Rpwm
t_dc(1)=round((h.pwm.Rpwm-2*sum(t_23))/4);
t_dc(2)=t_dc(1)+t_23(1);
t_dc(3)=t_dc(2)+t_23(2);
t_dc(4)=h.pwm.Rpwm-t_dc(3);
t_dc(5)=t_dc(4)+t_23(2);
t_dc(6)=t_dc(5)+t_23(1);
t_dc(7)=h.pwm.Rpwm;
```

J.3 Initialisation Code for the wm Calculation Block

```
Zmc=(Jm+Jc)/st;
Zc=Jc/st;
Zcs=kcs*st;
Zct=Zc+Zt;
Zeq=Zcs*Zct/(Zcs+Zct);
ZEcs=Zct/(Zcs+Zct);
ZEct=Zcs/(Zcs+Zct);
Mwm=1/(bm+Zmc+Zeq);
Mw1=1/Zct;
```

J.4 wht+1 Calculation and Nut Monitoring Blocks

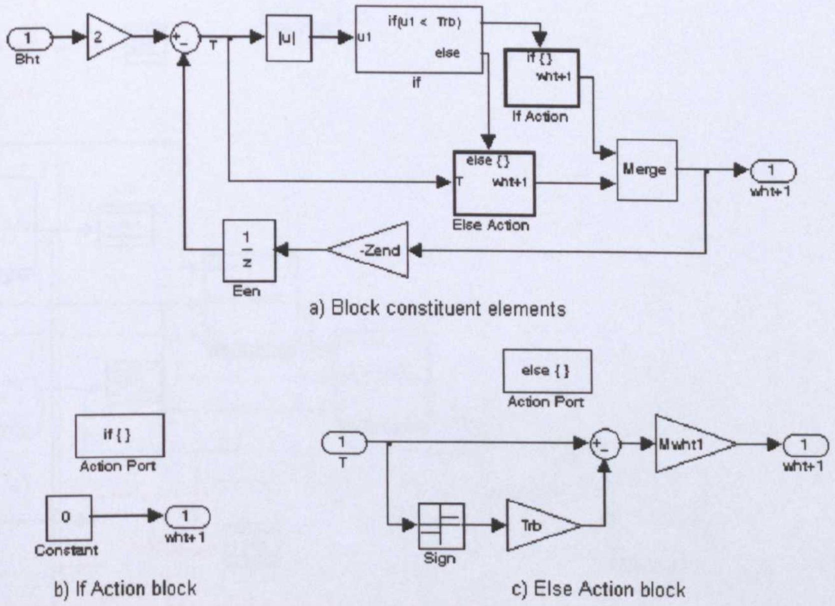


Figure J.1 w_{ht+1} Calculation block

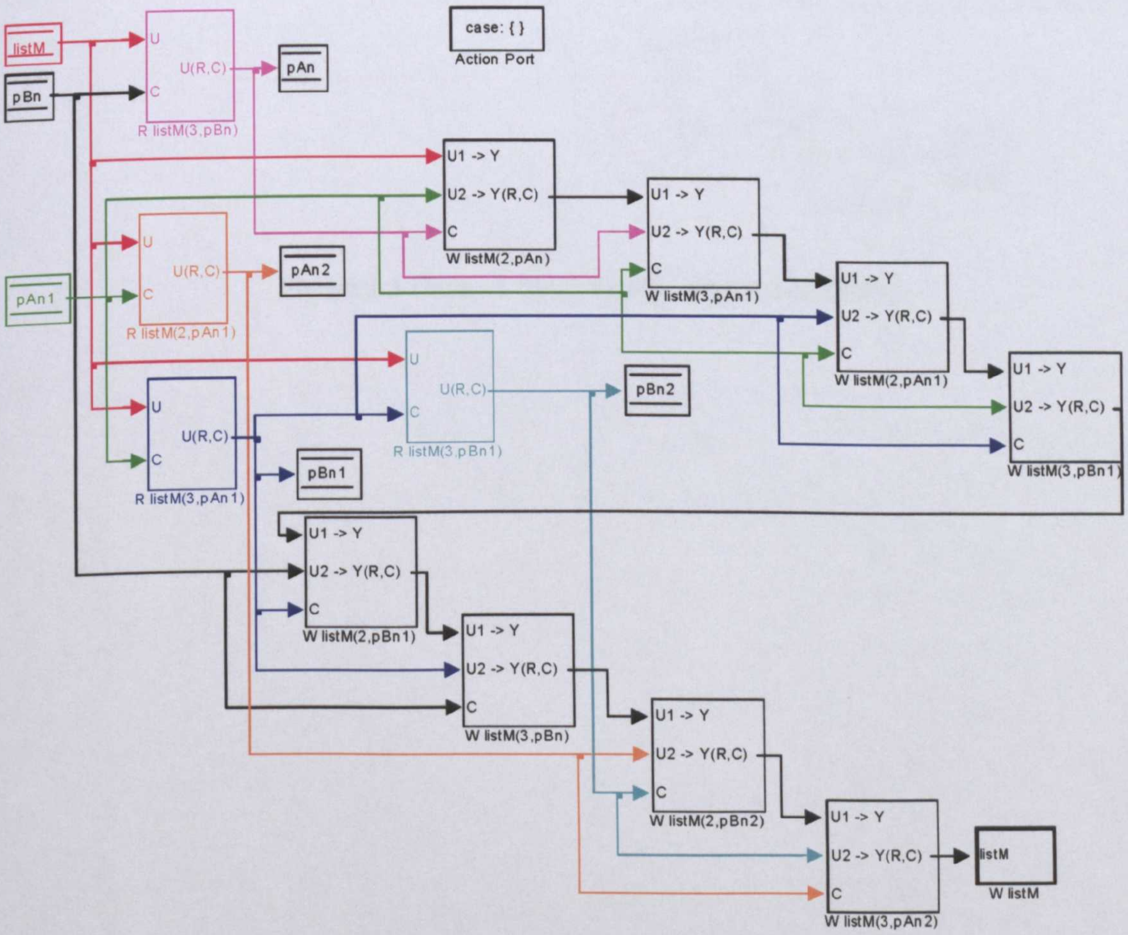


Figure J.2 Case 1 block (nut position monitoring)

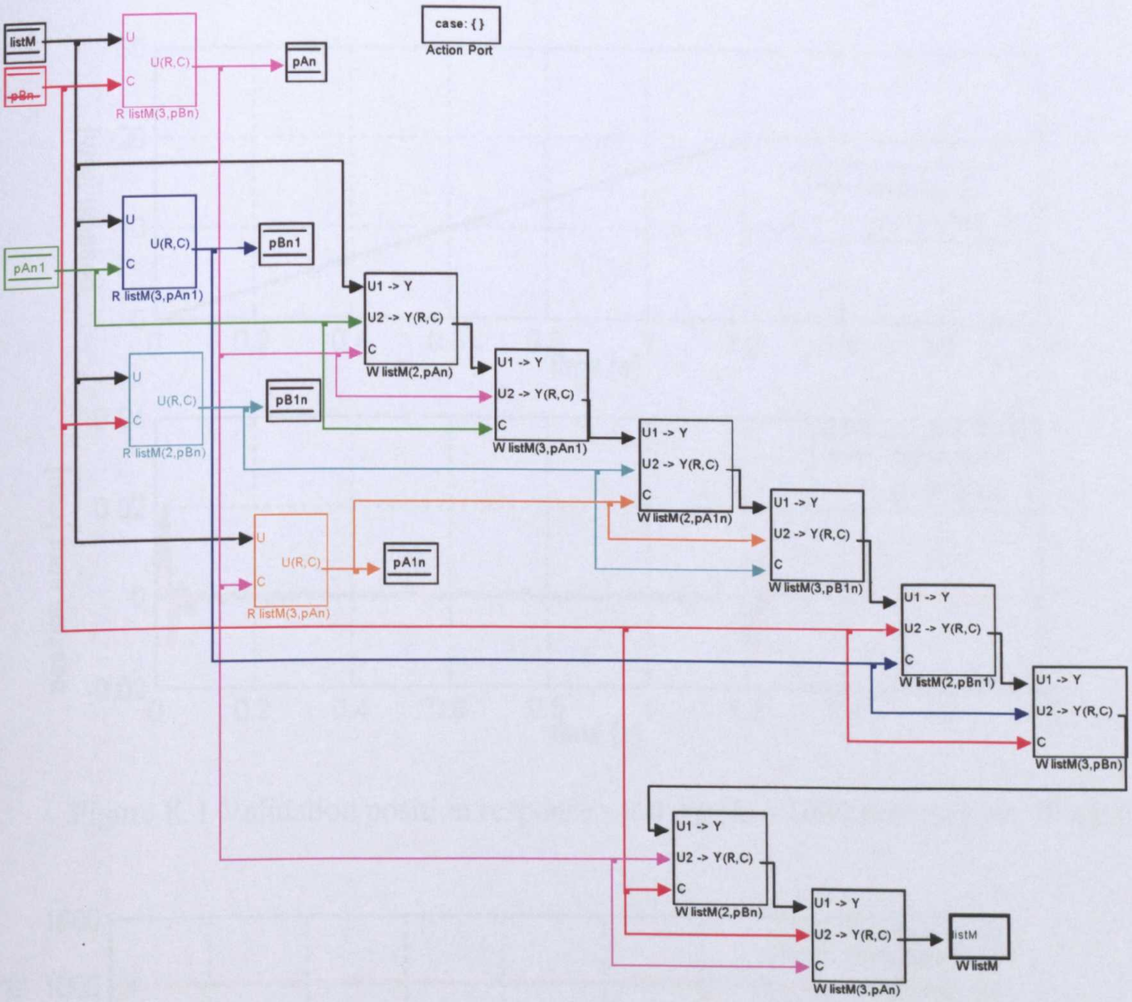


Figure J.3 Case –1 block (nut position monitoring)

APPENDIX K TEST RIG VALIDATION RESULTS

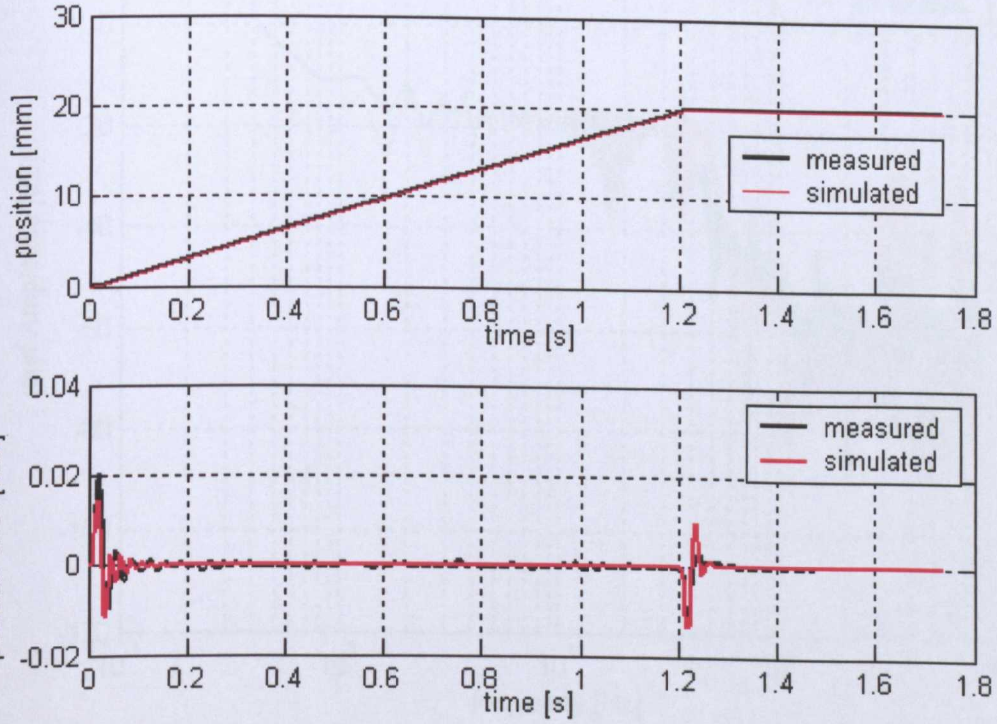


Figure K.1 Validation position response - test rig (fr = 1000 mm/min, d= 20 mm)

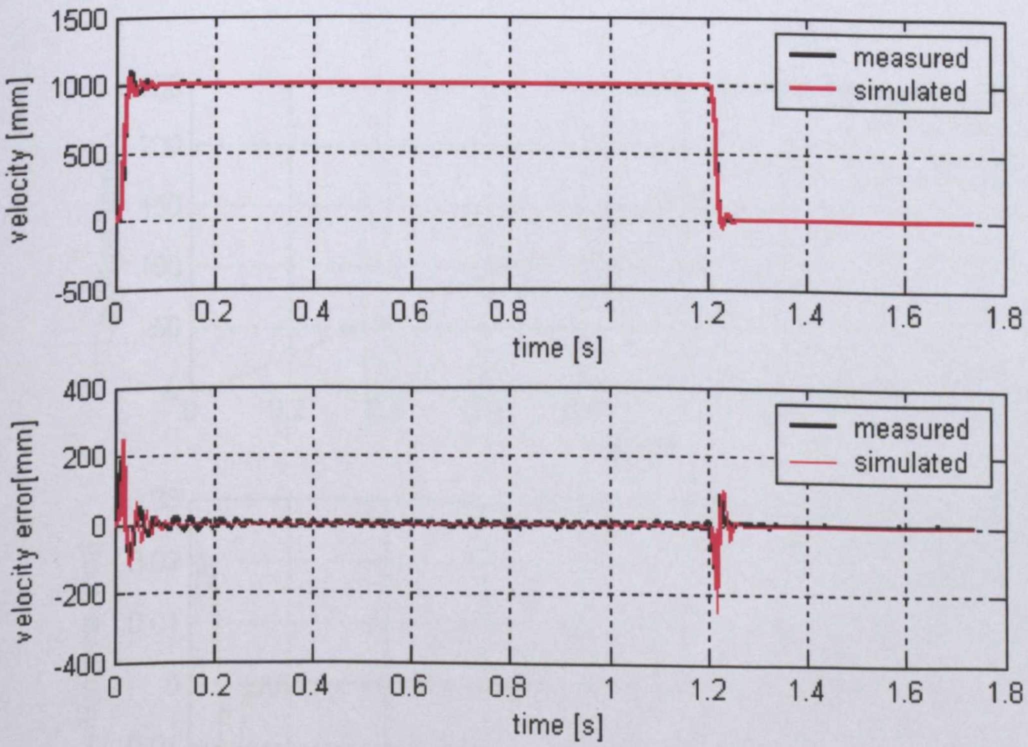


Figure K.2 Validation velocity response - test rig (fr = 1000 mm/min, d= 20 mm)

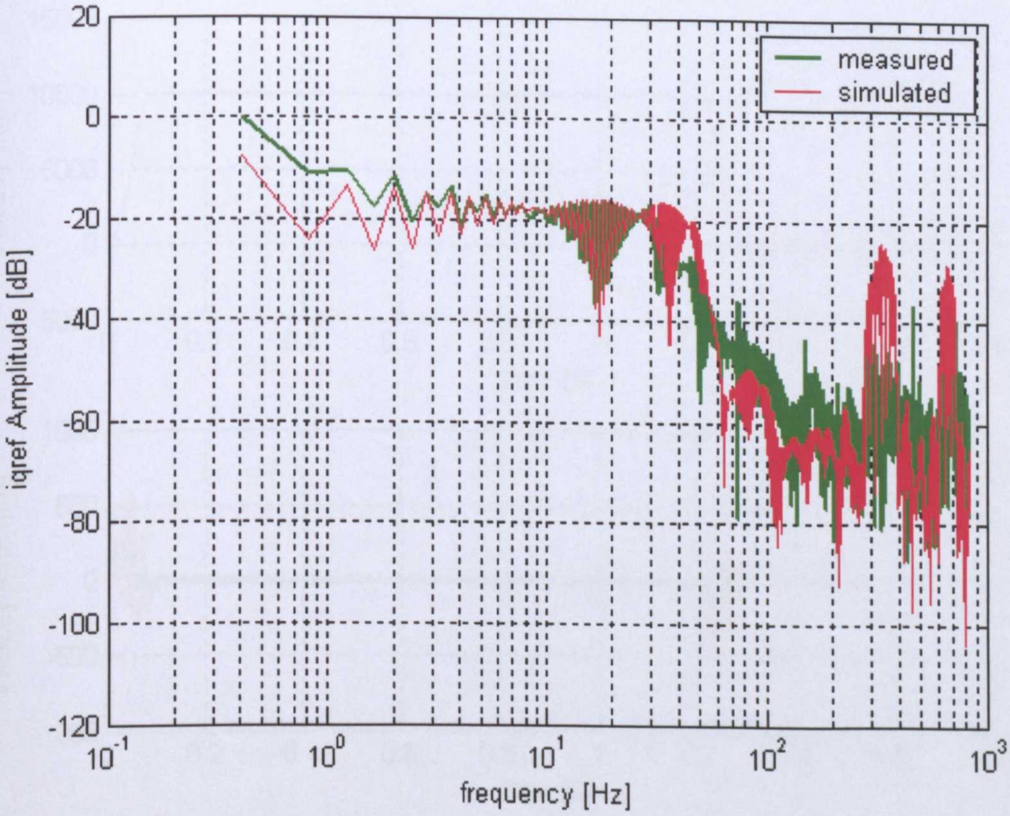


Figure K.3 Frequency response velocity loop - test rig (fr = 1000 mm/min, d= 20 mm)

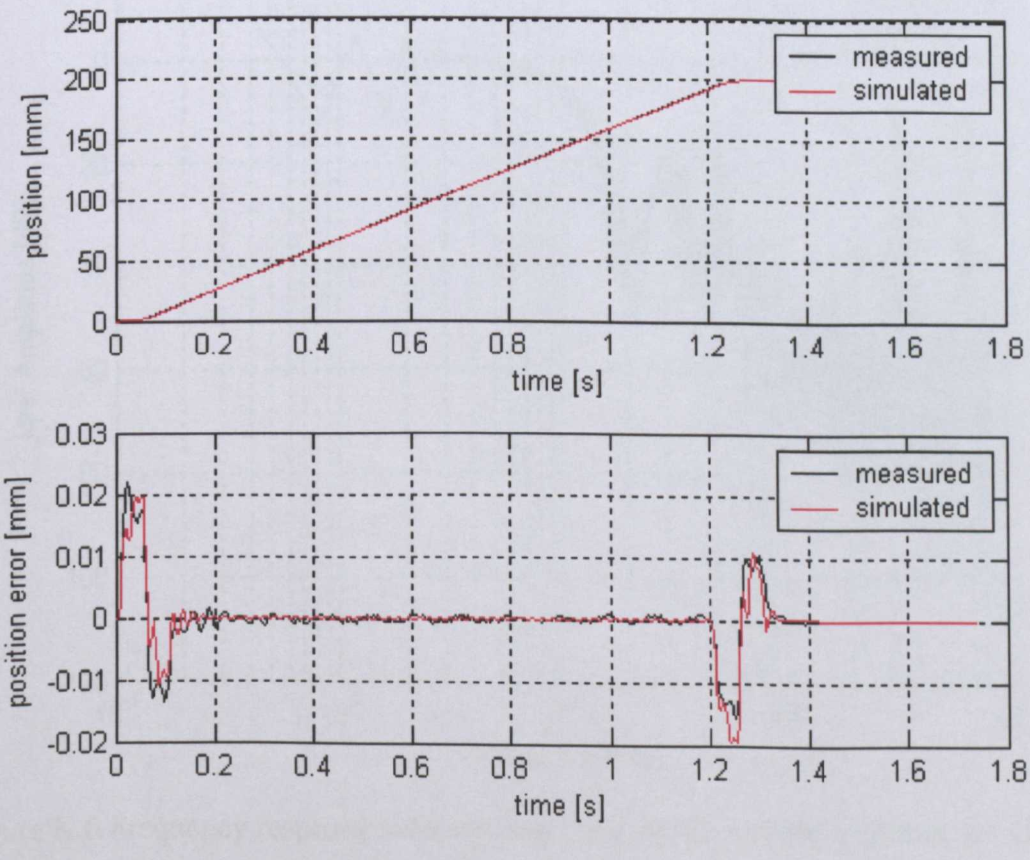


Figure K.4 Validation position response - test rig (fr = 10000 mm/min, d= 200 mm)

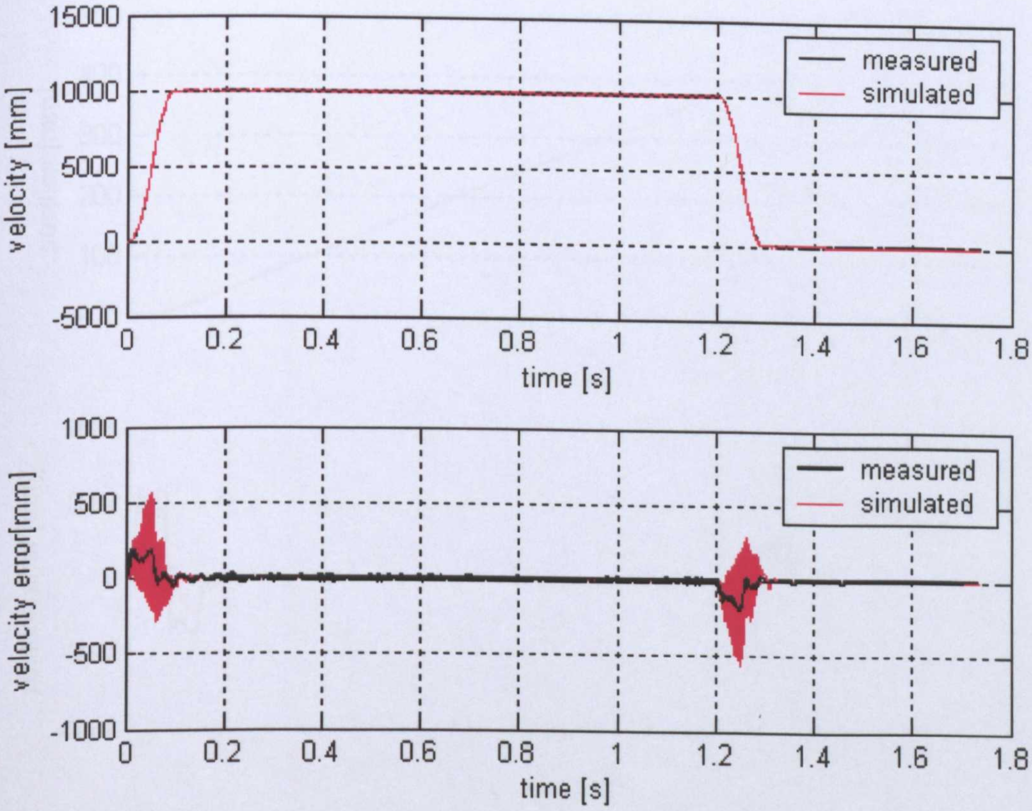


Figure K.5 Validation velocity response - test rig (fr = 10000 mm/min, d= 200 mm)

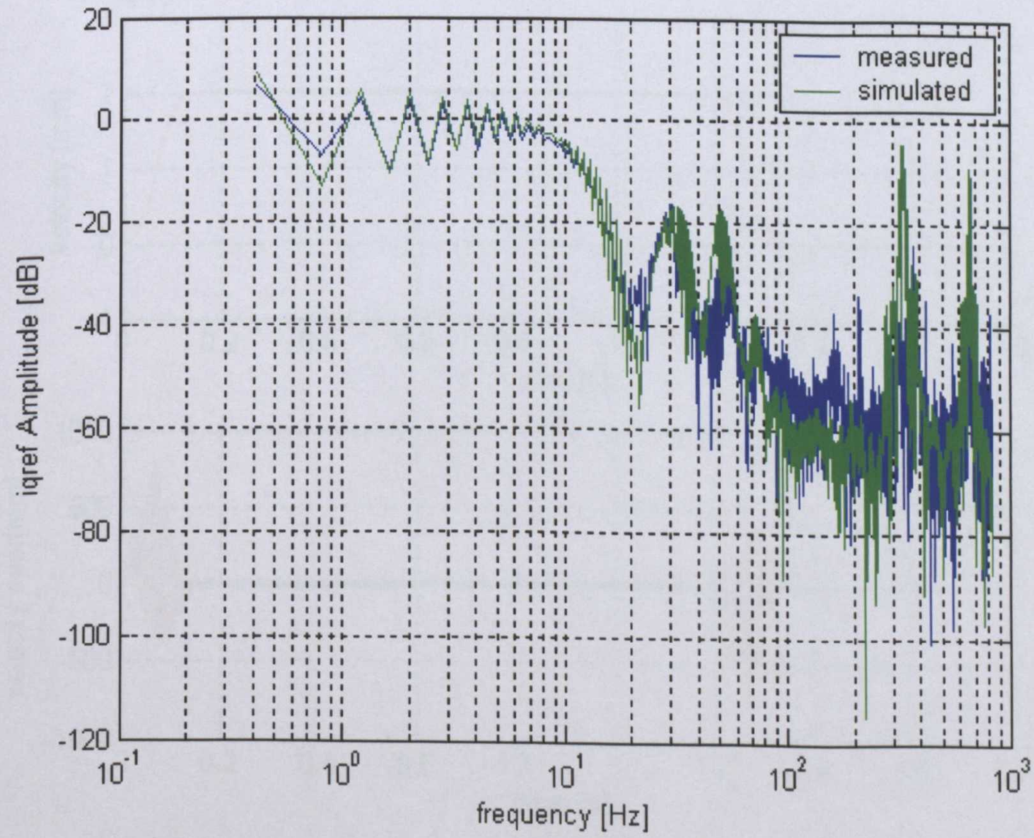


Figure K.6 Frequency response velocity loop - test rig (fr = 10000 mm/min, d= 200 mm)

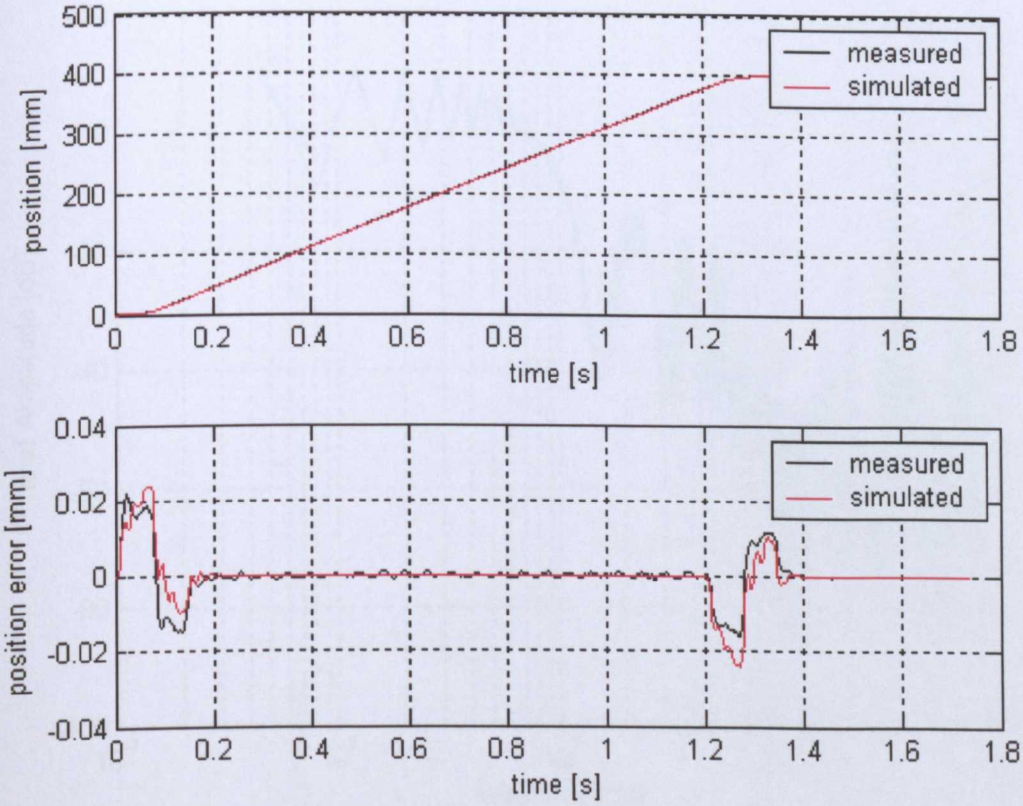


Figure K.7 Validation position response - test rig (fr = 20000 mm/min, d= 200 mm)

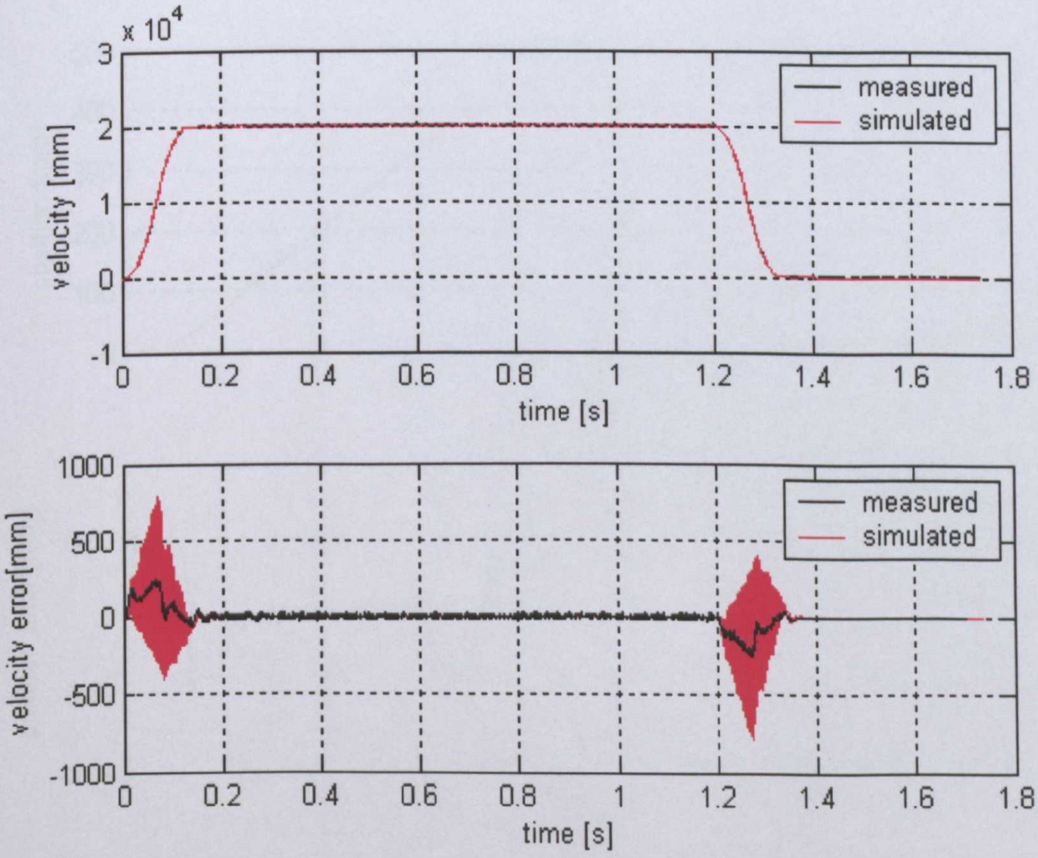


Figure K.8 Validation velocity response - test rig (fr = 20000 mm/min, d= 200 mm)

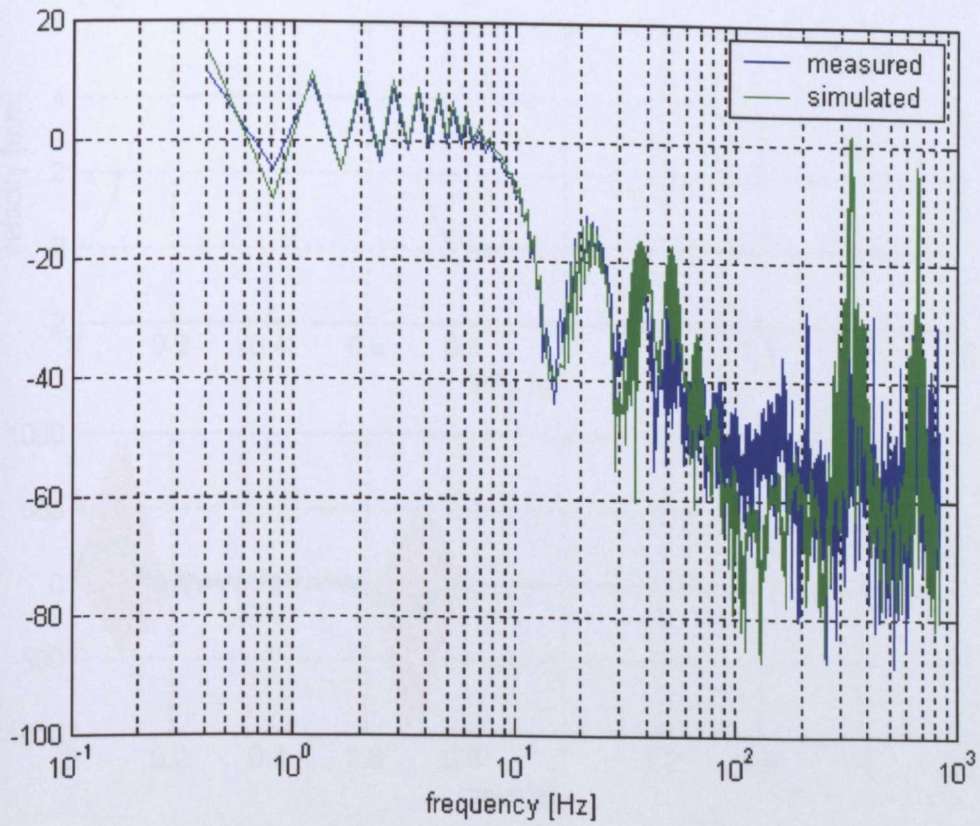


Figure K.9 Frequency response velocity loop - test rig (fr = 20000 mm/min, d= 200 mm)

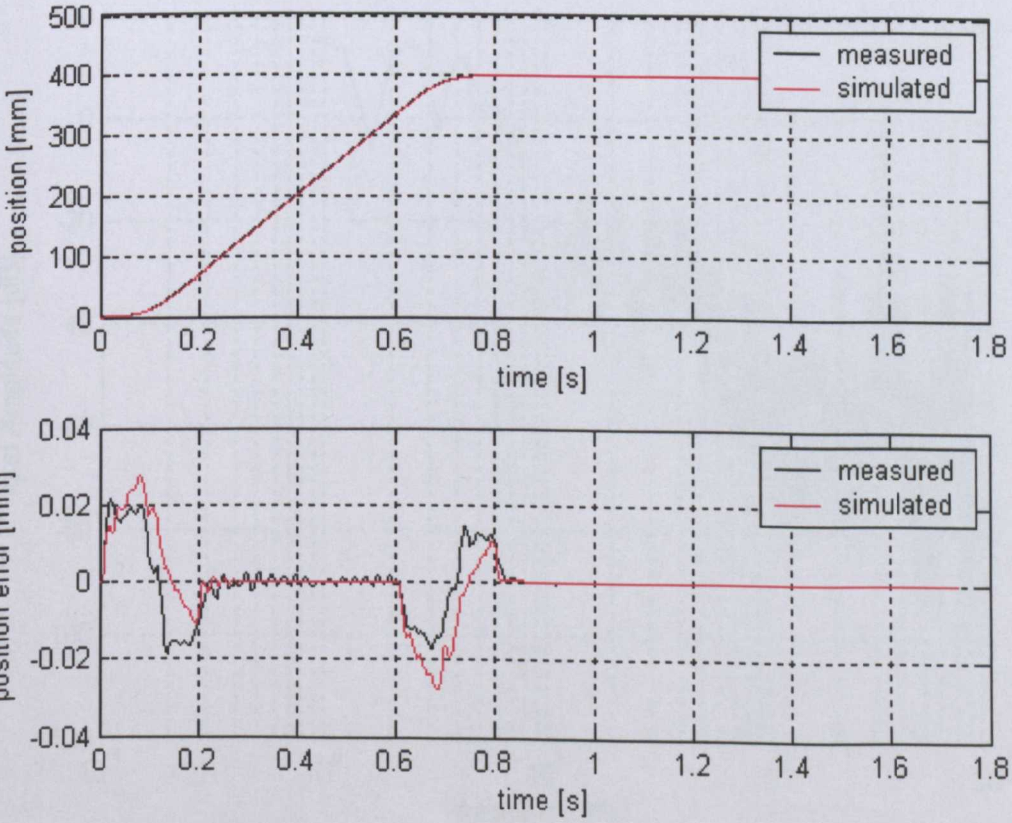


Figure K.10 Validation position response - test rig (fr = 40000 mm/min, d= 200 mm)

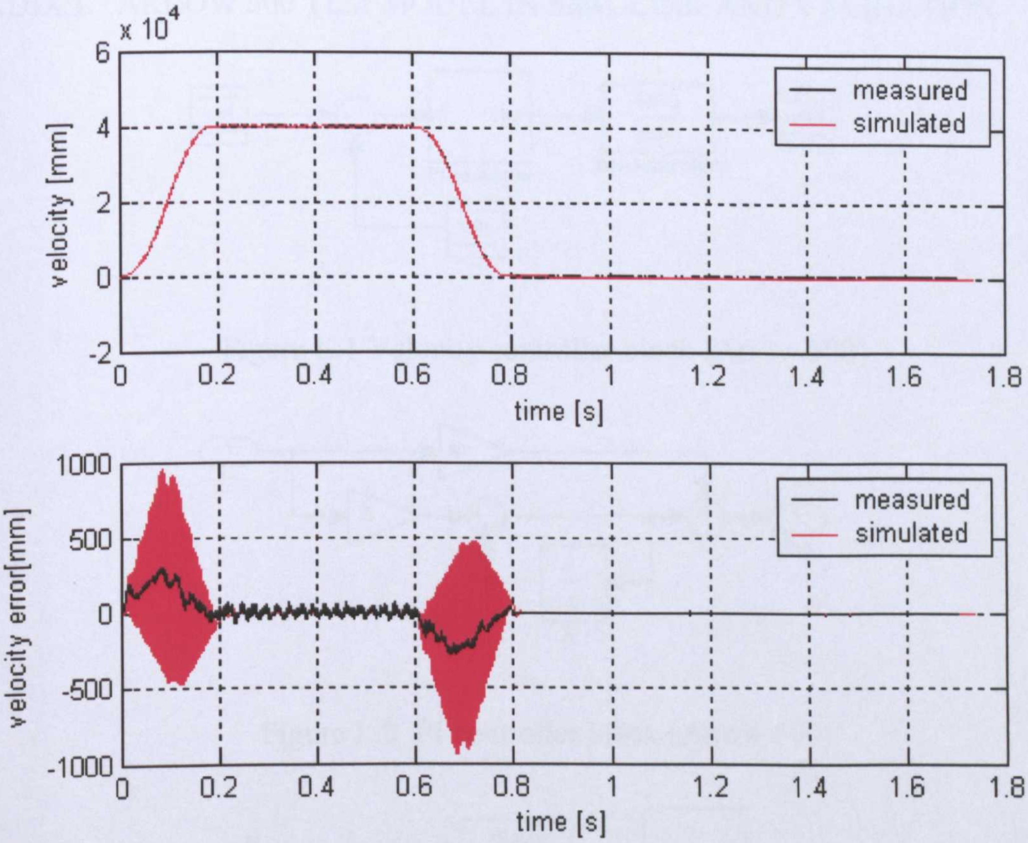


Figure K.11 Validation velocity response - test rig (fr = 40000 mm/min, d= 200 mm)

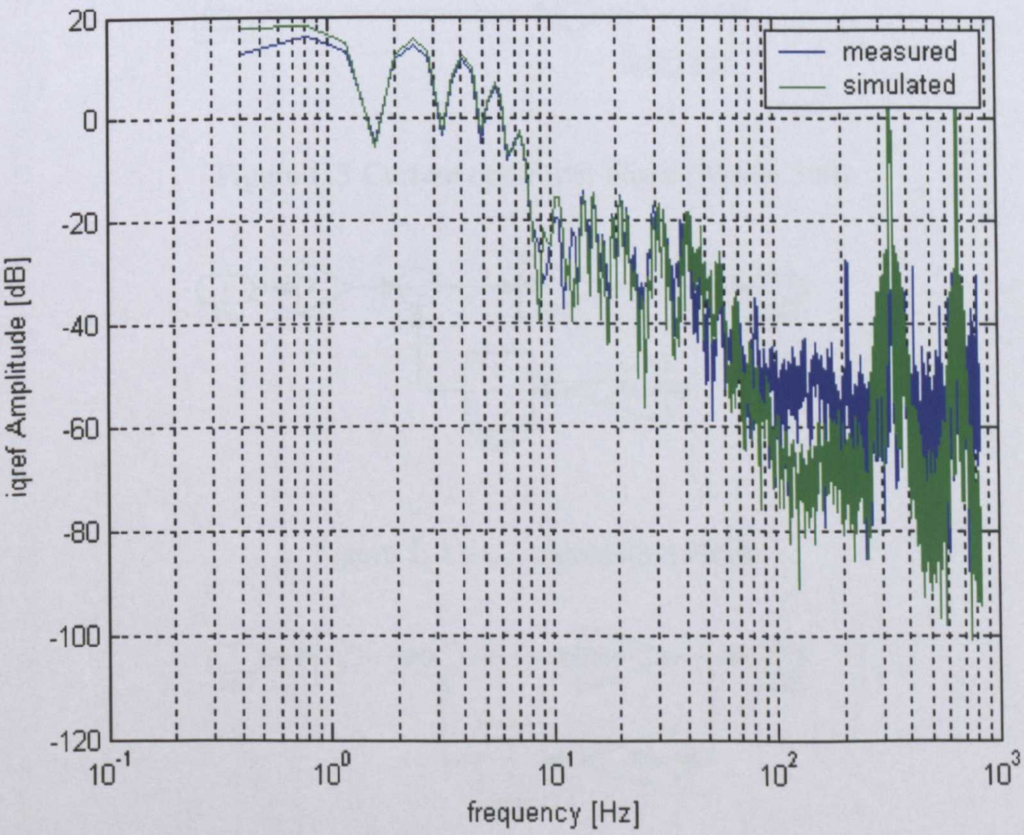


Figure K.12 Frequency response velocity loop - test rig (fr = 40000 mm/min, d= 200 mm)

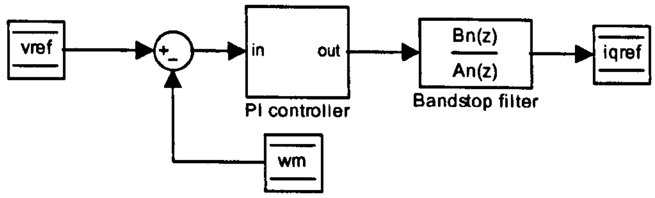


Figure L.1 Velocity controller block (Arrow 500)

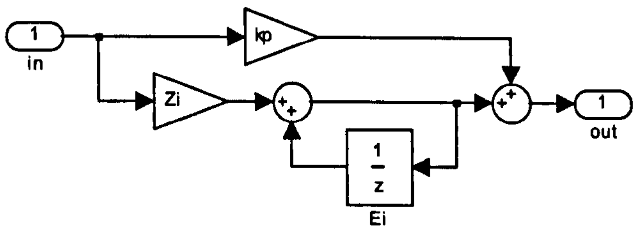


Figure L.2 PI controller block (Arrow 500)

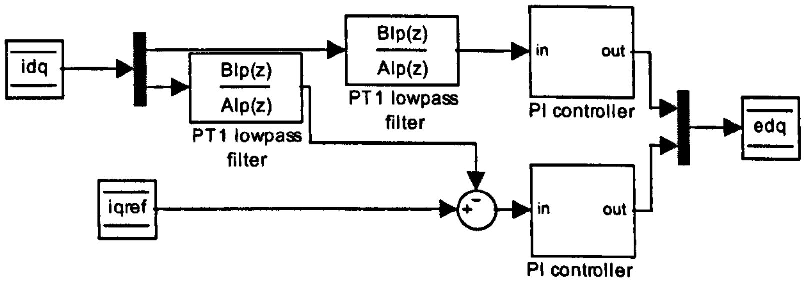


Figure L.3 Current controller block (Arrow 500)

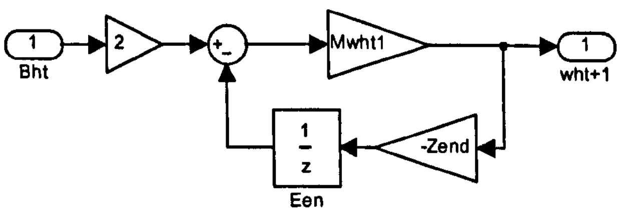


Figure L.4 w_{ht+1} calculation block

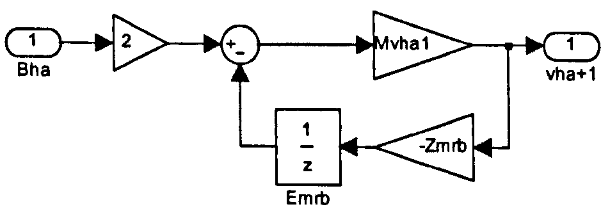


Figure L.5 Rear bearing mounting stiffness block (Arrow 500)

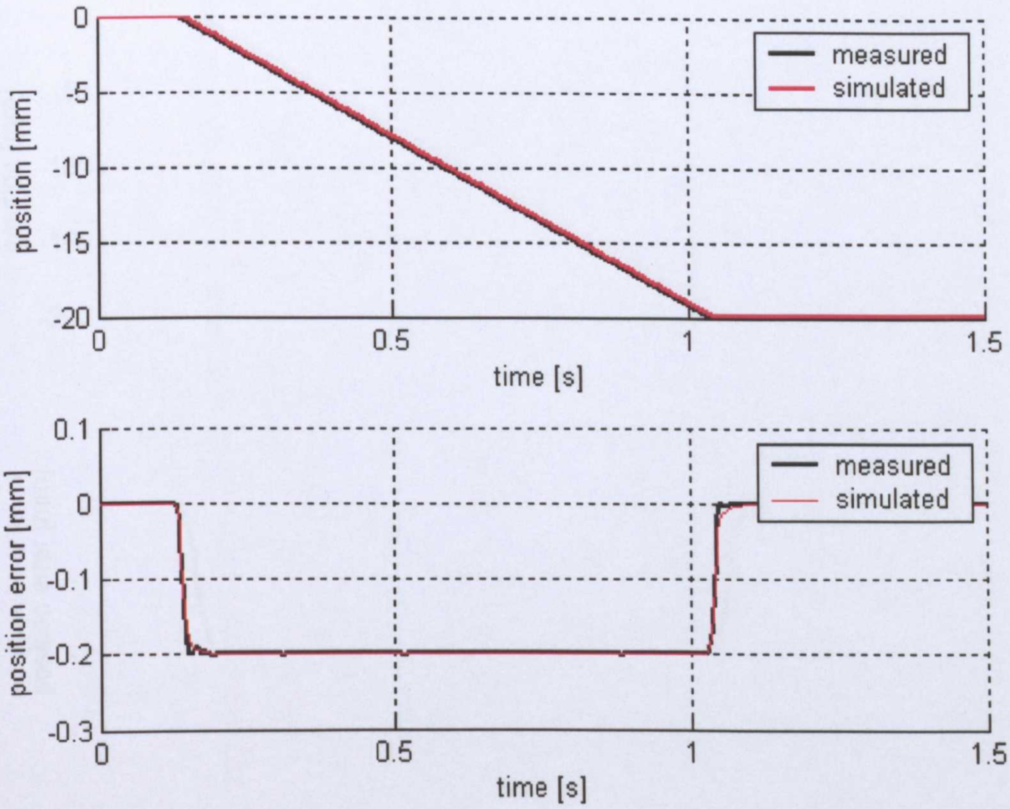


Figure L.6 Validation position response - arrow (fr = 1000 mm/min, d= 20 mm)

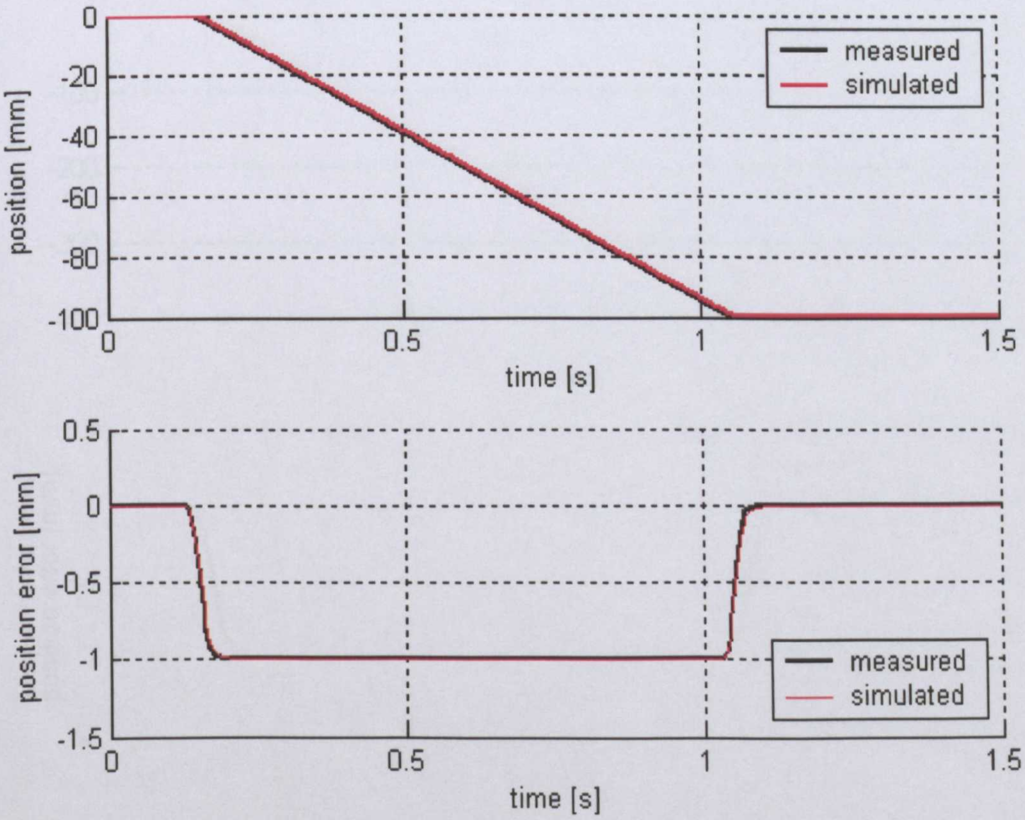


Figure L.7 Validation position response - arrow (fr = 5000 mm/min, d= 100 mm)

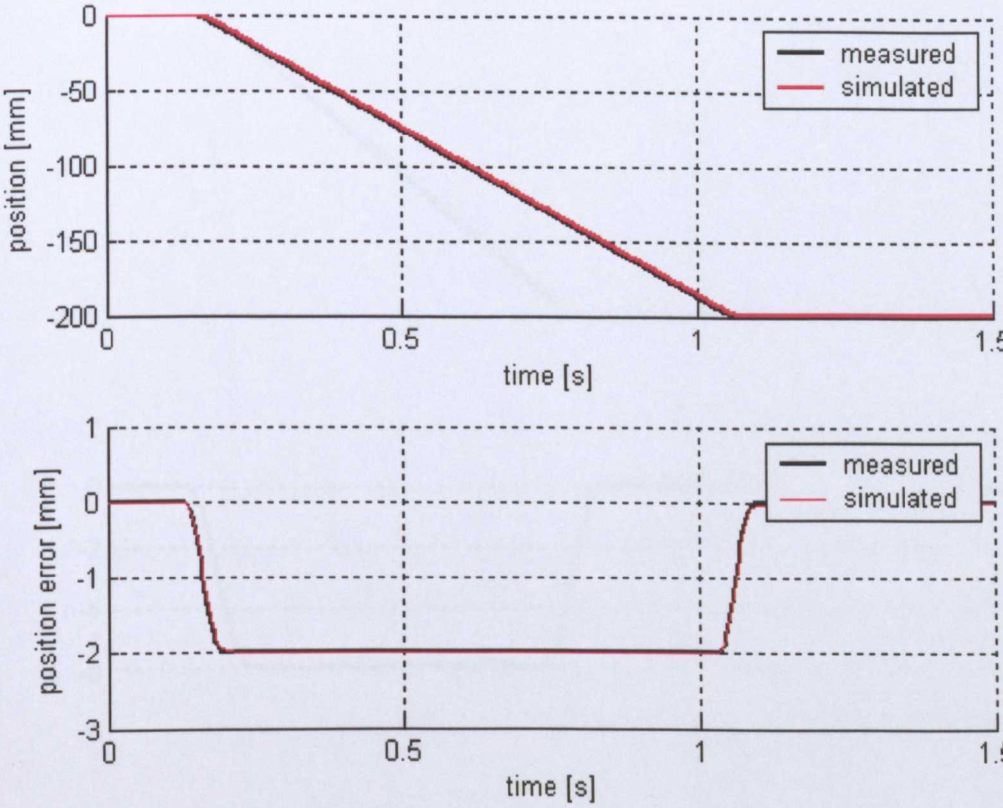


Figure L.8 Validation position response - arrow (fr = 10000 mm/min, d= 200 mm)

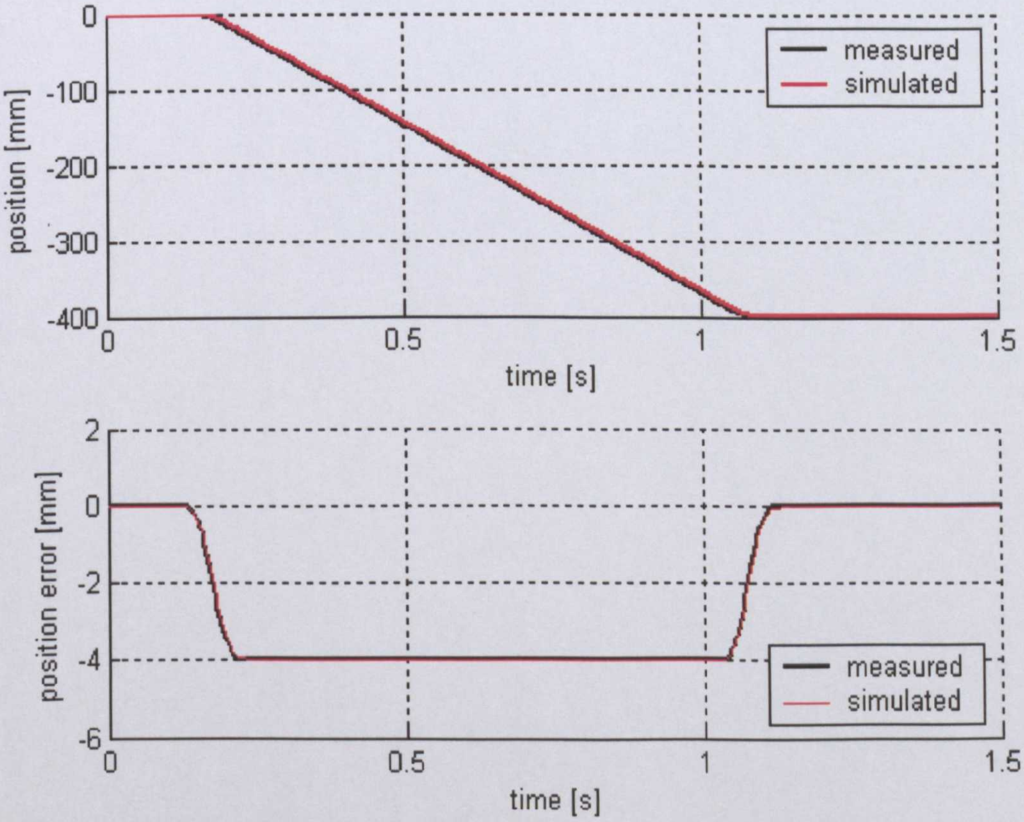


Figure L.9 Validation position response - arrow (fr = 20000 mm/min, d= 400 mm)

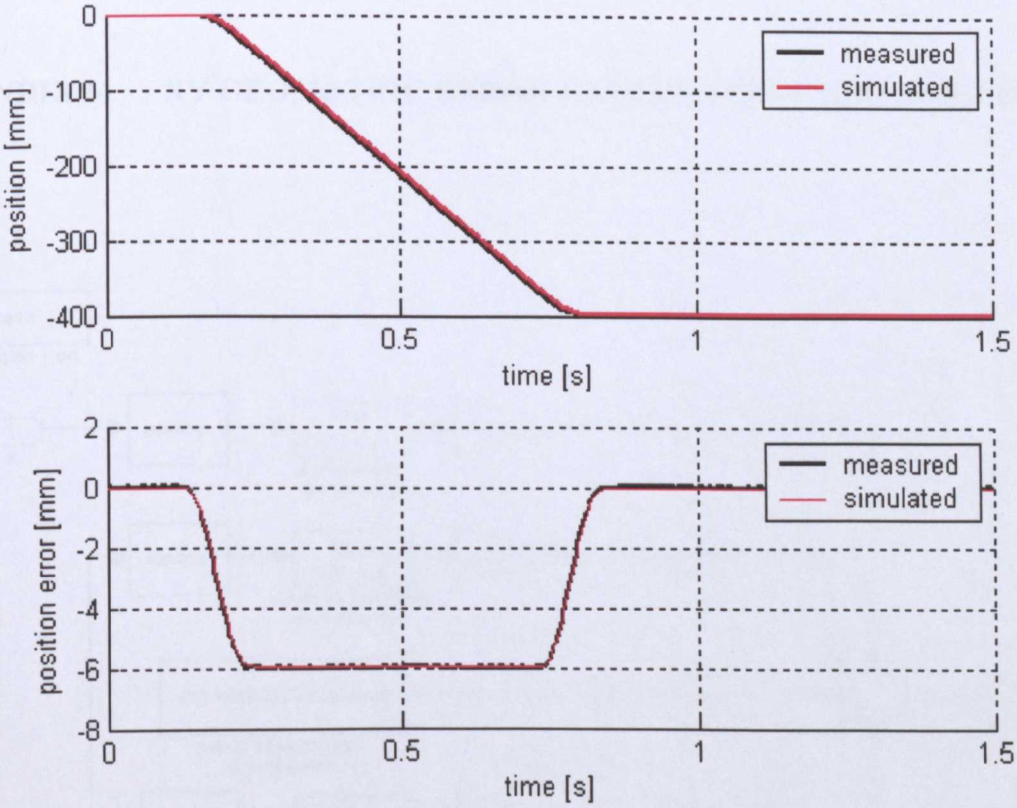


Figure L.10 Validation position response - arrow (fr = 30000 mm/min, d= 400 mm)

APPENDIX M XY GEOMETRIC ERROR CALCULATION SIMULINK BLOCK

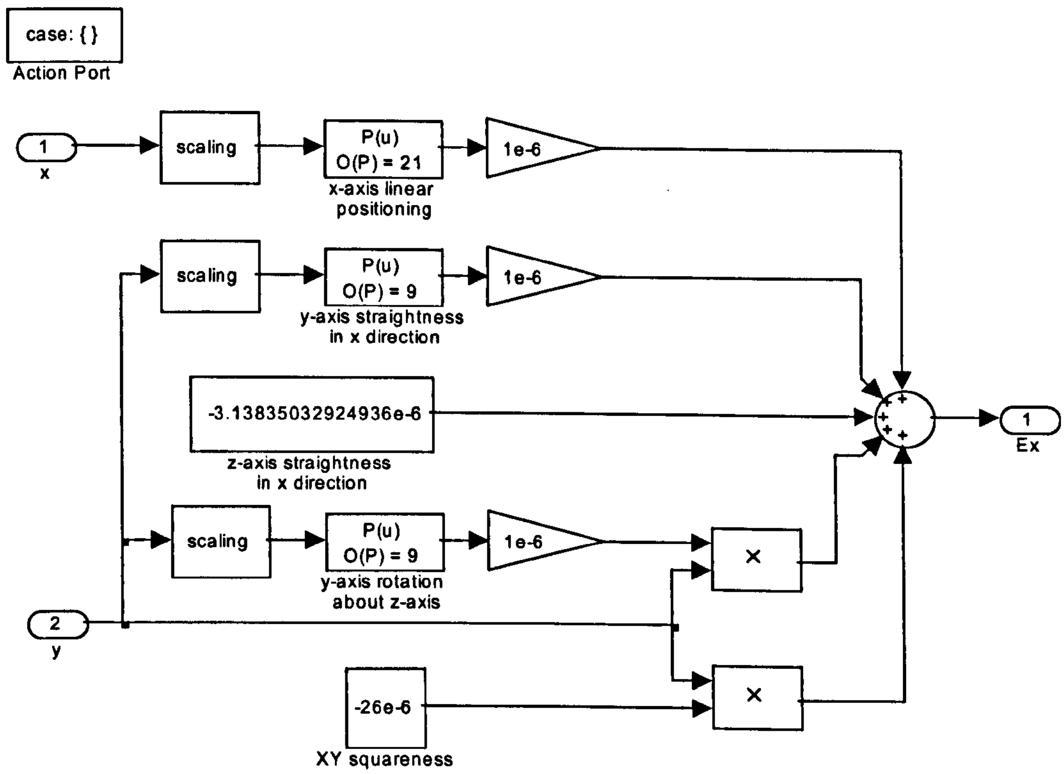


Figure M.1 X-Axis geometric error calculation (Case –1 Reverse)

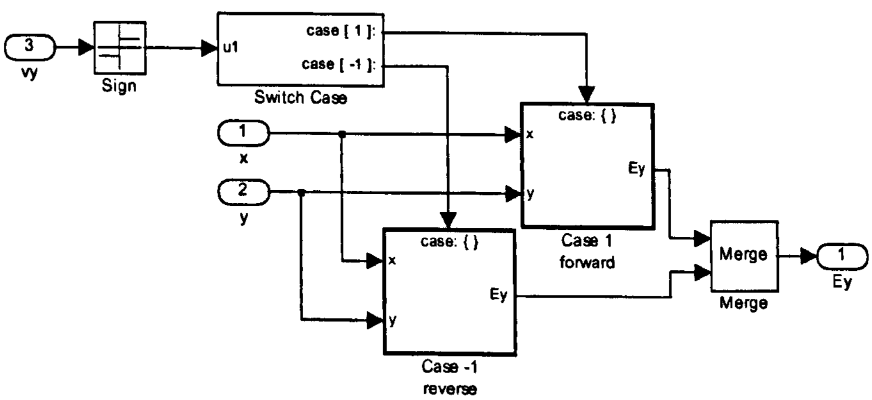


Figure M.2 Y-Axis geometric error calculation

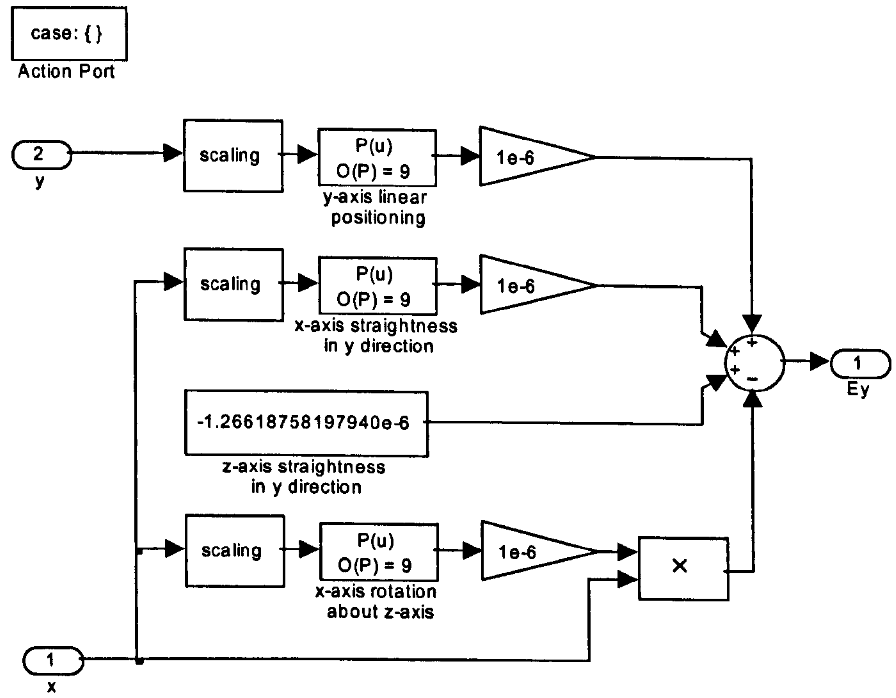


Figure M.3 Y-Axis geometric error calculation (Case 1: Forward)

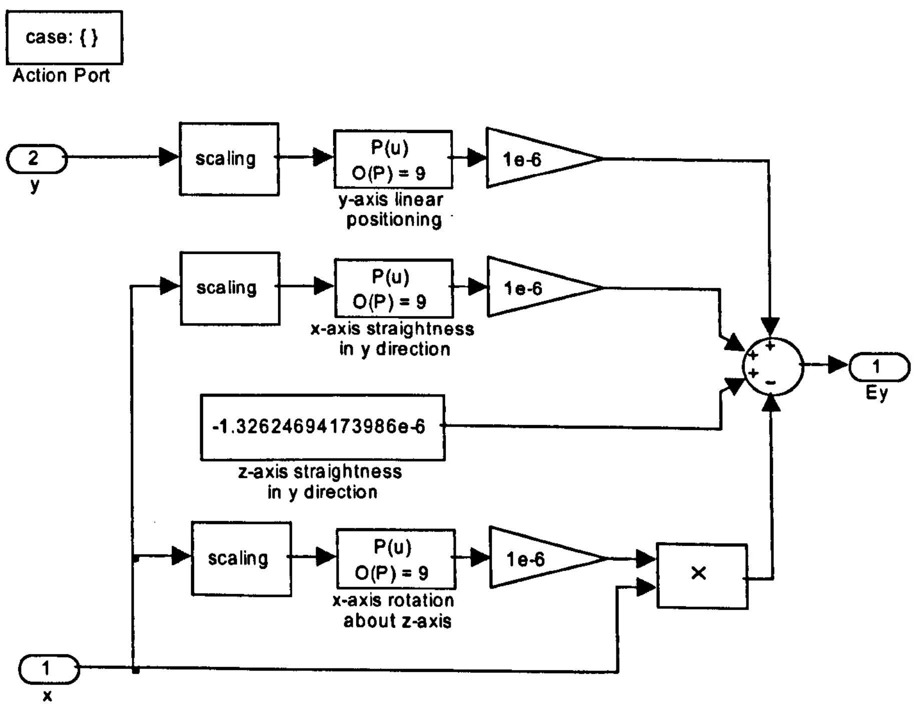


Figure M.4 Y-Axis geometric error calculation (Case -1: Reverse)

APPENDIX N MEASURED FREQUENCY RESPONSE Y-AXIS ARROW 500

Figure N.1 shows the controlled system frequency response measured on the y-axis. Poles at about 57, 75 and 555 Hz, with their corresponding zeros at 48, 61 and 450 Hz are identified. A natural frequency of 45 Hz on the y-axis is depicted in Figure N.2

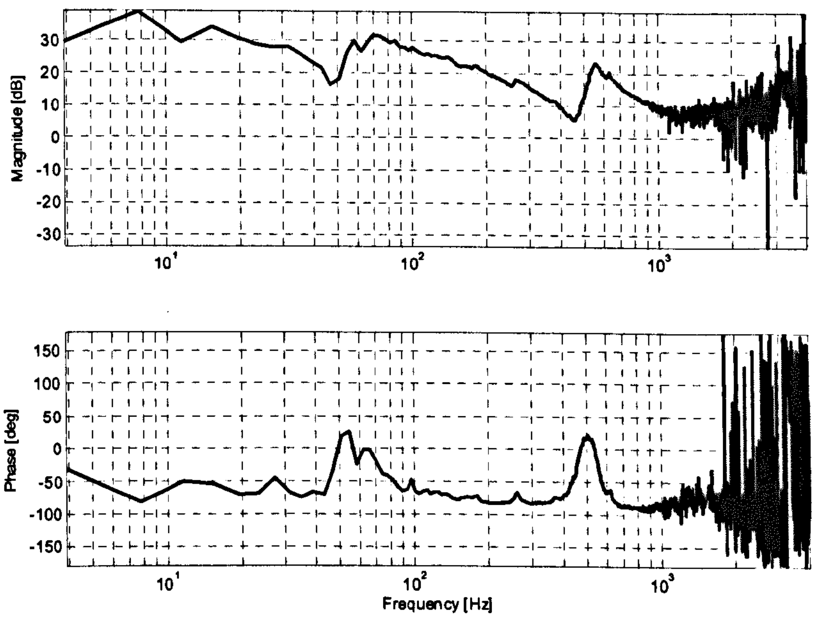


Figure N.1 Y-axis controlled frequency response

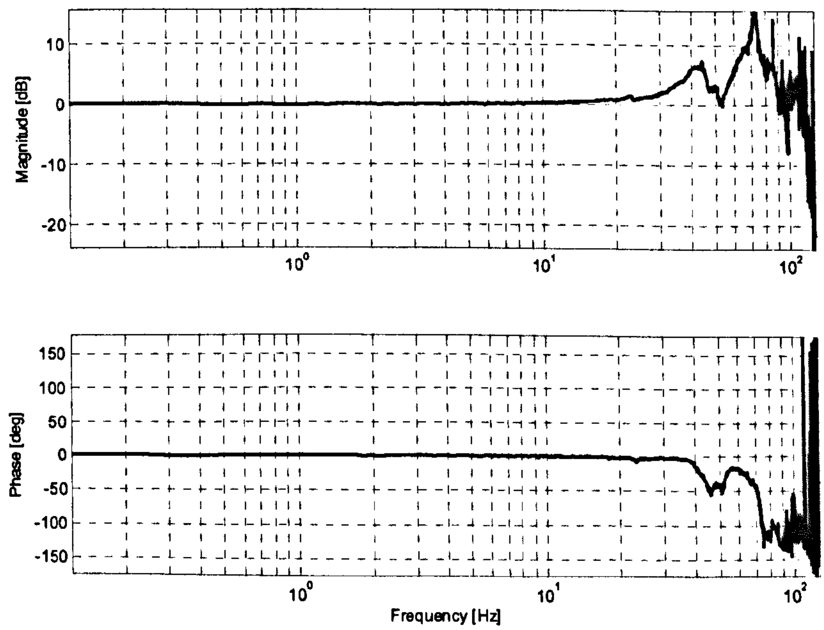


Figure N.2 Y-axis mechanical frequency response

APPENDIX O IDENTIFICATION OF MODAL PARAMETERS USING THE MODIFIED MORLET WAVELET

The idea of the continuous wavelet transform is to decompose a signal $f(t)$ into wavelet coefficients $W_\psi(a, b)$ using the basis of son wavelets $\psi_{a,b}(t)$ [142]. If $f(t)$ satisfies the condition, then:

$$\int_{-\infty}^{+\infty} |f(t)|^2 dt < 0 \tag{O.1}$$

The wavelet transform of $f(t)$ is expressed by the following product:

$$W_\psi(a,b) = \int f(t)\psi^*_{a,b}(t)dt \tag{O.2}$$

Where the asterisk denotes complex conjugation. This equation shows how a function $f(t)$ is discomposed into a set of basis functions $\psi_{a,b}(t)$, which are generated by dilatation and translation from the mother wavelet $\psi(t)$ as follows:

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right) \quad a > 0, \quad b \in R \tag{O.3}$$

