THE DEVELOPMENT, ANALYSIS AND EVALUATION OF AN OPTICAL TRACKER FOR TRACKING HIGH-SPEED, MANOEUVERING TARGETS

VOLUME I

RONALD ALFRED -ERND SCHNEIDER

A DISSERTATION SUBMITTED TO THE FACULTY OF ENGINEERING, UNIVERSITY OF THE WITWATERSRAND, JOHANNESBURG, FOR THE DEGREE OF MASTER OF SCIENCE.

PRETORIA, APRIL 1981

I, Ronald Alfred Bernd Schneider, Student number 78-6099/9, hereby formally declare that the dissertation submitted, titled

"The Development, Analysis and Evaluation of an Optical Tracker for Tracking High-speed, Manoeuvering Targets"

1 is my own unaided work,

2 that no substance or any part of it has been submitted in the past or is being submitted for a degree in any university, and

3 that [illegible] in this dissertation has been obtained by me while employed by Kentron (Pty) Ltd, Private Bag X336, Pretoria.

SIGNED, THIS 29TH DAY OF APRIL 1981.

CONTENTS PAGE

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

This dissertation presents the design for an optical tracker which accurately quantifies the line-of-sight between a target and the optical tracker. An optical tracker forms an essential element of defensive installations, yet publications on tracking schemes for optical trackers, as opposed to radar trackers, are hard to find.

In this dissertation a comprehensive analysis of an optical tracker for high-speed, manoeuvering targets has been done. The optical tracker was to be mobile, and therefore subject to vehicle motion perturbations. A literature survey was made, giving an overview of methods available for tracking such targets and quantifying the resulting line-of-sight. The options considered included line-of-sight data obtained as outputs of Kalman or Linear Regression filters of various orders. Tracking aids using linear filters, among others, were also presented in the literature.

The human operator was seen to be an integral part of the optical tracker. The literature survey pointed out some of the advantages inherent in including the human operator in series in the loop, but it was clear that difficulties would arise when trying to design an optimized controller around the human operator.

The approach taken in this dissertation was to make a mathe= matical model of the system so that the controller could be analyzed, simulated and evaluated in the shortest possible time and at the lowest possible cost. This included human operator models and, in some cases, actual human operators.

First the equations of motion for the gimballed platform containing the stabilized optical element were derived from first principles. Using the linearized version of these equations, classical controller design methods (e.g. Root Locus techniques, Bode diagrams) were applied to determine the trans= fer function of the optimum stabilization control loop. The non-linear equations of motion were used to construct a simu= lator on which the controller could be tuned. The performance of the stabilization loops was found with respect to base motion stabilization as well as to input command. Non-linear=

ities included were, among others, friction, stiction and voltage and current limits.

A human operator model was constructed using measured data from a simulated tracking task and a parameter estimation routine. This model was coupled to the stabilization loops to form a tracking loop, and it was shown that tracker instability could occur at low loop gains. After conside ing various tracking aid algorithms, it was found that a proportional-plus-integral-type aid improved the gain margin considerably.

The implementation of this tracking aid was checked by subjecting the simulator to tests using three different human operators. Results confirmed that for the tracking tasks, analyzed, the introduction of the tracking aid improved system performance fourfold. Furthermore, the tracking aid was implemented digitally, and it was found that the tracking aid could be adapted to form part of a linear filter. This was used to generate best estimates of the target position and rates in inertial space. Range information was also available, filtered, and together with the angular data, fed to the main control unit so that the aiming angles of the guns could be calculated.

This dissertation has shown that accuracies with which a highspeed target can be tracked and the sightline quantified were dependent on many factors, not least on the training and capabilities of the human operator. Other potential sources of inaccuracy were the digital signal processing, transducer noise, drifts and other stabilization errors. These were analyzed and their effects noted. Simulation showed that the design could achieve the specifications set for the optical tracker.

ACKNOWLEDGEMENTS

The work put into this dissertation was done at Kentron (Pty) Ltd, and the author would like to express his gratitude to the company for supplying the time and materials necessary for bringing this study to a level not normally required for in-use analysis. Special mention must be made of the checking and evaluation of many of the results of Chapters 4 and 5 by Mr I Alers. Mr F Beyers devoted much of his time to checking the system equations of motion, and deserves thanks for the encouragement and support given the author, especially during design reviews.

Dr O Rubin pointed out many additional problem areas during the course of this study and thanks are owed to Mr G Quinn and the staff of the Engineering Department for their help in the mechanical design of the stabilized platform. In the debugging of many of the computer programs used to obtain results cited in this report, Mr O v d Schyff played a valuable part.

Throughout the period of this study, the excellent super= vision of Dr K Garner of the University was apparent. His understanding of the problems involved in producing this dissertation and the constructive criticism given during the execution of the study are valued highly.

Finally, getting down to producing the necessary paperwork, it must be reported that this task was made that much easier by the efficient typing of the text, the labelling of the figures and drawing up of tables by Mrs I de Beer. The author is most grateful to everyone concerned.

1 INTRODUCTION

Much research and development has been carried out in the field of radar trackers, and a number of schemes have been published which present aids and controllers for radar tracking systems (references 6, 10, 21, 24). Applications of controllers for optical trackers have not been widely published. Such optical trackers can be both back-up systems to a radar tracking unit, as well as tracking systems in their own right. This dissert= ation contains the development of a controller for an optical tracker unit for military purposes.

As the human operator is an integral part of the controller the optical tracker is potentially more versatile than a radar tracker. Some important operational decisions can be made by the human operator before and during tracking and during firing of coupled weapons, if any. Most radar tracker studies (e.g. Rue (reference 21)) point out the need for high accuracy tracking (and controllers to achieve this) in order to maintain accep= table hit probabilities when engaging modern, high-speed and highly manoeuverable aerial targets. The tracking accuracy achieved by the optical tracker must be comparable to that of radar trackers (better than 2mrad rms pointing vector error), and this dissertation contains the analysis of a controller required to achieve such accuracies.

The human operator is, as mentoned above, an advantageous feature to incorporate into the system. Unfortunately, it was all the more difficult to design a controller with optimum performance around the human operator due to the adaptability and unpredictability of the operator. A trained operator could partly compensate for neuromuscular and overall system delay by predicting target motion and controlling the tracker accor= dingly. On the other hand, he was also required to monitor extensively the performance of the system during a tracking task. The tracker could be mounted on a moving vehicle, thus itself moving relative to inertial space, so that even a sta= tionary target would appear to the operator to move. What is more, this base motion was of a random nature, so that it would be very difficult for the operator to compensate for it. It was therefore unlikely that his full attention would be

given to the tracking task, and it would be advantageous to create a system which functioned autonomously for short periods of time.

Taking into account the indications of a short prestudy (McFadyen (reference 14)), and using an estimated system accuracy requirement of 1,5mrad rms tracking error, it was decided that the sightline should be stabilized in inertial space. This assured that the operator would not see any relative motion when looking at a stationary target through the optical system, and he would see the true motion in space of a moving target. This required the incorporation of a stabilized optical element into the system, chosen to be a mirror mounted on a two-degree-of-freedom gimballed platform. By prudent choice of inertial components and by careful design of the stabilizer control loops, most of the rotational and linear base motion of the tracker could be compensated for.

The sightline was required to sweep through a full 360° in azimuth, but the overall design of the tracker base or pedestal precluded the rotation of the stabilized mirror through large angles. It was therefore necessary to rotate the whole pedestal, requiring additional drive and control.

The pedestal and the stabilized platform made up the basic system hardware. A simulator was constructed representing the tracker and the performance of the human operator was measured. Various tracking aids were tested, and change in performance characteristics were noted, until sufficient system accuracy was achieved.

It must be realized too that the optical tracker may be part of an air defence system, so that data would be required from the tracker for the purpose of aiming the weapons. This specifically was accurate target kinematic data (position and rates), which was used by the weapon control computer to find the aiming angle of the weapon. This study was concerned with such target data.

Here is not simply a complete design report of an optical tracker. Instead an attempt has been made to highlight and analyze problem areas, specifically human operator performance. Solutions are offered which are "good" in the sense that they

satisfy the engineering requirement that the system meet the specifications, be economical and robust. This study was carried out with the restrictions of cost, time and materials imposed by any engineering project in the manufacturing industry.

The third chapter provides a conceptual layout of the tracker, giving nomenclature and defining coordinate systems. Certain aerial target attack profiles, which are representative of actual attacks the tracker may encounter in a given operational situation, are discussed.

The fourth chapter contains the derivation of the gimbal equations of motion and a mathematical model of the stabilized platform used to design the stabilizer control loops. The pedestal servo control loop is also discussed.

The fifth chapter is concerned with the measurement of the performance of and the derivation of a model of the human operator. This data is used to find a tracking aid which improves the system performance. Simulator performance figures are obtained and the viability of the tracking aid is demonstrated. The aid is expanded into a linear filtering algorithm which enables the required target data to be obtained.

The aim of this study is to find the configuration and parameters necessary for an optical tracker, in the presence of base motion, to track accurately a high-speed manoeuvering target. These parameters are to form the basis for specifications for the construction of such a tracker.

2 LITERATURE SURVEY

Having enumerated the requirements of the study, a start must be made to find solutions to the problem. In general, it can be said that the problem is to quantify a line-of-sight in the direction of a target. By implication this is an optical system and may be simply a pair of binoculars trained on the target. This will allow the target to be observed, but no information about the accurate position of the target in space can be obtained. This can be achieved easily by mounting the pair of binoculars or telescope on a tripod and reading off azimuth and elevation angles relative to some arbitrary zero position. A theodolite would suffice, some being capable of angular resolution of up to 10μrad (0,0006°).

The stationary theodolite would be a viable solution to the problem but for two factors: the tracker is mounted on a moving base and it must be able to accurately track a fast-moving aerial target. Both factors cause serious degradation of performance, not because of any limitation of the optics, but because of the limited capabilities of the human operator (despite the advantages of his adaptability), discussed in McRuer (reference 15). Random base disturbances are impossible for the human operator to predict and therefore his performance deteriorates. Targets also execute manoeuvres at random intervals, causing lags in the tracking action of the operator. It is therefore necessary to concentrate on the design of a base motion isolation loop and tracking loop which will enable the tracker to function successfully.

Rue (reference 21), besides discussing other reasons why tracking systems need to be accurately stabilized, points out the need for determining mathematical relationships between the various optical and mechanical elements. He derives the equation of motion for a general, two-axis gimballed mirror and then goes on to find the block diagram for a particular tracking and stabilization system. His methodology has been applied to the tracker designed in this report with modifications to cater for differences in configuration, layout and instrumentation. He does not discuss the implementation of any tracking aid or filter other than that it is the input to the block diagram of the tracker dynamics.

Springarn and Weidemann (reference 24) suggest the use of a Linear Regression Filter to obtain estimates of target position in inertial space. They derive a classical 2nd order filter (expanding memory) and give performance relationships for non-manoeuvering targets, showing that these are better than those for a Kalman Filter (2nd order) in polar coordinates. For manoeuvering targets they suggest the implementation of a similar filter with truncation (fading memory). The performance of this filter is dependent on the truncation time.

Singer (reference 23) uses a different approach to obtain accurate target position. He derives a 3rd order Kalman Filter executed in inertial coordinates and assumes that the rate of change of acceleration follows a gaussian probability distribution function. Fitts (reference 6) then uses the work of Singer to define and illustrate the concept of aided tracking. The performance of a conventional tracking system is shown to improve with the application of either position or rate aided tracking.

The computer implementation of the algorithm for this study was, however, subjected to realtime execution limitations, the computer to be used having no firmware implementation of trigonometric functions. Measurements, made in line-of-sight coordinates, had to be transformed using software routines to the inertial frame of reference in order that any of the above-mentioned schemes could be applied.

This placed a very stringent execution time limit on the algorithm used; so much so that the above schemes were not able to be implemented.

Algorithms executed in line-of-sight coordinates are, however, available in the literature. Gholson and Moose (reference 10) and Moose (reference 16) derive a linearized 2nd order target model for use in a Kalman Filter, augmented by an estimated manoeuver command modelled as a semi-markov process. Possible acceleration inputs are selected according to some *a priori* manoeuver probability. This scheme was successfully implemented and found to be very versatile.

Similar work has been done by Ricker and Williams (reference 20) who compute a weighted average of possible target manoeuvre commands and update the filter. Chan et al (reference 4) derive a least squares estimator of manoeuvre command and use this to update the filter.

Another important aspect of this project was that man-in-the-loop tracking was implicit to the system. The studies mentioned above are aimed more at radar tracker applications where the tracking loop is not necessarily closed through the human operator.

McRuer (reference 15) gives a wide survey of problems encountered when designing around the human operator in vehicular control tasks. Specifically, the versatility of the human operator as an information processing device and the ability to act, with training, as time-optimal controller are mentioned. The implementation of any tracking aid for the optical tracker can therefore be tested and judged only through experimentation.

Thus the literature survey has served to point the way to a number of possible solutions to the optical tracker design problem. No ready-cut answers were supplied however, so that "bottom up" design principles had to be applied and the best solution derived using basic techniques.

3 SYSTEM DESCRIPTION AND NOMENCLATURE

3.1 Introduction

In general, an optical tracker or optical director could have many different shapes and structures. Each structure will be best determined by the particular operational requirement of the director, and each particular operational requirement will dictate a set of performance requirements. The requirement specified for this study was that the director track a wide spectrum of targets (see below) which need high tracking rate capability with good stability characteristics. This demanded high resolution optics and led to the decision to use a mirror as the stabilized optical element. The mirror was large and heavy, so that special attention had to be given to the stabilization loop design. The mechanical layout of the gimbals was also determined by packaging considerations, as the stabilized optics had to be mounted on a rotatable pedestal.

The basic structure of the optical director is described below, performance limits given and axis systems and nomenclature defined. Target attack profiles are chosen and described, so that tracker performance can be evaluated.

3.2 The Optical Director

The optical director, shown schematically in Figure 3.1, is basically a rotating platform on which the human operator sits and on which the sighting system and supporting mechanics and electronics are mounted. The platform, or pedestal, can rotate through 360° about the k-axis relative to the base, which itself may or may not move with respect to inertial space. Axes and nomenclature are indicated on the figure.

3.3 The Stabilized Group

This unit is mounted on the optical director and contains the sensitive instruments needed for measuring the motion in inertial space of the sightline. Shown in Figure 3.2, the unit consists of a gimbal which moves the sightline in elevation from -25° (down elevation) to +85° (up elevation) relative to the pedestal, and a stable platform or cluster to which a mirror is attached (the stabilized optical element). The mirror moves the sightline through 20° in azimuth. The maximum sightline

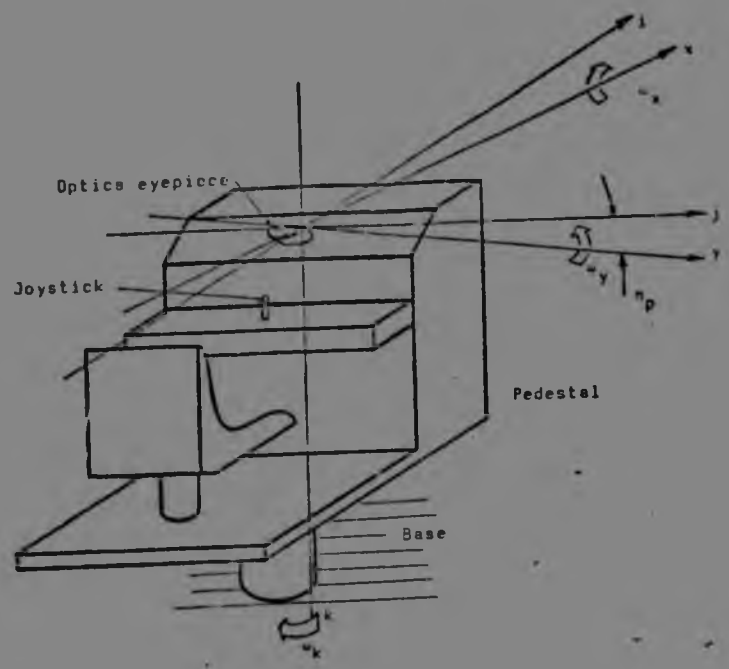

FIGURE 3.1 SCHEMATIC DIAGRAM OF OPTICAL DIRECTOR

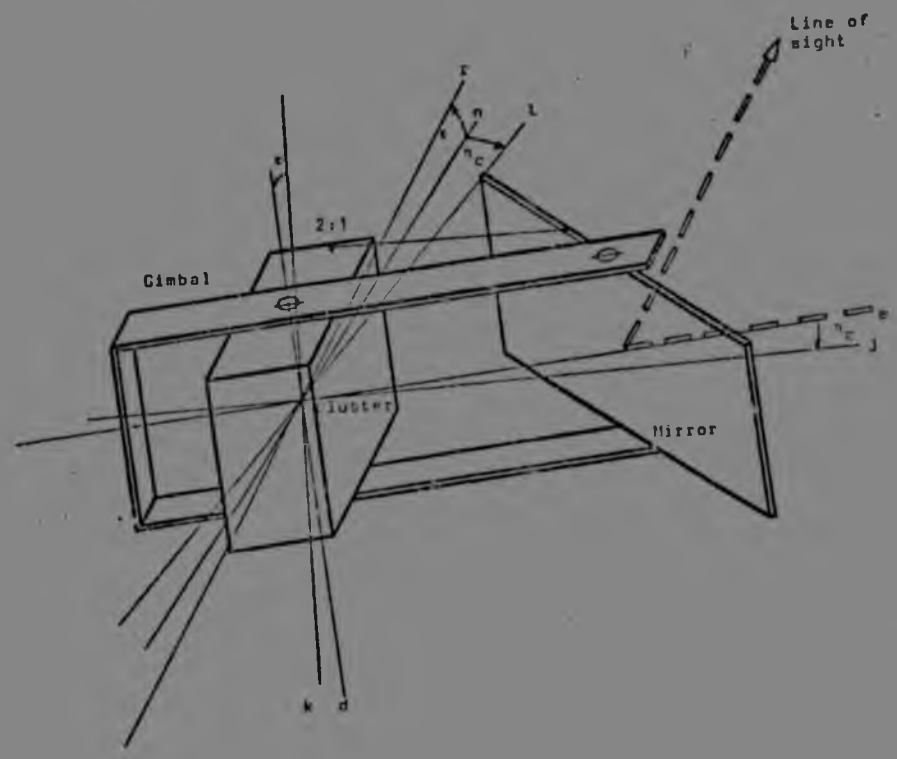

FIGURE 3.2 SCHEMATIC DIAGRAM OF STABILIZED GROUP

turning rate is 2,3rad/s (130deg/s) and the maximum angular acceleration is 2,5rad/s² 150deg/s²). Axes and nomenclature are indicated on the figure.

3.4 Target Profiles

The optical director is in general used as part of some surface defence installation. Attacks on this installation can be carried out by personnel, ground vehicles, helicopters, aircraft and missiles. This represents a wide spectrum of possible target manoeuvres which would require a prohibitive amount of time to investigate. It is however possible to select some typical target manoeuvres which would sufficiently test the tracker and allow conclusions to be made about its performance with respect to the whole spectrum of targets. Bearing in mind then that critical parameters for a tracker are tracking speed and stability, target manoeuvre selection was made to give high sightline turning rates (aircraft) and zero sightline turning rate but maximum range rate (approaching missile).

3.4.1 Medium Toss Bombing Attack

This is shown in Figure 3.3 and occurs during a bombing attack after which the aircraft executes an evasive manoeuvre. For the purpose of this analysis the aircraft was always tracked so that tracker ability in azimuth as well as elevation was evaluated. This was useful also for testing operator coordination.

3.4.2 Dive Attack

This is shown in Figure 3.4 and represents a rocket or strafing attack. As the aircraft passes overhead, very high sightline turning rates result, representing a worst case. Structural limits on the tracker prevent it from being able to look vertically up, nevertheless it is a useful point for comparitive tracking loop evaluation.

3.4.3 Missile Attack

The missile is assumed to fly straight at the tracker at 500m/s (Mach 1,7) at a constant altitude of 10m. Sightline turning rates were therefore zero, allowing the stability of the tracking loops to be tested. It also represented a worst case for the range estimator due to the high approach velocity.

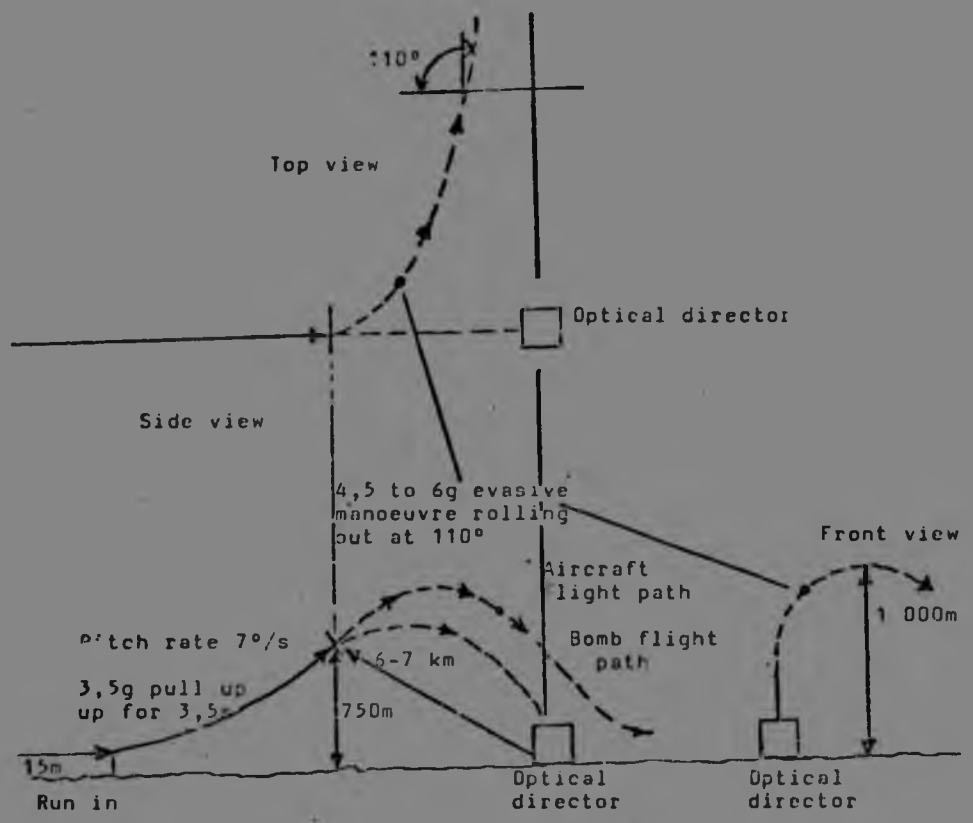

FIGURE 3.3 MEDIUM TOSS BOMB ATTACK

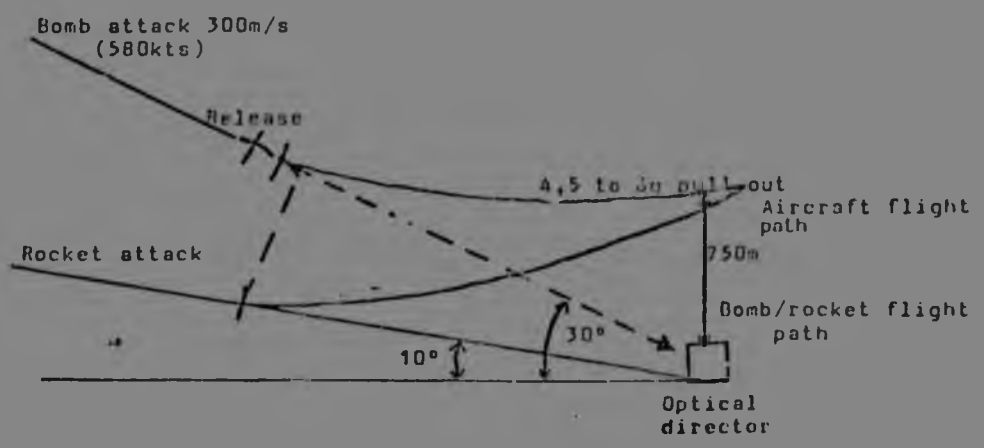

FIGURE 3.4 DIVE ATTACK

The system has now been defined to a degree that allows the equations of motion to be derived. This will be done in the following chapter.

4 SIGHTLINE STABILIZATION

4.1 Introduction

In Chapter 1 and the Literature Survey it was pointed out that in order to achieve the accuracies necessary to shoot down modern, high-speed aircraft, electro-optical pointing systems should be stabilized in inertial space. In this chapter base motion stabilization is discussed.

The motion of the base was divided into two components, namely rotational and linear motion. The approach taken in stabilizing each was different. First, however, the equations of motion for the system were derived and implemented according to the actual mechanical structure. The control loops were then designed and evaluated, and it is then shown that the proposed stabilization of the target in inertial space was achieved.

4.2 Stabilized Group Equations of Motion

In order to find the transfer function of the stabilized group, the equations of motion had to be known. First the rotational motion of a body in inertial space was considered (Figure 4.1). By Newton's Second Law, the torque acting on the body was given by

$$\underline{L} = \left(\frac{d}{dt}\right)_s \underline{H} \qquad 4.1$$

where $\underline{H}$ = the angular momentum of the body,
and s = the reference to a stationary (inertial) frame of reference.

The sum of the torques on the body was represented as

$$\Sigma \underline{L}_i = 0 \qquad 4.2$$

and

$$\underline{H} = (\underline{r} \times \underline{v})M \quad \text{or} \quad \underline{H} = \underline{r} \times (\underline{\omega} \times \underline{r})M \qquad 4.3$$

If the body had a Moment of Inertia $\underline{\underline{J}}$, then (Halfman (reference 12), and Thomson (reference 26))

$$\underline{H} = \underline{\underline{J}}\underline{\omega} \qquad 4.4$$

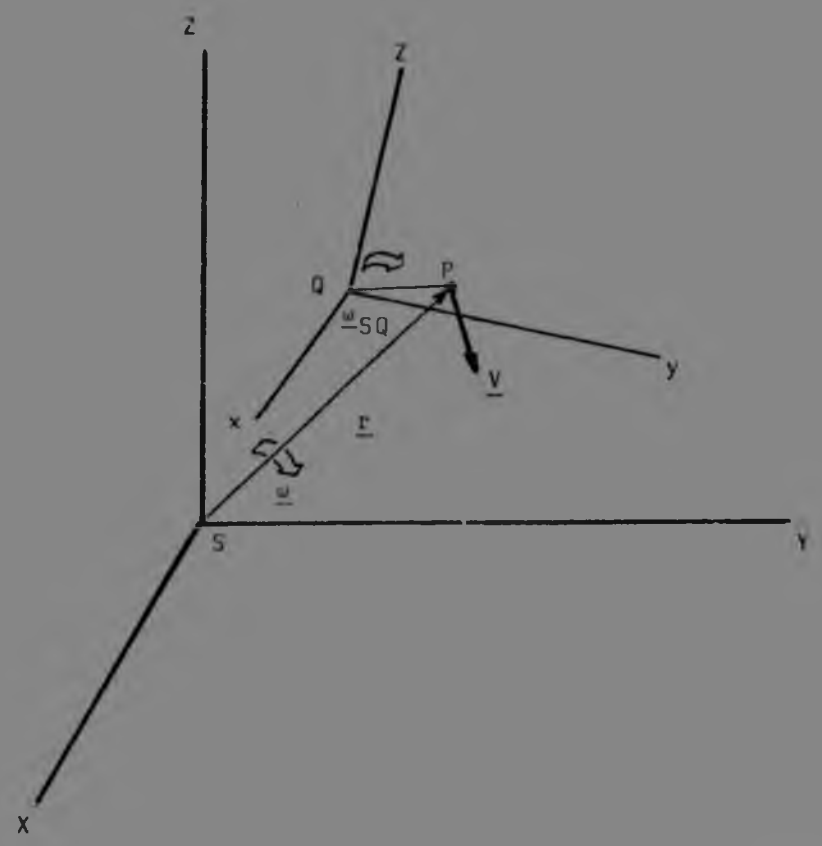

FIGURE 4.1 MOTION OF A BODY IN SPACE

The equation of Coriolis (Wrigley, et al (reference 28)) was also applicable. This equation relates the total change of state of a point in a rotating axis system relative to a fixed axis system, with the change of state of the point relative to the rotating axis system. It was formulated as follows:

$$(\frac{d}{dt})_S \underline{a} = (\frac{d}{dt})_Q \underline{a} + \underline{\omega}_{SQ} \times \underline{a} \qquad 4.5$$

$\underline{a}$ = general vector describing the state after point

$(\frac{d}{dt})_S$ = derivative in the S coordinate system

$(\frac{d}{dt})_Q$ = derivative in the Q coordinate system

$\underline{\omega}_{SQ}$ = rotation of the Q coordinate system relative to the S coordinate system

Referring to Figure 3.2, and using the equations given above, the equations of motion for the stabilized group could be found.

The following simplifying assumptions could be made, however. Consider the cluster and mirror to be one mechanically rigid unit with centre of gravity on the intersection of the axes. Similarly, let the centre of gravity of the gimbal be at the origin of the axis system. The derivation was then as follows. Let the inertia sensor of the cluster and gimbal be respectively given by

$$\underline{J}_C = \begin{pmatrix} J_R & J_{re} & J_{rd} \\ J_{re} & J_E & J_{ed} \\ J_{rd} & J_{ed} & J_{CD} \end{pmatrix}$$

and

$$\underline{J}_D = \begin{pmatrix} J_N & J_{nj} & J_{nd} \\ J_{nj} & J_J & J_{jd} \\ J_{nd} & J_{jd} & J_{DD} \end{pmatrix}$$

The torque exerted on the cluster by the drive was represented by

$$\underline{L}_{CD} = \begin{pmatrix} 0 \\ 0 \\ L_{Dd}-F_d \end{pmatrix} \qquad 4.6$$

where L_{Dd} = the drive torque (d-axis only)
and F_d = the friction constant

and the torque exerted on the gimbal by the drive by

$$\underline{L}_{OD} = \begin{pmatrix} 0 \\ L_{Dj}-F_j \\ 0 \end{pmatrix} \qquad 4.7$$

where L_{Dj} = the drive torque (j-axis only)
and F_j = the friction constant.

Using equation 4.4, the angular momentum of the cluster could be written as

$$\underline{H}_{IC} = \underline{\underline{J}}_C \underline{\omega}_{IC} \qquad 4.8$$

where $\underline{\underline{J}}_C$ = inertia of stable cluster
and $\underline{\omega}_{IC}$ = rotation rate of stable cluster with respect to inertial coordinate system

Incorporating equation 4.1, the sum of the torques was given by

$$\underline{L}_{IC} = \left(\frac{d}{dt}\right)_I \underline{H}_{IC} \qquad 4.9$$

and using the equation of Coriolis

$$\underline{L}_{IC} = (\underline{\omega}_{IC} \times \underline{\underline{J}}_C \underline{\omega}_{IC}) + \underline{\underline{J}}_C \dot{\underline{\omega}}_{IC} \qquad 4.10$$

This was the vector form of the cluster equation of motion. Similarly, the vector form of the equation of motion of the gimbal was given by

$$\underline{L}_{IO} = \left(\frac{d}{dt}\right)_I \underline{H}_{IO} = (\underline{\omega}_{IO} \times \underline{\underline{J}}_O \underline{\omega}_{IO}) + \underline{\underline{J}}_O \dot{\underline{\omega}}_{IO} + \underline{L}_{IC} \qquad 4.11$$

where $\underline{\underline{J}}_O$ = inertia of gimbal
and $\underline{\omega}_{IO}$ = rotation rate of gimbal with respect to inertial coordinate system

There were three distinct, independent rotation rates involved in this study. These were the rotation of the pedestal with respect to inertial space, the rotation of the gimbal relative to the pedestal (about the j-axis), and the rotation of the cluster relative to the gimbal (about the d-axis). Their relationship was as follows:

$$\underline{\omega}_{IO} = \underline{\omega}_{IP} + \begin{pmatrix} 0 \\ \dot{\epsilon} \\ 0 \end{pmatrix} \qquad 4.12$$

$$= \underline{\omega}_{IP} + \underline{\dot{\epsilon}}$$

where $\underline{\omega}_{IP}$ = the rotation rate of the pedestal in inertial space

and $\underline{\dot{\epsilon}}$ = the rotation rate of the gimbal relative to the pedestal.

$$\underline{\omega}_{IC} = \underline{\omega}_{IO} + \begin{pmatrix} 0 \\ 0 \\ \dot{\eta}_c \end{pmatrix} \qquad 4.13$$

$$\underline{\omega}_{IC} = \underline{\omega}_{IO} + \underline{\dot{\eta}}_c$$

where $\underline{\dot{\eta}}_c$ = the rotation rate of the cluster relative to the gimbal

The following coordinate transformations could also be defined. From the pedestal (i,j,k) axis system to the gimbal axis system (n,j,d)

$$\underline{\underline{T}}_{OB} = \begin{pmatrix} \cos\epsilon & 0 & -\sin\epsilon \\ 0 & 1 & 0 \\ \sin\epsilon & 0 & \cos\epsilon \end{pmatrix} \qquad 4.14$$

and from the gimbal axis system (n,j,d) to the cluster axis system (r,e,d)

$$\underline{\underline{T}}_{CO} = \begin{pmatrix} \cos\eta_c & \sin\eta_c & 0 \\ -\sin\eta_c & \cos\eta_c & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad 4.15$$

The above vector equations of motion (4.10 and 4.11) were useful only in the respective free rotation coordinates (the d- and j-axes). Using equations 4.12 to 4.15, they were expanded as follows:

$$\underline{u}_d \cdot \underline{\underline{J}}_C[(\underline{\underline{T}}_{OB}\dot{\underline{\omega}}_{IB}+\ddot{\underline{\varepsilon}}) + \ddot{\underline{\eta}}_c] = L_D$$
$$-\underline{u}_d \cdot [(\underline{\underline{T}}_{CO}(\underline{\underline{T}}_{OB}\underline{\omega}_{IB}+\dot{\underline{\varepsilon}}) + \dot{\underline{\eta}}_c) \times \underline{\underline{J}}_C(\underline{\underline{T}}_{CO}(\underline{\underline{T}}_{OB}\underline{\omega}_{IB}+\dot{\underline{\varepsilon}}) + \dot{\underline{\eta}}_c)] \quad 4.16$$
$$-F_d\ddot{\eta}$$

$$\underline{u}_j \cdot [\underline{\underline{J}}_O(\underline{\underline{T}}_{OB}\dot{\underline{\omega}}_{IB}+\ddot{\underline{\varepsilon}})] + \underline{u}_j \cdot [\underline{\underline{J}}_C(\underline{\underline{T}}_{OB}\dot{\underline{\omega}}_{IB}+\ddot{\underline{\varepsilon}}+\underline{\underline{T}}_{OC}\ddot{\underline{\eta}}_c)]$$
$$= L_{JD} - \underline{u}_j \cdot [(\underline{\underline{T}}_{OB}\underline{\omega}_{IB}+\dot{\underline{\varepsilon}}) \times \underline{\underline{J}}_O(\underline{\underline{T}}_{OB}\underline{\omega}_{IB} + \dot{\underline{\varepsilon}})]$$
$$-\underline{u}_j \cdot [(\underline{\underline{T}}_{OB}\underline{\omega}_{IB}+\dot{\underline{\varepsilon}} + \underline{\underline{T}}_{OC}\dot{\underline{\eta}}_c \times \underline{\underline{J}}_C(\underline{\underline{T}}_{OB}\underline{\omega}_{IB} + \dot{\underline{\varepsilon}} + \underline{\underline{T}}_{OC}\dot{\underline{\eta}}_c)]$$
$$-F_j\ddot{\varepsilon}$$

$$\underline{u}_j \cdot \underline{\underline{J}}_O(\underline{\underline{T}}_{OB}\dot{\underline{\omega}}_{IB} + \ddot{\underline{\varepsilon}})] + \underline{u}_j \cdot [\underline{\underline{J}}_C(\underline{\underline{T}}_{OB}\dot{\underline{\omega}}_{IB} + \ddot{\underline{\varepsilon}} + \underline{\underline{T}}_{OC}\ddot{\underline{\eta}}_c)] \quad 4.17$$
$$= J_{jT}\dot{\omega}_{oj}$$

where J_{jT} = the total moment of inertia of the gimbal and cluster, which could be found by applying the definition of moment of inertia

$$J = \int x^2 dm$$
$$J_{jT} = J_J + J_E\cos^2\eta + J_R\sin^2\eta - J_{re}\sin 2\eta \quad 4.18$$

These equations fully described the motion of the gimbal and cluster in response to drive torques. They contained the necessary cross-coupling terms so that, together with the inclusion of stiction and mechanical and electrical endstops, they could be used in a non-linear model of the system. By making suitable assumptions, these equations also lent themselves to linearization, so that linear control system design techniques could be applied.

4.3 Mechanical Design of the Stabilized Group

It is not intended to give a detailed description of the mechanical design of the stabilized group here. A sketch of the layout is shown in Figure 4.2. It is however worth nothing that the mechanical layout of the components has been done to minimize the crossproducts of inertia (using the Grey optimization algorithm, Engelbrecht (reference 5)) and therefore the cross-coupling between the axes, and to ensure that the centre of gravity lies on the rotation axes. The assumptions made above in the derivation of the equations of motion were therefore valid.

