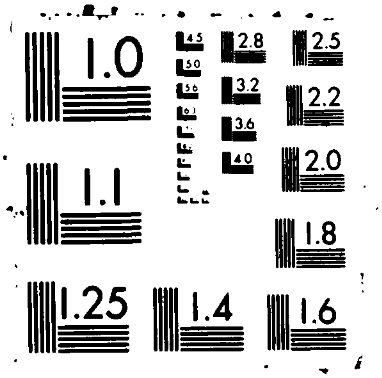
AD-A182 610 STOCHASTIC ADAPTIVE PARTICLE BEAM TRACKER USING MEER 1/3
FILTER FEEDBACK(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING
UNCLASSIFIED B A JOHNSON DEC 86 AFIT/GE/ENG/86D-27 F/G 14/2 NL


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

DTIC FILE COPY

AD-A182 610

STOCHASTIC ADAPTIVE PARTICLE BEAM TRACKER USING MEER FILTER FEEDBACK

THESIS

BRUCE A. JOHNSON
Captain, USAF

AFIT/GE/ENG/86D-27

DTIC
ELECTE
JUL 2 7 1987
S D
E

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

This document has been approved for public release and sale; its distribution is unlimited.

87 7 22 068

AFIT/GE/ENG/86D-27

STOCHASTIC ADAPTIVE PARTICLE BEAM TRACKER USING MEER FILTER FEEDBACK

THESIS

BRUCE A. JOHNSON
Captain, USAF

AFIT/GE/ENG/86D-27

This document has been approved
for public release and sale; its
distribution is unlimited.

AFIT/GE/ENG/86D-27

STOCHASTIC ADAPTIVE PARTICLE BEAM TRACKER USING MEER FILTER FEEDBACK

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Electrical Engineering

QUALITY INSPECTED

Accession For
NTIS GRA&I ☒
DTIC TAB ☐
Unannounced ☐
Justification
By
Distribution/
Availability Codes
Avail and/or
Dist Special
A-1

Bruce A. Johnson, B. S.
Captain, USAF

December 1986

Approval for public release; distribution unlimited

Preface

The purpose of this research was to continue the development of a stochastic controller that uses a Poisson space-time point process for feedback. The effort was motivated in part by the concurrent research being conducted at the Air Force Weapons Laboratory, Kirkland AFB, NM., and the Rome Air Development Center, Griffiss AFB, NY. The research was conducted with the primary goal of developing a neutral particle beam controller, but the issues discussed within this thesis should be valid for other applications.

I wish to express my appreciation to all those who helped me accomplish this research. Specifically, I would like to thank my readers, Lt Col Zdzislaw H. Lewantowicz and Capt Steve Rogers, and the creator of SOFE and SOFEPL, Stan Musick. Stan spent many late hours teaching me the finer points of SOFE/SOFEPL and numerous programming tricks on the Cyber computer. But, I am most indebted to my advisor, Dr. Peter S. Maybeck. His high expectations, technical guidance, suggestions and encouragement during my research kept me highly motivated even during the most difficult of times. His standards of excellent and outstanding leadership are matched only by his unending enthusiasm for his students and their works.

Last, I would like to thank Lauren, my parents and my close friends for their support and patience.

-- Bruce A. Johnson

Table of Contents

Page

Preface ii

List of Figures vi

List of Tables viii

List of Symbols x

Abstract xiii

I. Introduction 1

1.1 Background 2
1.2 Objectives 4
1.3 System Overview 6
1.4 Approach 8

II. Estimating the Position of the Beam - The Meer Filter 10

2.1 Particle Beam Model 10
2.2 Snyder-Fishman Filter 14
2.3 Multiple Model Adaptive Estimator . . 17
2.4 Pruning the Hypothesis Tree . . . 21
  2.4.1 The Best Half Method . . . 22
  2.4.2 The Merge Method 22
2.5 Simplifying the Meer Filter . . . 26
2.6 Summary 29

III. Controller Designs 30

3.1 The Target Model 31
3.2 Proportional Gain Controller . . . 34
  3.2.1 Proportional Gain Regulator . . 34
  3.2.2 Proportional Gain Tracker . . 38
3.3 Proportional-Plus-Integral Controller . 45
  3.3.1 PI Regulator 45
  3.3.2 PI Tracker 50
3.4 Reviewing the Cost Weighting Matrices . 56
3.5 Multiple Model Adaptive Controller . . 60
  3.5.1 Derivation of the MMAC . . . 60
  3.5.2 The Multiple Model Adaptive Tracker 69
3.6 Summary 70

IV. The Analyses 72

4.1 The Method - Monte Carlo Simulation . 72
4.2 The Tools - SOFE and SOFEPL . . . 77
4.3 The Beam and Target Parameters . . 86

4.4 Performance Analyses 88
4.4.1 Evaluating Alternative Weighting Matrices 89
4.4.2 Evaluating the Simplified Meer Filter 89
4.4.3 Sensitivity Analysis . . . 90
4.4.4 Robustness Analysis . . . 91
4.4.5 Evaluation the MMAC . . . 91
4.5 Summary 93

V. Results of the Monte Carlo Analyses . . 93

5.1 Evaluating Alternative Weighting Matrices 95
5.2 Evaluating the Simplified Meer Filter . 99
5.3 Sensitivity Analysis 101
5.3.1 Beam Time Constant 103
5.3.2 Beam Propagation Noise Strength . 104
5.3.3 Beam Dispersion 106
5.3.4 Signal to Noise Ratio . . . 107
5.3.5 Expected Number of Signal-Induced Events . . 108
5.3.6 Target Time Constant . . . 110
5.3.7 Target Dynamics Noise Strength . 114
5.3.8 Evaluating τ_T and Q_T with the Target Acceleration Power Density Maximum Set to 10.0 . 115
5.3.9 Target Measurement Noise Variance 115
5.4 Robustness Analysis 116
5.4.1 Beam Time Constant 117
5.4.2 Beam Propagation Noise Strength . 125
5.4.3 Beam Dispersion 125
5.4.4 Signal to Noise Ratio . . . 126
5.4.5 Expected Number of Signal-Induced Events . . 126
5.4.6 Target Time Constant . . . 127
5.4.7 Target Dynamics Noise Strength . 128
5.4.8 Evaluating τ_T and Q_T with the Target Acceleration Power Density Maximum Set to 10.0 . 129
5.4.9 Target Measurement Noise Variance 129
5.5 Evaluation the MMAC 130
5.6 Conclusions 139

VI. PI Controller Stability Analysis . . . 140

6.1 Derivation of the PI Controller Transfer Functions 141
6.2 The Zero-Input Stability Analysis . . 148
6.3 The Stability of the Kalman and Snyder-Fishman Filter Equations . . . 153
6.4 Stability Insights on the Meer Filter Structure 158
6.5 Conclusions 163

VII. Conclusions and Recommendations 165

7.1 Conclusions 165
7.2 Recommendations 167

Bibliography 170

Appendix A: Alternate Cost Weighting Methods . 173

Appendix B: Outline of SOFE Operations . . . 181

Appendix C: Controller Pole/Zero Locations as a Function of Different Sample Rates . 188

Appendix D: Sample Data Used to Calculate the Statistics in Table 5-25 . . . 191

Vita 192

List of Figures

Figure Page

1-1 System Overview 7

2-1 Signal and Noise Rate Parameters Functions . 13

2-2 Hypothesis Tree 18

2-3 Multiple Model Adaptive Estimator . . . 19

2-4 "Best Half" Method 23

2-5 "Merge" Method 26

2-6 Effective Gain Multiplier Verses Residual Size 27

3-1 Proportional Gain Regulator 37

3-2 Proportional Gain Tracker 42

3-3 Pseudo-Integral Term 46

3-4 Proportional-Plus-Integral Controller . . 47

3-5 Adaptive Stochastic Feedback Controller . 62

3-6 Multiple Model Adaptive Controller . . 64

4-1 Macro Level Flow Chart of SOFE (Modified) . 80

5-1 RMS Tracking Error for U=1, X_{11}=100, X_{55}=10 . 98

5-2 Applied Control and Relative Error Between the Target and Beam 98

5-3 Target Shaping Filter 110

5-4 PSD Functions 111

5-5 The Effects of Different τ_T's on the PSD . 113

5-6 RMS Error Between the Target and Beam Position: τ_{Bf}=20.0, τ_{Bt}=19.2 119

5-7 Mean and ±1 Standard Deviation Envelope of the Error between the True and Filter Position States: τ_{Bf}=20.0, τ_{Bt}=19.2 . . 119

5-8 RMS Error Between the Target and Beam Position: τ_{Bf}=20.0, τ_{Bt}=19.6 . . . 120

5-9 Mean and ±1 Standard Deviation Envelope of the Error between the True and Filter Position States: $\tau_{Bf}=20.0$, $\tau_{Bt}=19.6$ . . 120

5-10 RMS Error Between the Target and Beam Position: $\tau_{Bf}=20.0$, $\tau_{Bt}=20.0$ . . . 121

5-11 Mean and ±1 Standard Deviation Envelope of the Error between the True and Filter Position States: $\tau_{Bf}=20.0$, $\tau_{Bt}=20.0$ . . 121

5-12 RMS Error Between the Target and Beam Position: $\tau_{Bf}=20.0$, $\tau_{Bt}=20.4$ . . . 122

5-13 Mean and ±1 Standard Deviation Envelope of the Error between the True and Filter Position States: $\tau_{Bf}=20.0$, $\tau_{Bt}=20.4$ . . 122

5-14 RMS Error Between the Target and Beam Position: $\tau_{Bf}=20.0$, $\tau_{Bt}=21.0$ . . . 123

5-15 Mean and ±1 Standard Deviation Envelope of the Error between the True and Filter Position States: $\tau_{Bf}=20.0$, $\tau_{Bt}=21.0$ . . 124

5-16 MMAC RMS Error Between the Target and the Beam 133

5-17 Baseline RMS Error 133

5-18 MMAC Probabilistic Weighting from Monte Carlo Runs #1 and #2 136

6-1 Deterministic PI Controller 143

6-2 Stochastic PI Controller 146

6-3 Root Locus of the Stochastic PI Controller; Varying the Meer Filter Sample Rate . . 151

6-4 True Beam Offset from the Filter Beam Model: $R_t=R_f=0.5$ 160

6-5 True Beam Offset from the Filter Beam Model: $R_t=0.5$, $R_f=2.0$ 161

List of Tables

Table Page

3-1 Calculating the PI Controller Gains . . 55

5-1 RMS Errors from the Cost Weighting Matrices Analysis 96

5-2 The Applied Control During the Second Monte Carlo Run 96

5-3 Results of the Reduced Meer Filter Analysis . 100

5-4 Nominal Conditions for the Sensitivity and Robustness Analysis 102

5-5 The Sensitivity Analysis Results for τ_B . 104

5-6 The Sensitivity Analysis Results for g^2 . 106

5-7 The Sensitivity Analysis Results for R . . 107

5-8 The Sensitivity Analysis Results for SNR . 108

5-9 The Sensitivity Analysis Results for n . . 109

5-10 The Sensitivity Analysis Results for τ_T . 112

5-11 The Sensitivity Analysis Results for Q_T . 114

5-12 The Sensitivity Analysis Results for PSD . 115

5-13 The Sensitivity Analysis Results for R_T . 116

5-14 The Robustness Analysis Results for τ_B . . 118

5-15 Sensitivity of the Controller Gains to Changes in τ_B 124

5-16 The Robustness Analysis Results for g^2 . . 125

5-17 The Robustness Analysis Results for R . . 125

5-18 The Robustness Analysis Results for SNR . 126

5-19 The Robustness Analysis Results for n . . 127

5-20 The Robustness Analysis Results for τ_T . . 128

5-21 The Robustness Analysis Results for Q_T . . 128

5-22 The Robustness Analysis Results for PSD . 129

5-23 The Robustness Analysis Results for R_T . . 130

5-24 Results of the MMAC Analysis 134

5-25 Adaptive Filter Time Response to Increasing and Decreasing Q_T 138

6-1 Stochastic Filter Pole/Zero Placement as a Function of Filter Gain 153

6-2 The Filter's Probabilistic Weight that an Event Was Signal-Induced as a Function of R_f and the Residual 162

List of Symbols

Notation

Vector - underlined, lower case
Matrix - underlined, upper case
Scalar - not underlined

Variables

$\underline{a}$ - uncertain parameter vector
$\underline{A}$ - residual Covariance Matrix
$\underline{B}$ - projection matrix from the control input to the state space
$\underline{C}$ - projection matrix from the state variable to the state solution, $\underline{y}=\underline{C}\underline{x}$
D - filter depth
$\underline{e}$ - tracker or regulator error
$\underline{E}$ - PI controller gain on the setpoint, $\underline{y}_d$
$\underline{F}$ - system's dynamics matrix
$\underline{g}$ - square root of the beam dynamics driving (propagation) noise strength
$\underline{G}$ - projection matrix from the propagation noise input to the state space (sometimes it will include the noise strength)
$\underline{G}_c$ - controller gain on on the state, $\underline{x}$
h - hypothesis sequence
$\underline{H}$ - projection matrix from the state space to the measurement space
$\underline{I}$ - identity matrix
J - cost function
k - an elemental filter/controller in a multiple model structure
K - total number of elemental filter/controllers in a multiple model structure
$\underline{K}$ - filter gain, usually a function of time
$\underline{K}_c$ - solution to the Riccati equation
L - length of photo-detector array
m - dimension of a matrix (row); the physical dimensionality of the problem
m_e - mean error
n - number of expected events; Monte Carlo run number (Chapter 4 only); population of the sample statistics (Chapters 5 and 6)
$\underline{n}$ - dynamic driving noise process
N - total number of Monte Carlo runs in a simulation
N_t - total number of observations (events)
p, Pr - probability
$\underline{P}$ - filter error covariance matrix
PSD - power spectral density
$\underline{q}$ - pseudo-integral term (state)
$\underline{Q}$ - dynamics noise strength

Q_T - target dynamics noise strength
$\underline{r}$ - a spatial location on the detector array (Chapter 2 only); filter residual (in Chapter 2: $\underline{r}^*$)
$\underline{R}$ - beam dispersion; measurement noise variance
$\underline{R}_T$ - target measurement noise variance
R^M - real Euclidean m-space
RMS - root-mean-square
s - s-domain variable
SNR - signal-to-noise ratio
t - continuous time
t_0 - initial time
t_F - final time
t_i - discrete time
Δt - sample period
$\underline{u}$ - control input
$\underline{U}$ - cost weighting matrix on the control input
v - variance (Chapter 4 only)
$\underline{v}$ - measurement noise process
$\underline{w}$ - white noise
$\underline{W}$ - continuous-time cost weighting matrix
$\underline{x}$ - state variable
$\underline{x}_B$ - beam state variable
$\underline{x}_T$ - target state variable
$\underline{X}$ - cost weighting matrix on the state variable
$\underline{y}$ - solution to the state equation, $\underline{y}=\underline{C}\underline{x}$
$\underline{y}_c$ - the controlled variable (beam state)
$\underline{y}_r$ - the reference variable (target position state)
$\underline{y}_d$ - PI controller setpoint
z - z-domain variable (Chapter 6 only)
$\underline{z}$ - measurement
$\underline{z}_B$ - beam measurement
$\underline{z}_T$ - target measurement
$\underline{Z}$ - measurement time history
$\underline{\alpha}$ - realization of the uncertain parameter, $\underline{a}$; dummy variable used in integration
$\underline{\beta}$ - a Wiener process
ϵ - some epsilon distance
Λ - maximum amplitude of the rate function
$\underline{\Phi}$ - state transition matrix
$\underline{\xi}$ - dummy variable used in integration
$\lambda_S(\underline{r},\underline{x},t)$ - signal rate parameter (a function of position and time)
$\lambda_S(t)$ - signal arrival rate (a function of time only)
$\lambda_N(\underline{r},\underline{x},t)$ - noise rate parameter (a function of position and time)
$\lambda_N(t)$ - noise arrival rate (a function of time only)
σ - standard deviation
π - pi≈3.141715926
$\underline{\Pi}$ - a variable used in calculating the PI controller gains
τ - time constant; time variable used in integration
τ_B - beam time constant
τ_T - target time constant
ω - frequency

Superscript

$\hat{}$ - mean or expected value
$+$ - after measurement update
$-$ - prior to measurement update

Subscript

a - augmented
A - acceleration
B - beam
c - controlled
d - discrete
f - filter model
F - final
i,j,k- the i-th, j-th or k-th time period, sample period or hypothesis
k - the k-th elemental filter/controller; the k-th scalar state
M - Meer filter
N - noise
P - position
r - reference (target)
S - signal
SF - Snyder-Fishman filter
t - true or truth model
T - target
V - velocity
ZOH - zero order hold device
0 - initial (zero-th)

Operators

$\mathcal{L}$ - Laplace transformation
$\mathcal{Z}$ - z-transformation

AFIT/GE/ENG/86D-27

Abstract

The goal of this research was to develop a realizable proportional-plus-integral (PI) feedback tracker to control a neutral particle beam. The design is based on detecting the photo-electron events that are emitted from a laser-excited particle beam and the observed events are used by a Meer filter to locate the beam's centerline. The observed events are modeled by a Poisson space-time process and are composed of both signal- and noise-induced events. The Meer filter is a stochastic multiple model adaptive estimator which is composed of a bank of Snyder-Fishman filters and is designed to distinguish the signal-induced events from the noise-induced events. A target model is developed from a Gauss-Markov acceleration process, and the target states are estimated by a Kalman filter. The "optimal" PI controller design is based on the linear quadratic (LQ) controller synthesis technique and the "assumed" certainty equivalence property, and the Kalman filter provides the reference (target) states while the Meer filter supplies controlled (beam) states. The objectives of the research were to (1) select the "best" cost weighting matrices that minimize the RMS tracker error and enhance robustness, (2) simplify the Meer filter for easier on-line usage, (3) complete full-

scale sensitivity and robustness analyses over all the Kalman and Meer filter parameters, and (4) develop on-line adaptive estimation of those parameters that greatly affect stability robustness and tracker performance. During the research, an apparent stability problem was uncovered, and a fifth objective was to identify the source of the instability, and to propose a solution that would insure stability during parameter variations.

I. INTRODUCTION

This research is motivated in part by the problems in neutral particle beam pointing and tracking currently being investigated at the Air Force Weapons Laboratory, and at the Rome Air Development Center. Their goal is not only to estimate the position and direction of the beam, but use that information in an optimal way to control the pointing of the beam.

A method for sensing the location has been proposed in which the beam is illuminated by one or more lasers [23]. At certain angles of intersection and at different particle velocities, the particle electrons absorb photons from the laser beam and attain a higher energy state. When the particle beam electrons relax and spontaneously decay to their ground energy state, they expend the energy as light. By detecting the light energy, the position of the beam can be inferred. If the light energy were to arrive at the photo-detector at a sufficient signal rate as to produce an observable current, it might be modeled by a continuous-time, Gaussian process as done in many communication and control type problems. But, the assumption upon which this thesis is based is that the photon events do not arrive at such a sufficient rate. Instead, a discrete, Poisson space-time process is used to describe the arrival of the individual signal-induced events (the photons) and the noise-induced events (caused by dark currents within the detector, or other outside sources of noise). A space-time point process is a stochastic process having as realization points with random coordinates in both time and space. [9]. For this application, the time between events will assume a conditional Poisson process composed of Gaussian spatially distributed signal-induced events, and uniformly distributed noise-induced events.

1.1 Background

In 1975, Snyder and Fishman [25] developed an estimation algorithm called the Snyder-Fishman filter. The filter is a minimum-mean-square estimator that for this application, estimates the position of maximum intensity from the arrival of the signal-induced events (all events are assumed to be signal induced). The filter equations appear very similar to the Kalman filter equations with the notable exception that the Snyder-Fishman filter is based on the Poisson space-time process and thus the arrival times of the events are not known a priori. Instead, the events arrive as part of the Poisson distributed process. In the absence of noise-induced events, the Snyder-Fishman filter provides good results, but Santiago [24] found the filter's performance is severely degraded in the presence of noise.

In 1982, Meer [18] developed an adaptive filter designed to estimate the position of maximum intensity from the arrival of signal-induced events that are corrupted by a statistically independent noise process. This was accomplished through a multiple model structure in which a bank of elemental Snyder-Fishman filters are based on different hypothesis sequences. The hypotheses define which observed events were assumed to be due to the signal process and which were assumed to be due to the noise process, and each elemental filter is allowed to process the observed events only when its associated hypothesis defined the event as being signal-induced. Meer was the first to apply the point process to the neutral particle beam problem, and the "Meer" filter outperformed the Snyder-Fishman filter in a noisy environment. This successful filter development was the primary breakthrough required in controlling the beam location.

In 1983, Zicker [27] conducted a feasibility study of a simple proportional gain controller, considering both a regulator design to null out variations in the beam's location and a tracker design to maintain the beam on a maneuvering target. Zicker synthesized his stochastic controller designs from a deterministic optimal LQ controller assuming full state feedback. An LQ controller is a controller design based on a linear system model which uses a quadratic performance index to define optimal control. Then, the states were replaced by their best estimates according to the principle of assumed certain equivalence [14]. The Meer filter was used to provide the beam state estimate while a Kalman filter was used to provided the target state estimate.

Over the next two years, Moose [20] and Jamerson [8] replaced Zicker's proportion gain controller with proportional-plus-integral (PI) controllers, and the effort began to move from a question of feasibility to realizability and performance potential. The PI controller was selected because it possessed several characteristics which make it ideally suited for the tracking problem. First, control is based on both the current as well as the integral of previous tracking errors, thus providing a type-one system. Second, type-one systems are able to reject constant unmodeled disturbances that arise from linearized models, and are able to handle non-zero setpoint control with zero steady state error. Although performance improved considerably, especially when the controller was confronted with unmodeled disturbances, it came at the expense of some additional sluggishness due to the additional integrator in the controller. Also, Jamerson replaced Zicker's simple target model, which was based on a Gauss-Markov position model, with a more realistic Gauss-Markov acceleration model.

1.2 Objectives

Much of the previous work was spent on developing a stochastic filter based on a Poisson space-time point process which models both discrete signal- and noise-induced events, and then demonstrating that it can be used for feedback control. The next logical question that remains is whether a practical implementable controller can be designed which has the stability robustness to handle "real world" plant variations. Therefore, the objectives of this thesis are: (1) to evaluate the quadratic cost function in the LQ synthesis of Jamerson's PI controller and select the best set of cost weighting matrices that minimize the RMS tracker error and enhances robustness, (2) to evaluate the possibilities of a simplified filter to replace the multiple model structure of the Meer filter in order to reduce the computational loading, (3) to complete a full-scale sensitivity and robustness analysis on the six beam parameters and the three target parameters within the Meer and Kalman filters, and (4) to develop on-line adaptive estimation for those parameters that greatly affect stability robustness and tracker performance. During the research, an apparent stability problem has been uncovered, and a fifth objective has become to identify the source of the instability and to propose a solution that insured stability during parameter variations. The initial objective of evaluating an alternate cost weighting technique, such as implicit model following was dropped when the apparent stability problem surfaced. Discussions that pertain to developing an alternate cost weighting technique remain within the thesis (see Appendix A) as an aid for future research.

1.3 System Overview

An overview of the tracker design that is used throughout the thesis is shown in Figure 1-1. The purpose of a tracker is to generate a control input, $\underline{u}(t_i)$, that minimizes the difference between the the controller variable, $\underline{y}_c(t)$, and the reference (i.e., target) variable, $\underline{y}_r(t)$. In other words, we want the particle beam to track the target position.

The target model is generated through a shaping filter which uses an exponentially time-correlated noise process to simulate the target acceleration. The target velocity and position states are generated by integrating the acceleration state with respect to time. The target sensor is modeled as a sample and hold device that obtains a noise-corrupted measurement of the target's position, $\underline{z}_T(t_i)$, at a regular, prespecifed sample rate. The measurements are used by the Kalman filter to generate a target state estimate, $\underline{\hat{x}}_T$.

The particle beam dynamics (also referred to as the plant dynamics) are modeled as an exponentially time-correlated position process, the output of a shaping filter having one dominate pole. The position of the beam is inferred through the observations detected at the surface of the photo-detector. These observations, $\underline{z}_B(t)$, are assumed to be well modeled by a Poisson space-time point process, and can be of either signal or noise origin. The Meer filter is designed to discriminate between the two types of

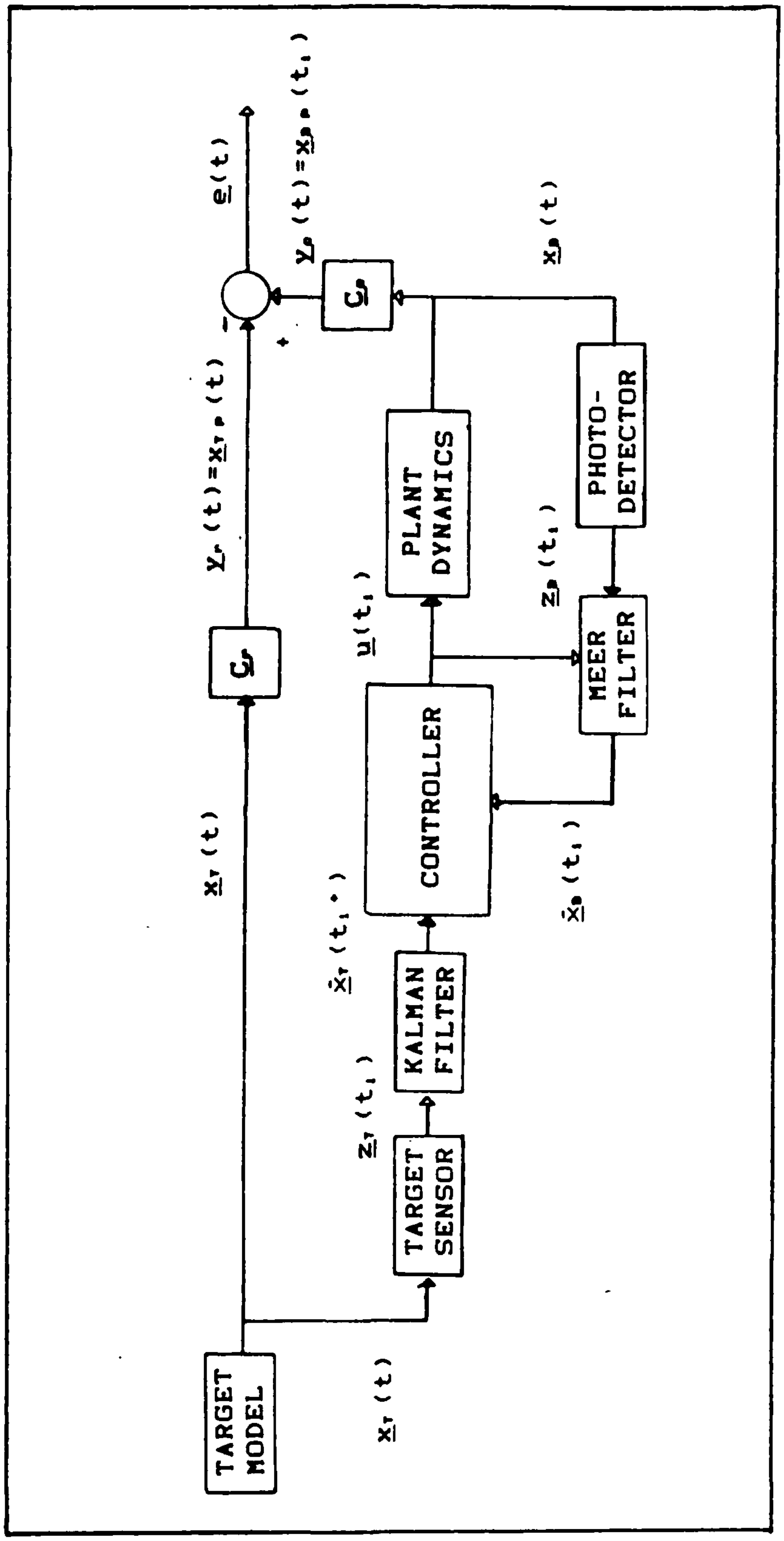


Figure 1-1. System Overview

events, and generate a beam state estimate, $\hat{\underline{x}}_B(t_i)$, from the signal-induced events.

The target state estimate and the beam state estimate are fed into the discrete-time controller algorithm to generate a command input to the plant. Because the target and the beam processes are assumed to be well modeled as Gaussian processes as produced by linear shaping filter, "near" optimal controller designs can be developed using the linear quadratic (LQ) full-state feedback controller synthesis process, and the assumed certainty equivalence property for incorporating filters into the loop. The proportional gain and proportional-plus-integral controllers are developed in theory (presented in Chapter 3), and the PI controller is implemented and evaluated on its own merit and within an adaptive multiple model controller structure.

The performance of the tracker is evaluated by calculating the tracker error, $\underline{e}(t)$, which is defined as the difference between the target and beam positions. Because of the adaptive nature of the controller and the time-variant nature of the Poisson space-time point process, the tracker error statistics have to be generated with a Monte Carlo simulation, as opposed to a covariance analysis or similar analytical methods.

1.4 Approach

The filter and controller theory is developed for the n-dimensional space, but for simplicity, the specific tracking system of interest will be designed and evaluated in one-space. This reduction in physical dimensionality shall simplify the physical insights about the design without loss of generality. Chapter 2 describes the Meer filter in considerable detail. The tracker model and controller designs are developed in Chapter 3. This includes the derivation of proportional gain, proportional-plus-integral and adaptive multiple model controller designs. Chapter 4 discusses the Monte Carlo simulation and the analyses to be performed, which are in accordance with the objectives stated in Section 1.2. Chapter 5 presents the results from the Monte Carlo analyses along with a detailed interpretation of the findings. One finding, an apparent instability problem found during the robustness analysis of the beam time constant is developed more fully in a separate chapter, Chapter 6. This chapter analyzes the stability and robustness characteristics of the deterministic and stochastic PI controller designs, the Kalman and Snyder-Fishman filters, and the Meer filter structure. The conclusions and recommendations for future research are provided in Chapter 7.

II. ESTIMATING THE POSITION OF THE BEAM - THE MEER FILTER

The first challenge in designing a controller for the pointing and tracking of a neutral particle beam is developing an estimator which can predict the location of the beam. Previous research resulted in defining the beam model and creating the Meer filter, a Multiple Model Adaptive Estimator (MMAE) composed of a bank of Snyder-Fishman filters. This chapter will explain the particle beam model, the Poisson space-time point process, and the Meer filter. The chapter concludes with a proposal to simplify the Meer filter under a given set of conditions.

2.1 Particle Beam Model

One recommended method for locating the center of a particle beam entails illuminating the beam by one or more lasers [23]. As a result, the particle electrons absorb the photons from the laser and jump to a higher, unstable energy state, and then spontaneously decay, returning to the ground energy state and dissipating the energy as photons. The photons radiate approximately in an isotropic manner and the position of the beam can be inferred from the location at which the photons strike a photo-detector array. The photoelectric events occur as a discrete process at random intervals. Because the optical sensors suffers from dark currents and background light, the sensors will corrupt the signal-generated data with erroneous, noise-induced events.

Meer noted that the signal-induced event could be modeled as a Poisson space-time point process on $[t_0,\infty) \times R^M$, and that this is a model upon which a Snyder-Fishman filter could be based.. Each event has associated with it the time of occurrence $t \in [t_0,\infty)$, and a spatial location $\underline{r} \in R^M$. A physical detector array will result in quantizing of R^M into a finite number of sections. For simplicity, the quantization is ignored without loss of generality, and the array is assumed to be continuous; that is, it is assumed that the value of $\underline{r}$ is known as an exact value from a continuous range of possible values.

The rate of occurrence is defined as the signal rate parameter, $\lambda_s(t,\underline{r},\underline{x}(t))$, and is assumed to have a spatial Gaussian function given as [18]:

$$\lambda_s(t,\underline{r},\underline{x}(t)) = \Lambda(t)\exp\{-[\underline{r}-\underline{H}(t)\underline{x}(t)]^T \underline{R}^{-1}(t)[\underline{r}-\underline{H}(t)\underline{x}(t)]/2\} \tag{2-1}$$

where

$\Lambda(t)$ is the maximum amplitude of the rate function
$\underline{R}$ is a symmetrical positive definite matrix defining the spread of the beam
$\underline{r}$ is a spatial location on the detector array
$\underline{H}(t)\underline{x}(t)$ is the signal-inferred beam centroid in R^M
$\underline{x}(t)$ is a stochastic process defining the state dynamics of the beam centroid
$\underline{H}(t)$ is an m x n projection matrix from the state space into the space of measured photoelectron events

The beam's spatial Gaussian function is a result of the beam's diffusion.

The state process is modeled as the Gauss-Markov output of the linear differential equation:

$$d\underline{x}(t) = \underline{F}(t)\underline{x}(t)dt + \underline{G}(t)d\underline{\beta}(t) \qquad (2\text{-}2)$$
$$\underline{x}(t_0) = \underline{x}_0$$

where $\underline{\beta}(t)$ is a Wiener process, whose hypothetical derivative is white Gaussian noise of unit strength, and $\underline{x}_0$ is a Gaussian random vector with mean $\underline{\hat{x}}_0$ and covariance $\underline{P}_0$.

Similar to the signal-induced events, the noise-induced events are modeled as a Poisson space-time point process on $[t_0, \infty) \times R^M$ with a noise rate parameter, $\lambda_N(t,\underline{r})$. The noise process is assumed to be statistically independent of the signal process [18]. The noise events are assumed to be uniformly distributed over the detector's field of view. The relationship between the signal- and noise-induced events is the signal-to-noise ratio, which is defined as the ratio of the average signal rate to the average noise rate. For a comparison of the two rate functions, see Figure 2-1. The signal-to-noise ratio is the ratio of the areas under the two curves in that figure.

Because both the signal and noise processes are Poisson point processes and independent of one another, their sum remains a Poisson point process. This can be shown by taking the product of the characteristic functions of the signal and noise processes. The resulting characteristic function will be a Poisson point process with the total rate

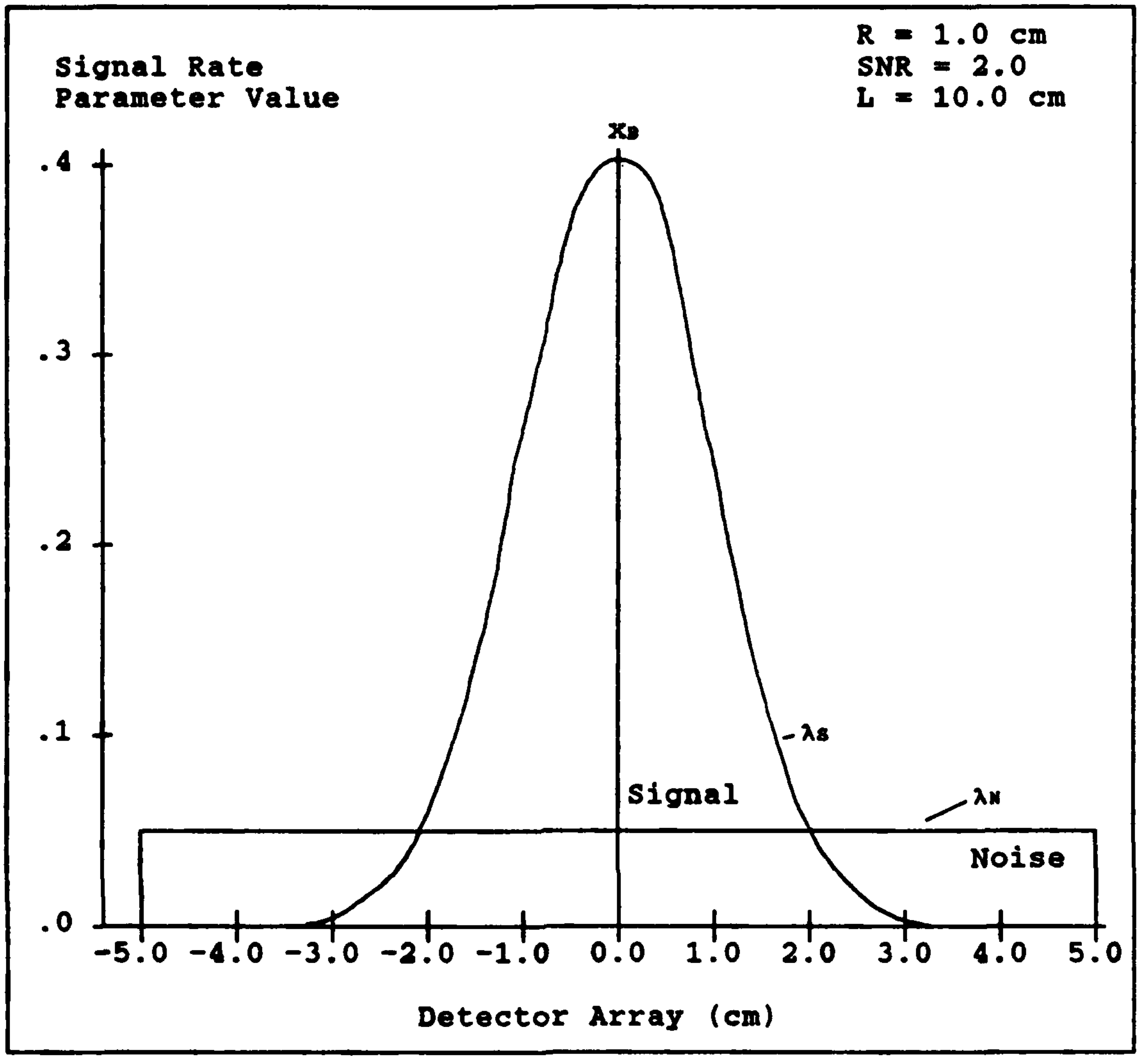


Figure 2-1. Signal and Noise Rate Parameter Functions

parameter equal to the sum of the signal and noise rate parameters:

$$\lambda(t,\underline{r}(t),\underline{x}(t)) = \lambda_S(t,\underline{r}(t),\underline{x}(t)) + \lambda_N(t,\underline{r}(t),\underline{x}(t)) \qquad (2\text{-}3)$$

2.2 Snyder-Fishman Filter

Snyder and Fishman developed an estimator which specifically handled the case of measurements that appear as a Poisson point process [25]. Their filter is similar in structure to the Kalman filter, but the Snyder-Fishman filter differs in two significant ways. The development of the Snyder-Fishman filter was based on the assumption that all measurements were signal-induced. In other words, the filter is limited to noise-free environments. The other difference is that the sample period is not fixed and the time between signal events is a Poisson distributed process.

With the absence of noise-induced events, the Snyder-Fishman filter estimates the beam's centroid as $\underline{H}(t)\underline{\hat{x}}(t)$, where $\underline{\hat{x}}(t)$ is the expected value of the beam states, $\underline{x}(t)$, conditioned on all the previous measurements,. The filter is described by the following differential equations [25]:

$$d\underline{\hat{x}}(t) = \underline{F}(t)\underline{\hat{x}}(t)dt + \int_{R^M} \underline{K}(t)[\underline{r}-\underline{H}(t)\underline{\hat{x}}(t)]\cdot N(dt \times d\underline{r}) + \underline{B}(t)\underline{u}(t)dt \qquad (2\text{-}4)$$

$$d\underline{P}(t) = \underline{F}(t)\underline{P}(t)dt + \underline{P}(t)\underline{F}^T(t)dt + \underline{G}(t)\underline{G}^T(t)dt - \int_{R^M} \underline{K}(t)\underline{H}(t)\underline{P}(t)N(dt \times d\underline{r}) \qquad (2\text{-}5)$$

$$\underline{K}(t) = \underline{P}(t)\underline{H}^T(t)[\underline{H}(t)\underline{P}(t)\underline{H}^T(t) + \underline{R}(t)]^{-1} \qquad (2\text{-}6)$$

where

$\hat{\underline{x}}(t_0) = \underline{x}_0$ and $\underline{P}(t_0) = \underline{P}_0$ are the initial conditions
$\underline{P}(t)$ is the filter-computed error covariance
$\underline{G}(t)\underline{G}^T(t) = \underline{G}(t)\underline{Q}(t)\underline{G}^T(t)$; $\underline{Q}(t) = \underline{I}$
$\underline{K}(t)$ is the filter gain

The notation found in Equations (2-4) and (2-5) involves counting integrals, where

$$\int_{R^M} f(t,\underline{r})N(dt \times d\underline{r}) = \begin{cases} 0 & N_t = 0 \\ \sum_{i=1}^{N_t} f(t_i,\underline{r}_i)\delta(t,t_i) & N_t \geq 1 \end{cases} \tag{2-7}$$

and $\delta(t,t_i)$ is a Kronecker delta. Simply stated, if no events are detected, then the integral equals zero. If an event is detected, the integral causes a jump discontinuity, $f(t,\underline{r})$, to be added to the solution of the differential equation for time t_i [25].

The propagation and update equations can be derived from Equations (2-4) and (2-5). The propagating equations for use between (signal-induced) events are:

$$d\hat{\underline{x}}(t) = \underline{F}(t)\hat{\underline{x}}(t)dt + \underline{B}(t)\underline{u}(t)dt \tag{2-8}$$

$$d\underline{P}(t) = \underline{F}(t)\underline{P}(t)dt + \underline{P}(t)\underline{F}^T(t)dt + \underline{G}(t)\underline{G}^T(t)dt \tag{2-9}$$

Equations (2-8) and (2-9) are actually implemented in the discrete controller designs (to be developed in Chapter 3) as stochastic difference equations:

$$\hat{\underline{x}}(t_{i+1}^-) = \underline{\Phi}(t_{i+1},t_i)\hat{\underline{x}}(t_i^+) + \underline{B}_d(t_i)\underline{u}(t_i) \tag{2-10}$$

$$\underline{P}(t_{i+1}^-) = \underline{\Phi}(t_{i+1},t_i)\underline{P}(t_i^+)\underline{\Phi}^T(t_{i+1},t_i) + \underline{G}_d(t_i)\underline{Q}_d(t_i)\underline{G}_d^T(t_i) \tag{2-11}$$

where $\underline{G}_d(t_i) = \underline{I}$ and the calculated as

$$\underline{Q}_d(t_i) = \int_{t_i}^{t_{i+1}} \underline{\Phi}(t_{i+1},\tau)\underline{G}(\tau)\underline{G}^T(\tau)\underline{\Phi}^T(t_{i+1},\tau)d\tau \tag{2-12}$$

When an event has been detected, a measurement update can take place as defined by the following equations:

$$\hat{\underline{x}}(t_i^+) = \hat{\underline{x}}(t_i^-) + \underline{K}(t_i)[\underline{z}_i - \underline{H}(t_i)\hat{\underline{x}}(t_i^-)] \tag{2-13}$$

$$\underline{P}(t_i^+) = \underline{P}(t_i^-) - \underline{K}(t_i)\underline{H}(t_i)\underline{P}(t_i^-) \tag{2-14}$$

where the Snyder-Fishman filter gain is defined by

$$\underline{K}(t_i) = \underline{P}(t_i^-)\underline{H}^T(t_i)[\underline{H}(t_i)\underline{P}(t_i^-)\underline{H}^T(t_i) + \underline{R}(t_i)]^{-1} \tag{2-15}$$

Equations (2-8) through (2-15) appear strikingly similar to the Kalman filter equations, except the measurement update times are not known a priori [25]. Instead, the measurement updates occur whenever an event is detected, and the time interval is defined as a random process.

2.3 Multiple Model Adaptive Estimator

The problem with using the Snyder-Fishman filter is that all the events are assumed to be signal-induced. To incorporate the noise-induced events, Meer developed a bank of Snyder-Fishman filters into a MMAE, where each filter in the bank is associated with an individual hypothesis. The hypothesis defines which events are signal-induced and which are noise-induced. The result is a hypothesis tree with each branch depicting a hypothesized noise/signal sequence, and each branch has an associated probability that its assumed sequence of events in fact occurred (see Figure 2-2). In other words, associated with each of these hypothesis sequences could be a specific Snyder-Fishman filter that performs measurement updates when it receives a hypothesized, signal-induced event and ignores the hypothesized noise-induced events (see Figure 2-3).

