

KALMAN FILTER NON-LINEAR CONTROL LAW
FOR A GYROSCOPE PULSE WIDTH MODULATION
TORQUE LOOP

by

Paul Allan DeBitetto

B.E.M.E., S.U.N.Y. at Stonybrook
(1986)

SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN
AERONAUTICS AND ASTRONAUTICS

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 1989

© Paul Allan DeBitetto, 1989

Signature of Author _____
Department of Aeronautics and Astronautics
May 6, 1989

Certified by _____
Professor Harold L. Alexander
Department of Aeronautics and Astronautics
Thesis Supervisor

Certified by _____
Michael E. Ash
Technical Supervisor, CSDL

Accepted by _____
Professor Harold Y. Wachman
Chairman, Departmental Graduate Committee

Aero

MASSACHUSETTS INSTITUTE
OF TECHNOLOGY

JUN 07 1989

AUG 6 '91

LIBRARIES

1

WITHDRAWN
M.I.T.
LIBRARIES

KALMAN FILTER NON-LINEAR CONTROL LAW FOR A GYROSCOPE PULSE WIDTH MODULATION TORQUE LOOP

by

Paul Allan DeBitetto

Submitted to the Department of Aeronautics and Astronautics
on May 12, 1989 in partial fulfillment of the
requirements for the Degree of Master of Science
in Aeronautics and Astronautics

ABSTRACT

This paper explores the optimization of a pulse width modulation torque loop for a floated gyroscope. Kalman filters are designed to estimate the float angular position and rate, external input rate, and the time derivative of the external input rate. Kalman filter state estimates are used in control algorithms derived from analytic solutions of the gyro dynamic equations. The performance of Kalman filter based controllers is compared to the performance of a simplified controller, which was derived based only on the recent history of average float rotation angle and commanded torque.

A pulse width modulation torquing scheme is employed for better resolution and reduced torquer non-linearities, than can be found in simple binary torquing schemes. While most previous torque loops have used analog control circuits, we propose implementing this method using microprocessors. Analysis of this non-linear microprocessor based pulse width modulation torquing scheme is carried out in the time domain. This was selected over the classical frequency domain analysis, which uses describing function approximations of non-linear control behavior.

A computer program simulates the derived algorithms and evaluates their closed loop performance in response to various inputs and different levels of measurement noise. Time domain plots of the simulated torque loop behavior are presented. In these simulations, the derived algorithms succeed in keeping the float at null with negligible steady state float hang-off. In each case, there is no moding of the torquing pattern, and with properly banded measurement noise, the float maintains its minimum mode of oscillation. At the simulated frequencies, excellent transient performance was observed for step, ramp, and sinusoidal inputs.

Fast Kalman filter updates of input-axis rate estimates provides an accurate, high-bandwidth alternative to counting commanded torque pulses for determination of external input rates.

Thesis Supervisor: Dr. Harold L. Alexander

**Title: Charles Stark Draper Assistant Professor of Aeronautics and
Astronautics**

Technical Supervisor: Dr. Michael E. Ash

Title: Technical Staff, The Charles Stark Draper Laboratory, Inc.

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to all those who have made this thesis possible and my educational experience at M.I.T. and Draper Laboratory a rewarding one.

Special thanks to:

My technical supervisor Dr. Michael E. Ash for all his technical assistance during the research and writing of this thesis. I also want to thank him for his genuine concern for my educational development; a true educator of the highest degree.

My thesis advisor Dr. Harold L. Alexander for his guidance and assistance in the preparation of this thesis.

Paul R. Kerrigan for providing me the opportunity to do my graduate work at Draper Laboratory.

Lisa E. Rosch for her gracious assistance in the preparation of this thesis.

I also want to thank members of the Draper Laboratory technical staff, David A. Moschella, John A. Hengeveld, Paul C. L. Leung, Charles R. Dauwalter, Marco J. Marchi for their technical advice and support.

Finally, I want to thank Laurie E. Chidester and my family for all the love and support they have given to me during my graduate studies.

This report was prepared at The Charles Stark Draper Laboratory, Inc. under Air Force Contract F04704-88-C-0100.

Publication of this report does not constitute approval by the Draper Laboratory or the sponsoring agency of the findings or conclusions contained herein. It is published for the exchange and stimulation of ideas.

I hereby assign my copyright of this thesis to The Charles Stark Draper Laboratory, Inc., Cambridge, Massachusetts.


Paul A. DeBitetto

Permission is hereby granted by The Charles Stark Draper Laboratory, Inc. to the Massachusetts Institute of Technology to reproduce any or all of this thesis.

TABLE OF CONTENTS

SECTION	PAGE
LIST OF FIGURES	11
LIST OF TABLES	29
PART I. INTRODUCTION AND BACKGROUND	
1. INTRODUCTION	30
2. USE OF ELECTROMAGNETIC TORQUING IN INERTIAL INSTRUMENTS.....	34
2.1 Single Degree of Freedom Floated Gyroscope	34
2.2 Use of Gyroscope Torquing	35
3. SIGNAL AND TORQUE GENERATORS.....	37
3.1 Signal Generator (SG).....	37
3.1.1 Readout Techniques.....	38
3.1.2 Readout Accuracy	38
3.2 Permanent Magnet Torque Generator (TG).....	39
4. PREVIOUS TYPES OF TORQUE LOOPS	42
4.1 Analog Loops.....	42
4.2 Pulse Torquing Loops	43
4.2.1 Binary Torquing	44
4.2.2 Ternary Torquing	46

TABLE OF CONTENTS (CONT.)

SECTION	PAGE
4.2.3 Pulse Width Modulation (PWM) Torquing.....	46
4.3 Torque Loop Analysis	49
4.4 Torque Loop Implementation Methods	51
5. PROPOSED PULSE WIDTH MODULATION TORQUE LOOP	53
5.1 Components and Timing	53
5.2 SG Prefiltering Computations	56
5.3 Torque Loop Control Computations	58

PART II. THEORY

6. TORQUE SUMMING MEMBER DYNAMIC EQUATIONS	59
6.1 Dynamic Differential Equations	59
6.2 Analytic Form of Commanded Torque	61
6.3 Runge-Kutta Numerical Integration	64
7. ESTIMATOR DESIGN	65
7.1 Introduction	65
7.2 Linear Discrete Time Systems	65
7.2.1 State Propagation	65
7.2.2 Transition Matrix Computation	68
7.2.2.1 Transition Matrix for Time-Invariant Systems	68
7.2.2.2 Transition Matrix for General Systems	69

TABLE OF CONTENTS (CONT.)

SECTION	PAGE
7.3 The Observer	70
7.3.1 General Form	70
7.3.2 Gain Matrix by Optimal Estimation	72
7.3.2.1 The Stochastic Model	72
7.3.2.2 The Covariance Matrix	73
7.3.2.3 Kalman Filter Update Procedure	74
7.3.3 Gain Matrix by Direct Pole-Placement	77
7.4 Torque Loop Estimator Equations	79
7.4.1 Torque Loop State Equations	79
7.4.2 Torque Loop Estimator Gains	83
7.4.2.1 Estimator Gains by Optimal Estimation	83
7.4.2.2 Estimator Gains by Direct Pole Placement	86
 8. NON-LINEAR CONTROLLER ALGORITHMS	 93
8.1 Introduction	93
8.2 General Description of Control Algorithms Based on Kalman Filter State Estimates	 94
8.2.1 Introduction	94
8.2.2 Position and Velocity States to be Sought	94
8.2.3 Torque Pulses to Reach Desired States	95
8.3 Three State Kalman Filter Control Algorithm	95
8.3.1 Analytic Solution of Dynamic Equations	95

TABLE OF CONTENTS (CONT.)

SECTION	PAGE
8.3.2 Position and Velocity States to be Sought	99
8.3.3 Torque Pulses to Reach Desired States	100
8.3.4 Newton Iterative Solution	102
8.4 Four State Kalman Filter Algorithm	103
8.4.1 Analytic Solution of Dynamic Equations	103
8.4.2 Position and Velocity States to be Sought.....	106
8.4.3 Torque Pulses to Reach Desired States	106
8.5 Simplified Controller	107
8.5.1 Introduction	107
8.5.2 Extrapolation of Average OA Angle Observable	107
8.5.3 Additional Torque Required to Force the Average OA Angle to Zero	108
8.5.4 Quantized Simplified Control Algorithm	109
9. DESCRIPTION OF SIMULATION SOFTWARE	111
9.1 Macintosh Software	111
9.2 IBM Mainframe Software	111
9.2.1 Executive Control	113
9.2.2 KALSIM Control Loops	114
9.2.3 INTGRT State and Transition Matrix Propagation	115
9.2.4 Kalman Filter Propagation	115
9.2.5 TRQCMD Controller Decisions	115
9.2.6 PLOT Options	115

TABLE OF CONTENTS (CONT.)

SECTION	PAGE
PART III. SIMULATION RESULTS	
10. DESCRIPTION OF SIMULATION SCENARIOS	116
11. STEADY STATE BEHAVIOR	118
11.1 Zero Input Response	118
11.1.1 Results Summary	118
11.1.2 Simulation Plots	119
11.2 Non Zero Constant Input.....	119
11.2.1 Results Summary	119
11.2.2 Simulation Plots	126
12. STEP INPUT RESPONSE	132
12.1 Results Summary	132
12.2 Simulation Plots	134
13. RAMP INPUT RESPONSE	162
13.1 Results Summary	162
13.2 Simulation Plots	163
14. SINUSOIDAL INPUT RESPONSE	191
14.1 Results Summary	191
14.2 Simulation Plots	192
15. RESPONSE TO SIGNAL GENERATOR NOISE	220
15.1 Results Summary	220

TABLE OF CONTENTS (CONT.)

SECTION	PAGE
15.2 Simulation Plots	221

PART IV. CONCLUSION

16. SUMMARY	255
17. IMPLEMENTATION AND RECOMMENDATIONS.....	257

APPENDICES

A. DERIVATION OF CONTINUOUS TO DISCRETE-TIME DIFFERENCE EQUATIONS USING VARIATIONS OF PARAMETERS METHOD.....	260
B. CONTINUOUS TO DISCRETE-TIME EQUATIONS TAKING INTO ACCOUNT PROCESSOR DELAY	263
C. EQUATIONS FOR CONTROL ALGORITHM DERIVATION	265
REFERENCES	271

LIST OF FIGURES

<u>FIGURE</u>	<u>PAGE</u>
2-1. Single Degree of Freedom Floated Gyroscope Coordinate Axes	34
3-1. E-Type Microsyn Signal Generator [1]	37
3-2. Section of Permanent Magnet Torque Generator [2]	40
3-3. Fully Assembled Permanent Magnet Torque Generator [2]	41
4-1. Analog Torque Loop Block Diagram [4]	42
4-2. Binary Pulse Torque Loop Block Diagram [4]	44
4-3. Relay Used in Ternary Pulse Torquing [4]	46
4-4. Pulse Width Modulation [7]	47
4-5. Two Possible Feedback Responses for Zero Input	48
4-6. Pulse Width Modulation with Analog Circuitry [4]	49
5-1. Proposed PWM Torque Loop Block Diagram	53
5-2. Torque Loop Events Time Line	56
6-1. Pulse Width Modulation Pulse Shape for Numerical Integration Simulations	62
7-1. Illustration of Transition Matrix Propagating the State Vector Through Time [16]	67
7-2. Closed Loop Estimator [17]	71
7-3. Flow Chart of Kalman Filter Update Procedure	76
7-4. Estimator Gain on External Torque vs. Time for $Q(4,4) = 960 \text{ dyne-cm/s}^2$...	87
7-5. Estimator Gain on External Torque vs. Time for $Q(4,4) = 9,600 \text{ dyne-cm/s}^2$...	87
7-6. Error Transient Response for "Fast" Estimator Gains.....	90
7-7. Error Transient Response for "Medium" Estimator Gains.....	92
8-1. Pulse Width Modulation Square Wave Pulse Shape for Deriving Control Laws	97

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
8-2. Two Switching Instants of Commanded Torque to Send Position and Velocity States to Desired Values	100
9-1. Flow Chart of IBM 3090 Mainframe Computer Simulation Software	112
11-1. Run #1 Float Position vs. Time (Zero Input).....	120
11-2. Run #1 Float Velocity vs. Time (Zero Input).....	120
11-3. Run #1 Commanded Torque vs. Time (Zero Input).....	121
11-4. Run #1 24 Pt. Average Position vs. Time (Zero Input).....	122
11-5. Run #1 24 Pt. Average Velocity vs. Time (Zero Input).....	122
11-6. Run #2 Float Position vs. Time (Zero Input).....	123
11-7. Run #2 Float Velocity vs. Time (Zero Input).....	123
11-8. Run #2 Commanded Torque vs. Time (Zero Input).....	124
11-9. Run #2 24 Pt. Average Position vs. Time (Zero Input).....	125
11-10. Run #2 24 Pt. Average Velocity vs. Time (Zero Input).....	125
11-11. Run #3 Float Position vs. Time (Constant 7.5°/hr Input).....	127
11-12. Run #3 Float Velocity vs. Time (Constant 7.5°/hr Input).....	127
11-13. Run #3 Commanded Torque vs. Time (Constant 7.5°/hr Input).....	128
11-14. Run #3 24 Pt. Average Position vs. Time (Constant 7.5°/hr Input).....	128
11-15. Run #3 24 Pt. Average Velocity vs. Time (Constant 7.5°/hr Input).....	129
11-16. Run #4 Float Position vs. Time (Constant 7.5°/hr Input).....	129
11-17. Run #4 Float Velocity vs. Time (Constant 7.5°/hr Input).....	130
11-18. Run #4 Commanded Torque vs. Time (Constant 7.5°/hr Input).....	130

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
11-19. Run #4 24 Pt. Average Position vs. Time (Constant 7.5°/hr Input).....	131
11-20. Run #4 24 Pt. Average Velocity vs. Time (Constant 7.5°/hr Input).....	131
12-1. Run #5 Float Position vs. Time (15°/hr Step Input).....	135
12-2. Run #5 Float Velocity vs. Time (15°/hr Step Input).....	135
12-3. Run #5 Commanded Torque vs. Time (15°/hr Step Input).....	136
12-4. Run #5 Position Estimation Error vs. Time (15°/hr Step Input).....	136
12-5. Run #5 Velocity Estimation Error vs. Time (15°/hr Step Input).....	137
12-6. Run #5 External Torque Estimation Error vs. Time (15°/hr Step Input).....	137
12-7. Run #5 24 Pt. Average Position vs. Time (15°/hr Step Input).....	138
12-8. Run #5 24 Pt. Average Velocity vs. Time (15°/hr Step Input).....	138
12-9. Run #6 Float Position vs. Time (15°/hr Step Input).....	139
12-10. Run #6 Float Velocity vs. Time (15°/hr Step Input).....	139
12-11. Run #6 Commanded Torque vs. Time (15°/hr Step Input).....	140
12-12. Run #6 Position Estimation Error vs. Time (15°/hr Step Input).....	140
12-13. Run #6 Velocity Estimation Error vs. Time (15°/hr Step Input).....	141
12-14. Run #6 External Torque Estimation Error vs. Time (15°/hr Step Input).....	141
12-15. Run #6 24 Pt. Average Position vs. Time (15°/hr Step Input).....	142
12-16. Run #6 24 Pt. Average Velocity vs. Time (15°/hr Step Input).....	142
12-17. Run #7 Float Position vs. Time (15°/hr Step Input).....	143
12-18. Run #7 Float Velocity vs. Time (15°/hr Step Input).....	143
12-19. Run #7 Commanded Torque vs. Time (15°/hr Step Input).....	144
12-20. Run #7 Position Estimation Error vs. Time (15°/hr Step Input).....	144

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
12-21. Run #7 Velocity Estimation Error vs. Time (15°/hr Step Input).....	145
12-22. Run #7 External Torque Estimation Error vs. Time (15°/hr Step Input).....	145
12-23. Run #7 24 Pt. Average Position vs. Time (15°/hr Step Input).....	146
12-24. Run #7 24 Pt. Average Velocity vs. Time (15°/hr Step Input).....	146
12-25. Run #8 Float Position vs. Time (15°/hr Step Input).....	147
12-26. Run #8 Float Velocity vs. Time (15°/hr Step Input).....	147
12-27. Run #8 Commanded Torque vs. Time (15°/hr Step Input).....	148
12-28. Run #8 Position Estimation Error vs. Time (15°/hr Step Input).....	148
12-29. Run #8 Velocity Estimation Error vs. Time (15°/hr Step Input).....	149
12-30. Run #8 External Torque Estimation Error vs. Time (15°/hr Step Input).....	149
12-31. Run #8 24 Pt. Average Position vs. Time (15°/hr Step Input).....	150
12-32. Run #8 24 Pt. Average Velocity vs. Time (15°/hr Step Input).....	150
12-33. Run #9 Float Position vs. Time (15°/hr Step Input).....	151
12-34. Run #9 Float Velocity vs. Time (15°/hr Step Input).....	151
12-35. Run #9 Commanded Torque vs. Time (15°/hr Step Input).....	152
12-36. Run #9 Position Estimation Error vs. Time (15°/hr Step Input).....	152
12-37. Run #9 Velocity Estimation Error vs. Time (15°/hr Step Input).....	153
12-38. Run #9 External Torque Estimation Error vs. Time (15°/hr Step Input).....	153
12-39. Run #9 24 Pt. Average Position vs. Time (15°/hr Step Input).....	154
12-40. Run #9 24 Pt. Average Velocity vs. Time (15°/hr Step Input).....	154
12-41. Run #10 Float Position vs. Time (15°/hr Step Input).....	155
12-42. Run #10 Float Velocity vs. Time (15°/hr Step Input).....	155

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
12-43. Run #10 Commanded Torque vs. Time (15°/hr Step Input).....	156
12-44. Run #10 Position Estimation Error vs. Time (15°/hr Step Input).....	156
12-45. Run #10 Velocity Estimation Error vs. Time (15°/hr Step Input).....	157
12-46. Run #10 External Torque Estimation Error vs. Time (15°/hr Step Input)....	157
12-47. Run #10 24 Pt. Average Position vs. Time (15°/hr Step Input).....	158
12-48. Run #10 24 Pt. Average Velocity vs. Time (15°/hr Step Input).....	158
12-49. Run #11 Float Position vs. Time (15°/hr Step Input).....	159
12-50. Run #11 Float Velocity vs. Time (15°/hr Step Input).....	159
12-51. Run #11 Commanded Torque vs. Time (15°/hr Step Input).....	160
12-52. Run #11 24 Pt. Average Position vs. Time (15°/hr Step Input).....	160
12-53. Run #11 24 Pt. Average Velocity vs. Time (15°/hr Step Input).....	161
13-1. Run #12 Float Position vs. Time (282°/hr/s Ramp Input).....	164
13-2. Run #12 Float Velocity vs. Time (282°/hr/s Ramp Input).....	164
13-3. Run #12 Commanded Torque vs. Time (282°/hr/s Ramp Input).....	165
13-4. Run #12 Position Estimation Error vs. Time (282°/hr/s Ramp Input).....	165
13-5. Run #12 Velocity Estimation Error vs. Time (282°/hr/s Ramp Input).....	166
13-6. Run #12 External Torque Estimation Error vs. Time (282°/hr/s Ramp Input).166	166
13-7. Run #12 24 Pt. Average Position vs. Time (282°/hr/s Ramp Input).....	167
13-8. Run #12 24 Pt. Average Velocity vs. Time (282°/hr/s Ramp Input).....	167
13-9. Run #13 Float Position vs. Time (282°/hr/s Ramp Input).....	168
13-10. Run #13 Float Velocity vs Time (282°/hr/s Ramp Input).....	168
13-11. Run #13 Commanded Torque vs. Time (282°/hr/s Ramp Input).....	169

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
13-12. Run #13 Position Estimation Error vs. Time (282°/hr/s Ramp Input).....	169
13-13. Run #13 Velocity Estimation Error vs. Time (282°/hr/s Ramp Input).....	170
13-14. Run #13 External Torque Estimation Error vs. Time (282°/hr/s Ramp Input).	170
13-15. Run #13 24 Pt. Average Position vs. Time (282°/hr/s Ramp Input).....	171
13-16. Run #13 24 Pt. Average Velocity vs Time (282°/hr/s Ramp Input).....	171
13-17. Run #14 Float Position vs. Time (282°/hr/s Ramp Input).....	172
13-18. Run #14 Float Velocity vs. Time (282°/hr/s Ramp Input).....	172
13-19. Run #14 Commanded Torque vs. Time (282°/hr/s Ramp Input).....	173
13-20. Run #14 Position Estimation Error vs. Time (282°/hr/s Ramp Input).....	173
13-21. Run #14 Velocity Estimation Error vs. Time (282°/hr/s Ramp Input).....	174
13-22. Run #14 External Torque Estimation Error vs. Time (282°/hr/s Ramp Input).	174
13-23. Run #14 24 Pt. Average Position vs. Time (282°/hr/s Ramp Input).....	175
13-24. Run #14 24 Pt. Average Velocity vs. Time (282°/hr/s Ramp Input).....	175
13-25. Run #15 Float Position vs. Time (282°/hr/s Ramp Input).....	176
13-26. Run #15 Float Velocity vs. Time (282°/hr/s Ramp Input).....	176
13-27. Run #15 Commanded Torque vs. Time (282°/hr/s Ramp Input).....	177
13-28. Run #15 Position Estimation Error vs. Time (282°/hr/s Ramp Input).....	178
13-29. Run #15 Velocity Estimation Error vs. Time (282°/hr/s Ramp Input).....	178
13-30. Run #15 External Torque Estimation Error vs. Time (282°/hr/s Ramp Input).	179
13-31. Run #15 24 Pt. Average Position vs. Time (282°/hr/s Ramp Input).....	179
13-32. Run #15 24 Pt. Average Velocity vs. Time (282°/hr/s Ramp Input).....	180
13-33. Run #16 Float Position vs. Time (282°/hr/s Ramp Input).....	180

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
13-34. Run #16 Float Velocity vs. Time (282°/hr/s Ramp Input).....	181
13-35. Run #16 Commanded Torque vs. Time (282°/hr/s Ramp Input).....	181
13-36. Run #16 Position Estimation Error vs. Time (282°/hr/s Ramp Input).....	181
13-37. Run #16 Velocity Estimation Error vs. Time (282°/hr/s Ramp Input).....	182
13-38. Run #16 External Torque Estimation Error vs. Time (282°/hr/s Ramp Input).....	182
13-39. Run #16 24 Pt. Average Position vs. Time (282°/hr/s Ramp Input).....	183
13-40. Run #16 24 Pt. Average Velocity vs. Time (282°/hr/s Ramp Input).....	183
13-41. Run #17 Float Position vs. Time (282°/hr/s Ramp Input).....	184
13-42. Run #17 Float Velocity vs. Time (282°/hr/s Ramp Input).....	184
13-43. Run #17 Commanded Torque vs. Time (282°/hr/s Ramp Input).....	185
13-44. Run #17 Position Estimation Error vs. Time (282°/hr/s Ramp Input).....	185
13-45. Run #17 Velocity Estimation Error vs. Time (282°/hr/s Ramp Input).....	186
13-46. Run #17 External Torque Estimation Error vs. Time (282°/hr/s Ramp Input).....	186
13-47. Run #17 24 Pt. Average Position vs. Time (282°/hr/s Ramp Input).....	187
13-48. Run #17 24 Pt. Average Velocity vs. Time (282°/hr/s Ramp Input).....	187
13-49. Run #18 Float Position vs. Time (282°/hr/s Ramp Input).....	188
13-50. Run #18 Float Velocity vs. Time (282°/hr/s Ramp Input).....	188
13-51. Run #18 Commanded Torque vs. Time (282°/hr/s Ramp Input).....	189
13-52. Run #18 24 Pt. Average Position vs. Time (282°/hr/s Ramp Input).....	189

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
13-53. Run #18 24 Pt. Average Velocity vs. Time (282°/hr/s Ramp Input).....	190
14-1. Run #19 Float Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	193
14-2. Run #19 Float Velocity vs Time (10 Hz, 11.83°/hr Sinusoidal Input).....	193
14-3. Run #19 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	194
14-4. Run #19 Position Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	194
14-5. Run #19 Velocity Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	195
14-6. Run #19 External Torque Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	195
14-7. Run #19 24 Pt. Average Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	196
14-8. Run #19 24 Pt. Average Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	196
14-9. Run #20 Float Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	197
14-10. Run #20 Float Velocity vs Time (10 Hz, 11.83°/hr Sinusoidal Input).....	197
14-11. Run #20 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).	198
14-12. Run #20 Position Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	198

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
14-13. Run #20 Velocity Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	199
14-14. Run #20 External Torque Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	199
14-15. Run #20 24 Pt. Average Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	200
14-16. Run #20 24 Pt. Average Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	200
14-17. Run #21 Float Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	201
14-18. Run #21 Float Velocity vs Time (10 Hz, 11.83°/hr Sinusoidal Input).....	201
14-19. Run #21 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	202
14-20. Run #21 Position Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	202
14-21. Run #21 Velocity Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal input).....	203
14-22. Run #21 External Torque Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	203
14-23. Run #21 24 Pt. Average Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	204

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
14-24. Run #21 24 Pt. Average Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	204
14-25. Run #22 Float Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	205
14-26. Run #22 Float Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	205
14-27. Run #22 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).	206
14-28. Run #22 Position Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	206
14-29. Run #22 Velocity Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	207
14-30. Run #22 External Torque Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	207
14-31. Run #22 24 Pt. Average Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	208
14-32. Run #22 24 Pt. Average Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	208
14-33. Run #23 Float Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	209
14-34. Run #23 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).	209
14-35. Run #23 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).	210
14-36. Run #23 Position Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	210

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
14-37. Run #23 Velocity Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	211
14-38. Run #23 External Torque Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	211
14-39. Run #23 24 Pt. Average Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	212
14-40. Run #23 24 Pt. Average Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	212
14-41. Run #24 Float Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	213
14-42. Run #24 Float Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	213
14-43. Run #24 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal)	214
14-44. Run #24 Position Estimation Error vs. Time (10 Hz, 11.83°/hr Input).Sinusoidal Input).....	214
14-45. Run #24 Velocity Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	215
14-46. Run #24 External Torque Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	215
14-47. Run #24 24 Pt. Average Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	216

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
14-48. Run #24 24 Pt. Average Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	216
14-49. Run #25 Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	217
14-50. Run #25 Float Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	217
14-51. Run #25 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	218
14-52. Run #25 24 Pt. Average Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input).....	218
14-53. Run #25 24 Pt. Average Velocity vs Time (10 Hz, 11.83°/hr Sinusoidal Input).....	219
15-1. Run #26 Float Position vs. Time (7.5°/hr with S.G. Noise Level=1 nrad).....	222
15-2. Run #26 Float Velocity vs. Time (7.5°/hr with S.G. Noise Level=1 nrad).....	222
15-3. Run #26 Commanded Torque vs. Time (7.5°/hr with S.G. Noise Level=1 nrad).....	223
15-4. Run #26 Position Estimation Error vs. Time (7.5°/hr with S.G. Noise Level=1 nrad).....	223
15-5. Run #26 Velocity Estimation Error vs. Time (7.5°/hr with S.G. Noise Level=1 nrad).....	224

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
15-6. Run #26 External Torque Estimation Error vs. Time (7.5°/hr with S.G. Noise Level=1 nrad).....	224
15-7. Run #26 24 Pt. Average Position vs. Time (7.5°/hr with S.G. Noise Level=1 nrad).....	225
15-8. Run #26 24 Pt. Average Velocity vs. Time (7.5°/hr with S.G. Noise Level=1 nrad).....	225
15-9. Run #27 Float Position vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	226
15-10. Run #27 Float Velocity vs Time (7.5°/hr with SG Noise Level=1 nrad).....	226
15-11. Run #27 Commanded Torque vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	227
15-12. Run #27 Position Estimation Error vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	227
15-13. Run #27 Velocity Estimation Error vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	228
15-14. Run #27 External Torque Estimation Error vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	228
15-15. Run #27 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	229

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
15-16. Run #27 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	229
15-17. Run #28 Float Position vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	230
15-18. Run #28 Float Velocity vs Time (7.5°/hr with SG Noise Level=10 nrad).....	230
15-19. Run #28 Commanded Torque vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	231
15-20. Run #28 Position Estimation Error vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	231
15-21. Run #28 Velocity Estimation Error vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	232
15-22. Run #28 External Torque Estimation Error vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	232
15-23. Run #28 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	233
15-24. Run #28 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	233
15-25. Run #29 Float Position vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	234

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
15-26. Run #29 Float Velocity vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	234
15-27. Run #29 Commanded Torque vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	235
15-28. Run #29 Position Estimation Error vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	235
15-29. Run #29 Velocity Estimation Error vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	236
15-30. Run #29 External Torque Estimation Error vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	236
15-31. Run #29 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	237
15-32. Run #29 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	237
15-33. Run #30 Float Position vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	238
15-34. Run #30 Float Velocity vs Time (7.5°/hr with SG Noise Level=10 nrad).....	238
15-35. Run #30 Commanded Torque vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	239

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
15-36. Run #30 Position Estimation Error vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	239
15-37. Run #30 Velocity Estimation Error vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	240
15-38. Run #30 External Torque Estimation Error vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	240
15-39. Run #30 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	241
15-40. Run #30 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	241
15-41. Run #31 Float Position vs. Time (7.5°/hr with SG Noise Level=100 nrad)..	242
15-42. Run #31 Float Velocity vs. Time (7.5°/hr with SG Noise Level=100 nrad)..	242
15-43. Run #31 Commanded Torque vs. Time (7.5°/hr with SG Noise Level=100 nrad).....	243
15-44. Run #31 Position Estimation Error vs. Time (7.5°/hr with SG Noise Level=100 nrad).....	243
15-45. Run #31 Velocity Estimation Error vs. Time (7.5°/hr with SG Noise Level=100 nrad).....	244
15-46. Run #31 External Torque Estimation Error vs. Time (7.5°/hr with SG Noise Level=100 nrad).....	244

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
15-47. Run #31 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=100 nrad).....	245
15-48. Run #31 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=100 nrad).....	245
15-49. Run #32 Float Position vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	246
15-50. Run #32 Float Velocity vs Time (7.5°/hr with SG Noise Level=1 nrad).....	246
15-51. Run #32 Commanded Torque vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	247
15-52. Run #32 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	247
15-53. Run #32 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=1 nrad).....	248
15-54. Run #33 Float Position vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	249
15-55. Run #33 Float Velocity vs Time (7.5°/hr with SG Noise Level=10 nrad).....	249
15-56. Run #33 Commanded Torque vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	250

LIST OF FIGURES (CONT.)

<u>FIGURE</u>	<u>PAGE</u>
15-57. Run #33 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	250
15-58. Run #33 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=10 nrad).....	251
15-59. Run #34 Float Position vs. Time (7.5°/hr with SG Noise Level=100 nrad).....	252
15-60. Run #34 Float Velocity vs. Time (7.5°/hr with SG Noise Level=100 nrad).....	252
15-61. Run #34 Commanded Torque vs. Time (7.5°/hr with SG Noise Level=100 nrad).....	253
15-62. Run #34 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=100 nrad).....	253
15-63. Run #34 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=100 nrad).....	254

LIST OF TABLES

<u>TABLE</u>	<u>PAGE</u>
5-1. Pulse Width Modulation Torque Loop Frequencies	54
7-1. "Fast", "Medium", and "Slow" Estimation Gains Used in Torque Loop Simulations	91
11-1. Summary of Results for Input Rate of $0^\circ/\text{hr}=0$ dyne-cm	118
11-2. Results Summary for Constant Input of $7.5^\circ/\text{hr}=3.742$ dyne-cm	119
12-1. Results Summary for Step input from $0-15^\circ/\text{hr}=0-7.58$ dyne-cm	132
13-1. Results Summary for Ramp Input of $282^\circ/\text{hr/s}=142.9$ dyne-cm/s	162
14-1. Results Summary for Sinusoidal Input; Freq.=10 Hz, Amp= $11.83^\circ/\text{hr}=5.99$ dyne-cm	191
15-1. Results Summary for Constant Input of $7.5^\circ/\text{hr}=3.742$ dyne-cm with SG Noise	220

1. INTRODUCTION

Pulse torquing gyroscope rebalance loops have been in use since the late 1950's using analog control circuitry. These control systems represent an intentional use of non-linear control to improve performance. However, a great deal of effort has been expended in trying to analyze and design pulse torque loops as linear systems, with the non-linearity of pulse torquing replaced with linear approximations or neglected all together. In order for rebalance loops to be analyzable with such techniques, they have generally been limited in complexity and sophistication. This thesis represents a back-to-basics approach to designing and analyzing a pulse torque loop and proposes the use of a pulse width modulated torque loop rather than the simpler binary or ternary pulse torque loops. Further, Kalman filtering is used to improve performance and rate estimation. However, in order for it to be realizable, full use of modern microprocessor technology is necessary.

Torque loops are used in a variety of gyroscopes and pulse rebalance accelerometers. The design, analysis, and simulations undertaken in this thesis are for the specific case of a single-degree-of-freedom floated gyroscope (see Section 2.2), although the design arrived at and simulation results obtained could be adapted to the other types of inertial devices as well.

The approach taken in this thesis is to base the torque control law on Kalman filter estimates of the output axis angular position and velocity of the float in a floated gyroscope, along with the estimate of the angular velocity inputs to the gyroscope. Also investigated are a Kalman filter based control law with an added state for estimating the rate of change of external inputs, and a simplified control algorithm not using a Kalman filter.

The only measurement available to the control system is the float angular position about the output axis. Using the estimates based on this observable quantity, a torque control decision is made using an analytic solution to the gyro dynamics, taking into account the non-linearity of pulse width modulated torquing.

The Kalman filter and control decision algorithms would have to be implemented on a microprocessor. The Kalman filter must operate at a high enough speed to accurately follow the gyro dynamic environment. In addition, the control law computations must be performed fast enough so the control system yields acceptable closed loop responses over a specified bandwidth of input frequencies.

It has been said that computer technology improves a factor of ten every four years (speed x memory size). Many of the same reasons for the increasingly wide-spread use of microprocessors such as speed, power, cost, and size, can be reiterated here for pulse torquing rebalance loops. The high speed capability of microprocessors (approximately 10 million instructions per second at present and increasing) results in the ability to sample at high rates and implement control algorithms with sufficient speed to insure acceptable closed loop bandwidths.

Electrical power requirements to operate a microprocessor controlled torque loop are significantly less than the power required by conventional torque loops constructed from analog circuitry. Also, cost to put into operation as well as cost to maintain a microprocessor system would be reduced. Physical size is a critical factor for control systems being integrated as part of an inertial guidance system, to be installed in a vehicle. A great deal of torque loop electronics can be replaced by microprocessor software, thereby also significantly reducing the physical space required. In addition, the microprocessor system would be more easily adaptable for other types of instruments. Changes in the loop would move from being a hardware problem to being a software problem.

The proposed torque loop incorporates two improvements. First is the use of microprocessor control employing Kalman filter control algorithms. Second, this thesis advocates the use of pulse width modulation as a viable method of pulse torquing.

Although pulse width modulated torque loops have been experimented with before, most conventional pulse torquing loops use the more easily implemented binary scheme of torquing (see Section 4.2). Pulse width modulation can achieve superior resolution in

accuracy over other torquing schemes such as binary or ternary. In addition, torque nonlinearities can be reduced using pulse width modulation, since pulse transitions are always identical and therefore net amounts of torque are more precisely produced (see Section 4.2). In addition, readout accuracy is greater than with analog torque loops, and there is a constant heat load dissipated into the gyroscope.

This thesis specifically addresses the design of the Kalman filter estimator and control algorithms for implementation in a microprocessor. Formal design of the hardware is not considered here, although feasibility issues are addressed. The thesis is divided into four parts.

Part I, "Introduction and Background", Chapters 1-5, discusses background information relevant to gyroscopic rebalance loops. A brief description of inertial instruments and the complementary signal and torque generator, vital components in all torque loops, is presented. In addition, the reader is made aware of previous types of torque loops and the work accomplished in this field. Chapter 5 finally presents an overall description of the proposed microprocessor controlled torque loop advocated in this thesis. A block diagram and a brief description of loop operation are included.

Part II, "Theory", Chapters 6-9, derives the formulas used in designing and simulating the torque control algorithms. First in Chapter 6, the differential equations describing the dynamic model of the gyroscope used for analysis and simulation are presented. Also discussed is the analytic form of commanded torque pulses assumed for simulation. Chapters 7 and 8 present the design of the estimator and derivations of the control algorithms respectively. Two slightly different Kalman filter models are developed. One uses three states, position velocity and external inputs, while the other incorporates an added state for the rate of change of external inputs.

Chapter 8 develops two different control algorithms based on whether three or four Kalman filter states are available. An additional third control law is presented which is a simplified control algorithm based only on the average history of float position and

previous commanded torque pulses. It is included because of its simplicity for implementation with reasonably good performance compared to the controllers based on the Kalman filter estimates.

Chapter 9 provides a description of the simulation software used to simulate the closed loop behavior of the proposed torque loop. The Kalman filters, control algorithms, and simulation numerical integration are coded on an IBM 3090 mainframe computer in double precision floating point arithmetic.

Part III, "Simulation Results", Chapters 10-15, presents the simulated behavior of the proposed torque loop. Plots of the closed loop time-domain responses of the two Kalman filter estimator models with their respective control laws, and the simplified controller, are shown for zero constant, step, ramp and sinusoidal external inputs. Run scenarios using different estimator gains for the same inputs are also included, along with the responses to three different levels of measurement noise.

Part IV, "Conclusion", Chapters 16-17 contains a summary and closing remarks concerning implementation details and recommendations for further work required to complete the design proposed in this thesis. Here it is reiterated that real-time microprocessor control can easily be achieved with the simplified control algorithm developed, and most probably achieved with the Kalman filter based control algorithms as well. Any uncertainty as to the practicality of implementing the latter algorithms will be extinguished shortly as microprocessor technology continues to rapidly expand.

2. USE OF ELECTROMAGNETIC TORQUING IN INERTIAL INSTRUMENTS

Electromagnetic torquers are used in spinning wheel gyroscopes and force rebalance accelerometers. This thesis concerns itself with a floated gyroscope torque loop, but the results apply to torque loops for accelerometers and other types of gyroscopes as well.

2.1 Single Degree of Freedom Floated Gyroscope

A diagram of a single degree of freedom floated gyroscope is given in Figure 2-1. A cylinder (called the float) contains a spinning wheel, which in the latest designs has gas bearings and a permanent magnet motor. The float is inside a cylindrical case with a viscous fluid between the case and the float.

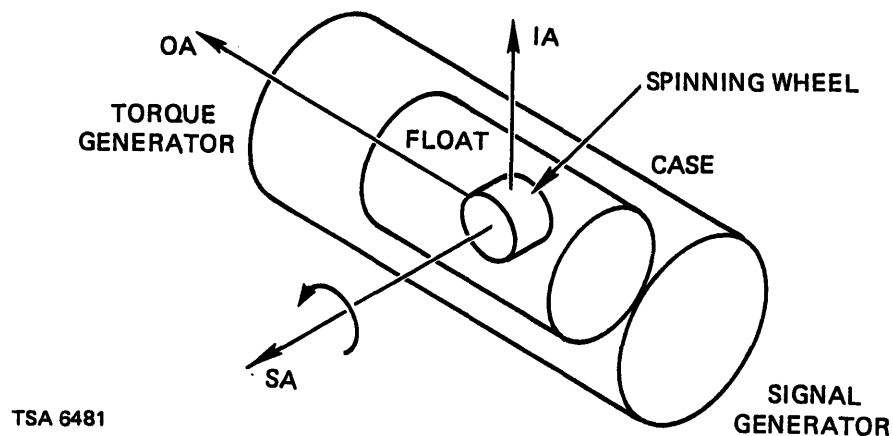


Figure 2-1. Single Degree of Freedom Floated Gyroscope Coordinate Axes.

The float is nearly neutrally buoyant in the fluid, with any buoyancy error being compensated by magnetic suspensions which keep the float centered in the case and allow only rotations about the output axis OA. Thin wires called flex leads run between the float and the case to bring in electric power to the wheel motor, and also to the torquer windings if they are on the float rather than the case.

The Spin Axis (SA) of the float is along the wheel spin axis, the Output Axis (OA) is perpendicular to SA and points along the float cylinder axis towards the torque generator, and the Input Axis (IA) completes the SA, IA, OA right handed coordinate system.

An angular velocity about OA causes the float to rotate about IA by gyroscopic precession, but this motion is suppressed by the fluid squeeze film damping in the narrow radial gaps and by the magnetic suspensions. Hence, the floated gyroscope is a single degree of freedom device, being designed to respond only to angular velocities about IA.

2.2 Use of Gyroscope Torquing

An angular velocity about IA causes the float to rotate about OA by gyroscopic precession. This rotation is sensed by the signal generator at the -OA end of the gyroscope, and if the instrument is being operated in torque-to-balance mode, the torque loop electronics cause current to flow through torquer windings at the +OA end to torque the float about OA back to the signal generator null position.

In a strapdown Inertial Measurement Unit (IMU) the gyroscopes would be operated in torque-to-balance mode. This mode is also used in gyrocompassing, where the gyroscope IA is alternately oriented east and west in the horizontal plane to detect the direction of north from the sensed earth rotation angular velocity as separated from the effect of gyroscope bias torque.

Instead of torque-to-balance mode, the instrument can be operated in servo mode, where the output of the signal generator is used to command a gimbal motor or other actuator to rotate the platform in which the gyroscope resides to cancel out the angular

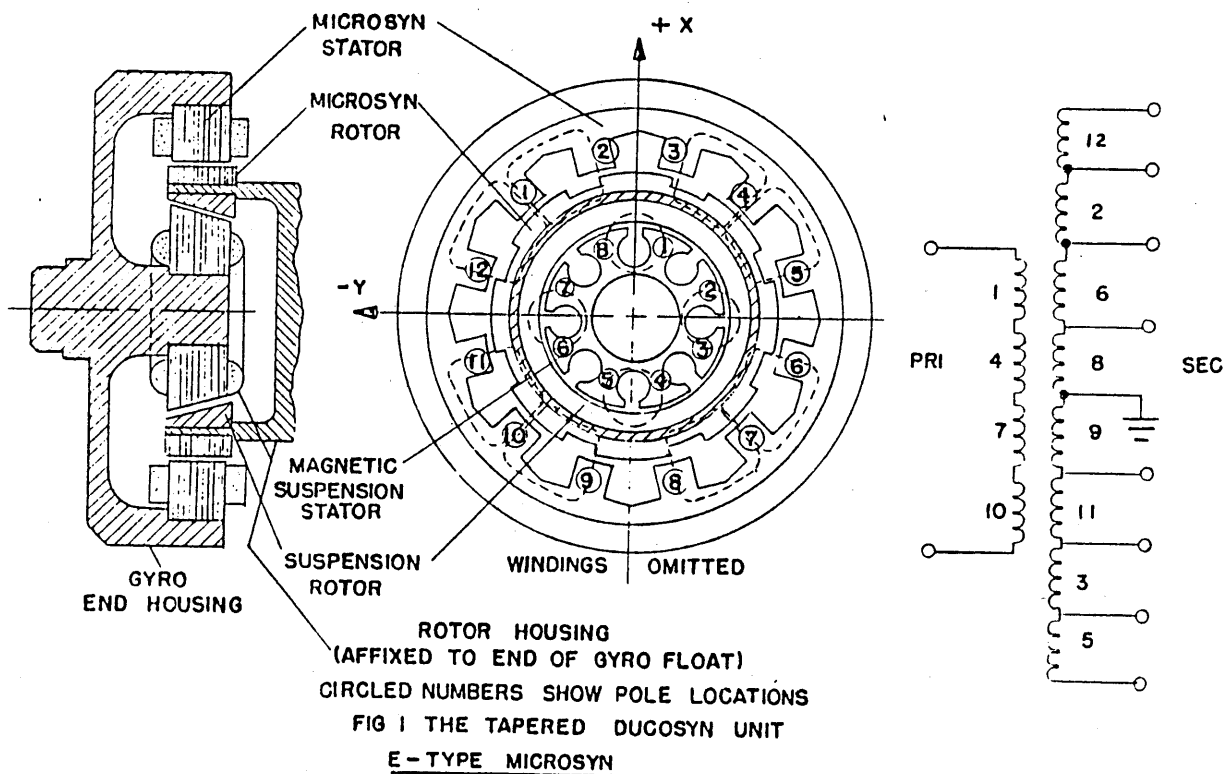
velocity along IA, and hence move the float about OA back to signal generator null. Servo mode would be used for a space-stabilized IMU.

In servo mode, the IMU inner platform is made to rotate by putting commanded torques into the gyroscopes. A gyroscope intended for a servo-mode application is often tested in torque-to-balance mode. Hence an improved gyroscope torque loop is of interest for servo-mode applications as well as strapdown and gyrocompassing applications.

3. SIGNAL AND TORQUE GENERATORS

3.1 Signal Generator (SG)

A microsyn signal generator is located in the -OA end housing of the gyroscope. Angular rotations of the float about the output axis OA are sensed by the SG which generates an electrical signal proportional to the angular displacement of the float from an aligned position called SG null. A microsyn is basically a reluctance transformer consisting of a stator core, a rotor core, and primary and secondary windings. The output signal's dependence on float angle is a result of angle-dependent inductive coupling of the various primary and secondary windings. Figure 3-1 shows a typical SG configuration.



8/74 CD5118

Figure 3-1. E-Type Microsyn Signal Generator [1]

A sinusoidal excitation voltage through the primary SG windings induces an output voltage in the secondary. The sign and magnitude of the winding coupling varies directly with the amount of float rotation about OA. The SG excitation frequency is 6.4, 19.2, or 38.4 KHz in various gyro designs. The magnetic suspension waveform is often 12.8 KHz. The relationship between the signal generator and suspension excitation frequencies is chosen to minimize cross-talk effects.

3.1.1 Readout Techniques

The output from the secondary winding may be passed through a phase sensitive demodulator to yield a DC voltage whose amplitude is proportional to the OA float rotation angle. Alternatively, the secondary output waveform can be sampled directly. Peak A/D sampling in phase with the excitation frequency can be done to directly measure the OA rotation, or A/D sampling all around the waveform can be done and an estimator used to determine the angular offset. An Analog Devices 12 bit, 1 MHz AD9003 A/D converter could give up to 50 samples over each cycle of a 19.2 KHz sinusoid.

In this thesis 19.2 KHz signal-generator angle measurements are assumed to be available. If the signal generator excitation frequency were in fact 19.2 KHz, as it is for the hypothetical gyro modeled in Section 6.1, then peak sampling could in principle be used. However, it is unlikely that peak sampling would yield low enough measurement noise. A demodulator and filter, or sampling all around the waveform and digital filtering, would probably be necessary.

3.1.2 Readout Accuracy

A sampled measurement noise of 100 nano-radians was assumed in the Kalman filter gain and covariance calculations. It is unclear what measurement noise is actually realizable. With the 800 Hz torquing frequency and the 12 dyne-cm torque pulse height assumed in the simulations, the float OA oscillations due to torque pulses are of the order of 20 nano-radians amplitude, so it would seem that a low measurement noise would be a requirement.

Signal amplification would be required to obtain an SG voltage in the ± 2.5 V range of the A/D converter. If the least significant bit of the A/D converter indicated 1 nano-radian, then the full scale range of a 12 bit A/D converter would be ± 2 μ rad. An angle larger than this would saturate the A/D converter. If this occurs, the Kalman filter must be told to either throw out the measurement update or just guarantee that the estimate is at least equal to SG full scale. However, the torque decision for an angle larger than this is straightforward; just torque the maximum amount in the appropriate direction.

Some experiments require SG readouts over the full $\pm 0.25^\circ$ stop-to-stop OA angle range. These experiments would require an SG gain of 1/2,000 of that used in a torque loop that is trying to keep the OA angle near signal generator null.

3.2 Permanent Magnet Torque Generator (TG)

At the opposite end of the gyro from the SG is a permanent magnet torque generator. The torque generator acts on signals recieved from the SG to torque the float about OA back to null. Figure 3-2 shows a simplified version of one pole of the torquer. The two primary components consist of a magnet and flux return path on the case, and a moving coil on the float in the gap between the magnet and return path. The permanent magnet and return path set up a magnetic field in the gap. The magnet and return path are mechanically connected by a non-magnetic material so that the gap and thus the flux remains constant.

A conductor within a magnetic field with current running through it will generate a force

$$\vec{F} = \vec{I} \times \vec{B}$$

where $\vec{F}$ is the force, $\vec{I}$ is the current and $\vec{B}$ is the magnetic flux. If the conductor is at a distance $\vec{r}$ from the float rotation axis, then a torque $\vec{M} = \vec{r} \times \vec{F}$ will be generated. If m conductors are used to make a coil and n coils are assembled onto a rotor, then the total torque produced is

$$\vec{M} = (mn)(\vec{r} \times \vec{F})$$

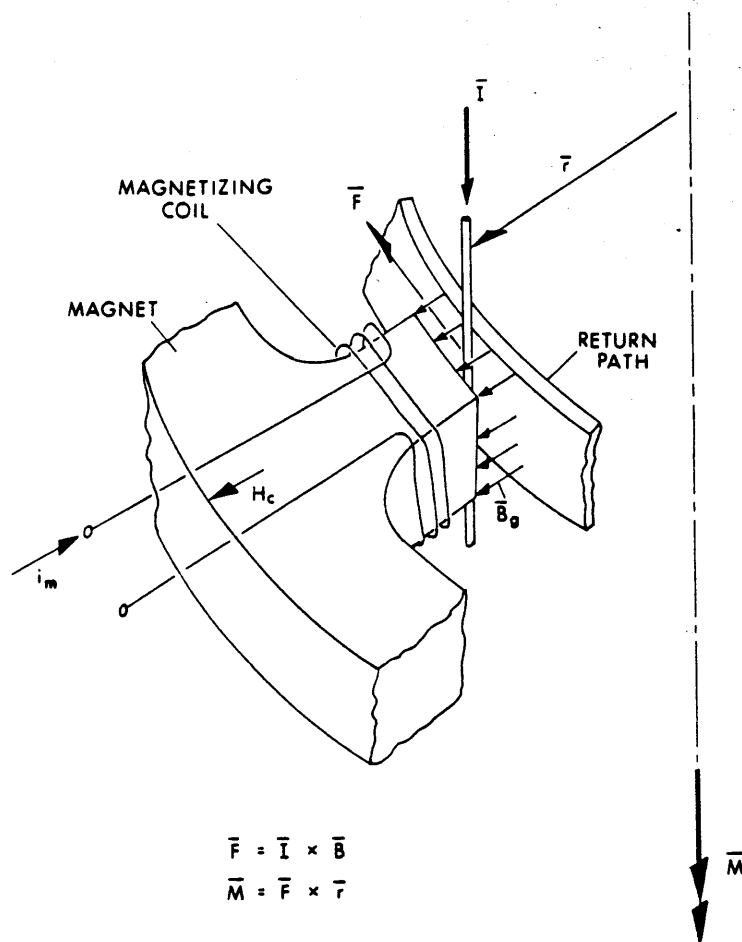


Figure 3-2. Section of Permanent Magnet Torque Generator [2]

At present, floated gyro torquer permanent magnets are made with Alnico magnets. The return path is a high permeability ferromagnetic material and the rotor structure has copper coils. The TG is assembled with the magnets demagnetized. The coil shown wrapped around the magnet pole in Figure 3-2 is used to magnetize the magnet once assembly is complete. This is accomplished by passing a large pulse of DC current through the winding. The assembled unit can be magnetized or demagnetized as required. This magnetizing winding is also used for torquer coil alignment by passing an AC signal

through the winding and reading the output across the moving coil. Figure 3-3 shows a fully assembled unit.

If Samarium-Cobalt magnets were used, they would have to be magnetized before assembly into the instrument, because the required magnetizing currents are too large for the coils that can be fit into an instrument. (Samarium-Cobalt magnets are used in the permanent magnet spin motors of the most recent floated gyroscope designs.)

Permanent magnet (PM) torquers have superior characteristics compared with the previously used soft iron torquers, especially for pulse-mode torquing. For PM torquers, the torque is proportional to the current thereby eliminating the linearity problem due to current tailoff as in soft iron torquers, where the torque is proportional to the current squared. Also, torques of opposite signs are produced by merely reversing the direction of current flow. PM torquers can achieve higher torque levels with negligible hysteresis and with lower power dissipation than soft iron torquers. The pulse rise time is also significantly faster. The time constant for the current rise or fall through torquer windings in a floated gyro permanent magnet torquer is of the order of $T=3\text{ }\mu\text{sec}$ [3]. After n time constants, the current is within e^{-n} of being at the new level.

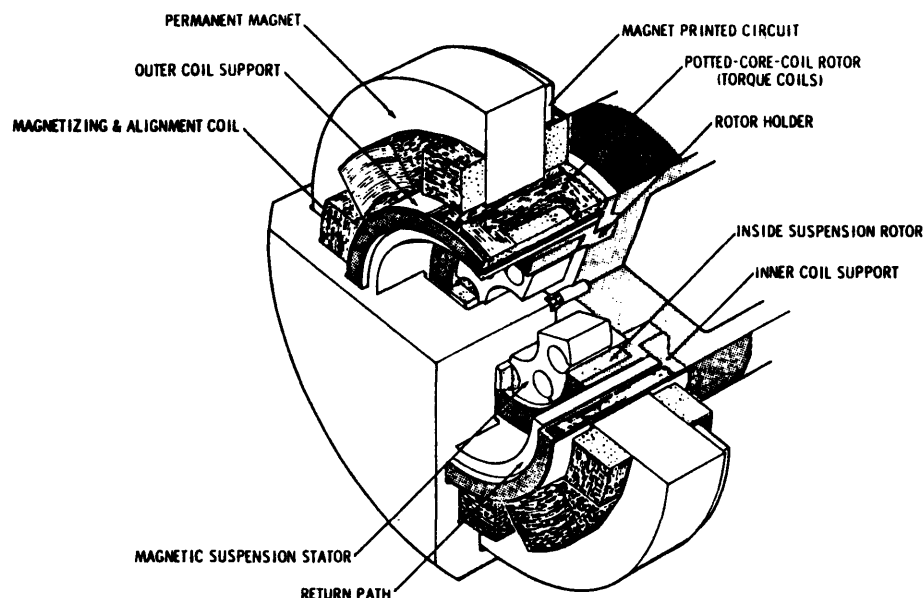


Figure 3-3. Fully Assembled Permanent Magnet Torque Generator [2]

4. PREVIOUS TYPES OF TORQUE LOOPS

Torque loops can be categorized as using linear torquing (analog loops), where the feedback is continuously variable, or pulse torquing. Pulse torque loops contain hard nonlinearities, because the feedback signal must be at one of a finite number of fixed predetermined levels at any instant. This chapter gives a brief description of the different types of torque loops and the analysis techniques most commonly employed. More in-depth descriptions can be found in the references cited.

4.1 Analog Loops

Figure 4-1 shows a block diagram for a typical analog torque loop. The rotation off SG null is used to generate an analog current through the torquer windings to move the float back to null. The sign and magnitude of the required current is a measure of the input-axis rotation rate; a current-to-voltage converter yields a signal that can be read by an A/D converter or digital voltmeter.

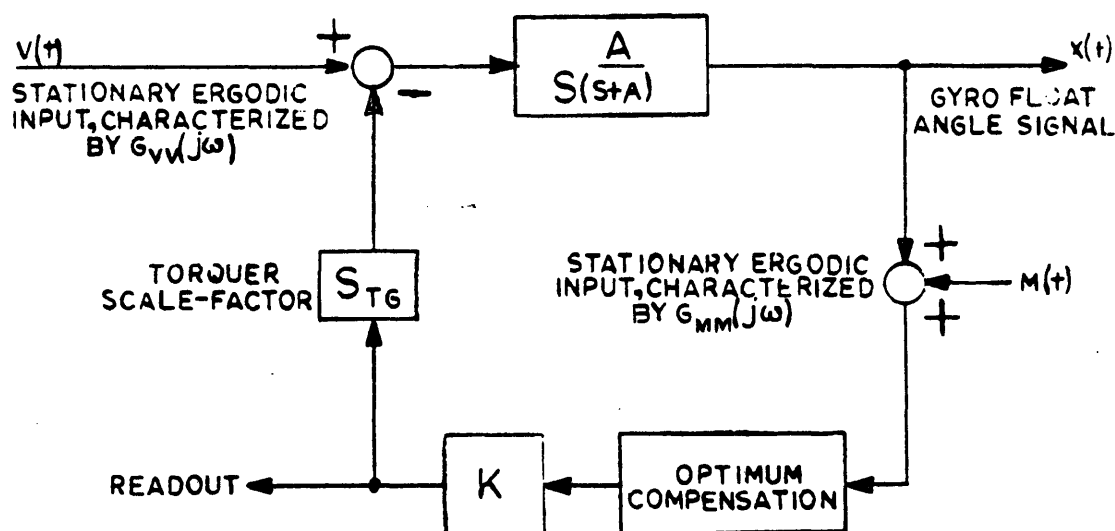


Figure 4-1. Analog Torque Loop Block Diagram [4]

These were the first types of torque loops developed. The loop is closed through a constant gain which is chosen to provide a response with specified bandwidth, while filtering out higher frequency noise. The transient response is usually sluggish with characteristic times on the order of seconds. For floated gyros with high fluid viscosities, stability compensation is usually not required.

Disadvantages of an analog torque loop are (1) the variable amount of current and hence heat input to the gyro, which causes irregular fluid-flow disturbances and gain changes; and (2) the difficulty in obtaining an accurate readout. The average of readings of voltmeters or A/D converters improves as the square root of the sample time T , up to the limit of accuracy of the voltmeter or A/D converter due to temperature and other sensitivities separate from those of the gyro.

4.2 Pulse Torquing Loops

Pulse torque loops were developed to reduce scale factor non-linearities and increase readout resolution. First, the scale factor S_{TG} in Figure 4-1 will vary unpredictably with feedback-current-induced heating. Pulse torquing eliminates the need for an accurate, continuously-variable torque generator, since pulses only require a constant accurate current source.

Second, a direct digital readout is desirable, because the torque readout electronics are simpler without an A/D converter and greater accuracy in the torque measurement can be obtained by counting pulses. The quantization error of pulse counts improves as T^{-1} , where T is the sample interval, and even faster with use of appropriate finite impulse response filters. It is experimentally observed that torque measurements provided by averaging pulse counts have lower noise than those provided by averaging A/D converter readings from an analog torque loop. This is because the temperature and other error sensitivities of an A/D converter are larger than the equivalent gyro error sensitivities.

Pulse torque loops work well when the input frequencies are substantially lower than the torquing frequency. In addition, torque pulse widths should not approach the characteristic time of the torque generator since scale factor non-linearities may occur due to eddy current effects. Because the feedback signal is periodic in nature, pulse torque loops represent non-linear stable control systems with self sustained oscillations or limit cycles. Three common pulse wave forms used for pulse torque loops are binary, ternary, and pulse width modulation.

4.2.1 Binary Torquing

Figure 4-2 shows a block diagram of a typical pulse torque loop employing binary torquing. The feedback torque consists of a positive or negative pulse of magnitude D occurring every T seconds. Each pulse imparts a velocity increment of either $+DT$ or $-DT$ to the float. The control torque is made proportional to the input rate. In the absence of any input $\omega(t)$, the float will oscillate about null. Any difference between the pulse commanded input and the actual external input will be stored in the float dynamics until the error becomes greater than one quanta of commanded torque. This is often referred to as torque quantization [5]. Minimum quantization is achieved only if the controller exhibits 1:1 moding,

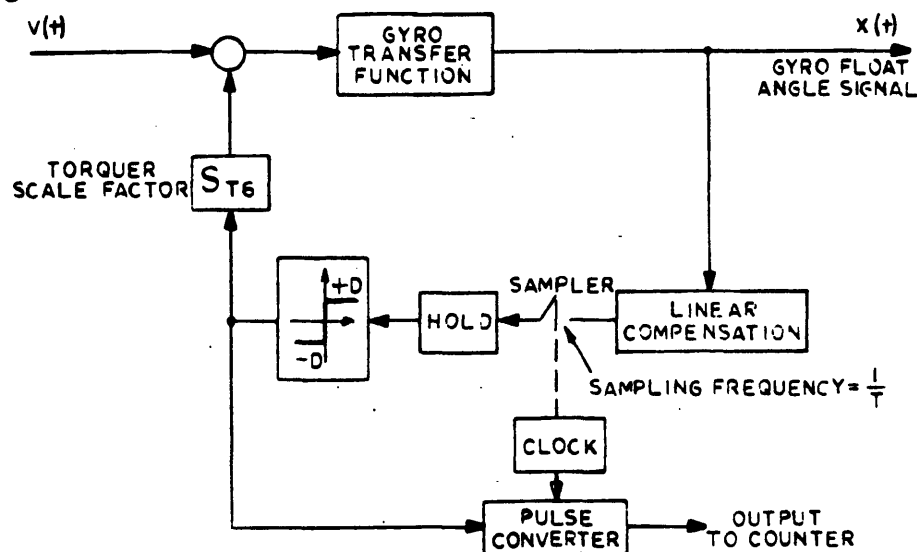


Figure 4-2. Binary Pulse Torque Loop Block Diagram [4]

that is, when the torquing pattern achieves the average of the external input in the shortest period possible. In this condition, the float will oscillate at the maximum possible frequency and the system will respond to deviations from null in the shortest time. Moding will be discussed further in Section 4.2.3.

The torque loop capability is defined by the amplitude of the feedback pulse D . If an external input rate of greater amplitude is applied, the float angle will diverge. As the torque pulsing frequency increases, the bandwidth of the loop will increase while the quantization error decreases. Quantization error can also be reduced by decreasing the pulse height D .

A source of noise for binary and all digital torque loops is the accuracy of pulse height control. However, this noise is less than that provided by the usual readout of an analog torque loop. The pulse height as controlled by a Zener diode approaches part in 10^6 accuracy. A Superconducting Quantum Interference Device (SQUID) can provide a part in 10^8 accuracy for the pulse current. (It could also provide this accuracy for the readout of the current in an analog loop, but we would still be left with the effects of the varying heat load in a gyro).

The advantages of binary torquing are that there is direct digital readout and there is a constant heat flow into the gyro. A disadvantage of binary torquing is that the quantization of torque is limited by the pulse width for a given amplitude. To obtain better quantization it is therefore necessary to increase the torquing frequency, or lower the pulse height. Increasing the frequency is practical only until the pulse width approaches the order of the inductive time constant of the torquer, at which point the delivered torque becomes highly non-linear. Another disadvantage is that non-linear amounts of torque can be produced due to the difference in net torque delivered during the transitions between two successive positive pulses ++ or negative pulses -- and a plus-minus +- or minus-plus -+ combination of pulses. Since binary torquing allows both types of transitions to occur in

unequal numbers, depending on the torque required, the differences between transitions described may lead to reduced accuracy in the delivered torque.

4.2.2 Ternary Torquing

A ternary loop is similar to the binary loop except the relay switching of torque has an added dead zone. The feedback signal now has three allowed values, $\pm D$ and zero. If the float is within a specified deadband, then no torque is applied. Figure 4-3 shows the form of the non-linearity.

The primary advantage of using ternary torquing is that for zero input there is no torque applied and therefore power is economized. A major disadvantage is the scale factor non-linearity induced by thermal gradients caused by non uniform heat dissipation in the torque generator. Also, since no torque is applied if the float is within the specified deadband, the float will not experience a well defined limit cycle and float oscillations can exhibit jump resonance in the closed loop response to sinusoidal inputs [6]. That is, a single input may produce two or more different responses.

4.2.3 Pulse Width Modulation (PWM) Torquing

Pulse width modulation (PWM) torquing is used for the torque loop simulations presented in this thesis. PWM torquing is different from binary torquing in that positive and negative pulses are always applied in succession, i.e., a plus follows a minus and vice versa. The float will therefore be forced to oscillate at the predetermined torquing frequency. Second, the ratio of positive to negative torquing periods is adjusted to set increments within the period of torquing. Figure 4-4 illustrates PWM torquing.

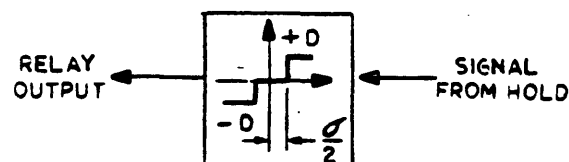


Figure 4-3. Relay Used in Ternary Pulse Torquing [4]

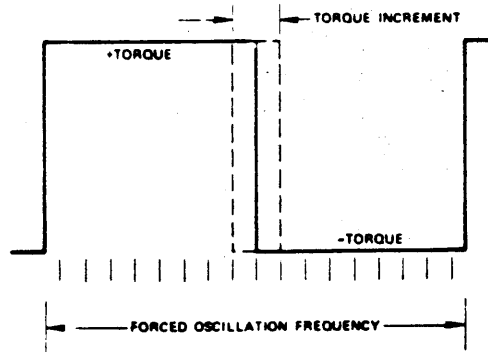


Figure 4-4. Pulse Width Modulation [7]

Delays are easier to adjust electronically than are other quantities. Each pulse interval Δt , e.g., $\Delta t = 1/800$ sec, is split into a fraction x_1 for a positive pulse and x_2 for a negative pulse. Each fraction has a certain quantization resolution, say a part in 256, so that $x_j = m_j/256$ with $m_1 + m_2 = 256$, $1 \leq m_j \leq 255$. The torque loop control law would choose m_1 and m_2 each torque interval so that the float is kept near null. The net torque is measured by the difference $m_1 - m_2$ over a sequence of torque intervals.

In practice, more restrictive minimum and maximum values of m_1 and m_2 are enforced depending on the pulse rise time. Each pulse interval thus always has well-formed positive and negative pulses.

Pulse width modulation torquing shares with binary torquing the advantage of having constant power input to the gyro. Improvements over binary torquing are (1) there are always the same pulse transitions $+-$ or $-+$ in each torquing period, and (2) there is greater quantization resolution. For the 800 Hz torquing example, binary torquing has quantization resolution of one part in 800 in 1 sec, whereas pulse width modulation torquing has $1/(800 \times 256) = 1/2.048 \times 10^5$ quantization resolution in 1 sec. Pure binary torquing cannot get this level of quantization resolution, because for the same pulse amplitude, the binary pulse width would have to be $1/2.048 \times 10^5 = 4.883 \mu\text{sec}$ long. Since the torquer inductive time constant is approximately $3 \mu\text{sec}$, a binary torquing frequency of 204.8 KHz would be highly non-linear and therefore impractical.

As in binary torquing, the PWM feedback torque will have a running average equal to the gyro input torque. The net torque delivered from the chain of positive and negative pulses over some period will therefore equal the input torque. The minimum period required to achieve the average of the input torque will be one torquing interval. For a constant input torque the relay switching instant that results will be referred to as the steady state switching instant. The external torque is determined with minimum quantization (the width between allowable switching instants times the height of the torque pulse D) and the float will oscillate at the highest frequency possible, the torquing frequency. There are infinite other possibilities for combinations of switching instants over longer periods which will produce an average torque equal to the external input. However, the input will be represented with greater quantization error. In addition, the float will oscillate with longer periods and larger amplitudes. These other possibilities will be referred to as higher modes and are undesirable in pulse rebalance loops. Figure 4-5 shows the steady state switching of feedback torque for zero input (solid wave form). The switching will occur half way through the torquing interval Δt . A moding response is also illustrated (dashed wave form).

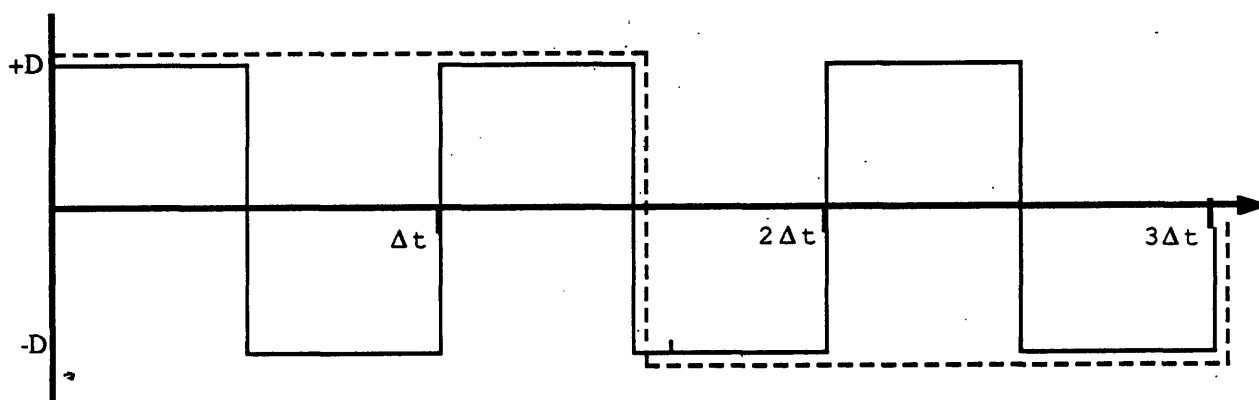


Figure 4-5. Two Possible Feedback Responses for Zero Input

Pulse width modulation is achieved in analog circuitry by adding a periodic wave form (usually a saw-tooth) to the output SG angle. Figure 4-6 illustrates the added dither signal. The sum is then interrogated at a frequency which is some multiple of the dither frequency. Current applied to the torque generator is allowed to switch only during interrogation and only once during the sawtooth period. The net effect is a pair of pulses, plus and minus, of different duration delivering a net acceleration driving the float towards null. The quantization is determined by the interrogation frequency and not by the dither frequency.

Because of the continuing advances in microprocessor technology, it is feasible to implement complicated high speed non-linear control laws in non-analog form, such as the Kalman filter pulse width modulation control scheme proposed in this thesis.

4.3 Torque Loop Analysis

The primary analysis and synthesis methods of pulse torque loops used up to now consist of (1) phase plane method, (2) describing function approximation for frequency domain analysis, and (3) time domain analysis.

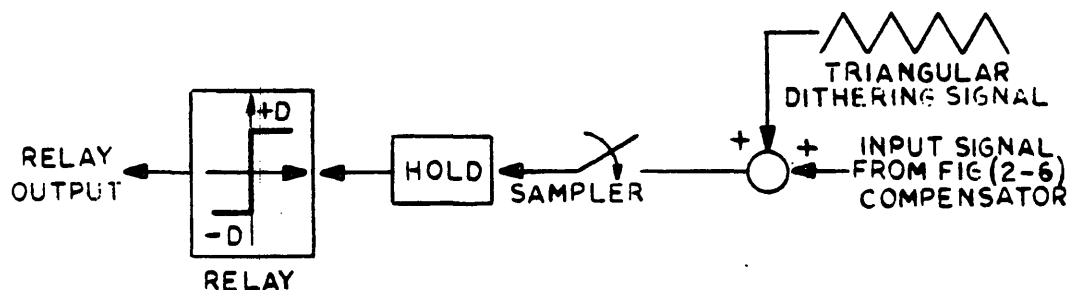


Figure 4-6. Pulse Width Modulation with Analog Circuitry [4]

The phase plane method is a graphical method based on representation of the differential equations which describe second order systems [5]. It allows visual analysis and synthesis of on-off control decision curves.

The describing functions method is based on the sinusoidal response of the closed loop system. It employs an approximation of the non-linearity based on the assumption that all harmonics but the first of the non-linearity can be neglected due to the low pass filtering in the system. The result is a gain expression for the non-linearity that makes the system susceptible to frequency domain analysis. Either the z plane or the s plane (with sampling treated as a range of possible phase lags) can be used. Designs employing describing functions should, however, be verified by computer simulation to check for acceptable transient response and stability. Further discussion of this method can be found in [8,9,10].

Time domain analysis, sometimes called the step by step transient method or piecewise continuous analysis, is usually not advocated as a practical technique for analyzing control systems. This method involves finding exact solutions to the dynamic equations in the intervals where the system is considered linear (interval where constant $+D$ or $-D$ occurs), and matching initial and final conditions across the non-linearity (the instant of switching from $+D$ to $-D$ or vice versa). Although this method of analysis is cumbersome and inflexible, this thesis does employ a time domain solution for two reasons. First, large algebraic equations can be manipulated more easily now with the assistance of software packages. Secondly, the Kalman filter provides accurate estimates of the entire state vector which can be implemented in an exact solution for accurate control decisions.

Previous investigator P. Hutchings [5] elegantly illustrates how all three of these techniques can be used to predict the behavior and provide proper linear compensation of a pulse rebalanced accelerometer. His analysis was based on zero inputs and heuristically extended to accept step inputs as well. The compensator he arrived at using frequency

domain analysis, however, did not yield good results for a ramp input. The only measurement employed in his system was the sign of the float offset from null.

4.4 Torque Loop Implementation Methods

Almost all previous torque loops have been implemented with analog circuitry. Most of the pulse torquing loops are analyzed as linear systems using describing function approximations and are implemented with binary torquing. PWM torque loops with high enough dither and interrogation frequency and input signals within the peak to peak amplitude of the dither frequency, will behave linearly and are usually designed by linear compensation techniques. An example of a PWM torque loop of this type can be found in [11]. Results indicated a loop which maintained minimum mode of operation but did produce significant steady state float hang-offs.

The closest approach to using a state estimator for improvement of the control signal can be found in Reference [7]. This work advocates the use of an interpolator to reduce quantization errors in pulse rebalanced instruments. Through implementation of phase-locked loop techniques, an independent model (estimator) of instrument dynamics was constructed from analog circuitry and used to compare the predicted response of the rebalance loop with the actual response. This method proved to be successful in suppressing quantization errors at high frequencies while preserving low frequency information. The difference is proportional to the difference between input torque and rebalance torque. An error signal is then generated proportional to this difference.

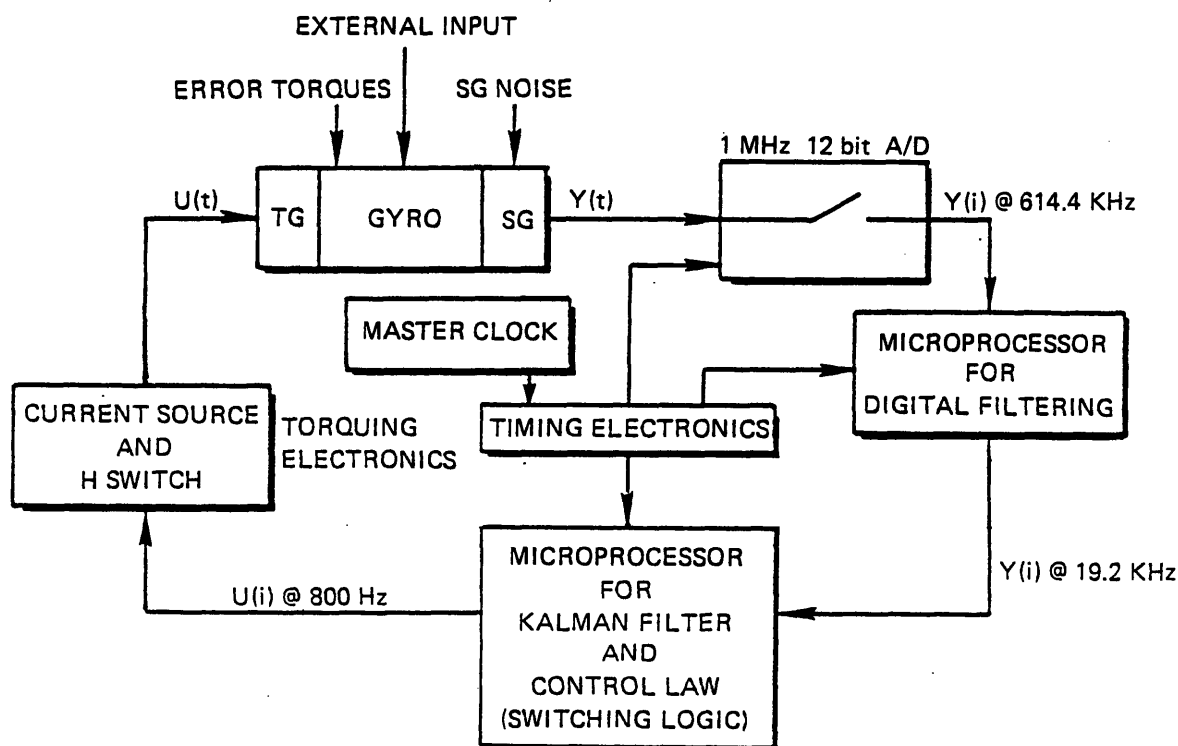
To the best knowledge of this author, one microprocessor controlled pulse torque loop was produced with limited success in the mid 1970's [12,13]. Control decisions were to be made in a Motorola M6800 microprocessor, by implementing a set of difference equations to approximate the proportional plus integral control compensation developed from frequency domain analysis. Binary, ternary, and PWM schemes of torquing were considered, although the final design used binary. Unfortunately the results were not made

available to the author although it was learned that lack of computational speed and accuracy in the available processors contributed to the indication of poor results achieved.

5. PROPOSED PULSE WIDTH MODULATED TORQUE LOOP

5.1 Components and Timing

Figure 5-1 shows a block diagram illustrating the essential components of the microprocessor controlled pulse width modulated torque loop simulated in this thesis. Table 5-1 lists the pertinent torque loop frequencies and shows the countdown multiples from a chosen standard clock frequency. The frequencies chosen are flexible to some degree. They were chosen for simulation to represent realizable values of timing attainable with existing technology. The frequencies must be fast enough so that the torque loop performance will yield an acceptable bandwidth, but slow enough for the processor to keep up with the computational load.



TSA 6510

Figure 5-1. Proposed PWM Torque Loop Block Diagram

Table 5-1. Pulse Width Modulation Torque Loop Frequencies

Countdown multiple from frequency standard	Δt μsec	Exact frequency (KHz)	
1	0.05086	19,660.8	Frequency Standard
32	1.6276	614.4	SG A/D Sample Rate
96	4.883	204.8	Pulse Width Modulation Quantization Interval
512	78.125	12.8	Suspension Excitation Frequency
1024	52.083	19.2	SG Excitation Frequency and Kalman Filter Measurement & Update Interval
8192	0.00125	0.8	Torquing Interval

The 12 bit 1 MHz analog-to-digital (A/D) converter reads the voltage on the SG secondary winding every 1.6276 μsec (614.4 KHz).

These measurements go to the prefiltering microprocessor, which estimates the amplitude of the sinusoid in phase with the SG excitation and outputs the result to the torque loop microprocessor every 32 points (19.2 KHz). This estimator could be as simple as computing a Fourier coefficient by a running sum over the last 32 points (one cycle of the sine wave; see Section 5.2).

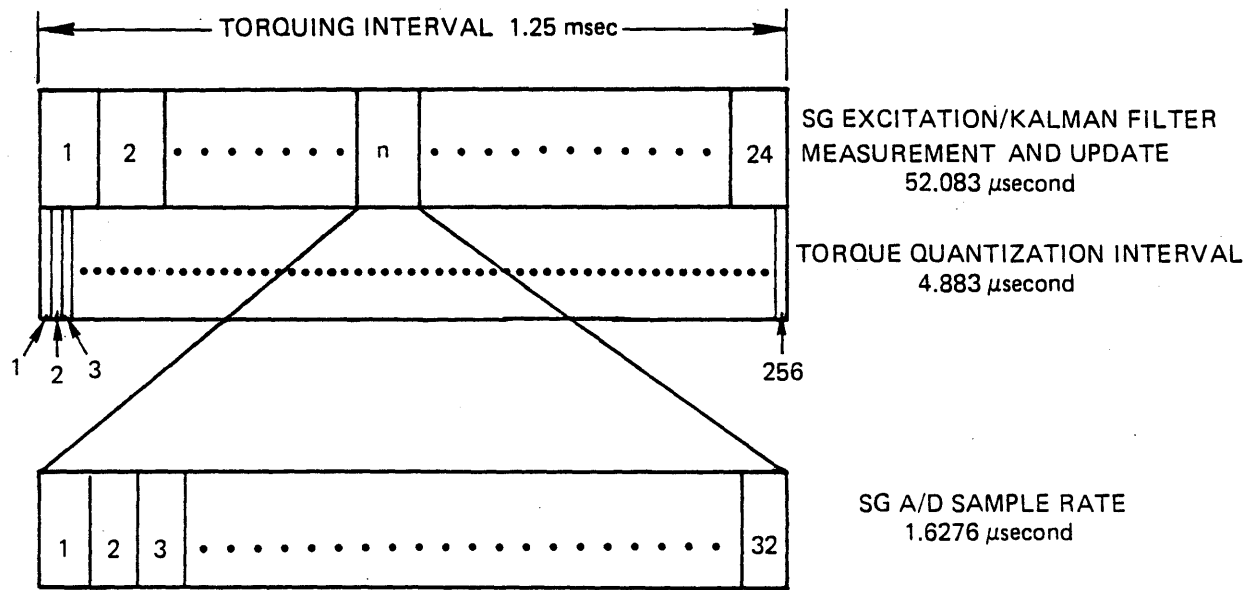
After 32 samples are processed, the torque loop microprocessor receives an interrupt to read the estimated OA angle from the prefiltering microprocessor. These measurements are used to update the Kalman filter estimates of float position and velocity, the external inputs, and also the rate of change of external inputs if the four-state estimator

is being employed. Alternatively, the computations for the simplified torque control algorithm can be done, as discussed later in this thesis.

After 24 readings and Kalman filter updates, which occur in parallel with the current torquing interval, a new torque word N is issued for the next torquing interval. Each torquing interval is 1.25 msec long and consists of a positive torque pulse followed by a negative pulse of equal amplitude. The instant when torque is changed from plus to minus, and therefore the ratio of total positive to negative torque delivered during a torquing interval, is specified by N .

The torquing interval is subdivided into n quantized allowable switching instants where torque may switch from plus to minus. The simulations in this thesis assume $n=256$. Therefore possible switching instants will occur at quantized intervals of $4.883 \mu\text{sec}$. The torque word is an 8 bit binary integer N between 5 and 250 specifying at which of the quantized instants the current in the torquer will switch. The reduced interval insures that successive pulses do not collide during transitions and therefore full, well-shaped torque transitions are always achieved.

Since each torquing interval begins with a positive pulse with minimum duration of $5 \times 4.883 = 24.415 \mu\text{sec}$, the torque loop microprocessor will have a minimum of this much time to perform the computations necessary to arrive at a torque word decision. Once a decision is reached, it will be executed at the appropriate instant. Figure 5-2 shows the timing of events during one torque interval of loop operation.



TSA 6511

Figure 5-2. Torque Loop Events Time Line

5.2 SG Prefiltering Computations

Let $\sin \omega t$ be the signal input to the primary SG winding, where $\omega/2\pi = 19.2$ KHz.

Let

$$t_j = j/(32 \times 19,200) \quad j = 0, 1, \dots, 31$$

be the sample times over one cycle of the excitation, with period $T = 1/19,200 = 52.083$ μ sec. The output of the secondary SG winding which is A/D sampled is

$$y(t_j) = y_j = a_0 + a_1 \sin \omega t_j + a_2 \cos \omega t_j + \dots$$

It is desired to estimate the a_1 coefficient, since the amplitude of the secondary winding voltage in-phase with the excitation on the primary winding is proportional to the OA angular rotation away from SG null.

Theoretically

$$a_1 = \frac{2}{T} \int_0^T y(t) \sin \omega t \, dt$$

$$\approx \frac{2}{T} \sum_{j=0}^{31} y_j \sin \omega t_j \Delta t$$

where

$$\Delta t = \frac{T}{32}$$

Thus, the estimate of a_1 is

$$\hat{a}_1 = \frac{1}{16} \sum_{j=0}^{31} y_j \sin \omega t_j$$

At the start of an SG primary excitation cycle, a running sum location is initialized to zero. Every 1.6276 μ sec, the SG secondary voltage is read through the A/D converters, the reading is multiplied by $\sin \omega t_j$ from a prestored table, and the product is added to the running sum. After 32 readings, the running sum is sent to the torque loop microprocessor and the running sum is re-initialized to zero for the start of the next 32 A/D readings.

If the torque loop microprocessor is fast enough, the SG prefilter computations could be in a task in the interrupt driven multi-tasking torque loop microprocessor software rather than being in a separate microprocessor.

The level of noise achievable from this or other SG analog or digital prefiltering has not been analyzed, since this thesis has concentrated on deriving and simulating the performance of pulse width modulation torque loop controllers presuming that OA angle measurements exist with adequate accuracy.

5.3 Torque Loop Control Computations

The derivation of candidate torque loop control algorithms is contained in Part II of this thesis, results from simulations of their performance is given in Part III, and possibilities for implementing the scheme described in this Chapter are discussed in Part IV.

PART II. THEORY

6. TORQUE SUMMING MEMBER DYNAMIC EQUATIONS

6.1 Dynamic Differential Equations

For the cylindrical float depicted in Figure 2-1, let

I = float moment of inertia about the Output Axis

$$= 62 \text{ gm-cm}^2$$

C = float rotational damping about the Output Axis

$$= 5.5 \times 10^4 \text{ dyne-cm-sec}$$

$$I/C = 0.00149 \text{ sec} = 1,490 \text{ } \mu\text{sec}$$

K = Output Axis elastic restraint

$$= 0 \text{ dyne-cm/rad}$$

(possibly non-zero because of flex leads)

H = wheel angular momentum

$$= 104,000 \text{ dyne-cm-sec}$$

$$I/H = 0.0006 \text{ sec} = 600 \text{ } \mu\text{sec}$$

The above values for I, C, K, and H are for a Draper Laboratory gyroscope design that has not been implemented. The pulse width modulation torque loop control law derived in this thesis was intended to be used in testing this proposed gyroscope. If the torque loop of this thesis were implemented on some existing gyroscope, the values of I and C for the existing instrument should be used in deriving the gains and other parameters of the torque loop.

Let

θ = float rotation angle about the Output Axis relative to the signal generator null (radians)

τ = error torques and effect of input-axis angular rates (dyne-cm)

$$= T_{\text{error}} + H \omega_i$$

ω_i = angular velocity about the Input Axis (radians/sec)

$T_c(t)$ = torque loop commanded torque at time t (dyne-cm)

Then the second order differential equation for θ as a function of time is

$$I \frac{d^2\theta}{dt^2} + C \frac{d\theta}{dt} + K\theta = \tau + T_c \quad (6-1)$$

For convenience, we write this second-order differential equation as a system of first order differential equations. Let

$$\theta_1 = \theta \quad (\text{radians})$$

$$\theta_2 = d\theta/dt \quad (\text{radians/sec})$$

Then assuming that $K=0$, Equation (6-1) can be written as

$$\frac{d\theta_1}{dt} = \theta_2$$

$$\frac{d\theta_2}{dt} = -\frac{C}{I} + \frac{\tau}{I} + \frac{T_c}{I} \quad (6-2)$$

The torque loop will seek a value of $T_c(t)$ which keeps θ_1 and θ_2 oscillating about zero with zero average values. The value of θ_1 is bounded by the $\pm 0.25^\circ$ mechanical stops about OA.

6.2 Analytic Form of Commanded Torque

When numerically integrating the true OA angular motion during a simulation, the external torque $\tau = H \omega_i$ is assumed to be a known function of time (constant, ramp, step, impulse, or sinusoid). In parallel with propagating the true motion of θ_1 and θ_2 , the controller's Kalman filter states θ_1 , θ_2 , τ , and sometimes $\dot{\tau}$ are propagated. In propagating the Kalman filter states, τ is not assumed known, but rather its value (and perhaps that of $\dot{\tau}$) is to be estimated along with the values of θ_1 , θ_2 from signal generator observations.

In both the true state and Kalman filter state numerical integrations, the value of the commanded torque $T_c(t)$ over the next $\Delta t = 1/800$ sec torque pulse interval is assumed known. During the interval Δt , the value of T_c is first positive and then negative, where the relative widths of the positive and negative pulses are determined according to the control law at the start of the Δt interval.

These relative widths have to be specified to some quantization level. A quantization of $1/256$ was chosen for this thesis, so that

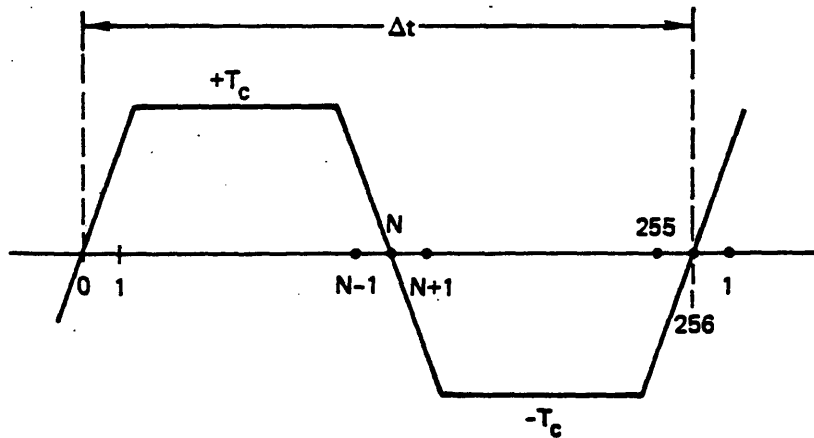
$$\Delta t = 1/800 \text{ sec} = 1,250 \text{ } \mu\text{sec}$$

$$\delta t = \Delta t / 256 = 4.8828125 \mu\text{sec}$$

The Kalman filter measurement updates are assumed to occur 24 times during each Δt interval, or every 52.083333 μsec .

The rise and fall times of the torque pulses are approximated in the simulations of this thesis by assuming a trapezoidal shape for the torque pulses. To go from 0 to $\pm T_c$ is assumed to take $\delta t = 4.88 \mu\text{sec}$, which is not exactly correct, but is what was assumed in the simulations. Then the shape of the width-modulated pulse train is as given in Figure 6-1.

Even though a trapezoidal pulse shape was assumed in the numerical integration simulations, the derivations of the various control laws assumed rectangular pulses. The excellent performance for these control laws reported in Part III of this thesis was still obtained for control models that were not quite the same as the simulation truth model.



TSA 6482

Figure 6-1. Pulse Width Modulation Pulse Shape for Numerical Integration Simulations

The torque word integer N between 0 and 256 in Figure 6-1 is chosen at the start of a Δt torquing interval by the control law based on the Kalman filter estimated states, or on the past average signal generator readings and previous torque word for the simplified controller. In order to always have a positive and a negative pulse in any Δt torquing interval and allow for pulse rise and fall times, it was required that $5 \leq N \leq 250$.

The range of N should have been somewhat narrower than this (see Chapter 17 for implementation details), but this range was what was assumed in the simulations. For steady state operation, this is of no concern. When trying to move the float angle to null as rapidly as possible, as when recovering from a start-up transient or a step input, if the saturated values of $N=5$ or $N=250$ were narrower, then the recovery time would be somewhat longer.

The value of T_c assumed for the simulations of this thesis was

$$\begin{aligned} T_c &= 12 \text{ dyne-cm} \\ \omega_{ic} &= T_c/H \\ &= 1.58 \text{ earth rate units (eru)} \\ &= 23.7 \text{ deg/hour} \end{aligned}$$

Thus the input angular velocities and steps in angular velocity had to be less than 1.5 eru.

Performance of the torque loop should scale directly as the maximum torquing capability, for applications which require a larger such capability. A strapdown application would require a larger torquing capability, whereas a gyrocompassing or laboratory tumble test application could use this or a smaller torquing capability.

6.3 Runge-Kutta Numerical Integration

A fourth-order Runge-Kutta numerical integration technique was used to propagate the true states (Section 6.1), the Kalman filter estimated states (Section 7.2.1), and the state transition matrix (Section 7.2.2.2).

As derived in Reference 5, the Runge-Kutta integration method is self starting (only needs the value of the integration variables at the start of an integration interval). It uses the value of the right hand side function at the start of the interval, its value at the middle of the interval evaluated twice with interim propagation values of the variables, and then at the end of the interval with further propagation values of the variables, before yielding the final values of the variables at the end of the integration step.

The error in the integration is proportional to (step size)⁵ [14]. Thus, arbitrary accuracy can be reached with small enough step size. As explained in Section 9.2.3, the step size was varied depending on how rapidly quantities were changing during a simulation.

7. ESTIMATOR DESIGN

7.1 Introduction

This chapter presents the development of the estimator contained within the microprocessor block in Figure 5-1. It is used to reconstruct the full gyro state vector given a limited number of observables and a model of the system dynamics. In order to keep the simulation flexible for adaptation to different control systems or for an upgraded model of gyro dynamics, an effort was made to use techniques applicable for a more general system model, even if they weren't required for the simple model chosen. The next two sections provide pertinent background information on linear discrete-time systems and estimator design. The last section of this chapter contains the torque loop estimator equations beginning with the discrete form of the continuous state equations and concluding with the estimator gains chosen for simulation.

7.2 Linear Discrete Time Systems

7.2.1 State Propagation

The dynamics of linear continuous-time systems can be represented by a first order vector differential equation of the form

$$\dot{\underline{x}}(t) = \underline{A}(t) \underline{x}(t) + \underline{B}(t) \underline{u}(t) + \underline{w}(t) \quad (7-1)$$

$$\underline{y}(t) = \underline{C}(t) \underline{x}(t) + \underline{v}(t)$$

where $\underline{x}(t)$ is the system state vector, $\underline{u}(t)$ is the deterministic control input, and $\underline{w}(t)$ is a random forcing function (process noise) acting on each of the states. The observed output $\underline{y}(t)$ can be expressed as a linear combination of the state vector plus another random disturbance vector $\underline{v}(t)$ usually referred to as measurement noise. The other matrices $\underline{A}(t)$,

$B(t)$, $C(t)$ describe the system dynamics. For a sampled data system, the controller and estimator have as inputs samples $y(t_i) = y(i)$ of the continuous output $y(t)$. Manipulation of the continuous-time equations is needed in order to arrive at the difference equations which represent the discrete-time behavior of the plant equation (7-1). The discrete time equations will be of the form [15]

$$\underline{x}(i+1) = A_d(i) \underline{x}(i) + B_d(i) \underline{u}(i) + \underline{w}_d(i) \quad (7-2)$$

$$\underline{y}(i) = C_d(i) \underline{x}(i) + \underline{v}_d(i)$$

where by Appendix A

$$A_d(i) = \Phi(t_{i+1}, t_i) \quad \text{the state transition matrix,}$$

$$B_d(i) = \int_{t_i}^{t_{i+1}} \Phi(t_{i+1}, \tau) B(\tau) \underline{u}(\tau) d\tau$$

$$\underline{w}_d(i) = \int_{t_i}^{t_{i+1}} \Phi(t_{i+1}, \tau) \underline{w}(\tau) d\tau$$

$$C_d(i) = C(t_i)$$

(7-3)

$$\underline{v}_d(i) = \underline{v}(t_i)$$

These equations assume there is no processor delay between the time the output is sampled and the instant the input control is adjusted. For a brief discussion of processor delay see Appendix B and Reference [15].

$\Phi(t_{i+1}, t_i)$ is called the transition matrix, and satisfies the differential equation

$$\frac{d\Phi}{dt}(t_{i+1}, t_i) = A(t_{i+1}) \Phi(t_{i+1}, t_i)$$

where $A(t_{i+1})$ is the system dynamics matrix of the homogeneous unforced matrix differential equation

$$\dot{\mathbf{x}}(t_{i+1}) = A(t_{i+1}) \mathbf{x}(t_{i+1}) \quad (7-4)$$

$\Phi(t_{i+1}, t_i)$ describes the influence of $\mathbf{x}(t_i)$ on $\mathbf{x}(t_{i+1})$ such that $\mathbf{x}(t_{i+1}) = \Phi(t_{i+1}, t_i) \mathbf{x}(t_i)$,

as illustrated in Figure 7-1.

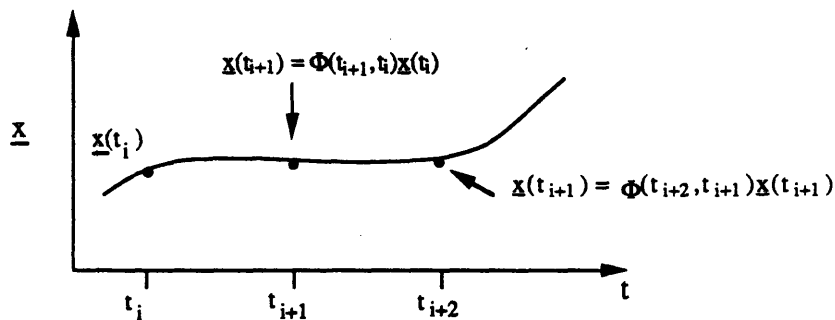


Figure 7-1. Illustration of Transition Matrix Propagating the State Vector Through Time [16]

The other discrete-form matrices B_d , C_d can be found by first solving the homogeneous form of the continuous-time system of equations (7-1). Then using the method of variation of parameters, a guessed solution of the form $\underline{x}_p(t_{i+1}) = \Phi(t_{i+1}, t_i) P(t_i)$ can be substituted into the full system of differential equations (7-1), from which the variable parameter $P(t_i)$ and thus the particular solution $\underline{x}_p(t_{i+1})$ can be solved for. Then using the law of superposition the total solution can be obtained by adding the homogeneous and particular solutions. See Appendix A for a more complete description of this derivation.

7.2.2 Transition Matrix Computation

7.2.2.1 Transition Matrix for Time-Invariant Systems

A system of equations whose properties do not change with time is considered stationary or time invariant. The continuous equations then have the form

$$\dot{\underline{x}}(t) = \underline{A} \underline{x}(t) + \underline{B} \underline{u}(t) + \underline{w}(t)$$

$$\underline{y}(t) = \underline{C} \underline{x}(t) + \underline{v}(t)$$

where the system matrices $\underline{A}$, $\underline{B}$, $\underline{C}$ are now constant.

The transition matrix $\Phi(t_{i+1}, t_i)$ will also be a constant, solving the unforced equation $\dot{\underline{x}}(t_{i+1}) = \underline{A} \underline{x}(t_i)$. The solution is a matrix exponential defined as [17]

$$e^{\underline{A}(t_{i+1}-t_i)} = \underline{I} + \underline{A} (t_{i+1}-t_i) + \frac{\underline{A}^2(t_{i+1}-t_i)^2}{2!} + \dots = \sum_{k=0}^{\infty} \frac{\underline{A}^k(t_{i+1}-t_i)^k}{k!} \quad (7-5)$$

7.2.2.2 Transition Matrix for General Systems

The state transition matrix is the partial derivative of the state at time t_{i+1} with respect to the state at time t_i . Consider a non-linear system where $x = (x^1, \dots, x^n)$.

$$\frac{dx^j}{dt} = F^j(\underline{x}, \underline{u}, t)$$

$$x^j = x_i^j \text{ when } t=t_i, j=1, \dots, n \quad (7-6)$$

Taking the partial derivative with respect to x_i^k yields

$$\frac{d(\partial x^j / \partial x_i^k)}{dt} = \frac{\partial F^j(\underline{x}, \underline{u}, t)}{\partial x_i^k}$$

$$\left. \frac{\partial x^j}{\partial x_i^k} \right|_{t_i} = \delta_{jk} \quad j=1, \dots, n \quad (7-7)$$

where the Kronecker delta is

$$\delta_{jk} = \begin{cases} 0 & j \neq k \\ 1 & j = k \end{cases}$$

Along with the n state equations (7-5), the n^2 partial derivative equations (7-7) are numerically integrated from time t_i to time t_{i+1} to obtain $\partial x^j(t_{i+1}) / \partial x_i^k$. Equations (7-7) are called the variational equations in Celestial Mechanics, and are used when fitting satellite and planetary orbits to observational data (see [18]).

The Kalman filter simulation software on the IBM mainframe computer propagates the state transition matrix by numerical integration of the variational equations. Although not needed for the torque loop model of this thesis, this approach allows extension to more general dynamic systems.

7.3 The Observer

7.3.1 General Form

Figure 7-2 shows the general form of a closed loop estimator. In order to use full-state feedback for control, all of the states must be available. Typically not all of the states are measured. However if the system is observable, an estimator can be used to reconstruct the entire state vector from incomplete state measurements.

Observability is defined as the ability to determine all of the states over time given a limited number of measurements at given instants of time. A mathematical test for observability is that the observability matrix defined by [17]

$$O \triangleq \begin{bmatrix} C \\ C\Phi \\ C\Phi^2 \\ \vdots \\ C\Phi^{n-1} \end{bmatrix} \quad (7-8)$$

be nonsingular. That is, the matrix should be of rank n , where n is the order of the system.

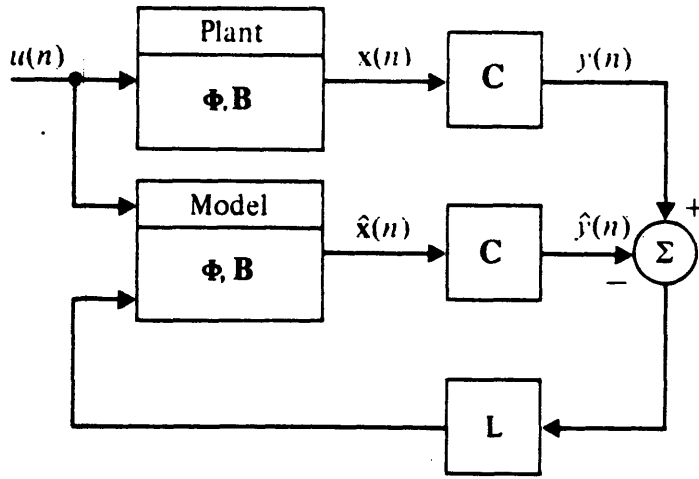


Figure 7-2. Closed Loop Estimator [17]

If the plant dynamics are given by $\underline{x}(i+1) = \Phi \underline{x}(i) + B \underline{u}(i)$, and the output $y(i) = C \underline{x}(i)$ is measured, then the idea is to construct a model of the plant dynamics $\hat{\underline{x}}(i+1) = \Phi \hat{\underline{x}}(i) + B \underline{u}(i)$ and to compare the model's output $\hat{y}(i) = C \hat{\underline{x}}(i)$ for the measured output. If the difference between the measured output and the estimated output is fed back for correction to the model, errors between the two can be corrected. The gain matrix L in Figure 7-2 represents the correction feedback gain. If we anticipate noisy measurements then we tend not to believe in the measurements and rely more heavily on the state estimates, resulting in small gains. Conversely if we are able to make precise measurements, then larger gains would reflect our increased confidence in measurements over state estimates.

If the estimate $\hat{\underline{x}}(i+1)$ is realized by including data only up to the i th time step, then this estimate is called a predictor estimator. The form of the estimates would be [17]

$$\hat{\underline{x}}(i+1) = \Phi \hat{\underline{x}}(i) + B \underline{u}(i) + L [y(i) - C \hat{\underline{x}}(i)] \quad (7-9)$$

This form is useful for coping with controller computation delay. Its drawback is that the state estimate $\hat{\mathbf{x}}(i)$ and the subsequent control $\mathbf{u}(i)$ will not benefit from the current measurement. A better way (if computation time allows), is to first predict the new estimate $\hat{\mathbf{x}}(i+1|i)$ by propagating the old estimate $\hat{\mathbf{x}}(i)$

$$\hat{\mathbf{x}}(i+1|i) = \Phi \hat{\mathbf{x}}(i) + \mathbf{B}u(i) \quad (7-10)$$

Then at time $(i+1)$, a new measurement $y(i+1)$ is made and used to correct our estimate $\hat{\mathbf{x}}(i+1|i)$ by

$$\hat{\mathbf{x}}(i+1) = \hat{\mathbf{x}}(i+1|i) + \mathbf{L}(y(i+1) - \mathbf{C} \hat{\mathbf{x}}(i+1|i)) \quad (7-11)$$

This is called a current estimator and is the basic form of the Kalman filter.

7.3.2 Estimator Gain Matrix Determination by Optimal Estimation

This section outlines the procedure for obtaining the optimal gain matrix $\mathbf{K}$ for the current estimator, based on a recursive weighted least squares solution. Derivation of the Kalman filter equations presented can be found in References [15], [16], and [17].

7.3.2.1 The Stochastic Model

Consider the discrete system model $\mathbf{x}(i+1) = \Phi(i) \mathbf{x}(i) + \mathbf{B} u(i) + \mathbf{w}(i)$ and measurement model $y(i) = \mathbf{C}(i) \mathbf{x}(i) + v(i)$. The process noise vector $\mathbf{w}(i)$ and the measurement noise vector $v(i)$ are assumed to be uncorrelated white noise random sequences with normal Gaussian distributions. No correlation is defined as

$$E[\mathbf{w}(i) \mathbf{w}^T(j)] = E[v(i) v^T(j)] = 0 \text{ for } i \neq j \quad (7-12)$$

where $E[\cdot]$ denotes the expected value. We also require the random sequences to have zero mean

$$E[\underline{w}(i)] = E[\underline{v}(i)] = 0 \quad (7-13)$$

and have mean square values

$$E[\underline{w}(i) \underline{w}^T(i)] = \mathbf{Q} ; E[\underline{v}(i) \underline{v}^T(i)] = \mathbf{R} \quad (7-14)$$

Also, there is assumed to be no correlation between the two sequences, that is

$$E[\underline{w}(i) \underline{v}^T(j)] = 0 \text{ for all } i, j \quad (7-15)$$

7.3.2.2 The Covariance Matrix

One more crucial element of the Kalman filter to be defined is the matrix $\mathbf{P}$ of the estimate covariances. If the error in the state estimate is given by $\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$ then the covariance matrix is defined as $\mathbf{P} = E[\tilde{\mathbf{x}} \tilde{\mathbf{x}}^T]$. Thus the covariance matrix provides us with an indication of estimate accuracy. A large value of $\mathbf{P}$ represents a poor opinion of the current estimation accuracy. As an illustrative example from Reference [15], if we have a system with two state variables, the error in knowledge of which may be represented by

$$\tilde{\mathbf{x}} = \begin{bmatrix} \tilde{x}_1 \\ \tilde{x}_2 \end{bmatrix}$$

then the covariance matrix for $\tilde{\mathbf{x}}$ is given by

$$\mathbf{P} = \begin{bmatrix} E[\tilde{x}_1^2] & E[\tilde{x}_1 \tilde{x}_2] \\ E[\tilde{x}_1 \tilde{x}_2] & E[\tilde{x}_2^2] \end{bmatrix}$$

The diagonal elements of $\mathbf{P}$ are the expected square errors in knowledge of the state variables. The off-diagonal terms represent the cross-correlation between the system state errors.

7.3.2.3 Kalman Filter Update Procedure

Step 1: Initialization

The Kalman filter state estimates are initialized to values representative of approximate system state values.

$$\hat{\mathbf{x}}(0) = E[\mathbf{x}(0)] \quad (7-16)$$

The covariance matrix $\mathbf{P}$ is also initialized according to an estimate of the accuracy of the initial value $\mathbf{x}(0)$.

$$\mathbf{P}(0) = E[(\mathbf{x}(0) - \hat{\mathbf{x}}(0)) (\mathbf{x}(0) - \hat{\mathbf{x}}(0))^T] \quad (7-17)$$

It is often assumed initially that the initial system state errors are uncorrelated. Hence, $\mathbf{P}(0)$ will be a diagonal matrix with the diagonal entries being the assumed initial mean square errors in each of the states. These a priori judgements of initial error are usually determined by best engineering judgement or previous knowledge of similar systems.

Step 2: State and Covariance Extrapolation

The present Kalman filter updated state estimate $\hat{\mathbf{x}}(i)$ will be extrapolated to $\hat{\mathbf{x}}(i+1|i)$ across the next sampling period by the Equation (7-10). The covariance matrix $\mathbf{P}(i)$ is extrapolated to $\mathbf{P}(i+1|i)$ by

$$\mathbf{P}(i+1|i) = \Phi \mathbf{P}(i) \Phi^T + \mathbf{Q} \quad (7-18)$$

where $\mathbf{Q}$ is the covariance of the assumed process noise vector $\mathbf{w}(i)$.

Step 3: Compute Optimal Gain (Weighting) Matrix, $\mathbf{K}(i+1)$

The Kalman filter gain matrix is calculated by

$$\mathbf{K}(i+1) = \mathbf{P}(i+1|i) \mathbf{C}(i+1)^T [\mathbf{C}(i+1) \mathbf{P}(i+1|i) \mathbf{C}(i+1)^T + \mathbf{R}]^{-1} \quad (7-19)$$

Step 4: Measurement Update

At the sampling instant, a new measurement is taken from which the extrapolated state estimate $\hat{\mathbf{x}}(i+1|i)$ and covariance matrix $\mathbf{P}(i+1|i)$ are updated by

$$\hat{\mathbf{x}}(i+1) = \hat{\mathbf{x}}(i+1|i) + \mathbf{K}(i+1) [\mathbf{y}(i+1) - \mathbf{C}(i+1) \hat{\mathbf{x}}(i+1|i)] \quad (7-20)$$

$$\mathbf{P}(i+1) = [\mathbf{I} - \mathbf{K}(i+1) \mathbf{C}(i+1)] \mathbf{P}(i+1|i) \quad (7-21)$$

where $\mathbf{I}$ is the identity matrix.

This concludes one cycle of the Kalman filter. These steps would repeat beginning with step 2. Figure 7-3 is a flow chart summarizing the above steps.

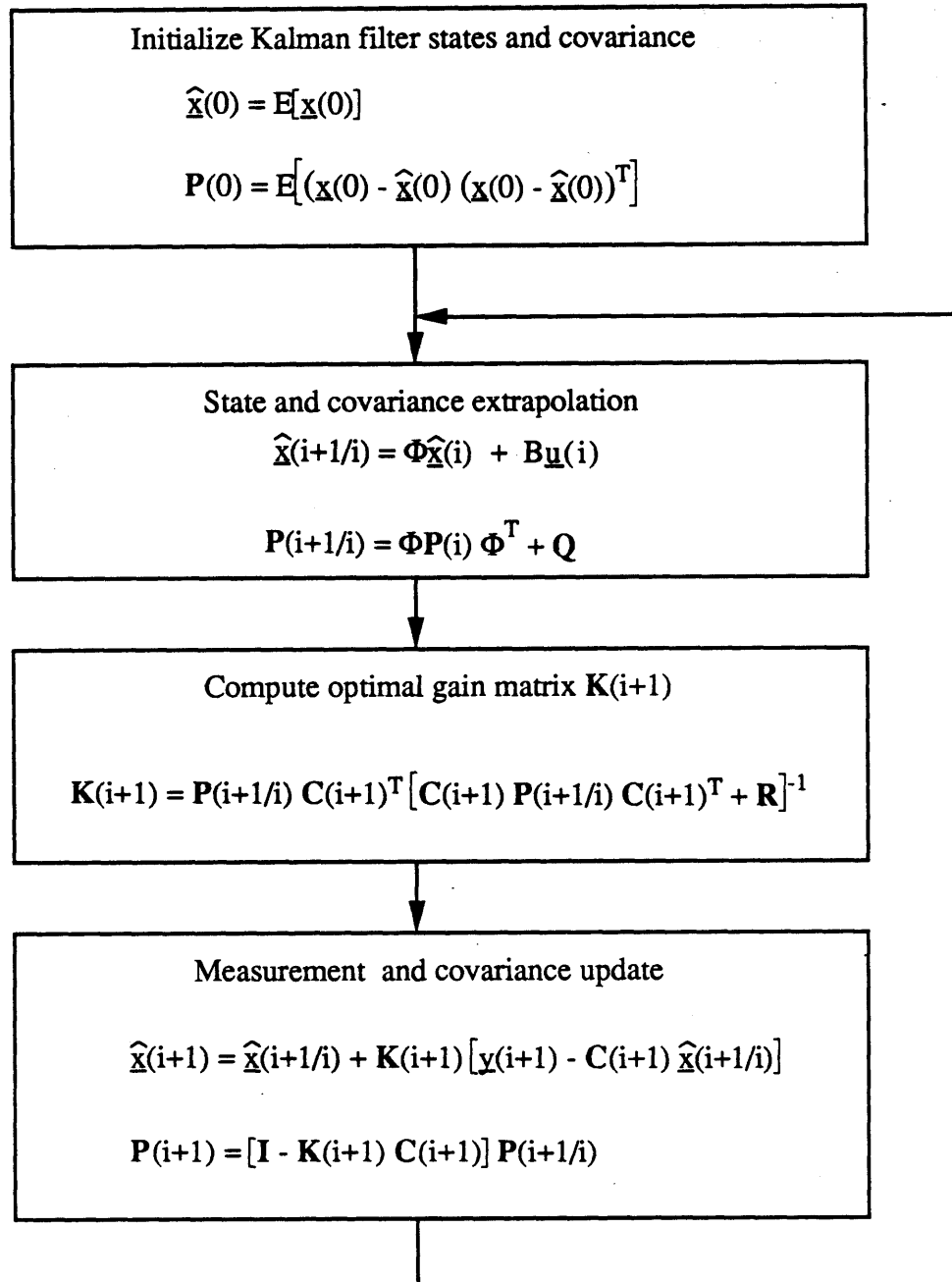


Figure 7-3. Flow Chart of Kalman Filter Update Procedure

7.3.3 Gain Matrix by Direct Pole-Placement

The actual and estimated state equations for the closed loop current estimator are repeated below.

$$\text{true state propagation } \underline{x}(i+1) = \Phi \underline{x}(i) + B \underline{u}(i)$$

$$\text{observables equation } y(i+1) = C \underline{x}(i+1)$$

$$\text{estimated state propagation } \hat{\underline{x}}(i+1|i) = \Phi \hat{\underline{x}}(i) + B \underline{u}(i)$$

$$\text{estimated state update } \hat{\underline{x}}(i+1) = \hat{\underline{x}}(i+1|i) + L[y(i+1) - C \hat{\underline{x}}(i+1|i)] \quad (7-22)$$

Subtracting the updated estimated state equations from the true state equations we obtain

$$\underline{x}(i+1) - \hat{\underline{x}}(i+1) = \Phi \underline{x}(i) + B \underline{u}(i) - \hat{\underline{x}}(i+1|i) - L[y(i+1) - C \hat{\underline{x}}(i+1|i)] \quad (7-23)$$

If we define the error as $\tilde{\underline{x}} = \underline{x} - \hat{\underline{x}}$ then we can rewrite the above as

$$\tilde{\underline{x}}(i+1) = \Phi \tilde{\underline{x}}(i) - L[C(\Phi \underline{x}(i) + B \underline{u}(i)) - C(\Phi \hat{\underline{x}}(i) + B \underline{u}(i))]$$

Simplifying yields,

$$\tilde{\underline{x}}(i+1) = [\Phi - LC\Phi] \tilde{\underline{x}}(i) \quad (7-24)$$

This is the difference equation describing the dynamic behavior of the error. The eigenvalues of $[\Phi - LC\Phi]$ are the poles of the estimator in the z plane and the characteristic equation is given by $\text{Det}[zI - \Phi + LC\Phi] = 0$.

If the desired roots of the estimator are known, and if the system is observable, then the gain matrix L can be determined by matching coefficients in like powers of z on either side of the equation [17].

$$\text{Det}[zI - \Phi + LC\Phi] = \alpha_e(z) \quad (7-25)$$

where $\alpha_e(z) = z^n - \alpha_1 z^{n-1} - \alpha_2 z^{n-2} \dots - \alpha_n$ is the characteristic equation derived from the desired root locations. Another way to solve for the gain matrix is to use Ackermann's formula [17]:

$$L = \alpha_e(\Phi) \begin{bmatrix} C\Phi \\ C\Phi^2 \\ \vdots \\ C\Phi^n \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \quad (7-26)$$

where $\alpha_e(F) = \Phi^{n-1} - \alpha_2 \Phi^{n-2} \dots - \alpha_n I$ and

$$\begin{bmatrix} C\Phi \\ C\Phi^2 \\ \vdots \\ C\Phi^n \end{bmatrix} \text{ is the observability matrix.}$$

7.4 Torque Loop Estimator Equations

7.4.1 Torque Loop State Equations

This section presents the state equations for both the three state and four state models used for simulation.

3 State Estimation: The Kalman filter states for the 3 state estimator are;

$$\theta_1 = \text{SG angle (radians)}$$

$$\theta_2 = \frac{d\theta_1}{dt} \text{ (rad/sec)}$$

$$\tau = \text{error torques plus effects of input-axis angular velocity (dyne-cm)}$$

The external inputs τ are modelled as being constant over each sampling period ($\dot{\tau} = 0$). This assumption is valid if we are assuming slowly varying changes in τ compared to the sampling rate. However, a breakdown of this assumption can be seen in the simulation results when more rapidly varying inputs are applied to the gyro.

This prompted the addition of a fourth state ($\tau' = \dot{\tau}$), since it was desired to have the estimator, and controller, behave better in the face of varying inputs. Now τ was modelled as varying linearly over a sample period and the assumption of a nonvarying state over the sampling period was extended to the fourth state τ' . This manner of extending the external input model by increasing the system type could be continued, but quickly becomes impractical and unprofitable. It was, however, of interest to add one integrator to the external input model in hopes of significantly improving estimator performance, such as eliminating any steady state estimator error for ramp inputs. The four-state Kalman filter will therefore be the same as the 3 state filter with the added state,

τ' = rate of change of error torques plus effects of angular rates from the outside world (dyne-cm/sec)

Using the gyro dynamic model defined in Chapter 3, with the control input $u(i) = \frac{T_c}{I}$ assumed zero, the overall state equations can be written as

3 state:

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\tau} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & \frac{-C}{I} & \frac{1}{I} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \\ \tau \end{bmatrix} + \begin{bmatrix} w_1(t) \\ w_2(t) \\ w_3(t) \end{bmatrix} \quad (7-27)$$

Since the SG measurement of the float angular position is the only observable variable, the output is written as

$$y(t) = [1 \ 0 \ 0] \begin{bmatrix} \theta_1 \\ \theta_2 \\ \tau \end{bmatrix} + v(t) \quad (7-28)$$

4 state:

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\tau} \\ \dot{\tau}' \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & \frac{-C}{I} & \frac{1}{I} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \\ \tau \\ \tau' \end{bmatrix} + \begin{bmatrix} w_1(t) \\ w_2(t) \\ w_3(t) \\ w_4(t) \end{bmatrix} \quad (7-29)$$

$$y(t) = [1 \ 0 \ 0 \ 0] \begin{bmatrix} \theta_1 \\ \theta_2 \\ \tau \\ \tau' \end{bmatrix} + v(t) \quad (7-30)$$

The control input T_c is excluded since it is a deterministic input which is applied identically to both the plant and the dynamic model. It therefore will have no influence on the design of the estimator.

It has been assumed throughout this simulation that all gyro parameters remain stationary in time. Hence the transition matrix $\Phi(t_{i+1}, t_i)$ will also be a constant matrix found by taking the matrix exponential of the continuous system dynamics matrix A , multiplied by the sampling period ($T = t_{i+1} - t_i = 52.083 \mu\text{sec}$).

$$A_d = \Phi = e^{AT} \quad (7-31)$$

Using the gyro parameters already presented, the homogeneous discrete-time difference equations of the estimators can be written as

3 state transition matrix:

$$\begin{bmatrix} \theta_1(i+1) \\ \theta_2(i+1) \\ \tau(i+1) \end{bmatrix} = \begin{bmatrix} 1 & 5.0898 \times 10^{-5} (\text{s}) & 2.1543 \times 10^{-11} (\text{rad/dyne-cm}) \\ 0 & 0.95485 & 8.2094 \times 10^{-7} (\text{rad/s-dyne-cm}) \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \theta_1(i) \\ \theta_2(i) \\ \tau(i) \end{bmatrix} \quad (7-32)$$

4 state transition matrix:

$$\begin{bmatrix} \theta_1(i+1) \\ \theta_2(i+1) \\ \tau(i+1) \\ \tau'(i+1) \end{bmatrix} = \begin{bmatrix} 1 & 5.0898 \times 10^{-5} \text{ (s)} & 2.1543 \times 10^{-11} \text{ (rad/dyne-cm)} & 3.7544 \times 10^{-16} \text{ (rad-s/dyne-cm)} \\ 0 & 0.9548 & 8.2094 \times 10^{-7} \text{ (rad/s dyne-cm)} & 2.1543 \times 10^{-11} \text{ (rad/dyne-cm)} \\ 0 & 0 & 1 & 5.2083 \times 10^{-5} \text{ (s)} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \theta_1(i) \\ \theta_2(i) \\ \tau(i) \\ \tau'(i) \end{bmatrix} \quad (7-33)$$

One might observe that the values in the transition matrices are relatively small numbers. This is because the simulation software was unfortunately set up to measure time in units of seconds. If all parameters were rescaled to measure time in msec, the four state transition matrix would become

$$\Phi = \begin{bmatrix} 1 & 5.0898 \times 10^{-2} \text{ (msec)} & 2.1543 \times 10^{-5} \text{ (rad-msec}^2\text{/gm-cm}^2\text{)} & 3.7544 \times 10^{-7} \text{ (rad-msec}^3\text{/gm-cm}^2\text{)} \\ 0 & 0.95485 & 8.2094 \times 10^{-4} \text{ (rad-msec/gm-cm}^2\text{)} & 2.1543 \times 10^{-5} \text{ (rad-msec}^2\text{/gm-cm}^2\text{)} \\ 0 & 0 & 1 & 5.2083 \times 10^{-2} \text{ (msec)} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (7-34)$$

It is fairly obvious that the system of equations given by (7-29) and (7-30) is completely observable, since essentially we are estimating positions and subsequent derivatives of position from measurements of position over time. As discussed in Section 7.3.1, proof of observability can be seen by showing that the determinant of the observability matrix O is not zero. Using the four state transition matrix above, the observability matrix is,

$$\mathbf{O} = \begin{bmatrix} \mathbf{C} \\ \mathbf{C}\Phi \\ \mathbf{C}\Phi^2 \\ \vdots \\ \mathbf{C}\Phi^{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 5.0898 \times 10^{-2} & 2.1543 \times 10^{-5} & 3.7544 \times 10^{-7} \\ 1 & 9.9498 \times 10^{-2} & 8.4870 \times 10^{-5} & 2.9694 \times 10^{-6} \\ 1 & 1.4590 \times 10^{-1} & 1.8810 \times 10^{-4} & 9.9086 \times 10^{-6} \end{bmatrix} \quad (7-35)$$

$$\text{Det } |\mathbf{O}| = 4.8463 \times 10^{-12} \neq 0$$

7.4.2 Torque Loop Estimator Gains

This section presents the values of the torque loop estimator gains used for simulation arrived at using optimal estimation and pole placement techniques. Three sets of steady state estimator gains are specified in order to illustrate the trade-off between noise rejection capabilities verses transient decay of the error. These will be classified as fast, medium, and slow estimators corresponding to the decay rate of error transients. In order to accurately compare controllers using three- and four-state estimators, both estimators will use the same gains except, of course, the three state estimator will assume the fourth gain is equal to zero. Discussion of gain selection will therefore be related to the four-state estimator only.

7.4.2.1 Estimator Gains by Optimal Estimation

The discrete form of (7-33) with noise is repeated below.

$$\underline{\mathbf{x}}(i+1) = \Phi \underline{\mathbf{x}}(i) + \underline{\mathbf{w}}(i)$$

$$\mathbf{y}(i+1) = [1 \ 0 \ 0 \ 0] \underline{\mathbf{x}}(i+1) + \mathbf{v}(i+1)$$

Estimation accuracy is indicated by the covariance matrix $\mathbf{P}$. For the four state filter $\mathbf{P}$ is defined as

$$P(i) = \begin{bmatrix} E(\tilde{\theta}_1^2) & E(\theta_1\theta_2) & E(\theta\tau) & E(\theta\tau) \\ E(\theta_2\theta_1) & E(\tilde{\theta}_2^2) & E(\theta_2\tau) & E(\theta_2\tau') \\ E(\tau\theta_1) & E(\tau\theta_2) & E(\tilde{\tau}^2) & E(\tau\tau') \\ E(\tau'\theta_1) & E(\tau'\theta_2) & E(\tau'\tau) & E(\tilde{\tau}'^2) \end{bmatrix} \quad (7-36)$$

where $\tilde{\theta}$ denotes state estimation error.

Initially it is assumed there is no correlation between the mean square errors. Therefore $P(0)$ will be diagonal matrix. The off diagonal entrees will fill in as the filter proceeds due to the interconnection of the errors through the differential equation. The initial values chosen for $P(0)$ will have no bearing on the steady state performance of the Kalman filter. $P(0)$ is chosen as

$$P(0) = \begin{bmatrix} (1.0 \times 10^{-7} \text{ rad})^2 & & & 0 \\ & (1.0 \times 10^5 \text{ rad/s})^2 & & \\ & & (1 \text{ dyne cm})^2 & \\ 0 & & & (50 \text{ dyne cm/s})^2 \end{bmatrix} \quad (7-37)$$

The maximum torquing capability of the simulated torque loop was 12 dyne-cm (1.5 earth rates).

The measurement noise $y(i)$ was assumed to be a zero-mean white noise sequence with covariance

$$R = (100 \mu\text{rad})^2 \quad (7-38)$$

The process noise $\underline{w}(i)$ was assumed to enter solely in the form of input-axis angular acceleration for the three state filter, and rate of change of acceleration (jerk) noise for the four state filter. It was also assumed to be a zero-mean white noise sequence with covariance matrix

$$Q = \begin{bmatrix} 0 & & & 0 \\ & 0 & & \\ & & 0 & \\ 0 & & & (960 \text{ dyne-cm/s})^2 \end{bmatrix} \quad (7-39)$$

The diagonal term in the 3 state filter was $(.05 \text{ dyne-cm})^2$.

Proper selection of values for the Q matrix is largely heuristic, and usually involves a certain amount of trial and error. Q , however, does directly influence the steady state performance of the filter by affecting the steady state values of the estimator gains. By varying the process noise model, we are essentially changing the eigenvalues or pole locations of the estimator error dynamics. For instance, if we assume no process noise is operating, then we might choose $Q = [0]$. The Kalman filter gains will then eventually approach zero, since the optimal thing to do in the face of zero disturbances and a perfect plant model is to ignore the noisy measurements and estimate open loop. One criteria helpful for arriving at values of Q , discussed in Reference [16], is to select noise levels which roughly match the possible range of state variation. Therefore if the k th state is likely to change by an amount Δx_k over an interval of interest Δt , then we could select,

$$\text{kth diagonal of } Q = \frac{\Delta x_k^2}{\Delta t} \quad (7-40)$$

Hence the value chosen in Equation (7-39), which represents a typical variation of τ' over a 52 μ sec observation interval.

Figures 7-4 and 7-5 are plots of the estimator gain versus time for the third state, external torque, obtained during torque loop simulations with two different values of $Q(4,4)$. It can be seen that the larger the assumed level of process noise, the larger the estimator gains are, reflecting a greater weighting on the measurements. Note again that time is measured in seconds giving rise to the large gain values. Using the covariance defined in this section, the steady-state gain matrix,

$$K = \begin{bmatrix} 0.067556 \\ 0.045358 \\ 3.4788 \\ 0.927 \end{bmatrix} \begin{array}{l} \text{rad/rad} \\ \text{rad/msec-rad} \\ \text{gm-cm/msec}^2\text{-rad} \\ \text{gm-cm/msec}^3\text{-rad} \end{array}$$

was chosen to represent a "slow" set of Kalman filter gains.

7.4.2.2 Estimator Gains by Direct Pole Placement

The steady-state estimator gain values achieved by the Kalman filter represent a balance between sensor and process noise. It is possible to assess this balance at the design stage and to precalculate the steady-state gains.