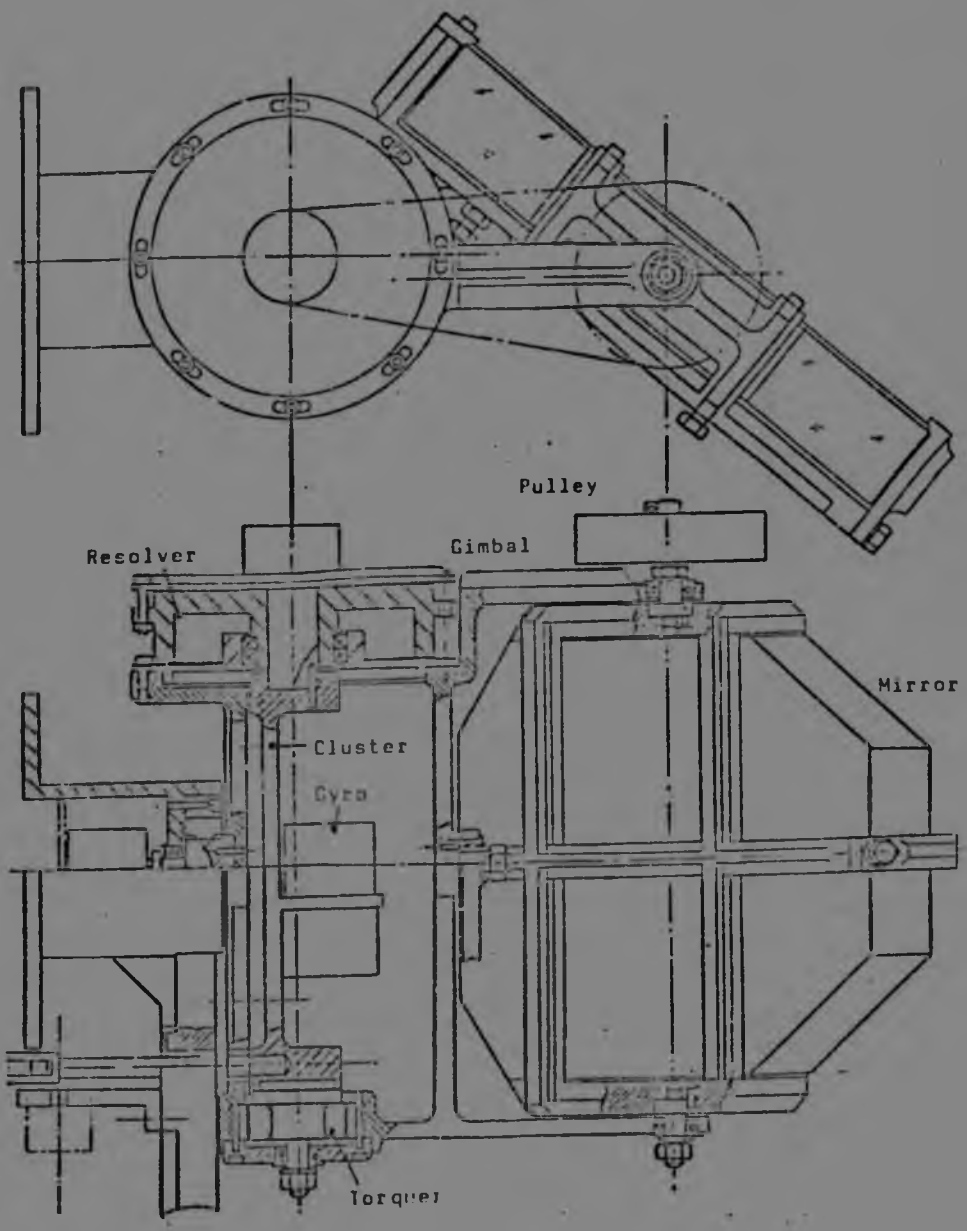

FIGURE 4.2 GIMBAL AND CLUSTER MECHANICAL LAYOUT

The inertia tensors were calculated from the mechanical layout, and were as follows:

Cluster inertias $J_R = 0,0189 kgm^2$
$J_E = 0,0218 kgm^2$
$J_{CD} = 0,0069 kgm^2$
$J_{re} = 0,0032 kgm^2$
$J_{ed} = 0,0$
$J_{rd} = 0,0$

Gimbal inertias $J_N = 0,0036 kgm^2$
$J_j = 0,0022 kgm^2$
$J_{OD} = 0,0021 kgm^2$
$J_{nj} = 0,0$
$J_{jd} = 0,0$
$J_{nd} = 0,0$

These values showed that the cross-coupling between the cluster and the gimbal was small, so that, in linearizing the equations of motion, the cluster and gimbal could be analyzed seperately without introducing too large an error.

Given the components to be used in the construction of the cluster and gimbal, as well as the results of some measurements, an estimate of the friction/stiction model required could be made. This is shown in Figure 4.3.

4.4 Vehicle Motion Amplitude Spectra

In order to evaluate the performance of the stabilization loops, an idea was required of the motion of the vehicle on which the pedestal was mounted. This spectrum of motion could be divided conveniently into two parts, (i) that which resulted from the passage of the vehicle over a particular terrain or through a series of manoeuvres (low frequencies generally), and (ii) that which resulted from vehicle structure vibrations, produced for example by the vehicle engine (generally high frequencies).

For this particular project, the spectral density of the low frequency motion (below 2rad/s) was specified. For comparison purposes, this had to be transformed to an amplitude spectrum. This was done as follows (Papoulis (reference 14)):

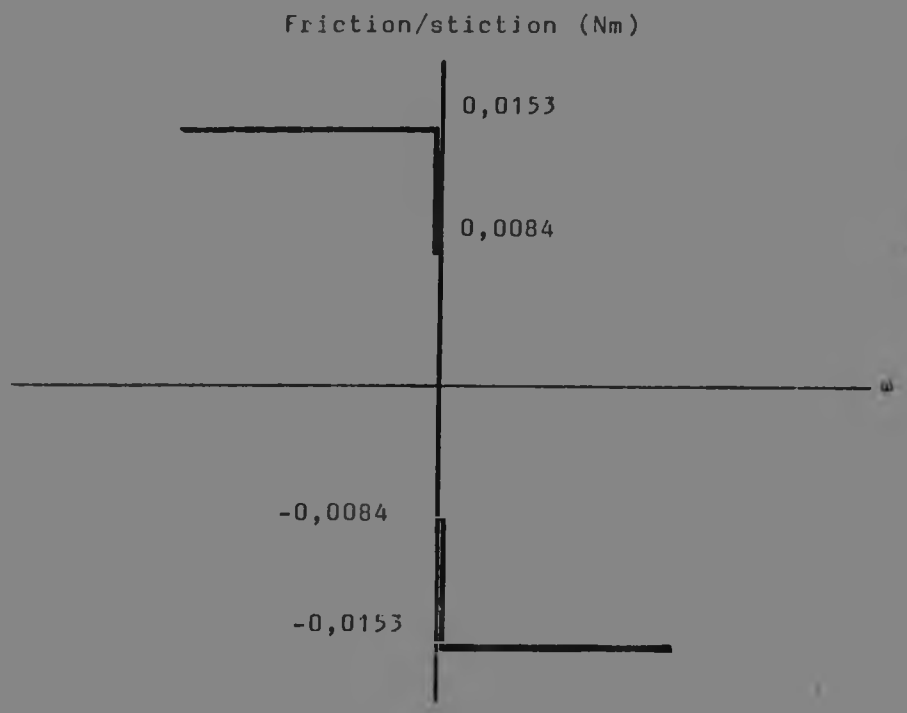

FIGURE 4.3 FRICTION/STICTION MODEL

The spectral density was defined as

$$G(f_k) = \frac{A^2(f_k)}{2\Delta f} \qquad 4.19$$

where $\Delta f = 1,T$

and T was the period over which the Fourier Transform was evaluated. $A(f_k)$ was the value of the amplitude spectrum at frequency f_k. Using the values supplied, the ampliture spec= trum was drawn and is shown in Figure 4.4, together with the spectrum at high frequencies.

Using the above data, it was also possible to find an approximate deterministic or random vehicle motion in the time domain. A simple Fourier cosine series expansion was used :

$$a(t) = \sum_{i=1}^{n} A(f_i) \cos\omega_i t \qquad 4.20$$

where $\omega_i = 2\pi f_i$

This gave a deterministic signal. In order to obtain a random signal, a random phase-shift could be incorporated into the above relation.

4.5 Stabilization Loop Design and Analysis

In the preceding paragraphs the equations of motion for the gimbal and cluster were derived. A sketch of the mechanical layout was presented which showed that cross-products of inertia could be minimized, the system statically balanced, and estimates of friction and stiction obtained. Next an energy spectrum of the estimated tracker base motion was presented and it was shown how a deterministic time signal could be derived, which was useful in evaluation of the stabilization loops. All tools were then available, so the loops could be designed.

The feature of the stabilization loops for rotational motion of the base, was the choice of a rate integrating gyroscope as prime sensing instrument. This had, as a byproduct, encouraged the development of the advanced technology necessary to process the gyro output signals. The hardware involved will not be discussed here, but the design of the compensator transfer functions for the stabilization loops will be presented. Upon

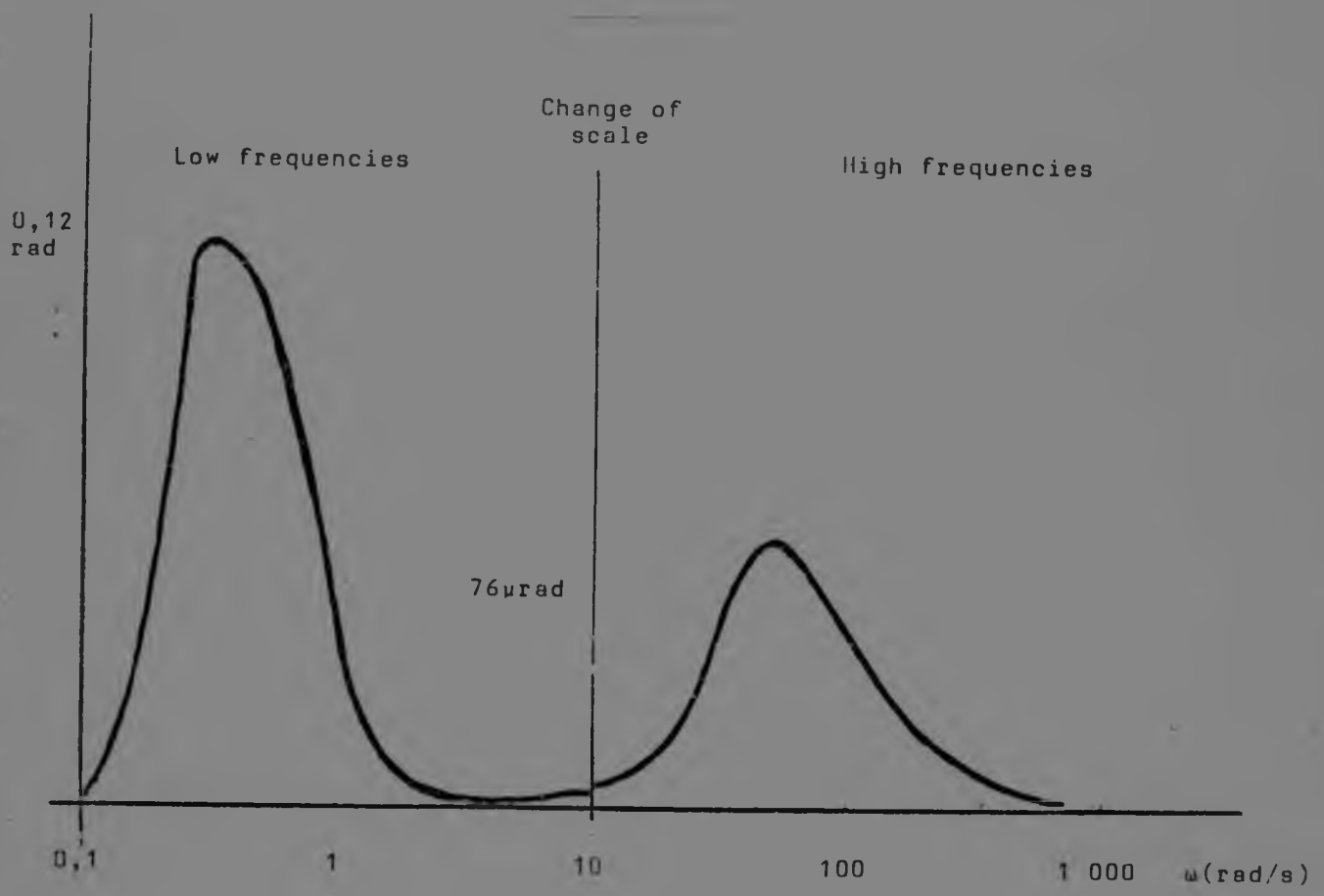

FIGURE 4.4 BASE MOTION AMPLITUDE SPECTRUM

linearization of the loop components, elementary control system design techniques could be applied, but the design was checked using a non-linear model and simulation techniques.

The stabilization loops for linear motion of the base will be discussed in the next paragraph.

4.5.1 Linear Control Loop Design

The equations of motion (equations 4.16 and 4.17) were used to model the gimbal and cluster dynamics, assuming that the cross-products of inertia could be neglected and that the friction torque was proportional to the rotation rate. The gimbal and cluster could therefore be decoupled, and adding a linearized model of the torquer and gyro, could be represented in block diagram form as shown in Figure 4.5. The values of the constants in the diagram are

Gyro pickoff constant	PK	=	859,5V/rad
Torquer constant	TKK	=	0,031Nm/A
Torquer resistance	R	=	4,3Ω
Torquer back emf	EMF	=	0,029V/rad/s
Cluster/gimbal and torquer inertia	DJ	=	0,02kgm²
Friction constant	BD	=	0,005Nms

There were two design criteria to be considered: Firstly, the system had to react quickly and without oscillation to an input step command. Secondly, and this was the prime duty of the stabilization loop, the base motion had to be well isolated. The decision on the loop constants to be used was therefore based on whether satisfactory base motion isolation was obtained.

Using a standard analysis technique, Mason's Analysis (Alers (reference 1)), the open loop transfer function was found and used to find a compensator transfer function which would result in a closed loop transfer function having characteristics satisfying the above criteria. The compensator was of the form

$$\text{GAIN}\ \frac{(1+\tau_{z1}s)(1+\tau_{z2}s)}{(1+\tau_{p1}s)(1+\tau_{p2}s)}$$

and the values of the gain and time constants were

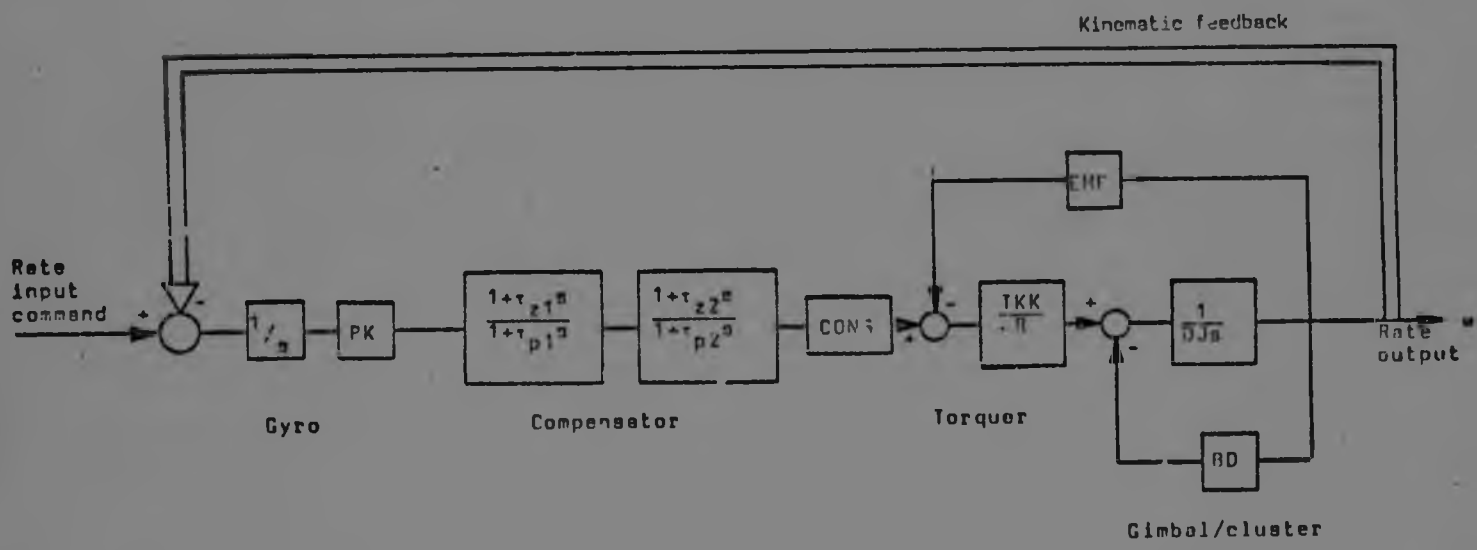

FIGURE 4.5 GIMBAL/CLUSTER DYNAMICS BLOCK DIAGRAM

τ_{z1} = 0,050s
τ_{p1} = 0,010s
τ_{z2} = 0,005s
τ_{p2} = 0,0014s
CONS 15V/V

The Root Locus diagram representing the transfer function from input to output, shown in Figure 4.6, illustrated that the loop remained critically damped for all values of loop gain. A very fast response could therefore be obtained without the loop becoming unstable. It must be pointed out here, however, that the gyro demodulator transfer function was omitted in this analysis. It consisted of a pole pair with natural frequency of 3 267rad/s and damping ratio 0,5 which become unstable for very high loop gains. The closed loop position of this pair was

$$-1\ 633{,}6 \pm j2\ 829{,}6 \text{rad/s}$$

for the above loop gain.

The disturbance response transfer function (from torquer (rate disturbance) to output angle) was also found and the frequency response plotted, shown in Figure 4.7. The good low frequency disturbance isolation, due to the gyro integrator, was clearly illustrated. The peak in the curve occured at the loop natural frequency, and represented the frequency at which disturbance motion isolation was at its worst. Given the vehicle (disturbance) motion spectrum of Figure 4.4, the net motion of the output could be found. This is shown in Figure 4.8, and it can be seen that the stabilized system output (solid line) is well below the required output (dashed line). These results are for the linearized loop, and although the non-linear simulation results were worse, the stabilization still met the requirements.

4.5 2 Non-linear Loop Sensitivity Analysis

Based on the block diagram of the linear loop (Figure 4.5), a non-linear model of the stabilization loop was constructed. The non-linearities consisted of a 7,5V limit on the gyro pickoff voltage, a 15,7V limit on the torquer input voltage, a 3,7V limit on the torquer current, the use of the nonlinear

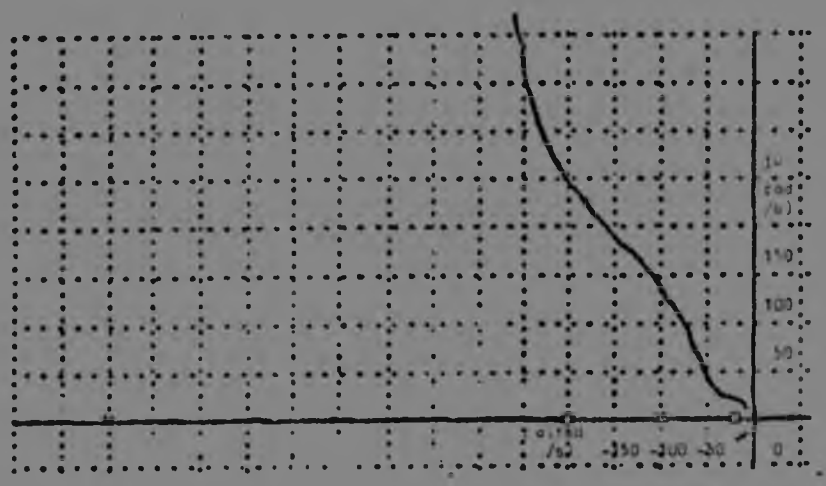

FIGURE 4.6 ROOT LOCUS PLOT

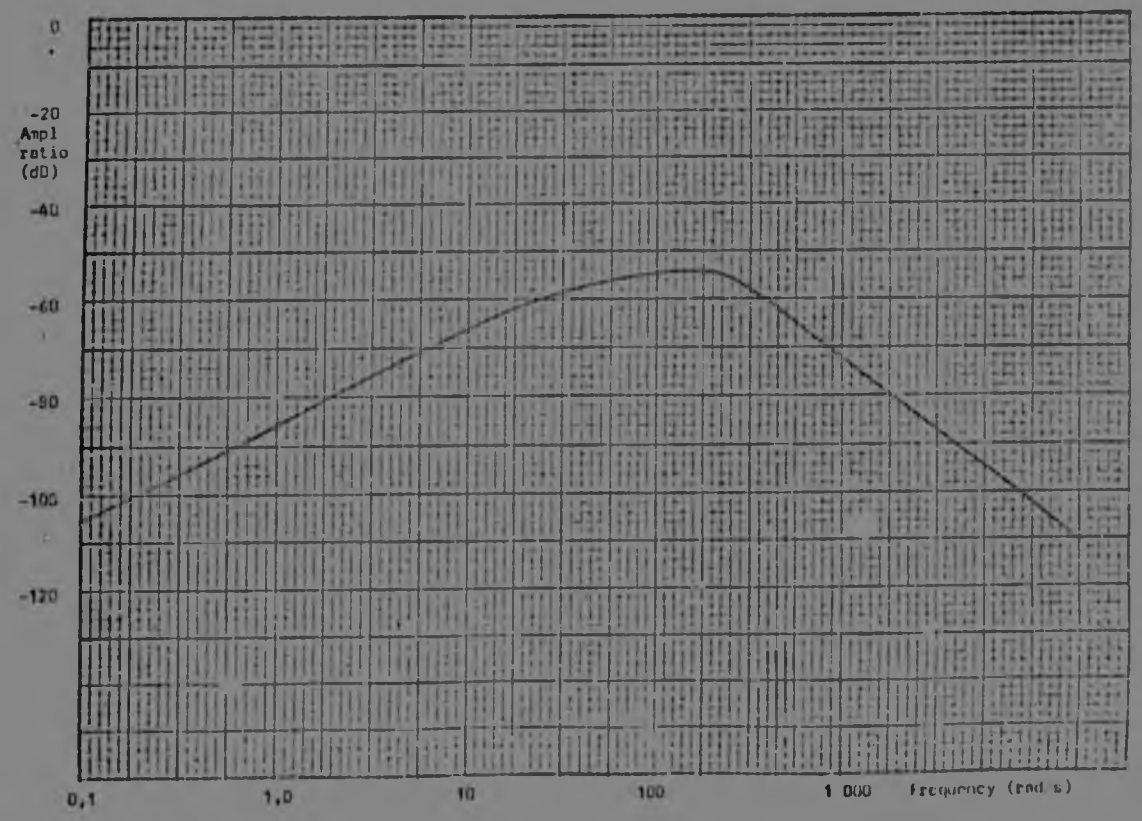

FIGURE 4.7 LINEAR MODEL BODE PLOT (DISTURBANCE RATE RESPONSE)

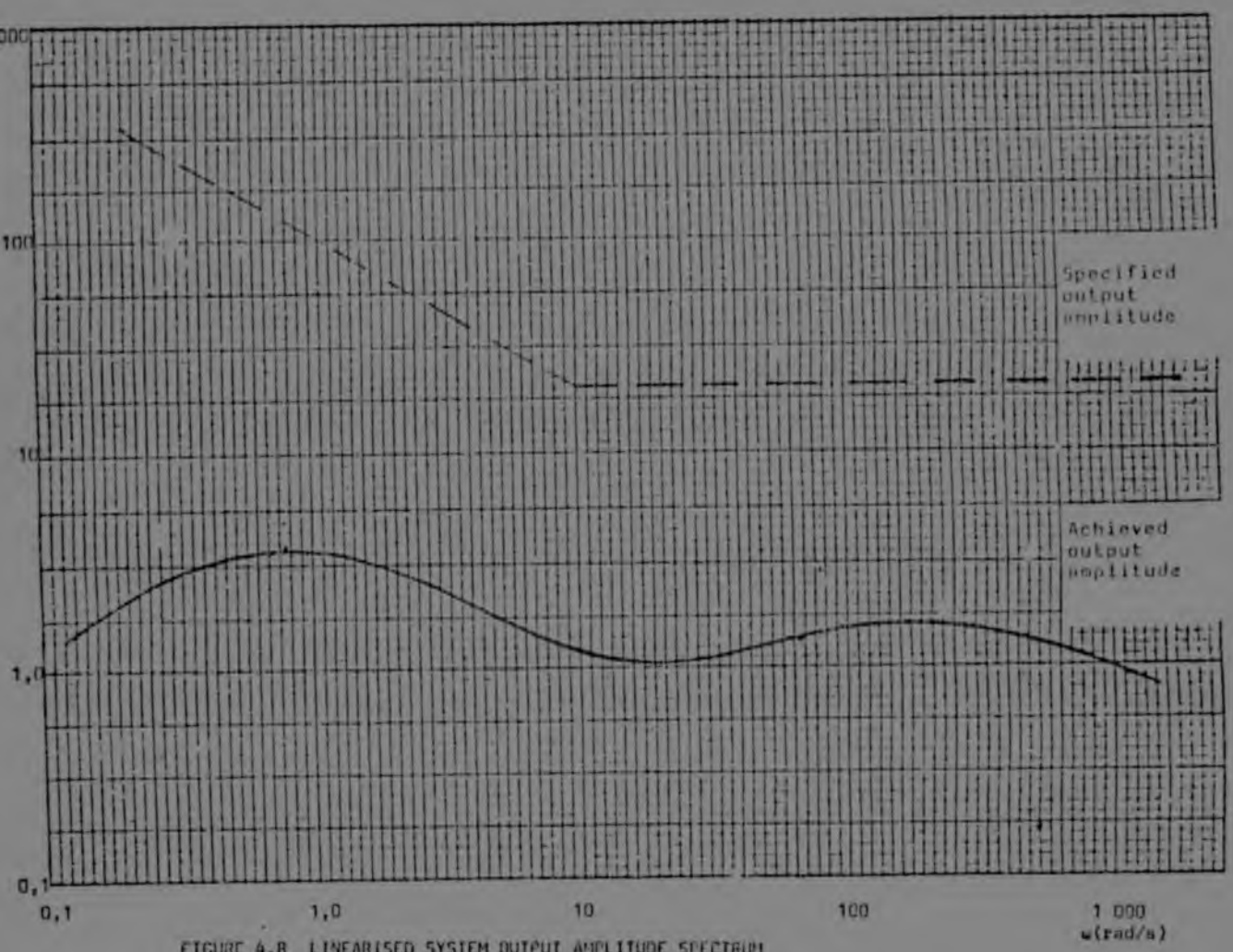

FIGURE 4.8 LINEARISED SYSTEM OUTPUT AMPLITUDE SPECTRUM

friction model (Figure 4.3) and the incorporation of the inertia cross-coupling terms. The cluster and gimbal stabilization loops were then no longer identical.

Subjecting the loops to sinusoidal inputs and disturbances and measuring the output amplitude (ie. the cluster and gimbal motion), the frequency responses shown in Figures 4.9, 4.10, 4.11 and 4.12 could be obtained. These show that the loops satisfy the design criteria.

The step response of the two loops is also shown in Figure 4.13 and 4.14. The difference in the responses was due mainly to the difference in the cluster and gimbal inertias.

For obvious reasons, the model implemented only approximated the real system. It was therefore desirable to know the sensitivity of the results from the model to model parameter inaccuracies.

Using low frequency base motion disturbance (see Figure 4.4) with no command input to the loops, the variation of stabilization error with variation of model parameters was investigated. The effects observed will be summarized as follows.

Due to stiction, the system developed an initial angle, which could be 0,2mrad or more. Thereafter the variation was smaller than 0,1mrad. Because this analysis was done in a "hands off" mode, the large initial angular error was not corrected. In the real system, it was expected that the human operator would zero the initial bias due to stiction, if it occured.

The error, apart from the bias, was not very sensitive to changes in the value of stiction, except when the value exceeded that of friction, when the error increased markedly.

A 30% increase in friction doubled the stabilization angular error.

A 30% decrease in friction reduced the stabilization angular error by 25%.

An increase of 10% in inertia reduced the stabilization angular error by 20%.

A decrease in inertia of 10% increased the stabilization angular error by 20%.

This indicated that the results of the model should be accurate to ± 20%, assuming a 10% uncertainty in the value of the parameters.

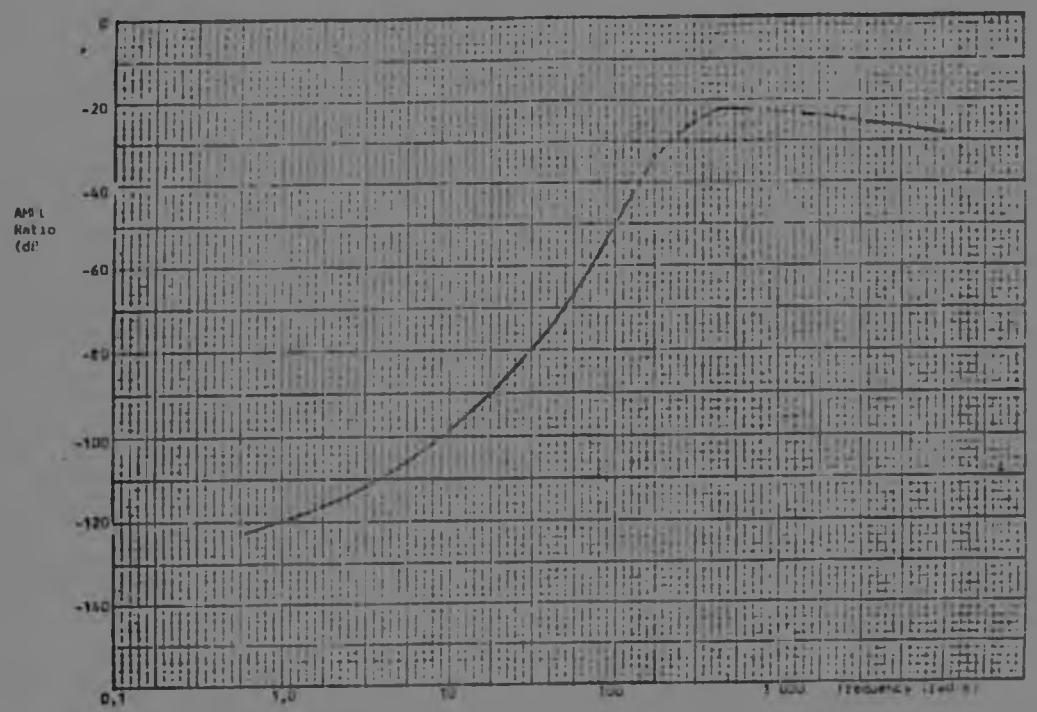

FIGURE 4.9 CLUSTER BODE PLOT- DISTURBANCE RATE RESPONSE (NON-LINEAR MODEL)

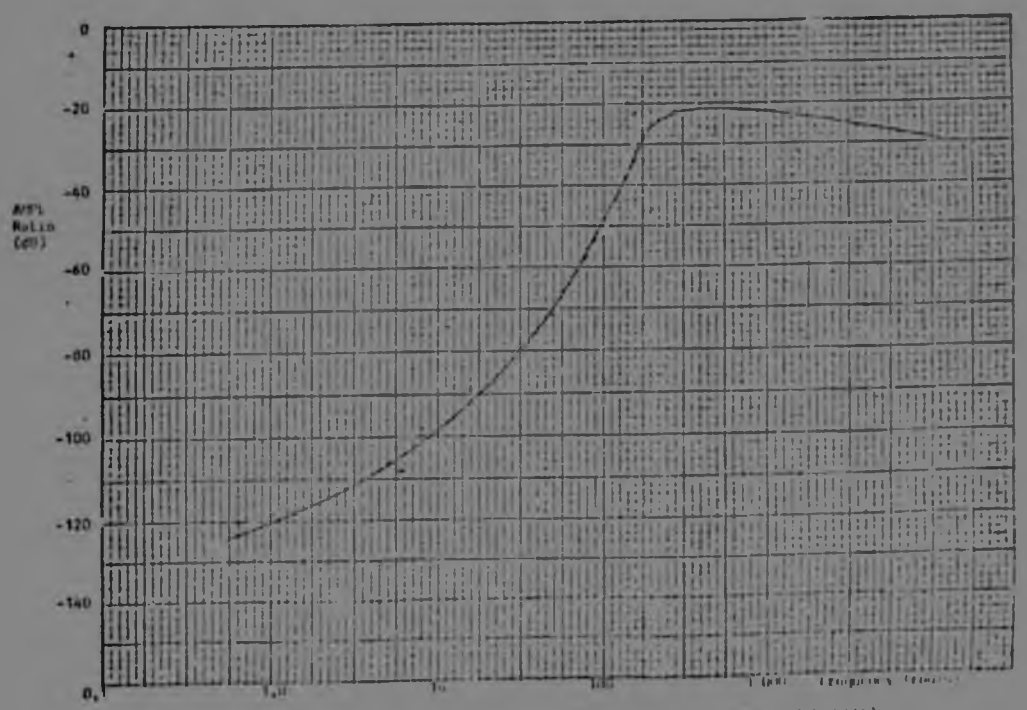

FIGURE 4.10 GLOBAL BODE PLOT- DISTURBANCE RATE RESPONSE (NON LINEAR MODEL)

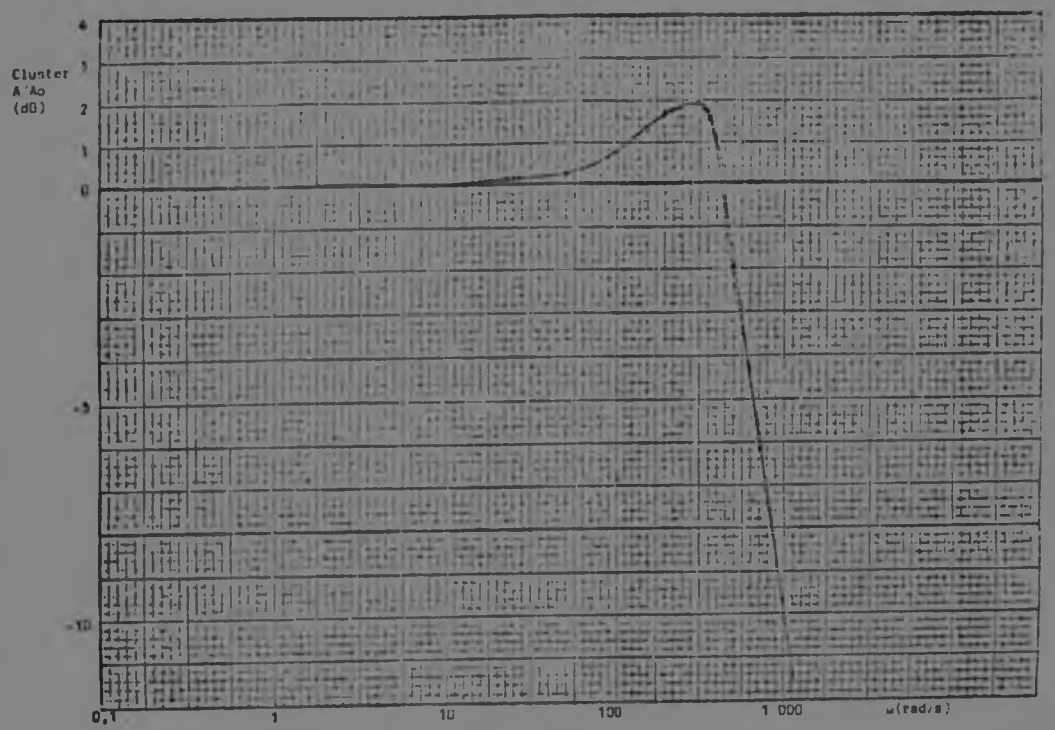

FIGURE 4.11 CLUSTER INPUT FREQUENCY RESPONSE (NON-LINEAR MODEL)

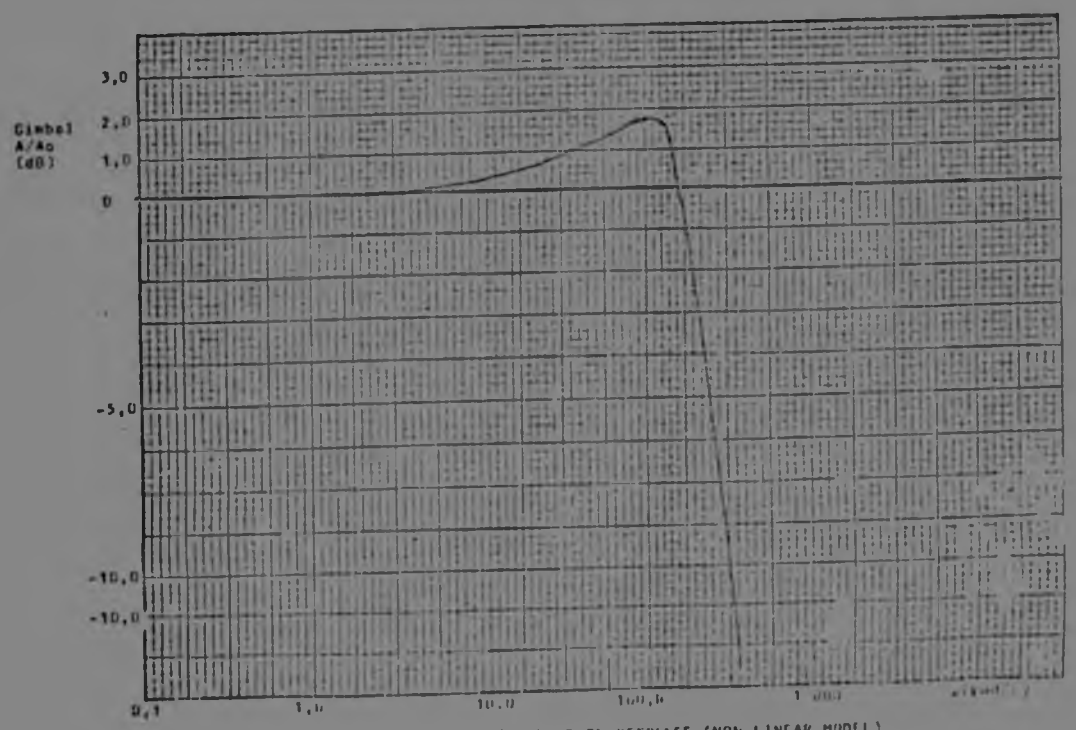

FIGURE 4.12 GIMBAL INPUT FREQUENCY RESPONSE (NON-LINEAR MODEL)

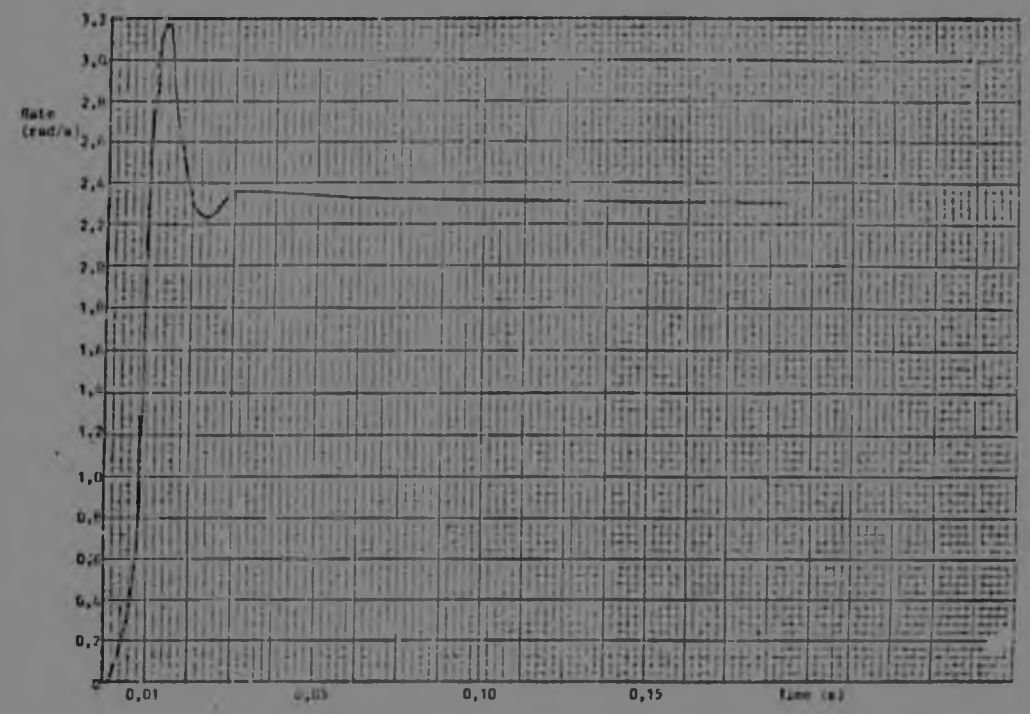

FIGURE 4.13 - CLUSTER STEP INPUT RESPONSE

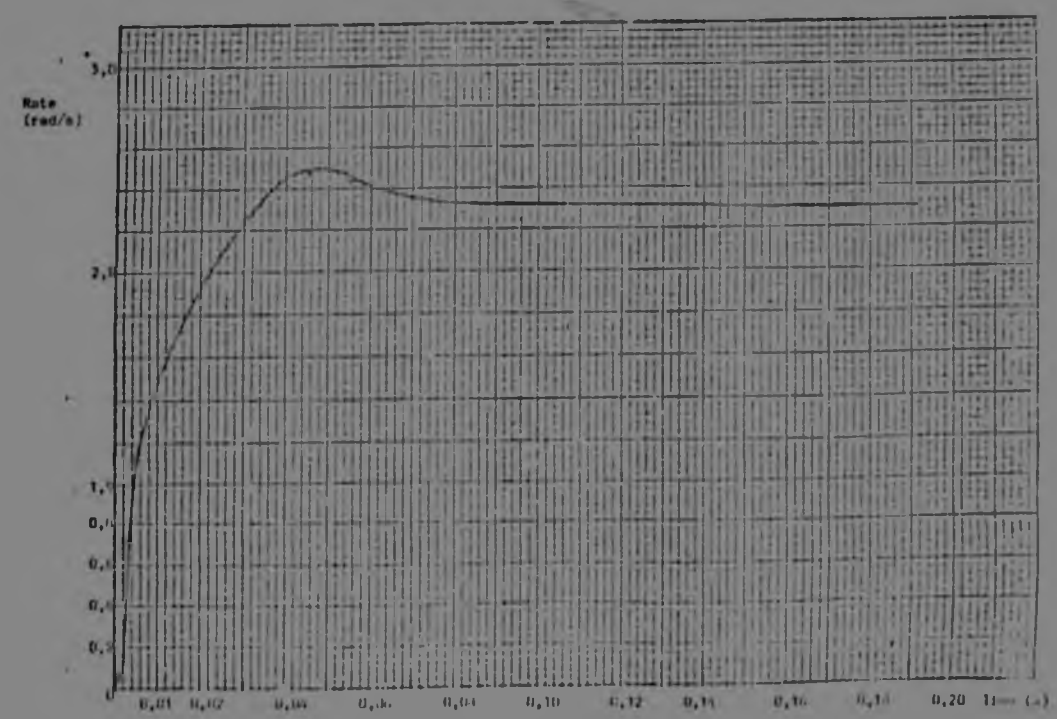

FIGURE 4.14 CLUSTER STEP INPUT RESPONSE

It must be remembered, that because all the measuring instruments were mounted on the cluster, and the elevation rate thus sensed in cluster axes, there was a component of the cluster rolling motion (about the r-axis) present in the elevation rate integrating gyro measurement. This had to be removed by measuring the roll rate using a rate gyro, as it was found that otherwise stabilization would be seriously degraded.