Where a is the dilatation or scale parameter defining the support width of the son wavelet and b the translation parameter localising the son wavelet function in the time domain. The factor $a^{-1/2}$ is used to ensure energy preservation in the wavelet transform. The function $\psi(t)$ must satisfy the admissibility condition:

$$0 < c_\psi = \int_{-\infty}^{+\infty} \frac{|\psi(\omega)|^2}{|\omega|} d\omega < \infty \tag{O.4}$$

Where $\psi(\omega)$ is the Fourier transform of $\psi(t)$. Then the wavelet transform can be inverted and the signal $f(t)$ recovered:

$$f(t) = \frac{1}{c_\psi} \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} W_\psi(a,b)\psi_{a,b}(t)\frac{da}{a^2}db \tag{O.5}$$

The wavelet transform is expressed in terms of the Fourier transform to explain the frequency localisation. Then, using Parseval's theorem for $F(\omega)$ the Fourier transform of the signal $f(t)$ and $a\psi^*(a\omega)e^{j\omega b}$ the Fourier transform of the son wavelet $\psi^*(t-b)/a$ gives

$$W_\psi(a,b) = \frac{\sqrt{a}}{2\pi}\int_{-\infty}^{+\infty} F(\omega)\psi^*(a\omega)e^{j\omega b} d\omega \tag{O.6}$$

The parameters a and b are discretised for computer calculation of $W_\psi(a,b)$. In this case, the wavelet is dilated and translated discretely by selecting

$$a = a_0^m \tag{O.7}$$

$$b = nb_0 a_0^m \tag{O.8}$$

Where a_0 and b_0 are fixed values with $a_0 > 1$, $b_0 > 0$, m, $n \in Z$. As a result, a discretised son wavelet and a corresponding discrete wavelet transform are obtained. The signal $f(t)$ is thus discomposed into sub-bands with a bandwidth that increases linearly with frequency. Octave-wide bands are achieved by doing a_0=2 and b_0=1 (dyadic discretisation). Thus, $a = 2^m$ and $b = n2^m$.

O.1 The Morlet Wavelet

The Morlet wavelet is defined by

$$\psi(t) = e^{j\omega_0 t} e^{-t^2/2} \tag{O.9}$$

Where ω_o is the central wavelet frequency. This value is generally chosen superior to five in order to verify the admissibility condition (equation O.1) as stated by Fasana et al [143]. The dilated version of the Fourier transform of $\psi(t)$ is real and is given by

$$\psi(a\omega) = \sqrt{2\pi} e^{-(a\omega-\omega_0)^2/2} \tag{O.10}$$

$\psi(a\omega)$ reaches its maximum value when $\omega = \omega_o/a$, thus the value of a at which the wavelet filter is focused on the frequency ω is determined from

$$a = \omega_0 / \omega \tag{O.11}$$

If the analysed frequency is important, the dilatation parameter becomes small and the spectrum of the Morlet wavelet function is wide. This effect produces a bad spectral resolution that makes it difficult to differentiate closed modes.

Lardies and Gouttebroze [144] proposed a modified Morlet wavelet function that offers a better compromise in terms of localisation, in both time and frequency for a signal, than the traditional Morlet wavelet function. They introduced a parameter N in equation (O.9) in order to get a narrower spectrum allowing a better resolution of closely spaced modes.

$$\psi(t) = e^{j\omega_0 t} e^{-t^2/N} \tag{O.12}$$

With $N > 0$, equation (O.10) becomes

$$\psi(a\omega) = \sqrt{N\pi} e^{-(a\omega-\omega_0)^2 N/4} \tag{O.13}$$

O.2 Implementation of the Wavelet Transform

A MATLAB program for the calculation of the CWT using the modified Morlet mother wavelet was written on the basis of equations presented in the preceding section. The program is included in Appendix 0.4. The CWT of a signal is calculated considering the following steps:

- Compute the FFT of the signal.
- For a given scale a, sample the wavelet with m data points within the range of the *Arg* function ($[-\pi \;\; \pi]$). Where m is the number of samples of the FFT of the signal. The samples are represented in counter clockwise direction from the positive x-axis ($[0,\ 2\pi/m,\ 4\pi/m\ \ldots\ \pi\text{-}4\pi/m,\ \pi\text{-}2\pi/m,\ -\pi,\ -(\pi\text{-}2\pi/m),\ -(\pi\text{-}4\pi/m)\ \ldots\ -4\pi/m,\ -2\pi/m]$).
- Compute the FFT of the wavelet at the scale a for a given wavelet shape factor R_f calculates as

$$R_f = \sqrt{N/2} \tag{O.14}$$

  Equation (O.13) becomes:

$$\psi(a\omega) = R_f \sqrt{2\pi} \quad e^{-R_f^2 (a\omega - \omega_0)^2 / 2} \tag{O.15}$$

  Where $R_f \omega_0 > 5$ and $N \geq 2$. Note that $N = 2$ for the Morlet mother wavelet.
- Multiply the FFT of the signal by the complex conjugated of the FFT of the wavelet.
- Compute the IFFT to obtain the wavelet coefficients for the scale a.

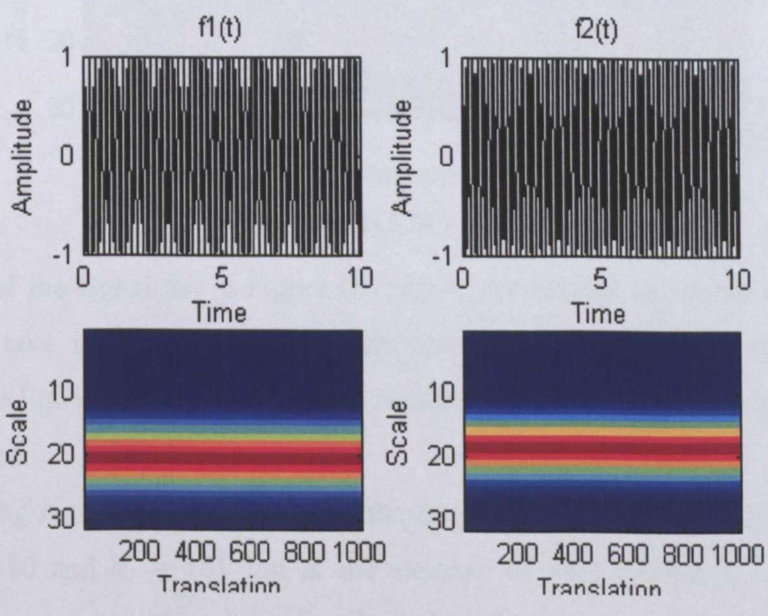


Figure O.1 $f_1(t)$ and $f_2(t)$ CWT (Morlet: $N = 2$)

The CWT of a signal $f(t)$ composed by the sum of two sinusoids, $f_1(t)$ and $f_2(t)$, with closed frequencies was calculated to show the effect of the shape factor R_f on the modified Wavelet results.

$$f_1(t) = \sin(2\pi t / 20) \tag{O.16}$$

$$f_2(t) = \sin(1.1 * 2\pi t / 20) \tag{O.17}$$

$$f(t) = f_1(t) + f_2(t) \tag{O.18}$$

Figure O.1 shows the calculated CWT for each signal ($f_1(t)$ and $f_2(t)$). The CWT of the signal $f(t)$ is illustrated in Figure O.2. Appendix 0.5 contains the MATLAB program used to calculate the CWT for this experiment.