The state estimate out of each filter is expressed as

$$\hat{\underline{x}}_j(t) = E\{\underline{x}(t) \mid h_j^{N_t}, Z^{N_t}\} \tag{2-16}$$

where $\hat{\underline{x}}_j(t)$ is the expected value of the beam states, $\underline{x}_j(t)$, conditioned on the j-th assumed hypothesis time histories, $h_j^{N_t}$, and the observations history, Z^{N_t}, of events $(t_i, \underline{r}_i)$, where $i = 1, 2, \ldots, N_t$. Therefore, the overall state estimate of the MMAE is the probabilistically weighted average of the individual filter state estimates as expressed by

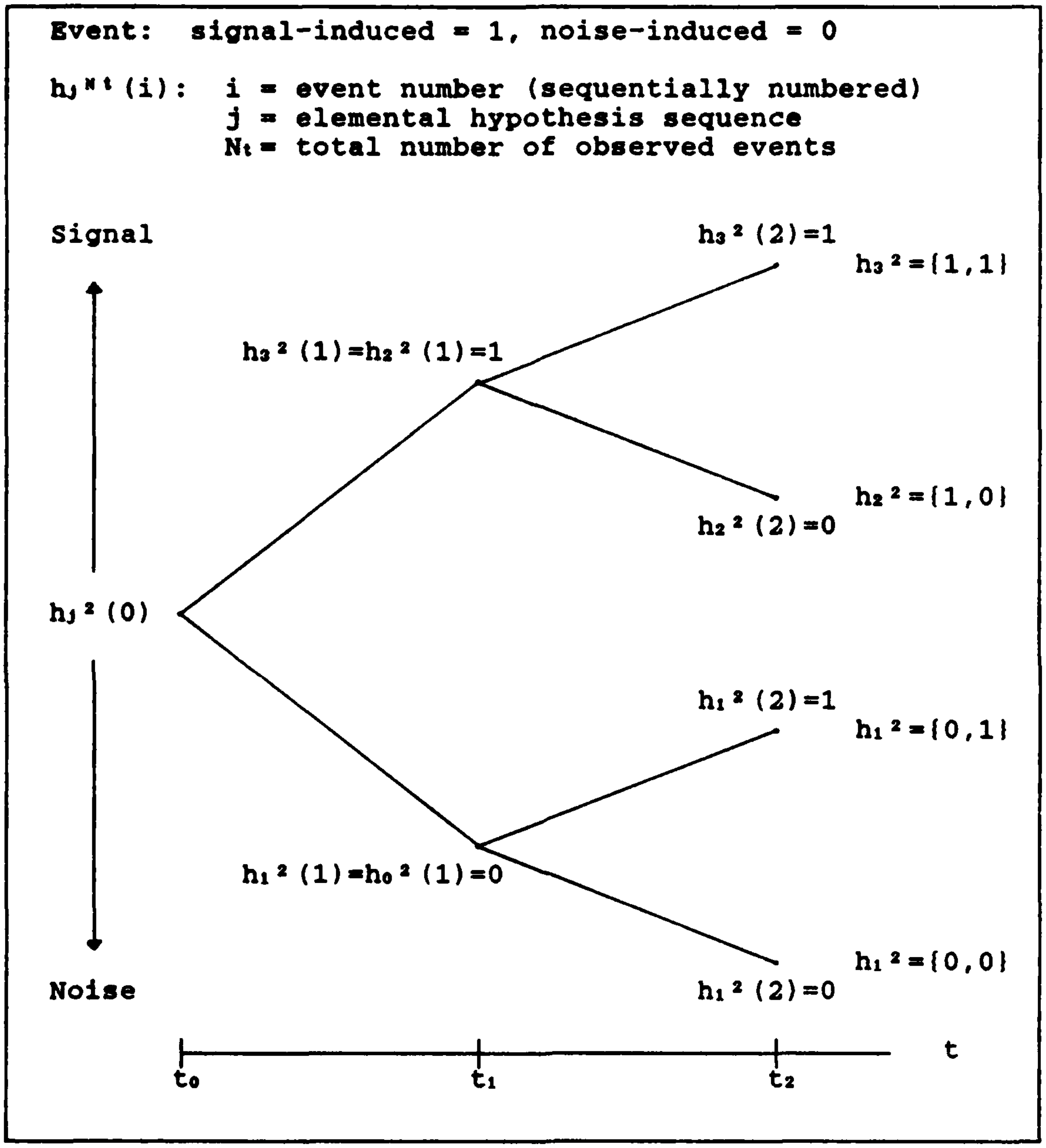

Event: signal-induced = 1, noise-induced = 0
hj^Nt(i): i = event number (sequentially numbered)
j = elemental hypothesis sequence
Nt = total number of observed events
Signal
Noise
hj²(0)
h3²(1)=h2²(1)=1
h1²(1)=h0²(1)=0
h3²(2)=1
h3²={1,1}
h2²(2)=0
h2²={1,0}
h1²(2)=1
h1²={0,1}
h1²(2)=0
h1²={0,0}
t0
t1
t2
t

Figure 2-2. Hypothesis Tree [18]

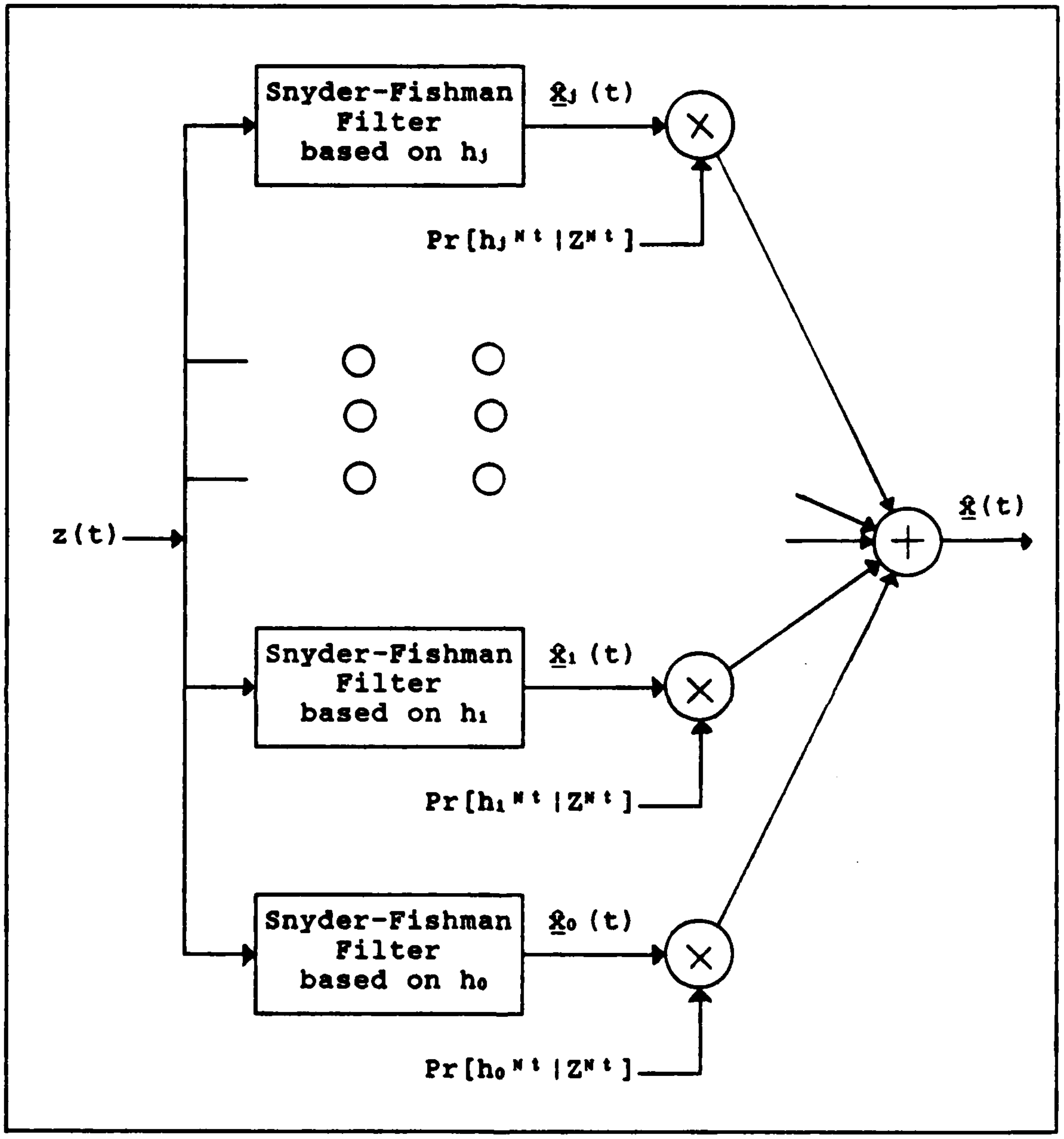

Snyder-Fishman Filter based on hj
x̂j (t)
Pr[hj^Nt | Z^Nt]
z(t)
x̂(t)
Snyder-Fishman Filter based on h1
x̂1 (t)
Pr[h1^Nt | Z^Nt]
Snyder-Fishman Filter based on h0
x̂0 (t)
Pr[h0^Nt | Z^Nt]

Figure 2-3. Multiple Model Adaptive Estimator

$$\underline{\hat{x}}(t) = \sum_{j=1}^{2^{N_t}-1} \Pr[h_j^{N_t} | Z^{N_t}] \cdot \underline{\hat{x}}_j(t) \tag{2-17}$$

with an MMAE computed error covariance defined as

$$\underline{P}(t) = \sum_{j=1}^{2^{N_t}-1} \Pr[h_j^{N_t} | Z^{N_t}] \cdot \{\underline{P}_j(t) + [\underline{\hat{x}}_j(t) - \underline{\hat{x}}(t)] \cdot [\underline{\hat{x}}_j(t) - \underline{\hat{x}}(t)]^T\} \tag{2-18}$$

The weighting probabilities, $\Pr[h_j^{N_t} | Z^{N_t}]$, are the conditional probabilities that a hypothesis sequence is correct, conditioned of the measurement history that has been observed.

The weighting probability starts at t_0, with $\Pr[h_j^0 | Z^0] = 1$. The subsequent weighting probabilities appear as:

$$\Pr[h_j^{N_t} | Z^{N_t}] = \frac{\hat{\lambda}_S(t_k, \underline{r}_k, \underline{\hat{x}}(t_k)) \text{ or } \hat{\lambda}_N(t_k, \underline{r}_k)}{\hat{\lambda}(t_k, \underline{r}_k, \underline{\hat{x}}(t_k))} \cdot \Pr[h_j^{N_t-1} | Z^{N_t-1}] \tag{2-19}$$

where $\hat{\lambda}_S(t_k, \underline{r}_k, \underline{\hat{x}}(t_k))$, $\hat{\lambda}_N(t_k, \underline{r}_k)$ and $\hat{\lambda}(t_k, \underline{r}_k, \underline{\hat{x}}(t_k)) = \hat{\lambda}_S(t_k, \underline{r}_k, \underline{\hat{x}}(t_k)) + \hat{\lambda}_N(t_k, \underline{r}_k)$ are estimates of the signal, noise and total rate parameters, and $\Pr[h_j^{N_t} | Z^{N_t}]$ is the probability of $h_j^{N_t}$ conditioned on the most recent event that occurred. The upward branches of the hypothesis tree are based on the assumption that the most recent event was signal-induced, and the downward branches of the hypothesis tree are based on the assumptions that the event was noise-induced (see Figure 2-2). The estimate of the signal rate parameter, $\hat{\lambda}_S(t_k, \underline{r}_k, \underline{\hat{x}}(t_k))$, appears in the numerator for a hypothetical signal-induced event, and is found by evaluating Equation (2-1) for $\underline{x}(t) = \underline{\hat{x}}(t)$. The estimate of the noise rate parameter, $\hat{\lambda}_N(t_k, \underline{r}_k)$, is used for a hypothetical noise-induced event, and $\hat{\lambda}_N = \lambda_N$. Because of the recursive nature of Equation (2-19), only knowledge of the most recent measurement is needed, rather than a growing memory of the entire history of measurements, for its evaluation [18].

2.4 Pruning the Hypothesis Tree

At this point, all the equations of the Meer filter have been derived in order to construct a full-scale particle beam estimator. The only problem is that we have developed a growing and soon to be unmanageable hypothesis tree, because the tree must grow two branches for each event to represent the possibility of either a signal or noise event. Therefore, if we have a total of N_t events, and N_t update cycles, there will be 2^{N_t} possible signal-versus-noise hypothesis sequences (frequently referred to as tree branches). The solution is to prune the hypothesis tree and keep the number of branches at a manageable size. The two proposed pruning algorithms are the "best half" and "merge" methods.

2.4.1 The Best Half Method [18] The "best half" method was the method Meer used in his dissertation to achieve an implementable filter. Initially, the hypothesis tree is allowed to grow until it reaches a prescribed memory depth of D. This produces a tree with 2^D hypothesis branches, with half of the tree originating from an assumed signal-induced event, and the other half emanating from a noise-induced event. With the (D+1)-th event, the conditional probability weighting factors associated with each branch are summed for each of these halves, and the more probable half is accepted, and the branches of the less probable half are eliminated (see Figure 2-4). The probability weight is normalized for the retained half of the tree so the sum of the retained probabilities will equal one.

The estimates are propagated forward until the next event, and the measurement update and pruning process occurs again. By doing so, the tree never grows past the memory depth of D.

2.4.2 The Merge Method An alternative algorithm to the "best half" method was proposed by Weiss and associates [26] and is designed to preserve some of the information that would be lost if the less probable half were eliminated. It is referred to as the "merge" method because

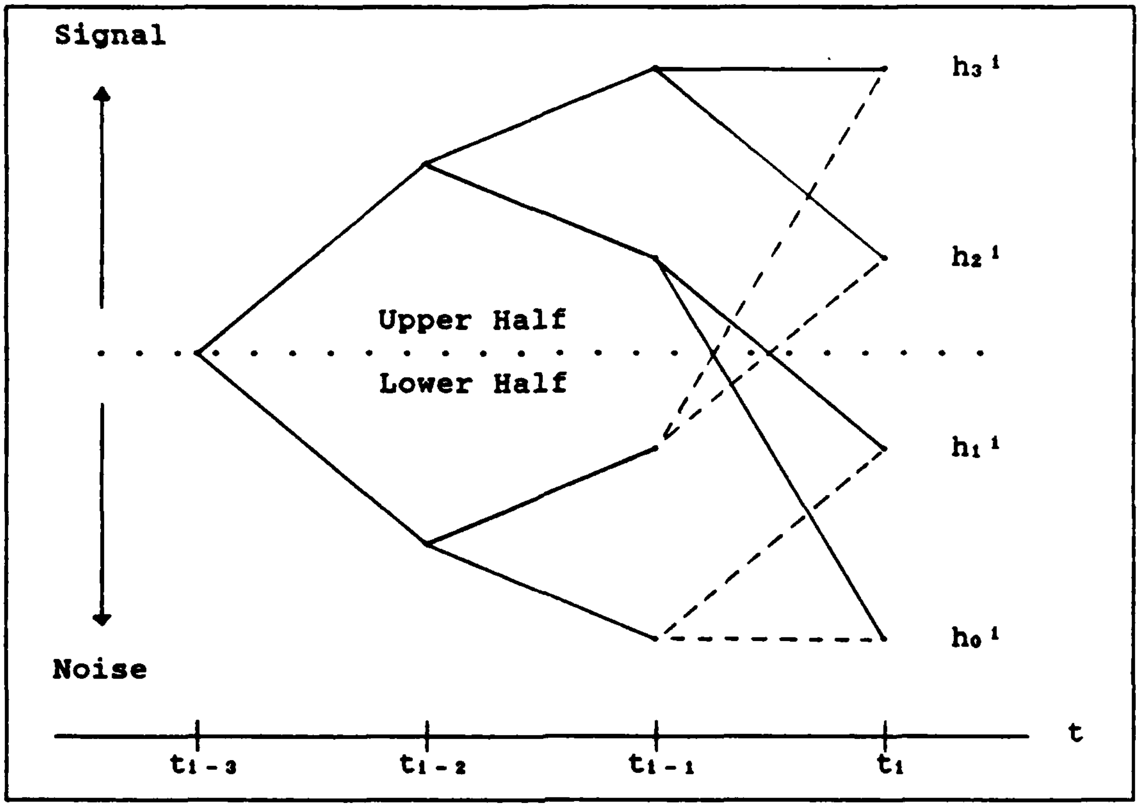


Figure 2-4. "Best Half" Method

the hypothesis sequences that are identical to each other in the assumptions made over the D-1 most recent events are paired up. That is, each pair of hypothesis sequences differ by the assumption made about the oldest event in the current sequence, and when the sequences are merged, only the oldest event is dropped. For example, if the hypothesis sequence is defined as $h_j^{N_t}$, where $j = 0,1,2,\ldots,2^{N_t}-1$, and N_t is the total number of observed events, one entire sequence could be written as

$$h_j^{N_t} = \{h_j^{N_t}(1),\ h_j^{N_t}(2),\ldots,\ h_j^{N_t}(N_t)\} = \{1,0,\ldots,1\} \qquad (2\text{-}20)$$

All signal-induced events are represented by a "1" while noise-induced events are labeled with a "0". Continuing with the example, it will be assumed that the memory depth is limited to two (D=2), and the total number of observed events is two ($N_t = 2$). Then the hypothesis sequences would be:

$$\begin{aligned}
h_3^2 &= \{h_3^2(1),\ h_3^2(2)\} = \{1,1\} \\
h_2^2 &= \{h_2^2(1),\ h_2^2(2)\} = \{1,0\} \\
h_1^2 &= \{h_1^2(1),\ h_1^2(2)\} = \{0,1\} \\
h_0^2 &= \{h_0^2(1),\ h_0^2(2)\} = \{0,0\}
\end{aligned} \tag{2-21}$$

As the next event is observed, $N_t = 3$, and there would be eight hypothesis sequences. If they are paired up by the D-1 most recent events, the sequenced pairs would be

$$\begin{aligned}
\textbf{Pair 1:}\quad h_7^3 &= \{[h_3^2(1),\ h_3^2(2)],\ h_7^3(3)\} = \{1,1,1\} \\
h_3^3 &= \{[h_1^2(1),\ h_1^2(2)],\ h_3^3(3)\} = \{0,1,1\} \\
\textbf{Pair 2:}\quad h_6^3 &= \{[h_2^2(1),\ h_2^2(2)],\ h_6^3(3)\} = \{1,0,1\} \\
h_2^3 &= \{[h_0^2(1),\ h_0^2(2)],\ h_2^3(3)\} = \{0,0,1\} \\
\textbf{Pair 3:}\quad h_5^3 &= \{[h_3^2(1),\ h_3^2(2)],\ h_5^3(3)\} = \{1,1,0\} \\
h_1^3 &= \{[h_1^2(1),\ h_1^2(2)],\ h_1^3(3)\} = \{0,1,0\} \\
\textbf{Pair 4:}\quad h_4^3 &= \{[h_2^2(1),\ h_2^2(2)],\ h_4^3(3)\} = \{1,0,0\} \\
h_0^3 &= \{[h_0^2(1),\ h_0^2(2)],\ h_0^3(3)\} = \{0,0,0\}
\end{aligned} \tag{2-22}$$

To merge the paired hypothesis sequences, the oldest, i.e. $(N_t - D)$-th, event is dropped, and the new weighted probability is the sum of the probabilities of the individual sequences that were merged. The state estimates and error covariances associated with each merged pair can be found using the following equations:

$$\underline{\hat{x}}_j'(t) = \{Pr[h_j^{Nt}|Z^{Nt}]\cdot\underline{\hat{x}}_j(t) + Pr[h_k^{Nt}|Z^{Nt}]\cdot\underline{\hat{x}}_k(t)\}/A \qquad (2\text{-}23)$$

$$\begin{aligned}\underline{P}_j'(t) = \{&Pr[h_j^{Nt}|Z^{Nt}]\cdot(\underline{P}_j(t)+[\underline{\hat{x}}_j(t)-\underline{\hat{x}}_j'(t)][\underline{\hat{x}}_j(t)-\underline{\hat{x}}_j'(t)]^T)\\ &+ Pr[h_k^{Nt}|Z^{Nt}]\cdot(\underline{P}_k(t)+[\underline{\hat{x}}_k(t)-\underline{\hat{x}}_j'(t)][\underline{\hat{x}}_k(t)-\underline{\hat{x}}_j'(t)]^T)\}/A\end{aligned} \qquad (2\text{-}24)$$

$$A = Pr[h_j'^{Nt}|Z^{Nt}] = Pr[h_j^{Nt}|Z^{Nt}] + Pr[h_k^{Nt}|Z^{Nt}] \qquad (2\text{-}25)$$

where

h_j^D and h_k^D denote the two different sequences within each pair (see Equation (2-19): $k = j + 2^D$ and $j = 0,1,\ldots,(2^D-1)$

D is the memory depth of the hypothesis tree after the pruning process has occurred

$\underline{\hat{x}}_j'$ and $\underline{P}_j'$ are the "merged" state estimates and error covariances as the number of elemental filters are reduced from 2^{D+1} to 2^D; $j' = 0,1,\ldots,(2^D-1)$

Equation (2-25) must be normalized so that the sum of the probabilities of each new estimate equals one. Figure 2-5 demonstrates the "merge" method. This method conceptually has an advantage over the "best half" method because it accounts for all possible time histories rather than deleting half of the branches from a decision tree. Unfortunately, it is computationally more burdensome.

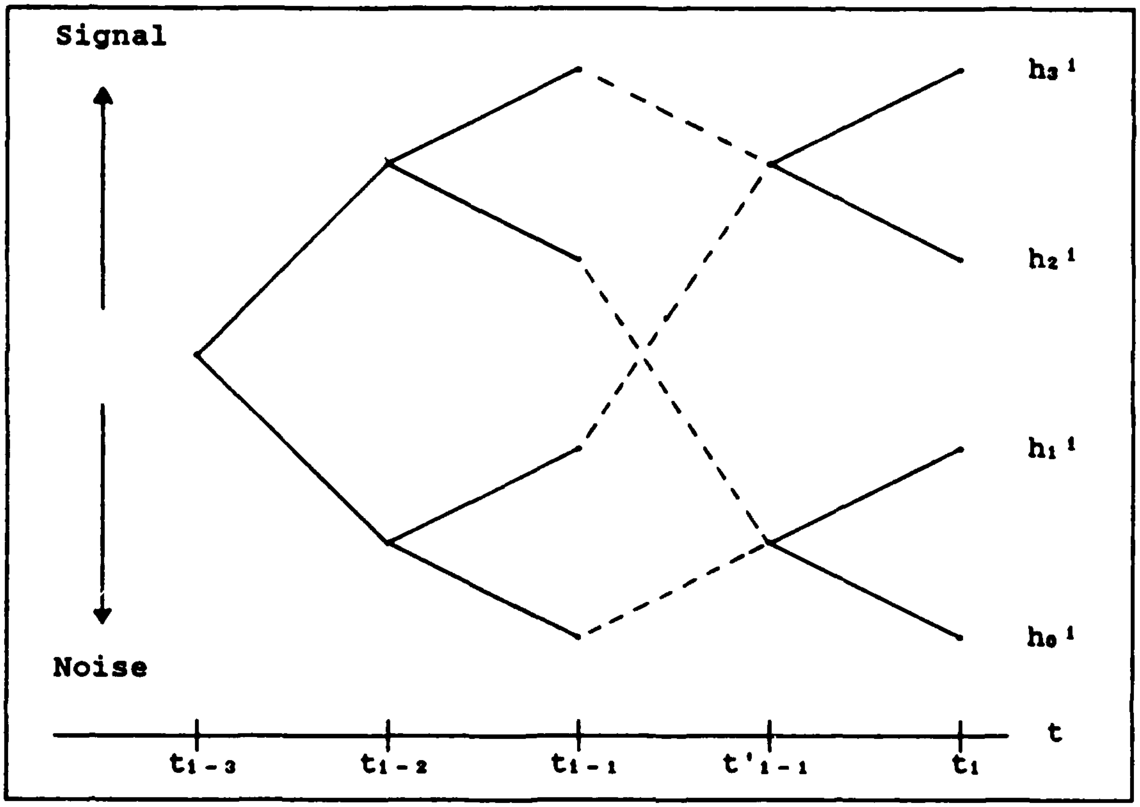


Figure 2-5. "Merge" Method

2.5 Simplifying the Meer Filter

Zicker ran a performance analysis on the Meer filter and found the Meer filter virtually insensitive to the depth parameter D. Varying D from D=1 to D=8 produced less than one percent change in rms errors. This is partially a function of using simple scalar first order Gauss-Markov model to define the beam centroid dynamics [17]. Assuming the first order beam model proves to be adequate, this insensitivity could be used to simplify the MMAE structure of the Meer filter by limiting the depth to D=1, and defining the filter gain, $\underline{K}(\underline{r}^*)$, as a function of the

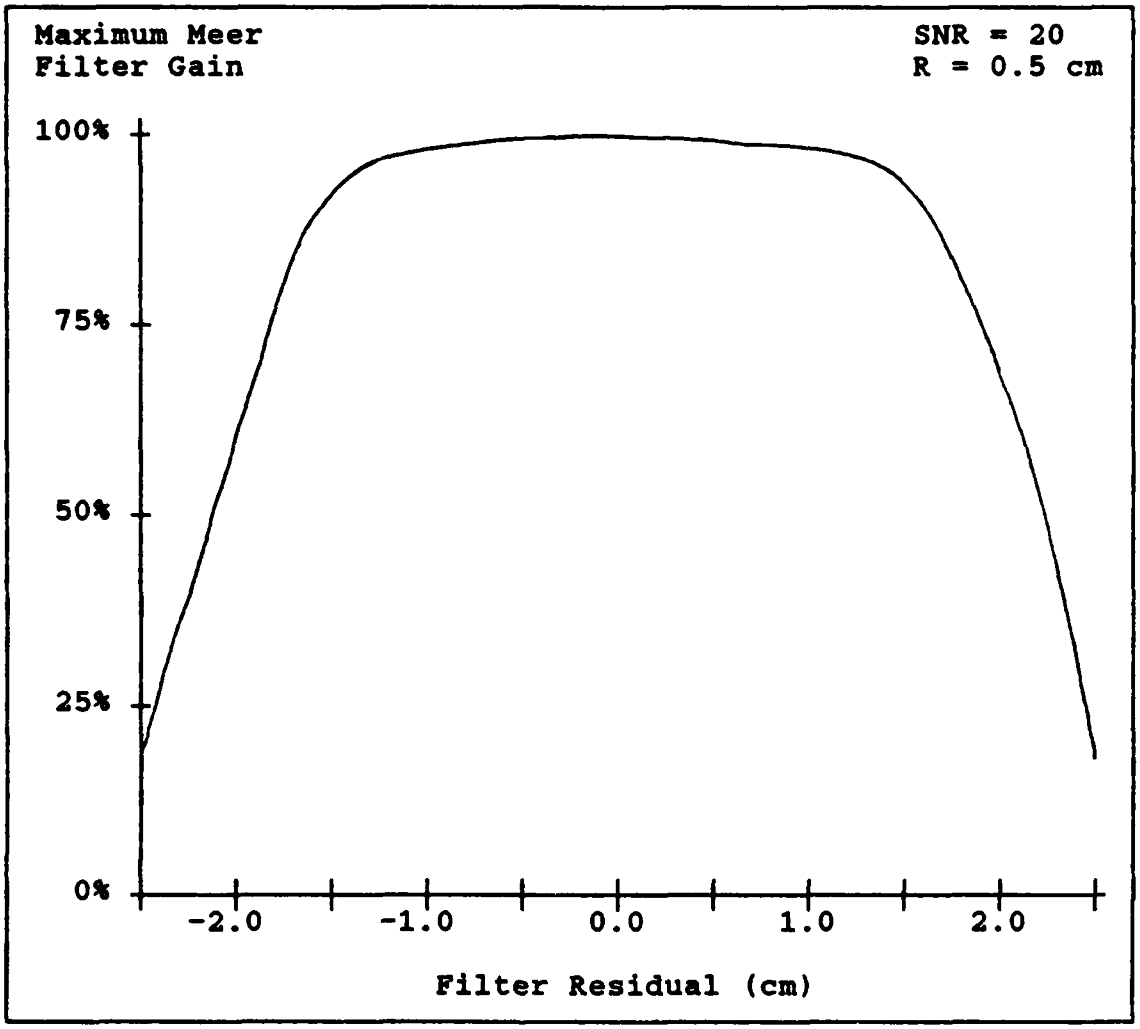


Figure 2-6. Meer Filter Gain Versus Residual Size

residual, $\underline{r}^*$. The insight comes from plotting $\underline{K}(\underline{r}^*)$ verses $\underline{r}^*$ as shown in Figure 2-6.

The mathematical development of the simplified algorithm can be established in the following manner. If the elemental Snyder-Fishman filter assumes that the event was signal-induced, the beam state estimate is

$$\hat{\underline{x}}_1(t_1^+) = \hat{\underline{x}}_1(t_1^-) + \underline{K}(t_1)[\underline{r} - \underline{H}_1(t_1)\hat{\underline{x}}_1(t_1^-)] \qquad (2\text{-}26)$$

The probability it was in fact a signal-induced event is

$$p_1(t_1) = \hat{\lambda}_S(t_1,\underline{r}_1,\underline{\hat{x}}(t_1)) \;/\; \hat{\lambda}(t_1,\underline{r}_1,\underline{\hat{x}}(t_1)) \qquad (2\text{-}27)$$

using the notation of Equation (2-19). If the elemental Snyder-Fishman filter assumed that the event was noise-induced, then $\underline{\hat{x}}_2(t_1^+) = \underline{\hat{x}}_2(t_1^-)$. The probability it was a noise-induced event is

$$p_2(t_1) = \hat{\lambda}_N(t_1,\underline{r}_1) \;/\; \hat{\lambda}(t_1,\underline{r}_1,\underline{\hat{x}}(t_1)) \qquad (2\text{-}28)$$

Therefore, the adaptive beam state estimate is

$$\begin{aligned} \underline{\hat{x}}_A(t_1^+) &= p_1(t_1)\underline{\hat{x}}_1(t_1^+) + p_2(t_1)\underline{\hat{x}}_2(t_1^+) \qquad (2\text{-}29) \\ &= \underline{\hat{x}}_A(t_1^-) + p_1(t_1)\underline{K}(t_1)[\underline{r} - \underline{H}_1(t_1)\underline{\hat{x}}_1(t_A^-)] \end{aligned}$$

where $\underline{\hat{x}}_A(t_1^-) = \underline{\hat{x}}_1(t_1^-) = \underline{\hat{x}}_2(t_1^-)$. Thus, the algorithm involves a single Snyder-Fishman-like filter, but with a modified gain of $[p_1(t_1)K(t_1)]$, and thus the gain is a function of the residual as seen in Figure 2-6.

Higher order Gauss-Markov beam models should have a greater sensitivity to D. Therefore, this simplification may be limited to this simple case.

2.7 Summary

The Meer filter is a Multiple Model Adaptive Estimator (MMAE) that is based on a space-time point process model for measurement events, and it can be used to estimate the position of a neutral particle beam. The Meer filter locates the beam by detecting the photo-electric event occurring from the spontaneously decaying excited particle within the beam. The time between events is a Poisson process, and this is properly modeled by the Snyder-Fishman filter. Meer introduced the MMAE structure to handle the noise-induced events which the Snyder-Fishman filter neglected. Because there is no deterministic mechanism for declaring whether an event is signal- or noise-induced, a hypothesis tree was developed. The depth of the tree was limited by incorporating either the "best half" or "merge" method. The results from Zicker's research indicated that the Meer filter could be reduced by limiting the depth to one, redefining the filter gain as a function of the residual, and eliminating the multiple model structure.

With the beam state estimator in hand, the next step is to develop the controller(s) to regulate the beam and track the target. The next chapter will address this.

III. CONTROLLER DESIGN

The next challenge is to design a controller which can place the beam on the target. Zicker developed the first controller for this application [27]. It was a proportional gain controller and was designed as an intuitive tool to gain insight into the tracker problem. But, because the proportional gain controller does not have type-one characteristics that allow rejection of unknown, constant disturbances that arise from linearized models, Moose developed a proportional-plus-integral (PI) controller [20]. Jamerson carried on Moose's research, and developed a different PI controller based on a more realistic target model [8]. Although the PI controller could reject constant unmodeled disturbances, it did so at the cost of additional sluggishness and had difficulty tracking a highly maneuverable target. The next step will be to develop a Multiple Model Adaptive Controller (MMAC) to handle parameter uncertainties caused by a highly manueverable target that can display rapid variations from a benign trajectory to evasive maneuvering.

All the controllers are based on the same fundamental assumptions: a linear system model, a deterministic optimal control law, full state feedback, and a quadratic cost criterion. The regulator's gain is based on a backward Riccati difference equation, assuming all the particle beam states are perfectly known. Then, by assumed certainty equivalence, the full state feedback is replaced by a filtered or "observed" state feedback provided by the Meer filter. The tracker portion of the controller is designed around the regulator, and directs the beam from the regulator's zero setpoint onto the target, using target state estimates generated by a Kalman filter.

Because the target model is used to developed the trackers, it will be presented first, followed by the proportional gain, proportional-plus-integral and the multiple model adaptive controller designs. The controllers are developed in the general vector form and then reduced to the specific one-dimensional tracker problem model that will be used during the analyses.

3.1 Target Model

Zicker [27] and Moose [20] designed their controllers for a simple target model composed of a first order Gauss-Markov position process. Research centered around feasibility studies and the emphasis was on gaining the knowledge and insight required to develop a realistic controller. Starting with Jamerson [8], the emphasis shifted to a more realistic target model built around a first order Gauss-Markov acceleration process. The linear, time-invariant state-space representation of this model is

$$\underline{\dot{x}}_T(t) = \underline{F}\,\underline{x}_T(t) + \underline{G}\,w_T(t)$$

$$\begin{bmatrix} \dot{x}_{TP}(t) \\ \dot{x}_{TV}(t) \\ \dot{x}_{TA}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -1/\tau_T \end{bmatrix} \begin{bmatrix} x_{TP}(t) \\ x_{TV}(t) \\ x_{TA}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ G \end{bmatrix} w_T(t) \quad (3-1)$$

where

$x_{TP}(t)$ is the target position state
$x_{TV}(t)$ is the target velocity state
$x_{TA}(t)$ is the target acceleration state
τ_T is the acceleration correlation time constant
$w_T(t)$ is a zero mean white Gaussian noise of strength Q_T associated with the target
$\underline{F}$ is the matrix that describes the system's dynamics
$\underline{G}$ is the matrix that maps the white noise effects into the state vector.

Equation (3-1) has three poles, of which two are at the origin of the s-plane. This equation of target motion is inherently astable, and the target is uncontrollable by the control input, u(t); therefore, the steady state Riccati equation used to generate controller gains may not have a solution [10,14]. All modes must be stabilizable to guarantee a solution of the steady-state Riccati equation. To insure a solution, the following sections will move the two poles at the origin of the s-plane to the left by some small epsilon. This will guarantee a solution for the steady state Riccati equation, but not affect the filter's development.

In order to use the target equation of motion defined by Equation (3-1), the target position will be measured as a discrete function defined by

$$z(t_i) = \underline{H}_T(t_i)\underline{x}_T(t_i) + v(t_i)$$

$$z(t_i) = [\ 1\ 0\ 0\ ] \begin{bmatrix} x_{TP}(t_i) \\ x_{TV}(t_i) \\ x_{TA}(t_i) \end{bmatrix} + v(t_i) \tag{3-2}$$

where $[\ x_{TP}(t_i)\ x_{TV}(t_i)\ x_{TA}(t_i)\ ]^T$ is the target state and $v(t_i)$ is a discrete-time zero-mean measurement corruption noise with a covariance R, and is assumed to be independent of the dynamics driving noise in Equation (3-1). How the target is actually detected and located is beyond the scope of this thesis. Between measurements, the state estimate and covariance matrix are propagated forward in time by

$$\hat{\underline{x}}_T(t_{i+1}^-) = \underline{\Phi}_T(t_{i+1}, t_i)\hat{\underline{x}}_T(t_i^+) \tag{3-3}$$

$$\underline{P}_T(t_{i+1}^-) = \underline{\Phi}_T(t_{i+1}, t_i)\underline{P}_T(t_{i+1})\underline{\Phi}_T^T(t_{i+1}, t_i) + \int_{t_i}^{t_{i+1}} \underline{\Phi}_T(t_{i+1}, \tau)\underline{G}_T(\tau)\underline{Q}_T(\tau)\underline{G}_T^T(\tau)\underline{\Phi}_T^T(t_{i+1}, \tau)d\tau \tag{3-4}$$

where $\underline{\Phi}_T(t_{i+1}, t_i)$ is the state transition matrix associated with $\underline{F}$ in Equation (3-1). During the measurement update, the state estimate and covariance matrix are updated by

$$\hat{\underline{x}}_T(t_i^+) = \hat{\underline{x}}_T(t_i^-) + \underline{K}(t_i)[\underline{z}(t_i) - \underline{H}_T(t_i)\hat{\underline{x}}_T(t_i^-)] \tag{3-5}$$

$$\underline{P}_T(t_i^+) = \underline{P}_T(t_i^-) - \underline{K}(t_i)\underline{H}_T(t_i)\underline{P}_T(t_i^-) \tag{3-6}$$

$$\underline{K}(t_i) = \underline{P}_T(t_i^-)\underline{H}_T^T(t_i)[\underline{H}_T(t_i)\underline{P}_T(t_i^-)\underline{H}_T^T(t_i) + \underline{R}_T(t_i)]^{-1} \tag{3-7}$$

where the initial conditions were

$$\hat{\underline{x}}(t_0) = \hat{\underline{x}}_0 \qquad\qquad \underline{P}(t_0) = \underline{P}_0$$

and the subscript "T" denotes "target."

As with any Kalman filter, if the measurement noise variance $R(t_i)$ is increased, then the filter must rely more on the dynamics and propagation equations. In a similar fashion, if the driving noise strength Q_T is increased, the filter must rely more heavily on the incoming measurement.

3.2 Proportional Gain Controller

If we had a linear quadratic Gaussian (LQG) stochastic controller problem, we could use the certainty equivalence property. But, the adapative mechanism in the Meer filter violates linearity; therefore, we must use the assumed certainty equivalence methodology [14]. It allows us to synthesze a sub-optimal, non-linear stochastic controller by generating the associated optimal deterministic full-state feedback controller, and then replacing the actual states with the conditional mean estimates from the stochastic filters.

3.2.1 Proportional Gain Regulator The function of the regulator is to drive the states of the beam to a zero setpoint defined as the center of the detector array. The linear discrete-time state equation that defines the beam is

$$\underline{x}_B(t_{i+1}) = \underline{\Phi}_B(t_{i+1},t_i)\underline{x}_B(t_i) + \underline{B}_d(t_i)\underline{u}_B(t_i) + \underline{G}_d\underline{w}_d(t_i) \quad (3\text{-}8)$$

where

$\underline{x}_B(t_i)$ is the position of the beam
$\underline{u}(t_i)$ is the control applied over the next period from t_i to t_{i+1}
$\underline{w}_d(t_i)$ is a zero mean white Gaussian discrete-time stochastic process of strength $\underline{Q}_d$, where

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

(Note that the Snyder and Fishman filter defines dynamics driving noise strength as $\underline{G}(t)\underline{G}^T(t)$; therefore, $\underline{Q}_B(t)=\underline{I}$.) The discrete, optimal deterministic control law, assuming perfect knowledge of $\underline{x}_B(t_i)$, is

$$\underline{u}^*(t_i) = -\underline{G}_c^*(t_i)\underline{x}_B(t_i) \tag{3-9}$$

where $\underline{G}_c^*(t_i)$ is the optimal controller gain. Because we do not have perfect knowledge of the beam position state, $\underline{x}_B(t_i)$, it is replaced by the conditional mean of the beam position state, $\hat{\underline{x}}_B(t_i)$, which is provided by the Meer filter. This is done in accordance with the assumed certainty equivalence synthesis methodology. The result is a discrete, optimal stochastic control law:

$$\underline{u}^*(t_i) = -\underline{G}_c^*(t_i)\hat{\underline{x}}_B(t_i) \tag{3-10}$$

The controller's optimality is defined by minimizing a quadratic cost function defined as

$$J = E\{ \sum_{i=0}^{N} \tfrac{1}{2}[\underline{x}^T(t_i)\underline{X}(t_i)\underline{x}(t_i) + \underline{u}^T(t_i)\underline{U}(t_i)\underline{u}(t_i)]$$

$$+ \tfrac{1}{2}\underline{x}^T(t_{N+1})\underline{X}_F(t_{N+1})\underline{x}(t_{N+1}) \qquad (3\text{-}11)$$

where

$\underline{x}(t_{N+1})$ is the final position to be achieved
$\underline{x}(t_i)$ is the position at time t_i
$\underline{X}$ and $\underline{X}_F$ are n-by-n constant weighting factors defined as a positive semi-definite matrices
$\underline{U}$ is an r-by-r weighting factor defined as a positive definite matrix

The controller gain, $\underline{G}_c(t_i)$, in Equation (3-10) is generated by solving the backwards Riccati recursion

$$\underline{G}_c^*(t_i) = [\underline{U}(t_i) + \underline{B}_d^T(t_i)\underline{K}_c(t_{i+1})\underline{B}_d(t_i)]^{-1}$$

$$\cdot[\underline{B}_d^T(t_i)\underline{K}_c(t_{i+1})\underline{\Phi}_B(t_{i+1},t_i)] \qquad (3\text{-}12)$$

$$\underline{K}_c(t_i) = \underline{X}(t_i) + \underline{\Phi}^T(t_{i+1},t_i)\underline{K}_c(t_{i+1})$$

$$\cdot[\underline{\Phi}^T(t_{i+1},t_i) - \underline{B}_d(t_i)]\cdot\underline{G}_c^*(t_i) \qquad (3\text{-}13)$$

solved backwards from the terminal condition

$$\underline{K}_c(t_{N+1}) = \underline{X}_F \qquad (3\text{-}14)$$

The resulting equations are sufficient to define the beam regulator as depicted in Figure 3-1.

For simplicity, the problem will be reduced to one physical dimension. That is, the beam and target models will be confined to one-space, R^1, and $\underline{B}_d$, $\underline{G}_d$ and $\underline{X}(t_i)$

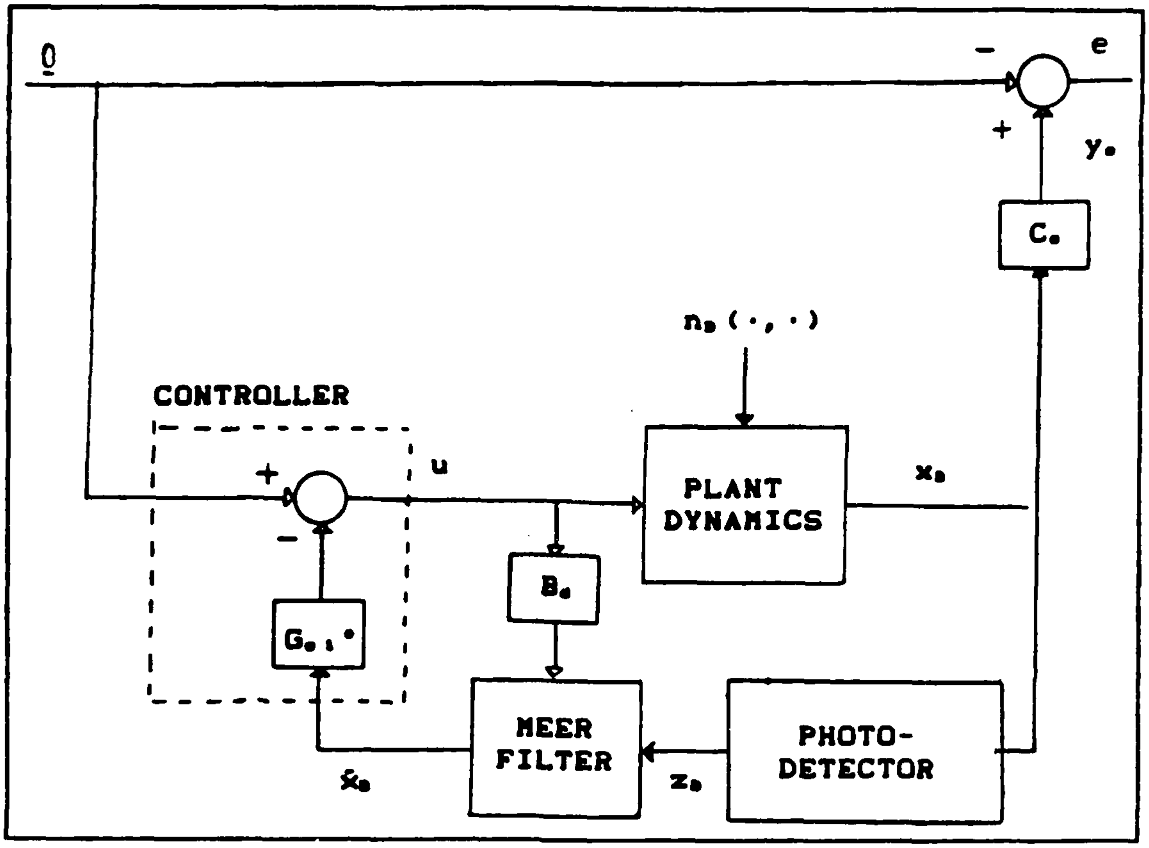


Figure 3-1. Proportional Gain Regulator

will be scalars equal to one. The reason the weighting matrix $\underline{X}(t_i)$ can be reduced to unity is that the X/U ratio is the important factor in determining the steady state gains, and, for the scalar case, either the numerator or denominator can be set to one, and the other weighting function adjusted to maintain the proper ratio. This simplification is limited to the scalar case.