A further source of error was the noise in the sensors, that is primarily the rate integrating gyro. This instrument not only produced signal output noise, but also drifting of the signal due to shock and temperature changes. Drift due to temperature changes has been eliminated by the creation of a controlled environment for the tracker optics and stable platform. Drift due to shock and day-to-day drift of the sensor null have been eliminated using manual offset adjustments. Only output signal noise therefore remained to be considered.

The peak value was specified as 70mV, which resulted in a platform input torque of 7,6mNm. This was less than the estimated stiction torque of 8,4mNm, so it was expected that the platform would show no jitter. When stiction was overcome, the 70mV peak noise resulted in a stabilization error of 4μrad (from Figure 4.9) at 500rad/s (the expected maximum), which is negligible.

In the above paragraphs, the equations of motion have been derived and implemented according to the mechanical layout of the stabilized group. Subjecting the model to expected base motion disturbances, it has been shown that effective rotational motion stabilization has been achieved. It has also been shown that the model is not unduly sensitive to errors in parameter choice.

In the following paragraphs a compensation method for linear motion will be derived and evaluated.

4.6 Linear Motion Compensation

A different approach has been taken in compensating for the linear motion of the vehicle. It was not practical to measure the linear velocities and instead the linear accelerations were measured. These were then integrated digitally in the appropriate set of axes, and the velocities obtained were used

to calculate the relevant sightline turning rates which would in effect correct for the linear motion.

4.6.1 Illustration of the Method

The method could best be illustrated by considering a stationary target and moving vehicle, shown in Figure 4.15. The relative motion induced a sightline turning rate, given by the vector equation

$$\underline{S} = \frac{\underline{V} \times \underline{R}}{|R|^2} \qquad 4.21$$

where $\underline{S}$ = the sightline turning rate
$\underline{V}$ = the relative velocity
$\underline{R}$ = the range.

The stable platform had only two degrees of freedom, elevation and azimuth. Therefore, using the nomenclature defined in Figure 3.2, the sightline turning rates in these axes could be written as

$$S_e = \frac{V_d}{R} \qquad 4.22$$

and $$S_d = -\frac{V_e}{R}$$

If, then, the gimbal and cluster were to be rotated at these rates, there would be no relative turning of the sightline, and linear motion of the vehicle would not be apparent to the operator.

To find out whether the linear motion of the vehicle played a significant role or not, the maximum expected heave velocity of the vehicle of 3,0ms was considered. At a range of 500m, the relevant sightline turning rate (S_e above) is 6mrad/s.

For sinusoidal motion at a frequency of 0,2Hz this would imply a position amplitude of approximately 6mrad. This could be seen as a stabilization error by the operator, and the figure was outside the requirement of the system specification. It was therefore desirable that this error should be removed by linear motion compensation.

As can be seen from equation 4.21, the compensation would be most effective when tracking moving targets at short range.

FIGURE 4.15 SIGHTLINE TURNING RATE ILLUSTRATION

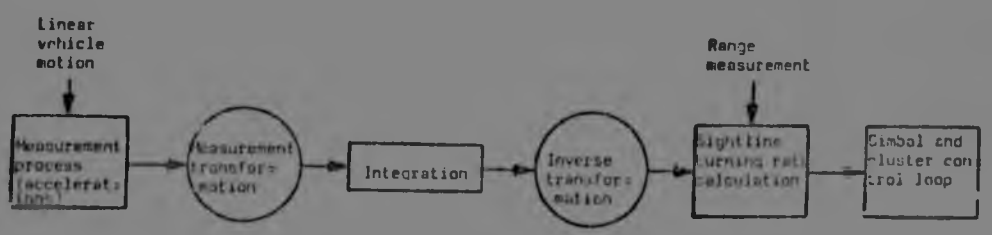

FIGURE 4.16 BLOCK DIAGRAM OF LINEAR MOTION STABILIZATION PROCESS

4.6.2 The Transformation Process

The linear motion of the vehicle was sensed by two accelero= meters mounted along the e- and d-axes (Figure 3.2). By their very nature, the acceleration measurements would be corrupted by 'g', the acceleration due to gravity. This was, however, known only in the inertial axis system, so the measure= ments and 'g' would have to be transformed to the same set of axes.

Furthermore, the axis system in which the linear motion measurements were made rotated with respect to inertial space, and as could be seen from the equation of Coriolis (equation 4.5), if the accelerations were integrated in this axis system, a component of the rolling motion of the axis system would be introduced. It was therefore advisable to transform the measurements to a stationary axis system (e.g. the inertial axis system) and to integrate them there. The 'g' vector was also subtracted here.

$$\underline{G}_I = \underline{\underline{T}}_{IC}\underline{G}_C \qquad 4.23$$

where $\underline{G}_C = \begin{pmatrix} 0 \\ G_e \\ G_d \end{pmatrix}$

is the vector of measured acceleration,

$\underline{G}_I$ = the measured accelerations in inertial space,

and $\underline{\underline{T}}_{IC} = \underline{\underline{T}}_{IS}\underline{\underline{T}}_{SB}\underline{\underline{T}}_{BO}\underline{\underline{T}}_{OC}$, where $\underline{\underline{T}}_{SB}$ has the same form as $\underline{\underline{T}}_{OC}$.

$\underline{\underline{T}}_{OC}$ and $\underline{\underline{T}}_{BO}$ can be found from equations 4.14 and 4.15

and $\underline{\underline{T}}_{IS}$ =

$$\begin{pmatrix} \cos\theta\cos\Psi & \cos\Psi\sin\theta\sin\phi - \sin\Psi\cos\phi & \cos\Psi\sin\theta\cos\phi + \sin\Psi\sin\phi \\ \cos\theta\sin\Psi & \sin\Psi\sin\theta\sin\phi + \cos\Psi\cos\phi & \sin\Psi\sin\theta\cos\phi - \cos\Psi\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{pmatrix}$$

where Ψ, θ, ϕ are the yaw, pitch and roll angles of the vehicle in inertial space.

The gravity vector could then be subtracted from the result and this integrated. The resulting velocity was transformed back to the cluster axis system using the inverse of the above

transformation. Because the transformation matrices were orthogonal, it was only necessary to use the transpose matrices.

Using the correct coordinates of the resulting velocity, the sightline turning rate could be found.

A word must be said about the choice of initial conditions for the integration process, however. The process may have been started at any time, and there was no way of knowing what the velocity at that time was. The initial velocity was therefore assumed to be zero, and the resulting error removed by averaging. That is, the average velocity was zero or a given value (e.g. the forward speed of the vehicle), and by comparing the average velocity obtained by the measurement and integration process with the actual average velocity, the error due to incorrect choice of initial condition could be corrected. This process of correction could be continually carried out, so that errors due to long-term drift could be compensated for.

The process is represented in block diagram form in Figure 4.16.

4.6.3 Simulation Results and Discussion

The above procedure and operations were implemented on a digital computer and, using a deterministic model of the vehicle motion and the non-linear model of the stable platform described in paragraph 4.5.2, the following results were obtained.

After more than 100s 'start-up' period, the calculated average velocity was within 2% of the actual average velocity. The stabilization error after a further 50s was less than 0,25mrad for a stationary target at 2 000m range.

It was found that the fact that all three accelerations were not measured did not significantly influence the results. This was a characteristic of the specific type of vehicle motion considered for this study.

A number of practical limitations were also illustrated. Due to the limited capacity of the on-board computer linear motion compensation could only be carried out every 250ms, compared with a sampling interval of 50ms, so that high frequency linear disturbances could not be compensated for. Quantization error and round-off error were also factors that contributed to error build-up. These will be analyzed in more detail in the following chapter.

A fu ber source of error was the noise introduced by the accelerometers, the angle encoders, the transformation and integration processes. The accelerometers had a scale factor non-linearity and hysteresis which could be summarized as output signal noise of 0,0125m/s² peak value. The transforma= tion and integration processes were executed numerically, and these could achieve accuracies of at least 0,1% of the irput value. Computation resolution was 24 bits (for the mantissa of a number) and integration step size was 50ms or less.

The angles used in the transformation were digitally enco= ded with 16 bit (100μrad) resolution and 44μrad rms accuracy. Simulation then showed that corrupting the measurements with noise resulted in a linear motion stabilization error increase of 1μrad, which is insignificant. This is due to the corrector characteristic of the loop

Despite these restrictions, the method described here signi= ficantly reduced the stabilization errors due to the motion of the vehicle, and together with the rotational stabilization loops, would effectively stabilize the target in inertial space, allowing the operator to give more attention to tracking the target accurately.

4.7 Pedestal follow-up Loop

It was pointed out in paragraph 3.3 that the mirror on the stabilized group was only able to rotate the sightline through 20° in azimuth. For some target tracking tasks and for target search tasks the sightline had to be rotated through larger angles. It was therefore necessary that the pedestal be able to rotate through 360°. The question then arose as to whether the pedestal control loop should form part of the stabilization of the sightline.

Obviously, the human operator would be most comfortable if the pedestal were stabilized in inertial space, thereby increasing his chances of successfully tracking the target. Because of the complexity of design required to achieve this, as well as the large amount of energy required to move the pedestal, operator, optics and ancillary equipment, a stabilized pedestal was considered impractical.

Even to stabilize the pedestal in azimuth would be problematical, due to the large inertia of the system and relatively high accelerations required. Therefore, instead of supplying some form of imperfect stabilization, which would only disorientate the operator, it was decided to stabilize the line of sight and allow the pedestal to undergo base motion. This suggested that a deadband be introduced for small azimuth angles which would prevent the pedestal servo amplifier from getting any input commands. The choice of the deadband was determined by the expected target attack profiles, described in paragraph 3.4.

Considering the lateral (azimuth) motion of the target during a medium toss bombing attack profile, most of the target manoeuvre took place within a sightline azimuth angle of 5°. The deadband was therefore chosen as ± 5°. The pedestal follow-up loop was then driven by the cluster angle η_c through a non-linear gain. A block diagram of the loop is shown in Figure 4.17, and the appropriate constants are

Servo Amplifier and Pedestal Drive

Servo amplifier gain	SAG	= 10,5V/V
Current feedback gain	PCFG	= 0,18V/A
Armature resistance	PRA	= 1,05Ω
Electrical time constant	PTE	= 0,0057s
Motor constant	PTK	= 1,6Nm/A
Moment of inertia	PJ	= 10,26kgm²
Pedestal friction term	PB	= 0,2Nm/rad/s

The pedestal drive motor torque was transmitted through a gearbox with ratio 6 : 1. The actual moment of inertia is thus given by

$$PJ = 6 \times 10,26$$
$$= 61,54 kgm^2$$

The friction term was also an approximation for the estimated friction torque, which is shown in Figure 4.18.

4.7.1 Linear Model: Stability check

The pedestal follow-up loop was too non-linear to allow the shaping networ and loop gain to be found using a linearized model. The linear model was, however, used to investigate the stability of the loop.

A quick analysis of the loop under open loop conditions showed that there were three poles at the origin. When the loop was closed, two of these poles moved into the right half plane, causing the loop to be unstable when PGAIN moved out of the deadband area (Figure 4.19).

This situation could be remedied by introducing a feedforward term, and the resulting root locus diagram is shown in Figure 4.20.

4.7.2 Non-linear Model: Loop Design

The optimum values of the shaping network and gain were now found using a model which contained all the system non-linearities, including the servo amplifier input voltage limit of 6V and current limit of 24A. Using the results of the linear analysis as a guide, the shaping network parameters were obtained.

$$pr_z = 0,12s$$
$$pr_p = 0,01s$$

The non-linear gain is shown in Figure 4.22.

The step response curves for three different step sizes are shown in Figure 4.21. It must be noted here that these curves illustrate the pedestal rotation rate response. Because the pedestal response time was so much slower than the cluster response time, the mirror reached the endstop before the pedestal had caught up. Although this only happened for large input commands, it was considered advisable to introduce a position

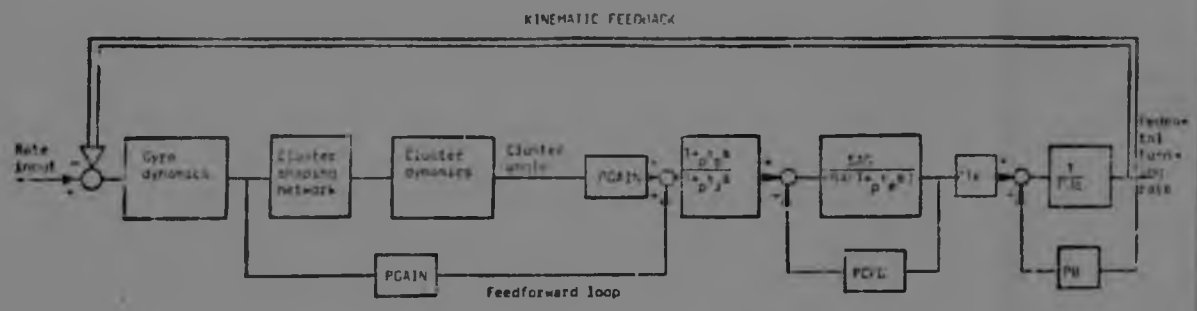

FIGURE A 17 PEDESTAL FOLLOW-UP LOOP BLOCK DIAGRAM

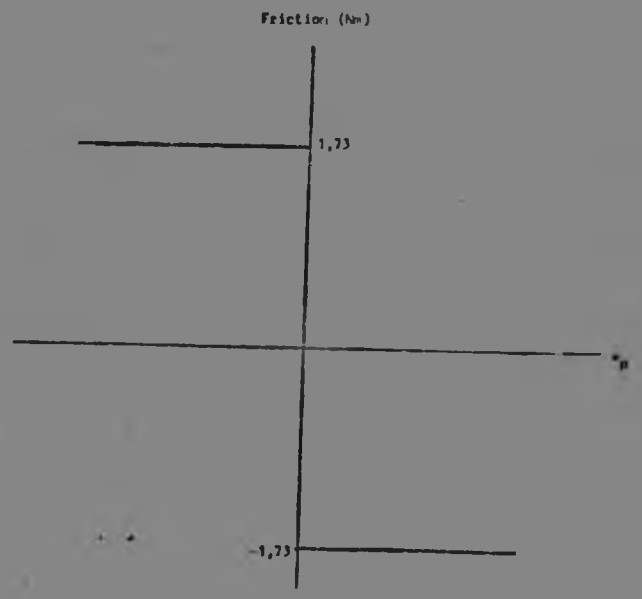

FIGURE A.18 PEDESTAL FRICTION MODEL

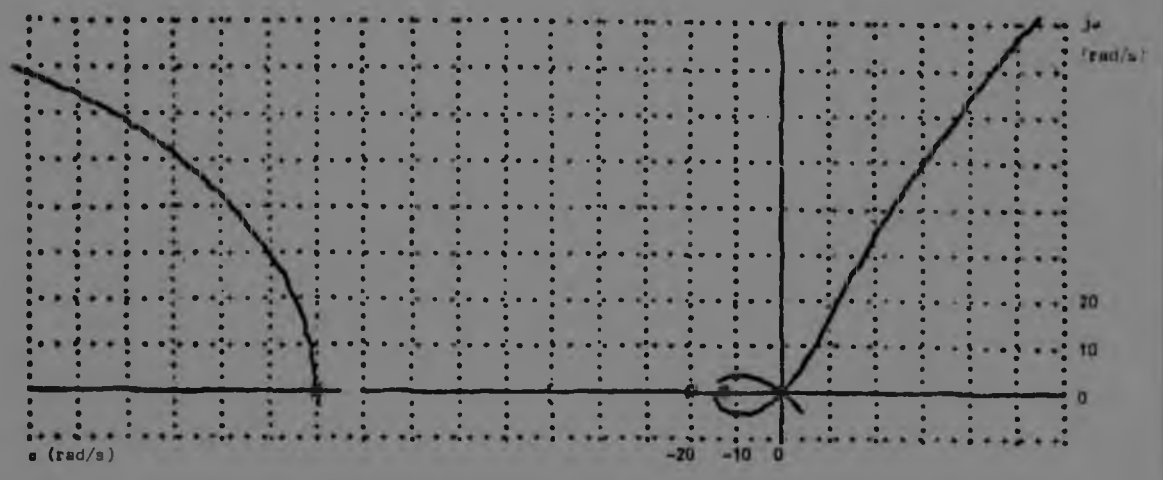

FIGURE 4.19 ROOT LOCUS DIAGRAM WITHOUT FEEDFORWARD

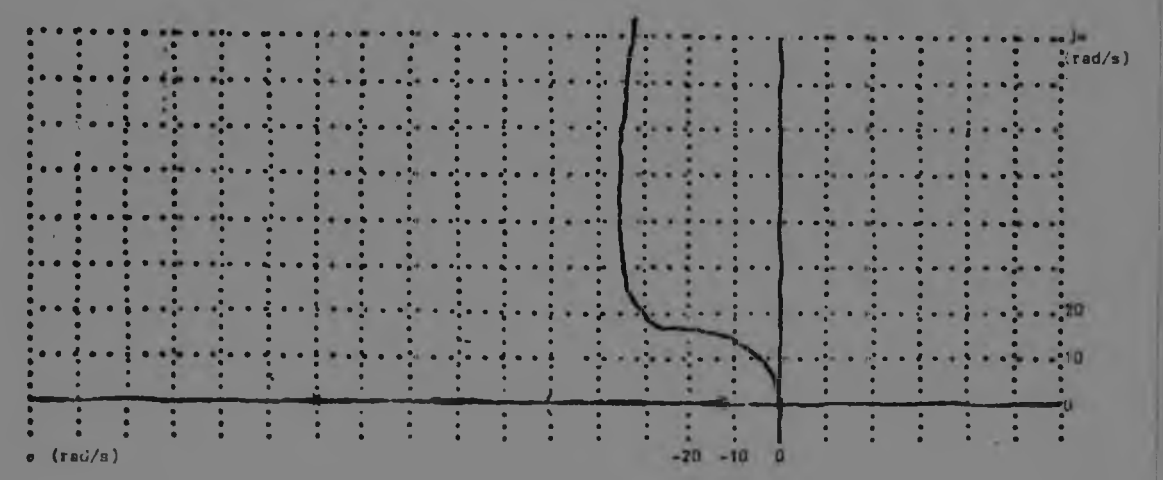

FIGURE 4.20 ROOT LOCUS DIAGRAM WITH FEEDFORWARD

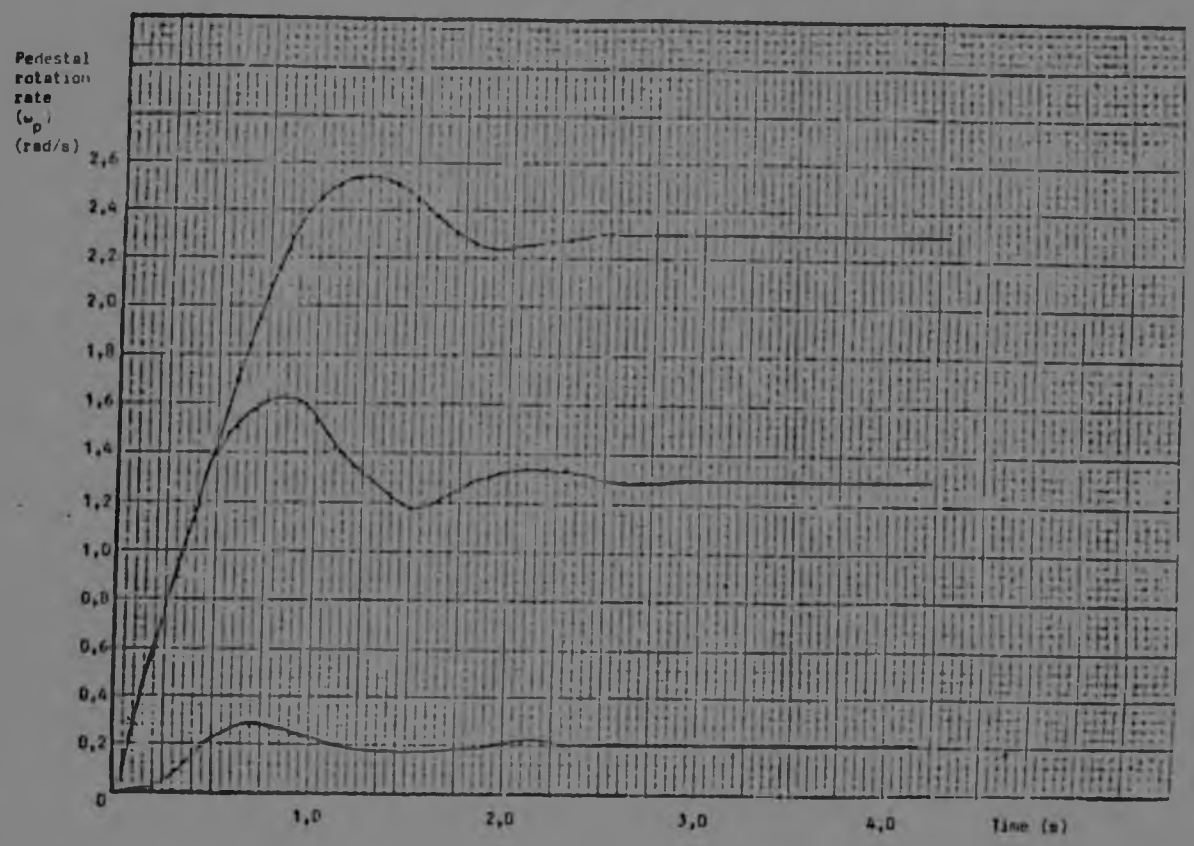

FIGURE 4.21 PEDESTAL STEP INPUT RESPONSE CURVES

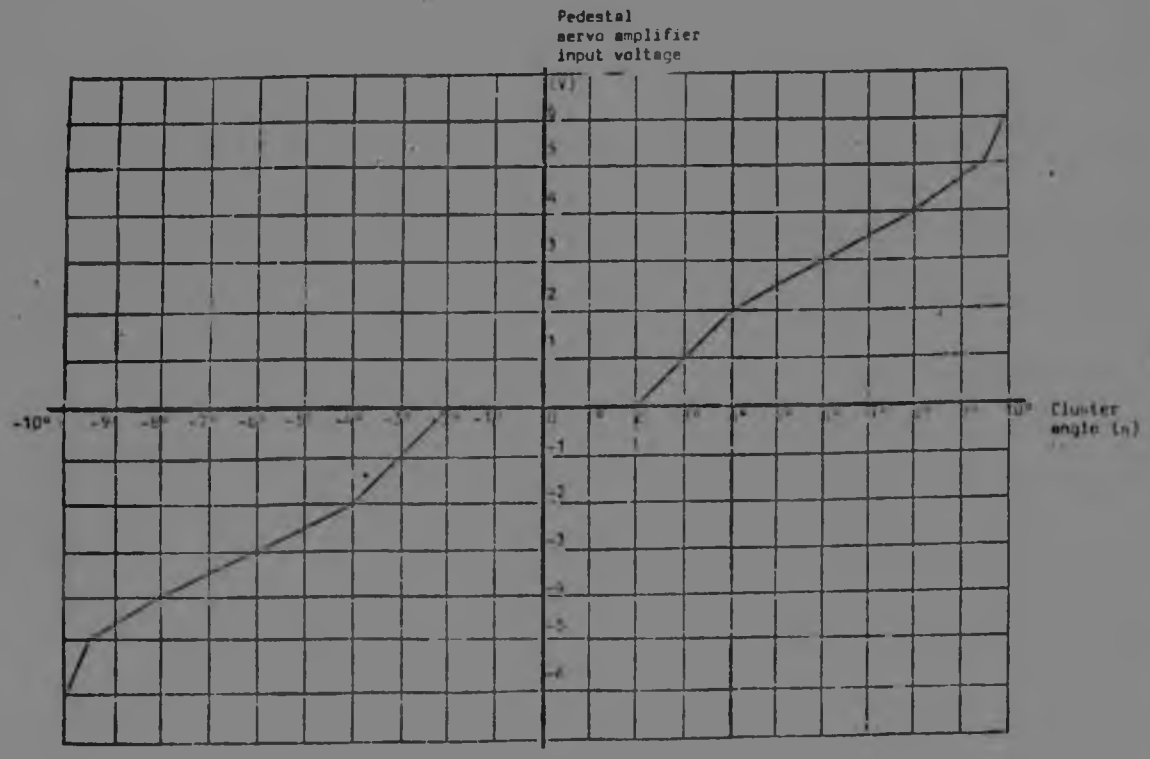

FIGURE 4.22 NON-LINEAR GAIN

feedback loop around the cluster which slowed the mirror so that the impact against the endstop was lessened. This simple loop was only operative for mirror deflections between 4° and 5°, and had a loop gain of 3rad/s/rad. As the pedestal rotation rate come close to that commanded, the mirror moved away from the endstop and the cluster angle settled at a non-zero steady-state value such that the command to the pedestal follow-up loop was sufficient to maintain the required rotation rate. The steady state cluster angle was small enough to allow for any additional deflections required for stabilization.

The frequency response of the pedestal follow-up loop was also obtained, and is shown in Figure 4.23. It is clear that the loop response time was much slower than that of the cluster (Figure 4.11).

A sensitivity analysis was performed on the parameters of the loop, and the results can be summarised as follows.

A decrease of 17% in the value of the pedestal inertia decreased the settling time of the step response by 1s. An increase of 17% in the inertia increased the settling time of the step response by 0,1s. The stability of the loop was not significantly affected.

A variation of 10% in the shaping network time constants did not significantly affect the settling time of the step response. A reduction in the feed forward gain (PGAN), however, resulted in more oscillations.

There were a number of possible variations in the gain PGAIN. Two were investigated, shown in Figures 4.24 and 4.25. The linear gain had a very small effect on the step input response (the settling time was 0,1s longer). The bang-bang type gain did not significantly improve the rise time for the step input response, as the current to the motor is limited anyway. Because of the deadband in the gain however, the response tended to oscillate around the final value, rather than approach it. This was considered a less attractive possibility for the gain.

4.6 Summary

In this chapter the requirement to stabilize the target effectively in inertial space has been met. In paragraph 4.2 the equations of motion were derived for a general gimballed system.

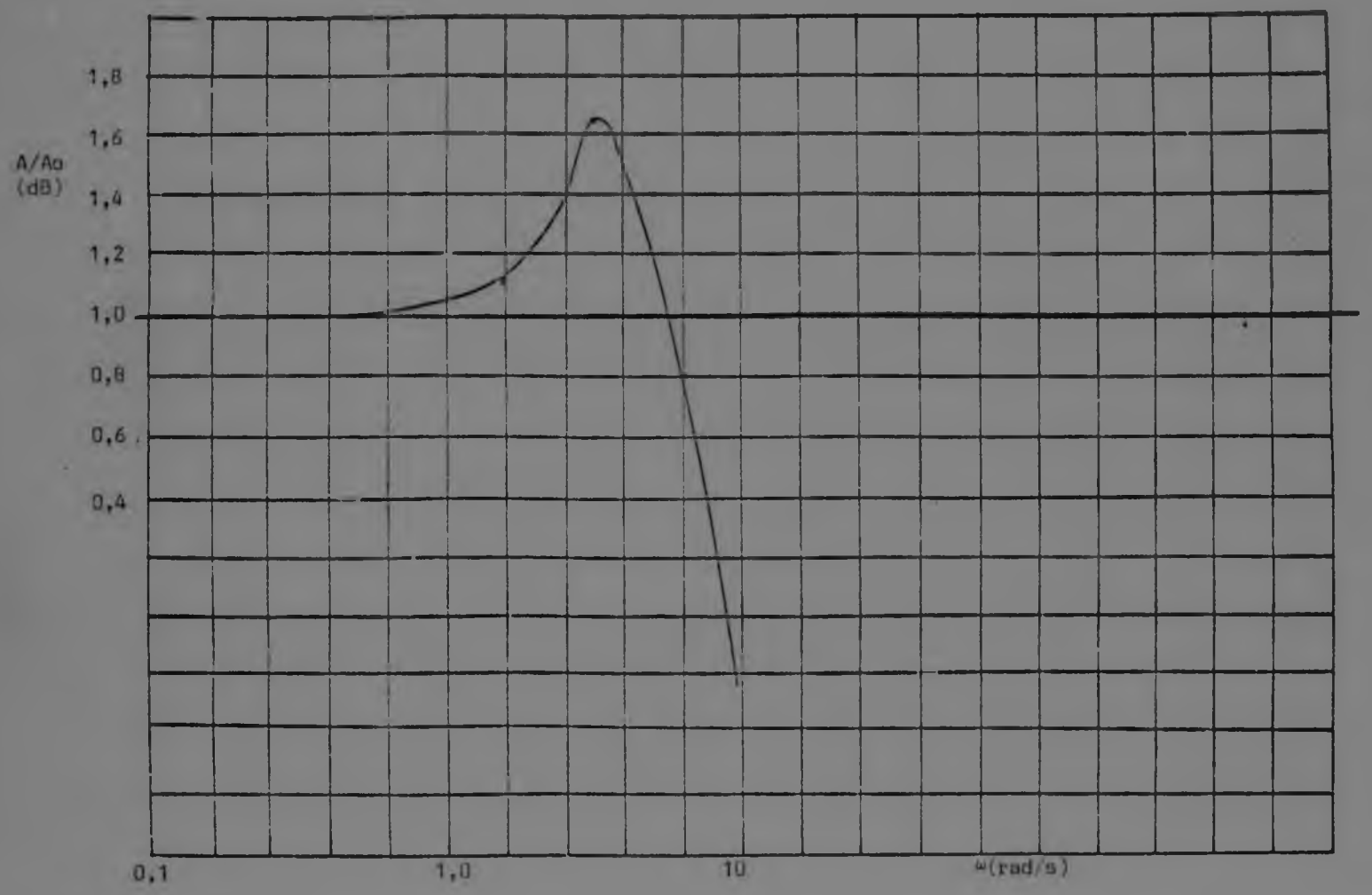

FIGURE 4.23 PEDESTAL FOLLOW-UP LOOP FREQUENCY RESPONSE

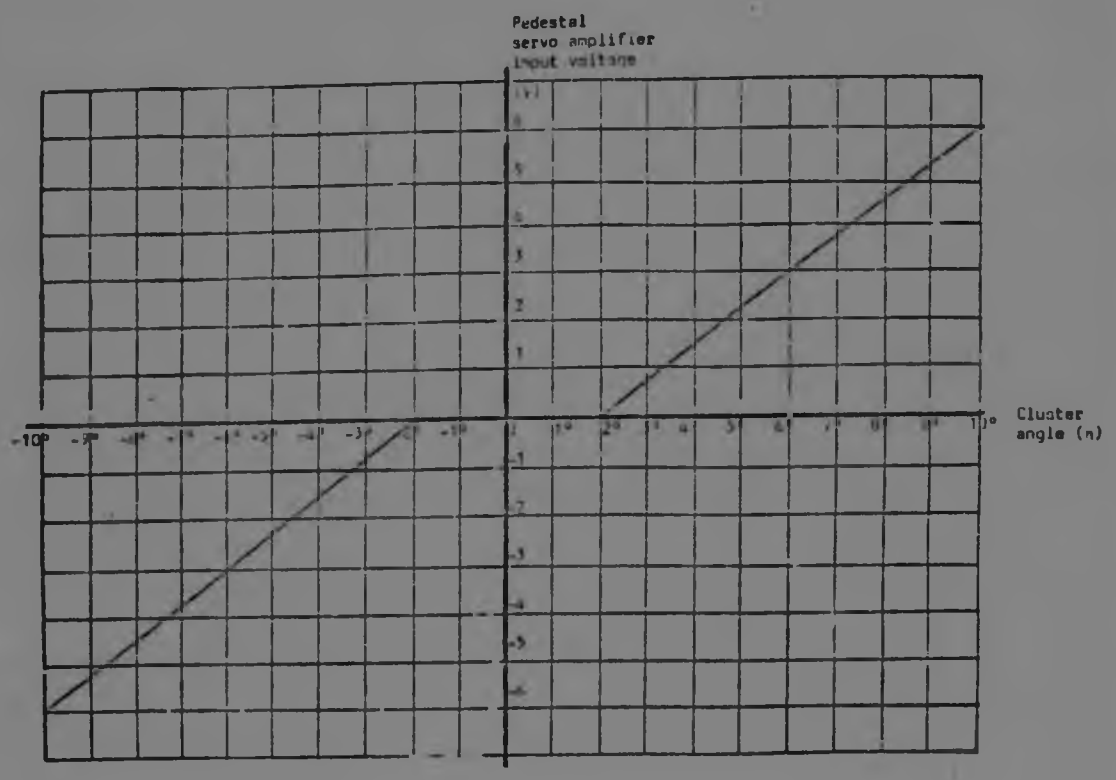

FIGURE 4.24 LINEAR GAIN

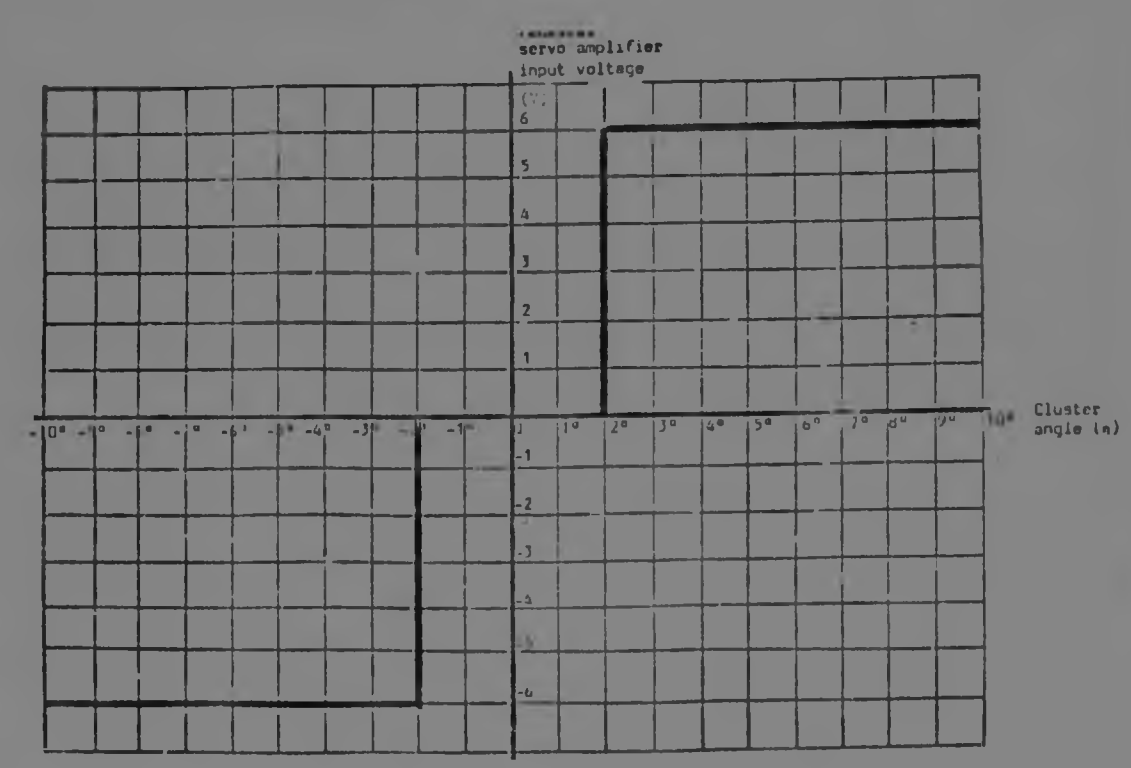

FIGURE 4.25 BANG-BANG GAIN

When considering the mechanical layout of the stabilized group (paragraph 4.3) it was possible to justify certain simplifying assumptions with respect to the moments of inertia and therefore decoupling of the elevation and azimuth motions. At this stage a mathematical model of the stabilized group was defined, which, when subjected to base motion disturbances described in paragraph 4.3, and when controlled by the loops derived in paragraph 4.5, allowed the evaluation of the rotational motion stabilization. A non-linear model of the stabilized group was used to determine the sensitivity of the design to system parameter and measurement errors. It was found that the model was sufficiently robust to assure that the results were valid.