Note how the CWT represents the signals $f_1(t)$ and $f_2(t)$ in the time/frequency domain (Figure O.1). A horizontal dark brown line centred at the scale a = 20 represents the signal $f_1(t)$; meaning an oscillation at a constant frequency of 5.5 Hz during the duration of the signal. The horizontal line level is moved to about 18 reflecting the oscillation frequency of $f_2(t)$ (5 Hz).

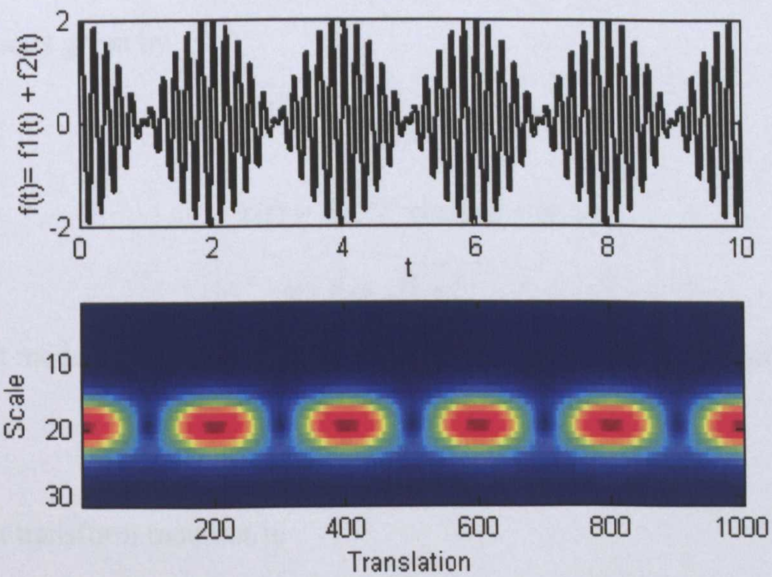


Figure O.2 $f(t)$ CWT (Morlet: N=2)

The CWT of the signal $f(t)$ in Figure O.2 shows the time at which the oscillations of the signal take its maximum amplitude (maximum concentration of energy). In contrast, the figure shows a bad spectral resolution and the two frequencies cannot be distinguished.

Increasing N to a hundred improves the frequency resolution of the closed spaced modes (a_1=20 and a_2 = 18), but at the expense of time resolution, as Figure O.3 illustrates.

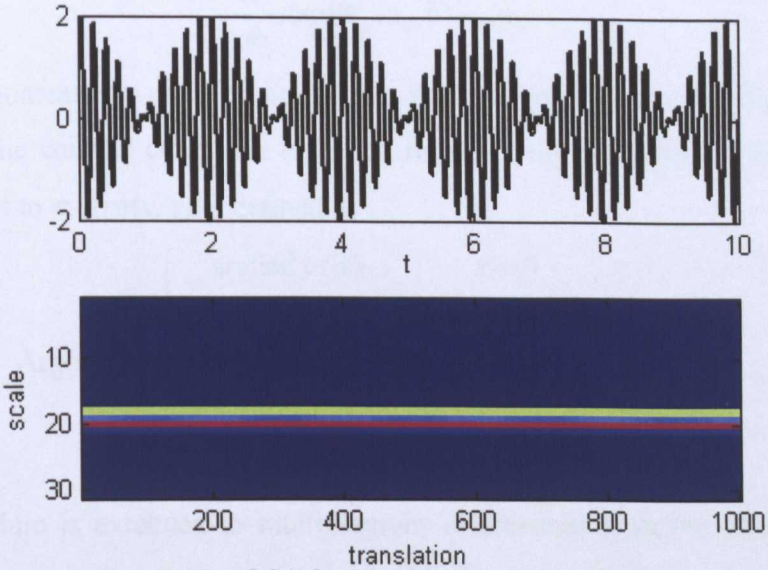


Figure O.3 CWT of $f(t)$ for N=128 (Modified Morlet wavelet)

O.3 Modal Parameters Identification Using Wavelets

As presented by Staszewski [127], the wavelet transform of a damped sinusoid $x(t)$ with ω_n the undamped natural frequency, ω_d the damped natural frequency and ζ the damping ratio is given by

$$W_\psi(a,b) = \frac{\sqrt{a}}{2} B e^{-\zeta\omega_n b} \psi^*(a\omega_d) e^{j(\omega_d b + \psi_o)} \tag{O.19}$$

Where, $$x(t) = B e^{-\zeta\omega_n t}\cos(\omega_d t + \psi_o) \tag{O.20}$$

$$\omega_d = \omega_n \sqrt{1-\zeta^2} \tag{O.21}$$

The wavelet modulus is localised at a constant value of the dilatation parameter noted a_o:

$$a = a_o = \omega_0 / \omega_d \tag{O.22}$$

The wavelet transform modulus is

$$\left|W_\psi(a_o,b)\right| = \frac{\sqrt{a_o}}{2} B e^{-\zeta\omega_n b} \left|\psi^*(a_o\omega_d)\right| \tag{O.23}$$

The damping ratio of the system can be estimated from the slope of the straight line of the logarithm of the wavelet transform modulus in equation (O.24)

$$\ln\left|W_\psi(a_o,b)\right| = -\zeta\omega_n b + \ln\left(\frac{\sqrt{a_o}}{2} B \left|\psi^*(a_o\omega_d)\right|\right) \tag{O.24}$$

And the wavelet transform phase is given by

$$Arg\left(W_\psi(a_o,b)\right) = \omega_d b + \psi_0 \tag{O.25}$$

Where, $$\frac{d}{db}Arg\left(W_{\psi}(a_o,b)\right)=\omega_d \tag{O.26}$$

Arg (the argument of a complex number) is the four-quadrant *arc tan* function, which represents the counter clockwise angle, in radians, from the positive x-axis. Values vary from $-\pi$ to π. $Arg(x, y)$ is defined as:

$$\mathrm{Arg(x,y)}=\begin{cases} \arctan(y/x) & x>0 \\ \pi/2 & x=0 \quad y>0 \\ -\pi/2 & x=0 \quad y<0 \\ \pi+\arctan(y/x) & x<0 \quad y\geq 0 \\ -\pi+\arctan(y/x) & x<0 \quad y<0 \end{cases} \tag{O.27}$$

This procedure is extended to multi-degrees of freedom systems, where ω_{nk} is the undamped natural frequency, ω_{dk} the damped natural frequency and ζ_k the damping ratio associated to the k^{th} mode, thus

$$W_{\psi}(a,b)=\frac{\sqrt{a}}{2}\sum_{k=1}^{p}B_k e^{-\zeta_k\omega_{nk}b}\psi^*(a\omega_{dk})e^{j(\omega_{dk}b+\psi_{ok})} \tag{O.28}$$

For a fixed value of the dilatation parameter ($a = a_i$), which maximises $\psi^*(a\omega_d)$, only the mode associated with a_i gives a relevant contribution to the wavelet transform, while the other terms are negligible. The wavelet transform of each separated mode i= 1, 2,... , p becomes

$$\ln\left|W_{\psi}(a_i,b)\right|=-\zeta_i\omega_{ni}b+\ln\left(\frac{\sqrt{a_i}}{2}B\left|\psi^*(a_i\omega_{di})\right|\right) \tag{O.29}$$

$$Arg\left(W_{\psi}(a_i,b)\right)=\omega_{di}b+\psi_{0i} \tag{O.30}$$

Thus, $$\frac{d}{db}Arg\left(W_{\psi}(a_i,b)\right)=\omega_{di} \tag{O.31}$$

The damped frequency and the damping ratio for each eigen-mode is estimated from the wavelet transform according to the following procedure:

- Calculate the dilatation parameters a_i by plotting the variations of the scale factor in time. The factor N is increased until good resolution is reached.
- The damped eigen-frequency in Hz ($f_{di}= \omega_{di}/2\pi$) is obtained from the slope of the phase of the wavelet transform - equation (O.32).
- Plot for each a_i the logarithm of the wavelet modulus as a function of time.
- The damping factors are estimated from the slope of the straight line of the wavelet modulus logarithm.

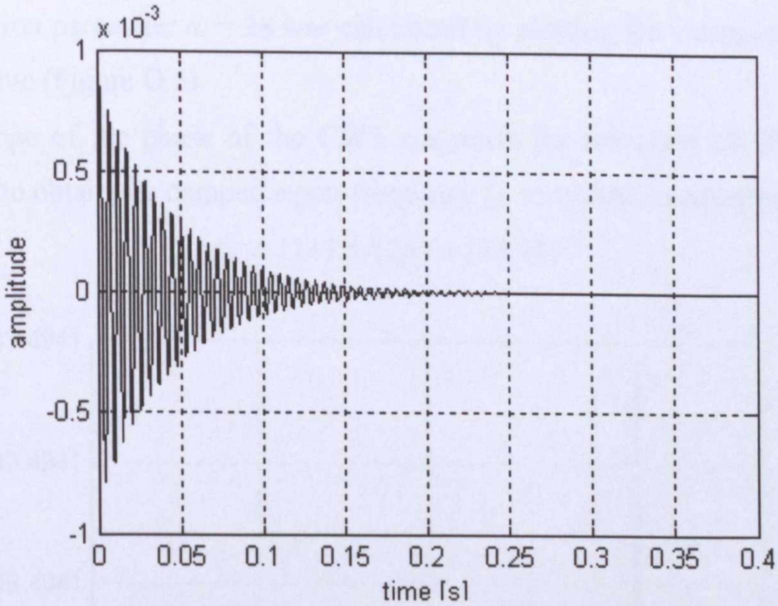


Figure O.4 Impulse response second-order system (equation (O.34))

This procedure was applied to calculate the damping factor and the damped natural frequency from the impulse response (Figure O.4) of a second order system represented for the following transfer function (see Appendix 0.6):

$$G(s) = \frac{1}{s^2 + 44.06s + 1.3218x10^6} \tag{O.32}$$

Where,
$$\omega_n = 1149.7 \tag{O.33}$$

$$\zeta = 0.019165 \tag{O.34}$$

$$f_d = \frac{\omega_n\sqrt{1-\zeta^2}}{2\pi} = 182.9477 \tag{O.35}$$

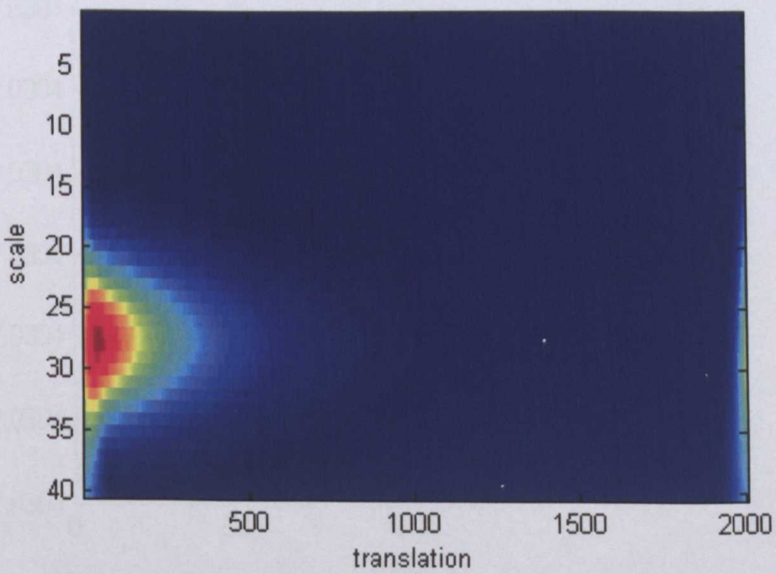


Figure O.5 CWT transform of the impulse response (equation (O.32))

The dilatation parameter $a_i = 28$ was calculated by plotting the variations of the scale factor in time (Figure O.5)