The error dynamics of the estimator will greatly affect the performance of the controller. Improvement in transient response is usually accompanied by an increase in estimator gains. Therefore by improving transient response, we are also trading off the estimator's ability to reject noise.

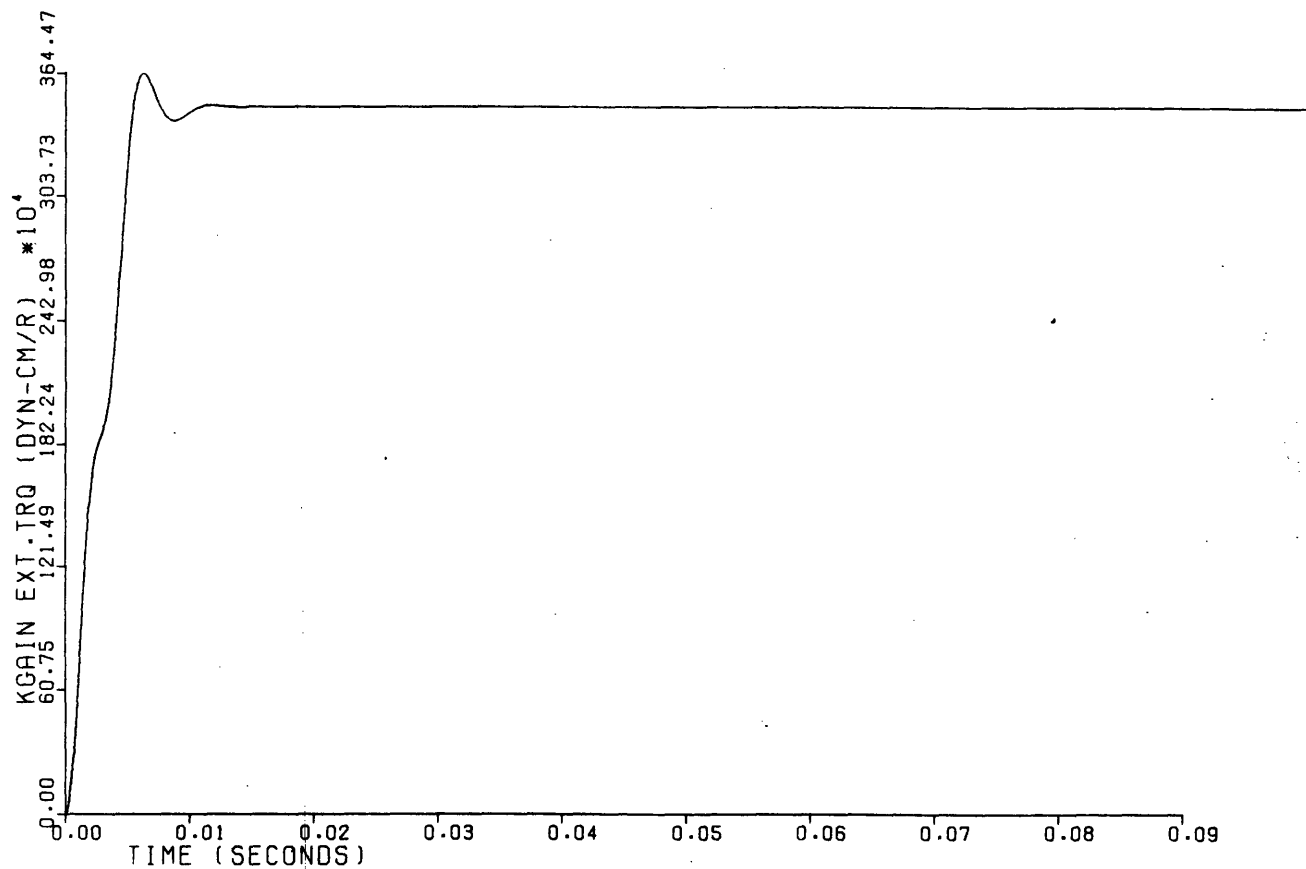


Figure 7-4. Estimator Gain on External Torque vs. Time for $Q(4,4) = 960 \text{ dyne-cm/s}^2$

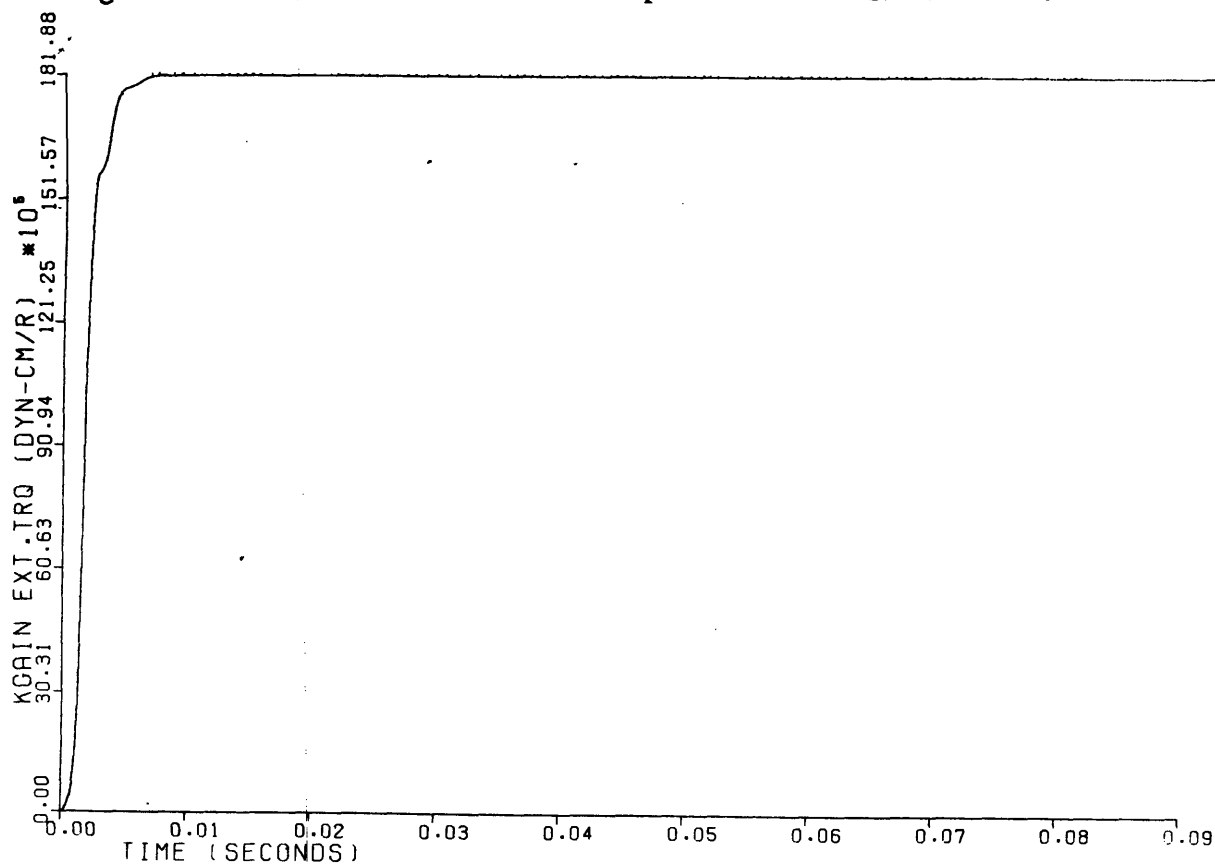


Figure 7-5. Estimator Gain on External Torque vs. Time for $Q(4,4) = 9,600 \text{ dyne-cm/s}^2$

A compromise has to be reached where the chosen L yields both an acceptable error transient response and acceptable noise rejection. Selecting L matrices using direct pole placement allows for a more direct manipulation of error dynamics. As discussed in Section 7.3.3, desired poles in the z -plane, with corresponding values of damping and natural frequencies, were used with Ackerman's formula to calculate the gain matrix L . L is then used to calculate the error system dynamics matrix $[\Phi - LC\Phi]$. A time response of the estimate error was plotted and evaluated, given initial errors in the estimator states.

"Fast" filter gains:

The following poles in the z -plane were selected which provided a fast transient response while still maintaining acceptable noise rejection. (See Chapter 15 for controller performance with varying SG noise levels.)

$$z_1, z_2 = .079395 \pm 0.22908 i$$

$$z_3, z_4 = 0.9141 \pm 0.058604 i \quad (7-42)$$

Using Ackerman's formula, the gain matrix L corresponding to the desired root locations is

$$L = \begin{bmatrix} 0.4 \\ 2.5374 \\ 437.58 \\ 461.00 \end{bmatrix} \begin{array}{l} \text{rad/rad} \\ \text{rad/msec-rad} \\ \text{gm-cm/msec}^2\text{-rad} \\ \text{gm-cm/msec}^3\text{-rad} \end{array} \quad (7-43)$$

By Equation (7-24) the set of difference equations describing the error transient response is

$$\tilde{\mathbf{x}}(i+1) = [\Phi - \mathbf{LC}\Phi] \tilde{\mathbf{x}}(i) \quad (7-44)$$

where the system dynamics matrix is given by (time in msec)

$$[\Phi - \mathbf{KC}\Phi] = \begin{bmatrix} 0.6 & 3.0539 \times 10^{-2} & 1.2926 \times 10^{-5} & 2.2526 \times 10^{-7} \\ -2.5374 & 0.8257 & 7.6628 \times 10^{-4} & 2.0590 \times 10^{-5} \\ -437.58 & -22.272 & 0.9906 & 5.1919 \times 10^{-2} \\ -461.0 & -23.464 & -9.9313 \times 10^{-3} & 0.9998 \end{bmatrix} \quad (7-45)$$

Figure 7-6 shows the error transient response of each state to an initial error in θ_1 , the float position. The settling time to within 10% is approximately 25 sampling periods.

"Medium" filter gains:

The poles of the fast filter gains were moved closer to the unit circle and with a reduced imaginary component. The result was a slowed error response with similar damping.

$$z_1, z_2 = 0.88948 \pm 0.09024 i \quad (7-46)$$

$$z_3, z_4 = 0.98027 \pm 0.029844 i$$

The corresponding gain matrix is

$$\mathbf{L} = \begin{bmatrix} 0.19485 \\ 0.39249 \\ 24.181 \\ 11.70 \end{bmatrix} \begin{matrix} \text{rad/rad} \\ \text{rad/msec-rad} \\ \text{gm-cm/msec}^2\text{-rad} \\ \text{gm-cm/msec}^3\text{-rad} \end{matrix} \quad (7-47)$$

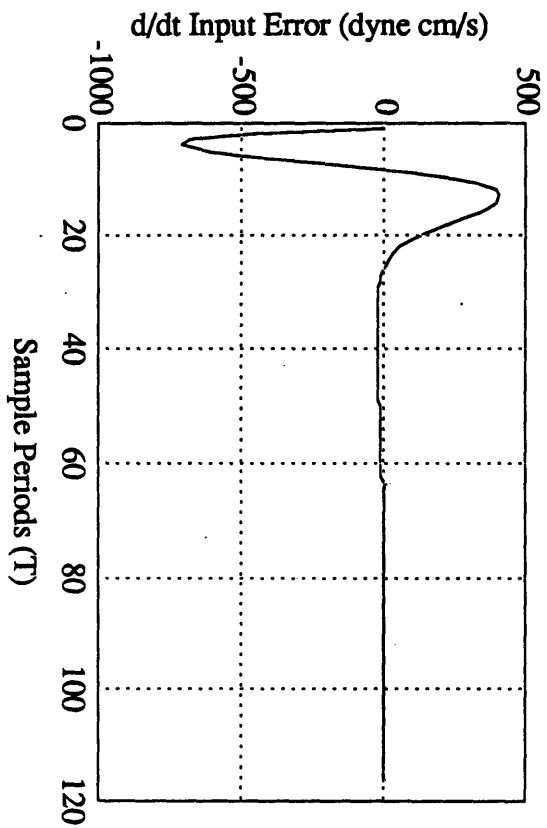
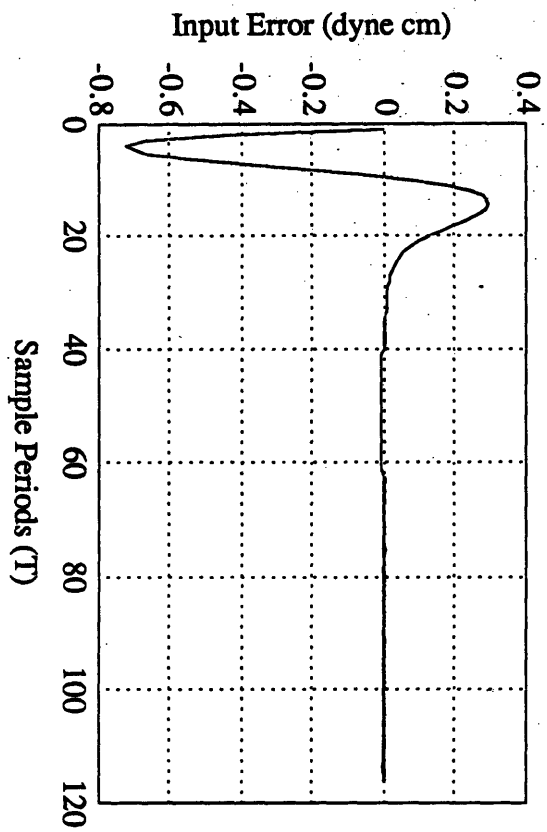
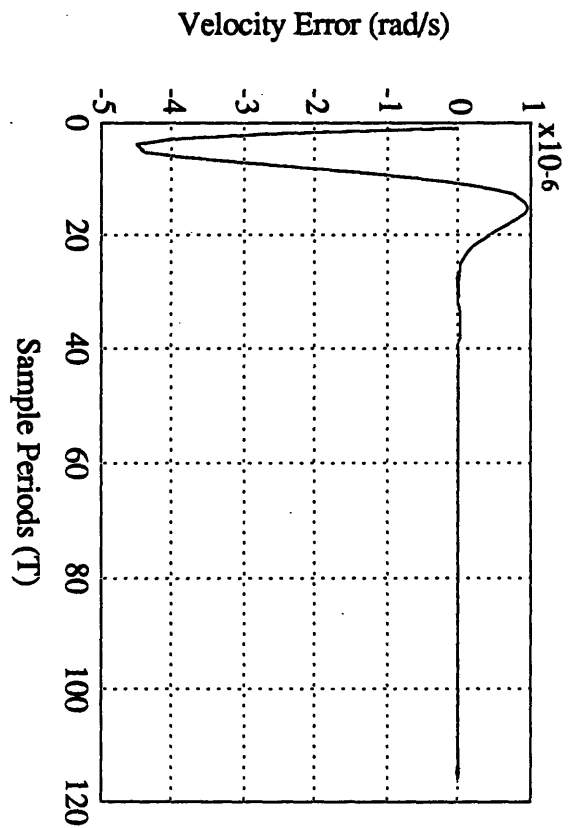
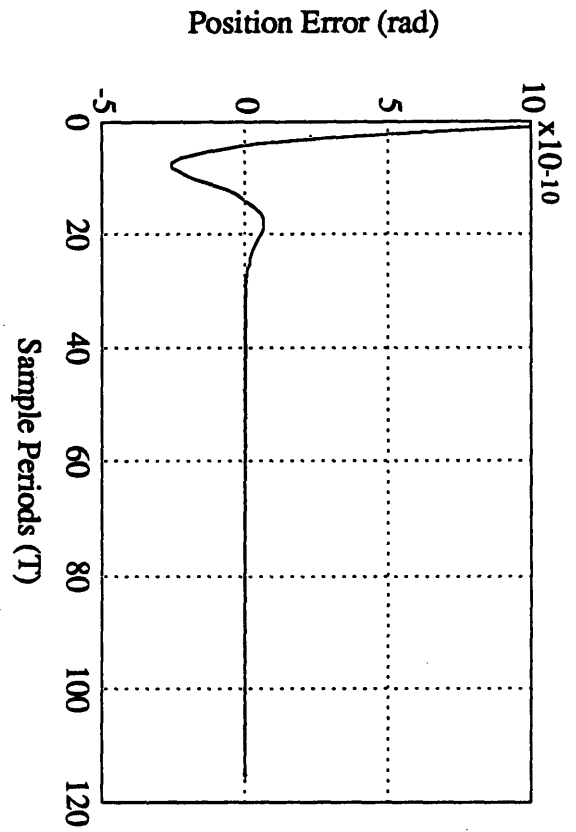


Figure 7-6. Error Transient Response for "Fast" Estimator Gains

The corresponding error dynamics matrix is

$$[\Phi - LC\Phi] = \begin{bmatrix} 0.8052 & 4.0981 \times 10^{-2} & 1.7345 \times 10^{-5} & 3.0229 \times 10^{-7} \\ -0.3925 & 0.9349 & 8.1248 \times 10^{-4} & 2.1396 \times 10^{-5} \\ -24.181 & -1.2308 & 0.9995 & 5.2074 \times 10^{-2} \\ -11.70 & -0.5951 & -2.5205 \times 10^{-4} & 1.00 \end{bmatrix} \quad (7-48)$$

Figure 7-7 shows the error transient response.

Table 7-1 is a tabulation of the three sets of gain matrices used for comparison of controller performance.

Table 7-1. "Fast", "Medium", and "Slow" Estimation Gains Used in Torque Loop Simulations

	<u>Estimator Gains</u>		
	"Fast"	"Medium"	"Slow"
L(1) rad/rad	0.4	0.19485	0.067556
L(2) rad/msec-rad	2.5374	0.39249	0.045358
L(3) gm-cm/msec ² -rad	437.58	24.181	3.4788
L(4) gm-cm/msec ³ -rad	461.00	11.70	0.927

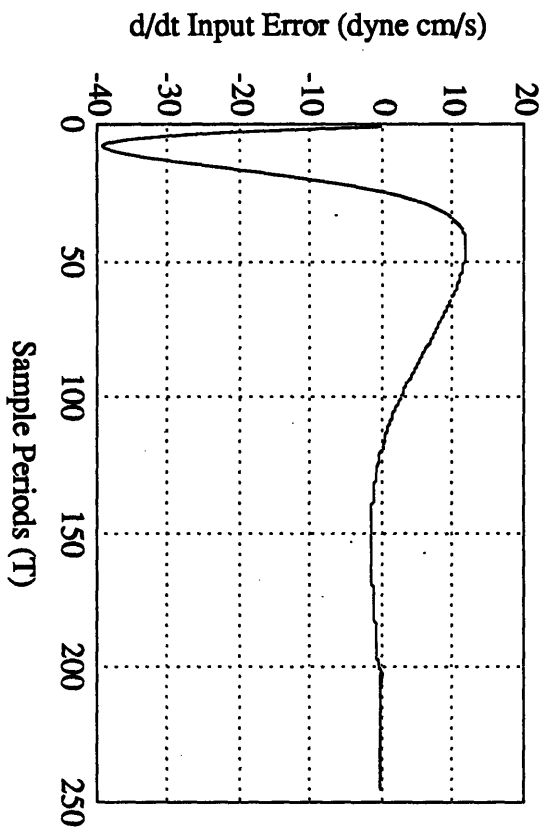
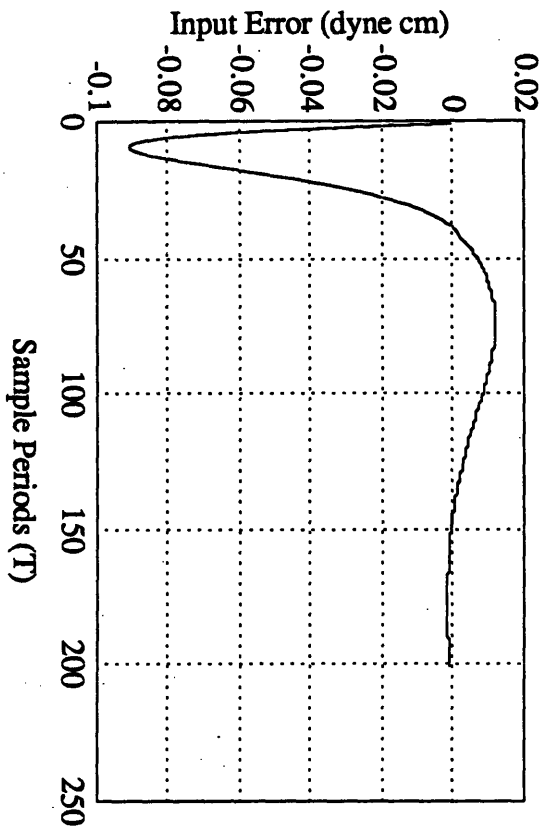
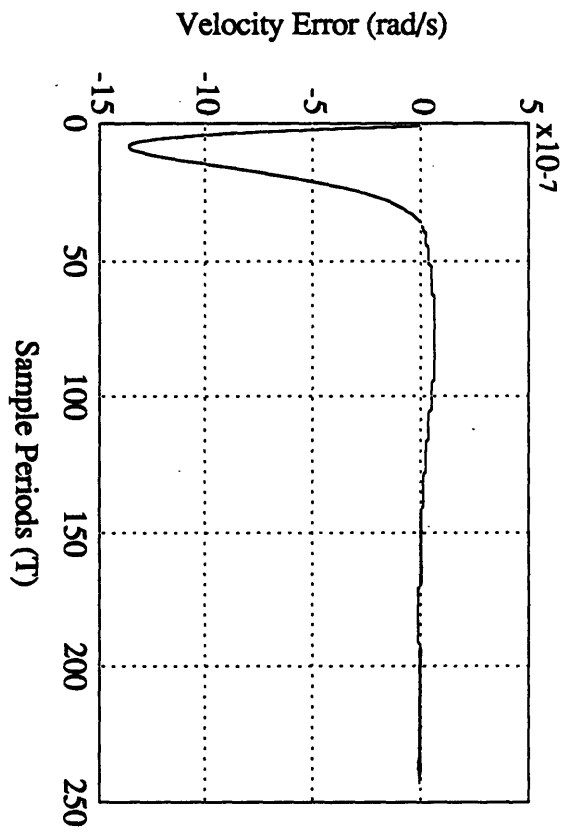
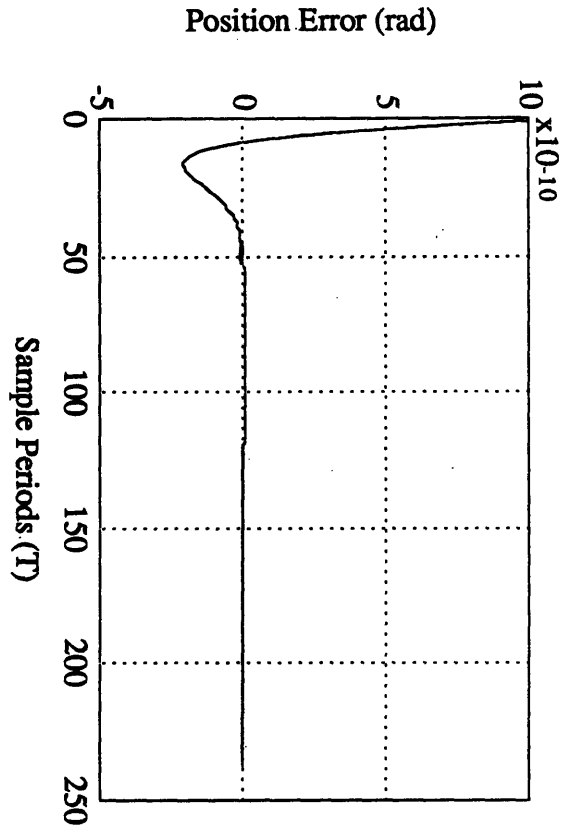


Figure 7-7. Error Transient Response for "Medium" Estimator Gains

8. NON-LINEAR CONTROLLER ALGORITHMS

8.1 Introduction

This chapter presents the derivations of the pulse width modulation controller algorithms used for simulation that ultimately would be implemented in a microprocessor. In each torque interval there is a positive and a negative torque pulse with net amount of torque in the interval being given by the relative widths of the two pulses. This relay switching of commanded torque represents the major non-linearity in the loop.

However, on either side of the torque transition the system can be considered a linear system. The controllers derived in this chapter are based on analytical solutions to the linear system dynamic equations with proper matching across the switching instants to insure continuity. Instantaneous torque transitions are assumed in the derivations, even though this is not exactly correct. The simulation software gave good results with the control laws derived herein, with finite simulated pulse rise and fall times.

Exact solutions are usually tedious to perform and also not easily suitable for standard compensation of the control system. However, with the advent of symbolic algebraic manipulation software and microprocessors of greater accuracy and speed, analytic solutions and their implementation are more easily achieved now than they have been in the past. In addition, since the Kalman filter is supposed to provide accurate state estimates, it seems appropriate to design a controller which uses all of these states for a control decision. There are three controller algorithms developed in this chapter. All three were derived with the two governing criteria of a good pulse-torquing rebalance loop in mind. These are,

- (1) Gyro float kept at null
- (2) Ensure no moding of the torquing pattern

All analysis is done in the time domain, mainly because observation of the moding pattern, which is observable in the time domain, is of major interest in the response of the controller. The first two controllers are derived from the estimated states provided by the three and four state Kalman filters. The final algorithm discussed was derived from the history of float position and previous torque pulses only. This algorithm began as a simple experiment but produced interesting results. It will be referred to as the simplified controller.

8.2 General Description of Control Algorithms Based on Kalman Filter State Estimates

8.2.1 Introduction

After twenty-four Kalman filter updates, the most recent state estimates of float position θ , float velocity $\dot{\theta}$, and external input τ (and $\dot{\tau}$ in the case of the four state Kalman filter) are made available for a control decision. There are two underlying steps towards a control decision. First, in order to ensure no moding and zero float hang-off, we use the estimated IA rate to determine a desirable steady-state torque pulse pattern and values of float position and velocity for the controller to aim for. Second, we decide what the next commanded torque, or switching instants, should be to take the current undesirable float states to the desirable states in the minimum time possible.

8.2.2 Position and Velocity States to be Sought

From the estimate of τ and possibly $\dot{\tau}$ and assuming the external inputs are no longer changing, the desired steady-state switching instant and resulting torquing pattern can be determined such that the integrated commanded torque over a single torque pulse most accurately represents the external input rate. This steady-state torquing pattern will be periodic at the torquing frequency with only slight variations due to quantization, and will not exhibit any other extraneous moding patterns.

When this steady-state torquing pattern is achieved, the float will oscillate at the torquing frequency. Thus the position and velocity of the float will repeat themselves at the

beginning and end of a torque pulse interval. In addition, the average position and velocity should be zero over a torque-pulse. The initial position and velocity states which satisfy both criteria are the states for the controller to aim for at the beginning of a torque-pulse. When the controller is successful in arriving at these states, the commanded torque will be the steady state torquing pattern calculated a priori.

8.2.3 Torque Pulses to Reach Desired States

In order to ensure zero moding and zero float hangoff, it is important to employ both position and velocity control of the float. Controlling two degrees of freedom, however, requires a minimum of two torque intervals for accomplishment. Once the desired steady state θ_o and $\dot{\theta}_o$ are determined, the controller decides what following combination of two torque words (switching instants) are required in order to take the present Kalman filter states θ , $\dot{\theta}$ to the desired states θ_o , $\dot{\theta}_o$ by the end of the second torquing interval. After the first pulse is executed, a check is made to see if the external inputs to the gyro have changed. If so, then a new set of θ_o , $\dot{\theta}_o$ are calculated and a new pair of torque words are determined and executed. If the external inputs have not changed appreciably after the first interval, then no new calculations are performed and the second previously calculated torque word is carried out.

Instead of calculating torque words in groups of two's, an equivalent method is to only calculate the first of the two switching instants and then to update this value after each torque interval. There are no advantages of either method over the other, except to say that the second method requires the same number of computations and thus the same amount of processor delay for every torque interval, whereas the first method does not.

8.3 Three State Kalman Filter Control Algorithm

8.3.1 Analytic Solution of Dynamic Equations

The first step in deriving the algorithm is to establish an analytical solution for the position and velocity of the float as a function of time over a complete torque interval.

The differential equations describing float motion over one torque interval are

$$\frac{d^2\theta}{dt^2} + \frac{C}{I} \frac{d\theta}{dt} = \frac{\tau + T_c}{I} \quad \text{for } 0 \leq t \leq x\Delta t \quad (8-1a)$$

$$\frac{d^2\theta}{dt^2} + \frac{C}{I} \frac{d\theta}{dt} = \frac{\tau - T_c}{I} \quad \text{for } x\Delta t \leq t \leq \Delta t \quad (8-1b)$$

where T_c is the amplitude of a positive or negative torque pulse and $\Delta t = 1/800$ sec is the torquing interval. The variable x between 0 and 1 is the fraction of a torque pulse period Δt where the torquer current switches. (See Figure 8-1).

Equation (8-1a) can be written as a system of two first order differential equations

$$\dot{\theta}_1 = \theta_2 \quad (8-2a)$$

$$0 \leq t \leq x\Delta t$$

$$\dot{\theta}_2 = -\frac{C}{I} \theta_2 + \frac{\tau + T_c}{I} \quad (8-2b)$$

where,

$$\theta_1 = \theta(t), \text{ and } \theta_2 = \dot{\theta}(t) = \dot{\theta}.$$

Equation (8-2b) is an ordinary first order differential equation of the form

$$\dot{u} - au = f(t) \quad \text{with } a = -C/I$$

This can be solved by looking for an integrating factor of e^{-at} , and rewriting the above equation as

$$e^{tC/I} \left(\dot{\theta}_2 + \frac{C}{I} \theta_2 \right) = e^{tC/I} \left(\frac{\tau + T_c}{I} \right) \quad (8-3)$$

The left hand side is an exact differential of $e^{tC/I} \theta_2$. We can integrate both sides of Equation (8-3) and solve for the velocity as a function of time from the beginning of a pulse interval to the torquer current switching instant.

$$\theta_2(t) = e^{\frac{-tC}{I}} \left[\theta_{2o} - \left(\frac{\tau - T_c}{C} \right) \right] + \left(\frac{\tau + T_c}{C} \right) \quad 0 \leq t \leq x \Delta t \quad (8-4)$$

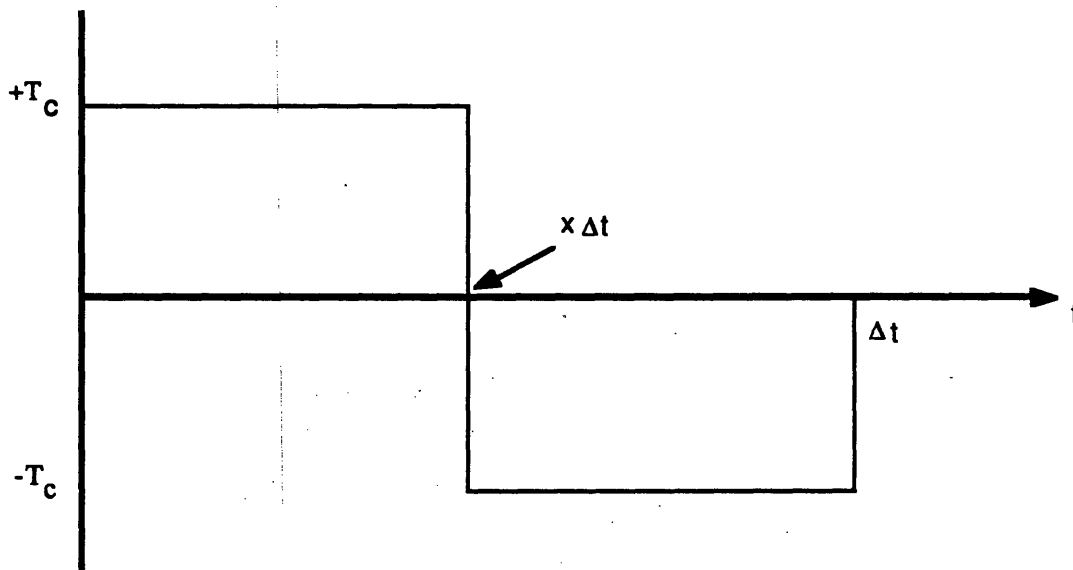


Figure 8-1. Pulse Width Modulation Square Wave Pulse Shape for Deriving Control Laws.

where θ_{20} is the initial float velocity at the beginning of a torque pulse.

From Equation (8-2a), the position as a function of time is

$$\theta_1(t) = -\frac{I}{C} e^{-tC/I} \left[\theta_{20} - \left(\frac{\tau + T_c}{C} \right) \right] + \frac{\tau + T_c}{C} t + G \quad 0 \leq t \leq x\Delta t \quad (8-5)$$

The constant of integration G is solved for by specifying the initial condition, $\theta_1(0) = \theta_{10}$, from which

$$G = \theta_{10} + \left[\theta_{20} - \left(\frac{\tau + T_c}{C} \right) \right] \frac{I}{C} \quad (8-6)$$

Equation (8-1b) is valid over the interval $x\Delta t \leq t \leq \Delta t$. The first order equations are now

$$\dot{\theta}_1 = \theta_2 \quad (8-7a)$$

$$x\Delta t \leq t \leq \Delta t$$

$$\dot{\theta}_2 = -\frac{C}{I} \theta_2 + \frac{\tau - T_c}{I} \quad (8-7b)$$

The form of the solutions will be similar. However, the initial conditions for this interval will be the final conditions from the previous interval. The solutions are

$$\theta_2(t) = \frac{\tau - T_c}{C} + \left[\theta_{20} - \left(\frac{\tau - T_c}{C} \right) + \frac{2T_c}{C} e^{x\Delta t C/I} \right] e^{-tC/I} \quad (8-8)$$

$$x\Delta t \leq t \leq \Delta t$$

$$\theta_1(t) = \frac{\tau - T_c}{C} t - \frac{GI}{C} e^{-tC/I} + \left[\theta_{1o} + \frac{2T_c}{C} x\Delta t + \left(\theta_{2o} - \left(\frac{\tau + T_c}{C} \right) \right) \frac{I}{C} + \frac{2T_c I}{C^2} \right] \quad (8-9)$$

8.3.2 Position and Velocity States to be Sought

As discussed in Section 8.2.2, the desired steady state position and velocity at the beginning of a torque interval can be determined by insisting that the average position and velocity are zero over the interval, and that the ending values are equal to the starting values. The first requirement can be written as

$$\frac{1}{\Delta t} \int_0^{\Delta t} \theta_2(t) dt = \frac{1}{\Delta t} \left[\int_0^{x_{ss}\Delta t} \theta_2(t) dt + \int_{x_{ss}\Delta t}^{\Delta t} \theta_2(t) dt \right] = 0 \quad (8-10)$$

$$\frac{1}{\Delta t} \int_0^{\Delta t} \theta_1(t) dt = \frac{1}{\Delta t} \left[\int_0^{x_{ss}\Delta t} \theta_1(t) dt + \int_{x_{ss}\Delta t}^{\Delta t} \theta_1(t) dt \right] = 0 \quad (8-11)$$

where x_{ss} is the steady state switching instant determined from the Kalman filter estimate of τ as

$$-\tau\Delta t = x_{ss}\Delta t T_c - (1-x)\Delta t T_c$$

or

$$x_{ss} = \frac{1}{2} - \frac{\tau}{2T_c} \quad (8-12)$$

Equations (8-10) and (8-11) were evaluated and solved to determine the initial steady state velocity θ_{2o}^{ss} and position θ_{1o}^{ss} , respectively. The general symbolic algebraic manipulation system called MACSYMA, which runs on the IBM mainframe computer, was used for this and subsequent formulas. The expressions for θ_{2o}^{ss} and θ_{1o}^{ss} are given by Equations (C-1) and (C-2) in Appendix C. These state values will repeat at the end of the interval since the integral of position and velocity is assumed zero and the steady state commanded torque is assumed to cancel the external input during one torque interval.

8.3.3 Torque Pulses to Reach Desired States

Given arbitrary Kalman filter state estimates θ_{1o} , θ_{2o} , which differ from the desired states θ_{1o}^{ss} , θ_{2o}^{ss} , the unknown combination of switching instants x_1 , x_2 must be determined such that by the end of the second pulse pair the desired state values are achieved. This is illustrated in Figure 8-2.

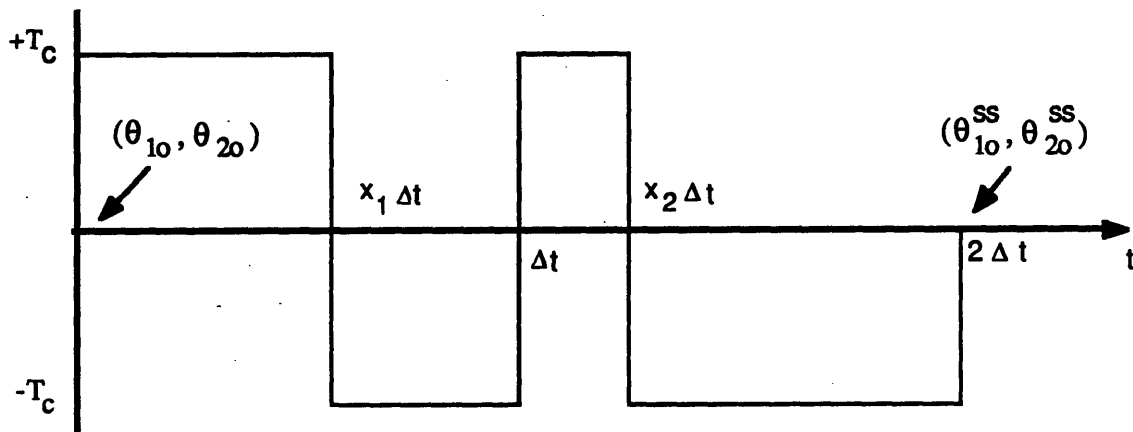


Figure 8-2. Two Switching Instants of Commanded Torque to Send Position and Velocity States to Desired Values

Using Equations (8-5) and (8-9) with the substituted values θ_{1o} , θ_{2o} and unknown x_1 , x_2 , an expression for float position by the end of the second pulse was determined. The expression for position after the first pulse is substituted as the initial condition for the second pulse. A similar expression for the velocity after two torque pulses was formed using Equations (8-4) and (8-8). The resulting expressions were set equal to the desired state values calculated a priori. This results in a system of transcendental equations in the unknowns x_1 and x_2 . See Equations (C-3) and (C-4) of Appendix C.

$$\theta_{2o}^{ss} = f(\theta_{1o}, \theta_{2o}, x_1, x_2, t) \quad (8-13a)$$

$$\theta_{1o}^{ss} = f(\theta_{1o}, \theta_{2o}, x_1, x_2, t) \quad (8-13b)$$

A closed form solution for x_1 and x_2 was not obtainable. However, a Newton iteration was used to rapidly converge to a solution from an initial guess. First, Equation (8-13a) was rearranged and solved for x_2 using MACSYMA:

$$x_2 = f(\theta_{1o}, \theta_{2o}, x_1, \theta_{2o}^{ss}, t) \quad (8-14)$$

See Equation (C-5) of Appendix C. This new expression was then substituted for x_2 in Equation (8-13b) and rearranged using MACSYMA into the form

$$g(\theta_{1o}, \theta_{2o}, \theta_{2o}^{ss}, x_1, t) = 0 \quad (8-15)$$

See Equation (C-6) of Appendix C. This is the non-linear equation coded in the simulation software and solved using Newton's method at each torque decision point during the simulation numerical integration.

8.3.4 Newton Iterative Solution

A derivation and discussion of Newton's method to solve non-linear equations can be found in [19].

Consider the non-linear equation of the form

$$g(x_1) = 0 \quad (8-16)$$

where x_1 is the only unknown. The solution can be reached by initially guessing a solution, then successively correcting this guess by the relation

$$x_1^{k+1} = \frac{-g(x_1)^k}{h(x_1)^k} + x_1^k \quad (8-17)$$

where g is the function and $h = dg/dx_1$ evaluated at the current approximation. This iterative process can be continued until the difference between successive values of x_1 are less than a specified value.

In order to insure stable and rapid convergence, a good initial guess of the solution is needed. The solution we are seeking, x_1 , is a real number between zero and one. Equation (8-15) is therefore evaluated five times over the interval (0,1) at steps of 0.25. If we plotted $g(x_1)$ versus x_1 , we are essentially looking for the point on the horizontal x axis where the curve intersects. Since we are only evaluating g at discrete points, we look for a change in sign of the function value between two successive evaluations. The solution then lies between the two points where the sign of g changed. A good initial guess is found by

extrapolating a line between the function values at the two points and selecting the value where the line intersects the x_1 axis as the initial guess.

This method was coded as part of the control algorithm in the simulation software. It provided efficient solutions of x_1 in four iterations or less and to four decimal place accuracy. After obtaining x_1 , substitution back into Equation (8-14) yielded x_2 in the software. Then x_1 , x_2 are multiplied by 256 and converted to integers with rounding to obtain the torque command word described in Sections 4.2.3 and 6.2.

8.4 Four State Kalman Filter Control Algorithm

Development of the four state algorithm procedures are similar to those used to derive the three state algorithm. The exception is that an extra integrator is placed on the third state τ , such that external inputs to the gyro are no longer modeled as constant, but are instead modeled as varying linearly over a torque pulse interval:

$$\tau(t) = \tau(0) + \dot{\tau}t \quad (8-18)$$

The fourth state $\dot{\tau}$ is assumed constant over a torque interval.

8.4.1 Analytic Solution of Dynamic Equations

The two first order differential equations including the new model of external inputs are written as

$$\dot{\theta}_1 = \theta_2 \quad (8-19a)$$

$$\dot{\theta}_2 = -\frac{C}{I} \theta_2 + \frac{\tau(0) + \dot{\tau}t \pm T_c}{I} \quad (8-19b)$$

Considering first the interval of $0 \leq t \leq x\Delta t$, (8-19b) is of the form $u' - au = f(t)$ with $a = -C/I$. Again we find as an integrating factor e^{-at} to multiply both sides of (8-19b).

$$e^{tC/I} \left(\dot{\theta}_2 - \frac{C}{I} \theta_2 \right) = e^{tC/I} \left(\frac{\tau + \dot{\tau}t + T_c}{I} \right) \quad (8-20)$$

Again noticing that the left hand side is an exact differential of $e^{tC/I} \theta_2$, we can integrate both sides of Equation (8-20):

$$e^{tC/I} \theta_2 \Big|_0^t = \int_0^t e^{sC/I} \left(\frac{\tau + T_c}{I} + \frac{\dot{\tau}s}{I} \right) ds \quad (8-21)$$

The right hand side can be integrated by parts and Equation (8-21) can be solved for θ_2 as a function of time.

$$\theta_2(t) = \frac{\dot{\tau}}{C} t + e^{-tC/I} \left[\theta_{20} + \frac{\dot{\tau}I}{C^2} - \left(\frac{\tau + T_c}{C} \right) \right] + \left(\frac{\tau + T_c}{C} \right) - \frac{\dot{\tau}I}{C^2} \quad 0 \leq t \leq x\Delta t \quad (8-22)$$

Equation (8-19b) is integrated to find θ_1 as a function of time.

$$\theta_1(t) = \frac{\dot{\tau}}{2C} t^2 + e^{-tC/I} \left[\theta_{20} + \frac{\dot{\tau}I}{C^2} - \left(\frac{\tau + T_c}{C} \right) \right] + \left[\left(\frac{\tau + T_c}{C} \right) - \frac{\dot{\tau}I}{C^2} \right] t + B \quad 0 \leq t \leq x\Delta t \quad (8-23)$$

Using the initial condition,

$$\theta_1(0) = \theta_{10} = \frac{-I}{C} \left[\theta_{20} + \frac{\dot{\tau}I}{C^2} - \left(\frac{\tau + T_c}{C} \right) \right] + B$$

B is solved for as

$$B = \theta_{10} + \frac{I}{C} \left[\theta_{20} + \frac{\tau I}{C^2} - \left(\frac{\tau + T_c}{C} \right) \right] \quad (8-24)$$

Over the interval $x\Delta t \leq t \leq \Delta t$ the solutions will again be similar to Equations (8-22) and (8-23) with θ_{20} and θ_{10} equal to Equations (8-22) and (8-23) evaluated at $x\Delta t$, respectively.

$$\theta_2(t) = \frac{\tau - T_c}{C} + \frac{\dot{\tau}}{C} t - \frac{\dot{\tau}I}{C^2} + E e^{-tC/I} \quad x\Delta t \leq t \leq \Delta t \quad (8-25)$$

The constant E is obtained by equating Equations (8-22) and (8-25) evaluated at $t = x\Delta t$:

$$E = \theta_{20} + \frac{\dot{\tau}I}{C^2} - \left(\frac{\tau + T_c}{C} \right) + \frac{2T_c}{C} e^{x\Delta t C/I} \quad (8-26)$$

$$\theta_1(t) = \left[\left(\frac{\tau - T_c}{C} \right) - \frac{\dot{\tau}I}{C^2} \right] t + \frac{\dot{\tau}}{2C} t^2 - \frac{EI}{C} e^{-tC/I} + F \quad x\Delta t \leq t \leq \Delta t \quad (8-27)$$

To obtain the constant F, Equations (8-23) and (8-27) are evaluated at $t = x\Delta t$ and equated:

$$F = \theta_{1o} + \frac{2T_c}{C} x\Delta t + \frac{I}{C} \left[\theta_{2o} + \frac{\dot{\tau}I}{C^2} - \left(\frac{\tau + T_c}{C} \right) + \frac{2T_c}{C} \right] \quad (8-28)$$

8.4.2 Position and Velocity States to be Sought

The procedures for determining the desired steady state position and velocity θ_{1o} , θ_{2o} proceed as in Section 8.3.2 using Equations (8-10) and (8-11). Since τ is assumed varying linearly over a torque pulse, the steady state switching instant x_{ss} is found using the average value of τ over a torque interval. With a torque period of 1.25 msec,

$$x_{ss} = 0.5 - \frac{\tau + \dot{\tau}(0.625 \text{ msec})}{2T_c} \quad (8-29)$$

The expressions for θ_{1o} , θ_{2o} , are written as Equations (C-7) and (C-8), respectively, in Appendix C.

8.4.3 Torque Pulses to Reach Desired States

The procedure outlined in Section 8.3.3 for determining the combination of switching instants x_1 , x_2 needed to bring the float to the desired states θ_{1o}^{ss} , θ_{2o}^{ss} in two torque pulses is identical to those required here. The two equations, derived similarly to Equations (8-13a,b) using Equations (8-22) -(8-28), are written as Equations (C-9) and (C-10) in Appendix C. Similarly the equation solving Equation (C-9) for x_2 and the non-linear equation solving for x_1 using a Newton iteration are represented as Equations (C-11) and (C-12), respectively, also in Appendix C.

8.5 Simplified Controller

8.5.1 Introduction

Instead of basing a control algorithm on 3 or 4 state Kalman filter estimates, one could attempt to derive a control law from the previous history of the average of the OA angle observables and the last torque word used. An ad hoc approach originated by M. Ash and verified using the simulation software has in fact yielded such a control law which behaves well in keeping the float oscillating about null when there is damping in the dynamic model.

For constant input the oscillation is just due to the torque pulses and is similar to the steady state oscillations in a Kalman filter based controller. However, in the face of a step input the simplified control law took a somewhat longer time to bring the float back to null compared to the Kalman filter control laws.

The simplified control law did not keep the float near null if the fluid damping C were zero, which is strange because the ad hoc derivation of the simplified control algorithm assumed that C was zero. The damping existing in an actual floated gyroscope apparently makes for more robust behavior of the simplified control law.

8.5.2 Extrapolation of Average OA Angle Observable

The OA angle is observed n times every $\Delta t = 1/800$ sec torquing interval, where for the simulations run for this thesis $n=24$. Suppose we are at a $1/800$ sec time point t_0 , and let $\theta_0, \theta_{-1}, \theta_{-2}$ be the average values of the 24 OA angle measurements in the past 3 torquing intervals. If the widths of the positive and then negative torque pulses in a $\Delta t = 1/800$ sec torquing interval were in fact what was required to keep the float at null, then the θ_j would be zero.

What would the average OA angle θ_1 be between times t_0 and $t_0 + \Delta t$ if there were no change from the torquing used in the intervals for which $\theta_0, \theta_{-1}, \theta_{-2}$ resulted? We could

suppose that θ_1 is the value of the quadratic polynomial at time $t_0 + \Delta t$ that passes through the tabulated points θ_j at times $t_0 + j\Delta t$ ($j=0, -1, -2$). The coefficients of this quadratic polynomial are

$$\begin{aligned} A_0 &= \theta_0 \\ A_1 &= \left(\frac{1}{2} \theta_{-2} - 2\theta_{-1} + \frac{3}{2} \theta_0 \right) / \Delta t \\ A_2 &= \left(\frac{1}{2} \theta_{-2} - \theta_{-1} + \frac{1}{2} \theta_0 \right) / \Delta t^2 \end{aligned} \quad (8-30)$$

so that

$$\begin{aligned} \theta_1 &= A_0 + A_1 \Delta t + A_2 \Delta t^2 \\ &= 3\theta_0 - 3\theta_{-1} + \theta_{-2} \end{aligned} \quad (8-31)$$

8.5.3 Additional Torque Required to Force the Average OA Angle to Zero

Besides the previously used torque, suppose there is an additional average amount T_{av} applied between time t_0 and $t_0 + \Delta t$ to force θ_1 to zero:

$$0 = A_0 + A_1 \Delta t + \left(A_2 + \frac{1}{2} \frac{T_{av}}{I} \right) \Delta t^2 \quad (8-32)$$

where with damping ignored distance $= \frac{1}{2} (\text{acceleration}) \Delta t^2$. Solving for T_{av} yields

$$T_{av} = - \frac{2\theta_1 I}{\Delta t^2} \quad (8-33)$$

In the interval Δt let the torque be $+T_c$ for time $x\Delta t$ and $-T_c$ for time $(1-x)\Delta t$, where $0 \leq x \leq 1$. Then

$$\begin{aligned} T_{av} &= T_c(x) - T_c(1-x) \\ &= T_c(2x-1) \end{aligned} \quad (8-34)$$

Combining Equations (8-33) and (8-34) yields

$$x = \frac{1}{2} \left(1 - \frac{2\theta_1 I}{\Delta t^2 T_c} \right) \quad (8-35)$$

8.5.4 Quantized Simplified Control Algorithm

Quantize the fraction x into an integer N by

$$N = 256x + 0.5 \quad (8-36)$$

where in the Fortran simulation software adding 0.5 before truncation to N provides rounding.

To handle the minimum allowed pulse width set

$$N = \begin{cases} 5 & \text{if } N < 5 \text{ or } x < 0.0195 \\ 250 & \text{if } N > 250 \text{ or } x > 0.9765 \end{cases} \quad (8-37)$$

and use the maximum amount of net positive or negative torquing

If $5 < N < 250$, let

$$M = N - 128 \quad (8-38)$$

Let N_{old} be the torque word from the previous torque interval and let the torque word for the new torque interval be

$$N = N_{old} + M \quad (8-39)$$

where correction (8-37) is applied if necessary to obtain $5 < N \leq 250$, where the torque word N is the width of the positive pulse and $256-N$ is the width of the negative pulse, both in time quantization units of $\Delta t/256 = 4.8828125 \mu\text{sec}$.

All this is very ad hoc, but this simplified control law works, as described in Part III.

9. DESCRIPTION OF SIMULATION SOFTWARE

9.1 Macintosh Software

The Macintosh application software package MATLAB for performing matrix operations and its associated control theory toolbox were used to assist in the development of the estimator. Direct pole placement and calculation of gain matrices was easily and accurately performed. In addition, this software was used to simulate the error transient response to initial estimate errors and produced the plots of errors versus time presented in Chapter 7.

9.2 IBM Mainframe Software

9.2.1 Executive Control

Software to numerically integrate the true OA float motion versus time and propagate the Kalman filter estimated motion was written in Fortran on the Draper Laboratory IBM 3090 mainframe computer by M. Ash and the author. A block diagram of the software is given in Figure 9-1.

The software was written in modular form with each subroutine being a separate member of the partitioned data set

PAD1182.TRQLOOP.FORT

The labeled commons containing variables passed between subroutines are members in the partitioned data set

PAD1182.TRQLOOP0.FORT

They are included in the compilation of the TRQLOOP subroutines by INCLUDE statements in the subroutines and a TSO command

"ALLOCATE FI(SYSLIB) DA(TRQLOOP0.FORT)"

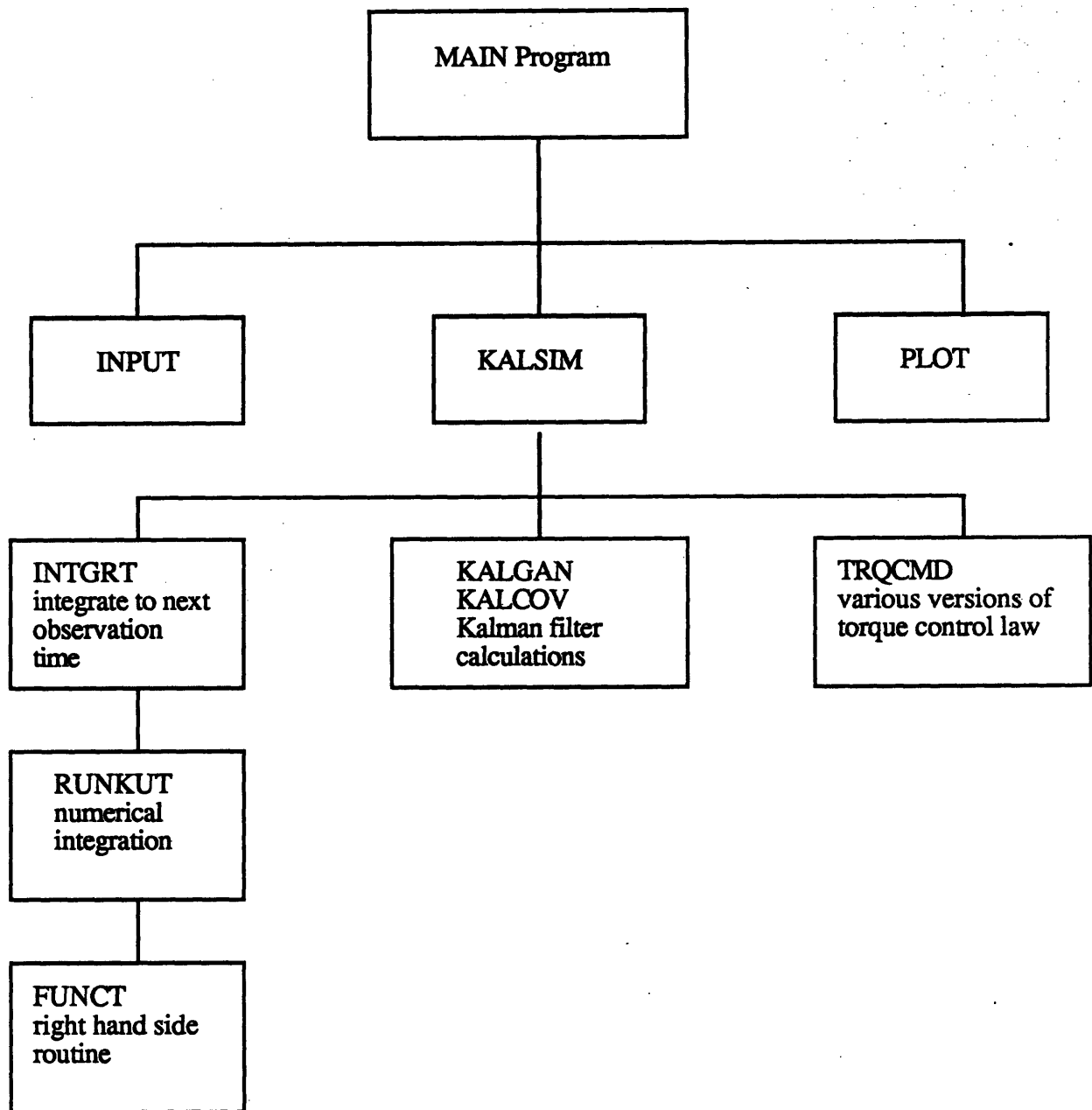


Figure 9-1. Flow Chart of IBM 3090 Mainframe Computer Simulation Software

The Job Control statements for making a load module and executing simulation runs are in the partitioned data set

PAD1182.TRQLOOP.CNTL

There are 30 subroutines and 5 labeled commons totalling over 3,000 lines of code in the program (not including plotting routines). An IMSL random number generator subroutine is also called.

The MAIN program calls subroutine INPUT to initialize and read the input control and data parameters. There is an A-format title and then Namelist input, so only parameters that differ from their initialized values need be input.

Next the MAIN program calls KALSIM to propagate the dynamic system motion and simulate the Kalman filter and control law behavior. When the simulation is completed, the plot software reads the output file of the simulation and generates graphs of the various states and other quantities versus time and each other. (The plot software is actually executed separately from TRQLOOP rather than being called from the MAIN program of TRQLOOP.)

9.2.2 KALSIM Control Loops

Numerical integration and Kalman filter quantities are initialized. Then there is an outer loop in steps of 1/800 sec until the ending time of the simulation computer run. At each step of the outer loop there are 24 observation time steps of $1/(24 \times 800)$ sec = 52.083 μ sec each. Time propagation uses 32 decimal place quadruple precision arithmetic, whereas the rest of the software uses 16 decimal place double precision arithmetic.

From one observation time to the next the true states, the Kalman filter estimated states, and the state transition matrix are numerically integrated by calling subroutine INTGRT. The OA angle observable is taken to be the true OA angle integrated state with the options of adding random number generated Gaussian noise and quantization noise.

The OA angle observable is used to update the Kalman filter estimated states every 52.083 μ sec. Then after 24 of these updates, the control law algorithm being simulated is applied to the Kalman filter estimated states (either the 3 state or 4 state model) to derive the commanded pulse width modulated torque for the next 24 observation times. If the simplified control law is being simulated, then the Kalman filter is skipped and the commanded torque is derived from the previous three sets of averages of 24 OA angle observations and the previous commanded torque.

An output file record is written every 52.083 μ sec with the true and Kalman filter estimated states, the actual input angular velocity or torque, and the torque loop commanded torque for later plotting.

9.2.3 INTGRT State and Transition Matrix Propagation

Every 52.083 μ sec subroutine KALSIM calls subroutine INTGRT, which calls the Runge-Kutta numerical integration subroutine RUNKUT to propagate the true and Kalman filter estimated states and the state transition matrix from one observation time to the next. Two Runge-Kutta steps are taken for the 52.083 μ sec observation interval, unless the interval contains a pulse transition, in which case 32 Runge-Kutta steps are used to cover the 52.083 μ sec interval.

RUNKUT calls function FUNCT to evaluate the right sides of the differential equations being numerically integrated. FUNCT in turn calls function RATIN to calculate the input angular velocity, which could be constant, a step, a pulse, a ramp, or a sinusoid of given amplitude and frequency, depending on the input options chosen for the simulation run. FUNCT also calls function TRQPLS to calculate the commanded torque at the given integration time given the torque decision made at the previous 1/800 sec decision time.

9.2.4 Kalman Filter Propagation

Every 52.083 μ sec subroutine KALSIM calls matrix manipulation subroutines and subroutines KALGAN and KALCOV to calculate the Kalman filter gain matrix, update the Kalman filter state estimates using the OA angle observable and the INTGRT propagated state and transition matrix, and update the state covariance matrix.

9.2.5 TROCMD Controller Decisions

Every 1/800 sec subroutine KALSIM calls one of various versions of subroutine TRQCMD to set up a labeled common with the next 1/800 sec commanded torque pulse information to be used in subroutine TRQPLS.

9.2.6 PLOT Options

At the end of a simulation computer run the PLOT program is called to read the file written in the simulation run, and plot versus time the true states and input angular velocity, the Kalman filter estimated states, the difference between these two vectors, and the commanded torque. The true or Kalman filter estimated OA velocity can also be plotted versus the OA position (phase state plot).

PART III. SIMULATION RESULTS

10. DESCRIPTION OF SIMULATION SCENARIOS

Part III (Chapters 10 to 15) presents the simulation runs illustrating performance of the control algorithms developed in Chapter 8. The runs presented in this thesis are by no means exhaustive. They were chosen for their ability to illustrate clearly the strengths and weaknesses of each controller. In addition, they were selected to represent approximations to realistic inputs experienced by inertial guidance systems for applications such as pointing and tracking and strap-down navigation. The following paragraphs give a brief description of the run scenarios selected and their purpose. Many other runs were performed which provided additional insights into the torque loop behavior, although they are not formally presented here.

Chapter 11 shows the controller responses to constant zero and nonzero input rates applied about the gyro input axis. The purpose was to observe the steady state moding pattern and look for any steady state float offset from signal generator null. For constant input rates, the three state and four state controllers behaved identically. In addition, except for initial transients, all three sets of estimator gains ("fast", "medium", and "slow") also produced the same results. Therefore, only one run is shown to represent all of the Kalman filter based controllers response to a constant input. The simplified controller performance is also presented here and in all the chapters.

Chapter 12 shows the responses to a step input which occurs half way through the simulation run period (0.1 sec). A step input provided insights into the transient response of the controllers. It was of particular interest to observe the settling time (how quickly the float was brought back to null and returned to steady state oscillation) and the stability (did the float return to the minimum mode of oscillation and does the torquing pattern represent the negative of the external input with minimum quantization). In addition, in steady state the Kalman filter outputted estimates of external input rate with an accuracy matching the eight decimal places allocated in the computer print output. This is of course expected since

both the actual and Kalman filter models were identical, and no noise was added to the simulated SG observable, except for the runs discussed in Chapter 15.

Chapter 13 illustrates the responses to a positive ramp input from zero, reaching the maximum torquing capability by the end of the simulation run period. A ramp tested the ability of the estimators to track a varying input rate, and observations of any steady state estimator errors were made. Also observed were float hang-off, whether the float maintained minimum mode of oscillation, and if any moding of the torquing pattern occurred.

Chapter 14 presents the responses to a 10 Hz sinusoidally varying input rate with an amplitude equal to half the torquing capability. The rapidly varying input represents a more severe test of the estimator's tracking ability. The difference in performance between the three and four state estimators at the three sets of estimator gains is more apparent. Again each run was examined for float modes of oscillation, how well the float was kept at null, and modes of the torquing pattern.

Chapter 15 shows the runs which investigated the performance of the torque loop with the addition of signal generator noise included in the measurements. Three different levels of noise were implemented on top of a constant non-zero input rate. The noise responses were similar for both the three state and four state controllers, thus again plots for both runs are not duplicated here.

All plots exhibited initial transients due to initial estimator errors. These transients should be ignored since we are interested in the behavior of the torque loop in steady state operation to various input rates.

11. STEADY STATE BEHAVIOR

11.1 Zero Input Response

11.1.1 Results Summary

Table 11-1 shows the summary of results for the unforced system.

Table 11-1. Summary of Results for Input Rate of $0^\circ/\text{hr}=0$ dyne-cm

RUN #	Description	Ave. float hang-off (nrad)	Steady state torquing pattern between 5 and 250
1	3 and 4 state controllers for all gains	-0.5327	128 with no fluctuation
2	simplified controller	+0.5306	128 ± 1 fluctuation

Both controllers exhibited a torque moding pattern achieving minimum quantization. Associated with that was a stable minimum mode of float oscillation of the forcing frequency. In addition, both controllers experienced a negligible hang-off of approximately 0.5 nrad.

The estimator based controller did not fluctuate about the steady state torque word. This will occur when the input torque is zero or an exact multiple of $T_C/128$. The controller should theoretically be able to precisely counteract the external input and keep the average float position accurately at null. The data, however, shows a slight hang-off and no attempt is made for correction. Thus the controller did not reach its sought after states exactly. Although the algorithm may calculate a slightly adjusted torque switching instant required to null out the slight hang-off, if the amount of adjustment is less than half of one quantum of torque (1 part in 256×12 dyne-cm) then the switching instant will be rounded back to the same torque word. Since the external input is zero, there will be no tendency

for the error between the external input and the commanded input to grow. Thus the float will never receive an added quantum of torque in the direction needed to null out the slight float offset. The simplified controller is based on the average float position and therefore does attempt to correct itself by adding and subtracting one quantum of correcting torque accordingly. However, an average hang-off of about 0.5 nrad still seems to persist.

11.1.2 Simulation Plots

Figures 11-1 through 11-10 present the plots for the responses to zero input.

11.2 Non Zero Constant Input

11.2.1 Results Summary

Table 11-2 shows a summary of results for the response to a non-zero constant input.

Table 11-2. Results Summary for Constant Input of $7.5^\circ/\text{hr} = 3.792 \text{ dyne-cm}$

RUN #	Description	Ave. float hang-off (nrad)	Steady state torquing pattern between 5 and 250
3	3 and 4 state controllers for all gains	-0.603	fluctuated between 87 and 88
4	simplified controller	+0.152	fluctuated between 87 and 88

Both controllers maintained the minimum mode of float oscillation and a steady state torquing pattern depicting the external input with minimum quantization.

This time the torque word of the Kalman filter controller did oscillate between 87 and 88. This is because the input was not an exact multiple of $12 \text{ dyne-cm}/128$. Thus the

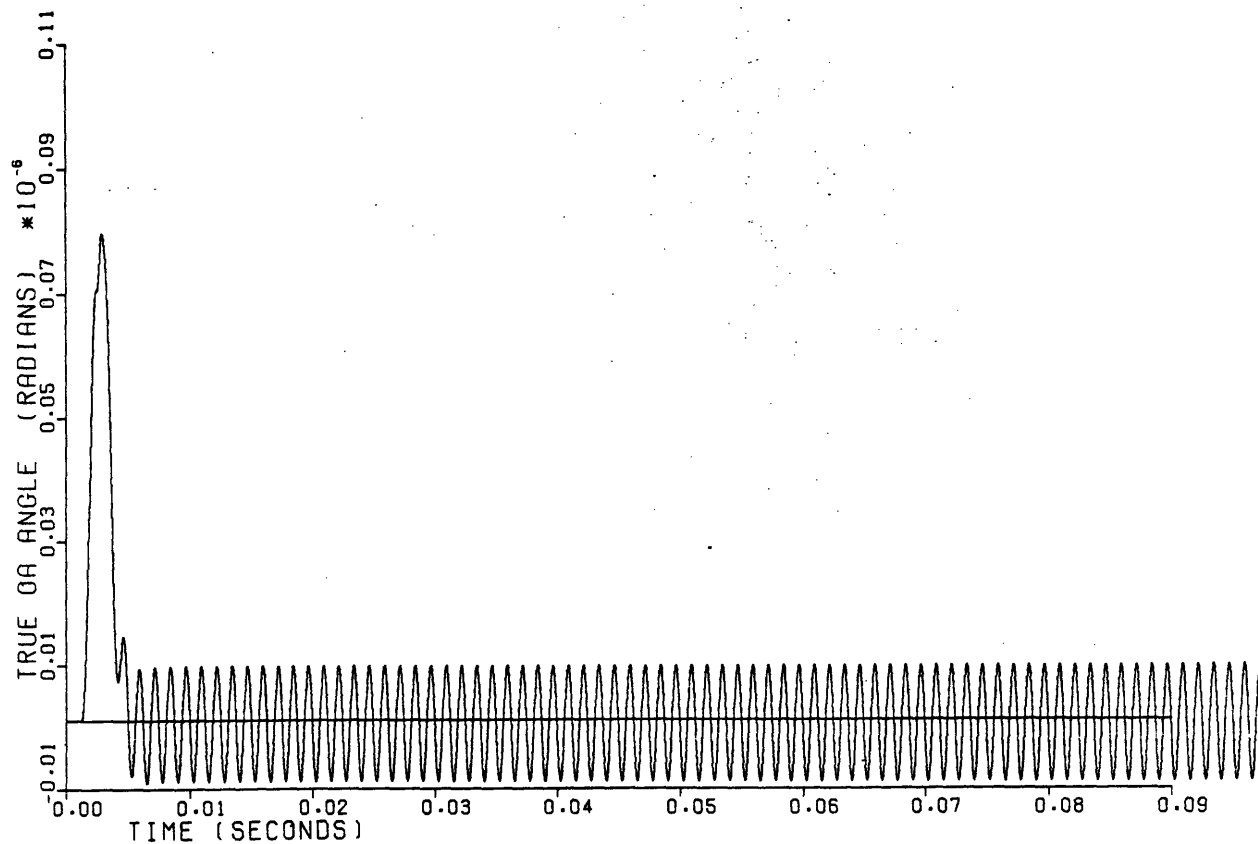


Figure 11-1. Run #1 Float Position vs. Time (Zero Input)

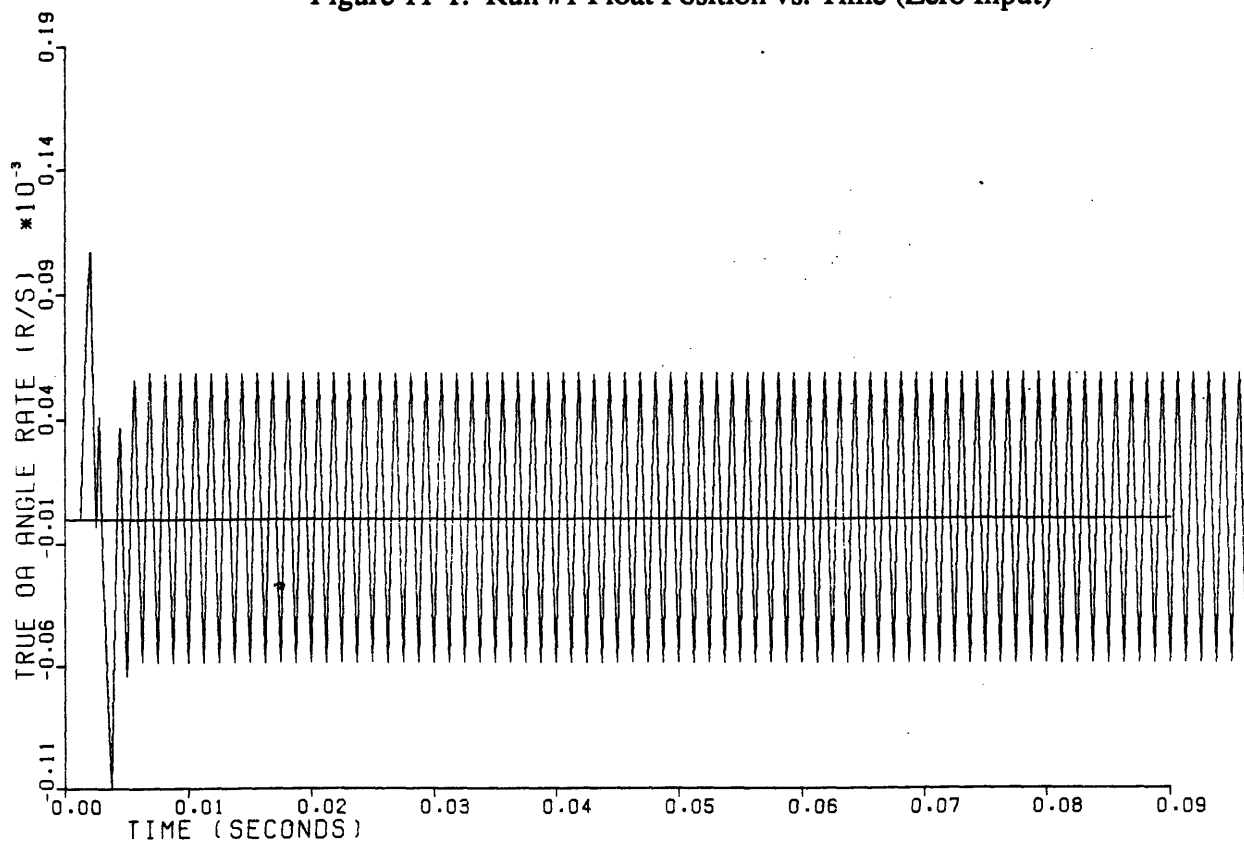


Figure 11-2. Run #1 Float Velocity vs. Time (Zero Input)

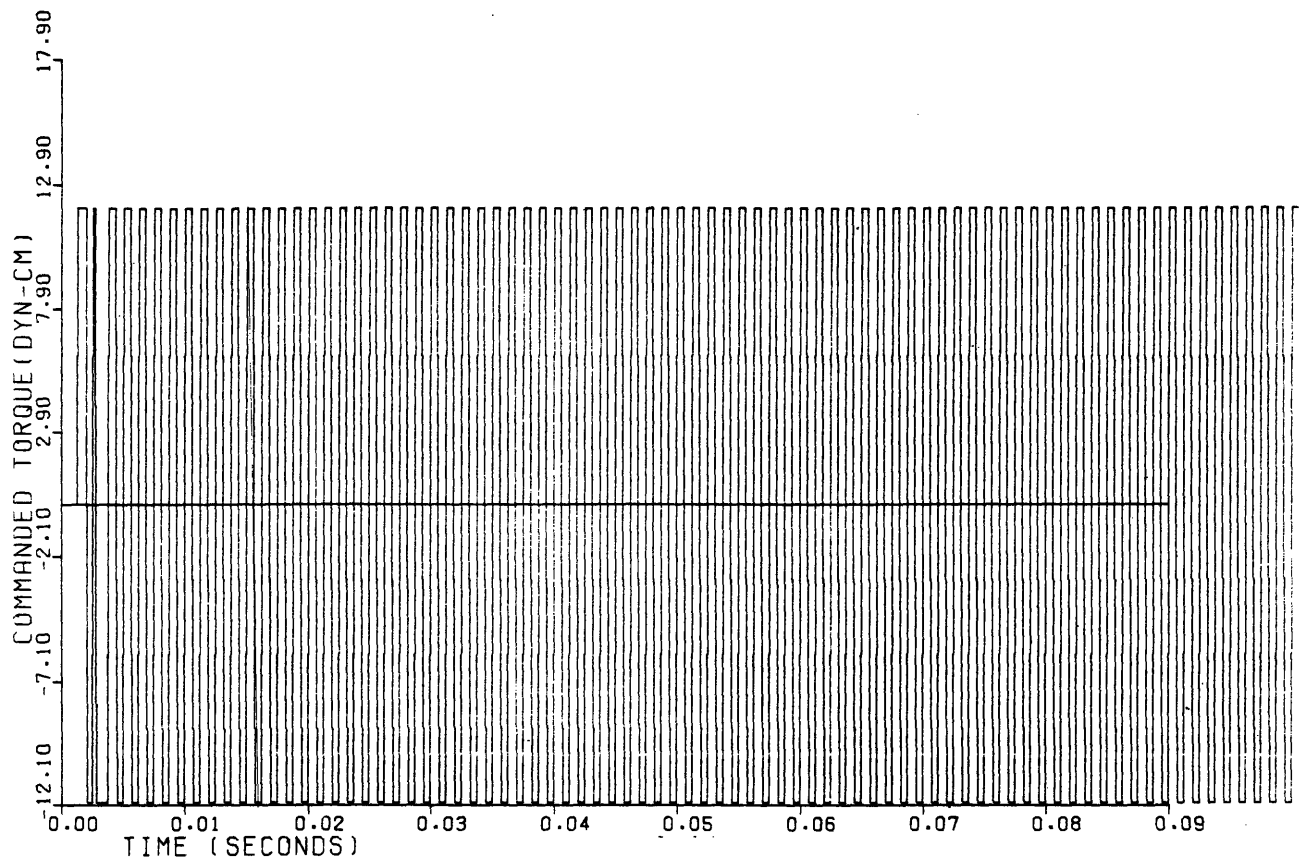


Figure 11-3. Run #1 Commanded Torque vs. Time (Zero Input)

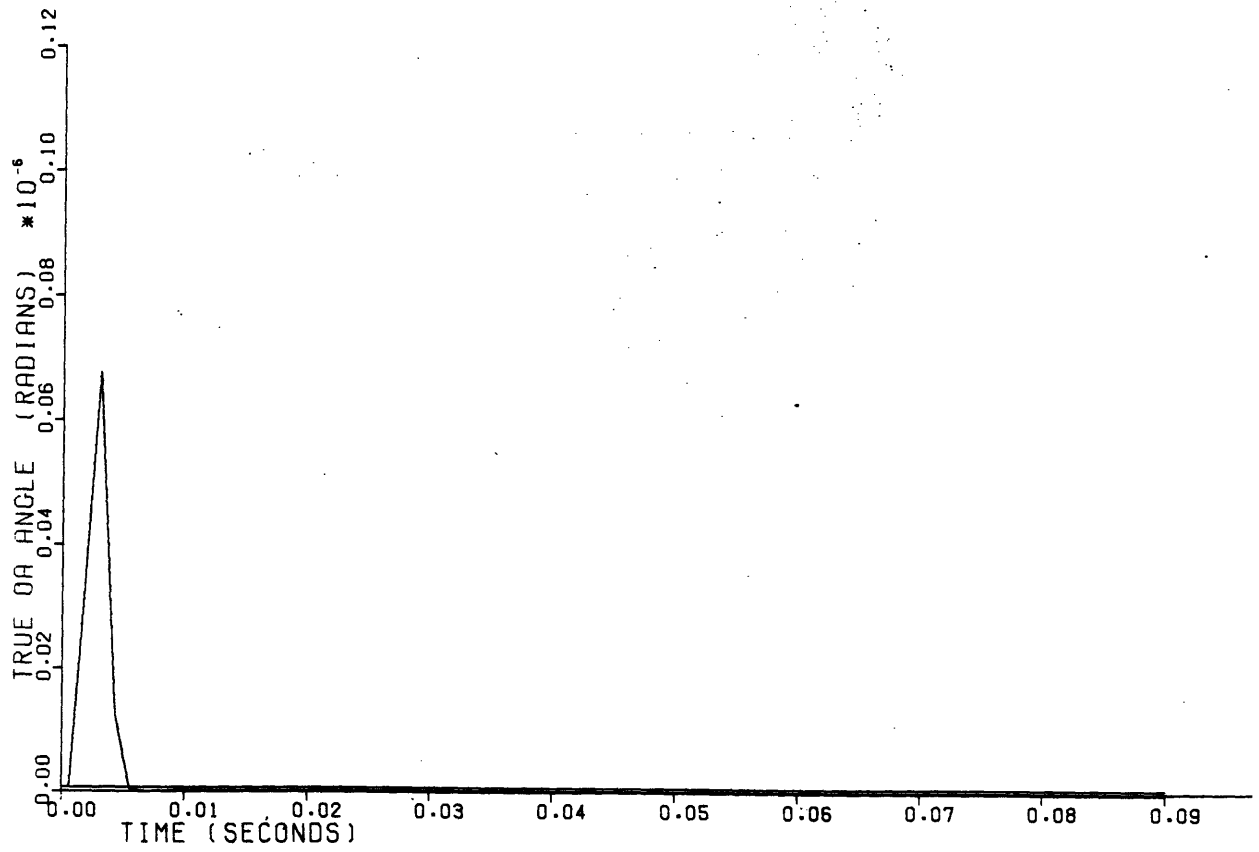


Figure 11-4. Run #1 24 Pt. Average Position vs. Time (Zero Input)

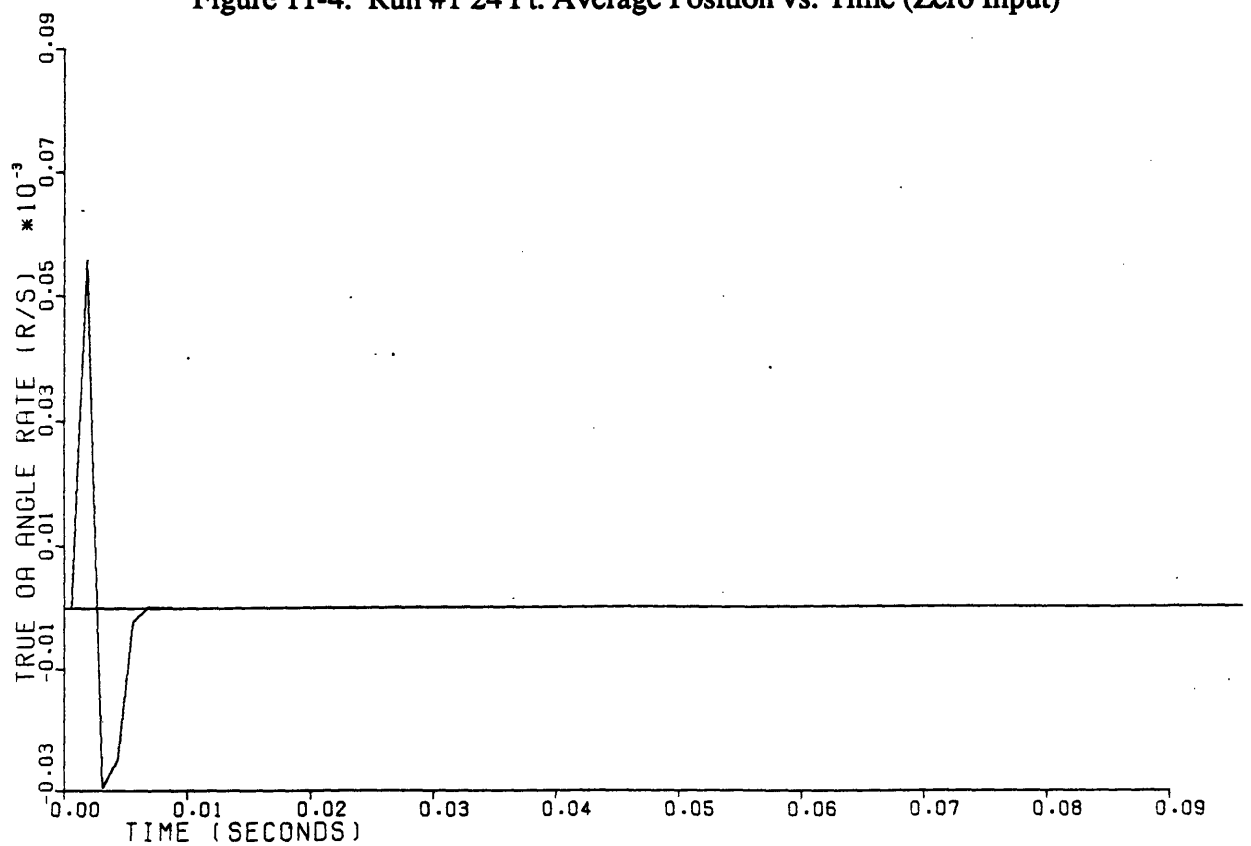


Figure 11-5. Run #1 24 Pt. Average Velocity vs. Time (Zero Input)

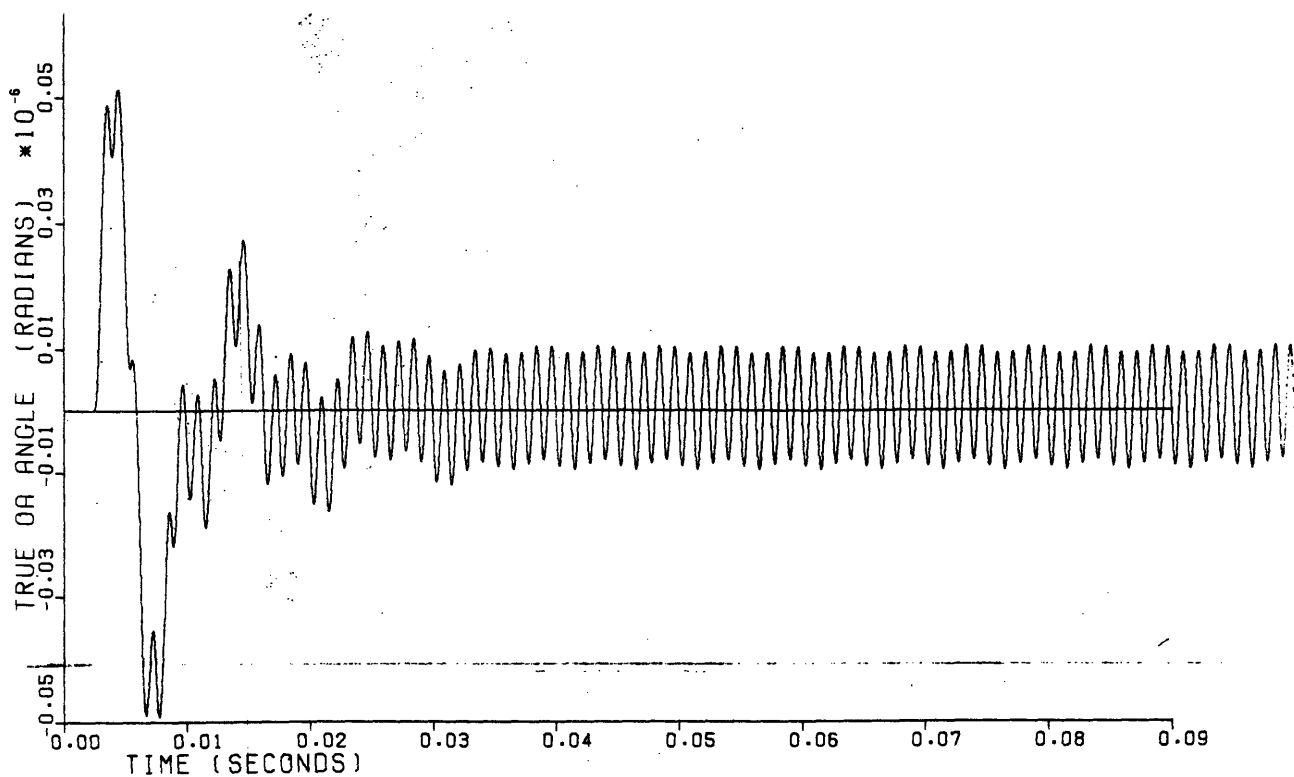


Figure 11-6. Run #2 Float Position vs. Time (Zero Input)

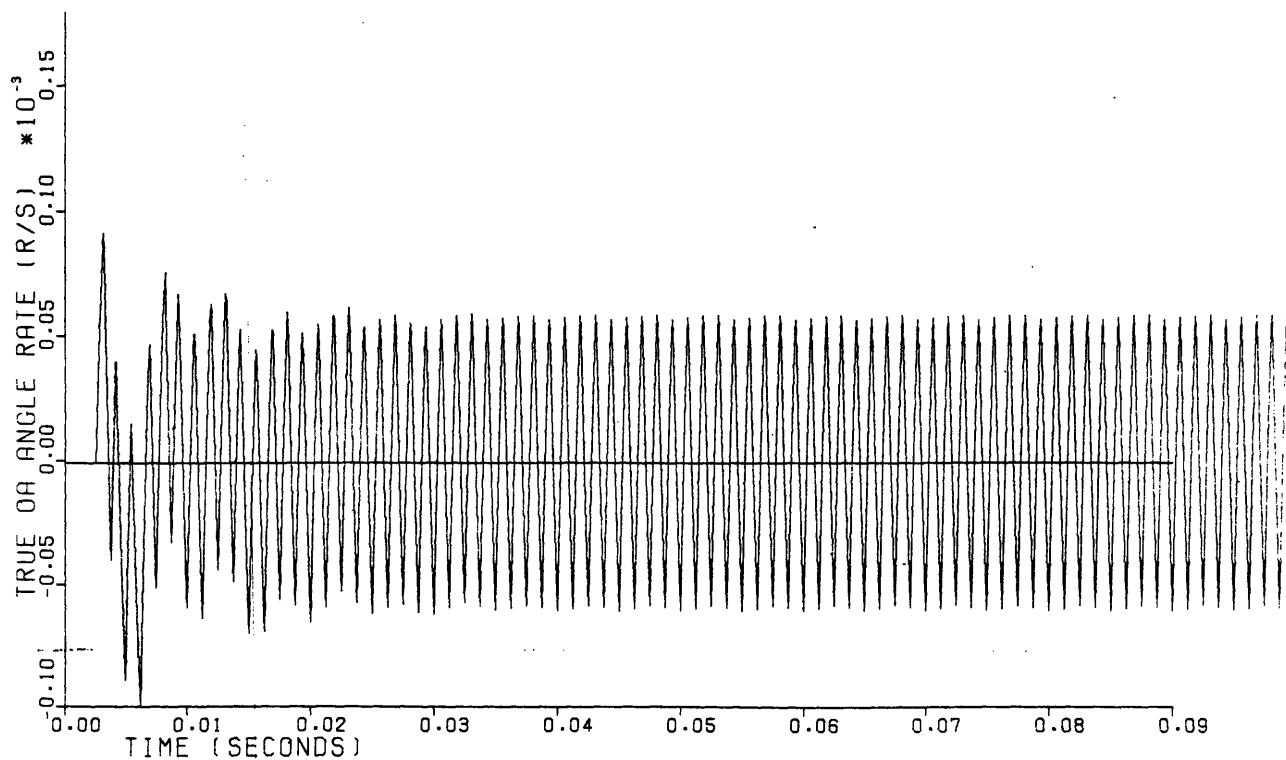


Figure 11-7. Run #2 Float Velocity vs. Time (Zero Input)

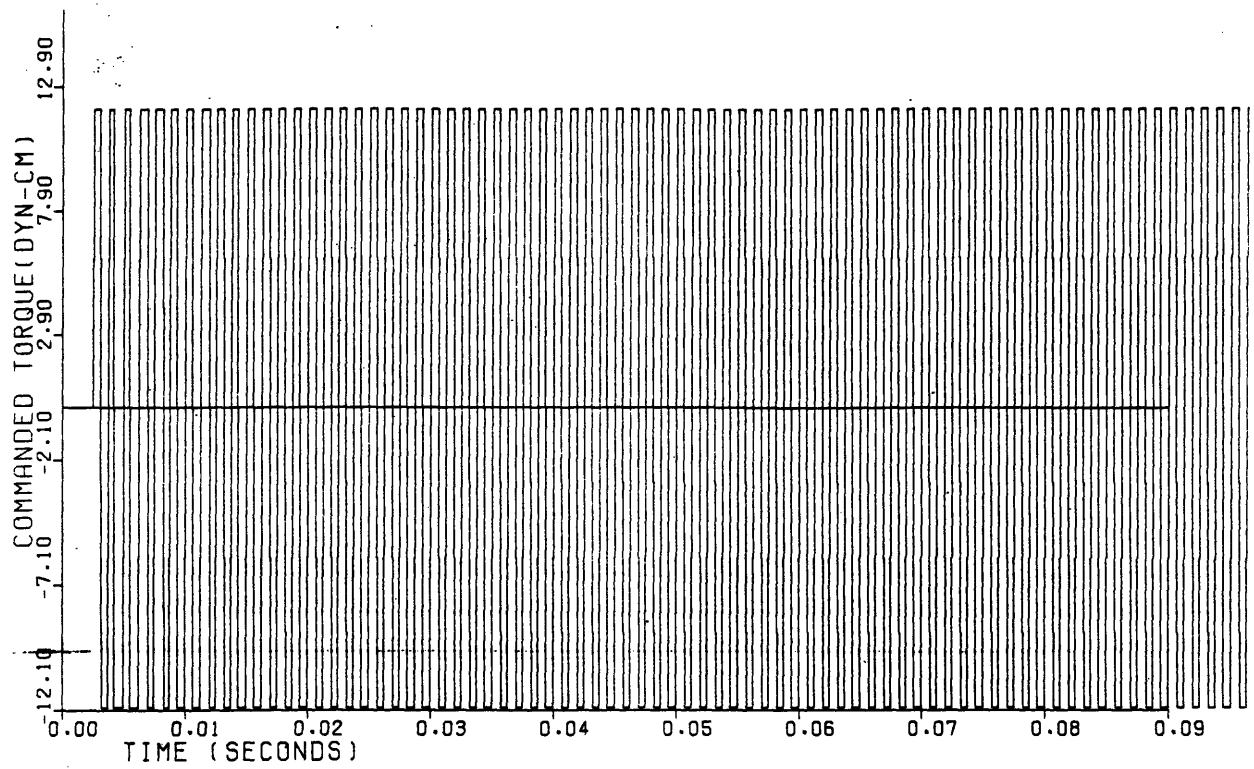


Figure 11-8. Run #2 Commanded Torque vs. Time (Zero Input)

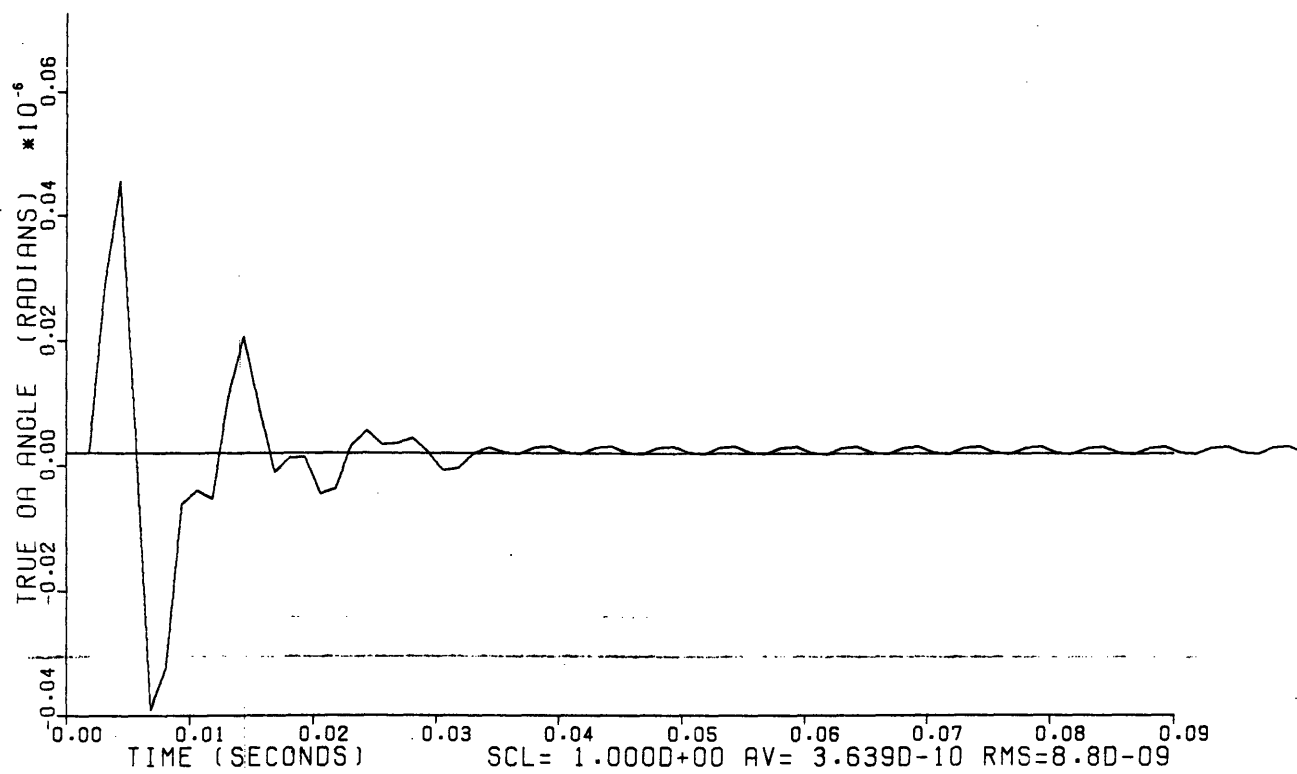


Figure 11-9. Run #2 24 Pt. Average Position vs. Time (Zero Input)

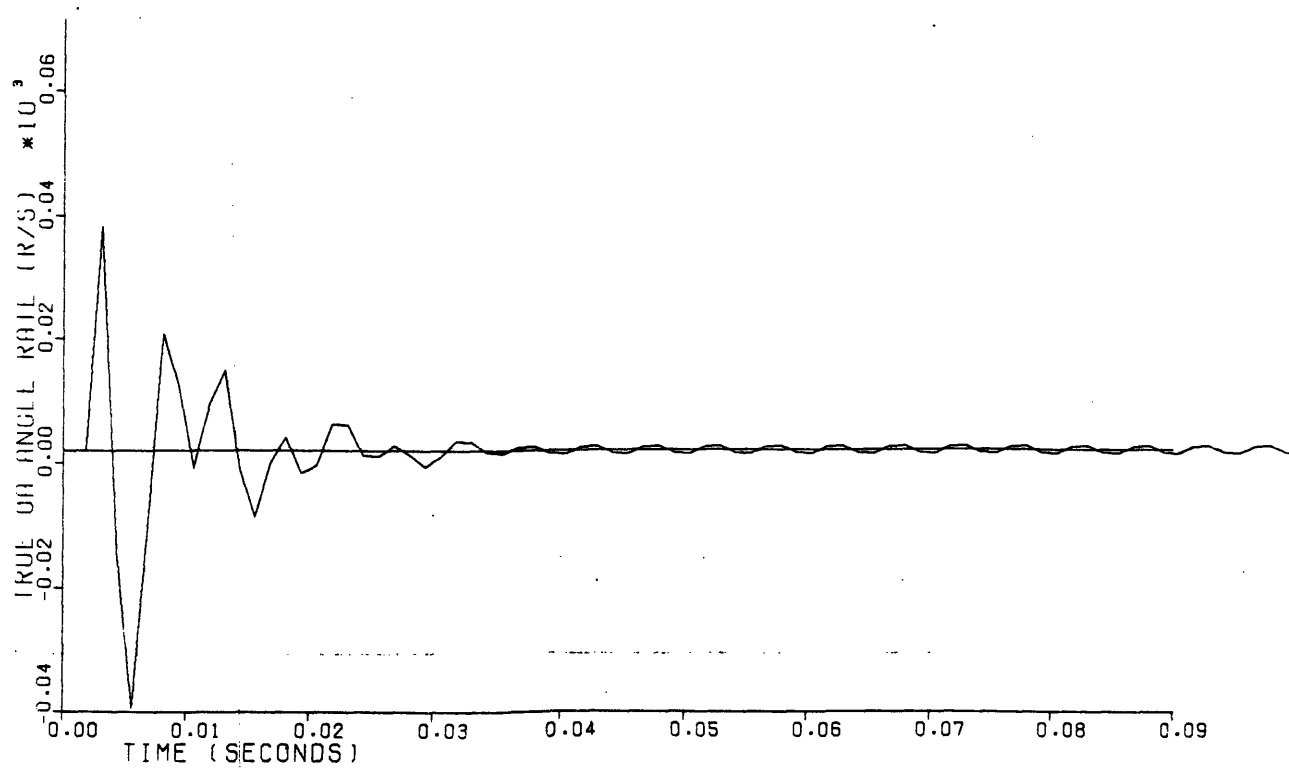


Figure 11-10. Run #2 24 Pt. Average Velocity vs. Time (Zero Input)

correcting torque applied does not exactly compensate the external input and the result in a growing error between the commanded and external input with a corresponding growth of average float hang-off. When the error becomes greater than half a quantum of commanded torque ($1/256 \times 12$ dyne-cm) then the torque word will get rounded to the next higher or lower value. The float receives an extra quantum of torque in the appropriate direction. Figure 11-11 shows the added quantum occurring approximately every 15 msec.

The amplitudes of float oscillation were the same for both runs although they appear different in the plots. This is because the simplified controller has a larger overshoot during the initial transient for which the automatic scaling adjusts. Both runs show again only slight float hang-offs, although the simplified controller has a hang-off about 1/4 that of the estimator based controller.

The Kalman filter controller seeks to obtain values of float angle and angle rate at the start of a torquing interval which results in a minimum float deviation from null. Although the simplified controller doesn't deliberately aim for these model states, the end result is that it does attain such values in steady state. The simplified controller takes longer to reach steady state compared to the Kalman filter controllers.

The accuracy of the external torque estimation, as seen by the steady state torquing pattern, can be improved by averaging the torque words over time (rectangular filtering) or by applying other digital filtering techniques. For the Kalman filter controllers the estimator errors for float position, velocity and external torque rapidly converge to zero. Thus the external input could be obtained by outputting the state estimate τ stored in the microprocessor. The accuracy would depend on the word length allocated in the processor. Improvements in accuracy would similarly be obtained through digital filtering.

11.2.2 Simulation Plots

Figures 11-11 through 11-20 present the plots for the responses to a constant non-zero input.

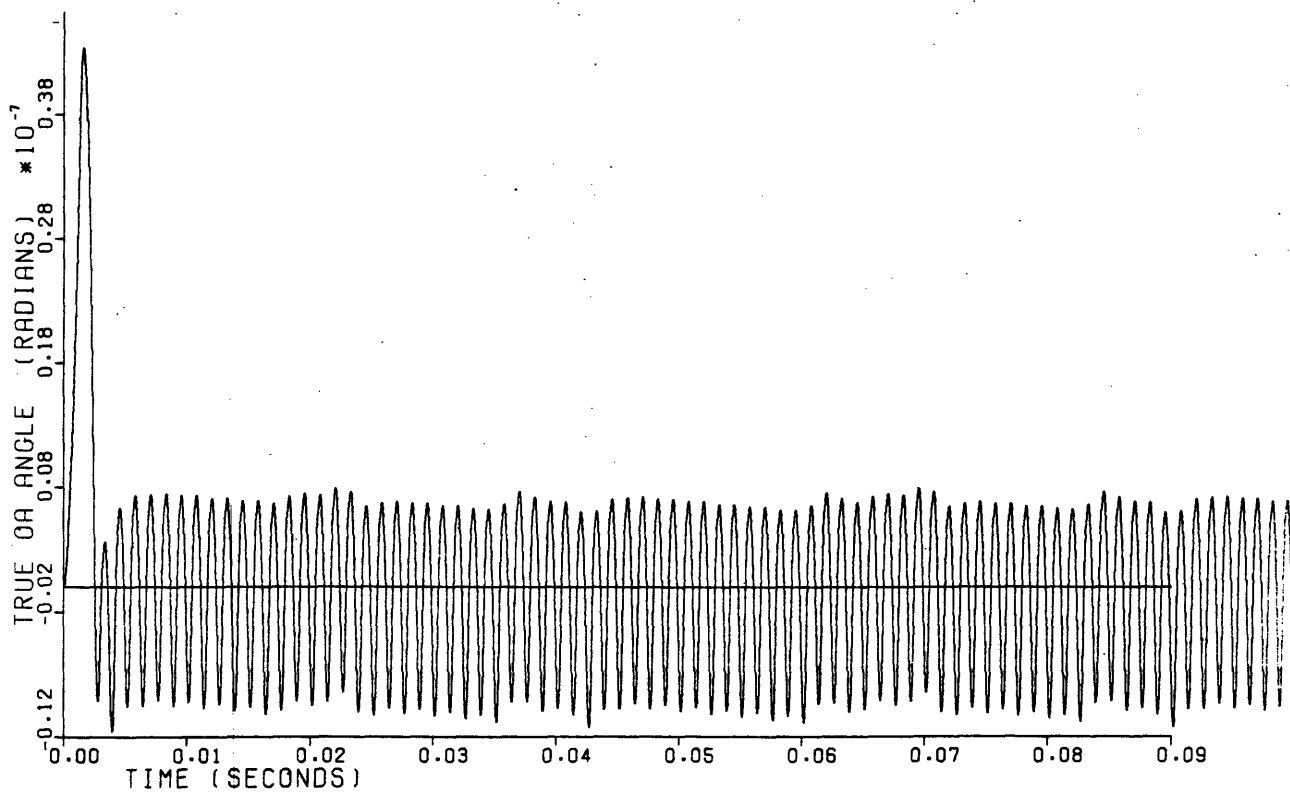


Figure 11-11. Run #3 Float Position vs. Time (Constant 7.5°/hr Input)

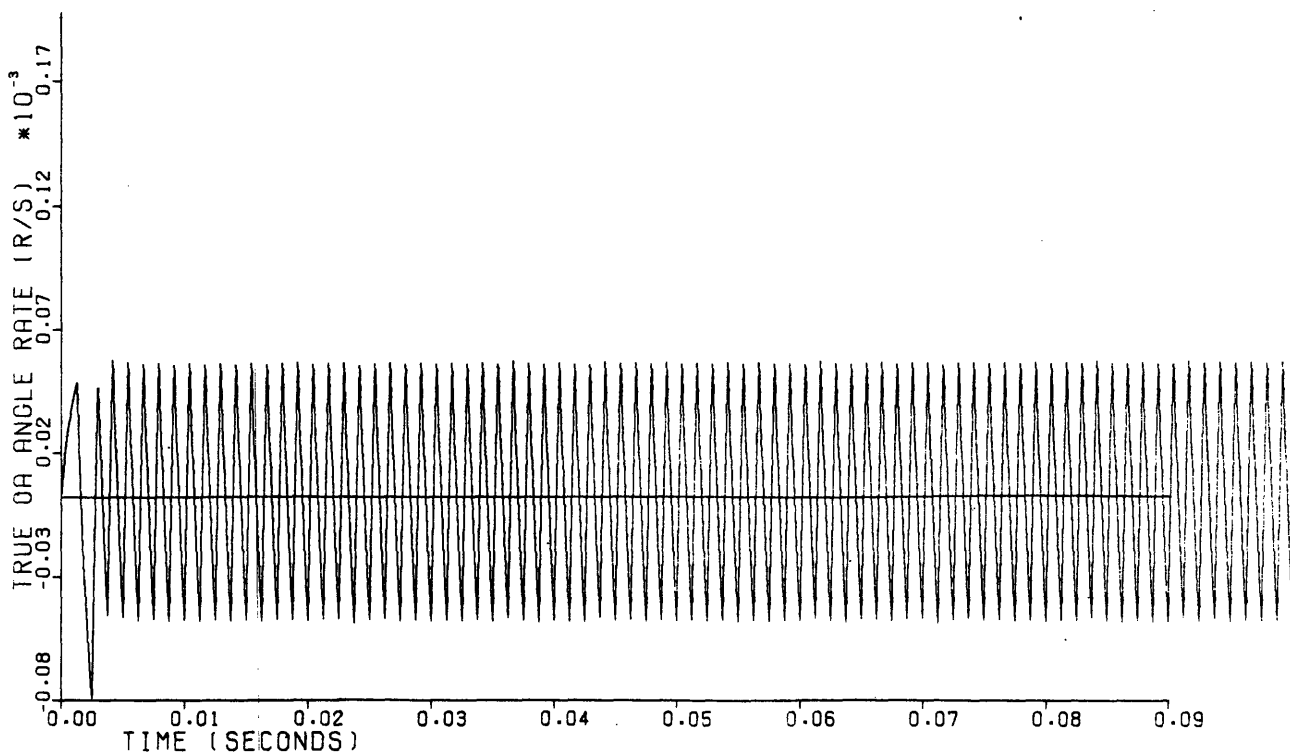


Figure 11-12. Run #3 Float Velocity vs. Time (Constant 7.5°/hr Input)

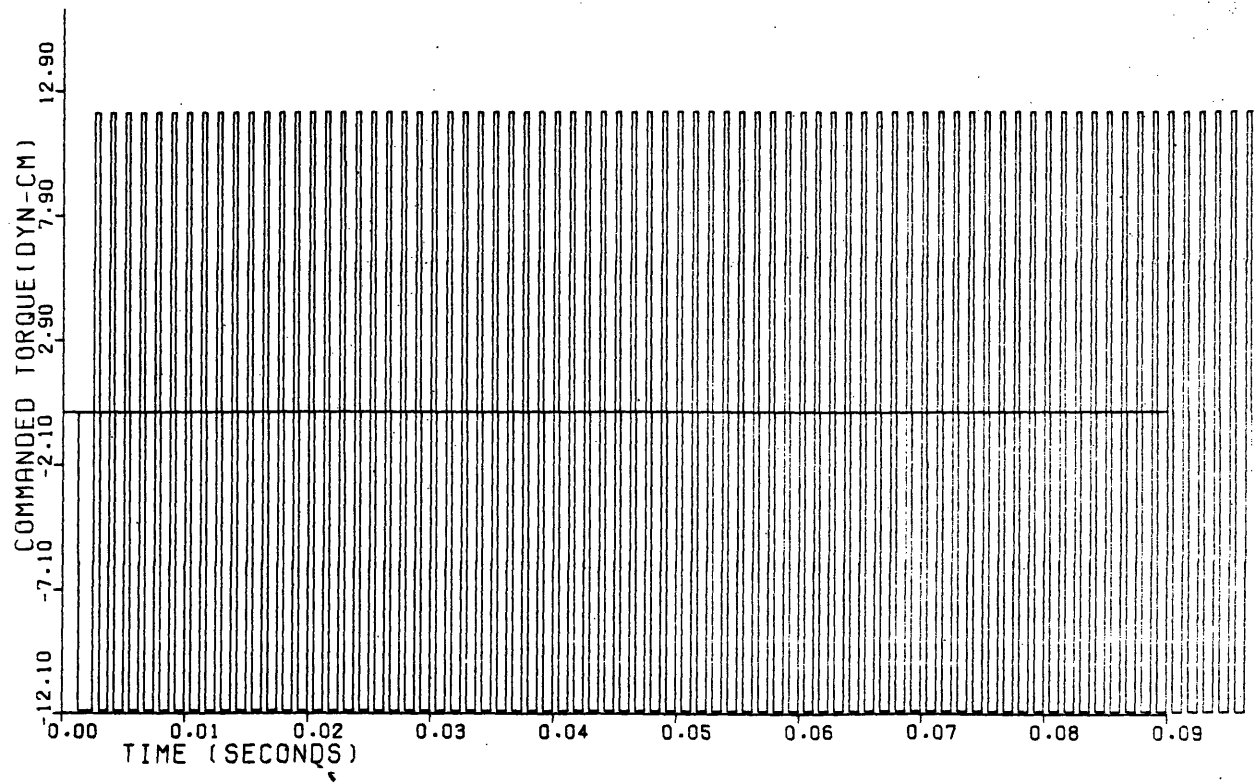


Figure 11-13. Run #3 Commanded Torque vs. Time (Constant 7.5°/hr Input)

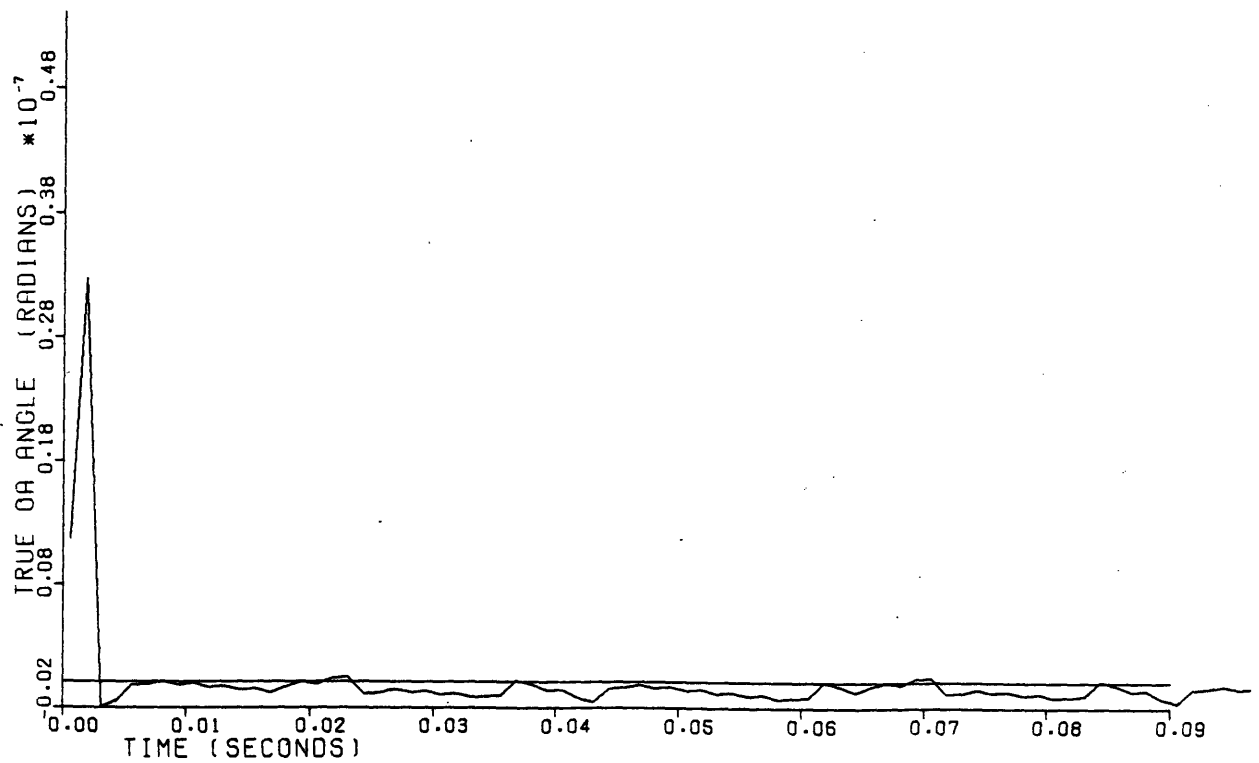


Figure 11-14. Run #3 24 Pt. Average Position vs. Time (Constant 7.5°/hr Input)

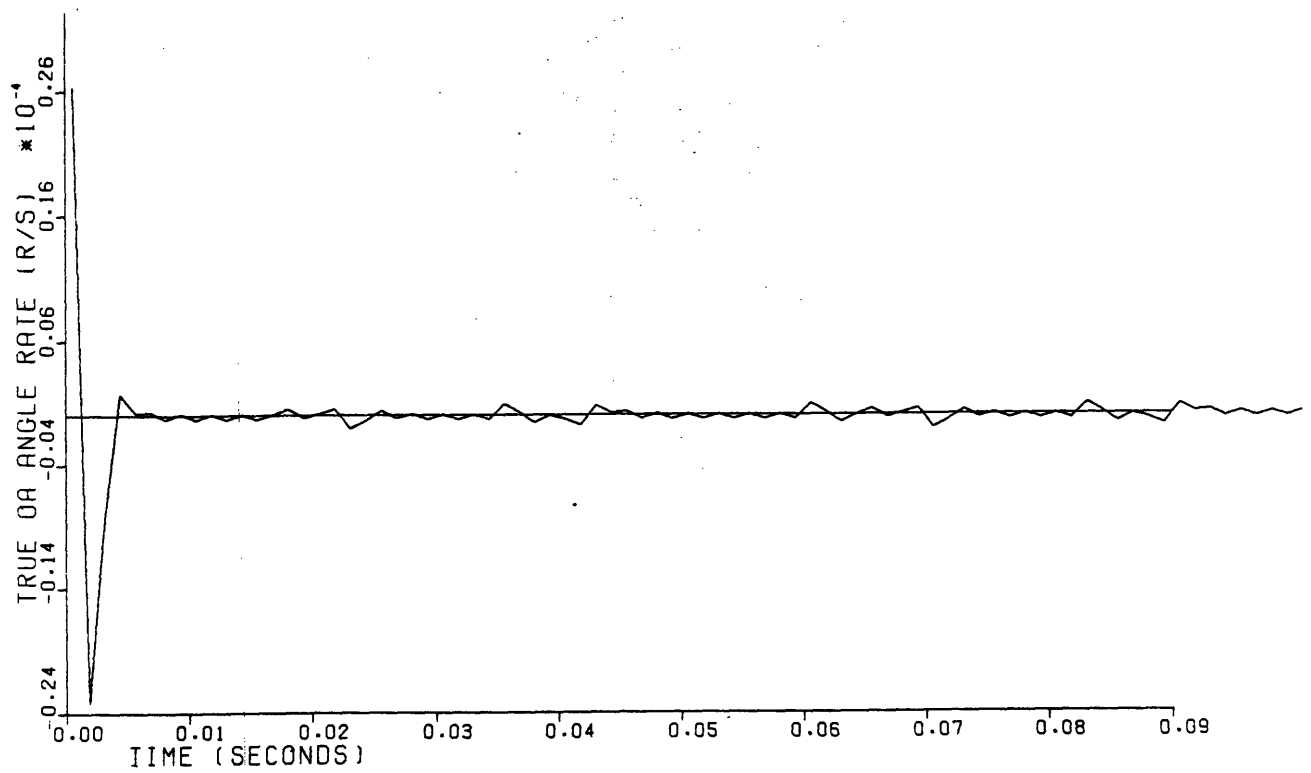


Figure 11-15. Run #3 24 Pt. Average Velocity vs. Time (Constant 7.5°/hr Input)

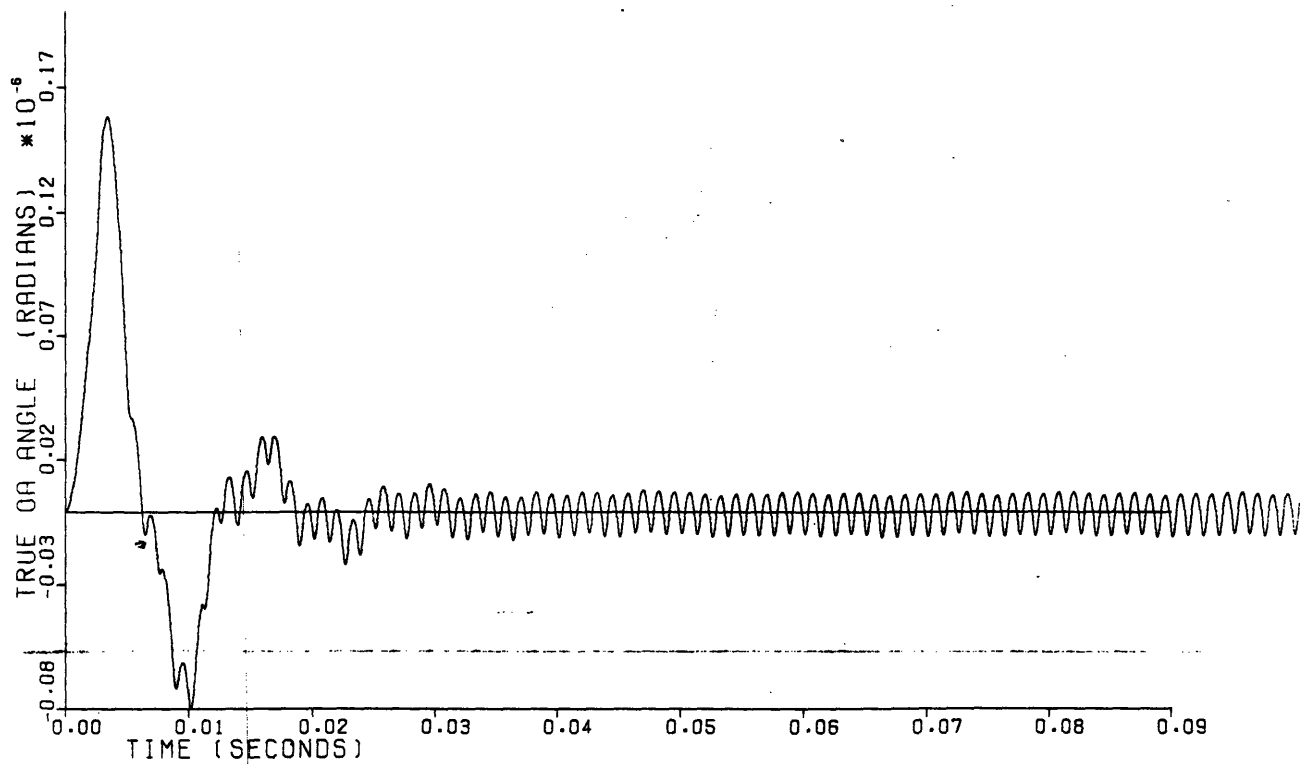


Figure 11-16. Run #4 Float Position vs. Time (Constant 7.5°/hr Input)

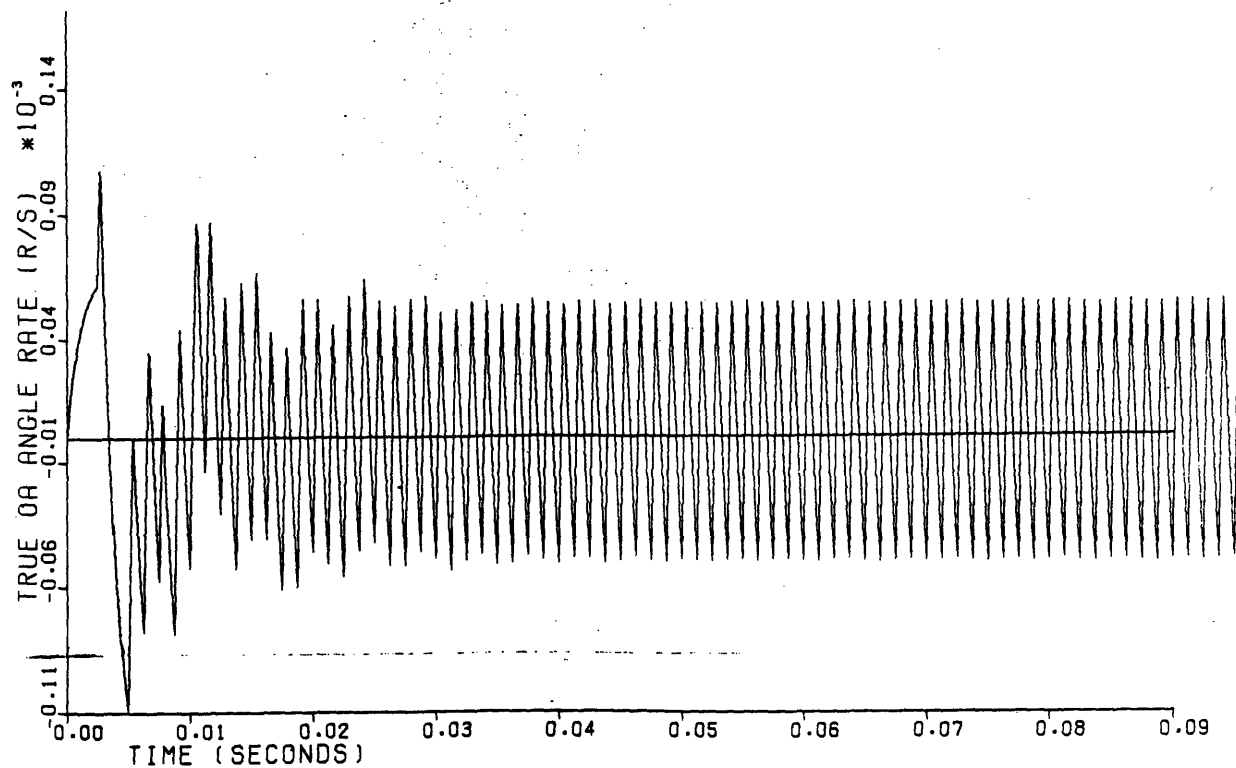


Figure 11-17. Run #4 Float Velocity vs. Time (Constant 7.5°/hr Input)

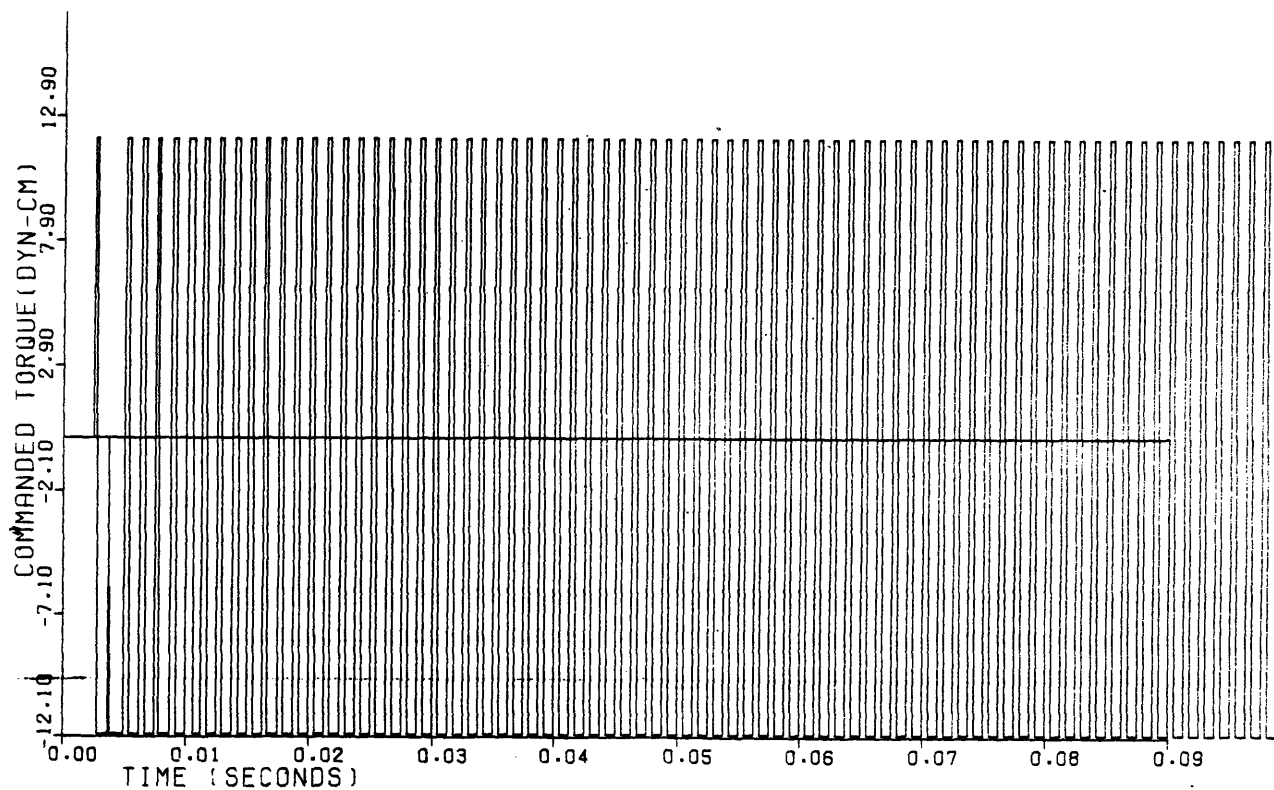


Figure 11-18. Run #4 Commanded Torque vs. Time (Constant 7.5°/hr Input)

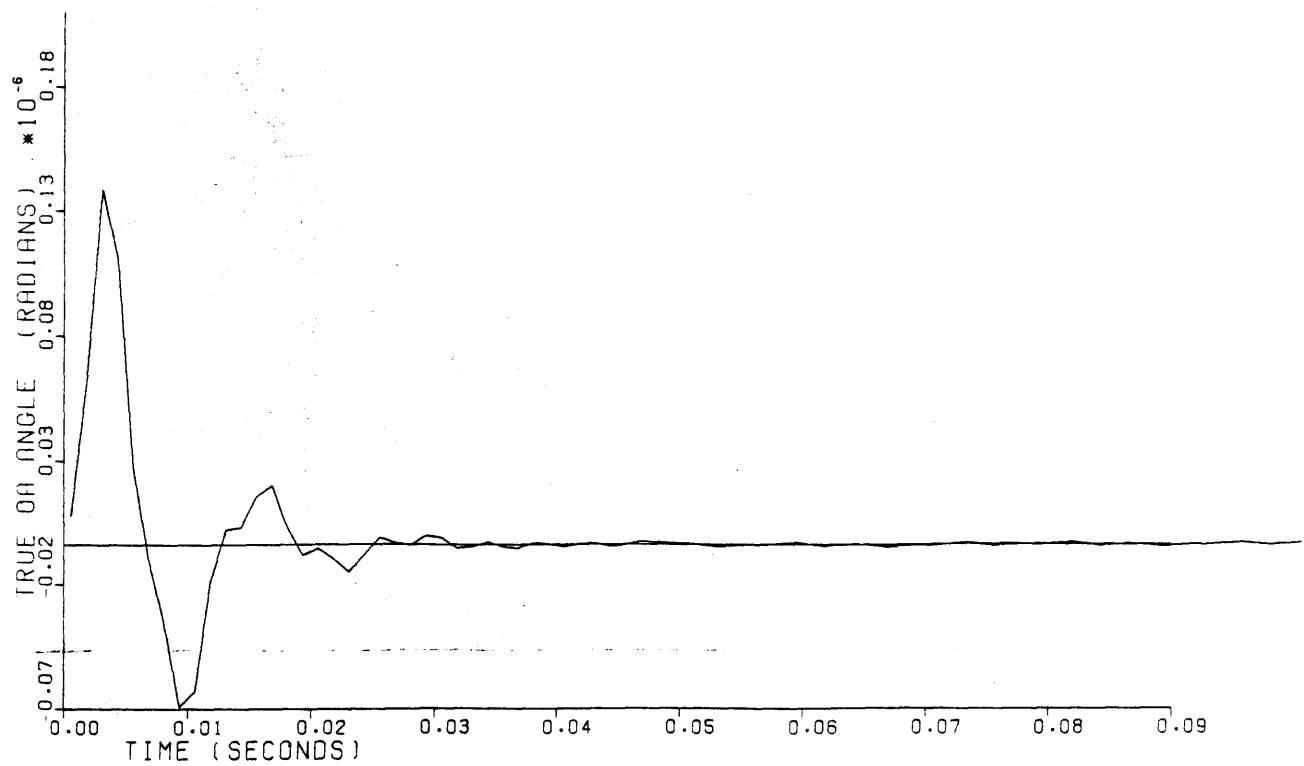


Figure 11-19. Run #4 24 Pt. Average Position vs. Time (Constant 7.5°/hr Input)

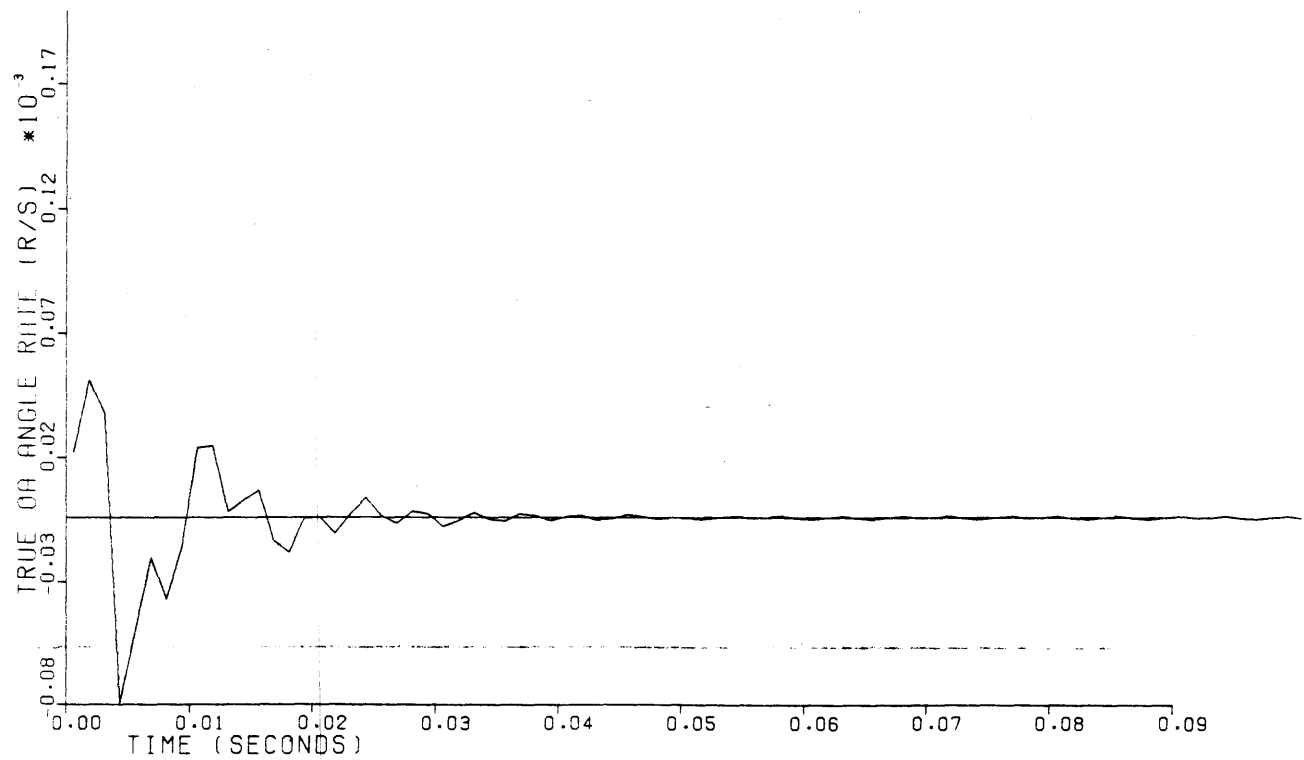


Figure 11-20. Run #4 24 Pt. Average Velocity vs. Time (Constant 7.5°/hr Input)

12. STEP INPUT RESPONSE

12.1 Results Summary

Table 12-3 shows the summary of results for a step input response.