An algorithm for compensating for the linear motion of the base in inertial space was derived and evaluated in paragraph 4.6. Here it was pointed out that due to computing time limitations only low frequency linear motion could be stabilized.

Because the movement of the stabilized mirror was limited in azimuth, a pedestal follow-up loop had to be implemented. The design of the control loop required was given in paragraph 4.7, and evaluation showed that the follow-up loop allowed the sightline to be rotated in an optimum manner through 360°, yet did not interfere with stabilization during tracking tasks.

It now remains to analyze the system described here in a tracking situation, so that it can be determined whether the tracking specifications will be met. This will be done in the following chapter.

5 TRACKING AID AND FILTER DEVELOPMENT

5.1 Introduction

In the previous chapter it was shown that stebilization loops have been designed which would keep the optical director pointing vector stable with respect to vehicle motion. The accuracy attained was 1,0mrad for periods of time greater than 3 minutes, which was an order of magnitude greater than the time taken for a typical engagement. The human operator could then give his full attention to the task at hand, namely the tracking of moving targets.

In this chapter it is described how the results of the stabilization analysis were used to construct a tracking system simulator, and the performance of the human operator was evaluated. Using measurements obtained for a given tracking task, the human operator transfer function was found. This was applied to a linear system model so that system stability could be investigated. The performance of the tracker for various aids was compared, and when a satisfactory controller response had been obtained, a tracking filter algorithm could be analyzed and implemented. The target states required for aiming the weapon could then be estimated. As had been done in previous sections, the sensitivity of the system to changes or errors in the algorithm parameters was determined, and it was shown that the design met the system requirements.

5.2 Tracker Simulator Construction

The purpose of constructing a tracker simulator was firstly, to enable the human operator characteristics to be measured and analyzed and secondly, to enable the tracking aid (compensator) and tracking filter to be evaluated. Simulator construction was seen to be an inexpensive and effective method of determining system performance without waiting for any hardware to be built. As the cost of developing the tracker exceeded R1m and took more than two years, errors discovered at an early stage could save time and money. The simulator would highlight possible problem areas which might require special attention during the design.

5.2.1 Stabilization Loop Representation

The human operator steered the sightline in space via his joy-

stick and the stabilization loops. The response time of the stabilization loops was much shorter than that of the target, so the stabilization loop transfer functions were considered unity for simulator purposes. The stabilization error was not neglected, however, and was simulated by adding a random rate to the loop at the average frequency of the base motion. The RMS value of these random rates was restricted to approximate the RMS stabilization error (0,5mrad).

5.2.2 Simulator Construction

The simulator, shown in Figure 5.1, was built around a digital computer, which generated the target profiles (in a random sequence, and with random initial conditions), and displayed the tracking error on the VDU, which had a crosshair on its centre. The operator had to keep the target (represented by a moving dot) on the crosshair.

The simulator only approximated the actual tracker in the following:

a The simulator was situated in a laboratory, so that the operator was exposed neither to the elements nor to base motion.

b The VDU size was only a fraction of the total field of view. The operator also had no indication of the range of the target due to the difficulty of realizing depth on the VDU.

c The operator used a joystick which was not the same as the actual joystick, nor did he have to watch other inputs (eg. the control panel), or give firing commands.

5.3 Human Operator Linear Open Loop Transfer Function

In order to understand the system more fully, a linearized transfer function of the human operator was determined. This transfer function consisted of a delay, which was found using cross-correlation techniques, and lead and lag terms, found using a paramter identification method. This analysis helped to explain the behaviour of the system under certain conditions and could be used to determine what changes in the system would result in improved performance.

5.3.1 Human Operator Delay

Using the simulator described above, the input to the human

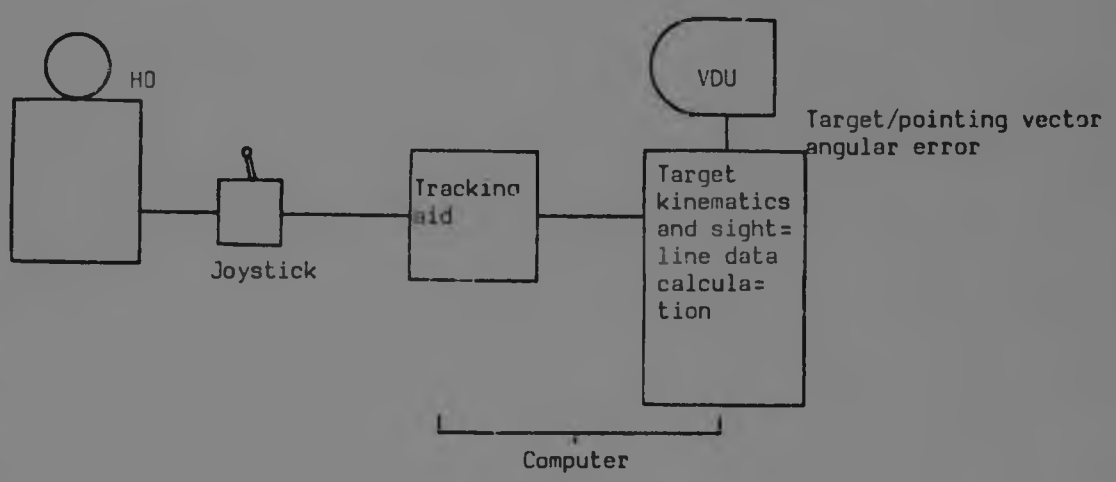

FIGURE 5.1 TRACKER SYSTEM SIMULATOR

operator tracking err[illegible] and the output of the human operator joystick output) were [illegible]p ed and recorded in an array. The cross-correlation between these two signals was then found (Papoulis (reference 17,18)).

The fourier transforms of the two signals were found using FFT algorithm (Brigham (reference 3)).

$f_1(t) \rightarrow F_1(\omega)$

$f_2(t) \rightarrow F_2(\omega)$

The cross energy spectrum was found by multiplying $F_2(\omega)$ by the conjugate of $F_1(\omega)$.

$f_{12}(\omega) = F^*_1(\omega)\, F_2(\omega)$

The cross-correlation between the two signals was then given by the inverse transformation of the cross-energy spectrum

$f_{12}(\omega) \rightarrow \rho_{12}.t$

Because the signals $f_1(t)$ and $f_2(t)$ were aperiodic, and [illegible]n FFT algorithm was used (ie., the signal was approximated [illegible] finite series of points), "leakage" of the spectra occured. This [illegible] inimized by multiplying the signals by the Hanning window, defined by

$h(t) = [illegible] \cos \frac{2\pi t}{T_C}$ $0 < t < T_C$ 5.1

A typical resulting cross-correlation function is shown in [illegible]. The delay of the operator was given by the first [illegible]inimum in the graph. The other peaks were harmonics of the first peak. The value was

$\tau_{HO} = 0,3$sec

[illegible] be for the above example was 0,2 samples/sec).

[illegible] Human Operator Gain, Lead and Lag Terms

The following open loop transfer function for the human operator was chosen

$$-\tau s \quad \frac{K(1 + \alpha s)}{(1 + \beta s)(1 + \gamma s)}$$

and the method used to find the values of the parameters is sketched briefly below.

The delay τ was found in paragraph 5.3.1.

α, β and γ were found using the parameter identification method described in Gabay and Merhav (reference 7.8). The [illegible]tion and human operator input were sampled every .025[illegible]. These signals were passed through filters with transfer function

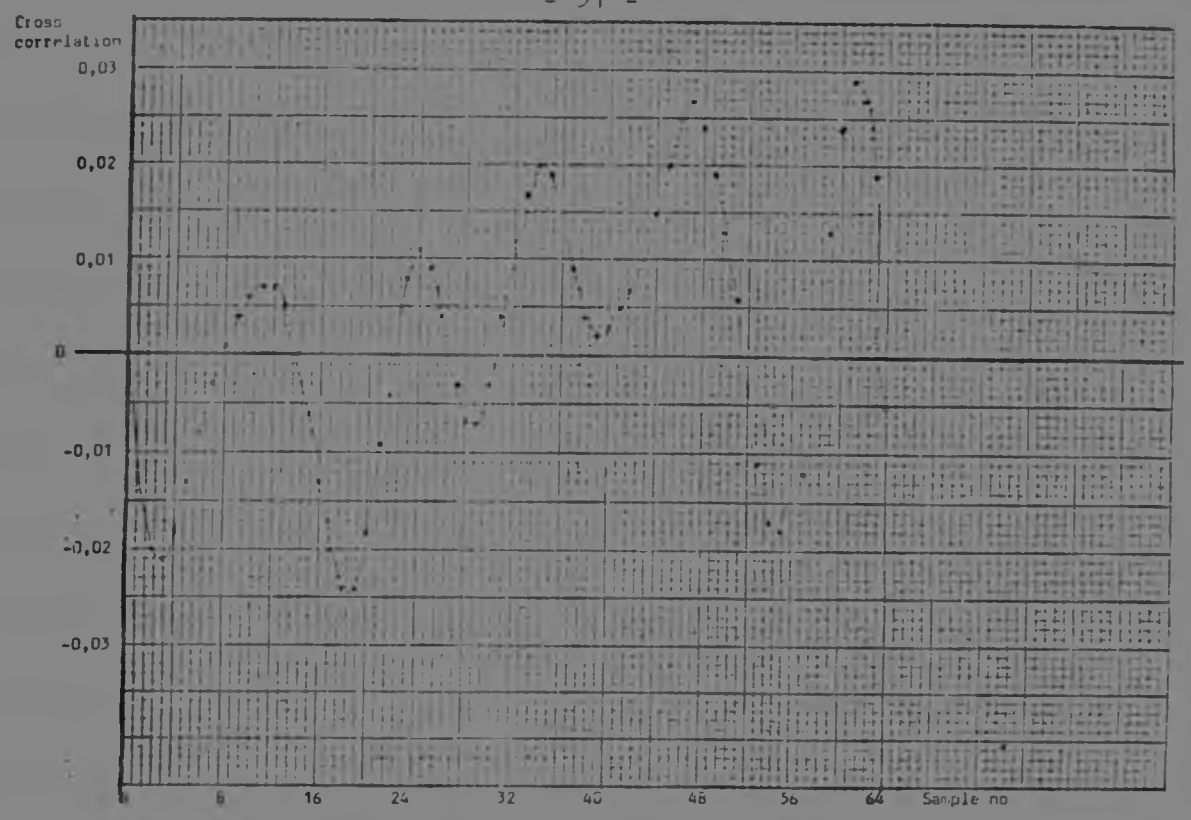

FIGURE 5.2 OPERATOR INPUT/OUTPUT CROSS-CORRELATION FUNCTION

FIGURE 5.3 OPERATOR GAIN ESTIMATES

$$H(s) = \left(\frac{30}{s + 30}\right)^4$$

and the derivatives of the signals were found. The input signal was also time-shifted by a delay of 0,2s and the expected values of the parameters were found by applying the orthogonality condition on the filtered input signal and the error signal.

A typical plot of the parameters is shown in Figure 5.3 and a number of interesting phenomena could be seen. The region of instability in the centre of the graphs corresponded to the target fly-over (dive attack profile, Figure 3.3). Here the identification algorithm settling time was too slow to cope with the change of parameters.

As the target approached and flew away, however, definite parameter values were identified. Note that the lead decreased and gain increased after fly-over, indicating that the operator automatically compensated for a change in situation (the target was flying away, so that the operator was more relaxed and could increase his gain without losing control or destabilizing the system). The lag term β remained more or less constant, but the γ term disappeared indicating a redundant term in the model.

The open loop transfer function for the human operator could therefore be represented from equation 5.2 as

$$HO(s) = e^{-0,3s} \frac{2,5\ (1 + 5s)}{1 + 0,003s)}$$

but it must be remembered that the values of these parameters underwent significant changes as the tracking task changed and the operator adapted.

Interestingly, this transfer function was not inconsistent with that found in the literature (reference 29).

5.4 Tracker System Linear Analysis

A linear model of the tracker system could now be constructed and analyzed. Using root locus diagrams, a compensator for improving the tracker performance was suggested.

5.4.1 Tracker Root Locus Diagram

The human operator transfer function time constants were given in equation 5.4. The delay was represented using the standard Padé approximation

$$e^{-\tau s} = \frac{1 - \tau/2s}{1 + \tau/2s}$$

The poles of the system were therefore (in rad/s)

0,0 ± j0,0 (error integration)

-6,67 ± j0,0 (Padé approximation

-333,3 ± j0,0 (Lag term)

and the zeros were

6,67 ± j0,0 (Padé approximation)

-0,83 ± j0,0 (Lead term)

The root locus diagram is sketched in Figure 5.4. It was obvious that the system became unstable for even small loop gains, due to the human operator delay pole: zero pair.

This could be changed by introducing a compensator into the system. A number of options were available: an integral element, a proportional-plus-integral element or a proportional-integral-derivative element.

After analysing the influence of each type of compensator, considering the system performance improvement obtained and the degree of complexity required for each, it was decided to implement the proportional-plus-integral element and to study the resulting system response to see if the system tracking accuracy could be met.

The root locus diagram incorporating the proportional-plus-integral element is shown in Figure 5.5. A much higher loop gain could now be tolerated before the system became unstable. It was also easier to maintain a good system response characteristic (i.e. fast response time with critical damping).

5.5 Tracking Aid Implementation

In evaluating the performance of any tracker which included a human operator, the question of the standard human operator arose. The human operator was himself a very adaptive controller, and he exhibited a strong learning ability.

In order to find out whether tracker performamnce improved when aided by a compensator element, some standard had therefore to be defined. The standard for this analysis has been chosen as the average performance of three untrained operators. The operators were required to do the tracking tasks, as described in paragraph 3.4 for three different tracker configurations:

a An uncompensated system. The joystick commands of the human operator directly drove the sightline.

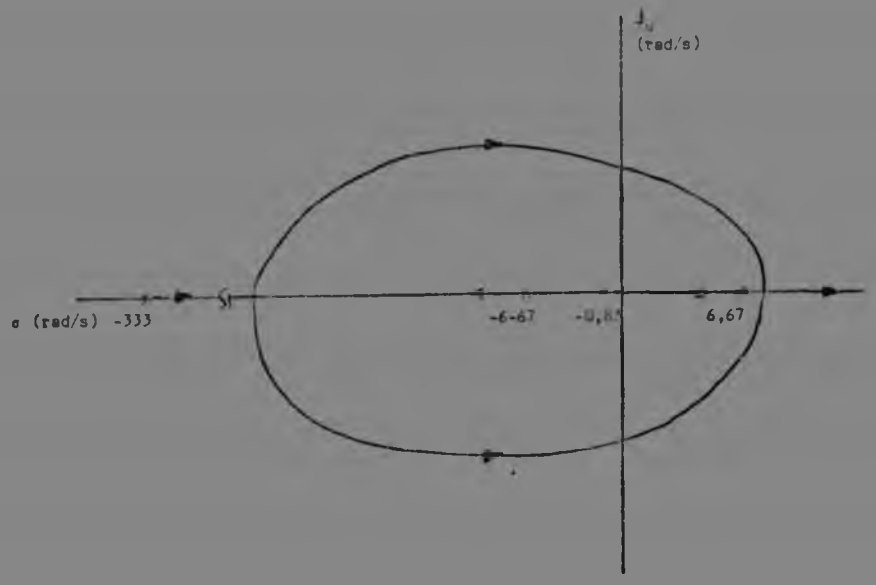

FIGURE 5.4 UNCOMPENSATED TRACKER ROOT LOCUS DIAGRAM

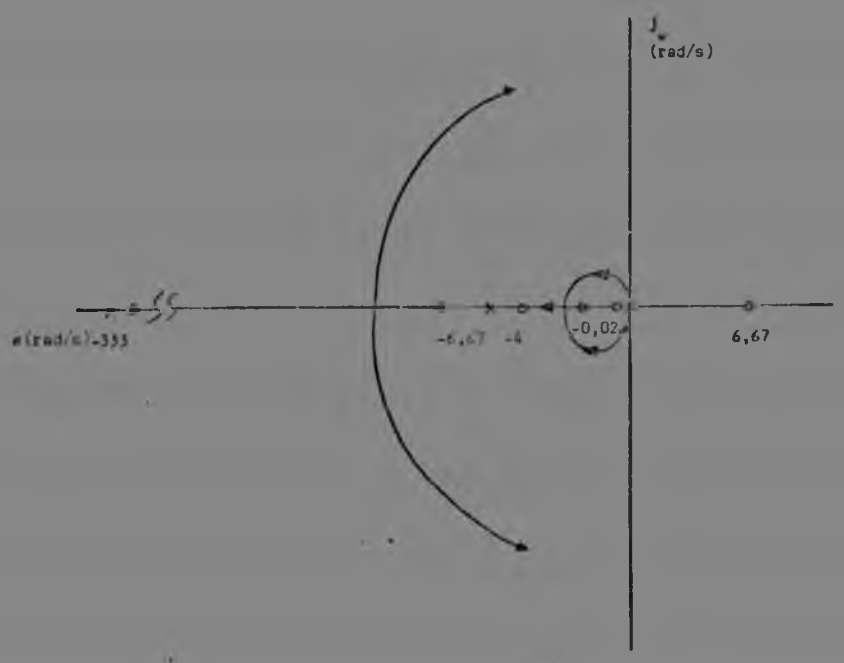

FIGURE 5.5 COMPENSATED TRACKER ROOT LOCUS DIAGRAM

b A system incorporating a non-linear joystick gain. The joystick commands were multiplied by a square law gain before driving the sightline.

c A compensated system. The joystick commands were multiplied by a square law gain and fed through a proportional-plus-integral element before driving the sightline.

The tracker configuration is summarized in block diagram form in Figure 5.6.

Each operator was given approximately 20min for each tracker configuration and the smallest RMS tracking angular error was noted. The average for each operator and the three target profiles was found and is shown in Table 5.1.

Although the results were not statistically valid, because the sampling of human operators was very small, they did indicate that the use of a non-linear joystick gain would improve the tracker performance and that the implementation of the proportional-plus-integral element as a tracking aid would enable the required system accuracy to be met. This was further confirmed by the learning curve for an operator, taken over approximately 40 hours, shown in Figure 5.7. This curve represented learning for the aided tracker, but included system experience gained on the unaided tracker.

According to Table 5.1, the accuracy achieved with the aided tracker did not meet the specification of 1,5mrad rms error. The operator learning curve (Figure 5.7) however, showed that an operator with 40 hours of training could achieve a rms error of 1,2mrad. Despite the fact that noise representing the stabilization error (0,5mrad rms) was introduced in the simulator, the actual error made by the assumptions in paragraph 5.2.2 could not be measured. Only evaluation of the actual system under similar operational conditions will determine the real tracking error.

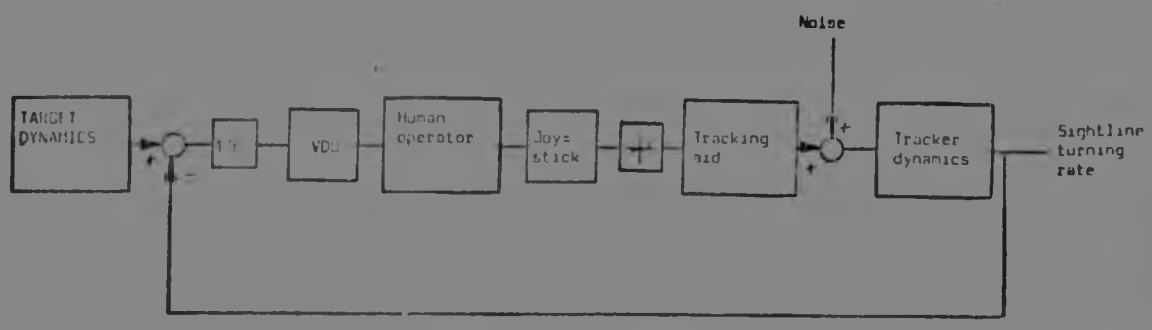

FIGURE 5.6 BLOCK DIAGRAM OF TRACKER CONFIGURATON

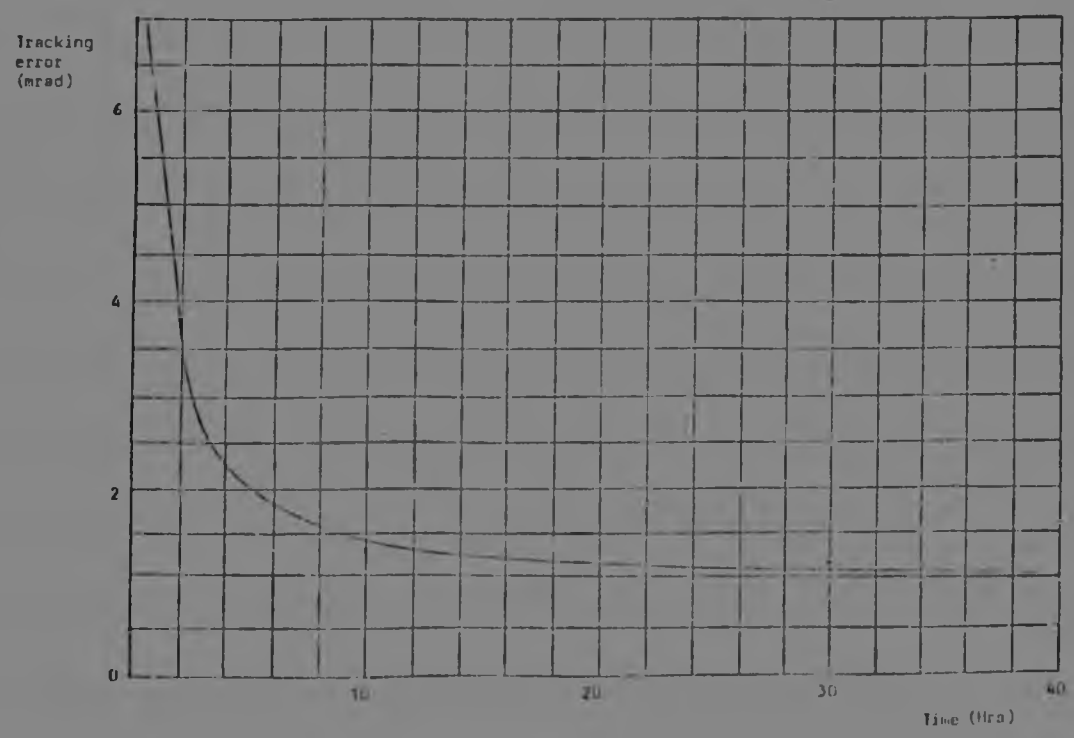

FIGURE 5.7 OPERATOR LEARNING CURVE

TABLE 5.1 RMS TRACKING ERROR

	UNAIDED TRACKER	TRACKER WITH NON-LINEAR JOY-STICK GAIN	AIDED TRACKER
Medium loss bombing attack (averege for 3 operators)	5,5mrad	3,0mrad	1,8mrad
Missile attack (average for 3 operators)	5,1mrad	2,8mrad	1,7mrad
Dive attack (average for 3 operators)	6,7mrad	3,9mrad	1,9mrad
Average performance (untrained operators)	5,8mrad	3,2mrad	1,8mrad

5.6 Adaptive Filtering Algorithm Application

It has been shown that the implementation of a proportional-plus-integral element in the tracker allowed the human operator to meet the system accuracy requirements. As has been pointed out before however, the target states had to be derived by the tracker for transmission to the weapon aiming mechanism, and it will be shown here that the tracking aid described above lends itself to expansion into a tracking filter algorithm. This algorithm is derived and evaluated below.

5.6.1 Tracking Filter Algorithm

Consider first the implementation of the proportional-plus-integral element, shown in Figure 5.8. As this was done digitally, the equivalent sampled discrete time representation is shown in Figure 5.9. The transfer function of the integrator with sampled input was given in the Z-domain by Gupta and Hasdorff (reference 10).

$$\frac{o}{i} = \frac{Z + K - 1}{Z - 1} \qquad 5.5$$

If the bilinear transformation $Z = \frac{1+w}{1-w}$ was applied, the proportional-plus-pintegral properties of the element were clearly seen.

$$\frac{o^*_1}{i^*_1} = C\,\frac{w + K/(2-K)}{w} \qquad 5.6$$

The expansion of the algorithm in this form to that of the adaptive tracking filter now followed readily. This is shown in Figure 5.10 and follows the pattern of the optimal linear filter (Sage and Melsa (reference 22)).

There was then some doubt as to the validity of the measurement process of the linear filter presented here with respect to that defined for the optimal linear filter. The answer lay in the joystick input. This was a function of the difference between the actual target state (or sightline turning rate) $\omega(k)$ and the estimated state $\bar{\omega}(k)$ (the crosshair or pointing vector turning rate).

Let the joystick input be

$$j(k) = \omega(k) - \bar{\omega}(k) + n(k) \qquad 5.7$$

where $n(k)$ is a zero mean white noise process with

$$\text{cov}\,\{n(k)\;\; n(k)\} = E(k)\,\delta(k-j)$$

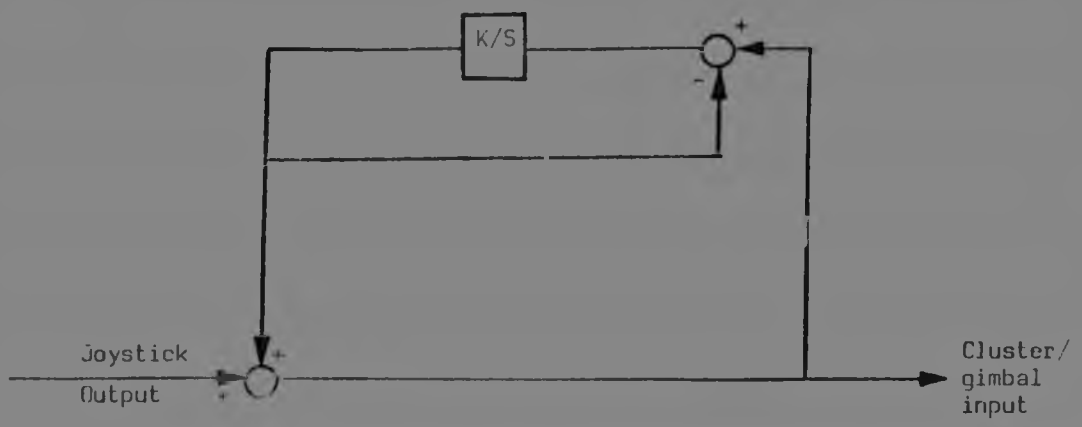

FIGURE 5.8 PROPORTIONAL-PLUS-INTEGRAL ELEMENT IMPLEMENTATION

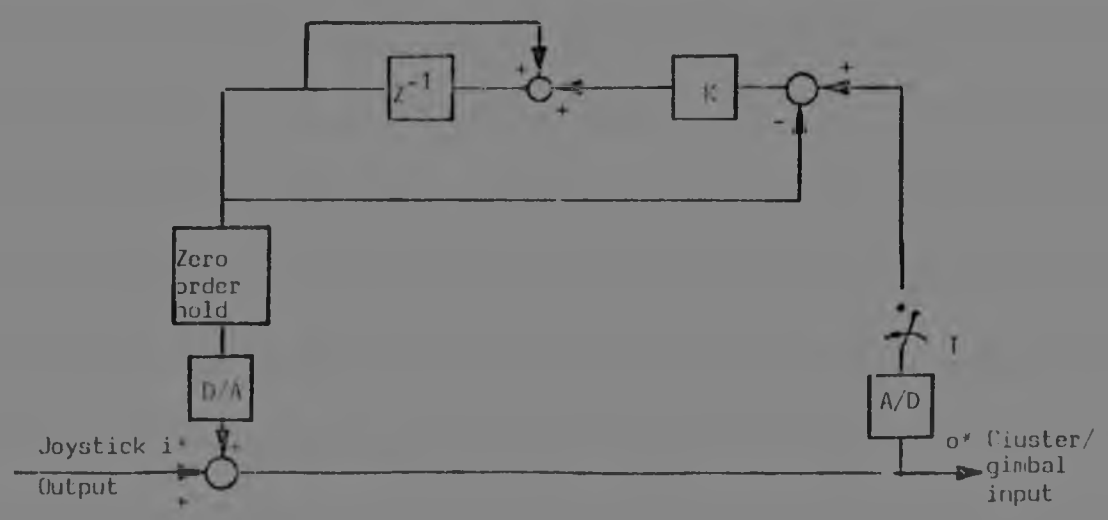

FIGURE 5.9 PROPORTIONAL-PLUS-INTEGRAL ELEMENT DISCRETE TIME REPRESENTATION

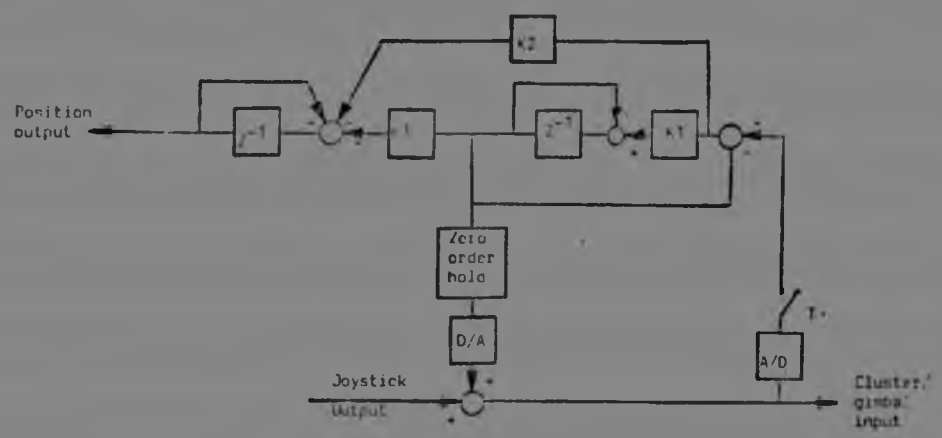

FIGURE 5.10 TRACKING FILTER IMPLEMENTATION

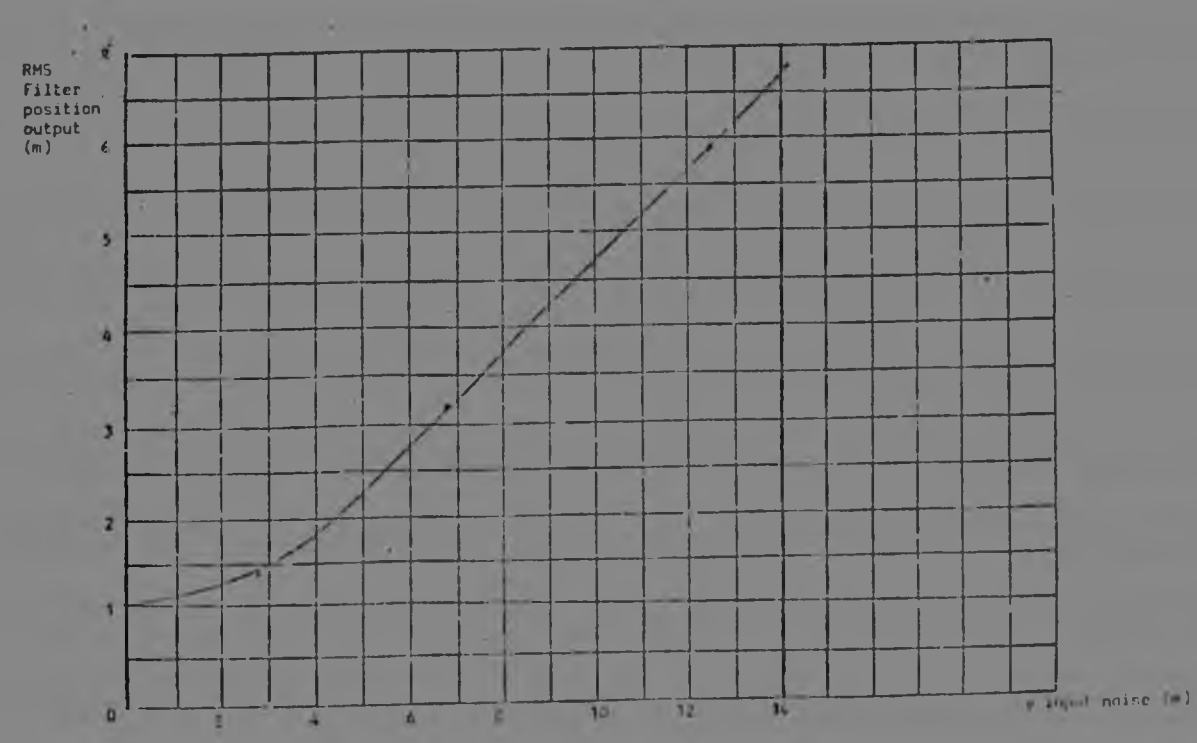

FIGURE 5.11 RANGE FILTER OUTPUT DEVIATION

The innovation process in the tracking filter

$$I(k) = K_1[z(k) - \hat{\omega}(k)] \qquad 5.8$$

where

$$z(k) = \bar{\omega}(k) + j(k) + v(k) \qquad 5.9$$

is the measurement process and v(k) is a zero mean white noise proces . Substituting equation 5.7 one obtains the following

$$z(k) = \omega(k) + \upsilon(k) \qquad 5.10$$

with $\upsilon(k) = n(k) + v(k)$, a zero mean white noise process with

$$\text{cov}\{\upsilon(k), \upsilon(k)\} = V(k)\,\delta(k-j)$$

The measurement process of the tracking filter algorithm was therefore consistent with that defined for the optimal linear filter, so that the tracking filter algorithm was the best linear estimator of the tracking process.

This algorithm was by no means the only option considered (see Chapter 2). In fact, because the target moved in inertial space, it would have been easier to model the target accurately in cartesian coordinates in inertial space. The measurements, however, (range, elevation rate, azimuth rate), were made in Polar coordinates in the cluster axis system, so that these would have to be transformed to inertial space. This transfor= mation was highly non-linear, so a non-linear filter would be obtained eventually.

The practical difficulty encountered with the transformation of the measurements, was the computation time required. Initial estimates showed that the transformation would require so much computational effort, that there would be insufficient time left to implement the other algorithms and control functions satisfactorily such as linear motion stabilization (refer to the discussion of Chapter 2). Furthermore, the tracking aid would have to be implemented seperately from the tracking filter.

The solution obtained here to the problem, namely the imple= mentation of a linearized filtering algorithm in Polar coordi= nates in the cluster axis system, was found to be the best trade-off between the different requirements. It was shown too, that reasonable filter output accuracy could be achieved.