The slope of the phase of the CWT transform for the scale 28 (Figure O.6) is calculated to obtain the damped eigen-frequency f_{di} according to equation (O.3).

$$f_{di} = 1149.5/(2\pi) = 182.9477 \qquad (O.36)$$

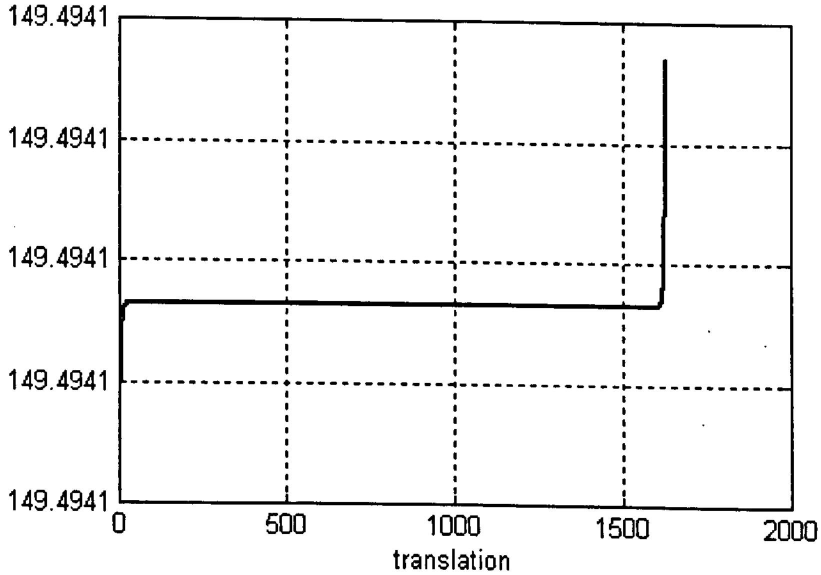


Figure O.6 Slope of the phase of the CWT transform (scale=28)

The damping factor is estimated, from the slope of the algorithm of the CWT transform for the scale 28 (Figure O.7), as

$$\zeta_i = \frac{22.0304}{1149.5} = 0.0191653 \qquad (O.37)$$

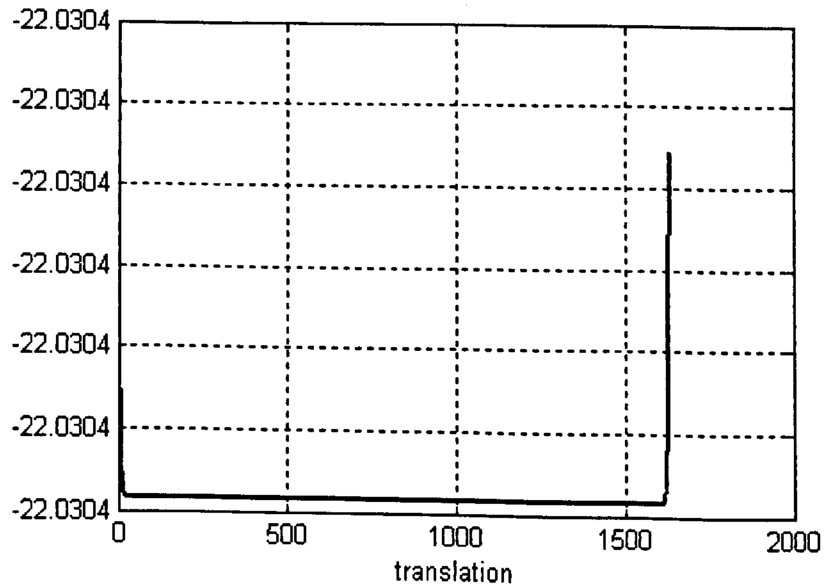


Figure O.7 Slope of the magnitude of the CWT transform (scale=28)

As seen, the method has identified the damped eigen-frequency and the damping factor accurately.

O.3.1 Implementation of the Identification Algorithm

A MATLAB program for the automatic calculation of the damping factor and damped natural frequencies using the modified Morlet mother wavelet was written on the basis of equations presented in the preceding section. The program is included in Appendix 0.7. The modal parameters are calculated considering the following steps:

a) Specify a set of scale factors to be addressed, for example [128 64 32 16 8 2]
b) Calculate the CWT of the signal for the first scale factor (N = 256). Starting with a large scale-factor assures enough frequency resolution to identify all the modes
c) Calculate the dilatation parameters a_i according to the following procedure (See Appendix 0.8 for more details):

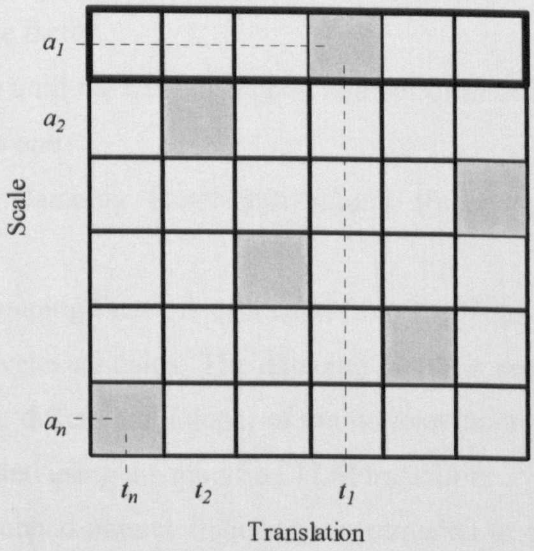


Figure O.8 Maximum CWT magnitude for each a row

i. For each row of the CWT map find the maximum value of the CWT magnitude and its position on the translation vector. (Figure O.8).
ii. Magnitudes lower than 10% of the maximum one are not considered.
iii. Verify which of the magnitudes are peak, by inspecting the preceding and following magnitudes on the corresponding translation (Figure O.9).
iv. Eliminate the scale with lower magnitude when contiguous scales are found.

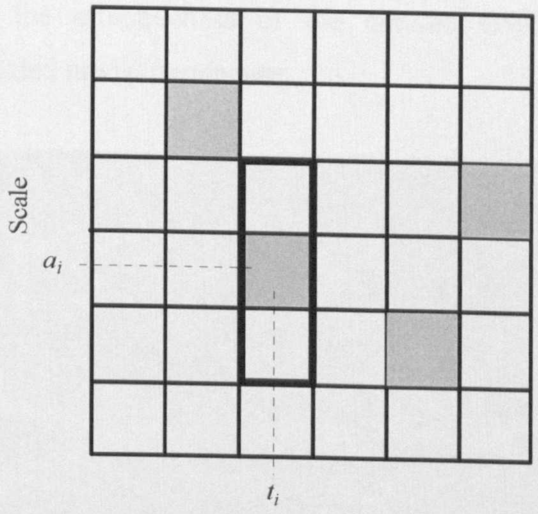


Figure O.9 CWT map fields to analyse for peak verification at the scale a_i

d) Repeat steps b) and c) for the next scale factor on the list (N = 128) and compare the calculated dilatation parameters with the ones obtained for the preceding scale factor.

e) Repeat step d) until the new calculated number of resonant frequencies differs to the previous one.

f) Calculate the damping factor and natural frequency for each dilatation parameter:

   i. The damping factor is estimated from the slope of the straight line of the wavelet modulus: The damping factor is set as the mean value of the first differential (slope) of the wavelet modulus. This differential is calculated using the modified TLM transform.

   ii. The damped natural frequency is estimated as the mean value of the first differential of the phase (equation (O.31)). The modified TLM transform is used to calculate the differential of the phase vector.

The algorithm was applied to the signal $f(t)$ described in section O.2 giving the results contained in Table O.1 (see Appendix 0.6).

	a_i	ζ		ω_d	
		Specified	By the CWT	Specified	By the CWT
First mode	18	0	2.1493e-12	5.5	5.5
Second mode	20	0	5.7388e-11	5	5

Table O.1 $f(t)$ Modal parameters identified by the CWT algorithm (N = 128)

These results show the effectiveness of the derived CWT algorithm for the identification of simulated modal parameters.

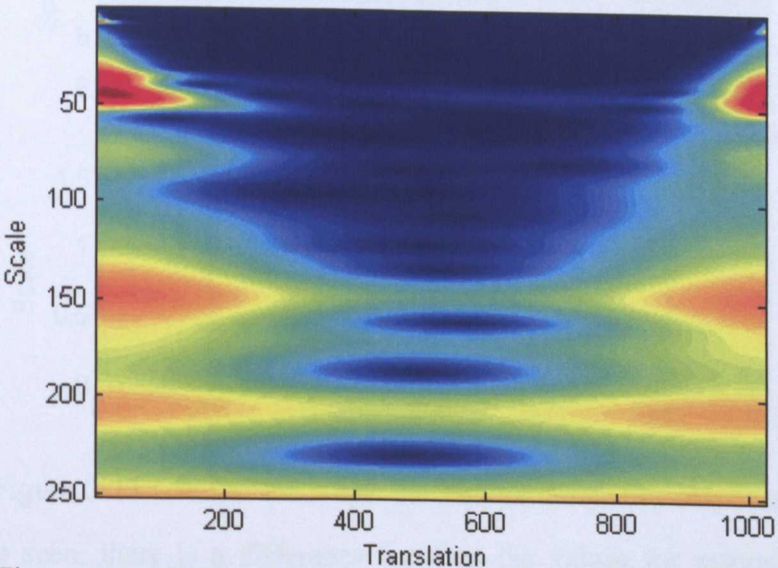


Figure O.10 CWT of the x-axis controlled frequency response ($N = 8$)

The derived CWT algorithm for the identification of modal parameters was applied to the Bode diagrams measured for the x-axis controlled frequency response (Figure 8.1) and mechanical frequency response (Figure 8.2). Table O.2 contains the values identified from the CWT of the controlled frequency, as shown in Figure O.10 (see Appendix 0.9). The damping factor and natural frequency identified for the x-axis are resumed in Table O.3 (see Appendix 0.10).

Scale (a_i)	ζ		f_d	
	Bode Diagram	Identified by the CWT	Bode Diagram	Identified by the CWT
8	0.1	0.036445	555	55O.58

Table O.2 Identified values for resonant frequencies and damping factors using Bode diagrams and wavelet analysis (x-axis controlled frequency response)

Scale (a_i)	ζ		f_d	
	Bode Diagram	Identified by the CWT	Bode Diagram	Identified by the CWT
4	0.25	0.033572	42.57	42.8478

Table O.3 Identified values for resonant frequencies and damping factors using Bode diagrams and wavelet analysis (x-axis mechanical frequency response)

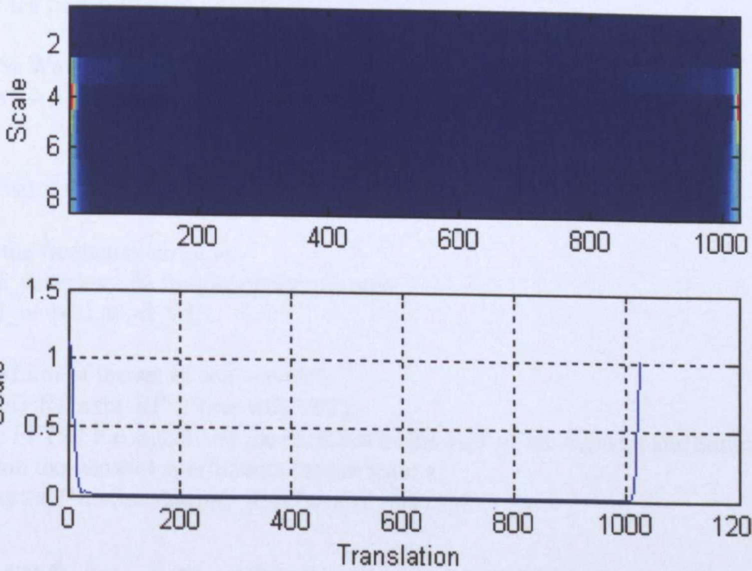


Figure O.11 CWT of the x-axis mechanical frequency response ($N = 8$)

As can be seen, there is a difference between the values for resonant frequencies identified by the wavelet algorithm and the bode diagrams. This could be explained by the fact that the CWT of the impulse response depends on the value of the resolution (N) selected for the analyses. A direction of study will be the derivation of a method for the specification of a resolution value that could lead to accurate identification results.

O.4 MATLAB Program for the Calculation of the CWT (Morlet Wavelet)