Table 12-1. Results Summary for Step Input from $0-15^\circ/\text{hr} = 0-7.58 \text{ dyne-cm}$

RUN #	Description	Maximum float deviation from null (nrad)	Settling time to S.S. operation (msec)
5	4 st. controller fast gains	0.4121	13
6	4 st. controller medium gains	0.07547	18
7	4 st. controller slow gains	0.3108	23
8	3 st. controller fast gains	0.07547	5.0
9	3 st. controller medium gains	0.08064	7.2
10	3 st. controller slow gains	0.1559	13
11	simplified controller	0.1798	33

After the transients (induced by the step change in external input) subsided, all controllers maintained minimum float oscillation and minimum quantization of the steady state torquing pattern. Each controller showed a comparatively large initial deviation in float position with respect to the steady state amplitude of float oscillation. For the runs shown, the step input happened to occur at the beginning of a torque pulse interval. The controllers were therefore unable to compensate for the new input rate until the next torque

word decision (1.25 msec later). Hence the period of poorly controlled float motion. From the standpoint of time lapse before compensating control, having the step input occur at the beginning of a torquing interval is a worst case scenario. On the other hand, the Kalman filter has an entire torquing interval to track the changing states and reduce the estimator error. If the step occurs near the end of a torquing interval then the time lapse before compensating control may be small, but the estimator errors may be large, thus generating erroneous decisions for control. Either way, there will be some degradation of transient performance to a step input due to time lapse before compensating control or the inability of the Kalman filter to catch up and reduce estimator errors in time for the next control decision.

It was somewhat surprising that the four state controller produced the largest initial float deviation. The rationale behind the observation is provided by the added fourth Kalman filter state $\dot{\tau}$ (slope of external input rate). A step input change in external input is read by the Kalman filter as an infinite change in slope $\dot{\tau}$. This will induce large estimator errors in the other states as well.

Since the four state controller uses slope information for a control decision, the large values of $\dot{\tau}$ coupled with the other state estimator errors will provoke erroneous control decisions until the errors diminish. The three state controller does not use slope information for a control decision nor does it try to estimate $\dot{\tau}$. The result is a maximum float derivation which is significantly reduced compared to the four state controller.

The maximum float derivation is also dependent upon the estimator gains. A slower estimator, with smaller gains, will place less weight on the measurements. Sharp changes in the input will be smoothed in the same manner that noise is filtered. The improvement can be seen in Run #6 where the response of the four state controller with medium gains is presented. Although the estimator errors take about three times as long to diminish compared to the estimation with fast gains, the maximum float deviation has been reduced by a factor of about 5.5. However, if the estimator is too slow, then the large

estimator errors at the time of a control decision will lead to erroneous control decisions and large float deviations. This is illustrated in Run #7.

As expected, settling time for the float to return to steady state oscillation is directly proportional to the settling time of the estimator transient error. The simplified controller produced the longest settling time and a maximum float deviation comparable to the three state controller with slow gains.

The conclusion is that the three state controller with fast gains will provide a superior step input response.

12.2 Simulation Plots

Figures 12-1 through 12-53 present the plots for the responses to a step input.

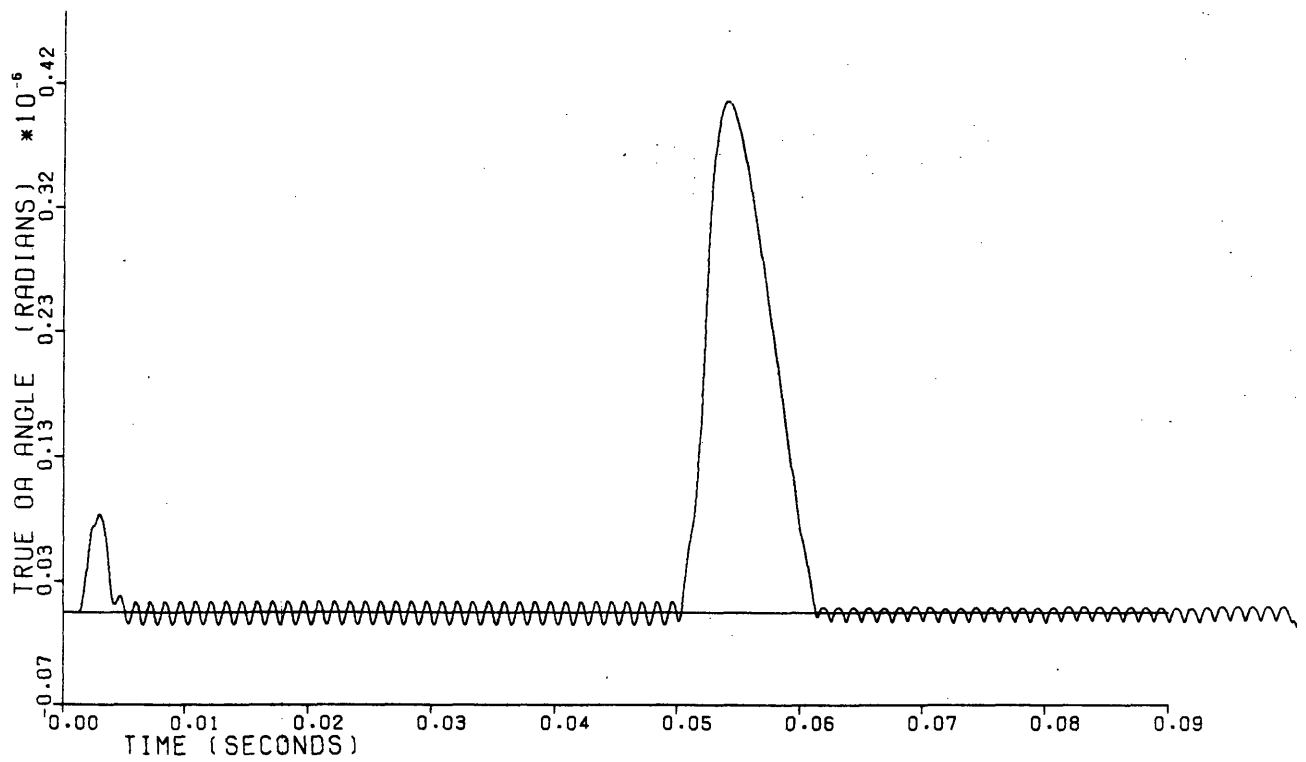


Figure 12-1. Run #5 Float Position vs. Time (15°/hr Step Input)

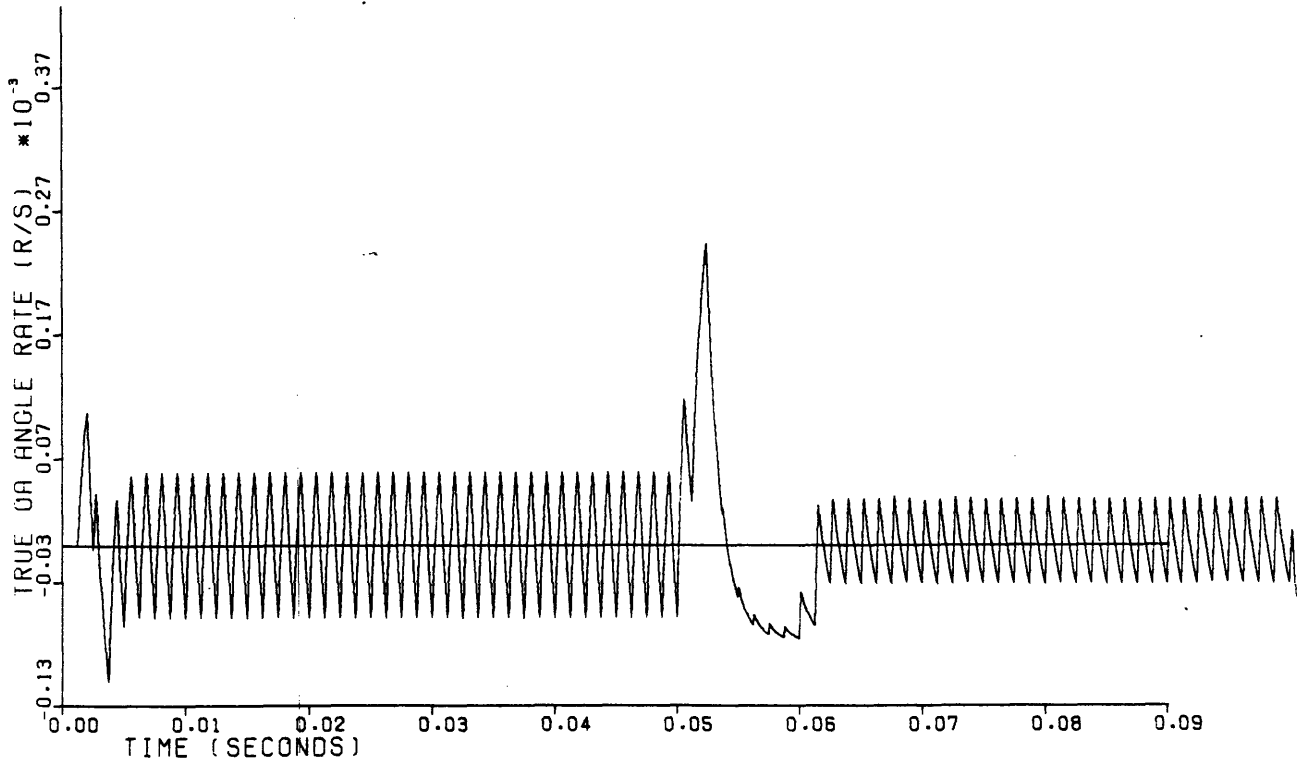


Figure 12-2. Run #5 Float Velocity vs. Time (15°/hr Step Input)

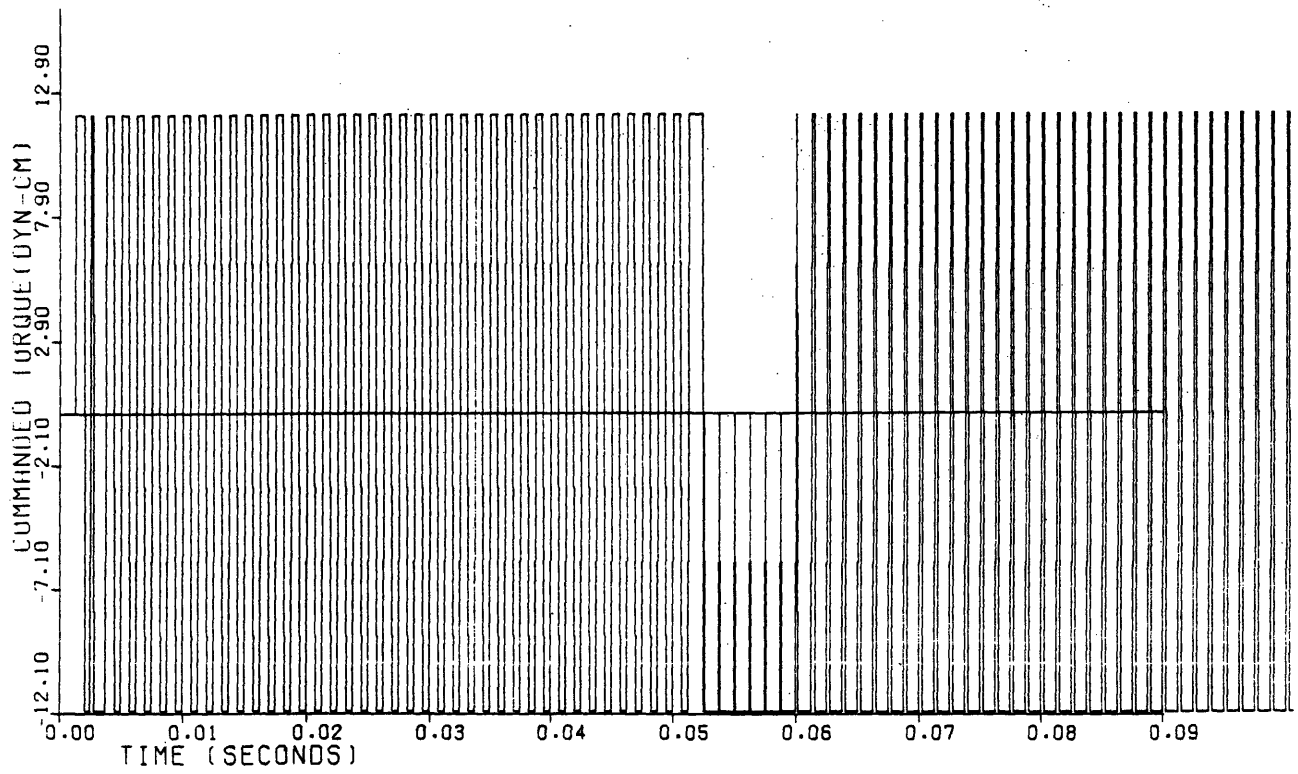


Figure 12-3. Run #5 Commanded Torque vs. Time (15°/hr Step Input)

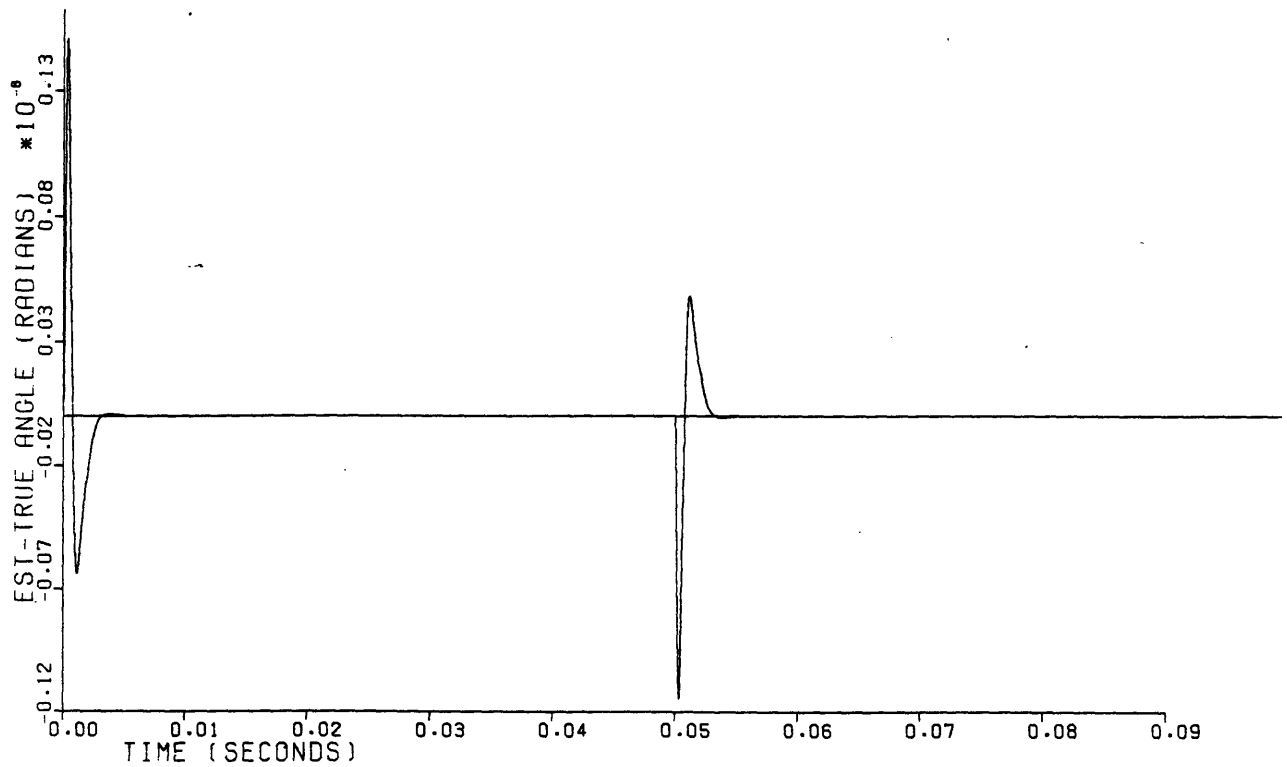


Figure 12-4. Run #5 Position Estimation Error vs. Time (15°/hr Step Input)

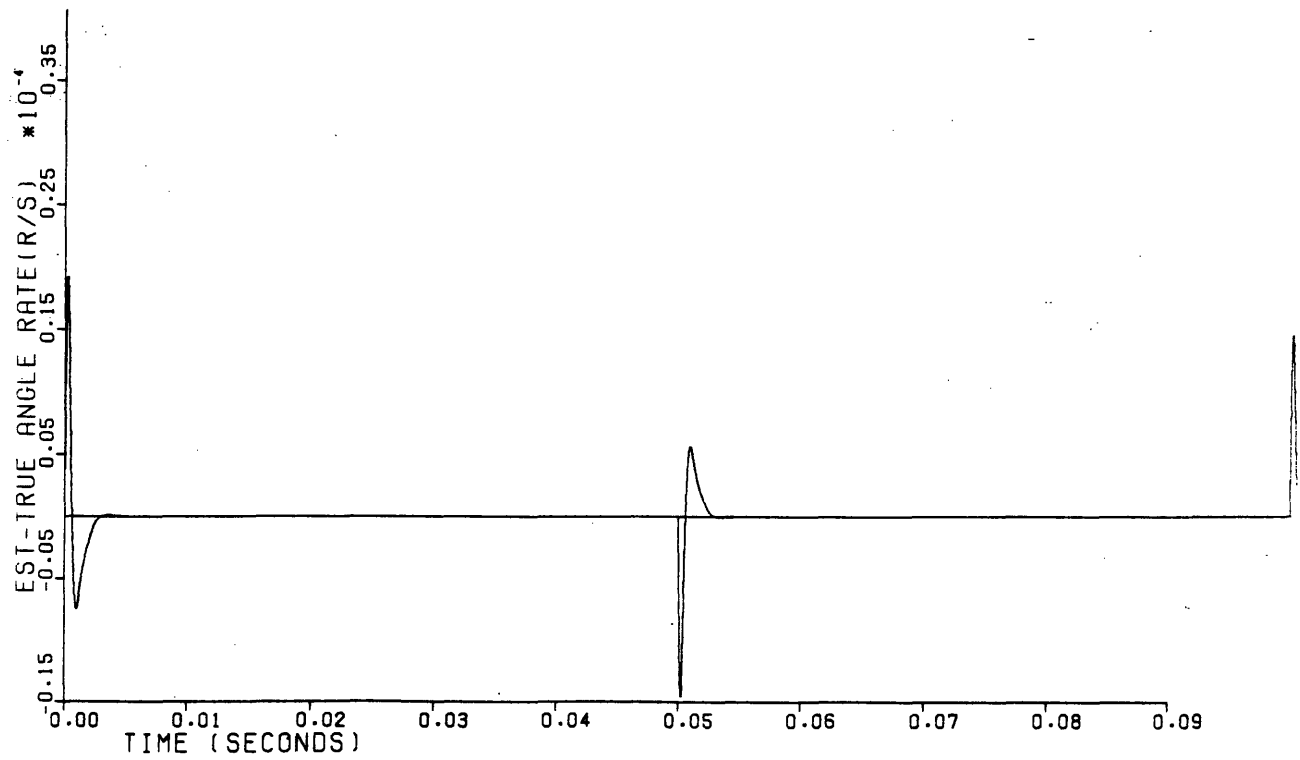


Figure 12-5. Run #5 Velocity Estimation Error vs. Time (15°/hr Step Input)

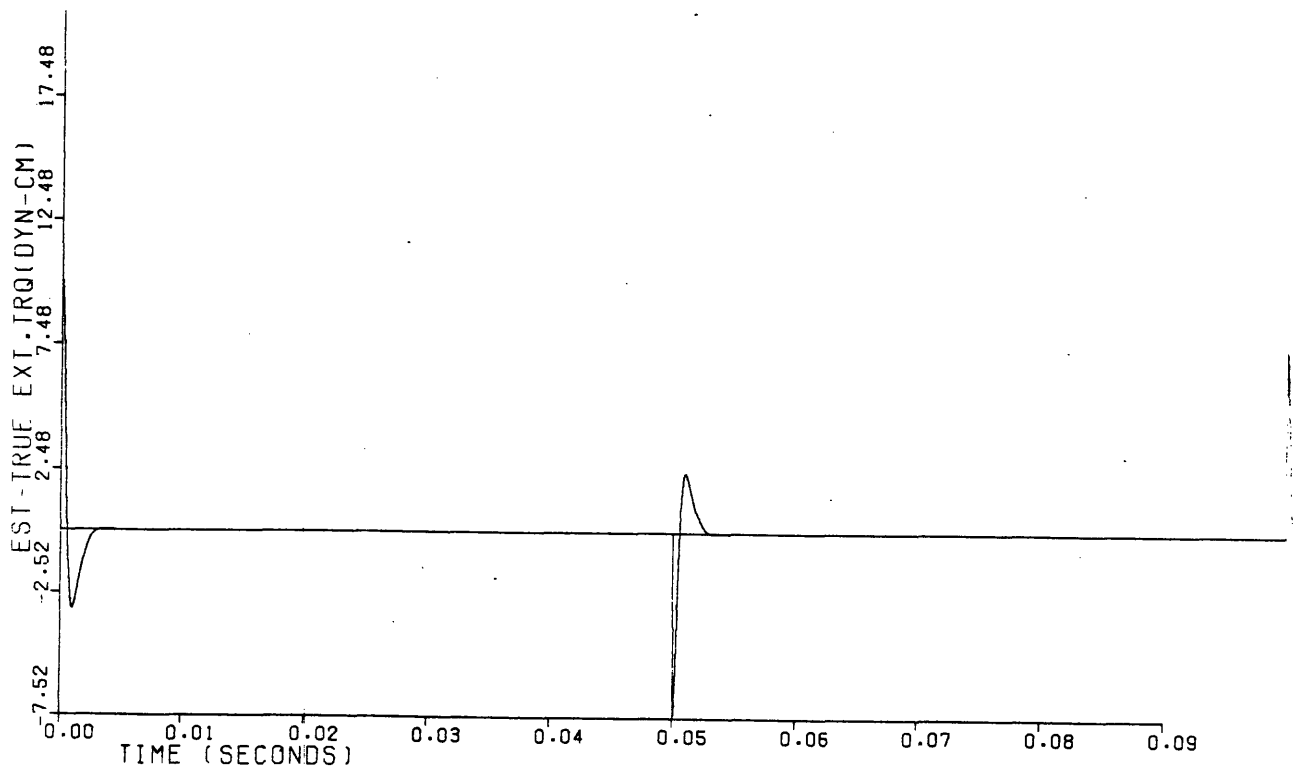


Figure 12-6. Run #5 External Torque Estimation Error vs. Time (15°/hr Step Input)

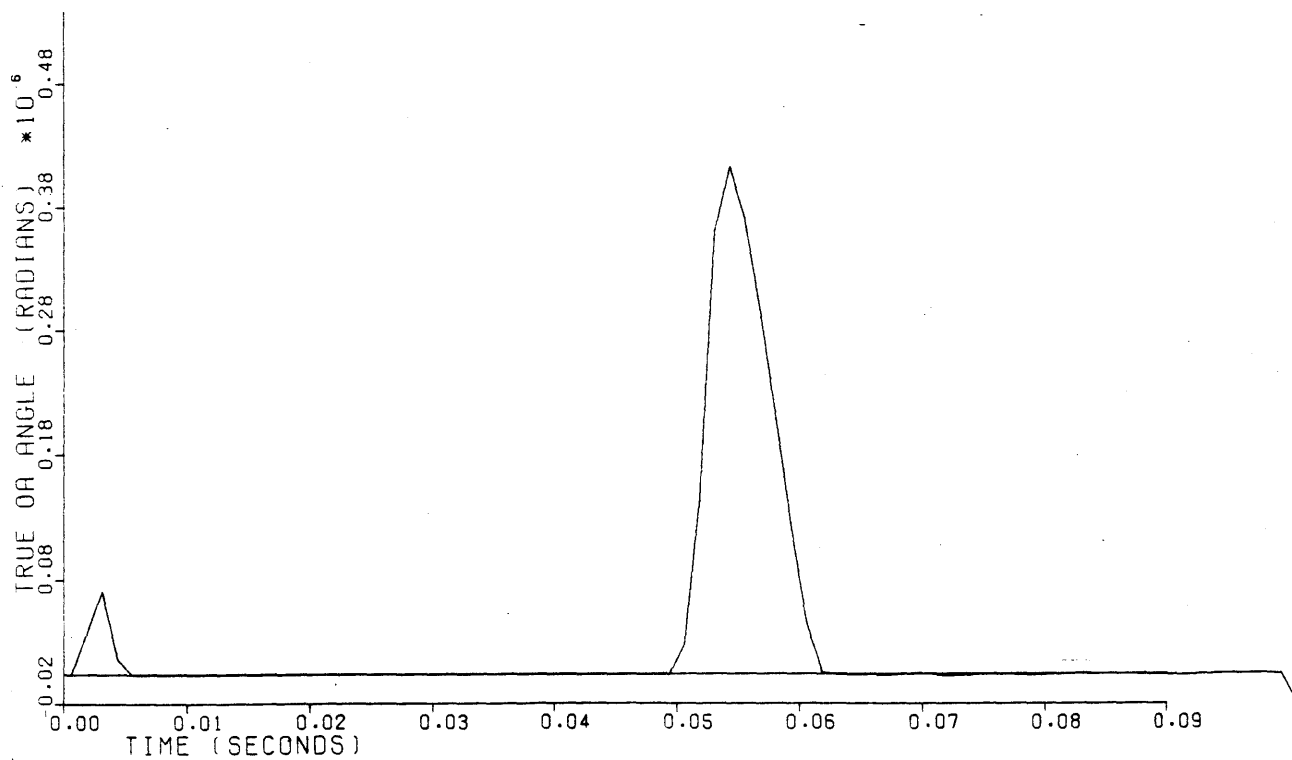


Figure 12-7. Run #5 24 Pt. Average Position vs. Time (15°/hr Step Input)

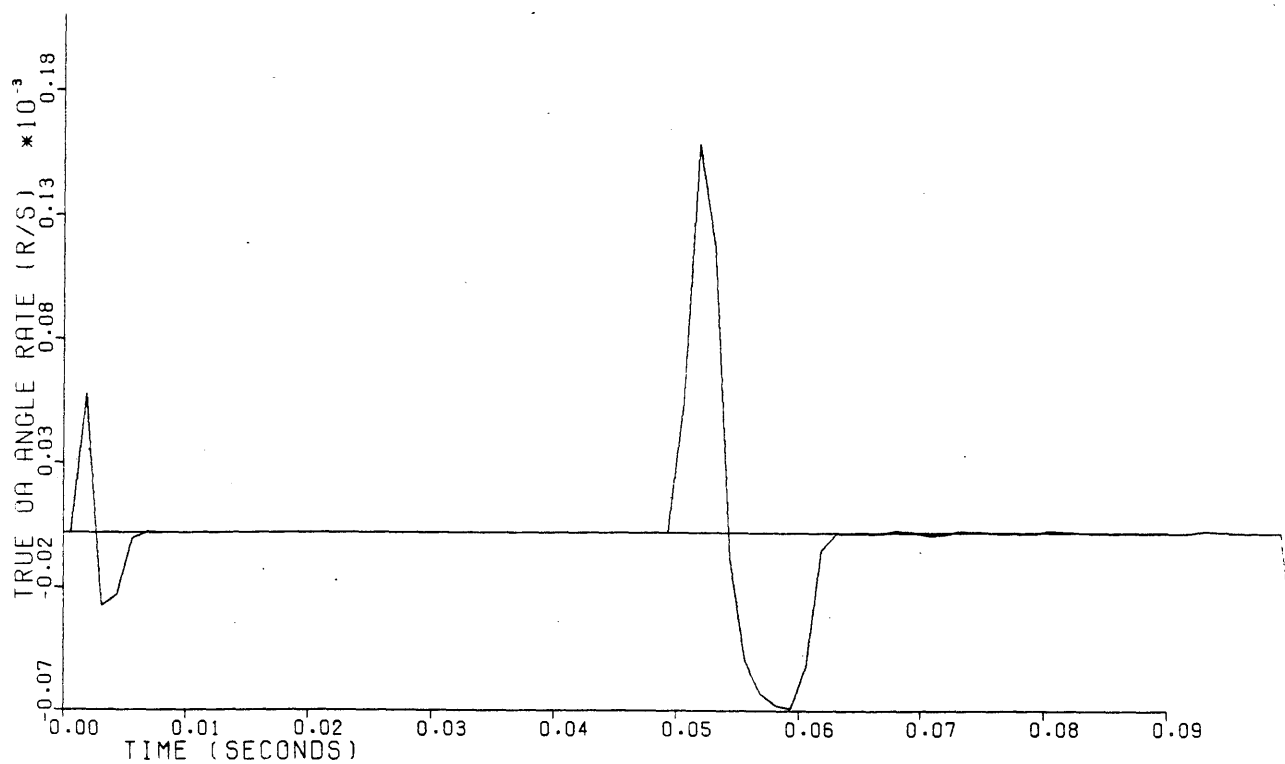


Figure 12-8. Run #5 24 Pt. Average Velocity vs. Time (15°/hr Step Input)

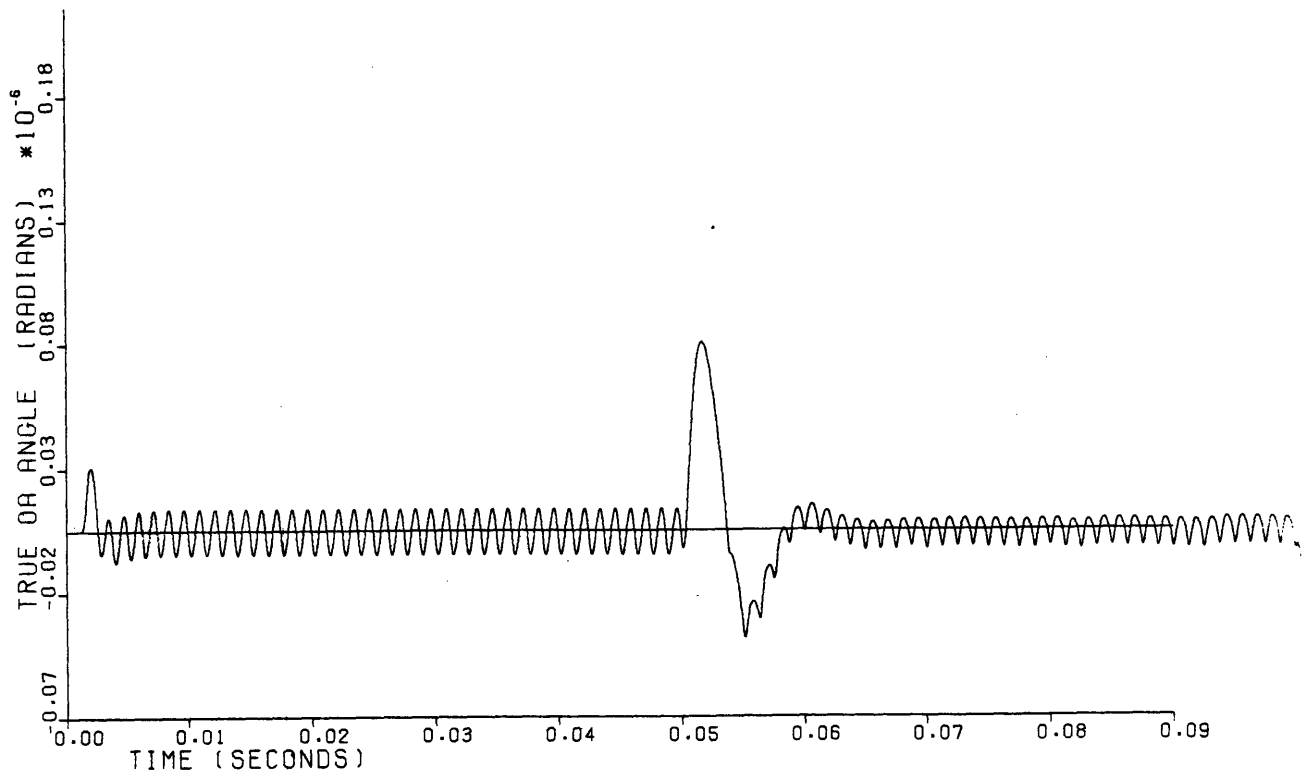


Figure 12-9. Run #6 Float Position vs. Time (15°/hr Step Input)

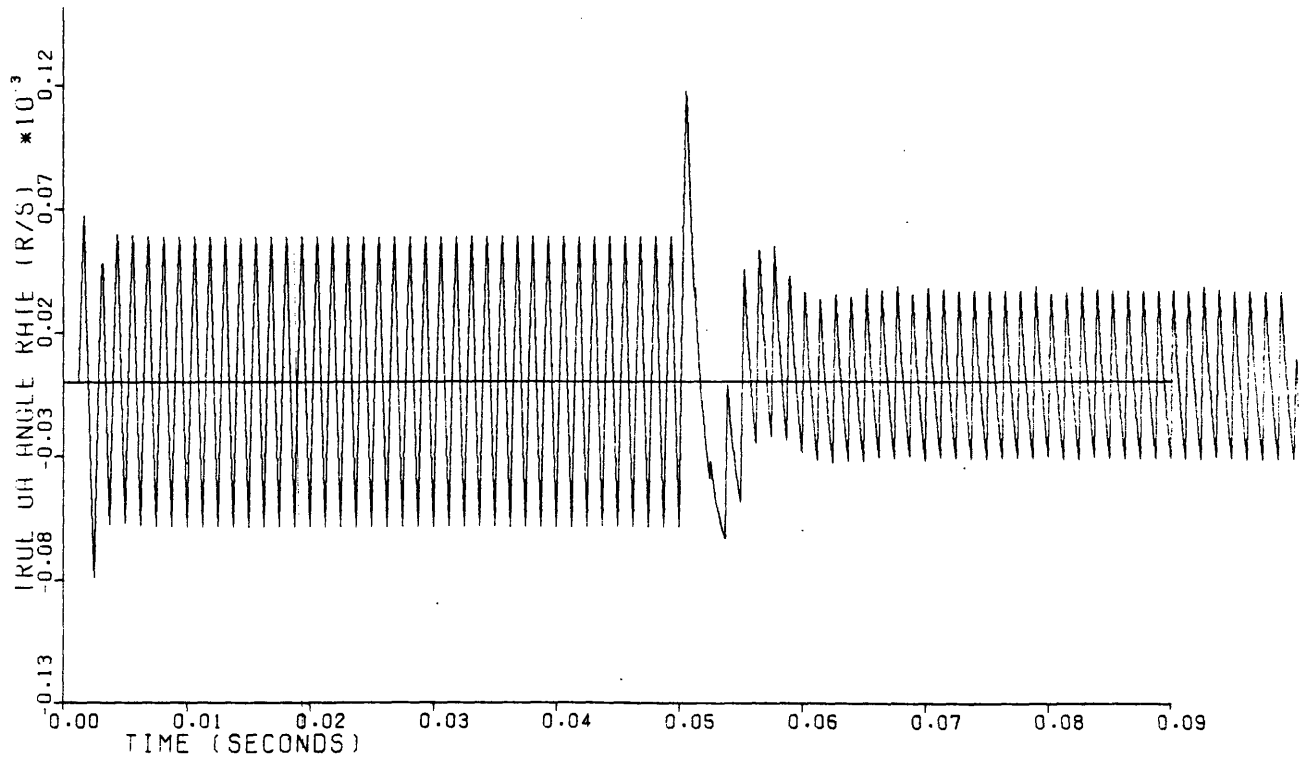


Figure 12-10. Run #6 Float Velocity vs. Time (15°/hr Step Input)

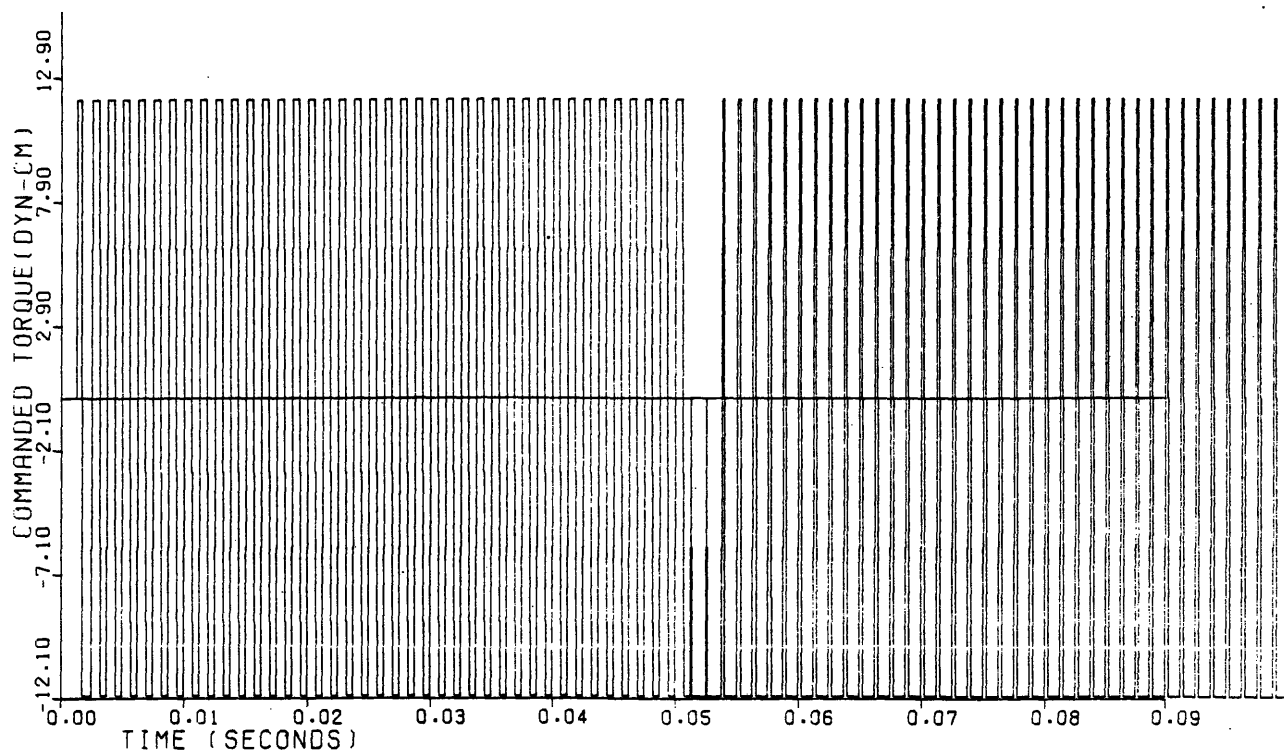


Figure 12-11. Run #6 Commanded Torque vs. Time (15°/hr Step Input)

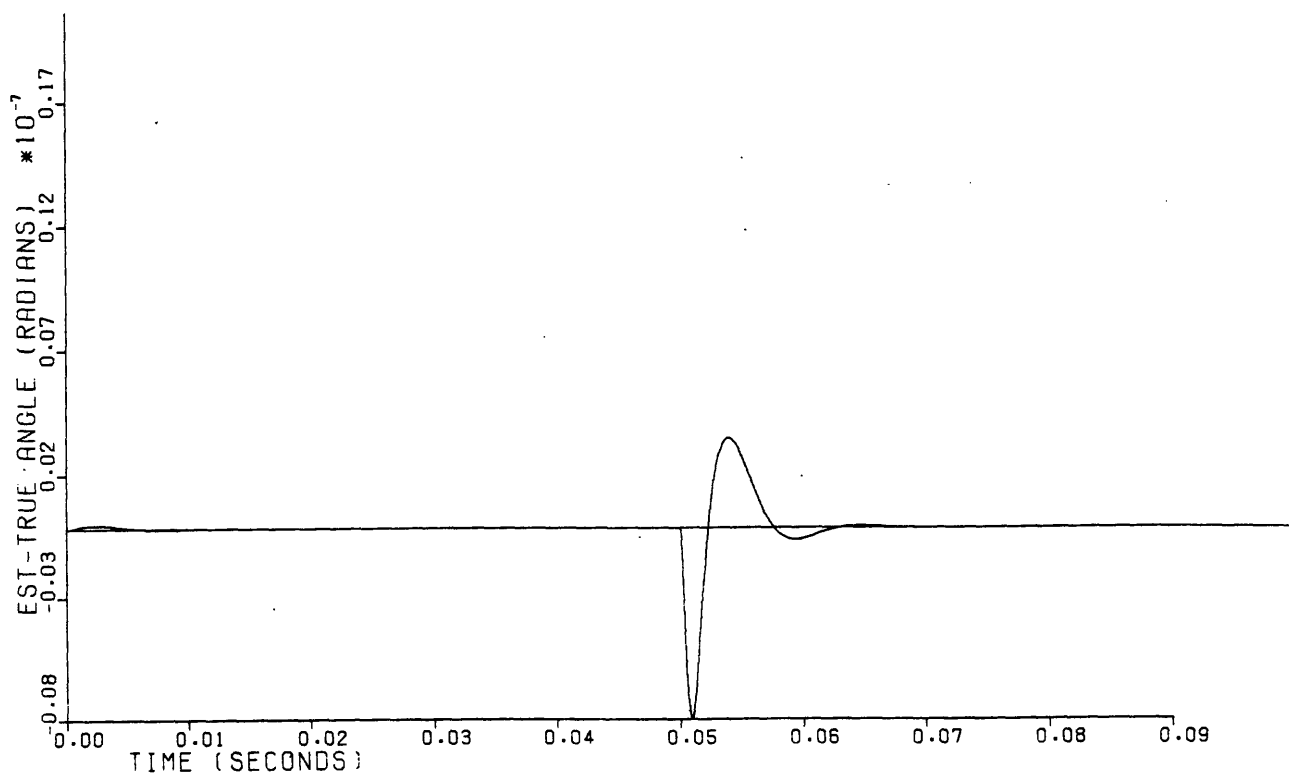


Figure 12-12. Run #6 Position Estimation Error vs. Time (15°/hr Step Input)

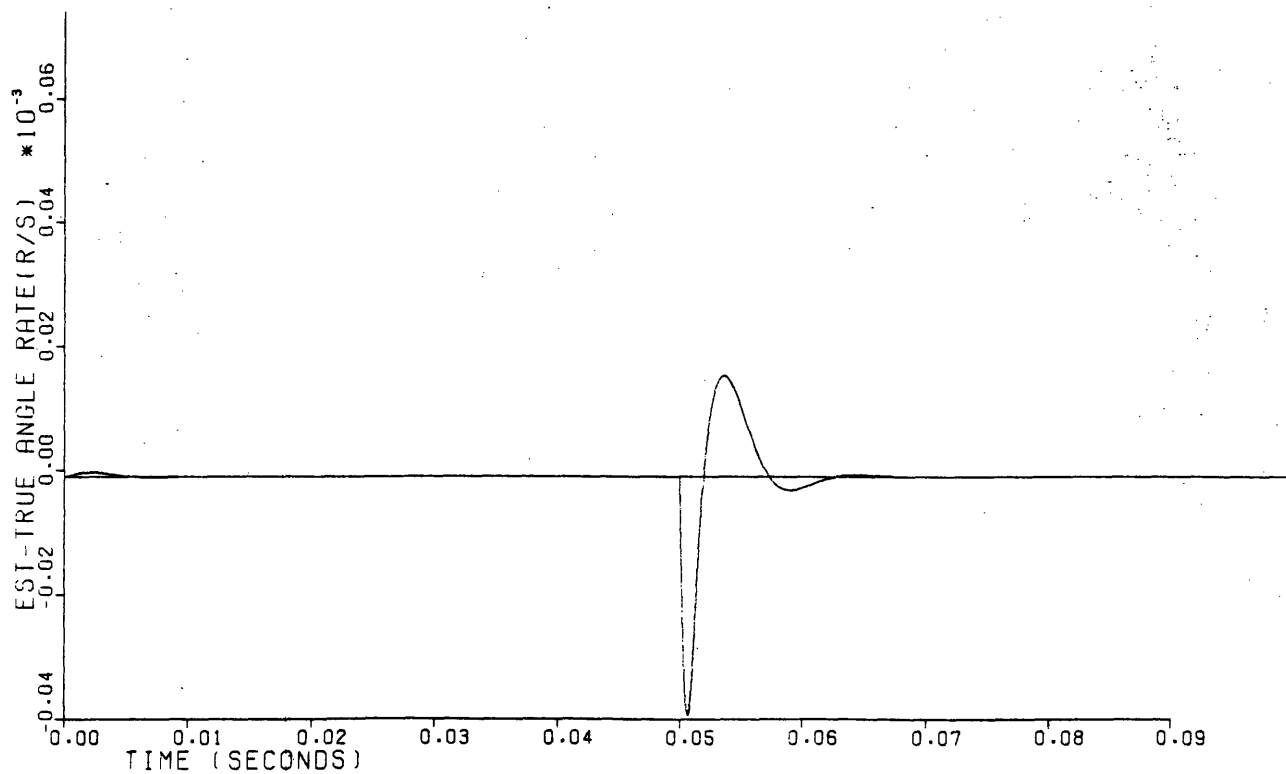


Figure 12-13. Run #6 Velocity Estimation Error vs. Time (15°/hr Step Input)

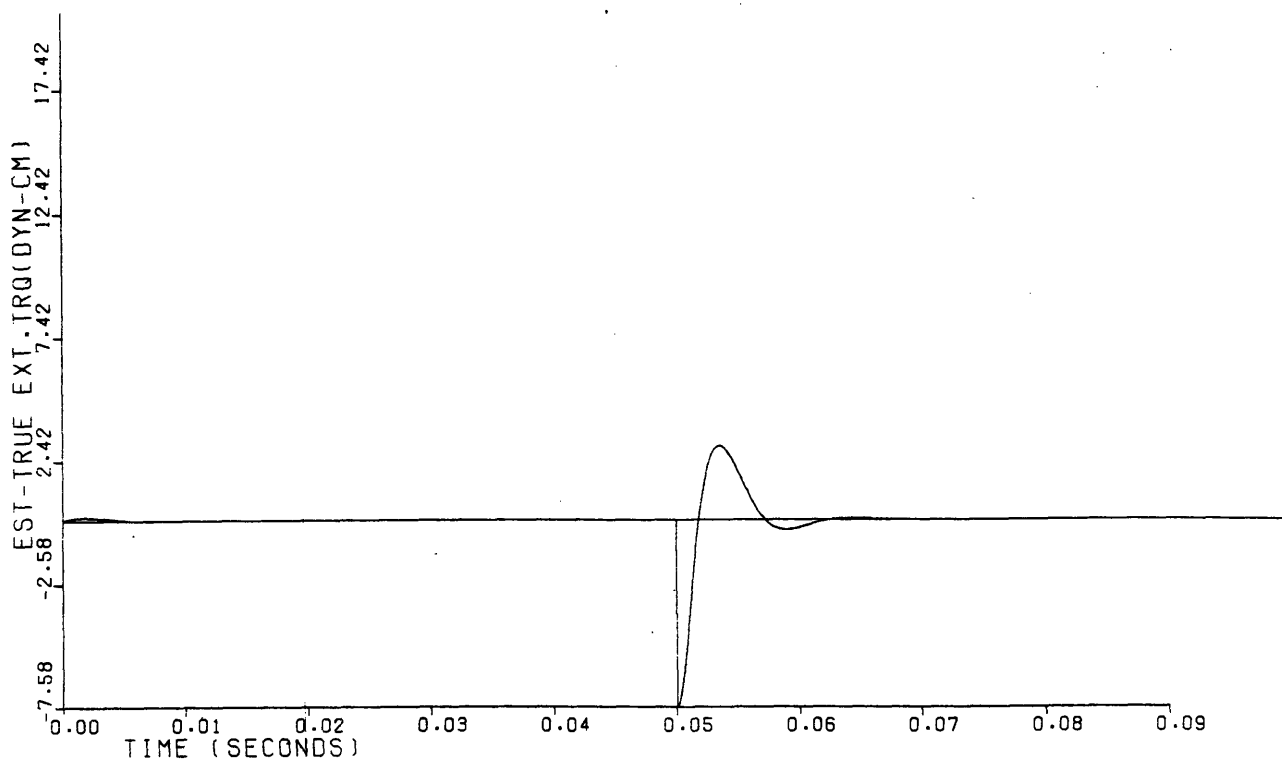


Figure 12-14. Run #6 External Torque Estimation Error vs. Time (15°/hr Step Input)

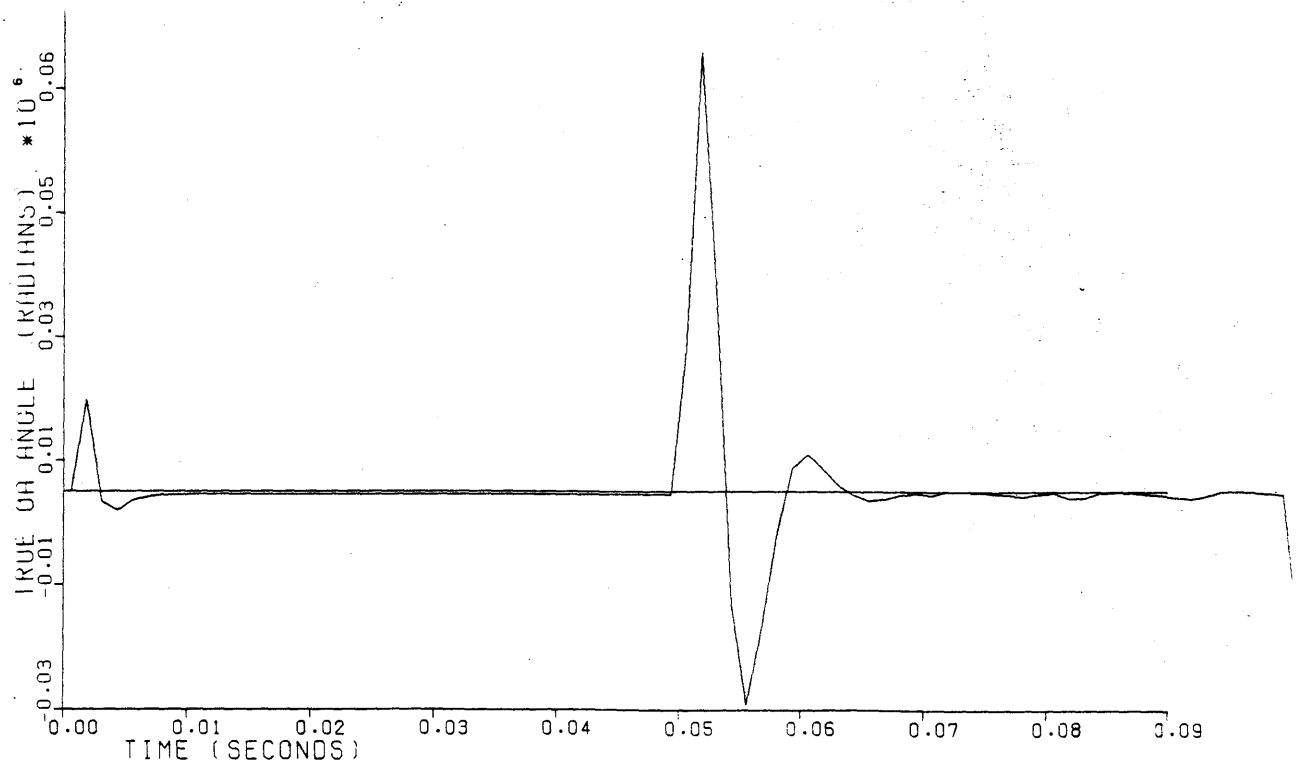


Figure 12-15. Run #6 24 Pt. Average Position vs. Time (15°/hr Step Input)

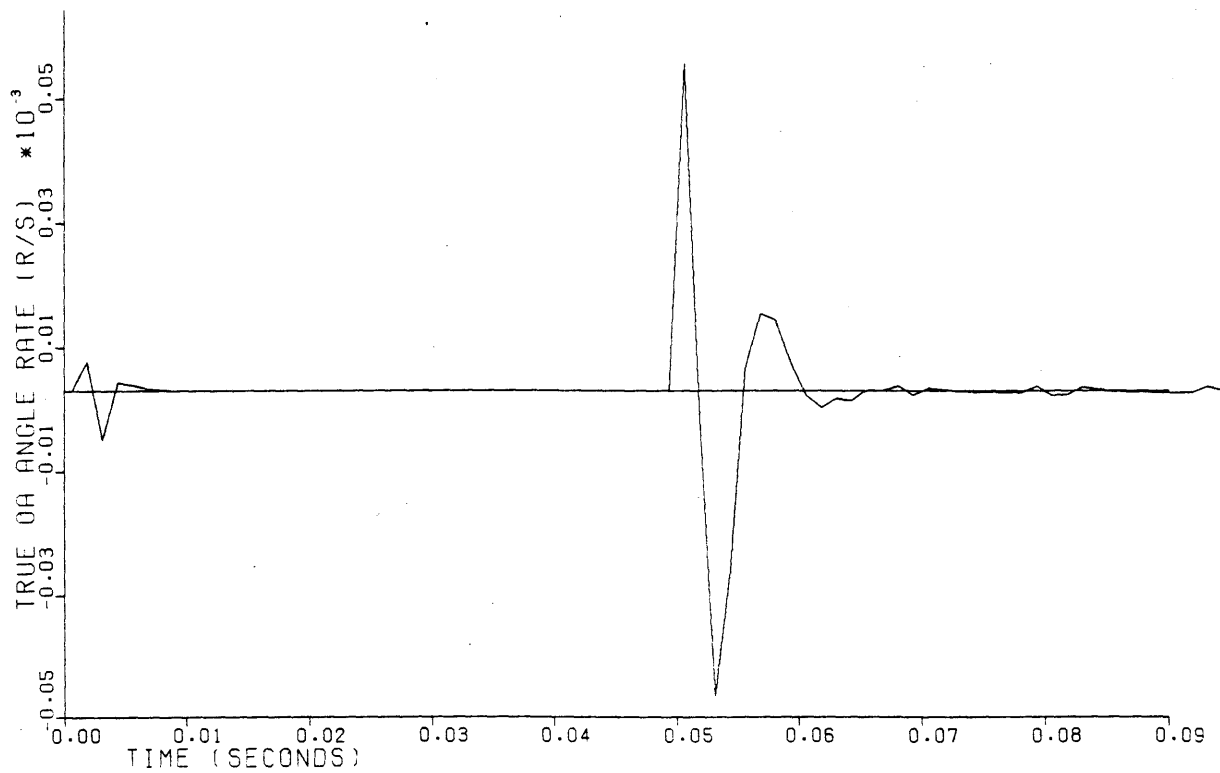


Figure 12-16. Run #6 24 Pt. Average Velocity vs. Time (15°/hr Step Input)

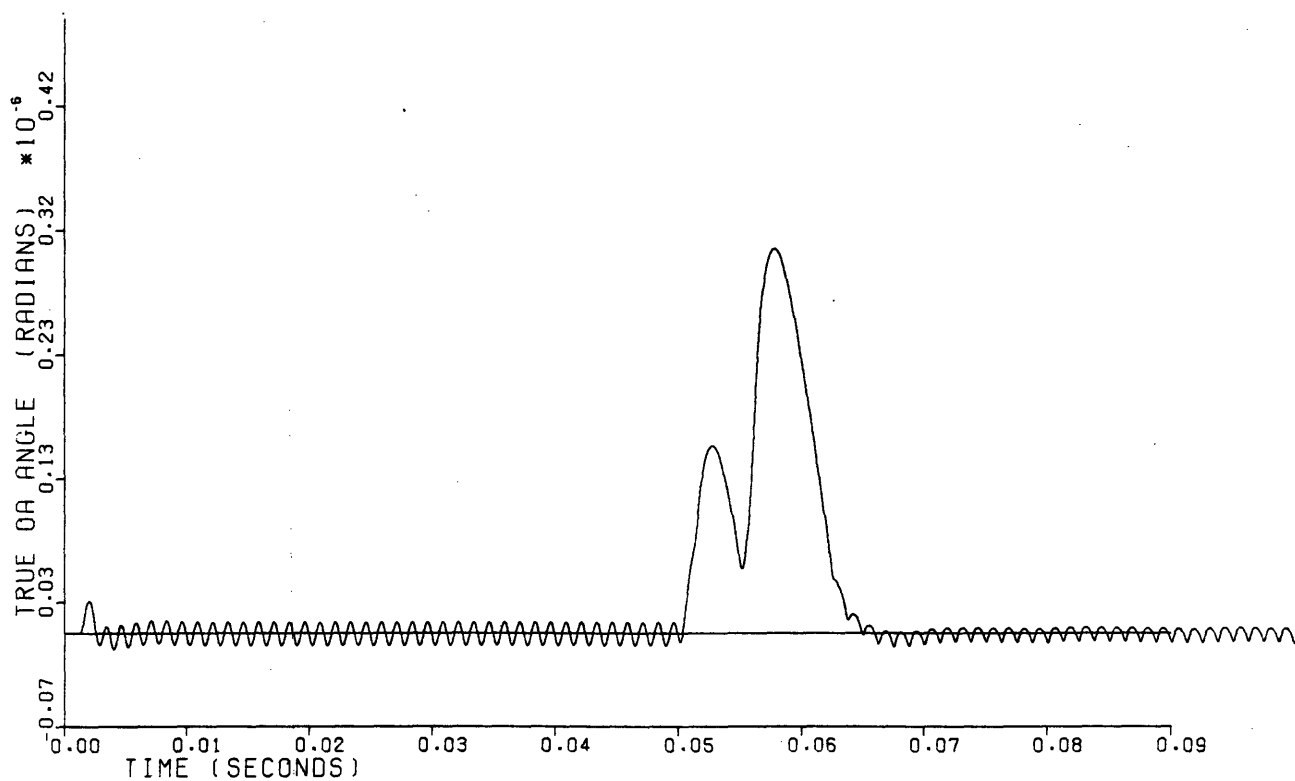


Figure 12-17. Run #7 Float Position vs. Time (15°/hr Step Input)

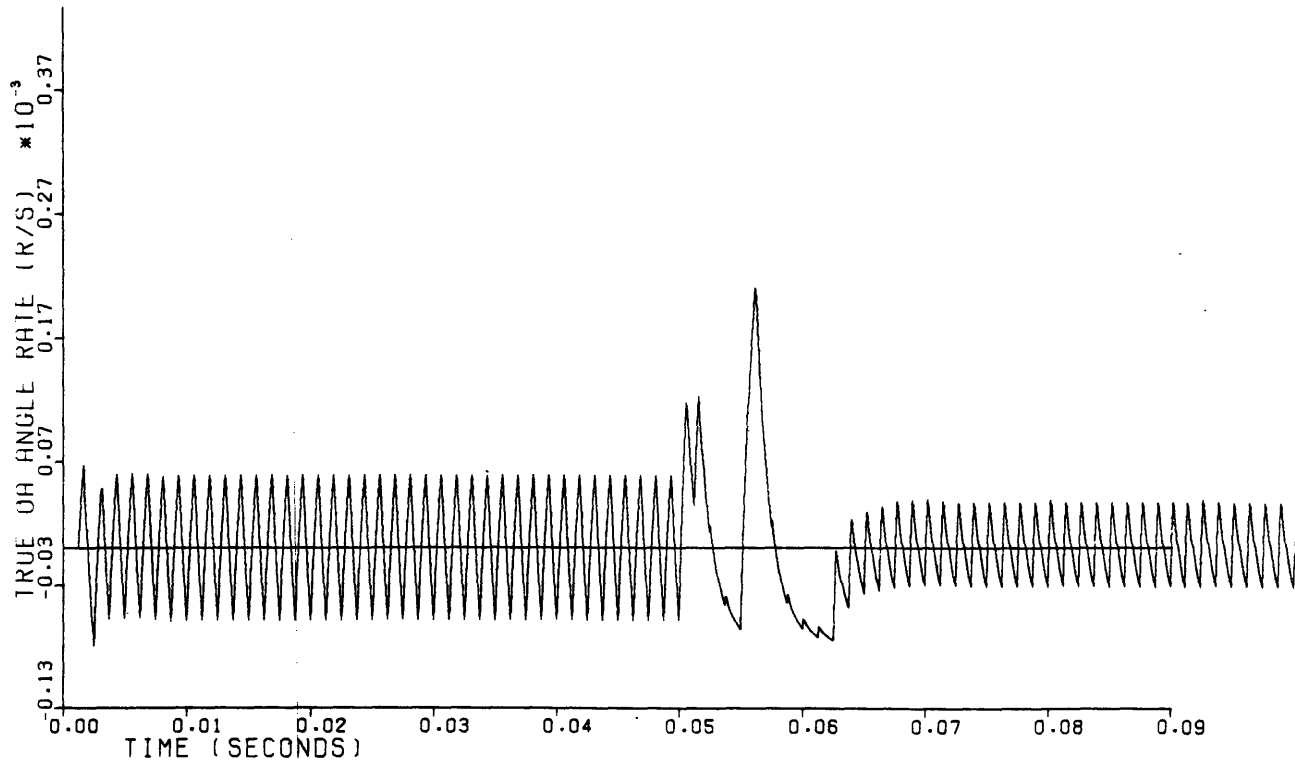


Figure 12-18. Run #7 Float Velocity vs. Time (15°/hr Step Input)

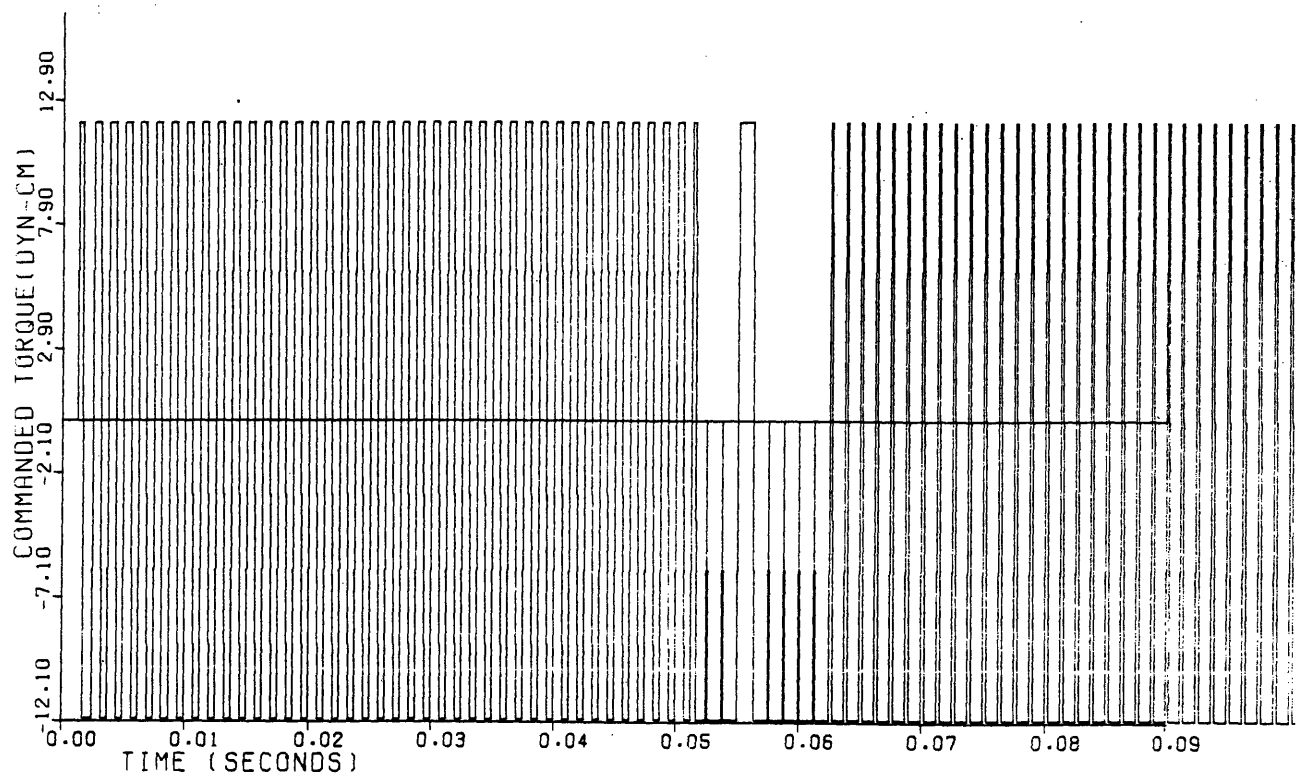


Figure 12-19. Run #7 Commanded Torque vs. Time (15°/hr Step Input)

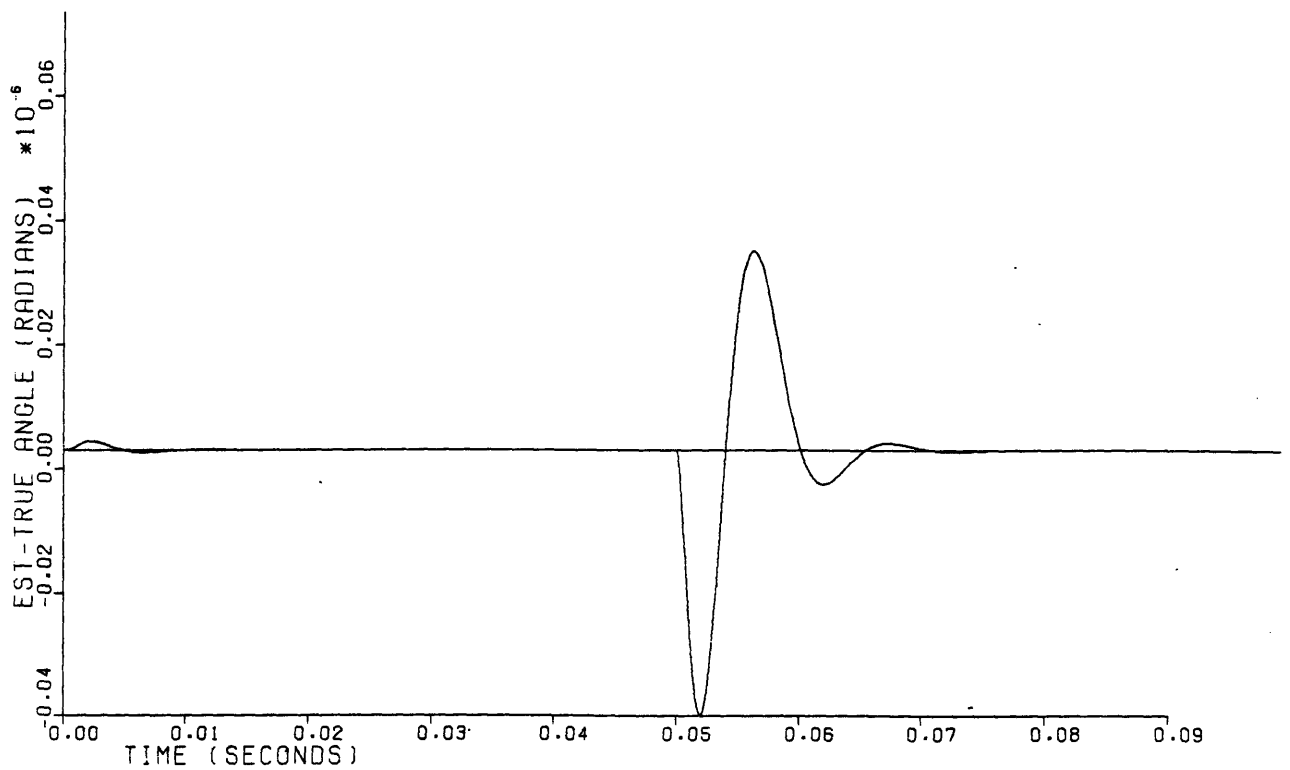


Figure 12-20. Run #7 Position Estimation Error vs. Time (15°/hr Step Input)

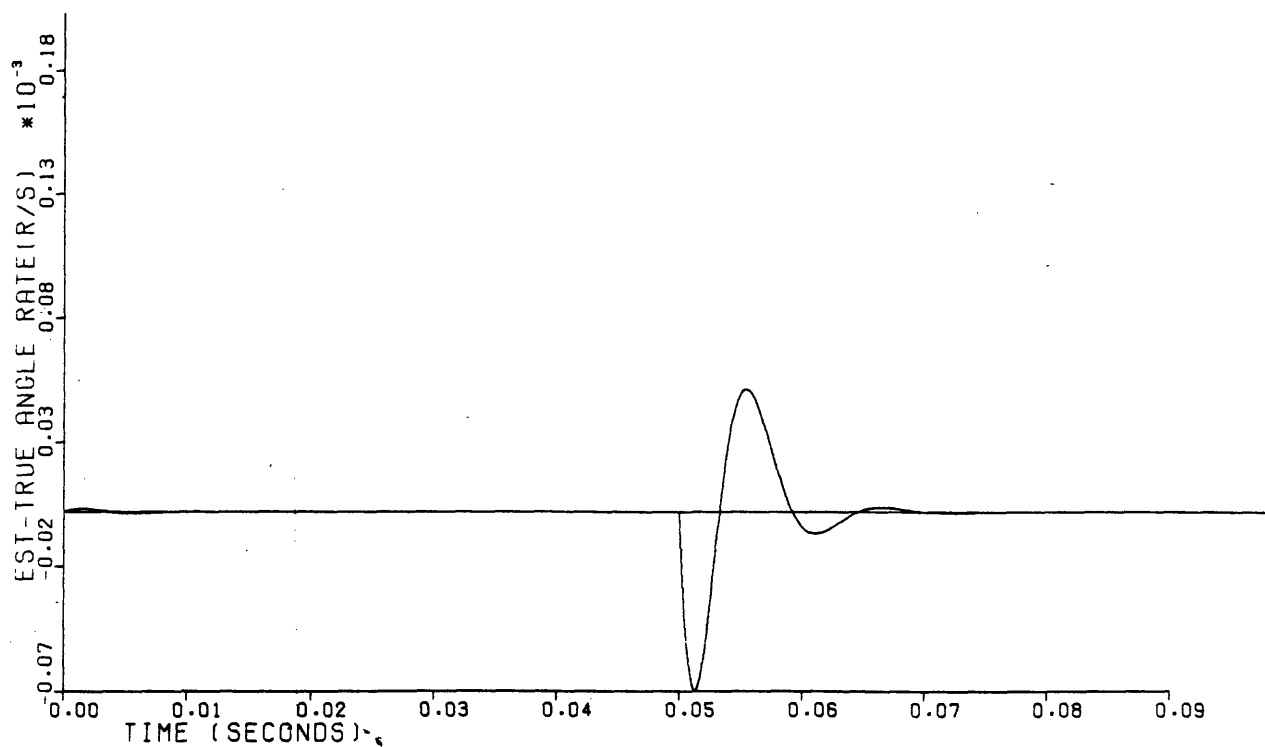


Figure 12-21. Run #7 Velocity Estimation Error vs. Time (15°/hr Step Input)

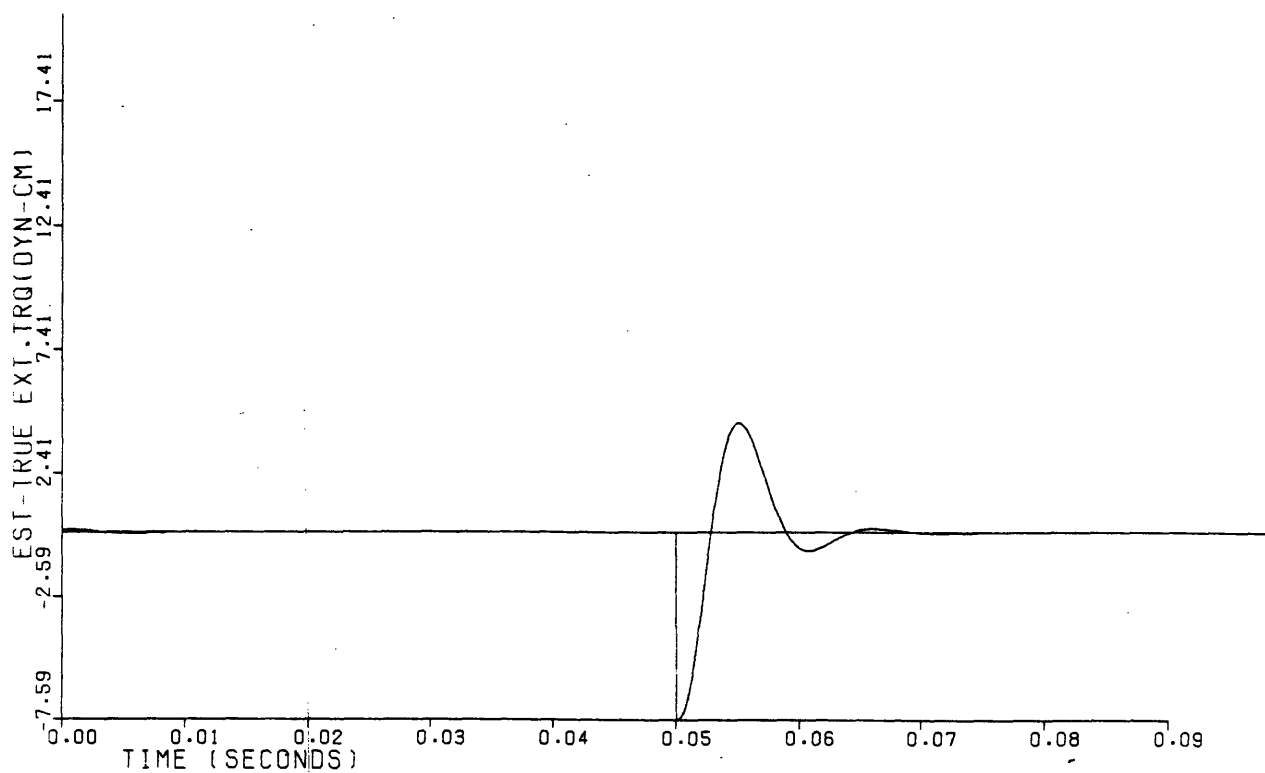


Figure 12-22. Run #7 External Torque Estimation Error vs. Time (15°/hr Step Input)

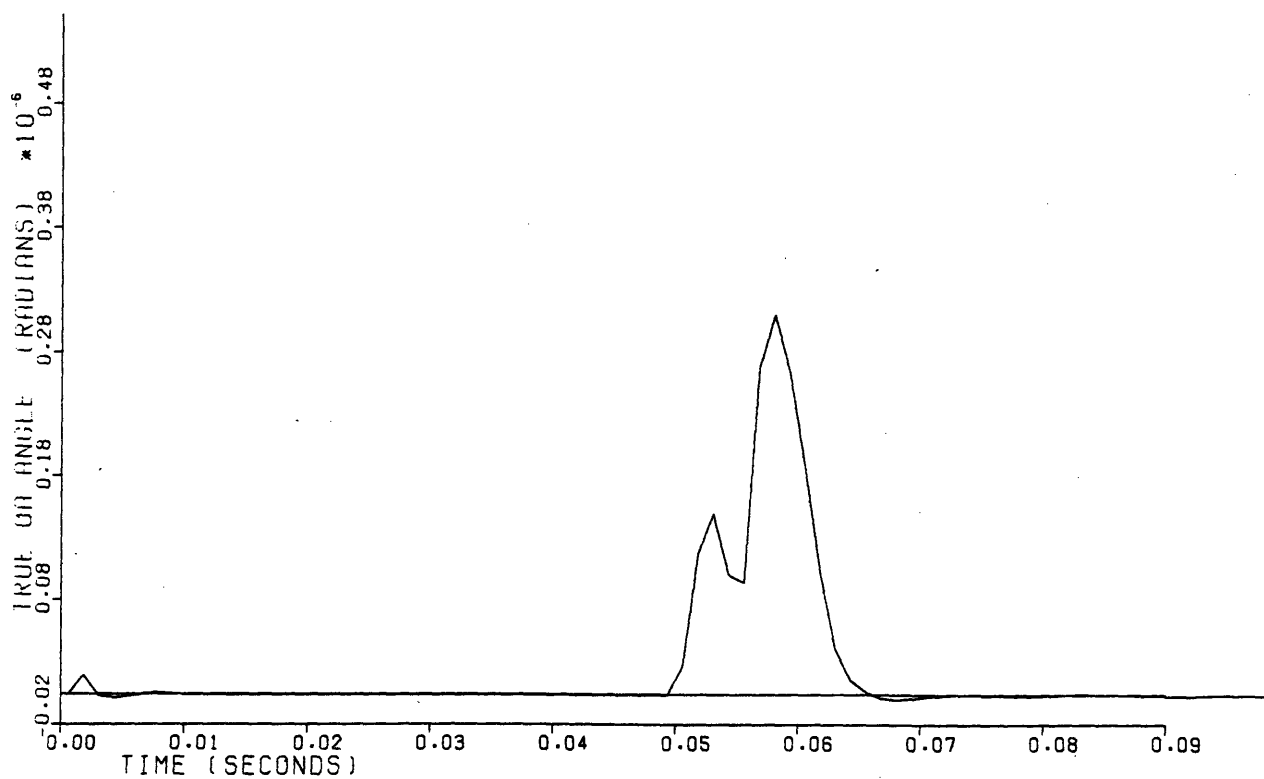


Figure 12-23. Run #7 24 Pt. Average Position vs. Time (15°/hr Step Input)

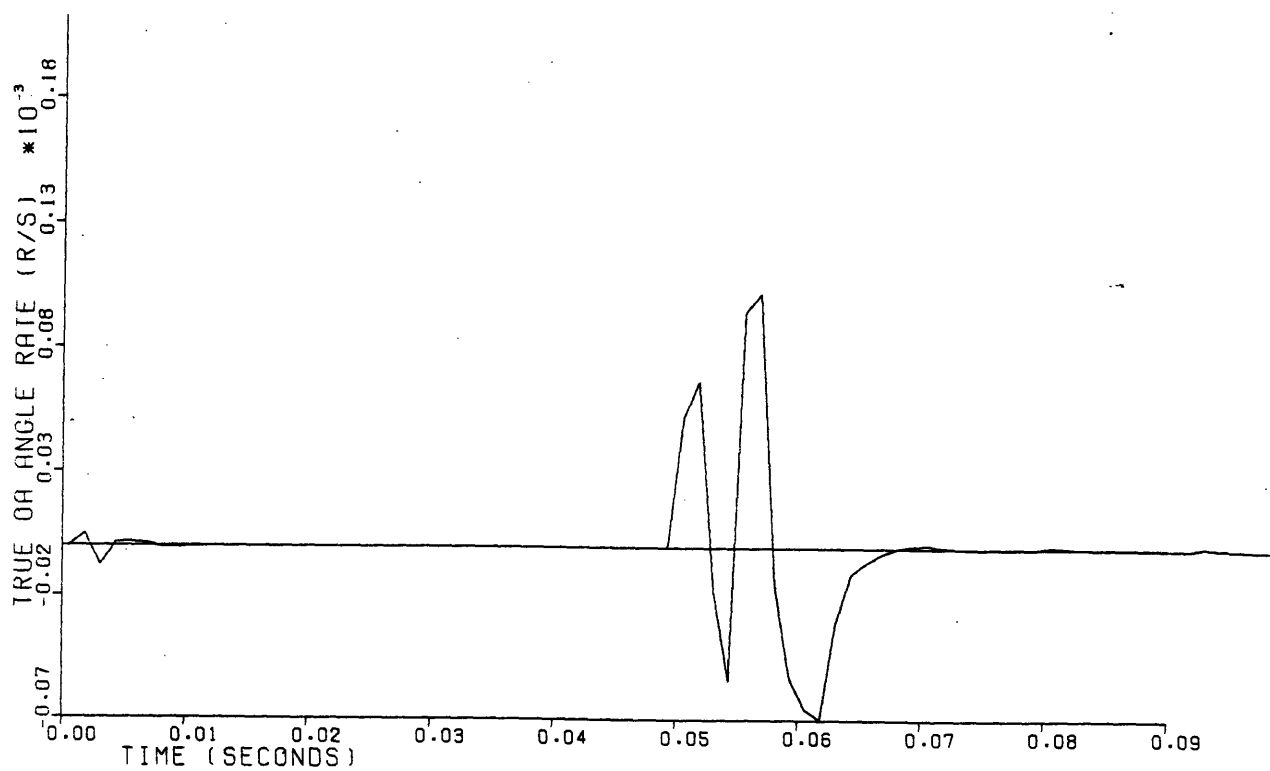


Figure 12-24. Run #7 24 Pt. Average Velocity vs. Time (15°/hr Step Input)

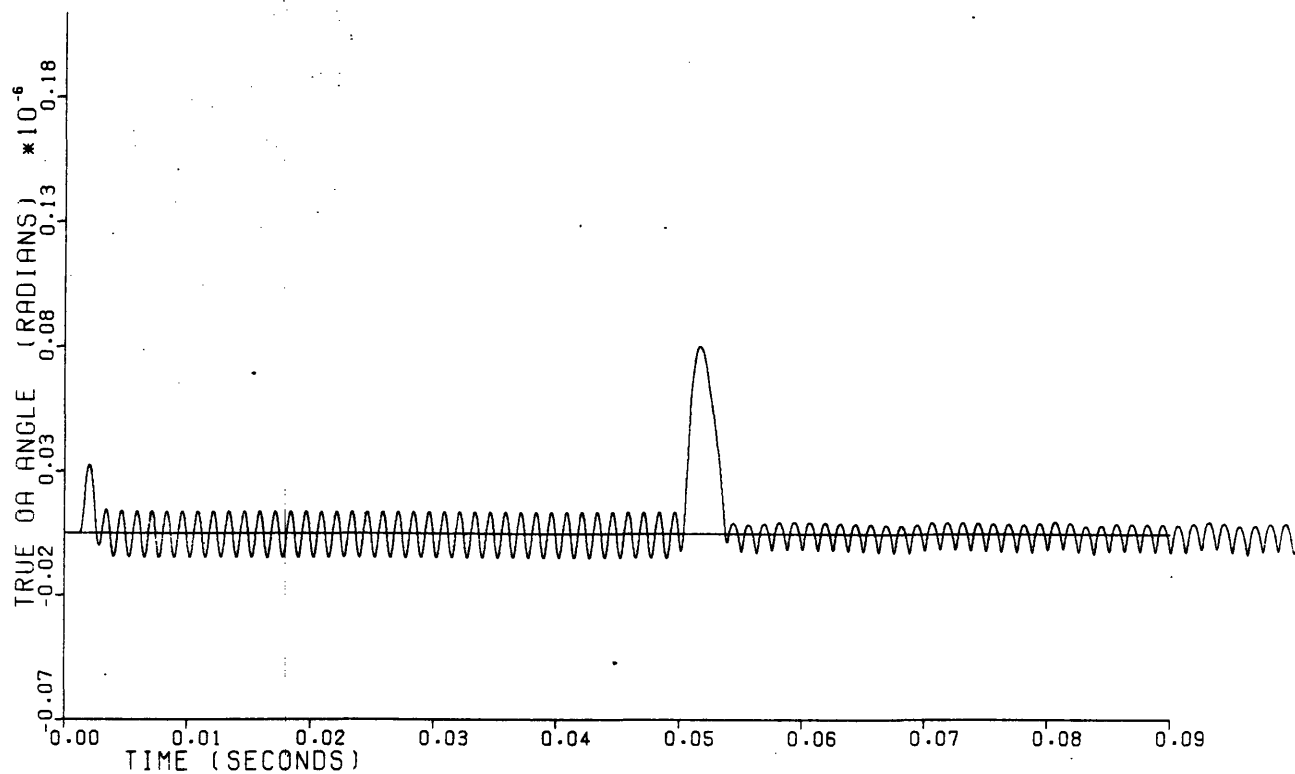


Figure 12-25. Run #8 Float Position vs. Time (15°/hr Step Input)

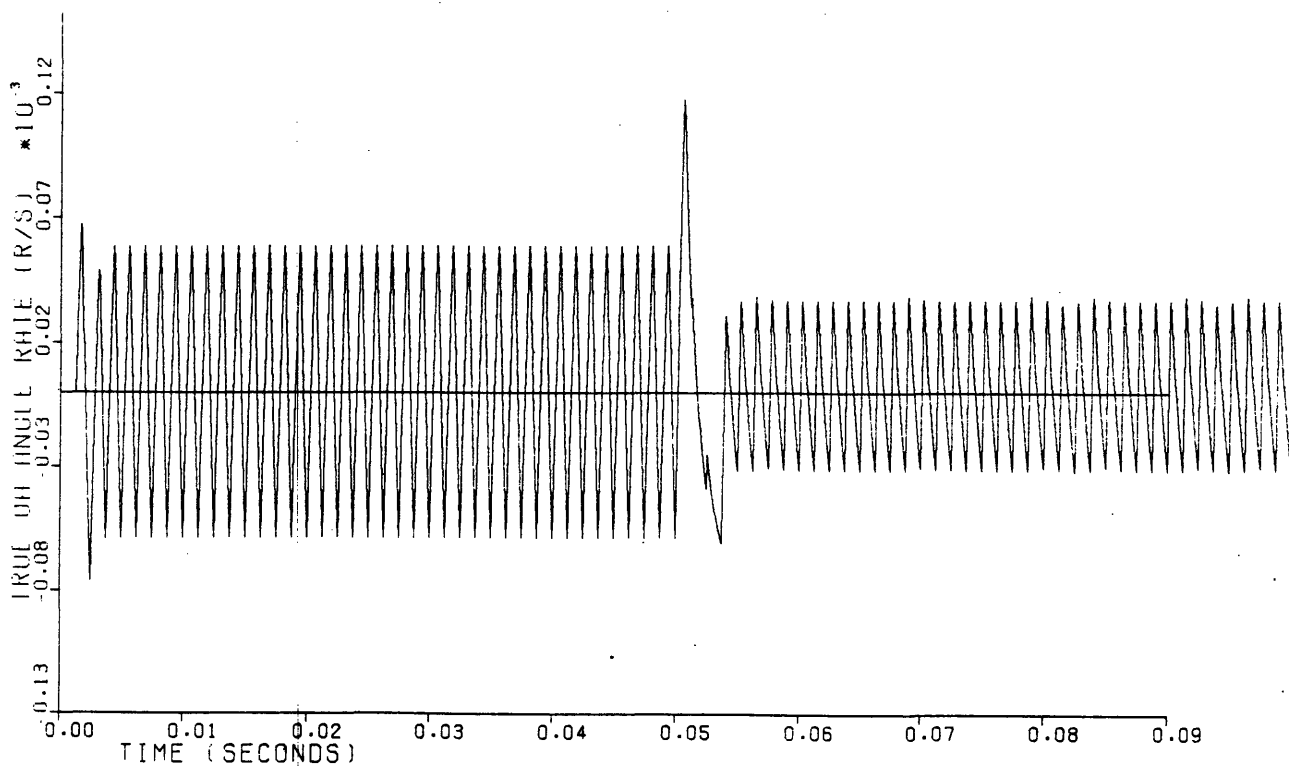


Figure 12-26. Run #8 Float Velocity vs. Time (15°/hr Step Input)

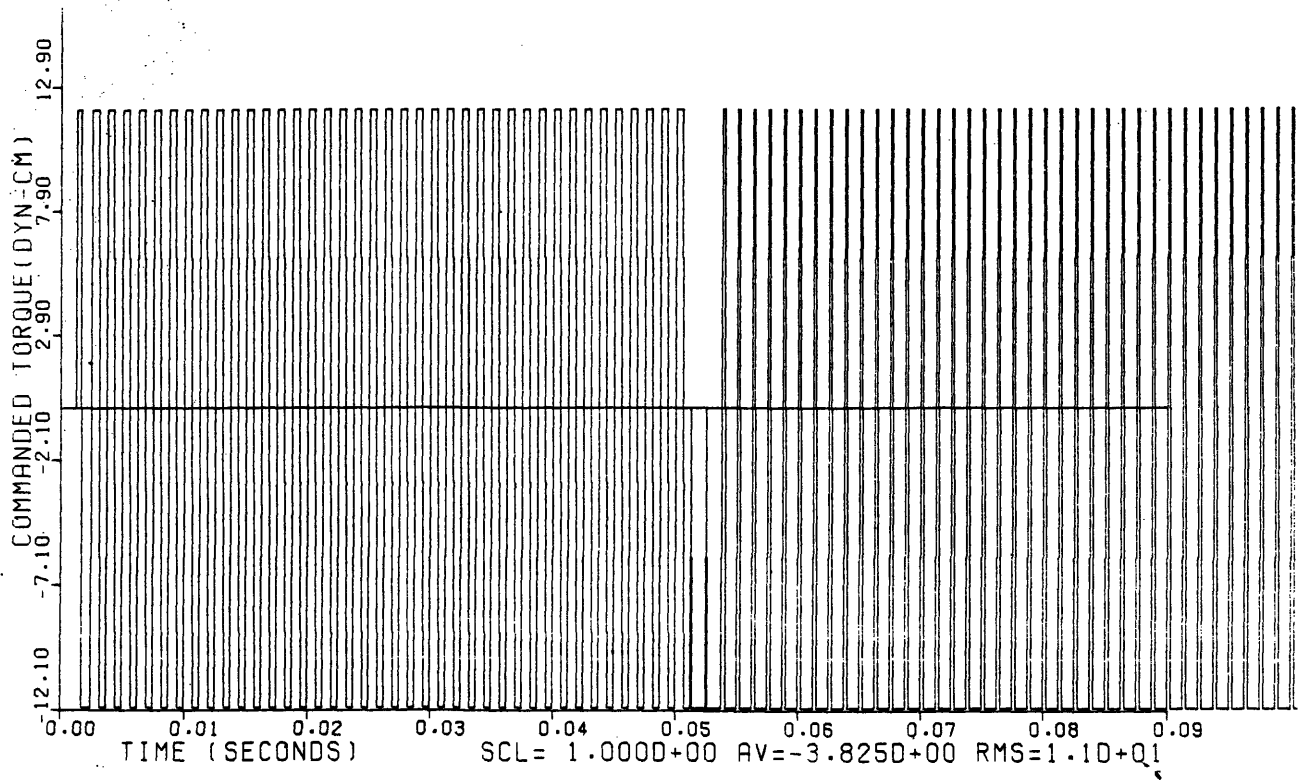


Figure 12-27. Run #8 Commanded Torque vs. Time (15°/hr Step Input)

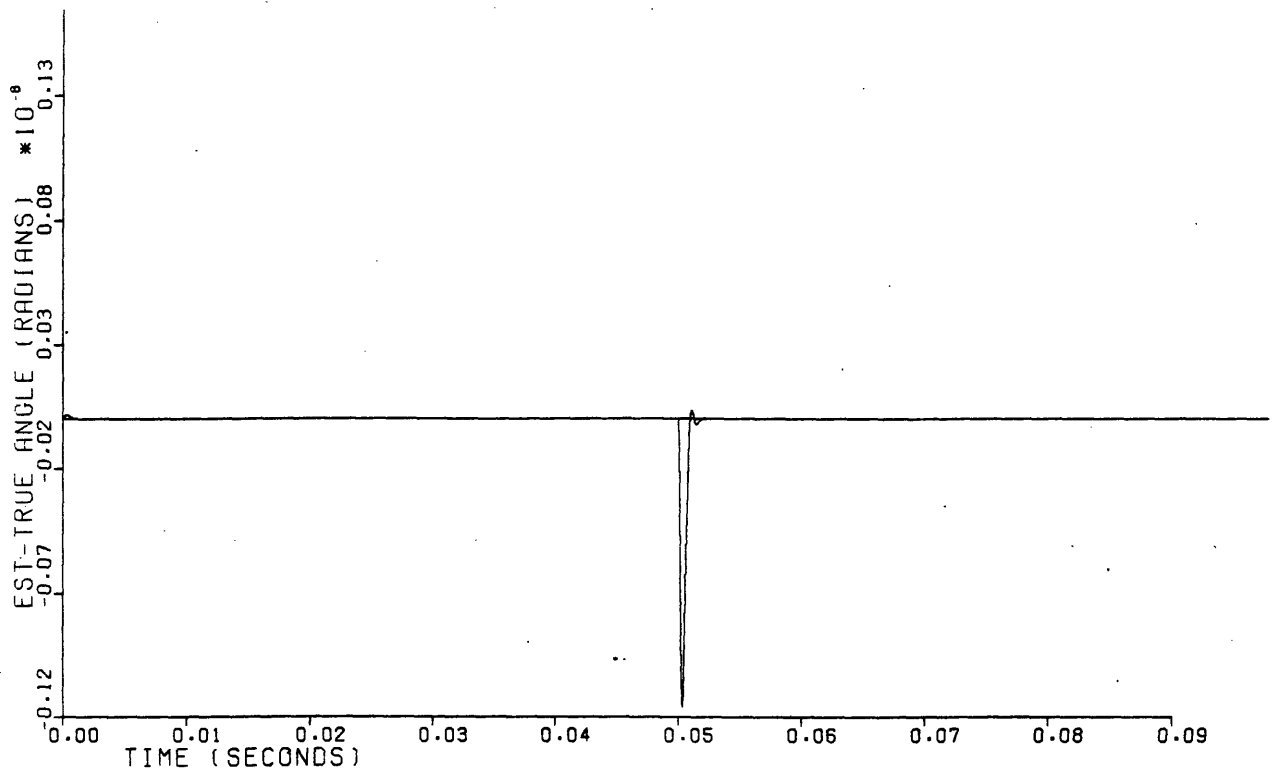


Figure 12-28. Run #8 Position Estimation Error vs. Time (15°/hr Step Input)

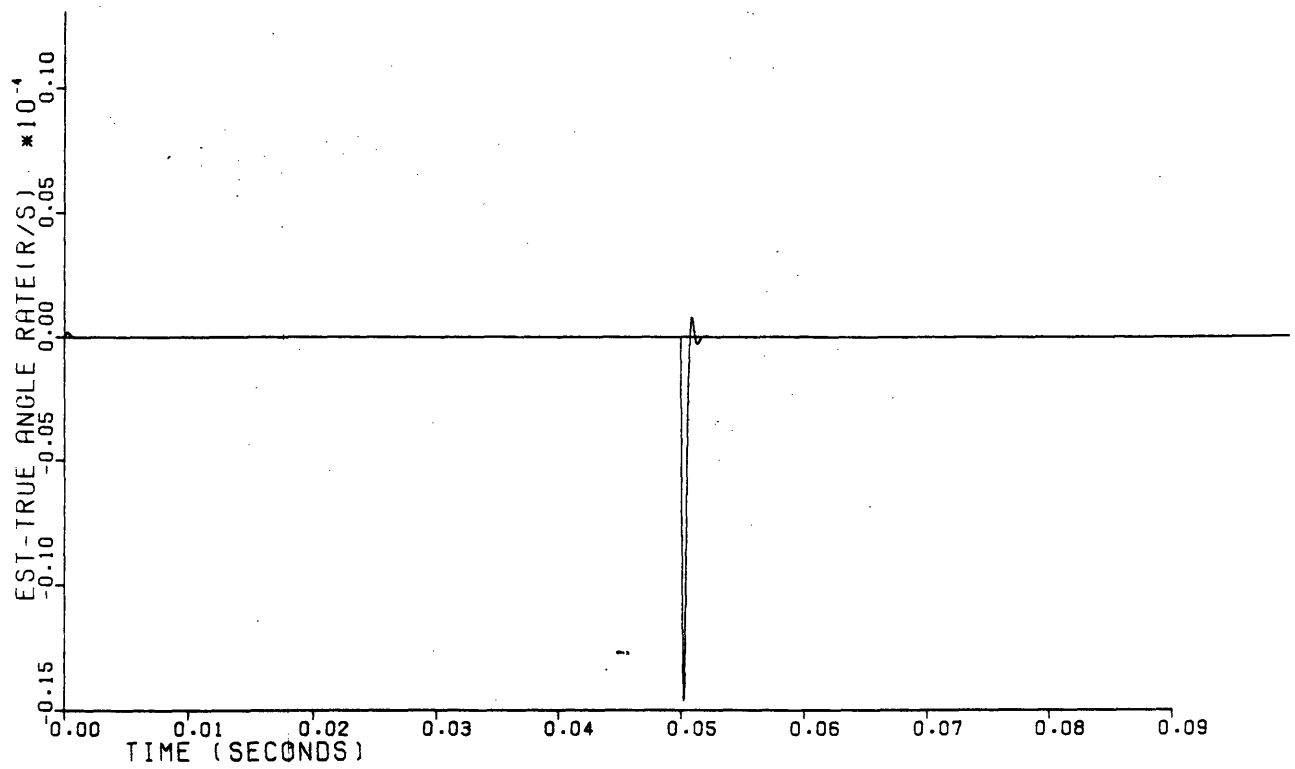


Figure 12-29. Run #8 Velocity Estimation Error vs. Time (15°/hr Step Input)

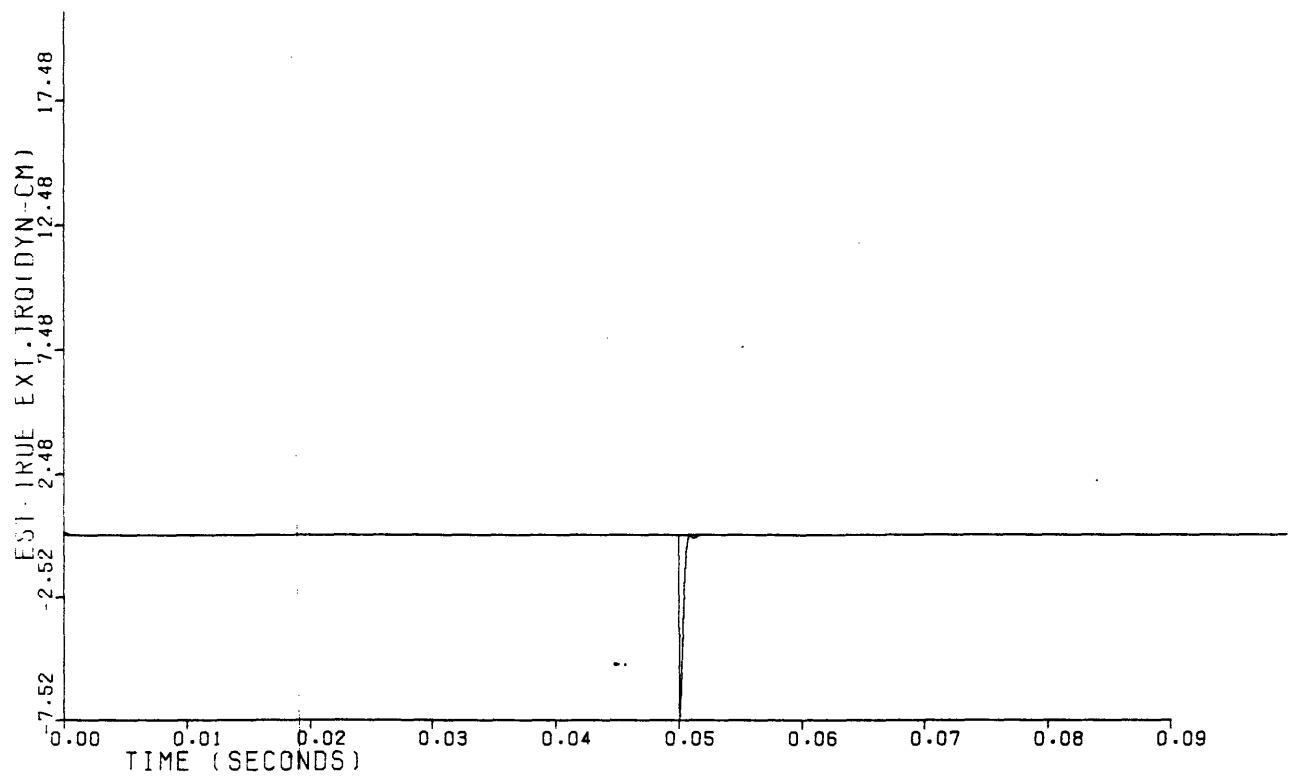


Figure 12-30. Run #8 External Torque Estimation Error vs. Time (15°/hr Step Input)

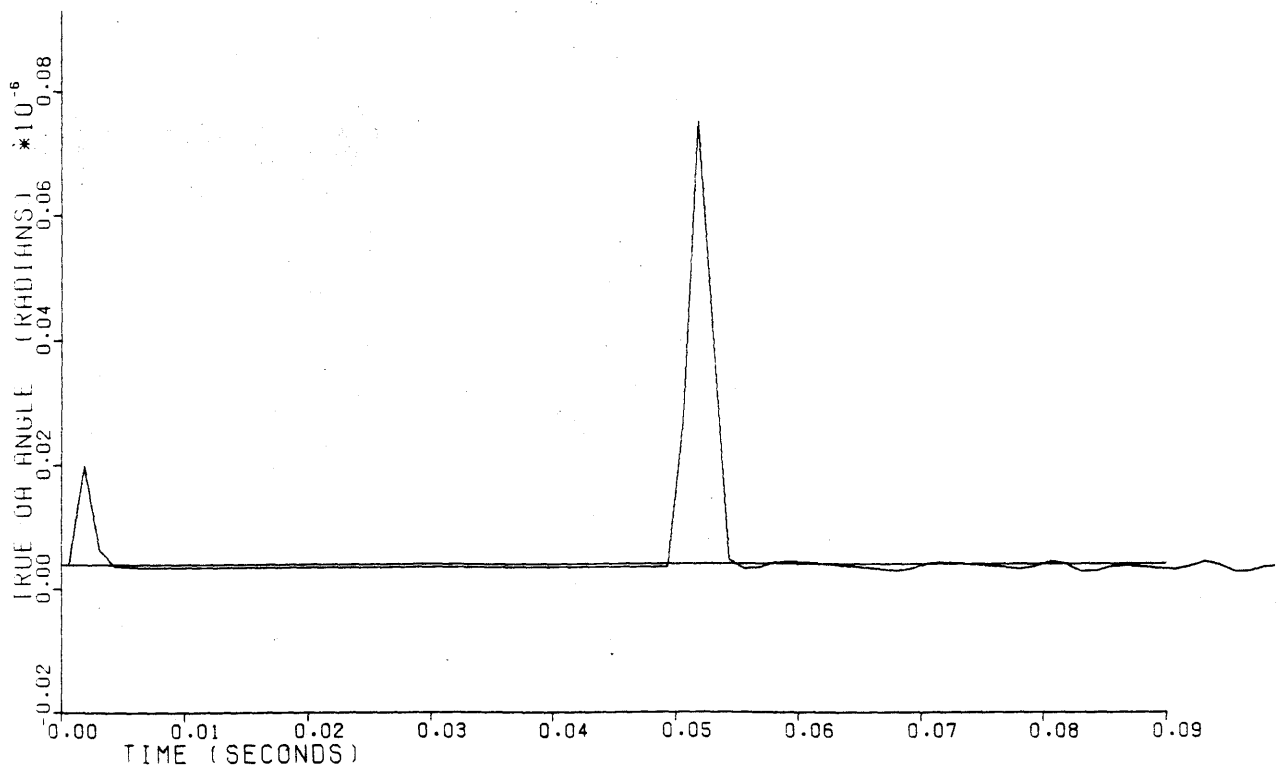


Figure 12-31. Run #8 24 Pt. Average Position vs. Time (15°/hr Step Input)

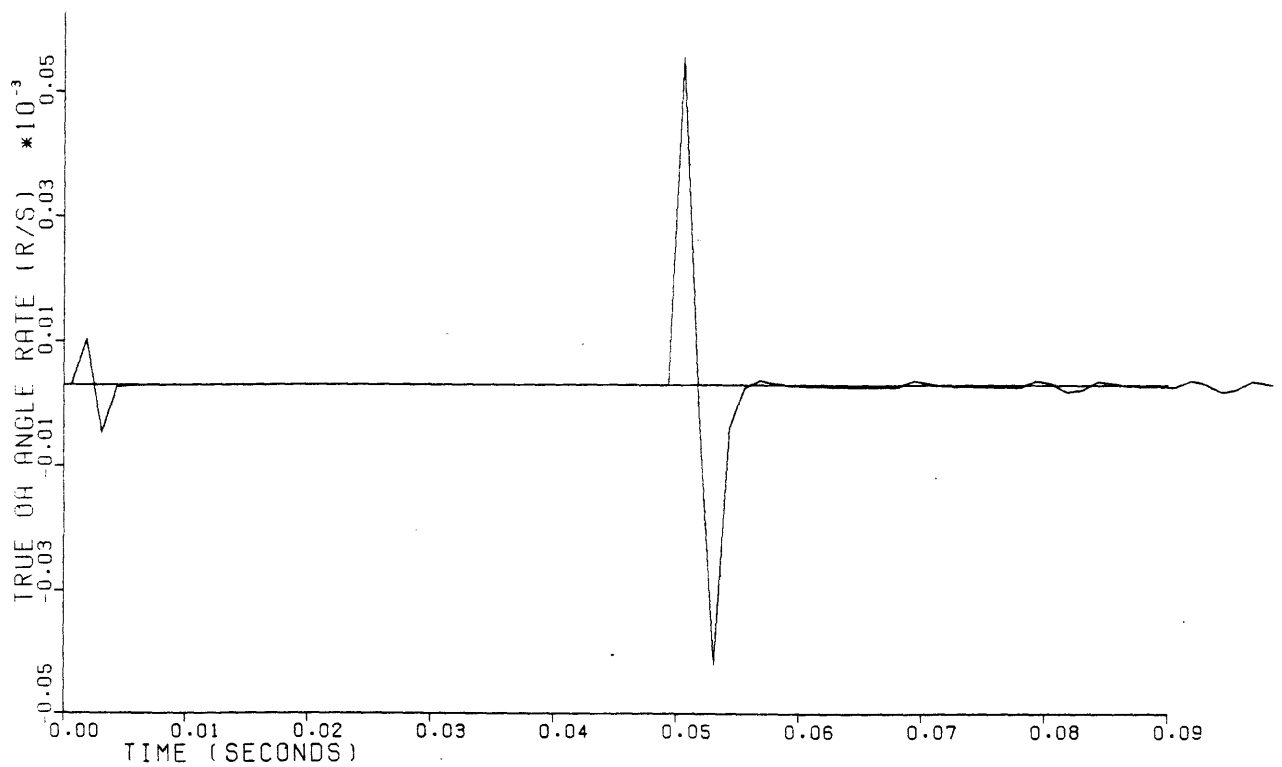


Figure 12-32. Run #8 24 Pt. Average Velocity vs. Time (15°/hr Step Input)

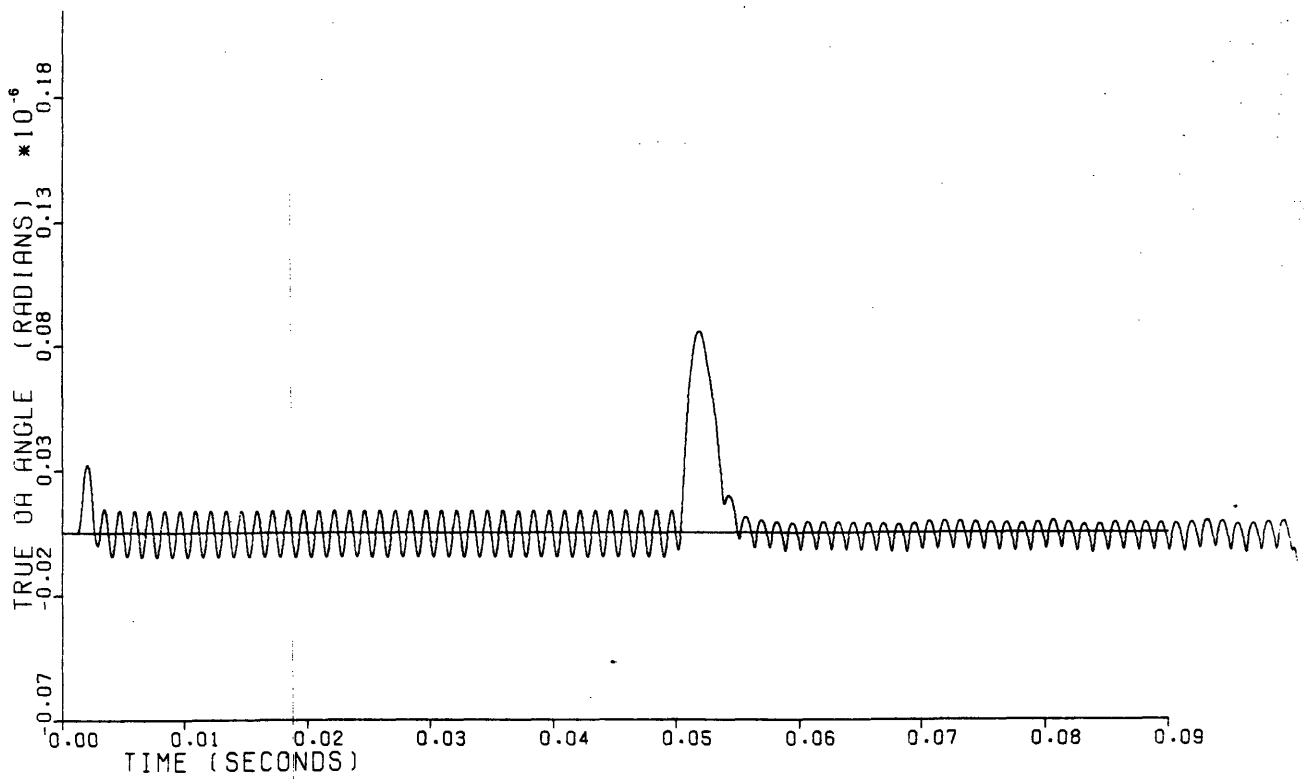


Figure 12-33. Run #9 Float Position vs. Time (15°/hr Step Input)

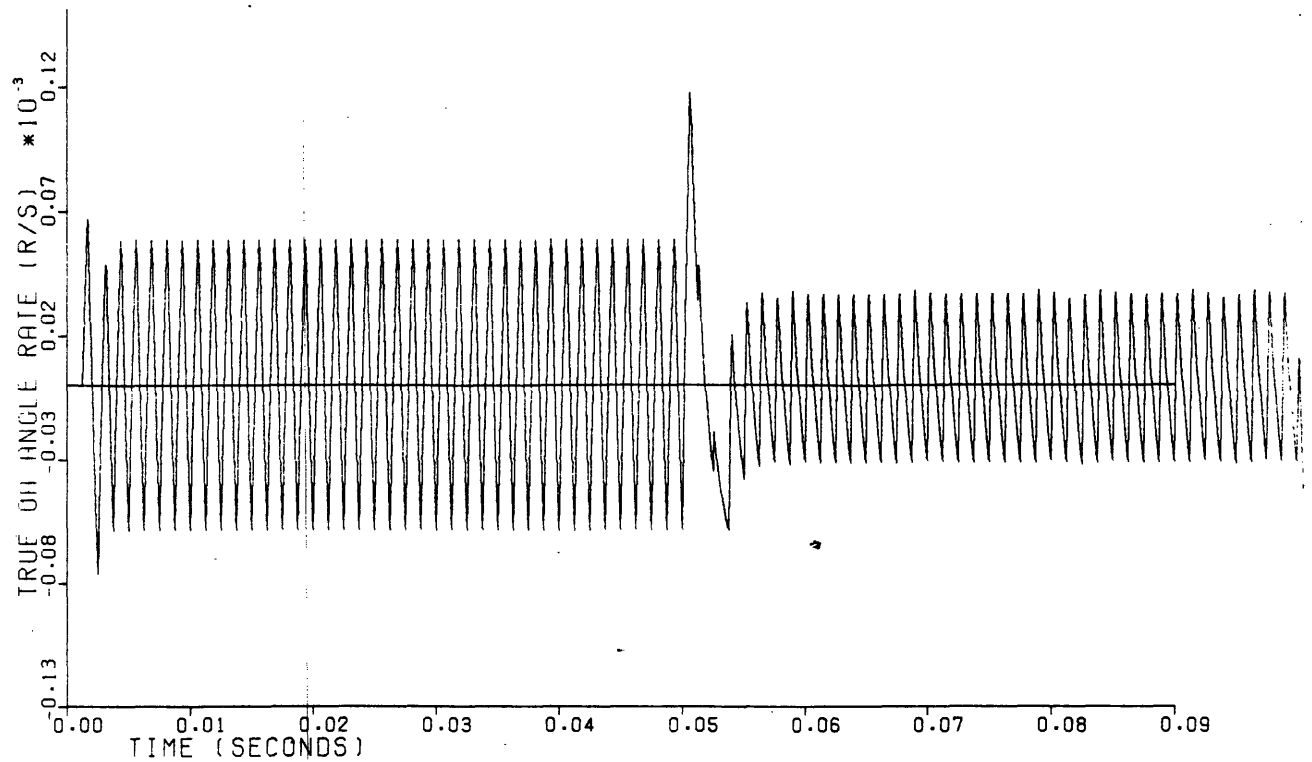


Figure 12-34. Run #9 Float Velocity vs. Time (15°/hr Step Input)

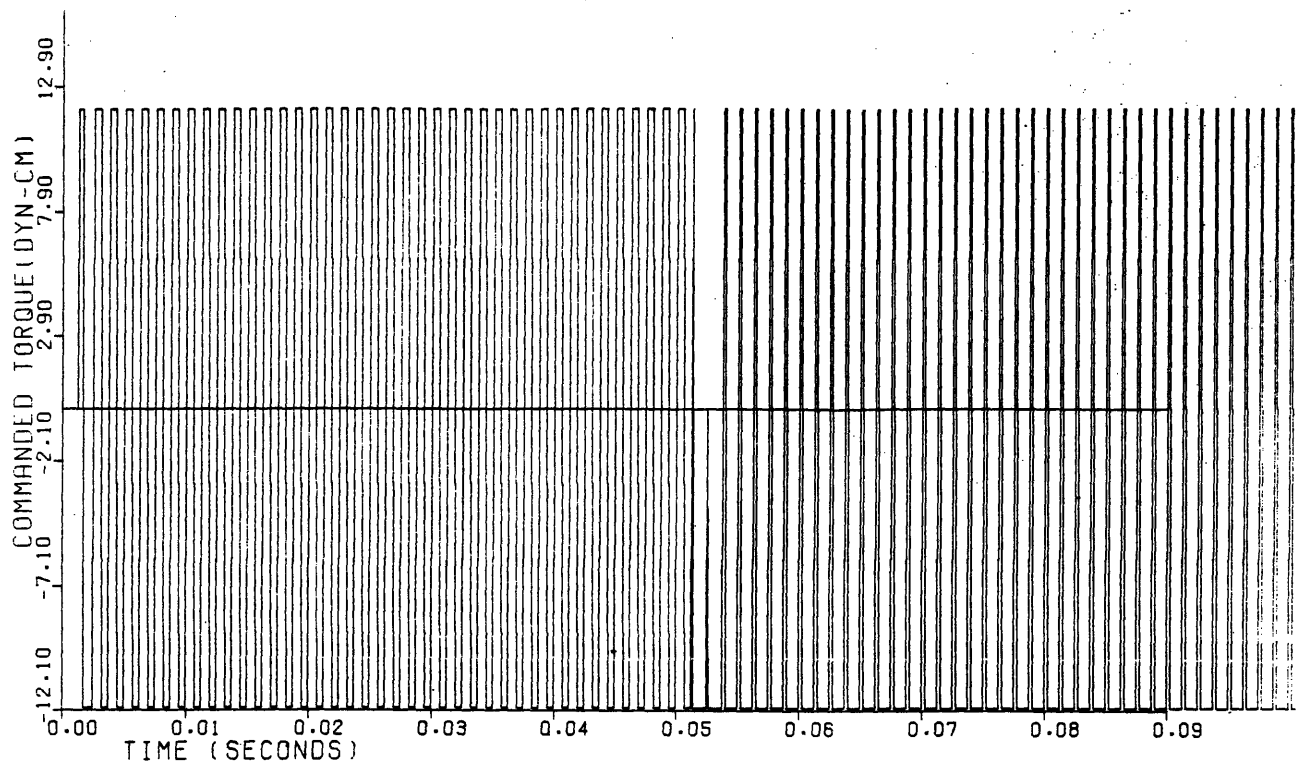


Figure 12-35. Run #9 Commanded Torque vs. Time (15°/hr Step Input)

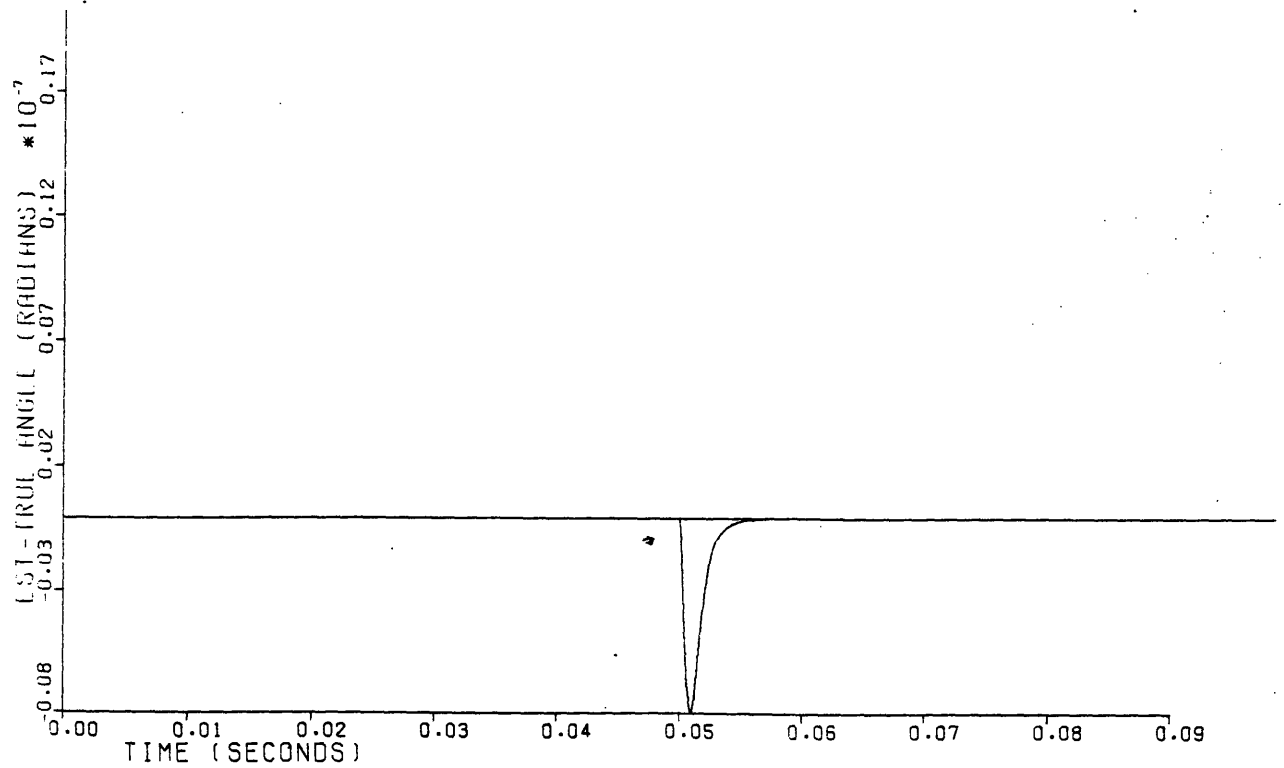


Figure 12-36. Run #9 Position Estimation Error vs. Time (15°/hr Step Input)

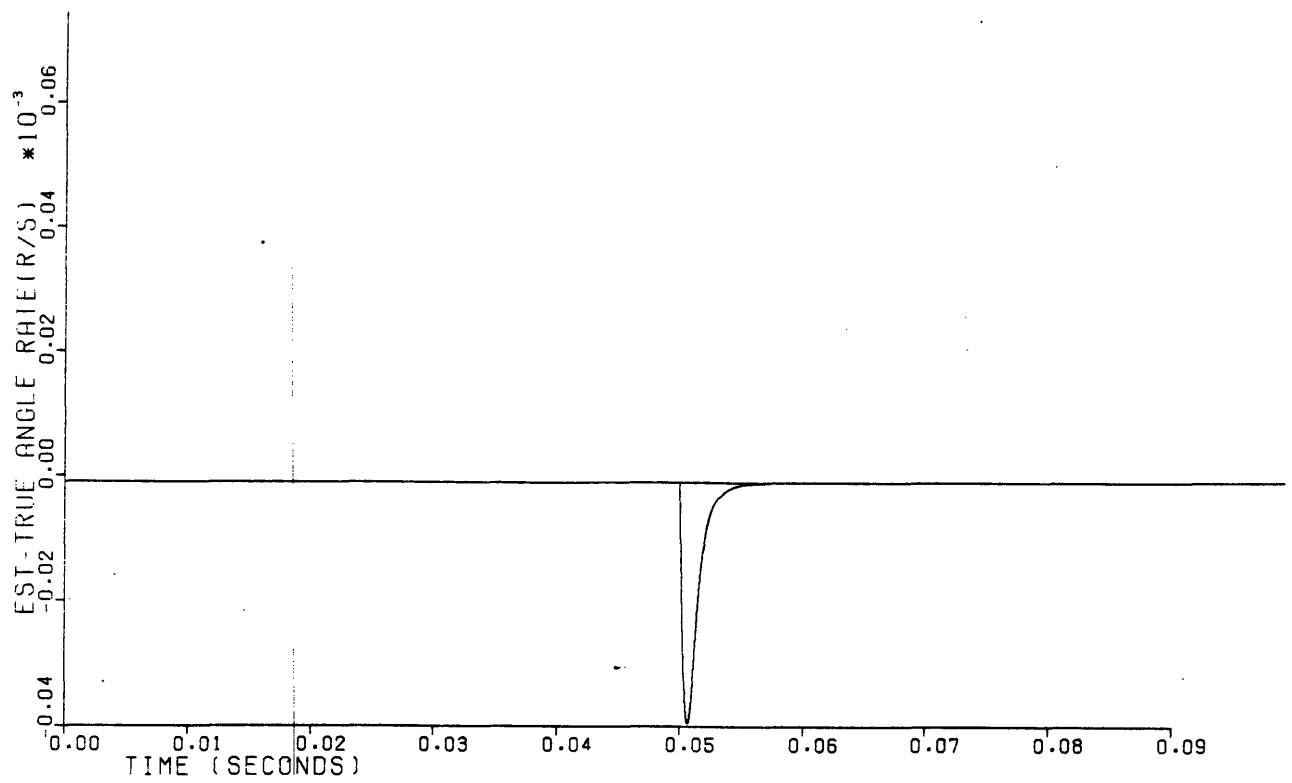


Figure 12-37. Run #9 Velocity Estimation Error vs. Time (15°/hr Step Input)