5.6.2 Range Filter Algorithm

An additional advantage of implementing the tracking filter in polar coordinates, cluster axes was that the range filter

algorithm could be decoupled and considered seperately. As suggested in reference 10, a second order linear discrete filter was implemented. The structure, being the same as that of the tracking filter, will be detailed below.

5.6.3 Matrix Equations of the Algorithms

The algorithms described above could be written in matrix form as follows. Given the message model (reference 9,22)

$$\underline{x}(k+1) = \underline{\Phi}\underline{x}(k) + \underline{\Gamma}(k)\,\underline{w}(k) \qquad 5.11$$

and measurement model

$$\underline{z}(k) = \underline{H}\underline{x}(k) + \underline{v}(k) \qquad 5.12$$

with $t_{k+1} = t_k + T$, T the algorithm step size of 50ms. The state vector is represented by

$$\underline{x}(k) = \begin{pmatrix} R(k) \\ \dot{R}(k) \\ \alpha_e(k) \\ \omega_e(k) \\ \alpha_a(k) \\ \omega_a(k) \end{pmatrix}$$

where $R(k)$ = range (m)
$\dot{R}(k)$ = range rate (m/s)
$\alpha_e(k)$ = elevation angle (rad)
$\omega_e(k)$ = elevation rate (rad/s)
$\alpha_a(k)$ = azimuth angle (rad)
$\omega_a(k)$ = azimuth rate (rad/s)

The matrix $\underline{\Phi}$ =
$$\begin{pmatrix} 1,0 & 0,05 & 0 & 0 & 0 & 0 \\ 0 & 1,0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1,0 & 0,05 & 0 & 0 \\ 0 & 0 & 0 & 1,0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1,0 & 0,05 \\ 0 & 0 & 0 & 0 & 0 & 1,0 \end{pmatrix}$$

is the state transition matrix, and

$$\underline{\Gamma}(k) = \begin{pmatrix} 0,4 & 0 & 0 \\ 1,5 & 0 & 0 \\ 0 & 0,05 & 0 \\ 0 & 0,3\times\frac{R(0)}{R(k)} & 0 \\ 0 & 0 & 0,05\times\frac{R(0)}{R(k)}/\cos\alpha_e(k) \\ 0 & 0 & 0,3\times\frac{R(0)}{R(k)}/\cos\alpha_e(k) \end{pmatrix}$$

is the input transition matrix

$w(k)$ and $v(k)$ zero mean white noise processes with

cov $\{w(k), w(k)\} = W \delta(k-j)$

and cov $\{v(k), v(k)\} = V \delta(k-j)$

and $Z(k) = \begin{pmatrix} Z_R(k) \\ Z_e(k) \\ Z_a(k) \end{pmatrix}$ the measurement process

and $H = \begin{pmatrix} 1,0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1,0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1,0 \end{pmatrix}$

The filter algorithm is then given by

$$\underline{x}(k+1) = \Phi \underline{x}(k) + \underline{K}(k+1)\,[\underline{Z}(k+1) - \underline{H}\,\Phi\,\underline{x}(k)] \qquad 5.13$$

$\underline{K}(k+1)$ is the filter gain, and is found from

$$\underline{K}(k+1) = \underline{V}_X((k+1)|k)\,\underline{H}^T\,[\underline{H}\,\underline{V}_X((k+1)|k)\,\underline{H}^T + \underline{V}]^{-1} \qquad 5.14$$

and $\underline{V}_X(k)$ is the state covariance matrix from

$$\underline{V}_X((k+1)|k) = \Phi\,\underline{V}_X(k)\,\Phi^T + \Gamma(k)\,W\,\Gamma^T(k) \qquad 5.15$$

$$\text{and } \underline{V}_X(k+1) = [\underline{I} - \underline{K}(k+1)\,\underline{H}]\,\underline{V}_X((k+1)|k) \qquad 5.16$$

The initial matrix $\underline{V}_X(0)$ is given by

$$\underline{V}_X(0) = \underline{I}$$

The noise covariance matrices are

$$\underline{V} = \begin{pmatrix} 1,0 & 0 & 0 \\ 0 & 10,0 & 0 \\ 0 & 0 & 10,0 \end{pmatrix} = \underline{W}$$

The values of the constants were found by simulation of the process, and the results obtained are described below.

5.6.4 Simulation Results

The matrix equations were implemented on a digital computer and the dive attack profile was generated. Appropriate "measurements" (range, elevation and azimuth rates) were made and corrupted with zero mean, gaussian noise. The response and filtering effect of the algorithms could then be studied and evaluated.

Fig. 5.11 shows the results obtained for the range algorithm. The best estimate of the range had a deviation of approximately 1m, but to improve this the filter tracking ability would become

degraded, and the bias at high range rates became unacceptably large. As it was, at fly-over there was a large bias in the results (Fig. 5.12) but this was not thought to be serious, as the optical director was mechanically unable to cope with such a situation anyway.

Fig. 5.13 gives the equivalent results for the elevation algorithm. Results for the azimuth algorithm were not obtained, but they were expected to be similar.

The filtering effect could be illustrated by a Bode plot. Consider the algorithm block diagram of Fig. 5.19. The discrete time transfer function for range was found to be

$$\left(\frac{O}{i}\right)_R = \frac{K_2 Z - K_2 + K_3 K_1}{Z^2 - (2-K_2)Z - K_2 + K_3 K_1 + 1} \qquad 5.17$$

and for elevation/azimuth

$$\left(\frac{O}{i}\right)_{e/a} = \frac{K_5 Z - K_5 + K_6 K_4}{(Z-1)(Z+K_4-1)} \qquad 5.18$$

Applying the bilinear transformation

$$Z = \frac{1+w}{1-2}$$

the following resulted

$$\left(\frac{O^*}{i^*}\right)_R = A\,\frac{(1-w)\,(w + K_3K_1/(2K_2-K_3K_1))}{K_3K_1 + (2K_2 - 2K_3K_1)\,w + (4-2K_2 + K_3K_1)\omega^2} \qquad 5.19$$

$$\left(\frac{O^*}{i^*}\right)_{e/a} = B\,\frac{(1-w)\,(w + K_6K_4/(2K_5-K_6K4))}{w\,(w + K_4/(2-K_4))} \qquad 5.20$$

The Bode plot is sketched in Figure 5.14 for a given set of gains, found during simulation of the algorithms.

The tracking filter algorithms were also implemented on the simulator. Here it was found that the accuracy of the estimated states was very dependent on the performance of the human operator, as could be expected. If the tracking error was kept below 1,2mrad, the elevation/azimuth angles were correct to within 1mrad deviation, and the rates to 1mrad/s deviation (see Table 5.2), but these estimated states developed large bias errors as soon as the tracking error increased.

This again illustrated the critical role of the human operator in the tracking loop.

5.6.5 Bias Correction

It was possible that large tracking filter biases could be

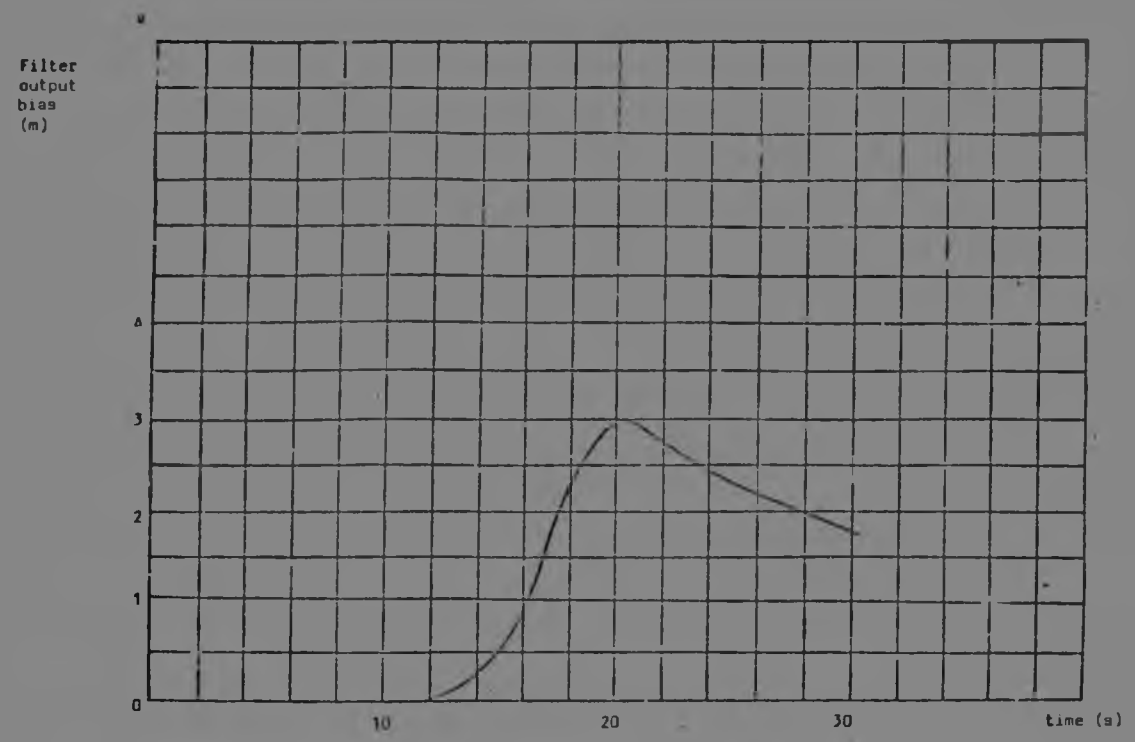

FIGURE 5.12 RANGE FILTER OUTPUT BIAS

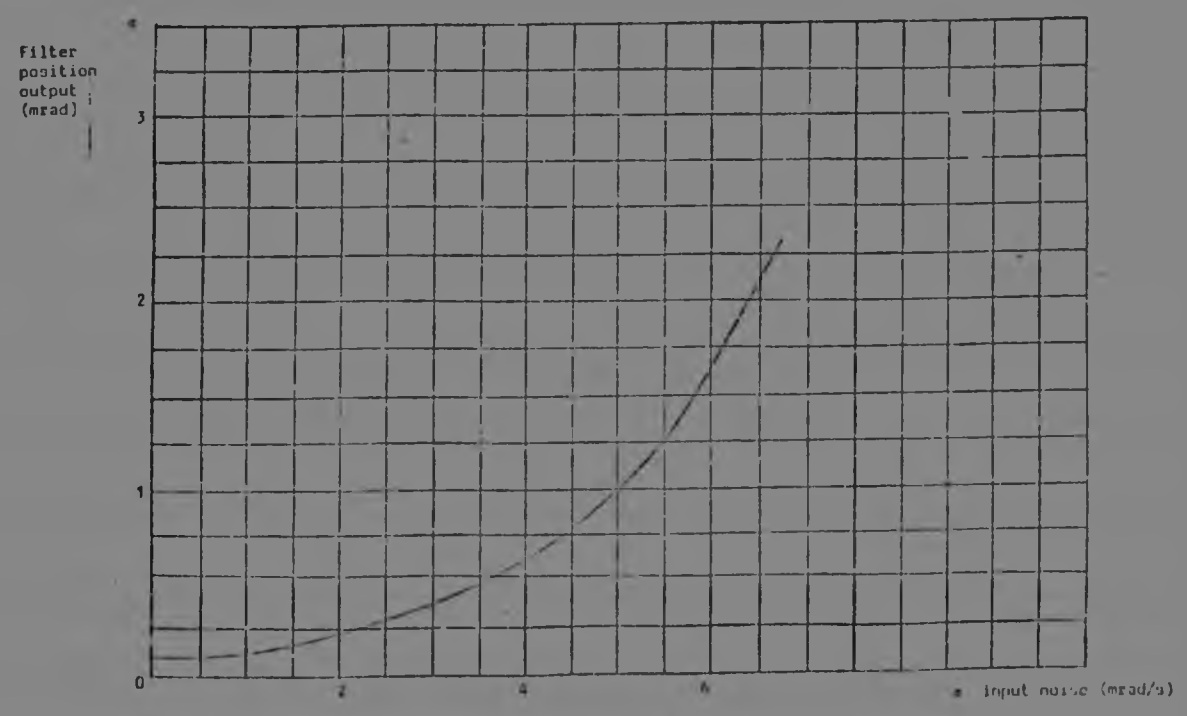

FIGURE 5.13 ELEVATION FILTER OUTPUT DEVIATION

developed. For this reason some form of correction would be needed over longer periods. It was therefore advisable that the position transducer outputs on the cluster and gimbal be transformed to inertial space, converted to cluster axes and he compared with the filter algorithm outputs. These outputs would then be updated periodically.

If many updates were available, the mean filter algorithm output would be the same as the actual measurements, as the measurement bias was zero.

The problem was the time required for the transformation. This would depend on the amount of time available in the computer, and the update or correction period would therefore be determined by computer limitations.

5.6.6 Parameter Sensitivity Analysis

The constants specified above for the algorithms were chosen so that optimal filter response and noise rejection was obtained. It was therefore desirable that they should be implemented as accurately as possible. It was, however, probable that further tuning of the filter would be required during testing and evaluation of the tracking system, as the values of the constants were obtained using simulation techniques. The sensitivity of the filter response to changes in parameter values is therefore discussed.

TABLE 5.2 TYPICAL TRACKING FILTER OUTPUT ACCURACIES (RMS)

PROFILE	SIGHTLINE POSITION ERROR (mrad)	ESTIMATED SIGHTLINE RATE ERROR (mrad)	ESTIMATED SIGHTLINE POSITION ERROR (mrad)
Medium toss bombing attack	1,4	0,42	0,30
Missile attack	0,6	0,12	0,04
Dive attack (Before cross-over)	1,5	0,64	0,80
Dive attack (including cross over)	2,5	1,20	25,00

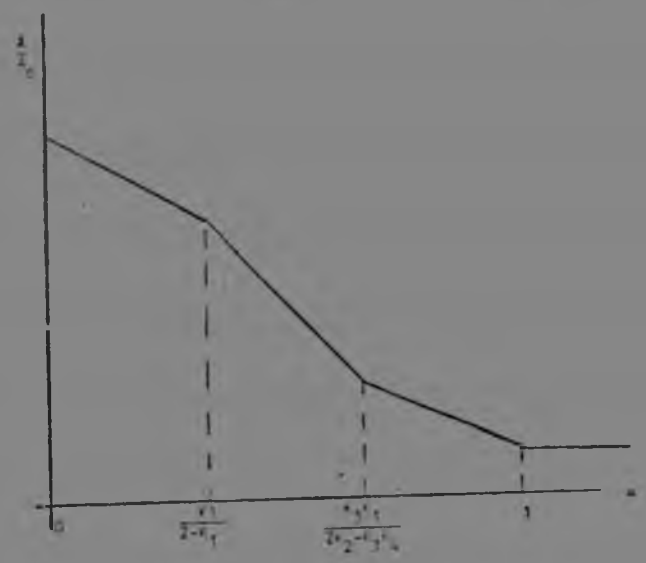

FIGURE 5.14 FILTER ALGORITHM BODE PLOT

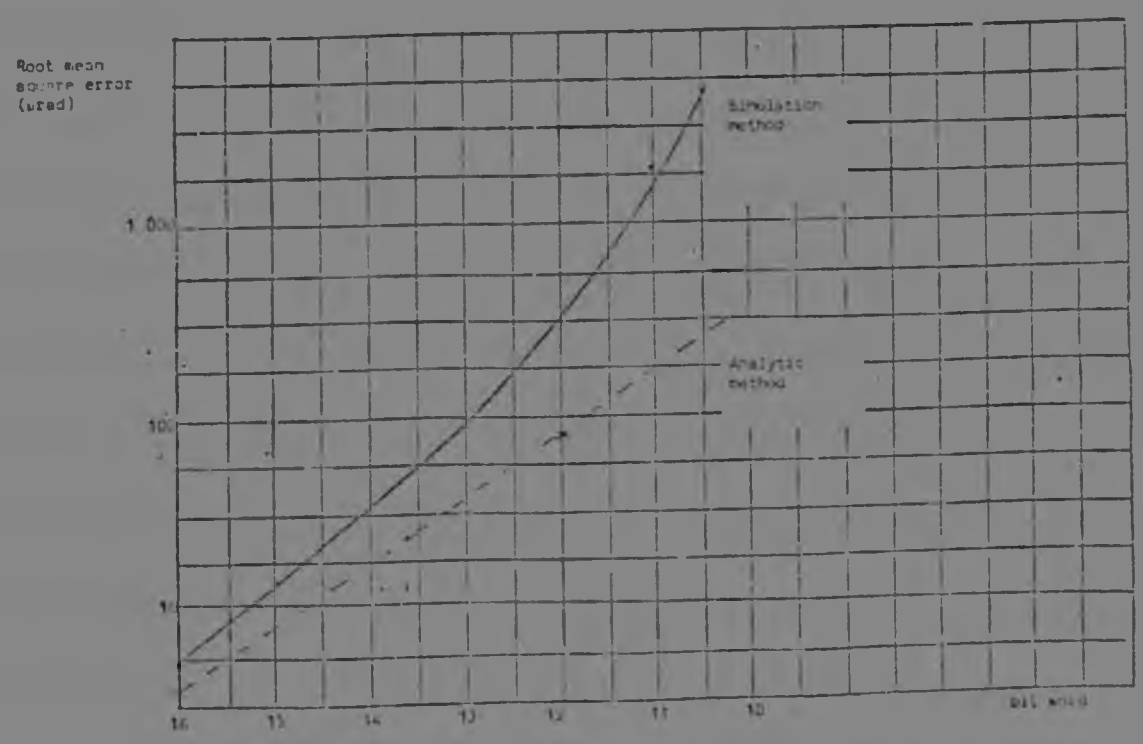

FIGURE 5.15 INPUT QUANTIZATION ERRORS

5.6.6.1 The Message Model

As this equation (equation 5.11) represents the kinematics of the target in inertial space, changes in these values of $\underline{\Phi}$ would drastically alter the filter behaviour. Alteration would imply redesign of the whole filter.

5.6.6.2 Gain Calculation

This is where tuning of the filter should be done. As there are a lot of parameters that could be changed, the sensitivity of each is discussed.

The $\underline{V}$ and $\underline{W}$ matrics should not be changed, except if the noise characteirstics of the process differed considerably from those considered during simulation.

The $\underline{r}$ matrix parameters had a sensitivity as follows: A 10% change in the first two parameters of the $\underline{r}$ matrix resulted in a 10% change in the deviation of the range and range rate filtered outputs.

A 10% variation in the other parameter values had an overall 10% variation in the deviation of either azimuth or elevation angle and rate filtered outputs. These outputs are coupled, so that the effect of changing the value of one parameter could not be predicted easily. Trial and error methods for filter tuning will have to be applied.

5.6.7 Sources of Error

The tracking filter outputs were intended to reflect accurately the position of the line-of-sight between the tracker and the target. Below are listed the sources of error which affected the accuracy of the filter outputs.

5.6.7.1 Filter parameter Errors

These arise mainly from target modelling inaccuracies and have been discussed above (see Table 5.2).

5.6.7.2 Digital Signal Processing Errors

When processing digital signals, three inherent sources of error arise: errors due to quantization of the input signal (sampling), round-off error during processing (i.e. in calculations) and quantization error of constants (Peled and Liu (reference 19)). All processes described above have been simulated on 32 bit computers, using floating point notation

(single precision) which used 24 bits to handle the mantissa of a given number. The microprocessor on which the algorithms were to be implemented in the tracker controller also had 32 bits with 24 bits used to handle the mantissa. This was considered accurate for the purposes of this study, so that round-off and coefficient errors were disregarded. The analog to digital converter however, through which the input signals had to pass, used at best 16 bits. The error due to quantization of the input was therefore investigated.

Takahashi et al (reference 25) derived a relationship between the mean square error (e) between an analog input signal and a digital output signal due to quantization interval (Δ)

$$e^2 = \frac{\Delta^2}{12}$$

The graph of e vs Δ is shown in Figure 5.15 (dashed line).

The tracking filter algorithm simulation was subjected to a known input. The output for quantization interval corresponding to 24 bits was used as standard, and the outputs for progressively larger quantization step sizes were compared. The graph is shown in Figure 5.15 (solid line).

It can be seen that, in order to maintain processing accuracy comparable to the accuracy of the measured data (40 rad rms) analog to digital conversion resolution of at least 15µbits should be used.

A further possible error could indirectly have been caused by the digital processing of the signal. Due to the nature of the process, the estimated target rate from the tracking aid was added to the loop approximately 45ms after the measurement had been made. The interval between samples was 50ms. This delay did not affect the performance of the tracker, nor was there any perceptible jitter when this was implemented on the simulator.

5.6.7.3 Transducer Noise

In chapters 3 and 4 the effect of noise in the angular transducer readings has been discussed. The overall result is to create a stabilization error of 0,5mrad RMS. This was simulated (see paragraph 5.2.1), so that the filtered parameter erros (Table 5.2) include errors due to transducer noise.

5.7 Summary

This chapter has shown that, assuming a stabilized platform, and using simulator techniques, it was possible for a human operator to track the targets discussed in Chapter 3 to the accuracy required by the system. This was, however, only possible using a tracking aid, and it was found that a pro= portional-plus-integral element gave the best results. This element could then easily be expanded into a filtering algorithm which gave best estimates of target states.

In paragraph 5.2 the construction of the tracker simulato was discussed. This simulator was used to give data for a particular tracking task from which the parameters for a linear transfer function of the human operator could be found. The human operator delay was estimated using cross-correlation techniques. In paragraph 5.4 a linear analysis of the tracker was done to investigate the stability of the loop. Already here it was shown that a proportional-plus-integral element improved the gain margin of the system.

In paragraph 5.5 tracking performance results of some compensated and uncompensated systems were compared. It was clear that the proportional-plus-integral element implementation gave a sufficiently good performance improvement to allow the trained human operator to meet the tracking accuracy specifica= tions. The final requirement of the tracker was to find the states of the target in inertial space, which was done by expanding the tracking aid into a linear filtering algorithm, shown in para= graph 5.6. This algorithm was evaluated and the sensitivity of parameters was investigated. Results indicate that the algorithm was stable and robust, and it was well suited to the application.

6 RESULTS AND CONCLUSIONS

In this study a number of aspects of the design of optical trackers not covered in the literature have been considered with application to a specific case. It has been shown that, provided the tracker motion can be stabilized, a tracking aid can be found which will enable the human operator to use the tracker to track high-speed manoeuvring targets to ~ accuracy sufficient for possible engagement of the target.

In order to compensate for the tracker motion, an element of the optics, in this case a mirror, was mounted on a gimballed platform so that it could be stabilized. The equations of motion were derived and, considering the tracker motion and the actual mechanical configuration of the system, feedback loops could be specified which simulation showed could stabilize the sightline to within 0,5mrad RMS for reasonable periods of time both for rotational and linear base motion. Tracker motion and target motion could then effectively be decoupled.

Consequently, a simulator was constructed which generated typical target attack profiles for the operator to track, against which his performance was measured. A simple linearized transfer function of the system including the operator was derived which enabled system stability to be checked. It was clear that the introduction of a proportional-plus-integral element would improve the gain margin for increased system stability.

This element was then implemented as a tracking aid, and the performance of the operator was measured and compared with the unaided or partly aided (non-linear joystick transfer function) case. It was now shown that the target could be tracked to within the required 1,5mrad RMS error by a trained human operator. Difficulties posed by the design around a human operator have therefore been overcome by simulation techniques.

Finally, the estimation of target states was looked into, and it was found that an optimal linear filter could be simply implemented. Simulation showed that good state estimation could be obtained and that the algorithm was robust.

It can therefore be said that the aim of this study, namely to find the configuration and parameters necessary to track manoeuvring targets accurately in the presence of tracker motion has been achieved. The results have also been used as specifications for a tracker which is being built. A summary of the relevant specifications follow:

6.1 STABILIZATION LOOPS

Rotational motion stabilization was achieved using the control loop given in block diagram form in Figure 4.4. For the elevation and azimuth loops the compensator was specified as

$$\frac{C\,(\tau_{z1}s+1)(\tau_{z2}+1)}{(\tau_{p1}a+1)(\tau_{p2}s+1)}$$

τ_{z1} = 0,050s ±10%
τ_{p1} = 0,010s ±10%
τ_{z2} = 0,005s ±10%
τ_{p2} = 0,0014s±10%
C = 15 ±5%

Linear motion stabilization was achieved using the scheme given in Figure 4.15. A digital integration process was specified with maximum step size of 0,01s and maximum sampling inteval of 0,25s.

The pedestal follow-up loop compensator was specified as

$$\frac{1+p\tau_z s}{1+p\tau_p s}$$

$p\tau_z$ = 0,12s ± 10%
$p\tau_p$ = 0,01s ± 10%

with loop gain as given in Figure 4.21 and feedforward gain

PGAN = 0,3 x |PGAIN| ± 5%

6.2 TRACKING AID AND FILTER

The algorithm specified is described in Figure 5.10 and was to have a cycle time of 50ms. The matrix equations of the algorithm and coefficient values were specified as given in paragraph 5.6.3.

7 SUGGESTIONS FOR FURTHER STUDY

This dissertation has clearly shown how critical a part the performance of the human operator plays in the tracking accuracy of the whole system, A lot of emphasis has therefore to be placed on the training of the human operator.

A study should therefore be undertaken to determine the best training program which will integrate the operator and the system to achieve optimal overall performance. Only then can maximum tracker effectivity be attained.

It would also be very interesting to determine how a tracking aid, which adapts to the performance of the human operator, affects the overall system accuracy. This would involve a fast and efficient parameter estimation algorithm coupled in some way to the tracking aid.

THE DEVELOPMENT, ANALYSIS AND EVALUATION OF AN OPTICAL TRACKER FOR TRACKING HIGH-SPEED MANOEUVERING TARGETS

VOLUME II

RONALD ALFRED BERND SCHNEIDER

A DISSERTATION SUBMITTED TO THE FACULTY OF ENGINEERING, UNIVERSITY OF THE WITWATERSRAND, JOHANNESBURG, FOR THE DEGREE OF MASTER OF SCIENCE

PRETORIA, APRIL 1981

CONTENTS

APPENDIX A

This appendix contains a set of computer program listings. These programs and subroutines are written in DSL77 or FORTRAN and were used to obtain the results for this dissertation. It must be noted that the output statements given in these listings do not cover all figures, curves and tables in this dissertation, but they can be obtained by a simple modifi= cation. On pages A2 to A30 are the listings of programs used in conjunction with the analyses of Chapter 4.

On pages A31 to A42 are the listings of programs used to obtain the human operator transfer function (Chapter 5).

On pages A43 to A50 are the listings of programs used in the Optical tracker simulator (Chapter 5).