```
function W_phi=morlet_mod(f,N,a,w0)
% morlet_mod calculates the modified Morlet wavelet (W_phi) of the signal f(t)
% in frequency domain
% Author: Veimar Yobany Moreno Castañeda
% Date:  December 2005
% University of Huddersfield (U.K.)
% School of Computing and Engineering
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
% INPUTS
% f  -  Signal
% N  -  Shape parameter N >= 2
% a  -  Scales array
% w0 -  Central wavelet frequency [rad/s]
% OUTPUT
% W_phi - the CWT of the signal f
len= length(f); % Number of samples for the FFT
if mod(len,2) ~= 0
   len=len-1;  % Number of samples must be even
   f=f(1:len);
end
F_f=fft(f); % Fourier transform of f(t)
num_samples=length(F_f);
% Verify N factor:
if N < 2 % N must be >= 2
```

```
  N=2;
  display('N must be >= 2')
end
Rf=sqrt(N/2); % Wavelet shape factor
% Verify Rf*w0 > 5 (Admissibility condition for the modified Morlet wavelet)
if Rf*w0 < 5
  w0=5.1/Rf;
  display('Rf*w0 > 5')
end
% Creation of the frequency array w:
d_w=2*pi/num_samples;  % frequency sample step
w=[0:d_w:pi-d_w -pi:d_w:-d_w];
aw=a(:)*w;
% Fourier transform of the set of son wavelets
F_phi=sqrt(2*pi)*Rf*exp(-Rf^2*(aw-w0).^2/2);
% Multiply the FFT of the signal and the modulus of the FFT of the wavelet and compute the
% IFFT to obtain the wavelet coefficients for the scale a
W_phi=diag(sqrt(a))*ifft(meshgrid(F_f,a).*conj(F_phi),[],2);
```

O.5 MATLAB Program Used for the Example of CWT Calculation