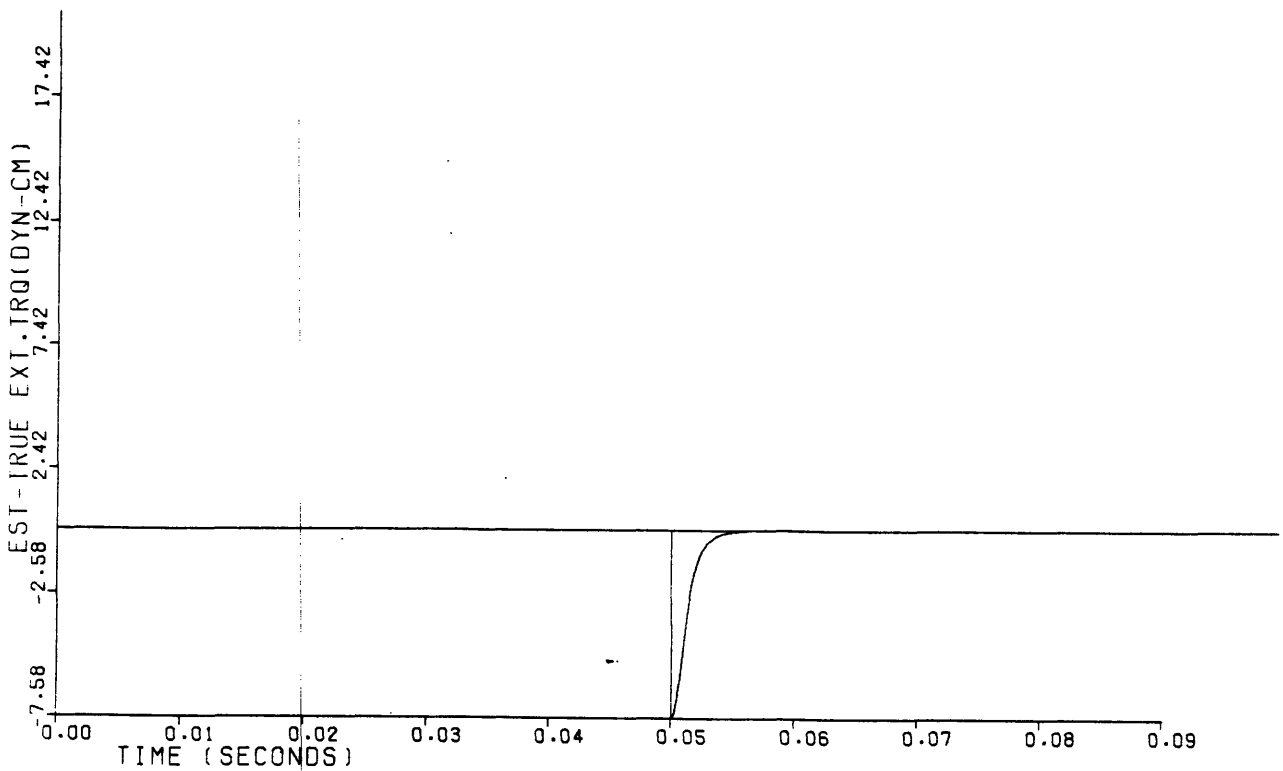


Figure 12-38. Run #9 External Torque Estimation Error vs. Time (15°/hr Step Input)

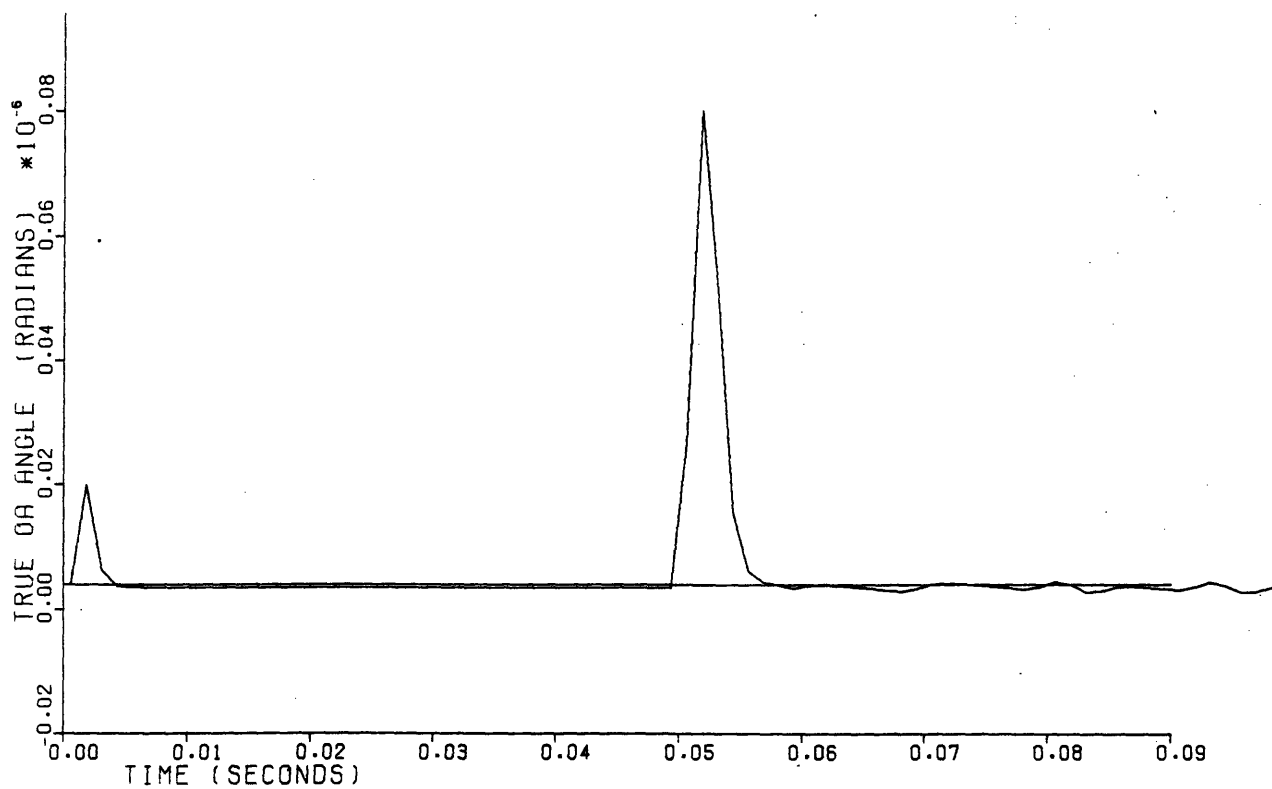


Figure 12-39. Run #9 24 Pt. Average Position vs. Time (15°/hr Step Input)

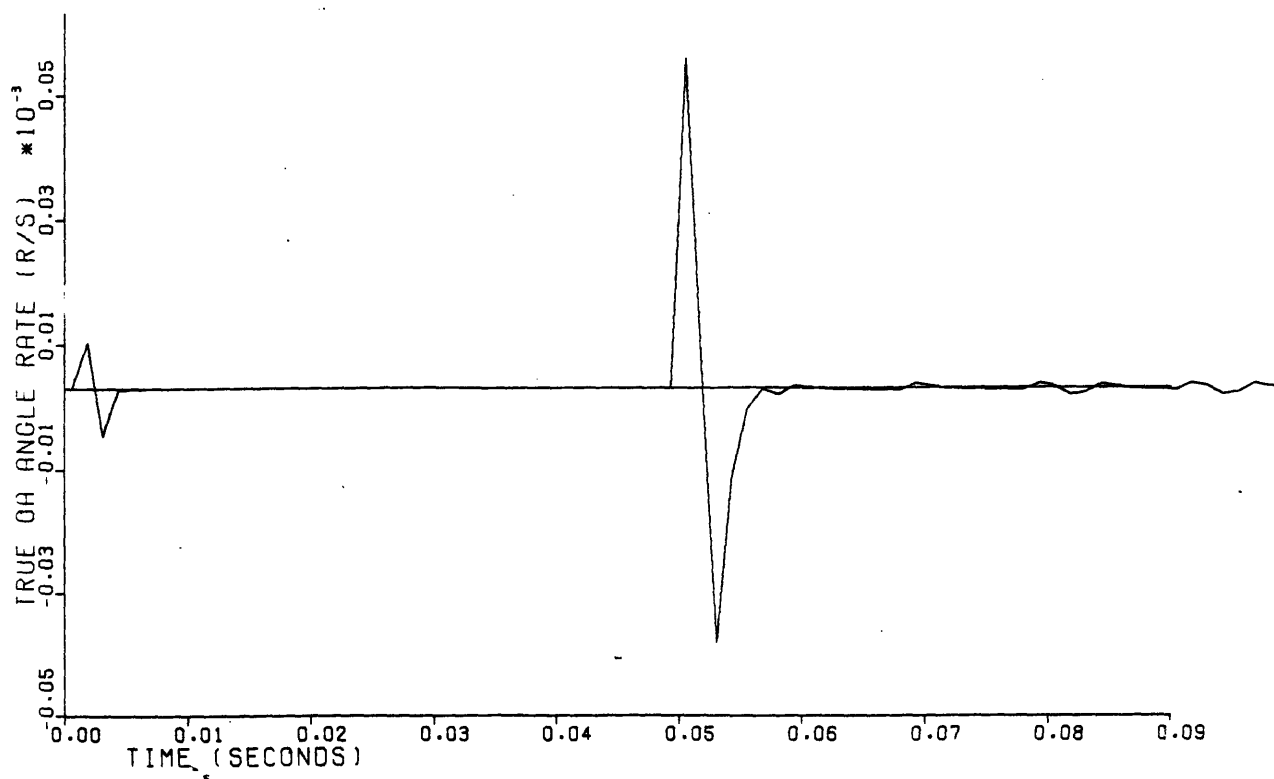


Figure 12-40. Run #9 24 Pt. Average Velocity vs. Time (15°/hr Step Input)

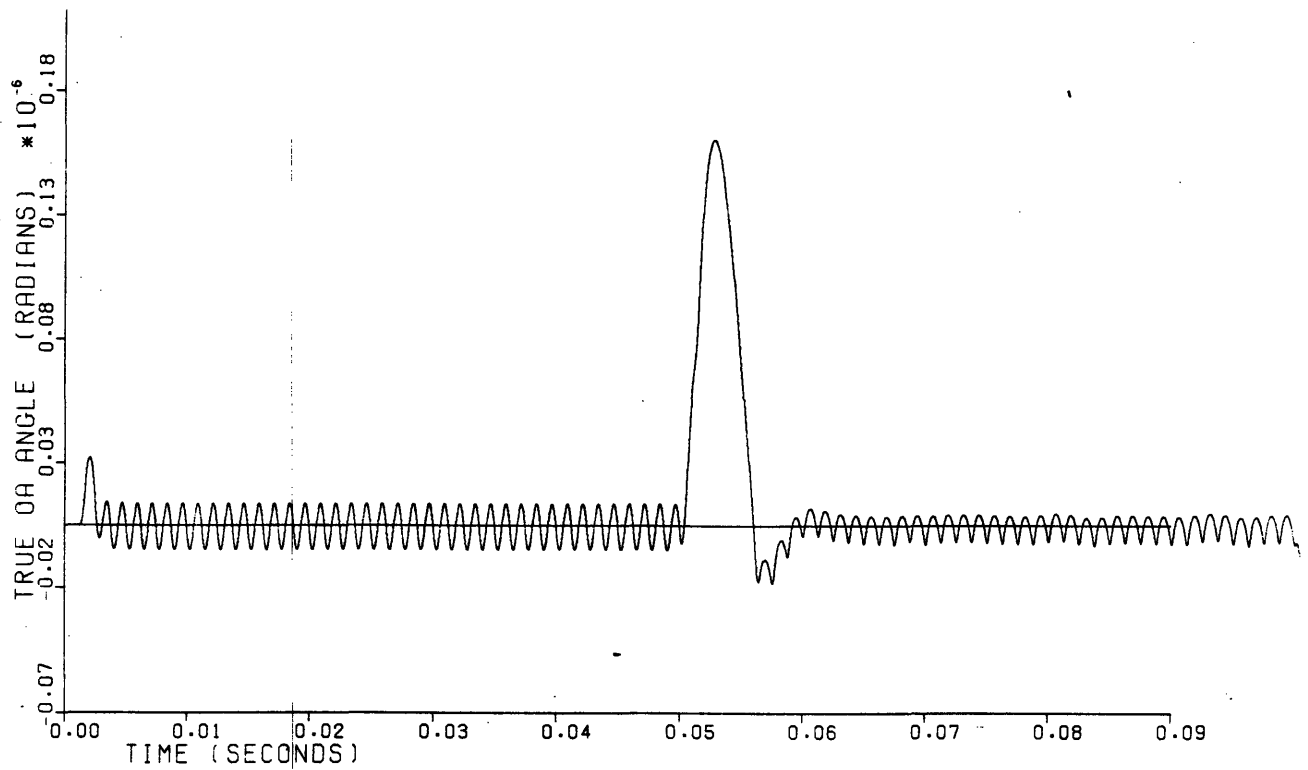


Figure 12-41. Run #10 Float Position vs. Time (15°/hr Step Input)

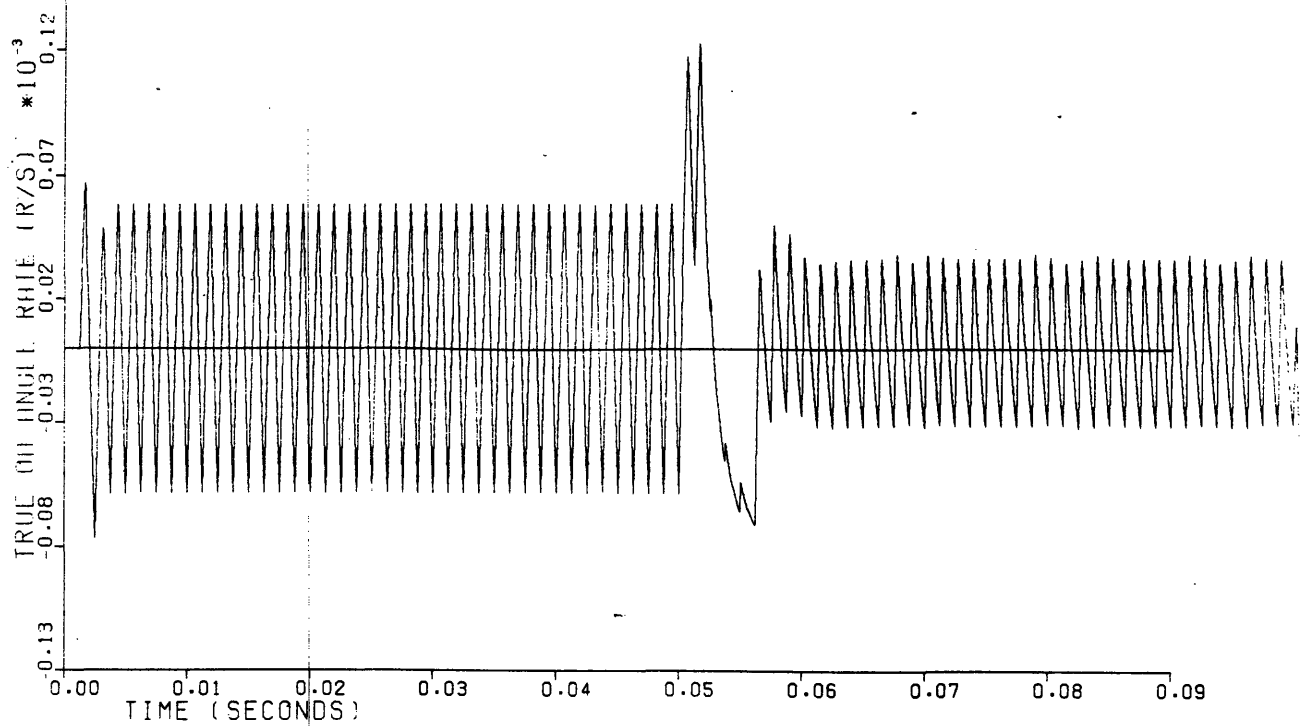


Figure 12-42. Run #10 Float Velocity vs. Time (15°/hr Step Input)

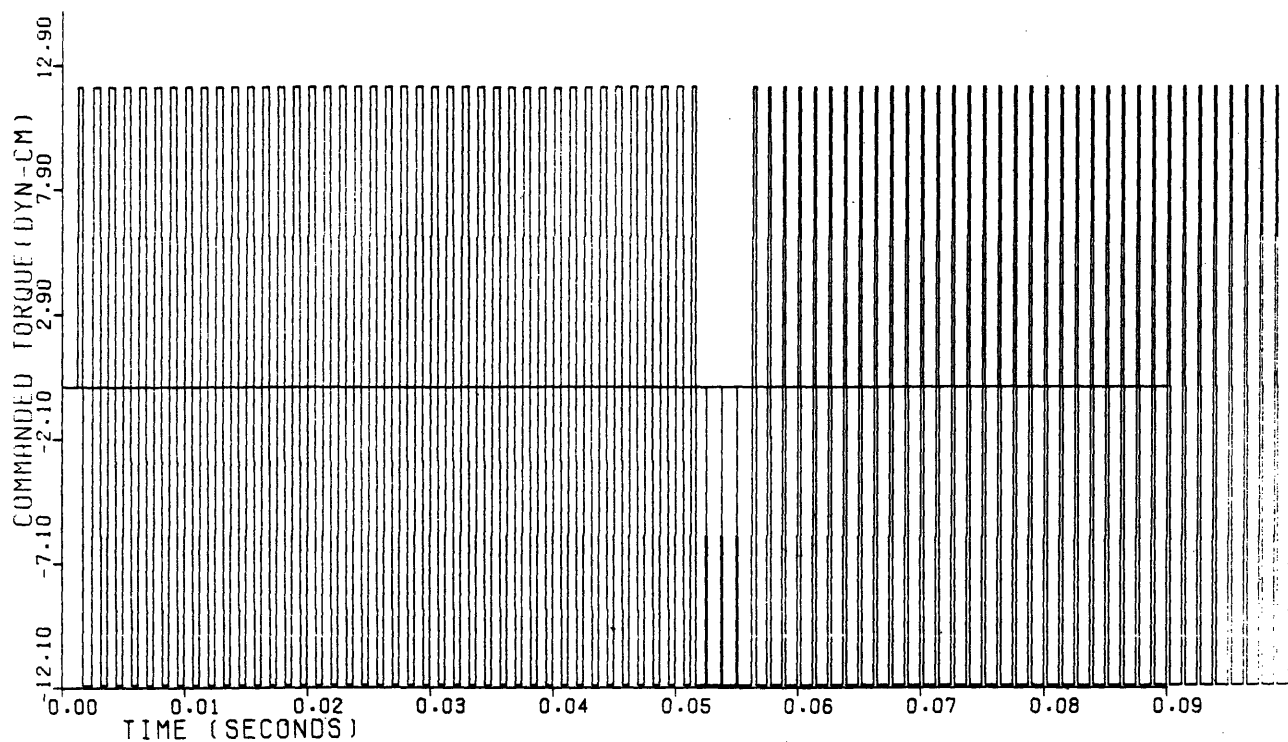


Figure 12-43. Run #10 Commanded Torque vs. Time (15°/hr Step Input)

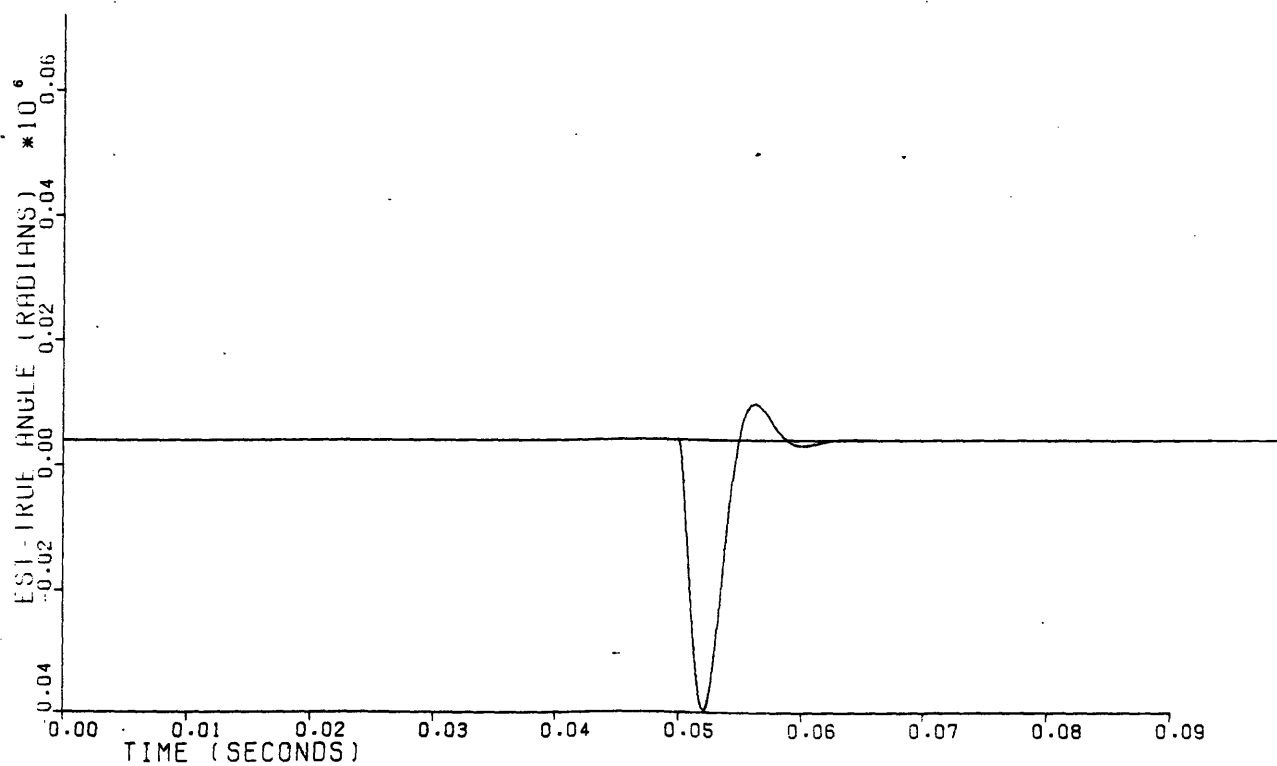


Figure 12-44. Run #10 Position Estimation Error vs. Time (15°/hr Step Input)

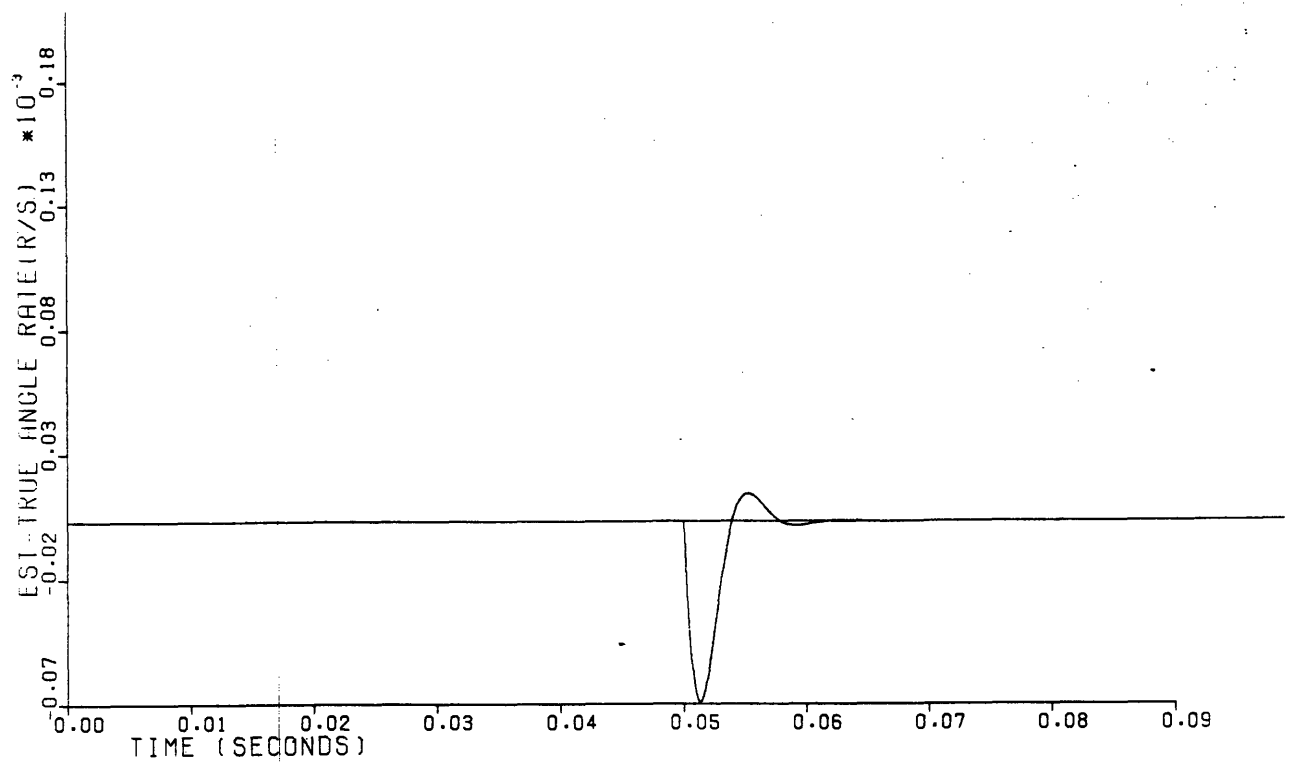


Figure 12-45. Run #10 Velocity Estimation Error vs. Time (15°/hr Step Input)

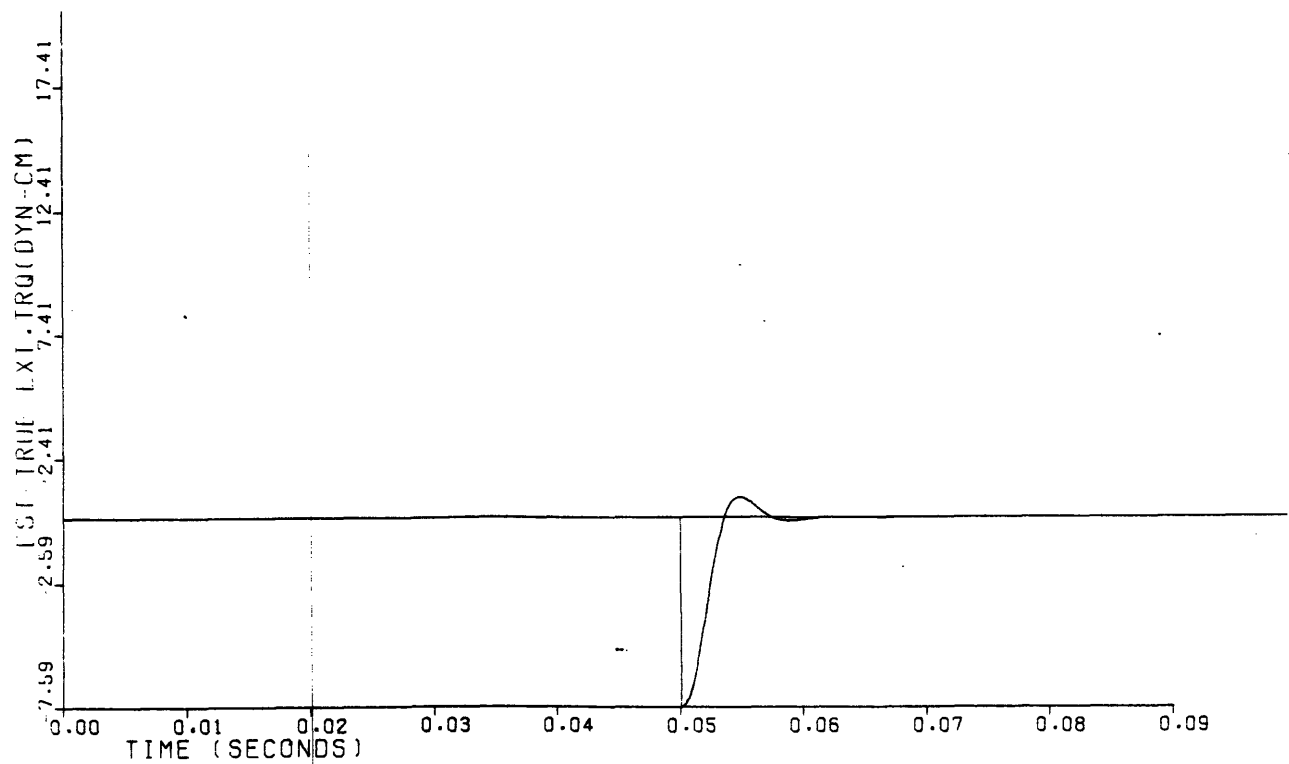


Figure 12-46. Run #10 External Torque Estimation Error vs. Time (15°/hr Step Input)

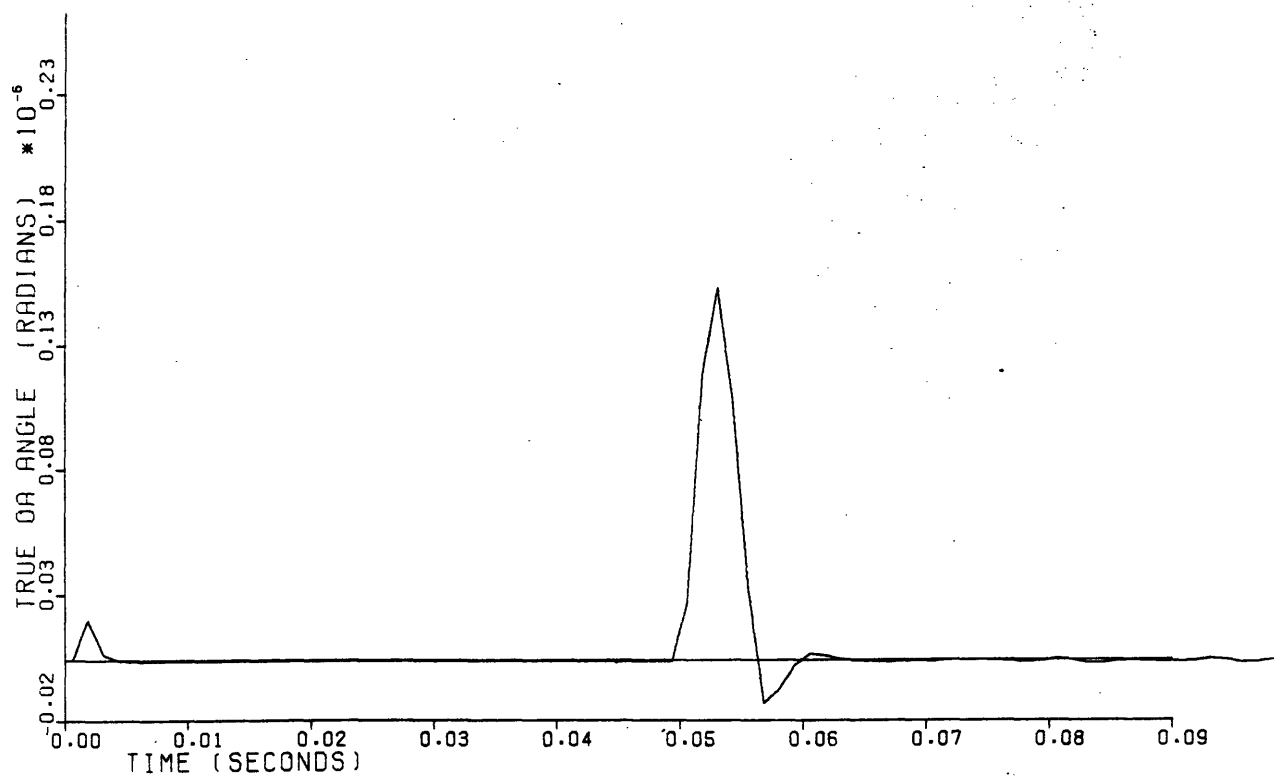


Figure 12-47. Run #10 24 Pt. Average Position vs. Time (15°/hr Step Input)

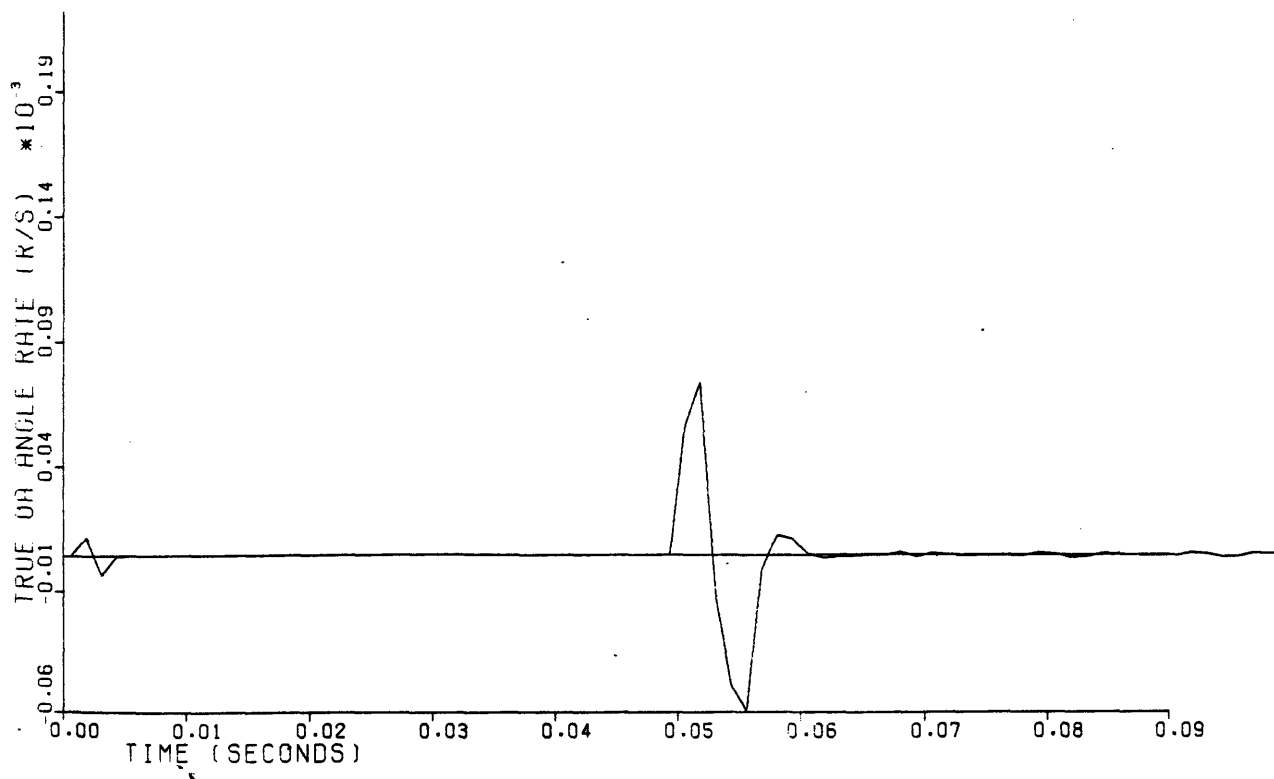


Figure 12-48. Run #10 24 Pt. Average Velocity vs. Time (15°/hr Step Input)

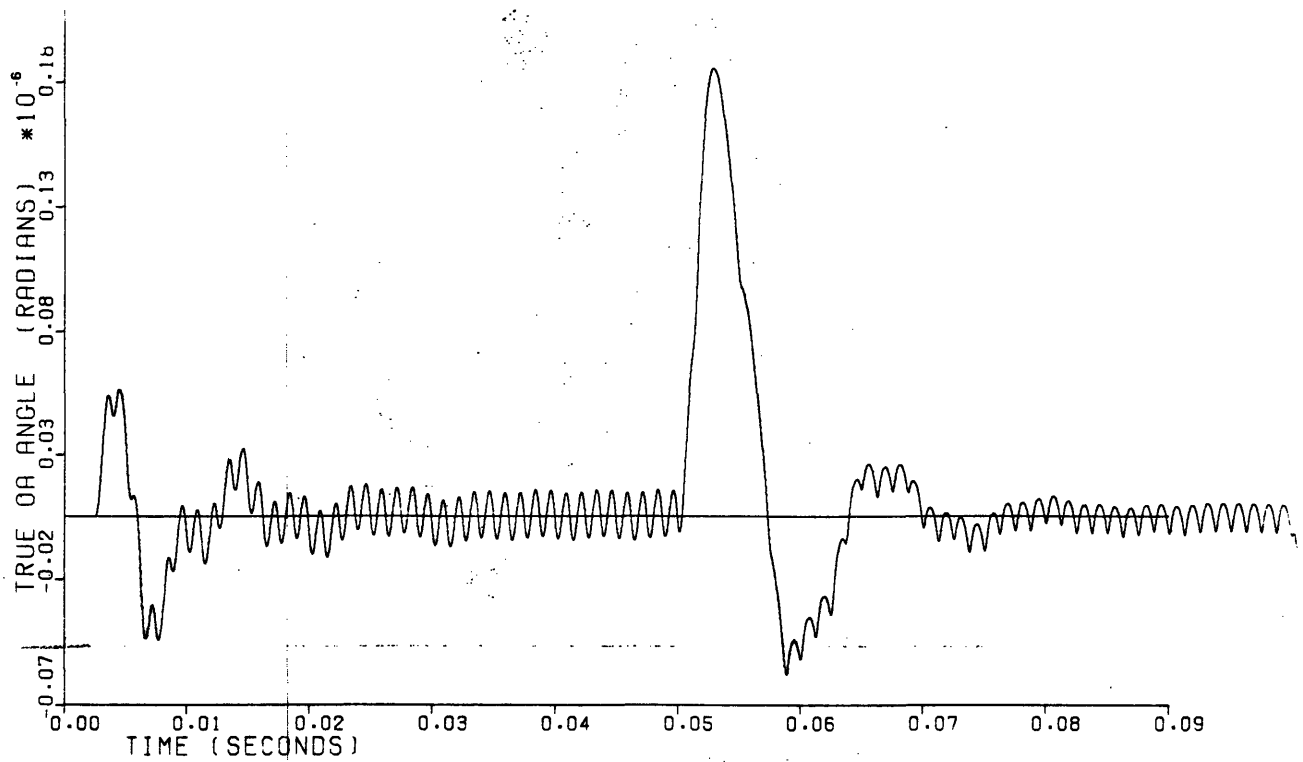


Figure 12-49. Run #11 Float Position vs. Time (15°/hr Step Input)

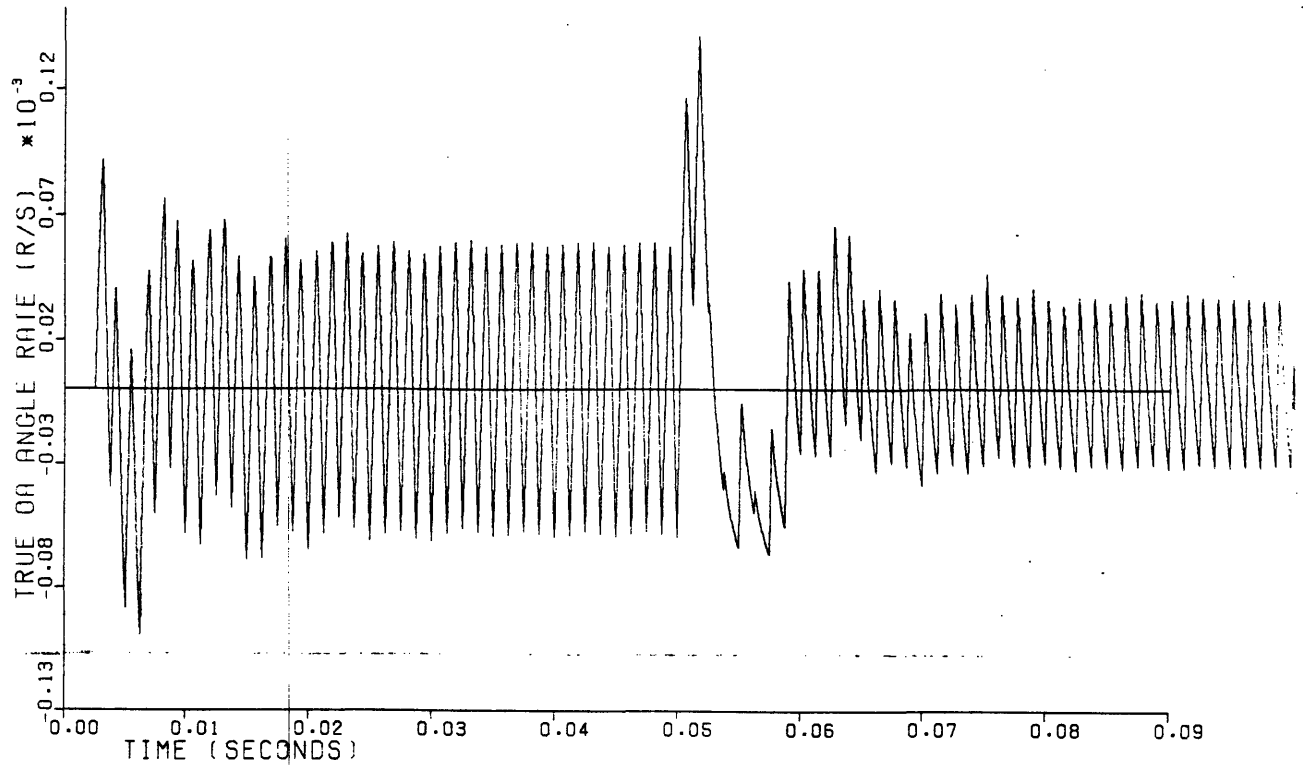


Figure 12-50. Run #11 Float Velocity vs. Time (15°/hr Step Input)

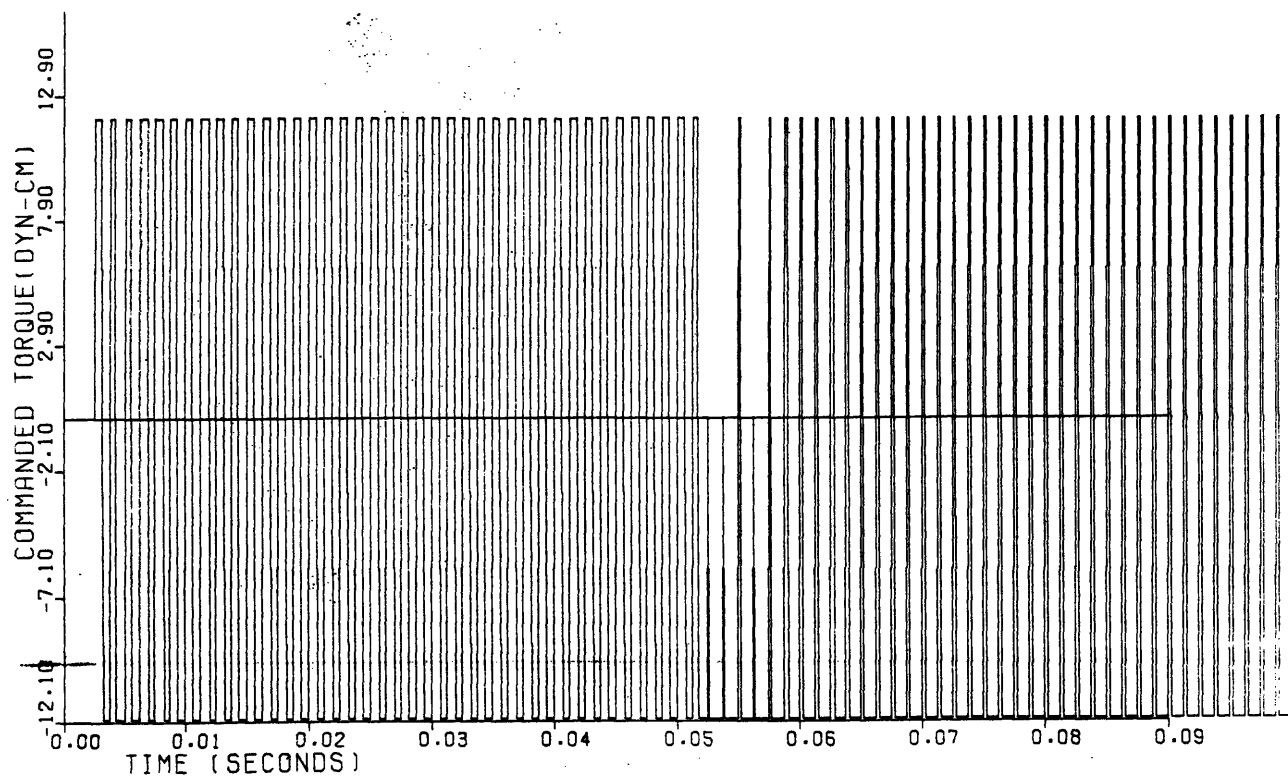


Figure 12-51. Run #11 Commanded Torque vs. Time (15°/hr Step Input)

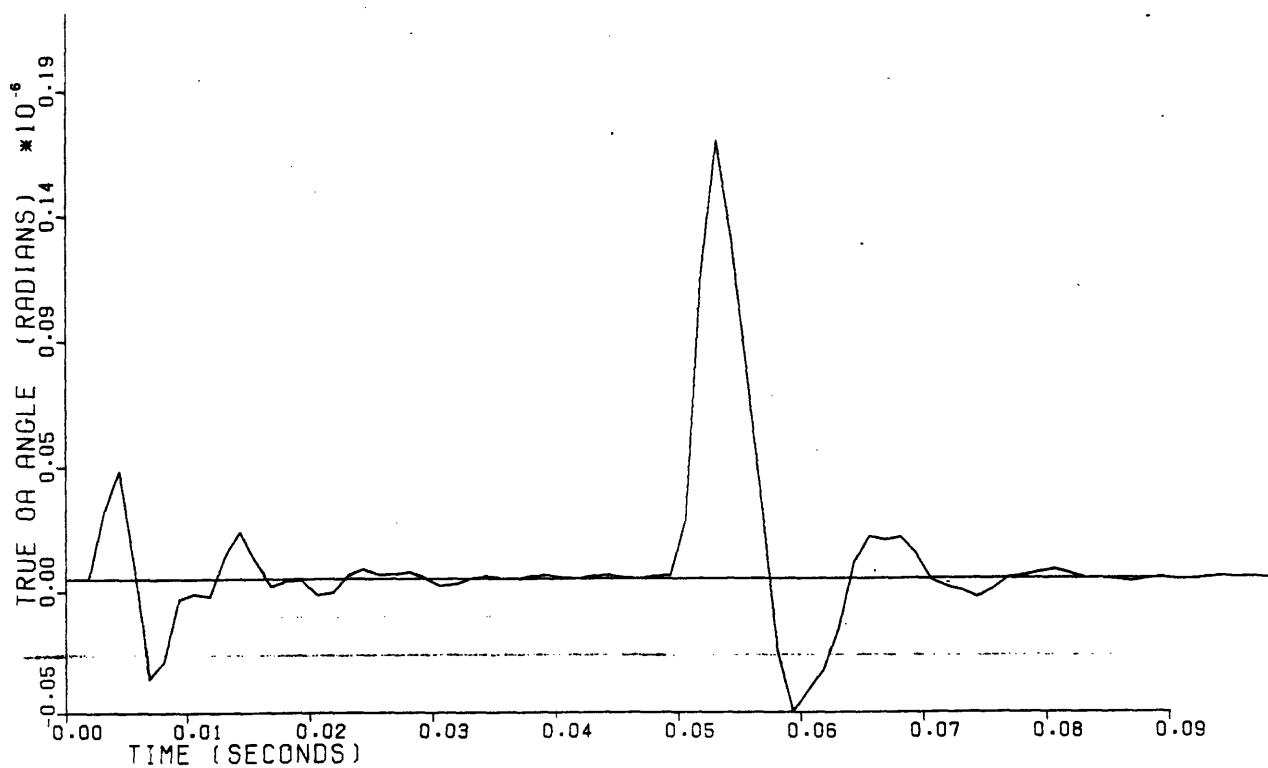


Figure 12-52. Run #11 24 Pt. Average Position vs. Time (15°/hr Step Input)

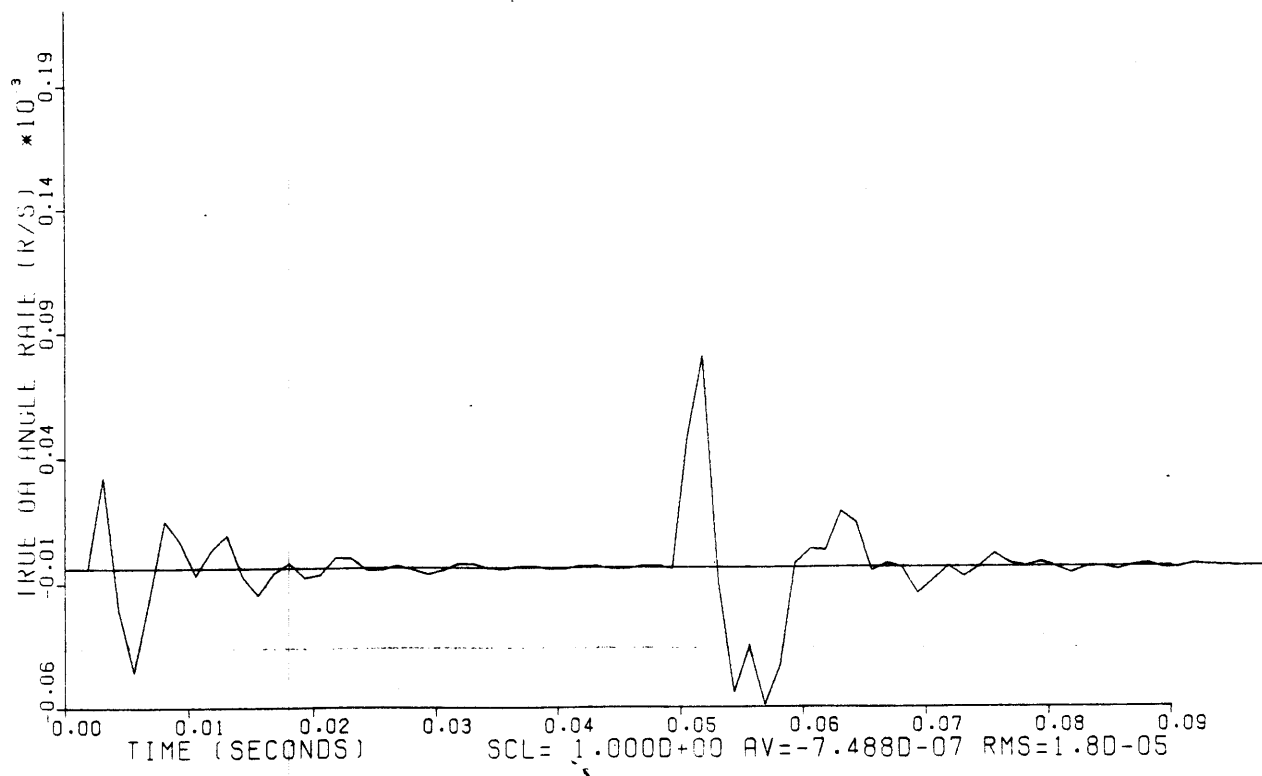


Figure 12-53. Run #11 24 Pt. Average Velocity vs. Time (15°/hr Step Input)

13. RAMP INPUT RESPONSE

13.1 Results Summary

Table 13-1 shows the summary of results for a ramp input response.

Table 13-1. Results Summary for Ramp Input of $282^\circ/\text{hr/s} = 142.9 \text{ dyne-cm/s}$
beginning @ 0.03 s

Run #	Description	Ave. float hang-off (nrad)	Steady State Estimation Errors		
			Position (nrad)	Velocity (rad/s)	External input (dyn-cm)
12	4 st. controller fast gains	-0.111	0.0	0.0	0.0
13	4 st. controller medium gains	-0.111	0.0	0.0	0.0
14	4 st. controller slow gains	-0.111	0.0	0.0	0.0
15	3 st. controller fast gains	1.151	0.0097	-1.1×10^{-7}	-0.0549
16	3 st. controller medium gains	3.726	-0.247	-1.089×10^{-6}	-0.2032
17	3 st. controller slow gains	7.546	-1.992	-2.722×10^{-6}	-0.2641
18	simplified controller	9.105	-	-	-

All controllers maintained a minimum mode of float oscillation and minimum quantization in the torquing pattern. As a result of the 800 Hz constant frequency of float oscillation, the amplitude of oscillations adjusts according to the level of external input.

The three state estimator propagates the third state τ (by integration) assuming it is constant over the sampling period. If the actual state τ is changing linearly, the estimator will track all states with a steady state estimation error. These errors contribute to the steady state average float hang-offs observed. The added integrator placed on the model of external input (addition of 4th state $\dot{\tau}$) raises the system type of the estimator thereby eliminating any steady state estimator errors for a ramp input. The average float hang-off is consequently reduced. The added state $\dot{\tau}$ reduced the average float hang-off of the best three state controller with fast gains, by about a factor of ten.

The average float hang-off of the four state controller was not affected by the different sets of estimator gains. This is because the steady state estimation errors are zero for all sets of gains. The three state controller did exhibit a correlation between the gains and the average float hang-off, since the estimation errors depended upon the gains chosen.

It was observed that the average float hang-offs of the three state controllers seemed to slowly ramp down towards null. Since the errors in position were generally less than the float hang-off, the controller was able to sense a net positive float offset and attempt to correct for it if the calculated switching instant was adjusted enough to kick the controller to the next quantized torque word.

The simplified controller performed comparably to the three state controller with slow gains. There was, however, no attempt to correct for the float hang-off.

13.2 Simulation Plots

Figures 13-1 through 13-53 present the plots for the responses to a ramp input.

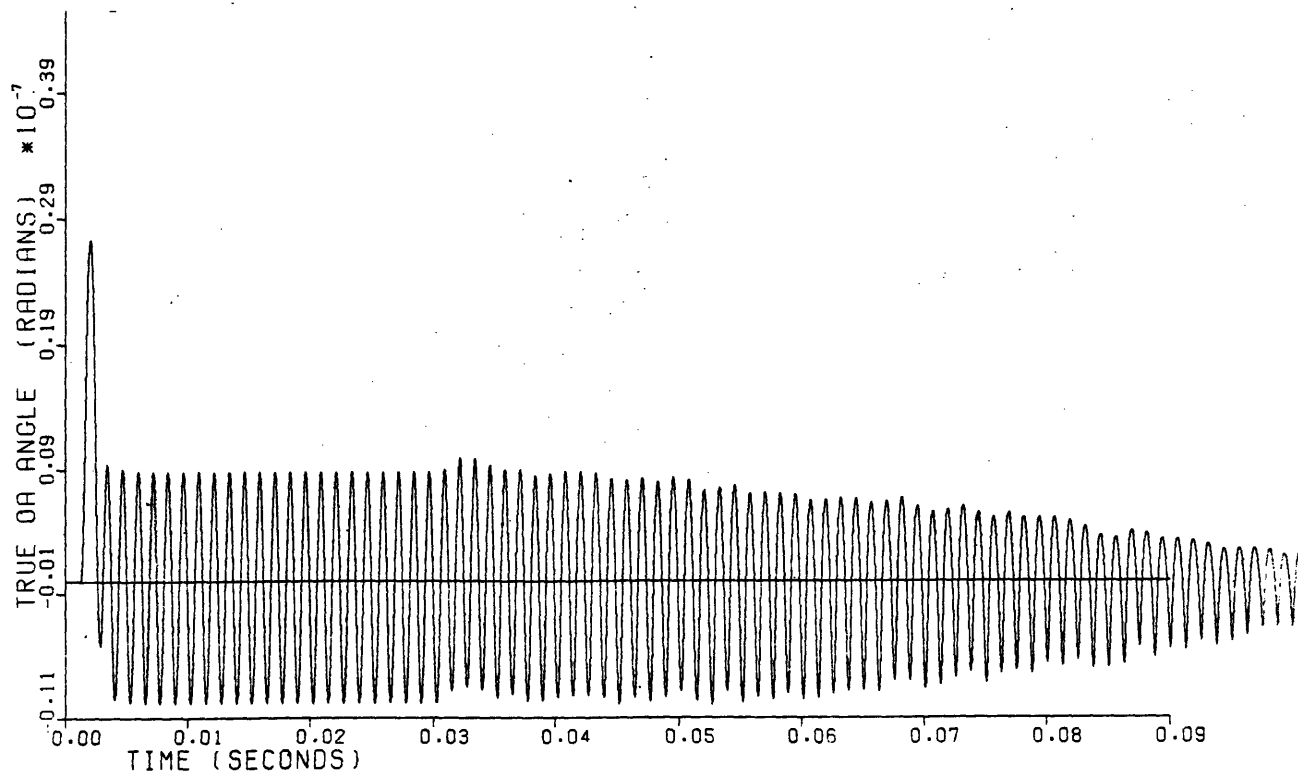


Figure 13-1. Run #12 Float Position vs. Time (282°/hr/s Ramp Input)

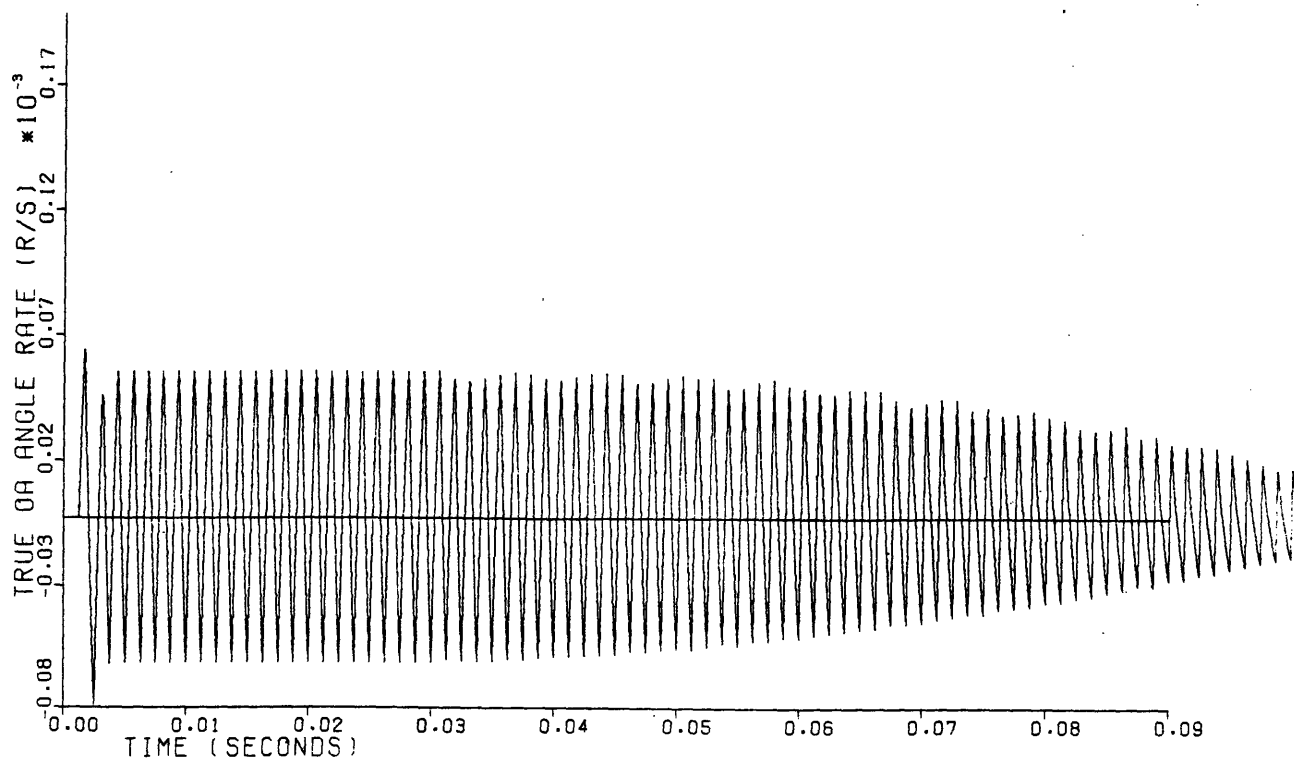


Figure 13-2. Run #12 Float Velocity vs. Time (282°/hr/s Ramp Input)

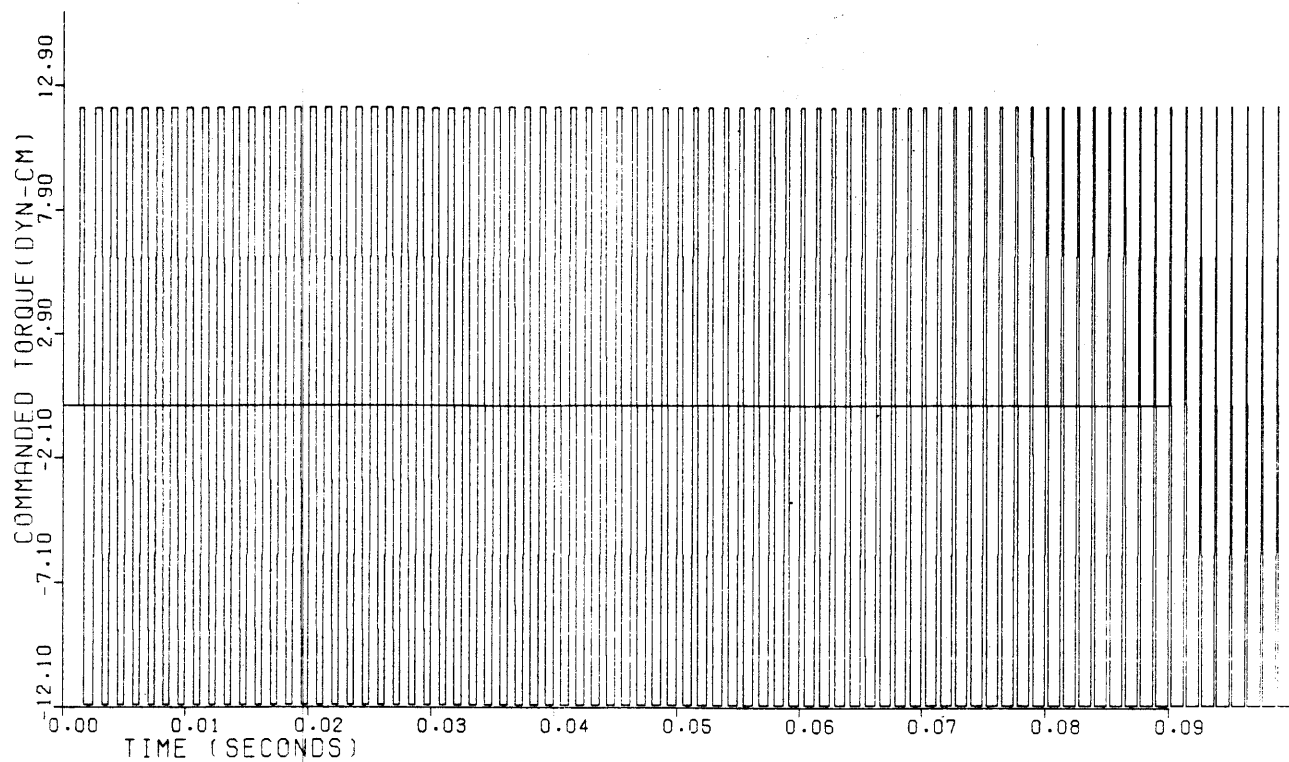


Figure 13-3. Run #12 Commanded Torque vs. Time (282°/hr/s Ramp Input)

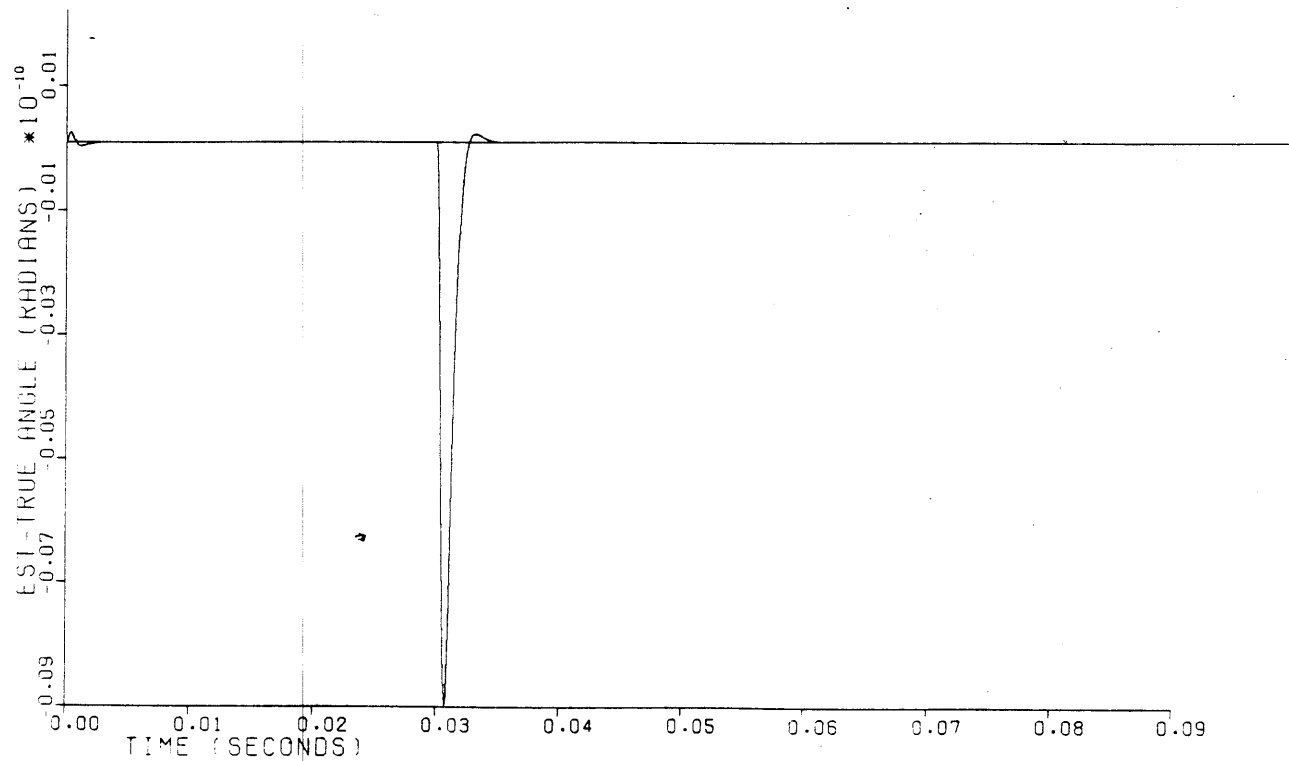


Figure 13-4. Run #12 Position Estimation Error vs. Time (282°/hr/s Ramp Input)

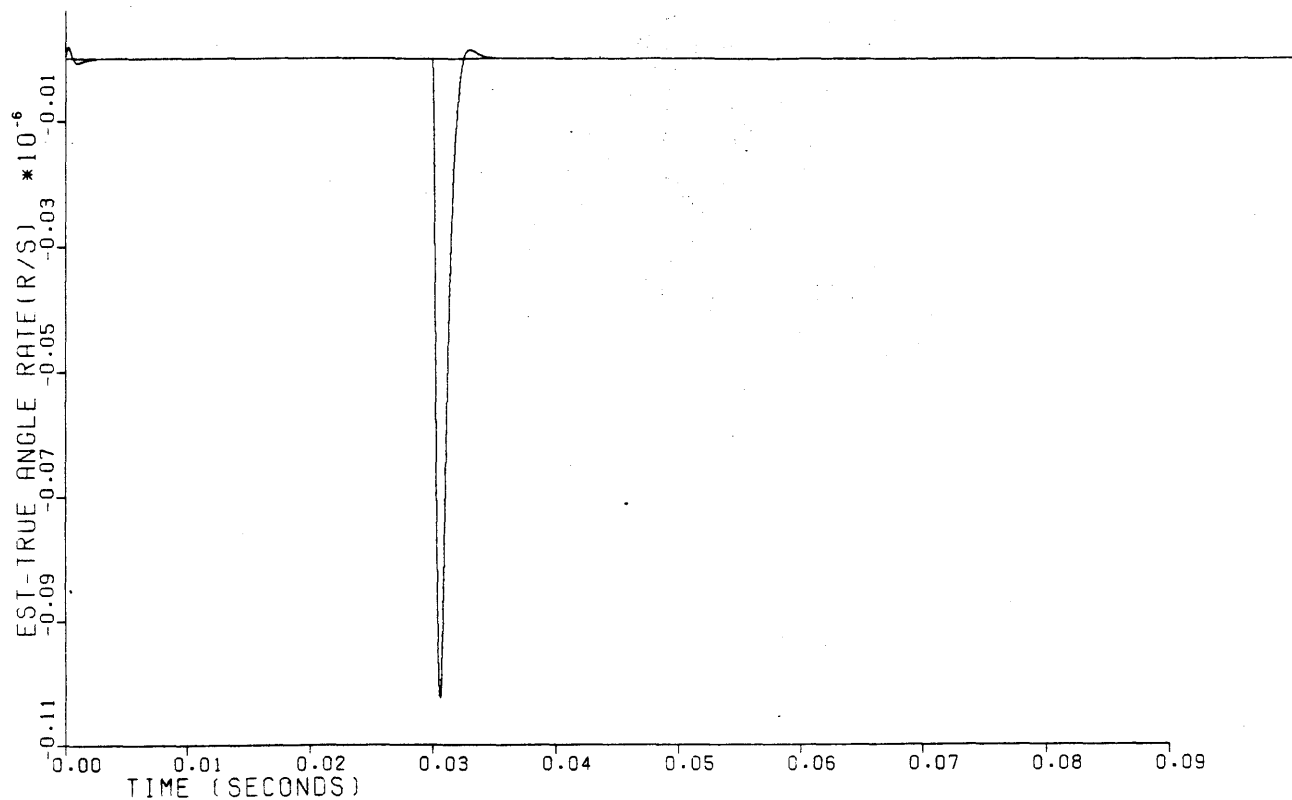


Figure 13-5. Run #12 Velocity Estimation Error vs. Time (282°/hr/s Ramp Input)

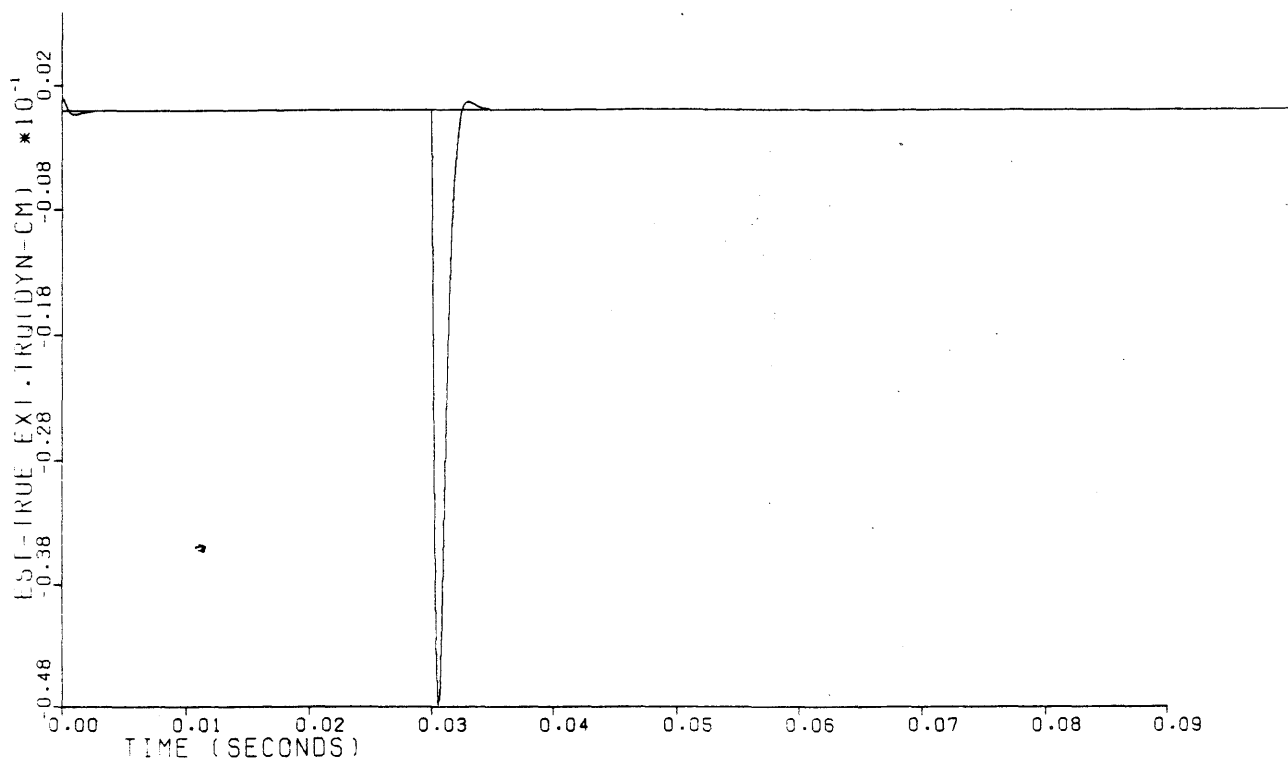


Figure 13-6. Run #12 External Torque Estimation Error vs. Time (282°/hr/s Ramp Input)

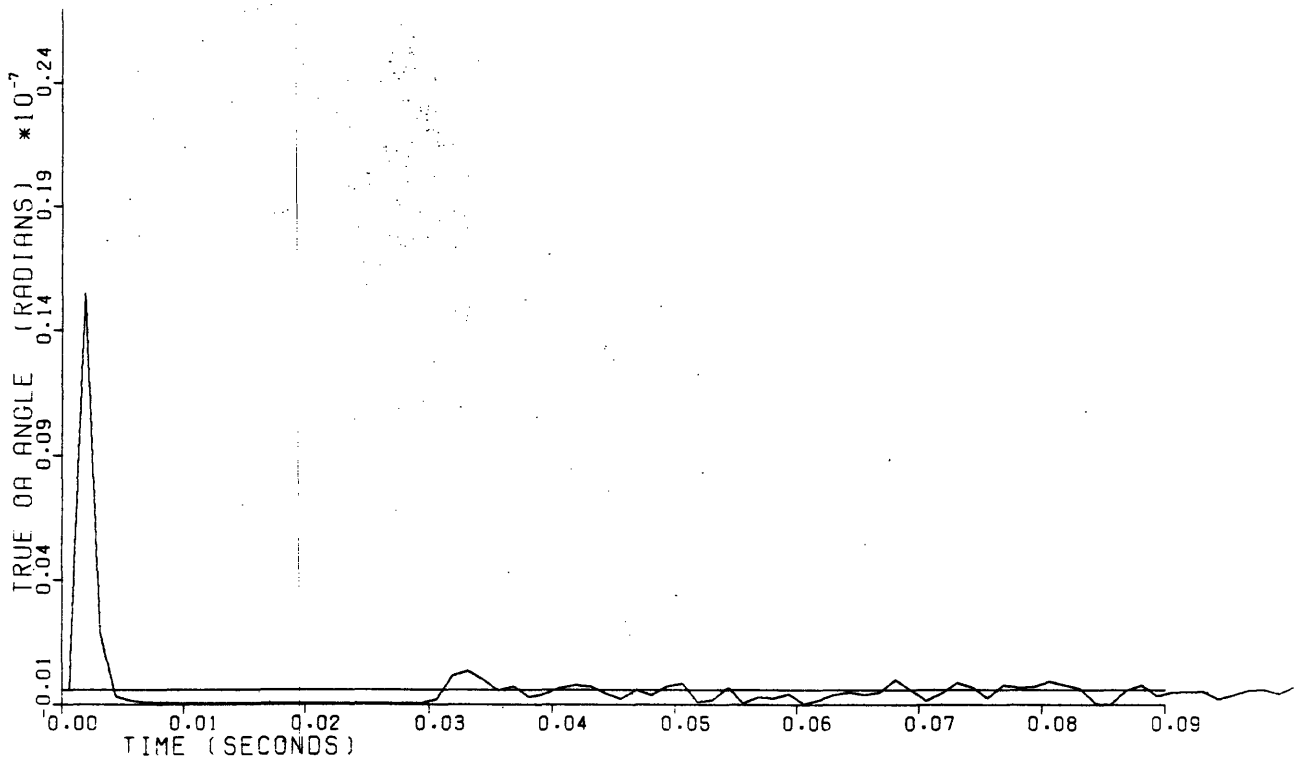


Figure 13-7. Run #12 24 Pt. Average Position vs. Time (282°/hr/s Ramp Input)

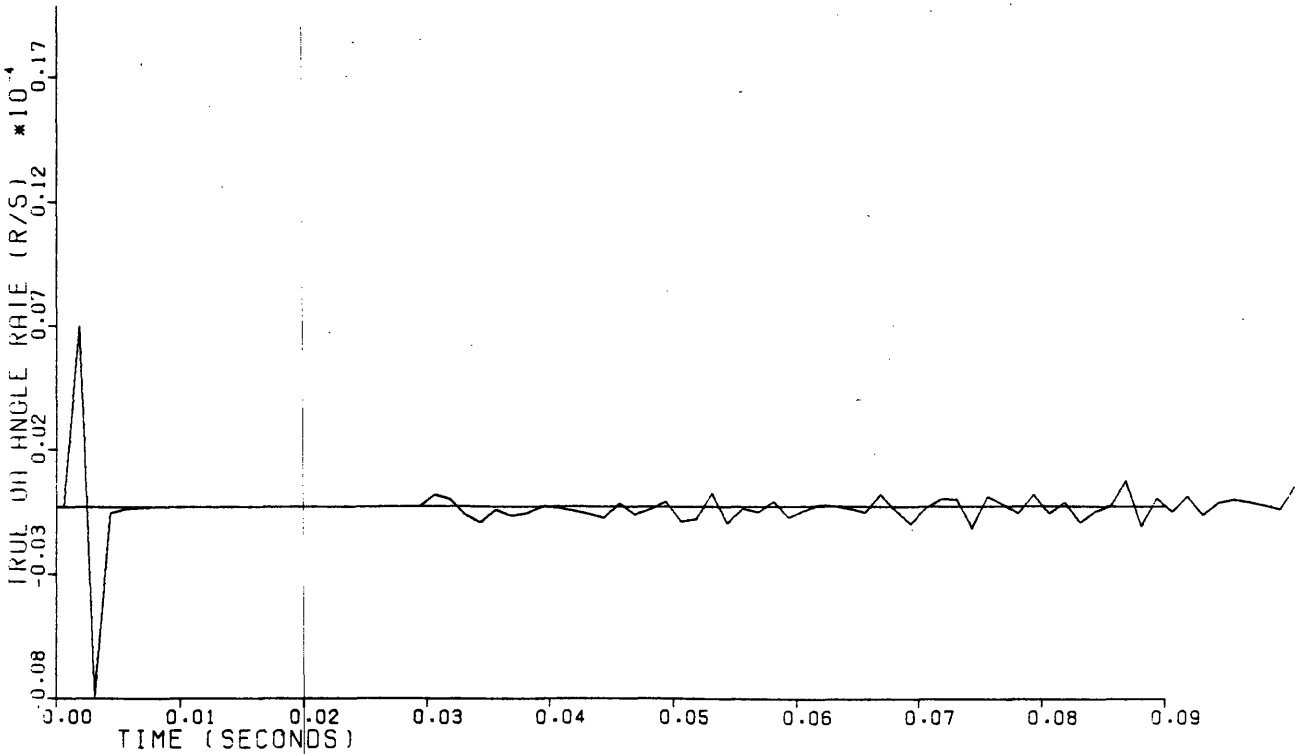


Figure 13-8. Run #12 24 Pt. Average Velocity vs. Time (282°/hr/s Ramp Input)

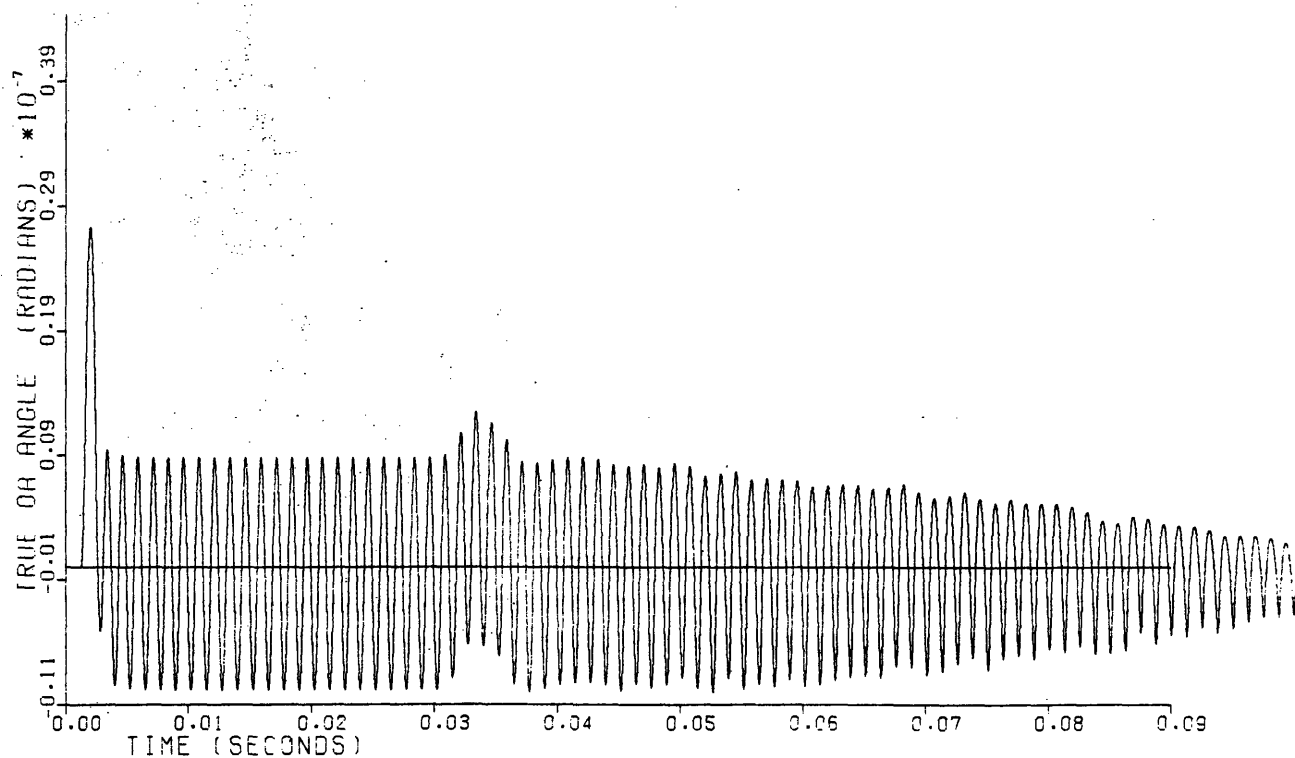


Figure 13-9. Run #13 Float Position vs. Time (282°/hr/s Ramp Input)

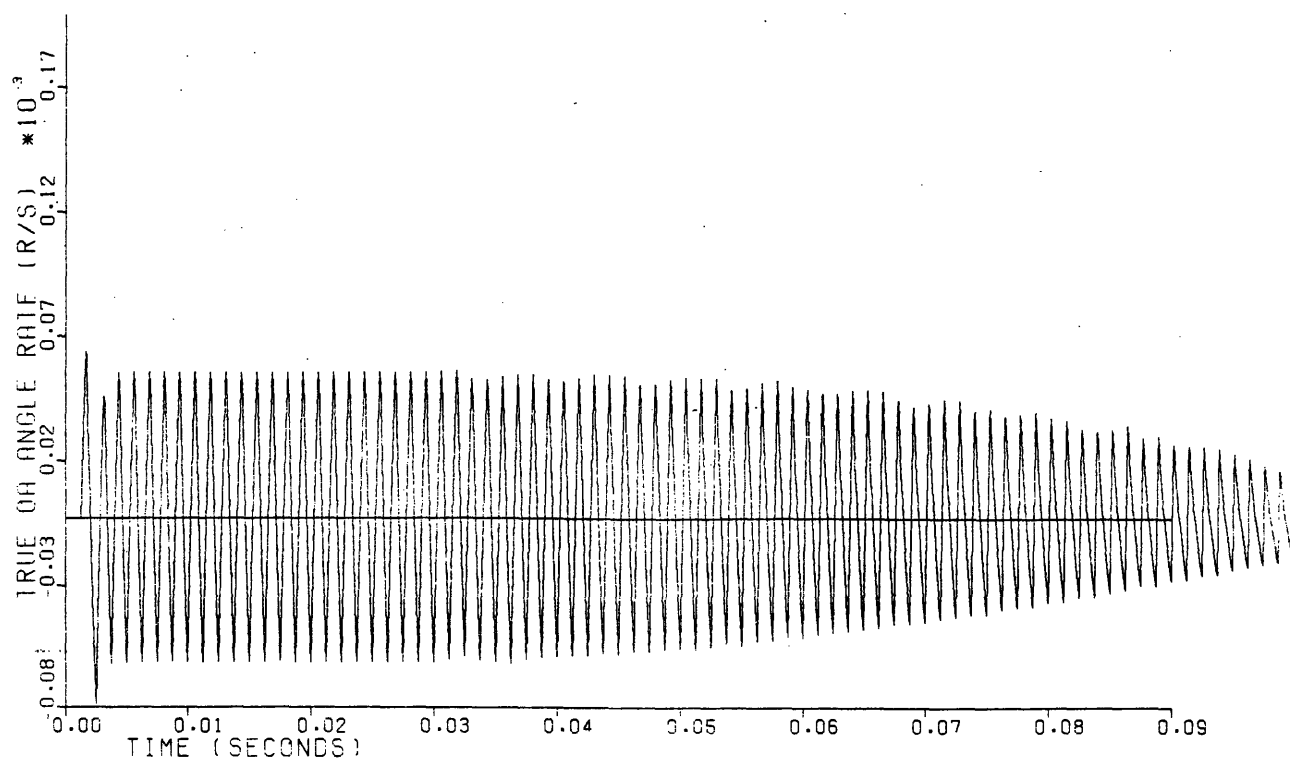


Figure 13-10. Run #13 Float Velocity vs Time (282°/hr/s Ramp Input)

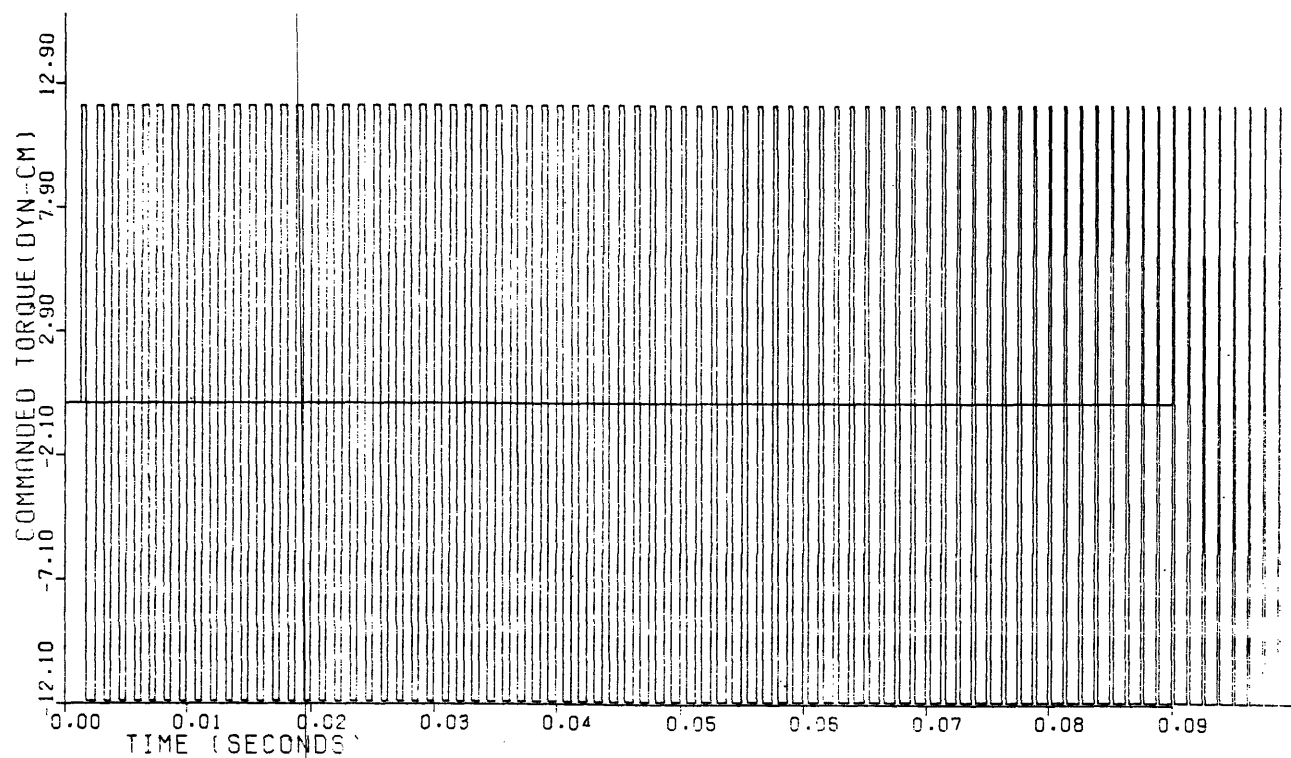


Figure 13-11. Commanded Torque vs. Time (282°/hr/s Ramp Input)

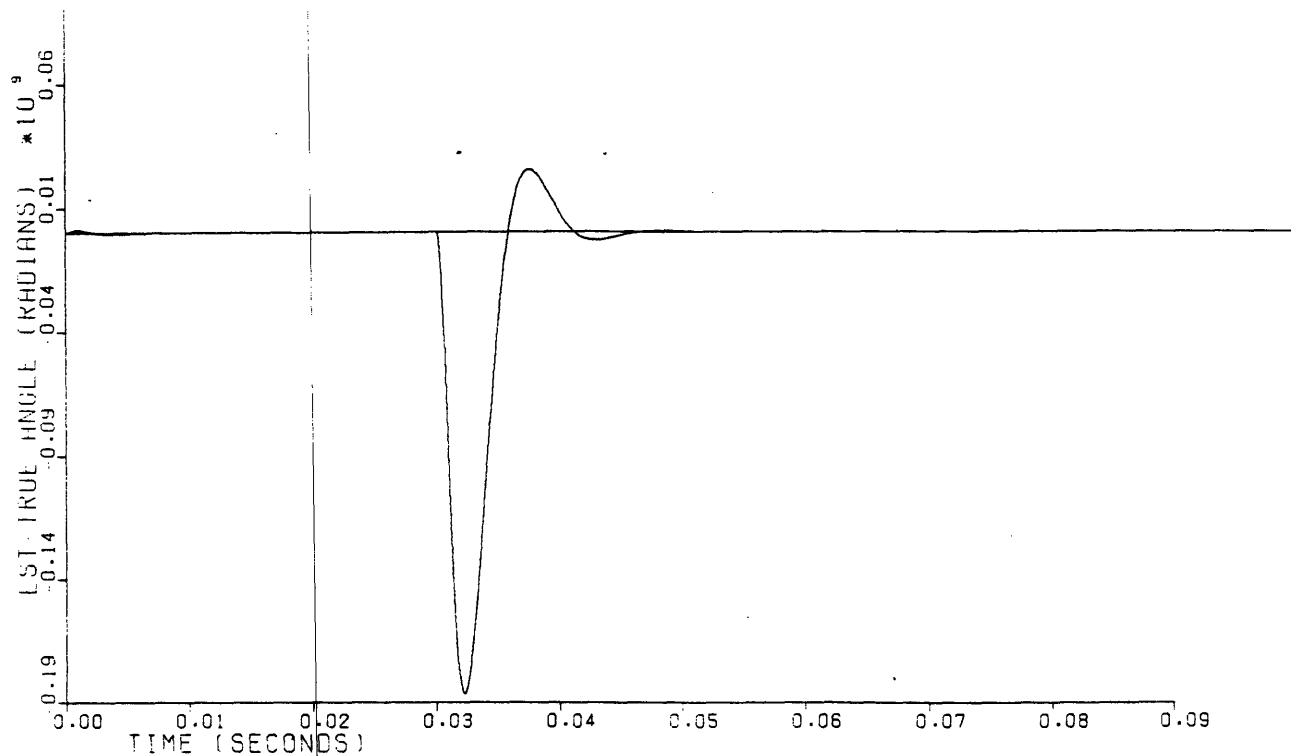


Figure 13-12. Run #13 Position Estimation Error vs. Time (282°/hr/s Ramp Input)

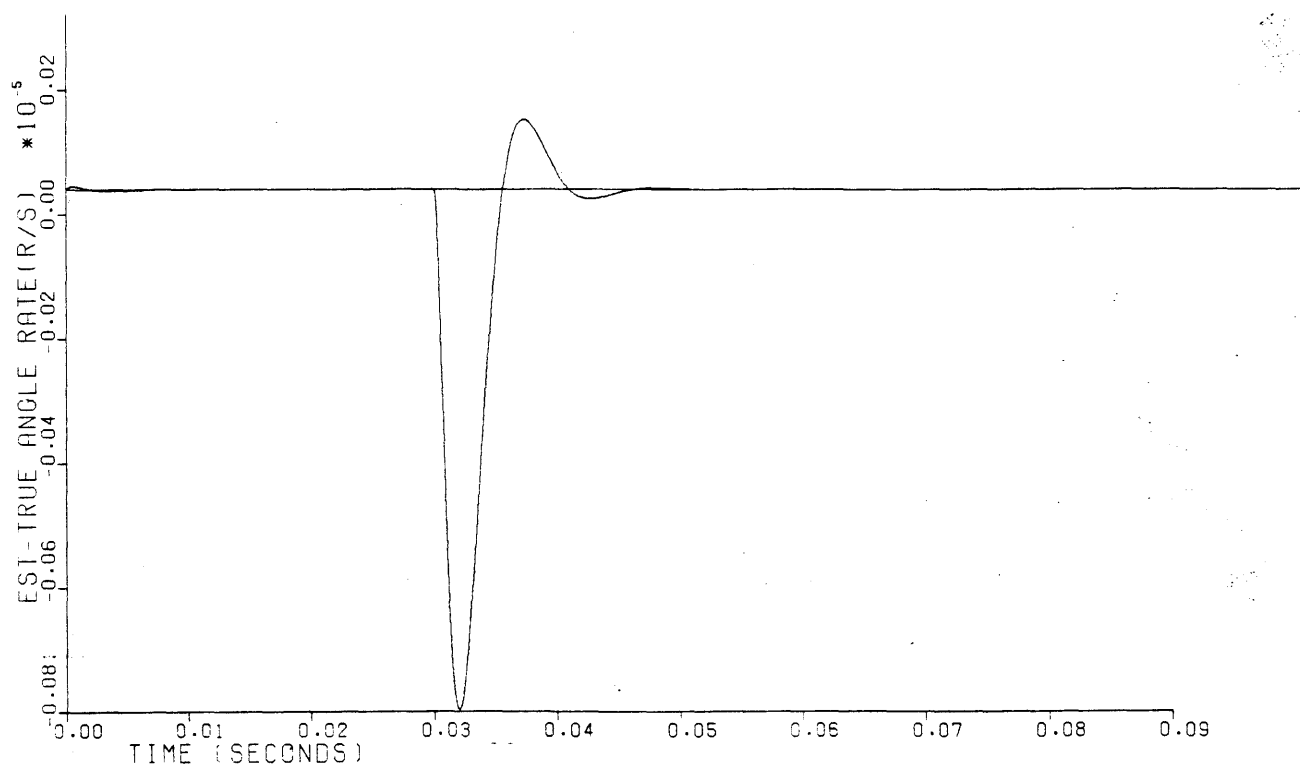


Figure 13-13. Run #13 Velocity Estimation Error vs. Time (282°/hr/s Ramp Input)

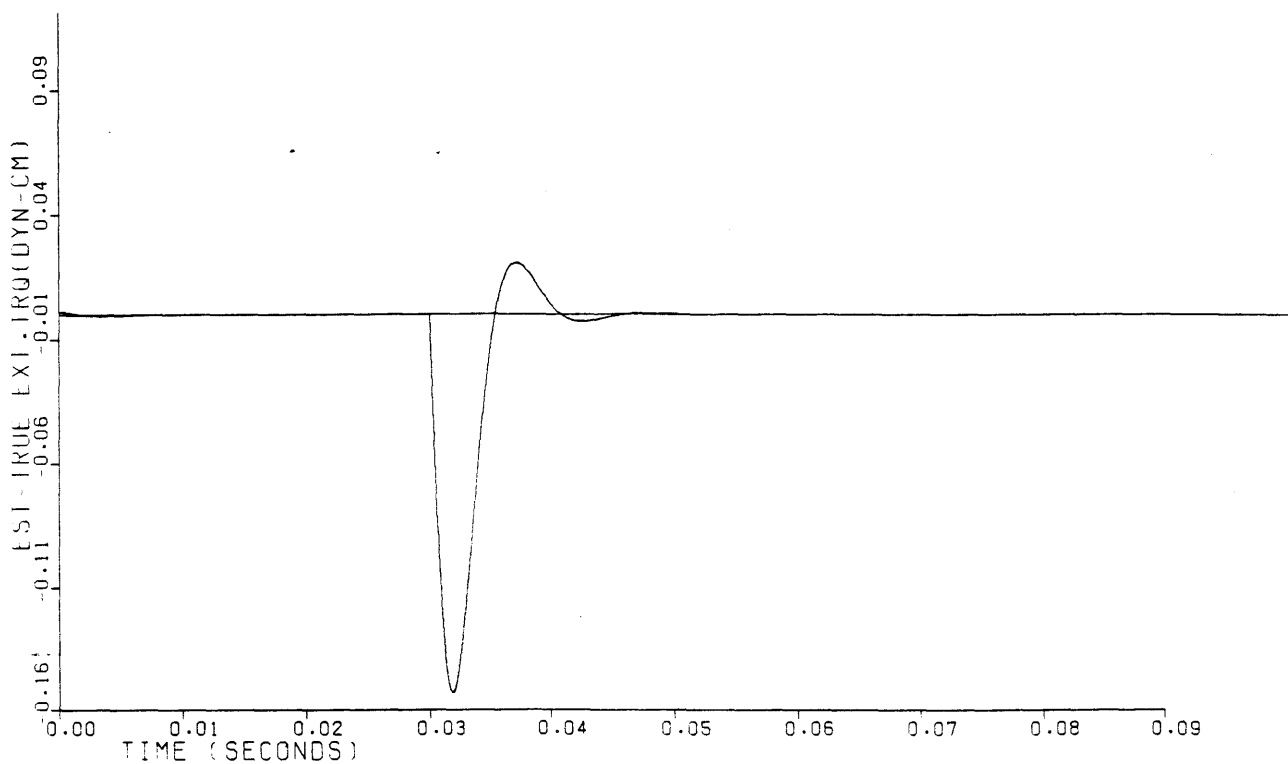


Figure 13-14. Run #13 External Torque Estimation Error vs. Time (282°/hr/s Ramp Input)

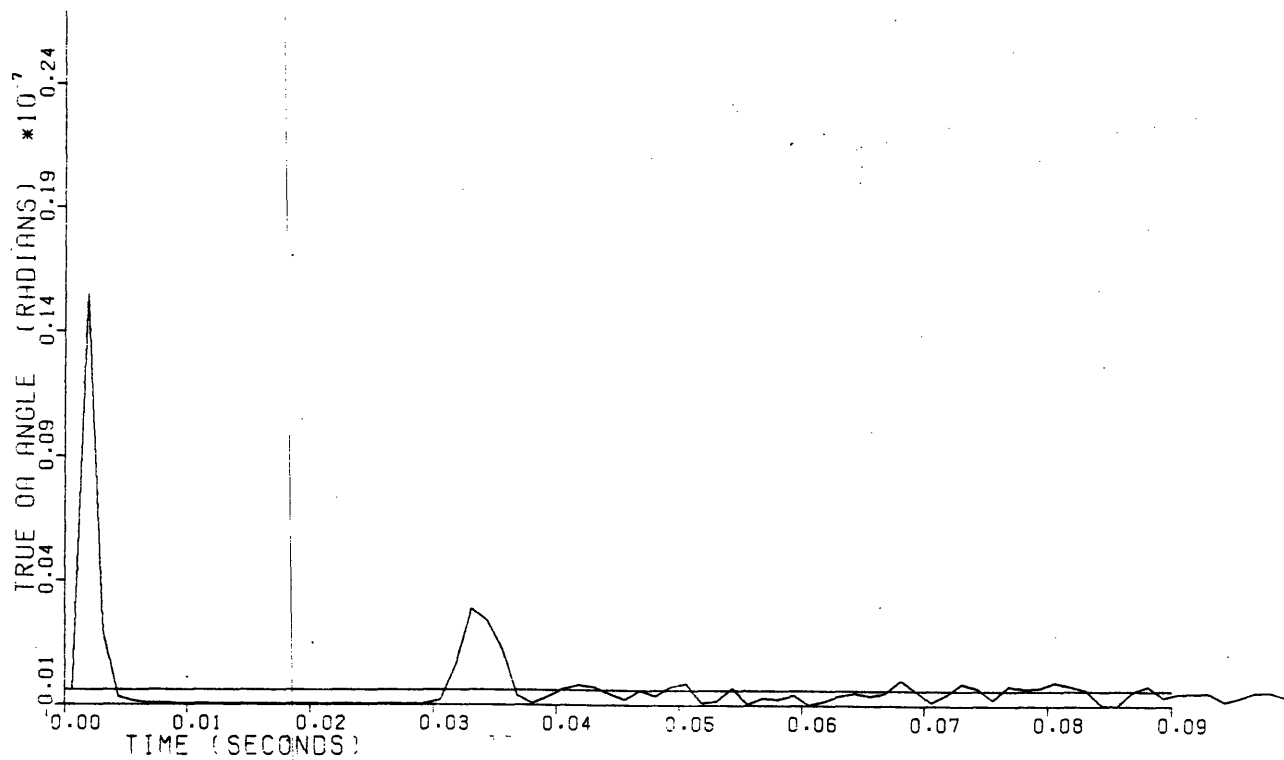


Figure 13-15. Run #13 24 Pt. Average Position vs. Time (282°/hr/s Ramp Input)

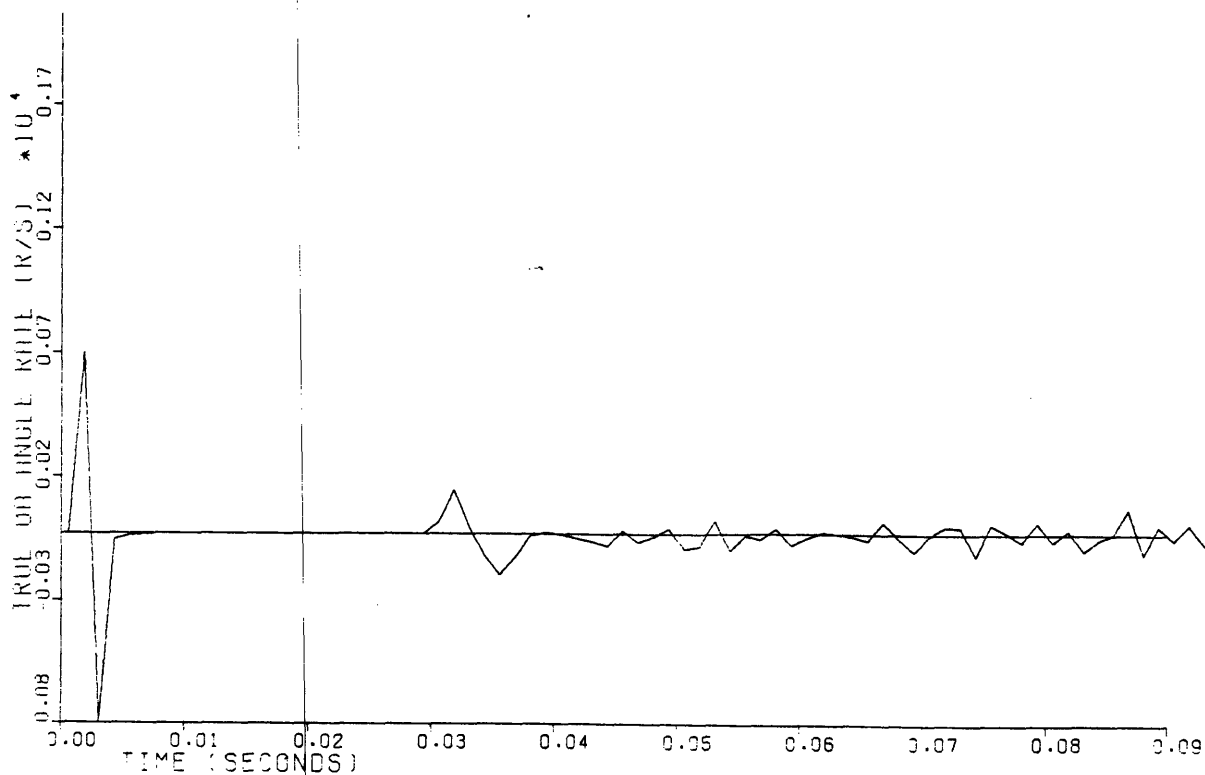


Figure 13-16. Run #13 24 Pt. Average Velocity vs Time (282°/hr/s Ramp Input)

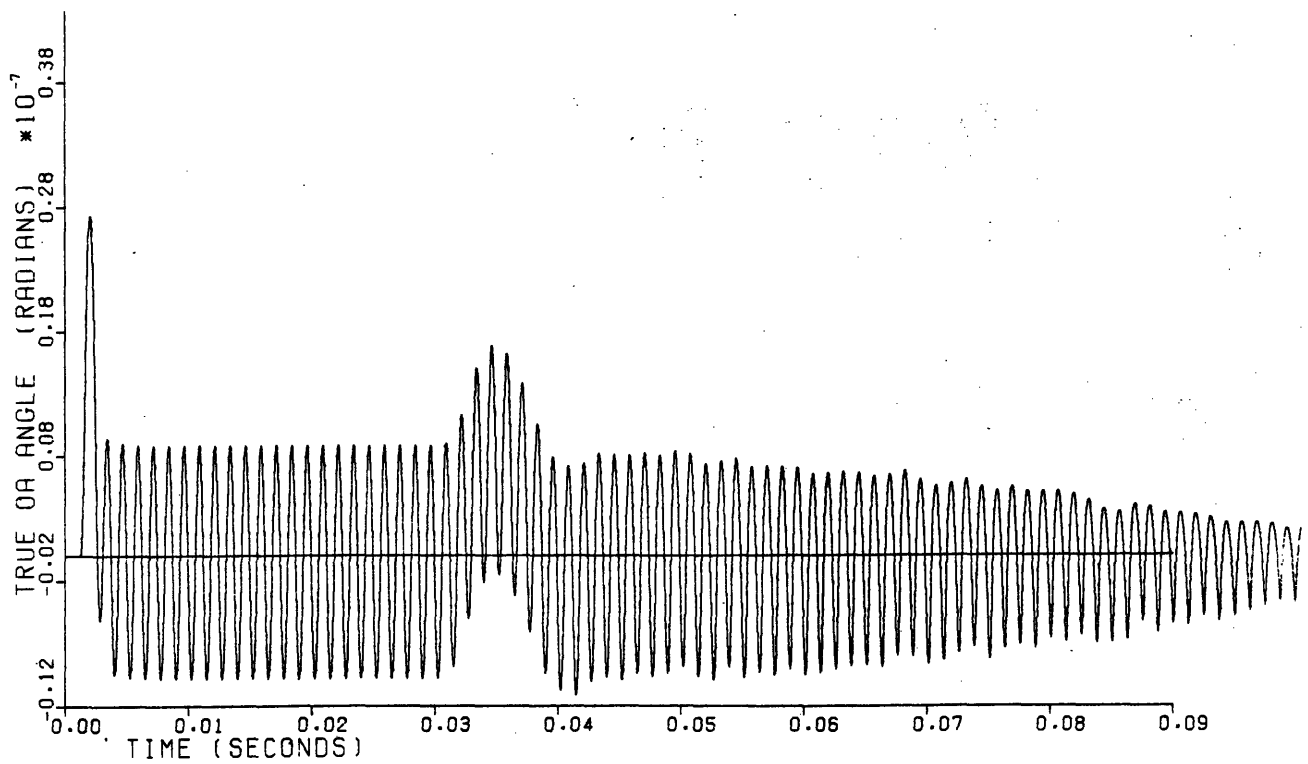


Figure 13-17. Run #14 Float Position vs. Time (282°/hr/s Ramp Input)

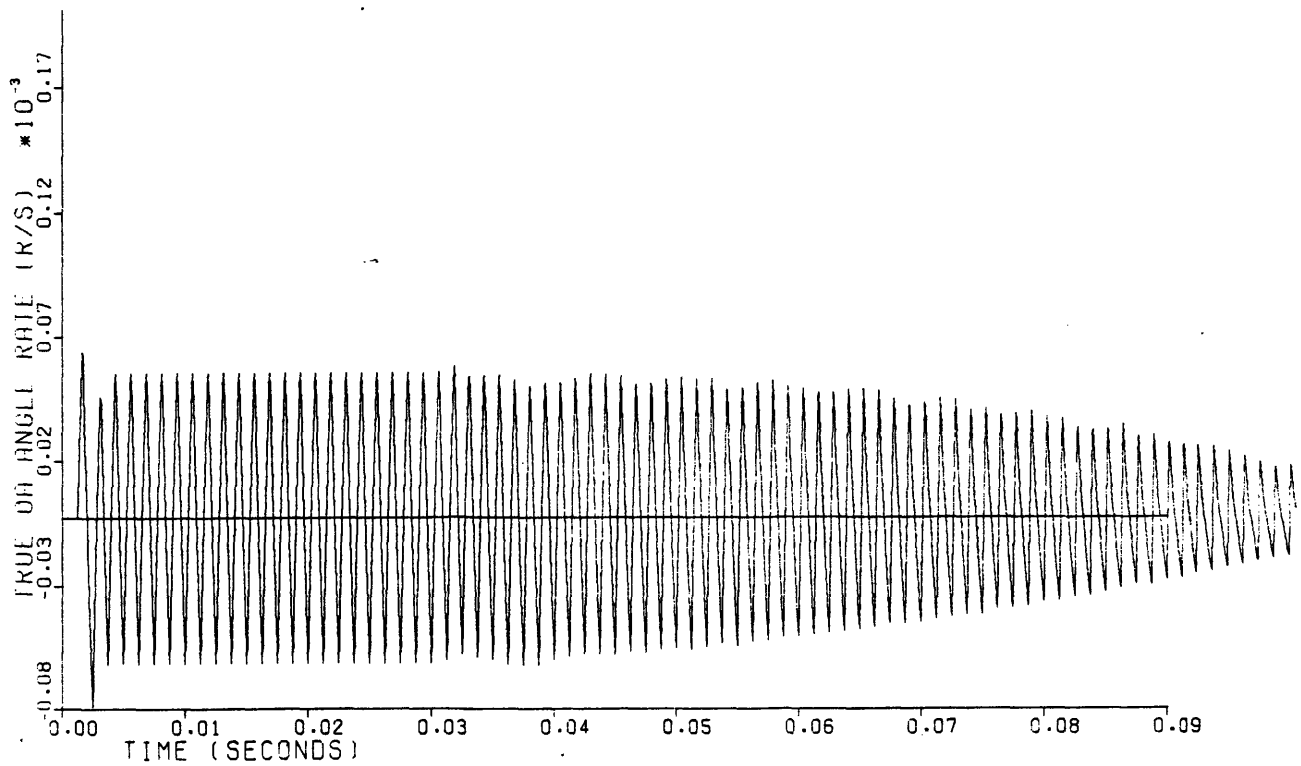


Figure 13-18. Run #14 Float Velocity vs. Time (282°/hr/s Ramp Input)

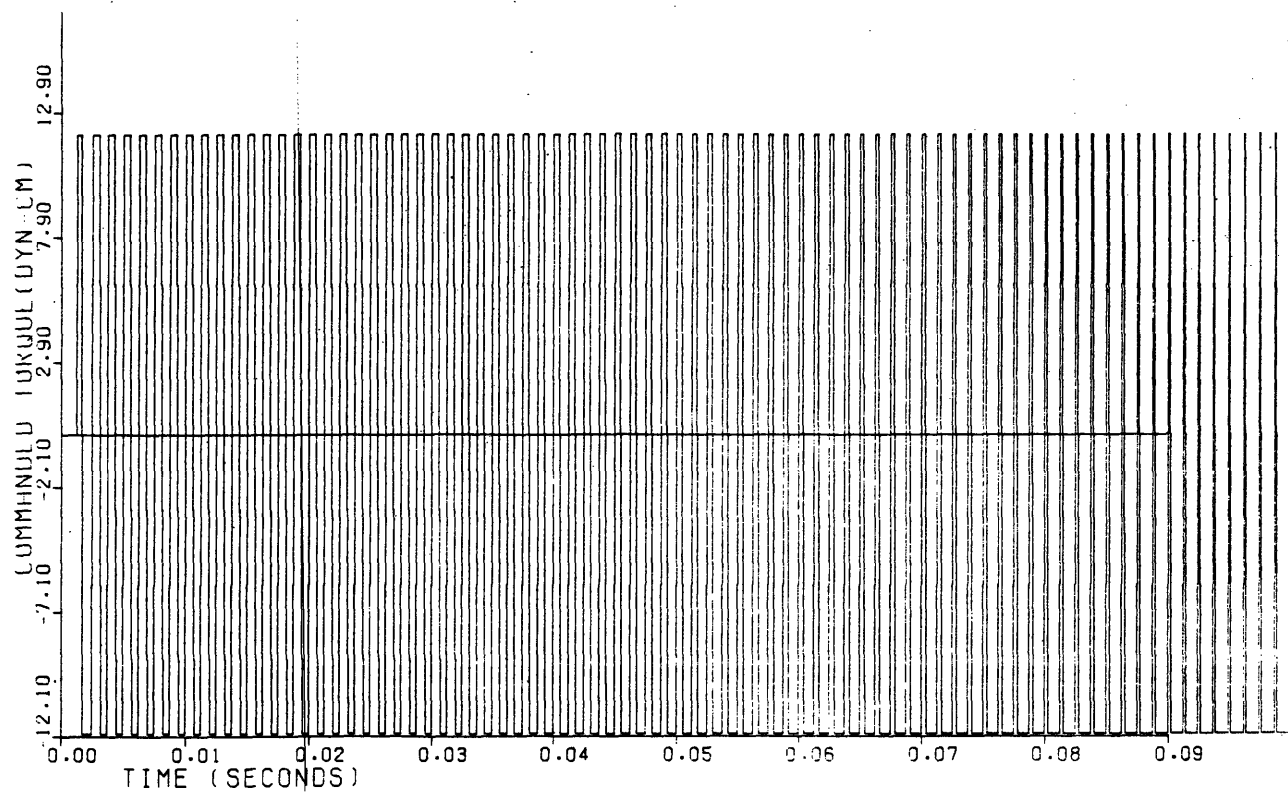


Figure 13-19: Run #14 Commanded Torque vs. Time (282°/hr/s Ramp Input)

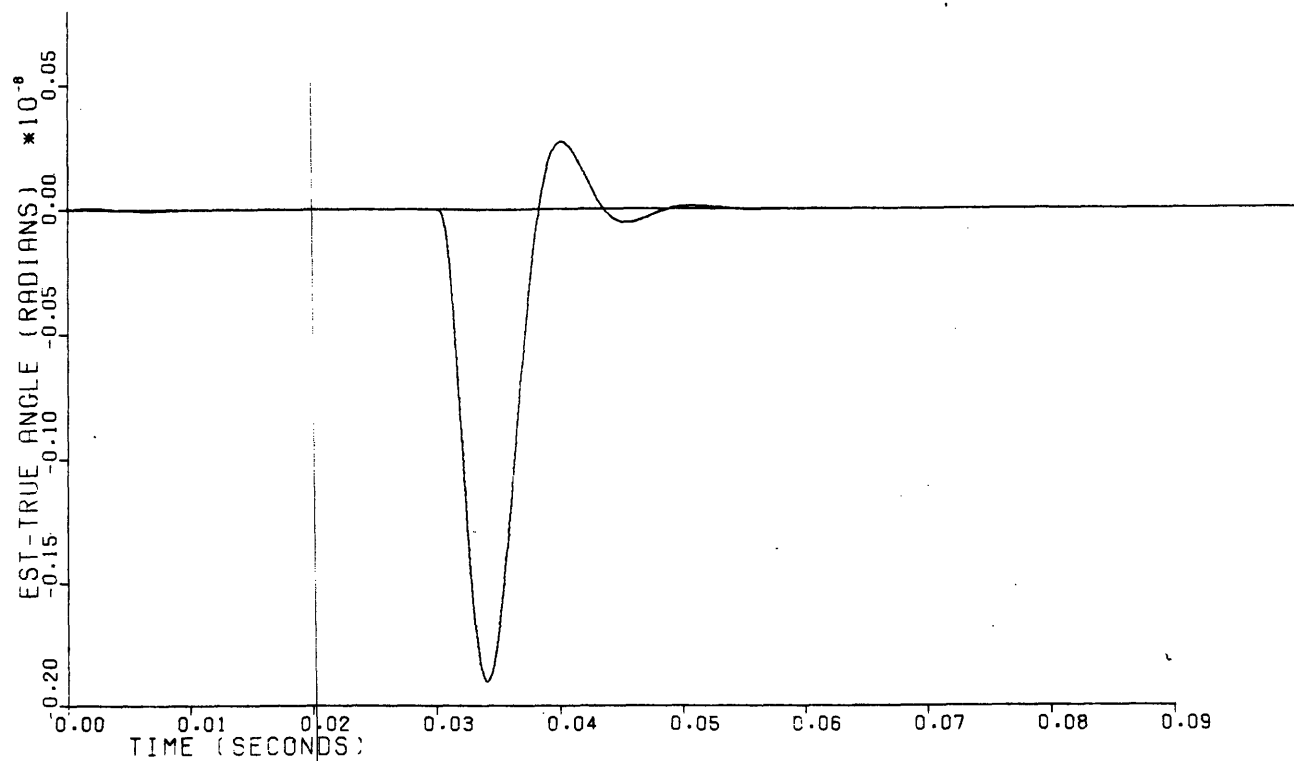


Figure 13-20: Run #14 Position Estimation Error vs. Time (282°/hr/s Ramp Input)

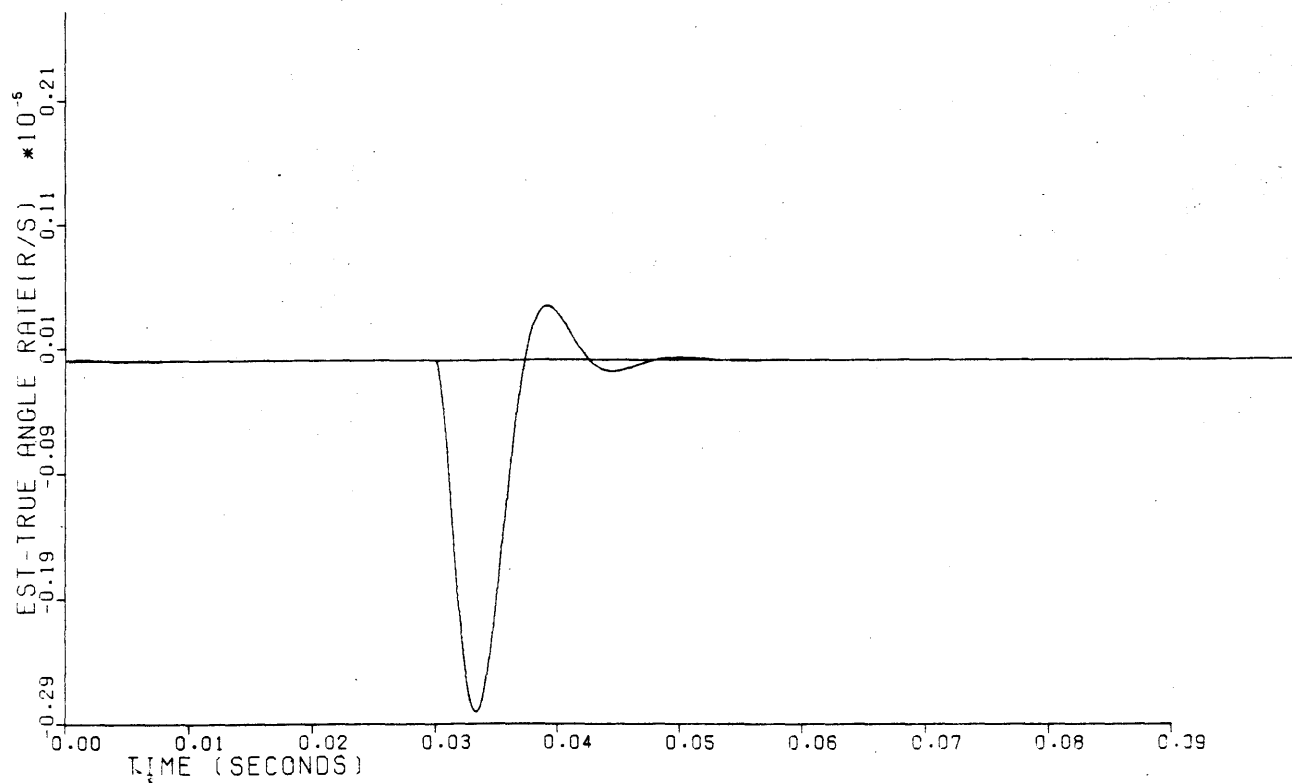


Figure 13-21. Run #14 Velocity Estimation Error vs. Time (282°/hr/s Ramp Input)

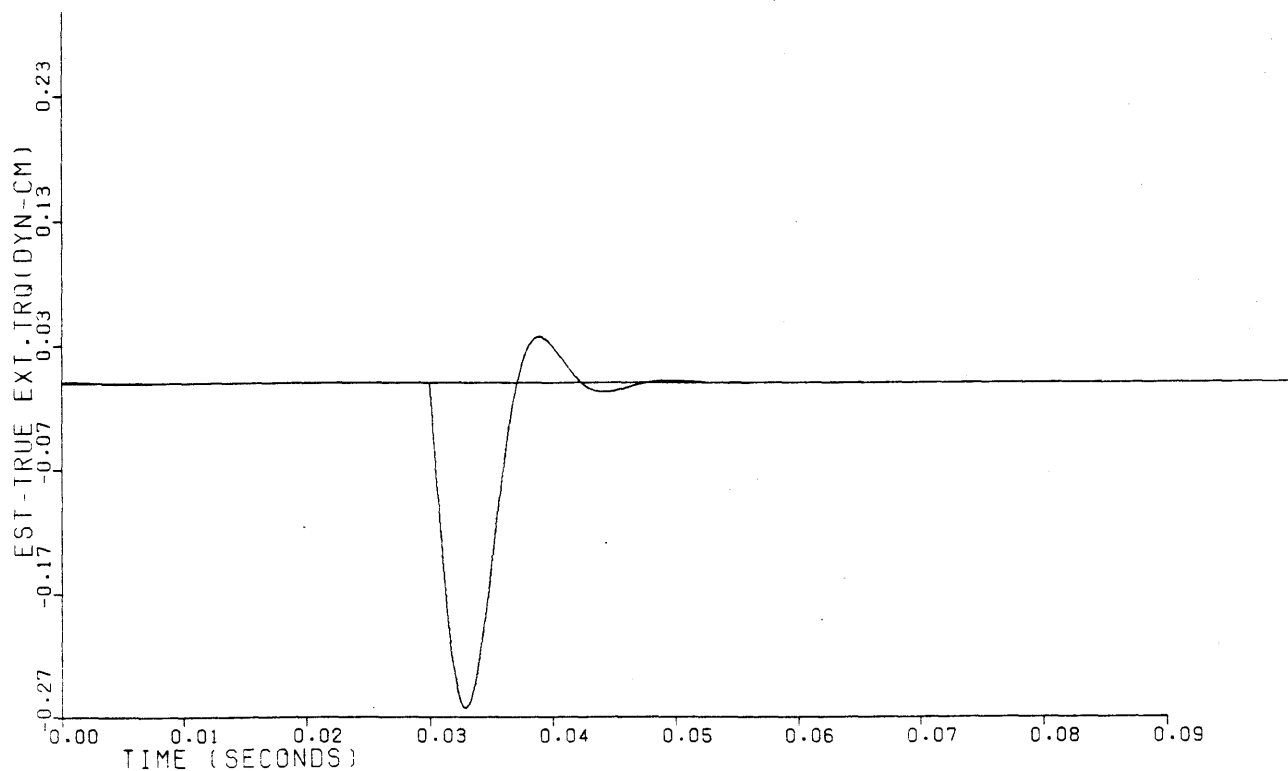


Figure 13-22. Run #14 External Torque Estimation Error vs. Time (282°/hr/s Ramp Input)

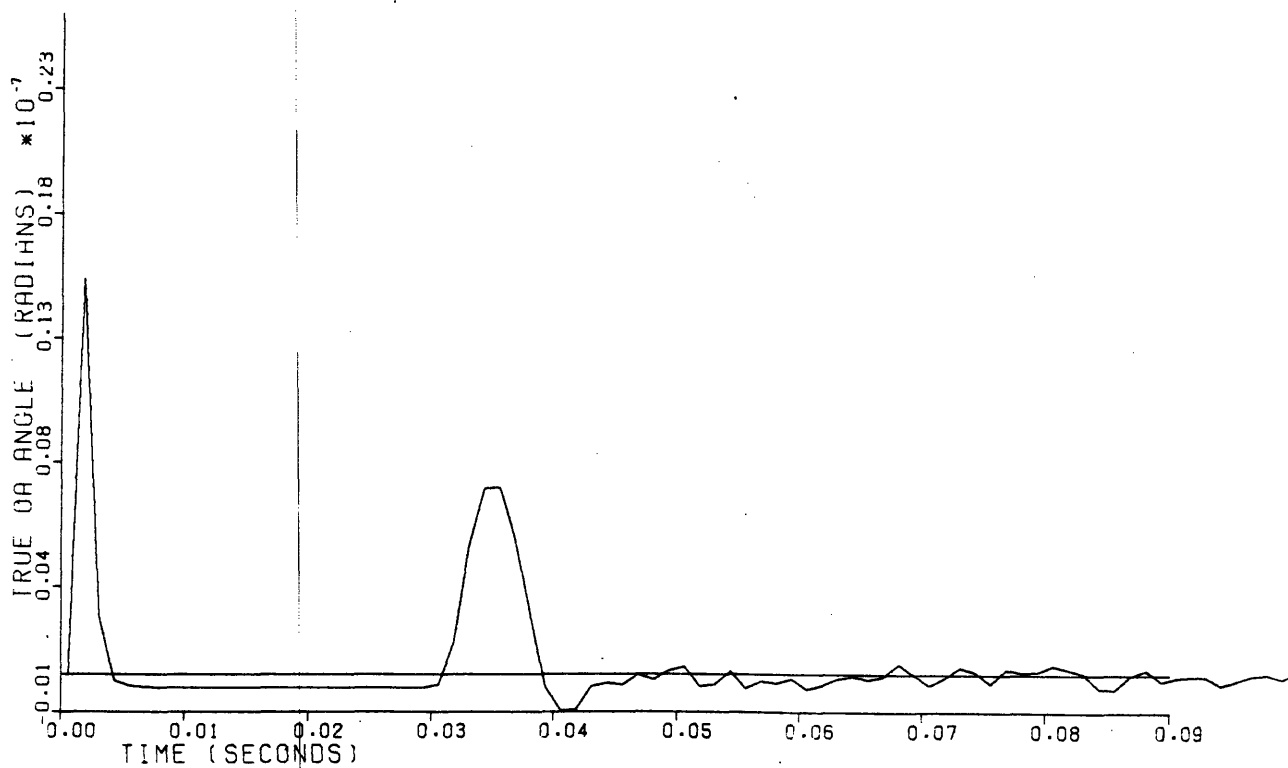


Figure 13-23. Run #14 24 Pt. Average Position vs. Time (282°/hr/s Ramp Input)