On pages A51 to A58 are the listings of programs used to analyse the performance of the tracking filter (Chapter 5).

```
C*****************************************
C     THIS PROGRAM FINDS THE TRANSFER     *
C     FUNCTION OF A SIMPLIFIED AND LINEA- *
C     RIZED GIMBAL/CLUSTER STABILIZATION  *
C     LOOP USING SIGNAL FLOW NOTATION.    *
C                                         *
C     SUBROUTINES USED:                   *
C       SEE ALERTS (REFERENCE 1) FOR      *
C       A DETAILED DESCRIPTION.           *
C                                         *
C*****************************************
C
C     PARAMETERS
C
C  LOOP GAIN (V/V)
      CONS = 15.0
C     DO 100 I=1,10
C     READ (5,*) A
C  GENERALIZED INERTIA (KGM*M) AND DAMPING (NM/RAD/S)
      DJ = .02
      BD = .005
C  GYRO INPUT GAIN (RAD/S/RAD/S) AND PICK-OFF CONSTANT (V/RAD/S)
      TKG = 1.000
      PK = 859.50
C  TORQUER CONSTANT (NM/A) AND RESISTANCE (OHMS)
      TKK = .031
      R=4.3
C  BACK EMF (V/RAD/S)
      EMF = .029
C
C     START ANALYSIS
C
      CALL DYSA(0)
      CALL PATH( 1. 2.  1.      . 0)
      CALL PATH( 2. 3.  TKG     . 0)
      CALL PATH( 3. 4.  1.      .-1)
      CALL PATH( 4. 5.  PK      . 0)
      CALL PATH( 5. 6.  1.      . 0)
      CALL CMPXPL(5,12,0.5,3257.3)
      CALL LEDLAG(12,17,0.050,0.010)
      CALL LEDLAG(17,23,.0050,.0014)
      CALL PATH(23,24.  1.0     . 0)
      CALL PATH(24,25.  CONS    . 0)
      CALL PATH(25,26.  1./R    . 0)
      CALL PATH(26,27.  TKK     . 0)
      CALL PATH(27,28.  1.      . 0)
      CALL PATH(29,28.  1./DJ   .-1)
      CALL PATH(29,28.  -BD     . 0)
      CALL PATH(28,25.  -EMF    . 0)
      CALL PATH(29,30.  1.      .-1)
      CALL PATH(29, 3.  -1.0    . 0)
      CALL SOLVE(27,30)
C
C     DO SENSITIVITY ANALYSIS
C
```

```
C      CALL SENST(1P,19.2,,50.,6.,1.23,-300.)
C
C      FIND STEP RESPONSE
C
C      CALL TRESP(.1)
C
C      FIND FREQUENCY RESPONSE
C
C      CALL FRESP (.1,1.,0)
C
  100  CONTINUE
       STOP
       END
```

```
C*****************************************
C     THIS PROGRAM FINDS THE TRANSFER     *
C     FUNCTION OF THE LINEARIZED PEDESTAL *
C     FOLLOW-UP LOOP. A LINEARIZED MODEL  *
C     OF THE CLUSTER/GIMBAL IS INCORPO-   *
C     RATED.                              *
C                                         *
C     SUBROUTINES USED:                   *
C       SEE ALEPS (REFERENCE 1) FOR A     *
C       DETAILED DESCRIPTION.             *
C*****************************************
C
C
C     GIMBAL/CLUSTER LOOP PARAMETERS
C
C  LOOP GAIN (V/V)
      CONS = 15.0
C     DO 400 I=1,10
C     READ (5,*) A
C  GENERALIZED INERTIA (KGM*M) AND DAMPING (NM/RAD/S)
      DJ = .02
      BD = .005
C  GYRO INPUT GAIN (RAD/S/RAD/S) AND PICK-OFF CONSTANT (V/RAD/S)
      TKG = 1.000
      RK = 859.50
C  TORQUER CONSTANT (NM/A) AND RESISTANCE (OHMS)
      TKK = .031
      R=4.3
C  BACK EMF (V/RAD/S)
      EMF = .029
C
C     PEDESTAL FOLLOW-UP LOOP PARAMETERS
C
C  SERVO AMP GAIN (V/V) AND C     NT FEEDBACK GAIN (V/A)
      SAG = 10.5
      PCFG = 0.18
C  SERVO RESISTANCE (OHMS) .CONSTANT (NM/A) AND TIME CONSTANT (RAD/S)
      PRA = 1.05
      PTK = 1.60
      PTF = .0057
C  LOOP GAIN (RAD/V)
      PGAIN = 35.
C  PEDESTAL INERTIA (KGM*M) AND DAMPING (NM/RAD/S)
      PJ = 10.26
      PB = 0.20
C
C     ANALYSIS
C
C  CLUSTER/GIMBAL LOOP
      CALL DYSA(0)
      CALL PATH( 1. 2.  1.      . 0)
      CALL PATH( 2. 3.  TKG     . 0)
      CALL PATH( 3. 4.  1.      .-1)
      CALL PATH( 4. 5.  PK      . 0)
      CALL PATH( 5. 6.  1.      . 0)
```

```
      CALL CMPXPL(6.12.0.5.7267.3)
      CALL LEDLAG(12.17.0.050.0.010)
      CALL LEDLAG(17.23..0050..0014)
      CALL PATH(23.24.   1.0     . 0)
      CALL PATH(24.25.   CONS    . 0)
      CALL PATH(25.26.   1./R    . 0)
      CALL PATH(26.27.   TKK     . 0)
      CALL PATH(27.28.   1.      . 0)
      CALL PATH(28.29.   1./DJ   .-1)
      CALL PATH(29.28.   -BD     . 0)
      CALL PATH(29.30.   1.      .-1)
      CALL PATH(29. 3.   -1.0    . 0)
C  PEDESTAL FOLLOW-UP LOOP
      CALL PATH( 5.31.    1.     . 0)
      CALL PATH(30.31.    PGAIN  . 0)
      CALL LEDLAG(31.40.0.030.0.0100)
      CALL PATH(40.41.    1.     . 0)
      CALL PATH(41.42.    SAG    . 0)
      CALL PATH(42.43.    1./PPA . 0)
      CALL REALPL(47.48.PTE)
      CALL PATH(48.49.    PTK    . 0)
      CALL PATH(48.41.    -PCFG  . 0)
      CALL PATH(49.50.    1./PJ  .-1)
      CALL PATH(50.49.    -PD    . 0)
      CALL PATH(50. 3.    -1.    . 0)
      CALL SOLVE(1.50.1)
C
C     DO SENSITIVITY ANALYSIS
C
C     CALL SENST(4.2.40..4000..50..4.20.-200.)
  400 CONTINUE
      STOP
      END
```

```
*  ********************************************
*       THIS PROGRAM USES A DETERMINISTIC BASE   *
*       MOTION PERSE ROTATION AND A NON-LINEAR   *
*       MODEL OF THE GIMBAL/CLUSTER DYNAMICS TO  *
*       TEST THE EFFECTIVENESS OF THE STABILI-   *
*       ZATION LOOPS.                            *
*                                                *
*       SUBROUTINES USED:                        *
*         BASEM                                  *
*         GYMBAL                                 *
*  ********************************************
*
*      PARAMETER DEFINITIONS
*
PARAM PHI0=0.0000....   :GIMBAL ANGLE INIT. COND. (RAD)
      THE0=0.0000....   :CLUSTER ANGLE INIT. COND. (RAD)
      ETAP=0.000....    :PEDESTAL ROTATION ANGLE (RAD)
      WCD0=0.0....      :CLUSTER ROTATION RATE INIT. COND. (RAD/S)
      WOJ0=0.0....      :GIMBAL ROTATION RATE INIT. COND. (RAD/S)
      ETA0=0.0....      :INITIAL CLUSTER ANGLE R-L. TO GIMBAL (RAD)
      EPS0=0.0....      :INITIAL GIMBAL ANGLE REL. TO PEDESTAL (RAD)
      CGY0=0.....       :CLUSTER GYRO ERROR INIT. COND. (RAD)
      GGY0=0.....       :GIMBAL GYRO ERROR INIT. COND. (RAD)
      GKP=850.5....     :GYRO PICK-OFF GAIN (V/RAD)
      KGFB=1.00....     :ESTIMATED GYRO FEEDBACK GAIN (RAD/S/V)
      CGAIN=15.000....  :CLUSTER LOOP GAIN (V/V)
      GGAIN=15.000....  :GIMBAL LOOP GAIN (V/V)
      TK=0.031....      :TORQUER CONSTANT (NM/A)
      TR=4.3....        :TORQUER RESISTANCE (OHMS)
      CBAT=0.00840....  :CLUSTER STICTION (NM)
      GBAT=0.00840....  :GIMBAL STICTION (NM)
      BEMF=0.029....    :BACK EMF (V/RAD/S)
      TZ1=0.050....     :COMPENSATOR ZERO TIME CONSTANT (RAD/S)
      TZ2=0.005....     :COMPENSATOR ZERO TIME CONSTANT (RAD/S)
      TP1=0.010....     :COMPENSATOR POLE TIME CONSTANT (RAD/S)
      TP2=0.001A....    :COMPENSATOR POLE TIME CONSTANT (RAD/S)
      TDD=0.0....       :CLUSTER LOOP INPUT SIGNAL (RAD/S)
      TJD=0.0....       :GIMBAL LOOP INPUT SIGNAL (RAD/S)
      ETATP=1.....      :CLUSTER INITIALIZATION RATE (RAD/S)
      EPSTP=1.....      :GIMBAL INITIALIZATION RATE (RAD/S)
      ZCROSS=1.....     :CLUSTER ZERO-CROSS INITIALIZER
      ZKROSS=1.         :GIMBAL ZERO-CROSS INITIALIZER
*
CONTRL DELT..0005,FINTI.100.0
INTEG ADAMS
*
INITIAL
*
DYNAMIC
*
      WCD = INTGR(WCD0,WCDT)     :CLUSTER ROTATION RATE (RAD/S)
      WOJ = INTGR(WOJ0,WOJT)     :GIMBAL ROTATION RATE (RAD/S)
      THE = INTGR(THE0,WCD)      :CLUSTER INERTIAL ANGLE (RAD)
      PHI = INTGR(PHI0,WOJ)      :GIMBAL INERTIAL ANGLE (RAD)
      THE2I = INTGR(THE0,THE2)   :INTEGRAL TO FIND RMS ERROR
```

```
      PHI21 = INTGR(PHI0,PHI2)   :INTEGRAL TO FIND RMS ERROR
      ETA = INTGR(ETA0,ETAT)     :CLUSTER ANGLE REL. TO GIMBAL (RAD)
      EPS = INTGR(EPS0,EPST)     :GIMBAL ANGLE REL. TO PEDESTAL (RAD)
      CGY = INTGR(CGY0,CGYT)     :CLUSTER GYRO ANGLE ERROR (RAD)
      GGY = INTGR(GGY0,GGYT)     :GIMBAL GYRO ANGLE ERROR (RAD)
*
*     FIND DETERMINISTIC BASE MOTION REPRESENTATION
*
PROCED THET,PHIT,ZT=DUM(TIME)
      CALL BASEM(TIME,ZT,PHIT,THET)
ENDPRO
*
*  TRANSFORM TO PEDESTAL ROTATION RATES
*
      WBX=-THET*SIN(ETAP) + PHIT*COS(ETAP)
      WBY= THET*COS(ETAP) + PHIT*SIN(ETAP)
      WBZ= 0.
*
*        CONTROL LOOP COMPONENTS
*
*       C = CLUSTER
*       G = GIMBAL
*       GY = GYRO
*       GP = GYRO PICKOFF
*       DM = DEMODULATOR
*       SN = SHAPING NETWORK
*       PT = PLATFORM TORQUER
*       O = OUTPUT
*
*  GYRO INPUT RATE ERROR SIGNAL
      CGYT = TDD-WCO
      GGYT = TJD-WOJ
*  GYRO PICK-OFF OUTPUT
      CGPO = CGY*GKP
      GGPO = GGY*GKP
*  SHAPING NETWORK OUTPUTS
      CSN01,Y1 = LEDLG(0.,TZ1,TP1,CGPO)
      GSN01,Y2 = LEDLG(0.,TZ1,TP1,GGPO)
      CSN02,Y3 = LEDLG(0.,TZ2,TP2,CSN01)
      GSN02,Y4 = LEDLG(0.,TZ2,TP2,GSN01)
*
*    FIND CLUSTER/GIMBAL MOTION ACCORDING TO A NON-LINEAR MODEL
*
PROCED WCOT,WOJT,ETAT,EPST,CPTO,GPTO = DUM1(ZCROSS,ZKROSS....
         TIME,WCO,WOJ,CSN02,GSN02,ETA,EPS,ETAP,EPSTP,....
         WBX,WBY,WBZ,CBAT,GBAT,ETAP,CGAIN,GGAIN,TK,TR,BEMF)
*
      CALL GYMBAL(WBX,WBY,WBZ,WCO,WOJ,CPTO,GPTO....
         ETAT,ETA,EPST,EP ,ETATP,EPSTP,ZCROSS,ZKROSS....
         CBAT,GBAT,WCO1,WOJT)
*
*  FIND RELATIVE ROTATION RATES CLUSTER:GIMBAL:PEDESTAL
      EPST = WOJ-WBY
      ETAT = WCO-(WBX*SIN(EPS)+WBZ*COS(EPS))
*  FIND TORQUER OUTPUTS
```

```
      CPTO = (CGAIN*GSNO2-BEXF*ETAI)*TK/TR
      GPTO = (GGAIN*GSNO2-HEXF*EPST)*TK/TR
ENDPRO
*
*     FIND RMS STABILIZER MOTION
*
      THE2=THE*THE
      PHI2=PHI*PHI
PROCED CRMSAN,GRMSAN = DUM2(TIME,THE2I,PHI2I,FINTI,DELT)
      IF (TIME .LT. (FINTI-DELT)) GO TO 10
      CRMSAN=SQRT(THE2I/TIME)
      GRMSAN=SQRT(PHI2I/TIME)
   10 CONTINUE
ENDPRO
*
TERMINAL
*
*
PRNTF 2.00,WCO,THE,CRMSAN,WOJ,PHI,GRMSAN
END
STOP
```

```
*  ********************************************
*     THIS PROGRAM USES A DETERMINISTIC BASE   *
*     MOTION REPRESENTATION AND THE NON-LINEAR*
*     MODEL OF THE CLUSTER/GIMBAL DYNAMICS TO  *
*     TEST THE EFFECT OF LINEAR MOTION STABI-  *
*     LIZATION.                                *
*                                              *
*     SUBROUTINES USED:                        *
*       BASEMC                                 *
*       SHPMOT                                 *
*       GYMBAL                                 *
*       LMTRAN                                 *
*       VCROSS                                 *
*       ILMTRN                                 *
*  ********************************************
*
*    PARAMETERS
*
PARAM PHI0=0.0000....    :INITIAL BASE ROLL ANGLE (RAD)
      THE0=0.0000....    :INITIAL BASE PITCH ANGLE (RAD)
      PHID0=0.000....    :INITIAL BASE ROLL RATE (RAD/S)
      THED0=0.000....    :INITIAL BASE PITCH RATE (RAD/S)
      XD0=-0.09874....   :INITIAL BASE FORWARD MOTION (M/S)
      YD0=-0.806?....    :INITIAL BASE LATERAL MOTION (M/S)
      ZD0=-1.20?....     :INITIAL BASE VERTICAL MOTION (M/S)
      PSI=0.000....      :BASE YAW ANGLE (RAD)
      ETAP=0.0....       :PEDESTAL ANGLE (RAD)
      WC00=0.0....       :INITIAL CLUSTER RATE (RAD/S)
      WDJ0=0.0....       :INITIAL GIMBAL RATE (RAD/S)
      ETA0=0.0....       :INITIAL CLUSTER ANGLE RELATIVE TO GIMBAL (RAD)
      EPS0=0.00....      :INITIAL GIMBAL ANGLE RELATIVE TO PEDESTAL (RAD)
      ZMD0=0.0....       :INITIAL MEASURED VERTICAL MOTION (M/S)
      YMD0=0.0....       :INITIAL MEASURED LATERAL MOTION (M/S)
      XMD0=0.0....       :INITIAL MEASURED FORWARD MOTION (M/S)
      XM0=0.0....        :INITIAL MEASURED FORWARD POSITION (M)
      YM0=0.0....        :INITIAL MEASURED LATERAL POSITION (M)
      ZM0=-0.0....       :INITIAL MEASURED VERTICAL POSITION (M)
      TAV=150.00....     :AVERAGING TIME (S)
      CDIF0=0.0....      :CLUSTER ERROR ANGLE INITIAL CONDITION (RAD)
      GDIF0=0.0....      :GIMBAL ERROR ANGLE INITIAL CONDITION (RAD)
      SW1=1.0....        :AVERAGE CALCULATION SWITCH
      CSLA0=0.0000....   :CLUSTER SIGHTLINE ANGLE INIT. COND. (RAD)
      GSLA0=0.0000....   :GIMBAL SIGHTLINE ANGLE INIT. COND. (RAD)
      PHASE=1.000....    :PHASE ANGLE (RAD)
      THASE=1.000....    :PHASE ANGLE (RAD)
      ZASE=1.000....     :PHASE ANGLE (RAD)
      ZCLUS=10.000....   :CLUSTER DISPLACEMENT FROM CENTRE OF ROT. (M)
      CGY0=0.....        :INITIAL CLUSTER GYRO OUTPUT
      GGY0=0.....        :INITIAL GIMBAL GYRO OUTPUT
      GGAIN =15.000....  :GIMBAL LOOP GAIN (V/V)
      CGAIN =15.000....  :CLUSTER LOOP GAIN (V/V)
      CLMGN=1.000....    :CLUSTER LINEAR MOTION LOOP GAIN (RAD/S/RAD/S)
      GLMGN=1.000....    :GIMBAL LINEAR MOTION LOOP GAIN (RAD/S/RAD/S)
      RANGE=2000.0....   :TARGET RANGE (M)
      GKP=850.5....      :GYRO PICK-OFF GAIN (V/RAD)
```

```
      TK=0.071....          :TORQUER CONSTANT (NM/A)
      TR=4.2....            :TORQUER RESISTANCE (OHMS)
      GBAT=0.00940....      :GIMBAL STICTION CONSTANT (NM)
      CBAT=0.00940....      :CLUSTER STICTION CONSTANT (NM)
      BEMF=0.029....        :BACK EMF (V/RAD/S)
      TZ1=0.050....         :ROT. STAB. LOOP COMPENSATOR ZERO (RAD/S)
      TZ2=0.005....         :ROT. STAB. LOOP COMPENSATOR ZERO (RAD/S)
      TP1=0.010....         :ROT. STAB. LOOP COMPENSATOR POLE (RAD/S)
      TP2=0.0014....        :ROT. STAB. LOOP COMPENSATOR POLE (RAD/S)
      TDD=0.0....           :CLUSTER LOOP INPUT COMMAND (RAD/S)
      TJD=0.0....           :GIMBAL LOOP INPUT COMMAND (RAD/S)
      ETATD=1.....          :CLUSTER MODEL INITIALIZER
      EPSTD=1.....          :GIMBAL MODEL INITIALIZER
      ZCROSS=1.....         :ZERO-CROSS INITIALIZER
      ZKROSS=1.....         :ZERO-CROSS INITIALIZER
      SINT=0.25             :LINEAR MOTION SAMPLING INTERVAL (S)
*
CONTRL DELT=.0005,CINTI=220.0
INTEG ADAMS
*
INITIAL
      SAMPT = 0.0           :INITIAL SAMPLE TIME (S)
      CDIF = 0.0            :INITIAL CLUSTER ANGLE DIFFERENCE (RAD)
      GDIF = 0.0            :INITIAL GIMBAL ANGLE DIFFERENCE (RAD)
*
DYNAMIC
*
      ZDT = INTGR(ZD0,ZDDT)       :VERTICAL BASE MOTION (M/S)
      YDT = INTGR(YD0,YDDT)       :LATERAL BASE MOTION (M/S)
      XDT = INTGR(XD0,XDDT)       :FORWARD BASE MOTION (M/S)
      XMDT   INTGR(XMD0,XMDDT)    :MEASURED FORWARD MOTION (M/S)
      YMDT = INTGR(YMD0,YMDDT)    :MEASURED LATERAL MOTION (M/S)
      ZMDT = INTGR(ZMD0,ZMDDT)    :MEASURED VERTICAL MOTION (M/S)
      XM = INTGR(XM0,XMDT)        :MEASURED FORWARD POSITION (M)
      YM = INTGR(YM0,YMDT)        :MEASURED LATERAL POSITION (M)
      ZM = INTGR(ZM0,ZMDT)        :MEASURED VERTICAL POSITION (M)
      THEDT = INTGR(THED0,THEDDT):BASE PITCH RATE (RAD/S)
      PHIDT = INTGR(PHID0,PHIDDT):BASE ROLL RATE (RAD/S)
      THE = INTGR(THE0,THEDT)     :BASE PITCH ANGLE (RAD)
      PHI = INTGR(PHI0,PHIDT)     :BASE ROLL ANGLE (RAD)
      WCD = INTGR(WCD0,WCDT)      :CLUSTER ROTATION RATE (RAD/S)
      WDJ = INTGR(WDJ0,WDJT)      :GIMBAL ROTATION RATE (RAD/S)
      CAN = INTGR(THE0,WCD)       :CALCULATED SIGHTLINE ANGLE (RAD)
      GAN = INTGR(PHI0,WDJ)       :CALCULATED SIGHTLINE ANGLE (RAD)
      ETA = INTGR(ETA0,ETAT)      :CLUSTER ANGLE REL TO GIMBAL (RAD)
      EPS = INTGR(EPS0,EPST)      :GIMBAL ANGLE REL TO PEDESTAL (RAD)
      CSLA = INTGR(CSLA0,SLD)     :CLUSTER SIGHTLINE ANGLE (RAD)
      GSLA = INTGR(GSLA0,SLJ)     :GIMBAL SIGHTLINE ANGLE (RAD)
      CDIFI = INTGR(CDIF0,CDIF)   :DIFFERENCE ANGLE INTEGRAL
      GDIFI = INTGR(GDIF0,GDIF)   :DIFFERENCE ANGLE INTEGRAL
      CGY = INTGR(CGY0,CGYT)      :CLUSTER GYRO INTEGRAL
      GGY = INTGR(GGY0,GCYT)      :GIMBAL GYRO INTEGRAL
*
*     FIND AVERAGE OF MOTION
*
```

```
      PROCED XMDTM=DUN(XMDT,YMDT,ZMDT,XDAV,YDAV,ZDAV,SW,TAV,...
     *        CAN,GAN,CSLA,GSLA)
              IF (SW .LT. 0.) GO TO 5
              XMDT=XMDT-XDAV
              YMDT=YMDT-YDAV
              ZMDT=ZMDT-ZDAV
              CAN=CSLA
              GAN=GSLA
              SW=-2.
            5 CONTINUE
              XMDTM=0.
      ENDPRO
      *
      *     ROTATIONAL MOTION STABILIZATION
      *
      *       C = CLUSTER
      *       G = GIMBAL
      *       GY = GYRO
      *       GP = GYRO PICKOFF
      *       DM = DEMODULATOR
      *       SN = SHAPING NETWORK
      *       PT = PLATFORM TORQUER
      *
      PROCED WCDT,WOJT,CGYT,GGYT,ZT,PHIT,THET,EPST,ETAT=DUMI(...
     *        TIME,WCD,WOJ,CSN02,GSN02,ETA,EPS,ETATP,EPSTP,...
     *        ZCROSS,ZKROSS,CBAT,GBAT,TDD,TJD,ETAP,...
     *        TDDL,TJDL,TK,TR,BEMF,CGAIN,GGAIN,TAV)
      *  ZERO ROTATIONAL MOTION
              CALL BASEMC(TIME,ZT,PHIT,THET)
      *  TRANSFORM TO PEDESTAL ROTATION RATES
      *
              WBX=-THET*SIN(ETAP) + PHIT*COS(ETAP)
              WBY= THET*COS(ETAP) + PHIT*SIN(ETAP)
              WBZ= 0.
              IF (TIME .LE. TAV) GO TO 30
              EPST = WOJ-WBY
              ETAT = WCD-(WBX*SIN(EPS)+WBZ*COS(EPS))
      *  NON-LINEAR CLUSTER/GIMBAL MODEL
              CALL GYMBAL(WBX,WBY,WBZ,WCD,WOJ,C-.0,GPTO,ETAT,ETA,...
     *           EPST,EPS,ETATP,EPSTP,ZCROSS,ZKROSS,CBAT,GBAT,WCDT,WOJT)
              EPST = WOJ-WBY
              ETAT = WCD-(WBX*SIN(EPS)+WBZ*COS(EPS))
      *  GYRO INPUT RATE ERROR SIGNAL
              CGYT = (TDDL+TDD)-WCD
              GGYT = (TJDL+TJD)-WOJ
      *  TORQUER OUTPUTS
              CPTO = (CGAIN*CSN   -BEMF*ETAT)*TK/TR
              GPTO = (GGAIN*GSNO  -BEMF*EPST)*TK/TR
              GO TO 31
           30 EPST=0.0
              ETAT=0.0
              CGYT=0.0
              GGYT=0.0
              WOJ=WBY
              WCD=(WBX*SIN(EPS)+WBZ*COS(EPS))
```

```
      WOJT=0.0
      WCDT=0.0
   31 CONTINUE
ENDPRO
*
*     LINEAR MOTION STABILIZATION
*
PROCED TDDL,TJDL,XDDT,YDDT,ZDDT,THEDDT,PHIDDT,EDT,DDT,EDT,...
        XDAV,YDAV,ZDAV,XMDDT,YMDDT,ZMDDT=DUM2(TIME,DELT,XM,YM,ZM,...
        PHASE,THASE,ZASE,ETAP,ETA,EPS,ZCLUS,XDT,YDT,ZDT,PSI,THE,...
        PHI,RANGE,CLMGN,GLMGN,SAMPT,SINT,TAV,SW,XMDT,YMDT,ZMDT)
*  GENERATE BASE MOTION (ACCELERATIONS)
      CALL SHOMOT(TIME,PHASE,THASE,ZASE,ZDDT,THEDDT,PHIDDT)
*  LINEAR ACCELERATIONS (M/S*S)
      XDDT=THEDDT*ZCLUS
      YDDT=PHIDDT*ZCLUS
*  TRANSFORM TO CLUSTER AXES
      CALL LMTRAN(XDDT,YDDT,ZDDT,PSI,THE,PHI,ETAP,EPS,ETA,RDDT,EDDT,DDDT)
*     IF (TIME .LE.TAV) GO TO 20
*     IF ((TIME-SAMPT) .LT. SINT) GO TO 25
   20 CALL LMTRAN(XDDT,YDDT,ZDDT,PSI,THE,PHI,ETAP,EPS,ETA,RDDT,...
         EDDT,DDDT)
      CALL LMTRAN(XMDDT,YMDDT,ZMDDT,PSI,THE,PHI,ETAP,EPS,ETA,0.....
         EDDT,DDDT)
      CALL LMTRAN(XMDT,YMDT,ZMDT,PSI,THE,PHI,ETAP,EPS,ETA,EMDT,...
         EMDT,DMDT)
      SAMPT = TIME
*  ADD TO CLUSTER/GIMBAL COMMAND
   25 TJDL=(GLMGN*DMDT/RANGE)*COS(ETA)
      TDDL=-(CLMGN*EMDT/RANGE)
*  AVERAGE VELOCITY COMPONENTS
      IF (TIME.EQ.0.)GO TO 10
      XDAV = XM/TIME
      YDAV = YM/TIME
      ZDAV = ZM/TIME
      IF (TIME.LT.TAV) GO TO 10
      IF (TIME.GE.(TAV+DELT)) GO TO 10
      SW = 1.
*     TAV=TAV+1.0
   10 CONTINUE
*
ENDPRO
*
*     FIND ACTUAL SIGHTLINE TUNING RATE ABOUT
*     D (CLUSTER) AND J (GIMBAL) AXES.
*
      SLD=-EDT/RANGE
      SLJ=(DDT/RANGE)*COS(ETA)
*
*    FIND RMS SIGHTLINE ERROR
*
PROCED CRMSER,GRMSER=DUM3(TIME,SW,CAN,CSLA,GAN,GSLA,...
            CDIFI,GDIFI,CDIF,GDIF)
      IF (SW .GE. -1.5) GO TO 15
      CDIF=(CAN-CSLA)*(CAN-CSLA)
```

```
      GDIF=(GAN-GSLA)*(GAN-GSLA)
      CRMSER=SQRT(CDIFI/TIME)
      GRMSER=SQRT(GDIFI/TIME)
   15 CONTINUE
ENDPRO
*
*         CONTROL LOOP COMPONENTS
*
      CGPO = CGY*GKP
      GGPO = GGY*GKP
      CSNO1,Y1 = LEDLG(0.,TZ1,TP1,CGPO)
      GSNO1,Y2 = LEDLG(0.,TZ1,TP1,GGPO)
      CSNO2,Y3 = LEDLG(0.,TZ2,TP2,CSNO1)
      GSNO2,Y4 = LEDLG(0.,TZ2,TP2,GSNO1)
*
TERMINAL
*
PRNTF 2.000,XMDT,XDT,YMDT,YDT,ZMDT,ZDT
PRNTS 5.000,XDAV,YDAV,ZDAV,CSLA,CAN,GSLA,GAN,CRMSER,GRMSER
END
STOP
```

```
*  ************************** ***************
*       THIS PROGRAM CONTAINS A NON-LINEAR      *
*       MODEL OF THE PEDESTAL AND GIMBAL        *
*       DYNAMICS SO THAT THE PEDESTAL FOLLOW-   *
*       UP LOOP CAN BE CHECKED                  *
*                                               *
*       SUBROUTINES USED:                       *
*         GYMBAL                                *
*         AMPPD                                 *
*  ***********************************************
*
*       GIMBAL/CLUSTER DYNAMICS PARAMETERS
*
PARAM PHI0=0.0000....    :GIMBAL ANGLE INIT. COND. (RAD)
      THE0=0.0000....    :CLUSTER ANGLE INIT.COND. (RAD)
      WC00=0.0....       :CLUSTER ROTATION RATE INIT.COND. (RAD/S)
      WOJ0=0.0....       :GIMBAL ROTATION RATE INIT.COND. (RAD/S)
      ETA0=0.0....       :INITIAL CLUSTER ANGLE REL. TO GIMBAL (RAD)
      EPS0=0.0....       :INITIAL GIMBAL ANGLE REL. TO PEDESTAL (RAD)
      CGY0=0.....        :CLUSTER GYRO ERROR INIT. COND. (RAD)
      GGY0=0.....        :GIMBAL GYRO ERROR INIT. COND. (RAD)
      GKP=P5?.?....      :GYRO PICK-OFF GAIN (V/RAD)
      GKFB=1.00....      :ESTIMATED GYRO FEEDBACK GAIN (RAD/S/V)
      CGAIN=15.000....   :CLUSTER LOOP GAIN (V/V)
      GGAIN=15.000....   :GIMBAL LOOP GAIN (V/V)
      TK=0.031....       :TORQUER CONSTANT (NM/A)
      TR=4.30....        :TORQUER RESISTANCE (OHMS)
      BEMF=0.029....     :BACK EMF (V/RAD/S)
      CBAT=0.03840....   :CLUSTER STICTION (NM)
      GBAT=0.00?40....   :GIMBAL STICTION (NM)
      TZ1=0.05....       :COMPENSATOR ZERO TIME CONSTANT (RAD/S)
      TZ2=0.005....      :COMPENSATOR ZERO TIME CONSTANT (RAD/S)
      TP1=0.010....      :COMPENSATOR POLE TIME CONSTANT (RAD/S)
      TP2=0.0014....     :COMPENSATOR POLE TIME CONSTANT (RAD/S)
      ETTP=1.0....       :CLUSTER INITIALIZATION RATE (RAD/S)
      EPSTP=1.0....      :GIMBAL INITIALIZATION RATE (RAD/S)
      ZCROSS=1.0....     :CLUSTER ZERO-CROSS INITIALIZER
      ZKROSS=1.0....     :GIMBAL ZERO-CROSS INITIALIZER
      TJD=0.0....        :GIMBAL LOOP INPUT SIGNAL (RAD/S)
      TAMP=1.30....      :CLUSTER LOOP SINE INPUT AMPLITUDE (RAD/S)
      TFRE0=1.500....    :CLUSTER LOOP SINE INPUT FREQUENCY (RAD/S)
      CAMP=0.1....       :INITIALIZER FOR AMPL. RATIO CALCULATION
      CDDIFF=.0500....   :INITIALIZER FOR AMPL. RATIO CALCULATION
      CDDIFF=.0500....   :INITIALIZER FOR AMPL. RATIO CALCULATION
      CPAMP=0.....       :INITIALIZER FOR AMPL. RATIO CALCULATION
      CAMPR0=.50....     :INITIALIZER FOR AMPL. RATIO CALCULATION
      DL=0.14....        :CLUSTER ANGLE DEADBAND LIMIT (RAD)
      EFB=3.0....        :CLUSTER LIMITER FEEDBACK GAIN (RAD/S/RAD)
      DTP=0.050....      :CLUSTER FEEDBACK LOOP COMPENSATOR POLE (RAD/S)
      DTZ=0.0050....     :CLUSTER FEEDBACK LOOP COMPENSATOR ZERO (RAD/S)
      GLIM=50.....       :GYRO OUTPUT LIMIT
      SW = 1.....        :LIMIT INITIALIZER
      PDCDT=0.           :LIMIT INITIALIZER
*
*     PEDESTAL DYNAMICS CONSTANTS
```

```
*
PARAM ETAPO=0.....      :INITIAL PEDESTAL ANGLE (RAD)
      ETATPO=0.....     :INITIAL PEDESTAL RATE (RAD)
      PGAN=0.03....     :FEEDFORWARD GAIN (V/V)
      SAG=10.50....     :SERVO AMPLIFIER GAIN (V/V)
      CFG=0.18....      :SERVO CURRENT FEEDBACK GAIN (V/A)
      TF=0.00570....    :SERVO TIME CONSTANT (RAD/S)
      PI=3.1415....
      RAP=1.05....      :SERVO RESISTANCE (OHMS)
      PB=1.77....       :NON-LINEAR FRICTION (NM)
      PKT=9.60....      :SERVO CONSTANT (NM/A)
      PKE=9.60....      :SERVO BACK EMF (V/RAD/S)
      PJ=61.54....      :PEDESTAL INERTIA (KGM*M)
      PTZ1=0.120....    :COMPENSATOR ZERO TIME CONSTANT (RAD/S)
      PTP1=0.010....    :COMPENSATOR POLE TIME CONSTANT (RAD/S)
      VLIM=6.00....     :SERVO INPUT VOLTAGE LIMIT (V)
      ILIM=24.0....     :SERVO CURRENT LIMIT (A)
      ELIM=.175         :CLUSTER ANGLE LIMIT (RAD)
*
*     LOOK-UP TABLE - CLUSTER AZ. ANGLE (RAD):PEDESTAL
*     INPUT COMMAND (V)
*
AFGEN FPC=-10.,-6.,-9.5,-5.,-9.,-4.,-6.,...
          -3.,-4.,-2.,-2.,0.,0.,0.,2.,0.,4.,2.....
          6.,3.,8.,4.,9.5,5.,10.,6.
*
CONTEL DELT,.0002,FINTI,5.0
INTEG ADAMS
*
INITIAL
      NVLIM=-VLIM
      NILIM=-ILIM
      NELIM=-ELIM
      NDL=-DL
      Y0 = 0.
      Y00 = 0.
      Y000=0.
*
DYNAMIC
*
      WCD = INTGR(WCD0,WCDT)    :CLUSTER ROTATION RATE (RAD/S)
      WOJ = INTGR(WOJ0,WOJT)    :GIMBAL ROTATION RATE (RAD/S)
      THE = INTGR(THE0,WCD)     :CLUSTER INERTIAL ANGLE (RAD)
      PHI = INTGR(PHI0,WOJ)     :GIMBAL INERTIAL ANGLE (RAD)
      ETA = INTGR(ETA0,ETAT)    :CLUSTER ANGLE REL. TO GIMBAL (RAD)
      EPS = INTGR(EPS0,EPST)    :GIMBAL ANGLE REL. TO PEDESTAL (RAD)
      CGY = INTGR(CGY0,CGYT)    :CLUSTER GYRO POSITION ERROR
      GGY = INTGR(GGY0,GGYT)    :GIMBAL GYRO POSITION ERROR
*
      ETATT=INTGR(ETATP0,ETADTP):PEDESTAL RATE REL. TO BASE (RAD)
      ETAP=INTGR(ETAP0,ETATP):PEDESTAL ANGLE REL. TO BASE (RAD)
*
      ETAL = LIMIT(NELIM,ELIM,ETA)
*  GENERATE STEP INPUT COMMAND
      TDDS = STEP(.001)
```

```
      TDD=TAMP*TDDS
*
PROCED ZT,PHIT,THET=DUM1(TIME,TAMP,TFREQ)
*   THIS SUBROUTINE ZEROS THE BASE MOTION PARAMETERS
      CALL BASEMC(TIME,ZT,PHIT,THET)
*   THIS SUBROUTINE GENERATES A SIMPLE SINE CURVE
*     CALL FRGEN(TIME,TAMP,TFREQ,TDD)
ENDPRO
*
*     CLUSTER/GIMBAL DYNAMICS
*
*        C = CLUSTER
*        G = GIMBAL
*        GY = GYRO
*        GP = GYRO PICKOFF
*        DM = DEMODULATOR
*        SN = SHAPING NETWORK
*        PT = PLATFORM TORQUER
*        P  = PEDESTAL
*
*   TRANSFORM TO PEDESTAL ROTATION RATES
      WBX= THET*SIN(ETAP) + PHIT*COS(ETAP)
      WBY= THET*COS(ETAP) - PHIT*SIN(ETAP)
      WBZ= ETATP
*   CLUSTER/GIMBAL NON-LINEAR  MODEL
PROCED WCD1,WOJ  ETAT,EPST,CPTQ,GPTQ=DUM2(WBX,WBY,WBZ,....
           ZCROSS,ZKROSS,TIME,WCD,WOJ,CSN02,ETA,EPS,ETTP,EPSTP,....
           ETAL,CPAT,GBAT,CGAIN,GGAIN,TK,TR,BEMF,PDCDT,S4)
*
      CALL GYRBAL(WBX,WBY,WBZ,WCD,WOJ,CPTQ,GPTQ,ETAT,ETA,....
         EPST,EPS,ETTP,EPSTP,ZCROSS,ZKROSS,CHAT,GBAT,WCDT,WOJT)
*   FIND RELATIVE ROTATION RATES
      EPST = WOJ-WBY
      ETAT = WCD-(WBX*SIN(EPS)+WBZ*COS(EPS))
*   CONTROL LOOP COMPONENTS WITH LIMITS
      CPTQ1 = CGAIN*CSN02
      ACPTQ1=ABS(CPTQ1)
      IF (ACPTQ1 .LT. 15.7) GO TO 80
      CPTQ1=SIGN(15.7,CPTQ1)
   80 CPTQ = (CPTQ1-BEMF*ETAT)*TK/TR
      GPTQ1 = GGAIN*GSN02
      AGPTQ1=ABS(GPTQ1)
      IF (AGPTQ1 .LT. 15.7) GO TO 81
      GPTQ1=SIGN(15.7,GPTQ1)
   81 GPTQ = (GPTQ1-BEMF*EPST)*TK/TR
      SOCDT=SIGN(1.,WCDT)
      SPDCDT=SIGN(1.,PDCDT)
      AETA = ABS(ETA)
      IF (SW .LT. 0.) GO TO 110
      IF (AETA .LT. .175) GO TO 100
      IF(SOCDT.NE.SPDCDT) GO TO 90
      ETAT = 0.
      WCDT = 0.
      WCD = WBX*SIN(EPS) + WBZ*COS(EPS)
      GO TO 120
```

```
      GO SW = -1.
         PAETA = AETA
  100 PCCDT = WCDT
         GO TO 120
  110 IF (AETA .LT. .175) GO TO 115
         IF (AETA.GT.PAETA) GO TO 115
         PCCDT = WCDT
         GO TO 120
  115 SW = 1.
         PCCDT = WCDT
  120 CONTINUE
ENDPRO
*  GYRO PICK-OFF OUTPUTS
      CGPO = GKP*CGY
      GGPO = GKP*GGY
*  LOOP TO PREVENT CLUSTER HITTING ENDSTOPS
      ETAI=DEADS(NDL,DL,ETA)
      AETAI=ABS(ETAI)
      ETAN=SIGN((AETAI+DL),ETAI)
      ETAIN=FCNSW(ETAI,ETAN,0.,ETAN)
      ETADT,TEMI=LEDLG(0.,DTP,DTZ,ETAIN)
      ETAF=FCNSW(ETAIN,ETADT,0.,ETADT)
      ETAFB=ETAF*ZFB