With these simplifying assumptions, the general regulator equations reduce to:

$$x_s(t_{i+1}) = \Phi_s x_s(t_i) + u(t_i) \qquad (3\text{-}15)$$

$$\Phi_B = \exp[-(t_{i+1}-t_i)/\tau_B] = \exp[-\Delta t/\tau_B] \qquad (3\text{-}16)$$

$$u(t_i) = -\underline{G}_c^*(t_i)\hat{x}_B(t_i) \qquad (3\text{-}17)$$

$$G_c^*(t_i) = K_c(t_{i+1})\Phi_B \;/\; [U + K_c(t_{i+1})] \qquad (3\text{-}18)$$

Because this is a pure tracking problem without a well defined terminating point, the solution in found by determining the steady state solution to the backward Riccati difference equation (i.e., solving for the positive root in a quadratic solution):

$$K_c(t_i) = K_c(t_{i+1}) = K_c$$

$$K_c^2 + K_c[U(1 - \Phi_B^2) - 1] - U = 0 \qquad (3\text{-}19)$$

3.2.2 <u>Proportional Gain Tracker</u> Now we wish to extend the regulator to a tracking problem using the same LQG synthesis process. The goal is to minimize the difference between the controlled beam variables, $\underline{y}_c$, and the target or reference variables, $\underline{y}_r$. The variables are the linear transformations of the states of the beam and target states expressed as:

$$\underline{y}_c(t_i) = \underline{C}_c(t_i)\underline{x}_B(t_i) \qquad (3\text{-}20)$$

$$\underline{y}_r(t_i) = \underline{C}_r(t_i)\underline{x}_T(t_i) \tag{3-21}$$

and the difference between them is referred to as the tracking error, expressed as

$$\begin{aligned} \underline{e}(t_i) &= \underline{C}_c(t_i)\underline{x}_B(t_i) - \underline{C}_r(t_i)\underline{x}_T(t_i) \\ &= [\ \underline{C}_c(t_i) \quad -\underline{C}_r(t_i)\ ]\ [\ \underline{x}_B(t_i) \quad \underline{x}_T(t_i)\ ]^T \\ &= \underline{C}_a(t_i)\underline{x}_a(t_i) \end{aligned} \tag{3-22}$$

The tracking error shall be regulated to zero by way of minimizing the tracker augmented cost function defined as

$$\begin{aligned} J = E\{ \sum_{i=0}^{N} \tfrac{1}{2}[\underline{x}_a^T(t_i)\underline{X}_a(t_i)\underline{x}_a(t_i) + \underline{u}^T(t_i)\underline{U}(t_i)\underline{u}(t_i)] \\ + \tfrac{1}{2}\underline{x}_a^T(t_{N+1})\underline{X}_{fa}(t_{N+1})\underline{x}_a(t_{N+1}) \end{aligned} \tag{3-23}$$

where

$$\underline{X}_a(t_i) = \underline{C}_a^T(t_i)\underline{Y}_a(t_i)\underline{C}_a(t_i)$$

$$\underline{X}_{fa}(t_i) = \underline{C}_a^T(t_{N+1})\underline{Y}_{fa}(t_{N+1})\underline{C}_a(t_{N+1})$$

Assuming we will have perfect knowledge of both the beam states and target states, we can obtain the full-state proportional gain control law of

$$\begin{aligned} \underline{u}^*(t_i) &= -\ [\ \underline{G}_{c1}^*(t_i) \quad \underline{G}_{c2}^*(t_i)\ ]\ [\ \underline{x}_B(t_i) \quad \underline{x}_T(t_i)\ ]^T \\ &= -\ \underline{G}_{c1}^*(t_i)\underline{x}_B(t_i) - \underline{G}_{c2}^*(t_i)\underline{x}_T(t_i) \end{aligned} \tag{3-24}$$

The solution to the augmented gain matrix is solved by an augmented backward Riccati recursion:

$$[\ \underline{G}_{c1}{}^{*}\quad \underline{G}_{c2}{}^{*}\ ] = \left\{ \underline{U} + [\ \underline{B}_d{}^{T}\ \ \underline{0}\ ] \begin{bmatrix} \underline{K}_{c11} & \underline{K}_{c12} \\ \underline{K}^{T}{}_{c12} & \underline{K}_{c22} \end{bmatrix} \begin{bmatrix} \underline{B}_d \\ \underline{0} \end{bmatrix} \right\}^{-1}$$

$$\cdot\ [\ \underline{B}_d{}^{T}\ \ \underline{0}\ ] \begin{bmatrix} \underline{K}_{c11} & \underline{K}_{c12} \\ \underline{K}^{T}{}_{c12} & \underline{K}_{c22} \end{bmatrix} \begin{bmatrix} \underline{\Phi}_c & \underline{0} \\ \underline{0} & \underline{\Phi}_r \end{bmatrix}$$

$$= [\underline{U} + \underline{B}_d{}^{T}\underline{K}_{c11}\underline{B}_d]^{-1} \cdot [\ \underline{B}_d{}^{T}\underline{K}_{c11}\underline{\Phi}_c \quad \underline{B}_d{}^{T}\underline{K}_{c12}\underline{\Phi}_r\ ] \tag{3-25}$$

$$\underline{K}_{c11}(t_i) = \underline{C}_c{}^{T}(t_i)\underline{Y}(t_i)\underline{C}_c(t_i) + \underline{\Phi}_c{}^{T}\underline{K}_{c11}(t_{i+1})\underline{\Phi}_c - \underline{\Phi}_c{}^{T}\underline{K}_{c11}(t_{i+1})\underline{B}_d(t_i)\underline{G}_{c1}{}^{*}(t_i) \tag{3-26}$$

$$\underline{K}_{c12}(t_i) = -\underline{C}_c{}^{T}(t_i)\underline{Y}(t_i)\underline{C}_r(t_i) + \underline{\Phi}_c{}^{T}\underline{K}_{c12}(t_{i+1})\underline{\Phi}_r - \underline{G}_{c1}{}^{*T}(t_i)[\ \underline{B}_d{}^{T}(t_i)\underline{K}_{c12}(t_{i+1})\underline{\Phi}_r\ ] \tag{3-27}$$

where $\underline{\Phi}_c$ is the beam state transition matrix, $\underline{\Phi}_B(t_{i+1},t_i)$ is this application, $\underline{\Phi}_r$ is the target state transition matrix, $\underline{\Phi}_T(t_{i+1},t_i)$, and Equations (3-26) and (3-27) are solved backwards from the terminal conditions

$$\underline{K}_{c11}(t_{N+1}) = \underline{C}_c{}^{T}(t_{N+1})\underline{Y}_F(t_{N+1})\underline{C}_c(t_{N+1}) \tag{3-28}$$

$$\underline{K}_{c12}(t_{N+1}) = -\underline{C}_c{}^{T}(t_{N+1})\underline{Y}_F(t_{N+1})\underline{C}_r(t_{N+1}) \tag{3-29}$$

By carefully partitioning the augmented backward Riccati recursion equations, one can see that the solution for $\underline{G}_{c1}^*(t_i)$ is identical to the feedback gain, $\underline{G}^*(t_i)$, found in the regulator (Note: Equations (3-25) and (3-26) are similar to Equations (3-12) and (3-13)). Thus, $\underline{K}_{c11}$ and $\underline{G}_{c1}^*$ are independent of the reference variable being tracked, and they are found by solving the deterministic regulator design. The tracker gain, $\underline{G}_{c2}$, is determined from Equations (3-27) and (3-29), and is a function of the feedback gain and the reference variable:

$$\underline{G}_{c2}^*(t_i) = [\underline{U}(t_i) + \underline{B}_d^T(t_i)\underline{K}_{c11}(t_{i+1})\underline{B}_d(t_i)]^{-1} \cdot \underline{B}_d^T(t_i)\underline{K}_{c12}(t_{i+1})\underline{\Phi}_r(t_{i+1},t_i) \qquad (3\text{-}30)$$

The resulting tracker model is diagrammed in Figure 3-2.

Once again, the thesis problem will be reduced to one dimension, and the tracker error will be the scalar difference between the beam and target positions. As foretold, the regulator design will provide the feedback gain, $\underline{G}_{c1}^*$. The problem setup is as follows:

$$\underline{Y} = X_{11} \qquad (3\text{-}31)$$

$$\underline{C}_c = 1 \qquad (3\text{-}32)$$

$$\underline{C}_r = [\ 1\ \ 0\ \ 0\ ] \qquad (3\text{-}33)$$

$$\underline{C}_a = [\ \underline{C}_c\ \ -\underline{C}_r\ ] = [\ 1\ -1\ \ 0\ \ 0\ ] \qquad (3\text{-}34)$$

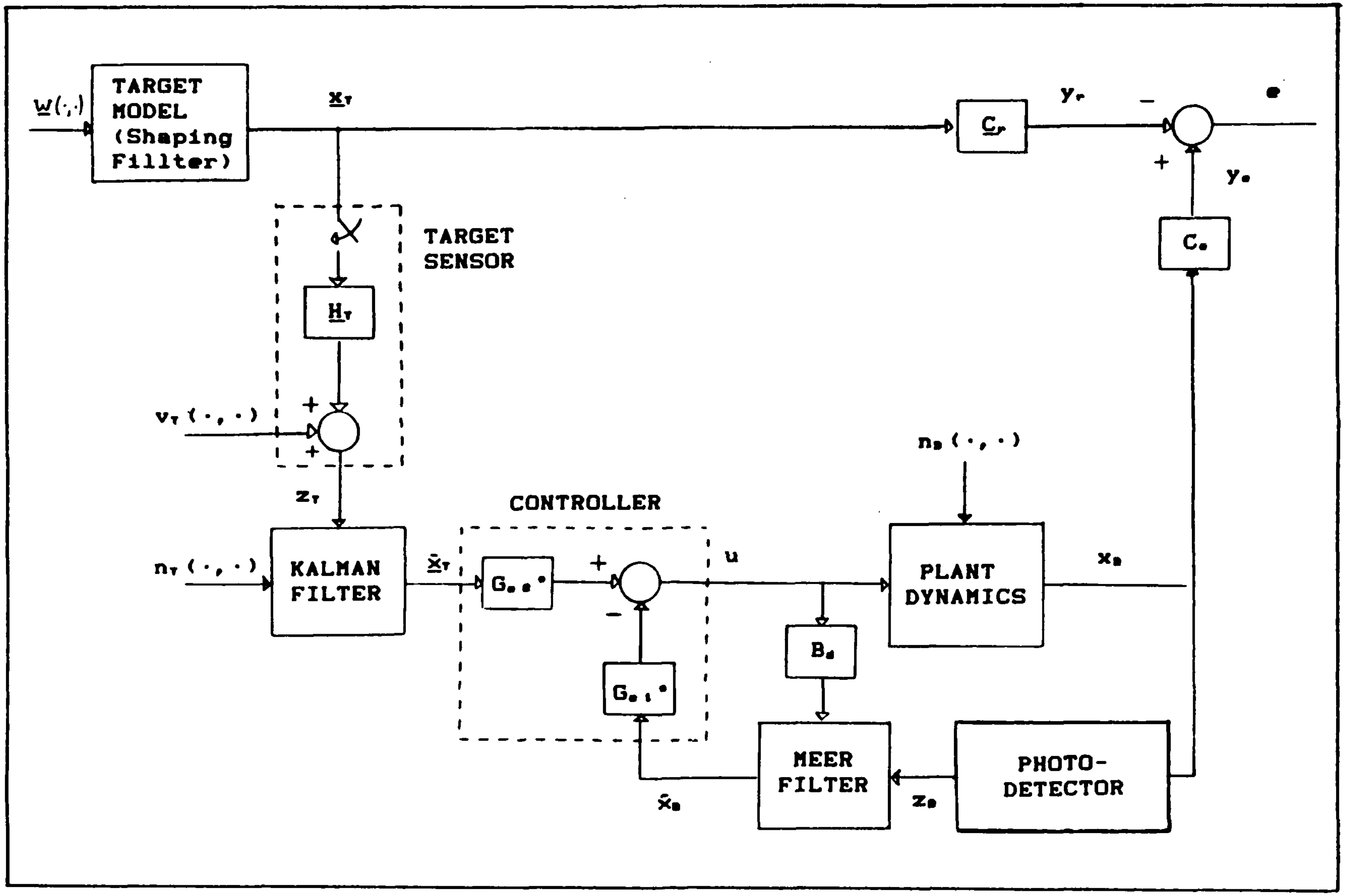


Figure 3-2. Proportional Gain Tracker

$$\underline{X}_a = \underline{C}_a{}^T \underline{Y} \underline{C}_a = \begin{bmatrix} X_{11} & -X_{11} & 0 & 0 \\ -X_{11} & X_{11} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{3-35}$$

$$u^*(t_i) = -[\ G_{c1}{}^* \ \ G_{c2P}{}^* \ \ G_{c2V}{}^* \ \ G_{c2A}{}^*\ ] \begin{bmatrix} x_B(t_i) \\ x_{TP}(t_i) \\ x_{TV}(t_i) \\ x_{TA}(t_i) \end{bmatrix} \tag{3-36}$$

The form of the augmented weighting matrix, $\underline{X}_a$, can be intuitively verified. We are trying to drive the quantity

$$e(t_i) = x_B(t_i) - x_{TP}(t_i) \tag{3-37}$$

towards zero, so the quadratic penalty placed on the error is

$$\tfrac{1}{2} X_{11}\, e(t_i)^2 = \tfrac{1}{2}[\ x_B(t_i) \ \ x_{TP}(t_i)\ ] \begin{bmatrix} X_{11} & -X_{11} \\ -X_{11} & X_{11} \end{bmatrix} \begin{bmatrix} x_B(t_i) \\ x_{TP}(t_i) \end{bmatrix} \tag{3-38}$$

which is the upper left block of Equation (3-35). The rest of the elements in $\underline{X}_a$ should be zero because we wish to disregard the velocity and acceleration components of the target when we calculate the error and the control input.

By restricting the tracker design to a time-invariant system with a target model driven by a stationary noise and by implementing constant cost matrices, we can solve for the steady state tracker gain $\underline{G}_{c2}{}^*$. As in the regulator design, B_d is set to one. The beam state transition matrix is a scalar function decribed by Equation (3-16), where τ_B is the time constant.

The target state transition matrix, $\underline{\Phi}_T(t_{i+1},t_i)$, must be stable to find the steady state tracker gain, since the augmented system must be stabilizable. If the target state transition matrix has any poles on or outside the unit circle (z-plane analysis), then the augmented system model will have modes that are astable or unstable, and uncontrollable. Because the original target state transition matrix would be astable, we have to move the two poles at the origin of the s-plane left by some small ε. The modified $\underline{F}$ matrix from Equation (3-1) is

$$\underline{F}' = \begin{bmatrix} -\varepsilon & 1 & 0 \\ 0 & -\varepsilon & 1 \\ 0 & 0 & -1/\tau \end{bmatrix} \qquad (3\text{-}39)$$

The resulting state transition matrix is

$$\underline{\Phi}_T(t_{i+1},t_i) = \mathcal{L}^{-1}\{[\, s\underline{I} - \underline{F}'\,]^{-1}\} = \underline{\Phi}_T \qquad (3\text{-}40)$$

where $\mathcal{L}^{-1}\{\cdot\}$ represents the inverse Laplace transformation. Previous experience found that the value of $\varepsilon = .0001$ ensured at least an order of magnitude difference between the poles forced away from the origin and the exponential time constants of either the target's acceleration, τ_T, or the beam's position, τ_B.

As a result, the tracker gain equations can be reduced to

$$\underline{G}_{c2}^{*} = \underline{K}_{c12}\underline{\Phi}_{T} \ / \ (U + K_{c11}) \qquad (3\text{-}41)$$

$$\underline{K}_{c12} = [\ X_{11} \quad 0 \quad 0\ ] + \Phi_{B}^{T}\underline{K}_{c12}\underline{\Phi}_{T} - \underline{G}_{c1}^{*T}\underline{K}_{c12}\underline{\Phi}_{T} \qquad (3\text{-}42)$$

3.3 Proportional-Plus-Integral Controller

Because a proportional gain controller could not handle constant unmodeled disturbances in the system, Moose developed the type-one, proportional-plus-integral (PI) controller. The PI controller design includes many of the same assumptions used in the proportional gain controller. Both the PI regulator and tracker are designed on the basis of linear system models and quadratic cost functions, and they both implement steady state controller gains. Once again, assumed certainty equivalence is used and the Meer filter provides the particle beam state estimate while a Kalman filter provides the target state estimate.

3.3.1 *Proportional-Plus-Integral Regulator* The function of this regulator is to guide the beam state to some definable setpoint, y_d. This setpoint can be non-zero but, for this application, the setpoint will be defined as zero. To handle a definable setpoint and unmodeled step disturbances, the regulator augments a set of pseudointegral states to the original plant state equation. Because we are using a discrete system, a true integration process is not possible. Instead, a pseudointegral or summation process is used to provide a type-one control, able to drive the steady state mean of the regulation error, $[\underline{y}_c(t_i)-\underline{y}_r(t_i)-\underline{y}_d]$, to zero, even in the face of unmodelled constant disturbances affecting the plant. Note that for the regulator, the reference state, $\underline{y}_r(t_i)$ is equal to the zero vector. The pseudointegral is defined as

$$\begin{aligned} \underline{q}(t_i) &= \underline{q}(t_0) + \sum_{j=0}^{i-1} [\underline{y}_c(t_j) - \underline{y}_r(t_j) - \underline{y}_d] \\ &= \underline{q}(t_{i-1}) + [\underline{y}_c(t_i) - \underline{y}_r(t_i) - \underline{y}_d] \qquad (3\text{-}43) \end{aligned}$$

Figures 3-3 and 3-4 demonstrates how the pseudointegral is implemented in the PI controller.

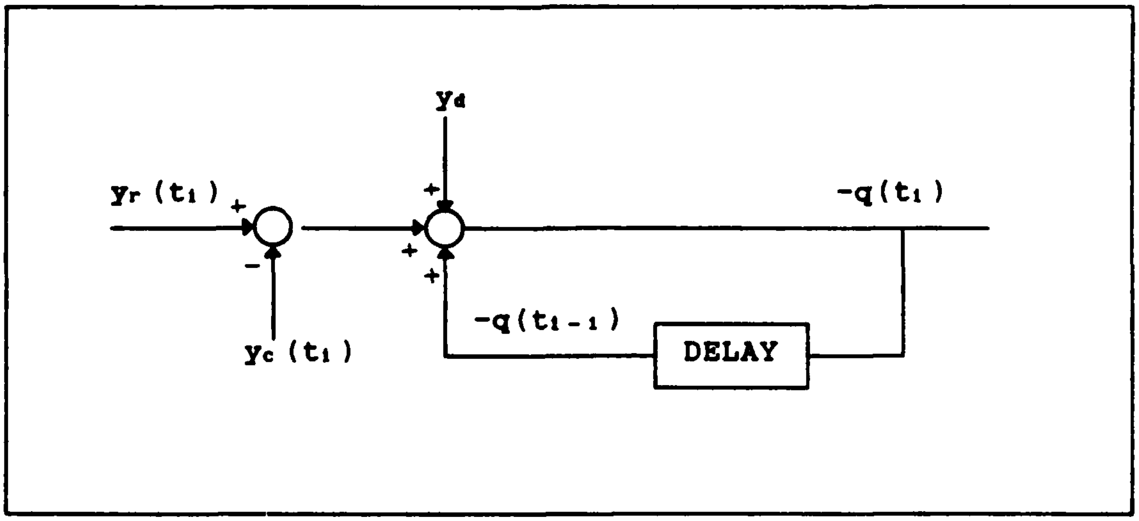


Figure 3-3. Pseudo-integral Term

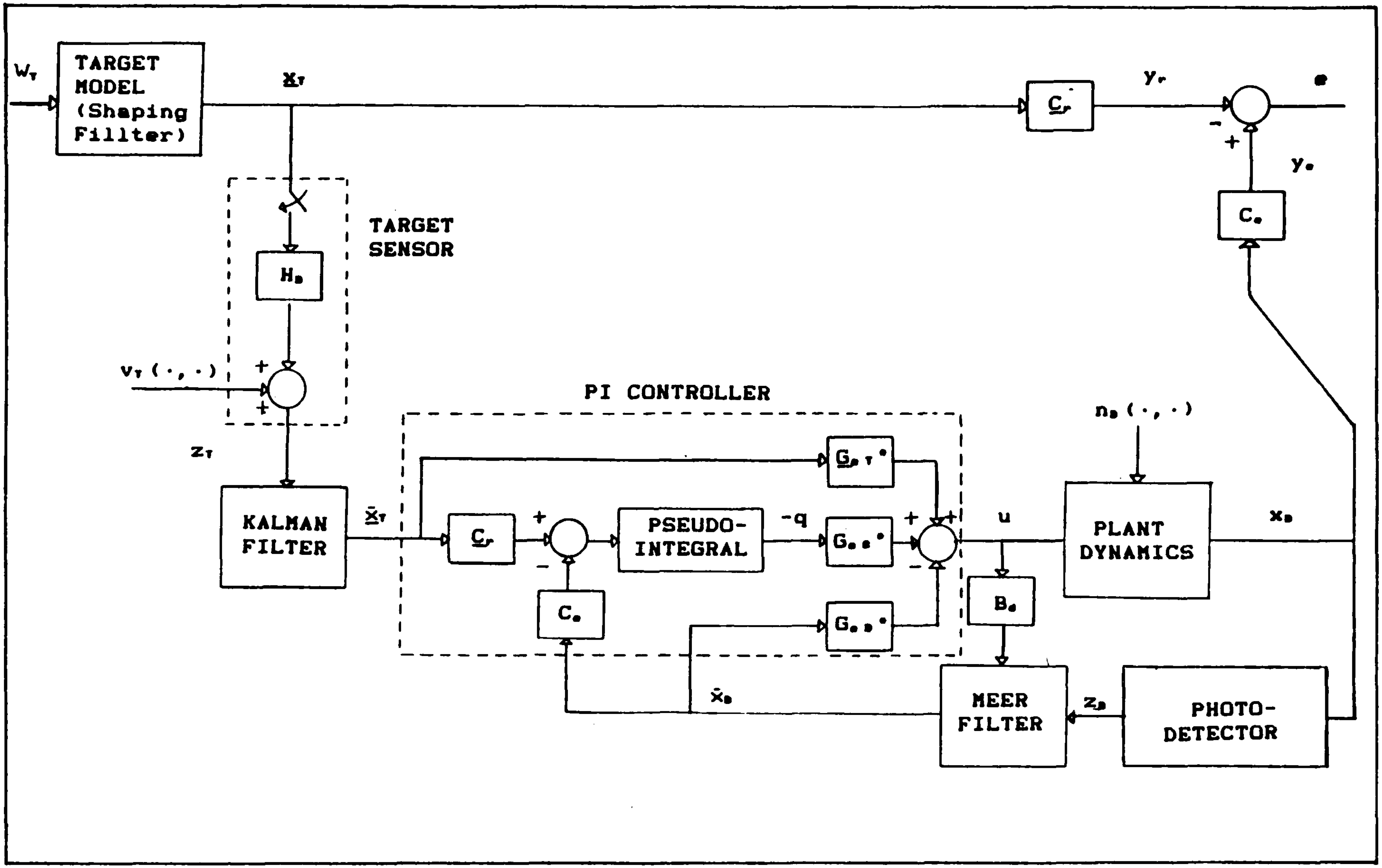


Figure 3-4. Proportional-Plus-Integral Controller

The PI control law is found by augmenting the pseudointegral (Equation (3-43)) with the beam state (Equation (3-8)) to form

$$\underline{x}_a(t_{i+1}) = \underline{\Phi}_a \underline{x}_a(t_i) + \underline{B}_{da}\underline{u}(t_i) + \underline{D}_a \underline{y}_d + \underline{G}_{da}\underline{w}_d(t_i)$$

$$\begin{bmatrix} \underline{x}_B(t_{i+1}) \\ \underline{q}(t_{i+1}) \end{bmatrix} = \begin{bmatrix} \underline{\Phi}_B & \underline{0} \\ \underline{C}_c & \underline{I} \end{bmatrix} \begin{bmatrix} \underline{x}_B(t_i) \\ \underline{q}(t_i) \end{bmatrix} + \begin{bmatrix} \underline{B}_d \\ \underline{0} \end{bmatrix} \underline{u}(t_i) - \begin{bmatrix} \underline{0} \\ \underline{I} \end{bmatrix} \underline{y}_d + \begin{bmatrix} \underline{G}_d \\ \underline{0} \end{bmatrix} \underline{w}_d(t_i)$$

(3-44)

The cost function retains the same structure as before, but includes a quadratic weighting on the pseudointegral in augumented form:

$$J = E\{ \sum_{i=0}^{N} \tfrac{1}{2}[\underline{x}_a^T(t_i)\underline{X}_a(t_i)\underline{x}_a(t_i) + \underline{u}^T(t_i)\underline{U}(t_i)\underline{u}(t_i)]$$

$$\tfrac{1}{2}[\underline{x}_a^T(t_{N+1})\underline{X}_{fa}(t_{N+1})\underline{x}_a(t_{N+1})\} \qquad (3\text{-}45)$$

The optimal control law

$$\underline{u}(t_i) = -\underline{G}_{ca}^*(t_i)\underline{x}_a(t_i) + E\underline{y}_d \qquad (3\text{-}46)$$

minimizes the cost function. The solution to the optimal gain functions are [14]

$$\underline{G}_{ca}^*(t_i) = [\ \underline{G}_{c1}^*(t_i) \quad \underline{G}_{c2}^*(t_i)\ ]$$

$$= [\underline{U} + \underline{B}_{da}^T \underline{K}_c(t_{i+1})\underline{B}_{da}]^{-1}[\underline{B}_{da}^T \underline{K}_c(t_{i+1})\underline{\Phi}_a(t_{i+1}, t_i)]$$

(3-47)

$$\underline{E} = [\, \underline{G}_{c1}{}^{*}(t_i) - \underline{G}_{c2}{}^{*}(t_i)\underline{K}_{c22}{}^{-1}(t_i)\underline{K}_{c12}{}^{T}(t_i)]\underline{\Pi}_{12} + \underline{\Pi}_{22} \tag{3-48}$$

where

$$\underline{\Pi} = \begin{bmatrix} \underline{\Pi}_{11} & \underline{\Pi}_{12} \\ \underline{\Pi}_{21} & \underline{\Pi}_{22} \end{bmatrix} = \begin{bmatrix} [\underline{\Phi}_{a}(t_{i+1},t_i) - \underline{I}] & \underline{B}_{d} \\ \underline{C}_{c} & \underline{0} \end{bmatrix}^{-1} \tag{3-49}$$

$\underline{K}_{c}(t_i)$ is derived from the backward Riccati difference equation

$$\begin{aligned} \underline{K}_{c}(t_i) &= \begin{bmatrix} \underline{K}_{c11}(t_i) & \underline{K}_{c12}(t_i) \\ \underline{K}_{c12}{}^{T}(t_i) & \underline{K}_{c22}(t_i) \end{bmatrix} \\ &= \underline{X}_{a} + \underline{\Phi}_{a}{}^{T}(t_{i+1},t_i)\underline{K}_{c}(t_{i+1})\underline{\Phi}_{a}(t_{i+1},t_i) \\ &\qquad - [\underline{B}_{da}{}^{T}\underline{K}_{c}(t_{i+1})\underline{\Phi}_{a}(t_{i+1},t_i)]^{T}\underline{G}_{c}{}^{*}(t_i) \end{aligned} \tag{3-50}$$

solved backwards from $\underline{K}_{c}(t_{N+1}) = \underline{X}_{fa}$.

If we assume a time-invariant system model and use constant weighting matrices, the regulator can be designed with steady state controller gains (provided the gain transients are short compared to the total time interval of interest, which is the case for this application). Because the tracker design will develop different controller gains, the specialization to the one-dimensional problem has been omitted.

3.3.2 Proportional-Plus-Integral Tracker For the PI tracker, we want to use the same PI controller structure as found in Figure 3-4 while implementing the tracking error equation used for the proportional gain tracker,

$$\begin{aligned} \underline{e}(t_i) &= \underline{C}_a(t_i)\underline{x}_a(t_i) \\ &= [\ \underline{C}_c(t_i)\ \ -\underline{C}_r(t_i)\ ]\ [\ \underline{x}_B^T(t_i)\ \ \underline{x}_T^T(t_i)]^T \end{aligned} \tag{3-51}$$

where $\underline{C}_a$ is an augmented matrix, and $\underline{x}_a$ is an augmented vector that include the beam state and the target state. Unless we wish to offset the beam from the target centroid (as to track some other appropriate point on the target or to lead the target), the set point, $\underline{y}_d$, will be defined as the zero vector. The remaining tracker development closely follows the PI design.

The PI control law is determined from the discretized, augmented system equation

$$\underline{x}_a(t_{i+1}) = \underline{\Phi}_a\underline{x}_a(t_i) + \underline{B}_{da}\underline{u}(t_i) + \underline{D}_a\underline{y}_d + \underline{G}_{da}\underline{w}_d(t_i)$$

$$\begin{bmatrix} \underline{x}_B(t_{i+1}) \\ \underline{x}_T(t_{i=1}) \\ \underline{q}(t_{i+1}) \end{bmatrix} = \begin{bmatrix} \underline{\Phi}_B & \underline{0} & \underline{0} \\ \underline{0} & \underline{\Phi}_T & \underline{0} \\ \underline{C}_c & -\underline{C}_r & \underline{I} \end{bmatrix} \begin{bmatrix} \underline{x}_B(t_i) \\ \underline{x}_T(t_i) \\ \underline{q}(t_i) \end{bmatrix} + \begin{bmatrix} \underline{B}_d \\ \underline{0} \\ \underline{0} \end{bmatrix} \underline{u}(t_i) - \begin{bmatrix} \underline{0} \\ \underline{0} \\ \underline{I} \end{bmatrix} \underline{y}_d + \begin{bmatrix} \underline{G}_d \\ \underline{0} \\ \underline{0} \end{bmatrix} \underline{w}_d(t_i) \tag{3-52}$$

where

$$\underline{x}_T(t_i) = \begin{bmatrix} x_{TP}(t_i) \\ x_{TV}(t_i) \\ x_{TA}(t_i) \end{bmatrix}$$

Equation (3-52) is composed of the beam state Equation (3-8), the discretized form of the tracker state equation (3-1) and the pseudointegral Equation (3-43) in which $\underline{y}_c(t_i)$ is expressed in terms of $\underline{x}_a(t_i)$. As mentioned before, the $\underline{F}$ matrix of Equation (3-1) is astable and the poles will be moved to the left of the s-plane origin.

Again, the cost function retains the same basic structure, but now it includes both the tracker and the pseudointegral cost terms. Because we are using the same PI control structure defined by Figure 3-4, the optimal control law, Equation (3-46), and the solution to the optimal gain functions, Equations (3-47) through (3-50), are still valid and will be used to design the PI tracker.

As was done with the the proportional gain tracker, the PI tracker will be restricted to a one-dimensional, time-invariant system using constant weighting matrices. The first restriction simplifies the problem substantially. The controlled variable, $\underline{y}_c$, the reference variable, $\underline{y}_r(t_i)$, and the setpoint, $\underline{y}_d$, are scalars defined as

$$y_c(t_i) = x_B(t_i) \tag{3-53}$$

$$y_r(t_i) = x_{TP}(t_i) \tag{3-54}$$

$$y_d = 0 \tag{3-55}$$

The latter two restrictions along with a decision to ignore short transients in Riccati solutions allow the problem to be solved using steady-state constant gains.

The PI tracker problem setup is as follows (see Equations (3-20) and (3-21):

$$\underline{C}_a = [\ \underline{C}_c \quad -\underline{C}_r\ ] = [\ 1 : -1 \quad 0 \quad 0\ ] \tag{3-56}$$

The state transition matrices of the beam and target are

$$\Phi_B = \exp[-(t_{i+1}-t_i)/\tau_B] = \exp[-\Delta t/\tau_B] \tag{3-57}$$

$$\underline{\Phi}_T = \mathcal{L}^{-1}\{[\ s\underline{I} - \underline{F}'\ ]^{-1}\} = \mathcal{L}^{-1}\left\{\begin{bmatrix} s+\varepsilon & -1 & 0 \\ 0 & s+\varepsilon & -1 \\ 0 & 0 & s+1/\tau_T \end{bmatrix}^{-1}\right\} \tag{3-58}$$

The augmented transition matrix is

$$\underline{\Phi}_a = \left[\begin{array}{c:ccc:c} \Phi_B & 0 & 0 & 0 & 0 \\ \hdashline 0 & & & & 0 \\ 0 & & \underline{\Phi}_T & & 0 \\ 0 & & & & 0 \\ \hdashline 1 & -1 & 0 & 0 & 1 \end{array}\right] \tag{3-59}$$

To be consistent with Jamerson's derivation [8], B_d is not assumed to equal one, but is derived from the continuous model:

$$B_d = \int_{t_i}^{t_{i+1}} \Phi_B(t_{i+1},\tau)B\,d\tau = B\tau_B(1-\Phi_B) \tag{3-60}$$

where B=1, and the discrete, augmented control input matrix, $\underline{B}_{da}$, is

$$\underline{B}_{da} = [\; B_d \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;]^T \tag{3-61}$$

Because the cost weighting function includes a cost weighting term on the pseudointegral state, Equation (3-35) must be redefined as

$$\underline{X}_a = \begin{bmatrix} X_{11} & -X_{11} & 0 & 0 & 0 \\ -X_{11} & X_{11} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & X_{55} \end{bmatrix} \tag{3-62}$$

The optimal control law that minimizes the cost function is

$$\begin{aligned} u(t_i) &= -\underline{G}_c^*(t_i)\underline{x}_a(t_i) + Ey_d \\ &= -[\underline{G}_{c1}^* \quad G_{c2}^*] \left[\begin{array}{c} x_B(t_i) \\ x_{TP}(t_i) \\ x_{TV}(t_i) \\ x_{TA}(t_i) \\ \hdashline q(t_i) \end{array} \right] + Ey_d \end{aligned} \tag{3-63}$$

where $\underline{G}_{c1}^* = [\; G_{c1B}^* \;\; G_{c1P}^* \;\; G_{c1V}^* \;\; G_{c1A}^* \;]$. The steady state controller gains are found by solving

$$\begin{aligned} \underline{G}_c^* &= [U + \underline{B}_{da}^T \underline{K}_{ca} \underline{B}_{da}]^{-1} [\underline{B}_{da}^T \underline{K}_{ca} \underline{\Phi}_a] \\ &= B_d [k_{11} \;\; k_{12} \;\; k_{13} \;\; k_{14} \;\; k_{15}] \underline{\Phi}_a \;/\; (U + B_d^2 k_{11}) \end{aligned} \tag{3-64}$$

$$E = [\underline{G}_{c1}^{*} - G_{c2}^{*}\underline{K}_{c12}^{T}/K_{c22}]\underline{\Pi}_{12} + \underline{\Pi}_{22} \qquad (3\text{-}65)$$

where

$$\underline{\Pi} = \begin{bmatrix} \underline{\Pi}_{11} & \underline{\Pi}_{12} \\ \underline{\Pi}_{21} & \underline{\Pi}_{22} \end{bmatrix} = \left[\begin{array}{cc:c} \Phi_s - 1 & \underline{0} & B_d \\ \underline{0} & \underline{\Phi}_T - \underline{I} & \underline{0} \\ \hdashline \underline{C}_c & -\underline{C}_r & 0 \end{array}\right]^{-1} \qquad (3\text{-}66)$$

$\underline{K}_{ca}$ is the steady state solution to the backward Riccati difference equation, and the solution is found by solving Equation (3-50) with the successive values of $\underline{K}_{ca}$ equal to one another:

$$\underline{K}_{ca} = \underline{X}_a + \underline{\Phi}_a^{T}\underline{K}_{ca}\underline{\Phi}_a - [\underline{B}_{da}^{T}\underline{K}_{ca}\underline{\Phi}_a]^{T}\underline{G}_c^{*} \qquad (3\text{-}67)$$

where the elements of $\underline{K}_{ca}$ are defined as

$$\underline{K}_{ca} = \begin{bmatrix} \underline{K}_{c11} & \underline{K}_{c12} \\ \underline{K}_{c12}^{T} & \underline{K}_{c22} \end{bmatrix} = \left[\begin{array}{cccc:c} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} \\ k_{21} & k_{22} & k_{23} & k_{24} & k_{25} \\ k_{31} & k_{32} & k_{33} & k_{34} & k_{35} \\ k_{41} & k_{42} & k_{43} & k_{44} & k_{45} \\ \hdashline k_{51} & k_{52} & k_{53} & k_{54} & k_{55} \end{array}\right] \qquad (3\text{-}68)$$

Table 3-1 contains the PI controller gains used for the sensitivity and robustness analyses (to be discussed in Chapter 4).

TABLE 3-1

Calculating the PI Controller Gains

1. Problem Setup

$\tau_B = 20$ $\tau_T = 10$ $\varepsilon = .0001$

$U = 1$ $X_{11} = 100$ $X_{55} = 10$

2. $\underline{B}_{da}$, $\underline{\Phi}_a$, $\underline{K}_{ca}$ and $\underline{\Pi}$ Matrices

$$\underline{B}_{da} = [\ .97541 \quad 0.0 \quad 0.0 \quad 0.0 \quad 0.0\ ]$$

$$\underline{\Phi}_a = \begin{bmatrix} .95123 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & .99990 & .99990 & .48371 & 0.0 \\ 0.0 & 0.0 & .99990 & .95158 & 0.0 \\ 0.0 & 0.0 & 0.0 & .90484 & 0.0 \\ 1.0 & -1.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

$$\underline{K}_{ca} = \begin{bmatrix} 138.57 & -138.63 & -1.0092 & .21089 & 37.367 \\ -138.63 & 151.13 & 62501 & 624373 & -37.370 \\ -1.0099 & 62501 & 6.25*10^{8} & 6.25*10^{9} & -.00963 \\ .20405 & 624373 & 6.25*10^{9} & 6.25*10^{10} & .69494 \\ 37.367 & -37.370 & -.00946 & .69654 & 47.099 \end{bmatrix}$$

$$\underline{\Pi} = \begin{bmatrix} 3.01*10^{-7} & -10001 & -1.00*10^{7} & -1.00*10^{8} & 1.0000 \\ 3.01*10^{-7} & -10001 & -1.00*10^{7} & -1.00*10^{8} & 1.8*10^{-8} \\ 0.0 & 0.0 & -10001 & -1.00*10^{5} & 0.0 \\ 0.0 & 0.0 & 0.0 & -10.508 & 0.0 \\ 1.0252 & -500.02 & -5.00*10^{6} & -5.00*10^{7} & .05000 \end{bmatrix}$$

3. PI Controller Gains

$$\underline{G}_c = [\ 1.2422 \quad -1.2921 \quad -1.0252 \quad -.49800 \quad .27436\ ]$$

$$E = 1.074560$$

Note:
ε is the distance that the two poles are moved left of the origin (see Equation 3-39)
$\underline{B}_{da}$ is calculated from Equations (3-60) and (3-61)
$\underline{\Phi}_a$ is calculated from Equations (3-57) through (3-59)
$\underline{K}_{ca}$ is calculated from Equation (3-67) - note the numerical precision that is required
$\underline{\Pi}$ is calculated from Equation (3-66)
$\underline{G}_c$ is calculated from Equation (3-64)
E is calculated from Equation (3-65)

3.4 Reviewing the Cost Weighting Matrices

This section will re-evaluate the quadratic cost function in both the proportional gain and the PI trackers in the hope of selecting a set of weighting matrices that will improve the robustness of the trackers. We will start with the more complex PI tracker cost function, and reduce the general quadratic cost equation to the present form expressed by Equation (3-45). After reviewing the relationship between the elements within the weighting matrices, we will draw the analogues applicable to the proportion gain tracker cost function.

The general form of the PI cost function is

$$J = \sum_{i=0}^{N} \left\{ \frac{1}{2} \begin{bmatrix} \underline{x}(t_i) \\ \underline{q}(t_i) \\ \underline{u}(t_i) \end{bmatrix}^T \begin{bmatrix} \underline{X}_{11} & \underline{X}_{12} & \underline{S}_1 \\ \underline{X}_{12}^T & \underline{X}_{22} & \underline{S}_2 \\ \underline{S}_1^T & \underline{S}_2^T & \underline{U} \end{bmatrix} \begin{bmatrix} \underline{x}(t_i) \\ \underline{q}(t_i) \\ \underline{u}(t_i) \end{bmatrix} \right\}$$

$$+ \begin{bmatrix} \underline{x}(t_{N+1}) \\ \underline{q}(t_{N+1}) \end{bmatrix}^T \begin{bmatrix} \underline{X}_{F11} & \underline{X}_{F12} \\ \underline{X}_{F12}^T & \underline{X}_{F22} \end{bmatrix} \begin{bmatrix} \underline{x}(t_{N+1}) \\ \underline{q}(t_{N+1}) \end{bmatrix} \qquad (3\text{-}69)$$

To generate a steady state control law, we can let N approach ∞. As a result, the $\underline{X}_F$ matrix will not effect the overall cost function, and can be dropped from any further consideration.

The $\underline{S}$ cross terms have two origins. First, the $\underline{S}$ term arises from the coupling between the control input, $\underline{u}(t_i)$, the pseudo-integral vector, $\underline{q}(t_i)$, and the dynamics state vector, $\underline{x}(t_i)$. This coupling arises from the linear relationship of the filter model, such as,

$$\dot{x}_k = -(1/\tau)x_k + u_k \tag{3-70}$$

and there might be a desire to put a quadratic cost on the scalar quantities such as $\dot{x}_k$ that are linear combinations of $\underline{u}$, $\underline{q}$ and $\underline{x}$. Second, the $\underline{S}$ term is frequently used to exert control over continuous-time dynamics for the entire sample period and not just at t_i [14]. The significance of this contribution to the $\underline{S}$ term depends on the sample rate. As the sample rate is increased, the $\underline{S}$ term's contribution approaches zero. For the controller designs in this application, it is assumed that the sample rate will be high enough to make the $\underline{S}$ terms insignificant. Without the $\underline{S}$ terms, Equation (3-69) assumes the form found in Equation (3-45).

What remains to be evaluated are the $\underline{X}_a$ and $\underline{U}$ cost weighting matrices required for the PI tracker steady state gain functions. The $\underline{X}_a$ matrix is defined as a positive semidefinite weighting matrix that assigns a cost to the tracker error between $x_B(t_i)$ and $x_{TP}(t_i)$. Therefore, there is no reason to weight the cross product terms resulting from the target velocity or acceleration, and their elements are set to zero as depicted by Equation (3-62). The X_{11} elements weight the present position error between the beam and target and X_{55} weights the pseudointegration value of those errors. Non-zero X_{55} allows the controller to have type 1 versus type 0 properties, which are desirable in tracking. If X_{55} is smaller than X_{11}, this tells the controller to place a greater emphasis on minimizing the integral of past errors. Therefore, the response to a large positive error history at t_i would be a large negative control input. Unfortunately, this can cause a very oscillatory response and a large initial overshoot that could drive the actuators into saturation [8]. If this were to occur, the oscillatory response can be dampened by increasing X_{55}.

The $\underline{U}$ matrix places a penalty on the control energy expenditure, and $\underline{U}$ must be positive definite. Because we are limiting the design to one dimension, $\underline{U}$ becomes a positive scalar. This leaves only three cost weighting elements, X_{11}, X_{55} and U, to be evaluated. If one of the elements is set to a constant value, then a robustness analysis can be conducted over the range of the remaining two elements. In other words, the selection of alternate weighting matrices that have the best robustness characteristics does not require us to evaluate the the three by three cost weighting matrix depicted by Equation (3-69), but instead requires us to confine our attention to a single relationship with only two unknown variables.

These findings can be extended to the proportional gain tracker. Since the proportional gain tracker does not have a pseudointegrator, it does not have a pseudointegral weighting element. Therefore, if we set U to a constant, then we only need to vary X_{11} to determine the best set of cost weighting matrices.

The method of selecting a set of weighting matrices is based on the simple sum of the quadratics on all the quantities of interest at each sample period. Therefore, any deviations from zero are penalized. If we had to design a system with specified performance limitations, we would pick the initial cost weighting elements as one over the square of the maximum allowable value [14]. From there, the controller's cost weighting matrix would be tuned according to the results of a performance analysis. Without prescribed design specifications, the cost weighting elements will be selected to minimize the RMS tracker error which will be calculated during the initial performance analysis (refer to Section 4.4).

There are several alternative methods of selecting cost weighting matrices. First, with a controller feedback sample period of one second, it might be desirable to include the cross terms in order to provide better control over the entire sample period. Second, the weighting matrices can be selected on the basis of their impact on the closed-loop system's pole placement. The effects this can have on stability and loop shaping can be observed by evaluating the effects different controller gains could have on the controller transfer function (see Equation (6-13)). Two formal techniques which exploit this are implicit model following [2,16,19] and the LQG/LTR Dual Method of Kwakernaak and Sivan [10]. Further discussions on these techniques has been included in Appendix A.

3.5 The Multiple Model Adaptive Controller

The previous two controller designs had difficulty tracking a target transitioning from a benign trajectory to an evasive flight path because of the dynamics noise strength parameter uncertainty induced by the increased maneuvering. That is, if a non-adaptive Kalman filter were tuned to a target flying straight and level, it will have great difficulty tracking the same target as the target tries to evade the tracker with a series of high-g jinking maneuvers. The reason is that the filter's parameters such as the target's acceleration time constant and the dynamics noise strength will not be representative of the new target's trajectory. One method to improve a system's performance is to construct a multiple model adaptive controller (MMAC). This section will start out by developing the general MMAC model and then tailor it to the tracker model that is used throughout the thesis. The uncertain parameters of initial interest are those associated with the target. Thus, the multiple model structures entail replications of the Kalman filter or of the entire controller.

3.5.1 Derivation of the Multiple Model Adaptive Controller

The MMAC is based on the premise that the robustness of a controller can be improved if the controller can provide on-line estimation of the uncertain parameters defined as $\underline{a}(t)$. The uncertain parameters can affect any or all of $\underline{\Phi}$, $\underline{B}_d$, $\underline{H}$, $\underline{Q}_d$ and $\underline{R}$, and thereby induce erroneous estimates of the state and the error covariance in a non-adaptive filter. As a result, we may not be able to design a single controller model exactly. Instead, we may find it possible to design an adaptive estimator/controller.

One means of generating an adaptive stochastic feedback controller is by designing a MMAE-based adaptive controller with adaptive controller gains, $\underline{G}_c^*[t_i,\underline{\hat{a}}(t_i^+)]$ (see Figure 3-5a). The concept is that the MMAE provides a state estimate, $\underline{\hat{x}}(t_i^+)$, and an uncertain parameter estimate, $\hat{a}(t_i^+)$, based on the past measurement history, $\underline{Z}_i$. The adaptive controller gain is calculated from the uncertain parameter estimate (gains are precomputed as a function of assumed parameter values, and then those functions are evaluated based on $\hat{a}(t_i^+)$), and the control input, $\underline{u}(t_i)$, is calculated from the product of the state estimate and the adaptive controller gain, as:

$$\underline{u}(t_i) = -\underline{G}_c^*[t_i,\underline{\hat{a}}(t_i)]\underline{\hat{x}}(t_i^+) \qquad (3\text{-}71)$$

If the uncertain parameters were confined to the dynamics driving noise and the measurement corruption noise statistics, then $\underline{G}_c^*(t_i)$ is not dependent on $\underline{\hat{a}}(t_i)$, and a non-adaptive controller gain, $\underline{G}_{c0}^*(t_i)$, is appropriate (see

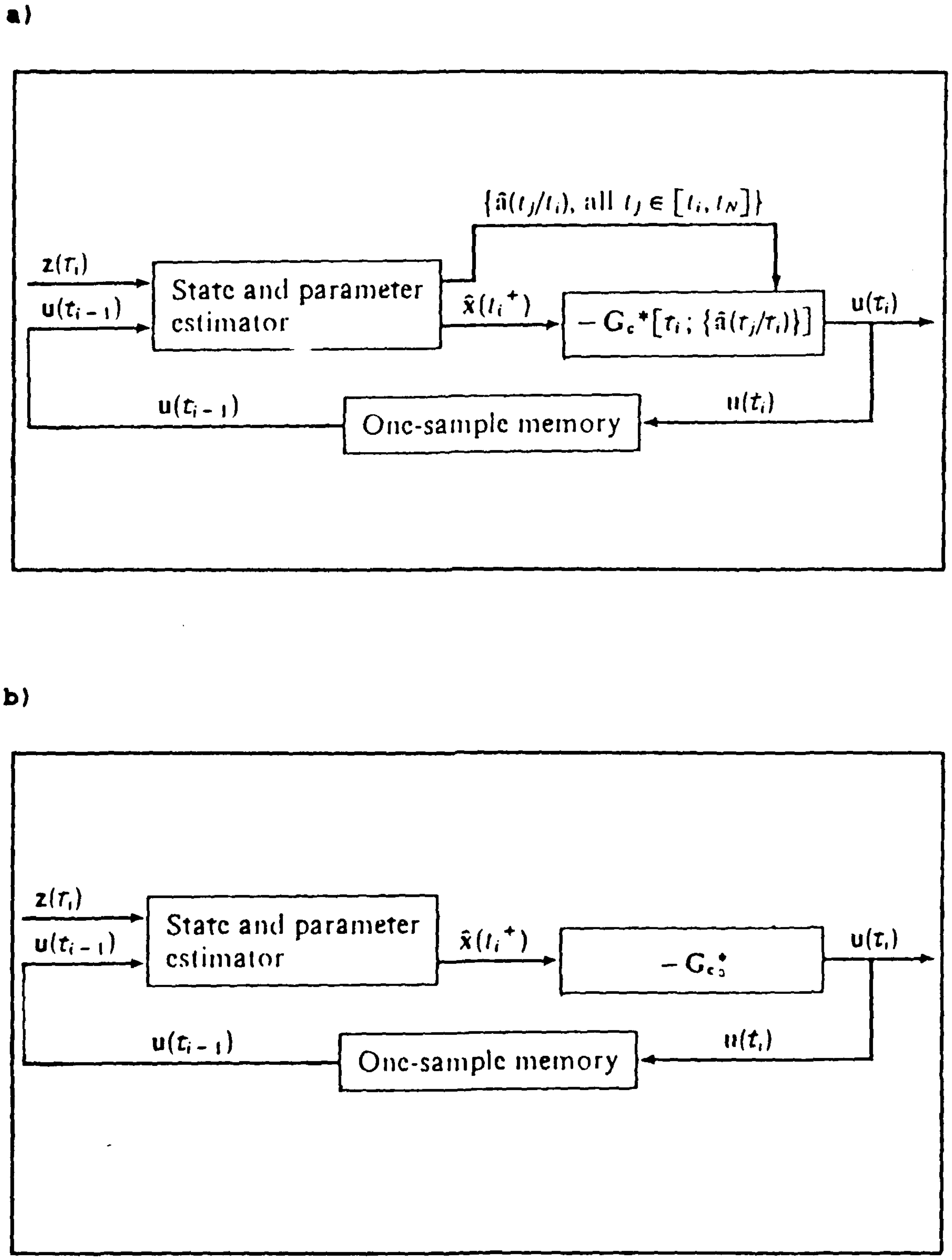


Figure 3-5. Adaptive Stochastic Feedback Controller: (a) With an adaptive feedback gain. (b) With non-adaptive gain.