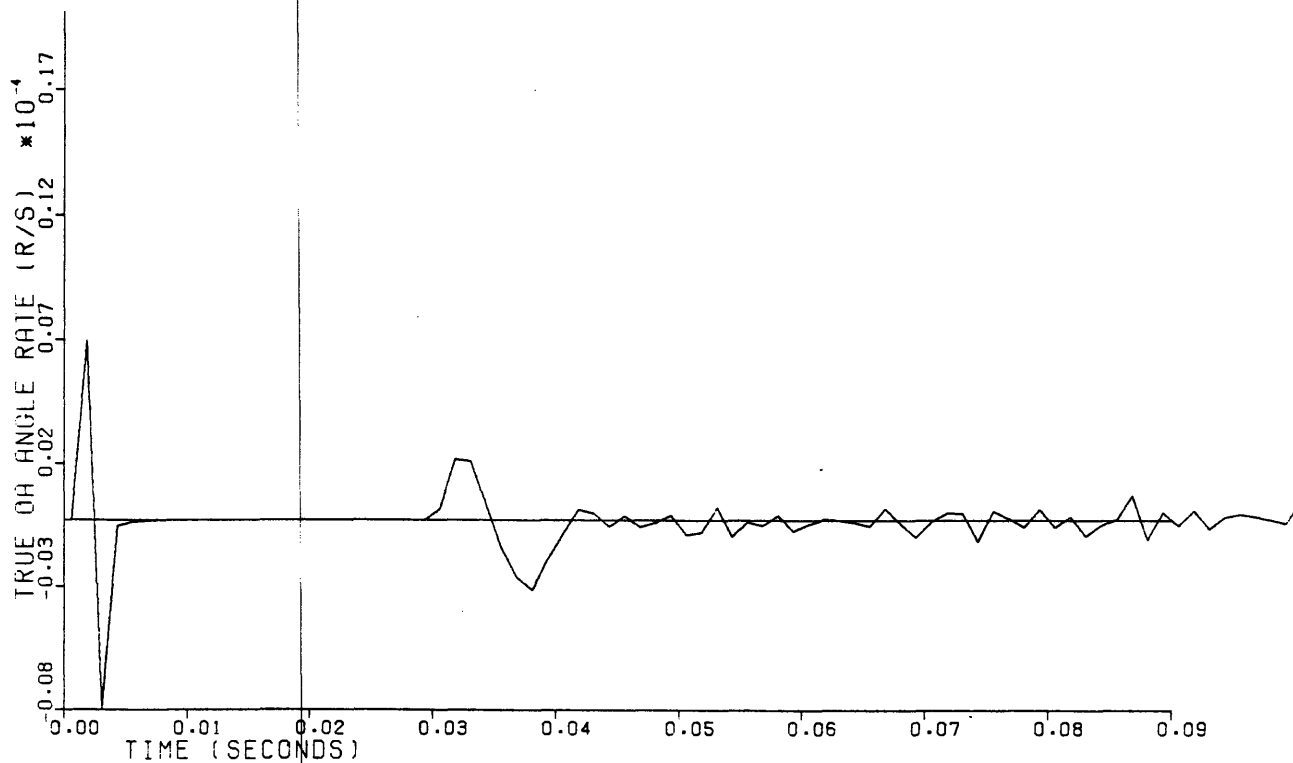


Figure 13-24. Run #14 24 Pt. Average Velocity vs. Time (282°/hr/s Ramp Input)

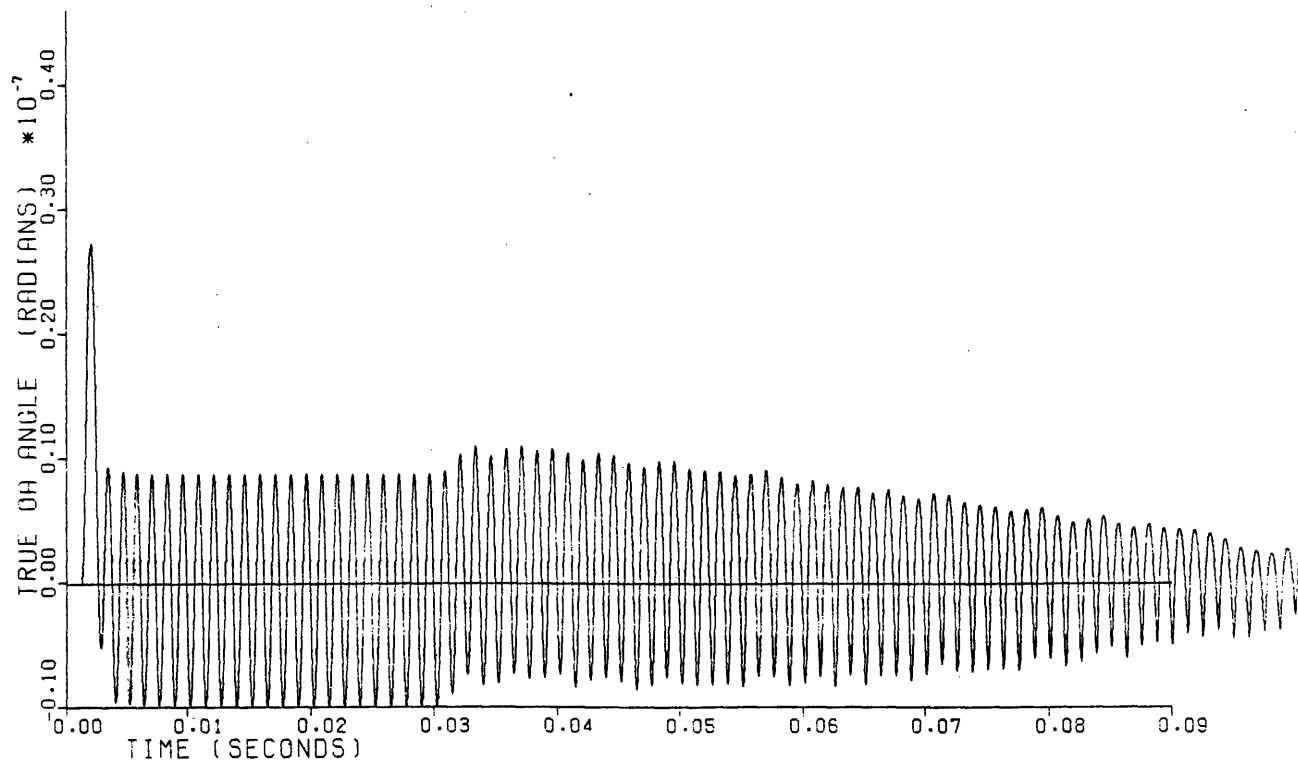


Figure 13-25. Run #15 Float Position vs. Time (282°/hr/s Ramp Input)

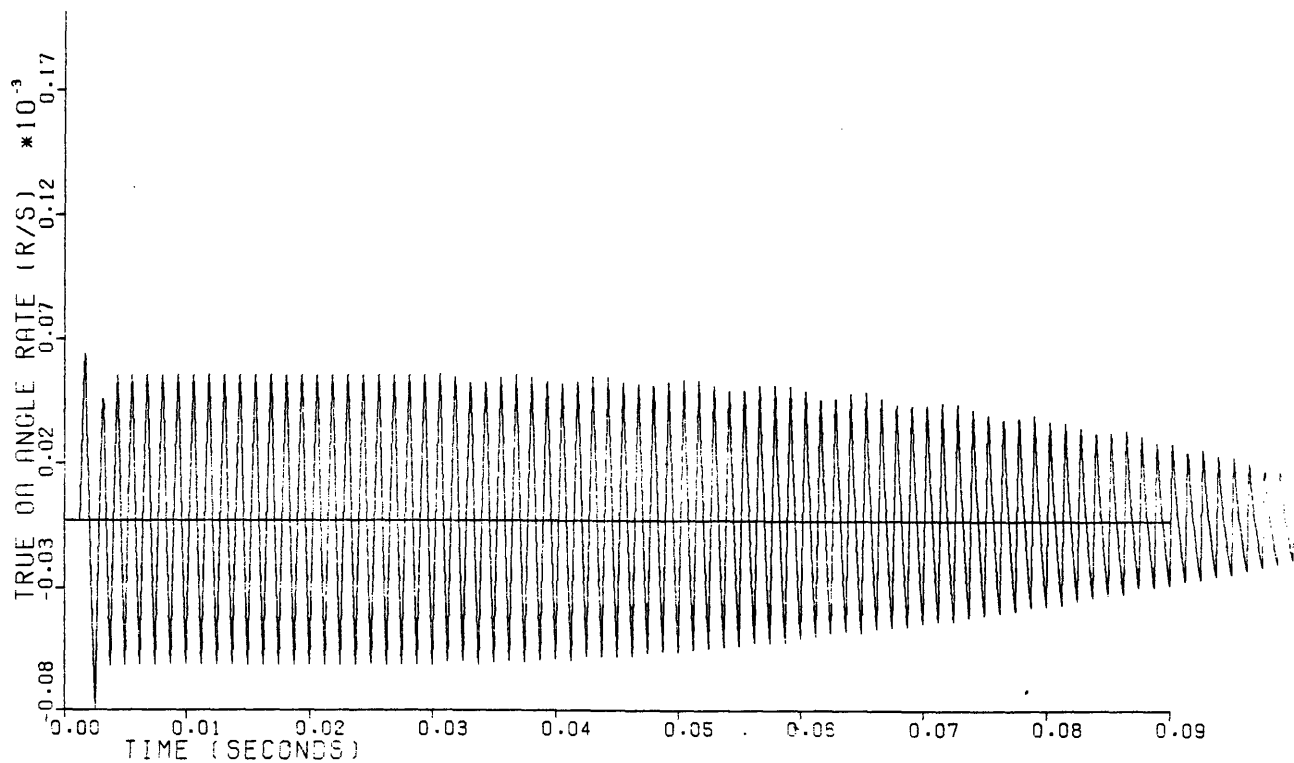


Figure 13-26. Run #15 Float Velocity vs. Time (282°/hr/s Ramp Input)

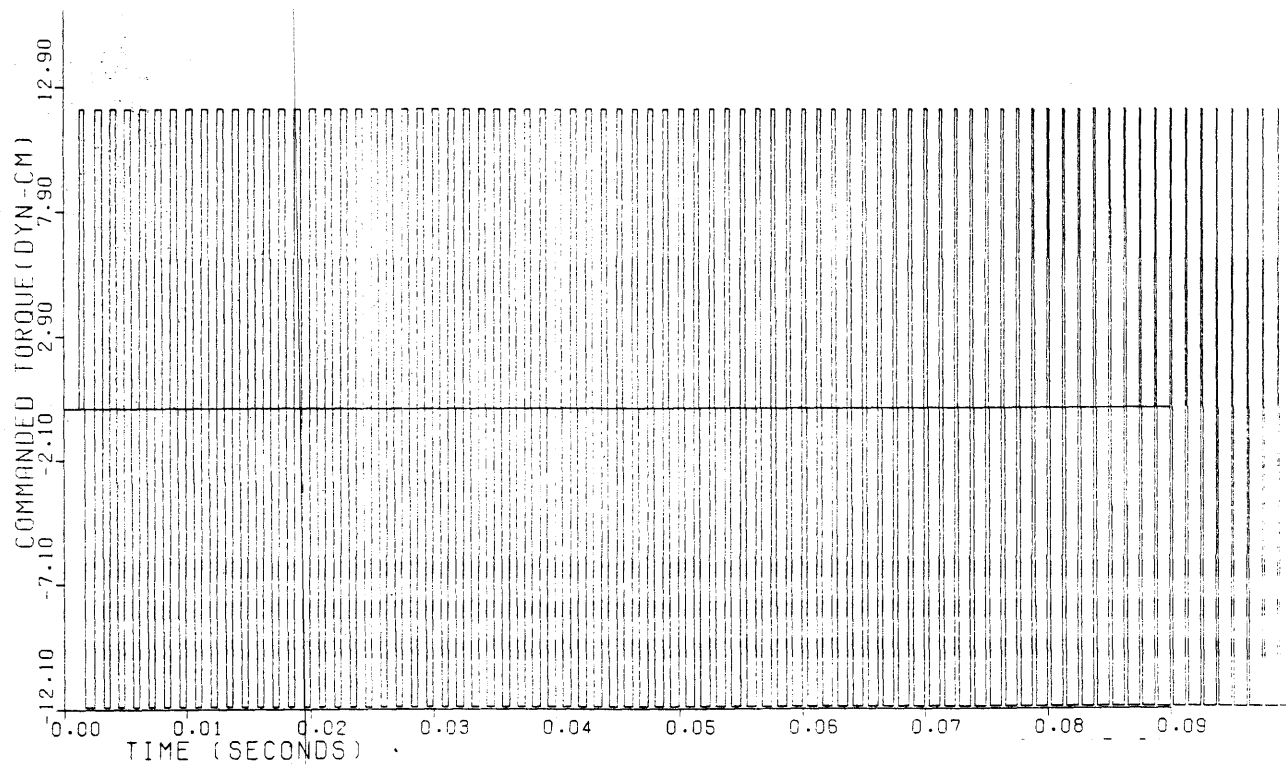


Figure 13-27. Run #15 Commanded Torque vs. Time (282°/hr/s Ramp Input)

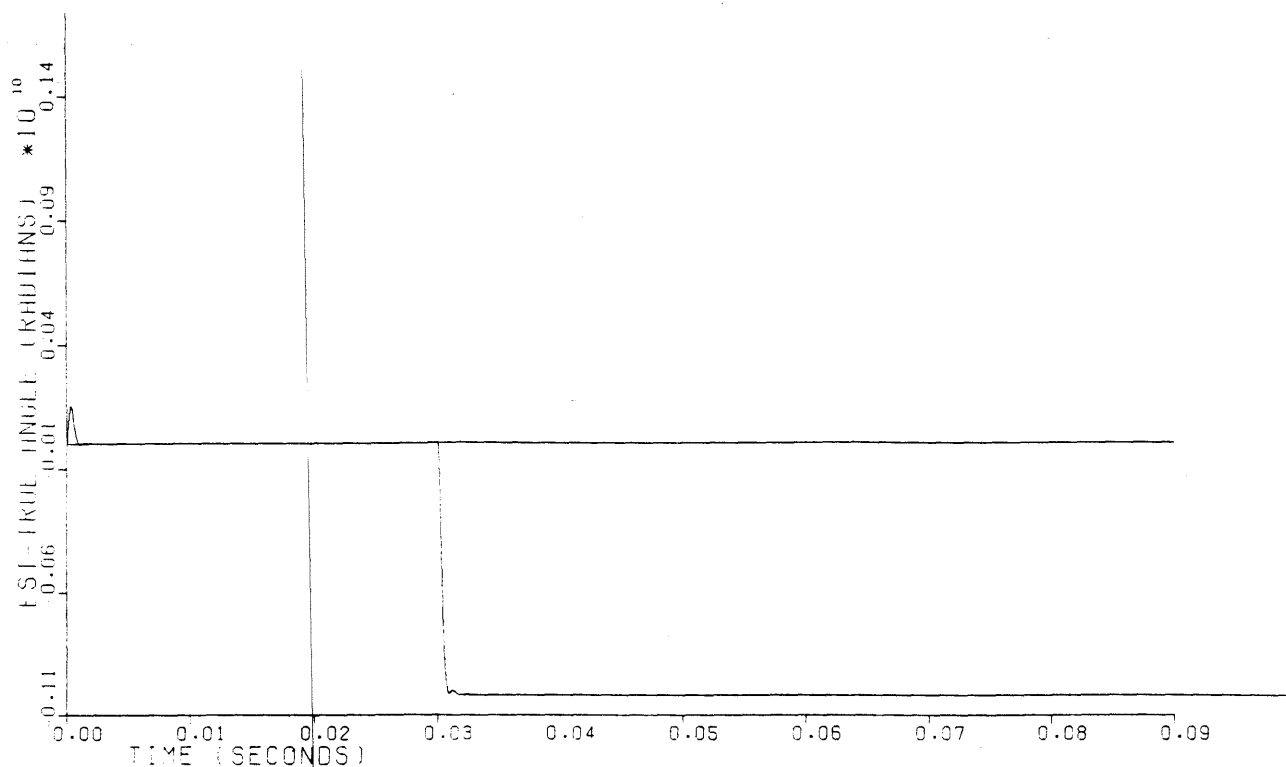


Figure 13-28. Run #15 Position Estimation Error vs. Time (282°/hr/s Ramp Input)

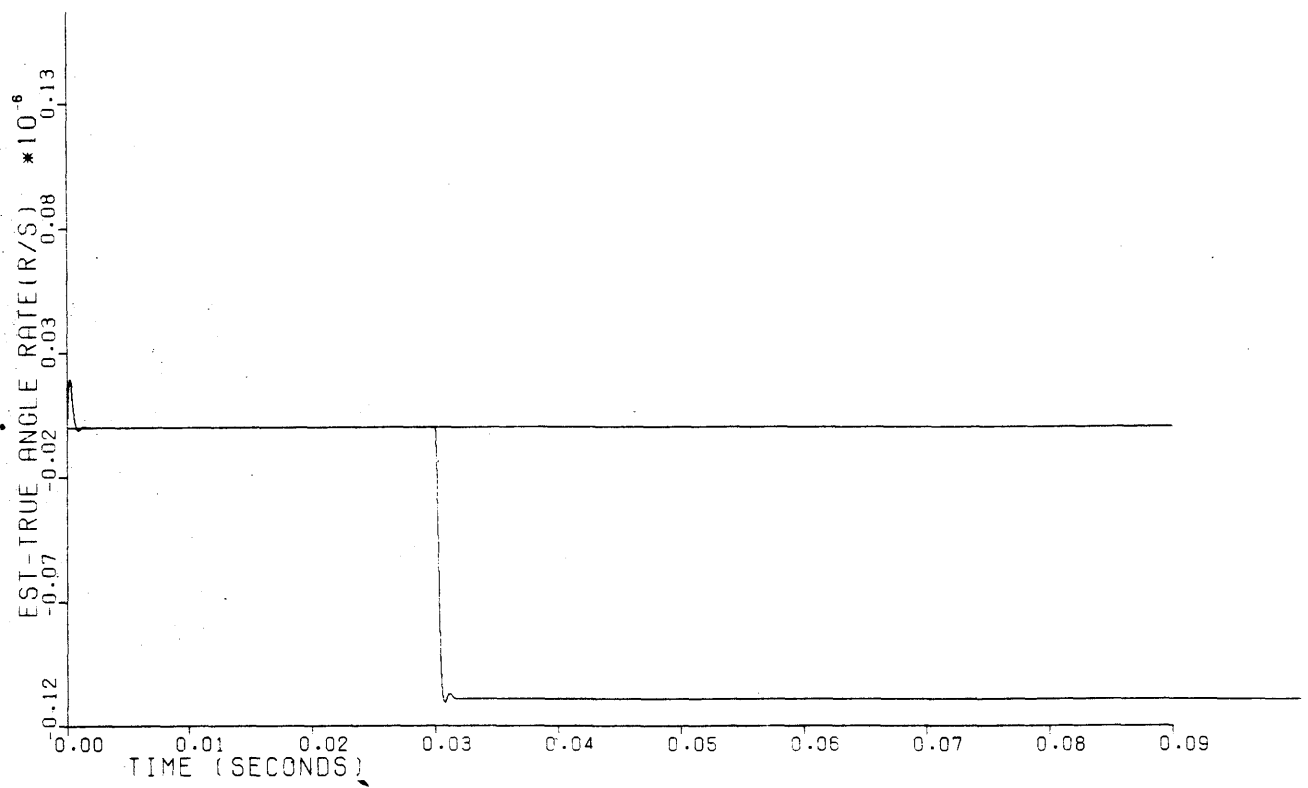


Figure 13-29. Run #15 Velocity Estimation Error vs. Time (282°/hr/s Ramp Input)

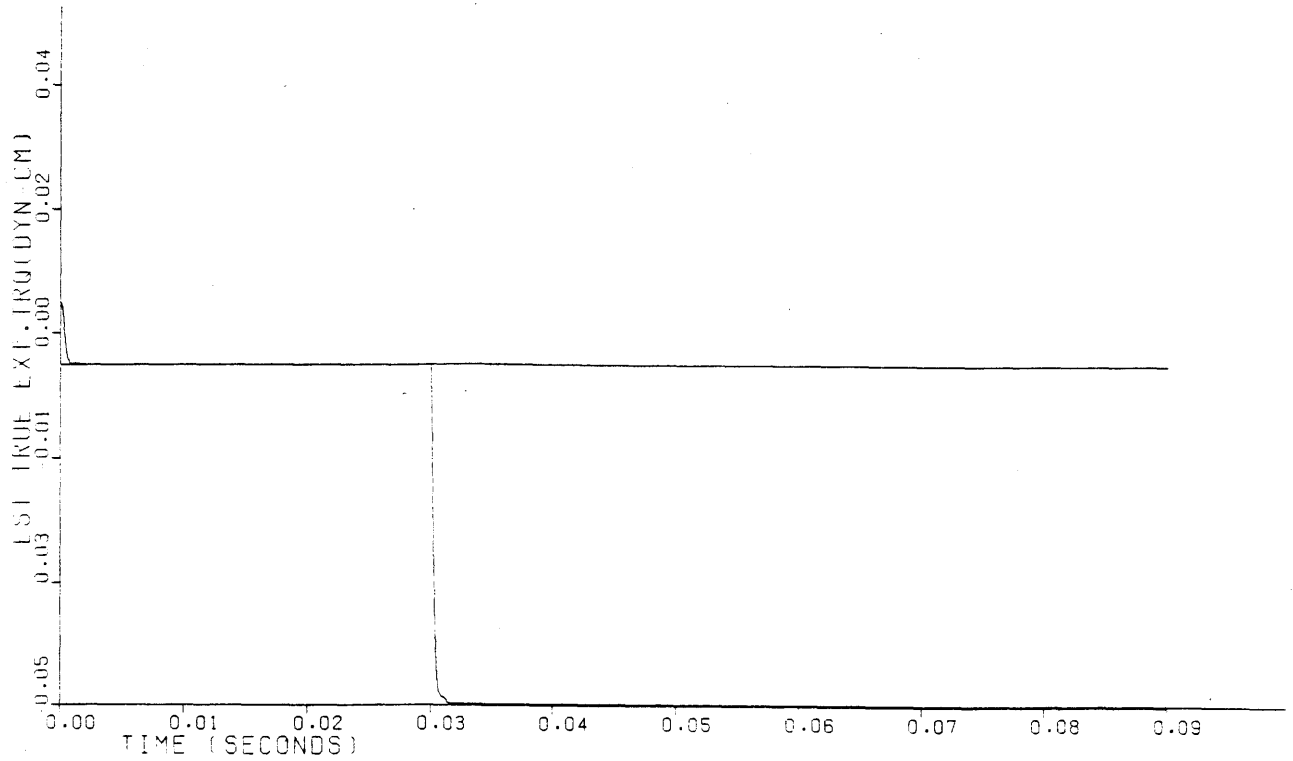


Figure 13-30. Run #15 External Torque Estimation Error vs. Time (282°/hr/s Ramp Input)

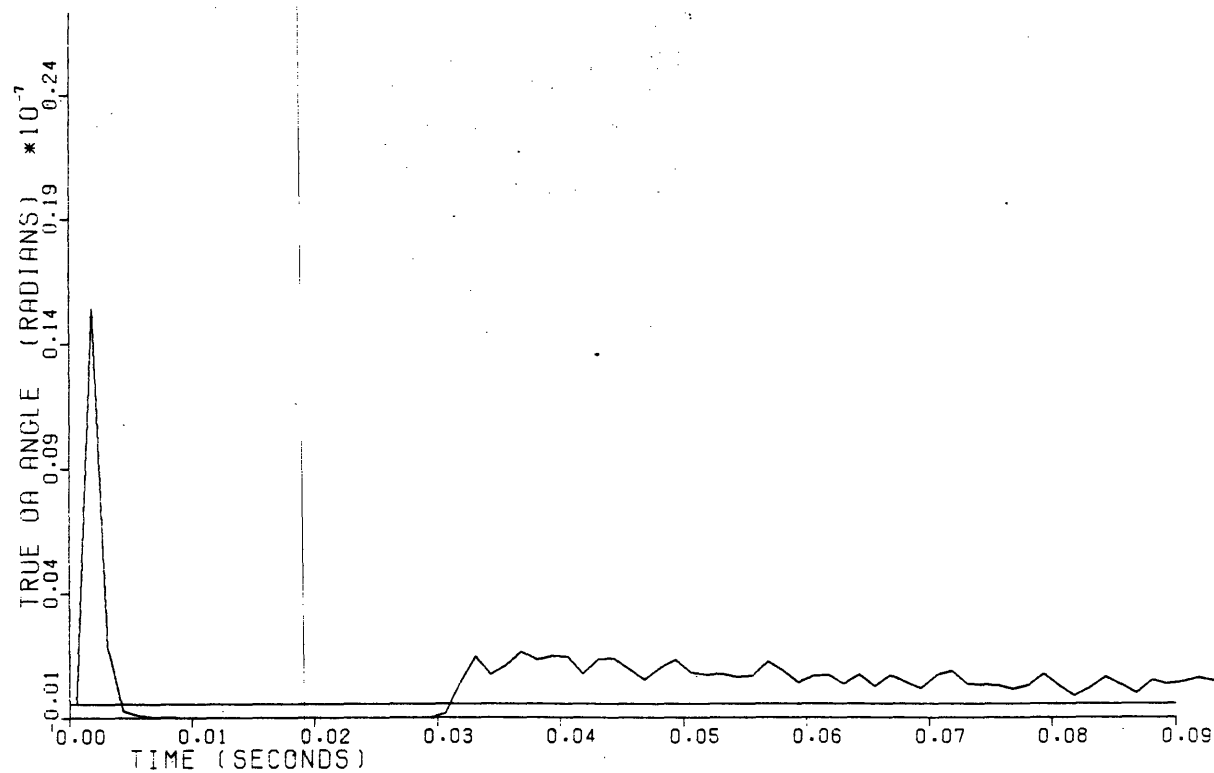


Figure 13-31. Run #15 24 Pt. Average Position vs. Time (282°/hr/s Ramp Input)

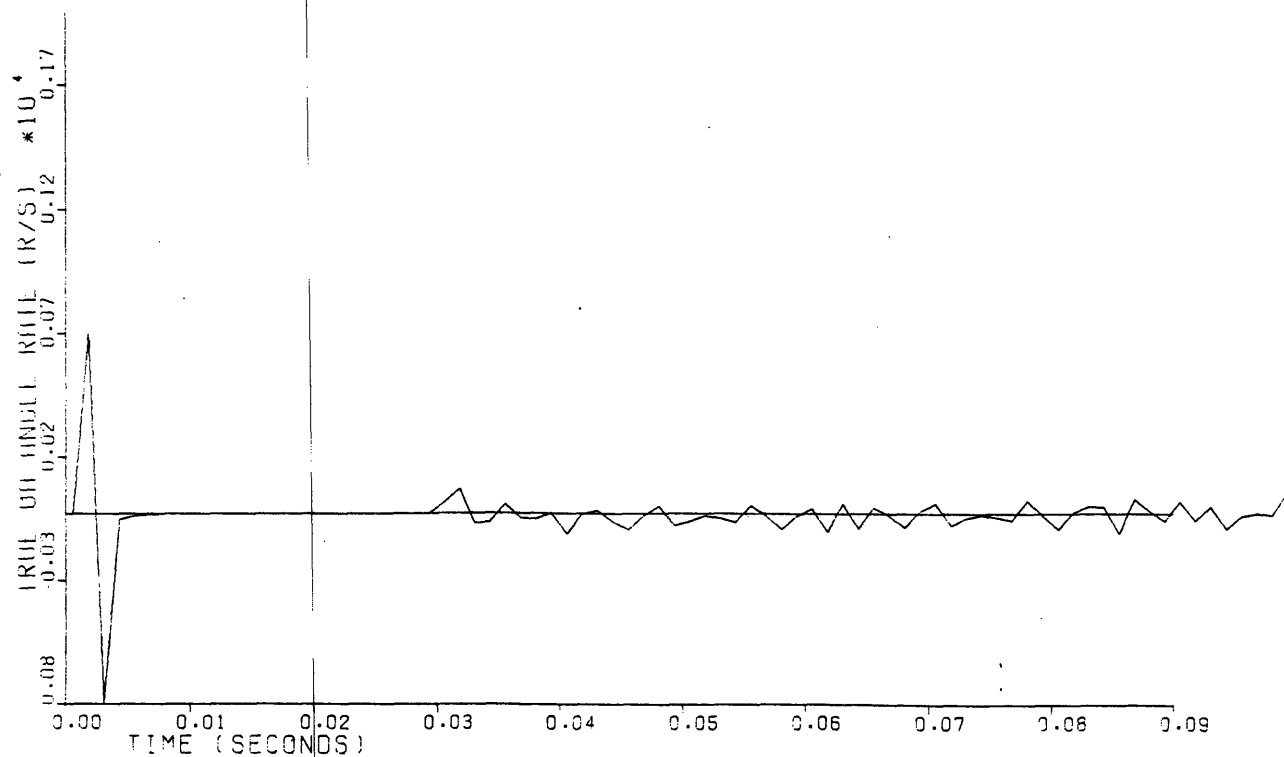


Figure 13-32. Run #15 24 Pt. Average Velocity vs. Time (282°/hr/s Ramp Input)

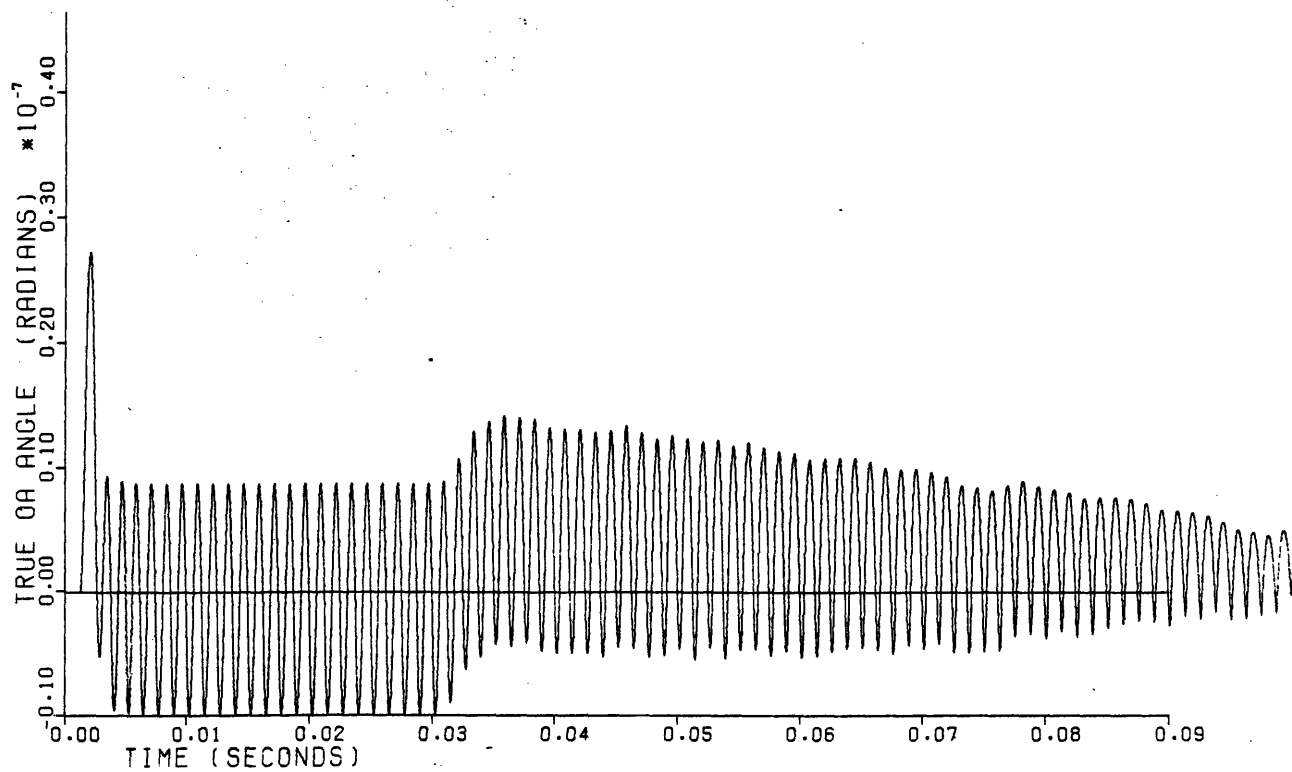


Figure 13-33. Run #16 Float Position vs. Time (282°/hr/s Ramp Input)

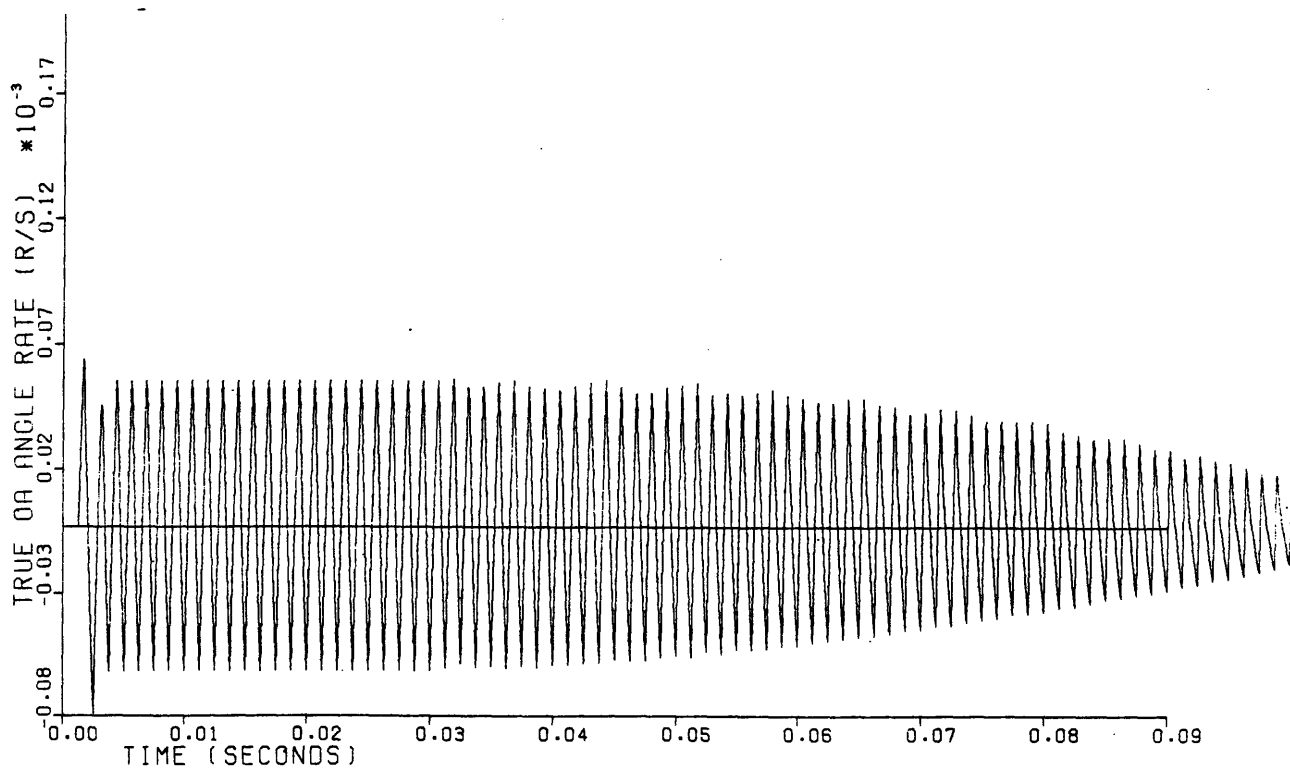


Figure 13-34. Run #16 Float Velocity vs. Time (282°/hr/s Ramp Input)

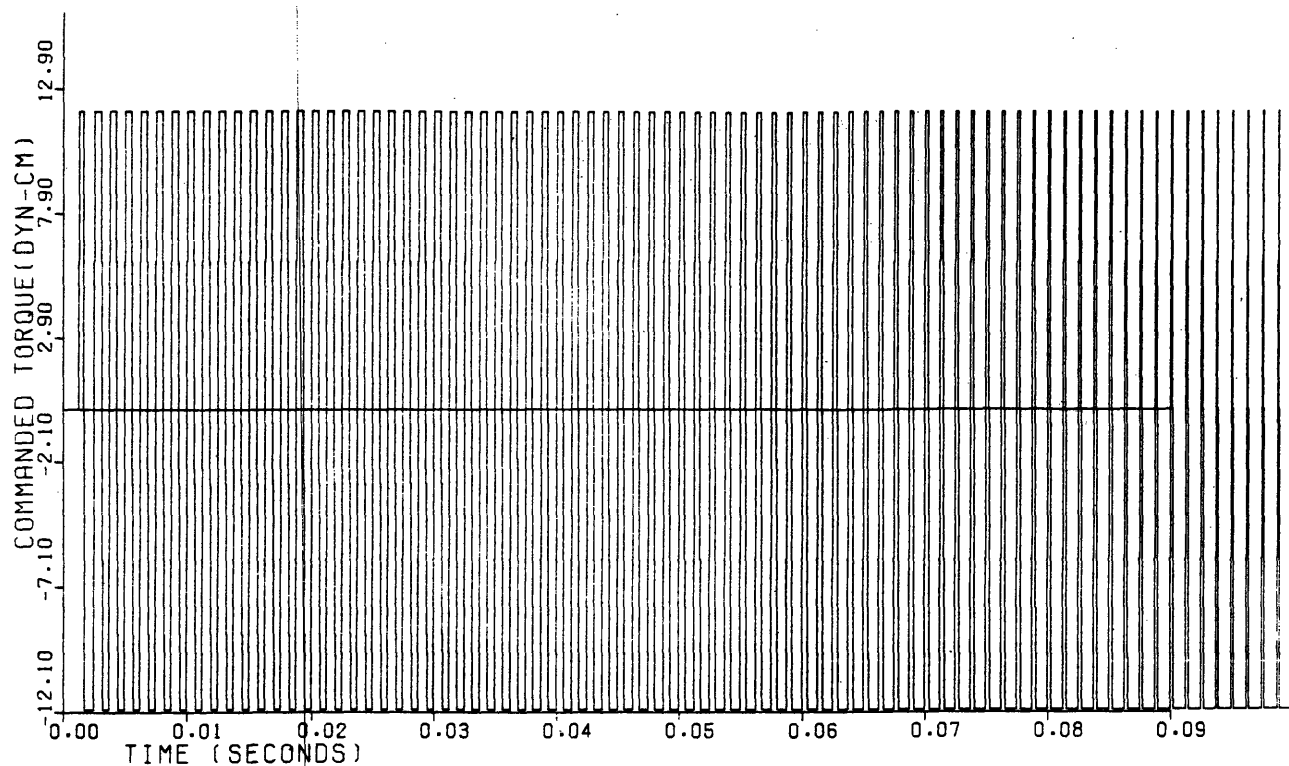


Figure 13-35. Run #16 Commanded Torque vs. Time (282°/hr/s Ramp Input)

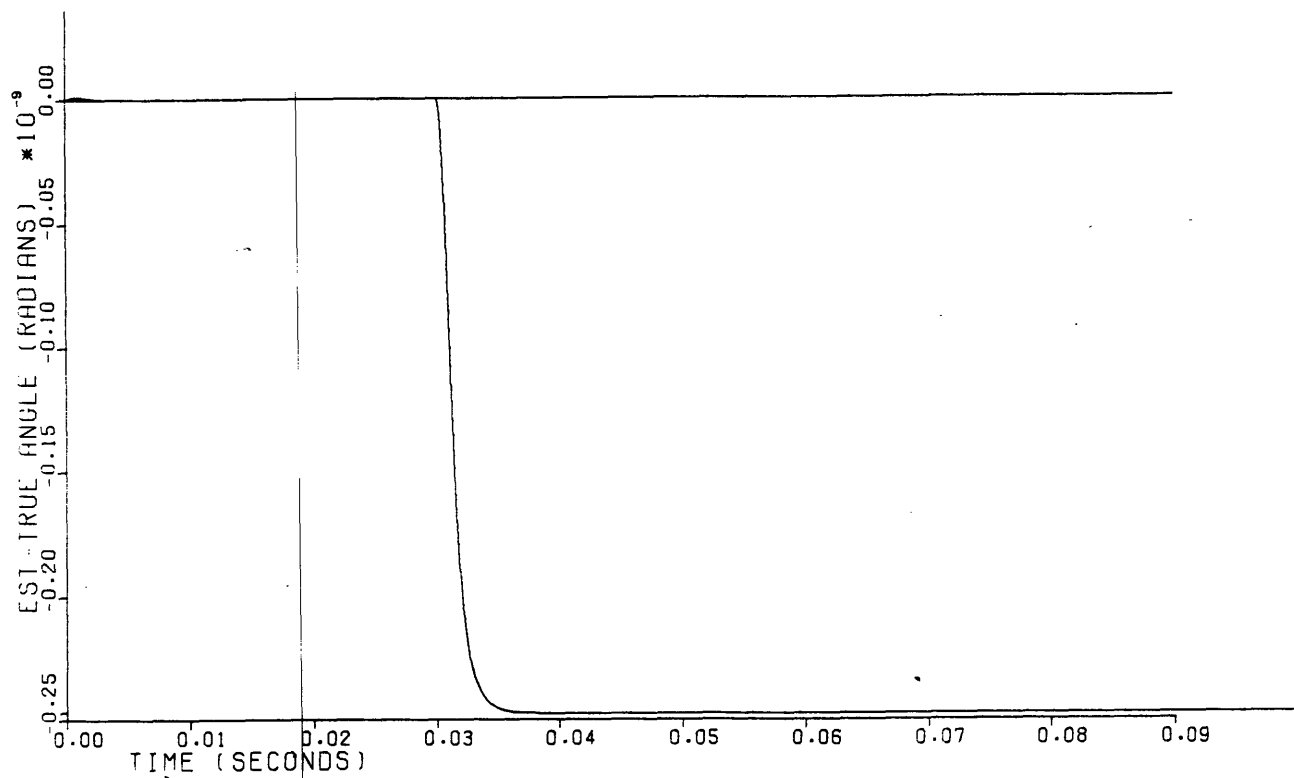


Figure 13-36. Run #16 Position Estimation Error vs. Time (282°/hr/s Ramp Input)

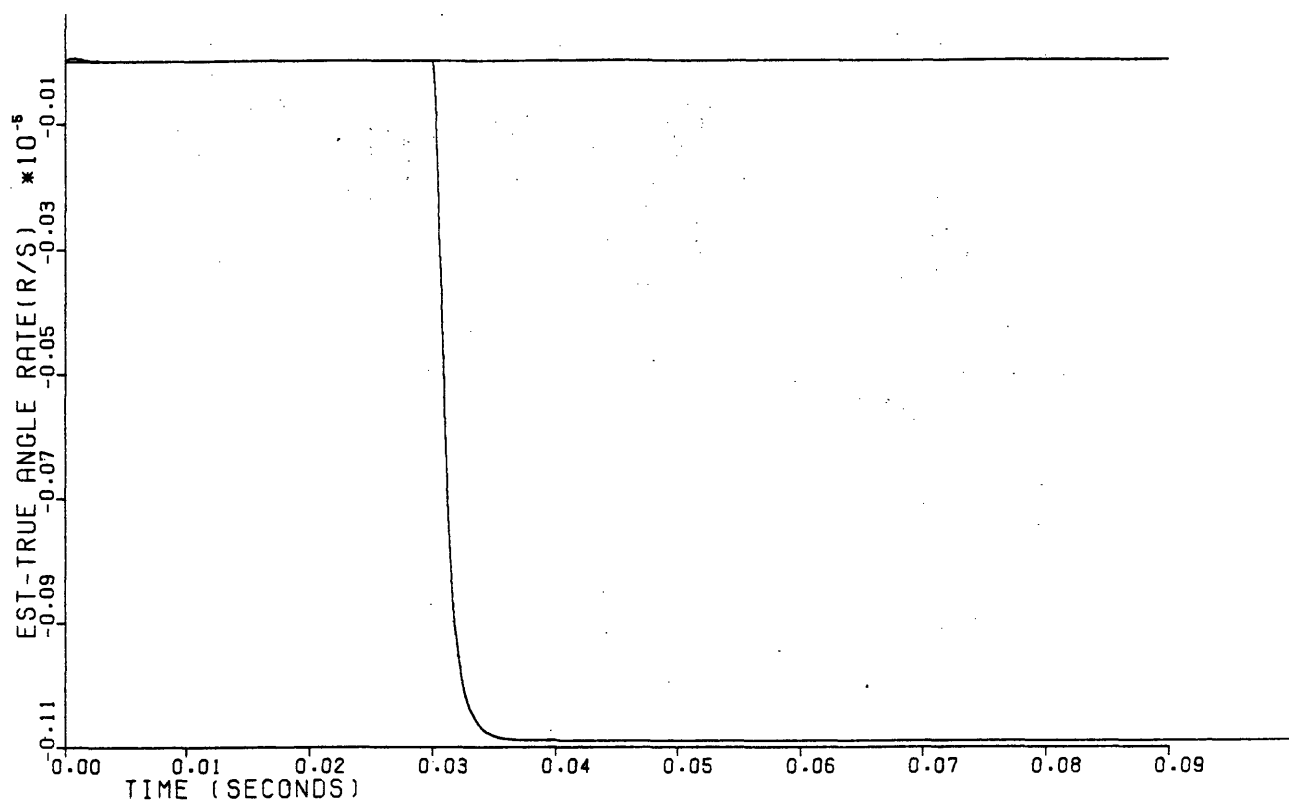


Figure 13-37. Run #16 Velocity Estimation Error vs. Time (282°/hr/s Ramp Input)

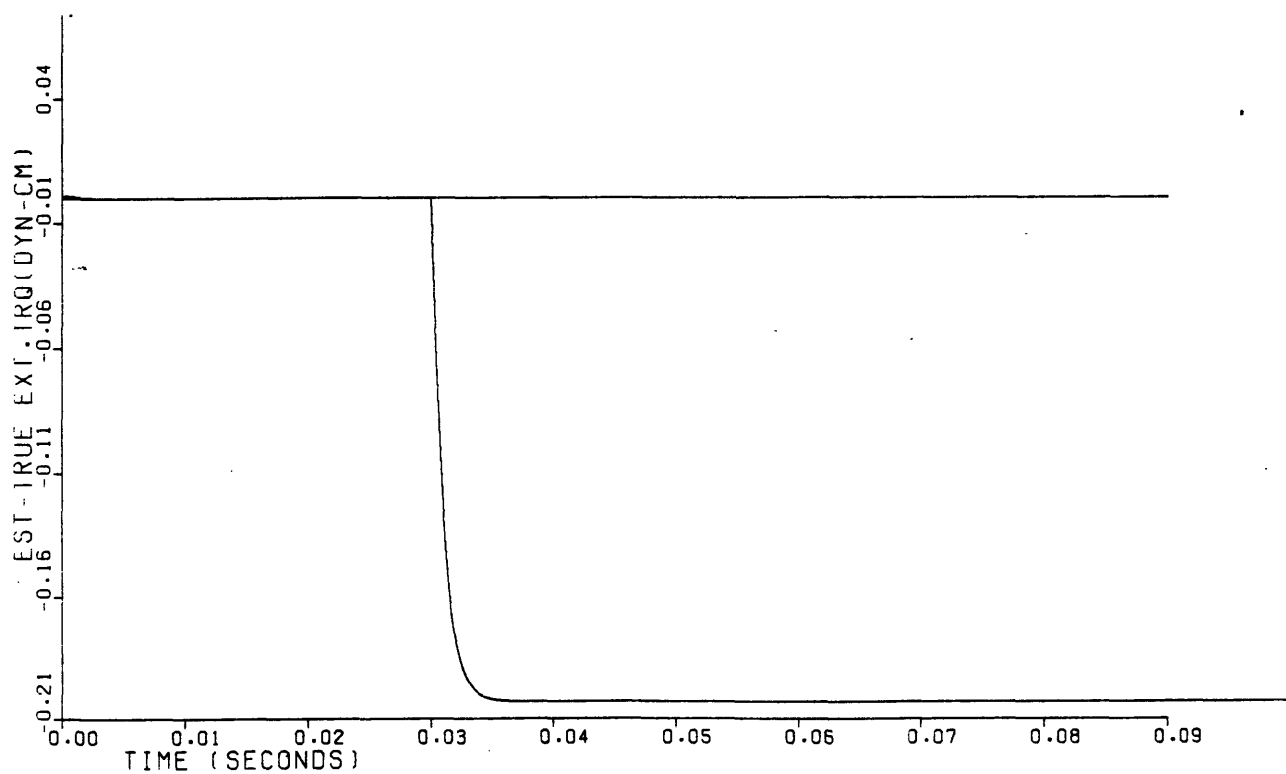


Figure 13-38. Run #16 External Torque Estimation Error vs. Time (282°/hr/s Ramp Input)

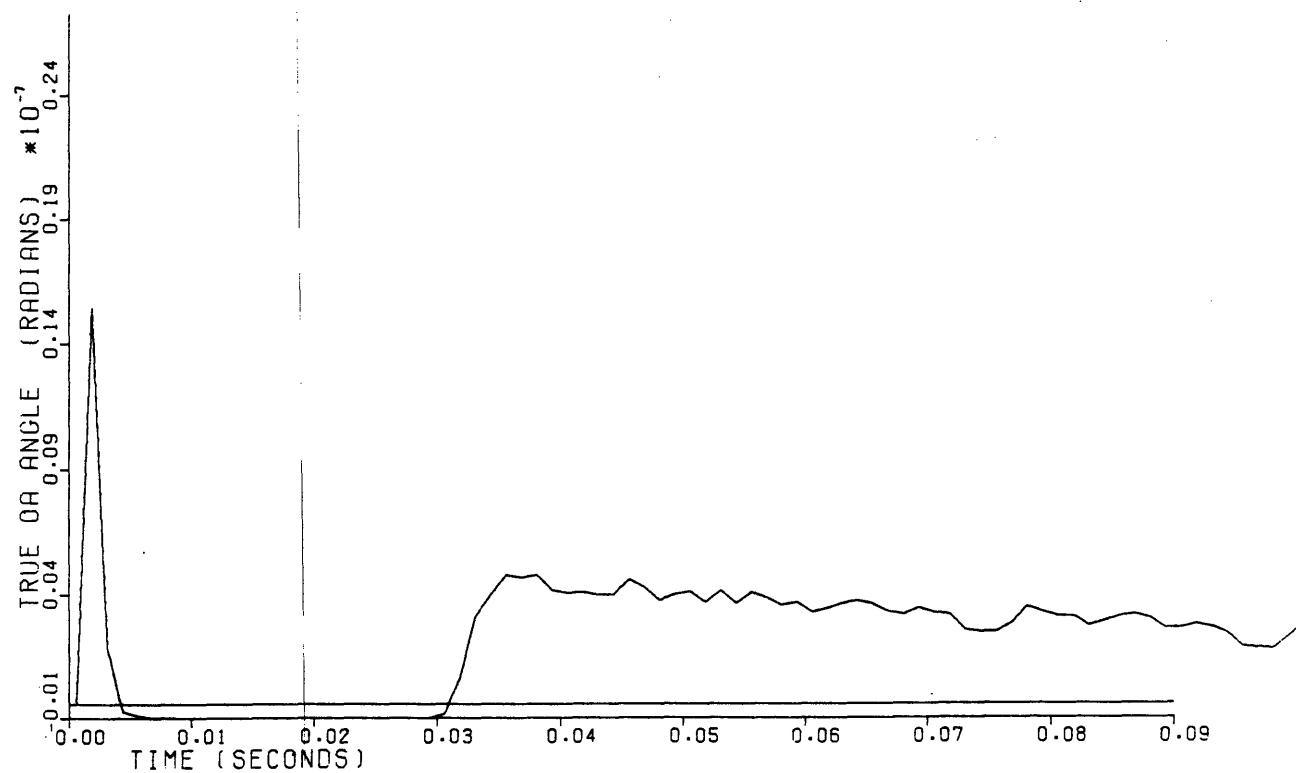


Figure 13-39. Run #16 24 Pt. Average Position vs. Time (282°/hr/s Ramp Input)

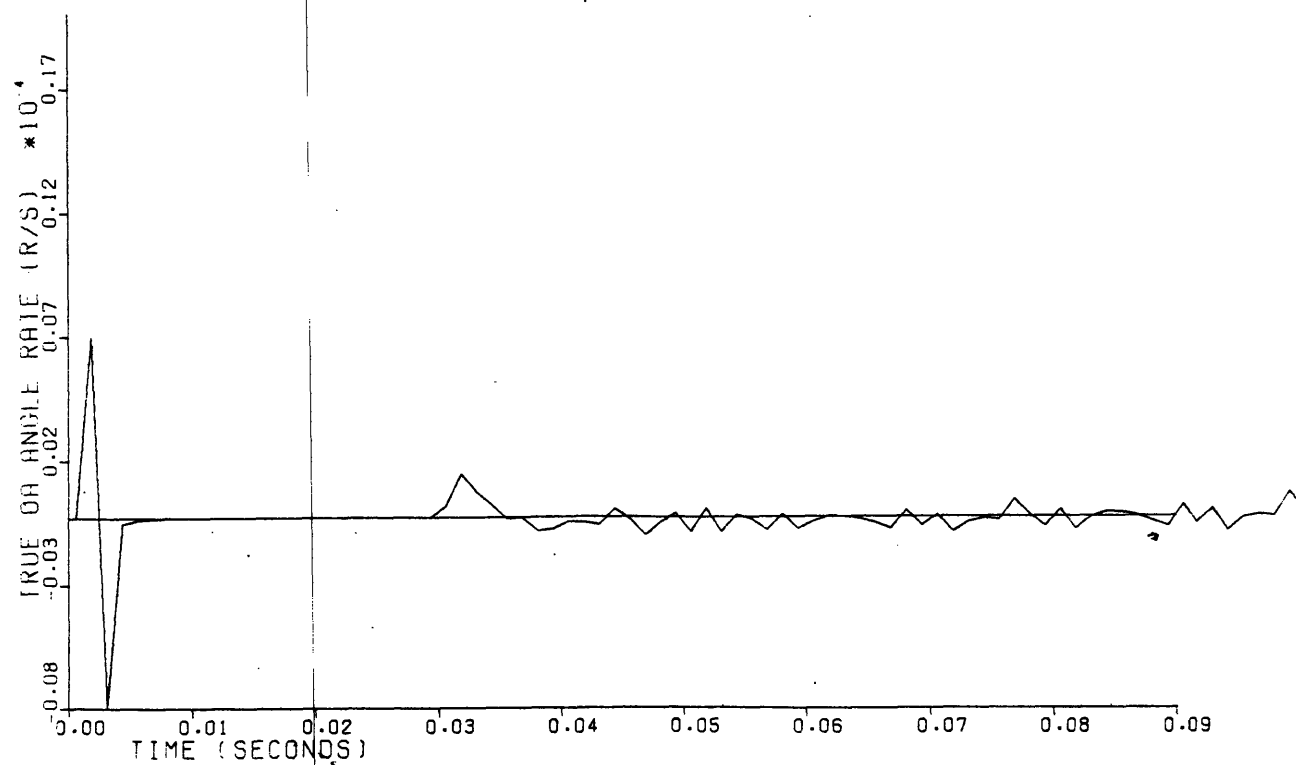


Figure 13-40. Run #16 24 Pt. Average Velocity vs. Time (282°/hr/s Ramp Input)

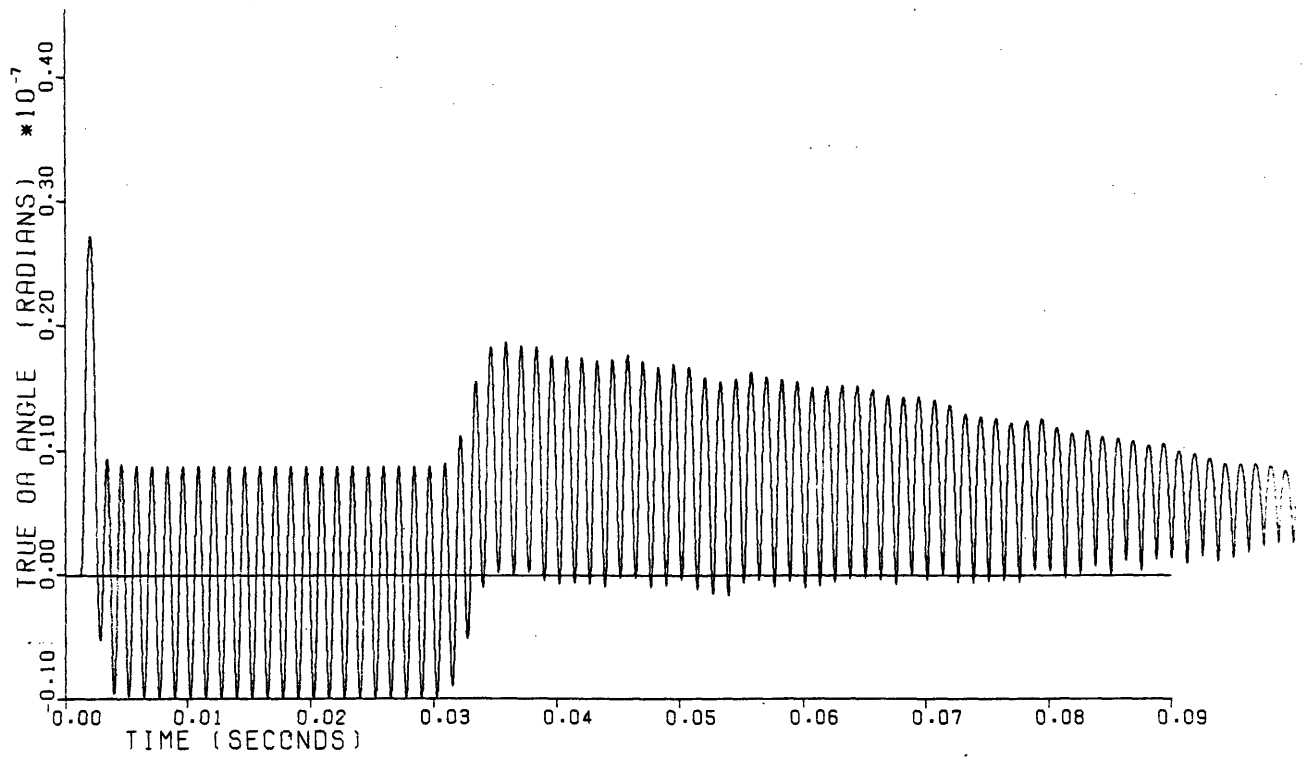


Figure 13-41. Run #17 Float Position vs. Time (282°/hr/s Ramp Input)

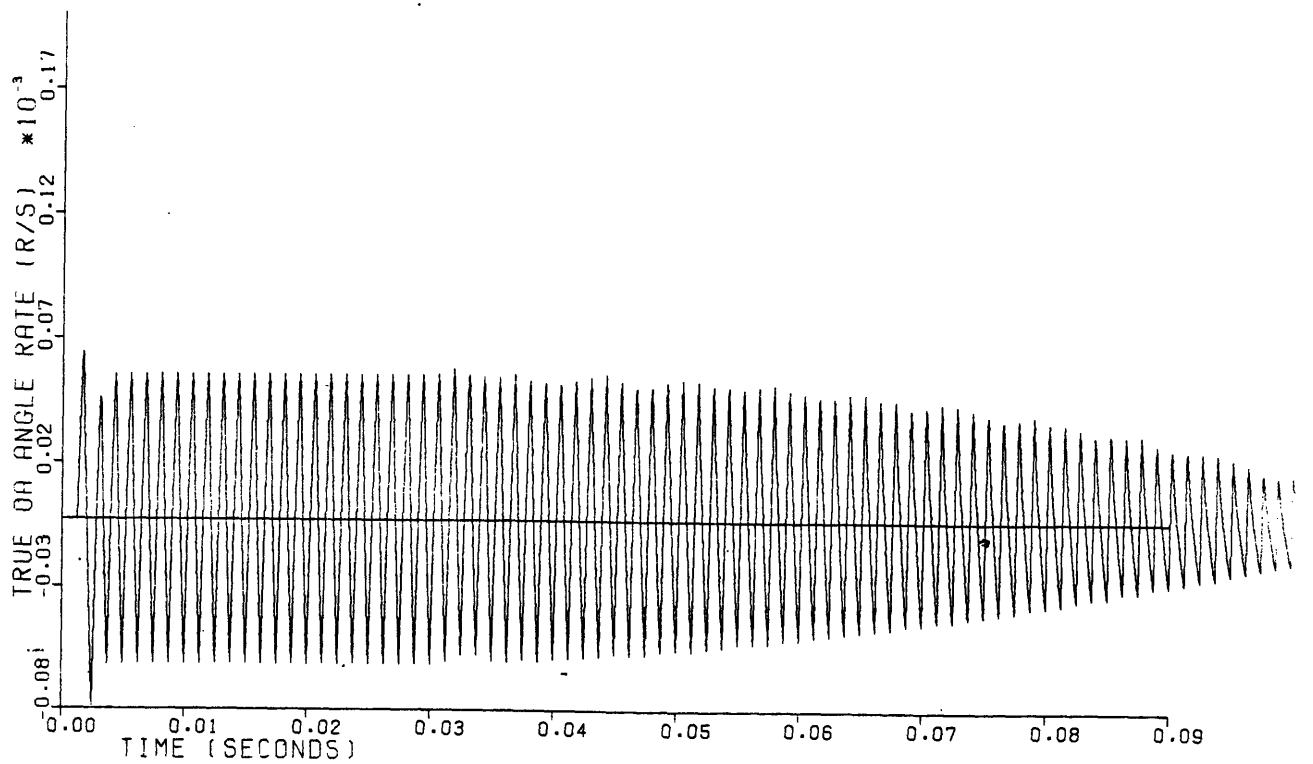


Figure 13-42. Run #17 Float Velocity vs. Time (282°/hr/s Ramp Input)

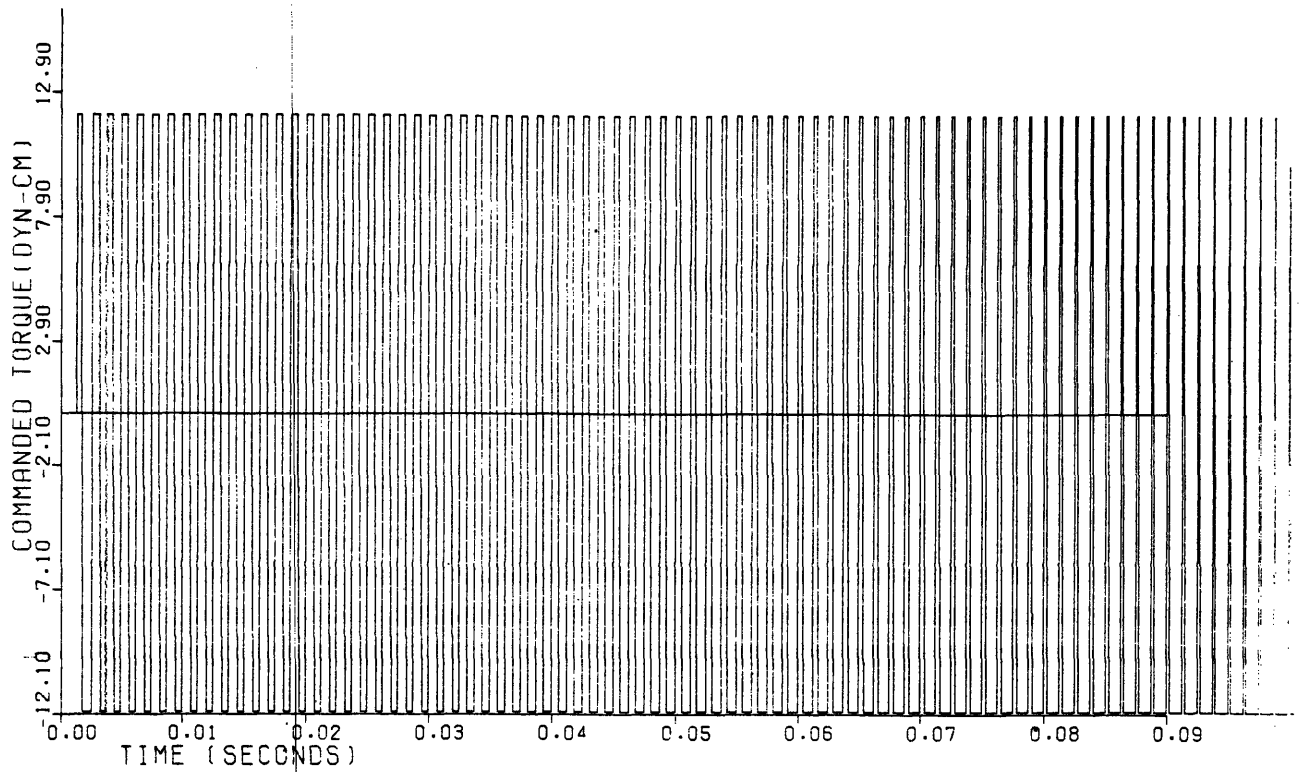


Figure 13-43. Run #17 Commanded Torque vs. Time (282°/hr/s Ramp Input)

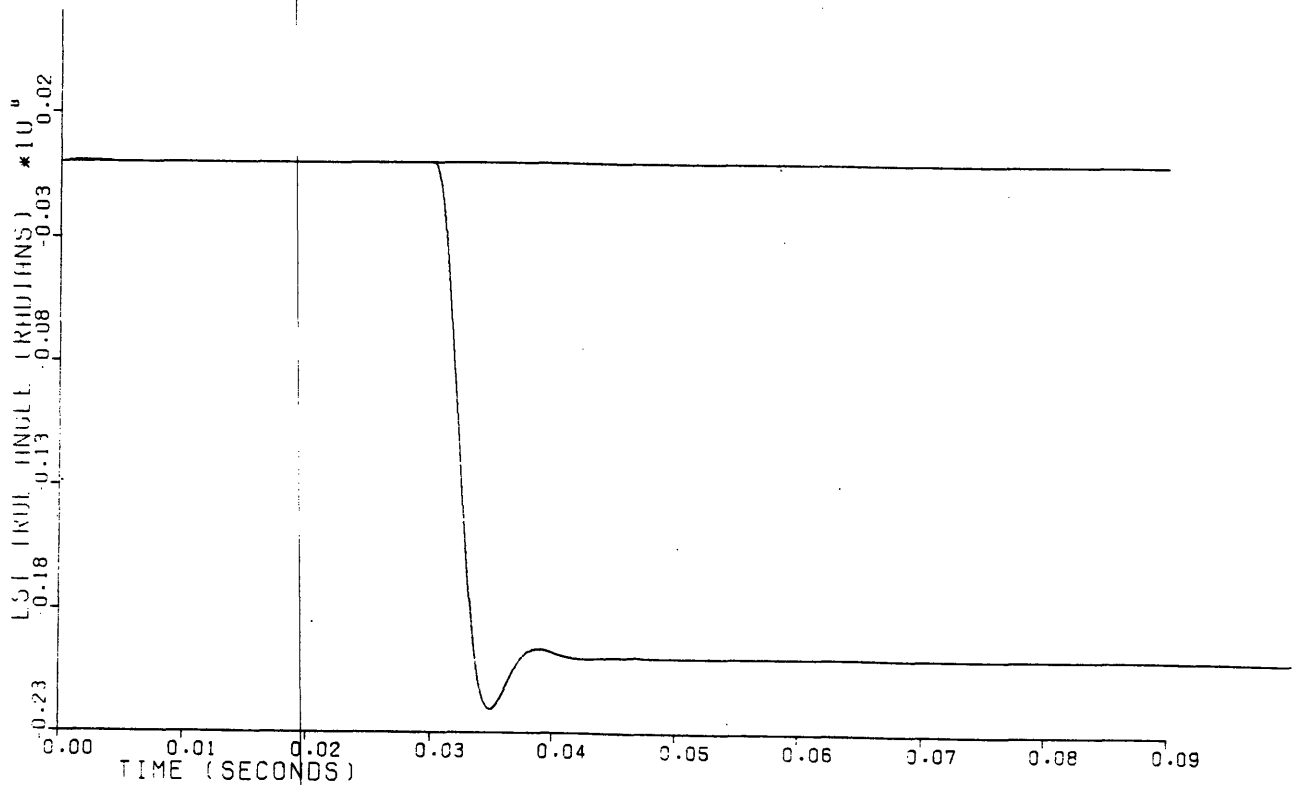


Figure 13-44. Run #17 Position Estimation Error vs. Time (282°/hr/s Ramp Input)

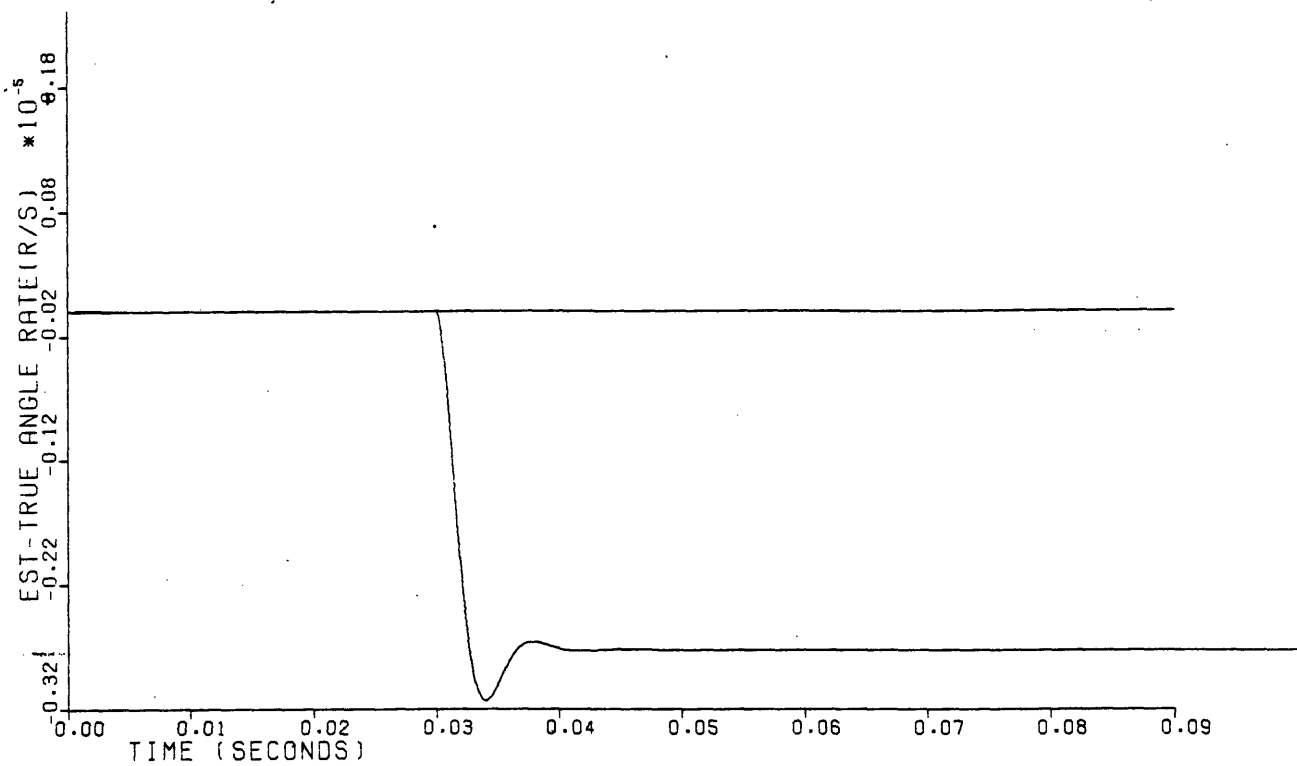


Figure 13-45. Run #17 Velocity Estimation Error vs. Time (282°/hr/s Ramp Input)

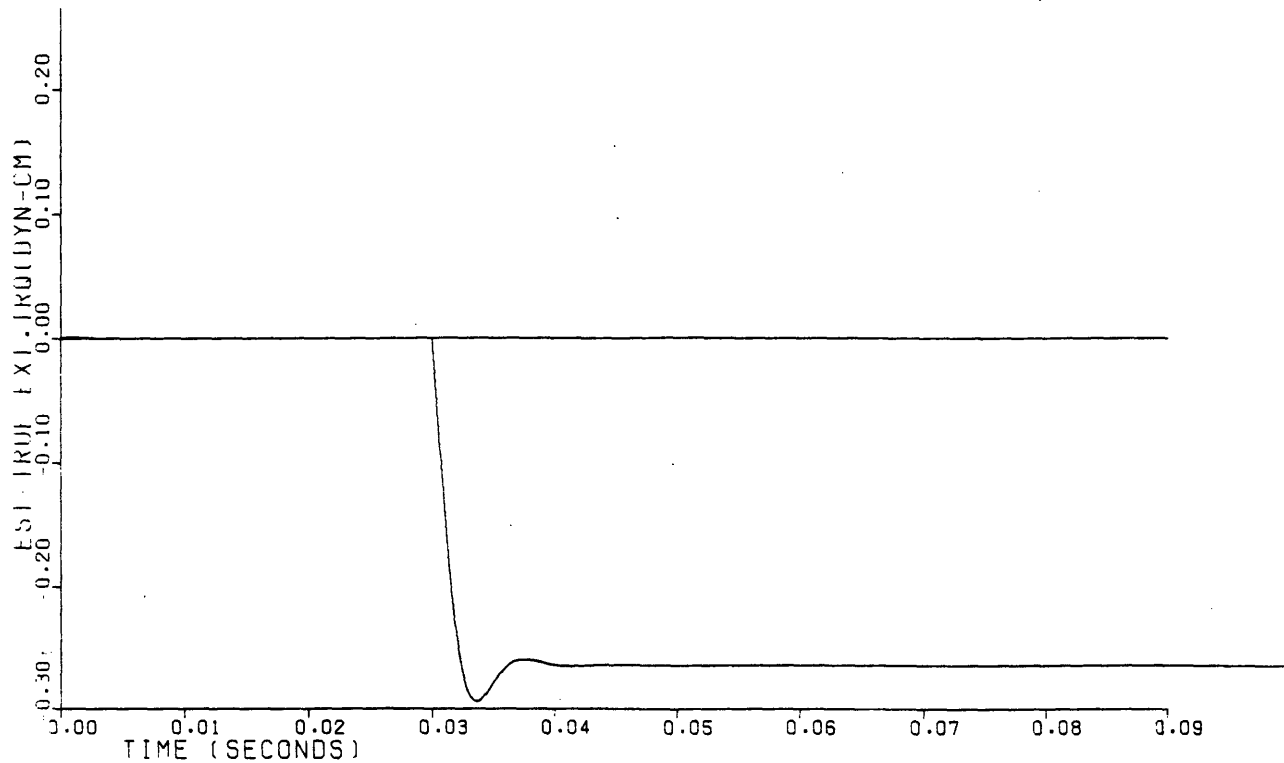


Figure 13-46. Run #17 External Torque Estimation Error vs. Time (282°/hr/s Ramp Input)

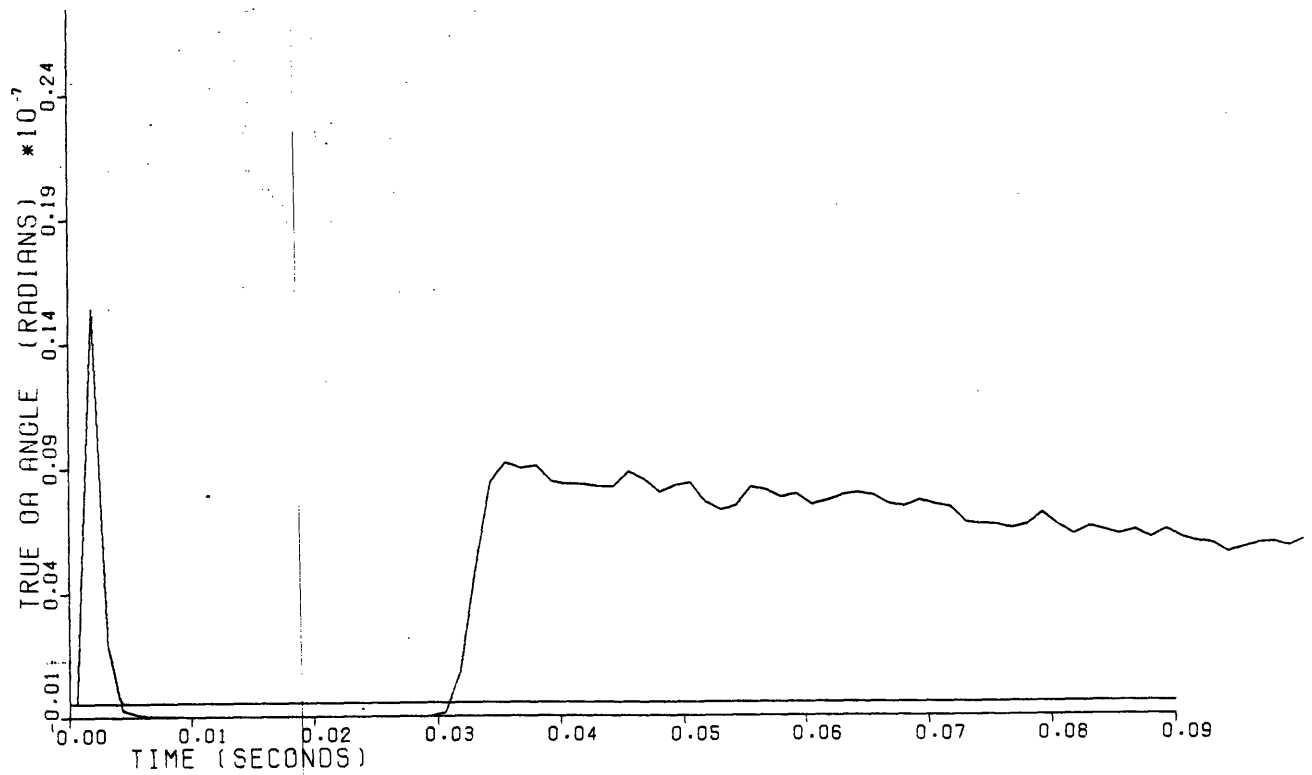


Figure 13-47. Run #17 24 Pt. Average Position vs. Time (282°/hr/s Ramp Input)

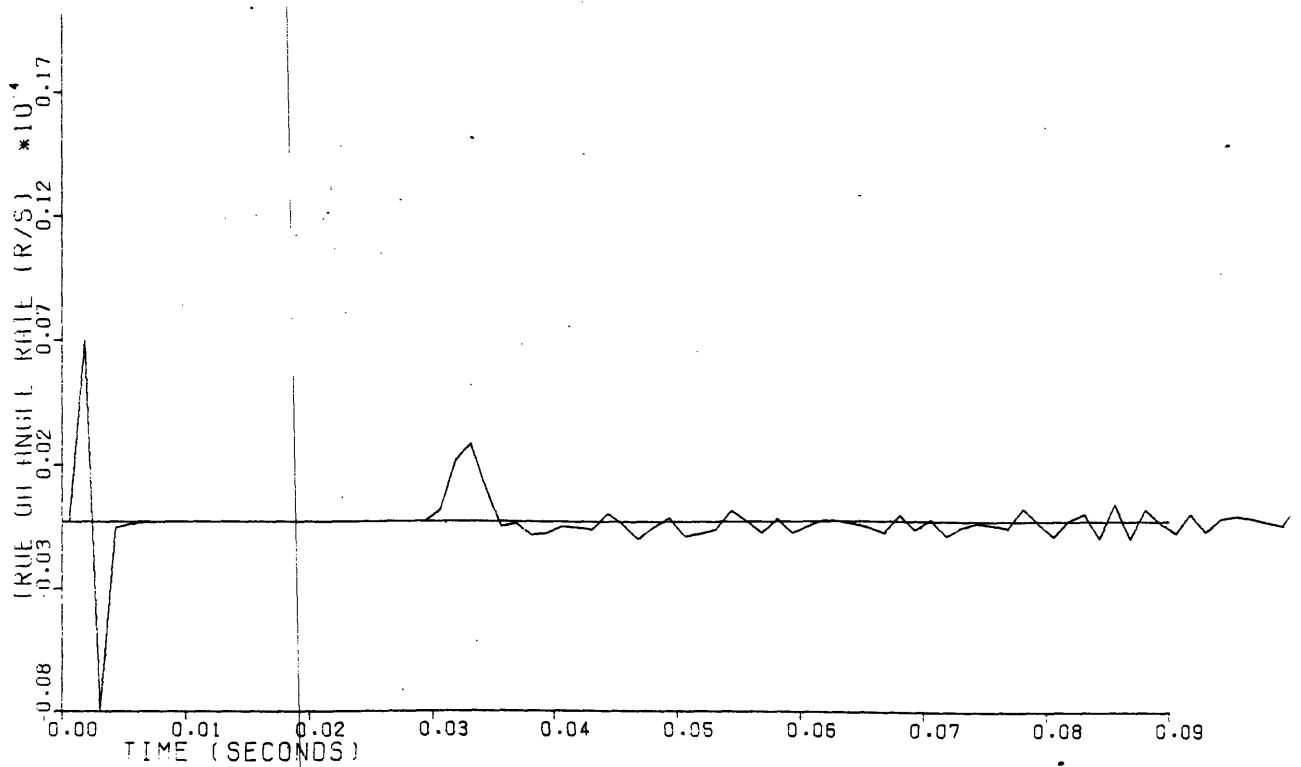


Figure 13-48. Run #17 24 Pt. Average Velocity vs. Time (282°/hr/s Ramp Input)

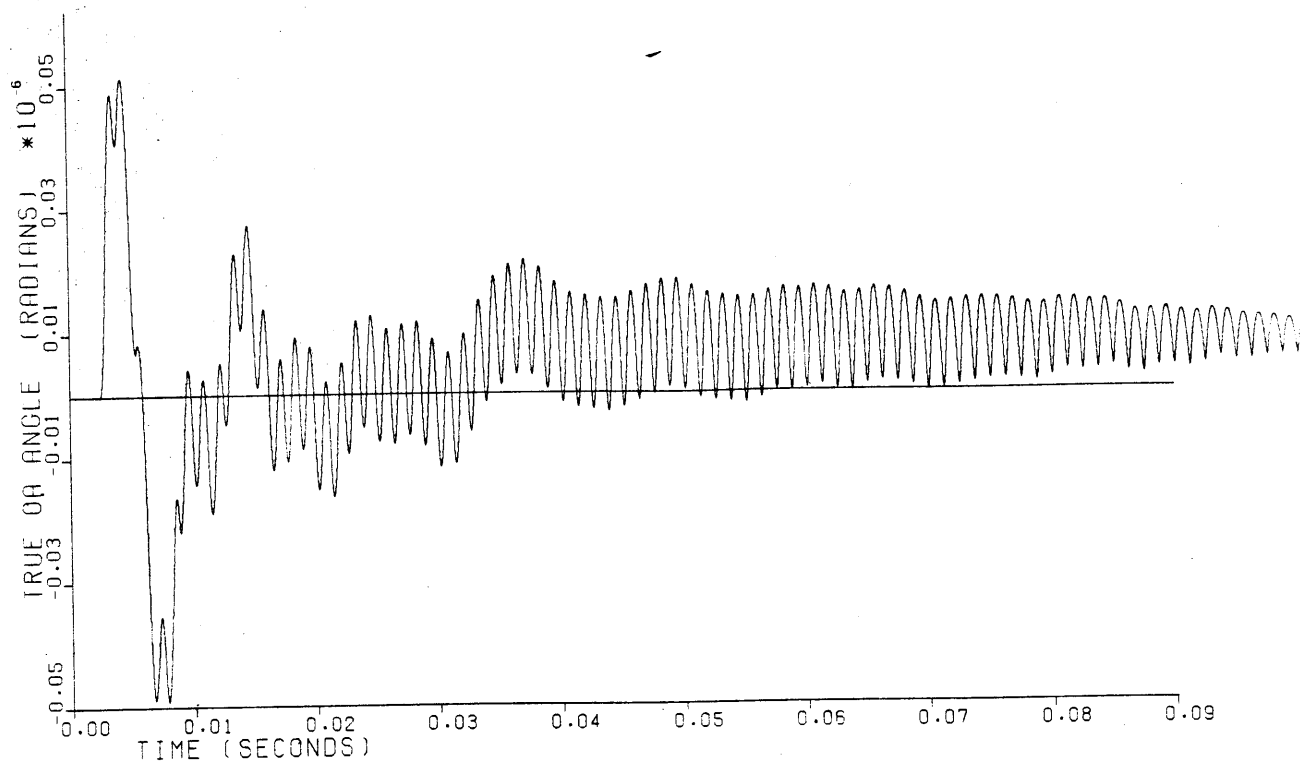


Figure 13-49. Run #18 Float Position vs. Time (282°/hr/s Ramp Input)

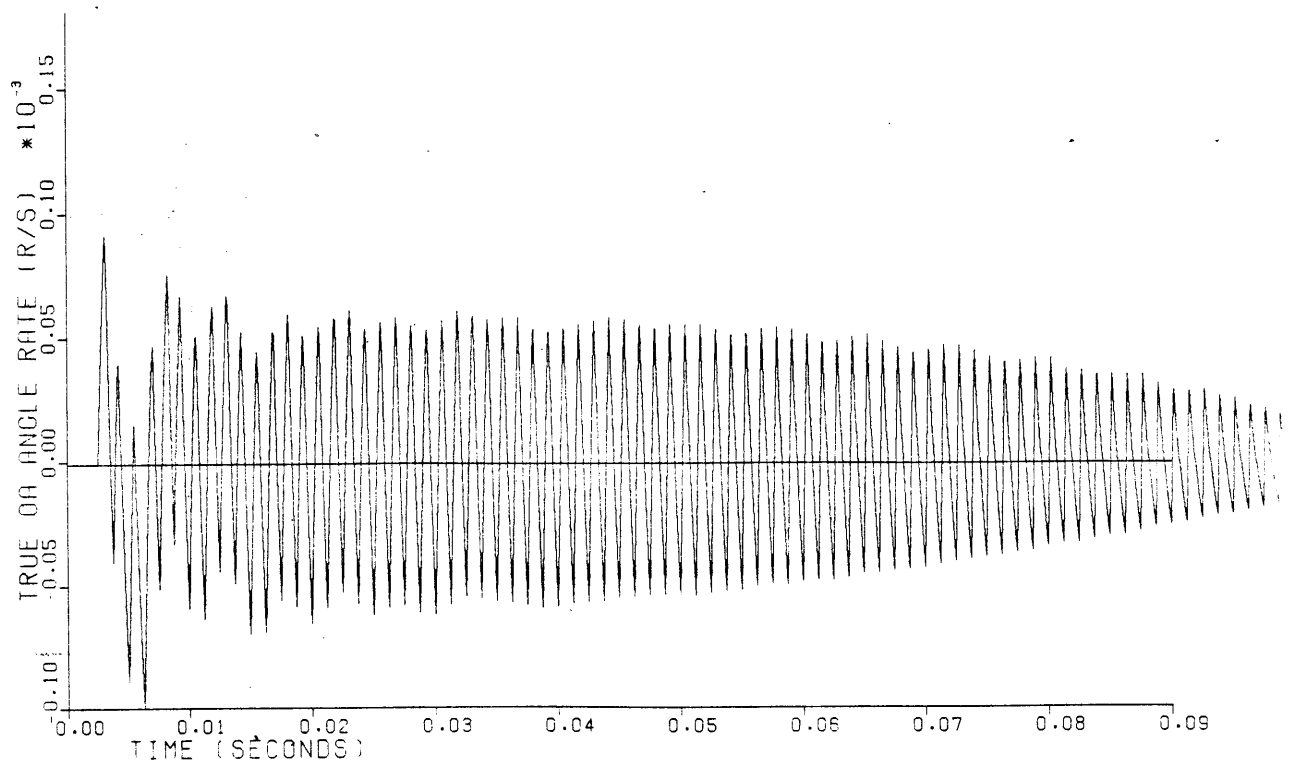


Figure 13-50. Run #18 Float Velocity vs. Time (282°/hr/s Ramp Input)

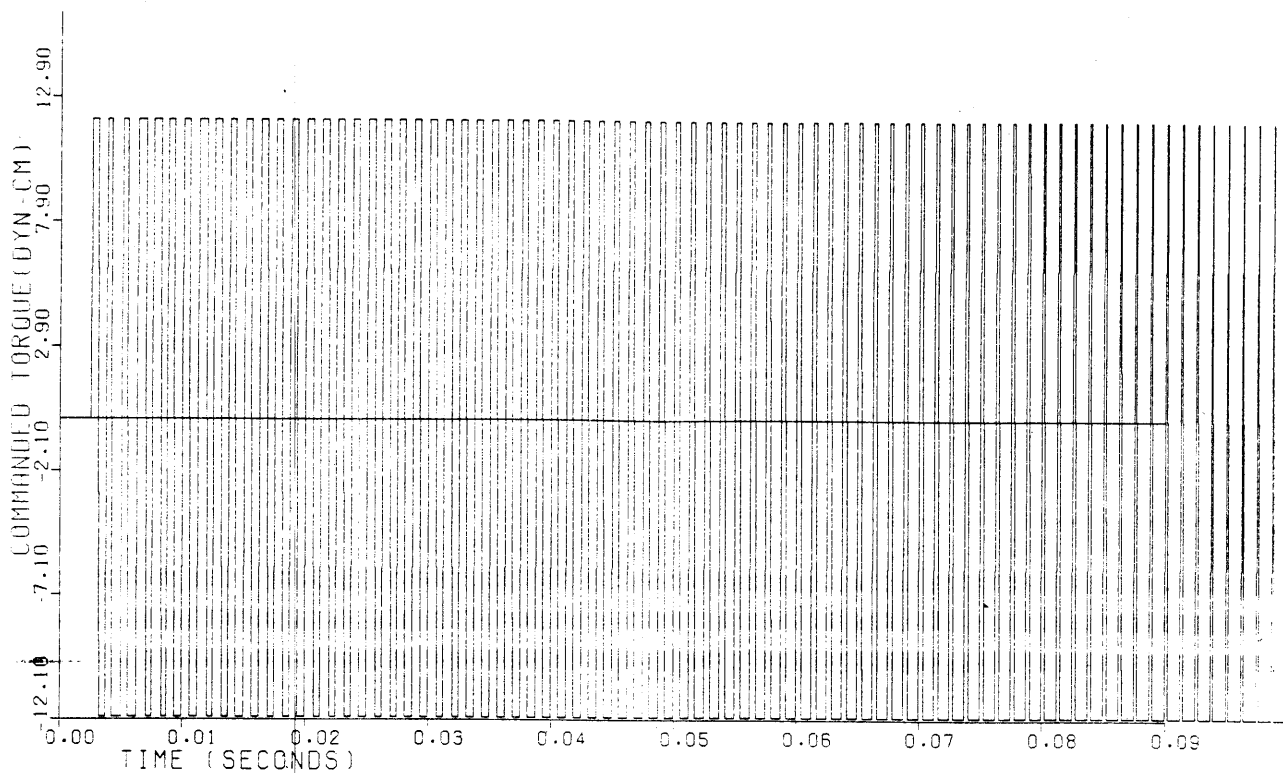


Figure 13-51. Run #18 Commanded Torque vs. Time (282°/hr/s Ramp Input)

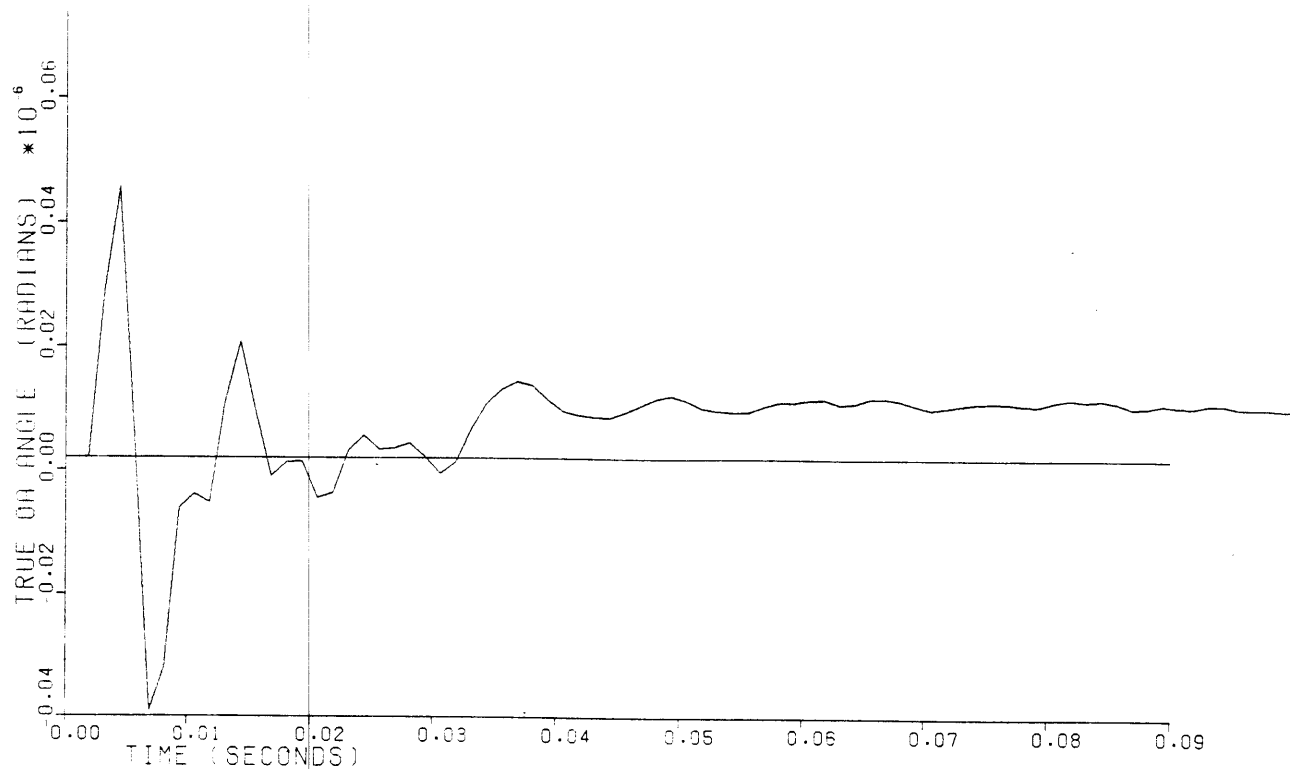


Figure 13-52. Run #18 24 Pt. Average Position vs. Time (282°/hr/s Ramp Input)

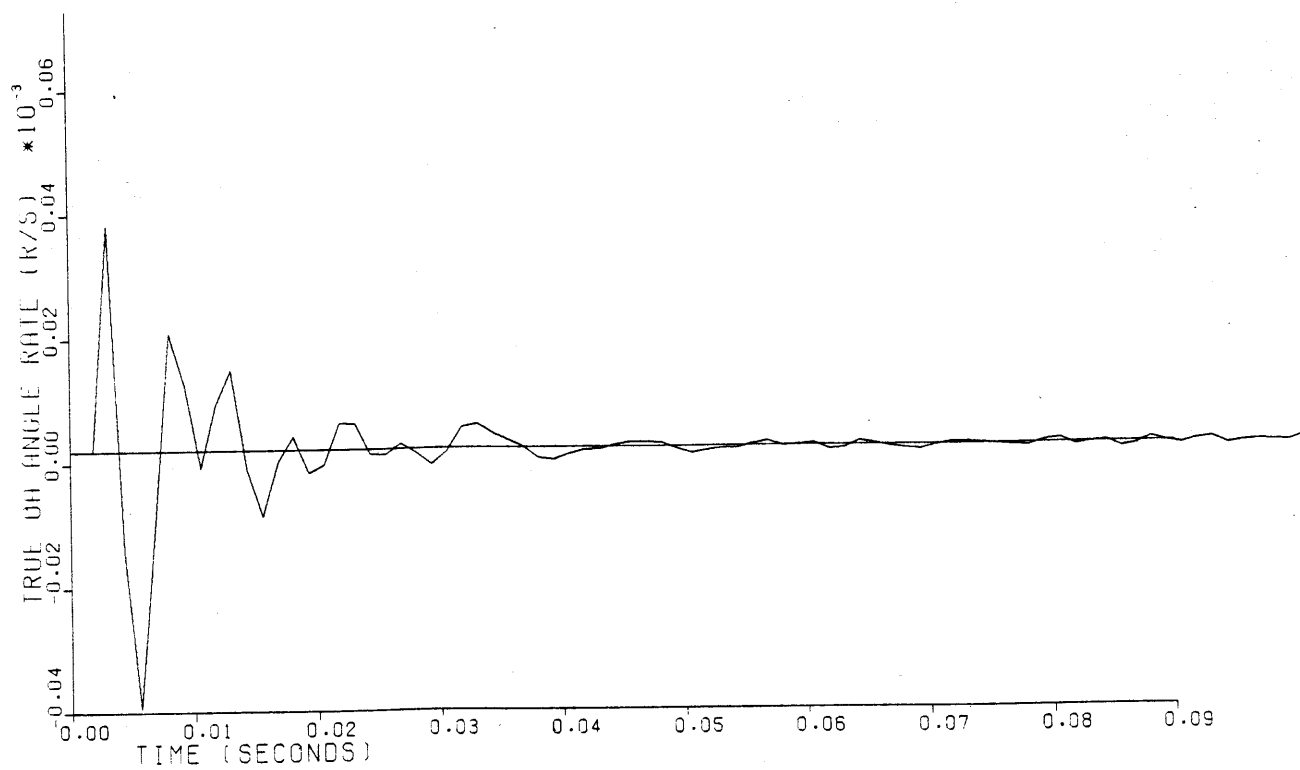


Figure 13-53. Run #18 24 Pt. Average Velocity vs. Time (282°/hr/s Ramp Input)

14. SINUSOIDAL INPUT RESPONSE

14.1 Results Summary

Table 14-1 shows the summary of results for a sinusoidal input response.

Table 14-1. Results Summary for Sinusoidal Input; Freq. = 10 Hz,
Amp = $11.83^\circ/\text{hr}$ = 5.99 dyne-cm

RUN #	Description	Max. avg. float deviation from null	Max. position estimator error (nrad)	Max. velocity estimator error (rad/s)	Max. external input estimator error (dyn-cm)
19	4 st. controller fast gains	1.94	0.002	1.8×10^{-8}	0.0085
20	4 st. controller medium gains	3.06	0.085	3.8×10^{-7}	0.073
21	4 st. controller slow gains	7.48	1.3	1.7×10^{-6}	0.17
22	3 st. controller fast gains	7.11	0.027	2.9×10^{-7}	0.145
23	3 st controller medium gains	16.2	0.67	2.9×10^{-6}	0.52
24	3 st. controller slow gains	26.9	5.1	7.2×10^{-6}	0.68
25	simplified controller	24.4	-	-	-

All controllers maintained the minimum mode of float oscillation at the torquing frequency and minimum quantization in the torquing pattern. As the external inputs varied,

the steady state amplitudes of float oscillation were properly adjusted to ensure 800 Hz oscillations. In addition to the steady state float oscillation, there was a slower oscillation of the average float position at the 10 Hz frequency of the varying external input. The sinusoidal input was an excellent test of the estimator's ability to track varying inputs where both the input rate and rate of change of the input rate varied.

The four state controller with the added state $\dot{\tau}$ performed significantly better in the face of a sinusoidal input. Comparing all sets of gains, the four state controller reduced the maximum average float offset of the three state controller by about a factor of 4. The estimation errors were improved by factors of about 15.5 for the fast gains, 7.5 for the medium gains, and 4.0 for the slow gains. In addition, as expected each controller experienced a reduction in the amplitude of estimation errors with an increase of the estimator gains. The simplified controller performed comparably to the three state controller with slow estimator gains.

14.2 Simulation Plots

Figures 14-1 through 14-53 present the plots for the responses to a sinusoidal input.

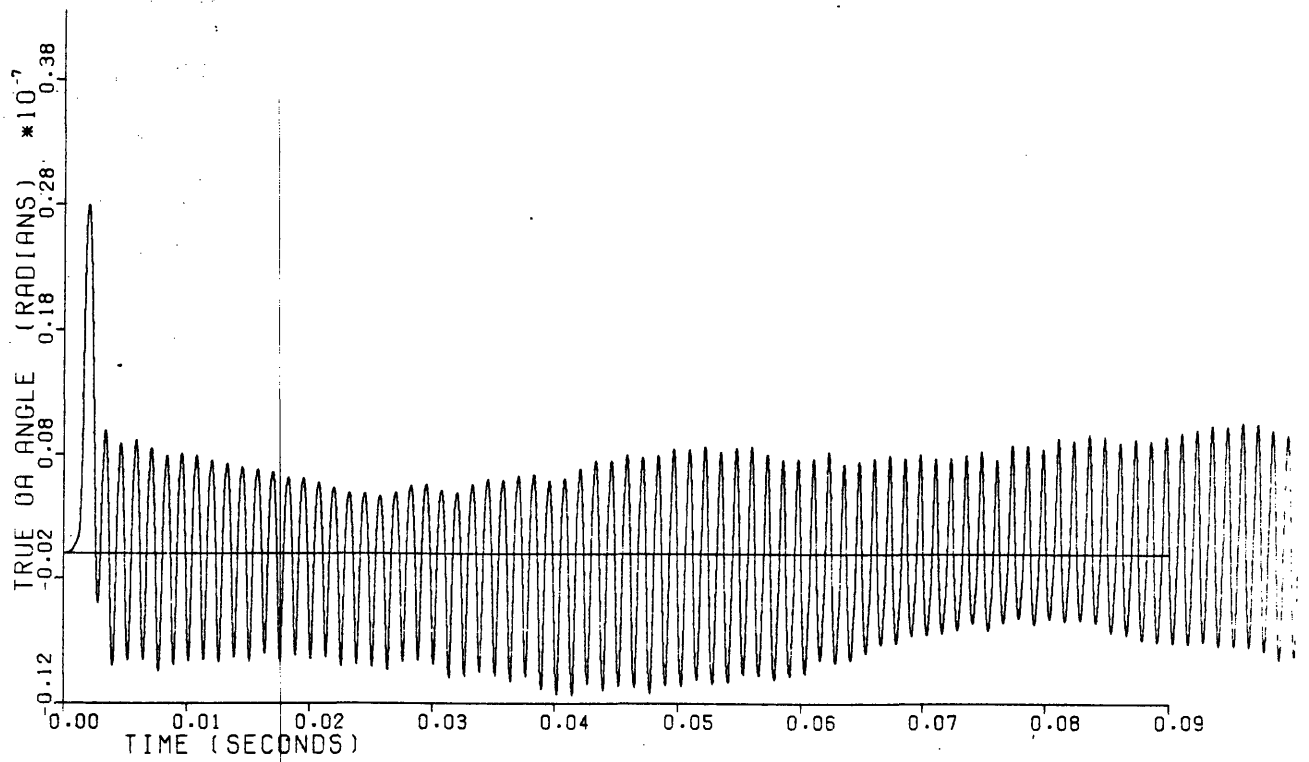


Figure 14-1. Run #19 Float Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

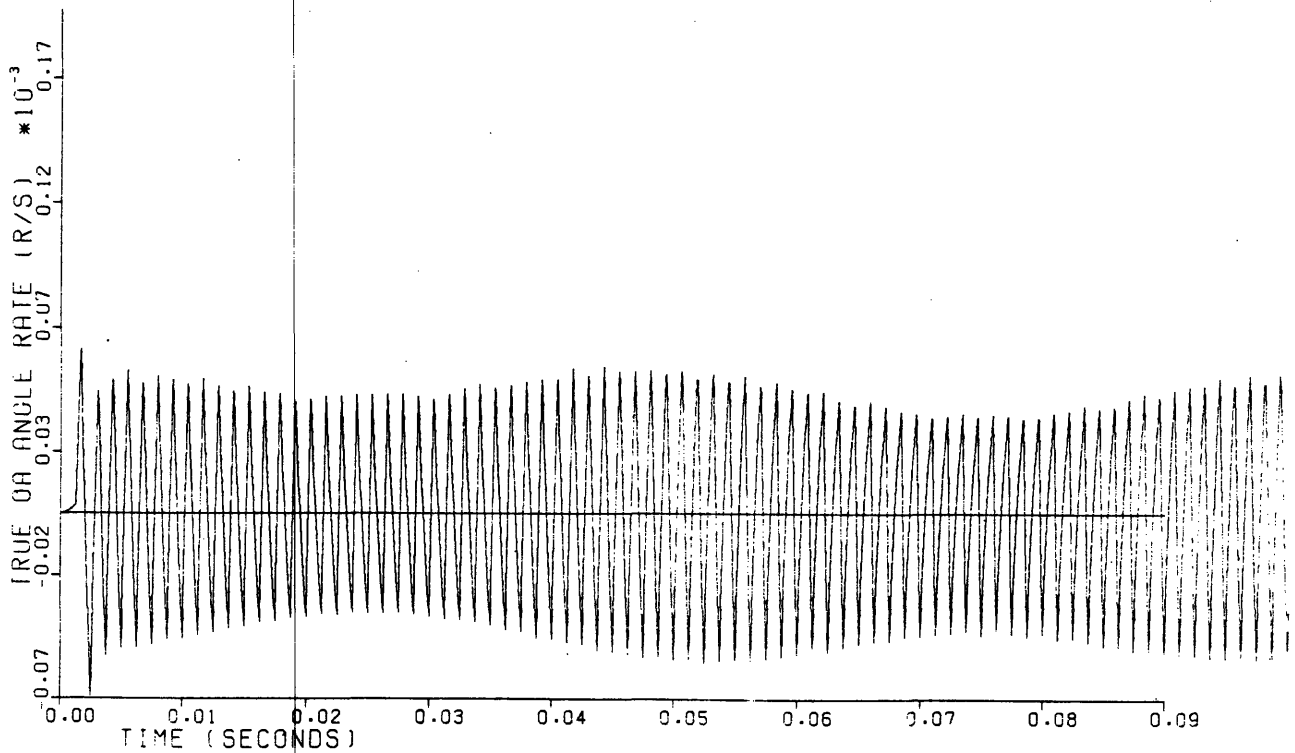


Figure 14-2. Run #19 Float Velocity vs Time (10 Hz, 11.83°/hr Sinusoidal Input)

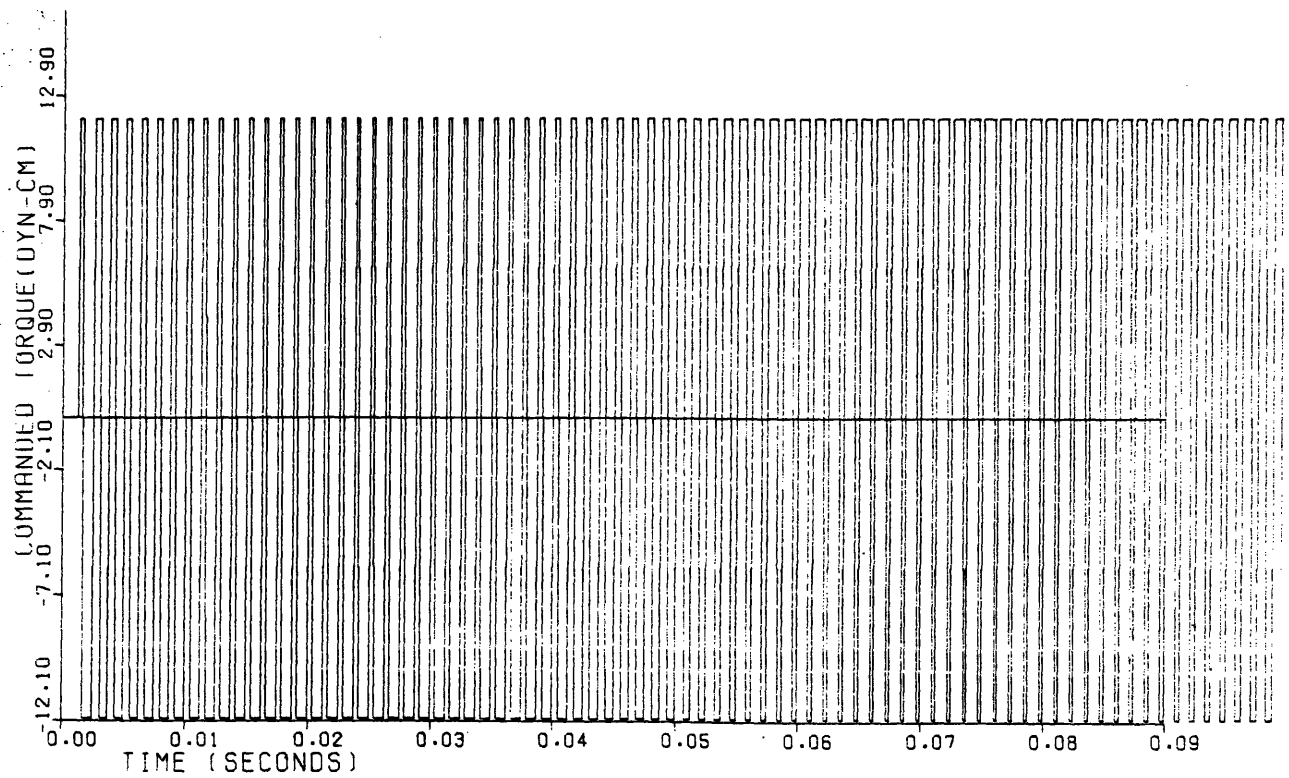


Figure 14-3. Run #19 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

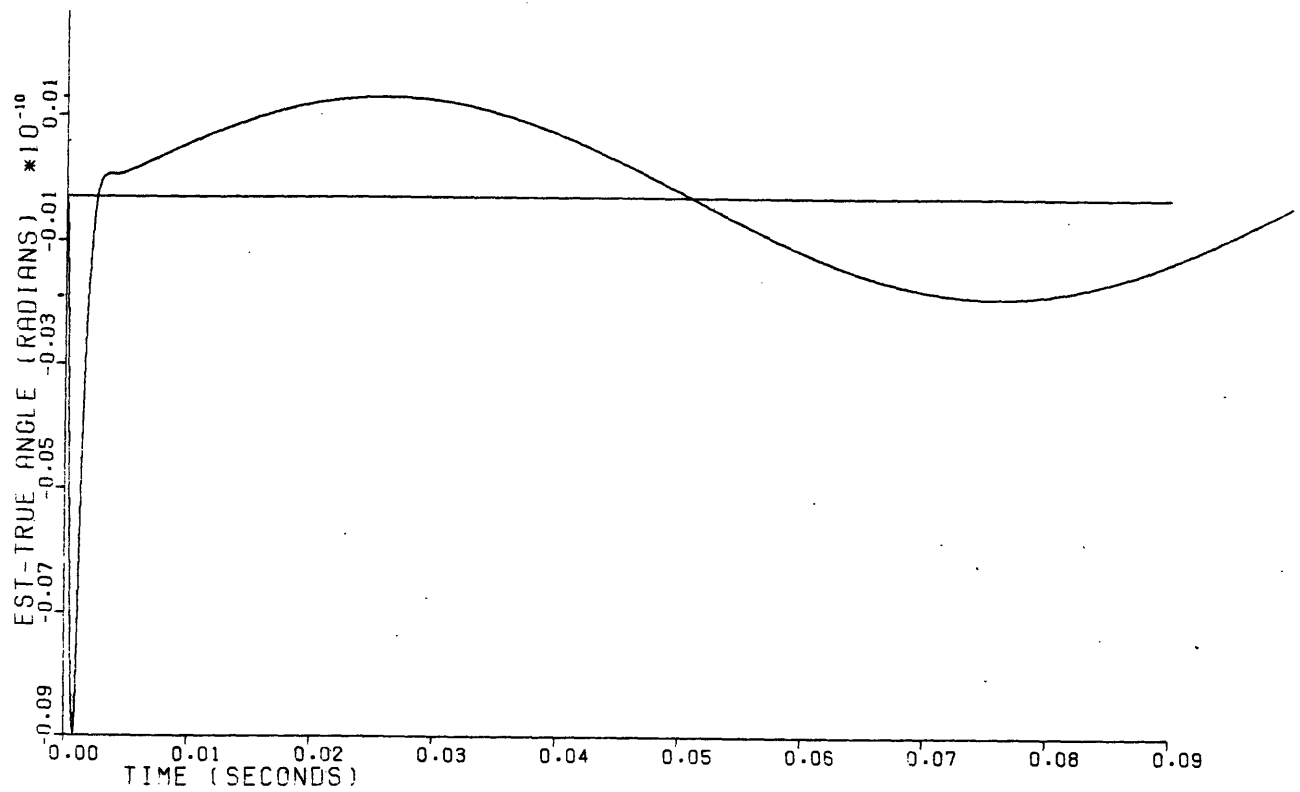


Figure 14-4. Run #19 Position Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

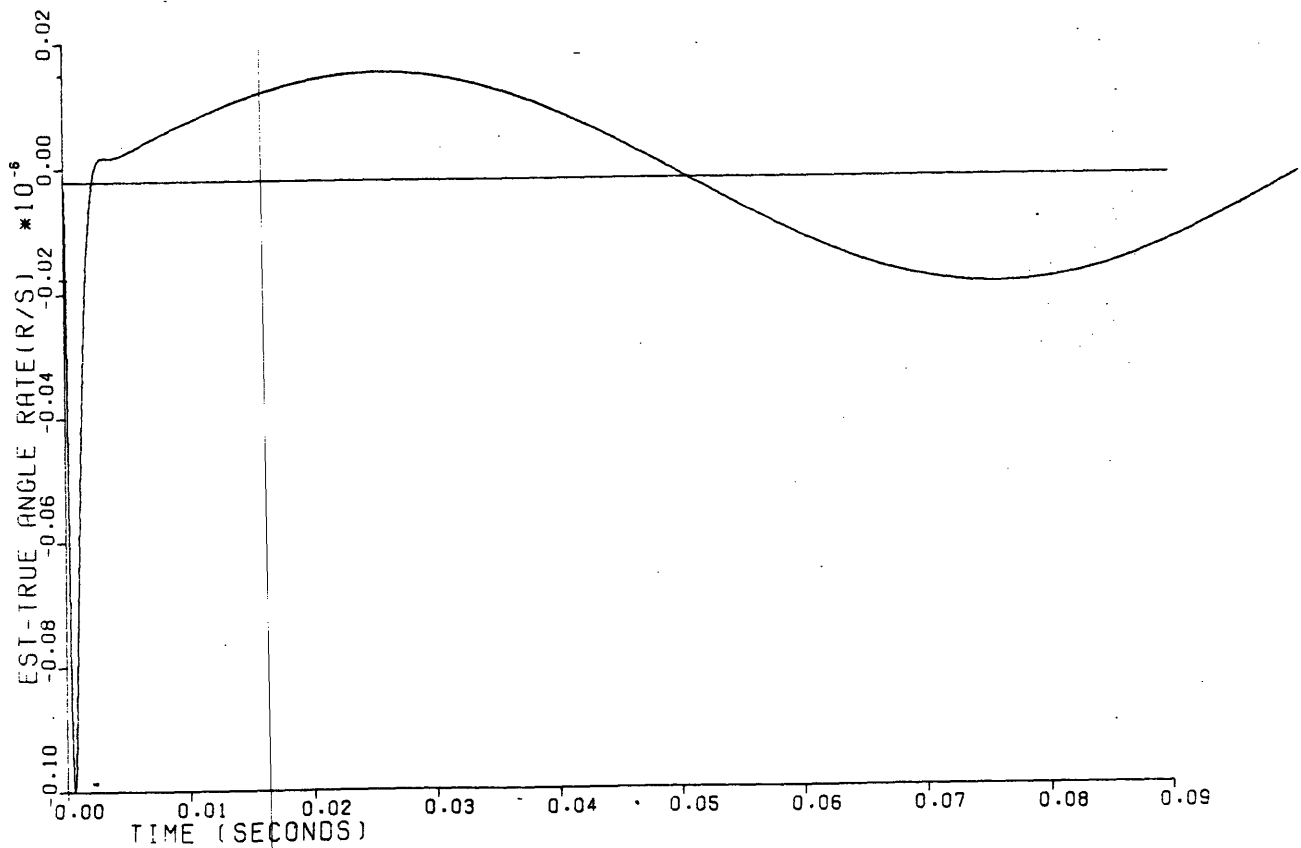


Figure 14-5. Run #19 Velocity Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

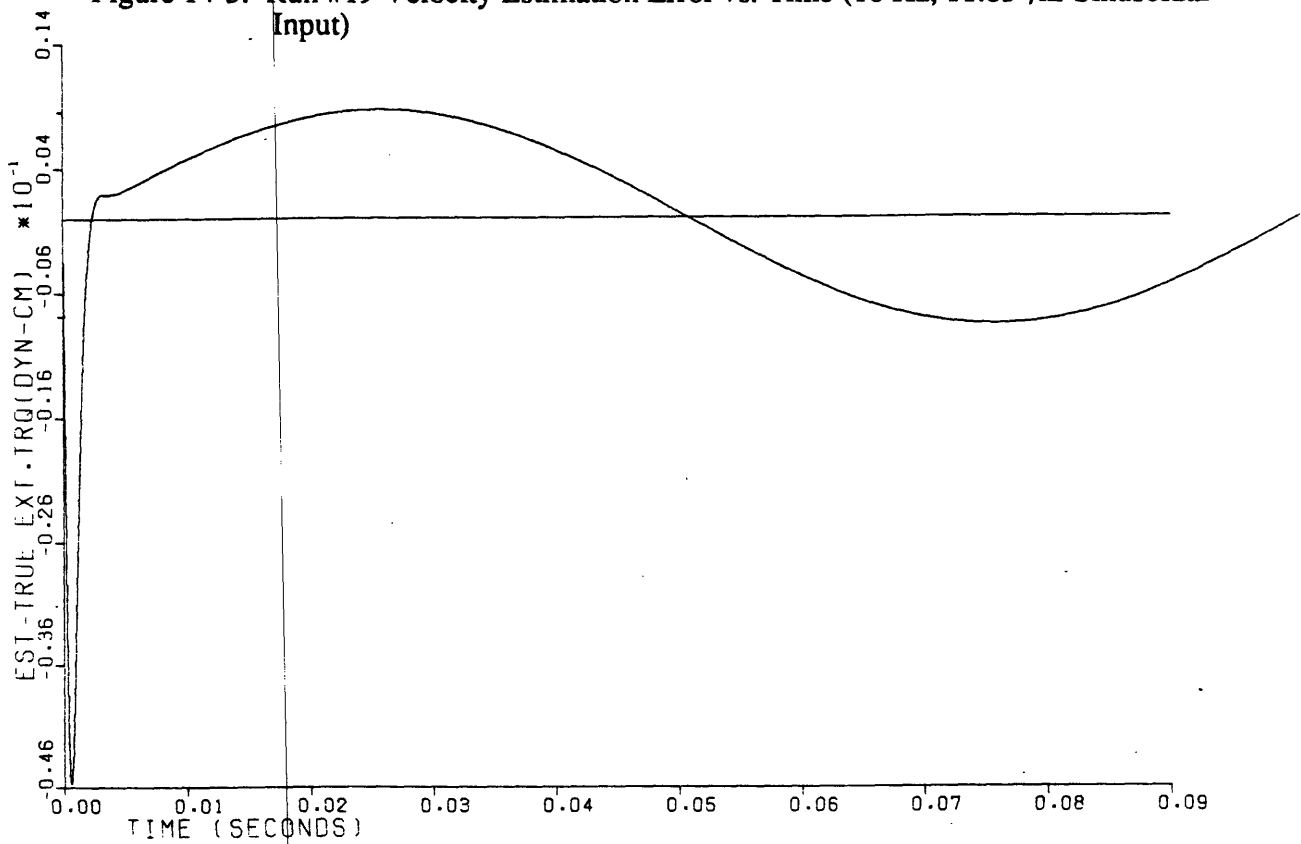


Figure 14-6. Run #19 External Torque Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

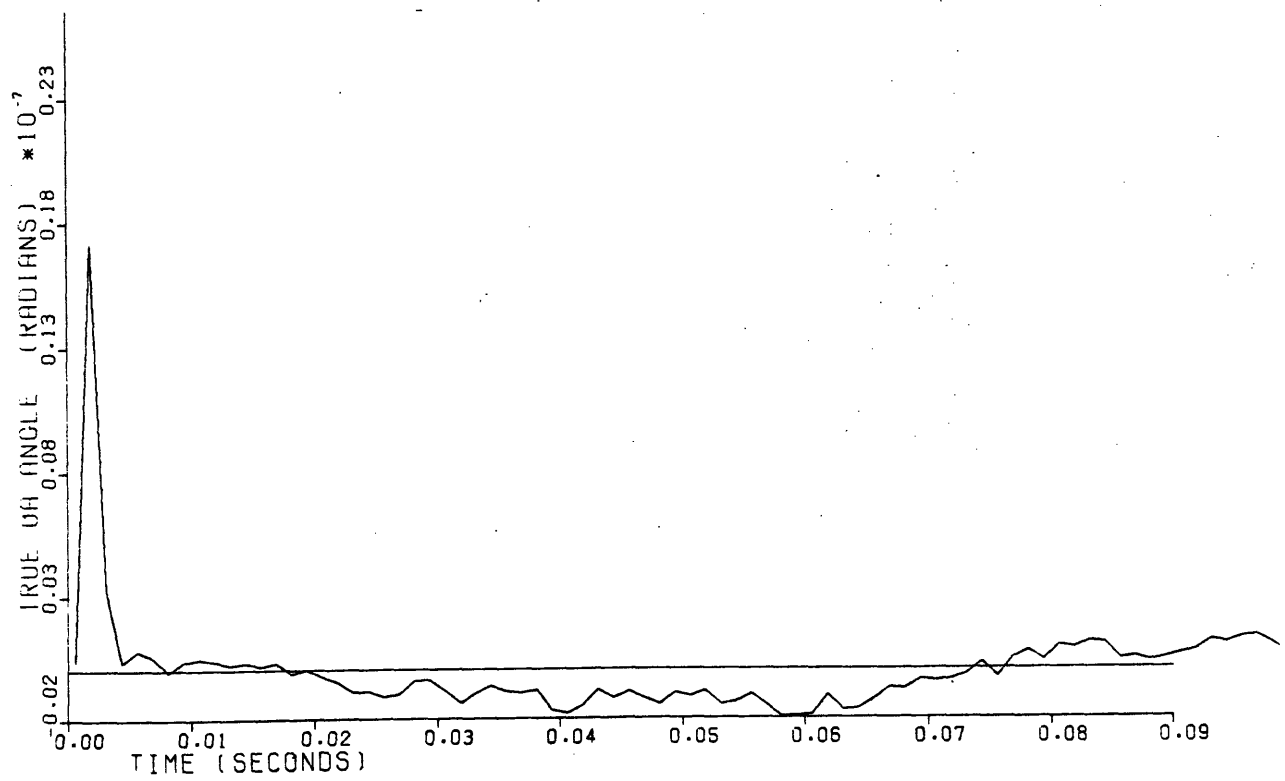


Figure 14-7. Run #19 24 Pt. Average Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

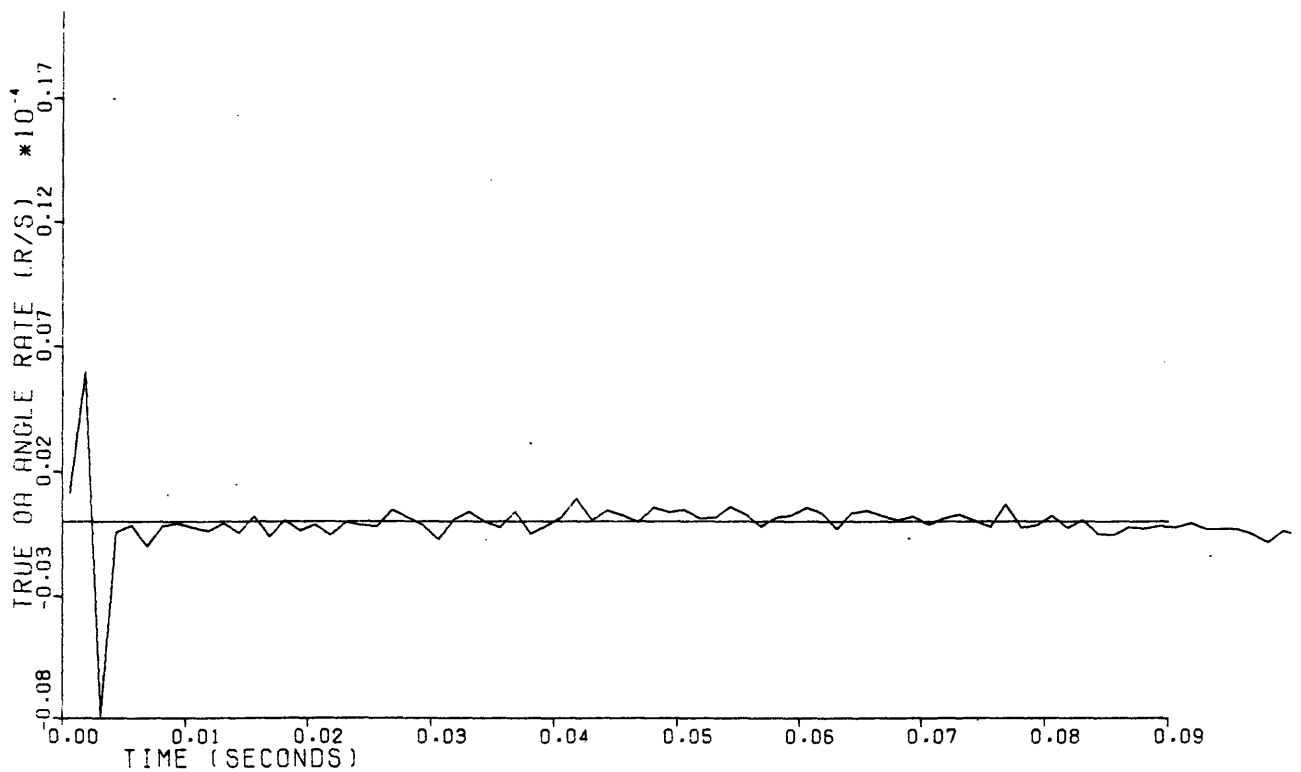


Figure 14-8. Run #19 24 Pt. Average Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