*  GYRO INTERNAL FEEDBACK LOOP (PROTECTION)
PROCED CGYT,GGYT,CGPOL,GGPOL=DUM4(GKFB,GKP,WCD,WOJ,TDD,TJD,...
          CGPO,GGPO,CGY,GGY,ETAFB,ETA)
      ACGPO=ABS(CGPO)
      AGGPO=ABS(GGPO)
      AETA=ABS(ETA)
      IF (ACGPO .LT. 7.5) GO TO 30
      CGYT = 0.-WCD-GKFB*CGY
      CGPOL = SIGN(7.5,CGPO)
      GO TO 31
   30 IF(AETA .GE. .175) ETAFB=0.0
      CGYT = TDD-WCD-ETAFB
      CGPOL = CGPO
   31 IF (AGGPO .LT. 7.5) GO TO 40
      GGYT = 0.-WOJ-GKFB*GGY
      GGPOL = SIGN(7.5,GGPO)
      GO TO 41
   40 GGYT = TJD-WOJ
      GGPOL = GGPO
   41 CONTINUE
ENDPRO
*  COMPENSATOR NETWORK REALIZED USING MODE-CONTROLLED INTEGRATORS
PROCED CSNO2,CS1,CS2,CP1,CP2,CQ1,CQ2=DUM6(Y00,Y0,TZ1,TZ2,TP1,TP2,...
          GLIM,CGPOL)
      CE=TZ1*CGPOL + Y0
      CS1=CGPOL - CE/TP1
      CSNO1=CE/TP1
      ACSN1=ABS(CSNO1)
      IF (ACSN1 .LT. GLIM) GO TO 45
      CP1=0.
      CP2=0.
      CSNO1=SIGN(GLIM,CSNO1)
```

```
      GO TO 46
   45 CP1=1.
      CP2=1.
   46 CONTINUE
      CFF=CSNO1*TZ2+Y00
      CS2=CSNO1-CFF/TP2
      CSNO2=CFF/TP2
      ACSN2=ABS(CSNO2)
      IF (ACSN2 .LT. GLIM) GO TO 47
      CO1=0.
      CO2=0.
      CSNO2=SIGN(GLIM,CSNO2)
      GO TO 48
   47 CO1=1.
      CO2=1.
   48 CONTINUE
ENDPRO
      Y1 = MODIN(0.,CP1,CP2,CS1)
      Y11 = MODIN(0.,CO1,CO2,CS2)
      GSNO1,Y2=LEDLG(0.,TZ1,TP1,GGPOL)
      GSNO2,Y4=LEDLG(0.,TZ2,TP2,GSNO1)
PROCED DY=DUM7(Y1,Y11,Y0,Y00)
      Y0 = Y1
      Y00 = Y11
ENDPRO
*
*     PEDES  - DYNAMICS
*
*  FIND GAIN
      ETAD=ETAL*180./(PI*COS(EPS))
      PGAIN=EFSEN(EPC,ETAD)
      PFF = PGAIN*PGAN
      APFF = ABS(PFF)
      PEPR=PGAIN + APFF*CGPOL
      PSNO1=LIMIT(NVLIM,VLIM,PEPR)
*  COMPENSATOR NETWORK REALIZED USING MODE CONTROLLED INTEGRATORS
PROCED VIN,VN,VP1,VP2=DUM8(PTZ1,PTP1,PSNO1,Y000,VLIM)
      VE=PTZ1*PSNO1 + Y000
      VN=PSNO1-VE/PTP1
      VIN=VE/PTP1
      AVIN=ABS(VIN)
      IF (AVIN .LT. VLIM) GO TO 51
      VP1=0.
      VP2=0.
      VIN=SIGN(VLIM,VIN)
      GO TO 52
   51 VP1=1.
      VP2=1.
   52 CONTINUE
ENDPRO
      Y01=MODIN(0.,VP1,VP2,VN)
PROCED DY1=DUM9(Y000,Y01)
      Y000=Y01
ENDPRO
*  PEDESTAL SERVO INPUT
```

```
      VINV=(VIN-LMI*CFG)*SAG
      VERR=VI*GAIN*ETATP
      IIN=VERR/RAR
      MI=REALP(0.,TF,IIN)
      LMI=LIMIT(NILIM,ILIM,MI)
*  FRICTION CALCULATION
PROCED FTOR=DUMS(ETATP,3P)
      IF (ETATP .EQ. 0.) GO TO 50
      FTOR = PR*SIGN(1.,ETATP)
      GO TO 60
   50 FTOR = 0.
   60 CONTINUE
ENDPRO
      PDT=LMI*PKT-FTOR
      ETADTP=PDT/PJ
*
*     FIND AMPLITUDE RATIO
*
PROCED CAMPRO,SCAMP=DUM3(CDDIFF,CAMP,...
        WCD,WDJ,CPAMP,CPDIFF,SA,CAMPRO)
      CAMPRO = CAMPRO
      CALL AMPRO(CDDIFF,CAMP,WCD,CPAMP,CPDIFF,CAMPRO)
      CAMPRO = CAMPRO
ENDPRO
*
TERMINAL
*
*
PRNTF .0100,CSNOP,WCD,ETAP,LMI,TDD,CGPOL
PRVDU .0100,TDD,ETA,ETAFB,CGPOL,WCD,ETATP
END
STOP
```

```
C*****************************************
C     THIS SUBROUTINE GENERATES A         *
C     DETERMINISTIC REPRESENTATION OF THE *
C     BASE MOTION RATES ACCORDING TO THE  *
C     RELEVANT POWER SPECTRA.             *
C                                         *
C     SUBROUTINES USED:                   *
C       NONE                              *
C*****************************************
C
      SUBROUTINE BASEM(TIME,ZT,PHIT,THET)
C
C  FREQUENCIES
      W1=.287
      W2=.316
      W3=.368
      W4=.459
      W5=.616
      W6=.795
      W7=.974
C  Z MOTION AMPLITUDES
      VZ1=-.036
      VZ2=-.111
      VZ3=-.306
      VZ4=-.523
      VZ5=-.515
      VZ6=-.299
      VZ7=-.147
C  PITCH RATE AMPLITUDES
      THET1=-.00032
      THET2=-.00104
      THET3=-.00331
      THET4=-.00753
      THET5=-.0112
      THET6=-.0122
      THET7=-.0094
C  ROLL RATE AMPLITUDES
      PHIT1=-.00023
      PHIT2=-.00090
      PHIT3=-.00364
      PHIT4=-.01157
      PHIT5=-.0323
      PHIT6=-.0658
      PHIT7=-.0145
C
      ZT = VZ1*SIN(W1*TIME)+VZ2*SIN(W2*TIME)+VZ3*SIN(W3*TIME)
     /     +VZ4*SIN(W4*TIME)+VZ5*SIN(W5*TIME)+VZ6*SIN(W6*TIME)
     /     +VZ7*SIN(W7*TIME)
C
      THET = THET1*SIN(W1*TIME)+THET2*SIN(W2*TIME)+THET3*SIN(W3*TIME)
     /       +THET4*SIN(W4*TIME)+THET5*SIN(W5*TIME)+THET6*SIN(W6*TIME)
     /       +THET7*SIN(W7*TIME)
C
      PHIT = PHIT1*SIN(W1*TIME)+PHIT2*SIN(W2*TIME)+PHIT3*SIN(W3*TIME)
     /       +PHIT4*SIN(W4*TIME)+PHIT5*SIN(W5*TIME)+PHIT6*SIN(W6*TIME)
     /       +PHIT7*SIN(W7*TIME)
C
      RETURN
      END
```

```
C*****************************************
C     THIS SUBROUTINE GENERATES DETERMI-  *
C     NISTIC BASE MOTION ACCELERATIONS    *
C     ACCORDING TO THE RESPECTIVE POWER   *
C     SPECTRA.                            *
C                                         *
C     SUBROUTINES USED:                   *
C       NONE                              *
C*****************************************
C
      SUBROUTINE SHPXDT(T,PHASE,THASE,ZASE,ZDDT,THEDDT,PHIDDT)
C
C     BASE MOTION ACCELERATION DATA
C
      ZDDT=-.0010*COS(.256*(T+ZASE))-.0105*COS(.287*(T+ZASE))
     /     -.0392*COS(.316*(T+ZASE))-.1124*COS(.368*(T+ZASE))
     /     -.1074*COS(.459*(T+ZASE))-.3134*COS(.616*(T+ZASE))
     /     -.3139*COS(.795*(T+ZASE))-.2979*COS(.974*(T+ZASE))
     /     -.0451*COS(2.112*(T+ZASE))
C
      THEDDT=-.0006*COS(.256*(T+THASE))-.0061*COS(.287*(T+THASE))
     /       -.0217*COS(.316*(T+THASE))-.0714*COS(.368*(T+THASE))
     /       -.1589*COS(.459*(T+THASE))-.2290*COS(.616*(T+THASE))
     /       -.2445*COS(.795*(T+THASE))-.2440*COS(.974*(T+THASE))
     /       -.2342*COS(2.112*(T+THASE))
      THEDDT=THEDDT*1.E-02
C
      PHIDDT=-.0003*COS(.256*(T+PHASE))-0.0001*COS(.287*(T+PHASE))
     /       -.0004*COS(.316*(T+PHASE))-0.0016*COS(.369*(T+PHASE))
     /       -.0054*COS(.459*(T+PHASE))-0.013A*COS(.616*(T+PHASE)
     /       -.0245*COS(.795*(T+PHASE))-0.0410*COS(.974*(T+PHASE )
     /       -.0098*COS(2.112*(T+PHASE))
C
      RETURN
      END
```

```
C*****************************************
C     THIS S     ITINE CONTAINS A NON-LINEAR *
C     MODEL    THE GIMBAL AND CLUSTER DYNA- *
C     MICS INCLUDING CROSS-COUPLING AND     *
C     STICTION.                             *
C                                           *
C     SUBROUTINES USED:                     *
C       VCROSS                              *
C       IBM STANDARD FORTRAN SUBROUTINE     *
C       PACKAGE                             *
C*****************************************
C
      SUBROUTINE GYMBAL(*WBX,*WBY,WBZ,*TCD,*TOJ,TDD,TJD,ETAT,ETA,
     /EPST,EPS,ETATP,EPSTP,ZCROSS,*KROSS,STIC,STIK,*CDT,*TOJT)
C
      DIMENSION WB(3),TCD(3,3),TDB(3,3),TDC(3,3),TMI(3),
     /WD(3),WI(3),TJC(3,3),TJD(3,3),ETT(3),EPT(3),COUP(3),
     /COUT(3)
C  DEFINE ARRAYS
      WB(1) = *WBX
      WB(2) = *WBY
      WB(3) = WBZ
C
      DO 15 I=1,3
      DO 10 J=1,3
      TJC(I,J) = 0.
      TJD(I,J) = 0.
      TCD(I,J) = 0.
      TDB(I,J) = 0.
   10 CONTINUE
   15 CONTINUE
C  BASE TO GIMBAL TRANSFORMATION (EPS=GIMBAL ELEV. ANGLE)
      TDB(1,1) = COS(EPS)
      TDB(1,3) =-SIN(EPS)
      TDB(2,2) = 1.
      TDB(3,1) = SIN(EPS)
      TDB(3,3) = COS(EPS)
C  GIMBAL TO CLUSTER TRANSFORMATION (ETA=CLUSTER AZIM. ANGLE)
      TCD(1,2) = SIN(ETA)
      TCD(1,1) = COS(ETA)
      TCD(2,1) =-SIN(ETA)
      TCD(2,2) = COS(ETA)
      TCD(3,3) = 1.
C  CLUSTER INERTIA TENSOR (KGM*M)
      TJC(1,1) = 0.0189
      TJC(2,2) = 0.0218
      TJC(3,3) = 0.0069
      TJC(1,2) = 0.0032
      TJC(2,1) = 0.0032
C  GIMBAL INERTIA TENSOR (KGM*M)
      TJD(1,1) = 0.0034
      TJD(2,2) = 0.0022
      TJD(3,3) = 0.0021
C
      TJJ = TJD(2,2)+TJC(2,2)*COS(ETA)*COS(ETA)
```

```
     /     +TJC(1,1)*SIN(ETA)*SIN(ETA)
C  FRICTION DAMPING TERM (N*M)
      BD = .015*SIGN(1.,ETAT)
      BJ = .015*SIGN(1.,EPST)
C
      EPT(1) = 0.
      EPT(2) = EPST
      EPT(3) = 0.
C
      ETT(1) = 0.
      ETT(2) = 0.
      ETT(3) = ETAT
C
C     CLUSTER EQUATION OF MOTION.
C
      CALL GMPRD(TCB,WB,TM1,3,3,1)
      CALL MADD(TM1,EPT,WD,3,1,0,0)
      CALL GMPRD(TCO,WD,TM1,3,3,1)
      CALL MADD(TM1,ETT,WI,3,1,0,0)
C
      CALL GMPRD(TJC,WI,TM1,3,3,1)
      CALL VCROSS(WI,TM1,COUP)
C  FIND CLUSTER DRIVE TORQUE
      CDRT = TDD-COUP(3)
      ACDRT = ABS(CDRT)
C  STICTION IN CLUSTER MOTION
      IF (ETAT.EQ.0.) GO TO 16
      IF (ETATP.EQ.0.) GO TO 16
      SN=SIGN(1.,ETAT)
      SNP=SIGN(1.,ETATP)
      IF (SN.EQ.SNP) GO TO 30
      ZCROSS=1.
   16 IF (ZCROSS.EQ.0.) GO TO 30
   20 WCDT=0.
      WCD=WPX*SIN(EPS)+WBZ*COS(EPS)
      IF (ACDRT.LT.STIC) GO TO 40
   30 IF (ETAT .EQ. 0.) GO TO 31
C  ACCELERATION UNDER FRICTION
      WCDT = (CDRT-BD)/TJC(3,3)
      ZCROSS=0.
      GO TO 40
C  ACCELERATION AFTER STICTION OVERCOME
   31 WCDT = CDRT/TJC(3,3)
      ZCROSS = 0.
   40 CONTINUE
C
C     GIMBAL EQUATION OF MOTION
C
      CALL GMPRD(TJO,WO,TM1,3,3,1)
      CALL VCROSS(WO,TM1,COUP)
C
      CALL GMTRA(TCO,TOC,3,3)
      CALL GMPRD(TOC,ETT,TM1,3,3,1)
      CALL MADD(WO,TM1,WI,3,1,0,0)
      CALL GMPRD(TJC,WI,TM1,3,3,1)
```

```
      CALL VCROSS(W1,TM1,COUT)
C  FIND GIMBAL DRIVE TORQUE
      GDRT = TJD-COUP(2)-COUT(2)
      AGDRT = ABS(GDRT)
C  STICTION IN GIMBAL MOTION
      IF (EPST.EQ.0.) GO TO 50
      IF (EPSTP.EQ.0.) GO TO 50
      SN=SIGN(1.,EPST)
      SNP=SIGN(1.,EPSTP)
      IF (SN.EQ.SNP) GO TO 70
      ZKPOSS=1.
   50 IF (ZKPOSS.EQ.0.) GO TO 70
   60 WDJT=0.
      WDJ=W8Y
      IF (AGDRT.LT.STIK) GO TO 80
   70 IF (EPST .EQ. 0.) GO TO 71
C  ACCELERATION UNDER FRICTION
      WDJT = (GDRT-RJ)/TJJ
      ZKPOSS=0.
      GO TO 80
C  ACCELERATION AFTER STICTION OVERCOME
   71 WDJT = GDRT/TJJ
      ZKPOSS=0.
   80 CONTINUE
C
      ETATP = ETAT
      EPSTP = EPST
      RETURN
      END
```

```
C*****************************************
C     THIS SUBROUTINE MEASURES THE        *
C     AMPLITUDE OF A RESPONSE TO A        *
C     SINE INPUT AND FINDS THE AMPLITUDE  *
C     RATIO.                              *
C*****************************************
C
      SUBROUTINE AMPRO(DDIFF,SAMP,AMP,PAMP,PDIFF,AMPLRO)
C
      AAMP = ABS(AMP)
      APAMP = ABS(PAMP)
      DIFF = AAMP-APAMP
      ADIFF = ABS(DIFF-PDIFF)
      ADDIFF = ABS(DDIFF)
      SNDIFF = SIGN(1.,DIFF)
      SNPDIF = SIGN(1.,PDIFF)
C
      IF(SAMP.EQ.0.) GO TO 10
      IF(ADIFF.GE.(ADDIFF*2.0)) GO TO 10
      IF(SNDIFF.EQ.SNPDIF) GO TO 10
      AMPLRO = (.3*AMPLRO+.7*((AAMP+APAMP)/2.)/SAMP)
      DDIFF = DIFF - PDIFF
C
   10 PAMP = AMP
      PDIFF = DIFF
      RETURN
      END
```

```
C*****************************************
C     THIS SUBROUTINE TRANSFORMS THE      *
C     LINEAR VELOCITY FROM INERTIAL       *
C     AXES TO CLUSTER AXES.               *
C                                         *
C     SUBROUTINES USED:                   *
C       IBM STANDARD FORTRAN SUBROUTINE   *
C       PACKAGE.                          *
C*****************************************
C
      SUBROUTINE LMTRAN(XDT,YDT,ZDT,PSI,THE,PHI,ETAP,
     /          EPS,ETA,RDT,SDT,QDT)
C
      DIMENSION TBS(3,3),TSB(3,3),TCO(3,3),TSI(3,3),EM(3),
     /TEMP(3,3),TEM(3,3),TE(3,3),CLM(3),SLM(3)
C
C     DO THE TRANSFORMATIONS
C
      DO 20 I=1,3
      DO 20 J=1,3
      TBS(I,J) = 0.
      TSB(I,J) = 0.
      TCO(I,J) = 0.
   10 CONTINUE
   20 CONTINUE
C
      TBS(1,1) = COS(ETAP)
      TBS(1,2) = SIN(ETAP)
      TBS(2,1) = -SIN(ETAP)
      TBS(2,2) = COS(ETAP)
      TBS(3,3) = 1.
C
      TSB(1,1) = COS(EPS)
      TSB(1,3) = -SIN(EPS)
      TSB(2,2) = 1.
      TSB(3,1) = SIN(EPS)
      TSB(3,3) = COS(EPS)
C
      TCO(1,1) = COS(ETA)
      TCO(1,2) = SIN(ETA)
      TCO(2,1) = -SIN(ETA)
      TCO(2,2) = COS(ETA)
      TCO(3,3) = 1.
C
      TSI(1,1) = COS(THE)*COS(PSI)
      TSI(1,2) = COS(THE)*SIN(PSI)
      TSI(1,3) = -SIN(THE)
      TSI(2,1) = COS(PSI)*SIN(THE)*SIN(PHI)-SIN(PSI)*COS(PHI)
      TSI(2,2) = SIN(PSI)*SIN(THE)*SIN(PHI)+COS(PSI)*COS(PHI)
      TSI(2,3) = COS(THE)*SIN(PHI)
      TSI(3,1) = COS(PSI)*SIN(THE)*COS(PHI)+SIN(PSI)*SIN(PHI)
      TSI(3,2) = SIN(PSI)*SIN(THE)*COS(PHI)-COS(PSI)*SIN(PHI)
      TSI(3,3) = COS(THE)*COS(PHI)
C
C     DEFINE VECTORS
```

```
C
      ZM(1) = XDT
      ZM(2) = YDT
      ZM(3) = ZDT
C
C     DO TRANSFORMATIONS
C
      CALL GMPRD(TBS,TSI,TEMP,3,3,3)
      CALL GMPRD(TOB,TEMP,TEM,3,3,3)
      CALL GMPRD(TCO,TEM,TF,3,3,3)
      CALL GMPRD(TF,ZM,CLM,3,3,1)
C
C     FIND LINEAR MOTION COMPONENTS
C
      XDT = CLM(1)
      YDT = CLM(2)
      ZDT = CLM(3)
C
      RETURN
      END
```

```
C*****************************************
C     THIS SUBROUTINE TRANSFORMS THE      *
C     LINEAR VELOCITY FROM CLUSTER        *
C     AXES TO INERTIAL AXES.              *
C                                         *
C     SUBROUTINES USED:                   *
C       IBM STANDARD FORTRAN SUBROUTINE   *
C       PACKAGE.                          *
C*****************************************
C
      SUBROUTINE ILMTRN(XDT,YDT,ZDT,PSI,THE,PHI,ETAP,
     /          EPS,ETA,GDT,FDT,DDT)
C
      DIMENSION TBS(3,3),TOB(3,3),TCO(3,3),TSI(3,3),ZM(3),
     /TEMP(3,3),TOM(3,3),TF(3,3),TFT(3,3),CLM(3),SLM(3)
C
C     DEFINE TRANSFORMATIONS
C
      DO 20 I=1,3
      DO 10 J=1,3
      TBS(I,J) = 0.
      TOB(I,J) = 0.
      TCO(I,J) = 0.
   10 CONTINUE
   20 CONTINUE
C
      TBS(1,1) = COS(ETAP)
      TBS(2,1) = SIN(ETAP)
      TBS(1,2) = -SIN(ETAP)
      TBS(2,2) = COS(ETAP)
      TBS(3,3) = 1.
C
      TOB(1,1) = COS(EPS)
      TOB(3,1) = -SIN(EPS)
      TOB(2,2) = 1.
      TOB(1,3) = SIN(EPS)
      TOB(3,3) = COS(EPS).
C
      TCO(1,1) = COS(ETA)
      TCO(2,1) = SIN(ETA)
      TCO(1,2) = -SIN(ETA)
      TCO(2,2) = COS(ETA)
      TCO(3,3) = 1.
C
      TSI(1,1) = COS(THE)*COS(PSI)
      TSI(1,2) = COS(THE)*SIN(PSI)
      TSI(1,3) = -SIN(THE)
      TSI(2,1) = COS(PSI)*SIN(THE)*SIN(PHI)-SIN(PSI)*COS(PHI)
      TSI(2,2) = SIN(PSI)*SIN(THE)*SIN(PHI)+COS(PSI)*COS(PHI)
      TSI(2,3) = COS(THE)*SIN(PHI)
      TSI(3,1) = COS(PSI)*SIN(THE)*COS(PHI)+SIN(PSI)*SIN(PHI)
      TSI(3,2) = SIN(PSI)*SIN(THE)*COS(PHI)-COS(PSI)*SIN(PHI)
      TSI(3,3) = COS(THE)*COS(PHI)
C
C     DEFINE VECTORS
```

```
C
      ZM(1) = XDT
      ZM(2) = YDT
      ZM(3) = ZDT
C
C     DO TRANSFORMATIONS
C
      CALL GMPRD(TBS,TSI,TEMP,3,3,3)
      CALL GMPRD(TCB,TEMP,TEM,3,3,3)
      CALL GMPRD(TCO,TEM,TE,3,3,3)
      CALL GMTRN(TE,TET,3,3)
      CALL GMPRD(TET,ZM,CLM,3,3,1)
C
C     FIND LINEAR MOTION COMPONENTS
C
      XDT = CLM(1)
      YDT = CLM(2)
      ZDT = CLM(3)
C
      RETURN
      END
```

```
C*****************************************************
C                                                     *
C     THIS SUBROUTINE FINDS THE CROSS                 *
C     PRODUCT OF TWO VECTORS:                         *
C                                                     *
C                  X X Y = Z                          *
C                                                     *
C     INPUT FORMAT:  X(3)                             *
C                    Y(3)                             *
C     OUTPUT FORMAT: Z(3)                             *
C                                                     *
C*****************************************************
C
      SUBROUTINE VCROSS(X,Y,Z)
C
      DIMENSION X(3),Y(3),Z(3)
C
      Z(1) = X(2)*Y(3) - X(3)*Y(2)
      Z(2) = X(3)*Y(1) - X(1)*Y(3)
      Z(3) = X(1)*Y(2) - X(2)*Y(1)
C
      RETURN
      END
```

```
*  ********************************************
*     THIS PROGRAM USES MEASURED DATA OF THE   *
*     HUMAN OPERATOR IN A TRACKING TASK TO     *
*     IDENTIFY THE PARAMETERS OF A HUMAN       *
*     OPERATOR MODEL OF THE FORM               *
*                                              *
*          G(S)=EXP(-TS)(N(S,B)/D(S,A))        *
*                                              *
*     ACCORDING TO GARAY AND MEHRAV (REFE-     *
*     RENCE 7).                                *
*                                              *
*     SUBROUTINES USED:                        *
*       CALFFT OR SPECT                        *
*       FFT                                    *
*       ESTIM                                  *
*                                              *
*  ********************************************
*
*     PARAMETERS
*
PARAM TC =10.00.....   :SAMPLING WINDOW CUT-OFF (S)
      TO = 0.20.....   :INPUT TIME SHIFT (S)
      T1 = 3.5.....    :TARGET VELOCITY SWITCHING TIME (S)
      T11 =11.5.....   :TARGET VELOCITY SWITCHING TIME (S)
      PI=3.1416.....
      C = 0.997.....   :EXPONENTIAL WEIGHTING COEFFICIENT
      PK1 = 0.30.....  :TRACKING AID GAIN
      PK2 = 0.05.....  :TRACKING AID GAIN
      PK3 = 0.05.....  :TRACKING AID GAIN
      AP1=0.043.....   :'STATE' FILTER TIME CONSTANT
      DLAY=1.0.....    :CORRELATION:IDENTIFICATION OPTION
      R0=6000.0.....   :TARGET INITIAL RANGE (M)
      V0 =350.0.....   :INITIAL TARGET SPEED (M/S)
      THER0=0.00.....  :INITIAL SIGHTLINE ANGLE ERROR (RAD)
      THT0=0.35.....   :INITIAL TARGET ELEVATION (RAD)
      THJD0=0.0.....   :INITIAL TRACKING AID OUTPUT (RAD/S)
      DNOS=64.....     :SAMPLE NUMBER FOR FFT ROUTINE
      SAINT=0.025.....  :IDENTIFIER SAMPLING INTERVAL
      SANT=0.05.....   :TARGET MOTION SAMPLING INTERVAL
      SKIP=1.....      :MEASURED DATA READ INITIALIZER
      ZE1=-1.0.....    :ESTIMATOR INITIALIZER
      ZE2=-1.0.....    :ESTIMATOR INITIALIZER
      ZE3=-1.0.....    :ESTIMATOR INITIALIZER
      ZE4=-1.0.....    :ESTIMATOR INITIALIZER
      P = 1.....       :ESTIMATOR SUBROUTINE INITIALIZER
      SUMN=0.0.....    :SUM INITIALIZER
      SUMU=0.0.....    :SUM INITIALIZER
      SUMN2=0.0.....   :SUM INITIALIZER
      SUMU2=0.0.....   :SUM INITIALIZER
*
CONTSL DELT.0.0080,FINTI.20.0
INTEG ADAMS
*
INITIAL
/UNCT DELAY (500.0)
```

```
      T02 = 2.*T0
      T03 = 3.*T0
      VZ0=0.              :INITIAL VERTICAL TARGET SPEED (M/S)
      FPA=0.              :INITIAL TARGET FLIGHT PATH ANGLE (RAD)
      SAMPL = 0.          :SAMPLING INITIALIZER
      SKIP0=0.0           :SAMPLING INITIALIZER
      DSAMPL=0.           :SAMPLING INITIALIZER
      DSN0=1.             :SAMPLING INITIALIZER
      DSAMNT=TC/DNOS      :SAMPLING INTERVAL FOR FFT
      THEST = 0.0         :INITIAL ESTIMATED SIGHTLINE ANGLE (RAD)
      PTIM = 0.0          :PREVIOUS SAMPLE TIME (S)
      DTIM = 0.0          :SAMPLE TIME INTERVAL (S)
*
DYNAMIC
*
      THER = INTGR(THER0,THERDT) :SIGHTLINE ERROR ANGLE (RAD)
      THER1 = INTGR(THER0,THER2) :ERROR ANGLE FOR RMS CALCULATION
      R = INTGR(R0,RDT)          :RANGE (M)
      THT = INTGR(THT0,THTDT)    :TARGET ELEVATION ANGLE (RAD)
*
*     INPUT JOYSTICK DATA
*
PROCED THTNOS,THJD=DAT(TIME,SKIP,SKIP0)
      IF (SKIP0 .LT. 0.) GO TO 15
      READ (7,10) THTNOS,THJD
  10  FORMAT(2F10.7)
      SKIP0 = -1.-SKIP
  15  SKIP0=SKIP0+SKIP
ENDPRO
      THJDT=THJD
*
*     SIMULATE TARGET MOTION
*
PROCED VX,VZ=TAR(TIME,T1,T11,FPA,V0,VZ0)
      IF (TIME .LT.  11) GO TO  20
      IF (FPA .LT. -.52) GO TO  30
      VZ = VZ0-40.*(TIME-T11)
      GO TO 30
  20  IF (TIME .LT. T1) GO TO 25
      IF (FPA .GE. .52) GO TO 30
      VX = V0+11.*(TIME-T1)
      VZ = 40.*(TIME-T1)
      VZ0 = VZ
      GO TO 30
  25  VZ = VZ0
      VX = V0
  30  CONTINUE
      VXN = -VX
      FPA = ATAN2(VZ,VXN)
ENDPRO
      THTD1=-(VZ/R)*COS(THT)-(VX/R)*SIN(THT)
      RDT=VX*COS(THT)-VZ*SIN(THT)
*
*     FIND TRACKING AID OUTPUT
*
```

```
PROCED THERDT,THRMS=DU2(THTDT,THJDT,THTNOS,THEST,THER  DTIM,PTIM,...
          SAMT,PK1,PK2,PK3,TIME)
      IF (TIME .EQ. 0.) GO TO 80
      IF ((DTIM-SAMT) .LT. 0.) GO TO 81
*  FIND RMS ERROR
      THRMS=SQRT(THERI/TIME)
      THEPST=THEST
      THTPST=THTST
      THTST=THJDT*PK2+THTPST
      THEST=THTPST*PK3+THJDT*PK1+THEPST
      PTIM=TIME
      GO TO 81
   80 THEST=THTDT
      THEPST=THEST
   81 DTIM=TIME-PTIM
*  FIND SIGHTLINE ERROR
      THERDT=THTDT-(THEPST+THJDT-THTNOS)
ENDPRO
      THER2=THER*THER
*
*     FIND STANDARD DEVIATIONS,NOISE-TO-SIGNAL RATIO
*
PROCED NTSR,SZ,SN=DU1(SUMN,SUMN2,SUMU,SUMU2,THTNOS,THTDT,TIME)
      IF (TIME .GT. 0.) GO TO 40
      COUNT = 1.
      GO TO 50
   40 CONTINUE
*  MEAN
      SUMN = SUMN+THTNOS
      SUMU = SUMU+THTDT
      TMN = SUMN/COUNT
      UMN = SUMU/COUNT
*  DEVIATION
      SUMN2 = SUMN2 + THTNOS*THTNOS
      SUMU2 = SUMU2 + THTDT*THTDT
      SN1 = SUMN2/COUNT - TMN*TMN
      ASN1 = ABS(SN1)
      SN = SQRT(ASN1)
      SZ1 = SUMU2/COUNT - UMN*UMN
      ASZ1 = ABS(SZ1)
      SZ = SQRT(ASZ1)
*  RATIO
      IF (SZ .EQ. 0.) GO TO 45
      NTSR = SQRT((SN*SN)/(SZ*SZ))
   45 COUNT = COUNT+1.
   50 CONTINUE
ENDPRO
*
*     FIND DELAY (CROSS-CORRELATION USING AN FFT)
*
PROCED DET,DGT=DU2(TIME,DSAMPL,DSAMNT,THER,THJD,DNOS,DSNO,TC,...
          PI,DLAY,FINTI,THEST)
      IF (TC .LE. FINTI) GO TO 55
      WRITE (*,100)
  100 FORMAT('HANNING END-TIME GREATER THAN FINTIM')
```

```
      GO TO 60
   55 IF (DLAY .LT. 1.) GO TO 60
      IF (TIME .LT. DSAMPL) GO TO 60
*  HANNING FUNCTION
      IF (TIME .GT. TC) GO TO 65
      DFT = 10.*THEST*(0.5-0.5*COS(2.*PI*TIME/TC))
      DGT = 10.*THJD*(0.5-0.5*COS(2.*PI*TIME/TC))
*  FIND SPECTRUM OF A SIGNAL
   65 CALL SPECT(DFT,DNOS,DSND)
*  FIND CROSS-CORRELATION OF TWO SIGNALS
*  65 CALL CRLFFT(DFT,DGT,DNOS,DSND,DSAMNT)
      DSAMPL = DSAMPL+DSAMNT
   60 CONTINUE
ENDPRO
*
*     IDENTIFY HUMAN OPERATOR TRANSFER FUNCTION PARAMETERS
*
*  'STATE' FILTER OUTPUTS
      Z = THER
      U = THTDT+THTNOS
      TZ1 = REALP(0.,RP1,Z)
      TZ2 = REALP(0.,RP1,TZ1)
      TZ3 = REALP(0.,RP1,TZ2)
      TZ4 = REALP(0.,RP1,TZ3)
      Z0 = REALP(0.,RP1,TZ4)
      Z1 = (TZ4-Z0)*(1./RP1)
      Z2 = (TZ3-Z0)*(1./RP1)*(1./RP1)-2.*(1./RP1)*Z1
      Z3 = (TZ2-Z0)*(1./RP1)*(1./RP1)*(1./RP1)-3.*(1./RP1)*(1./RP1)*...
           Z1-3.*(1./RP1)*Z2
      Z4 = (TZ1-Z0)*(1./RP1)*(1./RP1)*(1./RP1)*(1./RP1)-...
         4.*(1./RP1)*(1./RP1)*(1./RP1)*Z1-6.*(1./RP1)*(1./RP1)*Z2-...
         4.*(1./RP1)*Z3
      Z5 = (1.-Z0)*(1./RP1)*(1./RP1)*(1./RP1)*(1./RP1)*(1./RP1)-...
         5.*(1./RP1)*(1./RP1)*(1./RP1)*(1./RP1)*Z1-...
         10.*(1./RP1)*(1./RP1)*(1./RP1)*Z2-10.*(1./RP1)*(1./RP1)*Z3-...
         5.*(1./RP1)*Z4
*
      TUU1 = REALP(0.,RP1,U)
      TUU2 = REALP(0.,RP1,TUU1)
      TUU3 = REALP(0.,RP1,TUU2)
      U0 = REALP(0.,RP1,TUU3)
      U1 = ( UU3-U0)*(1./RP1)
      U2 = (TUU2-U0)*(1./RP1)*(1./RP1)-2.*(1./RP1)*U1
      U3 = (TUU1-U0)*(1./RP1)*(1./RP1)*(1./RP1)-3.*(1./RP1)*(1./RP1)*...
           U1-3.*(1./RP1)*U2
      U4 = (1.-U0)*(1./RP1)*(1./RP1)*(1./RP1)*(1./RP1)-...
         4.*(1./RP1)*(1./RP1)*(1./RP1)*U1-6.*(1./RP1)*(1./RP1)*U2-...
         4.*(1./RP1)*U3
*
      ET0 = Z2 - U1
      RO1 = Z3 - U2
      RO2 = Z4 - U3
      RO3 = Z
      RO4 = Z2
*
```

```
      UV1 = 0
      UV2 = DELAY(56,T0,U)
      UV3 = DELAY(112,T02,U)
      UV4 = DELAY(168,T03,U)
*  DO ESTIMATION
PROCES DRT1,DRT2,DRT3,DRT4,K,NRT=DU4(R01,R02,R03,...
          R04,ET0,UV1,UV2,UV3,UV4,C,P,SAINT,SAMPL,TIME,...
          ZE1,ZE2,ZE3,ZE4,DLAY)
       IF (DLAY .EQ. 1.) GO TO 70
       IF (TIME .LT. SAMPL) GO TO 70
       CALL ESTIM(R01,R02,R03,R04,ET0,UV1,UV2,UV3,...
         UV4,DRT1,DRT2,DRT3,DRT4,K,NRT,ZE1,ZE2,ZE3,ZE4,C,P)
       SAMPL = SAMPL + SAINT
   70 CONTINUE
ENDPRO
*
TERMINAL
*
PRNTF 0.100,DRT1,DRT2,DRT3,DRT4,K,NRT
PRVDU 0.100,THTDT,THER,THJDT,THRMS,DFT,DGT
END
STOP
```

```
C*****************************************
C     THIS SUBROUTINE FINDS THE AMPLITUDE *
C     SPECTRUM OF A TIME FUNCTION. IT CAN *
C     BE CALLED BY THE HUMAN OPERATOR     *
C     ANALYSIS PROGRAM WITH THE APPRO-    *
C     PRIATE MODIFICATION.                *
C                                         *
C     SUBROUTINES USED:                   *
C       FFT                               *
C*****************************************
C
      SUBROUTINE SPECT(FT,SON,SNO)
      DIMENSION XREAL(64),XIMAG(64),S(64)
C
      N=IFIX(SON)
      NO=IFIX(SNO)
      XREAL(NO)=FT
C
      IF (NO .LT. N) GO TO 500
      IF (NO .GE. N+1) GO TO 500
      NU=6
      N2=N/2
C
      DO 5 I=1,N
      XIMAG(I)=0.
    5 CONTINUE
C
      CALL FFT(XREAL,XIMAG,N,NU)
C
      DO 10 I=1,N
      S(I)=SQRT(XREAL(I)*XREAL(I)+XIMAG(I)*XIMAG(I))
   10 CONTINUE
C
      WRITE (6,300) (XREAL(I),I=1,N2),(XIMAG(I),I=1,N2),(S(I),I=1,N2)
  300 FORMAT (1H1,1X,'XREAL = ',8F8.4,/,9X,8F8.4,/,9X,8F8.4,
     /     /,9X,8F8.4,/,1X,'XIMAG =',8F8.4,/,9X,8F8.4,/,9X,8F8.4,
     /     /,9X,8F8.4,/,1X,'SPECTRUM =',8F8.4,/,11X,8F8.4,/,11X,8F8.4,
     /     /,11X,8F8.4)
  500 NO=NO+1
      SNO = FLOAT(NO)
      RETURN
      END
```

```
C*****************************************
C     THIS SUBROUTINE CONTAINS THE       *
C     RECURSION FORMULA FOR ESTIMATING   *
C     THE HUMAN OPERATOR TRANSFER        *
C     FUNCTION ACCORDING TO GABAY AND    *
C     MERHAV (REFERENCE 7).              *
C                                        *
C     SUBROUTINES USED:                  *
C       IBM STANDARD FORTRAN SUBROUTINE  *
C       PACKAGE                          *
C*****************************************
C
      SUBROUTINE ESTIM(RO1,RO2,RO3,RO4,ETO,U1,U2,U3,
     /     U4,DRT1,DRT2,DRT3,DRT4,CONST,URT,Z1,Z2,
     /   Z3,Z4,C,P)
C
      DIMENSION GAMMA(4),U(4,1),RHO(4,1),V(4,4),Z(4,1),
     /     PHOT(1,4),T1(1,4),T2(4,1),T3(4,1),VNUM(4,4),VTEM1(4,4),
     /     VTEM2(4,4),DEN(3),COF(3),DROOTR(2),DROOTI(2)
C
      M = 2
      CIN = 1./C
      IF (P.LT.0.) GO TO 50
C
C     FIND INITIAL VALUES
C
      DO 20 I=1,4
      DO 10 J=1,4
      V(I,J) = 0.
   10 CONTINUE
   20 CONTINUE
      P = -1.
      DO 30 I=1,4
      V(I,1) = 1.
   30 CONTINUE
      Z(1,1) = Z1
      Z(2,1) = Z2
      Z(3,1) = Z3
      Z(4,1) = Z4
      RETURN
C
C     FIND VECTORS
C
   50 CONTINUE
      RHO(1,1) = RO1
      RHO(2,1) = RO2
      RHO(3,1) = RO3
      RHO(4,1) = RO4
C
      U(1,1) = U1
      U(2,1) = U2
      U(3,1) = U3
      U(4,1) = U4
C
C     FIND V(K+1)
```

```
C
      CALL GMTRA(SHO,SHOT,4,1)
      CALL GMPRD(SHOT,V,T1,1,4,4)
      CALL GMPRD(U,T1,VTEM1,4,1,4)
      CALL GMPRD(V,VTEM1,VNUM,4,4,4)
C
      CALL GMPRD(T1,U,TEM,1,4,1)
      VDEN = 1./(TEM+C)
C
      CALL SMPY(VNUM,VDEN,VTEM1,4,4,0)
      CALL SMPY(V,CIN,VTEM2,4,4,0)
      CALL MSUB(VTEM2,VTEM1,V,4,4,0,0)
C
C     FIND Z(K+1)
C
      CALL SMPY(U,ETO,T2,4,1,0)
      CALL SMPY(Z,C,T3,4,1,0)
      CALL MADD(T2,T3,Z,4,1,0,0)
C
C     FIND GAMMA(K+1)
C
      CALL GMPRD(V,Z,T2,4,4,1)
      CALL SMPY(T2,-1.,GAMMA,4,1,0)
C
C     OUTPUT
C
      DEN(1) = 1.
      DEN(2) = GAMMA(1)
      DEN(3) = GAMMA(2)
C
C     FIND ROOTS OF POLYNOMIAL
C
      IF (GAMMA(3) .NE. 0.) GO TO 60
      DRT1 = 0.
      DRT2 = 0.
      DRT3 = 0.
      DRT4 = 0.
      URT = 0.
      CONST = 0.
      GO TO 100
   60 CONTINUE
C
C
      URT = GAMMA(4)/GAMMA(3)
      CONST = GAMMA(3)
C
      AGAM = ABS(GAMMA(2))
      IF (AGAM .LT. 0.05) GO TO 70
C     CALL POLRT(DEN,COF,M,DROOTR,DROOTI,IER)
      ADIF = ABS(GAMMA(1)*GAMMA(1)-4.*GAMMA(2))