Figure 3-5b). On might also choose to implement a nominal gain $G_{co}(t_i)$ rather than employ an adaptive gain, using adaptivity only to improve the accuracy of the state estimate. Last, one might assume steady-state conditions and use a steady-state controller gain, such as $\underline{G}_c^*(\hat{\underline{a}})$ or $\underline{G}_{co}^*$.

By extending the concept of an adaptive stochastic feedback controller, we can generate the multiple model adaptive controller (see Figure 3-6) [1,4,13,14]. This is done by replacing the bank of Kalman filters, used to estimate the state and uncertain parameters, with a bank of LQG controllers. Each controller is based on a particular set of parameter values, $\underline{a}_k(t_i)$, where k identifies the controller within the bank. Within each controller, a state estimate, $\hat{\underline{x}}_k(t_i^+)$, is computed by a Kalman filter based on the parameter value, $\underline{a}_k$, and is multiplied by the appropriate steady-state gain, $\underline{G}_c^*(\underline{a}_k)$, based on the same parameter assumption. The result is an optimal control law conditioned on $\underline{a}_k$ being the true parameter:

$$\underline{u}_k(t_i) = -\underline{G}_c^*(\underline{a}_k)\hat{\underline{x}}_k(t_i^+) \tag{3-72}$$

The adaptive control is generated by adding the probabilistically weighted $\underline{u}_k(t_i)$ values; as shown at the right most summing junction in Figure 3-6.

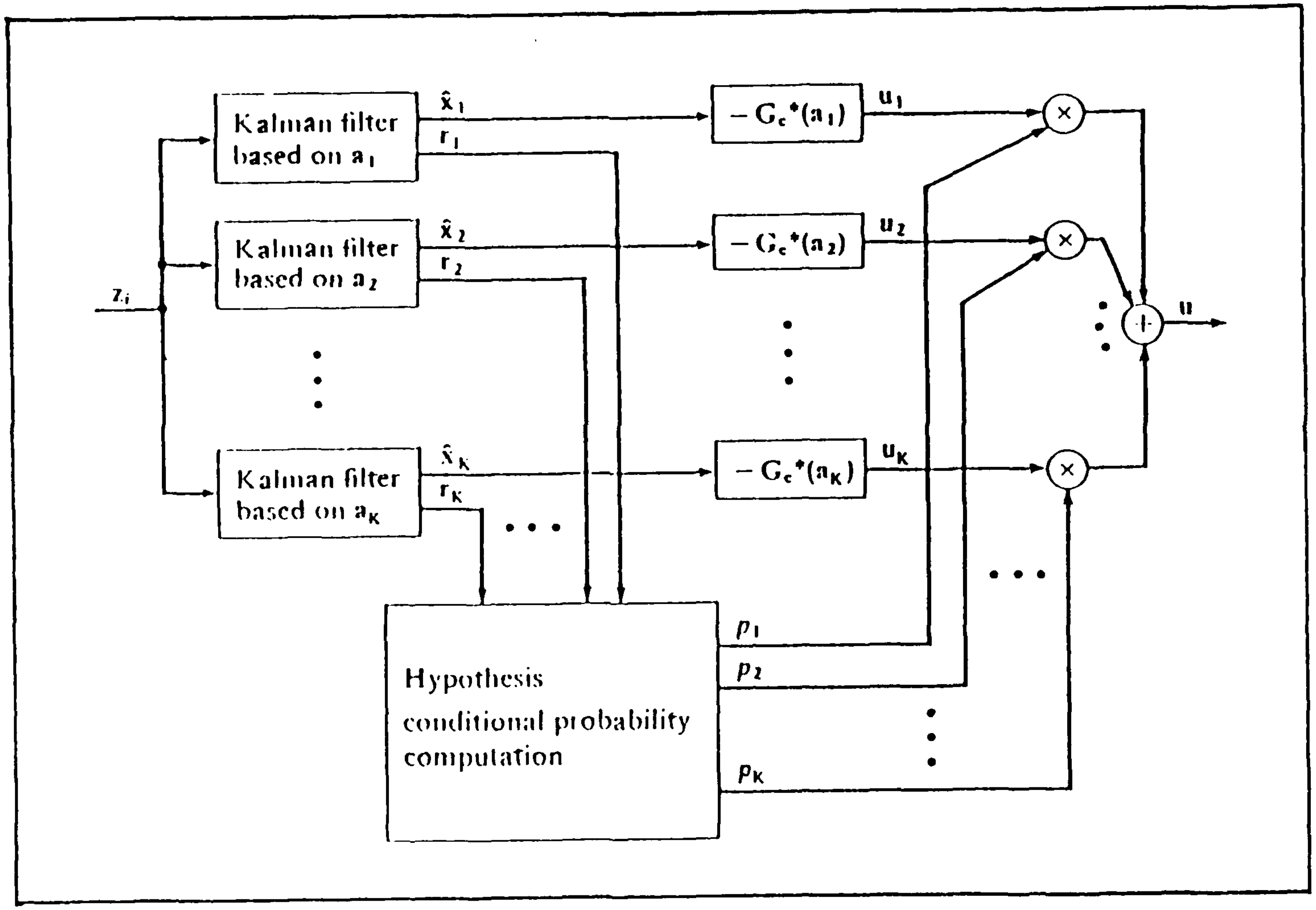

z_i
Kalman filter based on a_1
Kalman filter based on a_2
Kalman filter based on a_K
$\hat{x}_1$
r_1
$\hat{x}_2$
r_2
$\hat{x}_K$
r_K
$-G_c^*(a_1)$
$-G_c^*(a_2)$
$-G_c^*(a_K)$
u_1
u_2
u_K
u
Hypothesis conditional probability computation
p_1
p_2
p_K

Figure 3-6. Multiple Model Adaptive Controller

One way to develop the MMAC mathematically is first to develop the multiple model adaptive estimator using a Bayesian approach, and to draw the mathematical analogy between the adaptive feedback controller based on such an estimator and the MMAC. The Bayesian state estimate can be expressed as the expected value of the state conditioned on the measurement history $\underline{Z}_i$:

$$\hat{\underline{x}}(t_i^+) = E\{\underline{x}(t_i)|\underline{Z}_i\} = \int \underline{\xi}\cdot f_{\underline{x}(t)|\underline{z}}(\underline{\xi}|\underline{Z}_i)\,d\underline{\xi} \tag{3-73}$$

where $\underline{\xi}$ is a dummy variable of integration of $\underline{x}(t)$, and $f_{\underline{x}(t)|\underline{z}}(\underline{\xi}|\underline{Z}_i)$ is the conditional probability density function of $\underline{x}(t)$. If we let $\underline{a}(t)$ denote the vector of uncertain parameters, and assume $\underline{a}(t)$ can take on any value in the continuous range of A, Equation (3-73) can be rewritten to include $\underline{a}(t)$:

$$\hat{\underline{x}}(t_i^+) = \int \underline{\xi}\left[\int_A f_{\underline{x}(t),\underline{a}|\underline{z}}(\underline{\xi},\underline{\alpha}|\underline{Z}_i)\,d\underline{\alpha}\right]d\underline{\xi} \tag{3-74}$$

where $\underline{\alpha}$ is the dummy variable used to integrate $\underline{a}(t)$ over the range of A. According to Bayes' rule, the conditional probability density function can be expressed as

$$f_{\underline{x}(t),\underline{a}|\underline{z}}(\underline{\xi},\underline{\alpha}|\underline{Z}_i) = f_{\underline{x}(t)|\underline{a},\underline{z}}(\underline{\xi}|\underline{\alpha},\underline{Z}_i)\,f_{\underline{a}|\underline{z}}(\underline{\alpha}|\underline{Z}_i) \tag{3-75}$$

The first density function on the right hand side of Equation (3-75) is Gaussian, with a mean of $\underline{\hat{x}}(t_i^+)$ and a covariance of $\underline{P}(t_i^+)$ as computed by a Kalman filter based upon a particular $\underline{a}$. The second term is the conditional probability of $\underline{a}$ conditioned on the measurement history. According to Bayes' rule, it can be expressed as

$$
\begin{aligned}
f_{a|z}(\underline{\alpha}|\underline{Z}_i) &= f_{a|z(i),z(i-1)}(\underline{\alpha}|\underline{\zeta}_i, \underline{Z}_{i-1}) \\
&= \frac{f_{a,z(i)|z(i-1)}(\underline{\alpha},\underline{\zeta}_i|\underline{Z}_{i-1})}{f_{z(i)|z(i-1)}(\underline{\zeta}_i|\underline{Z}_{i-1})} \\
&= \frac{f_{z(i)|a,z(i-1)}(\underline{\zeta}_i|\underline{\alpha},\underline{Z}_{i-1})\, f_{a|z}(\underline{\alpha}|\underline{Z}_i)}{\int_A f_{z(i)|a,z(i-1)}(\underline{\zeta}_i|\underline{\alpha},\underline{Z}_{i-1})\, f_{a|z}(\underline{\alpha}|\underline{Z}_i)\, d\underline{\alpha}}
\end{aligned}
\tag{3-76}
$$

where $f_{z(i)|a,z(i-1)}(\underline{\zeta}_i|\underline{\alpha},\underline{Z}_{i-1})$ is Gaussian, with a mean of $\underline{H}(t_i)\underline{\hat{x}}(t_i^-)$ and a covariance of $[\underline{H}(t_i)\underline{P}(t_i^-)\underline{H}^T(t_i) + \underline{R}(t_i)]$. Equation (3-76) could conceptually be evaluated recursively starting from $f_a(\underline{\alpha})$. By combining Equations (3-74) with (3-75) and switching the order of integration, we can generate the state estimate from

$$
\underline{\hat{x}}(t_i^+) = \int_A \left[\int \underline{\xi}\, f_{x(t)|a,z}(\underline{\xi}|\underline{\alpha},\underline{Z}_i)\, d\underline{\xi} \right] f_{a|z}(\underline{\alpha}|\underline{Z}_i)\, d\underline{\alpha} \tag{3-77}
$$

Unfortunately, the double integration will make this approach computationally impractical, so we let the uncertain parameter vector assume only a finite number of values, such as the discrete vector set $\{\underline{a}_1, \underline{a}_2, \dots \underline{a}_K\}$, where K is the number of elemental filters in the MMAE, and eventually the number of controllers within the MMAC. Thus, the integration is replaced by a finite summation. This becomes the foundation of the multiple model structure, where each controller within the bank of controllers is based on an actual discrete parameter value.

To adopt the Bayesian approach to a multiple model structure, an a priori density function is required for $\underline{a}_k(t_0)$:

$$f_{a(t_0)}(\underline{\alpha}) = \sum_{k=1}^{K} p_k(t_0)\delta(\underline{\alpha} - \underline{a}_k) \tag{3-78}$$

where $p_k(t_0)$ is the probability $\underline{a}$ assumes the value $\underline{a}_k$ at time t_0. Therefore, the hypothesis conditional probability, $p_k(t_i)$, is a recursive relationship expressed as

$$\begin{aligned} p_k(t_i) &= \text{prob}\{\underline{a}=\underline{a}_k \mid \underline{Z}(t_i)=\underline{Z}_i\} \\ &= \frac{f_{z(i)|a,z(i-1)}(\underline{\zeta}_i \mid \underline{a}_k, \underline{Z}_{i-1})p_k(t_{i-1})}{\sum_{j=1}^{K} f_{z(i)|a,z(i-1)}(\underline{\zeta}_i \mid \underline{a}_j, \underline{Z}_{i-1})p_j(t_{i-1})} \end{aligned} \tag{3-79}$$

where $f_{z(i)|a,z(i-1)}(\underline{\zeta}_i \mid \underline{a}_k, \underline{Z}_{i-1})$ can be evaluated as

$$f_{z(i)|a,z(i-1)}(\underline{\zeta}_i \mid \underline{a}_k, \underline{Z}_{i-1}) = \frac{1}{(2\Pi)^{m/2}\,|\underline{A}_k(t_i)|^{1/2}} \cdot \exp\{\cdot\}$$

$$\{\cdot\} = \{-\tfrac{1}{2}\underline{r}_k^{T}(t_i)\underline{A}_k(t_i)^{-1}\underline{r}_k(t_i)\} \tag{3-80}$$

and

$\underline{r}_k(t_i)$ is the residual; $\underline{r}_k(t_i) = \underline{z}_i - \underline{H}_k(t_i)\hat{\underline{x}}_k(t_i^-)$

$\underline{A}_k(t_i) = \underline{H}_k(t_i)\underline{P}_k(t_i^-)\underline{H}_k(t_i) + \underline{R}_k(t_i)$

m is the number of measurements at time t_i; dimension of $\underline{r}_k(t_i)$

Therefore, the conditional mean is the sum of the probabilistically weighted state estimates from each controller within the bank:

$$\hat{\underline{x}}(t_i^+) = E\{\underline{x}(t_i)|\underline{Z}(t_i)=\underline{Z}_i\}$$

$$= \sum_{k=1}^{K} \hat{\underline{x}}_k(t_i^+)p_k(t_i) \qquad (3\text{-}81)$$

Note how equations (3-79) and (3-81) are closely related to equations (3-76) and (3-77) respectively.

The MMAC feedback control can be analogously derived, and is the probabilistically weighted sum of $\underline{u}_k(t_i)$ as expressed by

$$\underline{u}(t_i) = \sum_{k=1}^{K} \underline{u}_k(t_i)p_k(t_i)$$

$$= \sum_{k=1}^{K} -\underline{G}_c^*(\underline{a}_k)\hat{\underline{x}}_k(t_i^+)p_k(t_i) \qquad (3\text{-}82)$$

3.5.2 The Multiple Model Adaptive Tracker

Since this is the first implementation of a MMAC for this application and is to be viewed as a feasibility demonstration, the design will be simple. The MMAC will be based upon only one uncertain parameter, discretized into three possible values, and thus it will be composed of a bank of three controllers. The purpose of the MMAC is to allow the tracker to adapt to changes in the best modeled value of target dynamics driving noise strength, Q_T, because the tracker has no control over Q_T, and since mis-modeling of Q_T leads to large RMS errors. Each controller will be tuned to a different, quantized value of Q_{Tk}. For this application, each controller will be tuned to one of the following uncertain parameter values:

$$Q_{Ti} = 0.01 \qquad Q_{Ti} = 0.1 \qquad Q_{Ti} = 1.0$$

These three values approximately cover (for modeling purposes only) the range of trajectories from the straight and level flight, to the maximum manned-vehicle g-limit of 10g's, assuming a specific range to target or a defined relationship between real maneuvers and maneuvers as seen in the detector array plane.

Because Q_T is selected as the uncertain parameter and it is not used to calculate the steady-state controller gain, we can use the non-adaptive steady-state controller gain, $\underline{G}_{c0}^{*}$. If we had to calculate an adaptive controller gain, such as would be the case if the target acceleration time constant, τ_T, were the uncertain parameter, then each controller gain, $\underline{G}_c{}^*[a_k(t_i)]$, would be calculated from the same steady-state controller gain equations, Equations (3-57) through (3-67), but each controller gain function would be tuned to its specific uncertain parameter value.

Each controller within the MMAC structure operates independently of one another, and the embedded k-th elemental Kalman filter calculates the residual, $\underline{r}_k(t_i)$, the covariance of the measurement, $\underline{A}_k(t_k)$, and the best estimate of $\underline{x}_k(t_i)$. The residuals and covariances of the measurement from all of the elemental filters are used to calculate the probability, $p_k(t_i)$, which is used to weight each controller (see Equations (3-79) and (3-80)). Then, the probabilistically weighted control inputs, $u_k(t_i)p_k(t_i)$, are summed according to Equation (3-82). The controllers start out with equal probabilistic weighting. That is, each controller is initialized with an equal probability, $p_k(t_0) = 1/K$; the constant K represents the number of controllers within the MMAC structure (for this application, K = 3).

3.6 Summary

This chapter developed the target model and all the controllers to be used directly or in support of this thesis. The target was modeled through a first order Gauss-Markov acceleration process, where the state estimate (involving position, velocity and acceleration variables) and covariance error are provided by a Kalman filter. The first controller design developed the proportional gain regulator and tracker equations. But because the proportional gain controller does not have type-one characteristics that allow rejection of unknown, constant disturbances that arise from linearized models, the PI controllers were developed. Still, the performance of the tracker was sub-optimal. The next sections looked at improving the controllers' robustness by evaluating alternative weighting matrices and using the PI tracker in a multiple model adaptive controller. In all, the final objective is to design the optimal controller which can place the particle beam on a maneuvering target.

IV. THE ANALYSES

Before a design can be considered complete, it must be developed and its performance evaluated against some prespecified standard or baseline. This chapter discusses the different performance analyses used to evaluate the PI controller and also to develop and evaluate the adaptive PI controller. Both controller designs use the Meer filter to estimate the beam (controlled) state, and the Kalman filter to estimate the target (reference) state. The results of the performance analyses are listed in Chapter 5. The first section explains why Monte Carlo simulation was selected over covariance analysis. The next section addresses the software packages SOFE and SOFEPL which were used to generate the Monte Carlo simulation and error statistics. The following section provides an in-depth review of the parameters used in the Meer filter for beam location estimation and the Kalman filter for target state estimation. The last section discusses the actual analyses that are used.

4.1 The Method - Monte Carlo Simulation

The particle beam estimator/controller problem is modeled as a stochastic process and the performance can be best evaluated by observing the statistical behavior of the error processes. Specifically, we are interested in evaluating the controller's tracking error statistics, and it is assumed that the embedded estimators are already tuned for the appropriate dynamics driving noise strength, Q, and measurement corrupting noise covariance (spread dispersion of the beam in the Meer filter), R. Regardless, it will also be necessary to evaluate the Kalman and Meer filter's state estimation error statistics, to support some of the findings from the sensitivity and robustness analyses. The performance of a controller can often be analyzed by either covariance analysis or Monte Carlo simulation. The more efficient method is the covariance analysis, which requires only one software run to generate the time history of the estimation error covariance, $P_e(t_i)$, or other pertinent statistics. Unfortunately, covariance analysis cannot be applied to the multiple model adaptive controller because a proper covariance analysis is limited to linear stochastic controllers that use prespecified measurement update times. The adaptive mechanisms in both the controller and in the Meer filter violate the linearity, and the elemental Snyder-Fishman filters have a varying sample rate that is not prespecifiable.

Therefore, the less efficient Monte Carlo simulation is used to evaluate the particle beam estimator/controller. The Monte Carlo simulation is a complete computer simulation that requires numerous simulation runs to generate enough samples of the error process to approximate the filter's true statistics with sample statistics [12]. For this application, Zicker [17,27] found that the sample statistics sufficiently converged by 200 runs, with each run lasting 100 seconds. Zicker also found that the system transients dissipated by 50 seconds into the simulation. Therefore, these numbers have become the baseline for future Monte Carlo simulations and the statistics of the error processes are averaged over the last 50 seconds. Because the last 50 seconds should represent a steady-state condition, the time-average of the error process statistics, such as the average root-mean-square tracker error, can be used as a compact index of performance, and can be used to identify trends within any of the analyses.

The Meer or Kalman filter's state estimation error statistics, which are used to evaluate the filter's operation, are the mean and the standard deviation of the error between the truth and filter states. They will be calculated and plotted for each controller feedback sample period, t_i, between 50 and 100 seconds. For example, the derivation of the Meer filter's beam position state estimate error statistics are as follows: the error between the true beam position and the Meer filter estimate is

$$e_B(t_i, n) = x_{tB}(t_i, n) - \hat{x}_B(t_i, n) \tag{4-1}$$

where $50 \leq t_i \leq 100$ and n is the run number. Thus, the mean error is

$$\hat{e}_B(t_i) = [1/N] \cdot \sum_{n=1}^{N} e_B(t_i, n) \tag{4-2}$$

where N is the total number of runs (N=200). The variance and the standard deviation of the error are [5,7]

$$v_B(t_i) = [1/(N-1)]\cdot\sum_{n=1}^{N} \{e_B^2(t_i,n)\} - [N/(N-1)]\cdot\hat{e}_B^2(t_i) \qquad (4-3)$$

$$\sigma_B(t_i) = \{1 + 1/[4(N-1)]\}\cdot\{v_B(t_i)\}^{1/2} \qquad (4-4)$$

Equation (4-3) calculates the unbiased estimate of the variance and Equation (4-4) is an approximation that produces an unbiased estimate of the standard deviation. A full derivation of the exact equation and the limitations that apply to the approximation can be found in reference [6]. Because these statistics are calculated for the steady-state condition, it is desirable for $\hat{e}_B(t_i)$ to approximate zero (i.e., it is desirable for the bias to be negligibly small), and for $\sigma_B(t_i)$ to approach a steady state value from t_i=50 to 100 seconds. Deviations from these two conditions will indicate the filter is operating improperly. Note how the statistics are calculated for a finite sample population, and only when we have a large enough sample can we be sure the sample statistics closely approximate the true statistics.

The root-mean-square (RMS) tracking error, $RMS_e(t_i)$, and the time-averaged RMS tracker error, RMS_e, are the statistics used to evaluate the performance of the controller during the different analyses. The error for the tracker problem is defined as the difference between the true target state and the true beam position state:

$$e_t(t_i,n) = x_{tB}(t_i,n) - x_{tXP}(t_i,n) \tag{4-5}$$

The errors from each run are averaged and a mean error, $m_e(t_i)$, and a standard deviation of error, $\sigma_e(t_i)$, are calculated for each t_i:

$$m_e(t_i) = [1/N]\cdot\sum_{n=1}^{N}\{x_{tB}(t_i,n) - x_{tTP}(t_i,n)\} \tag{4-6}$$

$$\sigma_e(t_i) = \{1 + 1/[4(N-1)]\}$$
$$\cdot\{[1/(N-1)]\cdot\sum_{n=1}^{N}\{e_t^2(t_i,n)\} - [N/(N-1)]\cdot m_e^2(t_i)\}^{1/2} \tag{4-7}$$

The root-mean-square (RMS) was selected as the statistical performance parameter to be used to evaluate the controller. The RMS error can be generated for each sample period by using the equation:

$$RMS_e(t_i) = [m_e^2(t_i) + \sigma_e^2(t_i)]^{1/2} \tag{4-8}$$

The time-averaged RMS error is calculated by averaging each RMS error, $RMS_e(t_i)$, from 50 to 100 seconds:

$$RMS_e = [1/51]\cdot\sum_{i=50}^{100} RMS_e(t_i) \tag{4-9}$$

Once again, because the RMS error statistic is calculated for steady-state condition, $RMS_e(t_i)$ should appear as a constant value with only minor deviations. A controller with an increasing, ramp-like RMS error would indicate that the filter is unstable. Anytime an undesirable RMS error is found, the RMS error will be analyzed to see if the problem is primarily due to bias-like characteristic (i.e., an increasing mean) or fluctuations (i.e., an increasing σ_e).

4.2 The Tools - SOFE and SOFEPL

The performance analyses are generated from tailored software packages called SOFE and SOFEPL [7,21]. SOFE, which is short for Simulation for Optimal Filter Evaluation, is a general purpose Monte Carlo simulation program designed for evaluating systems that use Kalman filters. It contains both a truth model and a filter to be evaluated. The truth model is described by a set of stochastic differential equations that emulate the "real-world" system dynamics. The filter is also described by a set of differential equations representing the system's state propagation within a Kalman filter, and updates due to measurements. SOFE allows the user to specify the measurement format of the filter update, and if desired, to include feedback control. SOFEPL, which is short for SOFE Plotter, is a post-processing routine that can be programmed to perform various statistical functions, such as calculate the RMS error time history, and then plot the results. Both SOFE and SOFEPL were developed by Stanton H. Musick and others of the Air Force Avionics Laboratory.

SOFE had to be modified internally before it could be used, because the original version of SOFE was limited to Kalman or extended Kalman filter applications. In doing so, Meer [18] changed SOFE's code to accommodate the Snyder-Fishman estimators and the MMAE structure for indicating the likelihood of each photoelectric event being due to signal versus noise. The problem of adapting SOFE to this application did not affect the propagation or update equations, but lay in generating Poisson distributed random signal and noise events, and then using these events as measurement update for the Meer filter. The standard Kalman filter, such as found in the original version of SOFE, updates the filter at regular discrete time intervals, whereas the Meer filter updates only when it receives a signal event [27].

Because SOFE was intended for general applications, it allows the user to tailor SOFE to the user's needs through nine user-defined (i.e. user-written) subroutines. These subroutines allow the user to specify: the truth model's differential equations, the filter model's differential equation, truth model disturbances (optional), the projection matrix of the state vector onto the measurement vector ($\underline{H}$), the measurement corruption noise covariance matrix ($\underline{R}$), the system dynamics matrix (often composed of partial derivatives) ($\underline{F}$), dynamics driving noise strength matrix, ($\underline{Q}$), the initial filter covariance matrix ($\underline{P}_0$), the initial conditions for each run, a prespecified trajectory (optional), impulsive control (optional), and other calculations required during the measurement update such as feedback control. Meer's [18] alterations to the code incorporated several more user-defined subroutines that were used to implement the Meer filter. Zicker [27] followed Meer and added the "Merge" method of filter pruning and the first controller design. Moose and Jamerson [8,20] designed the follow-on controllers and Jamerson implemented the Gauss-Markov acceleration model for the target. Zicker, Moose and Jamerson's changes are limited to the user-defined subroutines and do not affect the executive routine of SOFE. The SOFE source code that Meer altered and that is used throughout the research is the FORTRAN 4, May 1982 version of SOFE.

The conceptual operation of the modified SOFE program is shown at a macro level in Figure 4-1. The fixed-sample-period controller requires that the discrete feedback control and the Meer filter propagation cycle operate at the same discrete sample rate, because the feedback law requires the most current particle beam state estimate. As mentioned before, the Meer filter's measurement update occurs whenever an event is observed. For simplicity, the time to the next signal and noise events are pre-calculated from the signal and noise rate parameters, and the integrator is told in advance where to stop for the Meer filter measurement

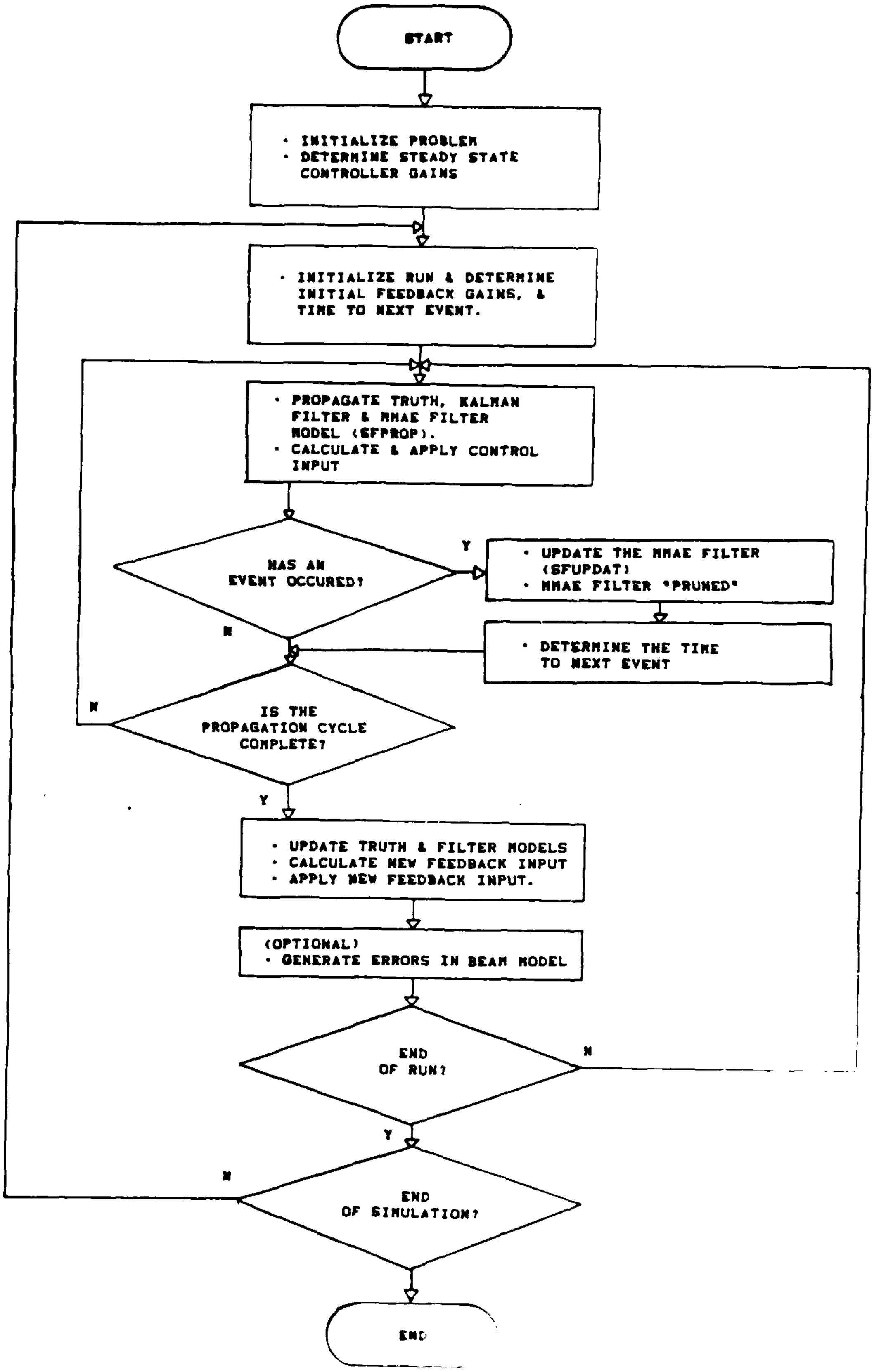
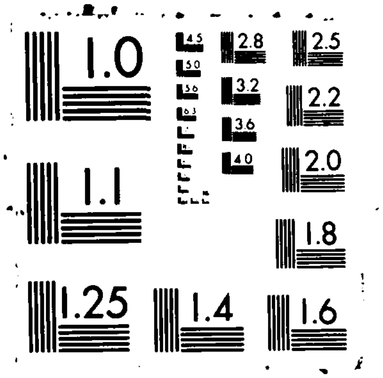

START
• INITIALIZE PROBLEM
• DETERMINE STEADY STATE CONTROLLER GAINS
• INITIALIZE RUN & DETERMINE INITIAL FEEDBACK GAINS, & TIME TO NEXT EVENT.
• PROPAGATE TRUTH, KALMAN FILTER & MMAE FILTER MODEL (SFPROP).
• CALCULATE & APPLY CONTROL INPUT
HAS AN EVENT OCCURED?
Y
• UPDATE THE MMAE FILTER (SFUPDAT)
• MMAE FILTER "PRUNED"
N
• DETERMINE THE TIME TO NEXT EVENT
N
IS THE PROPAGATION CYCLE COMPLETE?
Y
• UPDATE TRUTH & FILTER MODELS
• CALCULATE NEW FEEDBACK INPUT
• APPLY NEW FEEDBACK INPUT.
(OPTIONAL)
• GENERATE ERRORS IN BEAM MODEL
END OF RUN?
N
Y
N
END OF SIMULATION?
END

Figure 4-1. Macro Level Flow Chart [illegible]

AD-A182 610 STOCHASTIC ADAPTIVE PARTICLE BEAM TRACKER USING MEER 2/3
FILTER FEEDBACK(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING
UNCLASSIFIED B A JOHNSON DEC 86 AFIT/GE/ENG/86D-27 F/G 14/2 NL


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

update. A diagram of the top-down structure of the modified SOFE is presented in Appendix B.

The propagation of the truth states, the filter states and the filter-computed error covariances is accomplished by integrating the differential equations from the current time forward to some specified time. This time can be the integration step size, the time for the next Meer filter update or the time for the next Kalman filter update. The state differential equations have the general form

$$\dot{\underline{x}}(t) = \underline{f}[\underline{x}(t),\underline{u}(t),t] + \underline{G}(t)\underline{w}(t) \tag{4-10}$$

where $\underline{w}(t)$ is a zero-mean white Gaussian noise of strength $\underline{Q}$. SOFE solves Equation (4-10) by first using a fifth order Kutta-Merson integrator to solve the deterministic differential equation

$$\dot{\underline{x}}(t) = \underline{f}[\underline{x}(t),\underline{u}(t),t] \tag{4-11}$$

The stochastic term is added upon the completion of each integration through the expression

$$\underline{x}_S = \underline{x}_D + \underline{\mathrm{GAUSS}}(\underline{0},\underline{Q}_d) \tag{4-12}$$

where $\underline{x}_D$ is the deterministic solution to Equation (4-11), $\underline{x}_S$ is the stochastic solution, and the expression $\underline{\mathrm{GAUSS}}(\underline{0},\underline{Q}_d)$ is a randomly generated vector term of zero mean and a covariance of $\underline{Q}_d$. The dynamics driving noise strength, $\underline{Q}_d$ (the second moment of the equivalent discrete time noise representing $\underline{G}(t)\underline{w}(t)$), is calculated from

$$\underline{Q}_d = \int_{t_i}^{t_{i+1}} \underline{\Phi}(t_{i+1},\tau)\underline{G}(\tau)\underline{Q}(\tau)\underline{G}^T(\tau)\underline{\Phi}^T(t_{i+1},\tau)d\tau \qquad (4\text{-}13)$$

For both the Kalman and Meer filters, Q_d is scalar, and the integral in Equation (4-13) is solved as

$$Q_d = G^2 Q\tau[1-\exp(-2\Delta t/\tau)]/2 \qquad (4\text{-}14)$$

where τ is the target or beam time constant, $G = 1$, and Δt is the sample period over which the the noise is injected into the Equation (4-12).

The measurement noise is incorporated into the measurement update in a similar manner. The general measurement equation for either the Kalman filter or for the elemental Snyder-Fishman filter (which receives a hypothetical signal-induced event) is

$$\underline{z}(t_i) = \underline{h}[\underline{x}_t(t_i),t_i] + \underline{v}(t_i) \qquad (4\text{-}15)$$

where $\underline{z}(t_i)$ is the sum of the measurement function of the true states, $\underline{h}[\underline{x}_t(t_i),t_i]$, and the measurement corruption noise, $\underline{v}(t_i)$. SOFE injects the measurement noise by the following relationship:

$$\underline{z}(t_i) = \underline{\mathrm{GAUSS}}(\underline{h}[\underline{x}_t(t_i), t_i], \underline{R}] \qquad (4\text{-}16)$$

where $\underline{R}$ is the covariance of the target sensor measurement noise $\underline{v}(t_i)$, or the beam dispersion for the beam signal-induced photodetector event. If the event is signal-induced, Equation (4-16) gives the actual location of the event, not a "noise-corrupted" measurement of it. If the event in noise-induced, Equation (4-16) does not apply and the location of the noise-induced event is simulated by using a uniformly distributed mapping function.

SOFE updates the Kalman filter at regular time intervals since the Kalman filter has a fixed sample period. This is not true of the Meer filter because the Meer filter update is based on the arrival of the next signal or noise event. Thus, Meer [18] had to modify the SOFE source code so that SOFE would simulate a varying sample period (i.e., the time to the next signal- or noise-induced event) as a Poisson time process. The equation which will calculate the the varying sample period is derived from the Poisson process density function,

$$p(y) = -\{[\lambda(t)t]^y / y!\} \cdot \exp[-\lambda(t)t] \qquad (4\text{-}17)$$

where p(y) is the probability that y events occurred within time period t, and $\lambda(t)$ is the mean signal arrival rate. Because we are interested in generating a random sample period with the next event occurring at the completion of the sample period, y is set to zero (i.e., no events occur during the sample period), and the random sample period, t, is calculated from the inverse mapping function of Equation (4-17) [27]

$$t = -\hat{\lambda}(t) \cdot \ln[p(0)] \tag{4-18}$$

where p(0) is defined as the probability that t amount of time has passed before an event arrives. This probability has a range of $0 \leq p(0) \leq 1$. Because we are interested in generating samples of t_i, p(0) is selected randomly from the range of p(0), and this process is used to simulate the arrival time of the next signal- or noise-induced event.

The mean signal arrival rate is defined as

$$\hat{\lambda}_s(t) = n/(t_F - t_o) \tag{4-19}$$

where n is the number of signal-induced events expected over the duration of the simulation run, $(t_F - t_o)$. The relationship between the mean signal arrival rate and the expected signal rate parameter, $\hat{\underline{\lambda}}_s(t, \underline{r}, \underline{R}(t))$, can be shown by integrating Equation (2-1),

$$\begin{aligned} \hat{\lambda}_s(t) &= \int_{-\infty}^{\infty} \hat{\lambda}_s(t_i, \underline{\alpha}) d\underline{\alpha} \\ &= \int_{-\infty}^{\infty} \Lambda(t_i) \exp[-\underline{\alpha}^T \underline{R}^{-1}(t_i) \underline{\alpha}/2] d\underline{\alpha} \end{aligned} \tag{4-20}$$

where $\underline{\alpha}=[\underline{r}-\underline{H}(t)\underline{\hat{x}}(t)]$. Although the integral of the Gaussian density function does not have a closed-form expression, we know that the entire area under a probability density function must have a magnitude of one,

$$\int_{-\infty}^{\infty} [(2\Pi)^{m}|\underline{R}|]^{-1/2}\cdot\exp(-\underline{\alpha}^{T}\underline{R}^{-1}\underline{\alpha}/2)\,d\underline{\alpha} = 1 \qquad (4\text{-}21)$$

Thus, the maximum amplitude of the rate function, $\Lambda(t)$, can be shown as:

$$\Lambda(t) = [n/(t_F-t_0)]\cdot[(2\Pi)^{m}|\underline{R}(t)|]^{-1/2} \qquad (4\text{-}22)$$

where m is the dimensionality of the vector $\underline{\alpha}$.

The mean noise arrival rate is defined as

$$\hat{\lambda}_N(t) = \int_0^L \underline{\hat{\lambda}}_N(t,\underline{r})\,d\underline{r} = \underline{\hat{\lambda}}_N(t_1,r)\cdot\underline{L} \qquad (4\text{-}23)$$

where $\hat{\lambda}_N(t_1,\underline{r})$ is not a function of $\underline{r}$ in this application ($\hat{\lambda}_N(t,\underline{r})$ is defined as being uniformly distributed over the entire length of the detector).

The data provided by SOFE is statistically reduced and the results plotted by a second program called SOFEPL. The statistics used to evaluate the performance of either the Kalman or Meer filters are the mean and standard deviation of the error between the truth and filter state (see

Equations (4-2) and (4-4)). Although the SOFEPL program provides an array of statistical options, they are limited to the error statistics found between the truth and filter states (for filter performance evaluation) and are inadequate for evaluating a controller design. An alternative provided by SOFEPL is for the user to specify the desired error to be evaluated statistically. The procedure is used to generate the RMS time history as described by Equation (4-8), where the statistics are based on the error between the true beam and true target positions.

4.3 The Beam and Target Parameters

This section reviews the beam filter and target filter parameters since they can play such an important role in the design and performance of the MMAC. The Meer filter is designed around six parameters. They are as follows:

τ_B is the beam time constant.

g is the square root of the beam propagation noise strength, $g = (G^2Q)^{1/2}$ (Q=1 by assumption).

R is the beam dispersion measured as the variance of the Gaussian-shaped beam at the detector surface.

SNR is the signal-to-noise ratio (to be defined as in Equation (4-24)).

D is the MMAE filter depth.

n is the expected number of signal-induced events during a simulation run.

The three parameters associated with the Gauss-Markov acceleration target model are as follows:

τ_T is the target time constant.

Q_T is the target dynamics driving noise strength.

R_T is the target measurement noise variance.

The signal-to-noise ratio is defined for this application as the average number of signal induced events produced for every noise event. The SNR can be expressed as [18]:

$$\begin{aligned} \mathrm{SNR} &= [n/(t_F - t_0)]/[\lambda_N(t,\underline{r})\cdot L] \\ &= \Lambda(2\Pi R)^{1/2}/[\lambda_N(t,\underline{r})\cdot L] \end{aligned} \tag{4-24}$$

where n is the number of signal-induced events expected during one simulation run, $(t_f - t_0)$ is the duration of the simulation, $\lambda_N(t,\underline{r})$ is the noise arrival rate per length of the detector array, and L is the length of the detector array. For this one-dimensional application, the length of the detector is 10 cm.

The nominal values for the beam and tracker filter parameters used for the analyses are as follows:

The Beam Parameters

τ_B = 20 (sec)   D = 3

g = 0.2 $(\mathrm{cm}^2/\mathrm{sec})^{1/2}$ R = 0.5 (cm^2)

SNR = 20   n = 100 (signal events)

The Target Parameters

τ_T = 10 (sec) Q_T = 0.1 $(\mathrm{cm}^2/\mathrm{sec}^3)$ R_T = 0.5 (cm^2)

When the the Meer filter is simplified and the assumptions from Section 2.5 apply, the MMAE filter depth is reduced to one.

Because the fastest transients at the nominal condition have a time constant of ten seconds (rather benign dynamics), the Kalman filter and the feedback control sample period will be set to one second. This easily satisfies the Shannon's sampling theorem which states that the sampling rate should be at least twice the highest signal frequency content of interest. To provide an extra margin of insurance, the engineering guide of sampling ten times faster than the highest frequency is used.

4.4 Performance Analyses

Five different performance analyses are required to design and evaluate the MMAC, which is composed of a bank of PI controllers, of which each elemental controller uses its own Kalman filter to estimate the target state and a Meer filter to estimate the beam state. The first two analyses are the alternate controller cost weighting matrices analysis and the reduced Meer filter depth analysis. The results of these analyses will allow us to select the best set of cost weighting matrices for good on-line design performance and adequate robustness, and to allow us to justify using a filter depth of one (i.e. $D = 1$). The next two analyses are the sensitivity analysis and the robustness analysis. The results from these analyses are used to develop the MMAC. Last, another analysis is required to evaluate the operation of the MMAC. The first four analyses are performed on the PI tracker developed in Section 3.3.2. The MMAC analysis is performed on the adaptive controller developed in Section 3.5.2.

4.4.1 Evaluating Alternate Weighting Matrices. Although Jamerson did some work on selecting a suitable set of controller cost weighting matrices to achieve good controller performance and robustness, an exhaustive study was never accomplished. Therefore, a performance analysis is conducted to determine the best alternate cost weighting matrices which provide good performance at design conditions and that also enhance the MMAC's robustness. In other words, the goal is to select the best set(s) of weighting matrices that best minimize the averaged RMS error (without using excessive amounts of control) when the truth model matches the filter model, while simultaneously providing at least stability at off-design conditions.

4.4.2 Evaluating the Simplified Meer Filter. The second analysis evaluates the performance of the controller and the Meer filter when the filter depth is reduced from $D = 3$ to $D = 1$. This will reduce the number of propagating elemental Snyder-Fishman filters from eight to two, and in fact the filter can be expressed equivalently with only a single elemental filter, but with its gain expressed as a function of residual size. Although this will slightly increase the average RMS error, it should significantly decrease the computational loading (see

Section 2.5). It makes on-line feasibility much more reasonable.

4.4.3 Sensitivity Analysis. The purpose of the sensitivity analysis is to evaluate the performance of the PI controller as it is exposed to several different "real-world" environments. Because each environment could be described by a set of parameters, the sensitivity analysis is conducted by varying a single parameter in both the truth model and in the controller. The results lend insight on how the controller reacts if it knows the true environment and the results provide an absolute baseline of the best possible performance that could be expected from an adaptive controller. The test is based on the nominal parameter settings presented in the previous sections and most of the parameters are evaluated at one order of magnitude above and below their nominal values. The test evaluates all the beam and target parameters except the filter depth, which is set to one (D = 1). The cost weighting matrices are based on the results in Chapter 5 and are set to U=1, X_{11}=100 and X_{55}=10 (refer to Equation (3-62)).

4.4.4 Robustness Analysis. To complement the sensitivity analysis, a robustness analysis is performed to evaluate the controller's performance when an algorithm-assumed parameter differs from the "real-world" environment. The robustness analysis is conducted under the same guidelines as the sensitivity analysis, but differs in that the parameters are changed in the truth model without informing the controller's filters. The purpose of this study and the sensitivity analysis is to identify those parameters that are best suited for on-line adaptation by the MMAC, and to identify any other potential problems with the PI controller.

4.4.5 Evaluating the MMAC. Once the MMAC is developed, it is evaluated against a performance baseline. This baseline is defined as a PI controller which receives noise-corrupted measurements, $\underline{z}(t_i)$, but has access to the truth model's parameters. This baseline provides the best possible performance the MMAC could achieve if it could perfectly estimate the uncertain parameters. A well designed and tuned MMAC should closely approximate this baseline. Also, the MMAC is compared to an unknowledgeable and non-adaptive controller to see how much better performance the on-line parameter adaptation yields.