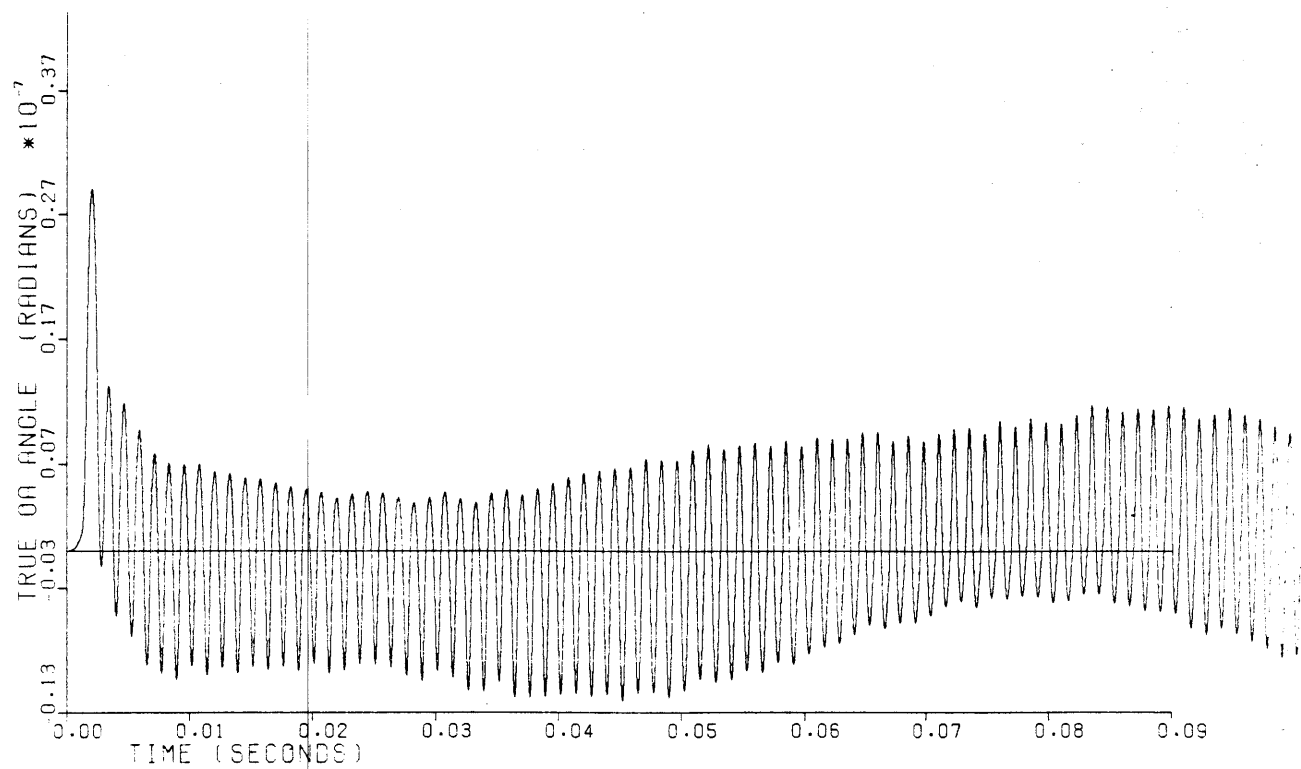


Figure 14-9. Run #20 Float Position vs. Time (10 Hz, $11.83^\circ/\text{hr}$ Sinusoidal Input)

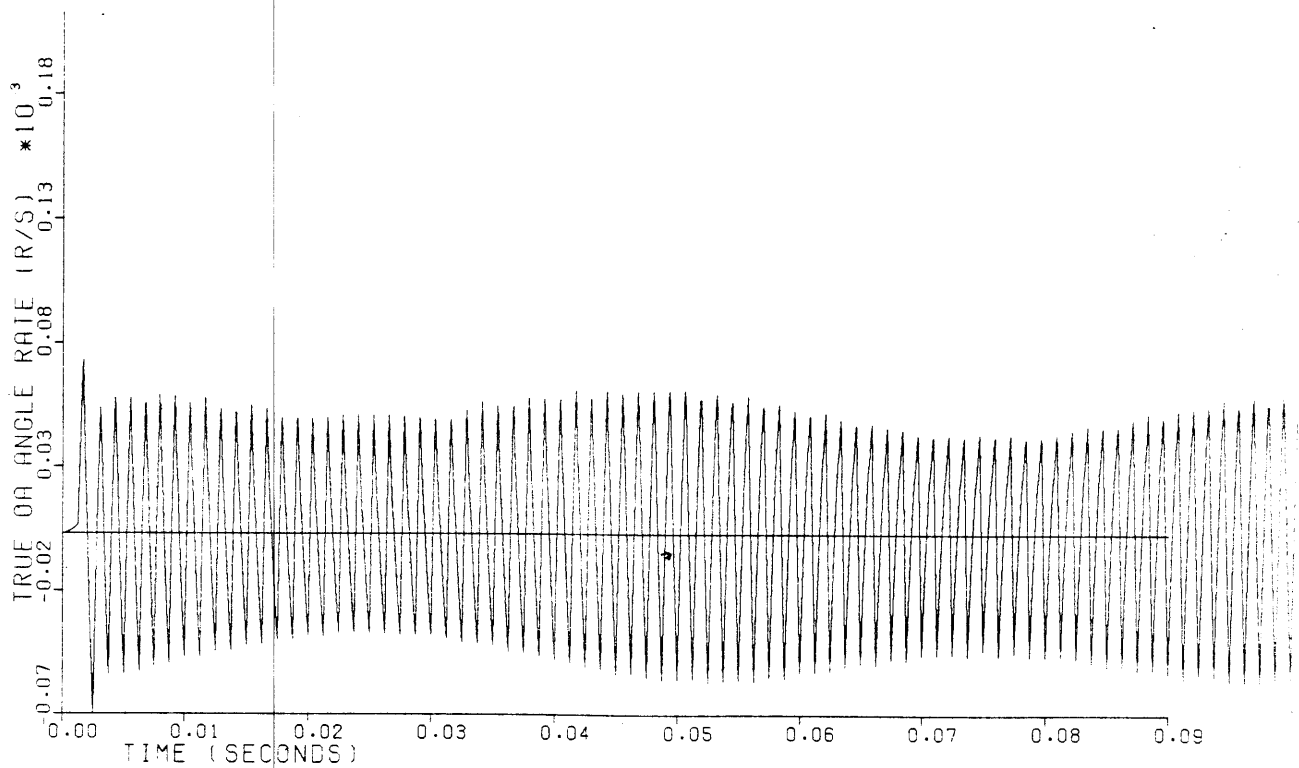


Figure 14-10. Run #20 Float Velocity vs Time (10 Hz, $11.83^\circ/\text{hr}$ Sinusoidal Input)

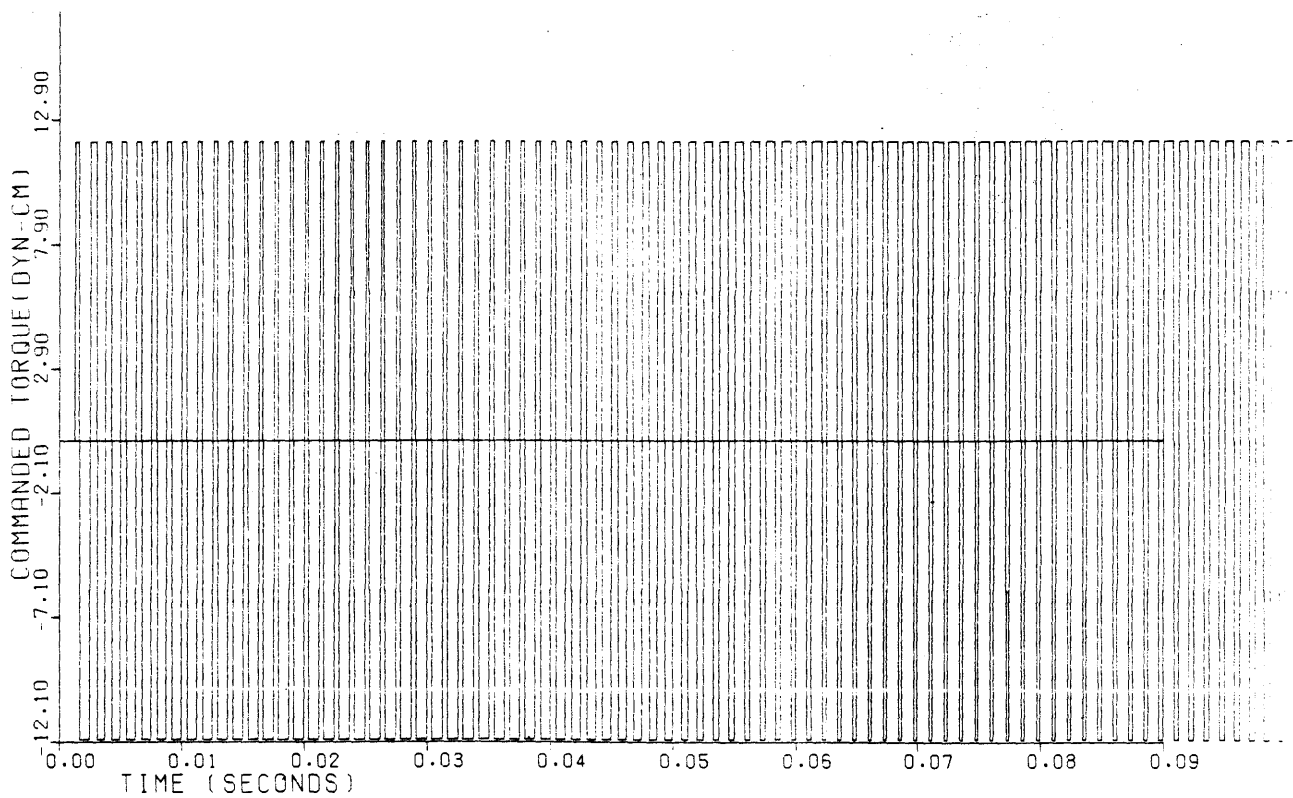


Figure 14-11. Run #20 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

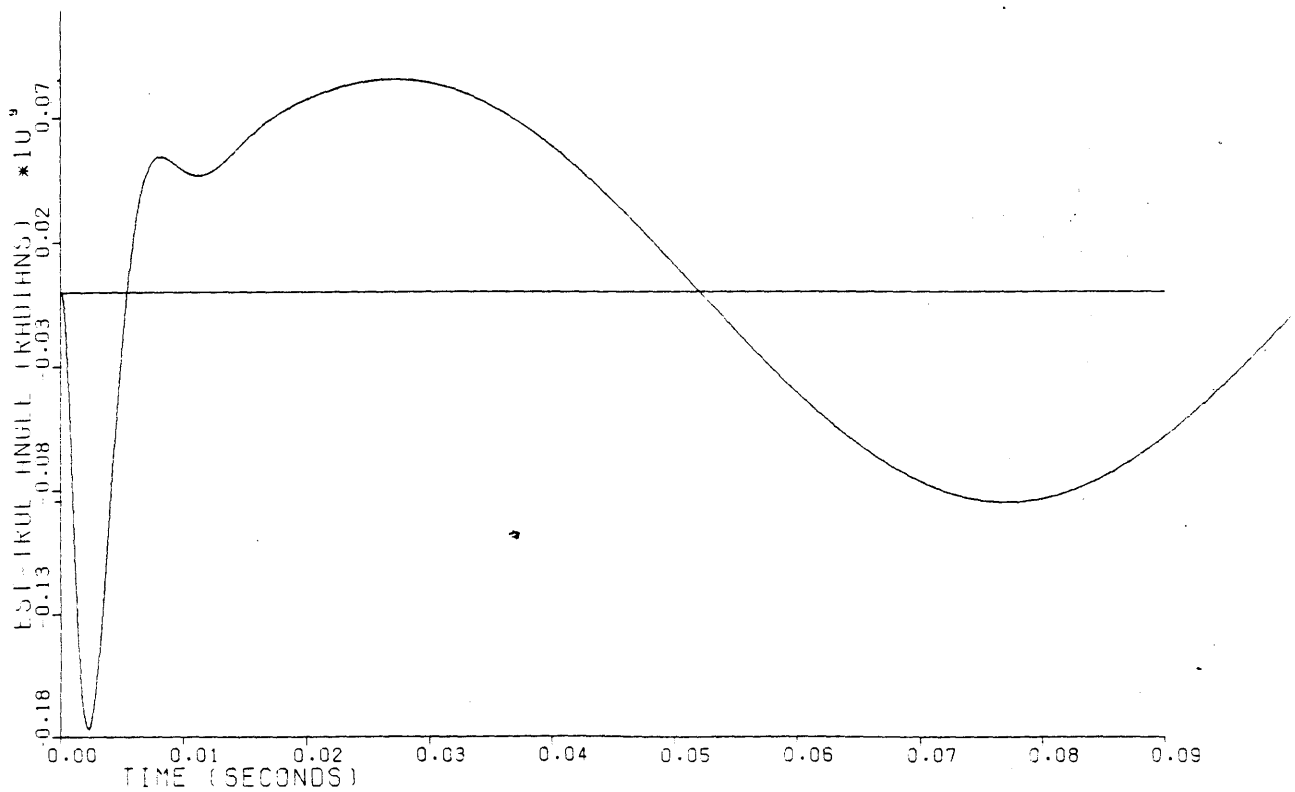


Figure 14-12. Run #20 Position Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

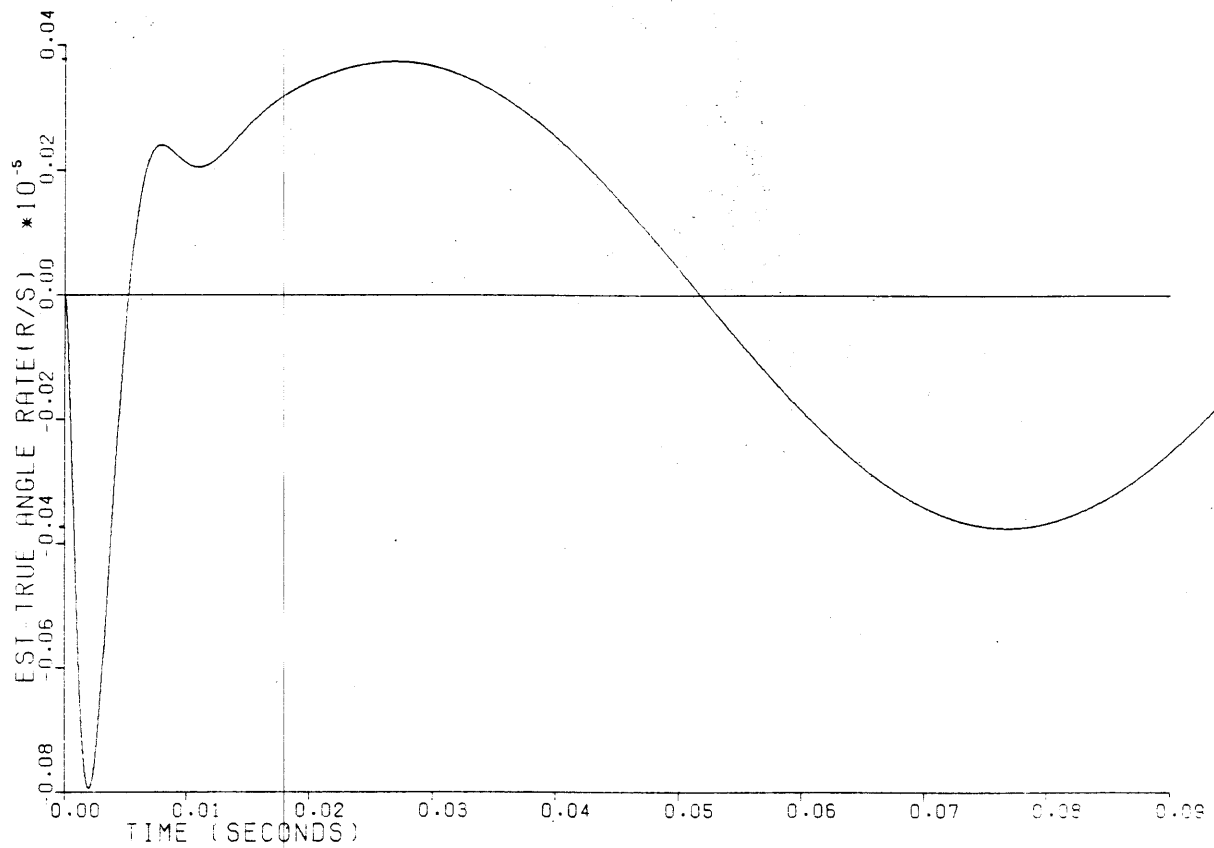


Figure 14-13. Run #20 Velocity Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

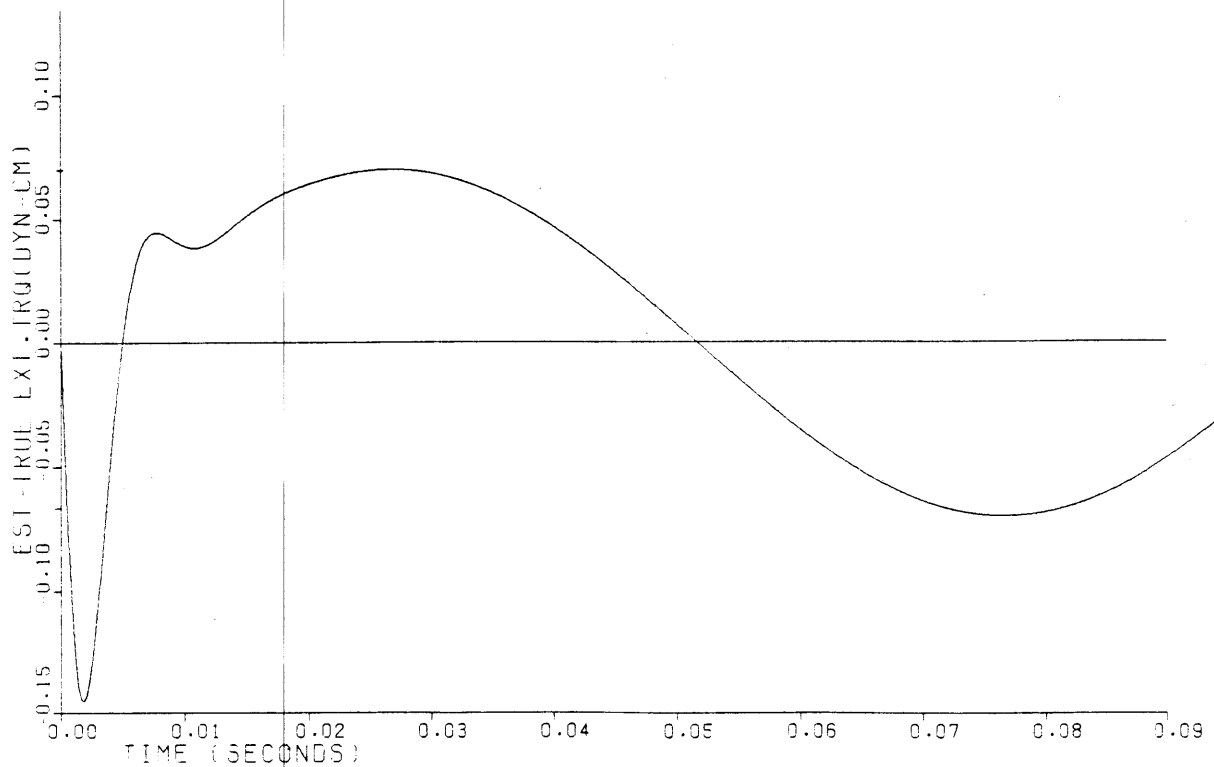


Figure 14-14. Run #20 External Torque Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

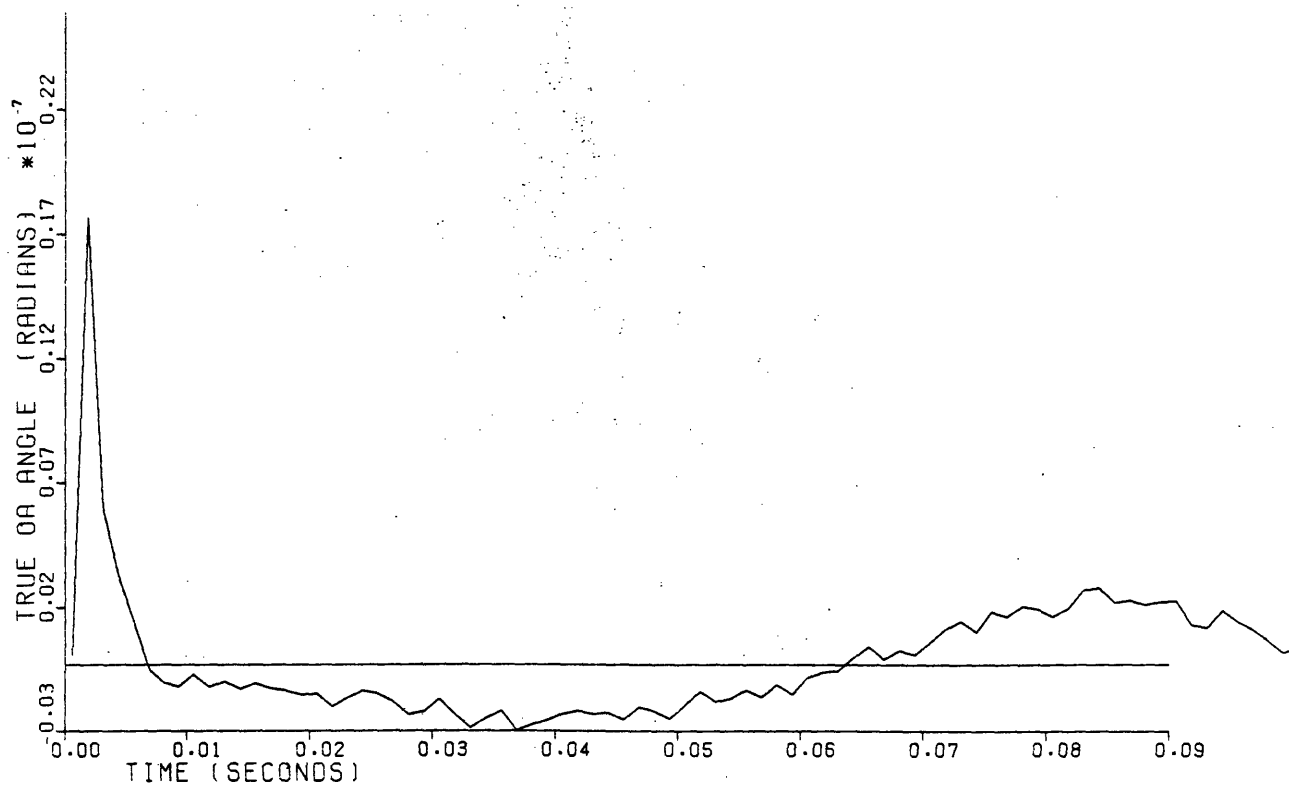


Figure 14-15. Run #20 24 Pt. Average Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

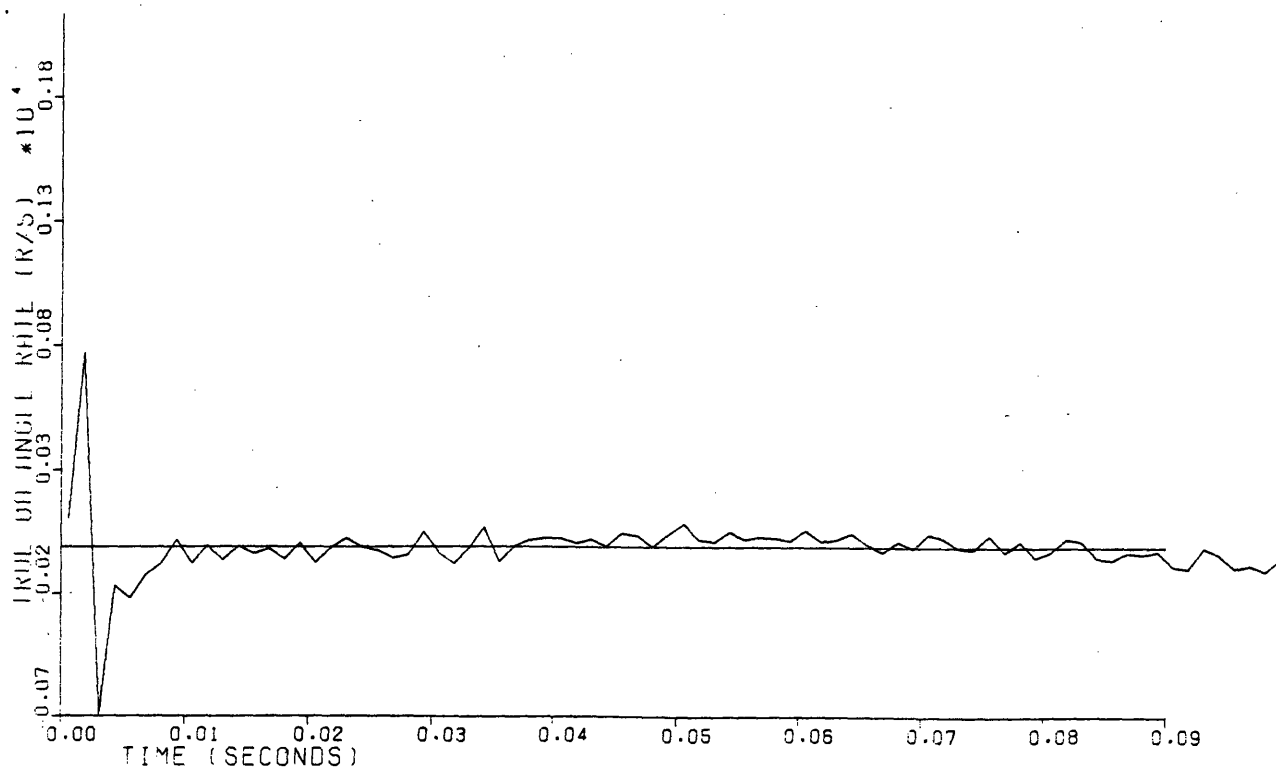


Figure 14-16. Run #20 24 Pt. Average Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

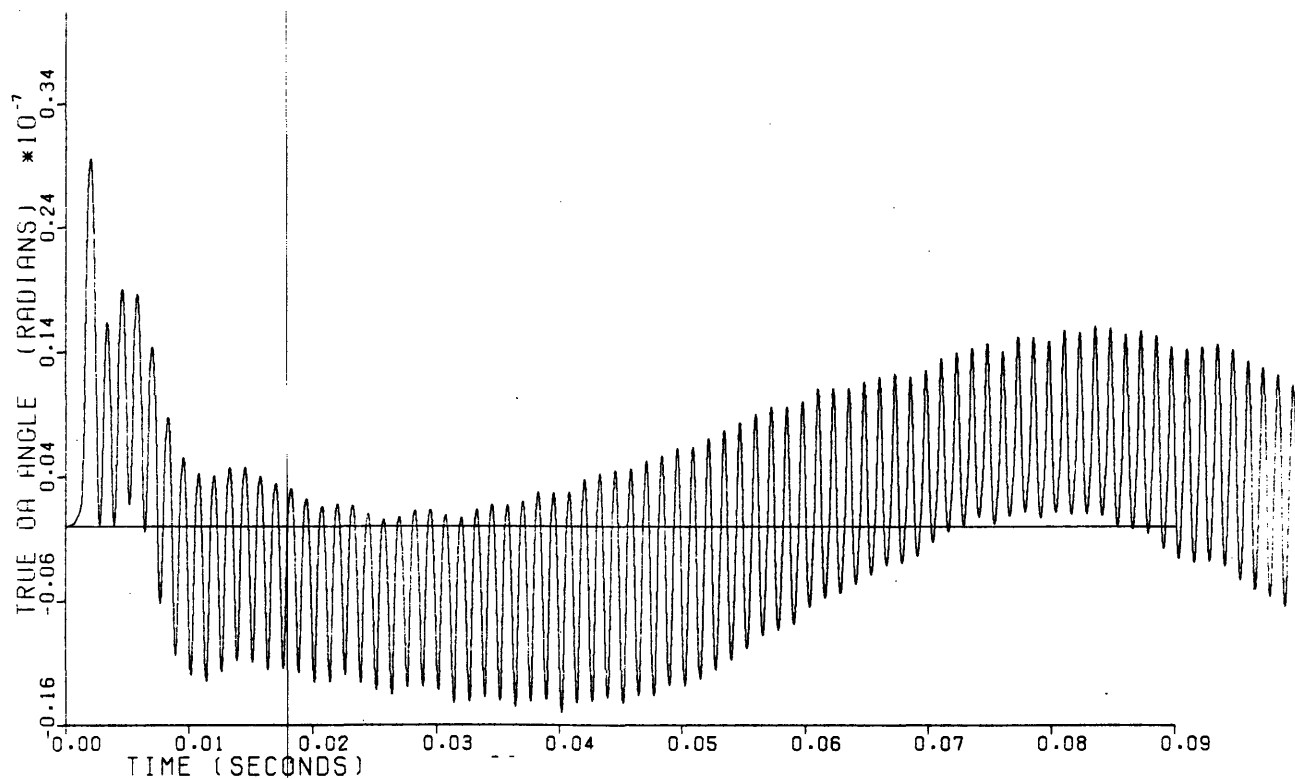


Figure 14-17. Run #21 Float Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

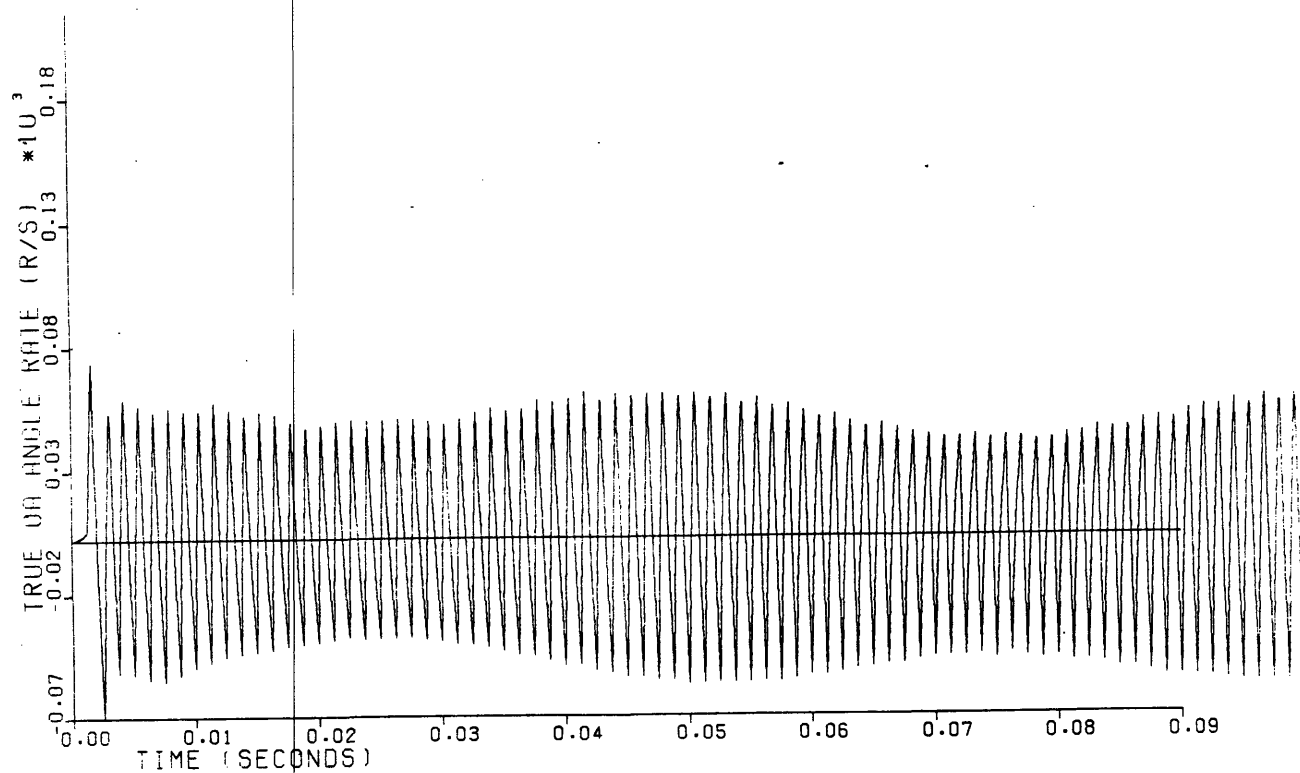


Figure 14-18. Run #21 Float Velocity vs Time (10 Hz, 11.83°/hr Sinusoidal Input)

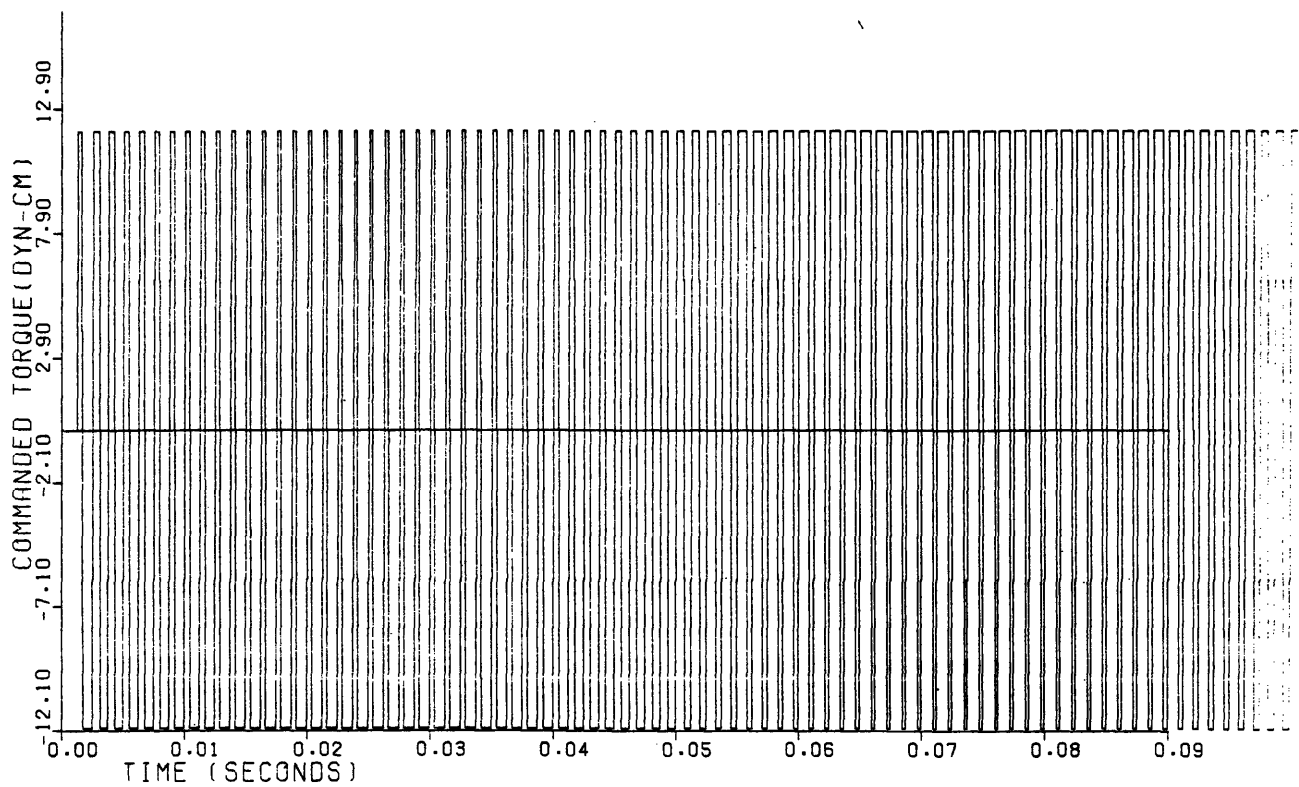


Figure 14-19. Run #21 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

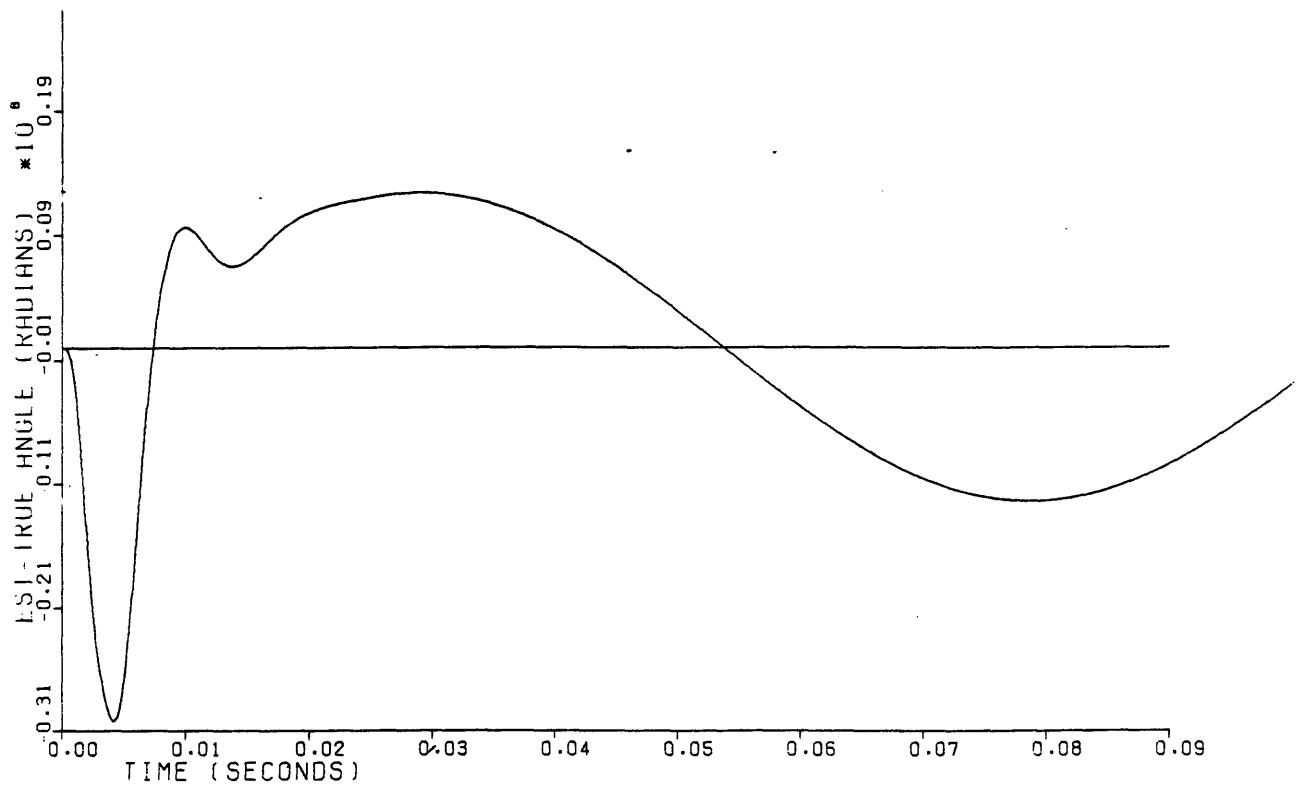


Figure 14-20. Run #21 Position Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

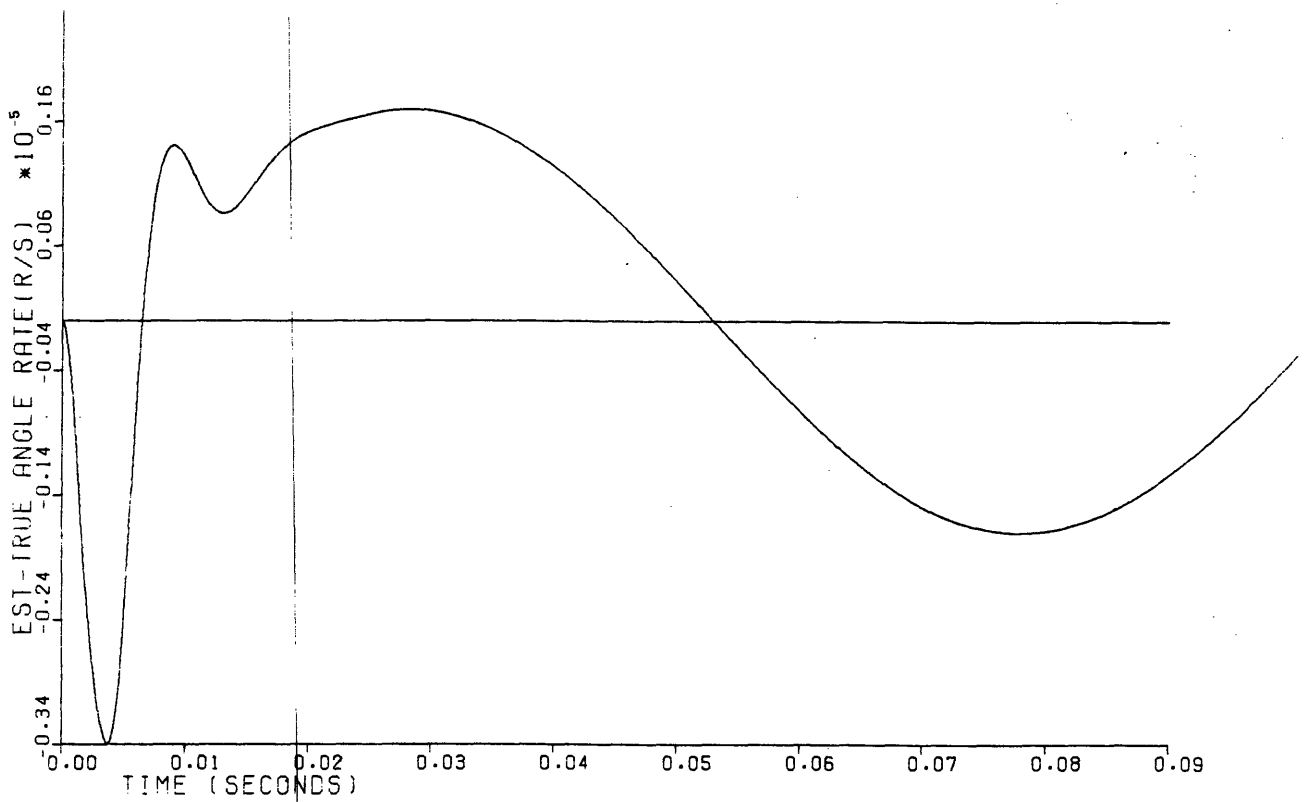


Figure 14-21. Run #21 Velocity Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

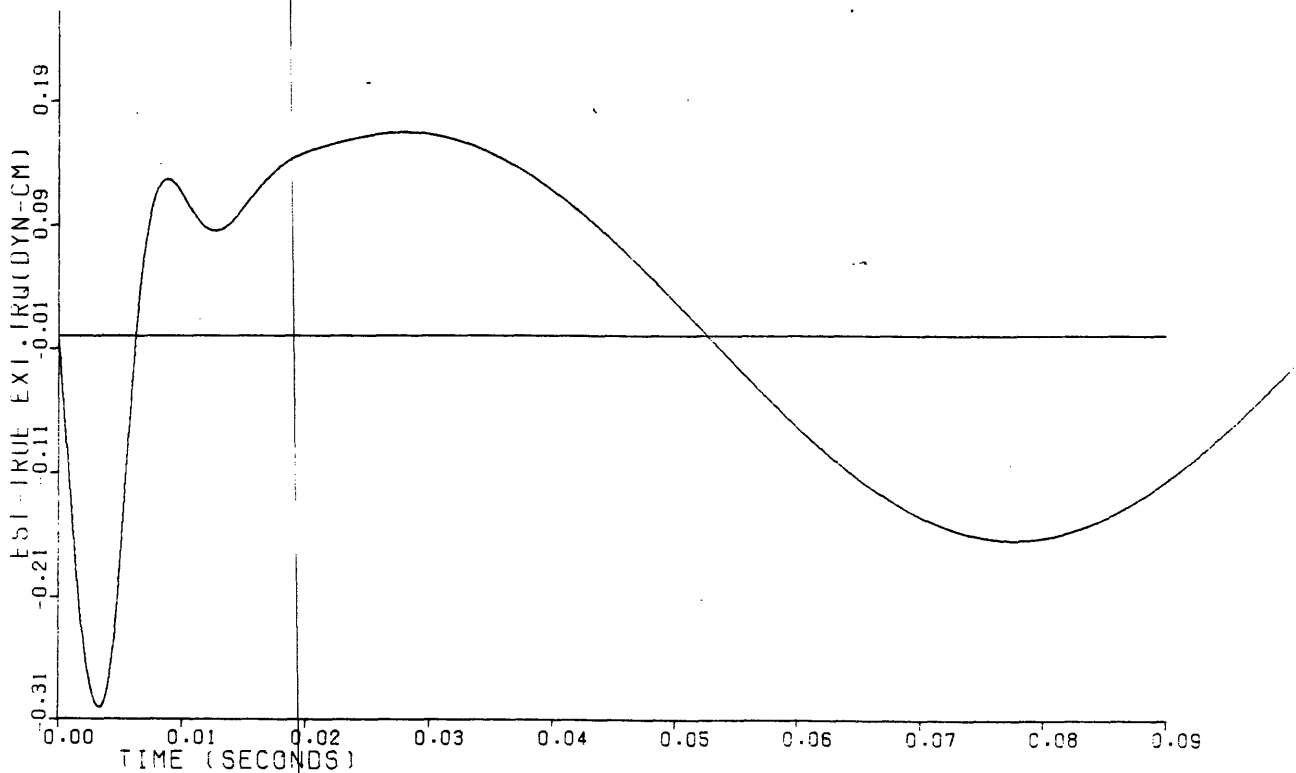


Figure 14-22. Run #21 External Torque Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

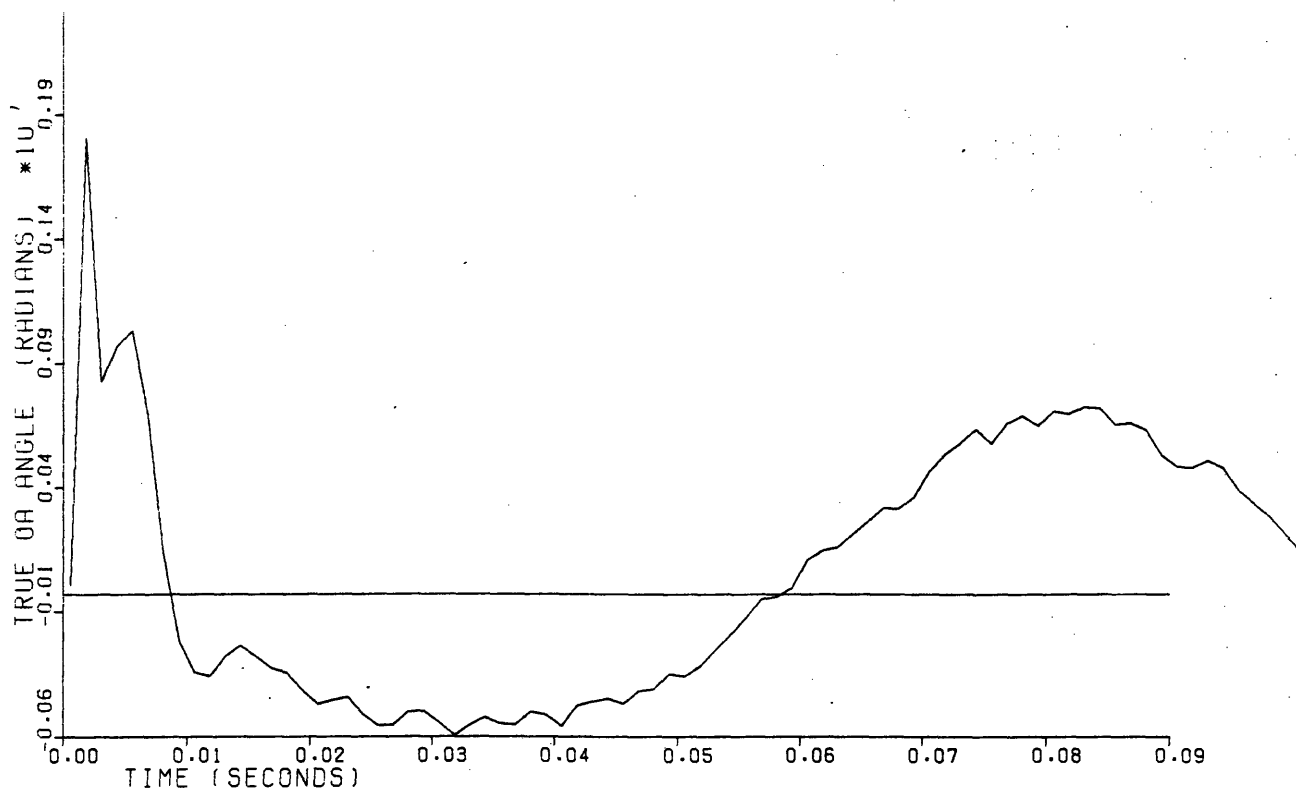


Figure 14-23. Run #21 24 Pt. Average Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

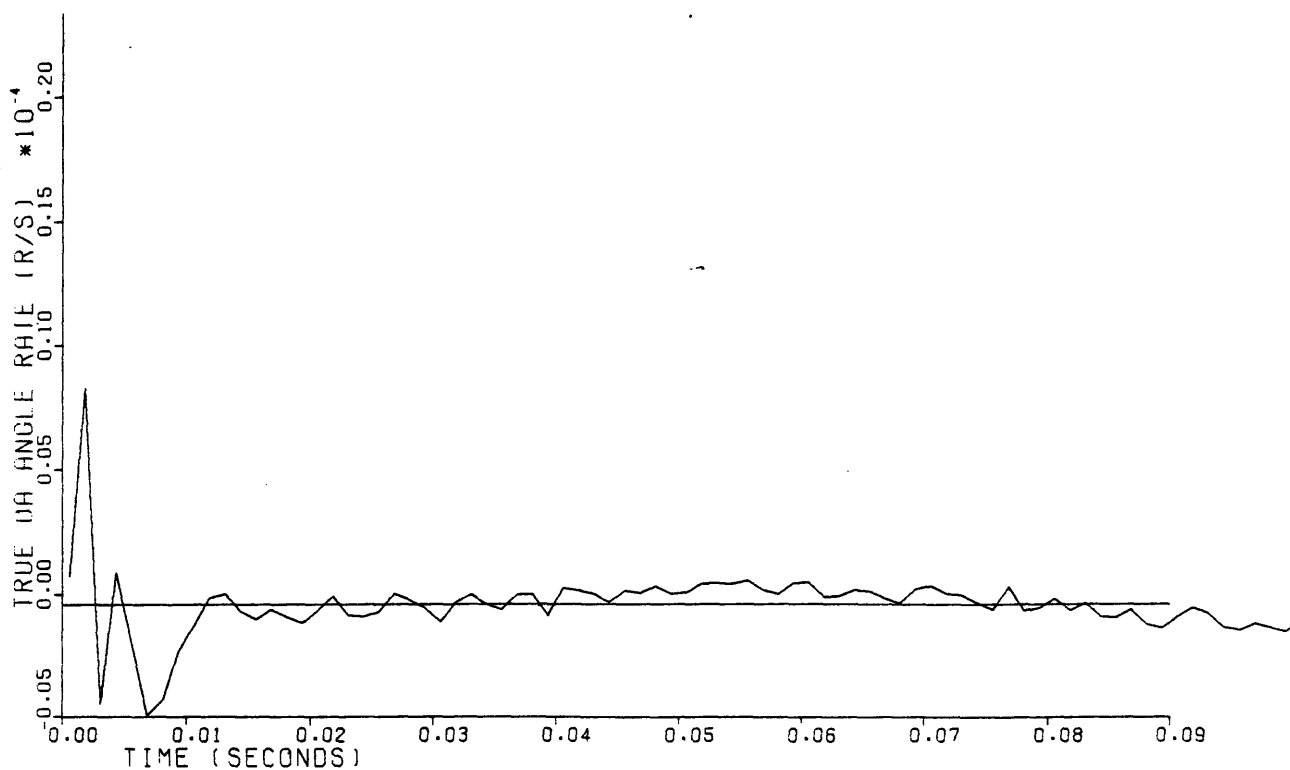


Figure 14-24. Run #21 24 Pt. Average Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

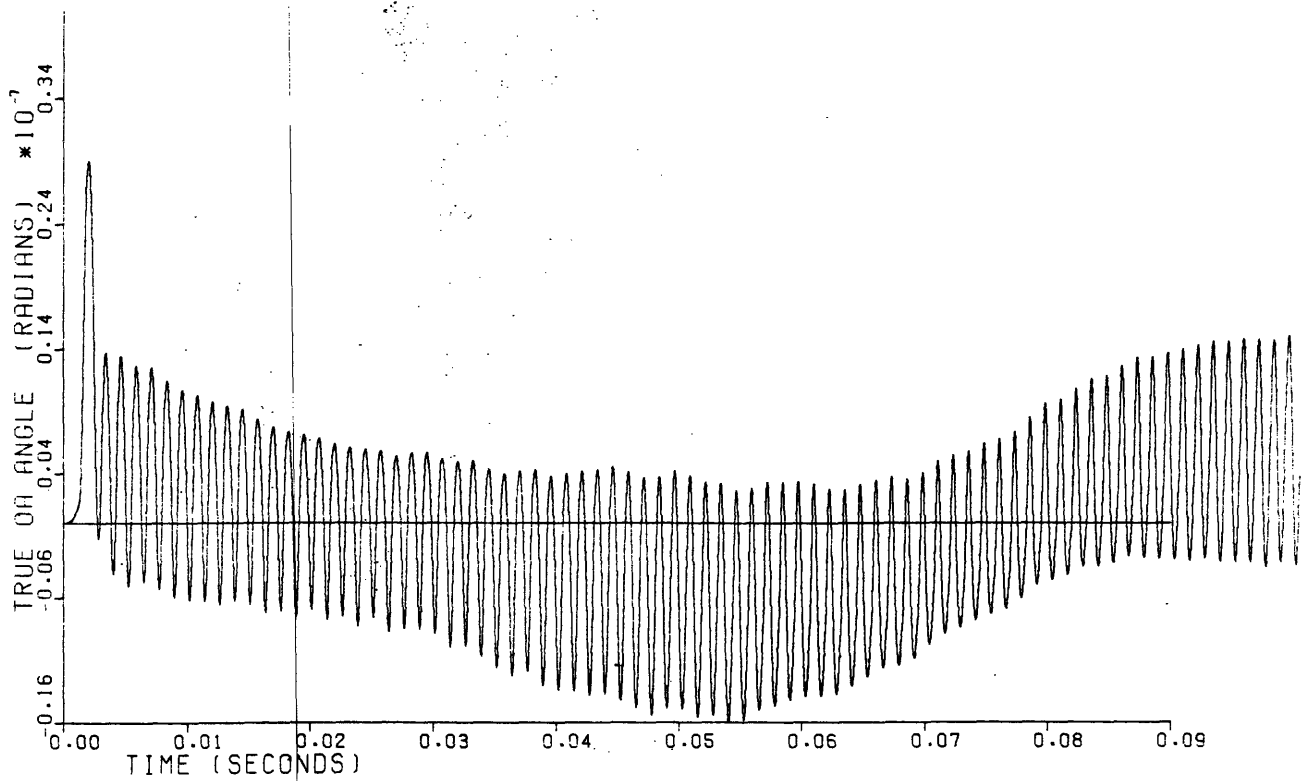


Figure 14-25. Run #22 Float Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

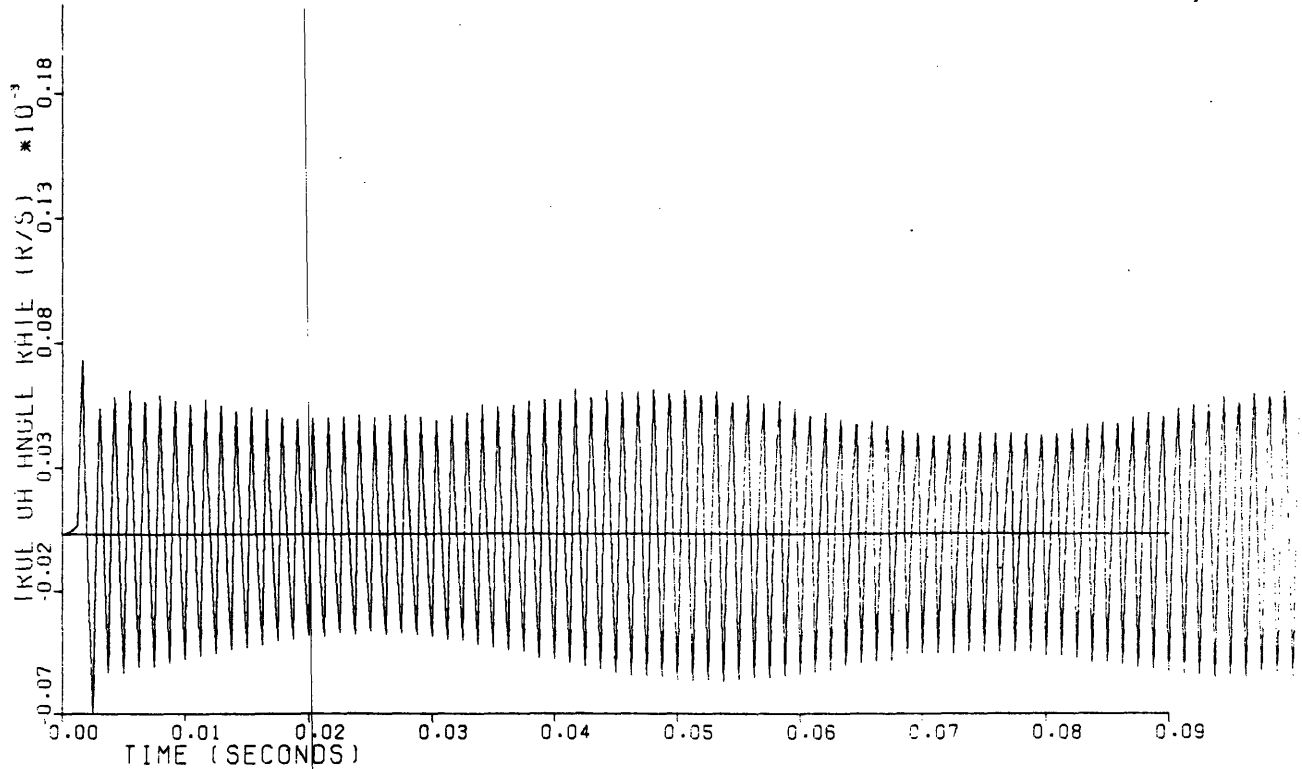


Figure 14-26. Run #22 Float Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

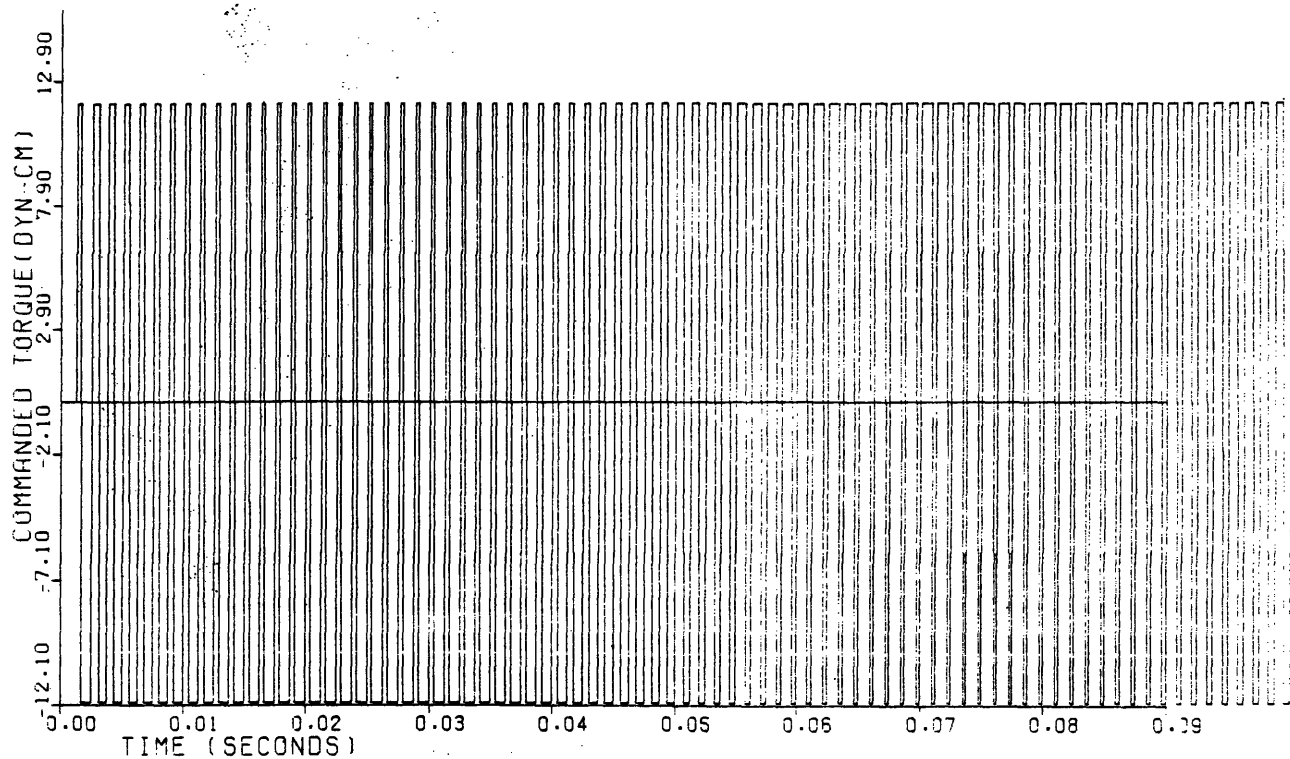


Figure 14-27. Run #22 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

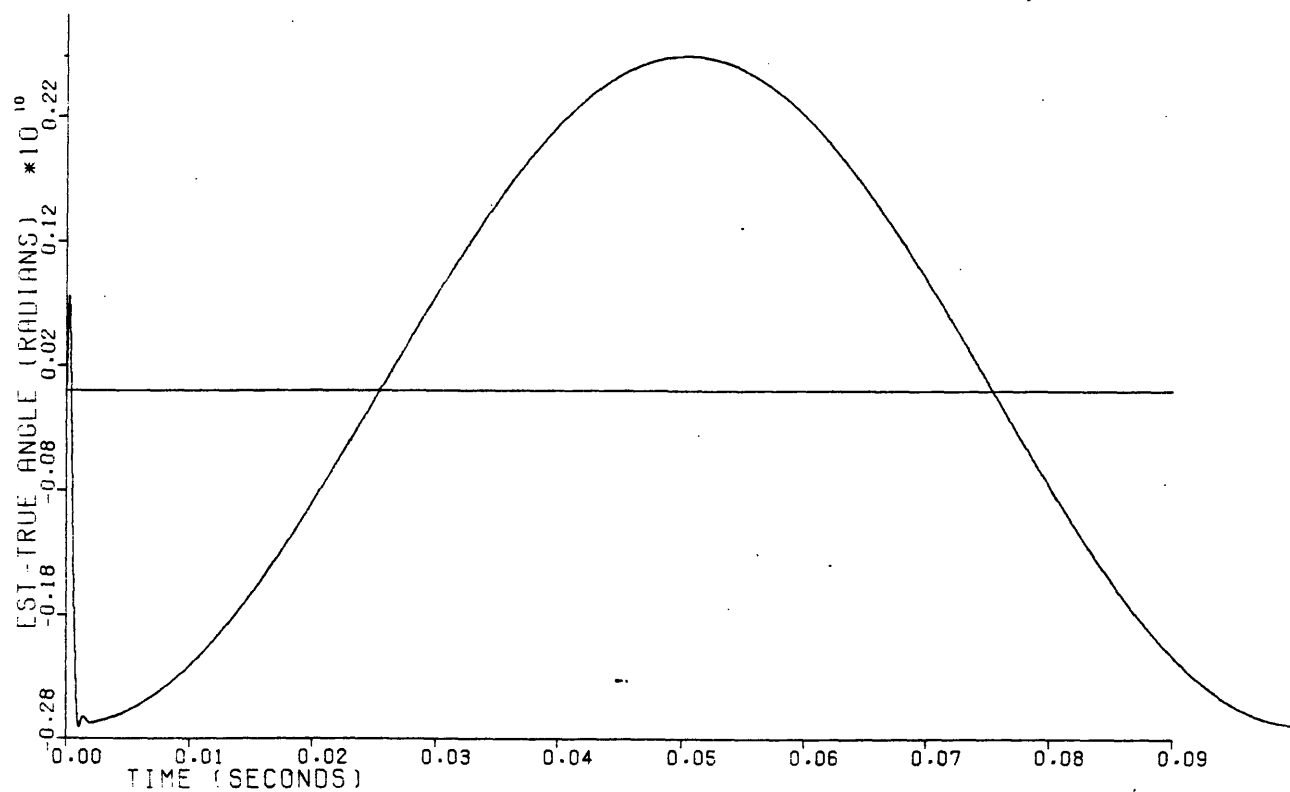


Figure 14-28. Run #22 Position Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

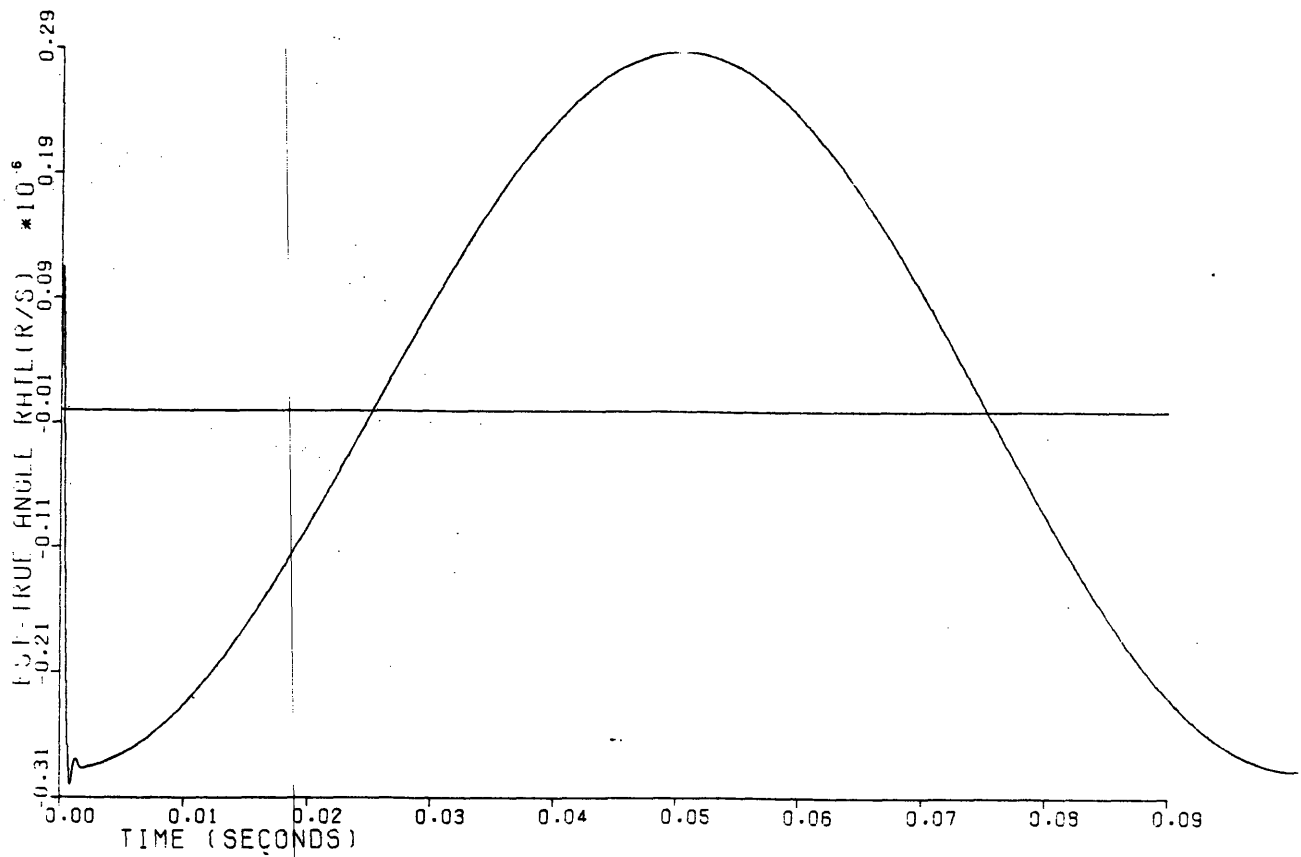


Figure 14-29. Run #22 Velocity Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

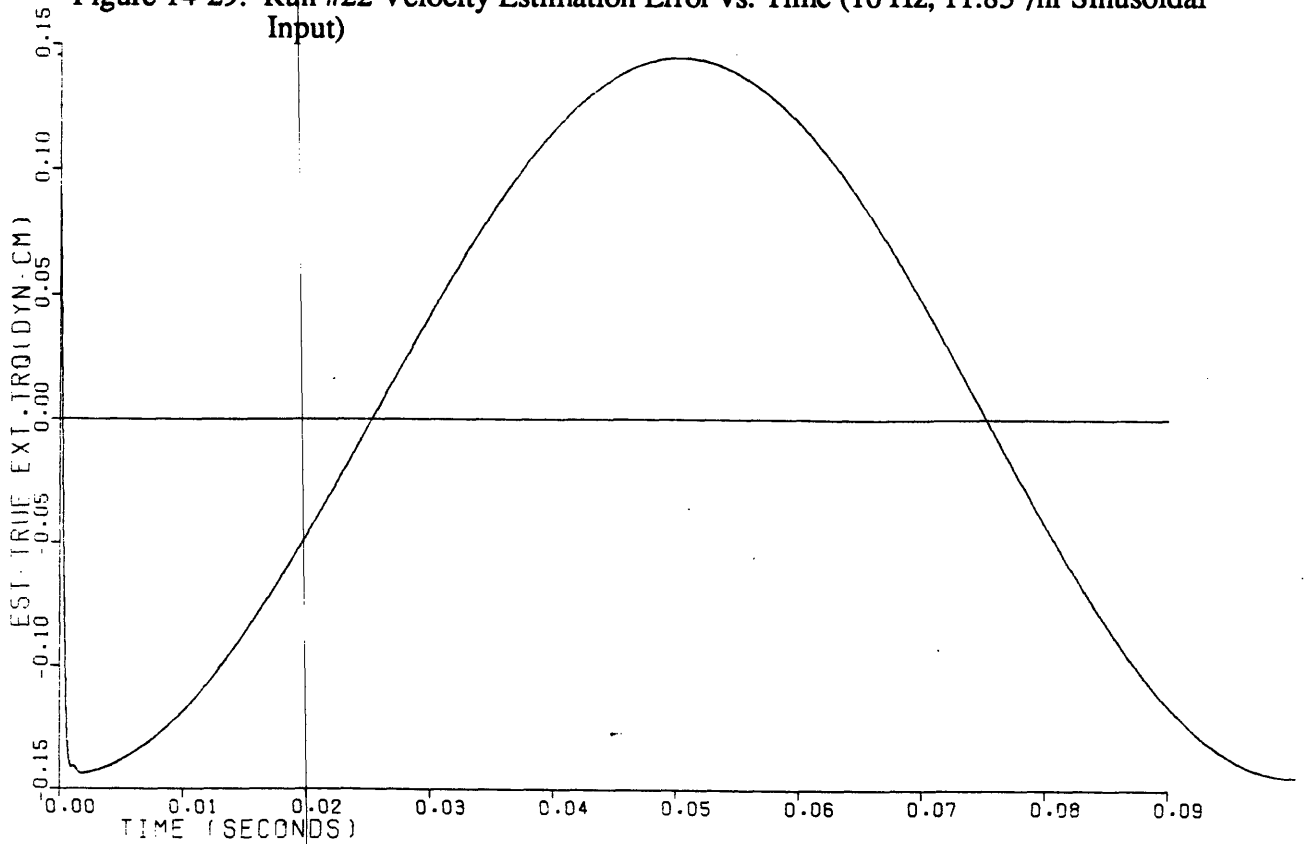


Figure 14-30. Run #22 External Torque Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

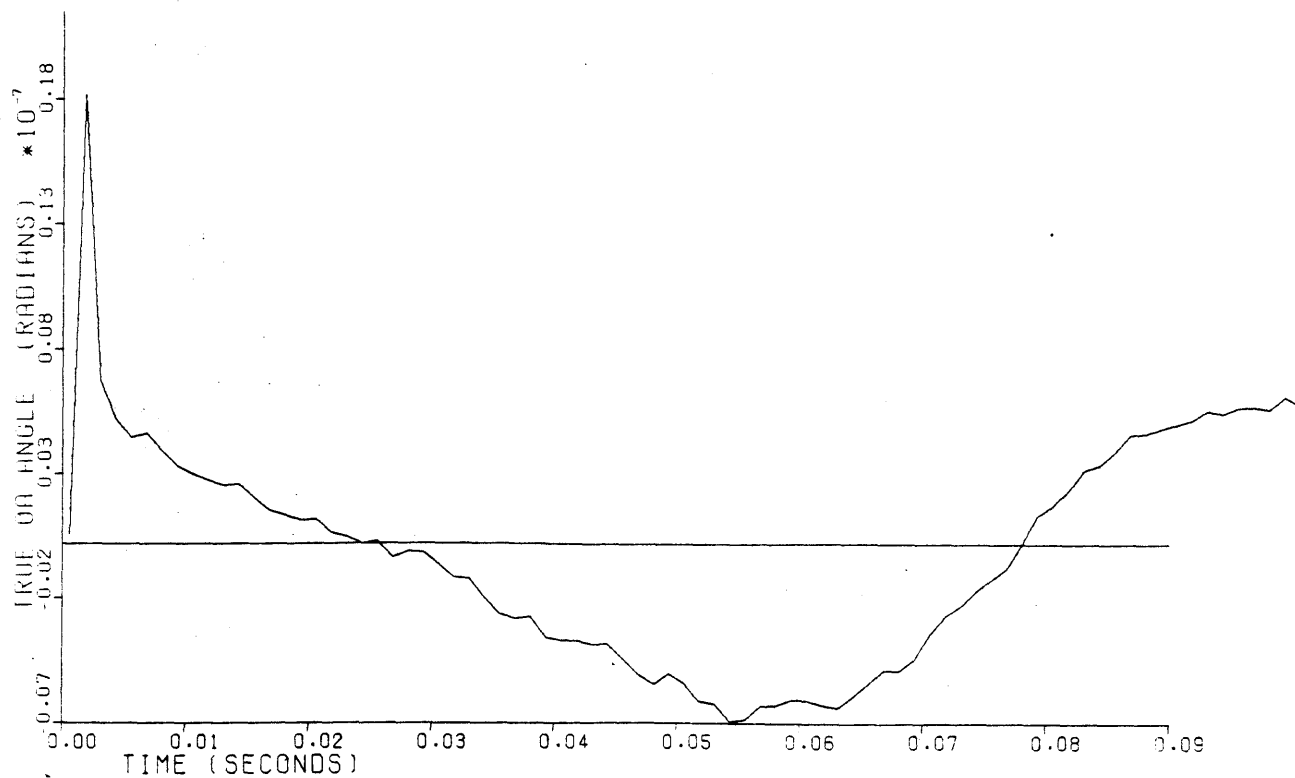


Figure 14-31. Run #22 24 Pt. Average Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

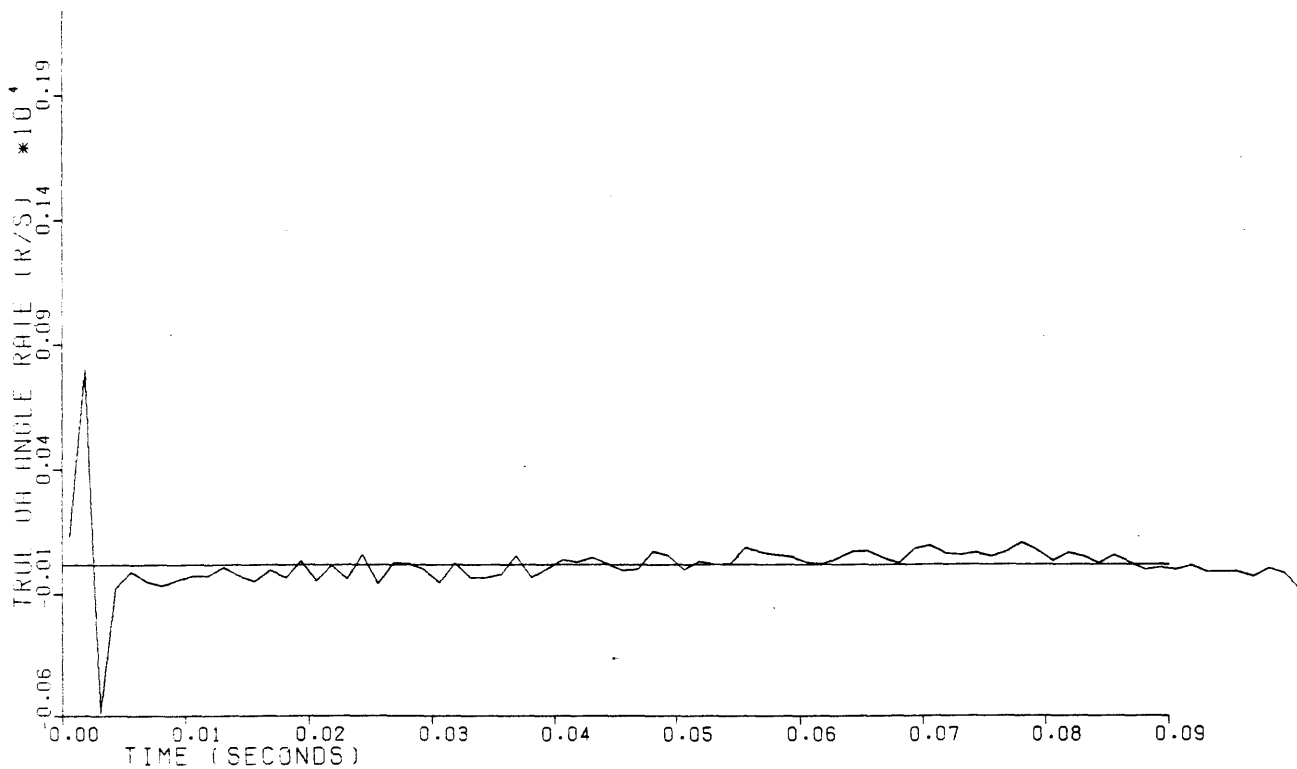


Figure 14-32. Run #22 24 Pt. Average Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

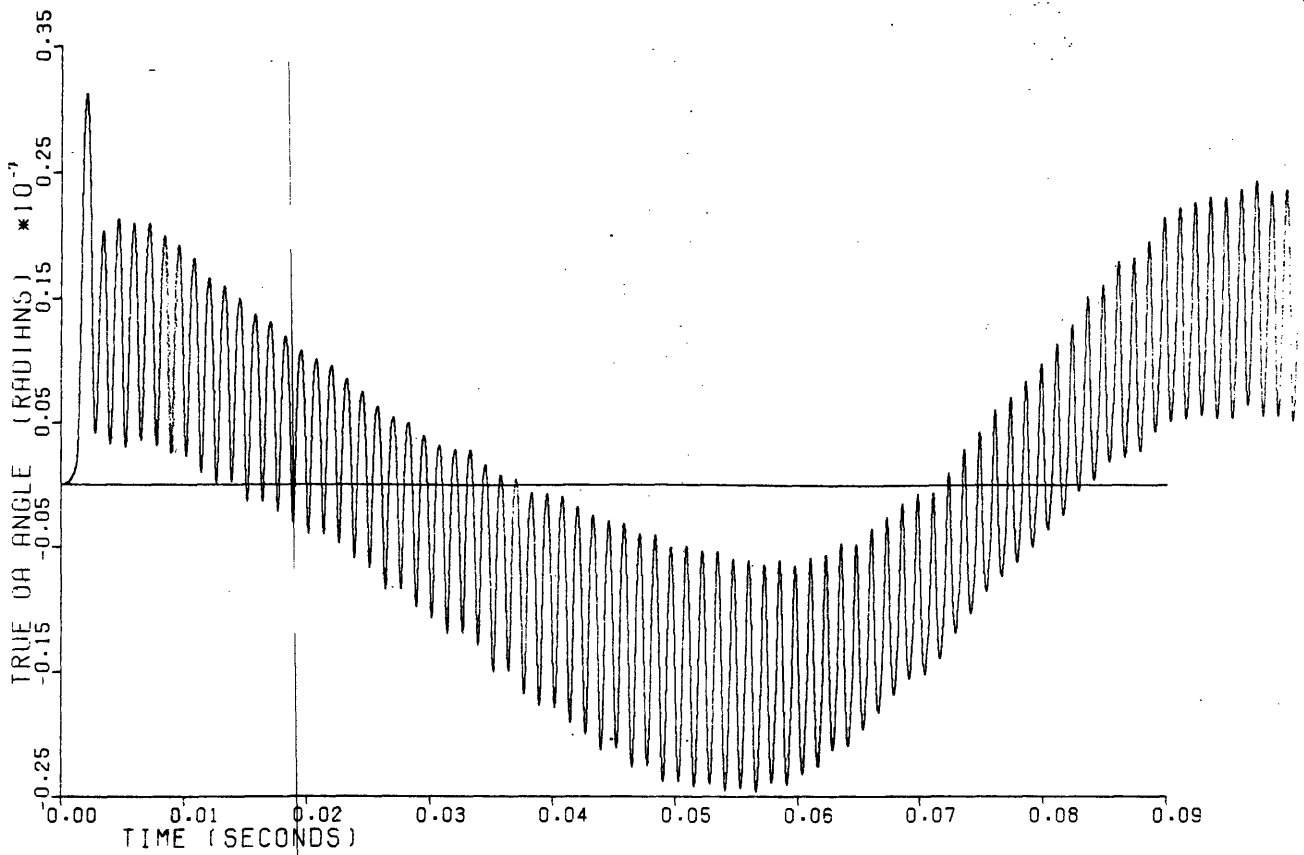


Figure 14-33. Run #23 Float Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

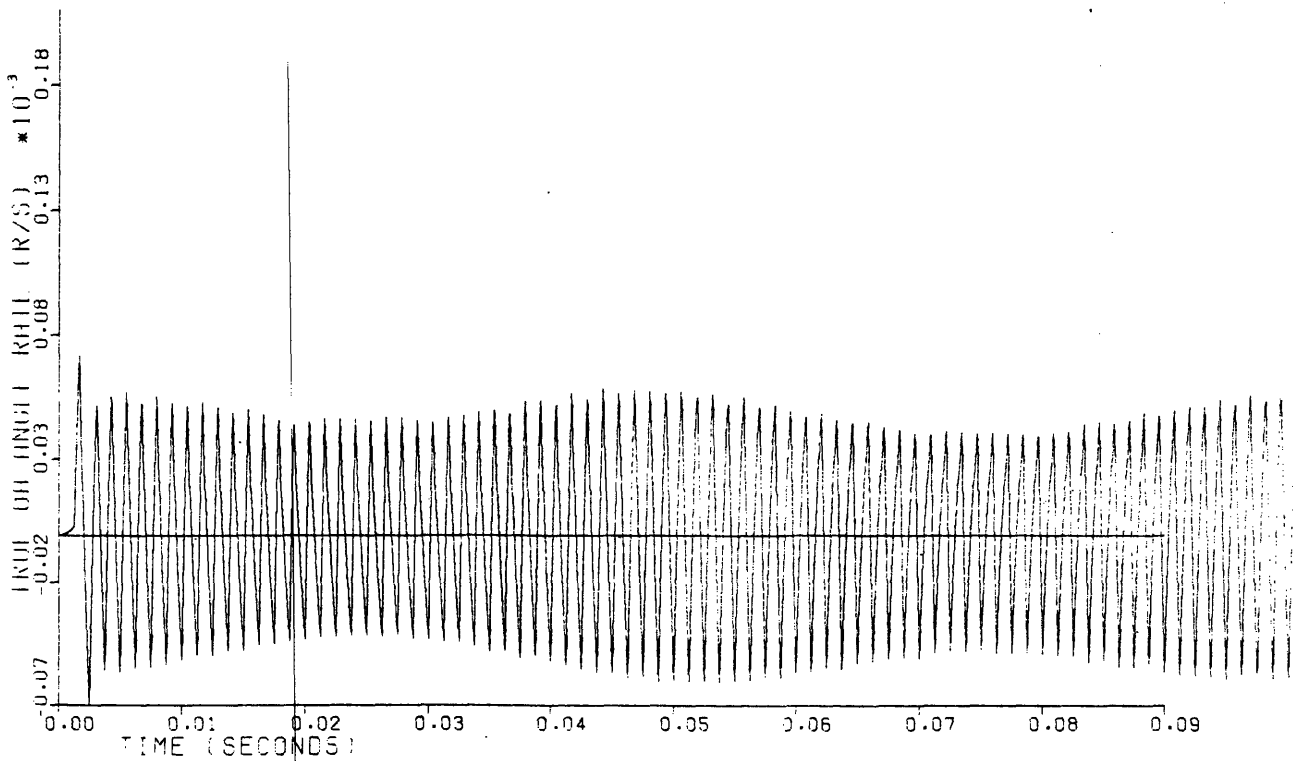


Figure 14-34. Run #23 Float Velocity : vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

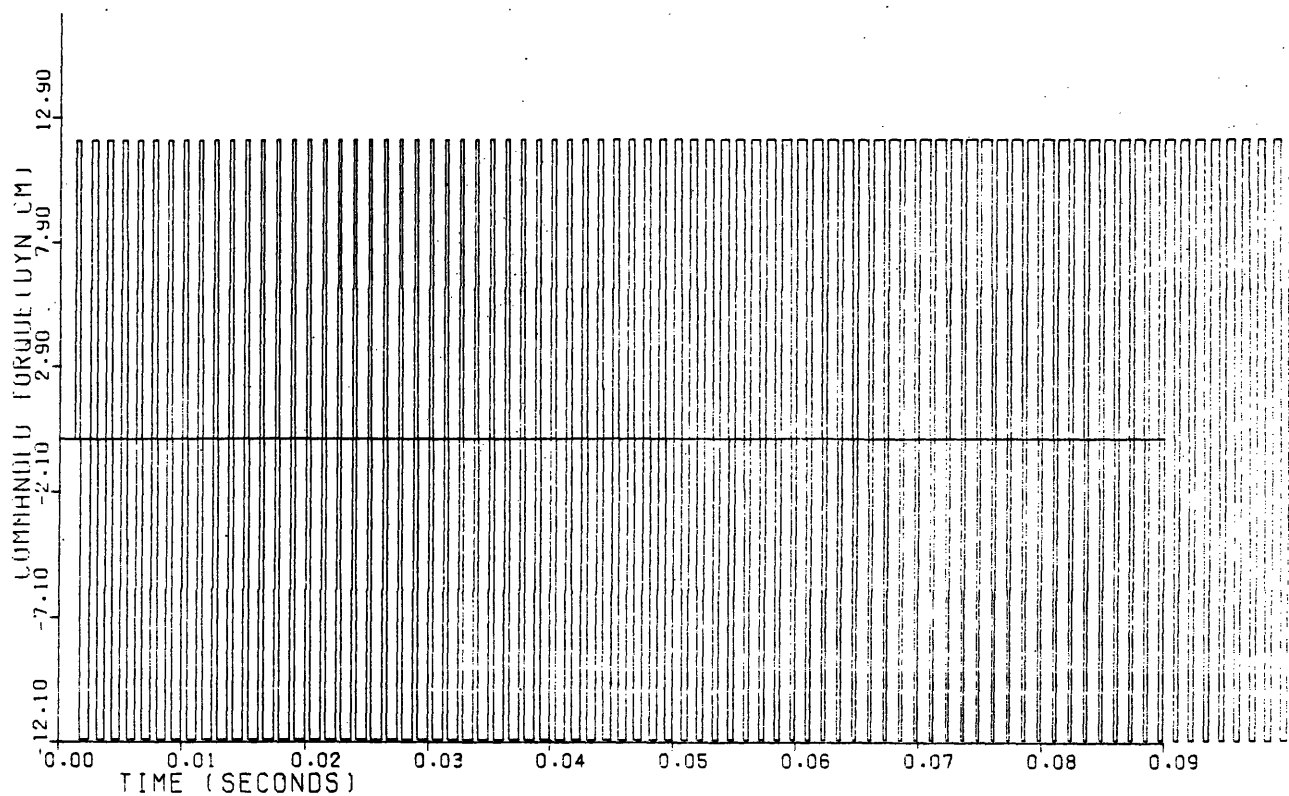


Figure 14-35. Run #23 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

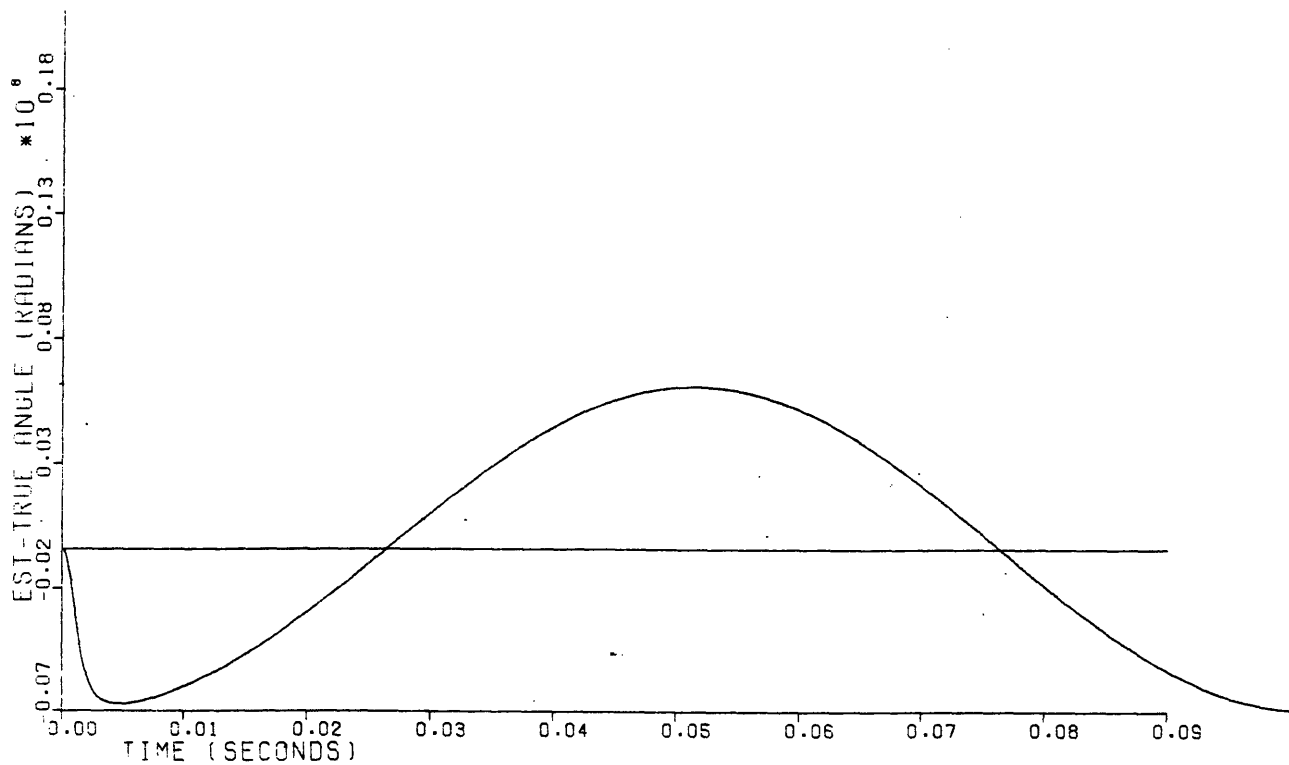


Figure 14-36. Run #23 Position Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

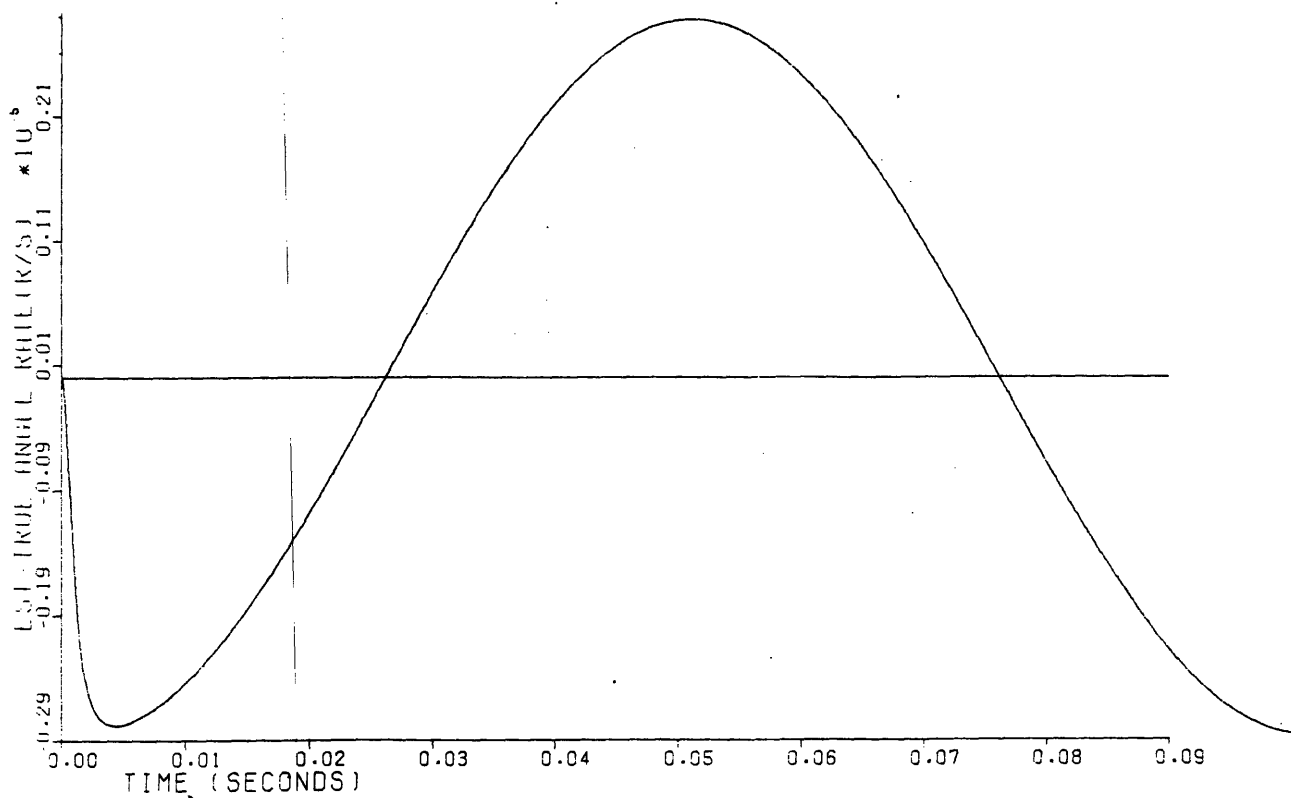


Figure 14-37. Run #23 Velocity Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

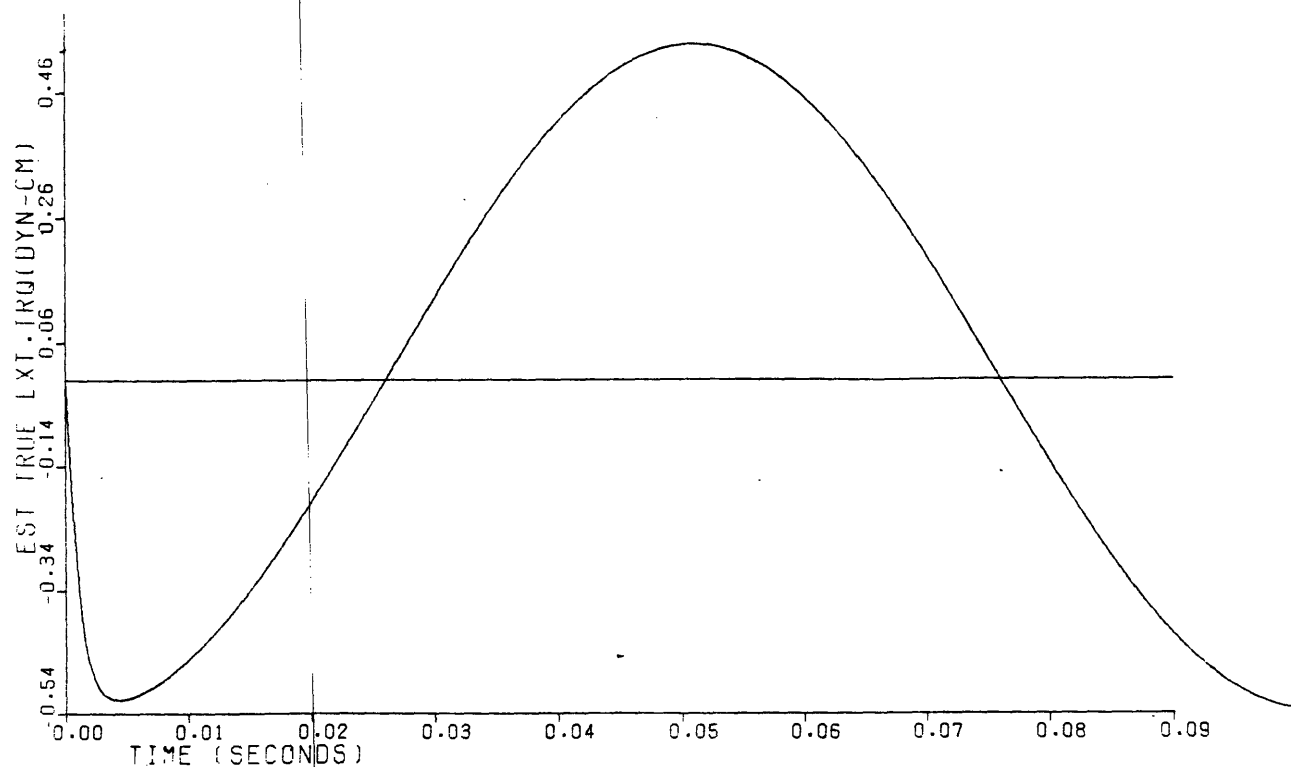


Figure 14-38. Run #23 External Torque Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

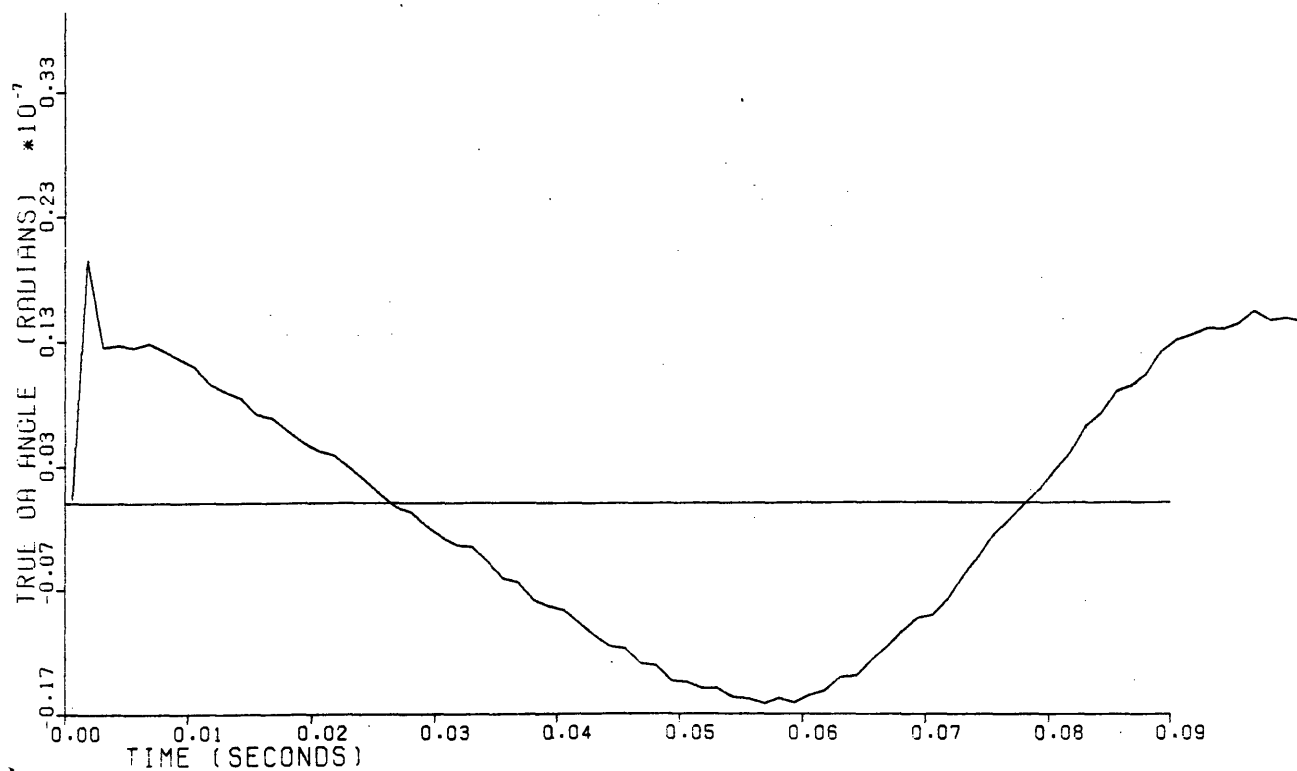


Figure 14-39. Run #23 24 Pt. Average Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

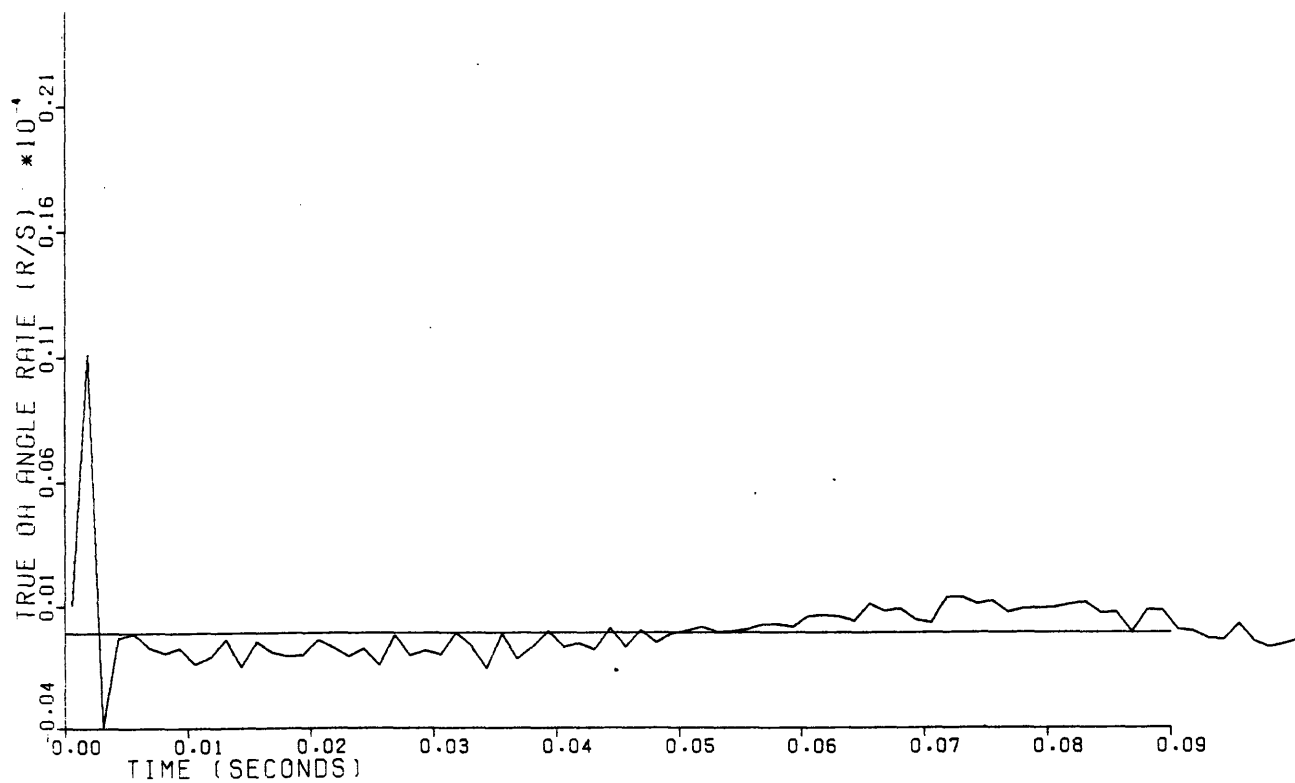


Figure 14-40. Run #23 24 Pt. Average Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

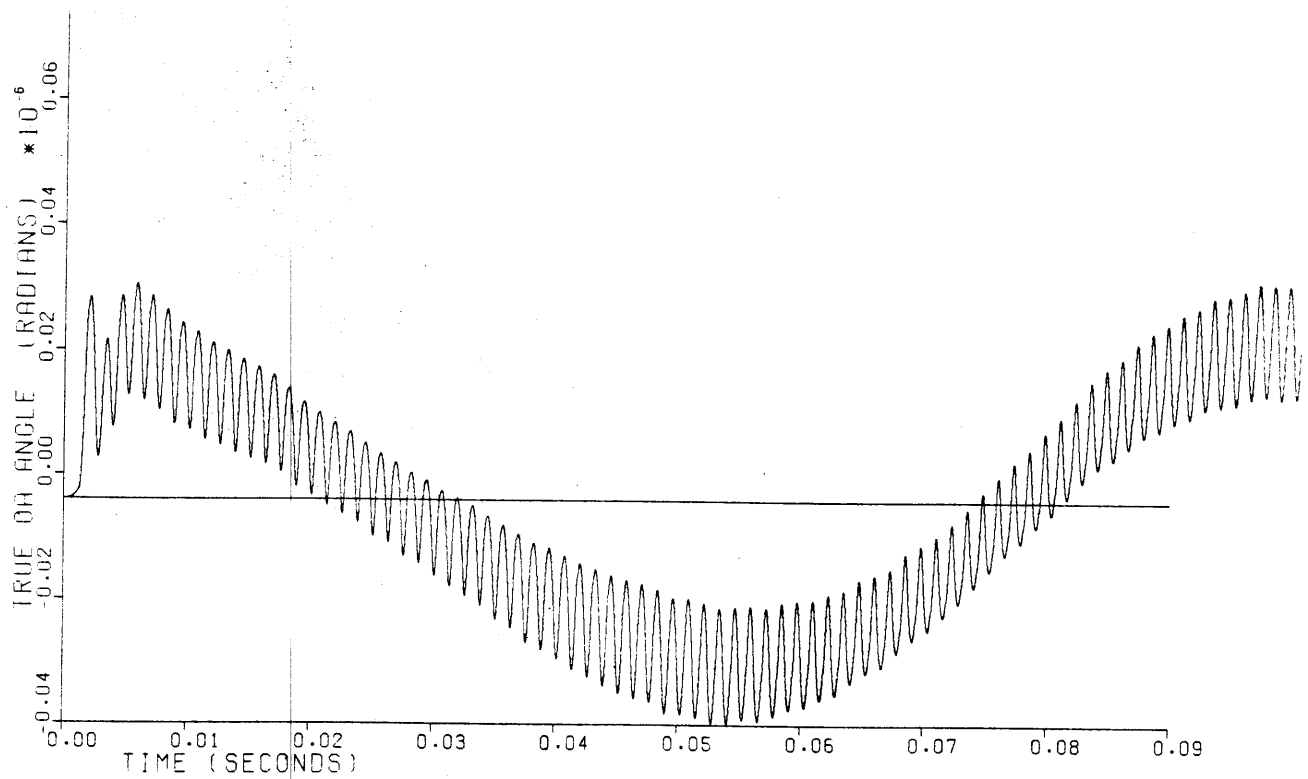


Figure 14-41. Run #24 Float Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

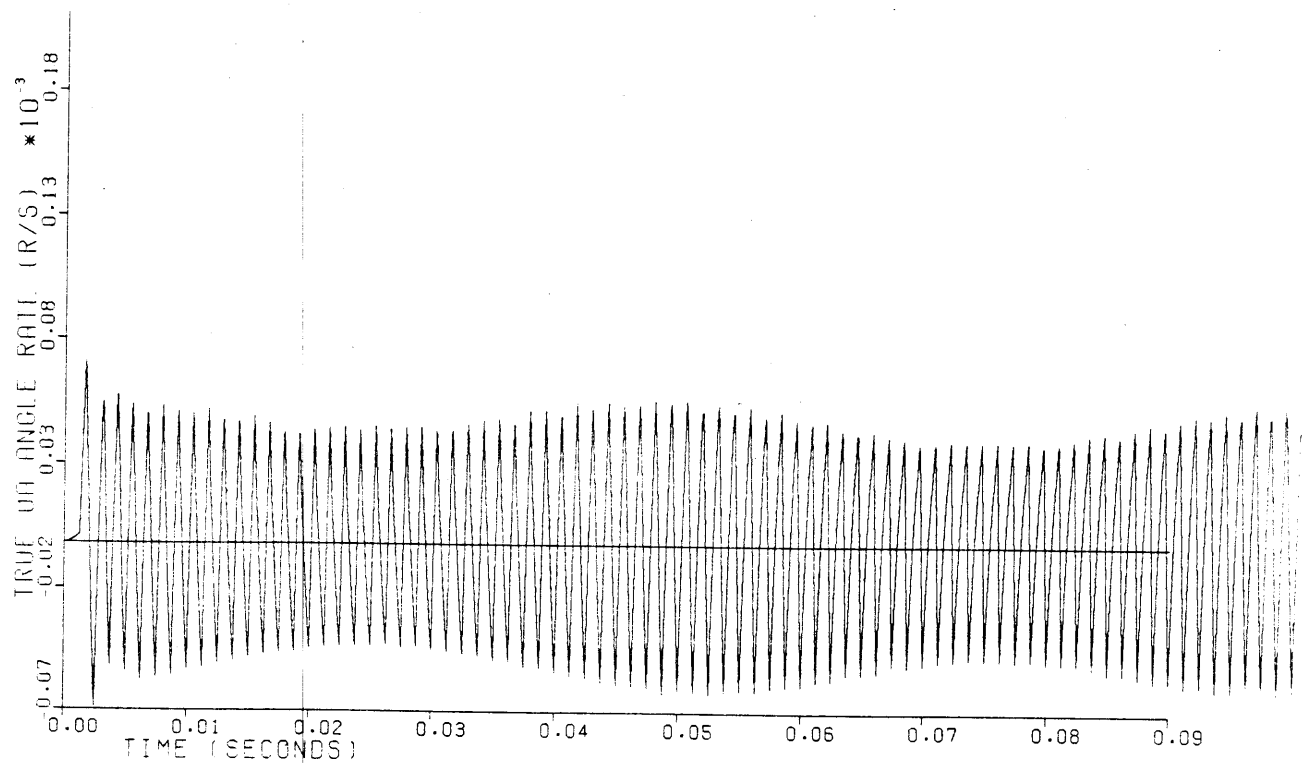


Figure 14-42. Run #24 Float Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

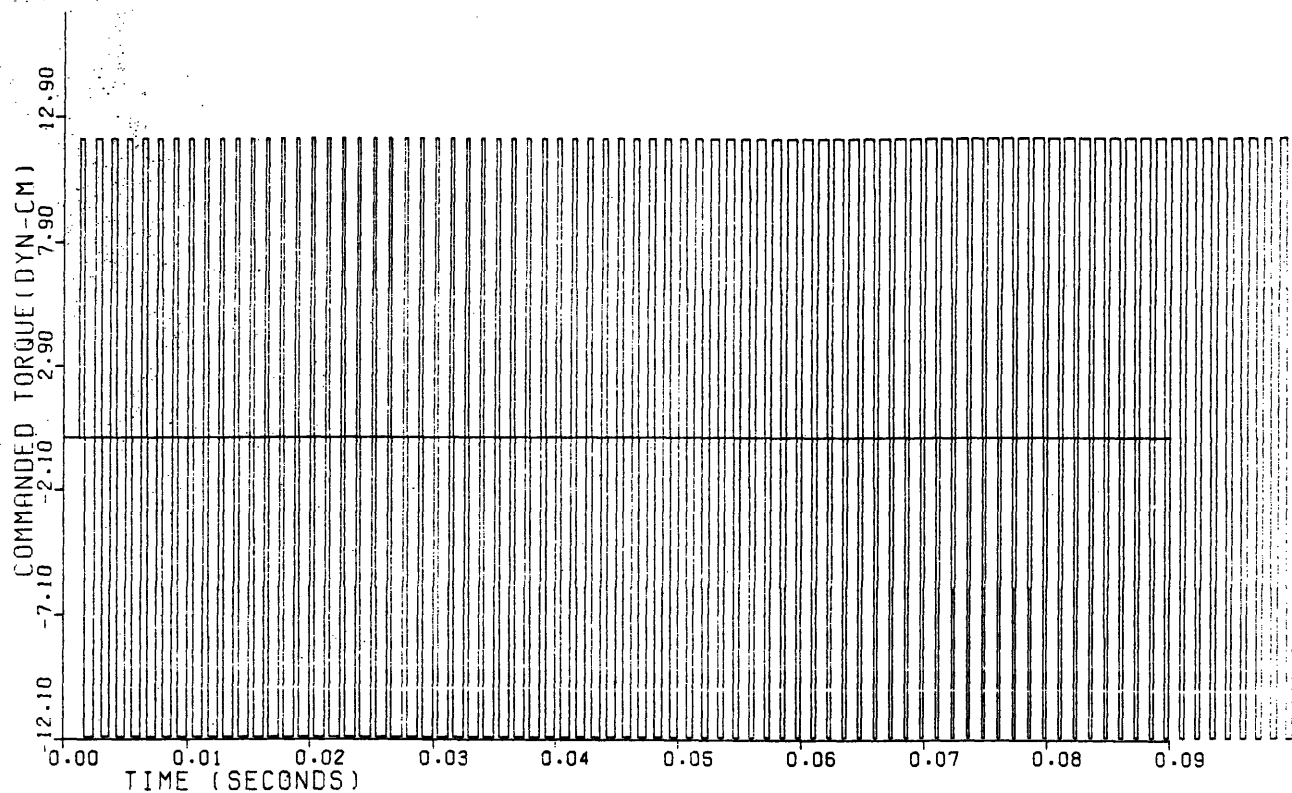


Figure 14-43. Run #24 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

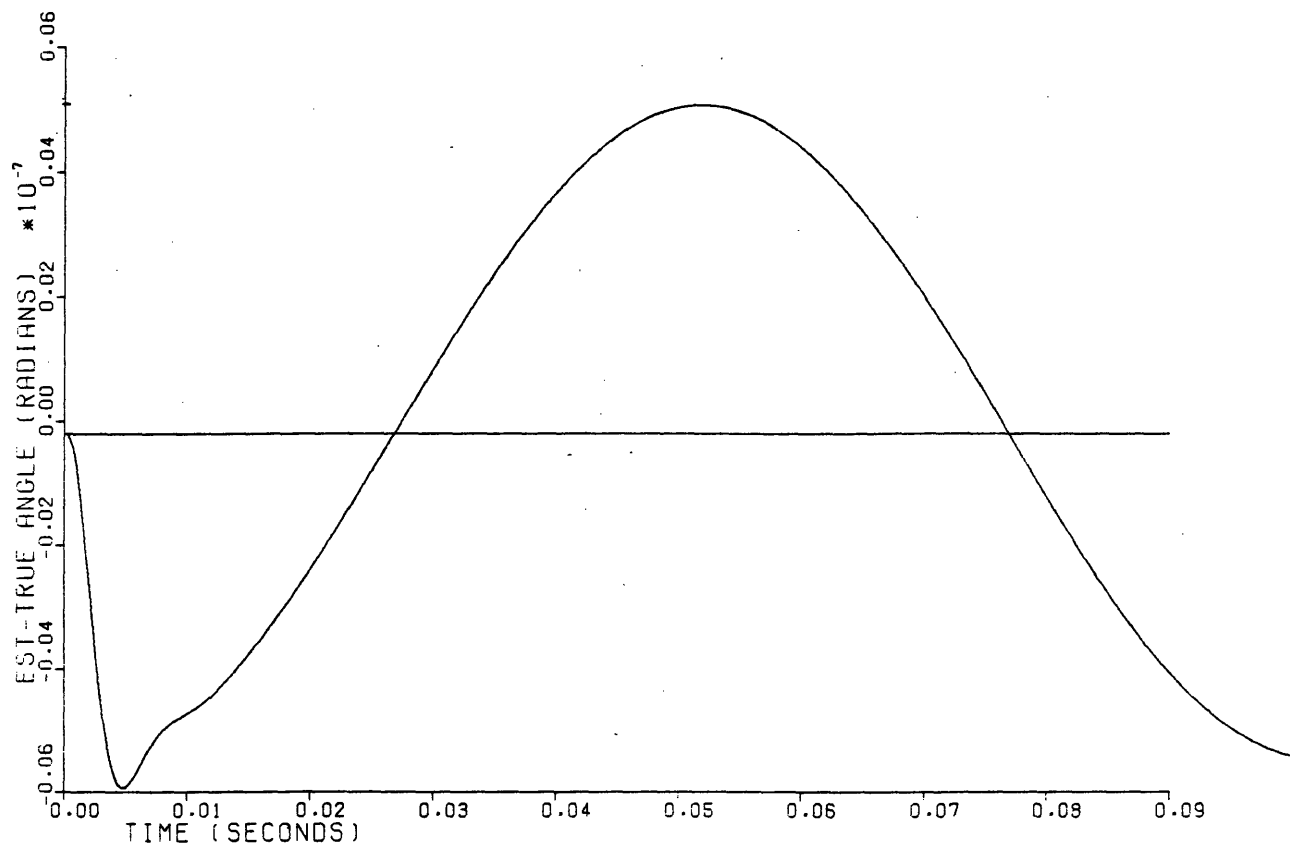


Figure 14-44. Run #24 Position Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

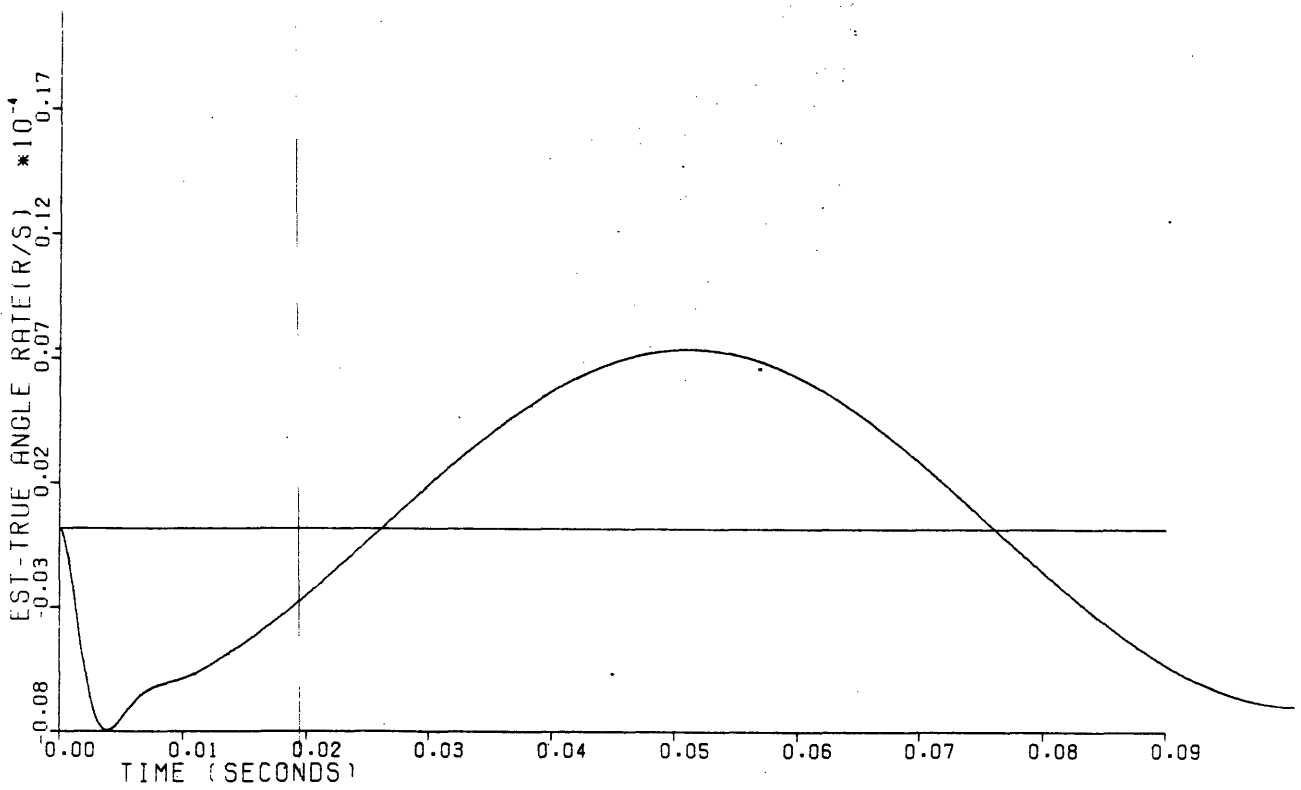


Figure 14-45. Run #24 Velocity Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

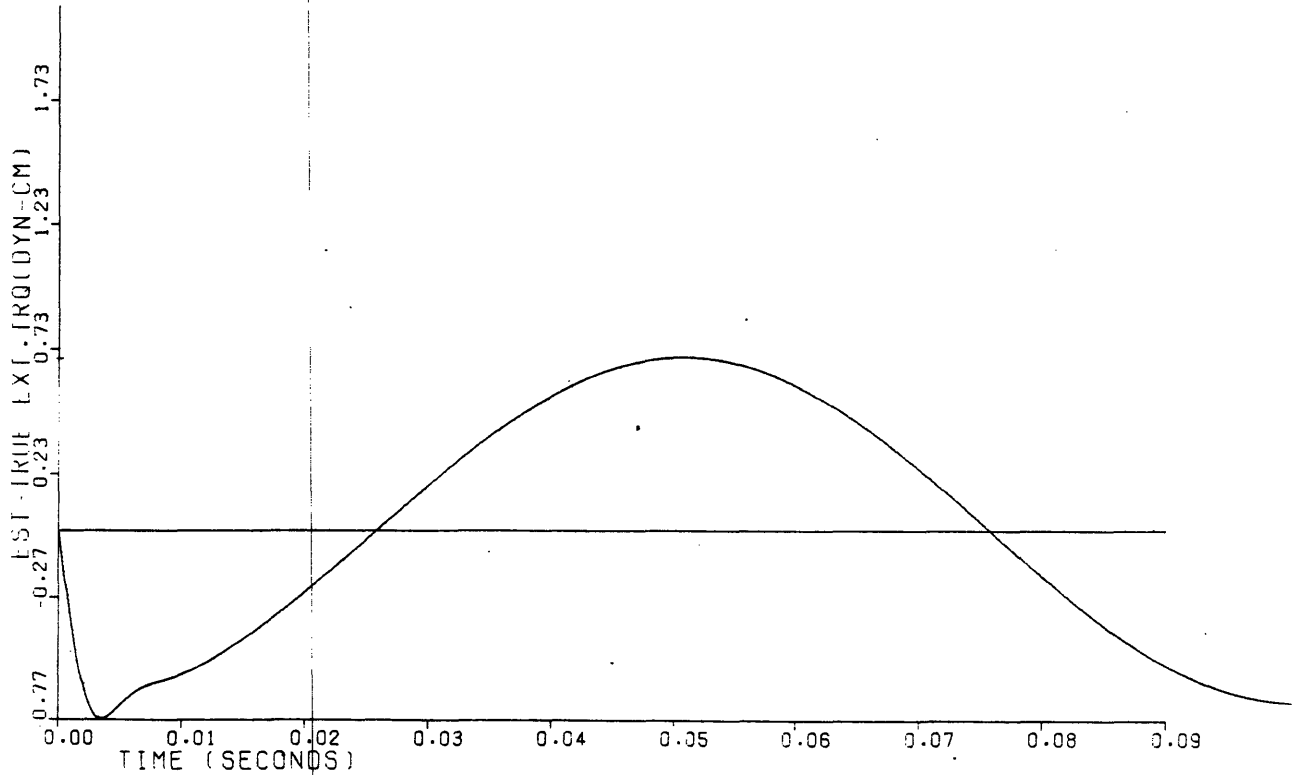


Figure 14-46. Run #24 External Torque Estimation Error vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

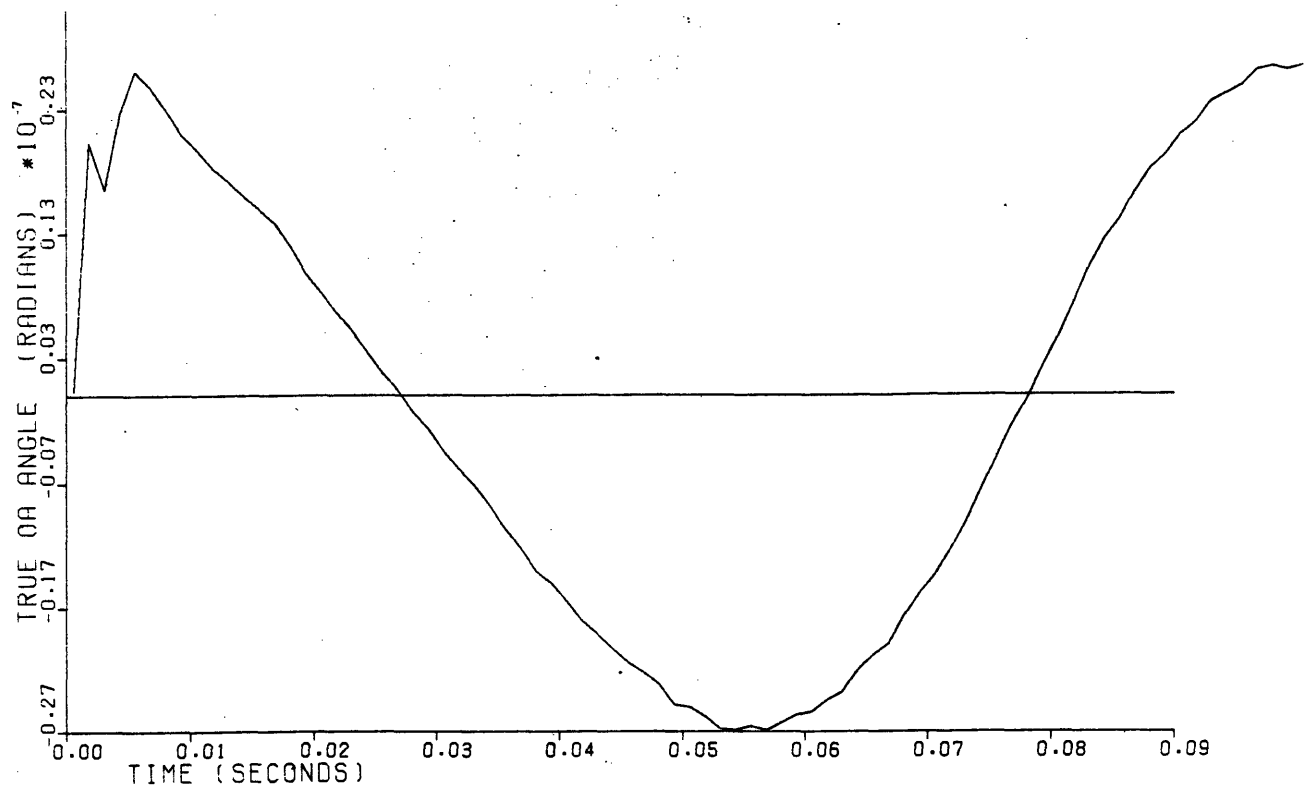


Figure 14-47. Run #24 24 Pt. Average Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