```
% cwt_example_0.m
% Program used to illustrate the effect of the parameter Rf on the CWT
% calculated for the Morlet wavelet
% Author: Veimar Yobany Moreno Castañeda
% Date:  December 2005
% University of Huddersfield (U.K.)
% School of Computing and Engineering
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
t=0:0.01:10; % Time array [s]
% Decaration of two closed sinusoids:
f1=sin(2*pi*t*5);  % f1(t)
f2=sin(1.1*2*pi*t*5);  % f2(t)
f=f1+f2;  % f(t)
% Calculation of the CWT
N=2;     % Scale factor
a=0:30;   % Scales array
w0=6;     % Morlet wavelet centre frequency
wf1=morlet_mod(f1,N,a,w0); % CWT of f1(t)
wf2=morlet_mod(f2,N,a,w0); % CWT of f2(t)
wf=morlet_mod(f,N,a,w0); % CWT of f(t)
% plot signal f1 and its CTW
subplot(2,2,1)
plot(t,f1,'k') % plot f1(t)
ylabel('Amplitude'), xlabel('Time'), title('f1(t)'), axis tight
subplot(2,2,3)
imagesc(abs(wf1)) % plot CWT of f1
xlabel('Translation'), ylabel('Scale')
% plot signal f2 and its CTW
subplot(2,2,2)
plot(t,f2,'k') % plot f2(t)
ylabel('Amplitude'), xlabel('Time'), title('f2(t)'), axis tight
subplot(2,2,4)
imagesc(abs(wf2)) % plot CWT of f2
xlabel('Translation'), ylabel('Scale')
pause
% plot signal f and its CTW
subplot(2,1,1)
plot(t,f,'k'), xlabel('t'), ylabel('f(t)= f1(t) + f2(t)')
```