4.5 Summary

The purpose of this chapter has been to explain the tools and methods used to develop and evaluate the MMAC, and to evaluate the performance of the Kalman and Meer filters within the MMAC structure. The Monte Carlo simulation provides the most viable method of evaluating the controller's performance. SOFE and SOFEPL provide the basic method of computing the filter state estimation error statistics and controller tracking error statistics which allow us to evaluate the filters and the controller designs. Four of the performance analyses, the alternate weighting matrices, the reduced Meer filter, the sensitivity, and the robustness analyses, provide the foundation to develop the MMAC. Upon completion, the MMAC is developed and evaluated against a baseline controller, which assumes that the baseline controller has access to the true parameters while receiving noise corrupted measurements. The MMAC is also compared to the results of a non-adaptive PI controller. The results of the five analyses are in the next chapter.

V. RESULTS OF THE MONTE CARLO ANALYSES

The results of the five Monte Carlo analyses discussed in Section 4.4 are presented in this chapter. The first analysis evaluates seven different sets of weighting matrices and found the set, U=1, X_{11}=100, X_{55}=10, produces the lowest steady state RMS tracking error. None of the seven sets of weighting matrices tested cause the controller to use excessive amounts of control. The second analysis shows that the controller can be reduced to a filter depth of one without any measurable increase in RMS error between the beam and target positions. Therefore, the rest of the research is conducted with the filter depth set to one (D=1). Unfortunately, an appreciable reduction in computer time is not seen. This is credited to the overall inefficiency of the source code, and it is assumed that a program designed to take advantage of a reduced filter depth would require less computer processing time. The sensitivity analysis provides a baseline of the best possible controller performance provided the filter knows the "real-world" environment, while the robustness analysis measures the ability of the controller to cope with a "real-world" environment that differs from the parameter values assumed in the filter/controller design. The performance achieved when the values of most "real world" parameters are allowed to vary shows good robustness characteristics. The two notable exceptions are the target dynamics driving noise, Q_T, which the MMAC is designed to handle, and the particle beam time constant, τ_B, which appears to be the source of a controller stability problem. Last, the MMAC provides good on-line adaptive estimation of Q_T. The MMAC significantly outperforms the non-adaptive controller, but the controller responds slowly to a decreasing target dynamics noise strength.

The results of the first four analyses are from existing SOFE software used by Meer, Zicker, Moose and Jamerson [8,18,20,27], which was modified for easier usage, and corrected for three software errors. These errors were detected either by validation testing or by reviewing the source code, and the errors may have affected the previous results. The errors are as follows: (1) the original SOFE source code, version May 1982, had a repeated line of code, line "LJ = LJ + K" in Subroutine PSQRT. Subroutine PSQRT computes the Cholesky upper triangular square root matrix of the Kalman filter covariance matrix, $\underline{P}(t_i)$. This error would miscalculate the upper triangular elements of the Cholesky square root matrix. (2) Jamerson's source code had an error in the parameter list of the call statement, "CALL RICDSD(·)". This library-obtained subroutine calculated the steady-state solution to the Riccati equation used to calculate the controller gains (see Equation 3-67). This error resulted in slightly higher controller gains than would be properly evaluated. (3) Zicker's source code, which was modified and used by Moose and Jamerson, calculated the true maximum amplitude of the rate function, Λ_t, as a function of the filter beam dispersion, R_f (see Equation (4-22)). The true beam dispersion, R_t, should have been used instead. This slightly affected the results from the previous robustness analyses for the beam dispersion.

The results from the MMAC analysis are from a corrected version of the existing software which is modified for adaptive estimation/control. All the subroutines used to implement the MMAC were verified independently of SOFE.

5.1 Evaluating Alternate Weighting Matrices

The purpose of this study is to select the cost weighting matrix that best minimizes the time-averaged RMS error, RMS_e, between the target and the beam without using excessive amounts of control. Therefore, seven different performance analyses have been performed with a different cost weighting matrix as defined by the steady-state cost function,

$$J = \Sigma \{1/2 \cdot \begin{bmatrix} x_B(t_i) \\ x_{TP}(t_i) \\ q(t_i) \\ u(t_i) \end{bmatrix}^T \begin{bmatrix} X_{11} & -X_{11} & 0 & 0 \\ -X_{11} & X_{11} & 0 & 0 \\ 0 & 0 & X_{55} & 0 \\ 0 & 0 & 0 & U \end{bmatrix} \begin{bmatrix} x_B(t_i) \\ x_{TP}(t_i) \\ q(t_i) \\ u(t_i) \end{bmatrix} \qquad (5\text{-}1)$$

The results as depicted by Table 5-1 and 5-2 show that the best cost weighting matrix is composed of U=1, X_{11}=100 and X_{55}=10. A closer analysis shows that a 10:1 ratio of X_{11} to X_{55} is the important relationship that future designs should

TABLE 5-1

The RMS Errors from the Cost Weighting Matrices Analysis

U	X_{11}	X_{55}	Average RMS$_e$	Minimum RMS$_e$ (t_i)	Maximum RMS$_e$ (t_i)	%
1	1	1	2.536	2.258	2.881	+3.6%
1	10	10	2.639	2.314	2.918	+7.8%
1	10	100	2.929	2.526	3.251	+19.7%
1	100	10	2.448	2.165	2.737	-
1	100	100	2.676	2.336	2.963	+9.3%
1	100	1000	2.956	2.546	3.286	+20.7%
1	1000	100	2.450	2.166	2.740	+0.1%

TABLE 5-2

The Applied Control During the Second Monte Carlo Run

U	X_{11}	X_{55}	RMS of $u(t_i)$	Minimum $u(t_i)$	Maximum $u(t_i)$	*
1	1	1	18.363	-50.896	18.046	3
1	10	10	18.413	-51.153	18.688	4
1	10	100	18.586	-52.456	20.231	6
1	100	10	18.330	-51.276	17.451	1
1	100	100	18.440	-51.365	18.753	5
1	100	1000	18.608	-52.478	20.470	7
1	1000	100	18.333	-51.283	17.740	2

Note: * is the order of performance with #1 being the best.

consider. This cost weighting matrix allows the controller to operate with the lowest average RMS error among the seven sets evaluated. Although none of the cost weighting matrices allow the controller to use significantly large amounts of control in relation to the severity of the target dynamics, the best cost weighting matrix provides the least oscillatory application of control as measured by the applied control RMS statistic (see Table 5-2). These findings are consistent with the theory presented in Section 3-4. As we increase the pseudo-integrator cost weighting element, X_{55}, with respect to the position states cost weighting element, X_{11}, the controller will begin to overcompensate large differences in the target and beam positions. This will cause the controller to be more oscillatory in the application of control and it will lead to larger RMS errors. Also, as the cost weighting of control, U, approaches the values of the other cost elements, the control will become more expensive to apply, and this will tend to raise the average RMS error. A plot of the RMS error statistic is plotted in Figure 5-1. Except for analyzing the pole placement of the controller (refer to Chapter 6), nothing was done to enhance the robustness of the controller in any explicit manner.

The relationship between the amount of applied control and the relative error between the target and the controlled beam is shown in Figure 5-2. This plot is from the second simulation run out of the Monte Carlo analysis and shows how

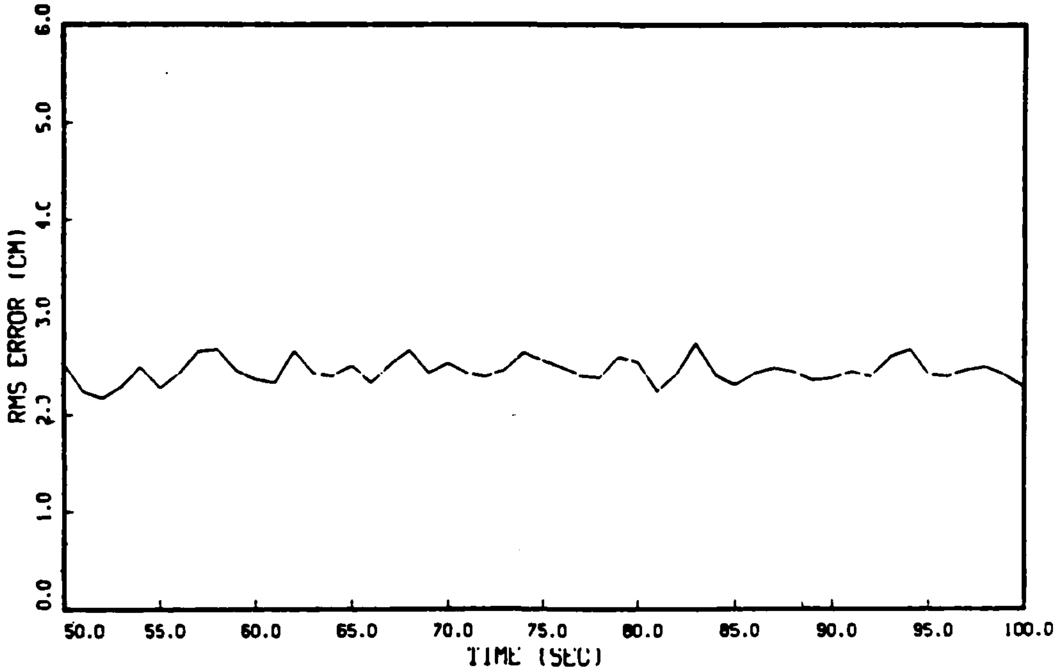


Figure 5-1. RMS Tracking Error for U=1, X_{11}=100, X_{22}=10

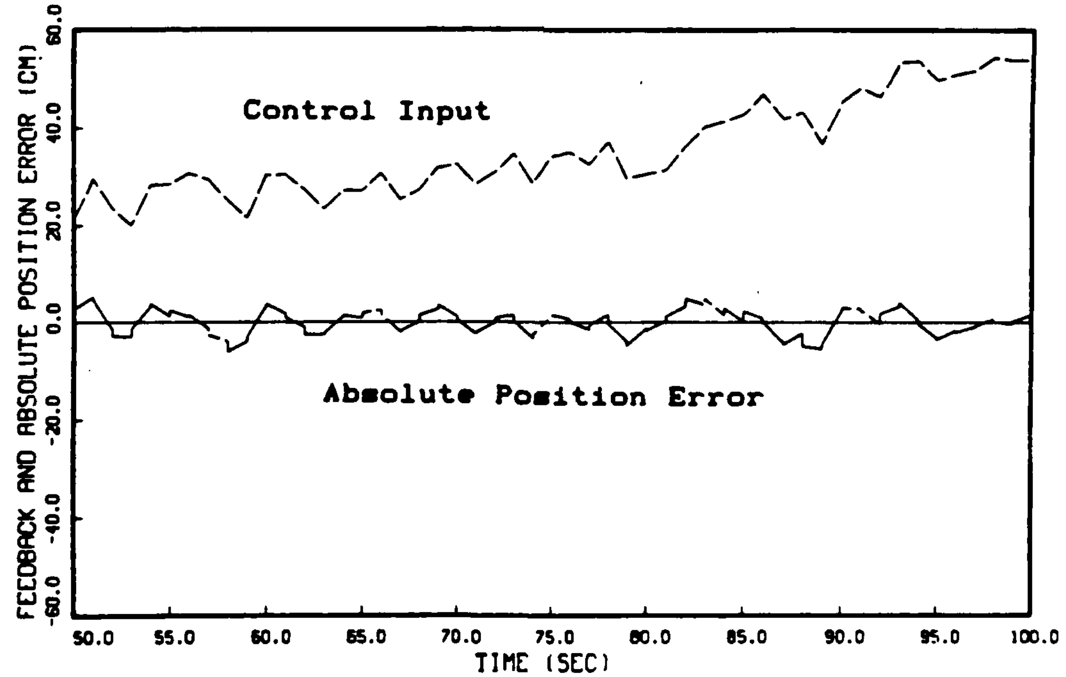


Figure 5-2. Applied Control and the Relative Error Between the Target and the Beam (Absolute position error = $x_B - x_{TP}$)

the applied control reacts to the target. In this example, the target is accelerating across the detector array of the tracking particle beam, which causes the magnitude of applied control to increase continuously. The plot starts at t_i=50 seconds because the first 50 seconds is being used to let the system's transient response attenuate. The discontinuities in the relative error are do to the Kalman filter updates.

The previous concerns [8] over the effect of a large initial overshoot caused by the beam position being offset from the target are important because the initial transient response can saturate the actuators and be very destablizing. They were ignored only because the emphasis is on the steady state operation of the estimator/controller, and it is suggested that during target acquisition a different cost weighting matrix or filter tuning parameter be used which is more forgiving of large initialization errors.

5.2 Evaluating the Simplified Meer Filter

The purpose of this analysis is to evaluate the performance of the controller when the depth of the Meer filter is reduced from D=3 to D=1. The goal is to reduce the computational loading of the Meer filter significantly at the cost of slightly degrading the performance of the controller. The insight was provided by Zicker [27] who found that when the Meer filter was reduced from a depth of 3 to 1, the average RMS error increased by only 0.004 cm.

The results from the controller's performance analyses show that the controller is even more tolerant to the changes in filter depth. As the data in Table 5-3 suggest, there is no detectable degradation of the controller's performance when the depth of the Meer filter is reduced to one. Unfortunately, the expected savings in CPU time did not appear.

TABLE 5-3

Results of the Reduced Meer Filter Analysis

Depth	U	X_{11}	X_{55}	Average RMS$_e$	Minimum RMS$_e$ (t_1)	Maximum RMS$_e$ (t_1)	CPU Time
3	1	100	10	2.44795	$2.1653\underline{0}$	$2.736\underline{17}$	511.5
1				2.44795	$2.1653\underline{3}$	$2.736\underline{70}$	521.6
3	1	100	100	2.67559	$2.3357\underline{6}$	$2.9627\underline{3}$	500.5
1				2.67559	$2.3357\underline{5}$	$2.9627\underline{2}$	497.0

Note: CPU Time is for the entire simulation, which includes data reduction and plotting; units are in seconds. Underlining of digits is used to accentuate the differences in computed statistics; units are in cm.

Because these findings did raise some suspicion, the analysis was supported by an in-depth review of the source code. The review indicated that the code was correct, and that the reason there was no apparent reduction in CPU time was due to the general inefficiency of the source code which did not take advantage of the reduced filter depth.

One of the reasons that contributes to the outstanding performance of the controller with the reduced Meer filter is the simple beam dynamics used by both the plant and filter. A more realistic plant might have had a much more complicated transfer function which could lead to greater, more discernible errors when the Meer filter's depth is reduced. In other words, a depth of one might be sufficient for a first-order Gauss-Markov scalar position process, but a greater depth might be required if a higher-order Gauss-Markov model were a significantly better model. Another reason is that the high SNR (SNR=20) provides few noise-induced events to test the Meer filter. Combining the effects of a high SNR with the effects of a tight beam dispersion (R=0.5 cm^2) and a long detector array (L=10 cm), results in the fact that most of the noise events should be easily detected as being induced by noise rather than signal, and the Meer filter does not require the extra filter depth to accomplish this adaptive decision-making.

5.3 Sensitivity Analysis

The purpose of the sensitivity analysis is to evaluate a "fully parameter-knowledgeable" controller, in which both the Kalman and Meer filters (as well as the controller gain computations) have access to the true parameters. The results provide the best possible performance that can be expected of the controller under various "real world" conditions, and the results define the baseline for the robustness analysis. Throughout the sensitivity and robustness (Section 5.4) analyses, the Meer and Kalman filter parameters are set to the the nominal conditions defined by Table 5-4, and then each "real world" parameter is evaluated individually through variations plus or minus one order of magnitude. In these sensitivity studies, the corresponding filter/controller-assumed parameter is varied accordingly, while in the later robustness studies, the filter/controller-assumed parameter is left unchanged. The sensitivity results from the beam parameters are presented first.

TABLE 5-4

Nominal Conditions for the Sensitivity and Robustness Analyses

Beam Parameters (see Section 4.3)	
τ_B = 20 (sec)	D = 1
g = 0.2 $(cm^2/sec)^{1/2}$	R = 0.5 (cm^2)
SNR = 20	n = 100 (signal events)
Target Parameters (see Section 4.3)	
τ_T = 10 (sec)	R_T = 0.5 (cm^2)
Q_T = 0.1 (cm^2/sec)	
Non-Zero Elements in the Cost Weighting Matrix (see Section 3.4 and Equation 3-62)	
U = 1 X_{11} = 100 X_{55} = 10	

5.3.1 Beam Time Constant - τ_B. The beam time constant represents the relative speed with which the particle beam dynamics can react to the applied control. The effect τ_B has on the plant dynamics can be seen by means of:

$$G_x(s) = \frac{B}{s - p_1} = \frac{B}{s + (1/\tau_B)} \qquad (5\text{-}2)$$

where $G_x(s)$ is the plant transfer function. The effects of increasing or decreasing τ_B has the effect of moving the pole closer to or away from the origin of the s-plane. The effects of moving the pole can be evaluated by looking at the plant's settling time for a step input,

$$t_s = 3.912/|p_1| = 3.912 \cdot \tau_B \qquad (5\text{-}3)$$

where t_s is defined as the time it takes the exponentially time-correlated plant dynamics to reach 98 percent of the final steady state value [3]. As τ_B is decreased, the pole moves farther from the origin of the s-plane, the beam dynamics become quicker, and it should be easier for the particle beam to track the target. This should result in a smaller average RMS error and this is confirmed by the results in Table 5-5. But, contrary to what one might expect, the large changes in τ_B do not result in large changes in the tracker error. This insensitivity is due to the fact that the system's frequency response (as can be observed in a Bode plot) is not only a function of the corner frequency, ω_c ($\omega_c = 1/\tau_B$), but also it is a function of the log magnitude of the open loop gain, K_M ($K_M = B\tau_B$), and the effects of changing τ_B are cancelled by changes in K_M. The filter's τ_B represents the time constant in the filter's model of the plant, whereas the true τ_B is the actual time constant of the particle beam plant. In a sensitivity analysis, the true and filter parameter values are equal.

TABLE 5-5

The Sensitivity Analysis Results for τ_B

τ_B (seconds)		Average RMS$_e$	%	Comments
Filter	True			
2	2	2.431	-0.7%	Quicker beam dynamics
20	20	2.448	-	Nominal conditions
200	200	2.456	+0.3%	Slower beam dynamics

5.3.2 <u>Beam Propagation Noise Strength</u> - g^2 The beam propagation noise strength indicates the confidence we have in the target model being correct and the magnitude of the random fluctuations that the beam position actually undergoes. As we increase the strength of the noise, g^2, we are increasing the magnitude of the random beam position fluctuations, and the Meer filter must depend more on the measurement update. This can be demonstrated by analyzing this relationship with an elemental Snyder-Fishman filter covariance propagation equation, and gain equation:

$$P(t_{i+1}^-) = \Phi^2(t_{i+1},t_i)P(t_i^+) + Q_d(t_i) \tag{5-4}$$

$$K(t_i) = P(t_i^-)/(P(t_i^-) + R) \tag{5-5}$$

where the closed-form solution of $Q_d(t_i)$ is

$$Q_d(t_i) = g^2\tau_B[1-\exp(-2\Delta t/\tau_B)]/2 \tag{5-6}$$

where Δt is the varying sample time between the arrival of events. As g^2 increases, so does $P(t_{i+1}^-)$ increase; therefore, $K(t_i)$ will increase toward a value of one, and the filter will begin to rely more heavily on the measurement update.

With a sensitivity analysis, we assume that the filter g^2 matches a hypothetical true g^2, which is used in the true state dynamics model,

$$x_{Bt}(t_{i+1}) = \Phi_{Bt}(t_{i+1},t_i)x_{Bt}(t_i) + B_d(t_i)u(t_i) + w_{dt}(t_i) \tag{5-7}$$

where $w_{dt}(t_i)$ is a zero-mean, white Gaussian, discrete-time, stochastic process with a variance as given in Equation (5-6). As we increase the true g^2 (and the filter g^2 correspondingly), the magnitude of the random fluctuations beam undergoes increases, and the filter must depend more heavily on the measurements which come from the stochastic truth model. Therefore, as the true g^2 increases and the actual RMS value of excursions of the beam increase, so should the average RMS error. The results in Table 5-6 confirm this.

TABLE 5-6

The Sensitivity Analysis Results for g^2

$g\ (cm^2/sec)^{1/2}$ Filter	True	Average RMS_e	%	Comments
0.02	0.02	2.419	-1.2%	$K(t_i)$ decreases
0.2	0.2	2.448	-	Nominal conditions
2.0	2.0	3.743	+53.9%	$K(t_i)$ increases

5.3.3 <u>Beam Dispersion</u> - <u>R</u> The particle beam is said to have a Gaussian-shaped dispersion as measured with respect to the surface of the photodetector. Because the beam is assumed to have Gaussian distribution (see Equation 2-1), it can be represented statistically by the true mean, $\hat{x}_{Bt}(t)$, and the true "variance", $R_t(t)$. As $R_t(t)$ increases, the signal-induced events spread over a larger area of the detector, making it more difficult to estimate the center of the beam, and forcing the filter to rely more heavily on its internal dynamics model. Although Table 5-7 supports this concept, the filter is relatively insensitive to equal changes in both the true and filter R's.

TABLE 5-7

The Sensitivity Analysis Results for R

R (cm^2) Filter	True	Average RMS_e	%	Comments
0.05	0.05	2.437	-0.4%	Depend on $z_B(t_i)$
0.5	0.5	2.448	-	Nominal conditions
5.0	5.0	2.475	+1.1%	Depend on filter

5.3.4 <u>Signal-to-Noise Ratio - SNR</u> The signal-to-noise ratio is the ratio of the number the signal events to the number of noise events that arrive over a period of time. Within the context of this simulation, the filter SNR is used to estimate the noise rate parameter (the SNR is defined in Equation (4-24),

$$\lambda_{Nf}(t,\underline{r}) = [n/(t_F - t_0)]/[SNR_f \cdot L] \tag{5-8}$$

The true SNR is used to generate the the actual noise parameter rate used to generate the noise events. Thus, when the SNR is decreased, we expect to see the increase of noise events to drive up the time-averaged RMS error, as depicted in Table 5-8. The table shows a discrepancy in that the time-averaged RMS error shows a slight increase when the SNR is increased. After reviewing the results from the robustness analysis (see Section 5.4.4), one can see that the controller is highly insensitive to changes in SNR, and that the small deviations of time-averaged RMS error is within the sample statistical error. This is also supported by the fact that the different noise arrival rates will cause different random-number generated signal- and noise-induced event patterns (i.e., the same random number generator seed was used throughout the study, and changes in signal or noise parameter rates will alter the simulation pattern of signal- and noise-induced events by requiring different numbers of calls to the random number generator).

TABLE 5-8

The Sensitivity Analysis Results for SNR

SNR Filter	True	Average RMS$_e$	%	Comments
2	2	2.469	+0.9%	1/3 noise events
20	20	2.448	-	Nominal conditions
200	200	2.466	+0.7%	Nearly noise free

5.3.5 Expected Number of Signal-Induced Events - n

The expected number of signal events is used to calculate the filter's maximum amplitude of the rate function, as by Equation (4-22),

$$\Lambda_f = n_f / [(2\pi R_f)^{1/2} \cdot (t_f - t_0)] \qquad (5\text{-}9)$$

which, in turn is used by the filter to calculate the conditional probability that a signal-induced event did occur (see Equations (2-1) and (2-27). The true n is used to calculate the signal arrival rate, $\lambda_s(t)$. As both the true and filter n are decreased by equal amounts, the filter must rely more heavily on the filter dynamics model because the mean Meer filter sample period is increased. Therefore, we should expect the time-average RMS error to increase as shown in Table 5-9. The RMS error statistics may be slightly affected by differences in the random-number generated signal- and noise-induced event patterns caused by a change in the signal parameter rate. The n=1000 performance run was not accomplished because the expected results did not warrant the high computer costs. More signal events should slightly increase the filter's accuracy. The n=10 performance run should have produced an unstable system (according to findings in Chapter 6) because the mean sample period of 10 seconds would place the poles of the discrete-time controller far outside the unit circle, but the perfectly matched filter time constants prevented us from seeing the instability.

TABLE 5-9

The Sensitivity Analysis Results for n

n (signal events) Filter	True	Average RMS_e	%	Comments
10	10	2.540	+3.8%	Emphasis on filter internal dynamics model
100	100	2.448	-	Nominal conditions
1000	1000	-	-	Not performed

5.3.6 Target Time Constant - τ_T The target acceleration time constant is used in the target shaping filter which simulates the target dynamics. The shaping filter is a linear, time-invariant system driven by a stationary white Gaussian noise process that has a mean of zero and a variance of Q_T. The target dynamics position process is modeled as the double integral of exponentially time-correlated acceleration; a third-order Markov process that is the output of a third order shaping filter which is shown in Figure 5-3. By definition, the white noise process

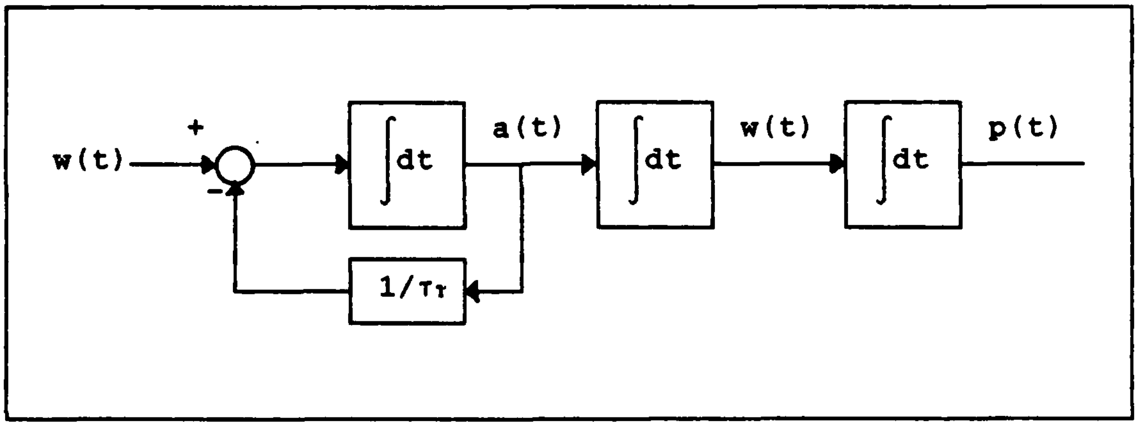


Figure 5-3. Target Shaping Filter

has a power spectral density (PSD) magnitude of Q_T and an infinitely wide bandwidth as shown in Figure 5-4a. When this white noise is passed through a first-order lag filter, as is done in the target shaping

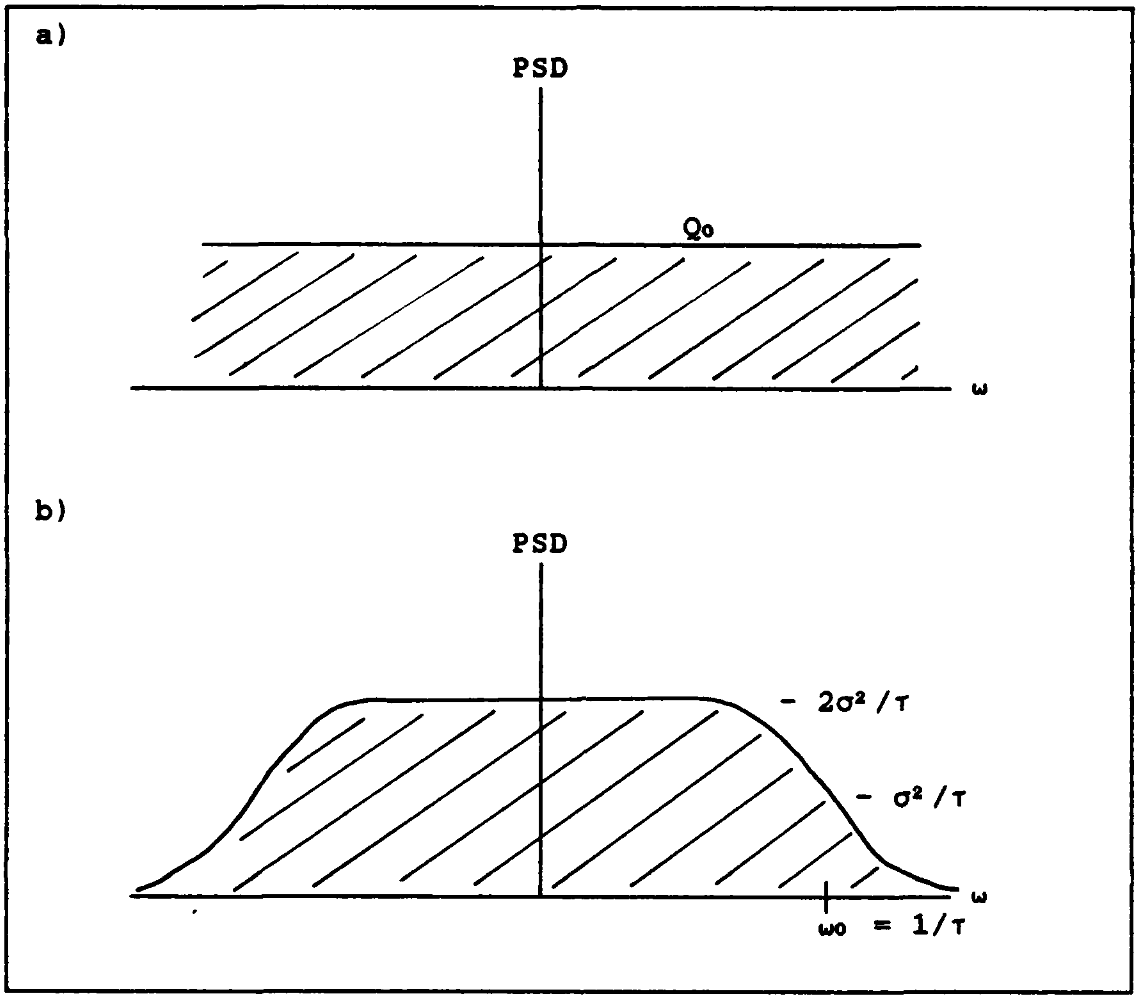


Figure 5-4. PSD Functions: a) White Noise, b) Target Acceleration

filter, it becomes the acceleration of the target, and has a power spectral density function of

$$PSD_A(\omega) = \frac{2\sigma_T^2/\tau_T}{\omega^2 + (1/\tau_T)^2} = \frac{Q_T}{\omega^2 + (1/\tau_T)^2} \qquad (5\text{-}10)$$

where the variance (mean squared value) of the target acceleration is defined as

$$\sigma_T^2 = (RMS_A)^2 = \tfrac{1}{2} Q_T \tau_T \qquad (5\text{-}11)$$

A plot of the acceleration power spectral density function is shown in Figure 5-4b. The effects of reducing τ_T would increase the bandwidth of the target acceleration and lead to faster dynamics, but with a much smaller amplitude as shown in Figure 5-5. The overall results of a small τ_T is a lower RMS_A (assuming Q_T is held constant). Therefore, smaller τ_T should lead to smaller time-averaged RMS errors as shown in Table 5-10.

TABLE 5-10

The Sensitivity Analysis Results for τ_T

τ_T (sec) Filter	True	Average RMS_e	%	Comments
1.0	1.0	1.325	-45.9%	σ_T^2=0.05, PSD=0.1
2.5	2.5	1.882	-23.1%	σ_T^2=0.125, PSD=0.625
5.0	5.0	2.225	-9.1%	σ_T^2=0.25, PSD=2.5
10.0	10.0	2.448	-	Nominal: σ_T^2=½, PSD=10
100.0	100.0	2.685	+9.7%	σ_T^2=5.0, PSD =1000

Note: PSD is calculated for PSD(ω=0), i.e., the low frequency asymptotic value at ω=0.

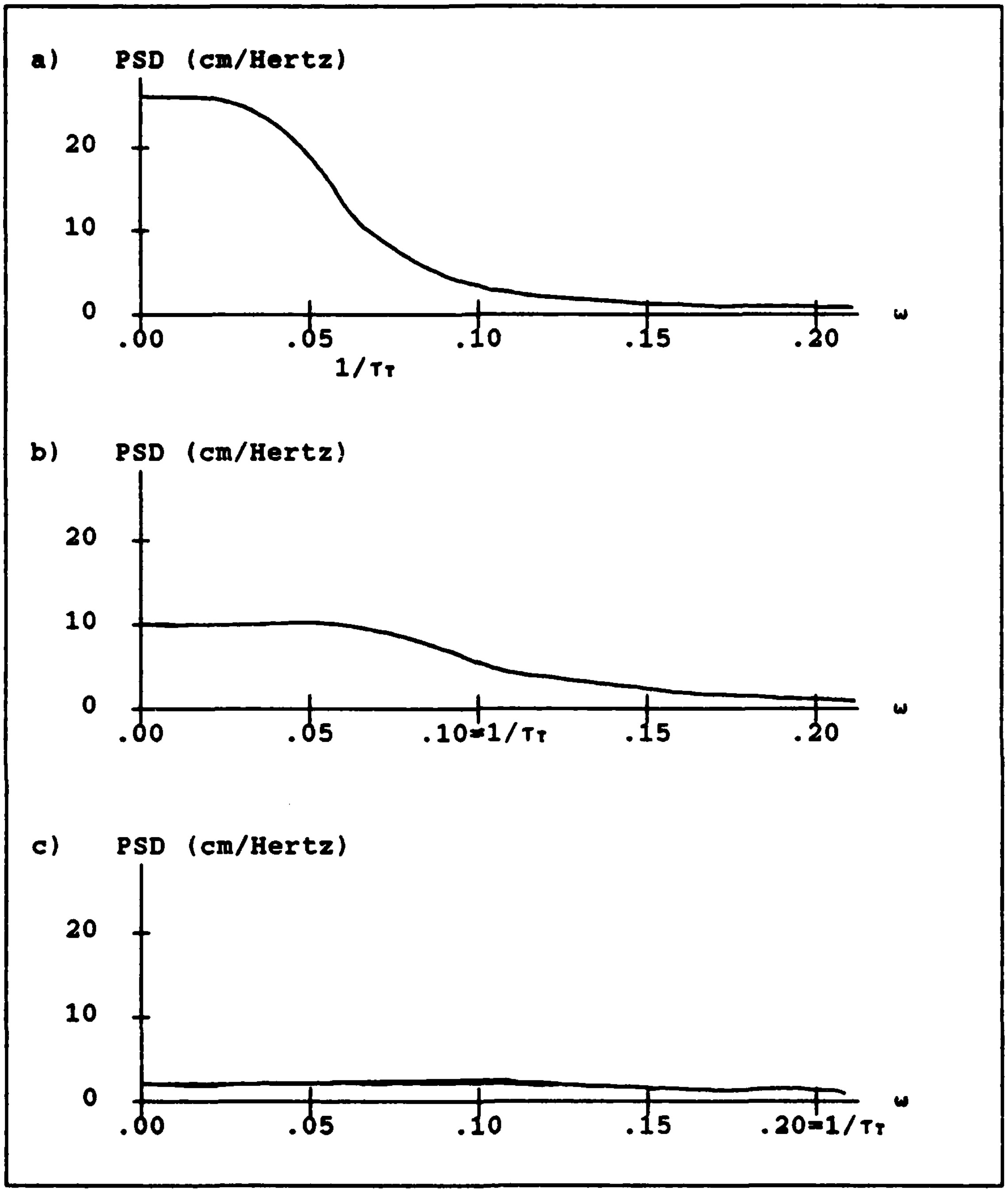


Figure 5-5. The Effects of Different τ_T's on the PSD:
a) τ_T = 15 (sec), b) τ_T =10 (sec), c) τ_T = 5 (sec)

5.3.7 Target Dynamics Noise Strength - Q_T The target dynamics noise strength is the strength of the white Gaussian noise used to drive the shaping filter. Although it does not effect the bandwidth of the target acceleration, it directly affects the amplitude of the power spectral density function as can be seen by setting ω to zero in Equation (5-10):

$$PSD(\omega=0) = Q_T \tau_T^2 \tag{5-12}$$

Therefore, an increase in Q_T will increase the RMS_A (see Equation (5-11)) and should result in an increase in RMS error. Table 5-11 confirms this. The true Q_T is the actual white noise strength used to drive the shaping filter, where the filter Q_T is used by the Kalman filter in the target state estimate and covariance propagation equations.

TABLE 5-11

The Sensitivity Analysis Results for Q_T

Q_T (cm^2/sec^5) Filter	True	Average RMSe	%	Comments
0.01	0.01	1.461	-40.3%	$\sigma_T^2=0.05$, PSD=1
0.1	0.1	2.448	-	Nominal: $\sigma_T^2=½$, PSD=10
1.0	1.0	4.644	+89.7%	$\sigma_T^2=5.0$, PSD=100

Note: PSD is calculated for PSD(ω=0).

5.3.8 Evaluating τ_T and Q_T with the Target Acceleration Power Spectral Density Maximum Set to 10.0

Although it appears that the filter is much more sensitive to Q_T, one more analysis is required to confirm this. For this analysis, the amplitude of the low frequency asymptote of PSD (see Figure 5-5) is held constant, and τ_T is varied. The results in Table 5-12 show that, for targets with the same low frequency acceleration PSD characteristics, those which can exhibit more violent maneuvering (i.e., the τ_T is smaller, or the acceleration process bandwidth $1/\tau_T$ is larger) are much more difficult to track.

TABLE 5-12

The Sensitivity Analysis Results for PSD

PSD = 10.0 (cm^2/Hz) at $\omega=0$					
Filter		True		Average	
τ_T	Q_T	τ_T	Q_T	RMS_e	%
1.0	10.0	1.0	10.0	5.384	+119.9%
10.0	0.1	10.0	0.1	2.448	-
100.0	0.001	100.0	0.001	1.089	-55.5%

Note: PSD is calculated for PSD($\omega=0$).

5.3.9 Target Measurement Noise Variance - R_T

The target measurement noise variance indicates our confidence in the measurement model. As the measurements becomes more severely corrupted, the measurements assume a wider and flatter Gaussian distribution centered around the true target position, and the variance, which is represented by the true R_T, increases. The filter R_T is the filter's value representing the actual R_T. As the filter R_T increases, the Kalman filter gain decreases and the filter must rely more heavily on its internal dynamics model, and tracking is more difficult with less precise sensors. Therefore, an increase in R_T should result in a higher time-averaged RMS error. Table 5-13 confirms this.

TABLE 5-13

The Sensitivity Analysis Results for R_T

R_T (cm^2)		Average		
Filter	True	RMS_e	%	Comments
0.05	0.05	1.518	-38.0%	Relies on measurements which are precise
0.5	0.5	2.448	-	Nominal conditions
5.0	5.0	4.473	+82.7%	Relies on filter's model and measurements are imprecise

5.4 Robustness Analysis

The purpose of the robustness analysis is to evaluate the controller's performance when the embedded Kalman or Meer filters mismodels the "real world". Initially, all the parameters are set to the nominal conditions, and then the true parameters are evaluated one at a time at values above and below the nominal condition to simulate the effects of the filters mismodeling the "real world". The results from the robustness analysis are evaluated against a baseline defined by the results of the sensitivity analysis. Tracking performance is highly insensitive to the changes in most of the true beam parameters, and by definition, this is a demonstration of excellent robustness characteristics (at the chosen nominal conditions). The notable exception is the beam time constant, for which the results indicate a possible stability problem. Some poor robustness characteristics were demonstrated for all three target parameters, of which, the target dynamics noise strength was selected as the parameter best suited for on-line adaptive estimation.

5.4.1 Beam Time Constant - τ_B Originally, the true τ_B was to be evaluated at plus and minus one order of magnitude of the nominal filter τ_B so that the results would correspond to the sensitivity analysis, but severe computational difficulties led to run-time computer failures before the simulations could be completed. To be able to collect useful data, the true τ_B was evaluated at plus five percent and minus four percent of the nominal filter τ_B. In all, five robustness analyses were conducted with true τ_B equal to 19.2, 19.6, 20.0 (nominal conditions), 20.4, and 21.0 seconds. The results of the analyses are presented in Table 5-14. The respective plots of RMS errors between the target and the beam position, and the mean and standard deviation of the error between the true and filter beam state are shown in Figures 15-6 through 15-15.

The results indicate a severe stability robustness problem. When the true τ_B is simulated at the values 19.2 or 21.0 seconds, the results in the both table and in the plots indicate that the controller's performance is suffering from a strong instability. Growing RMS values and standard deviations that increase with the severity of the mismodeling strongly support the concept that a stability problem exists. These findings led to a sensitivity evaluation of the controller gains and a stability analysis of the PI controller.

TABLE 5-14

The Robustness Analysis Results for τ_B

τ_B (sec)		Average RMS_e	Minimum RMS_e (t_i)	Maximum RMS_e (t_i)	Tuned RMS_e	%
Filter	True					
20	19.2	80.080*	4.555	338.690*	≈2.448	**
20	19.6	4.779	2.920	7.030	≈2.448	+195%
20	20.0	2.448	2.165	2.737	2.448	-
20	20.4	4.687	2.949	7.062	≈2.448	+191%
20	21.0	96.719*	5.301	364.960*	≈2.448	**

Note: % This is the percentage of difference between the average RMS error and the tuned RMS error, which is based on or approximated from the results of the sensitivity analysis (i.e., $\tau_{Bt}=\tau_{Bf}$).

* These values and the results from the Meer filter covariance analysis (see Figure 5-7) indicate that the controller frequently became unstable before the completion of the run (before 100 seconds).