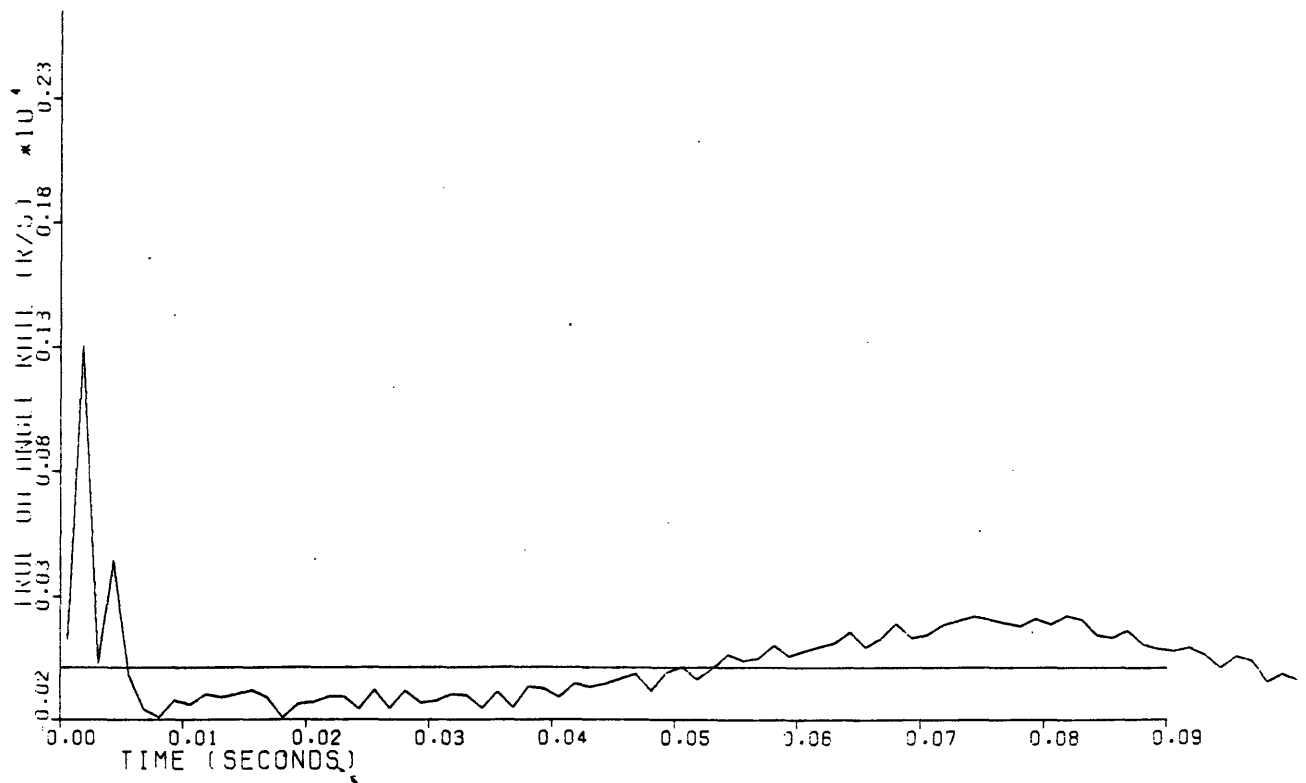


Figure 14-48. Run #24 24 Pt. Average Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

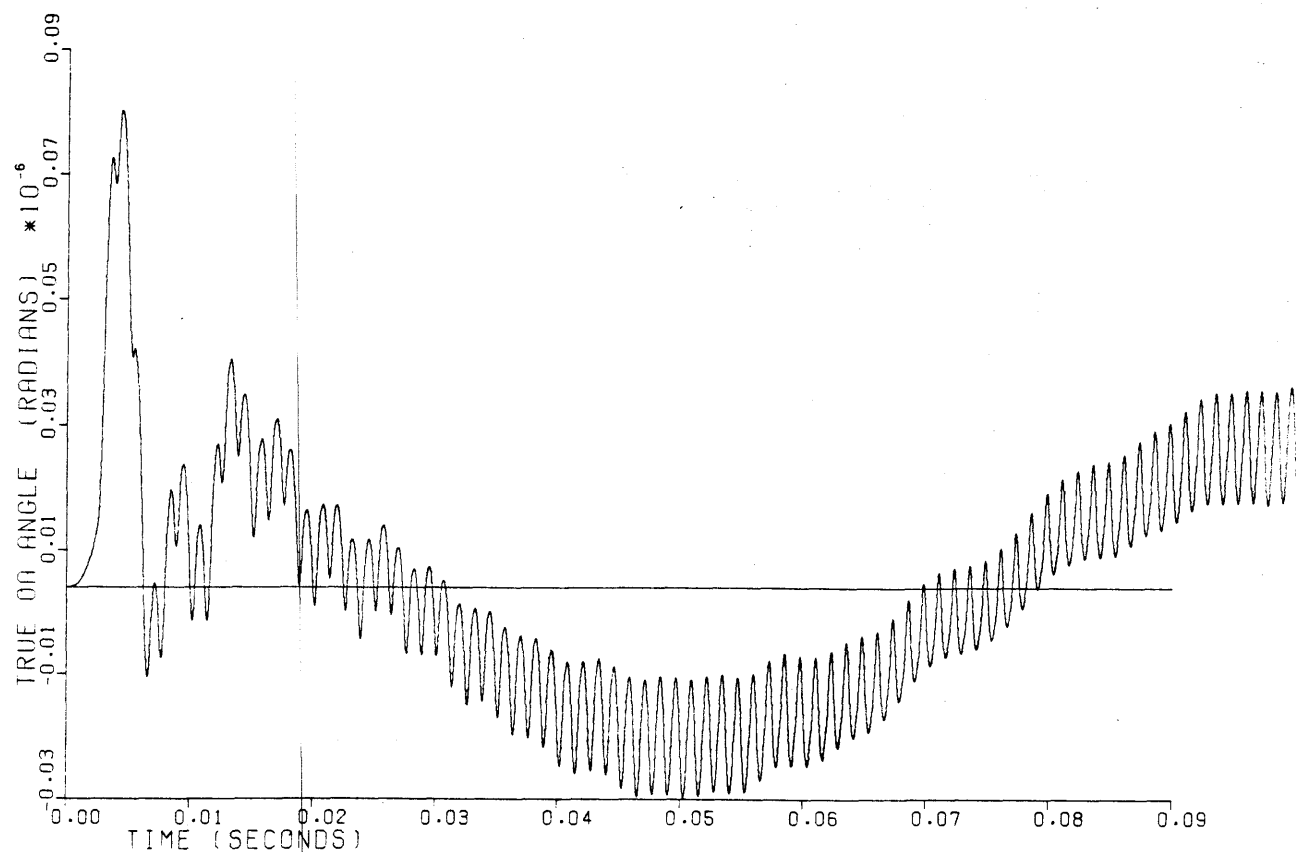


Figure 14-49. Run #25 Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

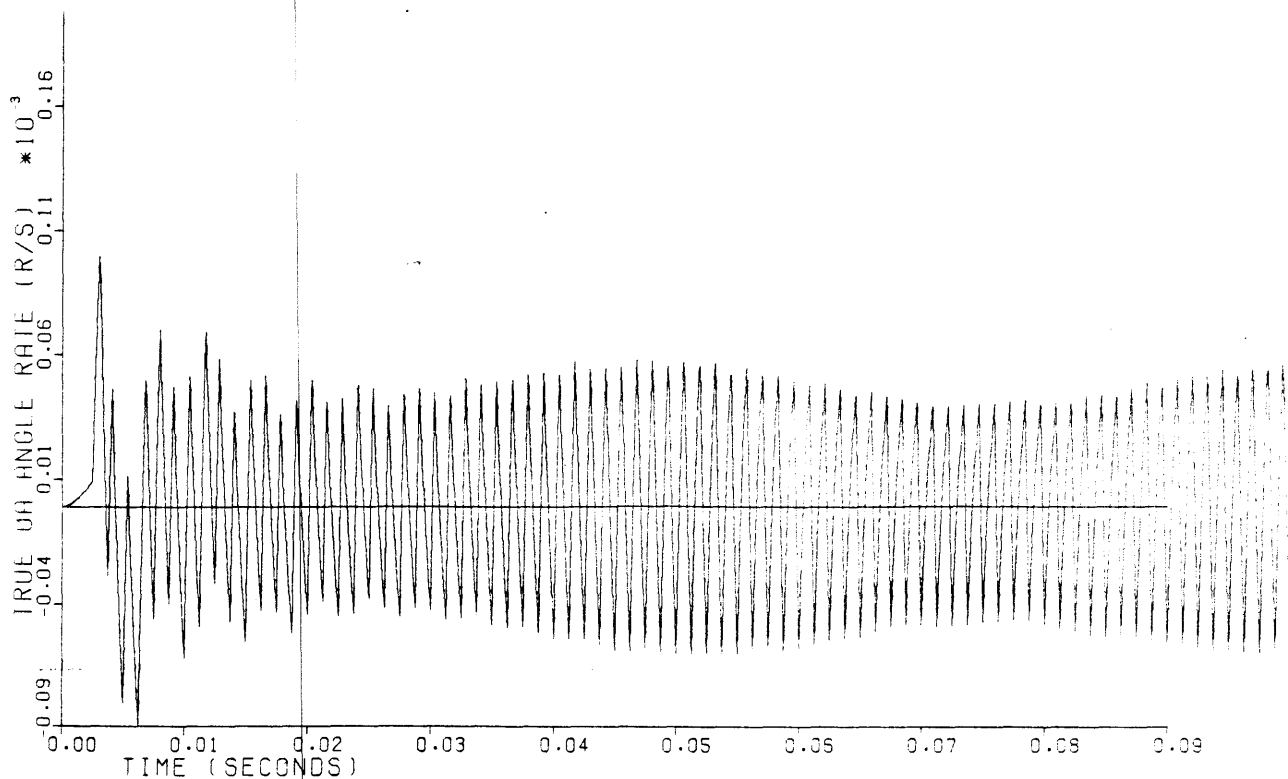


Figure 14-50. Run #25 Float Velocity vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

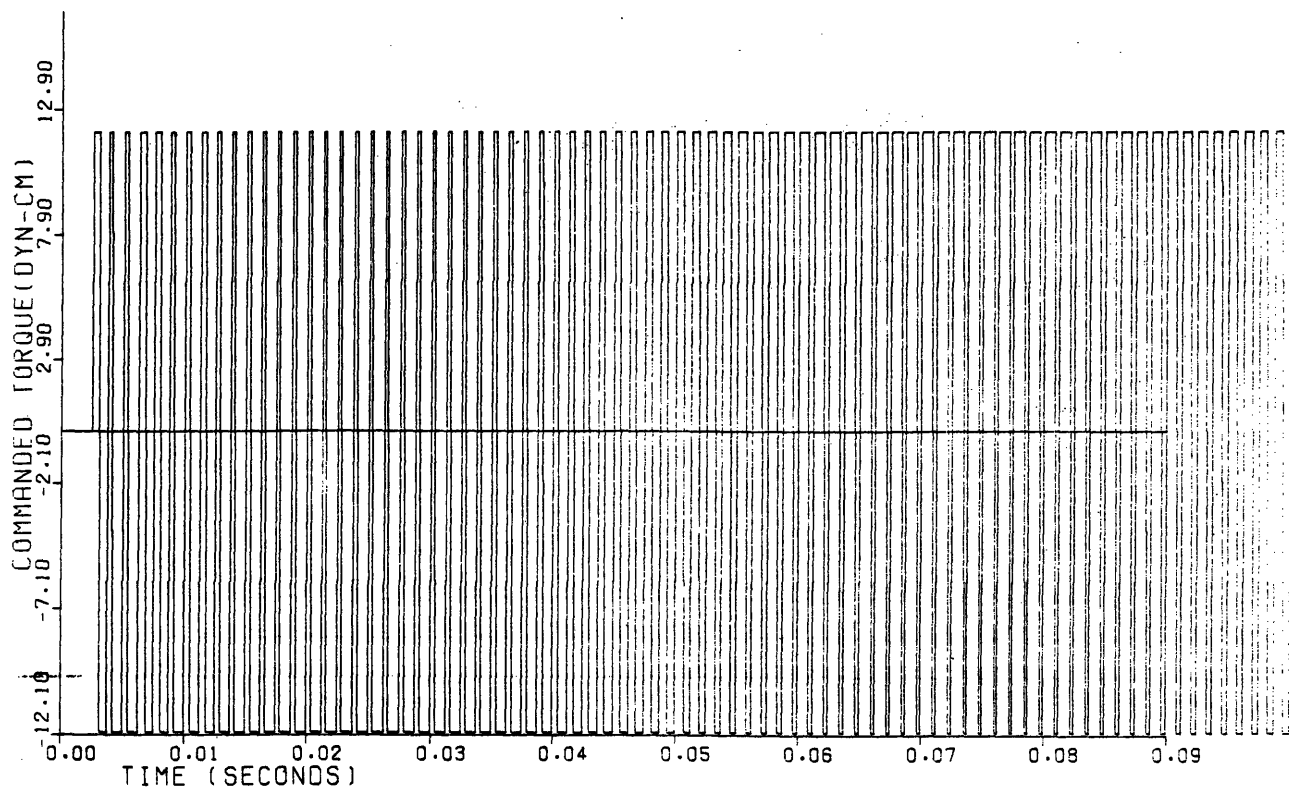


Figure 14-51. Run #25 Commanded Torque vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

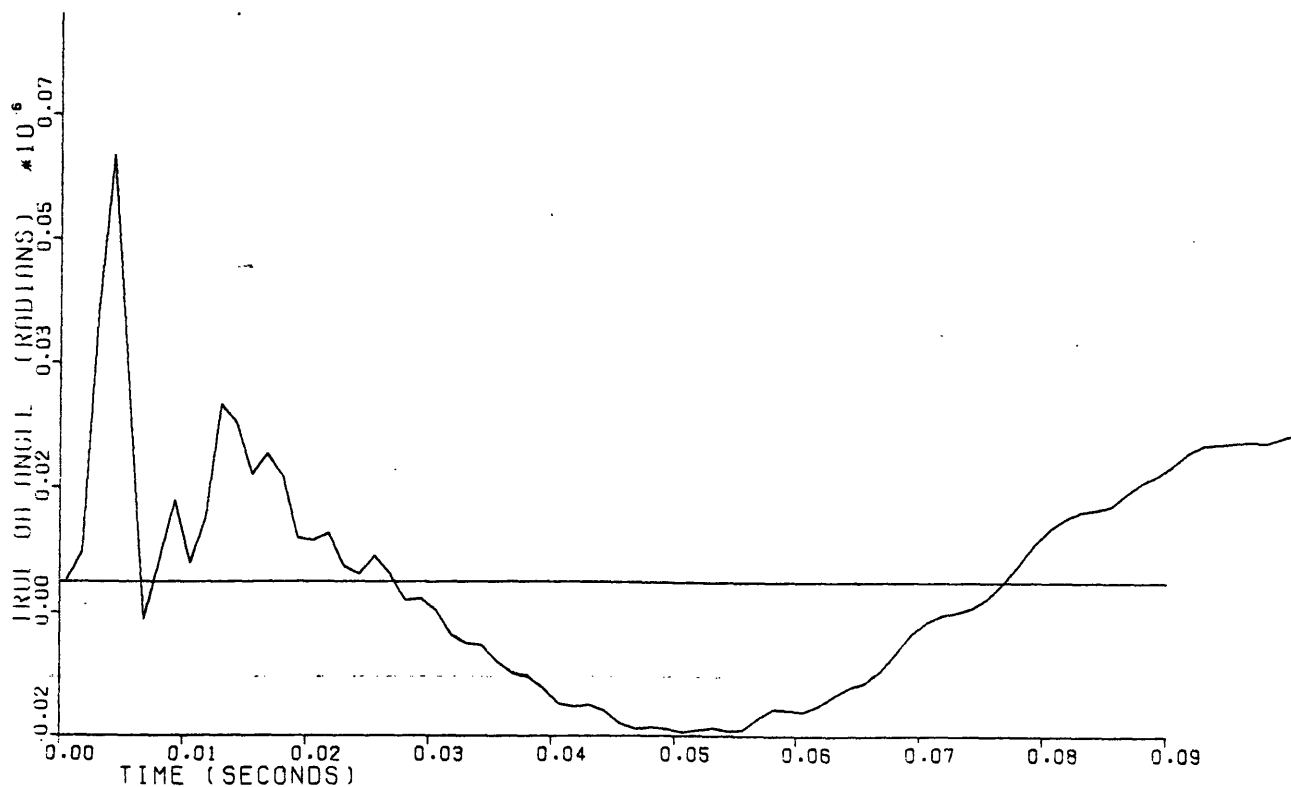


Figure 14-52. Run #25 24 Pt. Average Position vs. Time (10 Hz, 11.83°/hr Sinusoidal Input)

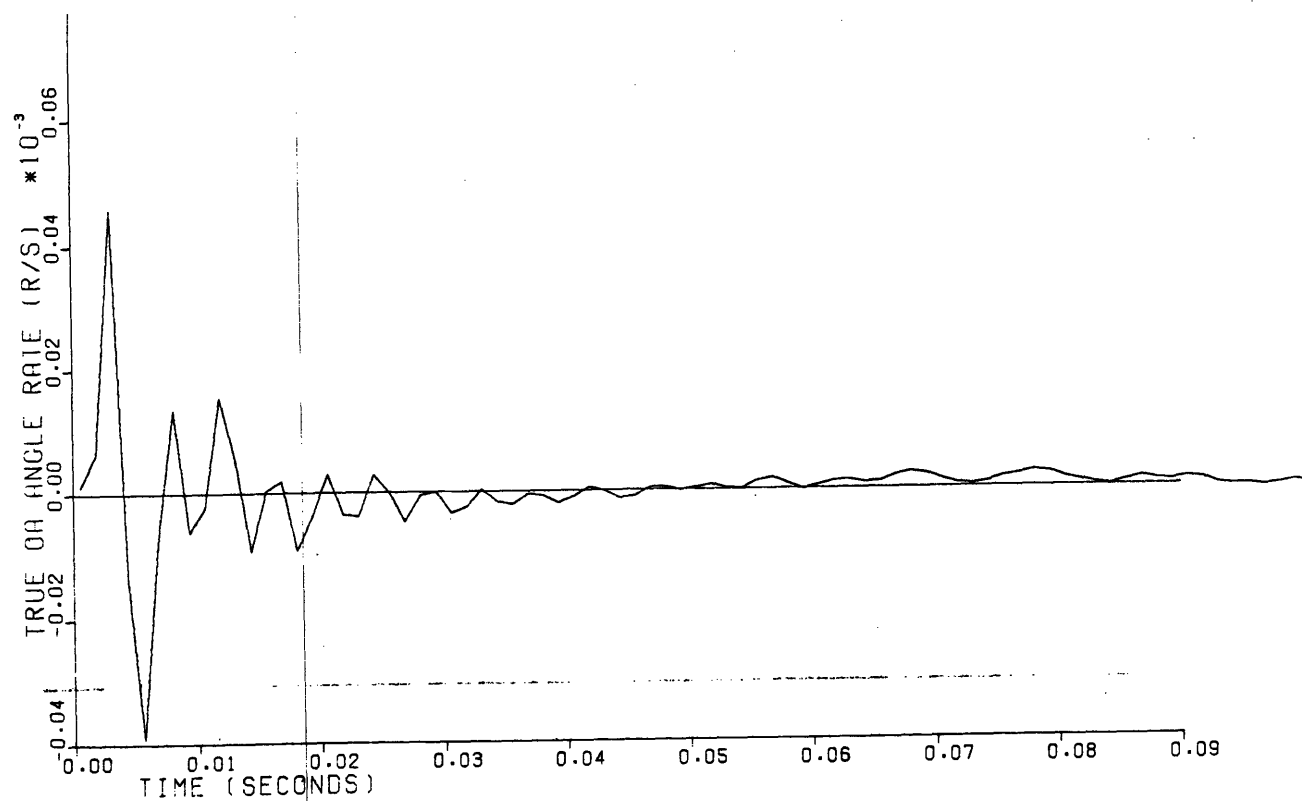


Figure 14-53. Run #25 24 Pt. Average Velocity vs Time (10 Hz, 11.83°/hr Sinusoidal Input)

15. RESPONSE TO SIGNAL GENERATOR NOISE

15.1 Results Summary

Table 15-1 shows the summary of results for the controller responses to signal generator noise. Since the three and four state controllers behaved identically with SG noise, only the four state controller at the three sets of gains are presented to represent both.

Table 15-1. Results Summary for Constant Input of $7.5^\circ/\text{hr} = 3.742 \text{ dyne-cm}$ with SG Noise.

Run #	Description	Noise level (nrad)	Max. float deviation from null (nrad)	Maintained min. mode of float oscillation	Maintained min. quantization in torquing pattern
26	4 st. controller fast gains	1	47.47	no	no
27		10	-	-	-
28		100	-	-	-
29	4 st. controller medium gains	1	12.22	yes	no
30		10	43.85	no	no
31		100	-	-	-
32	4 st. controller slow gains	1	11.55	yes	yes
33		10	19.46	yes	no
34		100	132.70	no	no
35	simplified controller	1	11.78	yes	yes
36		10	17.29	yes	no
37		100	114.50	no	no

As expected, the addition of SG noise degraded performance of the torque loop. The empty spaces in Table 15-1 indicate that the particular simulation run was terminated due to floating point numbers exceeding allowable limits. The ability of the controller to perform with noise is directly related to the ability of the Kalman filter estimator to reject noise. This ability is of course determined by the estimator gains, specifying how much weight is placed on the measurements versus the propagated estimate in determining the updated estimate. For the three sets of gains chosen, approximately a factor of ten improvement in the noise response is seen between each. At the higher noise levels and faster gains, the float no longer maintained an 800 Hz oscillation frequency and the torquing pattern no longer represented the actual external input with minimum quantization. The results were oscillations at slower frequencies and of greater amplitudes. Even so, all controllers did keep the average float position close to null. Again, the simplified controller behaved comparably to the Kalman filter controller with slow gains.

15.2 Simulation Plots

Figures 15-1 through 15-63 present the plots for the response to SG noise.

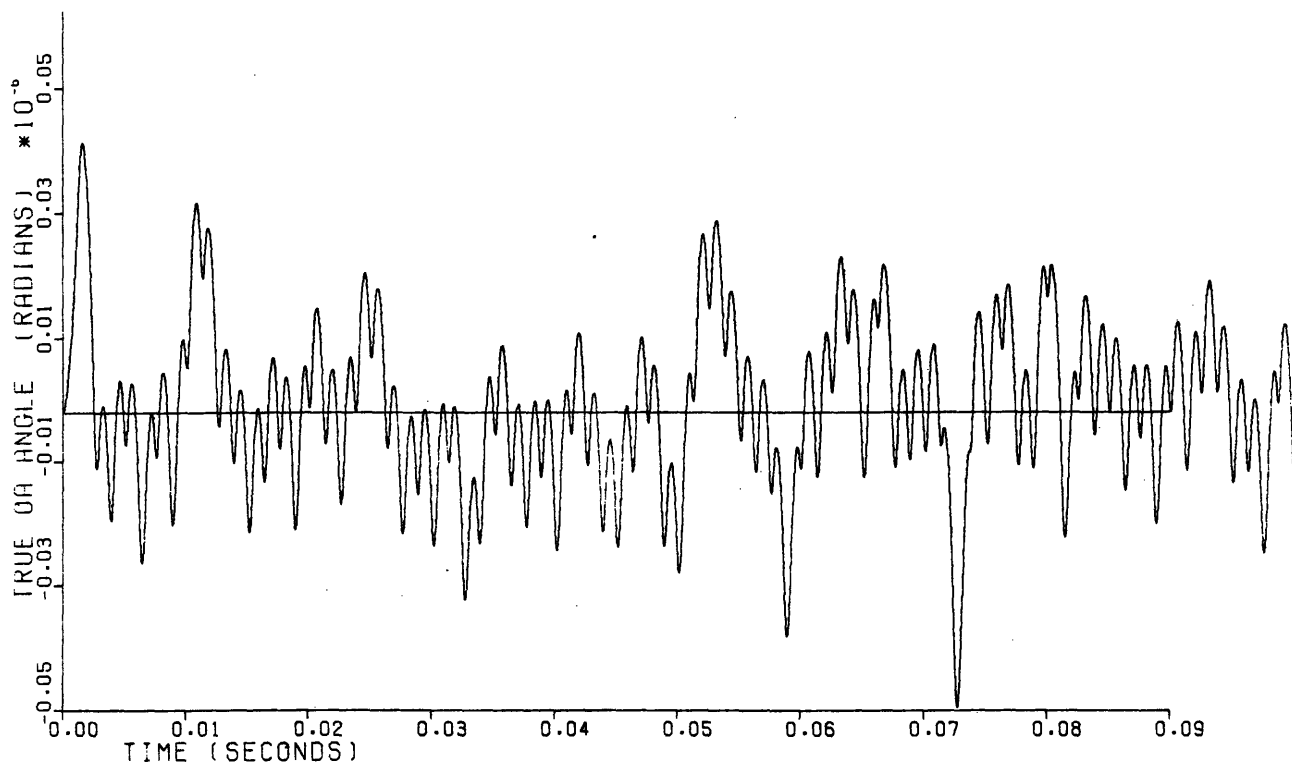


Figure 15-1. Run #26 Float Position vs. Time (7.5°/hr with S.G. Noise Level=1 nrad)

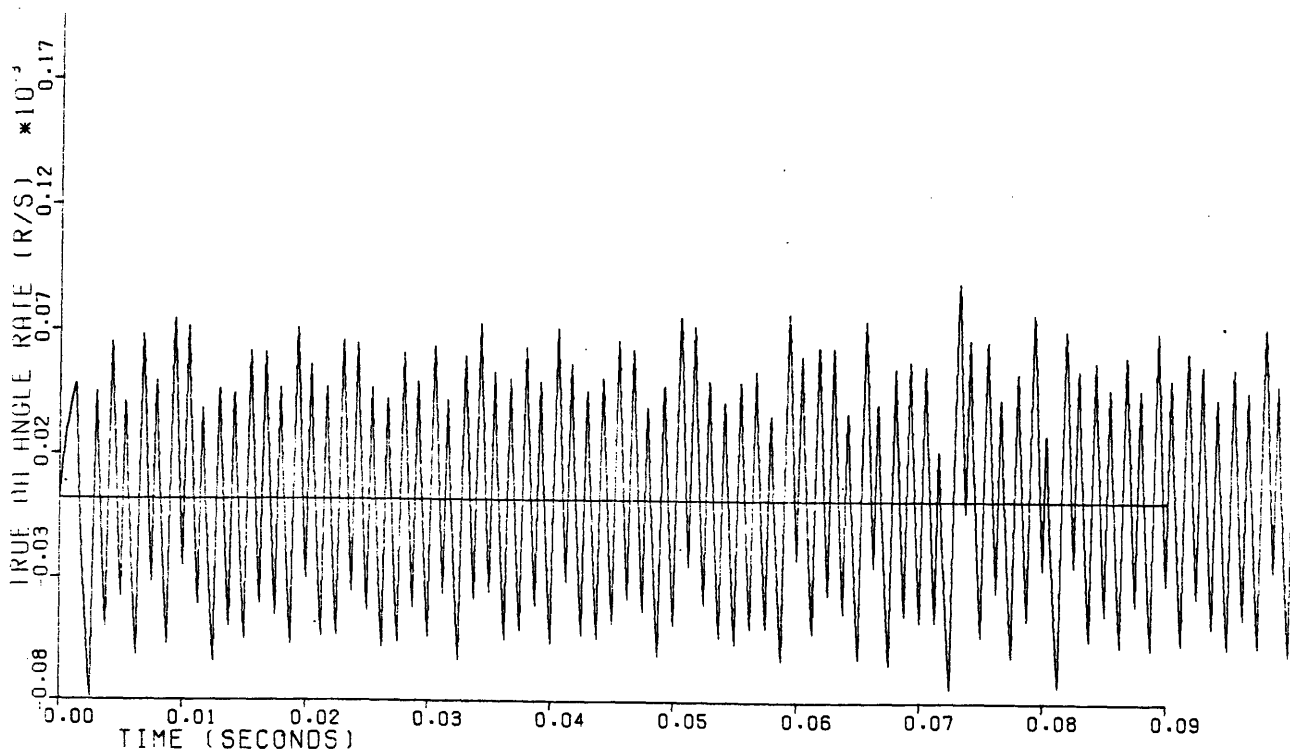


Figure 15-2. Run #26 Float Velocity vs. Time (7.5°/hr with S.G. Noise Level=1 nrad)

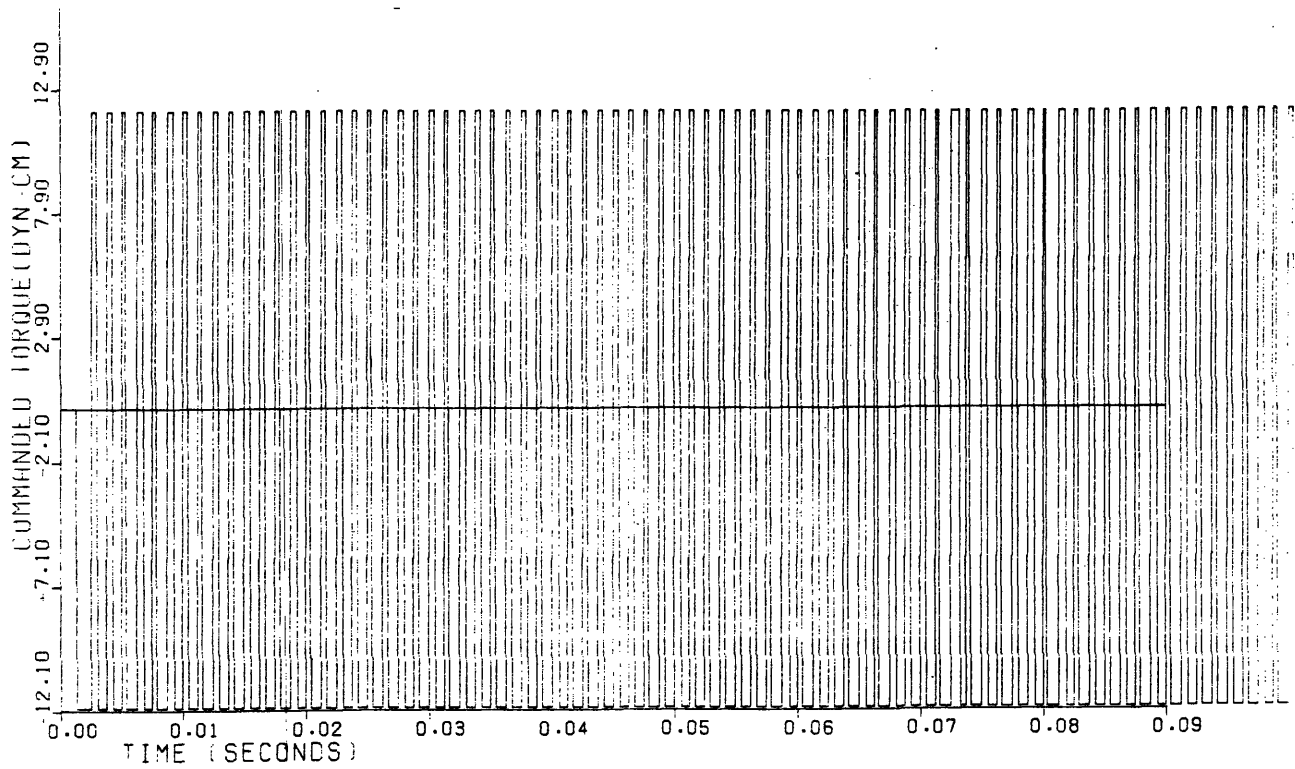


Figure 15-3. Run #26 Commanded Torque vs. Time (7.5°/hr with S.G. Noise Level=1 nrad)

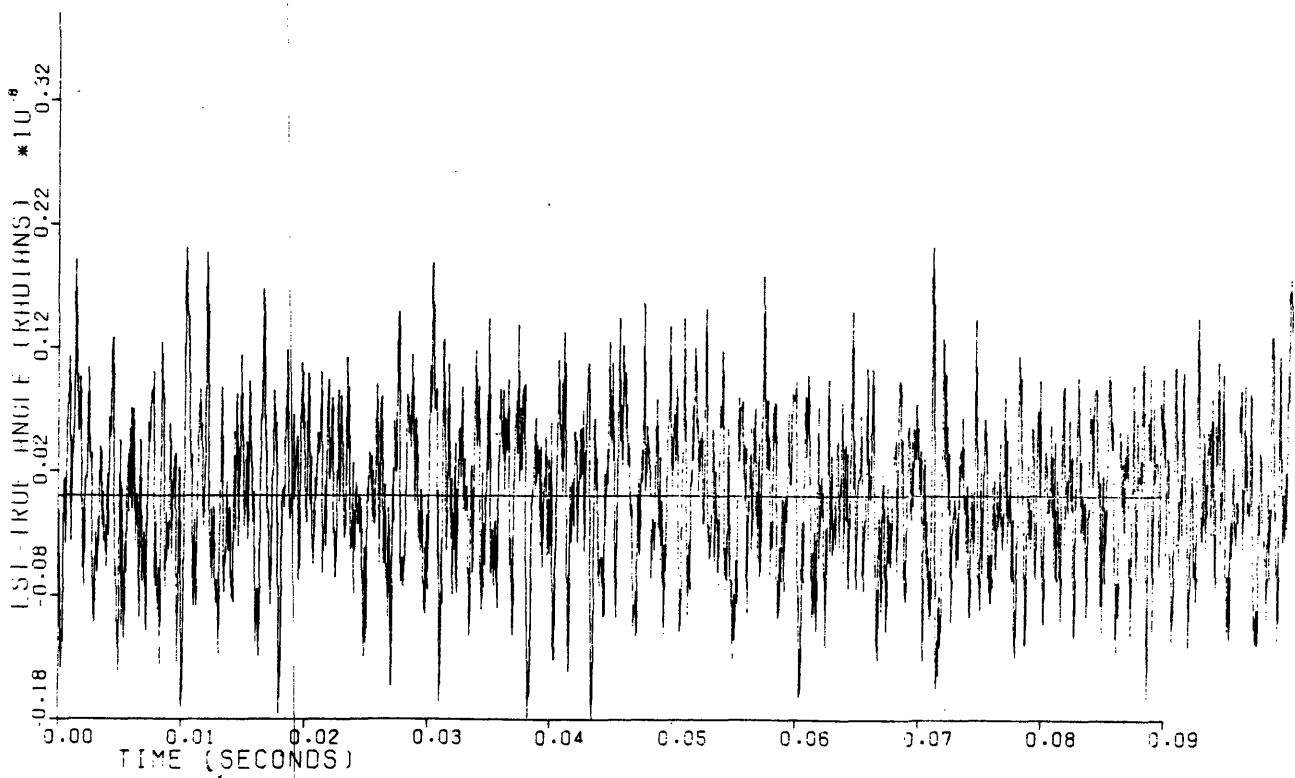


Figure 15-4. Run #26 Position Estimation Error vs. Time (7.5°/hr with S.G. Noise Level=1 nrad)

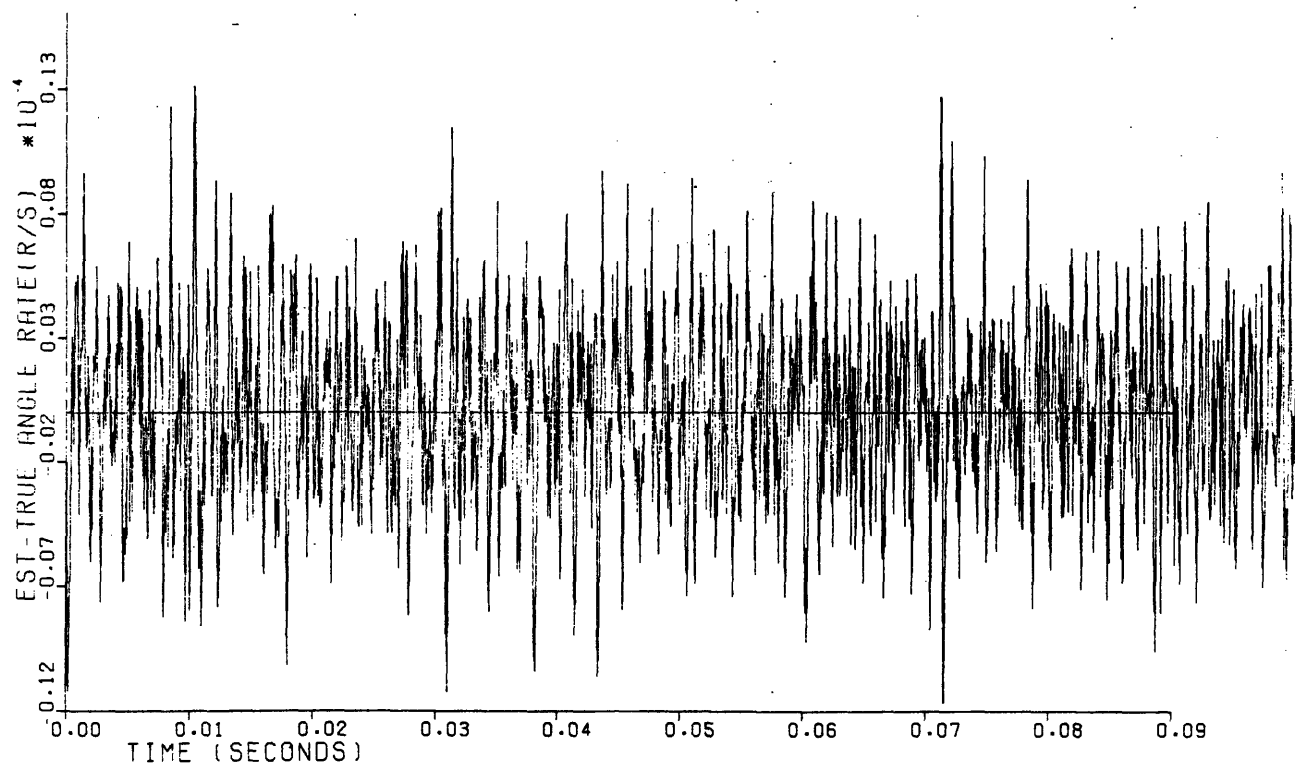


Figure 15-5. Run #26 Velocity Estimation Error vs. Time (7.5°/hr with S.G. Noise Level=1 nrad)

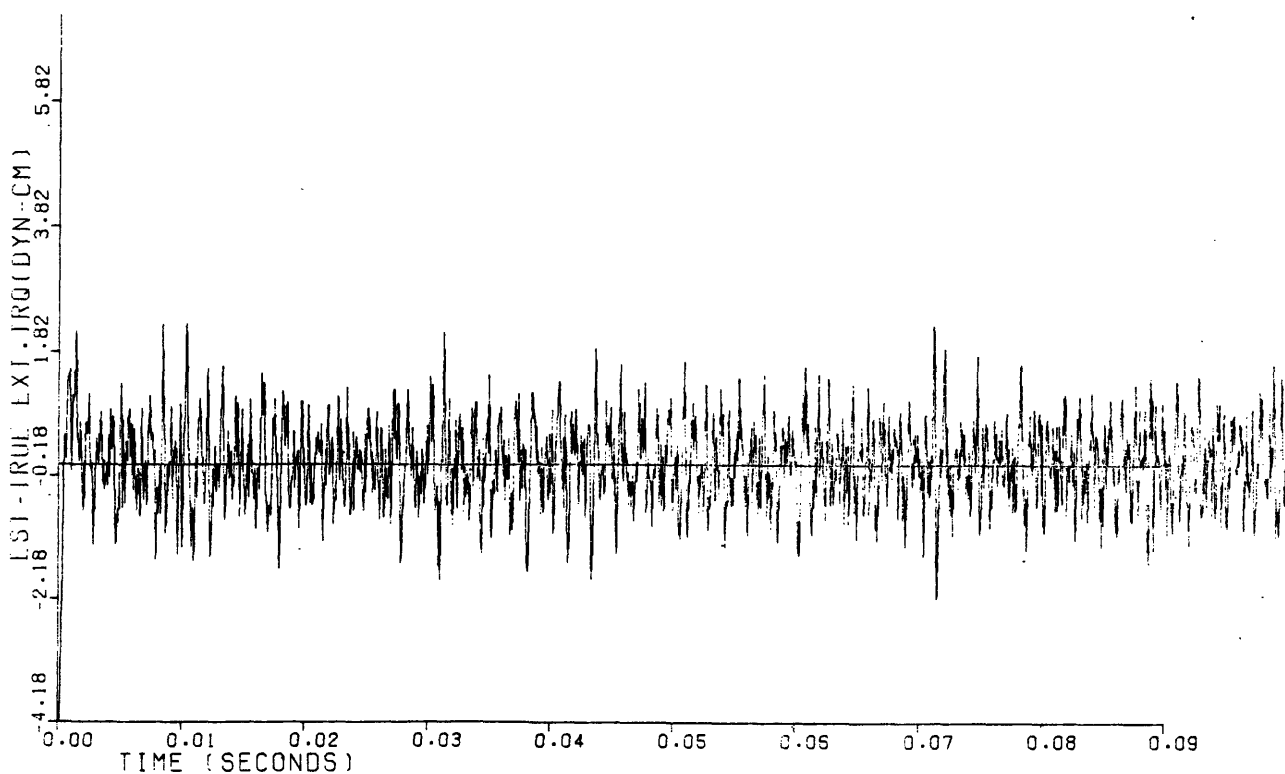


Figure 15-6. Run #26 External Torque Estimation Error vs. Time (7.5°/hr with S.G. Noise Level=1 nra)

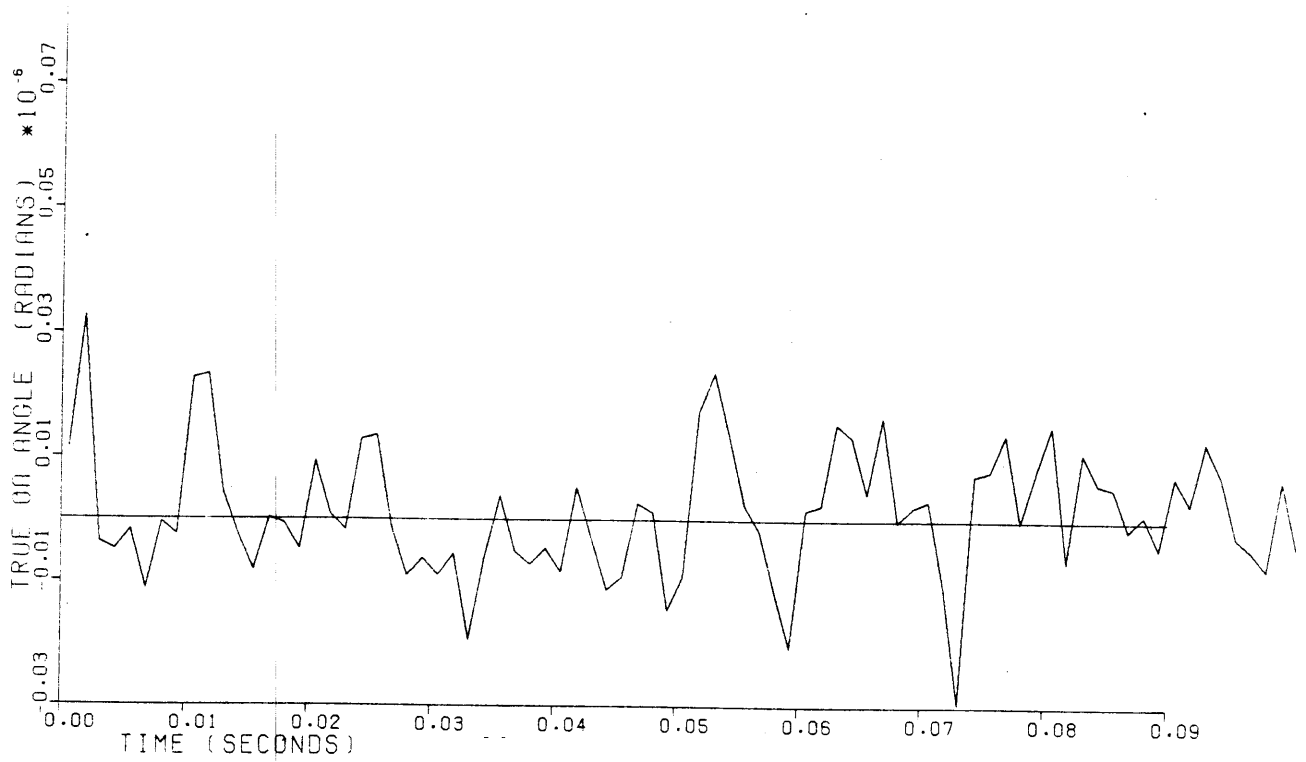


Figure 15-7. Run #26 24 Pt. Average Position vs. Time (7.5°/hr with S.G. Noise Level=1 nrad)

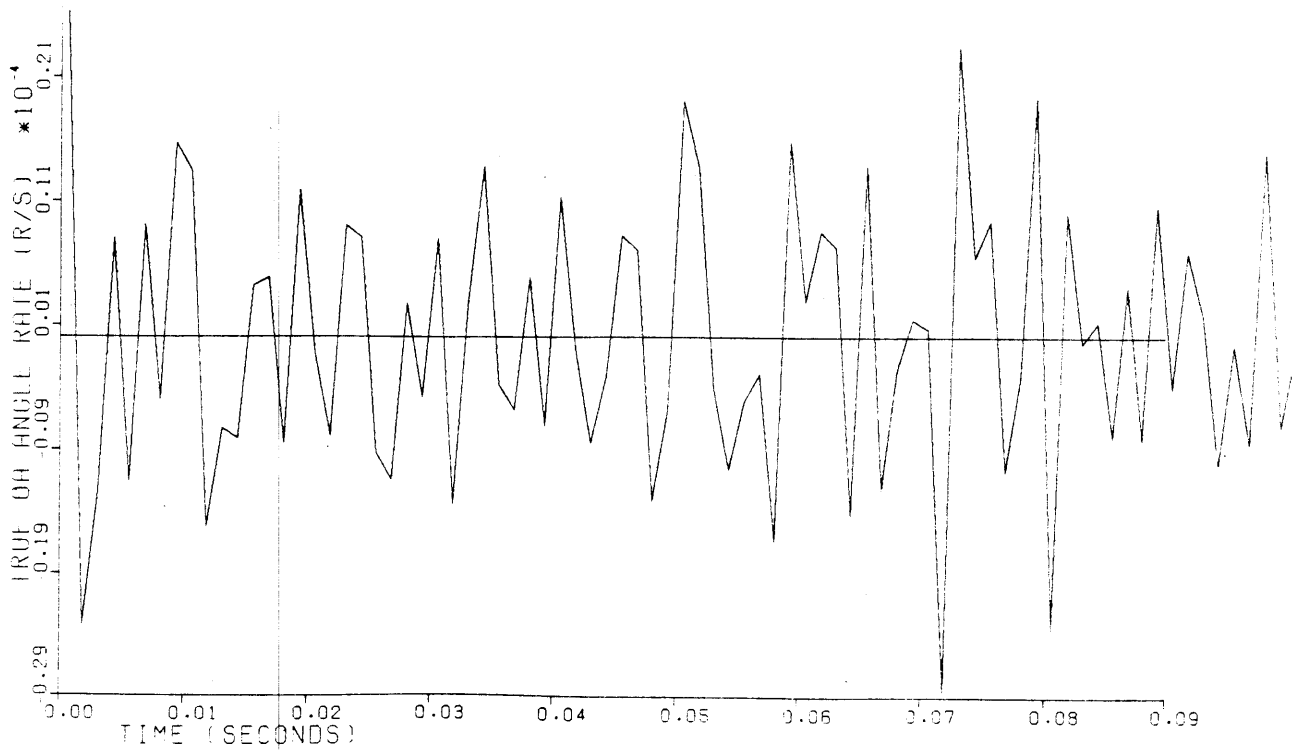


Figure 15-8. Run #26 24 Pt. Average Velocity vs. Time (7.5°/hr with S.G. Noise Level=1 nrad)

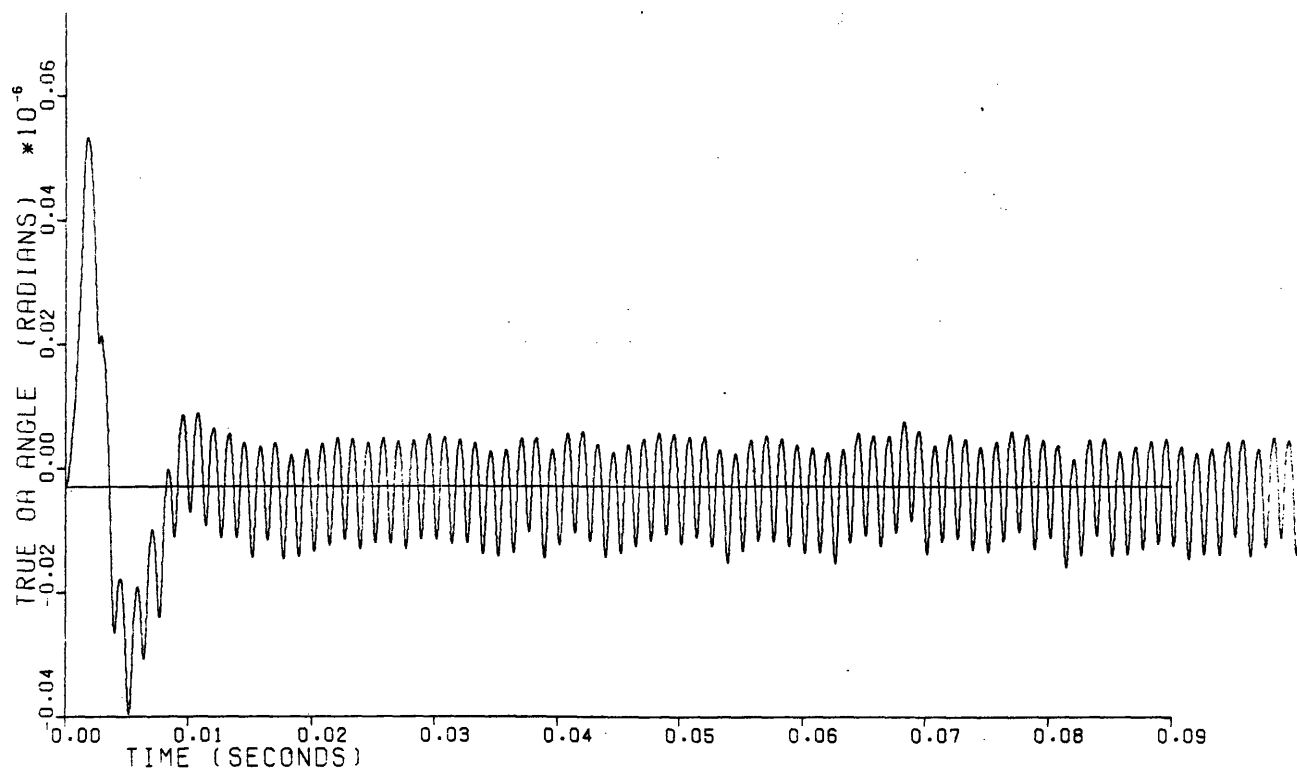


Figure 15-9. Run #27 Float Position vs. Time (7.5°/hr with SG Noise Level=1 nrad)

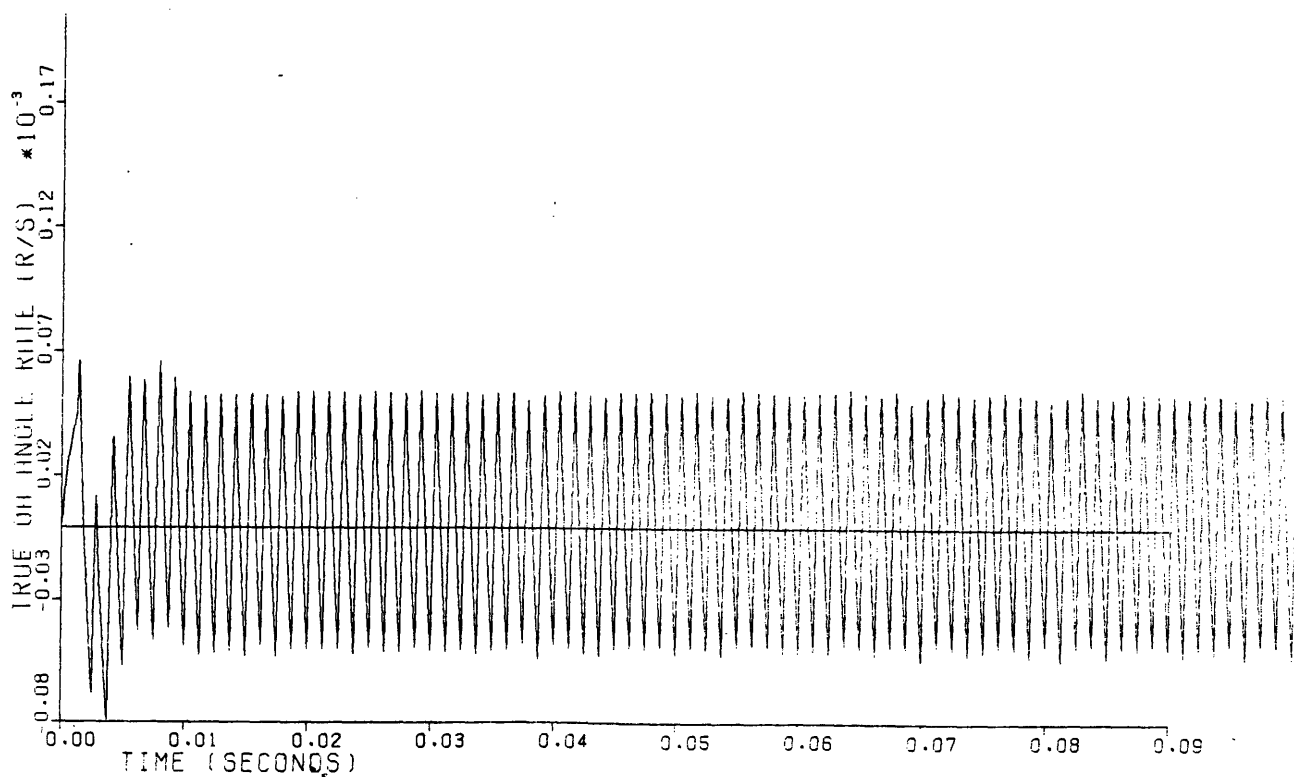


Figure 15-10. Run #27 Float Velocity vs Time ((7.5°/hr with SG Noise Level=1 nrad)

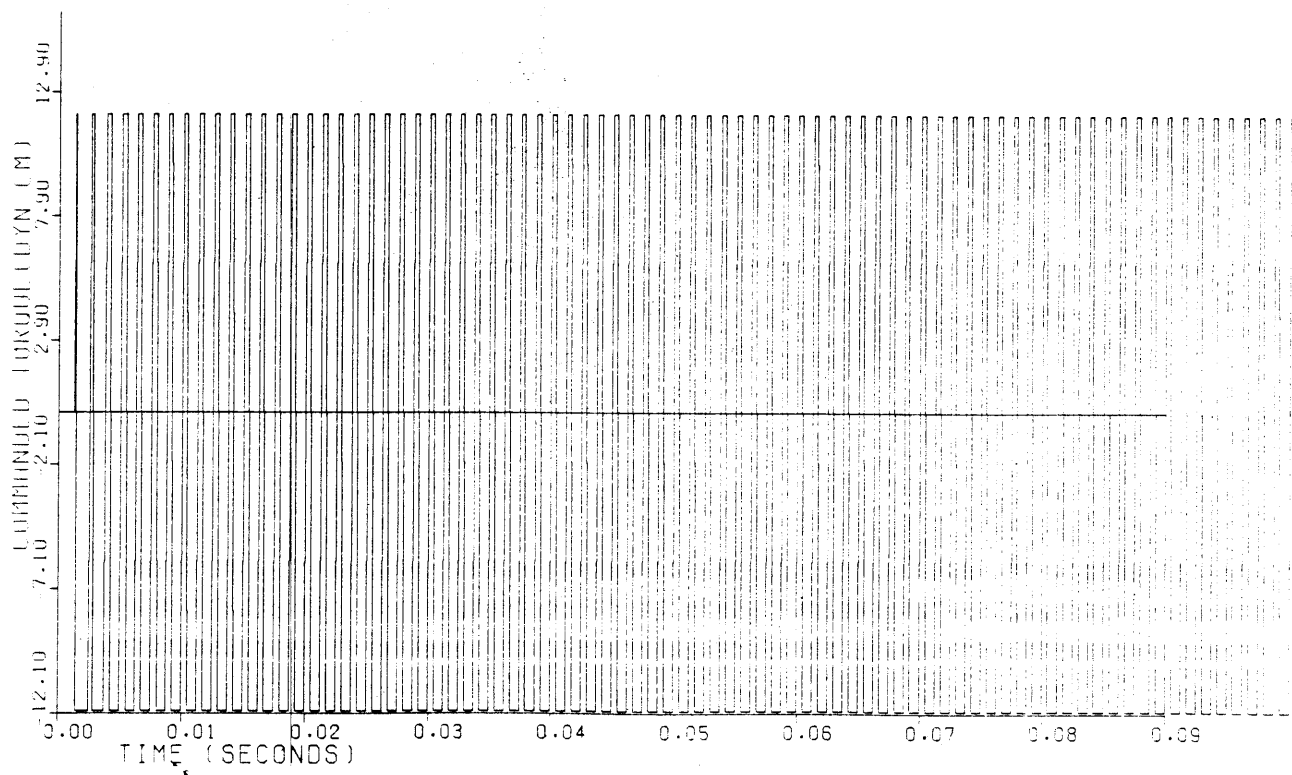


Figure 15-11. Run #27 Commanded Torque vs. Time (7.5°/hr with SG Noise Level=1 nrad)

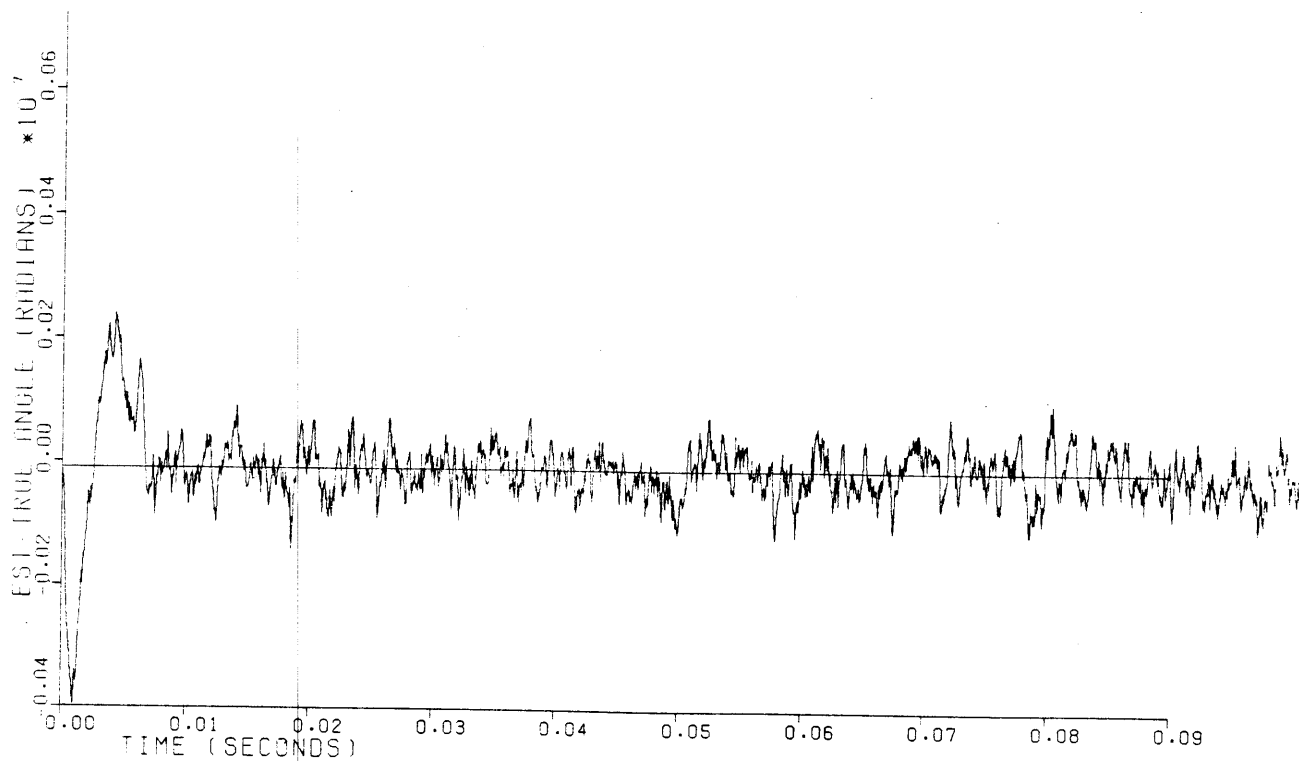


Figure 15-12. Run #27 Position Estimation Error vs. Time (7.5°/hr with SG Noise Level=1 nrad)

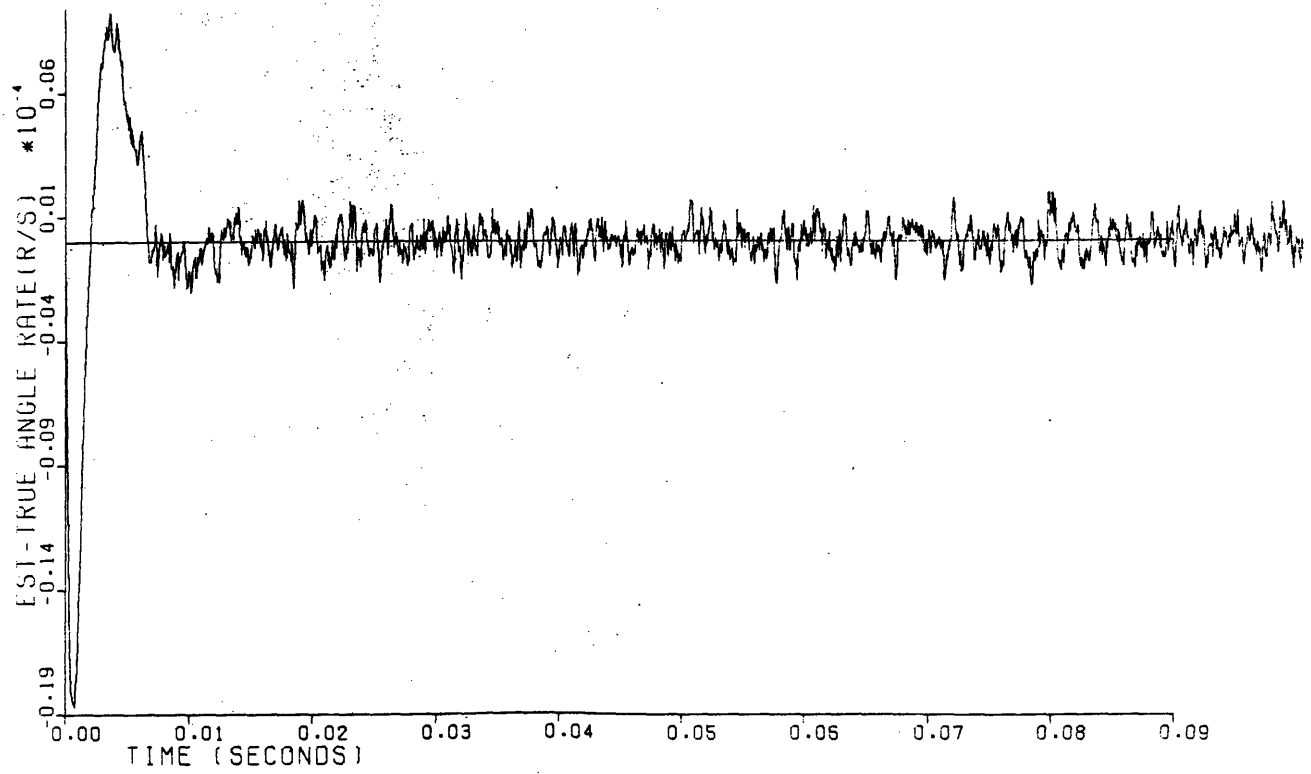


Figure 15-13. Run #27 Velocity Estimation Error vs. Time (7.5°/hr with SG Noise Level=1 nrad)

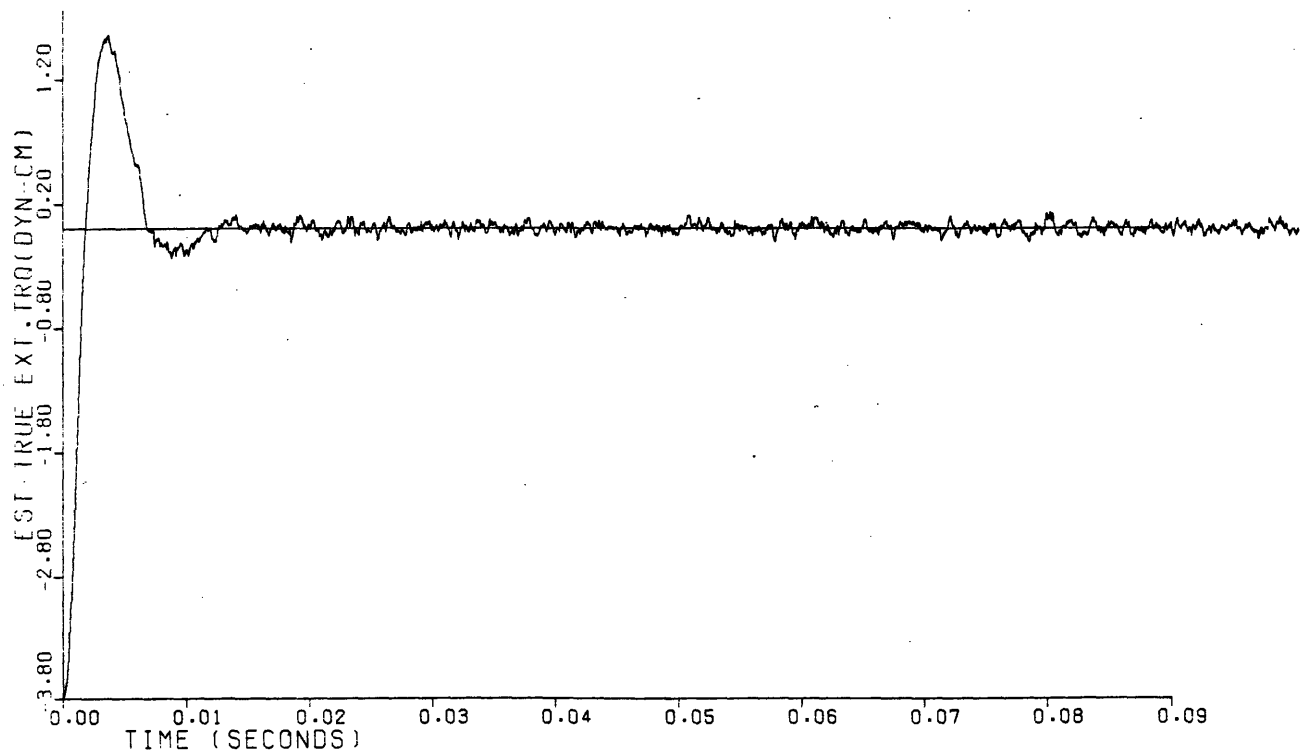


Figure 15-14. Run #27 External Torque Estimation Error vs. Time (7.5°/hr with SG Noise Level=1 nrad)

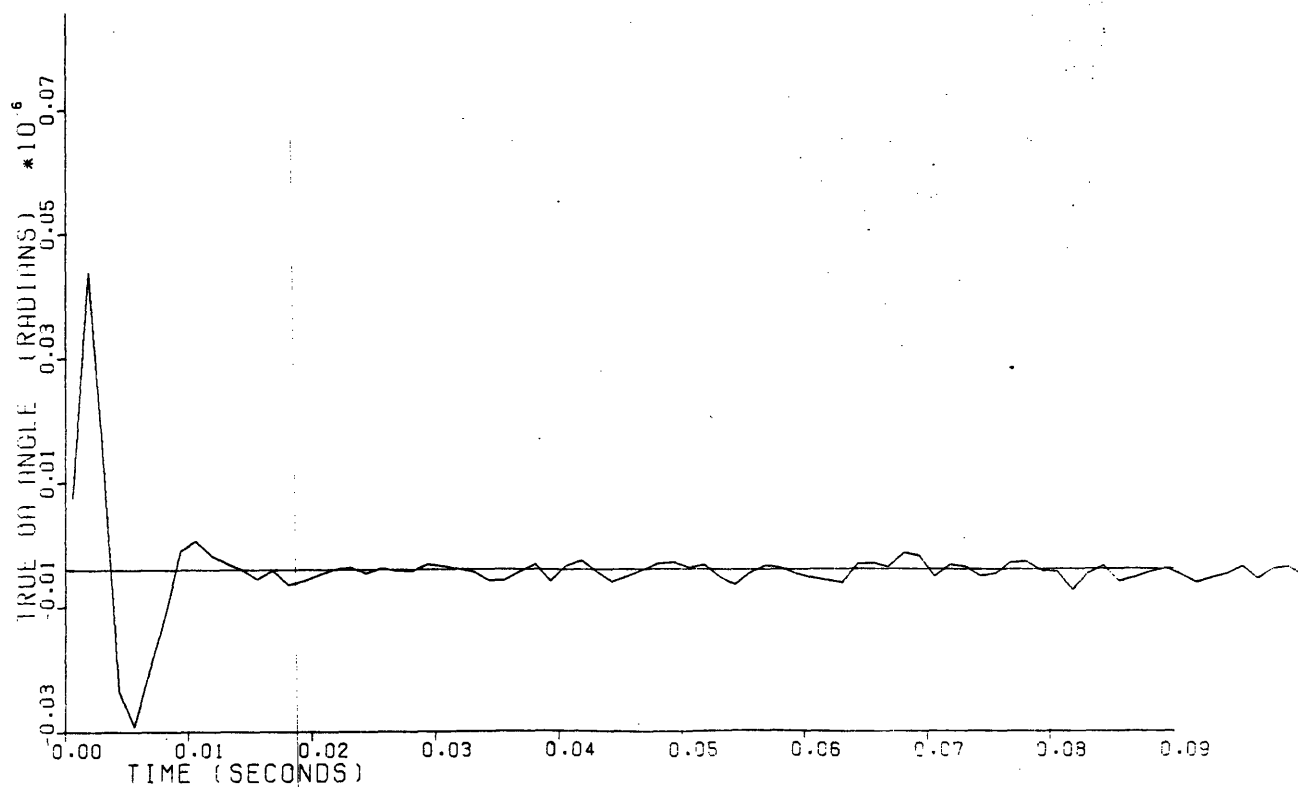


Figure 15-15. Run #27 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=1 nrad)

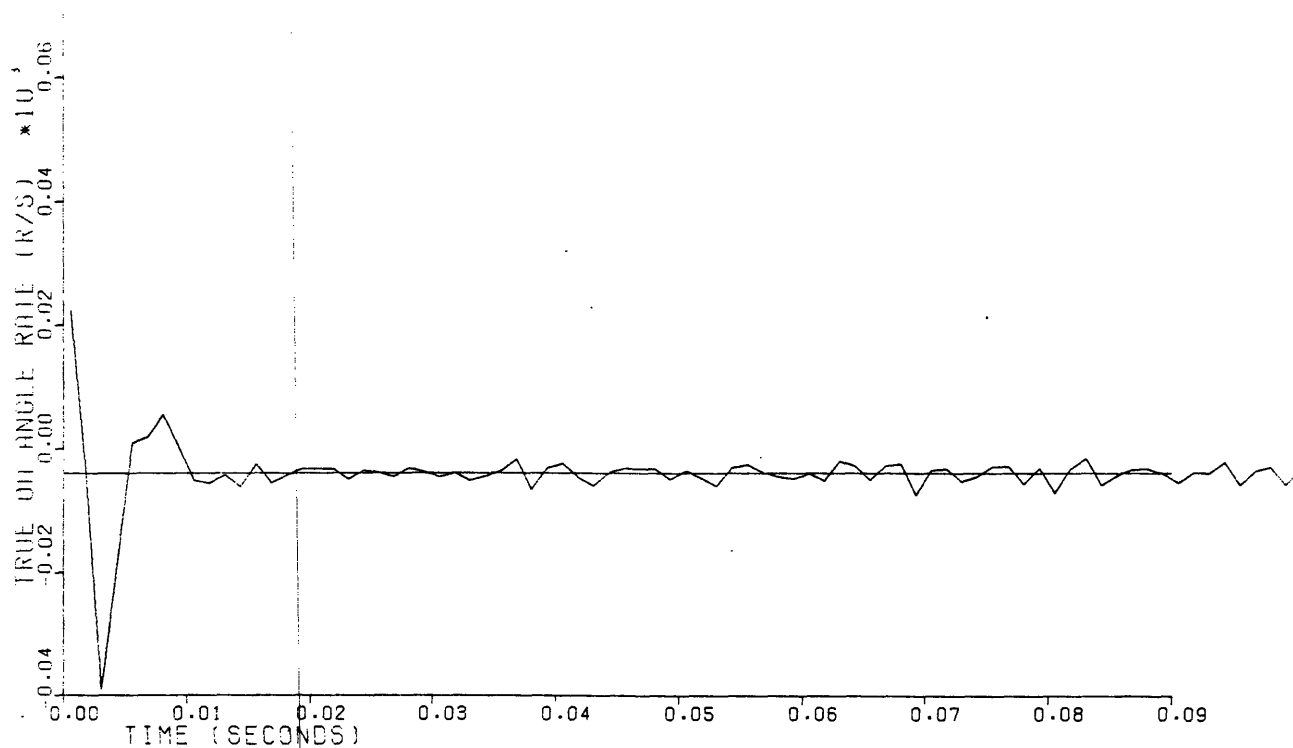


Figure 15-16. Run #27 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=1 nrad)

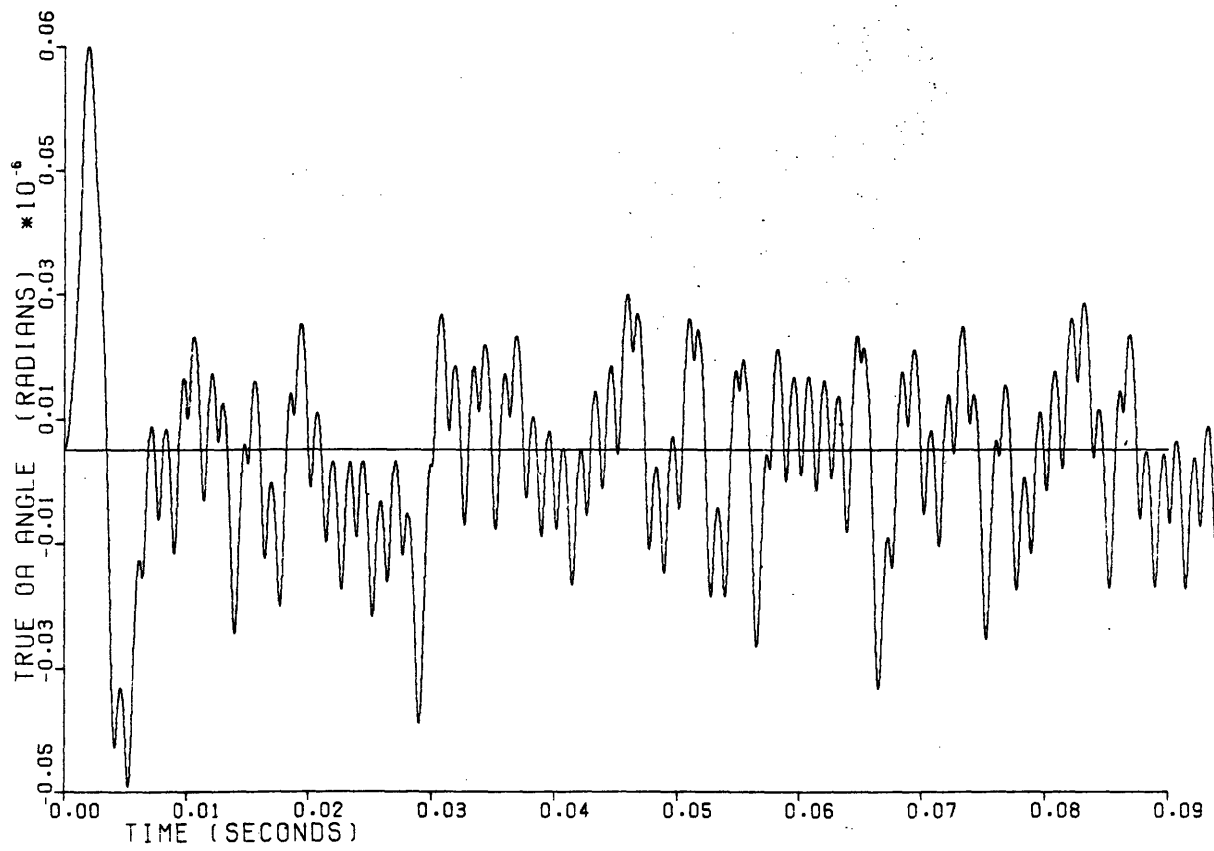


Figure 15-17. Run #28 Float Position vs. Time (7.5°/hr with SG Noise Level=10 nrad)

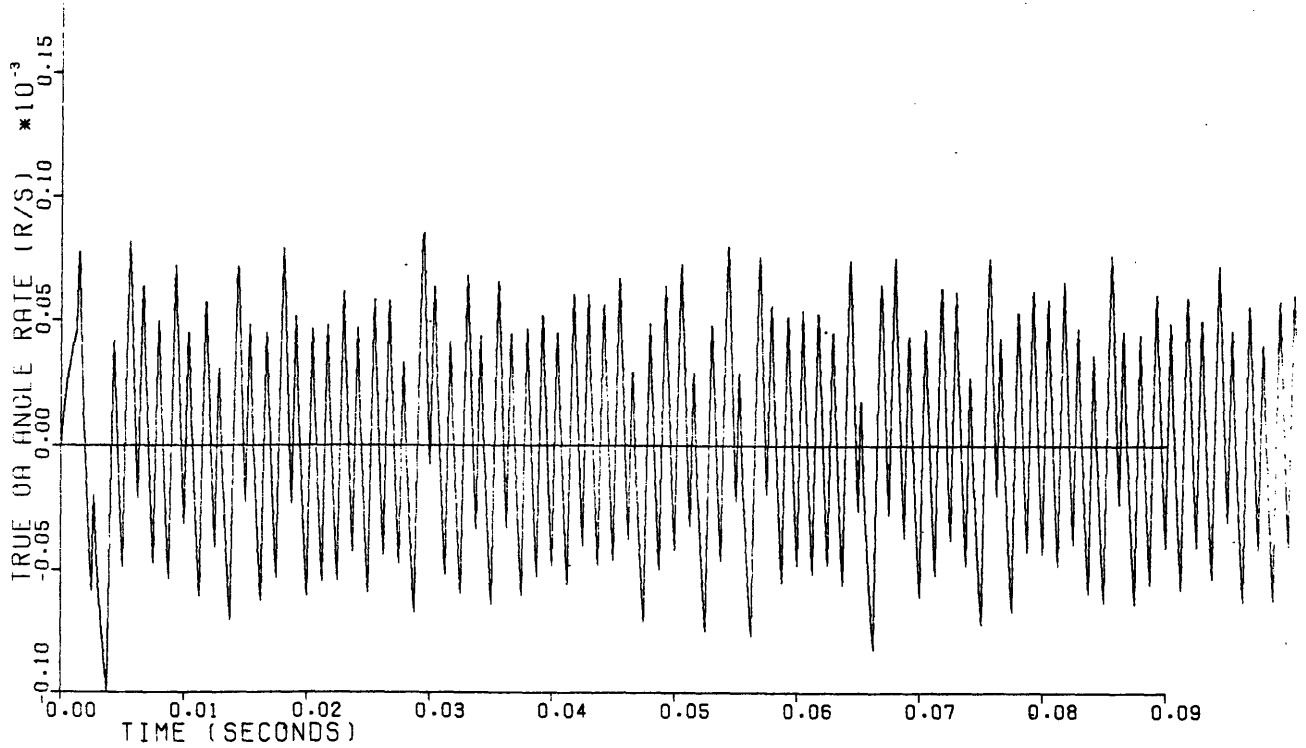


Figure 15-18. Run #28 Float Velocity vs Time (7.5°/hr with SG Noise Level=10 nrad)

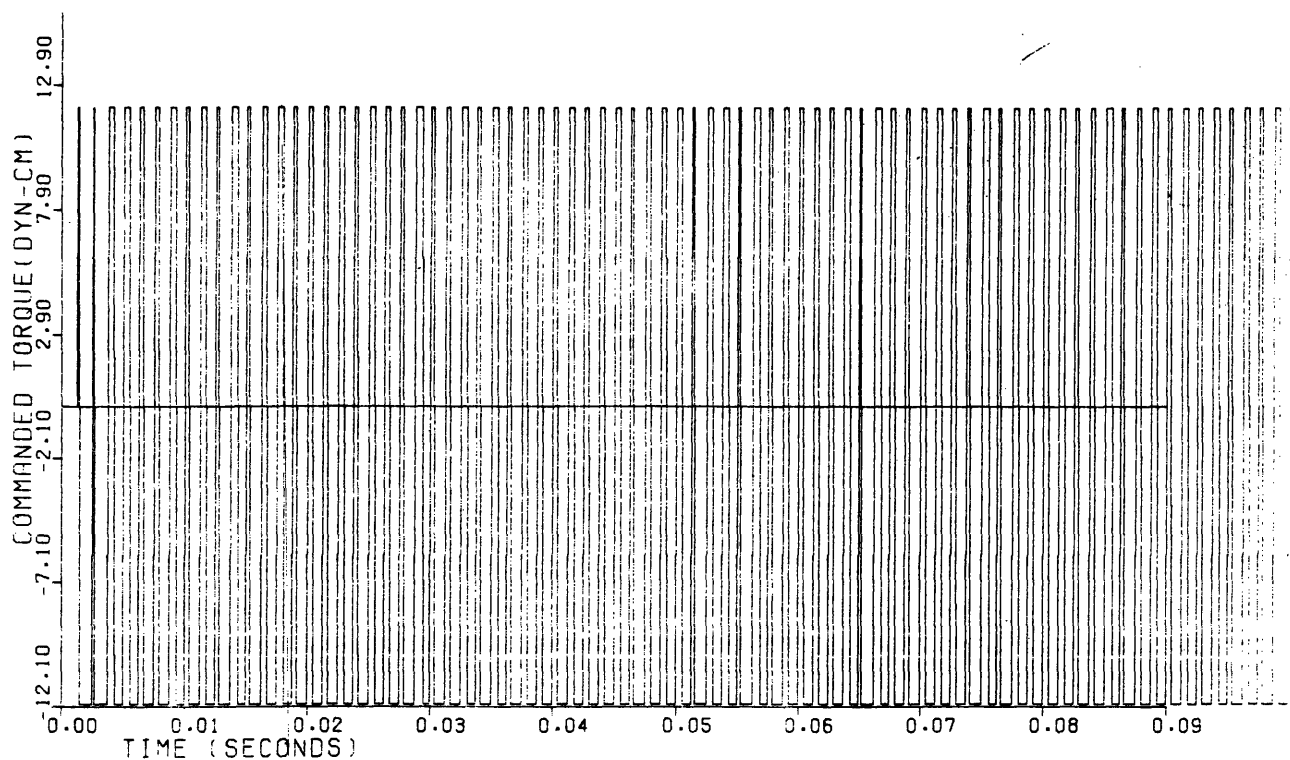


Figure 15-19. Run #28 Commanded Torque vs. Time (7.5°/hr with SG Noise Level=10 nrad)

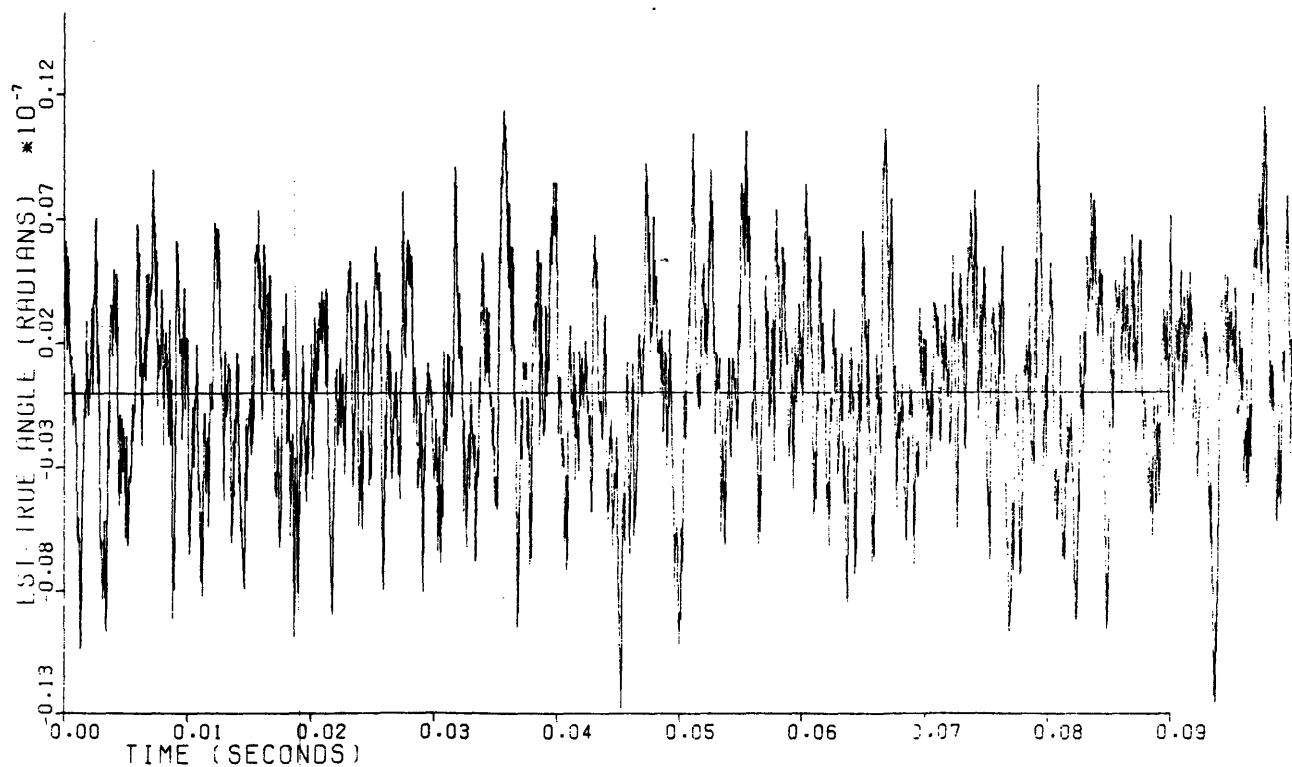


Figure 15-20. Run #28 Position Estimation Error vs. Time (7.5°/hr with SG Noise Level=10 nrad)

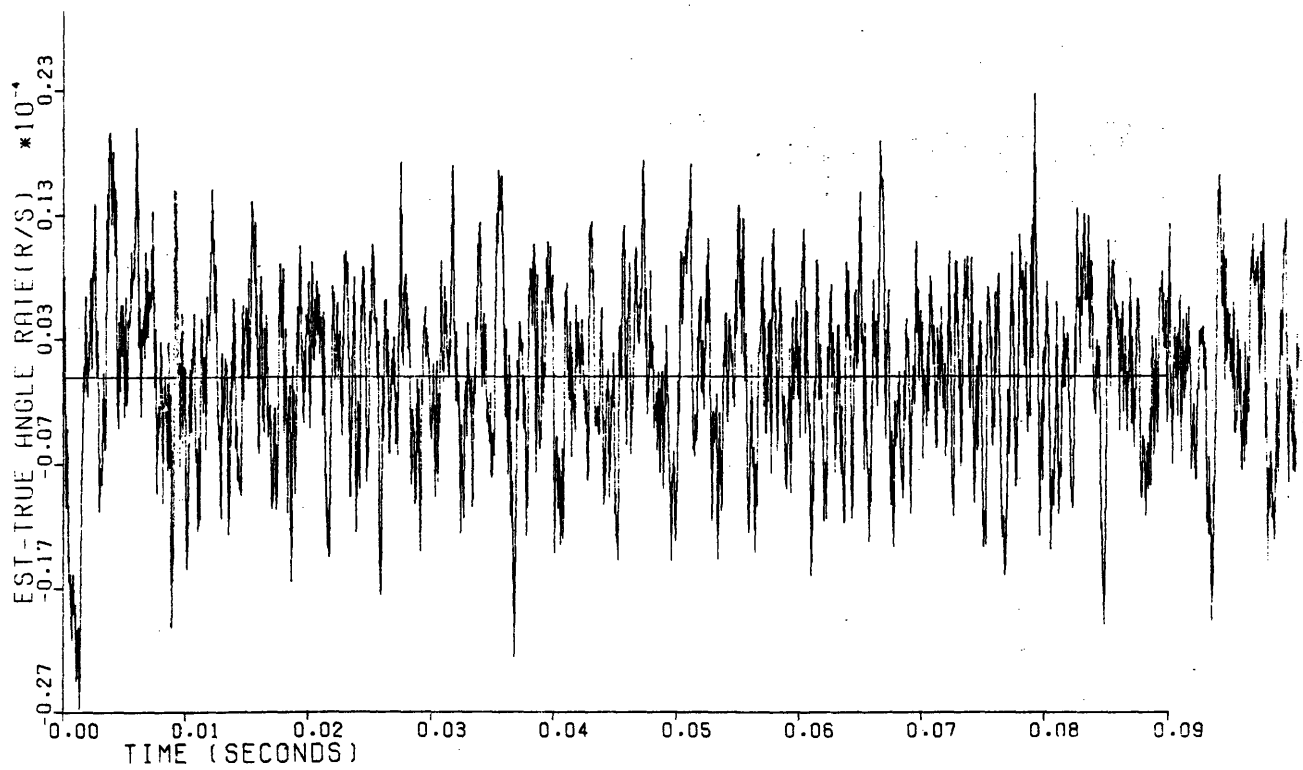


Figure 15-21. Run #28 Velocity Estimation Error vs. Time (7.5°/hr with SG Noise Level=10 nrad)

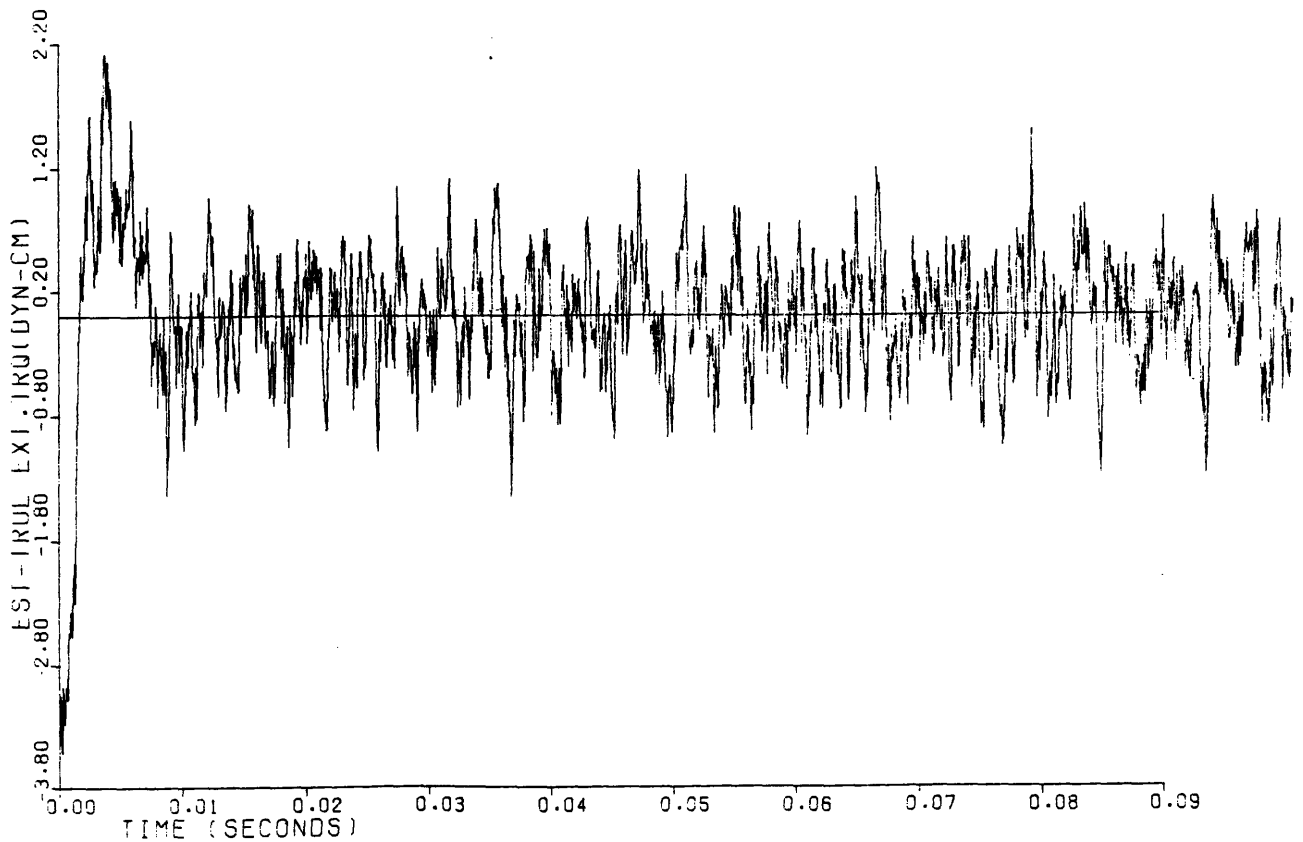


Figure 15-22. Run #28 External Torque Estimation Error vs. Time (7.5°/hr with SG Noise Level=10 m)

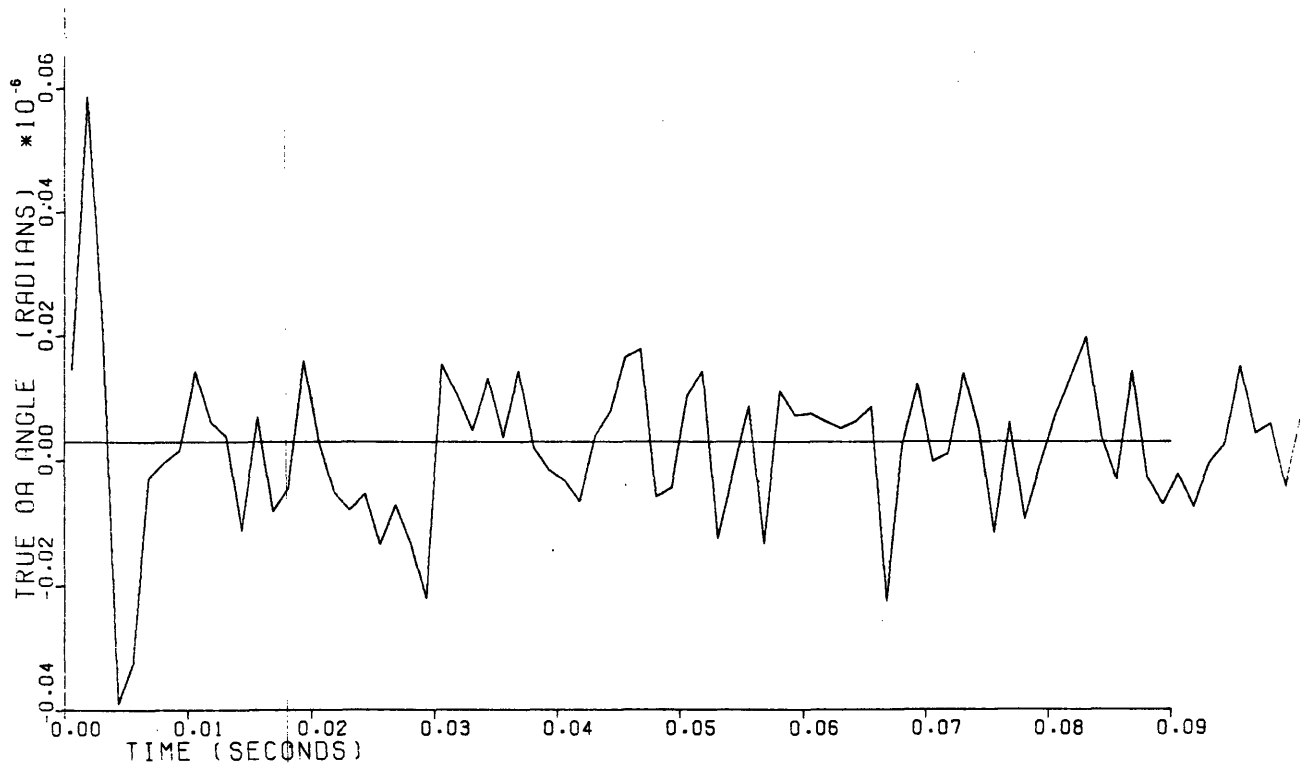


Figure 15-23. Run #28 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=10 nrad)

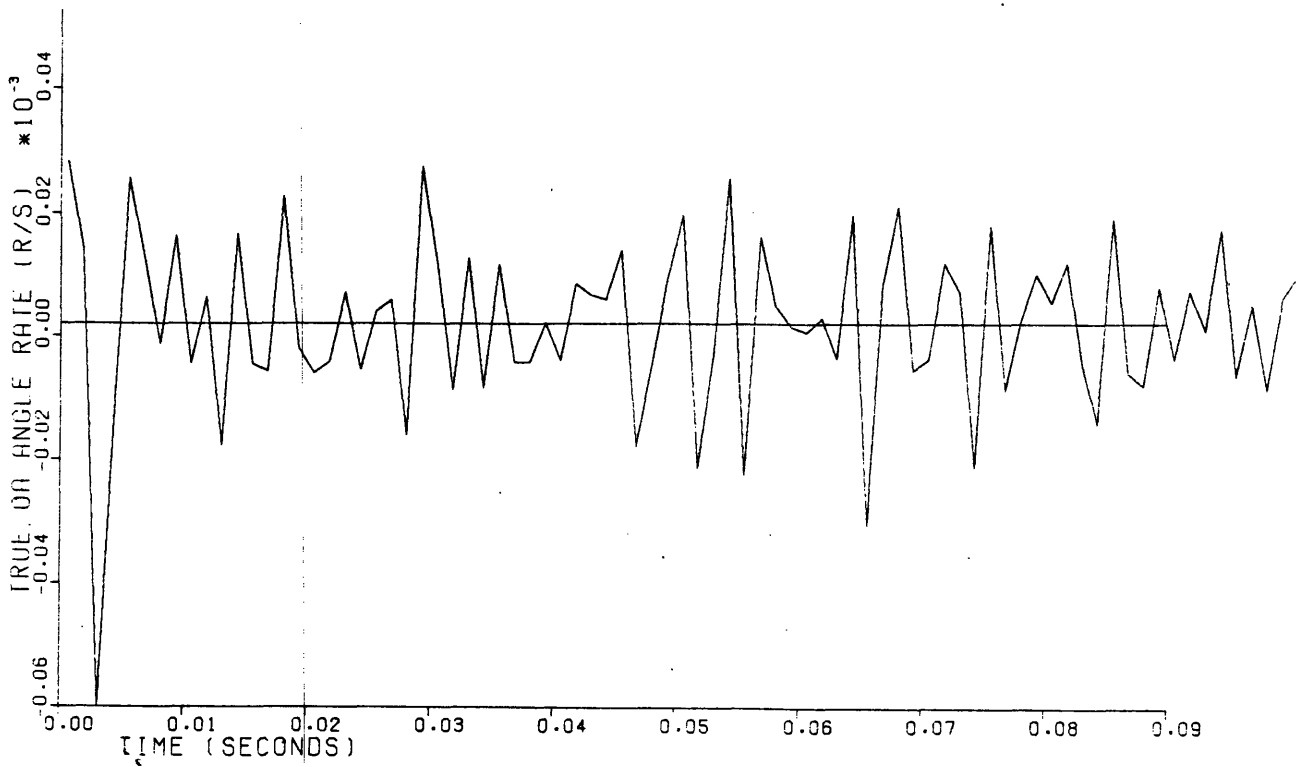


Figure 15-24. Run #28 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=10 nrad)

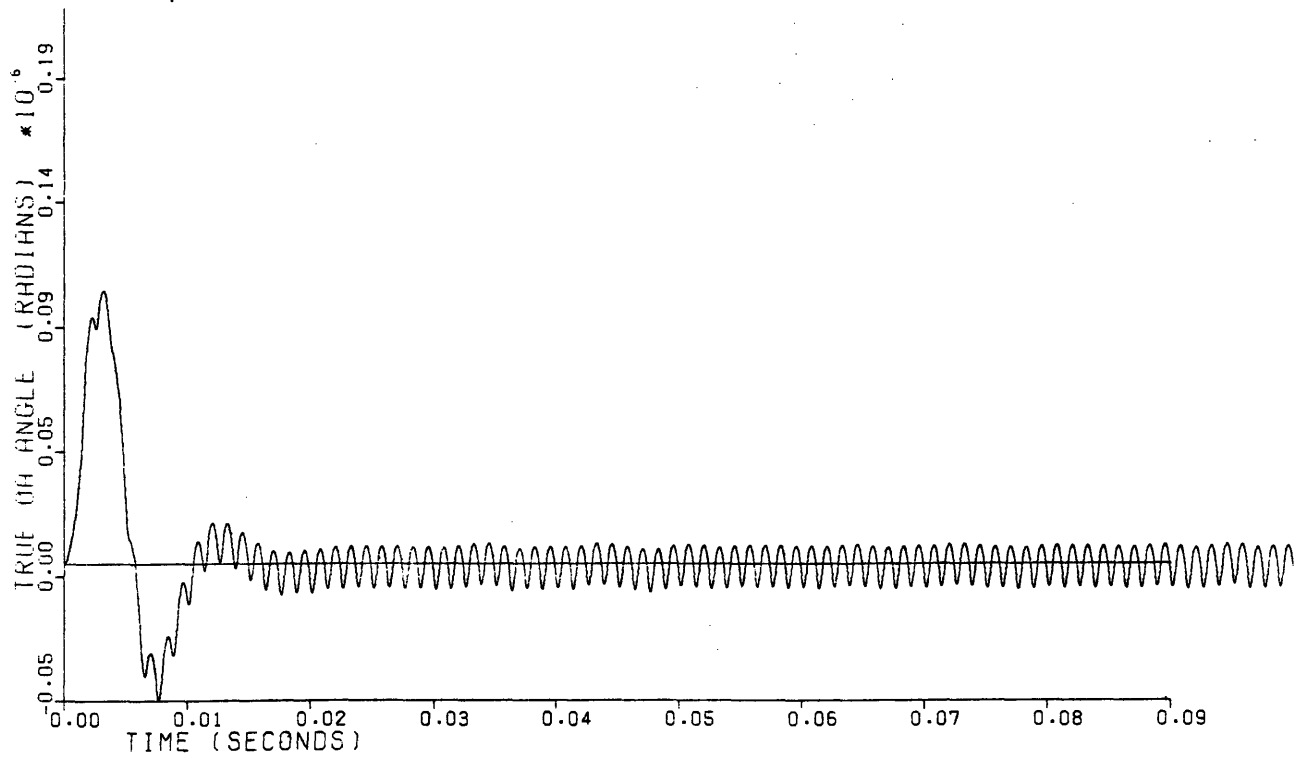


Figure 15-25. Run #29 Float Position vs. Time (7.5°/hr with SG Noise Level=1 nrad)

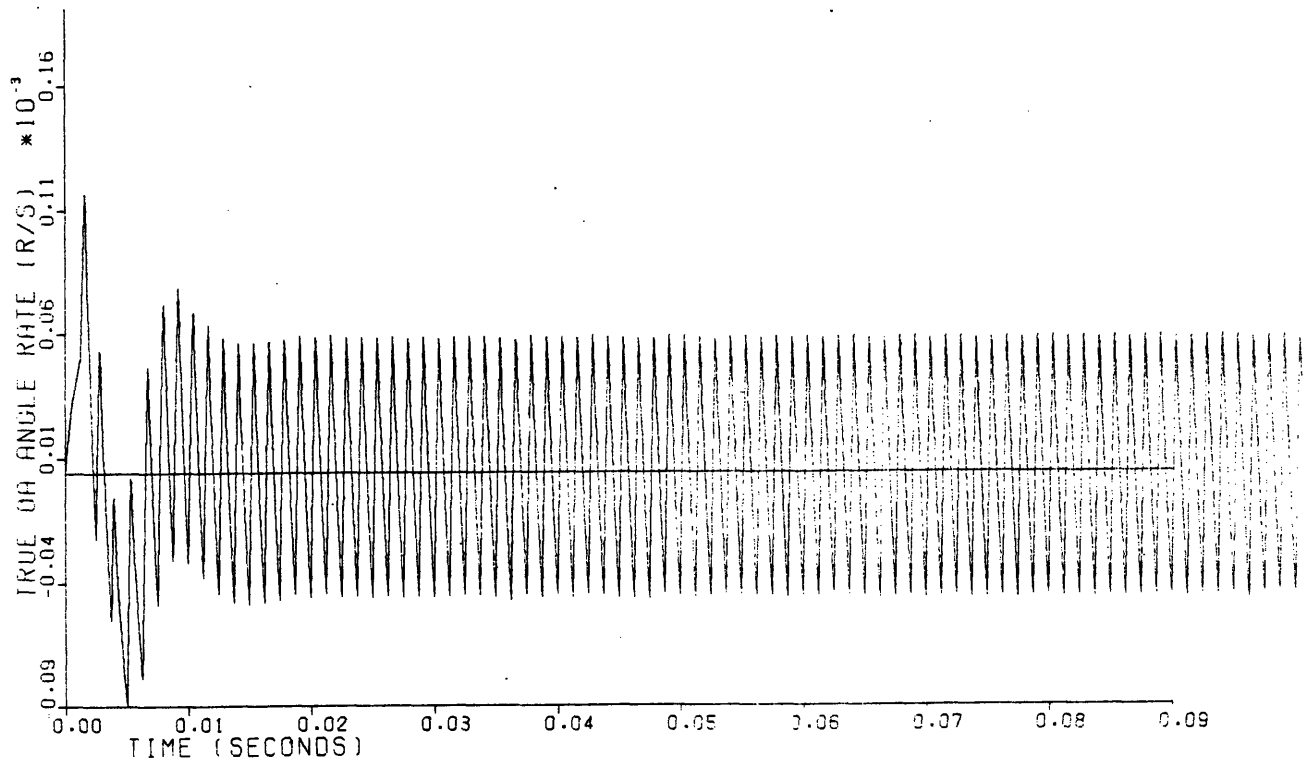


Figure 15-26. Run #29 Float Velocity vs. Time (7.5°/hr with SG Noise Level=1 nrad)

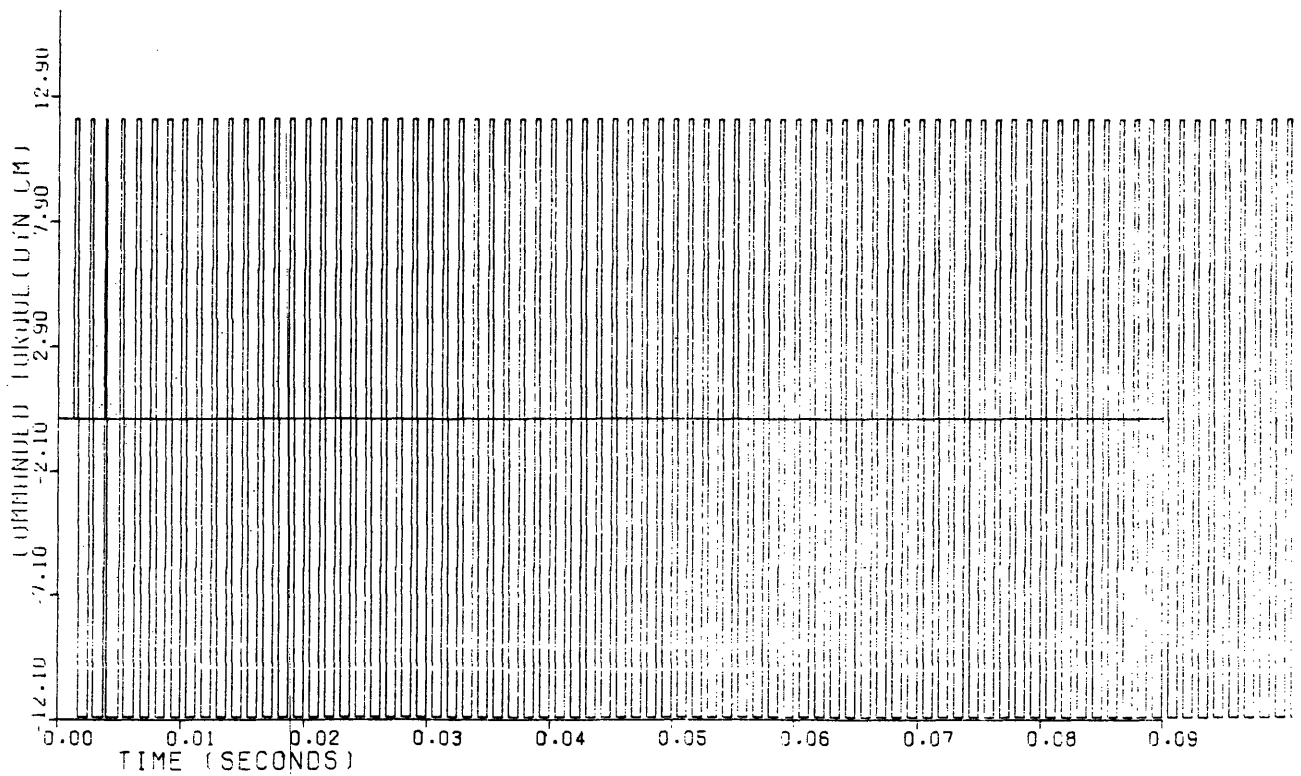


Figure 15-27. Run #29 Commanded Torque vs. Time (7.5°/hr with SG Noise Level=1 nrad)

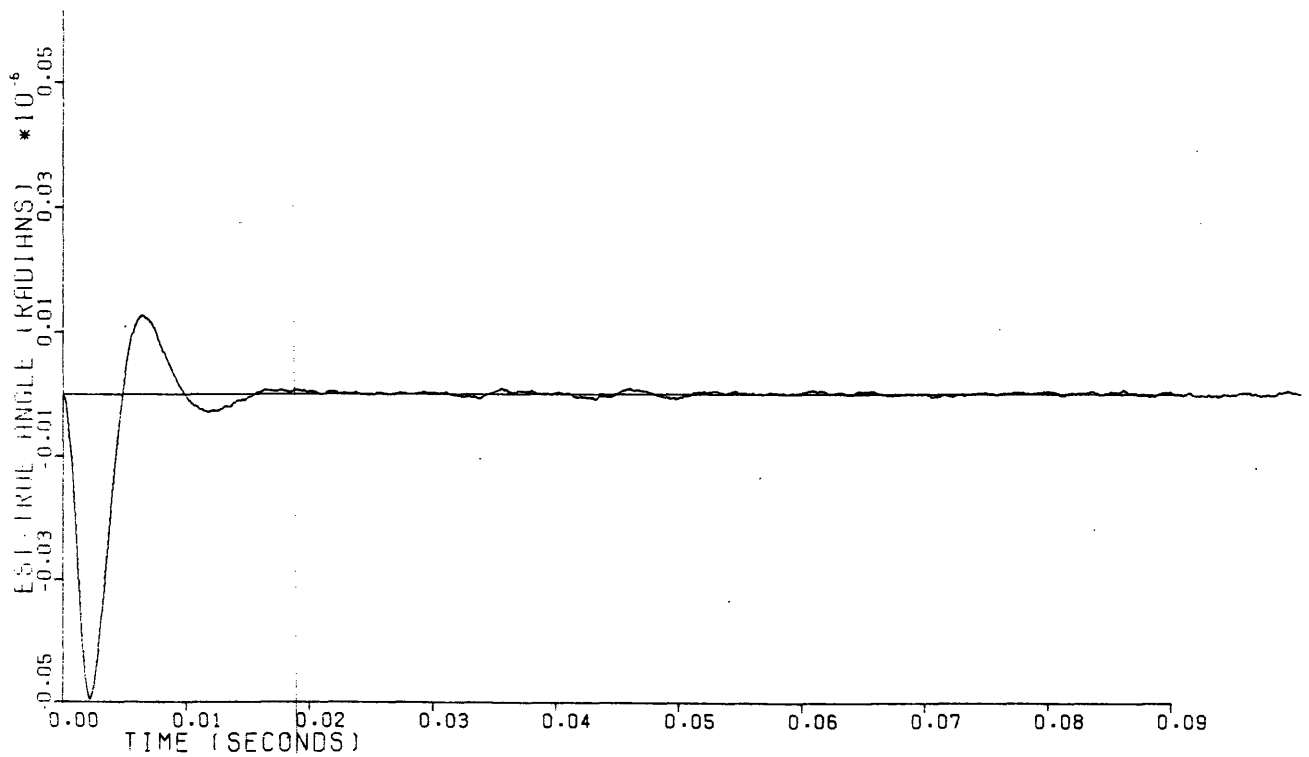


Figure 15-28. Run #29 Position Estimation Error vs. Time (7.5°/hr with SG Noise Level=1 nrad)

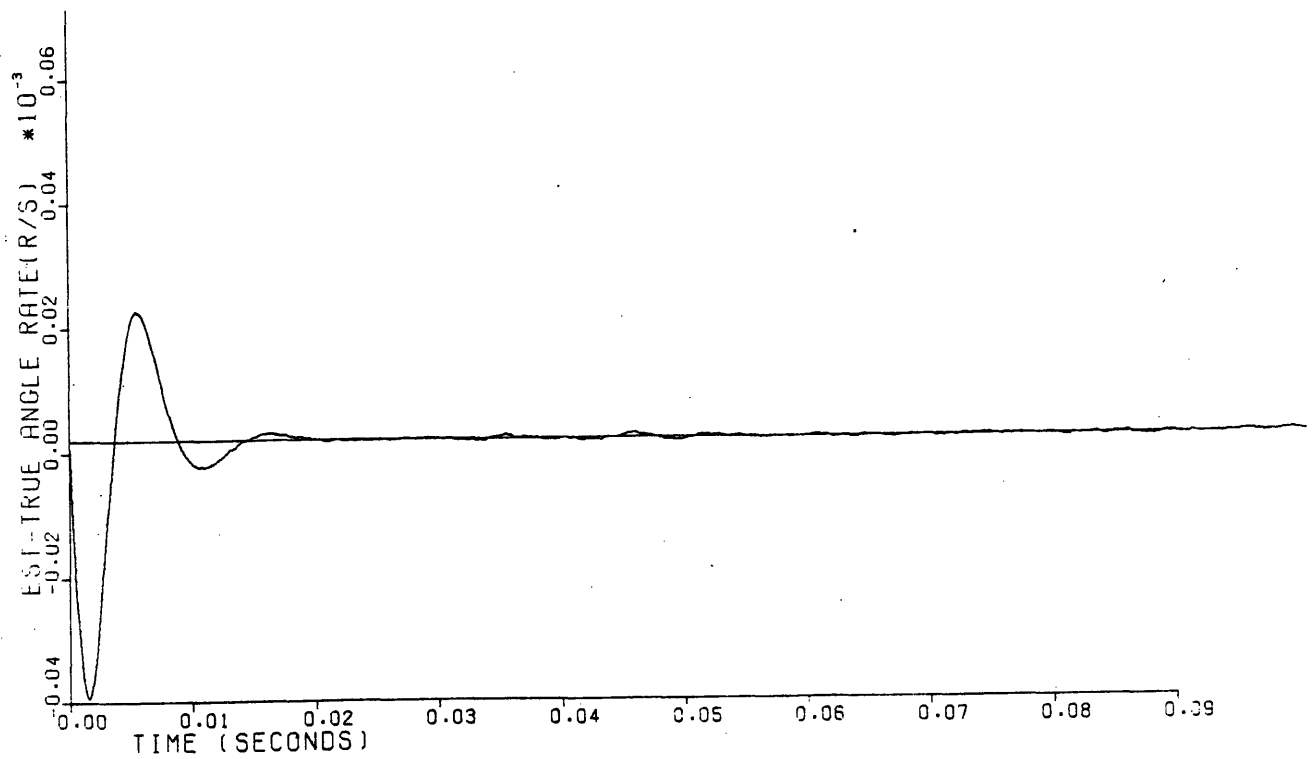


Figure 15-29. Run #29 Velocity Estimation Error vs. Time (7.5°/hr with SG Noise Level=1 nrad)

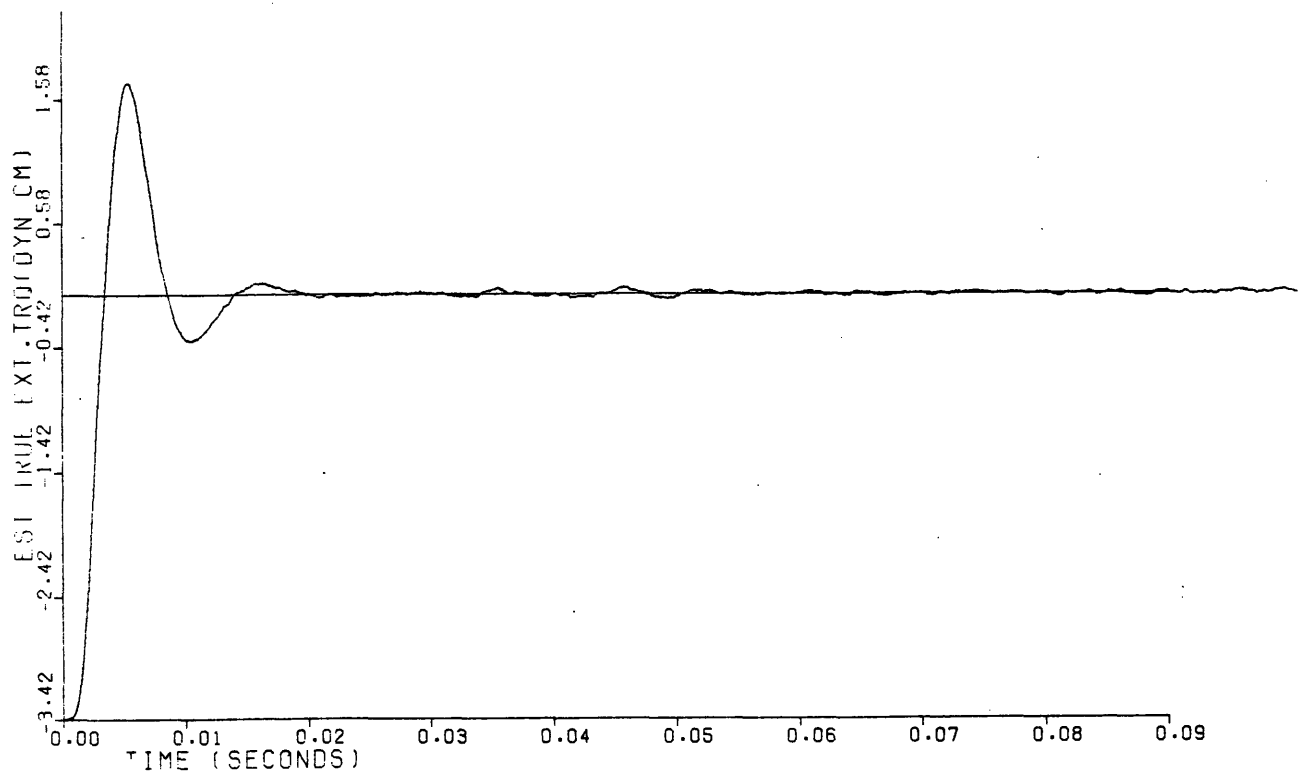


Figure 15-30. Run #29 External Torque Estimation Error vs. Time (7.5°/hr with SG Noise Level=1 nrad)

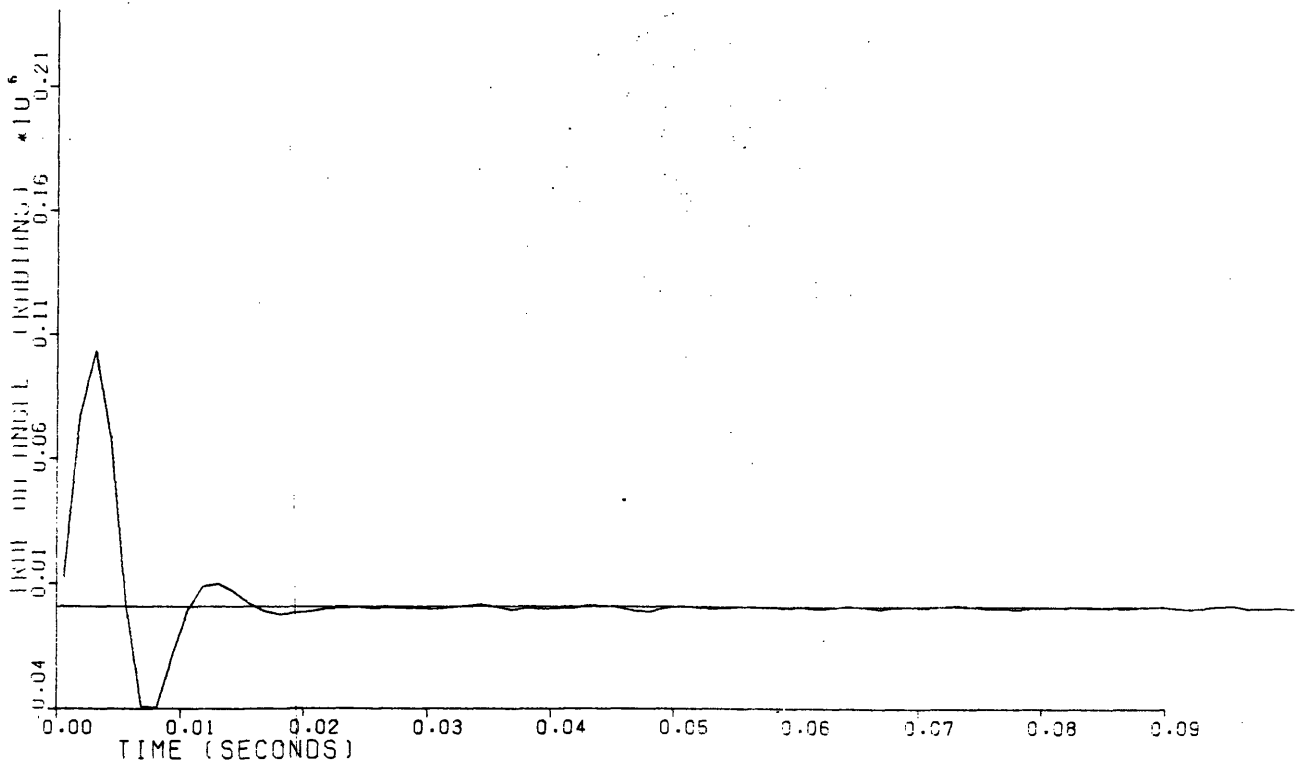


Figure 15-31. Run #29 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=1 nrad)

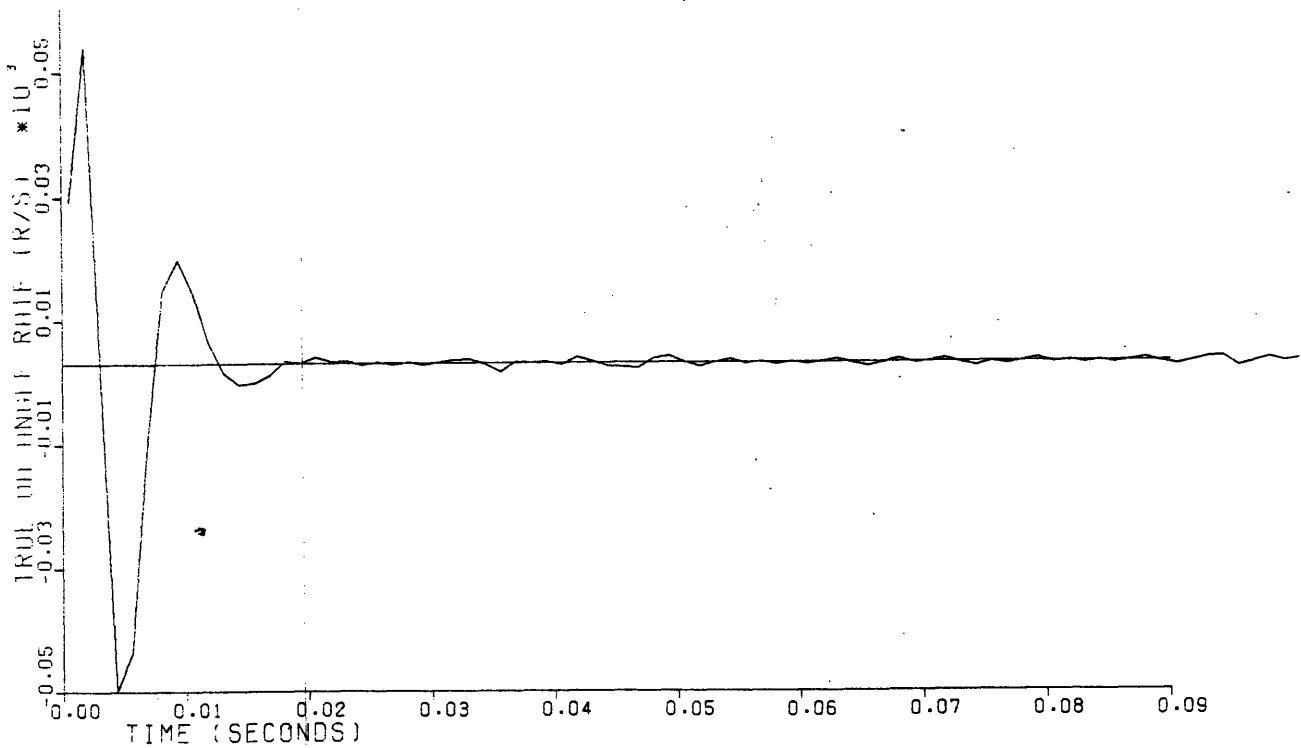


Figure 15-32. Run #29 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=1 nrad)