```
subplot(2,1,2)
%mesh(abs(wsig))
imagesc(abs(wf)), xlabel('Translation'), ylabel('Scale')
```

O.6 MATLAB Programs Used for the Example of Modal Parameters (ζ, ω_d) Identification Using the CWT

```
% cwt_ident_2ord_system.m
% Calculation of resonance frequency and damping factor for a second-order  system
% Author: Veimar Yobany Moreno Castañeda
% Date:   December 2005
% University of Huddersfield (U.K.)
% School of Computing and Engineering
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
clear all
clc
%define the transfer function
ld=16/1000; %screw lead [m]
M=350*(ld/(2*pi))^2; % mass referred to the motor shaft
C=0.1; % Bearings damping
K=3000;
wn=sqrt(K/M); % Underdamped natural frequency [rad/s]
cc=2*M*wn;  % critical damping
zita=C/cc; % Damping ratio
wr=wn*sqrt(1-zita^2); % Damped natural frequency [rad/s]
fr=wr/2/pi; % resonance frequency [Hz]
num = 1;
den = [1 2*zita*wn wn^2];
st=2e-4; % sample time
t=0:st:0.4;
y=impulse(num,den,t);
plot(t,y),grid,xlabel('time [s]'), ylabel('amplitude')
pause
scale=1:40;
scale_length=length(scale); % Length of scale
w0=2*pi; % Morlet wavelet centre frequency
N=2;
wf=morlet_mod(y,N,scale,w0); % CWT of f(t)
scale_position=find_scale(abs(wf));  % Position of the scales defining natural frequencies
disp(['Scales = ' num2str(scale_position)])
imagesc(abs(wf)),xlabel('translation'),ylabel('scale')
pause
%mesh(abs(out.wf)),xlabel('translation'),ylabel('scale')
cwt=wf(scale_position,:); % CWT for the given scale
ph=phase(cwt); % calculation of the phase of the cwt
slope=get_slope(ph,st);
plot(slope),grid,xlabel('translation'),ylabel('scale')
wd=mean(slope); % Damped natural frequency [rad/s]
fd=wd/(2*pi);
pause
 %% Calculation of the damping factor
ln_cwt=log(abs(cwt)); % the natural logarithm of the magnitude of cwt
if cov(ln_cwt) < 1e-3
   df_mean(i)=0;
else
   fun=get_slope(ln_cwt,st);
   slope=abs(mean(fun)/wd);
end
```