** Extremely large values.

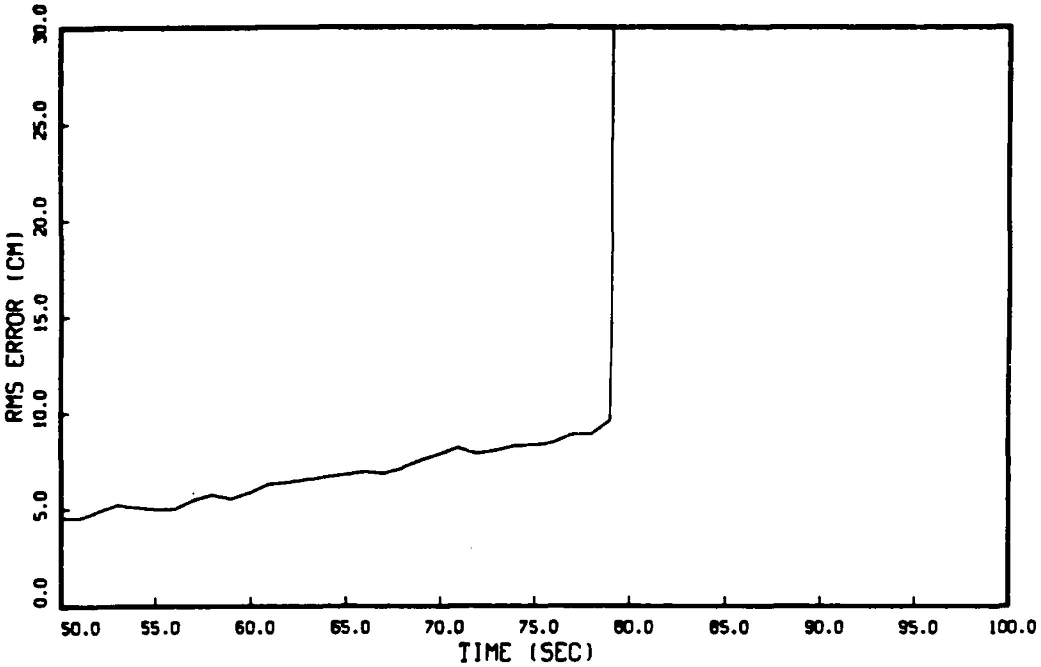


Figure 5-6. RMS Error Between the Target and Beam Position: τ_{Bf}=20.0 (sec), τ_{Bt}=19.2 (sec)

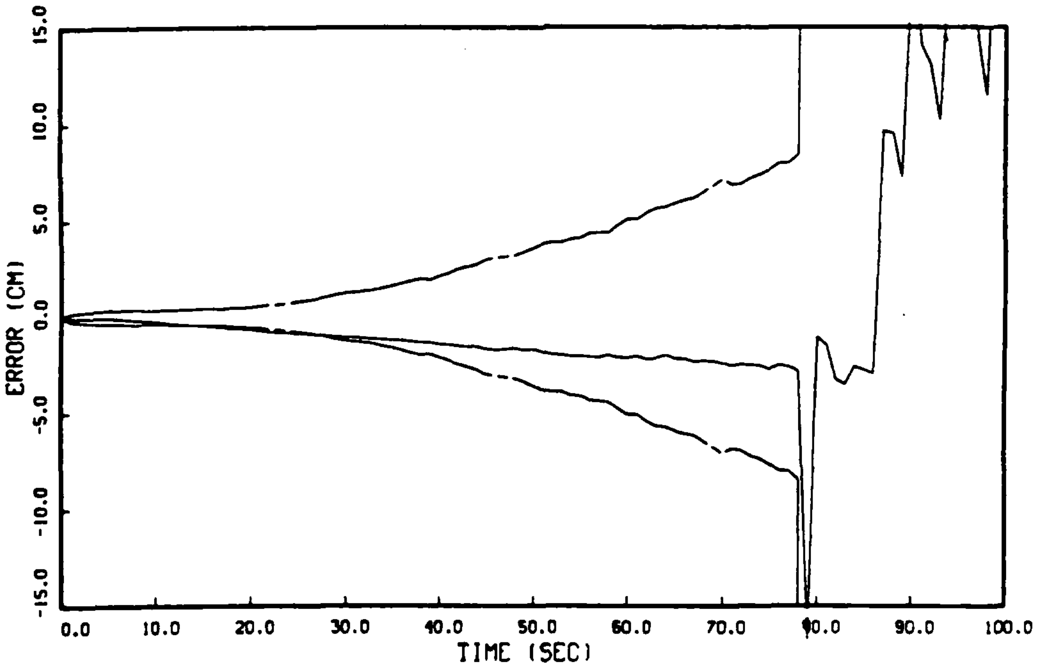


Figure 5-7. Mean and ± Standard Deviation Envelope of the Error between the True and Filter Position States: τ_{Bf}=20.0 (sec), τ_{Bt}=19.2 (sec)

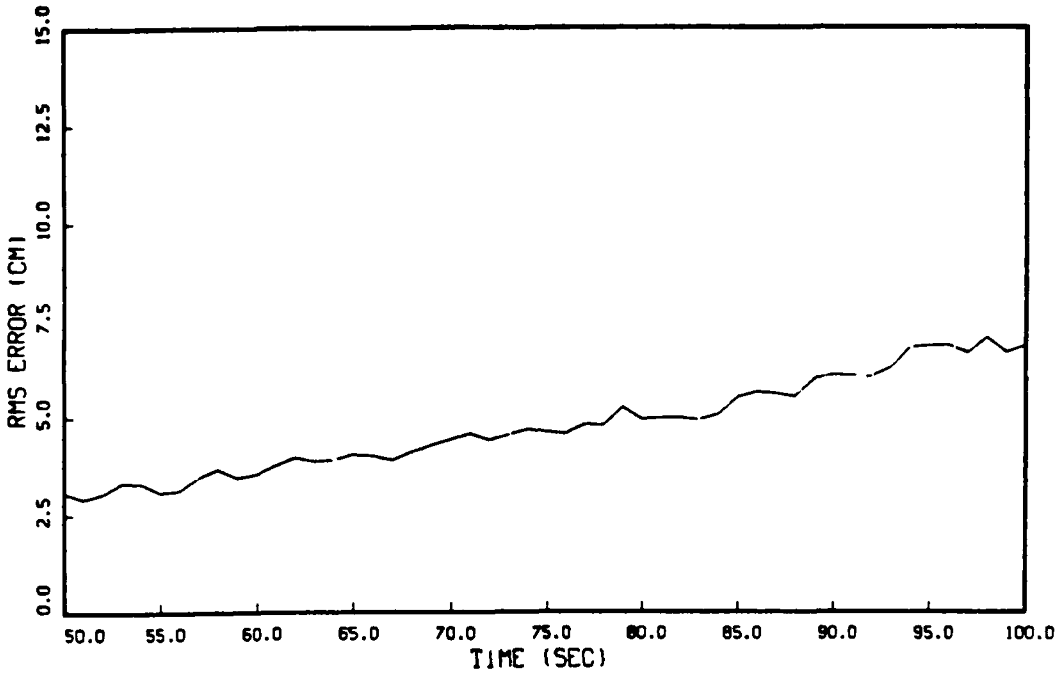


Figure 5-8. RMS Error Between the Target and Beam Position: τ_{Bf}=20.0 (sec), τ_{Bt}=19.6 (sec)

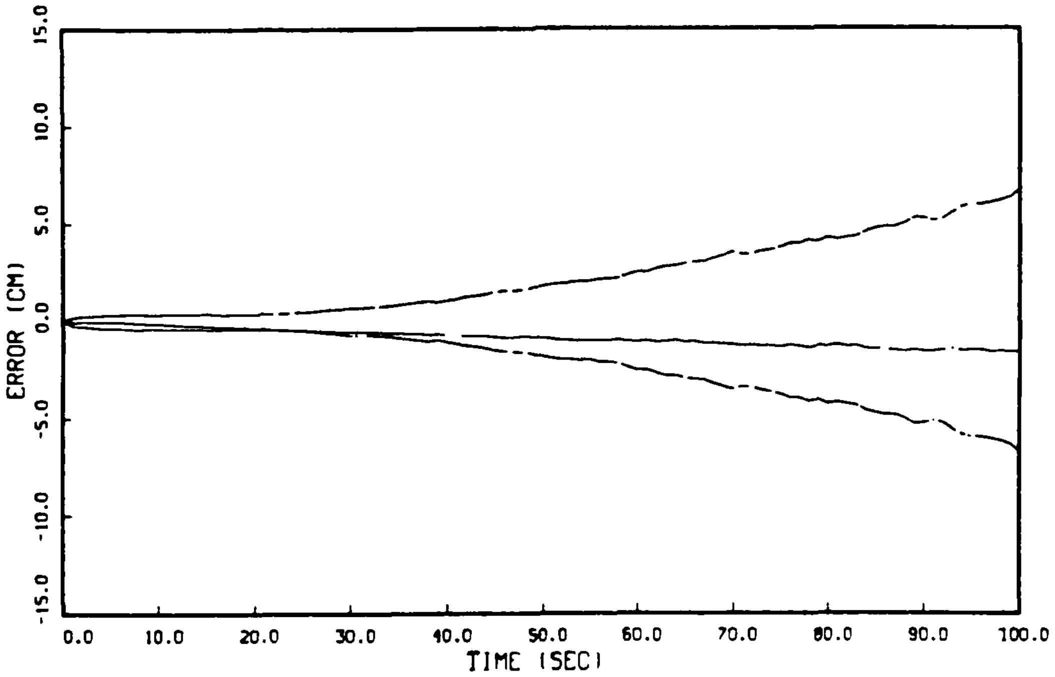


Figure 5-9. Mean and ± Standard Deviation Envelope of the Error between the True and Filter Position States: τ_{Bf}=20.0 (sec), τ_{Bt}=19.6 (sec)

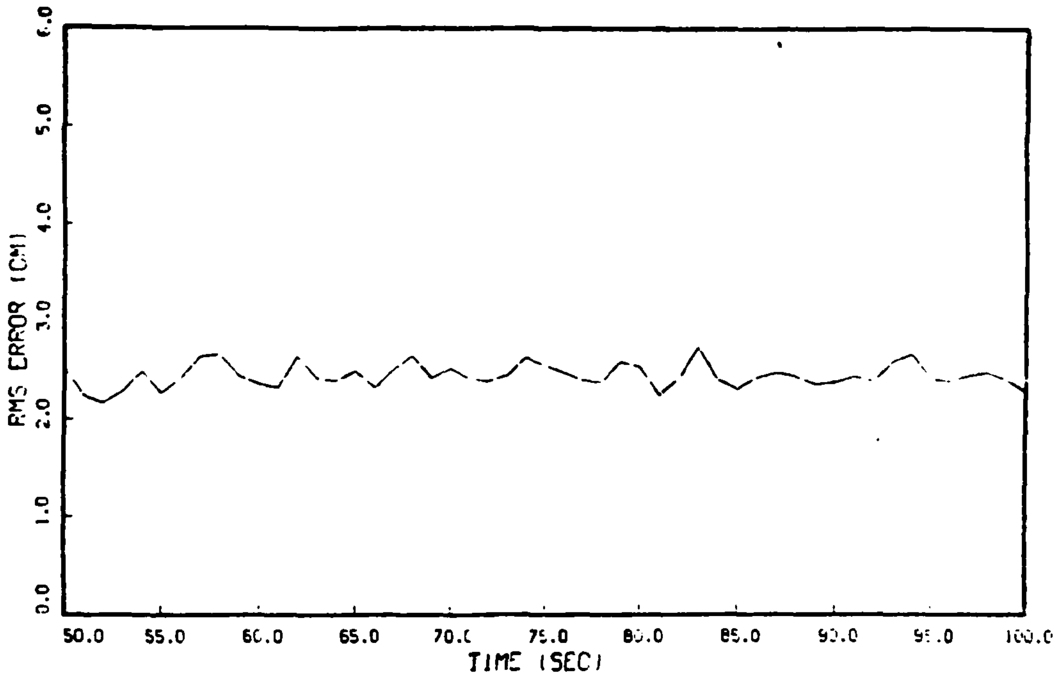


Figure 5-10. RMS Error Between the Target and Beam Position: T_{Bf}=20.0 (sec), T_{Bt}=20.0 (sec)

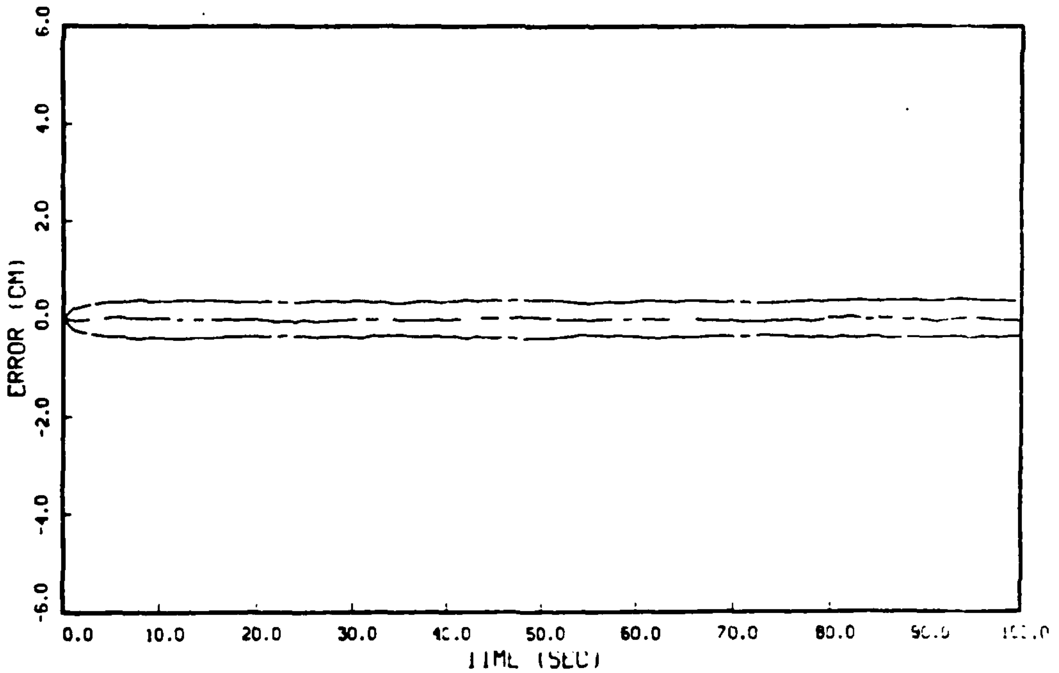


Figure 5-11. Mean and ± Standard Deviation Envelope of the Error between the True and Filter Position States: T_{Bf}=20.0 (sec), T_{Bt}=20.0 (sec)

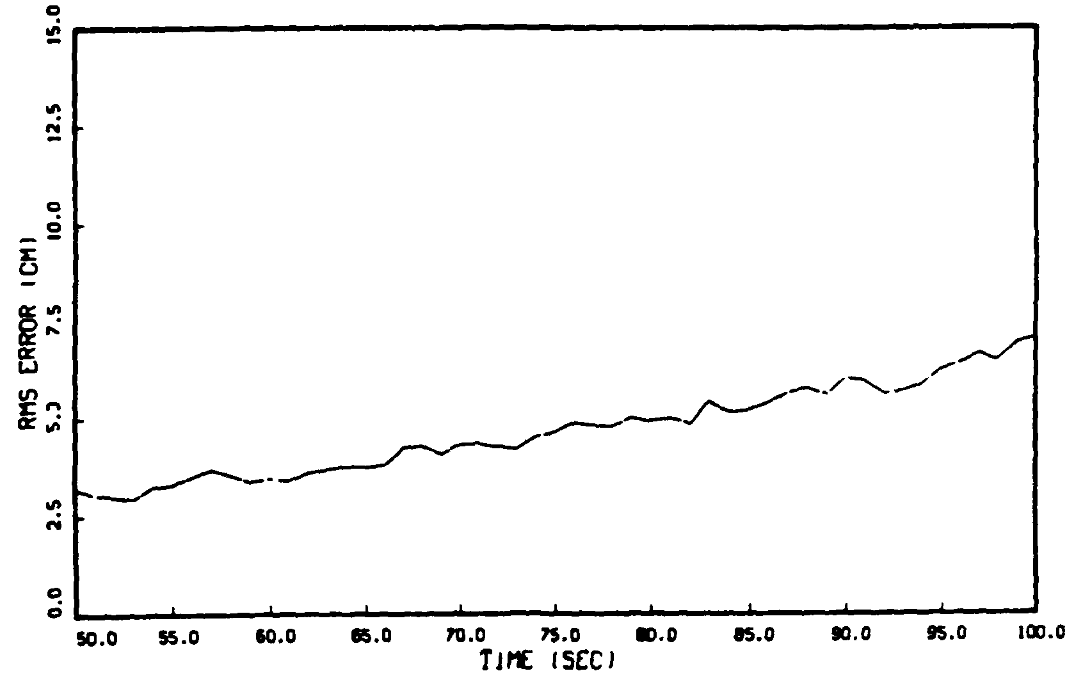


Figure 5-12. RMS Error Between the Target and Beam Position: $\tau_{Bf}=20.0$ (sec), $\tau_{Bt}=20.4$ (sec)

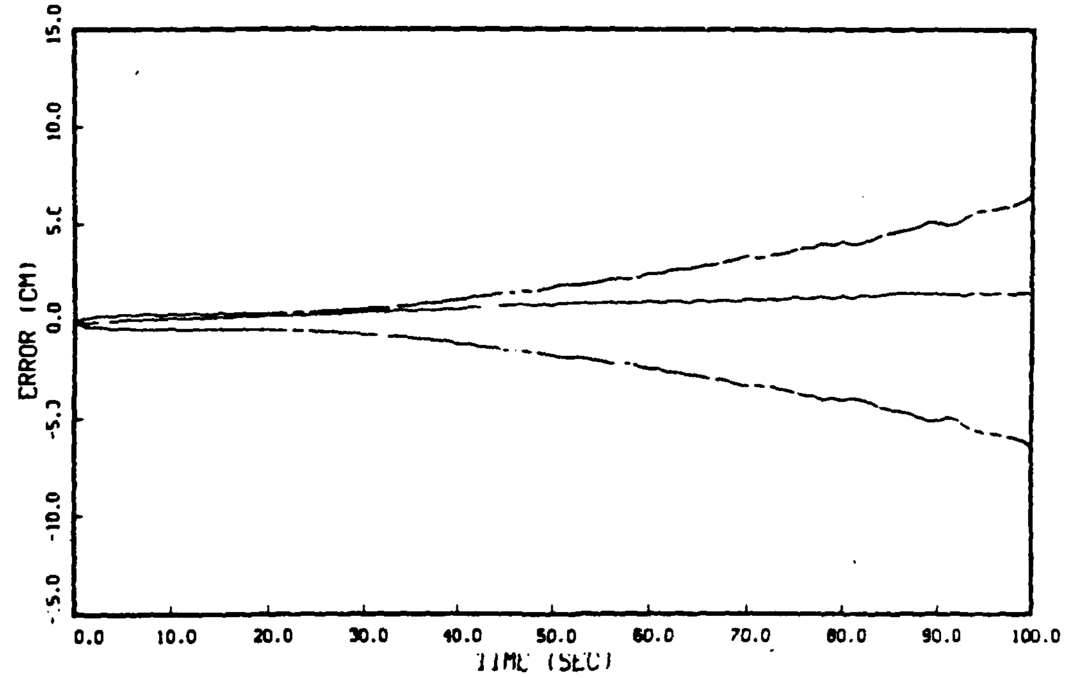


Figure 5-13. Mean and ± Standard Deviation Envelope of the Error between the True and Filter Position States: $\tau_{Bf}=20.0$ (sec), $\tau_{Bt}=20.4$ (sec)

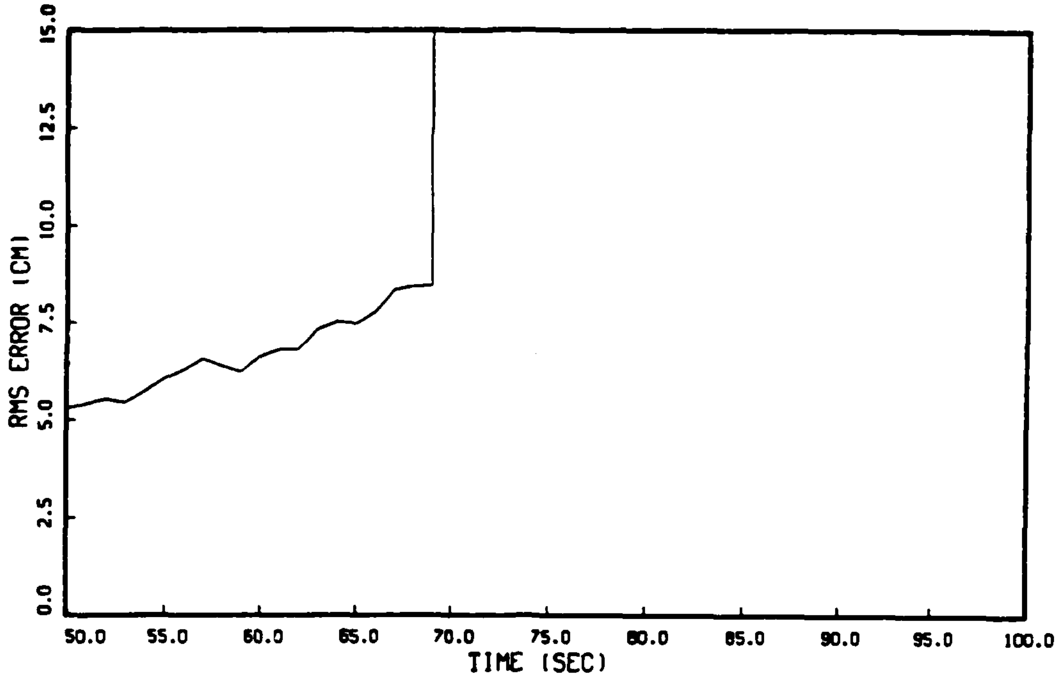


Figure 5-14. RMS Error Between the Target and Beam Position: τ_{Bf}=20.0 (sec), τ_{Bt}=21.1 (sec)

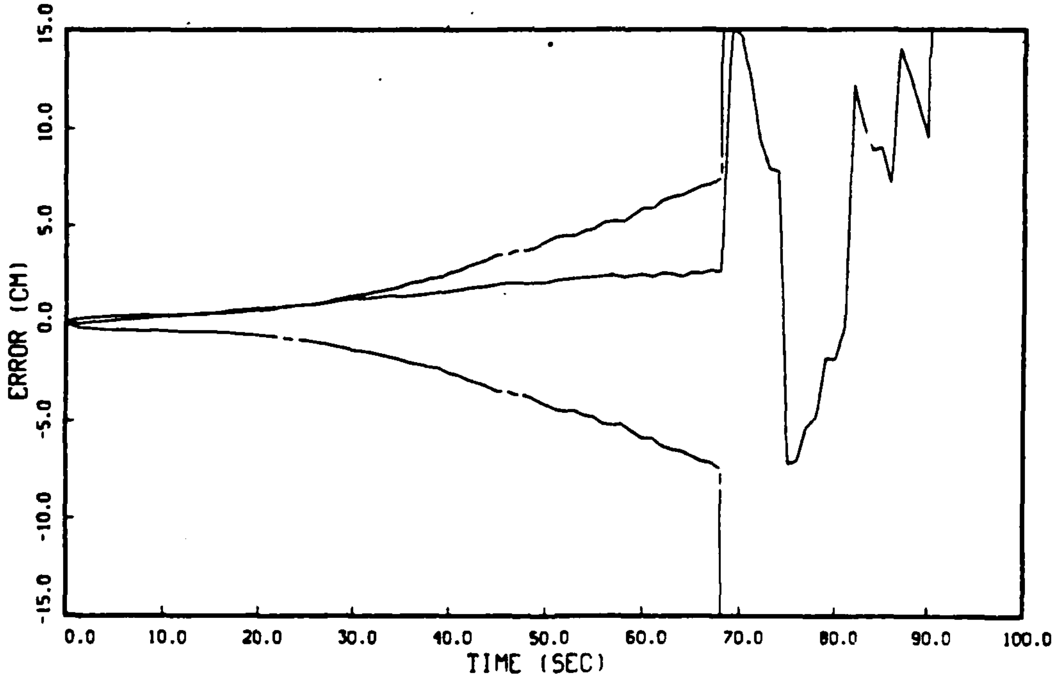


Figure 5-15. Mean and ± Standard Deviation Envelope of the Error between the True and Filter Position States: τ_{Bf}=20.0 (sec), τ_{Bt}=21.0 (sec)

The reason for the controller gain sensitivity evaluation is to see if the gains are sensitive to changes in τ_B. If the controller gains are found to be highly sensitive to changes in τ_B, it is possible that the poles of the full-state feedback system are being driven outside the unit circle, because of the gains, G_c^*, rather than due to the effects of the Meer filter being in the loop. This requires that the controller gain equations, Equations (3-65) through (3-70) be solved for different values of τ_B. The results in Table 5-15 suggest that the controller gains are relativity insensitive to changes in τ_B.

The results of the stability analysis are in Chapter 6, indicating that the stability robustness problem is induced by mismodeling of the real world in the Meer filter dynamics model. Since this is a totally different form of analysis, it is separated into a different chapter. Because we do not know the cause of the instability (at this point in time), the beam time constant would not be suitable for on-line adaptive estimation.

TABLE 5-15

Sensitivity of the Controller Gains to Changes in τ_B

τ_B	G_{c11}^*	G_{c12}^*	G_{c13}^*	G_{c14}^*	G_{c2}^*	E
19.0	$1.24\underline{13}$	$-1.29\underline{38}$	$-1.02\underline{65}$	$-0.48\underline{97}$	$0.27\underline{47}$	$1.07\underline{60}$
20.0	$1.24\underline{22}$	$-1.29\underline{21}$	$-1.02\underline{52}$	$-0.48\underline{80}$	$0.27\underline{44}$	$1.07\underline{46}$
21.0	$1.24\underline{31}$	$-1.29\underline{06}$	$-1.02\underline{40}$	$-0.48\underline{74}$	$0.27\underline{40}$	$1.07\underline{33}$

5.4.2 Beam Propagation Noise Strength - g^2 The controller is rather insensitive to mismodeling of the beam propagation noise strength, as is shown in Table 5-16.

TABLE 5-16

The Robustness Analysis Results for g^2

g^2 (cm^2/sec)		Average RMS$_e$	Tuned RMS$_e$	%
Filter	True			
0.2	0.02	2.427	2.419	+0.3%
0.2	0.2	2.448	2.448	-
0.2	2.0	4.032	3.743	+7.7%

5.4.3 Beam Dispersion - R The controller is rather insensitive to mismodeling of the beam dispersion parameter as can be seen from the results in Table 5-17.

TABLE 5-17

The Robustness Analysis Results for R

R (cm^2)		Average RMS$_e$	Tuned RMS$_e$	%
Filter	True			
0.5	0.05	2.439	2.437	+0.1%
0.5	0.5	2.448	2.448	-
0.5	5.0	2.549	2.475	+3.0%

5.4.4 Signal to Noise Ratio - SNR The results in Table 5-18 indicate that the controller is completely insensitive to SNR parameter mismodeling.

TABLE 5-18

The Robustness Analysis Results for SNR

SNR Filter	 True	Average RMS_e	Tuned RMS_e	%
20	2	2.496	2.496	0.0%
20	20	2.448	2.448	-
20	200	2.466	2.466	0.0%

5.4.5 Expected Number of Signal-Induced Events - n The controller is slightly sensitive to mismodeling of the number of signal events, as the results in Table 5-19 indicate. Note concerning the data -- the tuned (baseline) RMS error for $n_t=200$ is only an approximation and no data was collected at that value during the sensitivity analysis (see Table 5-9). The effect is depicted in Table 5-19 as approximations in the "Tuned RMS_e" column. Also, changes in the signal arrival rate will cause different random-number generated signal- and noise-induced event patterns, which could have contributed to the statistical differences in the time-averaged RMS errors.

TABLE 5-19

The Robustness Analysis Results for n

n (signal events)		Average RMS$_e$	Tuned RMS$_e$	%
Filter	True			
100	10	2.522	≈2.540	-0.7%
100	100	2.448	2.448	-
100	200	2.468	≈2.448	+0.8%

5.4.6 Target Time Constant - τ_T The controller appears to be highly sensitive to target time constant mismodeling when true target time constant is less than the filter time constant (see Table 5-20). This is as anticipated, since a small true τ_T is a harsher target to track than one with a long true τ_T. This lack of robustness indicates that the controller underreacts to the much quicker target even though the target RMS acceleration has decreased substantially (refer to Table 5-10 or Equation (5-11) to see the relationship between τ_T and RMS acceleration). Meanwhile, a much slower target dynamics characteristic in the real world allows the controller to track the target easily. The improvement over the tuned (baseline) value is not easily explained. It is doubtful the entire 1.3 percent difference can be explained as a statistical error since both the baseline sensitivity analysis and the robustness analysis used the same signal- and noise-induced event pattern.

TABLE 5-20

The Robustness Analysis Results for τ_T

τ_T (sec)		Average RMS_e	Tuned RMS_e	%
Filter	True			
10	1	2.087	1.325	+50.7%
10	10	2.448	2.448	-
10	100	2.649	2.685	-1.3%

5.4.7 <u>Target Dynamics Noise Strength</u> - <u>Q_T</u> The controller is highly sensitive to target dynamics driving noise strength mismodeling as indicated by the results in Table 5-21. Because of the high sensitivity and the fact that this is a parameter over which the target has the most control, it was selected for on-line adaptive estimation.

TABLE 5-21

The Robustness Analysis Results for Q_T

Q_T (cm^2/sec^5)		Average RMS_e	Tuned RMS_e	%
Filter	True			
1.0	0.01	3.340	1.460	+128.8%
0.1	0.01	2.035	1.460	+39.4%
0.1	0.1	2.448	2.448	-
0.1	1.0	4.998	4.644	+7.6%
0.01	1.0	7.300	4.644	+57.2%

Note: The filter assumed value of Q_T was changed for first and last entries.

5.4.8 Evaluating τ_T and Q_T with the Target Acceleration Power Spectral Density Maximum Set to 10.0

In this robustness study, the true parameters τ_T and Q_T are varied while the maximum PSD was held to a constant value of 10.0 cm as defined by Equation (5-12). The results in Table 5-23 confirmed the overall controller sensitivity to Q_T, but the findings are not as extreme as the results from the previous two sections might have suggested.

TABLE 5-22

The Robustness Analysis Results for Constant PSD

PSD = 10.0 (cm^2/Hz)				Average RMS$_e$	Tuned RMS$_e$	%
Filter		True				
τ_T	Q_T	τ_T	Q_T			
10	0.1	1	10.0	6.810	5.384	+26.5%
10	0.1	10	0.1	2.448	2.448	-
10	0.1	100	0.001	1.995	1.089	+83.2%

Note: PSD is calculated for PSD(ω=0).

5.4.9 Target Measurement Noise Variance - R_T

Table 5-22 shows that the controller is most sensitive to R_T mismodeling when the measurement of the target state is much more severely corrupted by noise than the filter assumes. This is expected. The importance of this analysis lies with the target's ability to deceive or corrupt tracking by way of jamming the controller's sensors.

TABLE 5-23

The Robustness Analysis Results for R_T

R_T (cm^2) Filter	True	Average RMS_e	Tuned RMS_e	%
0.1	0.01	1.623	1.518	+6.9%
0.1	0.1	2.448	2.448	-
0.1	1.0	6.324	4.473	+41.4%

5.5 Evaluating the Multiple Model Adaptive Controller

This section presents the results obtained during the evaluation of the MMAC design described in Section 3.5. The adaptive controller is composed of three elemental PI controllers, each tuned to a prespecified uncertain parameter value, Q_T, the target model white noise strength. The first and third elemental controllers are based on the target's upper and lower limits of Q_T, where Q_{T1} represents the white noise strength of a benign trajectory (Q_{T1}=0.01 cm^2/sec^5), and Q_{T3} is the maximum white noise strength associated with evasive maneuvering (Q_{T3}=1.0 cm^2/sec^5). The middle elemental controller is based on an intermediate Q_T equal to 0.07 cm^2/sec^5, as opposed to the previous nominal value of 0.1 cm^2/sec^5. This value of dynamics target noise strength was selected from a series of trial tests, each composed of 20 simulation runs. It was found that as Q_{T2} approached either the upper or lower limits, the adaptive controller had difficulty distinguishing Q_{T2} from the other Q_T's, and the controller error increased.

A successful adaptive controller must meet two important criteria. First, the adaptive controller must improve robustness significantly to warrant the additional computational load of an adaptive structure. This can be evaluated by comparing the Monte Carlo analysis results of the MMAC against a baseline performance, the sensitivity analysis results, and the robustness analysis results. The baseline performance is the best possible performance that can be expected of the adaptive controller. The baseline performance is done by informing the adaptive controller of the true Q_T and setting the probabilistic weighting of the "correct" elemental controller to one. The baseline performance in this evaluation used the filter-estimated states, because the filter states were used to generate the results found in Sections 5.1 through 5.4. (An alternative method could assume an "ultimate" baseline controller to have access to the truth states, but this is an artificial assumption that yields an unrealistic baseline of performance.)

The Monte Carlo analysis for the actual adaptive controller and baseline controller uses the following test format: from t=0.0 to t=50.0 seconds, the true Q_T is set to 0.01, and the first 40.0 seconds are used to let the system transients die out. At t=50.0 seconds, the true Q_T is changed to 1.00 and the target enters its evasive maneuvering phase of the test. At t=80.0 seconds, the true Q_T is reset to 0.01, and the target resumes a benign trajectory. Data is taken over three periods of time. The overall RMS error is calculated by averaging from t=40.0 to 100.0 seconds, while the controller's ability to adapt to the benign trajectory is measured by averaging from t=40.0 to 50.0 seconds, and the controller's ability to adapt to the evasive maneuvering is measured by averaging from t=70.0 to 80.0 seconds. The speed and effectiveness of the adaptation is indicated from the the time history of the RMS values, which can be observed in Figures 5-16 and 5-17.

The results in Table 5-24 show that the adaptive controller does an excellent job of estimating the true Q_T representing the evasive maneuvering, where performance is most critical. Large errors here could result in the loss of lock on the target (i.e., the target moves off the detector array). Also, the adaptive controller does a good job of adapting to the Q_T appropriate for a return to a benign trajectory, although the performance is not as good. The data provided by the sensitivity analysis provides a baseline of the best performance the PI controller could attain if it had access to the true Q_T. The differences between this baseline and the MMAC baseline demonstrate the degradation in tracker performance due to using on-line adaptation and selecting only three discretized values of Q_T for the MMAC structure. The data provided by the robustness analysis shows what the results would have been if we did not use on-line adaptive estimation of Q_T.

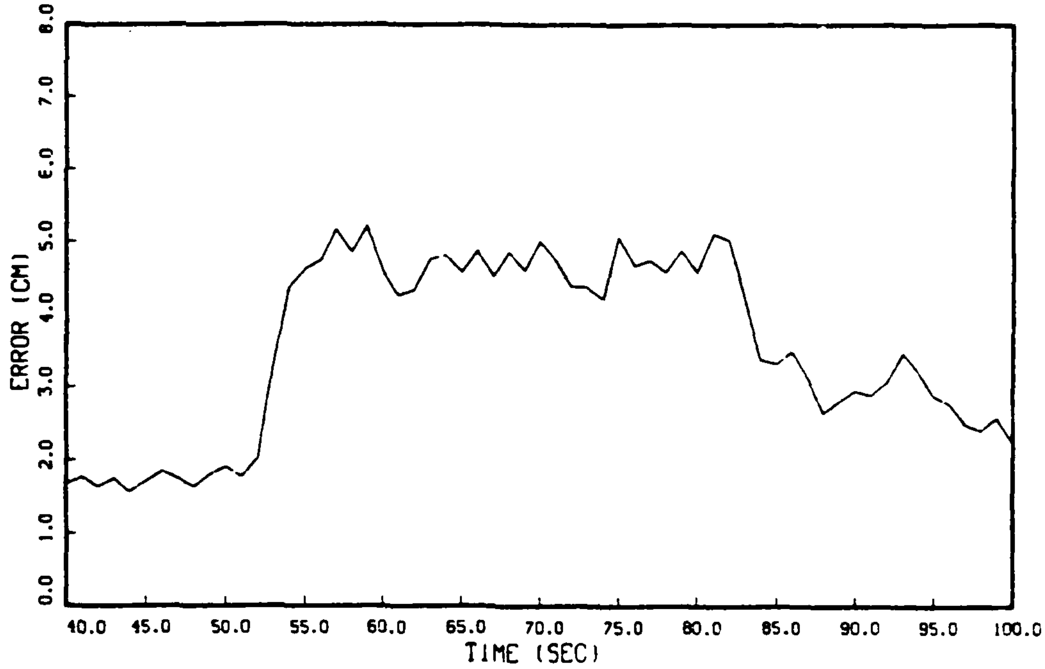


Figure 5-16. MMAC RMS Error Between the Target and the Beam

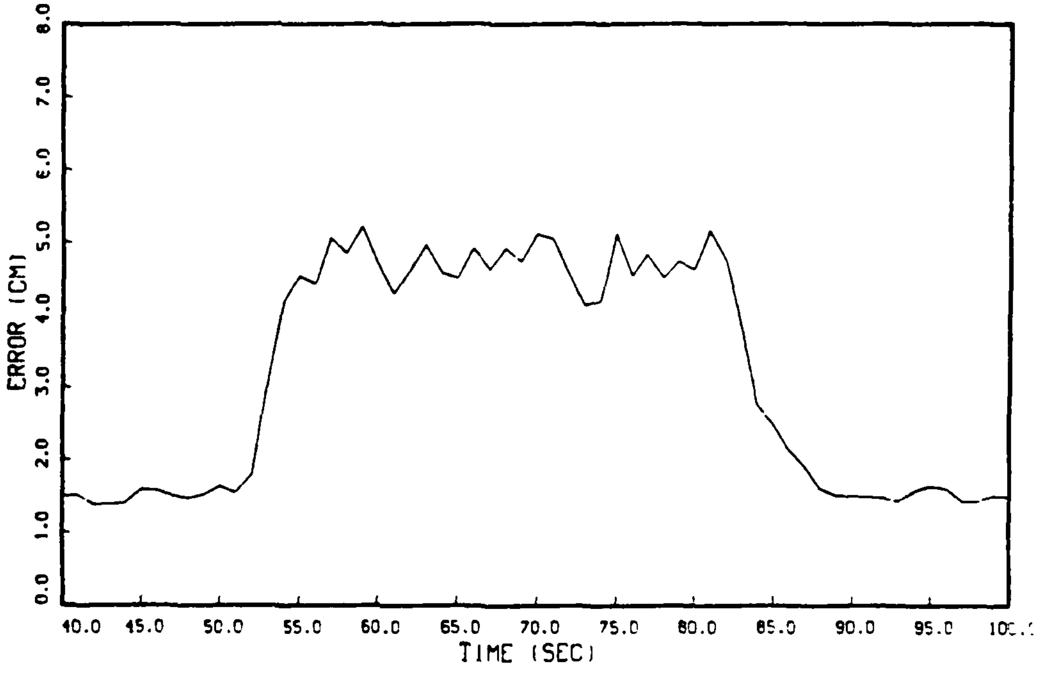


Figure 5-17. Baseline RMS Error

TABLE 5-24

Results of the MMAC Analysis

MMAC $Q_{T1}=0.01$ $Q_{T2}=0.07$ $Q_{T3}=1.00$	Baseline Average RMS_e	MMAC (actual) Average RMS_e	Sensitivity: Average RMS_e	Robustness: Average RMS_e		
				$Q_T=0.01$	$Q_T=0.1$	$Q_T=1.0$
Overall $40<t<100$	3.139	3.546	-	-	-	-
$Q_T=0.01$ $40<t<50$	1.634	1.736	1.461	-	2.034	3.340
$Q_T=1.00$ $70<t<80$	4.657	4.643	4.644	7.300	4.998	-

The performance of the MMAC algorithm is dependent upon significant differences between the residual characteristics in the "correct" and "mismodeled" elemental filters associated with the individual elemental controllers (see Figure 3-6). If the magnitude of the "correct" filter residual, $r_k(t_i)$, is much larger than the "mismodeled" filter residuals (relative to the internally computed residual variances, $A_k(t_i)$), then Equations (3-79) and (3-80) will cause the "correct" probability, $p_k(t_i)$, to increase until the "mismodeled" filter probabilities reach their lower bounds. If the "correct" and "mismodeled" residuals are indistinguishable, the filter will inappropriately select the "correct" model based on the smallest magnitude of variance residual, $A_k(t_i)$. If this happens, the controller probabilistic weighting will be erroneous, because the magnitudes of $A_k(t_i)$ are independent of both the residuals and the correctness of the different elemental controllers [1,13].

Although the actual residual values of the filters within the adaptive controller structure were not collected, the probabilistic weightings were recorded. The plots in Figure 5-18 are from the first two Monte Carlo runs and suggest the adaptive controller had little difficulty in distinguishing the "correct" elemental filter/controller from the "mismodeled" filter/controllers.

The second important criterion that should be used when evaluating an adaptive tracker is how quickly the tracker can detect and respond to changes in target dynamics noise strength. By definition, parameters are more slowly varying (if they vary at all) than the states [13]. Unfortunately, the true Q_T is controlled by the target, and although it varies slowly when it is compared to the target states, the target can quickly change Q_T with respect to the one second controller sample period used in this study. (This is unrealistically long, but has been chosen to be constant with the previous feasibility studies [8,20,27].) Therefore, it is important to see how fast the MMAC can adapt to changes in Q_T. The data is taken from a single simulation run that lasted 2,000 seconds, with the initial Q_T set to 0.01 cm^2/sec^5. At 65 seconds, and at every 100

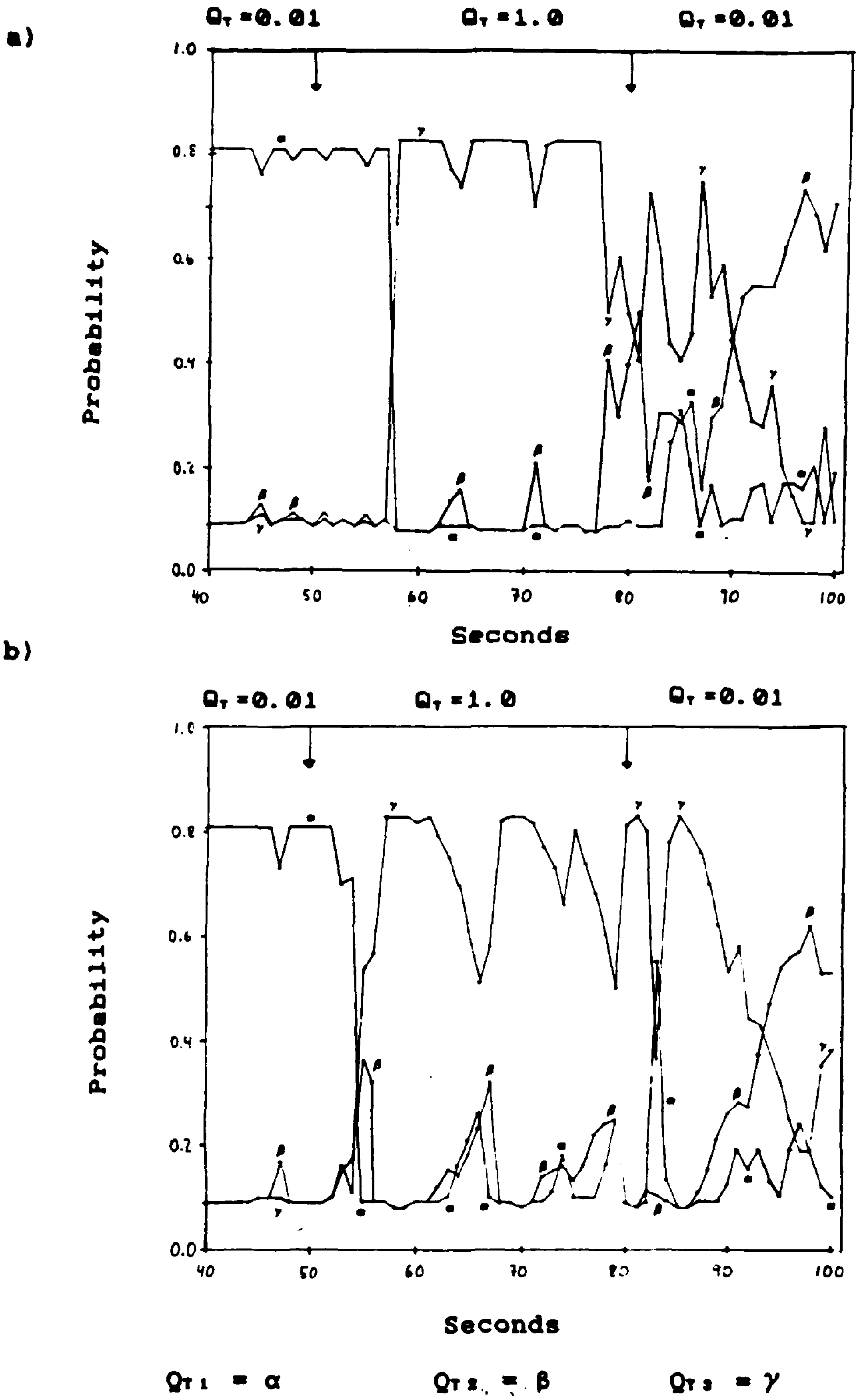


Figure 5-18. MMAC Probabilistic Weighting from a) Monte Carlo Run #1, b) Run #2

seconds thereafter, the true Q_T is increased to 1.0 cm^2/sec^5, and at 90 seconds and at every 100 seconds thereafter, the true Q_T resumes 0.01 cm^2/sec^2.

The results in Table 5-25 indicate that the adaptive controller takes approximately five seconds to detect a change in Q_T, and it is somewhat slow in adapting to the new Q_T. In addition, it takes the adaptive controller much longer to react to a decrease in Q_T than it takes to react to an increase in Q_T. When you go from benign to a harsh trajectory, $[r_k^2/A_k]$ of the benign filter k becomes very large while the corresponding term in the filter based on a harsh trajectory model is small, so the probabilities calculated in Equation (3-79) causes the probability weighting to shift to the latter filter. However, when the actual trajectory switches back to a benign trajectory, the value of $[r_k^2/A_k]$ in the mismatched filter is not extraordinarily large compared to that of the matched filter, so the probability weighting shift is not as rapid [11,15]. This effect can be observed in Figures 5-16 and 5-17. A possible solution to quicken the adaptive controller's reaction to changes in Q_T is to increase both the controller and Kalman filter sample rates. The other problem of the filter/controller slowly adapting to a decreasing Q_T is inherent to the design, and has been previously noted in other studies of multiple model adaptive algorithms [11,15].

TABLE 5-25

Adaptive Filter Time Response to Increasing and Decreasing Q_T

A. Q_T increases from 0.01 to 1.00.				
Sample Statistics	Time responses in seconds t_D	 t_R	 t_S	 t_T
$\hat{x}$ (mean)	5.30	8.45	6.90	13.75
σ (std dev)	2.27	8.11	6.70	7.04

B. Q_T decreases from 1.00 to 0.01.				
Sample Statistics	Time response in seconds t_D	 t_R*	 t_S*	 t_T*
$\hat{x}$ (mean)	5.05	31.5	20.5	36.5
σ (std dev)	2.06	15.3	25.8	15.2

Note: The sample population is n=20, except when denoted by *, then n=19 (i.e., the 2000 second run from which this data is tabulated, terminated before the information on t_R, t_S and t_T was available). The sampled data is in Appendix D.

Definitions:

t_D - the delay time is the time it takes the adaptive controller to react first to a new Q_T.

t_S - the settling time is the time required by the "correct" elemental filter to reach a $p_k > 0.5$, and stay above it thereafter. It is measured from the time the adaptive controller first reacts to the new Q_T.

t_R - the rise time is the time it takes the "correct" elemental filter to surpass a p_k of 0.75. Note: This is within 10 percent of the upper bound. It is measured from the time the adaptive controller first reacts to the new Q_T.

t_T - the transition time is the time it takes the "correct" elemental filter to detect and react to a change in Q_T. $t_T = t_D + t_R$

5.6 Conclusions

Five different Monte Carlo analyses were accomplished. The first four evaluated the non-adaptive PI controller while the fifth examined the MMAC. It was found that the PI controller performed best with a 10:1 ratio between the cost weighting elements X_{11} and X_{55}, and that the estimator/controller suffered no detrimental effects when the Meer filter's depth was reduced from three to one. The performance was found to be the most sensitive to equal changes in Q_T (the target dynamics driving noise strength) and R_T (the measurement corruption noise variance) in both the truth model and the filters, while τ_B (the beam time constant) and Q_T were the parameters that yielded the most significant robustness difficulties. The severe lack of robustness characteristics with respect to τ_B appears to be a stability problem which will be discussed in Chapter 6. The target driving noise strength was selected for on-line adaptive estimation, and the adaptive controller performed well.

VI. PI CONTROLLER STABILITY ANALYSIS

The purpose of this chapter is to evaluate how the Meer and Snyder-Fishman filters affect the stability of the PI controller. The motivation behind this analysis is to isolate the stability robustness problem mentioned in Section 5.4.1, and if possible, identify the reasons the PI controller lacks robustness with respect to the beam time constant. The source of the instability is believed to lie within one of three areas: with the mismodeling of the filter's state transition matrix ($\underline{\Phi}$), within the inherent structure of the Meer filter, or with the varying sample period which is the product of the Poisson process for measurement event times.

The chapter will start by evaluating the stability of the PI full-state feedback controller structure that uses the steady-state gains found in Table 3-1. The generic PI controller structure is time-invariant and linear; therefore, showing that the poles of the system lie inside the unit circle (z-domain) will prove that the closed loop system with a full-state feedback controller structure is asymptotically stable in the global sense (a form of zero-input stability). Then, the Meer filter is cascaded with the controller transfer function, and the stability is evaluated as functions of the Meer filter sample rate and the filter gain. Because the sample rate of the Meer filter is not fixed, the system is no longer time-invariant, such a simple stability evaluation is not possible. A rigorous proof of global stability for stochastic control systems that use either the Meer or Snyder-Fishman filters in conjunction with Kalman filters does not exist. Instead, an ad hoc method is employed where the effects of different sample rates are evaluated as being time-invariant, and the zero-input stability analysis is used to evaluate the pole movement within the stochastic controller. Next, a stability demonstration is used to verify that an analogous Kalman filter for this application is stable and inferences are drawn to the possible sources of instability within Snyder-Fishman filter. Last, we need to analyze the effects the Meer filter structure has on the Snyder-Fishman filter equations, and thus the effects the Meer filter structure has on the PI controller. The emphasis in this chapter is not to provide the rigorous conditions for global stability, but to analyze the possible causes of the controller's instability. In other words, we are looking for answers to the findings in Section 5.4.1.

6.1 Derivation of the PI Controller Transfer Functions

In a linear, time-invariant system, we find that the stability is determined by the location of the poles in the system transfer function. A system is said to be asymptotically stable if and only if all the poles lie within the stable region. For the continuous system, the stable region is defined as the left-hand plane of the Laplace s-domain plot. Analogously for the discrete system model, the stable region is defined as the area inside of the unit circle of a z-domain plot. In other words, a discrete, linear, time-invariant system model, such as the closed loop system with deterministic full-state feedback PI controller, is asymptotically stable if and only if all the poles of the discrete closed loop transfer function lie within the unit circle (i.e., have a magnitude less than one).