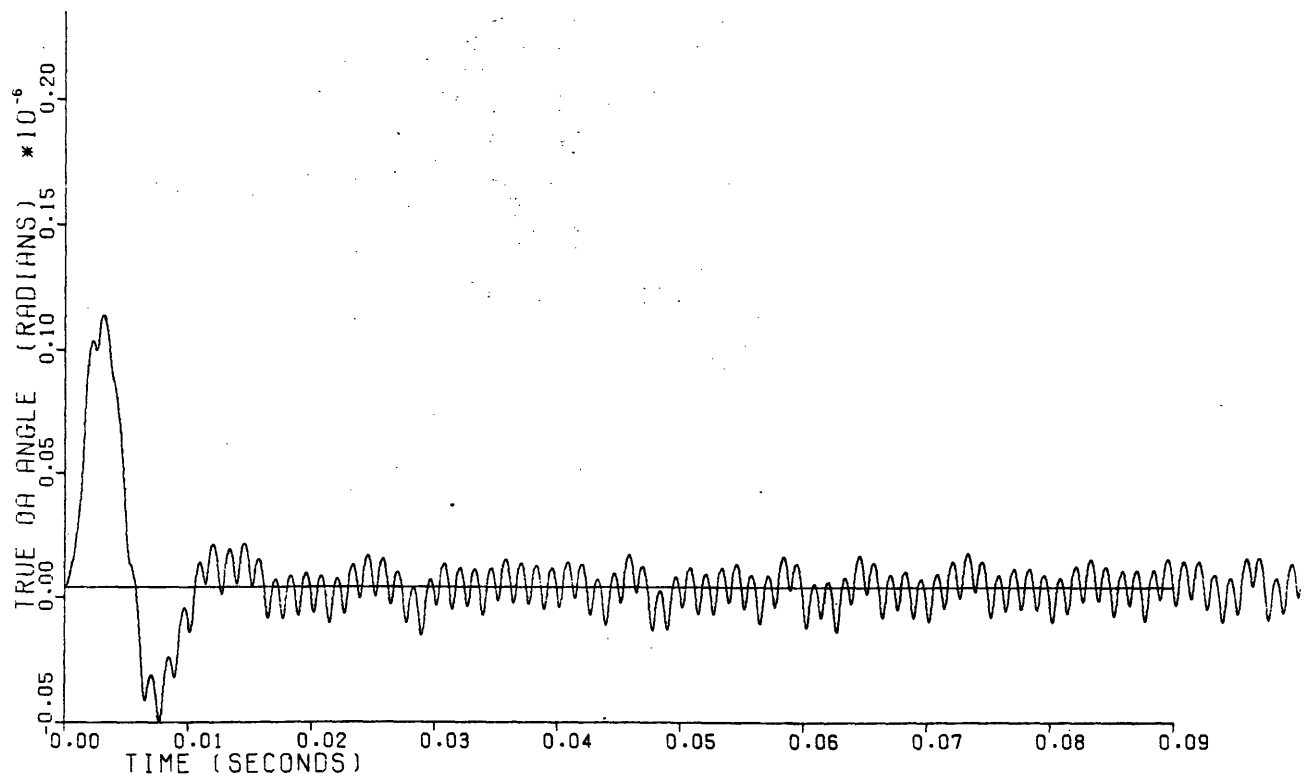


Figure 15-33. Run #30 Float Position vs. Time (7.5°/hr with SG Noise Level=10 nrad)

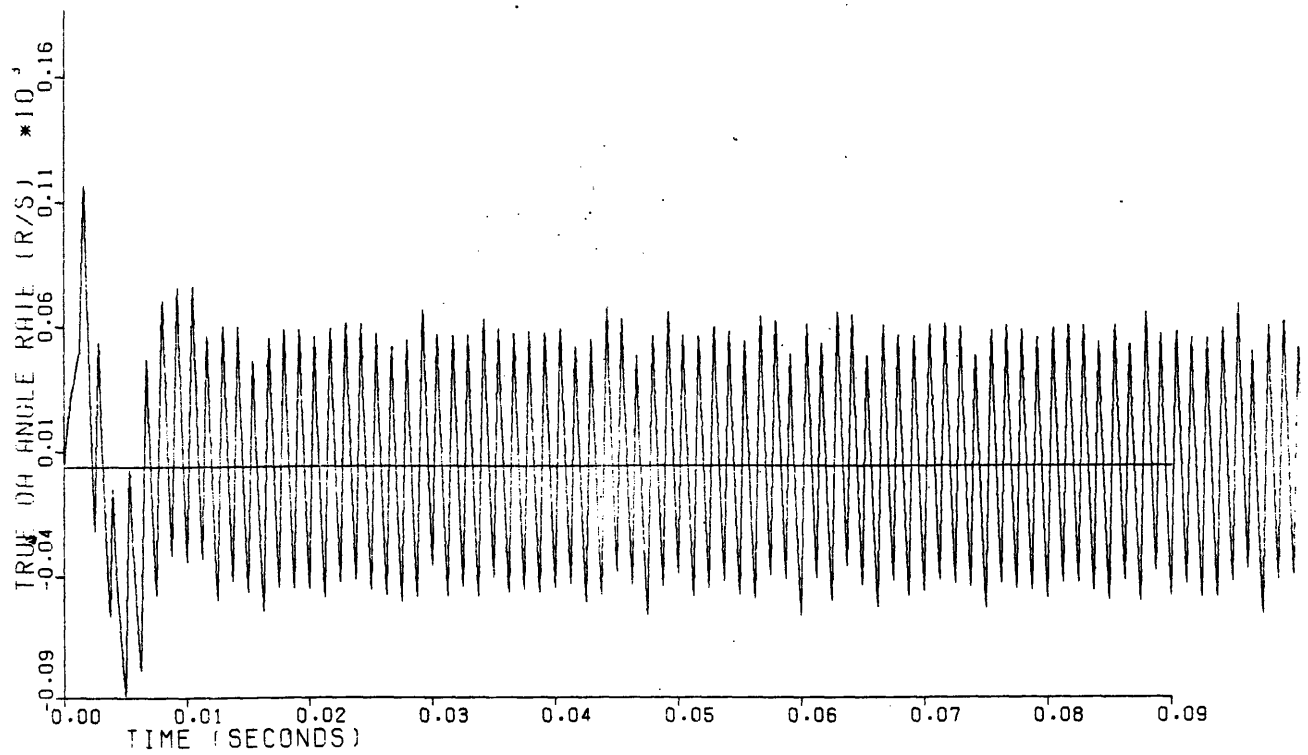


Figure 15-34. Run #30 Float Velocity vs Time (7.5°/hr with SG Noise Level=10 nrad)

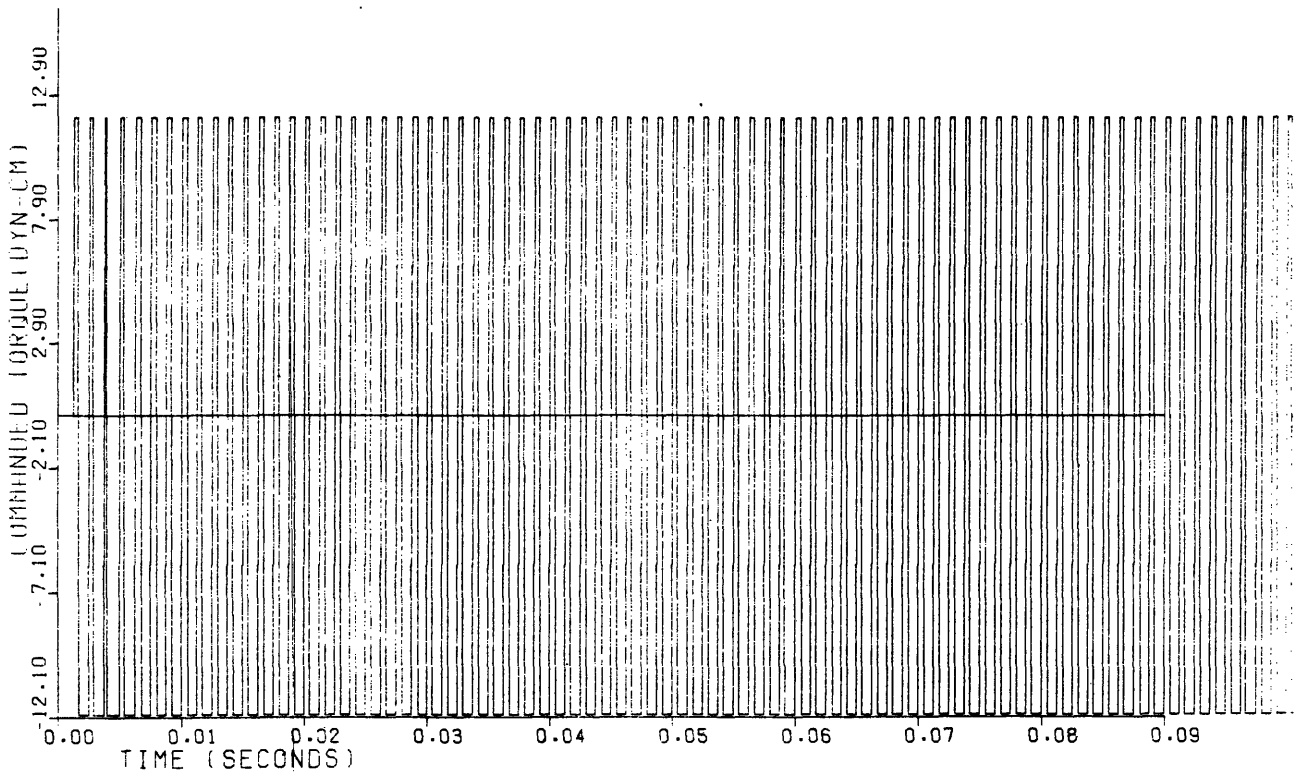


Figure 15-35. Run #30 Commanded Torque vs. Time ($7.5^\circ/\text{hr}$ with SG Noise Level=10 nrad)

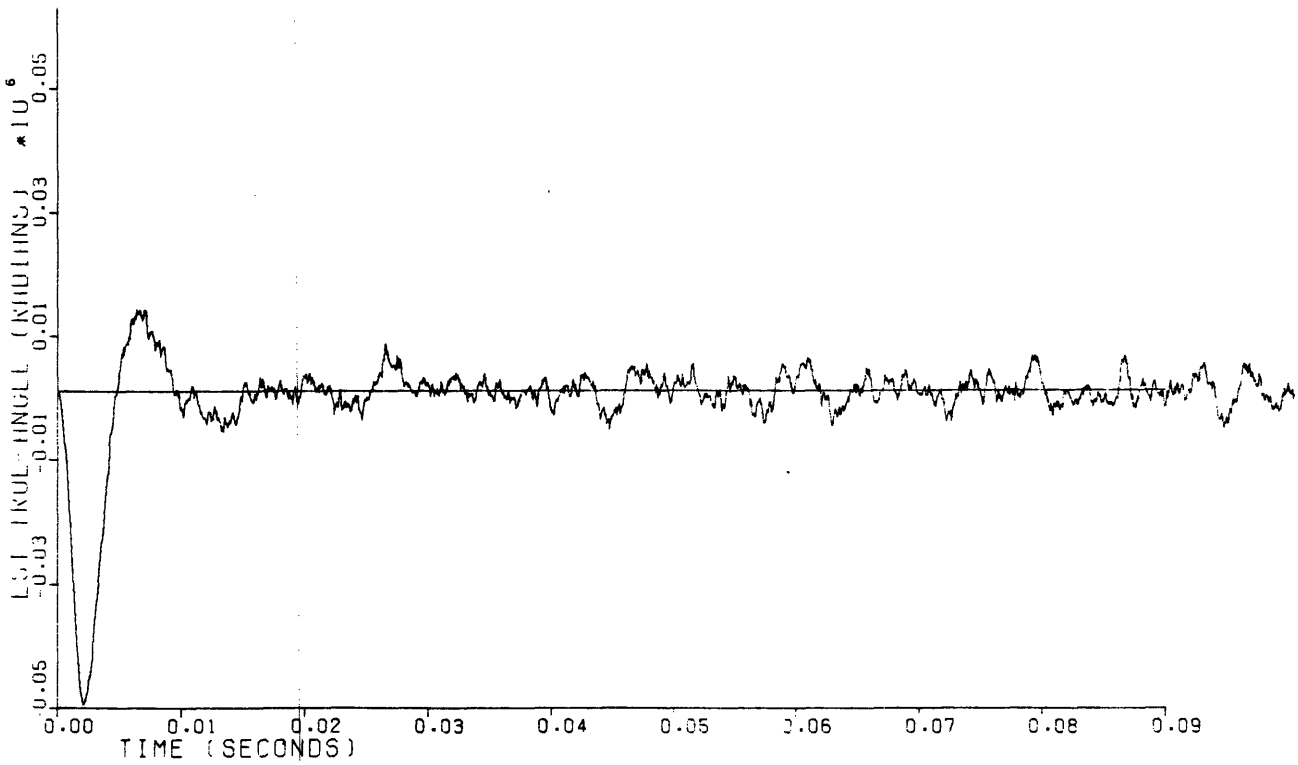


Figure 15-36. Run #30 Position Estimation Error vs. Time ($7.5^\circ/\text{hr}$ with SG Noise Level=10 nrad)

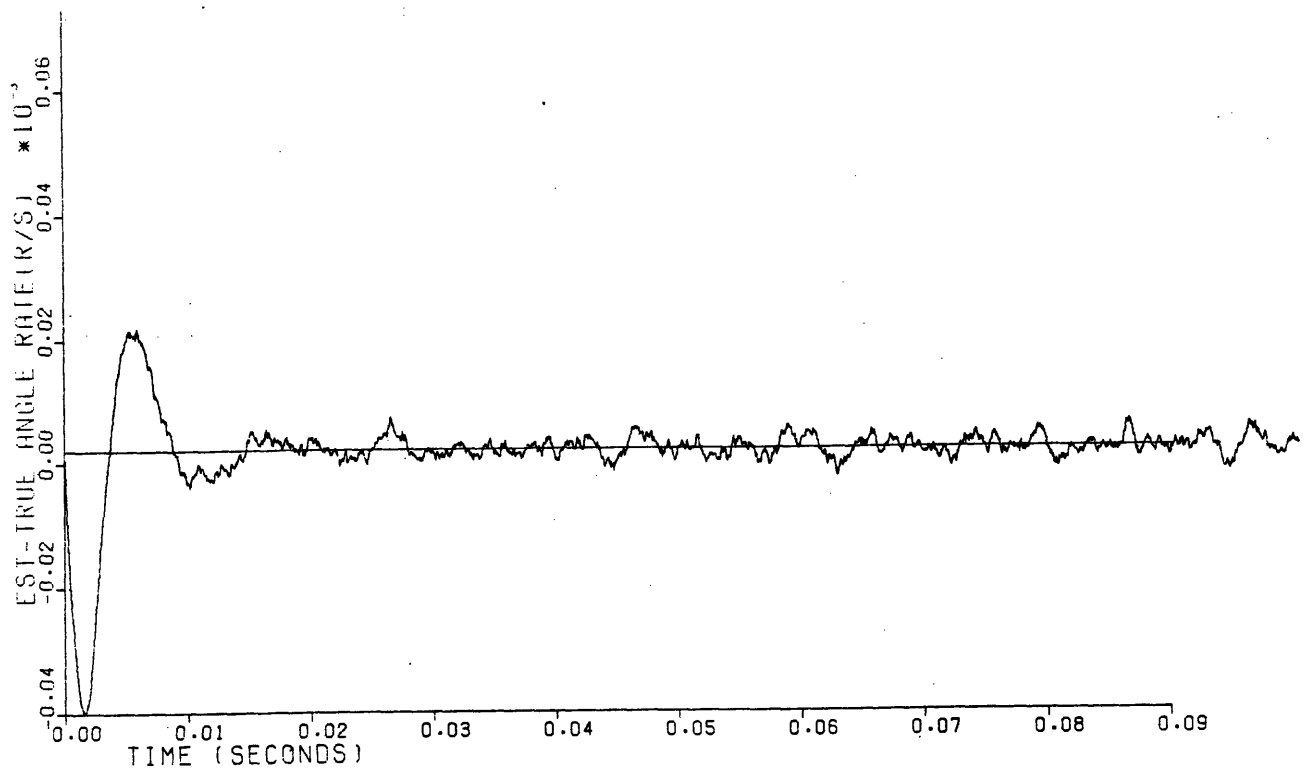


Figure 15-37. Run #30 Velocity Estimation Error vs. Time (7.5°/hr with SG Noise Level=10 nrad)

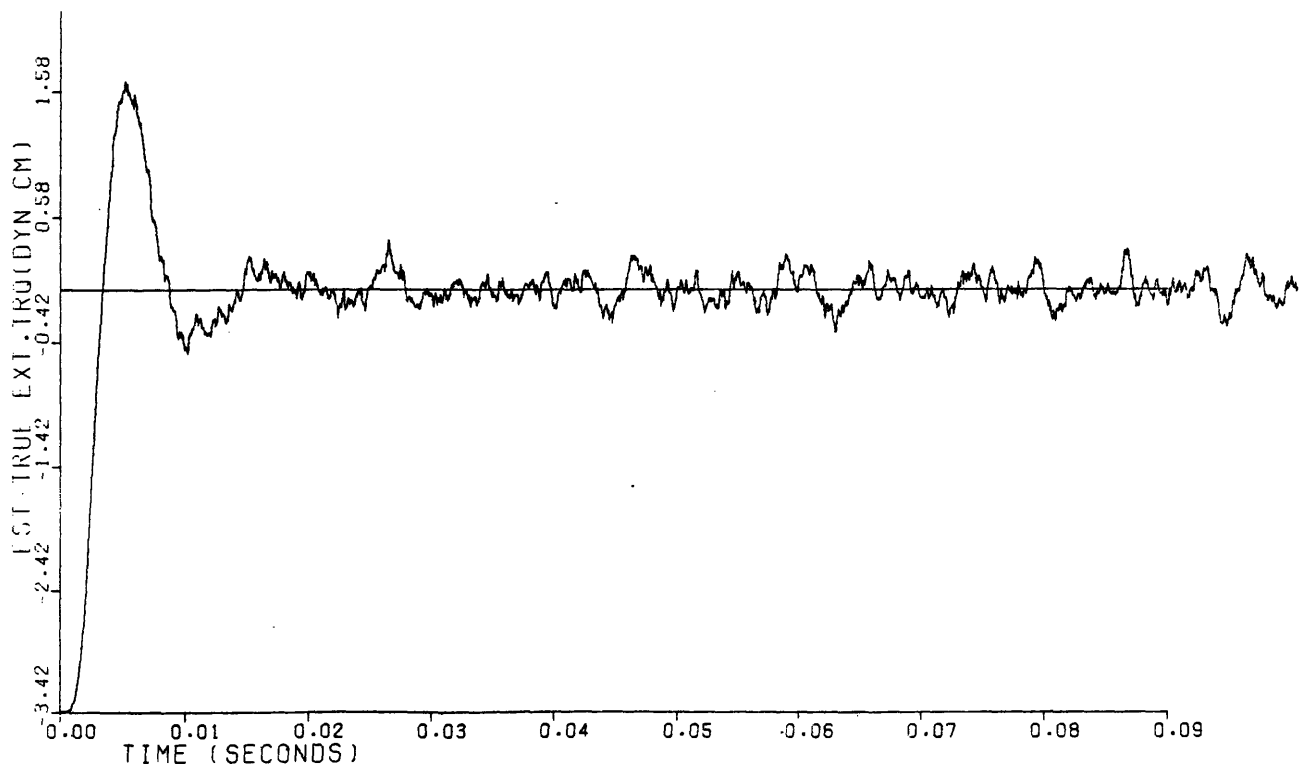


Figure 15-38. Run #30 External Torque Estimation Error vs. Time (7.5°/hr with SG Noise Level=10 nrad)

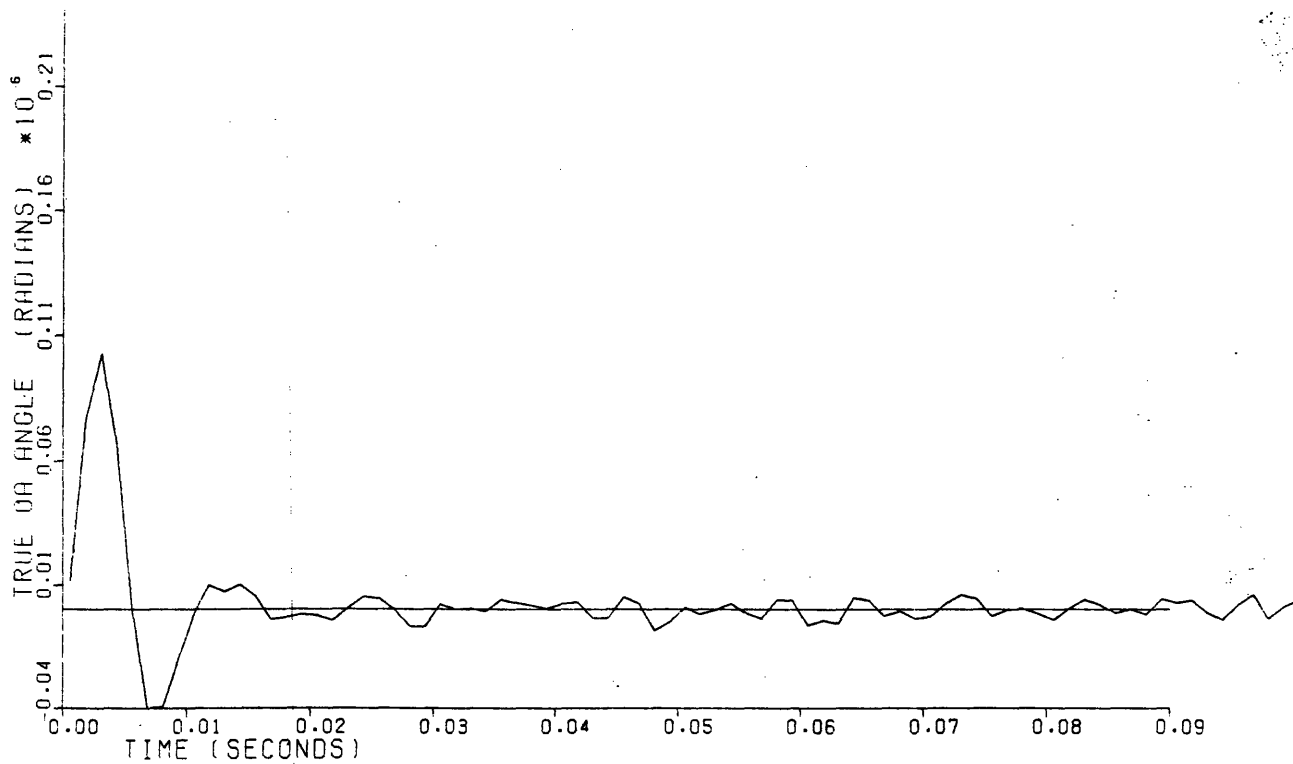


Figure 15-39. Run #30 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=10 nrad)

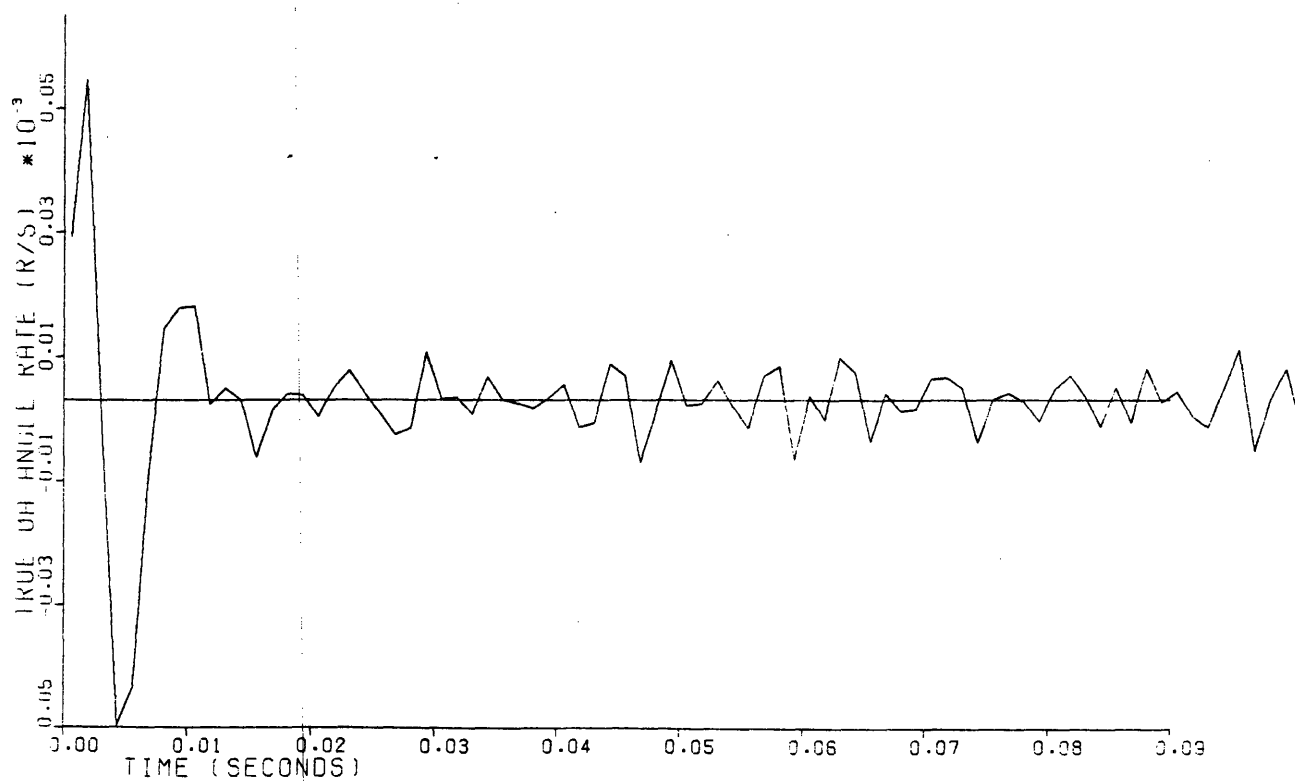


Figure 15-40. Run #30 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=10 nrad)

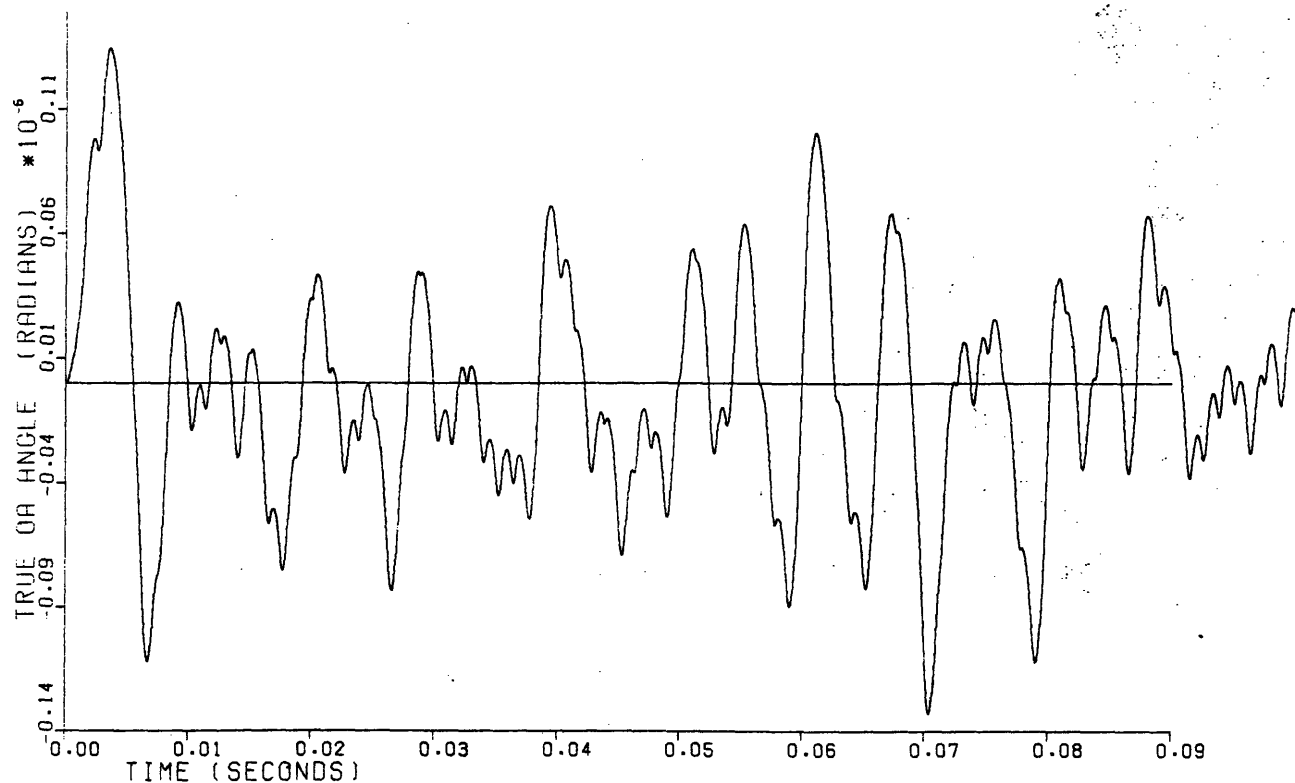


Figure 15-41. Run #31 Float Position vs. Time (7.5°/hr with SG Noise Level=100 nrad)

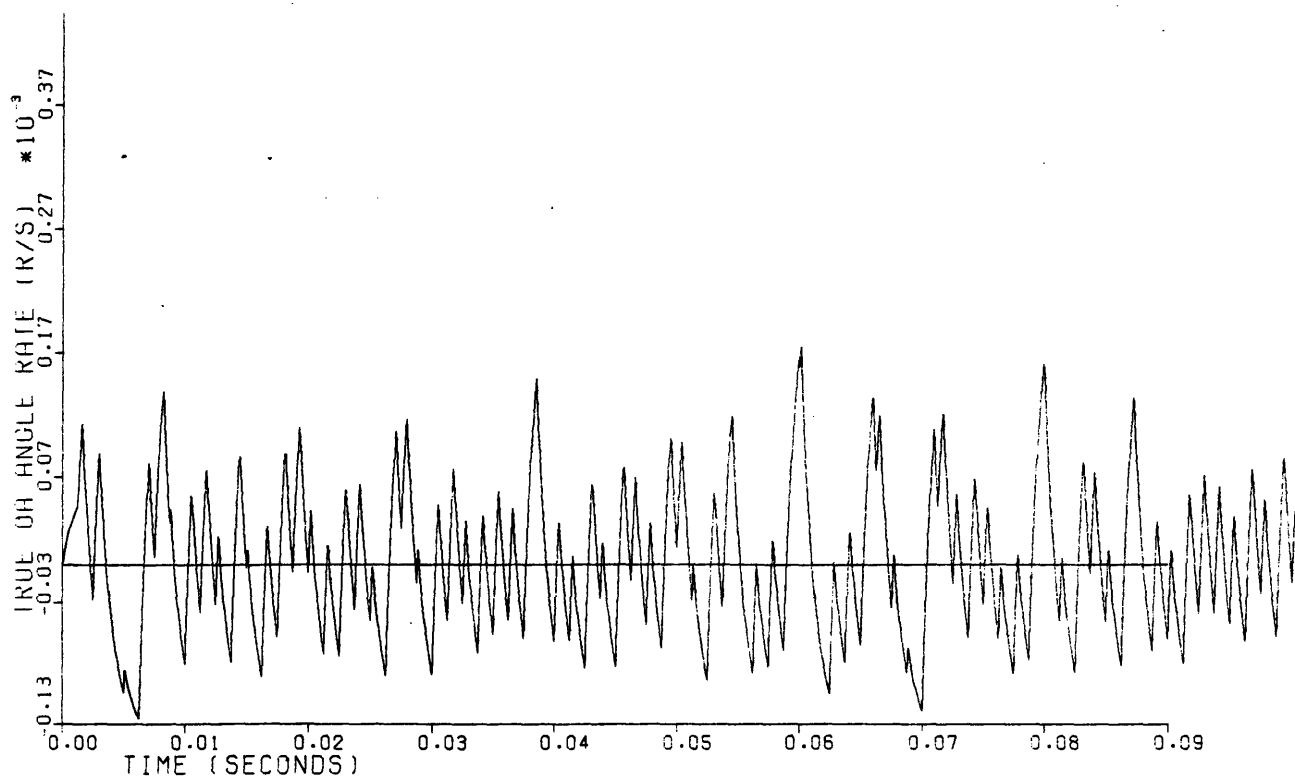


Figure 15-42. Run #31 Float Velocity vs. Time (7.5°/hr with SG Noise Level=100 nrad)

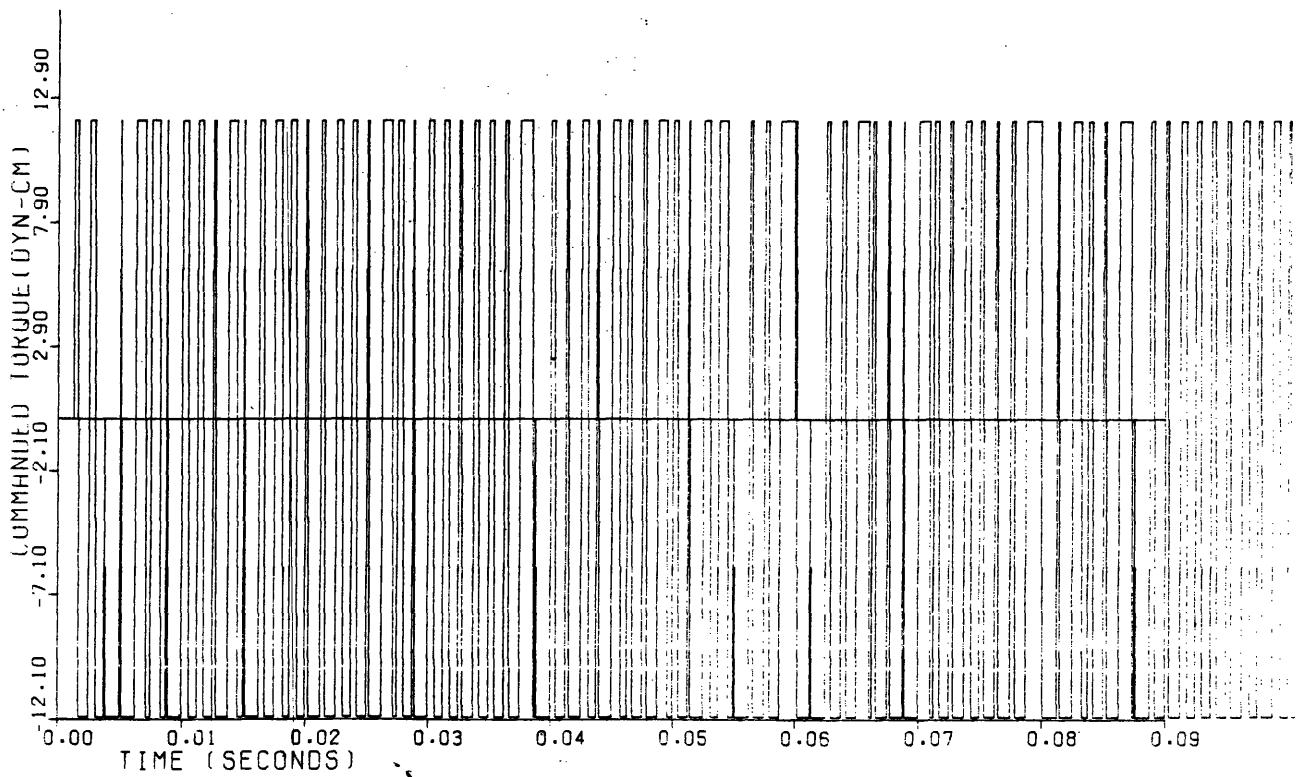


Figure 15-43. Run #31 Commanded Torque vs. Time (7.5°/hr with SG Noise Level=100 nrad)

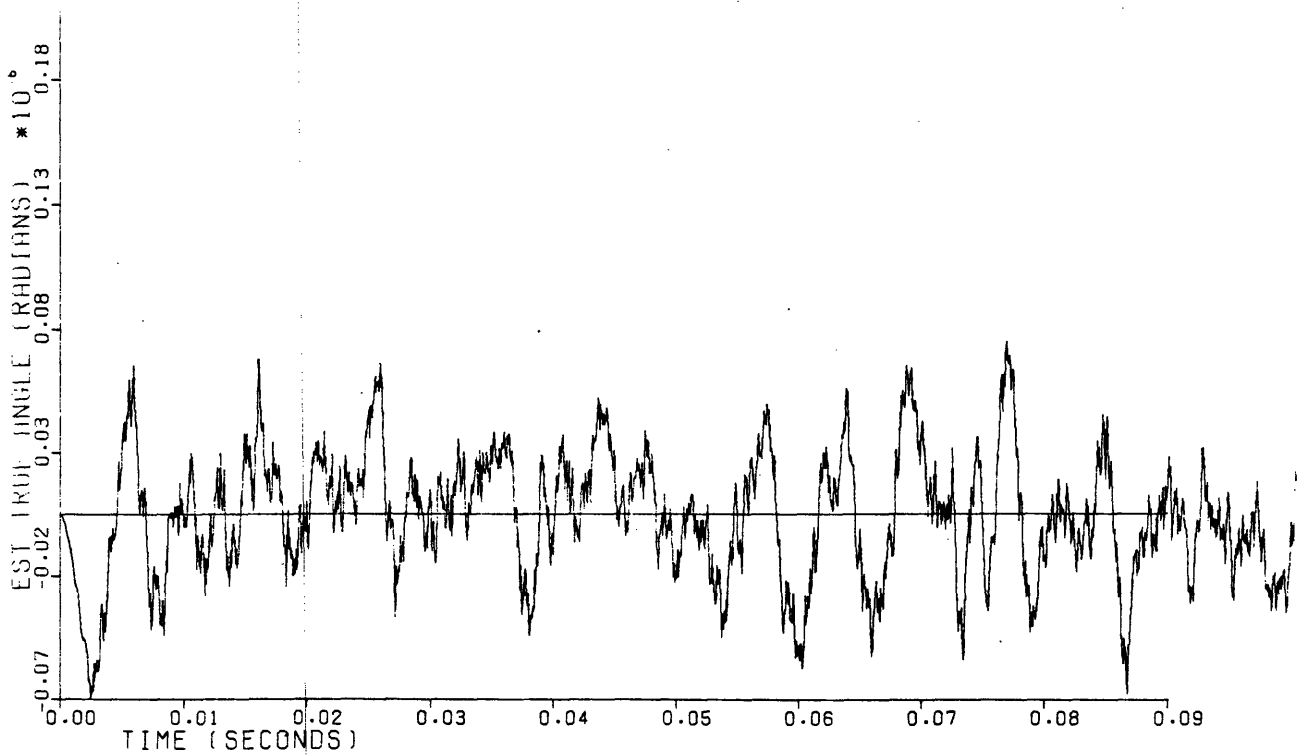


Figure 15-44. Run #31 Position Estimation Error vs. Time (7.5°/hr with SG Noise Level=100 nrad)

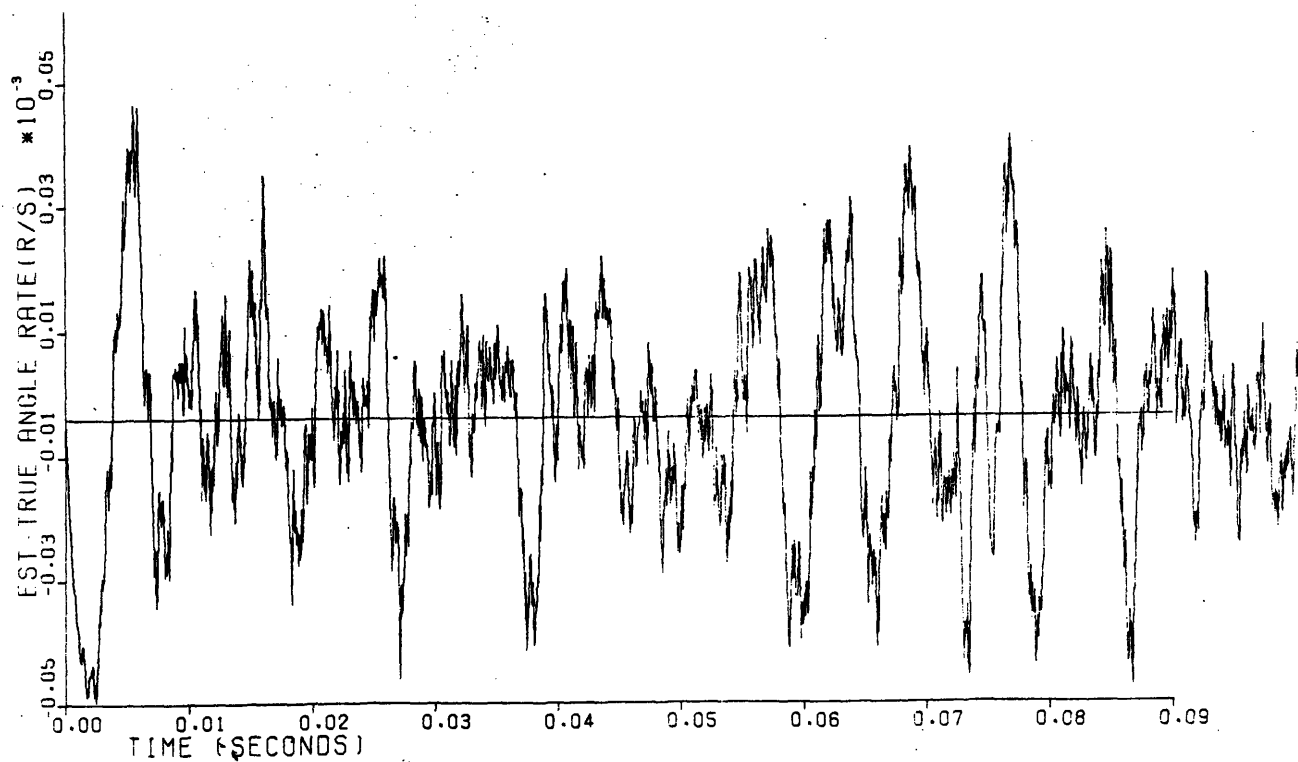


Figure 15-45. Run #31 Velocity Estimation Error vs. Time (7.5°/hr with SG Noise Level=100 nrad)

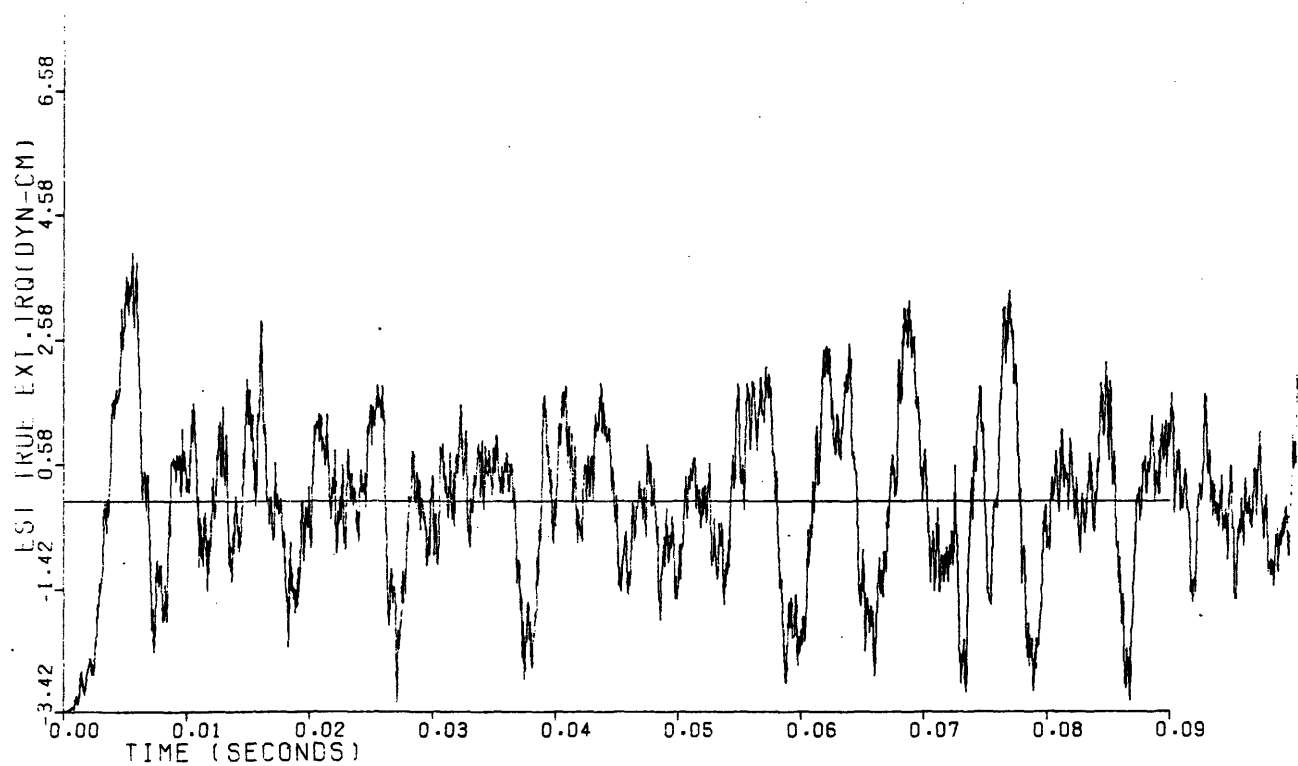


Figure 15-46. Run #31 External Torque Estimation Error vs. Time (7.5°/hr with SG Noise Level=100 m)

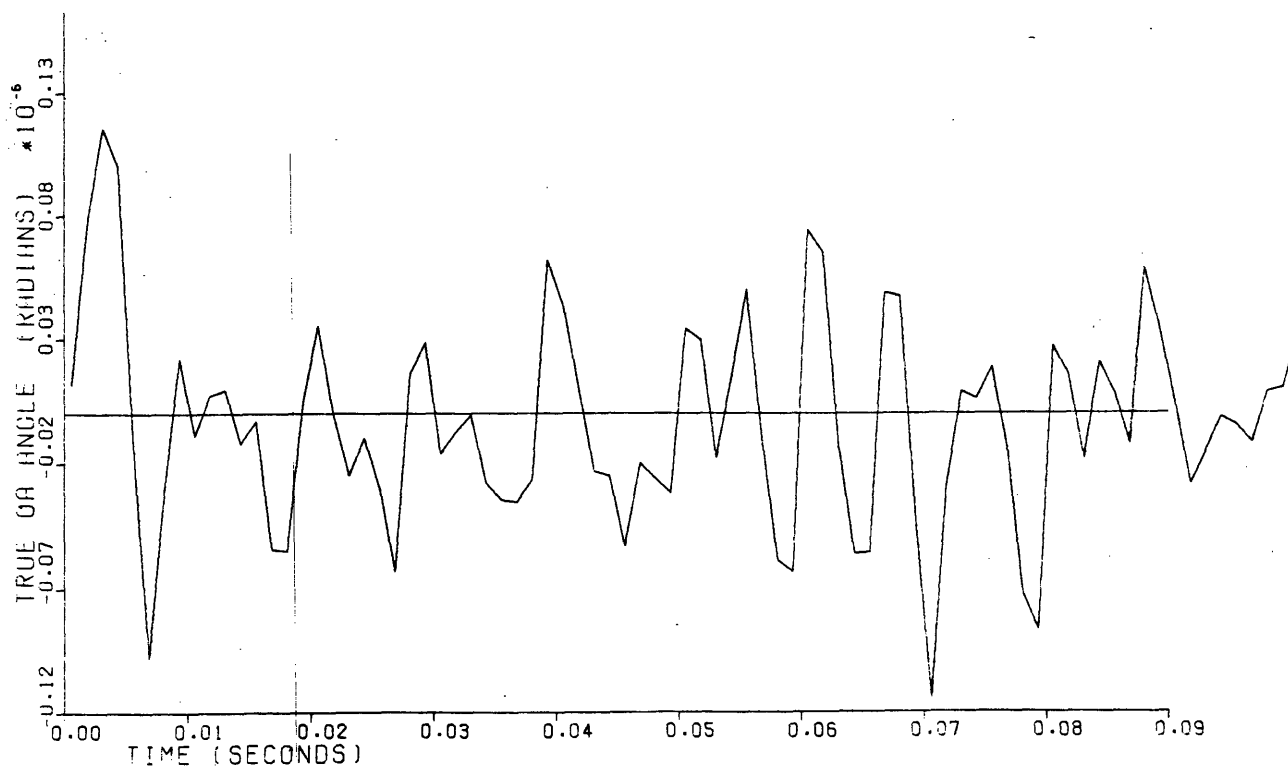


Figure 15-47. Run #31 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=100 nrad)

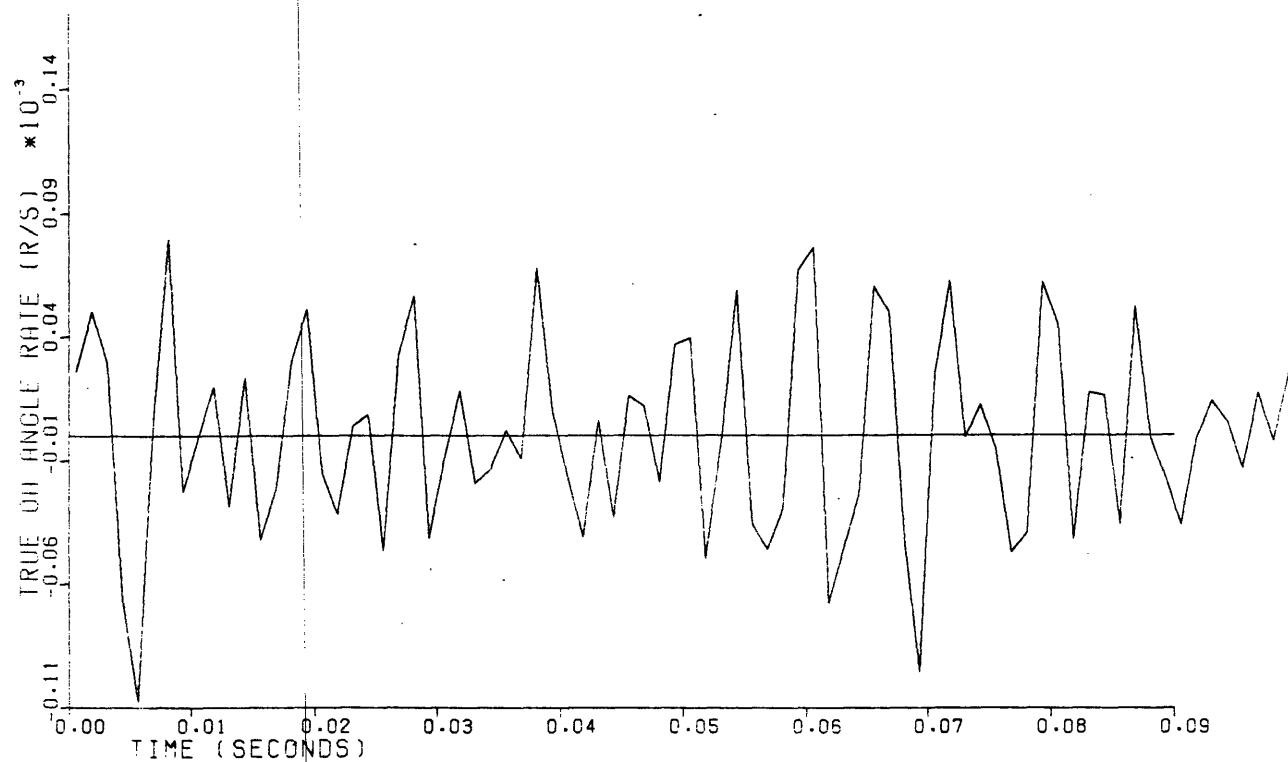


Figure 15-48. Run #31 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=100 nrad)

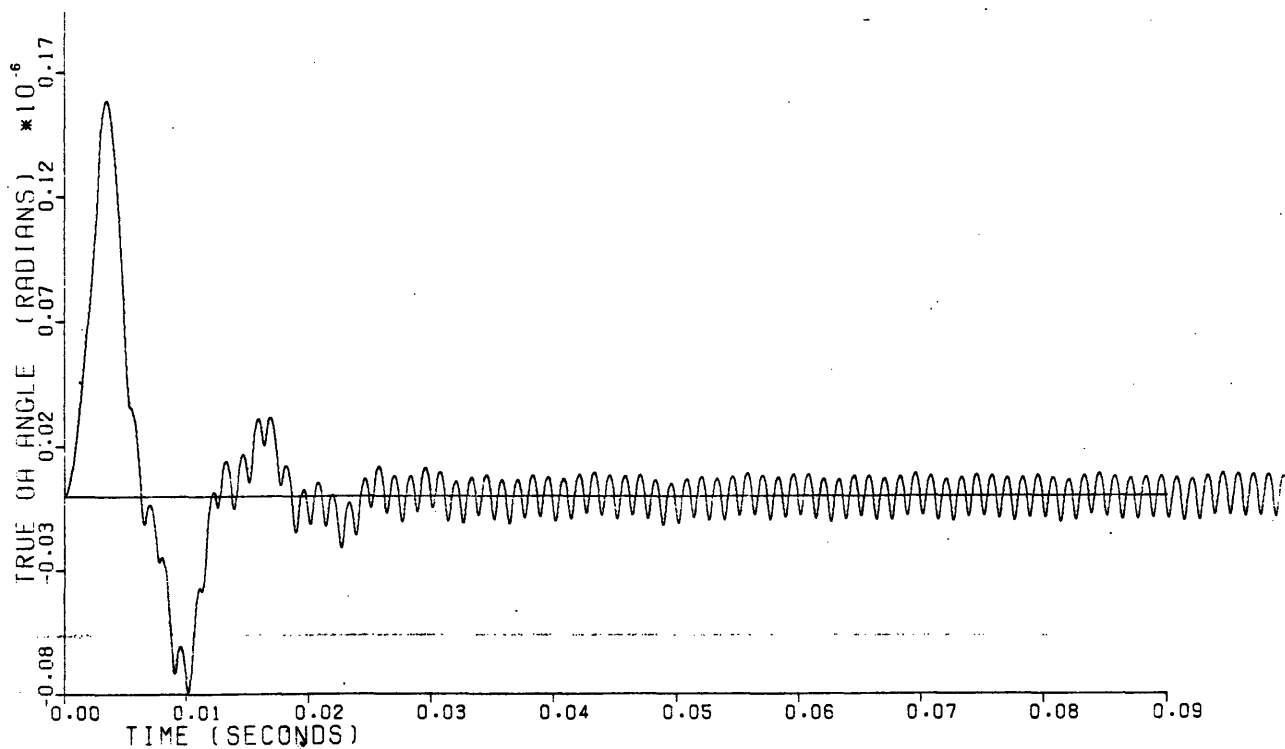


Figure 15-49. Run #32 Float Position vs. Time (7.5°/hr with SG Noise Level=1 nrad)

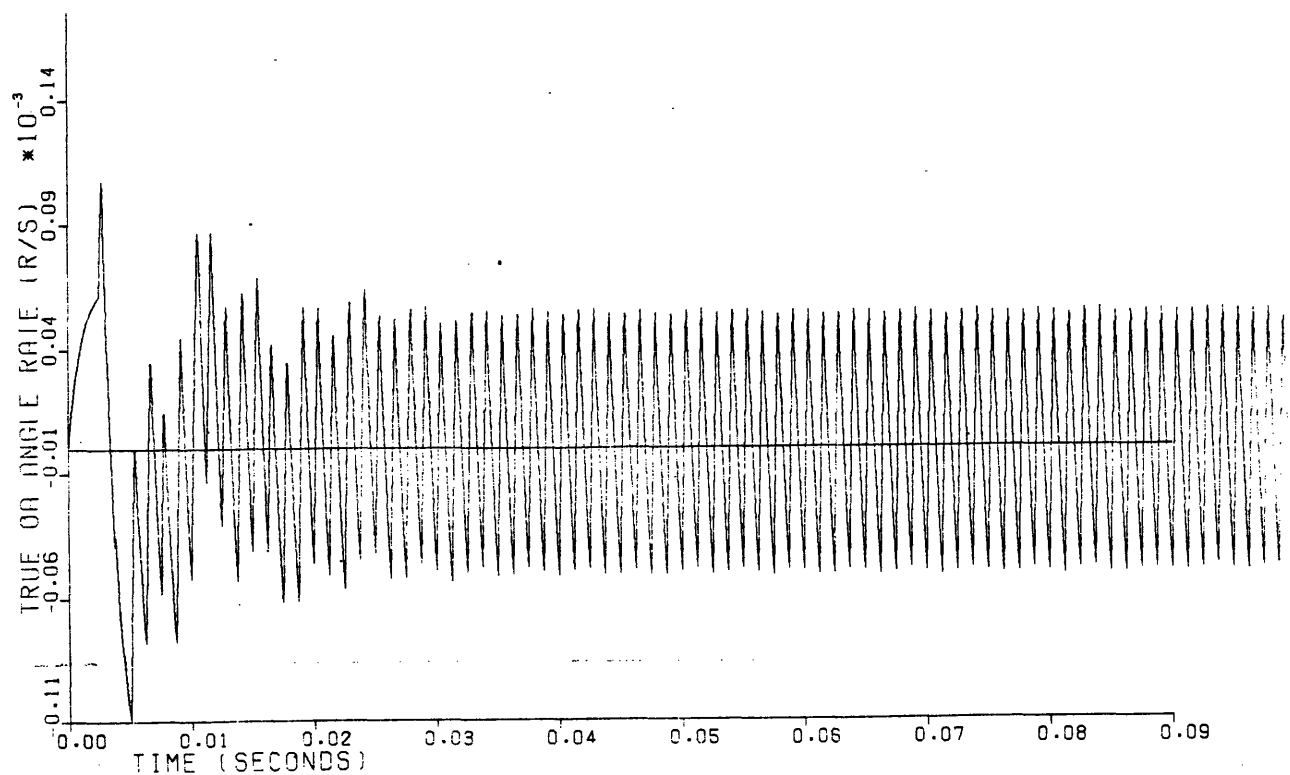


Figure 15-50. Run #32 Float Velocity vs Time (7.5°/hr with SG Noise Level=1 nrad)

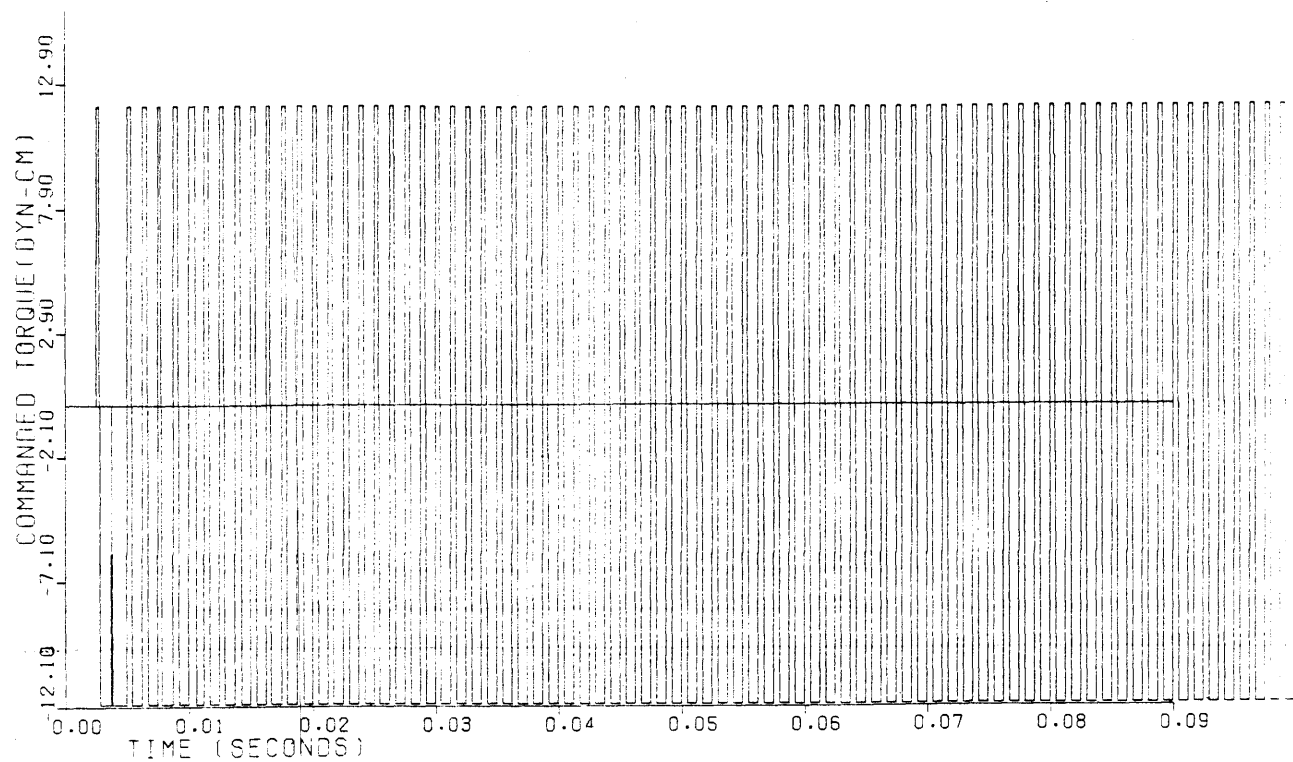


Figure 15-51. Run #32 Commanded Torque vs. Time (7.5°/hr with SG Noise Level=1 nrad)

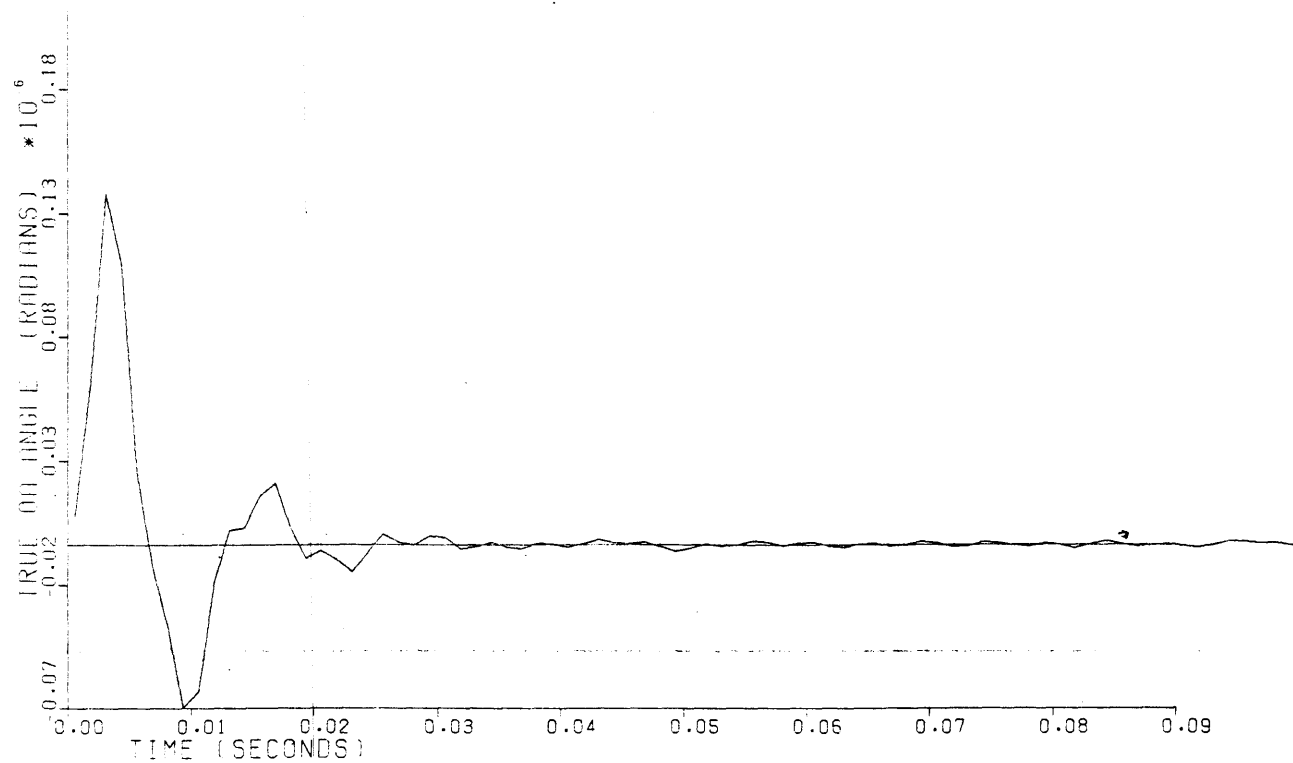


Figure 15-52. Run #32 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=1 nrad)

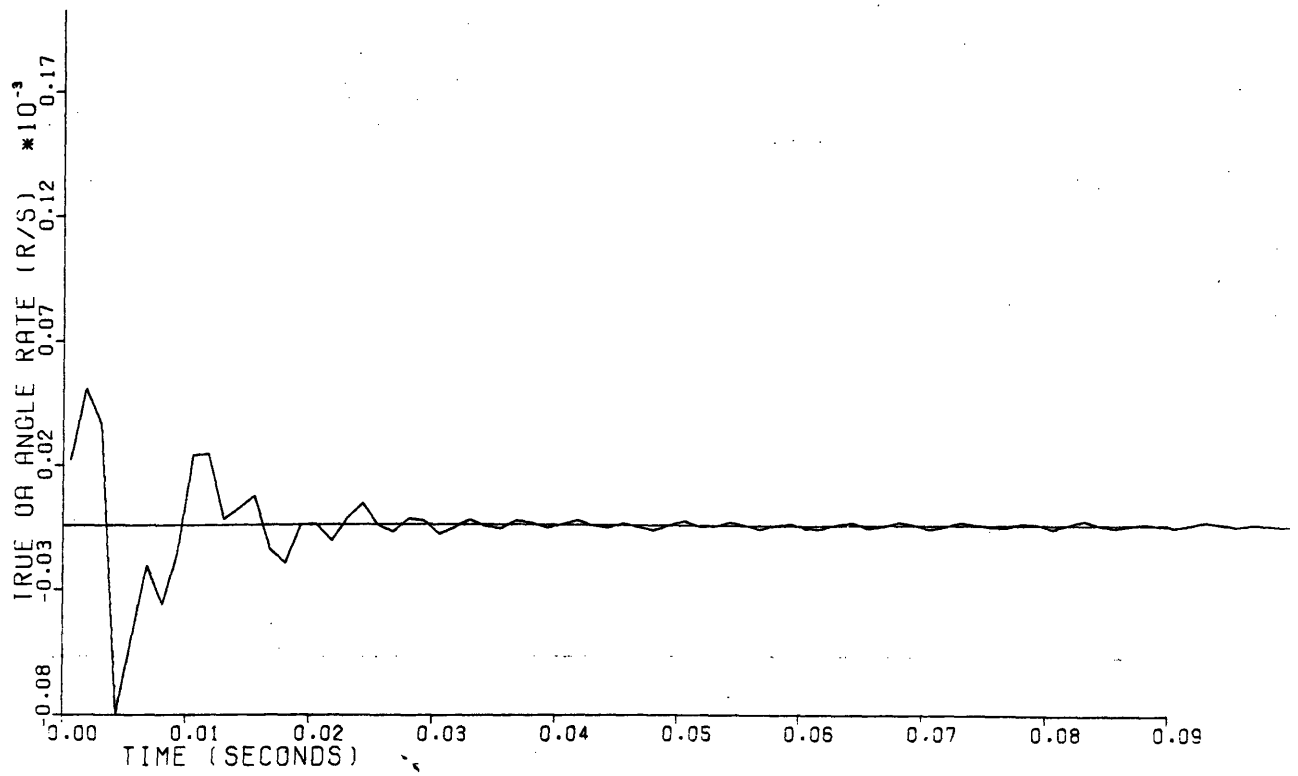


Figure 15-53. Run #32 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=1 nrad)

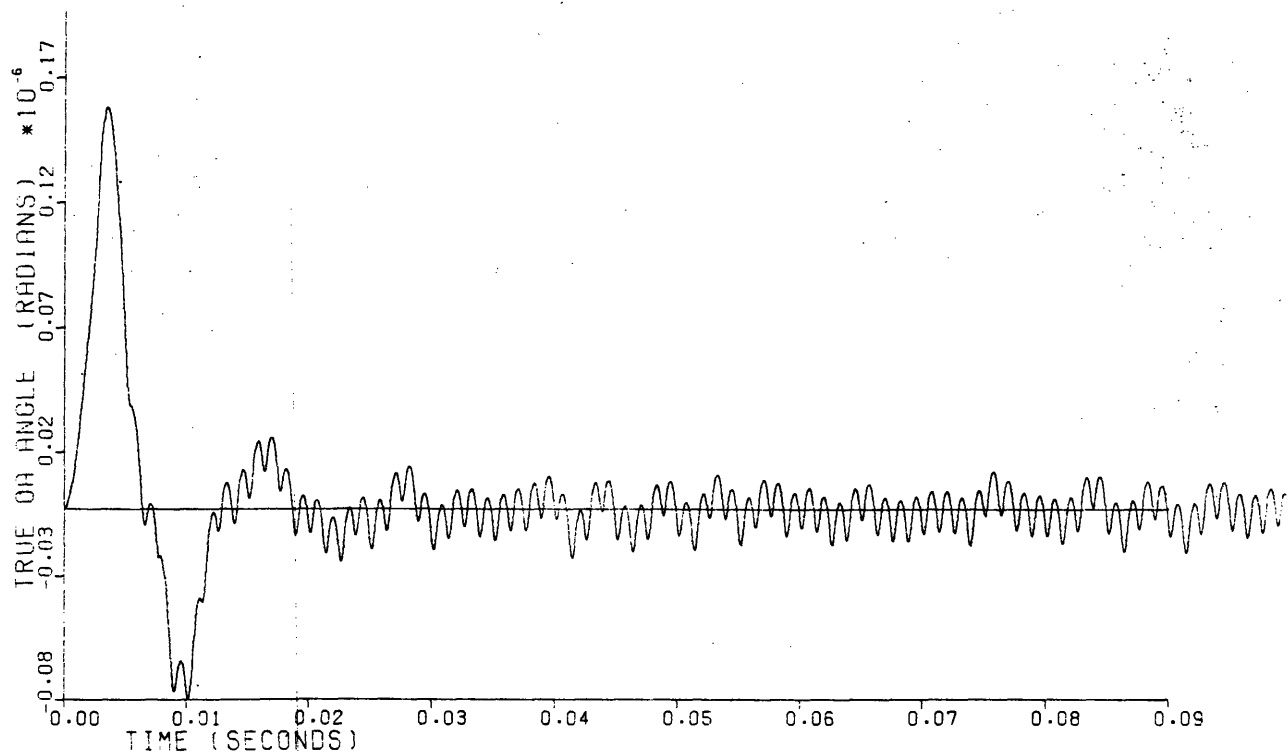


Figure 15-54. Run #33 Float Position vs. Time (7.5°/hr with SG Noise Level=10 nrad)

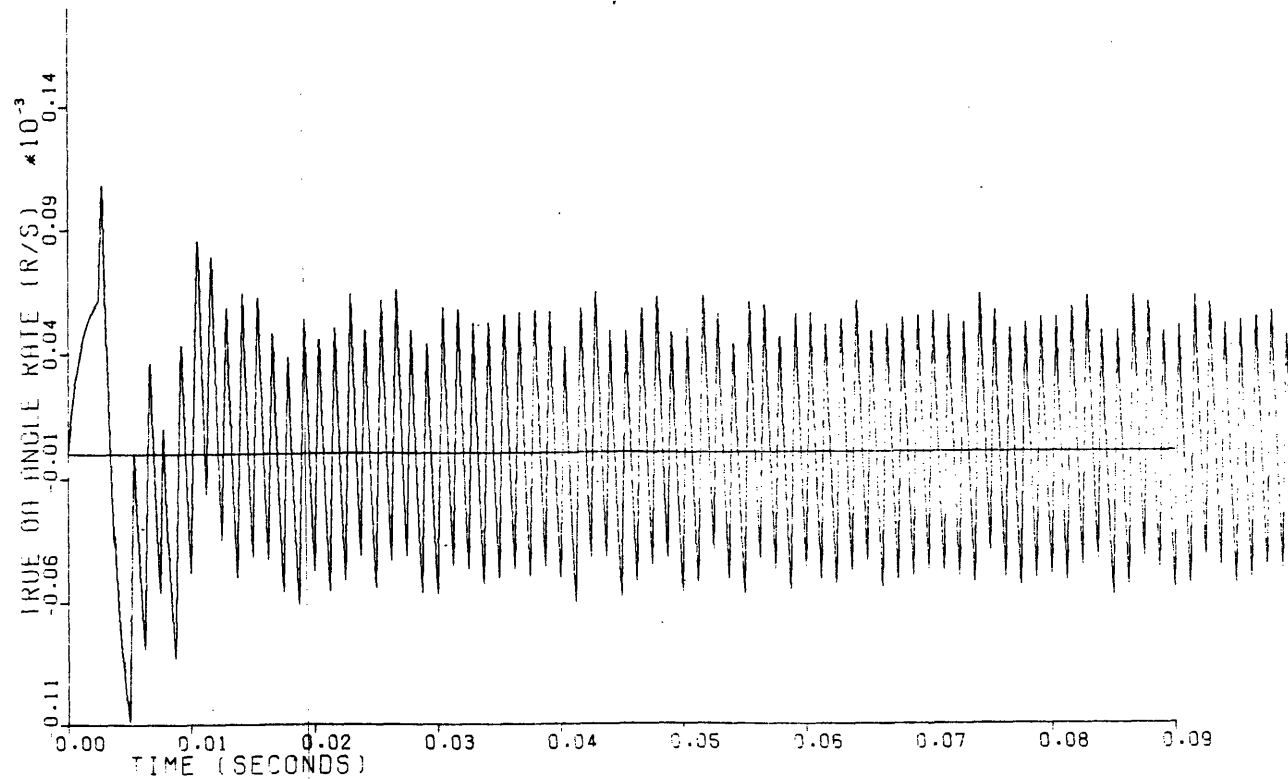


Figure 15-55. Run #33 Float Velocity vs Time (7.5°/hr with SG Noise Level=10 nrad)

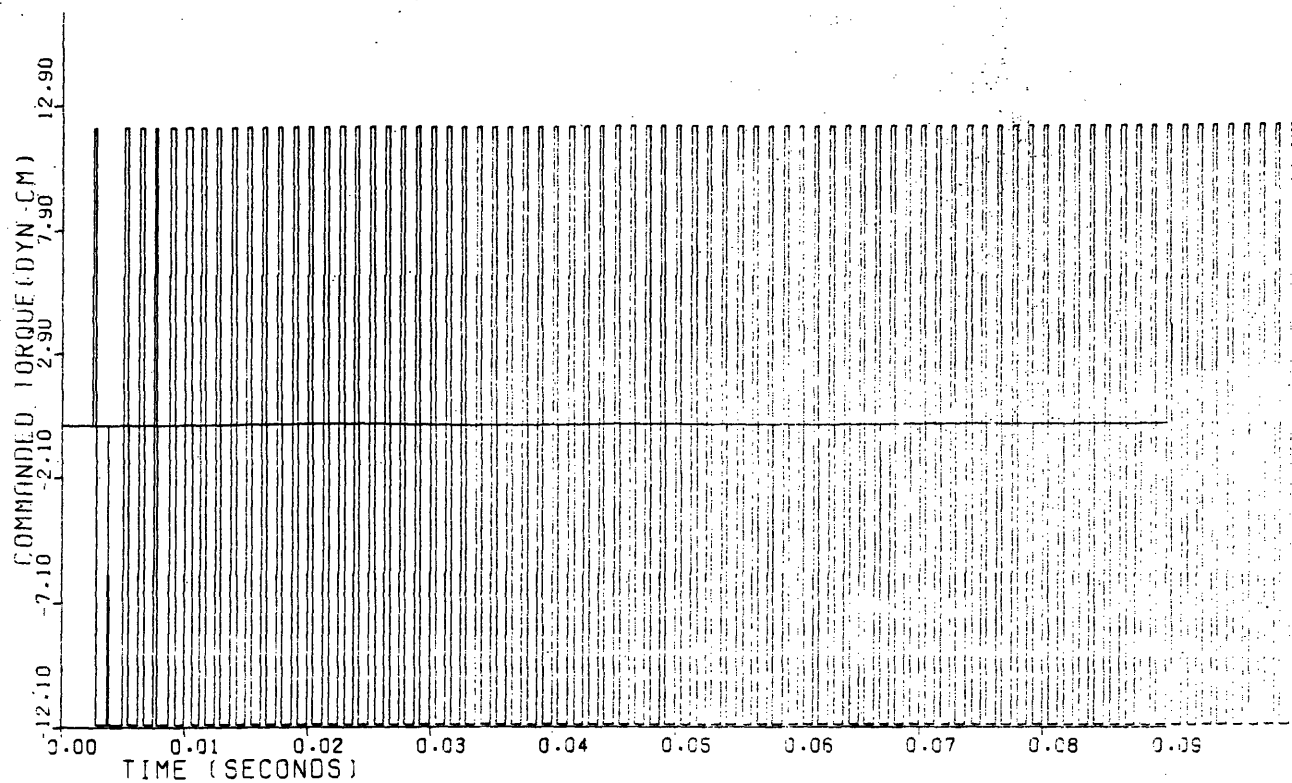


Figure 15-56. Run #33 Commanded Torque vs. Time (7.5°/hr with SG Noise Level=10 nrad)

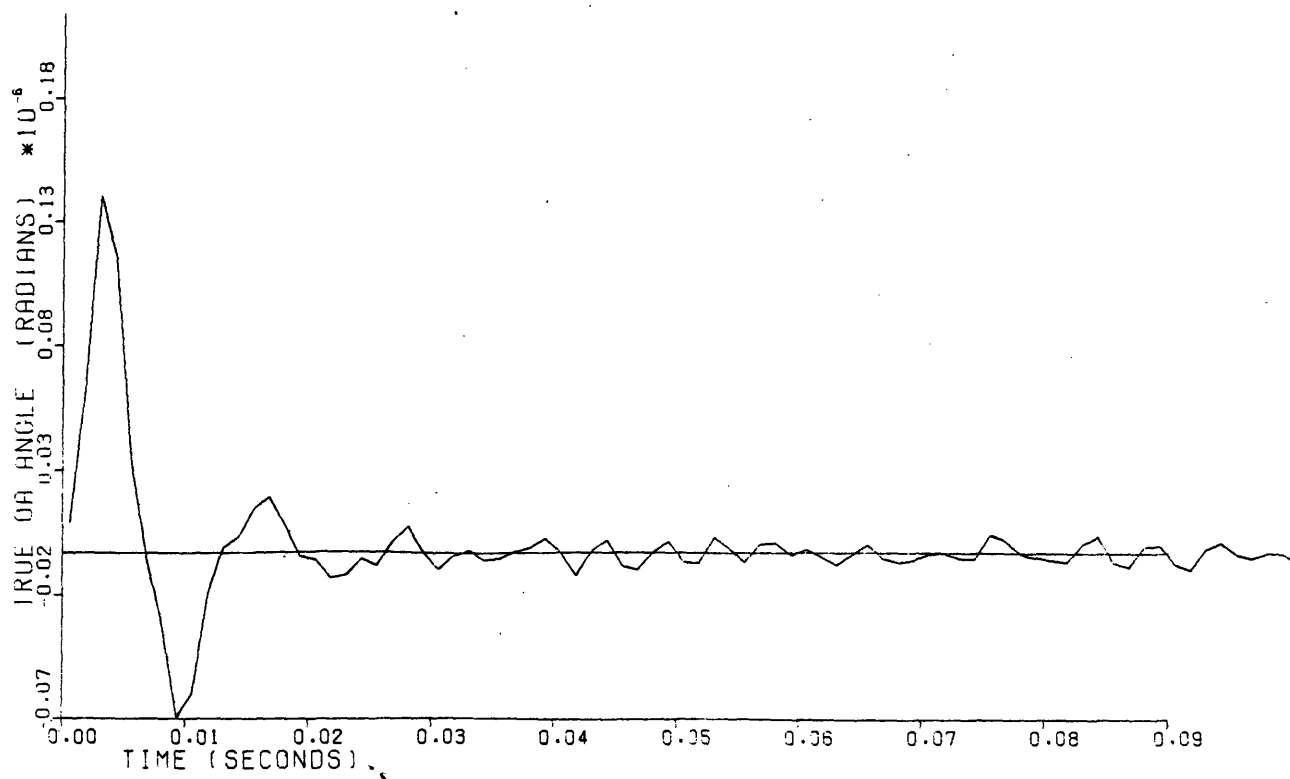


Figure 15-57. Run #33 24 Pt. Average Position vs. Time (7.5°/hr with SG Noise Level=10 nrad)

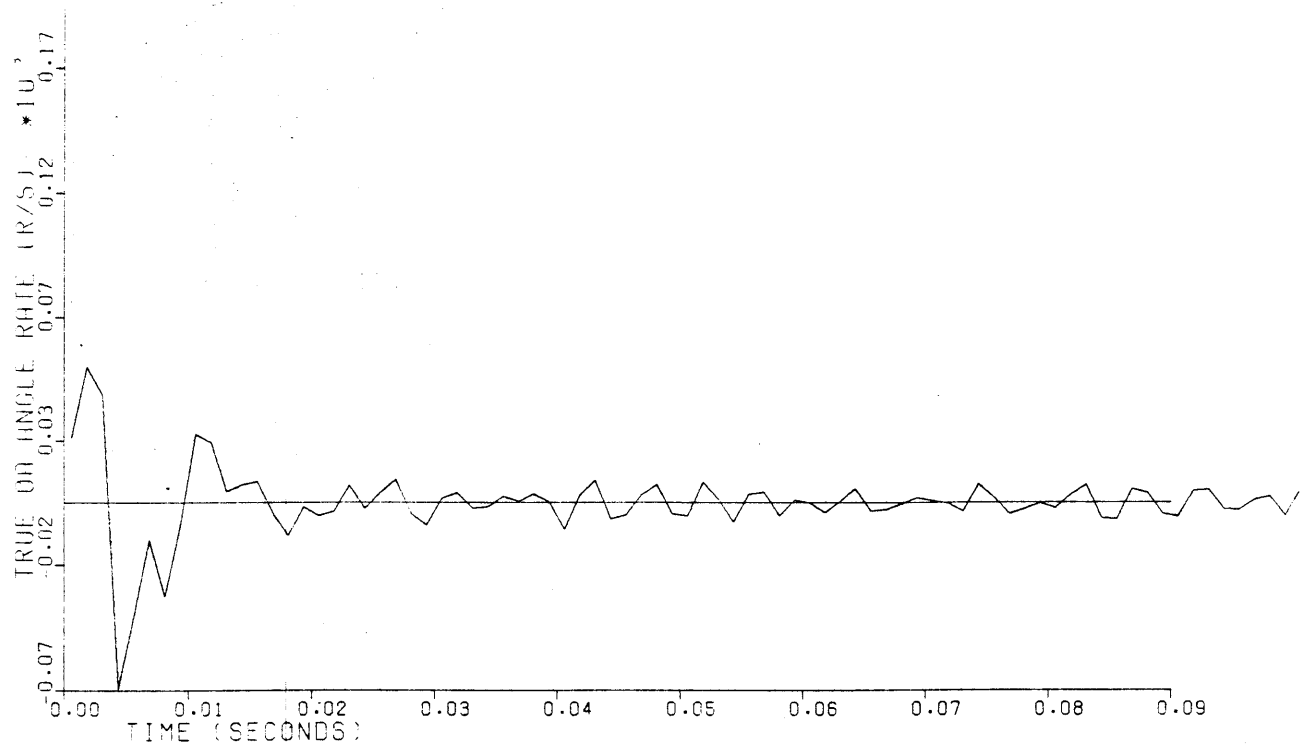


Figure 15-58. Run #33 24 Pt. Average Velocity vs. Time (7.5°/hr with SG Noise Level=10 nrad)

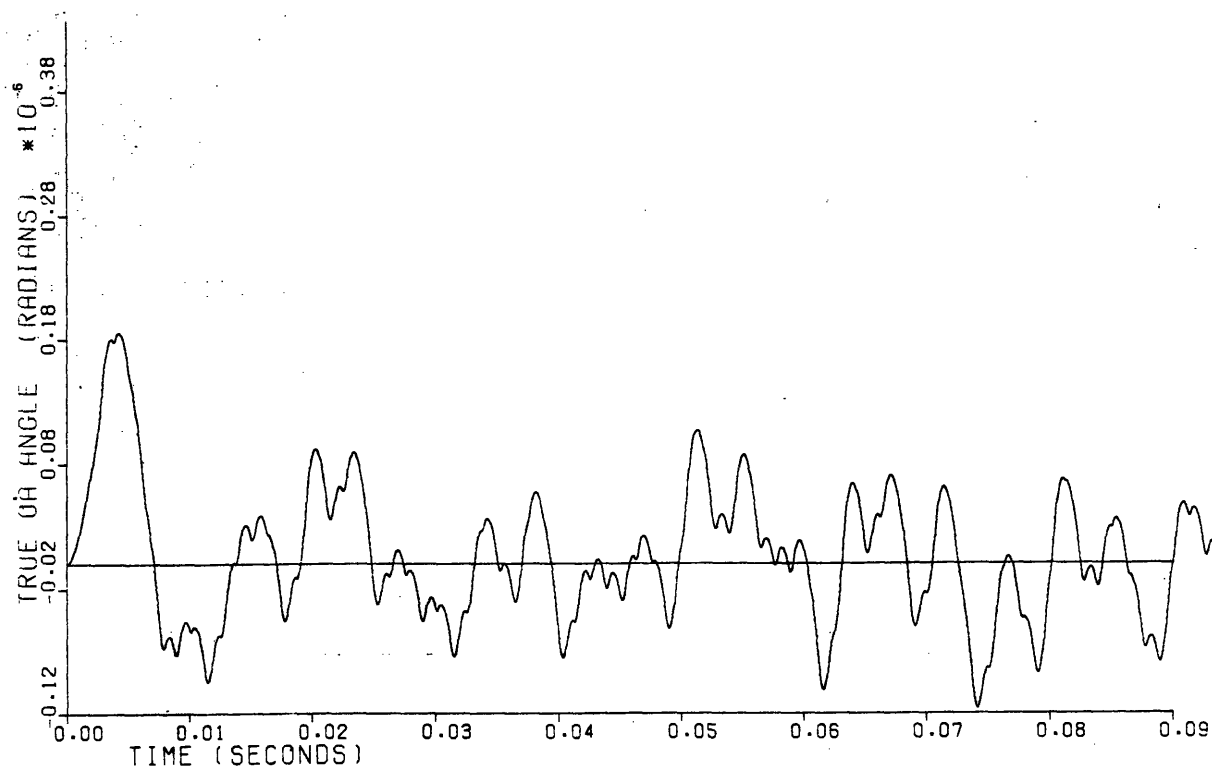


Figure 15-59. Run #34 Float Position vs. Time (7.5°/hr with SG Noise Level=100 nrad)

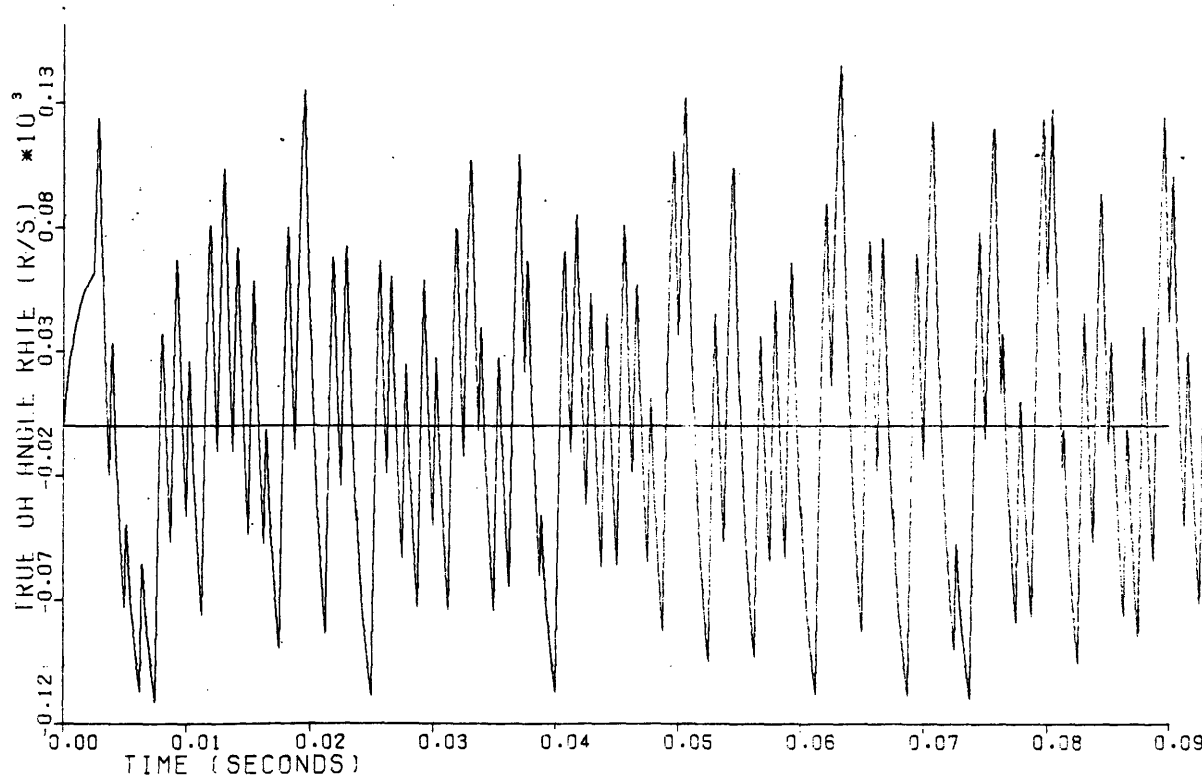


Figure 15-60. Run #34 Float Velocity vs. Time (7.5°/hr with SG Noise Level=100 nrad)

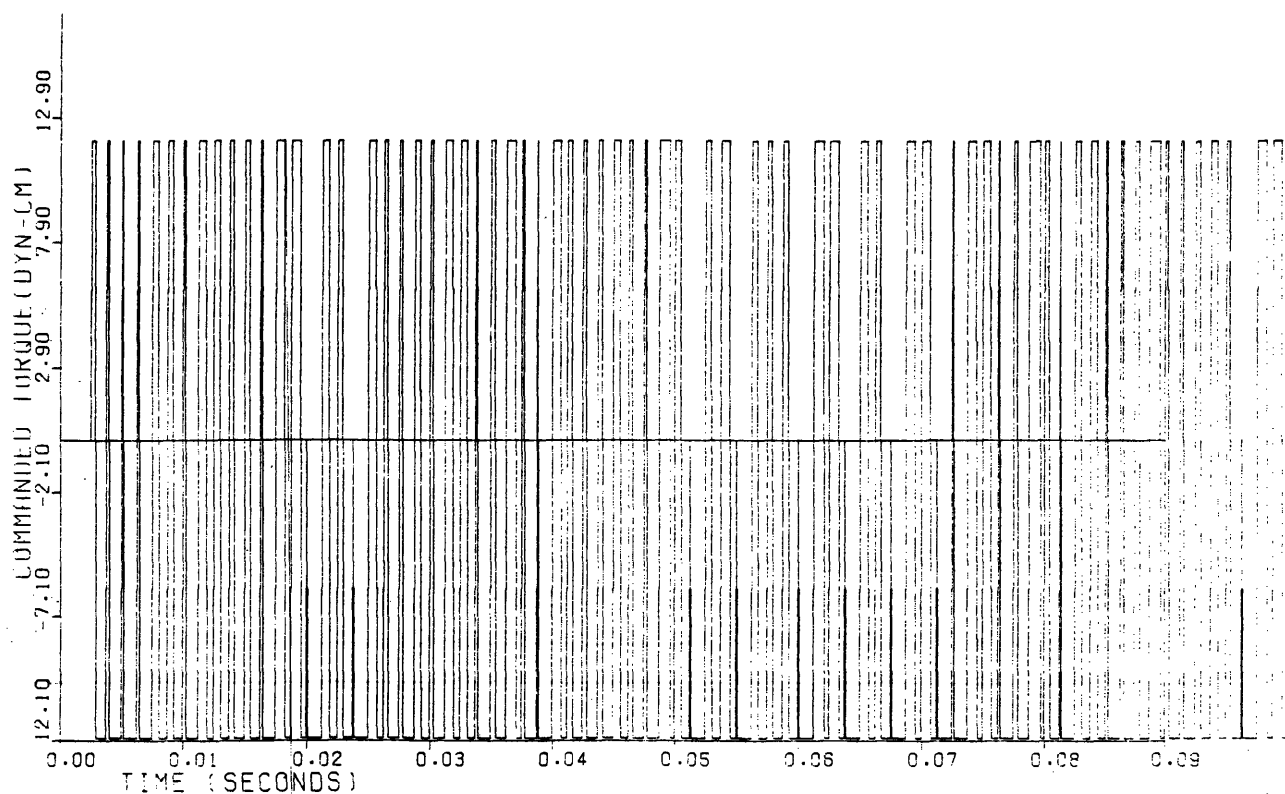


Figure 15-61. Run #34 Commanded Torque vs. Time ($7.5^\circ/\text{hr}$ with SG Noise Level=100 nrad)

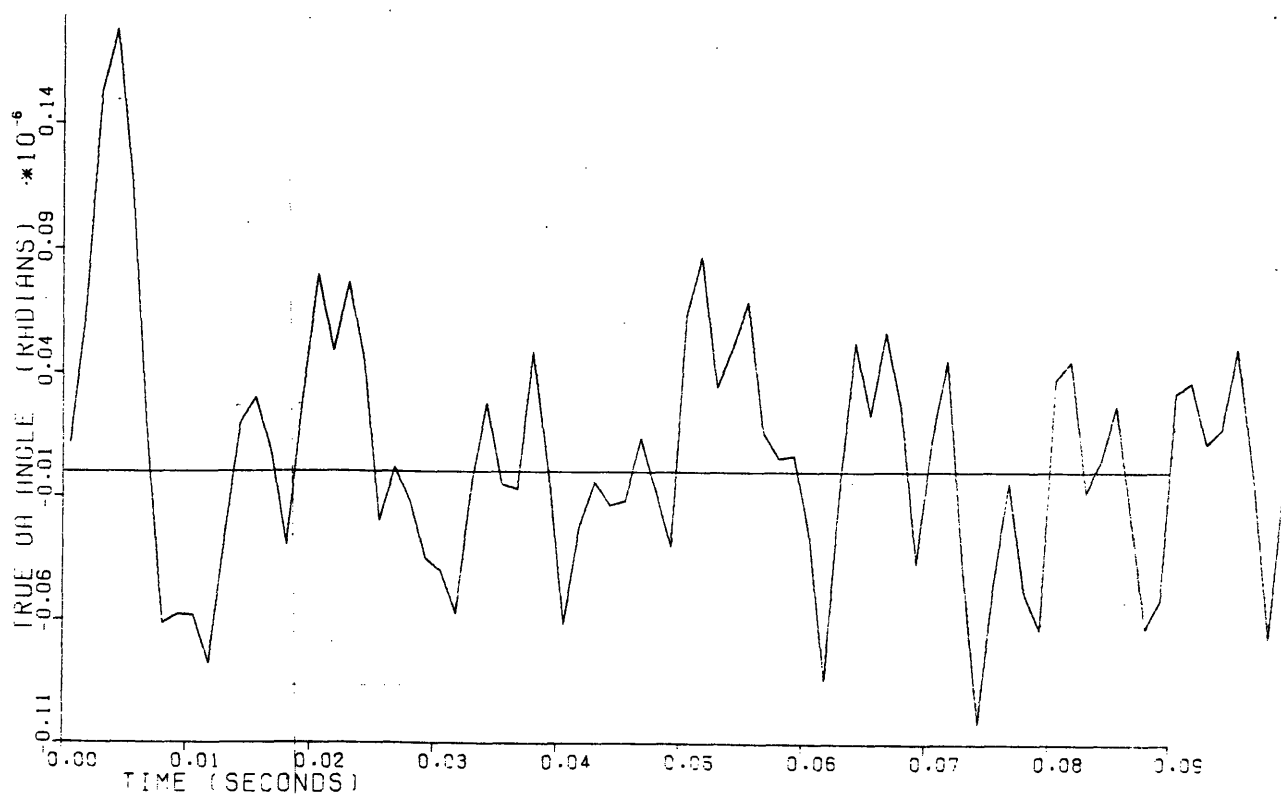


Figure 15-62. Run #34 24 Pt. Average Position vs. Time ($7.5^\circ/\text{hr}$ with SG Noise Level=100 nrad)

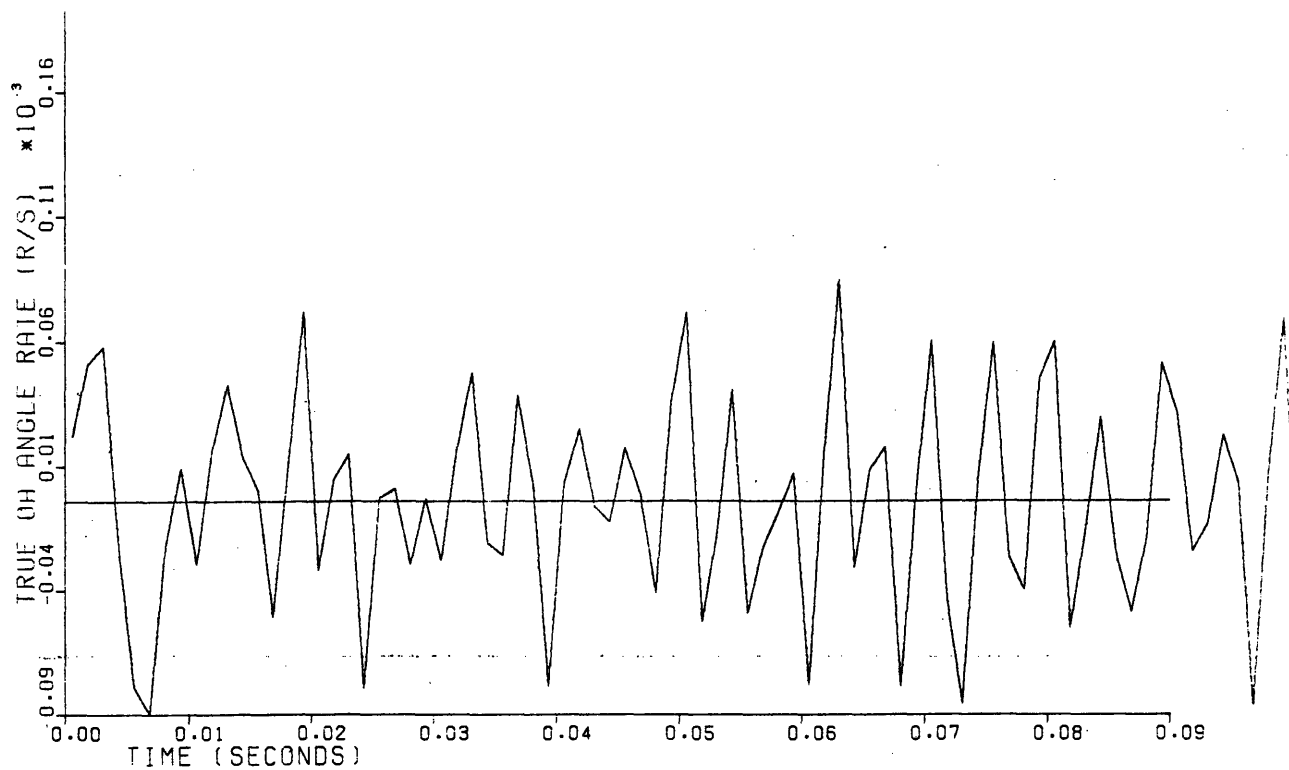


Figure 15-63. Run #34 24 Pt. Average Velocity vs. Time ($7.5^\circ/\text{hr}$ with SG Noise Level=100 nrad)

PART IV CONCLUSION

16. SUMMARY

The purpose of this thesis work was to develop the algorithms to be implemented in a torque loop microprocessor for the proposed pulse width modulation torque loop. These algorithms consist of a Kalman filter to estimate the gyro state vector and modelled external inputs to the gyro, and analytically derived control laws implementing control based on the estimated states. In addition, a simplified controller based solely on the recent history of average OA angle and previous torque commands was developed and compared to the Kalman filter based controllers.

A digital simulation of the proposed torque loop and developed algorithms was provided illustrating the steady state tracking accuracy and transient response to zero, constant, step, ramp, and sinusoidal rate inputs. Plots of the time domain responses of each controller for each input scenario were presented in order to identify the frequencies of float oscillations and moding of the torquing patterns. Two slightly different Kalman filter based controllers were developed, of which one employed an added state in the Kalman filter to track the rate of change of external inputs to the gyro. The transient responses of each of these controllers were compared for each run scenario and for three different values of estimator gains. In addition, the noise response for all controllers was presented for three different levels of noise assumed in SG signal.

The results indicated that a Kalman filter can be successfully employed at the chosen torque loop frequencies to provide accurate state estimates for the range of time varying inputs selected. The Kalman filter with added fourth state significantly improved the controller's response to time varying inputs except for a step input. Whether to employ the added state will depend on the application, such as pointing and tracking with high slew rates versus stabilization with near constant or slowly varying inputs. The added time and memory to perform the additional computations required by the four state filter will also be a deciding factor.

All controllers satisfied the two main criteria of insuring no moding of the torquing pattern, and therefore resisting all but the minimum mode of float oscillation, and keeping the float near null. Moding did occur when large enough levels of SG noise were added, depending upon the values of the estimator gains chosen. A final choice of estimator gains would require a more formal trade-off study of the estimator tracking ability versus its ability to reject noise. Factors affecting these abilities will be the number of Kalman filter updates per torquing interval, i.e. the Kalman filter frequency, and how much noise in the SG signal can be removed prior to entering the Kalman filter. Increasing the number of updates per torquing interval will allow the use of smaller estimator gains and produce better noise rejection capability. For the simulation runs presented, the simplified controller performed comparably to the Kalman filter based controller employing "slow" estimate update gains.

17. IMPLEMENTATION AND RECOMMENDATIONS

The algorithms developed in this thesis were originally supposed to be implemented on a Motorola DSP56001 digital signal processor, which executes 10.25 million instructions per second in 24 bit fixed point arithmetic with 56 bit accumulators. The simplified controller can be and has been converted to fixed point arithmetic relatively easily.

Torque loop simulations of the simplified controller in fixed point arithmetic with quantization simulating the 24 bit microprocessor word has been performed. Although not presented here, results of these simulations were similar to the floating point runs except for degradation of performance which occurred when float angles overflowed the allocated word length. When saturation occurred, torque words for maximum positive torque (250) or maximum negative torque (5) were commanded.

Converting the Kalman filter equations and associated controllers to fixed point arithmetic will require creative simplification and efficient organization of those equations. There are other microprocessors, such as a version of the Texas Instrument's TMS320, and the soon to arrive Motorola DSP96002 which are capable of floating point operations at approximately the same speed as their fixed-point counterparts. Use of these other processors should be considered.

Assuming a processor which performs 10.25 million instructions per second and a minimum interval $5 \times 4.883 = 24.415 \mu\text{sec}$ of positive torque at the beginning of each torquing interval, then a minimum of 250 instructions can be performed between the start of a torquing interval and the moment when the first allowed switching instant may occur. It has been estimated that the simplified controller can be implemented in about 20 instructions (multiply two numbers and add or subtract the result to a register). For the simplified controller, it should be possible to combine the SG prefilter and control algorithms in one microprocessor through multitasking interrupt driven software.

The Kalman filter based controllers can probably be implemented in approximately 240 instructions. This includes the four Newton-Raphson iterations and control law calculations at the start of a torquing interval. Implementing the Kalman filter is greatly simplified by using fixed gains. In addition since there is only one observable quantity, the update uses scalar equations with no matrix operations.

In order to insure that accurately known amounts of torque are applied, the torque pulses should deliver net torque with an accuracy of one part in 10^6 . Assuming a torquer inductive time constant of approximately 3 μsec , to get within 10^{-6} of the new level requires about 14 time constants or about 42 μsec . A part in 256 quantization interval for 800 Hz torquing is 4.883 μsec . Hence 14 time constants is 8.6 quantization intervals. With a torque word N satisfying $5 \leq N \leq 250$ (see Section 6.2), the pulse has only gotten to within 3×10^{-4} of its new level. Therefore we should have $9 \leq N \leq 246$ in an actual implementation.

The simple second order model used to represent gyro dynamics is by no means a complete model of the known dynamics of a single degree of freedom floated gyroscope, therefore extension of the model is recommended.

The very high frequency dynamics (> 100 Hz) of the torque and signal generators have not been included since the objective here was to characterize the low frequency (< 100 Hz) behavior of the torque loop. The TG and SG can be modelled as first order low pass filters with respective time constants τ_{TG} , τ_{SG} . In addition, this thesis assumes all noise processes are zero mean Gaussian distributions. Characterization of other noise processes should be included as well.

Gyro error sources such as thermal gradients, unbalance between plus and minus torquer scale factors, float motion induced scale factor changes, and crosscoupling between axes have not been modeled. Their effects can and should be compensated for outside the torque loop. Having a microprocessor controlled torque loop is a step towards the goal of an "intelligent inertial instrument", capable of performing real time correction of the torque

output for error effects using temperature, magnetic suspension, and other readout signals, and then applying an appropriate finite impulse response digital filter to the high quantization resolution pulse width modulation torque word or Kalman filter state.

Good control system design is an iterative process. It is therefore recommended that the algorithms developed in this thesis be implemented in hardware and tested. Further analysis and upgrading of the model may then become necessary if the simple model does not produce acceptable results.

The results of this thesis can be applied to other inertial instruments besides the single degree of freedom floated gyroscope. The Kalman filter and derived control laws can be directly applied to a PIPA (Pulsed Integrating Pendulum Accelerometer) since construction is basically similar to the floated gyroscope with the absence of a spinning wheel. If it is desired to adapt the system to non-floated instruments, such as a pendulous forced rebalance accelerometer or dry tuned rotor gyroscopes, the effects of low damping would have to be investigated.

It has been shown that the Kalman filter based controllers, with pulse width modulation torquing, are capable of providing superior float angle control. Negligible steady state float hang-offs (< 1 nrad) are observed for constant or slowly varying input rates. In addition, at the simulated clock frequencies, excellent transient performance was observed for step, ramp and 10 Hz sinusoidal inputs. Possible sampling rates of the torque loop will depend on the microprocessor chosen for implementation, however, the simulated frequencies were selected to be conservative. Finally, having the external inputs be a state in the Kalman filter provides an alternative to the conventional method of counting torque pulses for accurate determination of the angular velocity applied to the gyro. In addition, the microprocessor implemented Kalman filter will be able to provide fast updates of input-axis rate estimates and therefore a high bandwidth of the external input rate measurement.

APPENDIX A

DERIVATION OF CONTINUOUS TO DISCRETE-TIME DIFFERENCE EQUATIONS USING VARIATION OF PARAMETERS METHOD

Consider the continuous-time linear system equations given by

$$\dot{\mathbf{x}}(t) = \mathbf{A}(t) \mathbf{x}(t) + \mathbf{B}(t) \mathbf{u}(t) \quad (\text{A-1})$$

where the control input forcing function $\mathbf{B}(t) \mathbf{u}(t)$ has been included. The noise vector $\mathbf{w}(t)$ is assumed zero, however. Since the equations are linear, its effect can be added later. Following the derivation in Chapter 3 of Reference [3], the solution of the homogeneous equation with $\mathbf{u}(t)=0$ is

$$\mathbf{x}(t) = \Phi(t, t_0) \mathbf{x}(t_0) \quad (\text{A-2})$$

where $\Phi(t, t_0)$ is the transition matrix as defined in Section 7.2 and t_0 is some initial starting time. Using the method of Variation of Parameters, a particular solution can be found by assuming a form of solution

$$\mathbf{x}_p(t) = \Phi(t, t_0) \mathbf{p}(t) \quad (\text{A-3})$$

and substituting into Equation (A-1) yielding

$$\frac{d}{dt} \left[\Phi(t, t_0) \underline{p}(t) \right] = A(t) \Phi(t, t_0) \underline{p}(t) + B(t) \underline{u}(t)$$

or

$$A(t) \Phi(t, t_0) \underline{p}(t) + \Phi(t, t_0) \dot{\underline{p}}(t) = A(t) \Phi(t, t_0) \underline{p}(t) + B(t) \underline{u}(t)$$

This reduces to

$$\dot{\underline{p}}(t) = \Phi(t_0, t) B(t) \underline{u}(t)$$

by recognizing that $\Phi^{-1}(t, t_0) = \Phi(t_0, t)$ (proved in Reference [3]). Solving for $\underline{p}(t)$,

$$\underline{p}(t) = \underline{p}(t_0) + \int_{t_0}^t \Phi(t_0, \tau) B(\tau) \underline{u}(\tau) d\tau$$

Substituting into Equation (A-3), we get

$$\underline{x}(t) = \Phi(t, t_0) \underline{p}(t_0) + \int_{t_0}^t \Phi(t, t_0) \Phi(t_0, \tau) B(\tau) \underline{u}(\tau) d\tau$$

But since $\underline{p}(t_0) = \underline{x}(t_0)$, the final solution can be written as

$$\mathbf{x}(t) = \Phi(t, t_0) \mathbf{x}(t_0) + \int_{t_0}^t \Phi(t, \tau) \mathbf{B}(\tau) \mathbf{u}(\tau) d\tau, \quad (\text{A-4})$$

the sum of the homogeneous and particular solutions. Since for the discrete formulation we are interested in the system states only at discrete instants $t_i, t_{i+1}, t_{i+2}, \dots$, the discrete-time difference equation can be obtained by writing Equation (A-4) with slightly changed notation as

$$\mathbf{x}(t_{i+1}) = \Phi(t_{i+1}, t_i) \mathbf{x}(t_i) + \int_{t_i}^{t_{i+1}} \Phi(t_{i+1}, \tau) \mathbf{B}(\tau) \mathbf{u}(\tau) d\tau \quad (\text{A-5})$$

For a more thorough discussion of discretizing continuous-time equations see References [1], [2], and [3].

APPENDIX B

CONTINUOUS TO DISCRETE-TIME EQUATIONS TAKING INTO ACCOUNT PROCESSOR DELAY

In digital control systems, it is often the case that the instants at which the output is sampled do not necessarily coincide with the instants at which the control inputs are applied. The linear discrete-time system is assumed to be operating a linear continuous-time system with piecewise constant inputs. The control instants are represented by the time values t_i , $i=0,1,2,\dots$. These are the times at which the continuous-time variable $y(t)$ are sampled. Furthermore it is assumed that $t_i < t_i' < t_{i+1}$.

The continuous time equations are now

$$\dot{\underline{x}}(t) = \underline{A}(t) \underline{x}(t) + \underline{B}(t) \underline{u}(t) + \underline{w}(t)$$

(B-1)

$$g(t') = \underline{C}(t') \underline{x}(t') + \underline{u}(t')$$

Figure B-1 illustrates processor delay. The discrete-time equations derived from (B-1) will be [2]

$$\underline{x}(t_{i+1}) = \Phi(t_{i+1}, t_i) \underline{x}(t_i) + \left[\int_{t_i}^{t_{i+1}} \Phi(t_{i+1}, \tau) \underline{B}(\tau) d\tau \right] \underline{u}(t_i)$$

$$\begin{aligned}
 & + \int_{t_i}^{t_{i+1}} \Phi(t_{i+1}, \tau) \underline{w}(\tau) d\tau \\
 & \qquad \qquad \qquad (B-2) \\
 y(t_i) = & C(t_i) \Phi(t_i, t_i) \underline{x}(t_i) + \left[C(t_i) \int_{t_i}^{t_i} \Phi(t_i, \tau) B(\tau) d\tau \right] \underline{u}(t_i) + \\
 & + C(t_i) \int_{t_i}^{t_i} \Phi(t_i, \tau) \underline{w}(\tau) d\tau + \underline{u}(t_i)
 \end{aligned}$$

If $t_i = t_i'$ (processor delay assumed zero), Equations (B-2) reverts back to the earlier form used in Equations (7-3). See Reference [2] for a more complete discussion on conversion of continuous to discrete-time systems with processor delay.

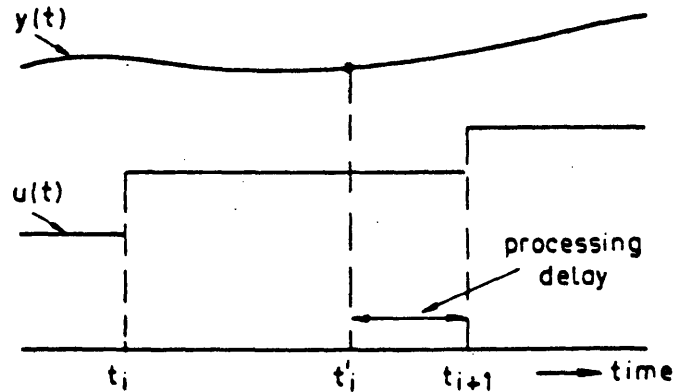


Figure B-1. Relationship of Control Actuation Instant t_i and Observation Instant t_i' []

APPENDIX C

EQUATIONS FOR CONTROL ALGORITHM DERIVATION

This Appendix lists the equations developed in Chapter 8 used in solving for the desired states θ_{1o}^{ss} , θ_{2o}^{ss} and the two switching instants required to take the float with arbitrary initial states θ_{1o} , θ_{2o} to the desired states in two torque intervals. A symbolic manipulation expert system, MACSYMA, was used to arrive at the equations below. The results were also written to a file in Fortran compatible code for inclusion in the simulation software.

MACSYMA was written at MIT in the LISP language, and runs on the Draper Laboratory IBM 3090 mainframe computer using a LISP compiler written at Draper Laboratory.

Since the MACSYMA output used symbols slightly different than those previously defined, a list of symbol conversions is presented in Table C-1.

Table C-1. Symbol conversions.

$DT = \Delta t$	$x = x_{ss}$
$TC = T_c$	$SSV = \theta_{2o}^{ss}$
$TD = \tau$	$SSP = \theta_{1o}^{ss}$
$STD = \dot{\tau}$	$INitlv = \theta_{2o}$
$\%E = \text{Exponential}$	$INitlp = \theta_{1o}$

$$\begin{aligned}
 \text{SSV} = & \left(2 I \text{ TC } \frac{C \text{ DT X}}{I} - 2 C \text{ DT } \frac{C \text{ DT}}{I} \text{ TC X} \right. \\
 & + \left((I - C \text{ DT}) \frac{C \text{ DT}}{I} - I \right) \text{ TD} + \left((C \text{ DT} - I) \frac{C \text{ DT}}{I} - I \right) \text{ TC} \Big) \\
 & \left. / (C I \frac{C \text{ DT}}{I} - C I) \right)
 \end{aligned} \tag{C-1}$$

$$\begin{aligned}
 \text{SSP} = & - \frac{C \text{ DT}}{I} \left(4 I^2 \text{ TC } \frac{C \text{ DT X}}{I} - 2 C^2 \text{ DT } \frac{C \text{ DT}}{I} \text{ TC X} \right. \\
 & + (4 C^2 \text{ DT} - 4 C \text{ DT } I) \frac{C \text{ DT}}{I} \text{ TC X} + ((2 I^2 - 2 C \text{ DT } I + C^2 \text{ DT}) \frac{C \text{ DT}}{I} \\
 & - 2 I) \text{ TD} + ((-2 I^2 + 2 C \text{ DT } I - C^2 \text{ DT}) \frac{C \text{ DT}}{I} - 2 I) \text{ TC} \\
 & \left. + ((2 C^2 \text{ DT } I - 2 C I^2) \frac{C \text{ DT}}{I} + 2 C I^2) \text{SSV} / (2 C^3 \text{ DT}) \right)
 \end{aligned} \tag{C-2}$$

$$\begin{aligned}
 \text{SSV} = & \frac{2 C \text{ DT}}{I} \left(2 \text{ TC } \frac{C \text{ DT X2}}{I} + \frac{C \text{ DT}}{I} + 2 \text{ TC } \frac{C \text{ DT X1}}{I} \right. \\
 & + (\frac{2 C \text{ DT}}{I} - I) \text{ TD} + (- \frac{2 C \text{ DT}}{I} - 2 \frac{C \text{ DT}}{I} - I) \text{ TC} + C \text{ INITLV} \Big) / C
 \end{aligned} \tag{C-3}$$

$$\begin{aligned}
SSP = & - \frac{2 C DT}{I} \frac{1}{2 I TC} \frac{1}{2 E} - \frac{C DT X2}{I} \frac{1}{2 E} + \frac{C DT}{I} \frac{1}{2 E} - \frac{2 C DT}{I} \frac{1}{2 E} \frac{1}{TC X2} \\
& + \frac{C DT X1}{I} \frac{1}{2 E} - \frac{2 C DT}{I} \frac{1}{2 E} \frac{1}{TC X1} + \left(\frac{1}{I} - \frac{2 C DT}{I} \right) \frac{1}{2 E} - \frac{1}{I} TD \\
& + \left(\frac{2 C DT}{I} - \frac{1}{I} \right) \frac{1}{2 E} - \frac{2 I}{I} \frac{1}{2 E} - \frac{1}{I} TC + \left(\frac{C I}{I} - \frac{C I}{I} \right) \frac{1}{2 E}) INITLV \\
& - \frac{2 C DT}{I} \frac{1}{2 E} \frac{1}{INITLP} \frac{1}{C}
\end{aligned} \tag{C-4}$$

$$\begin{aligned}
X2 = & \left(\frac{1}{I} \log \left(- \frac{2 TC}{I} \frac{1}{2 E} + \left(\frac{1}{2 E} - 1 \right) TD \right) \right. \\
& + \left(- \frac{2 C DT}{I} - \frac{2 C DT}{I} - 1 \right) TC - \frac{C}{I} \frac{1}{2 E} SSV + \frac{C}{I} \frac{1}{2 E} INITLV \bigg) / (2 TC) - \frac{C DT}{I} \\
& / (C DT)
\end{aligned} \tag{C-5}$$

$$\begin{aligned}
0 = & \left(\frac{2 I}{I} TC \log \left(- \frac{2 TC}{I} \frac{1}{2 E} + \left(\frac{1}{2 E} - 1 \right) TD \right) \right. \\
& + \left(- \frac{2 C DT}{I} - \frac{2 C DT}{I} - 1 \right) TC - \frac{C}{I} \frac{1}{2 E} SSV + \frac{C}{I} \frac{1}{2 E} INITLV \bigg) / (2 TC) \\
& + \frac{2 C DT}{I} TC X1 + \frac{2 C DT}{I} TD - \frac{4 C DT}{I} TC - \frac{C I}{I} SSV + \frac{C I}{I} INITLV + \frac{C^2}{I} INITLP \frac{1}{C} - SSP
\end{aligned} \tag{C-6}$$

$$\begin{aligned}
& \frac{C \, DT \, X}{I} \quad \frac{C \, DT}{I} \\
SSV = & (4 \, C \, I \, TC \, \cancel{X}E - 4 \, C^2 \, DT \, \cancel{X}E \, TC \, X \\
& + ((2 \, C \, I - 2 \, C^2 \, DT) \, \cancel{X}E \, I - 2 \, C \, I) \, TD \\
& + ((2 \, C^2 \, DT - 2 \, C \, I) \, \cancel{X}E \, I - 2 \, C \, I) \, TC \\
& + ((-2 \, I^3 + 2 \, C \, DT \, I^2 - C^2 \, DT^2 \, I) \, \cancel{X}E \, I^3 + 2 \, I^3) \, STD) \\
& / (2 \, C^2 \, I \, \cancel{X}E \, I^2 - 2 \, C^2 \, I)
\end{aligned}$$

(C-7)

$$\begin{aligned}
& \frac{C \, DT}{I} \quad \frac{C \, DT \, X}{I} \quad \frac{C \, DT}{I} \\
SSP = & - \cancel{X}E \, I^2 (12 \, C \, I \, TC \, \cancel{X}E \, I^3 - 6 \, C^3 \, DT^2 \, \cancel{X}E \, TC \, X^2 \\
& + (12 \, C^3 \, DT^2 - 12 \, C^2 \, DT \, I) \, \cancel{X}E \, I^2 \, TC \, X \\
& + ((6 \, C \, I^2 - 6 \, C^2 \, DT \, I + 3 \, C^3 \, DT^2) \, \cancel{X}E \, I^3 - 6 \, C \, I^2) \, TD \\
& + ((-6 \, C \, I^2 + 6 \, C^2 \, DT \, I - 3 \, C^3 \, DT^2) \, \cancel{X}E \, I^3 - 6 \, C \, I^2) \, TC \\
& + ((-6 \, I^4 + 6 \, C \, DT \, I^3 - 3 \, C^2 \, DT^2 \, I + C^3 \, DT^3 \, I) \, \cancel{X}E \, I^4 + 6 \, I^4) \, STD \\
& + ((6 \, C^3 \, DT \, I - 6 \, C^2 \, I^2) \, \cancel{X}E \, I^3 + 6 \, C^2 \, I^2) \, SSV \, 1/(6 \, C^4 \, DT)
\end{aligned}$$

(C-8)

$$\begin{aligned}
SSV = & \frac{2 C DT}{I} (2 C TC \frac{1}{2} E) + \frac{C DT X2}{I} + \frac{C DT}{I} + \frac{C DT X1}{I} \\
& + (C \frac{1}{2} E - C) TD + (- C \frac{1}{2} E - 2 C \frac{1}{2} E - C) TC \\
& + ((C DT I - I^2) \frac{1}{2} E + C DT I \frac{1}{2} E + I^2 STD + C^2 INITLV)/C^2
\end{aligned} \tag{C-9}$$

$$\begin{aligned}
SSP = & - \frac{1}{2} E (2 C I TC \frac{1}{2} E) - 2 C^2 DT \frac{1}{2} E TC X2 \\
& + 2 C I TC \frac{1}{2} E - 2 C^2 DT \frac{1}{2} E TC X1 \\
& + ((C I - 2 C^2 DT) \frac{1}{2} E - C I) TD + ((2 C^2 DT - C I) \frac{1}{2} E \\
& - 2 C I \frac{1}{2} E - C I) TC + ((- I^3 + C DT I - C^2 DT I) \frac{1}{2} E \\
& + C DT I^2 \frac{1}{2} E + I^3 STD + (C I - C^2 I \frac{1}{2} E) INITLV \\
& - C \frac{1}{2} E INITLV)/C^3
\end{aligned} \tag{C-10}$$

$$\begin{aligned}
 X2 = & \left(I \log \left(- \frac{2 C TC}{I} \frac{ZE}{I} + \frac{C DT X1}{I} + \frac{2 C DT}{I} - C \right) TD \right. \\
 & + \left(- C \frac{ZE}{I} - 2 C \frac{ZE}{I} - C \right) TC + \left(\frac{C DT I}{I} - I \right) \frac{ZE}{I} \\
 & + C DT I \frac{ZE}{I} + I^2 STD - C^2 \frac{ZE}{I} SSV + C^2 INITLV) / (2 C TC) - C DT) \\
 & / (C DT)
 \end{aligned}
 \tag{C-11}$$

$$\begin{aligned}
 0 = & \left(2 I TC \log \left(- \frac{2 C TC}{I} \frac{ZE}{I} + \frac{C DT X1}{I} + \frac{2 C DT}{I} - C \right) TD \right. \\
 & + \left(- C \frac{ZE}{I} - 2 C \frac{ZE}{I} - C \right) TC + \left(\frac{C DT I}{I} - I \right) \frac{ZE}{I} \\
 & + C DT I \frac{ZE}{I} + I^2 STD - C^2 \frac{ZE}{I} SSV + C^2 INITLV) / (2 C TC) \\
 & + 2 C DT TC X1 + 2 C DT TD - 4 C DT TC + C DT^2 I STD - C I SSV + C I INITLV \\
 & + C^2 INITLP) / C^2 - SSP
 \end{aligned}
 \tag{C-12}$$

REFERENCES

1. "CIGTF Tutorial on TGG," C. S. Draper Laboratory Report P-1923, 1984.
2. Goodwin, S. H., "Preliminary Performance Results of a Permanent Magnet Torque Generator in a Gyroscope," C. S. Draper Laboratory Report E-1479, March 1964.
3. Dauwalter, C. R., Private Communication, March 1989.
4. Musoff, H., "Design of Optimal Torque-to-Balance Loops," C. S. Draper Laboratory Report E-2447, 1970.
5. Hutchings, P. B., "Compensation of a Sample Data Contractor Servo," Master's Thesis, M.I.T., May 1959.
6. Koenigsberg, W. D., "Jump Resonant Frequency Islands in Non Linear Feedback Control Systems," C. S. Draper Laboratory Report P-002, September 1973.
7. Carson, R. C., Jr., "The Interpolator: A Method to Reduce Quantization Errors in Pulse Rebalance Instruments," C. S. Draper Laboratory Report P-128; April 1975.
8. Lory, C. B., "Compensation of Pulse-Rebalanced Inertial Instruments," Master's Thesis, M.I.T., January 1968.
9. Gelb, A. and Vander Velde, W., *Multiple-Input Describing Functions and Non-Linear System Design*, McGraw Hill Book Co., August 1968.
10. Carson, R. C., and Harris, R. A., "Digital Torquing of a PIPA Interim Report #1," C. S. Draper Laboratory Memo SOSS-SYS-70310, 4 December 1970.

11. Freier, L. J., Hockney, R. L., Koenigsberg, W. D., and Lemos, L. K., "P-95 Pulse-Width-Modulated Gyro Rebalance Loop Investigation," C. S. Draper Laboratory Report R-958, February 1976.
12. Salamin, E., "Microprocessor Controlled, Pulsewidth Modulated Rebalance Loop-Computer Simulation," C. S. Draper Laboratory Memo 76-15, May 1976.
13. Gootkind, D., "Non Linear Lead Compensation in the Gyro Torque Loop," C. S. Draper Laboratory Memo 77-10G-191, September 1977.
14. Conte, S. D., *Elementary Numerical Analysis*, McGraw-Hill Inc., 1965.
15. Kwakernak, H. and Sivan, R., *Linear Optimal Control Systems*, Wiley-Interscience, N.Y., 1972.
16. Gelb, A., Kasper, J. F., Nash, R. A., Price, C. F., and Sutherland, A. A., *Applied Optimal Estimation*, M.I.T. Press, Cambridge, Mass., 1974.
17. Franklin, G. F. and Powell, J. D., *Digital Control of Dynamic Systems*, Addison-Wesley Publishing Company, Reading, Mass., 1980.
18. Ash, M. E., "Generation of Planetary Ephemerides on an Electronic Computer," M.I.T. Lincoln Laboratory Technical Report 391, Chapter V, 2 June 1965.
19. Strang, G., *Introduction to Applied Mathematics*, Wellesley-Cambridge Press, Wellesley, MA, 1986.