```
plot(fun),grid,xlabel('translation'),ylabel('slope')
disp(['ref damping factor = ' num2str(zita)])
disp(['ref Damped natural frequency [Hz]= ' num2str(fr)])

% cwt_example_00.m
% Program used to illustrate the calculation of modal parameters (dampling factor and damped
% natural frequency using the continuous wavelet transform for the modified Morlet wavelet
% Author: Veimar Yobany Moreno Castañeda
% Date:   December 2005
% University of Huddersfield (U.K.)
% School of Computing and Engineering
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
clc, clear all
st=0.01; % sample time [s]
t=0:st:10; % Time array [s]
t_len=length(t);
% Decaration of two closed sinusoids:
f1=sin(2*pi*t*5);  % f1(t)
f2=sin(1.1*2*pi*t*5);  % f2(t)
f=f1+f2;  % f(t)
N=128;
scale=0:30;
w0=6; % Morlet wavelet centre frequency
wf=morlet_mod(f,N,scale,w0); % CWT of f(t)
scale_position=find_scale(abs(wf));  % Position of the scales defining natural frequencies
disp(['Scales = ' num2str(scale_position)])
scale_length=length(scale_position);
f_d=[];%zeros(scale_length,t_len-1);
d_f=[];%zeros(scale_length,t_len-1);
for i=1:scale_length
  if scale_position(i) ~= 0
    cwt=wf(scale_position(i),:);
    %% Calculation of the damped natural frequency (eigen frequency)
       ph=phase(cwt); % calculation of the phase of the cwt
    fd=get_slope(ph,st); % Damped natural frequency [rad/s]
    for ii=1:length(fd)
      f_d(i,ii)=fd(ii);
    end
    fd_mean(i)=mean(f_d(i,:));
    %% Calculation of the damping factor
       ln_cwt=log(abs(cwt)); % the natural logarithm of the magnitude of cwt
    df_mean(i)=abs(mean(get_slope(ln_cwt,st))/fd_mean(i));
    for ii=1:length(ln_cwt)
      d_f(i,ii)=ln_cwt(ii);
    end
  end
end
subplot(2,1,1)
plot(d_f), grid, xlabel('sample'),  ylabel('ln(W(a,b)')
subplot(2,1,2)
plot(f_d'/(2*pi)), grid, xlabel('sample'), ylabel('fd [Hz]'), legend('fd1','fd2')
disp(['Damping factor = ' num2str(df_mean)])
disp(['Damped natural frequency [Hz]= ' num2str(fd_mean/(2*pi))])
```

O.7 MATLAB Program Used for the Automatic Modal Parameters (ζ, ω_d) Identification Using the CWT

```
function out=id_modal_parameters(f,st,N);
```

```
% id_modal_parameters calculates the modal parameters (damping factor and damped natural frequency)
% for the function f(t)
% Author: Veimar Yobany Moreno Castañeda
% Date:  December 2005
% University of Huddersfield (U.K.)
% School of Computing and Engineering
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
% INPUT
% f -  Signal on time domain
% st -  Signal sample time [s]
% N  -  Scale factorModified (Morlet Wavelet)
% OUTPUT
% out - Structure that containing the calculated modal parameters
% out.damping   - vector of damping factors
% out.freq      - vector of damped natural frequencies
% out.scale     - vector of scales associated to the natural frequencies
% out.wf        - CWT of the signal
% out.ph        - CWT phase
% out.N         - scale factor
sf=1/st;   % sample frequency [Hz]
scale=0:40;
scale_length=length(scale); % Length of scale
w0=6; % Morlet wavelet centre frequency
wf=morlet_mod(f,N,scale,w0); % CWT of f(t)
scale_position=find_scale(abs(wf));  % Position of the scales defining natural frequencies
disp(['Scales = ' num2str(scale_position)])
fd_mean=[];
d_f=[];
for i=1:length(scale_position)
   if scale_position(i) ~= 0
      cwt=wf(scale_position(i),:);
      %% Calculation of the damped natural frequency (eigen frequency)
         ph=phase(cwt); % calculation of the phase of the cwt
      fd_mean(i)=mean(get_slope(ph,st)); % Damped natural frequency [rad/s]
      %% Calculation of the damping factor
         ln_cwt=log(abs(cwt)); % the natural logarithm of the magnitude of cwt
      d_f(i)=abs(mean(get_slope(ln_cwt,st))/fd_mean(i));
      if d_f(i) < 1e-6
         d_f(i)=0;
      end
   end
end
out.freq=fd_mean;  % Damping factor vector
out.damping=d_f;   % Natural Frequency vector
out.scale=scale_position; % Scale vector
out.wf=wf;  % CWT
out.ph=ph;  % CWT phase
out.N=N;    % scale factor
disp(['Damping factor = ' num2str(d_f)]),
disp(['Damped natural frequency [Hz]= ' num2str(fd_mean/(2*pi))])
```

O.8 MATLAB Program Used for the Calculation of the Dilatation Parameters

```
function scale=find_scale(Wf);
% find_scale calculates the scales defining natural frequencies on a frequency/time plot
% Author: Veimar Yobany Moreno Castañeda
% Date:  December 2005
% University of Huddersfield (U.K.)
```

```
% School of Computing and Engineering
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
% INPUT
% Wf -  The magnitude of the calculated wavelet transform
% OUTPUT
% scale - Scales defining natural frequencies on Wf
% Variables declaration:
scale=[];  % Scales defining natural frequencies on Wf
[scale_length,translation_length]=size(Wf);
scale_position=zeros(1,scale_length);  % position  b of the maximum value of Wf for each scale
Wf_maximum=scale_position;             % to hold the maximum value of Wf for each scale
% Find the maximum values of the CWT magnitude and their position on the translation vector
for ii=2:scale_length
   [Wf_maximum(ii),scale_position(ii)]=max(Wf(ii,:));
end
% Magnitudes lower than 10% of the maximum one are not considered
Wf_maximum_factor=0.1*max(Wf_maximum);
for ii=2:scale_length
   if Wf_maximum(ii) < Wf_maximum_factor
     Wf_maximum(ii)=0;
  end
end
% Verify which of the Wf_maximum values are picks
k=0; % to control the number of elements of the vector scale
for iii=2:scale_length-1 % first and last scales are not analysed
   if Wf_maximum(iii)>0 &&  Wf_maximum(iii) == max(Wf(iii-1:iii+1,scale_position(iii)))
      k=k+1;
      scale(k)=iii;  % Wf_maximum(iii) is a pick
   end
end
% Eliminate the scale with lower magnitude when contiguous scales are found
if k > 1
   for i=1:k-1
      if scale(i) == scale(i+1)-1
         if Wf_maximum(scale(i)) > Wf_maximum(scale(i+1))
            scale(i+1)=0;
         else
            scale(i)=0;
         end
      end
   end
end

function slope=get_slope(f,st)
% get_slope calculates the first differential of the function f(t)
% Author: Veimar Yobany Moreno Castañeda
% Date:   December 2005
% University of Huddersfield (U.K.)
% School of Computing and Engineering
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
% INPUT
% f  -  Signal on time domain
% st -  Signal sample time [s]
% OUTPUT
% slope - vector containing the first derivative
len=length(f);
[mp,pos]=max(f);
ff=f(pos:len-pos*2);
f_dif=mtlm_dif(f,st); % Find first differential
s_dif=mtlm_dif(f_dif,st); % Find second differential
```

```
lim=5*st;
% The interval to take into account is the one for which the second derivative is approx cero
klow=2;
while abs(s_dif(klow)) > lim,
   klow=klow+1;
end
kup=klow;
while abs(s_dif(kup)) < lim,
   kup=kup+1;
   if kup == len
      break;
   end
end
slope=f_dif(klow:kup);
```

O.9 MATLAB Program Used for the identification of damping factors and resonant frequencies, x-axis mechanical frequency response

```
% ident_x_mech_freq_resp.m
% mechanical frequency response
% identification of damping factors and resonant frequencies
% Author: Veimar Yobany Moreno Castañeda
% Date:   December 2005
% University of Huddersfield (U.K.)
% School of Computing and Engineering
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
clear all, clc
load x_mech_freq_resp
Fsig=bode2fft(ph,amp);
[ph1,amp1]=fft2bode(Fsig);
y=id_modal_parameters_f(Fsig,1/125,8);
subplot(2,1,1)
imagesc(abs(y.wf)), ylabel('Scale')
subplot(2,1,2)
plot(abs(y.wf(y.scale,:))), grid, xlabel('Translation'), ylabel('Scale'), subplot(111)
```

O.10 MATLAB Program Used for the identification of damping factors and resonant frequencies, x-axis control system frequency response

```
% ident_x_cont_syst_freq_resp.m
% control system frequency response (velocity)
% identification of damping factors and resonant frequencies
% Author: Veimar Yobany Moreno Castañeda
% Date:   December 2005
% University of Huddersfield (U.K.)
% School of Computing and Engineering
% This is a Copyrighted material, for copying permissions send email to m_veimar@hotmail.com
clear all, clc
load x_cont_syst_freq_resp_nfilt
Fsig=bode2fft(ph,amp);
[ph1,amp1]=fft2bode(Fsig);
y=id_modal_parameters_f(Fsig,1/4000,8);
imagesc(abs(y.wf))
xlabel('Translation'), ylabel('Scale')
```