The first step is to analyze the stability of the deterministic PI controller loop shown in Figure 6-1. The transfer function is formed by solving the pseudo-integrator equation,

$$q(z) = \frac{z}{z - 1} [y_c(z) - y_r(z)] \tag{6-1}$$

feedback control equation,

$$u(z) = -G_{c1} y_c(z) - G_{c2} q(z) + y_r(z) \tag{6-2}$$

and the plant equation in terms of $y_c(z)/y_r(z)$. The plant equation is found by taking the impulse transform of a zero-order hold attached to the plant dynamics model,

$$Y_c(s) = G_{ZOH}(s) P(s) U^*(s) \tag{6-3}$$

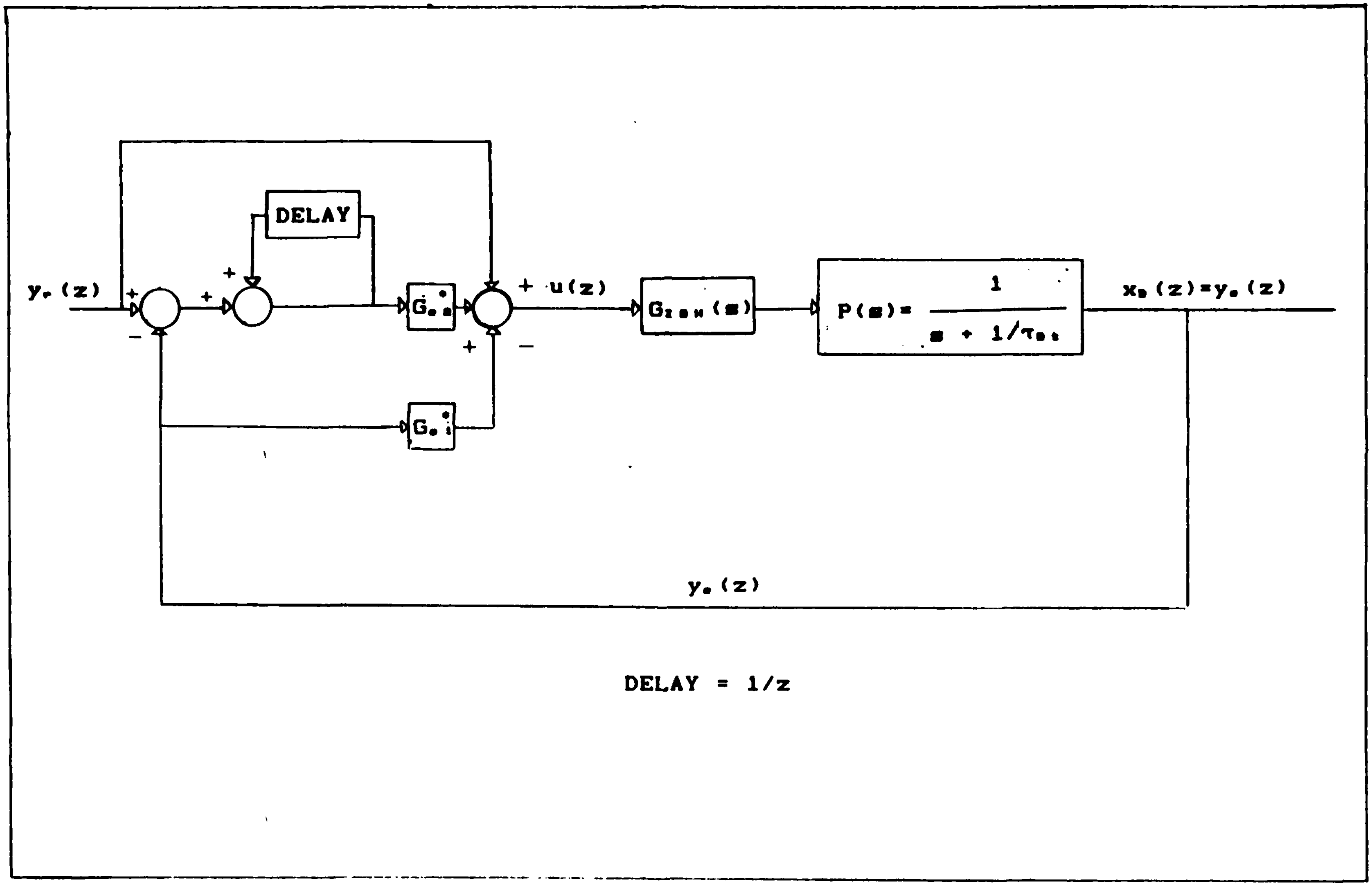


Figure 6-1. Deterministic PI Controller

where $G_{ZOH}(s)$ is the transfer function (in the Laplace domain) representing a zero-order hold and $P(s)$ is the plant transfer function

$$G_{ZOH}(s) = [1 - \exp(\Delta ts)]/s \qquad (6\text{-}4)$$

$$P(s) = \frac{1}{s - 1/\tau_{Bt}} \qquad (6\text{-}5)$$

and then transforming it into the z-domain,

$$y_c(z) = G_{ZOH}P(z)u(z) = \frac{\tau_{Bt}[1-\exp(-\Delta t/\tau_{Bt})]}{z - \exp(-\Delta t/\tau_{Bt})} \cdot u(z) \qquad (6\text{-}6)$$

where

$$G_{ZOH}P(z) = \mathcal{Z}\{[1 - \exp(-st)/s][1/(s + 1/\tau_{Bt})]\} \qquad (6\text{-}7)$$

The deterministic PI controller transfer function is

$$\frac{y_c(z)}{y_r(z)} = \frac{\tau_{Bt}[1-\exp(-\Delta t/\tau_{Bt})][z(G_{c2}+1)-1]}{[z-\exp(-\Delta t/\tau_{Bt})][z-1] + \tau_{Bt}[1-\exp(-\Delta t/\tau_{Bt})][z(G_{c1}+G_{c2})-G_{c1}]} \qquad (6\text{-}8)$$

Assuming we want to know the location of the poles for the nominal conditions (Δt=1.0 seconds, τ_{Bt}=20.0 seconds, G_{c1}^*=1.2422, and G_{c2}^*=0.27436), Equation (6-8) becomes

$$\frac{y_c(z)}{y_r(z)} = \frac{.7654[z - .7847]}{z^2 - .4719z + .2605} \qquad (6\text{-}9)$$

The poles of the closed loop system with deterministic PI full-state feedback controller transfer function are located at $p_1=0.7982$ and $p_2=-0.3263$ which are well inside the unit circle, and therefore the deterministic controller is asymptotically stable.

The next step is to derive the stochastic PI controller loop transfer function with Meer filter in the loop. For simplicity, the Kalman filter, which is used to estimate the reference variable, will be removed from the overall controller structure, and its stability will be verified in Section 6.3. This can be done because the Kalman filter transfer function is not included in the feedback loop of the regulator associated with the tracker. Therefore, instead of analyzing the stability of the PI tracker, we will analyze the stochastic PI regulator, and this simplification will not effect the pole/zero movement of interest [14].

The stochastic controller loop transfer function is derived in the same manner as was the deterministic controller transfer function, except that an approximation to a Meer/Snyder-Fishman filter is included in the feedback loop (see Figure 6-2). Note that the approximation being made is that a fixed sample period is being used rather than taking measurements at times dictated by a Poisson process.

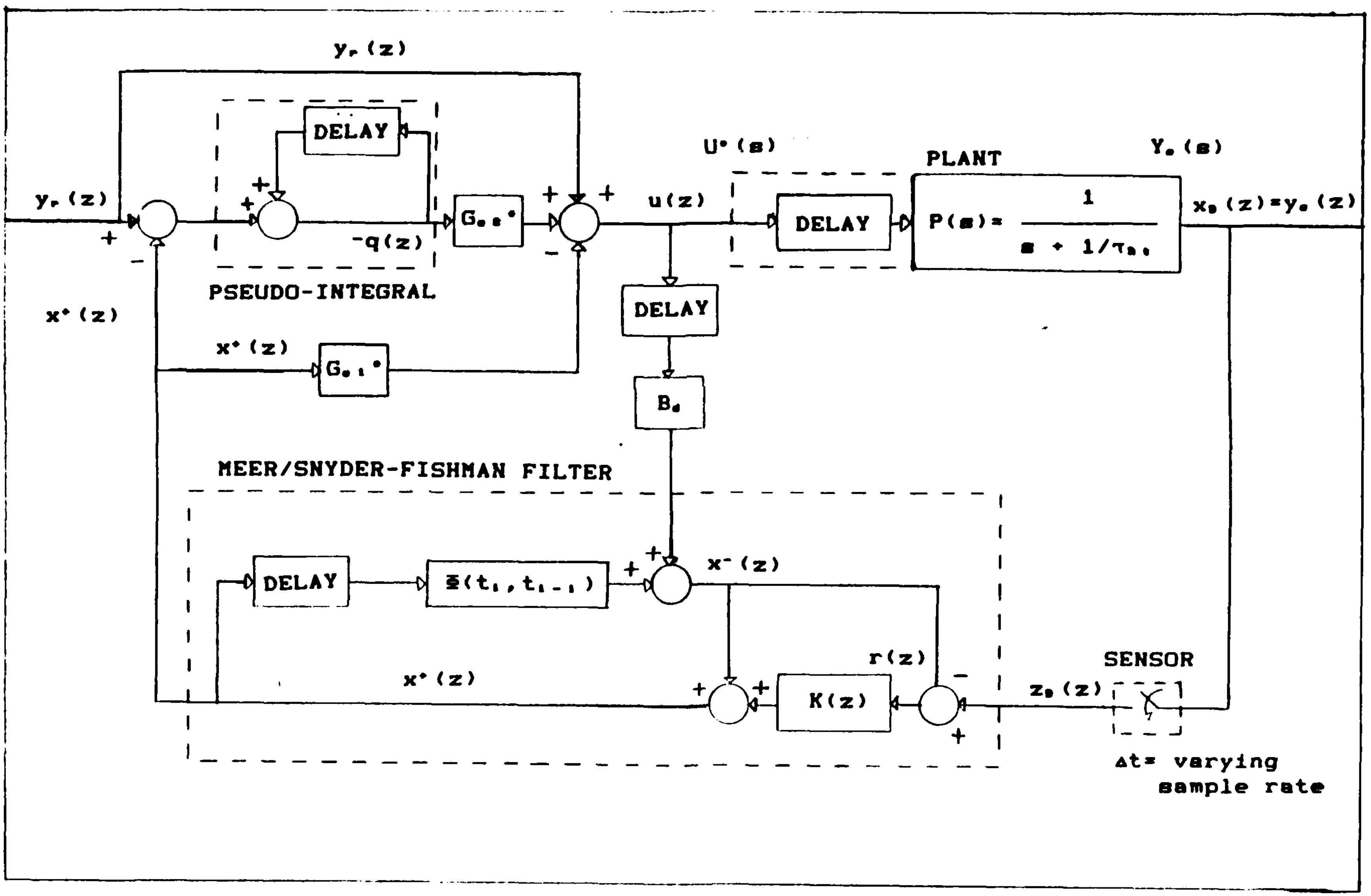


Figure 6-2. Stochastic PI Controller

Thus, the approximation results in a Kalman filter structure as shown in Figure 6-2. Such an approximation is made only in an attempt to discern basic stability robustness characteristics of the loop, possibly as a function of a chosen fixed sample period, to gain insights into the actual behavior of a Meer or Snyder-Fishman filter in the loop. Therefore, the transfer function is formed by solving the equations for the approximated Meer(or Snyder-Fishman) filter,

$$\hat{x}^{+}(z) = \frac{zK(z)y_c(z) + B_d u(z)[1-K(z)]}{z - \Phi_B[1-K(z)]} \tag{6-10}$$

the pseudo-integrator,

$$q(z) = \frac{z}{z-1}[\hat{x}^{+}(z) - y_r(z)] \tag{6-11}$$

the feedback control,

$$u(z) = -G_{c1}\hat{x}^{+}(z) - G_{c2}q(z) + y_r(z) \tag{6-12}$$

and the plant model, Equation (6-6), in terms of $y_c(z)/y_r(z)$. The term B_d is the discrete-time control input matrix and can be calculated from Equation (3-60). The stochastic PI controller closed loop transfer function is:

$$\frac{y_c(z)}{y_r(z)} = \frac{[\tau_{Bt}(1-\exp(-\Delta t/\tau_{Bt}))][z(G_{c2}+1)-1][z-\Phi_B(1-K(z))]}{z^3 + z_2[A+D-C] + z[B-AC-E] - BC} \tag{6-13}$$

where

$A = [1-K(z)][B_d(G_{c1}+G_{c2})-\Phi_B]-1$

$B = [1-K(z)][\Phi_B - B_d G_{c1}]$

$C = \exp(-\Delta t/\tau_{Bt})$

$D = \tau_{Bt}K(z)[G_{c1}+G_{c2}][1-\exp(-\Delta t/\tau_{Bt})]$

$E = \tau_{Bt}K(z)[G_{c1}][1-\exp(-\Delta t/\tau_{Bt})]$

6.2 The Zero-Input Stability Analysis

The purpose of the zero-input stability analysis is to see what conditions will drive the stochastic PI controller unstable. Two variables of the Meer filter will be analyzed. First, the stochastic filter will be set to a steady-state condition with a "fixed" sample period of one second. This is to simulate a rather well behaved Poisson distributed sample process in which the time "between" events has closely approximated the mean signal arrival rate of one second between events. The reasoning is that we need to define some sort of baseline to be able the calculate a filter gain. Then, we want to vary the time to the "next" signal-induced event (for the varying sample period analysis, we will discount the effects that noise-induced events have on the simplified Meer filter gain) and observe the effects that different sample periods have on the controller's pole locations. In other words, we want to see in an approximate manner how the Poisson distributed sample rate will affect the controller's stability. Theoretically, as the time to the "next" event is increased, one or more poles will migrate beyond the limits of the unit circle, and the controller is forced to operate in the unstable region for the duration of that sample period [28].

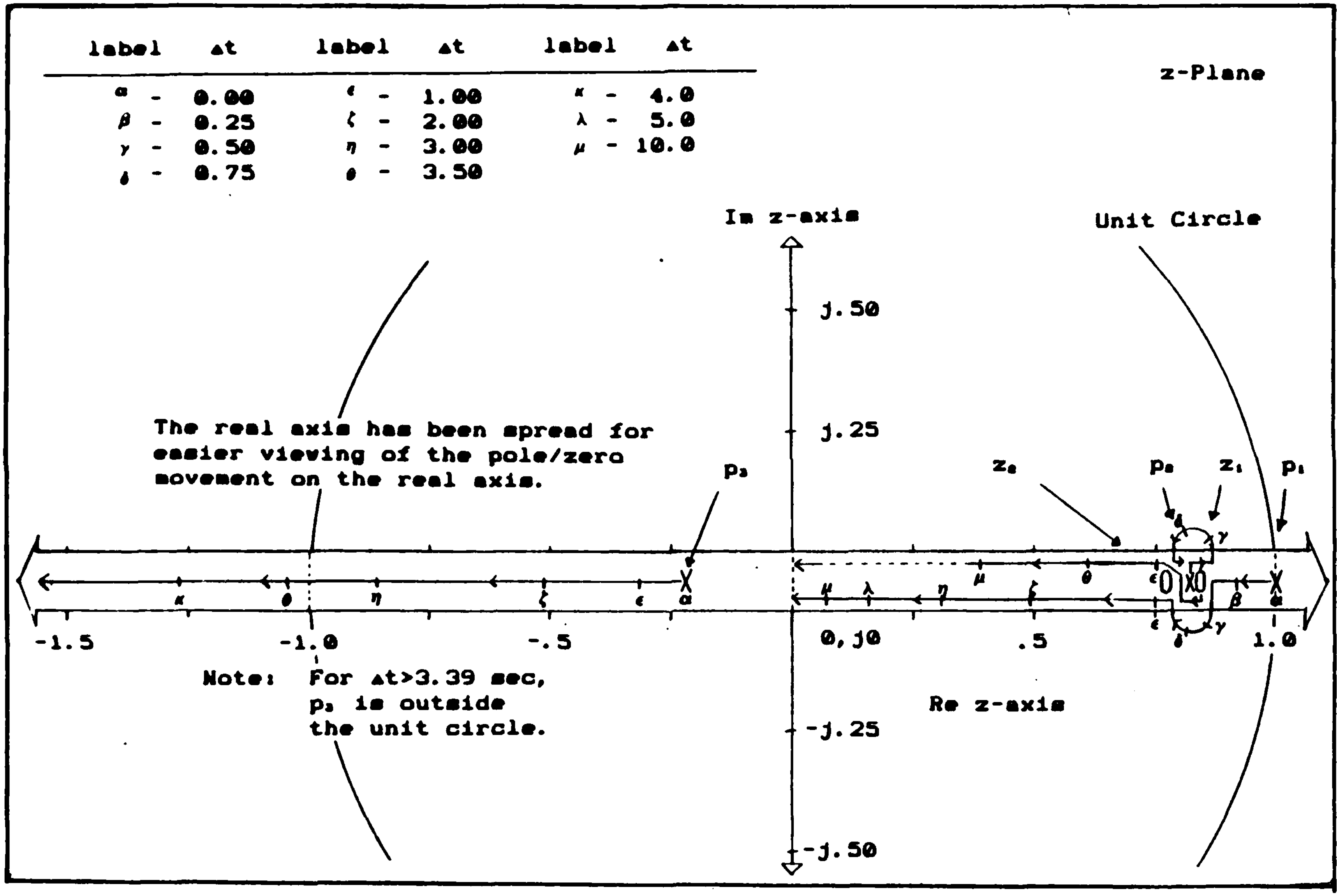
Second, the controller's stability will be evaluated with respect to changes in the Meer filter gain, $K(t_i)$. Unlike Kalman or Snyder-Fishman filters gains, the simplified Meer filter gain (assuming a filter depth of one) is a function of the filter's residuals (see Section 2.5); therefore, the Meer filter is susceptible to large swings in filter gain due to noise-induced events. As it will be seen, this can have a significant effect on the stability of a controller.

The effects that a varying sample rate and changing filter gains have on controller stability can be evaluated by solving for the stochastic PI controller closed loop transfer function, Equation (6-13), for a range of different sample periods and filter gains. This is done by solving the state transition matrix, Φ_B, for the "next" sample period. The control input matrix, B_d, is calculated from Equation (3-60) for the controller sample rate of one Hertz (i.e., a sample period of $\Delta t=1$ second). For the evaluation of the varying sample rate, the filter gain is calculated from the Snyder-Fishman covariance equations (see Equations (2-11) and (2-14), where $P(t_{i-1}^{+})$ is calculated for the steady-state condition defined by a "fixed" sample period of one second and $P(t_i^{-})$ is calculated for the duration of the "next" sample period. The filter gain is calculated from $P(t_i^{-})$ using the Snyder-Fishman filter gain equation, Equation (2-15). Changes in the filter gain are evaluated by solving the stochastic PI controller transfer function for a range of different filter gains at different sample periods.

6.2.1 The Effects of a Varying the Sample Period

It was found that the PI feedback controller (using Meer filter to estimate the beam states, and operating under nominal conditions) is driven unstable for sample periods that are longer than 3.368 seconds. This can be observed from the stochastic controller root locus plotted in Figure 6-3. Although sample periods longer than 3.368 seconds occurs only 3.45 percent of the time (as calculated from Equation (4-17), and assuming $\hat{\lambda}_s=1.0$ signal-induced event per second), it means that the filter is operating in the unstable region for 15.05 percent of the time (as calculated by integrating over the Poisson density function from $t_i=3.368$ to $t_{i+1}=\infty$).

The effect of mismodeling τ_s was also studied, and the results showed (see Appendix C) that the controller becomes unstable at shorter sample periods when the true τ_s is decreased from the Meer filter-assumed value.



label	Δt	label	Δt	label	Δt
α -	0.00	ϵ -	1.00	κ -	4.0
β -	0.25	ζ -	2.00	λ -	5.0
γ -	0.50	η -	3.00	μ -	10.0
δ -	0.75	θ -	3.50		

The real axis has been spread for easier viewing of the pole/zero movement on the real axis.

Note: For Δt>3.39 sec, p_3 is outside the unit circle.

Figure 6-3. Root Locus of the Stochastic PI Controller; Varying the Meer Filter Sample Rate

6.2.2 The Effects of a Varying Stochastic Filter Gain

The analysis was done with the sample period set to 1.0 and 3.5 seconds, and the gain was varied from 0.0 to 1.0. That is, the filter gain in this analysis was calculated across its entire range, and the filter gain was not based on a specific filter beam dynamic driving noise strength, g^2, nor a measurement noise variance, R. The results for a sample period of one second demonstrated that the controller is stable for the entire range of the filter gain (see Table 6-1). This is expected, for we effectively evaluated the controller as if it were using a Kalman filter instead of a Meer filter. The controller is third order, but the filter employs a pole-zero cancellation (during ideally modeled conditions and not true for robustness), and the effects of the filter within the controller structure is never felt. This is not the case when the controller's sample period is 3.5 seconds long. The controller is only stable for values of $K(t_i)$ that are less than 0.27. This is a relatively small filter gain, which indicates that the filter assumes that its internal dynamics model is relatively accurate during that sample period. The results suggest that the stability of the controller is highly sensitive to the magnitude of the filter dynamic driving noise strength, g^2, which is used to calculate the filter gain. The higher the g^2, the more susceptible the stochastic controller is to being driven unstable by long sample periods (particularly if the filter-assumed model does not match the real world exactly, which is always the case).

TABLE 6-1

Stochastic Filter Pole/Zero Placement as a Function of $K(t_i)$

Pole locations (nominal condition)

Filter Gain	$\Delta t = 1.0$ seconds			$\Delta t = 3.5$ seconds		
	p_1	p_2	p_3	p_1	p_2	p_3
1.00	.7982	-.3263	.0000	.8183	-3.8484	.0000
0.90	.7982	-.3263	.0951	.8183	-3.4365	.0111
0.75	.7982	-.3263	.2378	.8183	-2.8248	.0338
0.50	.7982	-.3263	.4756	.8183	-1.8375	.1032
0.25	.7982	-.3263	.7134	.8183	-.9711	.2949
0.10	.7982	-.3263	.8561	.8183	-.6075	.5657
0.00	.7982	-.3263	.9512	.8183	-.4560	.8395

Zero locations (Note: the z_2/p_3 cancellation at $\Delta t=1.0$)

Filter Gain	$\Delta t = 1.0$ seconds		$\Delta t = 3.5$ seconds	
	z_1	z_2	z_1	z_2
1.00	.7847	.0000	.7847	.0000
0.90	.7847	.0951	.7847	.0839
0.75	.7847	.2378	.7847	.2099
0.50	.7847	.4756	.7847	.4197
0.25	.7847	.7134	.7847	.6296
0.10	.7847	.8561	.7847	.7555
0.00	.7847	.9512	.7847	.8395

6.3 The Stability of the Kalman and Snyder-Fishman Filter Equations

This section applies the second Lyapunov stability theory to the homogeneous portion of the Kalman filter to show that it meets the zero-input stability criteria. Then, the same stability criteria will be applied approximately to the Snyder-Fishman filter and although the stochastic particle beam model is technically stable, an observability problem is apparent. The stability criteria to be used in this section presents a set of non-restrictive conditions for which the Kalman filter algorithm is asymptotically stable in the global sense. Asymptotic global stability is the strongest form of stability, which means that, given any initial condition within a defined region, its state plane trajectory will always converge to an equilibrium point [3]. If that system is linear and asymptotically stable, there will be only one equilibrium point. This is an important concept, because for linear systems, the sufficient conditions of asymptotic stability are indistinguishable from conditions for bounded input-bounded output (BIBO) stability [12]. When BIBO stability criteria are applied to the filter, it means that, for a bounded input into the filter, the output (state estimate) will be bounded. If the region of the initial conditions that meet the stability criteria is expanded to include the entire state space, then that system is said to be stable in the global sense. If the system model upon which the Kalman filter is based is stochastically controllable with respect to the points of entry of the dynamics driving noise $w_d(\cdot,\cdot)$, and stochastically observable from the points of exit of the measurements from the system model, then the filter is asymptotically stable in the global sense [12,14].

Therefore, a Kalman filter as defined by the target model in Section 3.1, is said to be stochastically controllable if there exist positive numbers α and β, $0<\alpha<\beta<\infty$, and a positive integer N, such that, for all $i \geq N$,

$$\alpha \underline{I} \leq \sum_{j=i-N+1}^{i} \underline{\Phi}(t_i, t_j) \underline{G}_d(t_{j-1}) \underline{Q}_d(t_{j-1}) \underline{G}_d^T(t_{j-1}) \underline{\Phi}^T(t_i, t_j) \leq \beta \underline{I} \tag{6-14}$$

and is said to be stochastically observable if there exist positive numbers α and β, $0<\alpha<\beta<\infty$, and a positive integer N, such that, for all $i \geq N$

$$\alpha \underline{I} \leq \sum_{j=i-N+1}^{i} \underline{\Phi}^T(t_j, t_i) \underline{H}^T(t_j) \underline{R}^{-1}(t_j) \underline{H}(t_j) \underline{\Phi}(t_j, t_i) \leq \beta \underline{I} \tag{6-15}$$

Because Equations (6-14) and (6-15) are bounded both above and below, this is a stronger condition than just using the positive definiteness of the controllability and observability gramians. If the target model upon which the Kalman filter is based is stochastically controllable and observable, then the filter is asymptotically globally stable. For time invariant system models, we would only need detectable and stabilizable to ensure stability of the filter [10].

When Equation (6-14) is applied to the target model, it becomes:

$$\alpha\underline{I} \le \sum_{j=i-N+1}^{i} \begin{bmatrix} A^2Q_d & ABQ_d & ACQ_d \\ ABQ_d & B^2Q_d & BCQ_d \\ ACQ_d & BCQ_d & C^2Q_d \end{bmatrix} \le \beta\underline{I} \qquad (6\text{-}16)$$

where

$A = \tau_B[\Delta t - B]$

$B = \tau_B[1 - C]$

$C = \exp(-\Delta t/\tau_B)$

Q_d = the discrete target dynamic driving noise strength

$\Delta t = t_i - t_j$ (Note: $t_i - t_j$ is not the sample period)

Because Q_d does not equal zero, and the matrix in Equation (6-16) is a positive definite for all forward running time, there will always be positive α and β such that $0<\alpha<\beta<\infty$; therefore, the target model is fully controllable.

When Equation (6-15) is applied to the target model, it becomes

$$\alpha\underline{I} \le \sum_{j=i-N+1}^{i} \begin{bmatrix} 1/R_T & -\Delta t/R_T & C/R_T \\ -\Delta t/R_T & \Delta t^2/R_T & -\Delta tC/R_T \\ C/R_T & -\Delta tC/R_T & C^2/R_T \end{bmatrix} \le \beta\underline{I} \qquad (6\text{-}17)$$

where

$C = \exp(-\Delta t/\tau_B)$

R_T = target measurement variance

$\Delta t = t_j - t_i$ (Note: backwards time)

Because R does not equal zero or infinity, and the matrix in Equation (6-17) is positive definite for all backwards running time, there will always be positive α and β such that $0<\alpha<\beta<\infty$; therefore, the target model is fully observable.

Since the target model upon which the Kalman filter is based is fully controllable and fully observable, the filter is uniformly asymptotically stable in the global sense. Uniformly stable implies that the system in stable for all time, regardless of what absolute time it is. The same observations can be made by referring to the target shaping filter in Figure 5-3. It is obvious that model is controllable from the entry point of the white noise, and observable from the point of extraction of the target position state.

When the same zero-input stability criteria is applied to the homogeneous part of a Kalman filter that approximates the Snyder-Fishman filter, the Poisson distributed sample rate does not violate any condition which define controllability and observability. Note that the Kalman and Snyder-Fishman filter structures are almost identical (see Section 2.2).

When stochastic controllability Equation (6-14) is applied to the particle beam model, the resulting expression is

$$\alpha \leq \sum_{j=i-N+1}^{i} Q_d \exp[-2(t_i - t_j)/\tau_B] \leq \beta \qquad (6\text{-}18)$$

As long as Q_d does not equal zero, and none of the sample periods are allowed to approach ∞ (theoretically possible with the Poisson process), then there shall be positive numbers α and β where $0<\alpha<\beta<\infty$, and the model used for the basis of the filter design will be controllable.

Stochastic observability is also met. This can be shown by applying Equation (6-15) to the stochastic particle beam model to attain

$$\alpha \le \sum_{j=i-N+1}^{i} \exp[2(t_i - t_j)/\tau_B]/R \le \beta \qquad (6\text{-}19)$$

But, a fundamental observability problem could arise for the longer sample periods, because as the length of the sample period grows, the upper bound must grow at an exponential rate. Regardless, the Snyder-Fishman filter equations are technically asymptotically stable assuming we define the range of the Poisson process intervals between measurements as not to include infinity.

6.4 Stability Insights on the Meer Filter Structure

This section will re-evaluate the method in which the Meer filter gain is calculated for a filter depth of one, with particular emphasis on gaining an understanding of the severe robustness problems associated with mismodeling of the beam time constant, τ_B. As it was shown in Section 2.5, the Meer filter gain, $K_M(t_i)$, is related to the Snyder-Fishman filter gain, $K_{SF}(t_i)$, by the following expression for a depth of one:

$$K_M(t_i) = p_1(t_i)K_{SF}(t_i) \qquad (6\text{-}20)$$

where $p_1(t_i)$ is the probability that the event at t_i was signal induced (see Equation 2-27). As the filter residual, $r(t_i)$, increases, $p_1(t_i)$ decreases. If the mismodeling of τ_B is severe enough, the residual will be large enough so that $p_1(t_i)$ will approach zero and drive the Meer filter gain to zero regardless of the Snyder-Fishman filter gain. This suggests the possibility that the Meer filter structure can severely inhibit the robustness of the controller, and that the apparent instability might be in part caused by or at least aggravated by a robustness problem.

The concept is that the combined effects of long sample periods and mismodeling τ_B will cause large residuals. Ideally, a large residual indicates that the event is most likely noise-induced, and the event receives a probabilistic weight as such. But, if the true center of the beam is offset from the filter's estimate of the center, then the Meer filter will begin to weight a large number of signal-induced events as being noise-induced (see Figure 6-4). It appears that the filter could counteract this effect. As more events are labeled noise-induced, the time between signal events will appear to increase, and the filter covariance will increase. This will cause the elemental Snyder-Fishman filter to rely more on the incoming

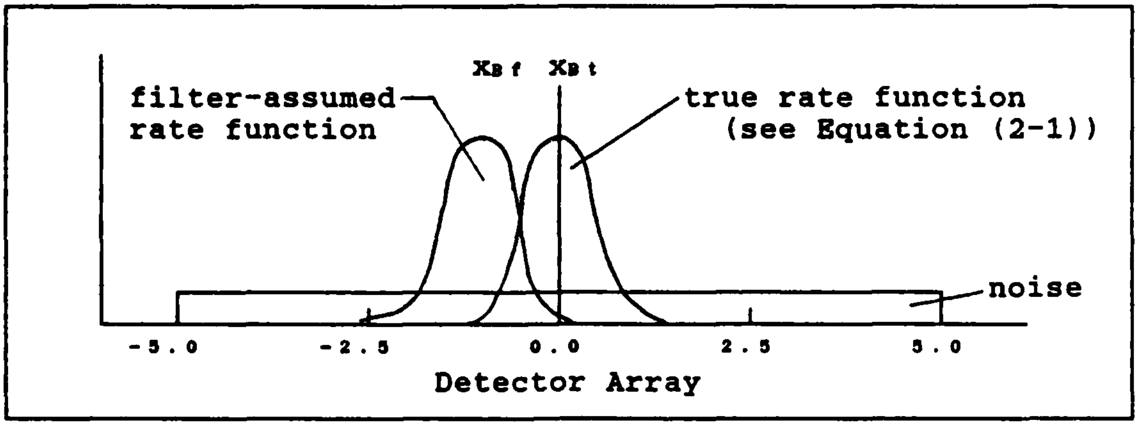

X_{Bf}
X_{Bt}
filter-assumed rate function
true rate function (see Equation (2-1))
noise
-5.0
-2.5
0.0
2.5
5.0
Detector Array

Figure 6-4. True Beam Offset from Filter Beam:
$R_t = R_f = 0.5$ cm, $r(t_i) = 1.0$ cm

measurements. This does happen, but it is to no avail because the elemental Snyder-Fishman filter gain is multiplied with the probabilistic weight that it was a signal-induced event, $p_i(t_i)$. This product forms the Meer filter gain. As the filter beam center diverges from the true beam center, $p_i(t_i)$ approaches zero and forces the Meer filter gain to zero. Meanwhile, no new measurements are incorporated, and the filter never recovers.

This led to the hypothesis that some of the robustness can be recovered by computing the filter R as a function of some time-weighted average of the previous N residuals. By increasing the filter R when the most recent residuals are consistently larger than originally anticipated, the Meer filter is told to look for the signal-induced events across a larger section of the detector (see Figure (6-5)). The

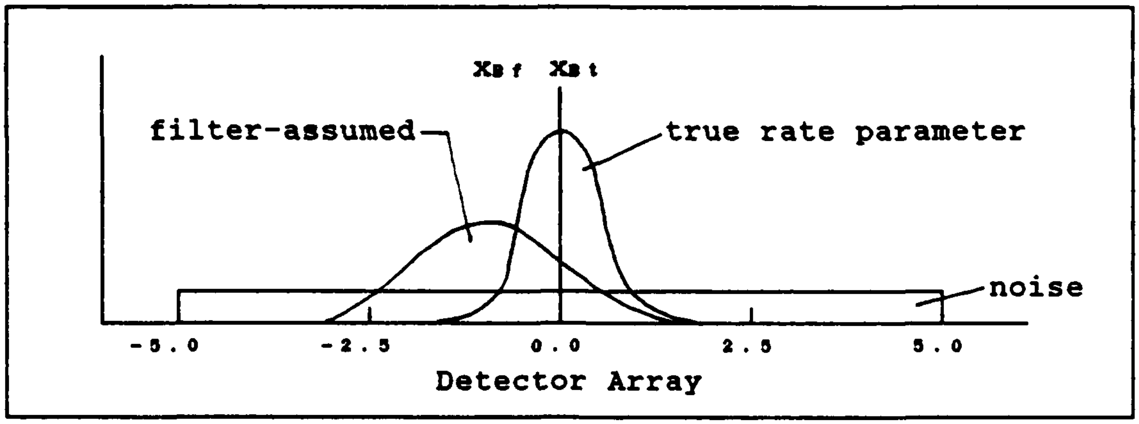


Figure 6-5. True Beam Offset from Filter Beam: R_t=0.5 cm, R_f=1.0 cm, $r(t_i)$= 1.0 cm

advantage of this concept is that most of the signal-events can be recovered, and the filter should have much better robustness with respect to the mismodeling of the beam time constant. The robustness analysis found that the controller is rather insensitive to increases of the filter R (see Section 5.4.3), and although the larger filter R will cause filter to weight more noise-induced events as signal-induced events, this should have no significant effect on the RMS tracking error. This conclusion is drawn from the results found during the sensitivity and robustness analyses conducted on the beam dispersion parameter (see Sections 5.3.3 and 5.4.3).

This concept was simulated through a measurement model that received 1000 events. The true beam had a dispersion parameter R_t of 0.5 cm, and was located at the center of the detector. The measurement model was offset from the true beam center by a prespecified residual. The simulation included noise-induced events as specified by the SNR, and

TABLE 6-2

The Filter's Probabilistic Weight that an Event Was Signal-Induced as a Function of R_f and the Filter Residual

(Simulated Results: Sample Population = 1000 events)

1. R_f=0.5	Filter residual:			r = $z_B - \hat{x}_B$	(cm)		
Probability	0.0	0.5	1.0	1.5	2.0	2.5	3.0
$\hat{p}_1(t_1)$	.973	.965	.938	.851	.655	.454	.255
$1.0 \le p_1 < .99$	.889	.754	.506	.371	.124	.019	.007
$.99 \le p_1 < .95$	.081	.197	.291	.185	.300	.137	.021
$.95 \le p_1 < .90$	.003	.024	.080	.097	.081	.103	.038
$.90 \le p_1 < .75$	.004	.005	.065	.146	.039	.145	.068
$.75 \le p_1 < .50$	.003	.001	.024	.102	.117	.083	.126
$.50 \le p_1 < .30$	.000	.002	.008	.034	.085	.016	.082
$.30 \le p_1 \le .15$	.001	.005	.005	.031	.067	.053	.077
$.15 \le p_1 < .05$	.004	.005	.001	.017	.081	.111	.053
$.05 \le p_1 \le .00$	.015	.016	.020	.017	.106	.333	.528

2. R_f=1.0	Filter residual:			r = $z_B - \hat{x}_B$	(cm)		
Probability	0.0	0.5	1.0	1.5	2.0	2.5	3.0
$\hat{p}_1(t_1)$	.986	.986	.975	.971	.935	.844	.667
$1.0 \le p_1 < .99$	.970	.958	.849	.558	.434	.201	.047
$.99 \le p_1 < .95$	.011	.021	.122	.374	.267	.295	.265
$.95 \le p_1 < .90$	.002	.003	.005	.036	.135	.092	.125
$.90 \le p_1 < .75$	.004	.003	.002	.013	.116	.201	.065
$.75 \le p_1 < .50$	.002	.006	.003	.006	.025	.117	.178
$.50 \le p_1 < .30$	.001	.001	.001	.000	.007	.048	.140
$.30 \le p_1 < .15$	.004	.002	.003	.001	.002	.022	.076
$.15 \le p_1 < .05$	.003	.002	.001	.000	.002	.011	.047
$.05 \le p_1 < .00$	.003	.004	.014	.012	.012	.013	.054

3. R_f=2.0	Filter residual:			r = $z_B - \hat{x}_B$	(cm)		
Probability	0.0	0.5	1.0	1.5	2.0	2.5	3.0
$\hat{p}_1(t_1)$	.997	.996	.993	.991	.985	.980	.963
$1.0 \le p_1 < .99$	.976	.980	.980	.962	.834	.598	.476
$.99 \le p_1 < .95$	.015	.008	.007	.025	.148	.371	.382
$.95 \le p_1 < .90$	.005	.004	.004	.003	.003	.012	.097
$.90 \le p_1 < .75$	.003	.006	.003	.003	.004	.007	.031
$.75 \le p_1 < .50$	.001	.000	.001	.003	.003	.003	.000
$.50 \le p_1 < .30$	.000	.001	.001	.001	.000	.002	.002
$.30 \le p_1 < .15$	.000	.001	.002	.001	.002	.000	.003
$.15 \le p_1 < .05$	.000	.000	.002	.002	.001	.001	.002
$.05 \le p_1 < .00$	.000	.000	.000	.000	.005	.006	.007

Note: R_t=0.5, SNR=20 (952 signal-induced events, 48 noise-induced events) the simulation was conducted with the filter R equal to 0.5, 1.0 and 2.0. The results are in Table 6-2, and confirm the idea that, as the offset increased, $p_1(t_1)$ would decrease, and that an increase in R could compensate for the offset and still be able to identify the most severely degrading noise-induced events properly. Under nominal conditions and assuming a filter residual equal to 0.0 cm, the filter generated an average probability that an event was signal-induced, $\hat{p}$, of .973. More significant is the fact that the Meer filter was able to distinguish the most severely degrading noise-induced events by giving them an appropriate probabalistic weighting of .05 or less. As the filter residual increased to 3.0 cm, $\hat{p}$ decreased to .255, and the filter had great difficulty in distinguishing the signal-induced events. But, as the filter beam dispersion (R_f) is increased, the Meer filter was able to identify more of the signal-induced events and up to some point, be able to distinguish the most severly degrading noise-induced events.

6.5 Conclusions

The controller stability evaluation led to several conclusions. First, the apparent controller instability is really a combination of an intermittent instability problem and a stability robustness problem. Second, it appears that there are three sources of the controller instability. The varying sample rate (based on the mean signal parameter rate of one event per second) can drive the controller unstable during the longer sample periods. The mismodeling of τ_B can lead to large residuals, especially during the longer sample periods. Finally, as the residuals increases, the Meer filter structure will assume that more of the incoming measurements are noise-induced and the filter will inadvertently drive the Meer filter gain to zero. All three of these problems can be rectified. First, by insuring the design has a much higher mean signal rate parameter, we can reduce the chance that a controller will be driven unstable by a long sample period. Second, developing on-line adaptive estimation of τ_B can reduce the large residuals. Third, by on-line computing of the filter R as a time-weighted average of the recent filter residuals, we can inhibit the Meer filter from misdiagnosing the signal-induced events as being noise-induced, and thereby, keep the Meer filter structure from driving the filter gain inappropriately low.

VII. CONCLUSION AND RECOMMENDATIONS

7.1 Conclusion

Five Monte Carlo analyses and a PI controller stability analysis were performed. The first Monte Carlo analysis looked at seven different cost weighting matrices in an effort to minimize the steady state RMS tracking error, and to enhance the robustness of the PI controller. The significant finding was that a 10 to 1 ratio between the cost weighting element on the states, X_{11}, and the cost weighting element on the pseudo-integral term, X_{55}, resulted in the lowest RMS error. This prevents X_{55} from inducing any degrading oscillations into the controller, while allowing the controller to minimize previous tracking errors and compensate for non-linear disturbances. The results of next Monte Carlo analysis demonstrated that the Meer filter structure can be reduced to a depth of one without any apparent degradation of the controller's performance. This reduction in filter depth should lead to significant savings in on-line computer processing. The third and fourth Monte Carlo analyses were the sensitivity and robustness analyses. The results showed the PI tracker performance was most sensitive to equal changes in the true and filter target dynamics driving noise strengths (Q_T) and to equal changes in the true and filter target measurement noise variances (R_T). This means that a PI controller with perfectly tuned Meer and Kalman filters (i.e., the filters have access to the true parameters), was most sensitive to the changes in the true Q_T and R_T. When the controller was evaluated without access to the truth model's parameters, it was found that parameter variations in the beam time constant, τ_B, and Q_T caused the most significant robustness difficulties. The last Monte Carlo analysis evaluated the on-line adaptive estimation of Q_T. The results showed that the multiple model adaptive controller performed well.

Because the PI controller lacked robustness with respect to the beam time constant, τ_T, an in-depth stability analysis was performed on the PI controller. This was composed of a zero-input stability analysis on both the deterministic full-state feedback and stochastic PI controllers, a controllability and observability evaluation of the Kalman and Snyder-Fishman filter equations, and an analysis on the Meer structure. The apparent stability problem was do to the combined effects of a low signal rate parameter that frequently lead to long, destablizing sample periods, the mismodeling of τ_B that can lead to large signal-induced residuals, and an inherent design problem that would allow the Meer filter to evaluate the large signal-induced residuals as being noise-induced. All three problems can be corrected by using a higher signal rate parameter, by developing on-line adaptive estimation of τ_B, and by calculating the filter target measurement noise variance, R_T, as a function of the time-weighted average of the residual.

7.2 Recommendations

These recommendations are divided into two parts. The first part will suggest ways to improve SOFE and the SOFE user-defined subroutines that provide for the Monte Carlo analyses in this thesis. They are as follows: (1) The large number of runs (200) and the long settling time required for the controller (the first 50 seconds of each run) were based on Zicker's research [27], and it appears to be excessive. It is recommend that the number of runs and the length of the initial transient response be reviewed. This could save a considerable amount of computer time. (2) It is suggested that the length of the detector array (L) be increased. It was found that the target frequently exceeded the 10 cm. length of the detector, especially during conditions other than nominal. Alternatives would be to change the scales so that 10 centimeters would correspond to a large physical field of view, or change the plant or target model gains to reduce the RMS tracker error. (3) It is suggested that the software be updated. The SOFE program used for this research was complied in May 1982 and was written in FORTRAN 4. The current version of SOFE was written in FORTRAN 77 and developed using top-down structured programming, and the current version (Version 2.5) is supported by a new user's manual. The problems with using an older version of the software is the increasing difficulty in obtaining technical support and the inherent difficulty of reading FORTRAN 4 code. (The only SOFE (i.e., not user-defined) subroutine altered by Meer was "ADVANS"; therefore, this should be the only subroutine in the SOFE source code that would have to be rewritten.) Second, if the simplified Meer filter (i.e., with a filter depth of one) is accepted for all further research, much of thee existing Meer filter code can be deleted. Most of the Meer filter code is overhead used to support greater filter depths. This should result in a much more efficient code.

The second half of the recommendations suggests ways of improving the filter's stability robustness. They are as follows: (1) Use a higher signal rate parameter that will generate a higher mean sample rate. This will insure that the controller is stable more than the current 85 percent of the time for nominal conditions. (2) Develop on-line adaptive estimation of τ_B, which could be accomplished through multiple model adaptive methods. If the Meer filter is kept at a filter depth of one, this should be a simple task. The goal is to reduce the large residuals that occur during the mismodeling of τ_B. (3) Develop a variable filter R which is always larger than the true R and is evaluated as a function of the time-weighted average of the the most resent residuals. This should be done with the intent of keeping the mean conditional probability that an observation was signal-induced, $\hat{p}_1(t_i)$, near the true (based on the truth model simulation of signal and noise rates) value of $\hat{p}_1$. At the same time, the filter R must be kept small enough to be able to identify the severely degrading noise events. The goal is to maintain stability robustness by preventing the Meer filter from driving $p_i(t_i)$ to extremely low values, which would effectively "shut down" the filter measurement model. (4) Consider using implicit model-following (see Section 3.4 and Appendix A) as a means to enhance the controller's robustness and to obtain good response characteristics [16]. Once the PI controller demonstrates good robustness for all meaningful parameter variations, the effects of unmodeled beam dynamics and timing delays can be studied. (5) Consider reducing the RMS tracker error by increasing the gain of the particle beam plant (see Section 5.3.1 and specifically Equation (5-2)). This can be done be increasing the control input gain matrix, $\underline{B}(t)$ (for this application, B is a scalar). The effects of different control input gains can be analyzed with frequency domain techniques (like using Bode plots).

Bibliography

1. Athans, M., and Chang, C. B., Adaptive Estimation and Parameter Identification Using Multiple Model Estimation Algorithm, ISD-TR-76-184, Tech. Note 1976-28. Lincoln Laboratory, Lexington, Massachusetts, June 1976.

2. Broussard, J. R., and P. W Berry,, "The Relationship Between Implicit Model Following and Eigenvalue-Eigenvector Placement," IEEE Transactions on Automatic Control, Vol AC-25, No. 3, pp 591-594, June 1980.

3. D'Azzo, John J. and Constantine H. Houpis, Linear Control System Analysis and Design (Second Edition). New York: McGraw-Hill Book Company, 1981.

4. Deshpande, J. G., T. N. Upadhyay and D. G. Lainiotis, "Adaptive Control of Linear Stochastic Systems," Automatica, 9: 107-116 (January 1973).

5. Dixon, W. J. and F. J. Massey, Introduction to Statistical Analysis (Second Edition), McGraw Hill Book Company, New York, 1957.

6. Deming, W. E., Some Theory of Sampling, John Wiley and Sons, New York, 1950.

7. Feldman, R. E. and S. H. Musick, "SOFEPL: A Plotting Postprocessor for 'SOFE', User's Manual", AFWAL-TR-80-1109, Air Force Wright Aeronautical Laboratories, Wright-Patterson AFB, OH, November 1982.

8. Jamerson, Lawrence C. Particle Beam Tracker for an Acceleration Taget, MS Thesis, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, Ohio, December 1985.

9. Fishman, P. M. and D. L. Snyder, "The Statistical Analysis of Space-Time Point Processes," IEEE Transactions on Information Theory, IT-22: 257-265, (May 1976).

10. Kwakernaak, Huibert and Rapheal Sivan, Linear Optimal Control Systems, New York: John Wiley and Sons, Inc., 1972.

11. Korn, J., and L. Beean, "Application of Multiple Model Adaptive Estimation Algorithms to Manuever Detection and Estimation," Technical Report TR-152, Alphatech, Inc., Burlington, Mass., June 1983.

12. Maybeck, Peter S. Stochastic Models, Estimation, and Control, Volume 1. New York: Academic Press, 1979.

13. Maybeck, Peter S. Stochastic Models, Estimation, and Control, Volume 2. New York: Academic Press, 1982.

14. Maybeck, Peter S. Stochastic Models, Estimation, and Control, Volume 3. New York: Academic Press, 1982.

15. Maybeck, Peter S. and Karl P. Hentz, "Investigation of Moving-Bank Multiple Model Adaptive Algorithms," Proceedings of the IEEE 24th Conference on Decision and Control, pp. 1874-1881, IEEE Press, Fort Lauderdale, FL, Dec 85.

16. Maybeck, Peter S., William G. Miller and Jean M. Howey. "Robustness Enhancement fro LQG Digital Flight Controller Design", Preceedings of the National Aerospace and Electronic Conference, IEEE Press, Dayton, OH, May 84.

17. Maybeck, Peter S. and William L Zicker. "MMAE - Based Control with Space Time Point Process Observations," IEEE Transaction on Aerospace and Electronic Systems, AES-21: pp. 292-300 (May 1985).

18. Meer, David E. Multiple Model Adaptive Estimation for Space-Time Point Process Observations, PhD Dissertation, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, Ohio, September, 1982.

19. Miller, W. G., "Robustness Multivariable Controller Desing Via Implicit Model-Following Methods," M.S thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio Dec. 1983

20. Moose, William J. A Proportional-Plus-Integral Controller for a Particle Beam Weapon, MS Thesis, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, Ohio, December 1984.

21. Musick, S. H., "SOFE: A Generalized Digital Simulation for Optimal Filter Evaluation, User's Manual," AFWAL-TR-80-1108, Air Force Wright Aeronautical Laboritories, Wright-Paterson AFB, OH, October 1980.

22. Papoulis, Athanasios. Probability, Random Variables, and Stochastic Processes. New York: McGraw-Hill Book Company, 1965.

23. Rohringer, G. Particle Beam Diagnostics by Resonant Scattering, 1 June 1977--30 November 1977. Contract DASG60-77-C-0120. General Research Corporation, Santa Barbara, CA, December 1977.

24. Santiago, J., "Fundamental Limitations of Optical Trackers," Master's Thesis, AFIT, Wright-Patterson AFB OH, December 1978.

25. Synder, Donald C. and P. M. Fishman, "How to Track a Swarm of Fireflies by Observing Their Flashes," IEEE Transactions on Information Theory, IT-21: 692-695 (Nov 1975).

26. Weiss, J. L., T. N. Upadhyay, and R Tenney. "Finite Computable Filters for Linear Systems Subject to Time Varying Model Uncertainty," paper persented at NAECON May 83 of results obtained in a Masters Thesis by Jerold L. Weiss, "A Comparison of Finite Filtering Methods for Status Directed Processes," Massachusetts Institute of Technology, Cambridge, Massachusetts, June 1983.

27. Zicker, William L. Pointing and Tracking of Particle Beams, MS Thesis, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, Ohio, December 1983.

28. Houpis, Constantine H., and Gary B. Lamont. Digital Control Systems. New York: McGraw-Hill Book Company, 1985.

Appendix A

Alternate Cost Weighting Methods

This appendix will derive the implicit model-following method [16,19], and discuss the LQG/LTR (linear quadratic Gaussian/loop transfer recover) Dual Method of Kwakernaak and Sivan [10]. Both of these methods are designed to enhance the controller's stability robustness.

To be able to derive the implicit model-following cost function, we must start with the general form of the cost function:

$$J_c = E\{\tfrac{1}{2}\underline{x}^T(t_{N+1})\underline{X}_F\,\underline{x}(t_{N+1}) + \int_{t_0}^{t_{N+1}} \tfrac{1}{2}[\underline{x}^T(t)\underline{W}_{xx}(t)\underline{x}(t) + \\ + \underline{u}^T(t)\underline{W}_{uu}(t)\underline{u}(t) + 2\underline{x}^T(t)\underline{W}_{xu}(t)\underline{u}(t)]dt \qquad \text{(A-1)}$$

Because we are designing for steady state, the cost function can be re-defined as:

$$J_c = \tfrac{1}{2}\sum_{i=0}^{\infty}\int_{t_i}^{t_{i+1}} [\underline{x}^T(t)\underline{W}_{xx}\underline{x}(t) + \underline{u}^T(t)\underline{W}_{uu}\underline{u}(t) \\ + 2\underline{x}^T(t)\underline{W}_{xu}\underline{u}(t)]dt \qquad \text{(A-2)}$$

Now, for all $t \epsilon [t_i, t_{i+1})$, $\underline{x}(t)$ is described by:

$$\underline{x}(t) = \underline{\Phi}(t,t_i)\underline{x}(t_i) + \int_{t_i}^{t} \underline{\Phi}(t,\tau)\underline{B}(\tau)\underline{u}(\tau)d\tau$$

$$+ \int_{t_i}^{t} \underline{\Phi}(t,\tau)\underline{G}(\tau)\underline{u}(\tau)d\beta(\tau) \qquad \text{(A-3)}$$

and $\underline{u}(t) = \underline{u}(t_i)$, and $\underline{u}(t_i)$ is constant over the entire sample period; therefore we can define

$$\underline{B}^*(t,t_i) = \int_{t_i}^{t} \underline{\Phi}(t,\tau)\underline{B}(\tau)d\tau \qquad \text{(A-4)}$$

Therefore, the deterministic form of Equation (A-3) is

$$\underline{x}(t) = \underline{\Phi}(t,t_i)\underline{x}(t_i) + \underline{B}^*(t,t_i)\underline{u}(t_i) \qquad \text{(A-5)}$$

and the last integral term in Equation (A-3) has been dropped because the only effect it will have on the cost function is in the form of a loss function, defined as $L_r(t_i)$, and because it cannot be affected by the control input [14]. Said another way, the first part of an assumed certainty equivalence design is the synthesis of the LQ deterministic optimal controller, so the stochastic driving term can be neglected. By substituting Equation (A-5) into Equation (A-2), we get

$$J_c = \frac{1}{2}\sum_{i=0}^{\infty}\int_{t_i}^{t_{i+1}} [(\underline{x}^T(t_i)\underline{\Phi}^T + \underline{u}^T(t_i)\underline{B}^{*T})\underline{W}_{xx}(\underline{\Phi x}(t_i) + \underline{B}^*\underline{u}(t_i))$$

$$+ \underline{u}^T(t_i)\underline{W}_{uu}\underline{u}(t_i)$$

$$+ 2(\underline{x}^T(t_i)\underline{\Phi}^T + \underline{u}^T(t_i)\underline{B}^{*T})\underline{W}_{xu}\underline{u}(t_i)]dt$$

$$= \frac{1}{2}\sum_{i=0}^{\infty}\int_{t_i}^{t_{i+1}} \frac{1}{2}[\underline{x}^T(t_i)\underline{\Phi}^T\underline{W}_{xx}\underline{\Phi x}(t_i)$$

$$+ \underline{u}^T(t_i)\underline{B}^{*T}\underline{W}_{xx}\underline{B}^*\underline{u}(t_i) + \underline{u}^T(t_i)\underline{W}_{uu}\underline{u}(t_i)$$

$$+ \underline{u}^T(t_i)\underline{B}^{*T}\underline{W}_{xu}\underline{u}(t_i) + \underline{u}^T(t_i)\underline{W}_{xu}{}^T\underline{B}^*\underline{u}(t_i)$$

$$+ 2\underline{x}^T(t_i)\underline{\Phi}^T\underline{W}_{xx}\underline{B}^*\underline{u}(t_i) + 2\underline{x}^T(t_i)\underline{\Phi}^T\underline{W}_{xu}\underline{u}(t_i)]dt$$

and since $\underline{x}(t_i)$ and $\underline{u}(t_i)$ are constants, this becomes

$$J_c = \frac{1}{2}\sum_{i=0}^{\infty}[\underline{x}^T(t_i)\int_{t_i}^{t_{i+1}}(\underline{\Phi}^T\underline{W}_{xx}\underline{\Phi})dt\cdot\underline{x}(t_i)$$

$$+ \underline{u}^T(t_i)\int_{t_i}^{t_{i+1}}(\underline{B}^{*T}\underline{W}_{xx}\underline{B}^* + \underline{W}_{uu} + \underline{B}^{*T}\underline{W}_{xu} + \underline{W}_{xu}{}^T\underline{B}^*)dt\cdot\underline{u}(t_i)$$

$$+ 2\underline{x}^T(t_i)\int_{t_i}^{t_{i+1}}(\underline{\Phi}^T\underline{W}_{xx}\underline{B}^* + \underline{\Phi}^T\underline{W}_{xu})dt\cdot\underline{u}(t_i) \qquad \text{(A-6)}$$

By substituting Equation (A-6) into

$$J_c = \tfrac{1}{2}\sum_{i=0}^{\infty}[\underline{x}^T(t_i)\underline{X}(t_i)\underline{x}(t_i) + \underline{u}^T(t_i)\underline{U}(t_i)\underline{u}(t_i) + 2\underline{x}^T(t_i)\underline{S}(t_i)\underline{u}(t_i)]dt \qquad \text{(A-7)}$$

it yields

$$\underline{X}(t_i) = \int_{t_i}^{t_{i+1}} (\underline{\Phi}^T(t,t_i)\underline{W}_{xx}\underline{\Phi}(t,t_i))dt \qquad \text{(A-8)}$$

$$\underline{U}(t_i) = \int_{t_i}^{t_{i+1}} (\underline{B}^{*T}(t,t_i)\underline{W}_{xx}\underline{B}^*(t,t_i) + \underline{W}_{uu} + \underline{B}^{*T}(t,t_i)\underline{W}_{xu} + \underline{W}_{xu}{}^T\underline{B}^*(t,t_i))dt \qquad \text{(A-9)}$$

$$\underline{S}(t_i) = \int_{t_i}^{t_{i+1}} (\underline{\Phi}^T(t,t_i)\underline{W}_{xx}\underline{B}^*(t,t_i) + \underline{\Phi}^T(t,t_i)\underline{W}_{xu})dt \qquad \text{(A-10)}$$

By definition, $\underline{X}(t_i)$ and $\underline{W}_{xx}(t)$ are symmetrical positive semi-definite, $\underline{U}(t_i)$ and $\underline{W}_{uu}(t)$ are symmetrical positive definite, and $\underline{S}(t_i)$ and $\underline{W}_{xu}(t)$ are defined such that the expression

$$[\underline{X}(t_i) - \underline{S}(t_i)\underline{U}^{-1}(t_i)\underline{S}(t_i)^T]$$

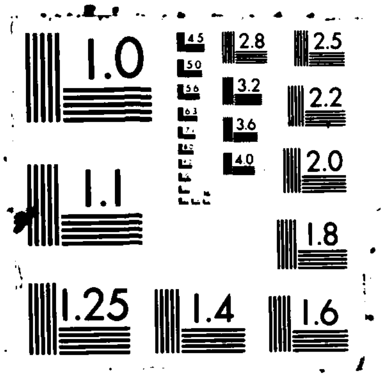
is positive semi-definite.

Instead of using the quadratic cost term $\underline{x}^T(t)\underline{W}_{xx}(t)\underline{x}(t)$ in the integral in the integral on Equation (A-2) that is used to drive $\underline{x}(t)$ to "zero", we can define a model of desired characteristics as:

AD-A182 618 STOCHASTIC ADAPTIVE PARTICLE BEAM TRACKER USING MEER FILTER FEEDBACK(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING 3/3

UNCLASSIFIED B A JOHNSON DEC 86 AFIT/GE/ENG/86D-27 F/G 14/2 NL

END
8-87
DTIC


1.0
4.5
5.0
5.6
6.3
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

$$\dot{\underline{y}}_M(t) = \underline{F}_M(t)\underline{y}_M(t) \tag{A-11}$$

and then put a quadratic penalty on the deviations of the actual system outputs ($\underline{y} = \underline{C}\underline{x}$) from those characteristics:

$$\begin{aligned} \{\dot{\underline{y}} - \underline{F}_M\underline{y}\}^T\underline{W}_{yy}\{\dot{\underline{y}} - \underline{F}_M\underline{y}\} & \\ &= \{\underline{C}\dot{\underline{x}} - \underline{F}_M\underline{y}\}^T\underline{W}_{yy}\{\underline{C}\dot{\underline{x}} - \underline{F}_M\underline{y}\} \\ &= \{[\underline{C}\underline{F}-\underline{F}_M\underline{C}]\underline{x} + \underline{C}\underline{B}\underline{u}\}^T\underline{W}_{yy}\{[\underline{C}\underline{F}-\underline{F}_M\underline{C}]\underline{x} + \underline{C}\underline{B}\underline{u}\} \end{aligned} \tag{A-12}$$

The $\underline{x}^T\underline{W}_{xu}\underline{u}$ quadratic cost term in Equation (A-2) is replaced with

$$\{\dot{\underline{y}} - \underline{F}_M\underline{y}\}^T\underline{W}_{yu}\underline{u} = \{[\underline{C}\underline{F}-\underline{F}_M\underline{C}]\underline{x} + \underline{C}\underline{B}\underline{u}\}^T\underline{W}_{yu}\underline{u} \tag{A-13}$$

Therefore, the implicit model-following cost function is:

$$\begin{aligned} \underline{J}_I = \tfrac{1}{2}\sum_{i=0}^{\infty}\int_{t_i}^{t_{i+1}} & \{[\underline{C}\underline{F}-\underline{F}_M\underline{C}]\underline{x} + \underline{C}\underline{B}\underline{u}\}^T\underline{W}_{yy}\{[\underline{C}\underline{F}-\underline{F}_M\underline{C}]\underline{x} + \underline{C}\underline{B}\underline{u}\} \\ & + \underline{u}^T\underline{W}_{uu}\underline{u} + \{[\underline{C}\underline{F}-\underline{F}_M\underline{C}]\underline{x} + \underline{C}\underline{B}\underline{u}\}^T\underline{W}_{yu}\underline{u}\}dt \end{aligned} \tag{A-14}$$

Using the same derivation as above, this can be written as

$$\begin{aligned} J_I = \tfrac{1}{2}\sum_{i=0}^{\infty} [& \underline{x}^T(t_i)\underline{X}_I(t_i)\underline{x}(t_i) + \underline{u}^T(t_i)\underline{U}_I(t_i)\underline{u}(t_i) \\ & + 2\underline{x}^T(t_i)\underline{S}_I(t_i)\underline{u}(t_i)]dt \end{aligned} \tag{A-15}$$

where

$$\underline{X}_I(t_i) = \int_{t_i}^{t_{i+1}} \{\underline{\Phi}^T(t,t_i)(\underline{CF}-\underline{F}_M\underline{C})^T\,\underline{W}_{yy}(\underline{CF}-\underline{F}_M\underline{C})\underline{\Phi}(t,t_i)\}dt \qquad \text{(A-16)}$$

$$\begin{aligned}\underline{U}_I(t_i) = \int_{t_i}^{t_{i+1}} \{&[(\underline{CF}-\underline{F}_M\underline{C})\underline{B}^*(t,t_i)+\underline{CB}]^T\,\underline{W}_{yy}[(\underline{CF}-\underline{F}_M\underline{C})\underline{B}^*(t,t_i)+\underline{CB}]\\ &+ \underline{W}_{uu} + [(\underline{CF}-\underline{F}_M\underline{C})\underline{B}^*(t,t_i)+\underline{CB}]^T\,\underline{W}_{yu}\\ &+ \underline{W}_{yu}^T[(\underline{CF}-\underline{F}_M\underline{C})\underline{B}^*(t,t_i)+\underline{CB}]\}dt \qquad \text{(A-17)}\end{aligned}$$

$$\begin{aligned}\underline{S}_I(t_i) = \int_{t_i}^{t_{i+1}} \{&\underline{\Phi}^T(t,t_i)(\underline{CF}-\underline{F}_M\underline{C})^T\,\underline{W}_{yy}[(\underline{CF}-\underline{F}_M\underline{C})\underline{B}^*(t,t_i)+\underline{CB}]\\ &+ \underline{\Phi}^T(t,t_i)(\underline{CF}-\underline{F}_M\underline{C})^T\,\underline{W}_{yu}\}dt \qquad \text{(A-18)}\end{aligned}$$

If we were to use the implicit model-following in the PI tracker as described in Section 3.3.2, we would use the same augmented states which include a pseudo-integral term, $\underline{C}(t_i)$ would be defined by Equation (3-56), $\underline{B}(t_i)=[1\ 0\ 0\ 0\ 0]^T$, and $\underline{\Phi}(t,t_i)$ would be

$$\underline{\Phi}(t,t_i) = \begin{bmatrix} A & 0 & 0 & 0 & 0 \\ 0 & 1 & \Delta t & D & 0 \\ 0 & 0 & 1 & C & 0 \\ 0 & 0 & 0 & B & 0 \\ 1 & -1 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(A-19)}$$

where

$A = \exp(-\Delta t/\tau_B)$

$B = \exp(-\Delta t/\tau_T)$

$C = \tau_T[1 - B]$

$D = \tau_T[\Delta t - C]$

$\Delta t = t - t_i$

The cost weighting matrices $\underline{W}_{xx}(t)$, $\underline{W}_{xu}(t)$ and $\underline{W}_{uu}$ of the general cost function, and $\underline{W}_{yy}(t)$ and $\underline{W}_{yu}(t)$ of the implicit model-following method must be selected as to represent the physical objective we are trying to accomplish with the individual states. For example (see Equation (3-62)),

$$\underline{W}_{xx}(t) = \underline{W}_{yy}(t) = \begin{bmatrix} a & -a & 0 & 0 & 0 \\ -a & a & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & a/10 \end{bmatrix} \qquad \text{(A-20)}$$

will drive the error between the target and the beam states to zero, while applying a 10 to 1 ratio to the pseudo-integral term. The 10 to 1 ratio is suggested because it delivered the best performance (see Section 5.1). The cost weighting matrix, $\underline{W}_{yy}(t)$, must be symmetrical, positive semi-definite, and the $\underline{W}_{yu}(t)$ term must be such that

$$[\underline{X}_I(t_i) - \underline{S}_I(t_i)\underline{U}^{-1}(t_i)\underline{S}_I^T(t_i)]$$

is a positive semi-definite matrix.

As the result of the implicit model-following technique, Equation (A-11) is embedded into the cost function, and the desired model does not appear explicitly in the final controller structure. That is, the additional states used to achieve the desired characteristics of the controller are not augmented to the original system model states. Implicit model-following gives us the flexibility to select the model of the desired closed loop system characteristics by allowing us to choose $\underline{F}_M$ to have a desired eigenstructure. In other words, we can select $\underline{F}_M$ in such a manner as to place the poles and orient the eigenvectors to develop a controller with good loop shape (desirable feedback control characteristics) and enhance robustness [2].

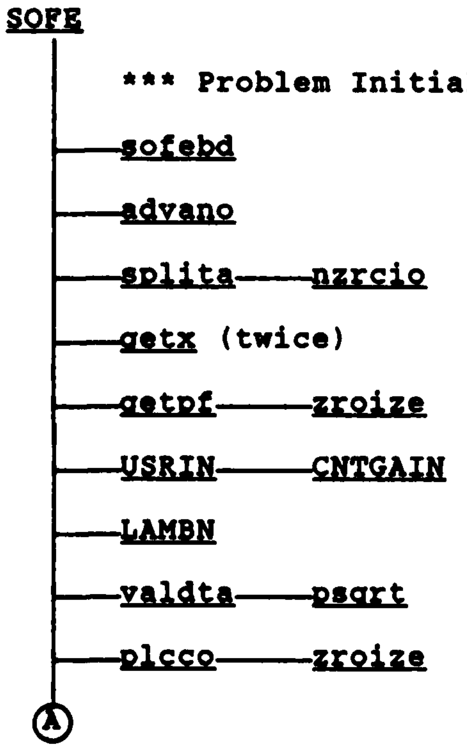
Another method that can be used to enhance robustness is LQG/LTR Dual Method of Kwakernaak and Sivan [10]. This is a specific method of choosing $\underline{W}_{xx}$ that enhances the system stability robustness at the plant output. This is done by treating the Kalman filter (or Snyder-Fishman filter as it would be the case for this application) as a dynamic compensator and using it to establish good loop shapes in order to achieve good command following, good disturbance rejection and low sensitivity to plant variations from the nominal conditions. Then $\underline{W}_{xx}$ is chosen iteratively as some initial $\underline{W}_{xx}$ value plus the $q^2 \underline{C}^T \underline{V} \underline{C}$, where $\underline{V}$ is a positive definite weighting matrix and q^2 is a scaler that is increased to acheive as much of the desirable robustness characteristics as possible in the implementable filter-controller structure.

Appendix B

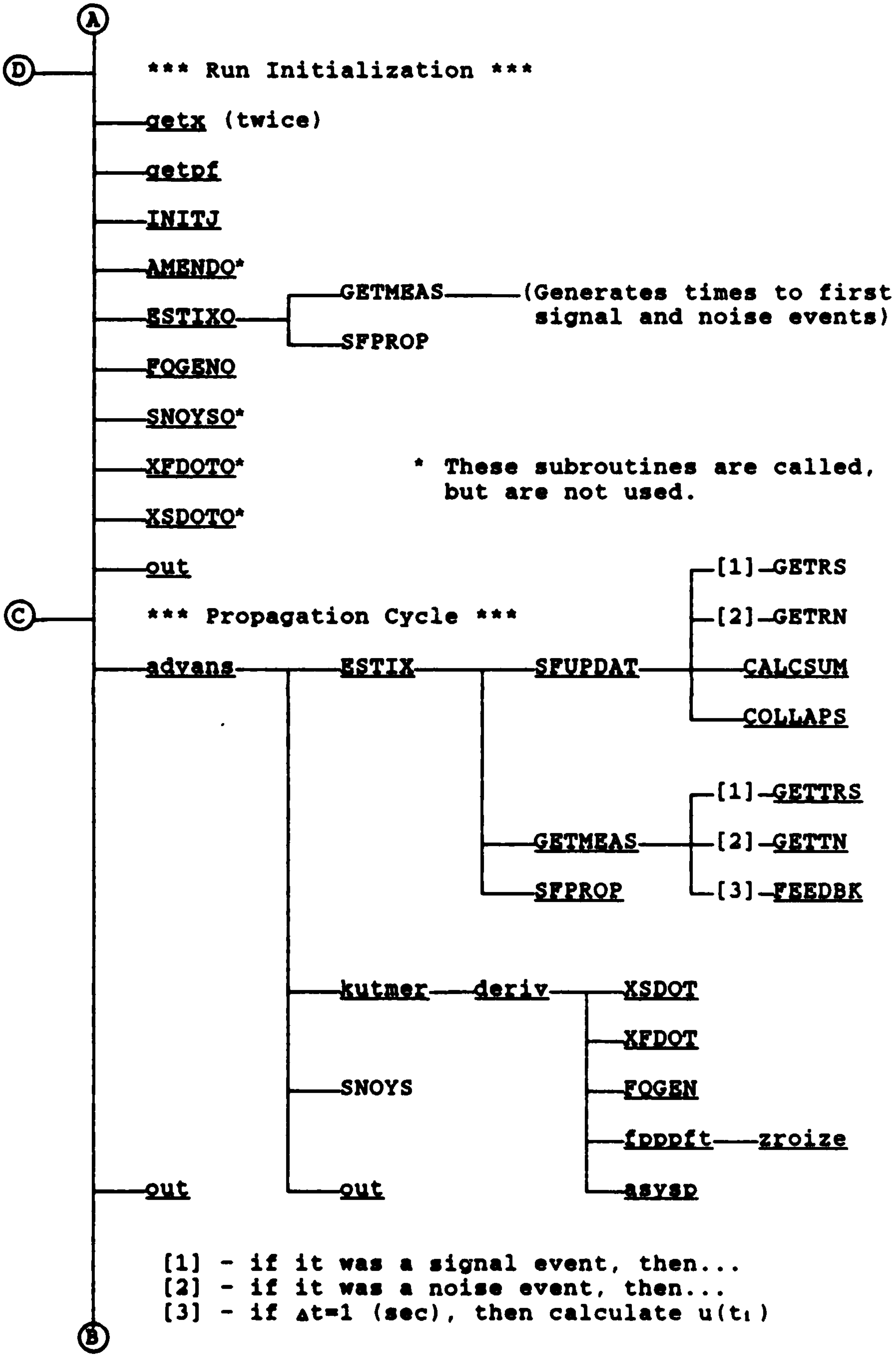
Outline of SOFE Operations

B.1 Structure of the Software

This section provides a functional outline of subroutines used in SOFE [7,21] for this application. The lower case letters are the internal SOFE subroutines, and the upper case letters are the user-defined subroutines. Because a shaping filter was used to generate the target trajectory rather than using an externally generated trajectory, the following subroutines were not called: span, TRAJO, move, interp, icsevu, or icsicu.

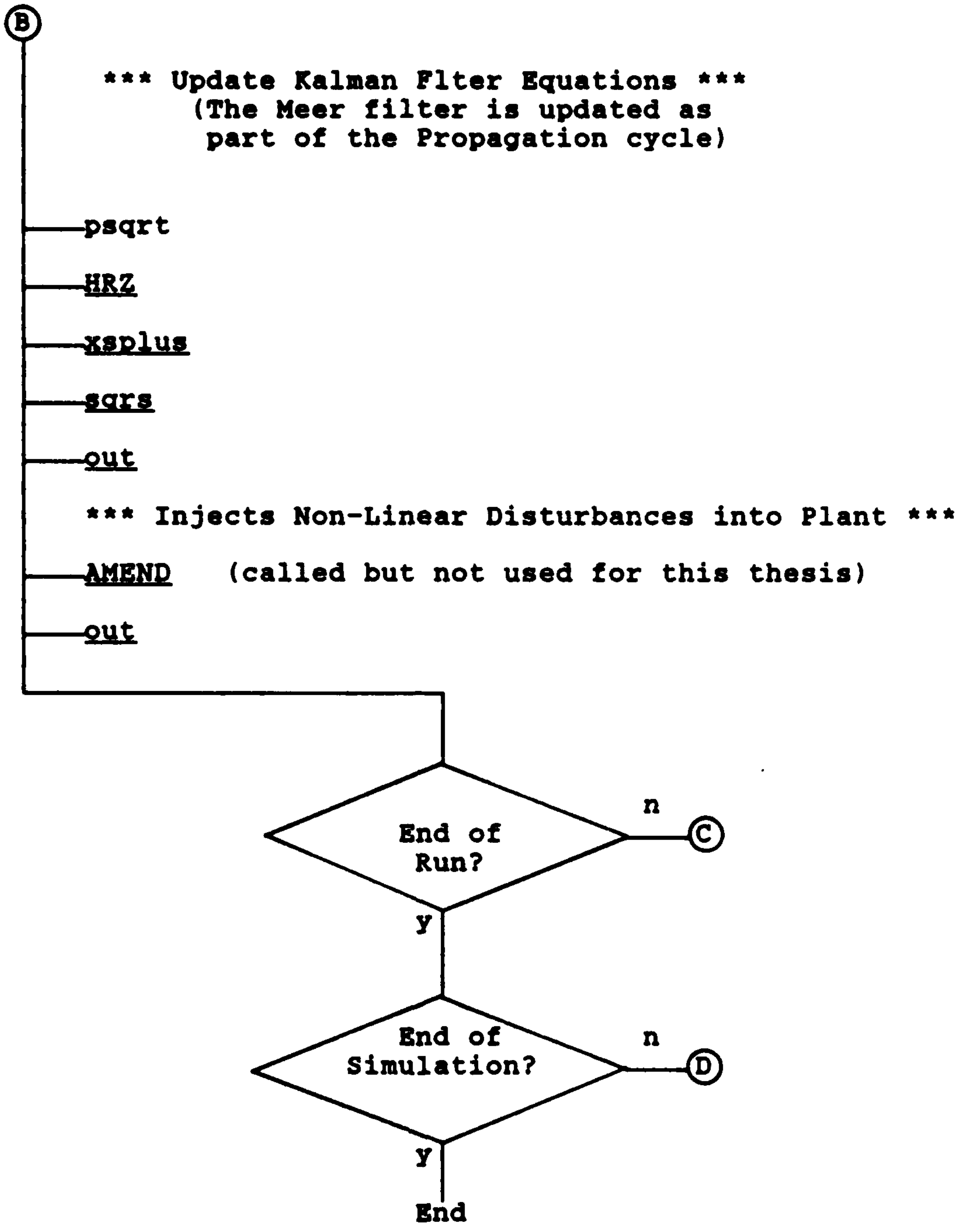


(Calculates λ_N from SNR)



* These subroutines are called, but are not used.

[1] - if it was a signal event, then...
[2] - if it was a noise event, then...
[3] - if $\Delta t=1$ (sec), then calculate $u(t_i)$


B
*** Update Kalman Flter Equations ***
(The Meer filter is updated as
part of the Propagation cycle)
psqrt
HRZ
xsplus
sqrs
out
*** Injects Non-Linear Disturbances into Plant ***
AMEND (called but not used for this thesis)
out
End of
Run?
n
C
y
End of
Simulation?
n
D
y
End

Notes:

GETRS, GETRN, HRZ and SNOYS use the function GAUSS (see Equations (4-12) and (4-16) for HRZ and SNOYS)

GETTS and GETTN use the function Poisson (see Equation (4-18))

Subroutine "out" calls several different subroutines and collects the data required for the SOFE output file, and by SOFEPL for statistical reduction of the tracker and filter state errors (see Equations (4-1) through (4-9)).

Definitions of the Subroutines:

advans	controls propagation between measurement updates
advano	initializes advans
asysp	adds $\underline{GQG}^T$ together for: $d\underline{P}/dt=\underline{FP}+\underline{PF}^T+\underline{GQG}^T$, the differental form of Equation (3-4).
deriv	evaluates the derivatives of the filter and truth states
fppft	adds $FP+PF^T$ together.
gauss	Gaussian random number generator
getpf	reads the initial covariance matrix, $\underline{P}_0$
getx	reads in the initial truth and filter states, $\underline{x}_0$
kutmer	Kutta-Merson integration algorithm
nzrcio	reads the non-zero input from the disk
out	outputs all scheduled data
psqrt	takes the Cholesky upper triangular square root of $\underline{P}(t_1)$
sofebd	reads the PRDATA block data file from the disk PRDATA is the SOFE simulation specification file.
splita	partitions the unlabeled common block 'A' into the states, filter covariances and other data
sqrs	squares the Cholesky square root to form $\underline{P}(t_1)$
valdta	validates the input data
xsplus	performs the measurement update on the filter and truth model.
AMEND	used to inject non-linear disturbances into the beam dynamics (not used - variable VTRK set to 0)
AMENDO	initializes AMEND (not used)

ESTIX	used to control the Meer filter updates and propagation cycles
ESTIXO	used to initialize the Meer filter propagation cycle
FQGEN	calculates the non-zero elements in $\underline{F}(t)$ and $\underline{Q}(t)$
FQGENO	initializes $\underline{F}(t)$ and $\underline{Q}(t)$
HRZ	calculates $\underline{H}_T(t_i)$, $r_T(t_i)$, $A_T(t_i)$ and $z_T(t_i)$
HRZO	initializes $\underline{H}_T(t_0)$
SNOYS	used to inject the white noise into the target shaping filter and the beam dynamics model
SNOYSO	initializes SNOYS (not used)
USRIN	reads USR1DAT block data file from disk, does initial filter and gain calculations, sets up truth and filter parameter, prints out non-scheduled data
INITJ	initializes the Meer filter states and covariance matrices
XFDOT	filter state derivatives
XFDOTO	alternate method for initializing XFDOT (not used)
XSDOT	truth state derivatives
XSDOTO	alternate method for initializing XSDOT (not used)
GETTS	generates the time to the next signal event
GETTN	generates the time to the next noise event
LAMBN	generates the noise rate parameter (see Equation (4-24))
GETRN	generates the spatial location of the noise event, r_N (see Equation (2-2))
GETRS	generates the spatial location of the signal event, r_S (see Equation (2-1))
GETMEAS	generates the time to the next signal, noise or measurement update
SFUPDAT	get the next event and updates the Meer filter (see Equations (2-3), (2-13) through (2-15)

COLLAPS	implements either the "best half" or "merge" algorithms (see Section 2.4)
POISSON	Poisson random number generator (see Equation (4-18))
SFPROP	propagates the Meer filter (see Equations (2-4) through (2-7), (2-10) and (2-11))
FEEDBK	calculates $\underline{u}(t_i)$ (see Equation (3-63))
CNTGAIN	calculates $\underline{G}_c{}^*$ and $\underline{E}$ (see Equations (3-64) and (3-65))

B.2 Subroutines Modified to Implement the MMAC

FQGEN, FQGENO, HRZ, HRZO, XFDOT and XSDOT were modified to included the increase in dimensionality of $\underline{x}_{Tf}(t_i)$, $\underline{x}_{Tf}(t_0)$, $\underline{x}_{Tt}(t_i)$, $\underline{x}_{Tt}(t_0)$, $\underline{F}_T(t)$, $\underline{Q}_T(t)$, $\underline{P}_T(t_i)$, $\underline{P}_T(t_0)$ and $\underline{H}_T(t_i)$, which occurred as a result of embedding the 3 elemental controllers into SOFE. The measurement updates were done recursively, using the same measurement for all three elemental filters. The elemental filter residuals, $r_k(t_i)$, and residual variances, $A_k(t_i)$ (see Equation (3-80)), were calucated in subroutine HRZ, and the probability that an elemental filter was correct, $p_k(t_i)$ (see Equation (3-79)), was calculated in a new subroutine called CPROB. Because the control input was the probabalistic sum of the elemental control inputs (see Equation (3-82)), subroutine FEEDBACK was replaced with a new subroutine called MMAC. CNTGAIN was modified to calculate the elemental steady-state controller gains, $\underline{G}_{ck}{}^*(\underline{a}_k)$, and they were a function of the discretized uncertain parameter, $\underline{a}_k$. SNOYS was modified to extend the white noise to all the elemental filter/controllers, and USRIN had to be modified to accommodate the additional filter/controllers. The parameter variations were simulated through subroutine ESTIX, and initialized in ESTIXO.

Appendix C

Controller Pole/Zero Location as a Function of Different Sample Rates

C.1 Pole/Zero Locations for $\tau_{Bt}=19.0$

Δt	pole(1)	pole(2)	pole(3)	zero(1)	zero(2)
.01	.99731	.81799	-.21466	.82532	.78471
.5	.84321±j.017096		-.26176	.78703	.78471
1.0	.74763	.79959	-.32636	.75103	.78471
2.0	.48560	.81479	-.52990	.68687	.78471
3.0	.31488	.81769	-.85387	.63136	.78471
3.5	.25903	.81862	-1.05451	.60630	.78471
4.605	.17996	.82017	-1.55394	.55625	.78471
5.0	.16120	.82060	-1.74383	.53992	.78471
10.0	.06537	.82386	-4.19311	.38312	.78471

The PI controller goes unstable at Δt>3.369.

C.2 Pole/Zero Locations for $\tau_{Bt}=19.6$

Δt	pole(1)	pole(2)	pole(3)	zero(1)	zero(2)
.01	.99732	.81799	-.21465	.82532	.78471
.5	.84267±j.016724		-.26175	.78703	.78471
1.0	.74969	.79878	-.32632	.75103	.78471
2.0	.48742	.81461	-.52973	.68687	.78471
3.0	.31649	.81752	-.85382	.63136	.78471
3.5	.26049	.81845	-1.05478	.60630	.78471
4.605	.18113	.81999	-1.55572	.55625	.78471
5.0	.16229	.82041	-1.74642	.53992	.78471
10.0	.06609	.82364	-4.21612	.38312	.78471

The PI controller goes unstable at Δt>3.369.

C.3 Pole/Zero Locations for $\tau_{Bt}=20.0$

Δt	pole(1)	pole(2)	pole(3)	zero(1)	zero(2)
0.0	1.0	.81786	-.21466	.82613	.78471
.01	.99733	.81799	-.21466	.82532	.78471
.1	.97299	.81930	-.22234	-	.78471
.25	.93059	.82244	-.23606	-	.78471
.50	.84290±j.01648		-.26175	.78704	.78471
.75	.80872±j.03280		-.29157	-	.78471
1.0	.75103	.79822	-.32629	.75103	.78471
2.0	.48858	.81449	-.52963	.68687	.78471
3.0	.31712	.81741	-.85378	.63136	.78471
3.5	.26142	.81834	-1.05494	.60630	.78471
4.0	.21939	.81911	-1.27438	.58280	.78471
4.605	.18167	.81988	-1.55878	.55625	.78471
5.0	.16299	.82030	-1.74806	.53992	.78471
10.0	.06655	.82351	-4.23078	.38312	.78471

The PI controller goes unstable at $\Delta t>3.368$.

C.4 Pole/Zero Locations for $\tau_{Bt}=20.4$

Δt	pole(1)	pole(2)	pole(3)	zero(1)	zero(2)
.01	.99735	.81799	-.21466	.82532	.78471
.5	.84311±j.01624		-.26174	.78704	.78471
1.0	.75235	.79766	-.32627	.75103	.78471
2.0	.48970	.81438	-.52953	.68687	.78471
3.0	.31851	.81730	-.85375	.63136	.78471
3.5	.26231	.81824	-1.05510	.60630	.78471
4.605	.18260	.81976	-1.55793	.55625	.78471
5.0	.16367	.82019	-1.74964	.53992	.78471
10.0	.06699	.82337	-4.24494	.38312	.78471

The PI controller goes unstable at $\Delta t>3.368$.

C.5 Pole/Zero Locations for $\tau_{Bt}=21.0$

Δt	pole(1)	pole(2)	pole(3)	zero(1)	zero(2)
.01	.99736	.81799	-.21466	.82532	.78471
.5	.84343±j.01588		-.26174	.78704	.78471
1.0	.75431	.79679	-.32624	.75103	.78471
2.0	.49130	.81422	-.52939	.68687	.78471
3.0	.31992	.81714	-.85370	.63136	.78471
3.5	.26359	.81808	-1.05532	.60630	.78471
4.605	.18363	.81960	-1.55947	.55625	.78471
5.0	.16464	.82002	-1.75189	.53992	.78471
10.0	.06762	.82318	-4.26528	.38312	.78471

The PI controller goes unstable at $\Delta t>3.368$.

C.6 Equivalent Deterministic Full-State Feedback Controller Transfer Function Derived in the s-Domain

$$\frac{Y_c(s)}{Y_r(s)} = \frac{s + G_{c2}^*}{s(s + 1/\tau_{Bt}) + G_{c1}^*(s + G_{c2}^*/G_{c1}^*)} \qquad \text{(C-1)}$$

Assuming we want to know the location of the poles for $G_{c1}^*=1.2422$ and $G_{c2}^*=0.27436$, Equation (C-1) becomes

$$\frac{Y_c(s)}{Y_r(s)} = \frac{s + .27436}{(s + 1.024365)(s + .267835)} \qquad \text{(C-2)}$$

These equations are the continuous-time, s-domain form of Equations (6-8) and (6-9).

Appendix D

Sample Data Used to Calculate the Statistics in Table 5-25

In an effort to estimate the time it takes the MMAC to adapt to changes in Q_T, a single Monte Carlo simulation run of 2000 seconds was performed. The uncertain parameter, Q_T, was varied from 0.01 to 1.00 cm^2/sec^5, and the following data was collected.

	Q_T Increasing from .01 to 1.0				Q_T Decreasing from 1.0 to .01			
Sample	t_D	t_R	t_S	t_T	t_D	t_R	t_S	t_T
1	4	1	1	5	7	34	21	41
2	9	7	6	16	6	16	8	22
3	4	2	1	6	3	20	8	23
4	4	2	1	6	6	17	4	23
5	3	12	12	15	3	33	26	36
6	7	2	2	9	4	38	31	42
7	4	16	16	20	6	62	52	68
8	7	1	1	8	5	32	13	37
9	4	8	8	11	3	34	27	37
10	3	11	11	15	3	71	66	74
11	5	19	18	22	2	21	10	23
12	5	22	22	27	4	36	26	40
13	5	7	6	12	5	31	18	36
14	6	10	2	16	3	8	3	11
15	8	2	2	10	8	25	15	33
16	1	30	14	31	8	24	17	32
17	5	2	1	7	10	23	10	33
18	9	2	1	11	5	46	18	51
19	6	11	11	17	5	27	17	32
20	9	2	2	11	5	-	-	-

t_D, t_R, t_S and t_T are defined in Table 5-25.

Times are in seconds (or number of sample periods, $\Delta t=1.0$)

See Table 5-25 for more detail.

VITA

Bruce A. Johnson was born on 1 August 1959. He attended the United States Air Force Academy and graduated with a B.S.E.E. in May 1982. Upon graduation, he received a regular commision in the United States Air Force. His first assignment was at Electronic Systems Division, Hanscom AFB, Massachusettes, where he served as Electrical Engineer/Project Manager on the Adaptable Surface Interface Terminal of the Joint Tactical Information Distribution System, and served as the Executive Officer to the Deputy for Tactical Systems. In May 1985, he entered the Air Force Institute of Technology and received his M.S.E.E. in December 1986. In April 1987, Captain Johnson was assigned to Headquarters Air Force.

Permanent Address: 613 Sebastopol St.
Claremont, CA 91711

UNCLAS FIED
SECURITY C SIFICATION OF THIS PAGE

AD A182 610

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED

1b. RESTRICTIVE MARKINGS

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION / DOWNGRADING SCHEDULE

3. DISTRIBUTION / AVAILABILITY OF REPORT
Approved for public release
Distribution unlimited

4. PERFORMING ORGANIZATION REPORT NUMBER(S) AFIT/GE/ENG/86D-27

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION
School of Engineering
AF Insititute of Technology

6b. OFFICE SYMBOL *(If applicable)* AFIT/EN

7a. NAME OF MONITORING ORGANIZATION

6c. ADDRESS *(City, State, and ZIP Code)*

7b. ADDRESS *(City, State, and ZIP Code)*

8a. NAME OF FUNDING / SPONSORING ORGANIZATION

8b. OFFICE SYMBOL *(If applicable)*

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c. ADDRESS *(City, State, and ZIP Code)*

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO	WORK UNIT ACCESSION NO.

11. TITLE *(Include Security Classification)*
STOCHASTIC ADAPTIVE PARTICLE BEAM TRACKER USING MEER FILTER FEEDBACK (UNCLASSIFIED)

12. PERSONAL AUTHOR(S) Bruce A. Johnson, B.S.E.E., Captain, USAF

13a. TYPE OF REPORT MS Thesis

13b. TIME COVERED FROM ________ TO ______

14. DATE OF REPORT *(Year, Month, Day)* 1986 December

15. PAGE COUNT 216

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB-GROUP
09	04	
12	01	

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Adaptive Estimation, Adaptive Control, Multiple Model Adaptive Controller, Kalman Filtering, Particle Beam, Space-Time Point Process, (more)

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

Title: Stockastic Adpative Particle Beam Tracker Using Meer Filter Feedback

Thesis Chairman: Dr. Peter S. Maybeck
Professer, Dept pf Eoectrical Engineering
Air Force Institute of Technology

Approved for public release: IAW AFR 190-1.
Lynn Wolaver 5 May 87
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (~~ATC~~)
Wright-Patterson AFB OH 45433

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL Dr. Peter S. maybeck

22b. TELEPHONE *(Include Area Code)* (513) 225-3450

22c. OFFICE SYMBOL AFIT/ENG

DD Form 1473, JUN 86 *Previous editions are obsolete.* SECURITY CLASSIFICATION OF THIS PAGE

Block #18 - Subject Terms

Poisson Process, Stochastic Esitmation, Target Tracking

Block #19

Abstract

The goal of this research was to develop a realizable proportional-plus-integral (PI) feedback tracker to control a neutral particle beam. The design is based on detecting the photo-electron events that are emitted from a laser-excited particle beam and the observed events are used by a Meer filter to locate the beam's centerline. The observed events are modeled by a Poisson space-time process and are composed of both signal- and noise-induced events. The Meer filter is a stochastic multiple model adaptive estimator which is composed of a bank of Snyder-Fishman filters and is designed to distinguish the signal-induced events from the noise-induced events. A target model is developed from a Gauss-Markov acceleration process, and the target states are estimated by a Kalman filter. The "optimal" PI controller design is based on the linear quadratic (LQ) controller synthesis technique and the "assumed" certainty equivalence property, and the Kalman filter provides the reference (target) states while the Meer filter supplies controlled (beam) states. The objectives of the research were to (1) select the "best" cost weighting matrices that minimize the RMS tracker error and enhance robustness, (2) simplify the Meer filter for easier on-line usage, (3) complete full-scale sensitivity and robustness analyses over all the Kalman and Meer filter parameters, and (4) develop on-line adaptive estimation of those parameters that greatly affect stability robustness and tracker performance. During the research, an apparent stability problem was uncovered, and a fifth objective was to identify the source of the instability, and to propose a solution that would insure stability during parameter variations.

END
8-87
DTIC