      DRT1 = (-GAMMA(1)+SQRT(ADIF))/(2.*GAMMA(2))
      DRT2 = (-GAMMA(1)-SQRT(ADIF))/(2.*GAMMA(2))
      GO TO 80
C
   70 DRT1 = 0.
```

```
      DRT2 = GAMMA(1)
   80 DRT3 = 0.
      DRT4 = 0.
C
C  80 DRT1 = DRDOTR(1)
C     DRT2 = DRDOTR(2)
C     DRT3 = DRDOTI(1)
C     DRT4 = DRDOTI(2)
C
      IF(IER .LE. 0) GO TO 100
      WRITE (6,*) IER
  100 CONTINUE
C
      RETURN
      END
```

```
C*****************************************
C     THIS SUBROUTINE FINDS THE CROSS-     *
C     CORRELATION FUNCTION OF THE HUMAN    *
C     OPERATOR INPUT AND OUTPUT FUNCTIONS  *
C     SO THAT THE DELAY CAN BE REDUCED.IT  *
C     CAN BE CALLED BY THE HUMAN OPERATOR  *
C     ANALYSIS PROGRAM WITH THE APPROPRIATE*
C     MODIFICATION.                        *
C                                          *
C     SUBROUTINES USED:                    *
C       FFT                                *
C*****************************************
C
      SUBROUTINE CALFFT(FT,GT,SON,SNO,T)
      DIMENSION XREAL1(64),XIMAG1(64),XREAL2(64),XIMAG2(64),
     /   XREAL(64),XIMAG(64)
      N = IFIX(SON)
      NO = IFIX(SNO)
      XREAL1(NO)=FT
      XREAL2(NO)=GT
      IF (NO.LT.N) GO TO 500
      IF (NO.GE.(N+1)) GO TO 500
      N2 = N/2
      NU = 6.
      DELF = 1./(SON*T)
      DO 5 I=1,N
      XIMAG1(I) = 0.
      XIMAG2(I) = 0.
    5 CONTINUE
C
C     FIND AMPLITUDE SPECTRA
C
      CALL FFT(XREAL1,XIMAG1,N,NU)
      CALL FFT(XREAL2,XIMAG2,N,NU)
C
      DO 10 I=1,N
      XREAL1(I) = XREAL1(I)*T
      XIMAG1(I) = -XIMAG1(I)*T
      XREAL2(I) = XREAL2(I)*T
      XIMAG2(I) = -XIMAG2(I)*T
   10 CONTINUE
C
C     FIND PRODUCT OF TRANSFORMS
C
      DO 20 I=1,N
      XREAL(I) = XREAL1(I)*XREAL2(I)-XIMAG1(I)*XIMAG2(I)
      XIMAG(I) = XREAL1(I)*XIMAG2(I)+XREAL2(I)*XIMAG1(I)
   20 CONTINUE
C
C     FIND INVERSE TRANSFORM TO GIVE CROSS-CORRELATION
C
      N21 = N2-1
      DO 110 I=1,N21
      XREAL(N/2+1+I) = XREAL(N/2+1-I)
      XIMAG(N/2+1+I) =-XIMAG(N/2+1-I)
```

```
  110 CONTINUE
      DO 120 I=1,N
      XIMAG(I) = -XIMAG(I)
  120 CONTINUE

      CALL FFT(XREAL,XIMAG,N,NU)
C
      DO 210 I=1,N
      XREAL(I) = XREAL(I)*DELF
      XIMAG(I) = XIMAG(I)*DELF
  210 CONTINUE
C
      WRITE (6,300) (XREAL(I),I=1,N2),(XIMAG(I),I=1,N2)
  300 FORMAT (1H1,1X,'XREAL = ',F6.3,2X,F6.3,2X,F6.3,2X,
     /   F6.3,2X,F6.3,2X,F6.3,/,9X,F6.3,2X,F6.3,2X,F6.3,2X,
     /   F6.3,2X,F6.3,2X,F6.3,/,9X,F6.3,2X,F6.3,2X,F6.3,2X,
     /   F6.3,2X,F6.3,2X,F6.3,/,9X,F6.3,2X,F6.3,2X,F6.3,2X,
     /   F6.3,2X,F6.3,2X,F6.3,/,9X,F6.3,2X,F6.3,2X,F6.3,2X,
     /   F6.3,2X,F6.3,2X,F6.3,/,9X,F6.3,2X,F6.3,/,1X,'XIMAG = ',
     /   F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,/,9X,
     /   F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,/,9X,
     /   F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,/,9X,
     /   F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,/,9X,
     /   F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,/,9X,
     /   F6.3,2X,F6.3)
      WRITE (6,400) (XREAL(I),I=N2,N),(XIMAG(I),I=N2,N)
  400 FORMAT (1H ,1X,'XREAL = ',F6.3,2X,F6.3,2X,F6.3,2X,
     /   F6.3,2X,F6.3,2X,F6.3,/,9X,F6.3,2X,F6.3,2X,F6.3,2X,
     /   F6.3,2X,F6.3,2X,F6.3,/,9X,F6.3,2X,F6.3,2X,F6.3,2X,
     /   F6.3,2X,F6.3,2X,F6.3,/,9X,F6.3,2X,F6.3,2X,F6.3,2X,
     /   F6.3,2X,F6.3,2X,F6.3,/,9X,F6.3,2X,F6.3,/,1X,'XIMAG = ',
     /   F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,/,9X,
     /   F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,/,9X,
     /   F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,/,9X,
     /   F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,2X,F6.3,/,9X,
     /   F6.3,2X,F6.3)
  500 NO = NO+1
      SNO = FLOAT(NO)
      RETURN
      END
```

```
C*************************************
C     THIS SUBROUTINE FINDS THE      *
C     FAST FOURIER TRANSFORM OF      *
C     A FUNCTION AND IS APPLIED      *
C     AS GIVEN IN BRIGHAM (REFE-     *
C     RENCE 3).                      *
C*************************************
C
      SUBROUTINE FFT(XREAL,XIMAG,N,NU)
      DIMENSION XREAL(N),XIMAG(N)
      N2=N/2
      NU1=NU-1
      K=0
      DO 100 L=1,NU
  102 DO 101 I=1,N2
      P=IBITR(K/2**NU1,NU)
      ARG=6.283185*P/FLOAT(N)
      C=COS(ARG)
      S=SIN(ARG)
      K1=K+1
      K1N2=K1+N2
      TREAL=XREAL(K1N2)*C+XIMAG(K1N2)*S
      TIMAG=XIMAG(K1N2)*C-XREAL(K1N2)*S
      XREAL(K1N2)=XREAL(K1)-TREAL
      XIMAG(K1N2)=XIMAG(K1)-TIMAG
      XREAL(K1)=XREAL(K1)+TREAL
      XIMAG(K1)=XIMAG(K1)+TIMAG
  101 K=K+1
      K=K+N2
      IF(K.LT.N) GO TO 102
      K=0
      NU1=NU1-1
  100 N2=N2/2
      DO 103 K=1,N
      I=IBITR(K-1,NU)+1
      IF(I.LE.K) GO TO 103
      TREAL=XREAL(K)
      TIMAG=XIMAG(K)
      XREAL(K)=XREAL(I)
      XIMAG(K)=XIMAG(I)
      XREAL(I)=TREAL
      XIMAG(I)=TIMAG
  103 CONTINUE
      RETURN
      END
C
      FUNCTION IBITR(J,NU)
      J1=J
      IBITR=0
      DO 200 I=1,NU
      J2=J1/2
      IBITR=IBITR*2+(J1-2*J2)
  200 J1=J2
      RETURN
      END
```

```
*
*   THIS PROGRAM CONTAINS THE TRACKER SIMU-
*   LATOR ROUTINES, GENERATING A DIVE-ATTACK
*   A MISSILE ATTACK, AND A LO   TOSS BOMBING
*   ATTACK. INPUT IS FROM THE JOYSTICK AND
*   THE RMS TRACKING ERROR IS FOUND.
*
*********************************************
*
*  PARAMETERS
*
PARAM CN1=2.Ø....       :INPUT CHANEL 1
      CN2=3.Ø....       :INPUT CHANEL 2
      ON1=1.Ø....       :OUTPUT CHANEL 1
      ON2=Ø.Ø....       :OUTPUT CHANEL 2
      VX0=-350.Ø....    :INITIAL TARGET X-VELOCITY
      VXM0=-500.Ø....   :INITIAL MISSILE X-VELOCITY (M/S)
      VY0=Ø.Ø....       :INITIAL TARGET Y-VELOCITY (M/S)
      VZ0=Ø.Ø....       :INITIAL TARGET Z-VELOCITY (M/S)
      JOYCL=Ø.1ØØ....   :JOYSTICK SCALE FACTOR (RAD/S/V)
      OSCL=2Ø.ØØ....    :OUTPUT SCALE FACTOR (V/RAD)
      THER0=Ø.Ø....     :INITIAL ELEVATION SIGHTLINE ERROR (RAD)
      PSER0=Ø.Ø....     :INITIAL AZIMUTH SIGHTLINE ERROR (RAD)
      BEGN=428.....     :RANDOM NUMBER GENERATOR INITIALIZER
      TNOISE=Ø.ØØ1....  :LOOP NOISE SCALE FACTOR
      SNOS=1.Ø....      :NOISE GENERATOR SAMPLING TIME (S)
      SAMT=Ø.Ø5Ø....    :LOOP SAMPLING INTERVAL (S)
      PSLIM=Ø.Ø5....    :INITIAL AZIMUTH ERROR LIMIT (RAD)
      TF0=Ø.2....       :FILTER GAIN
      PF0=Ø.3           :FILTER GAIN
      TF01=Ø.Ø5....     :FILTER GAIN
      TF02=Ø.Ø1....     :FILTER GAIN
      PF01=Ø.Ø5....     :FILTER GAIN
      PF02=Ø.Ø1....     :FILTER GAIN
      SW=1.Ø....        :JOYSTICK TRANSFER FUNCTION SWITCH
      T1=Ø.Ø....        :TARGET TYPE INITIAL PARAMETER
      T2=Ø.Ø....        :TARGET TYPE INITIAL PARAMETER
      T3=Ø.Ø....        :TARGET TYPE INITIAL PARAMETER
      FSW=1.Ø....       :FILTER IN-OUT SWITCH
*
CONTRL DELT,.ØØ5,FINT1,3Ø.1Ø
INTEG ADAMS
*
INITIAL
*
FUNCT RND(1.,Ø.)
FUNCT ADC(2.,Ø.)
FUNCT DAC(2.,Ø.)
      FPA=Ø.Ø       :INITIAL TARGET FLIGHT PATH ANGLE
      THEST=Ø.Ø     :INITIAL ELEVATION RATE ESTIMATE (RAD/S)
      PSEST=Ø.Ø     :INITIAL AZIMUTH RATE ESTIMATE (RAD/S)
      PTIM=Ø.Ø      :SAMPLE LOOP INITIALIZER
      DTIM=Ø.Ø      :SAMPLE LOOP INITIALIZER
      DTNOS=Ø.Ø     :SAMPLE LOOP INITIALIZER
      PTNOS=Ø.Ø     :SAMPLE LOOP INITIALIZER
INS
*  CHOOSE INITIAL TARGET ATTACK PROFILE
      NPSLIM=-PSLIM
      IF (T1.GT. Ø.) GO TO 6
      IF (T2.GT. Ø.) GO TO 6
      IF (T3.GT. Ø.) GO TO 6
      RNO=RND(BEGN)
      TSW=RNO*1ØØ.
      RNO1=RNO-Ø.5
```

```
      PSIC=LIMIT(NPSLIM,PSLIM,RNO1)
      IF (TSW .GT. 60.) GO TO 5
      IF (TSW .GT. 30.) GO TO 4
      F0=0800.0
      THT0=0.35
      GO TO 6
    4 F0=8500.0
      THT0=0.0017
      GO TO 6
    5 F0=1500.0
      THT0=0.0
    6 CONTINUE
*
DYNAMIC
*
      PSER=INTGR(PSER0,PSERDT)   :SIGHTLINE AZIMUTH ANGLE ERROR (RAD)
      THER=INTGR(THER0,THERDT)   :SIGHTLINE ELEVATION ANGLE EROR (RAD)
      R=INTGR(R0,RDT)            :RANGE (M)
      THT=INTGR(THT0,THTDT)      :TARGET ELEVATION ANGLE (RAD)
      PST=INTGR(PSI0,PSTDT)      :TARGET AZIMUTH ANGLE (RAD)
      THG=INTG (THT0,THGDT)      :GIMBAL ELEVATION ANGLE (RAD)
      PSC=INTGR(PSI0,PSCDT)      :CLUSTER AZIMUTH ANGLE (RAD)
      TH1=INTGR(THER0,TH2R)      :ERROR INTEGRALS
      PS1=INTGR(PSER0,PS2R)
      T1=INTGR(THER0,T2R)
      P1=INTGR(PSER0,P2R)
      TT1=INTGR(THER0,TT2R)
      PP1=INTGR(PSER0,PT2R)
*
*  INPUT JOYSTICK DATA
*
PROCED THJDT,PSJDT=DAT(TIME,TJSCL,CN1,CN2,SW)
      CALL ADC(CN1,THJ)
      CALL ADC(CN2,PSJ)
*  LINEAR JOYSTICK TRANSFER FUNCTION
      THJDT=THJ*TJSCL
      PSJDT=-PSJ*TJSCL
      IF (SW .EQ. 0.) GO TO 10
*  NON-LINEAR JOYSTICK TRANSFER FUNCTION
      TH2J=THJDT*THJDT
      PS2J=PSJDT*PSJDT
      THJDT=SIGN(TH2J,THJDT)
      PSJDT=SIGN(PS2J,PSJDT)
 10   CONTINUE
ENDPRO
*
*  FIND TARGET MOTION
*
PROCED RDT,THTDT,PSTDT=TAR(TIME,FPA,TSW,T1,T2,T3,R,THT,PST,VZ0)
      IF (TSW .GT. 60.) GO TO 30
      IF (TSW .GT. 30.) GO TO 20
      IF (T1 .GT .0.) GO TO 40
*  TARGET DIVE ATTACK PROFILE
      CALL DIVAT(TIME,FPA,THT,R,THTDT,RDT,VZ0)
      PSTDT=0.0
      IF (TIME.LT.30.) GO TO 40
      T1=1.
      CALL FINIS
      GO TO 40
*  TARGET MEDIUM TOSS BOMB ATTACK PROFILE
 20   IF(T2 .GT. 0.) GO TO 40
      CALL LNGTS(TIME,THT,PST,R,THTDT,PSTDT,RDT)
      IF (TIME.LT.16.) GO TO 40
      T2=1.
```

```
      CALL FINIS
      GO TO 40
*  MISSILE ATTACK PROFILE
  30  IF (T3 .GT. 0.) GO TO 40
      CALL MISATT(TIME,THT,R,THTDT,RDT)
      PSTDT=0.0
      IF (TIME.LT.17.) GO TO 40
      T3=1.
      CALL FINIS
  40  CONTINUE
ENDPRO
*
*  ADD NOISE TO LOOP
*
      TNOS=RND(BEGN)-0.5
      PNOS=RND(BEGN)-0.5
PROCED THTNOS,PSTNOS=NOS(TIME,TNOS,PNOS,TKNOIS,DTNOS,...
        PTNOS,SNOS)
      IF ((DTNOS-SNOS) .LT. 0.) GO TO 60
      THTNOS=TKNOIS*TNOS
      PSTNOS=TKNOIS*PNOS
      PTNOS=TIME
  60  DTNOS=TIME-PTNOS
ENDPRO
*
*  APPLY FILTER/AID
*
      TTSQR=(TEST-THG)*(TEST-THG)
      PTSQR=(PEST-PSC)*(PEST-PSC)
      THSQR=THER*THER
      PSQR=PSER*PSER
      TDF=(THEST-THGDT)*(THEST-THGDT)
      PDF=(PSEST-PSCDT)*(PSEST-PSCDT)
PROCED THERDT,PSERDT,TERMS,TDTRMS,TEST,PEST,TERMS,THGDT,...
            PSCDT=FIL(THTDT,SAT,TIME,THT,PST,PSTDT,FSEST,PSJDT,...
              FSW,TFK,PFK,PSTDT,PDOR,DTIM,PTIM,TFK1,TFK2,PFK1,PFK2,...
          THEST,TTI,PTI,PST,THT,THJDT,THTNOS,TTI,PPI,P)
      IF (P .EQ. 0.) GO TO 80
      IF (TIME .EQ. 0.) GO TO 79
      THERMS=(SQRT(THTI))/TIME+(SQRT( SI))/TIME
      TDTRMS=(SQRT(TTI))/TIME+(SQRT(PI))/TIME
      TERMS=(SQRT(TTI))/TIME+(SQRT(PPI))/TIME
  79  IF (FSW .EQ. 0.) GO TO 82
      IF (( TIM-SAM1).LT.0.0) GO TO 81
      THPEST=THEST
      PSPEST=PSEST
      TPEST=TEST
      PPEST=PEST
      THEST=THJDT*TFK+THPEST
      TEST=TPEST+TFK1*THPEST+TFK2*THJDT
      PSEST=PSJDT*PFK+PSPEST
      PEST=PPEST+PFK1*PSPEST+PFK2*PSJDT
      TTIM=TIME
      GO TO 81
  82  THEST=THTDT
      PSEST=PSTDT
      THPEST=THEST
      PSPEST=PSEST
      TEST=THT
      PEST=PST
      P=1.0
  81  DTIM=TIME-PTIM
      THGDT=THPEST+THJDT
      PSCDT=PSPEST+PSJDT
```

```
      THTRDT=THTDT-THGDT+THTNOS
      PSERDT=PSTDT-PSGDT+PSTNOS
      GO TO 83
   82 THTRDT=THTDT-THJDT+THTNOS
      PSERDT=PSTDT-PSJDT+PSTNOS
   83 CONTINUE
ENDPRO
*
*  DISPLAY ERROR
*
      OUTP=THER*OSCL
      OUPP=PSER*OSCL
      TOUTP=LIMIT(-10.,10.,OUTP)
      TOUPP=LIMIT(-10.,10.,OUPP)
PROCED DUM=DIS(TOUTP,TOUPP,ON1,ON2)
      CALL DAC(ON1,TOUTP)
      CALL DAC(ON2,TOUPP)
ENDPRO
*
TERMINAL
*
PRINTF 1.00,THTDT,THER,THJDT,THRMS
PRVLU 1.00,THJDT,PSJDT,TOUTP,T_RMS,THRMS,TDTRMS
*
END
*  CHOOSE ANOTHER ATTACK PROFILE
 101  RNO=RND(REGN)
      IF (T1.GT.0. .AND  T2.GT.0. .AND. T3.GT.0.) GO TO 106
      TSW=RNO*100.
      RNO1=RNO-0.5
      PSI0=LIMIT(RPSLIM,PSLIM,RNO1)
      IF (TSW .GT. 60.) GO TO 105
      IF (TSW .GT. 30.) GO TO 104
      IF (T1 .GT. 0.) GO TO 101
      P0=6000.0
      THT0=0.35
      GO TO 106
 104  IF (T2 .GT. 0.) GO TO 101
      P0=8500.0
      THT0=0.0017
      GO TO 106
 105  IF (T3 .GT. 0.) GO TO 101
      P0=8500.0
      THT0=0.0
 106  CONTINUE
*
END
END
STOP
```

```
C     ********************************
C     THIS PROGRAM SIMULATES A MEDIUM *
C     TOSS FOR TRACKING PURPOSES.     *
C     ********************************
      SUBROUTINE LNGTS(TIME,THT,PST,R,THTDT,PSTDT,RDT)
C
C  PARAMETERS
C
C  ACCELERATION CONSTANTS (M/S**S)
      GZ1=-40.0
      GX1=16.0
      GY2=30.0
      GX2=50.0
      GZ2=60.0
      GZ3=-30.0
C  SWITCHING TIMES (S)
      T0=1.0
      T1=6.0
      T11=12.0
      T12=13.0
      T13=14.0
C  INITIAL VELOCITY COMPONENTS (M/S)
      VZ0=0.0
      VY0=0.0
      VX0=-350.0
C  INITIAL DISPLACEMENTS (M)
      X0=8500.
      Y0=0.0
      Z0=-15.0
C  INITIAL ANGLES (RAD)
      THL0=1.765E-03
      PSI0=0.0
C  INITIAL RANGE (M)
      R0=6500.0
C
C  GENERATE TARGET MOTION IN RECTANGULAR COORDINATES
C
      IF (TIME .GT. T13) GO TO 35
      IF (TIME .GT. T12) GO TO 30
      IF (TIME .GT. T11) GO TO 20
      IF (TIME .GT. T1) GO TO 10
      IF (TIME .GT. T0) GO TO 5
C
      VX=VX0
      VY=VY0
      VZ=VZ0
C
      GO TO 40
    5 VZ=GZ1*(TIME-T0)
      VX=VX0+GX1*(TIME-T0)
      VY=0.0
      GO TO 40
   10 VX=VX0D+GX2*(TIME-T1)
      VY=GY2*(TIME-T1)
      VZ=VZ0D+GZ2*(TIME-T1)
      GO TO 50
   20 VX=VX0D+GX2*(TIME-T1)
      VY=GY2*(TIME-T1)
      VZ=VZ0DD+GZ3*(TIME-T11)
      GO TO 60
   30 VX=VX0D+GX2*(TIME-T1)
      VZ=VZ0DD+GZ3*(TIME-T11)
      GO TO 60
   35 VZ=VZ0DD+GZ3*(TIME-T11)
      GO TO 60
```

```
40    VXDD=VX
      VZDD=VZ
      GO TO 60
50    VZDD=VZ
60    CONTINUE
C
C  TARGET MOTION IN POLAR COORDINATES
C
      RDT=VX*COS(THT)*COS(PST)+VY*COS(THT)*SIN PST)
     *     +VZ*SIN(THT)
      THTDT=-(VZ/R)*COS(THT)-(VX/R)*COS(PST)*SIN(THT)-
     *       (VY/R)*SIN(PST)*SIN(THT)
      PSTDT=(VX/R)*(SIN(PST)/COS(THT))+(VY/R)*.COS(PST)/COS(THT)
C
      RETURN
      END
```

```
C  ***************************************
C  *                                     *
C  *   THIS SUBROUTINE SIMULATES         *
C  *   A DIVE ATTACK FOR TRACKING        *
C  *   PURPOSES.                         *
C  *                                     *
C  ***************************************
      SUBROUTINE DIVAT(TIME,FPA,THT,R,THTDT,RDT,VZ0)
C
C     PARAMETERS
C
      T1=3.5
      T11=11.50
      V0=-350.3
      PI=3.1416
C  GENERATE TARGET MOTION IN RECTANGULAR COORDINATES
      IF (TIME.LT.T11) GO TO 20
      IF (FPA.LT.-.52) GO TO 30
      VZ=VZ0-40.*(TIME-T11)
      GO TO 30
   20 IF (TIME .LT. T1) GO TO 25
      IF (FPA .GE. .52) GO TO 30
      VX=V0+11.*(TIME-T1)
      VZ=40.*(TIME-T1)
      VZ0=VZ
      GO TO 30
   25 VZ=VZ0
      VX=V0
   30 CONTINUE
      VXN=-VX
      FPA=ATAN2(VZ,VXN)
C  TARGET MOTION IN POLAR COORDINATES
      THTDT=-(VZ/R)*COS(THT)-(VX/R)*SIN(THT)
      RDT=VX*COS(THT)-VZ*SIN(THT)
C
      RETURN
      END
```

```
C  ********************************
C  THIS SUBROUTINE SIMULATES      *
C  A MISSILE ATTACK FOR           *
C  TRAC ING PURPOSES.             *
C  ********************************
      SUBROUTINE MSAT(TIME,THT,R,THTDT,RDT)
C
C     PARAMETERS
C
      VX=-500.0
      VZ=0.0
C  TARGET MOTION IN POLAR COORDINATES
      THTDT=-(VZ/R)*COS(THT)-(VX/R)*SIN(THT)
      RDT=VX*COS(THT)-VZ*SIN(THT)
C
      RETURN
      END
```

```
*****************************************
*     THIS PROGRAM GENERATES TARGET DIVE  *
*     ATTACK PROFILES USED TO SIMULATE    *
*     AND ANALYZE THE TRACKING FILTER     *
*     APPLICATION.                        *
*                                         *
*     SUBROUTINES USED:                   *
*       FILTER                            *
*       KGAIN                             *
*****************************************
*
*     PARAMETERS
*
PARAM T1 = 7.5,...        :TARGET VELOCITY SWITCHING TIME (S)
      T11 =10.5,....      :TARGET VELOCITY SWITCHING TIME (S)
      V0 = -350.0,....    :INITIAL TARGET VELOCITY (M/S)
      R0=6000.0,....      :INITIAL TARGET RANGE (M)
      THERO=0.00,....     :INITIAL SIGHTLINE ANGLE ERROR (RAD)
      THTO=0.35,....      :INITIAL TARGET ELEVATION ANGLE (RAD)
      G=50.....           :MAXIMUM TARGET ACCELERATION (M/S)
      PI=3.14159,....
      B=100.....          :NOISE GENERATOR INITIALIZER
      SRANGE=5.....       :RANGE MEASUREMENT NOISE STD DEVIATION (M)
      SELEV=.0000050,...  :ELEVATION RATE MEASUREMENT NOISE STD DEV (RAD/S)
      SBEAR=.000001,....  :AZIMUTH RATE MEASUREMENT NOISE STD DEV (RAD/S)
      PSI=0.....          :AZIMUTH ANGLE (RAD)
      XHPI1=4000.....     :FILTER INITIAL VALUE
      XHPI2=-328.800,...  :FILTER INITIAL VALUE
      XHPI3=-.350....     :FILTER INITIAL VALUE
      XHPI4=0.020....     :FILTER INITIAL VALUE
      XHPI5=0.....        :FILTER INITIAL VALUE
      XHPI6=0.....        :FILTER INITIAL VALUE
      W=0.....            :FILTER SUBROUTINE INITIALIZER
      V=0.0....           :FILTER SUBROUTINE INITIALIZER
      C1=0.4,....         :FILTER GAIN MULTIPLIER
      C2=30.00,....       :FILTER GAIN MULTIPLIER
      C3=.050,....        :FILTER GAIN MULTIPLIER
      C4=1.00,....        :FILTER GAIN MULTIPLIER
      C5=1.00,....        :FILTER GAIN MULTIPLIER
      C6=0.05             :FILTER GAIN MULTIPLIER
*
CONTRL DELT=.05,FINTI=30.
*
INITIAL
INTEG ADAMS
*
      ERTHE = 0.          :INITIAL FILTER ANGLE OUTPUT ERROR (RAD)
      VZ0=0.0             :INITIAL TARGET Z-VELOCITY (M/S)
      VX0=V0              :INITIAL TARGET X-VELOCITY (M/S)
      X0=R0*COS(THT0)     :INITIAL TARGET X-COORDINATE (M)
      Z0=R0*SIN(THT0)     :INITIAL TARGET Z-COORDINATE (M)
      GPA=0.0             :INITIAL FLIGHT PATH ANGLE (RAD)
*   INITIALIZED FILTER VALUES
      XHP1 = XHPI1
      XHP2 = XHPI2
```

```
      XHP3 = XHPI3
      XHP4 = XHPI4
      XHP5 = XHPI5
      XHP6 = XHPI6
*
DYNAMIC
*
      X=INTGR(X0,VX)              :TARGET X-COORDINATE (M)
      Z=INTGR(Z0,VZ)              :TARGET Z-COORDINATE (M)
      VX=INTGR(VX0,GX)            :TARGET X-VELOCITY (M/S)
      VZ=INTGR(VZ0,GZ)            :TARGET Z-VELOCITY (M/S)
      THEPI = INTGR(THEP0,THE2)   :SIGHTLINE ANGLE ERROR INTEGRAL
      THETI = INTGR(THEP0,THT2)   :SIGHTLINE RATE ERROR INTEGRAL
      DI = INTGR(THEP0,D2)        :RANGE ERROR INTEGRAL
      DTI = INTGR(THEP0,DT2)      :RANGE RATE ERROR INTEGRAL
      ZI=INTGR(THEP0,EZ2)         :RANGE MEASUREMENT ERROR INTEGRAL
*
*
*     FIND TARGET MOTION
*
PROCED GX,GZ=TAR(TIME,T1,T11,FPA,V0,VZ0,G)
      IF (TIME .LT. T11) GO TO  20
      IF (FPA .GE. .52) GO TO  25
      GX = G*SIN(FPA)
      GZ = G*COS(FPA)
      GO TO 30
  20  IF (TIME .LT. T1) GO TO 25
      IF (FPA .LT. -.52) GO TO 25
      GX = -G*SIN(FP )
      GZ = -G*COS(FPA)
      GO TO 30
  25  GZ = 0.0
      GX = 0.0
  30  CONTINUE
      VXN = -VX
      FPA = ATAN2(VZ,VXN)
ENDPRO
*
      EZ2=EZ1*EZ1
PROCED TME,THET,THETT,D,DT,DTT,XHAT1,XHAT2,XHAT3,XHAT4,XHAT5,...
         XHAT6,XISA1,XISA2,XISA3,XISA4,XISA5,XISA6,ERRD,ERRTHE,...
         ERRDT,ERRTHT,EZ1=ONAM(X,Z,VX,VZ,ERTHE,...
        XHP1,XHP2,XHP3,XHP4,XHP5,XHP6,W,V,R,  SRANGE,SELEV,SBEAR)
      D=SQRT(X*X+Z*Z)
      THE=ATAN2(Z,X)
      THET=(X*VZ-Z*VX)/(D*D)
      DT=(X*VX+Z*VZ)/D
      DT1=(VX*VX+VZ*VZ+X*GX+Z*GZ)/D-DT*DT/D
      THETT=(X*GZ-Z*GX)/(D*D)-2.*DT*THET/D
*
*     DETERMINE MEASUREMENT (NORMALLY
*     DISTRIBUTED ABOUT CALCULATED MEAN).
*
      IF (R.EQ.0.) GO TO 100
      I=689
```

```
      R=0.
  100 CONTINUE
      CALL GAUSS(1,SRANGE,0.,Z1)
      CALL GAUSS(1,SELEV,THET,Z2)
      CALL GAUSS(1,SHEAR,PSI,Z3)
      EZ1= 0-Z1
*
*     FEED MEASUREMENT TO FILTER.
*
      CALL FILTER(Z1,Z2,Z3,XHAT1,XHAT2,XHAT3,XHAT4,XHAT5,XHAT6,....
         XHP1,XHP2,XHP3,XHP4,XHP5,XHP6,X1SA1,X1SA2,X1SA3,X1SA4,....
         X1SA5,X1SA6,W,V,C1,C2,C3,C4,C5,C6)
      XHP1=XHAT1
      XHP2=XHAT2
      XHP3=XHAT3
      XHP4=XHAT4
      XHP5=XHAT5
      XHP6=XHAT6
*
*     FIND FILTER ERRORS
*
      ERRD = D-XHP1
      ERRTHE = THE-XHP3
      ERRTHT = THET-XHP4
      ERRDT = DT-XHP2
      ERTHE = ERRTHE
ENDPRO
*
*     FIND RMS ERROR
*
      THE2=ERRTHE*ERRTHE
      THT2=ERRTHT*ERRTHT
      D2=ERRD*ERRD
      DT2=ERRDT*ERRDT
PROCED THERMS,DRMS,THTRMS,DTRMS,ZRMS=RMS(TIME,THERI,THETI,....
         DI,DTI,ZI)
      IF (TIME .EQ. 0.) GO TO 50
      THERMS=SQRT(THERI/TIME)
      THTRMS=SQRT(THTI/TIME)
      DRMS=SQRT(DI/TIME)
      DTRMS=SQRT(DTI/TIME)
      ZRMS=SQRT(ZI/TIME)
   50 CONTINUE
ENDPRO
*
TERMINAL
PRNTF 0.10,ERRD,ERRDT,EZ1,ZRMS,DRMS,DTRMS
PRVDU 0.10,ERRD,ERRDT,ERRTHE,ERRTHT,DRMS,DTRMS
END
STOP
```

```
C*****************************************
C     THIS SUBROUTINE COMPUTES THE FILTER    *
C     ALGORITHM FOR ESTIMATING THE TARGET    *
C     MOTION PARAMETERS.                     *
C                                            *
C     SUBROUTINES USED:                      *
C       KGAIN                                *
C       IBM STANDARD FORTRAN SUBROUTINE      *
C       PACKAGE                              *
C*****************************************
C
      SUBROUTINE FILTER(Z1,Z2,Z3,XHAT1,XHAT2,XHAT3,
     /XHAT4,XHAT5,XHAT6,XHATP1,XHATP2,XHATP3,XHATP4,XHATP5
     /,XHATP6,X1SA1,X1SA2,X1SA3,X1SA4,X1SA5,X1SA6,Y,V,
     /C1,C2,C3,C4,C5,C6)
C
      DIMENSION X(6),Z(3),VX(6,6),PHI(6,6),
     /GAMMA(6,3),Q(3,3),R(3,3),H(3,6),
     /PHIT(6,6),GAMMAT(3,6),TM10(6),TM20(6),TM30(3),
     /TM40(3),TM50(3),TM60(3),TM70(6),TM80(6),X1SA(6),
     /TM90(6),GAIN(6,3),VINOV(6),XHAT(6)
C
C C     DEFINE ALL ARRAY TERMS
C
C     ZERO TERMS
C
      IF (V.GE.2.) GO TO 6
   5  V = V+1.
      XHAT1 = XHATP1
      XHAT2 = XHATP2
      XHAT3 = XHATP3
      XHAT4 = XHATP4
      XHAT5 = XHATP5
      XHAT6 = XHATP6
      X0=XHATP1
      RETURN
   6  CONTINUE
C
      DO 20 I=1,6
      DO 15 J=1,6
      PHI(I,J) = 0.
      VX(I,J) = 0.
  15  CONTINUE
  20  CONTINUE
C
      DO 30 I=1,6
      DO 25 J=1,3
      GAMMA(I,J) = 0.
  25  CONTINUE
  30  CONTINUE
C
      DO 40 I=1,3
      DO 35 J=1,3
      Q(I,J) = 0.
      R(I,J) = 0.
```

```
   35  CONTINUE
   40  CONTINUE
C
       DO 50 I=1,3
       DO 45 J=1,6
       H(I,J) = 0.
   45  CONTINUE
   50  CONTINUE
C
C      STATE VECTOR
C
       X(1) = XHATP1
       X(2) = XHATP2
       X(3) = XHATP3
       X(4) = XHATP4
       X(5) = XHATP5
       X(6) = XHATP6
C
C      MEASUREMENT VECTOR
C
       Z(1) = Z1
       Z(2) = Z2
       Z(3) = Z3
C
C      STATE TRANSITION MATRIX
C
       PHI(1,1) = 1.
       PHI(1,2) = 0.05
       PHI(2,2) = 1.0
       PHI(3,3) = 1.
       PHI(3,4) = .05
       PHI(4,4) = 1.0
       PHI(5,5) = 1.
       PHI(5,6) = .05
       PHI(6,6) = 1.0
C
C      NOISE TRANSFORMATION
C
       GAMMA(1,1) = C1*1.00
       GAMMA(2,1) = C2*0.05
       GAMMA(3,2) = C3
       GAMMA(4,2) = C4*X0/X(1)
       GAMMA(5,2) = C5*X0/(X(1)*COS(X(3)))
       GAMMA(6,3) = C6*X0/(X(1)*COS(X(3)))
C
C      MEASUREMENT TRANSITION
C
       H(1,1) = 1.
       H(2,4) = 1.
       H(3,6) = 1.
C
C      INITIAL STATE COVARIANCE MATRIX
C
       IF (Y.NE.0.) GO TO 55
       VX(1,1) = 0.01
```

```
      VX(2,2) = 0.01
      VX(3,3) = 1.
      VX(4,4) = 1.
      VX(5,5) = 1.
      VX(6,6) = 1.
   55 CONTINUE
C
C       STUDRANCE COVARIANCE MATRIX
C
      Q(1,1) = 5.00
      Q(2,2) = 10.
      Q(3,3) = 1.0
C
C     MEASUREMENT COVARIANCE MATRIX
C
      R(1,1) = 5.00
      R(2,2) = 10.0
      R(3,3) = 1.0
C
C     CALCULATE GAIN
C
      CALL KGAIN(PHI,X,Z,GAMMA,H,VX,Q,R,GAIN,Y)
C
C     CALCULATE INOVATION
C
      CALL GMPRO(PHI,X,X1SA,6,6,1)
      CALL GMPRO(H,X1SA,TM30,3,6,1)
      CALL MSUB(Z,TM30,TM40,3,1,0,0)
C
C     MULTIPLY BY GAIN
C
      CALL GMPRO(GAIN,TM40,VINOV,6,3,1)
C
C     ADD TO STATE EQUATION
C
      CALL MADD(X1SA,VINOV,XHAT,6,1,0,0)
C
C     OUTPUTS
C
      XHAT1 = XHAT(1)
      XHAT2 = XHAT(2)
      XHAT3 = XHAT(3)
      XHAT4 = XHAT(4)
      XHAT5 = XHAT(5)
      XHAT6 = XHAT(6)
C
      X1SA1 = X1SA(1)
      X1SA2 = X1SA(2)
      X1SA3 = X1SA(3)
      X1SA4 = X1SA(4)
      X1SA5 = X1SA(5)
      X1SA6 = X1SA(6)
C     WRITE (6,99)(X(I),I=1,6),(Z(I),I=1,3),((GAMMA(I,J),J=1,3),I=1,6)
C99   FORMAT (/,1X,'STATE VECTOR = ',2(2X,E12.4),2X,E12.4,2X,
C    /E12.4,2X,E12.4,2X,E12.4,/,1X,'MEASUREMENT VECTOR = ',
```

```
C     /F12.4,2X,F12.4,2X,E12.4,/,1X,'GAMMA MATRIX = ',
C     /F12.4,2X,E12.4,2X,E12.4,/,16X,F12.4,2X,E12.4,2X,E12.4,/,16X,
C     /E12.4,2X,F12.4,2X,E12.4,/,16X,F12.4,2X,E12.4,2X,E12.4,/,16X,
C     /F12.4,2X,E12.4,2X,E12.4,/,16X,F12.4,2X,F12.4,2X,E12.4)
C
      WRITE (6,100) ((GAIN(I,J),J=1,3),I=1,6),((VX(I,J),J=1,6),I=1,6)
  100 FORMAT (/,1X,'GAIN = ',E12.4,2X,E12.4,2X,E12.4,/,7X,
     /E12.4,2X,E12.4,2X,E12.4,/,7X,E12.4,2X,E12.4,2X,E12.4,/,7X,
     /E12.4,2X,E12.4,2X,E12.4,/,7X,E12.4,2X,E12.4,2X,E12.4,/,7X,
     /E12.4,2X,E12.4,2X,E12.4,/,/,1X,'VARIANCE = ',E12.4,2X,E12.4,2X,
     /E12.4,2X,E12.4,2X,E12.4,2X,E12.4,/,12X,E12.4,2X,E12.4,2X,
     /E12.4,2X,E12.4,2X,E12.4,2X,E12.4,/,12X,E12.4,2X,E12.4,2X,
     /E12.4,2X,E12.4,2X,E12.4,2X,E12.4,/,12X,E12.4,2X,E12.4,2X,
     /E12.4,2X,E12.4,2X,E12.4,2X,E12.4,/,12X,E12.4,2X,E12.4,2X,
     /E12.4,2X,E12.4,2X,E12.4,2X,E12.4,/,12X,E12.4,2X,E12.4,2X,
     /E12.4,2X,E12.4,2X,E12.4,2X,E12.4)
      RETURN
      END
```

```
C*****************************************************
C     THIS SUBROUTINE FINDS THE GAIN FOR THE          *
C     FILTER ALGORITHM.                               *
C                                                     *
C     SUBROUTINES USED:                               *
C       IBM STANDARD FORTRAN SUBROUTINE PACKAGE       *
C*****************************************************
C
      SUBROUTINE KGAIN(PHI,X,Z,GAMMA,H,VX,U  ,GAIN,Y)
C
      DIMENSION X(6),Z(3),VX(6,6),PHI(6,6),
     /GAMMA(6,3),Q(3,3),R(3,3),H(3,6),TM1(6),TM2(6),
     /TM3(6),TM4(6),TM5(6),HT(6,3),
     /PHIT(6,6),GAMMAT(3,6),TM10(6,6),TM20(6,6),TM30(6,3),
     /TM40(6,6),TM50(6,3),TM60(3,6),TM70(3,3),SIGMA(3,3),
     /APVAR(6,6),GAIN(6,3),EVEC1(3),EVEC2(3),UNM(6,6)
C
C     DEFINE UNIT MATRIX
C
      DO 15 I=1,6
      DO 10 J=1,6
      UNM(I,J) = 0.
      UNM(I,I) = 1.
   10 CONTINUE
   15 CONTINUE
C
C     CALCULATE A PRIORI VARIANCE
C
      CALL GMTRA(PHI,PHIT,6,6)
      CALL GMTRA(GAMMA,GAMMAT,6,3)
      CALL GMTRA(H,HT,3,6)
      CALL GMPRD(PHI,VX,TM10,6,6,6)
      CALL GMPRD(TM10,PHIT,TM20,6,6,6)
      CALL GMPRD(GAMMA,Q,TM30,6,3,3)
      CALL GMPRD(TM30,GAMMAT,TM40,6,3,6)
      CALL MADD(TM40,TM20,APVAR,6,6,0,0)
      CALL GMPRD(H,APVAR,TM60,3,6,6)
      CALL GMPRD(TM60,HT,TM70,3,6,3)
      CALL MADD(TM70,R,SIGMA,3,3,0,0)
C
C     CALCULATE GAIN
C
      CALL MINV(SIGMA,3,D,LVEC1,MVEC2)
      CALL GMPRD(HT,SIGMA,TM50,6,3,3)
      CALL GMPRD(APVAR,TM50,GAIN,6,6,3)
C
C     CALCULATE A POSTERIORI VARIANCE
C
      CALL GMPRD(GAIN,H,TM10,6,3,6)
      CALL MSUB(UNM,TM10,TM20,6,6,0,0)
      CALL GMPRD(TM20,APVAR,VX,6,6,6)
C
C     WRITE (6,100) ((SIGMA(I,J),J=1,3),I=1,3)
C    /             ((GAIN(I,J),J=1,3),I=1,6)
C100  FORMAT (/,1X,'SIGMA INV = ',3E12.4,/,13X,3E12.4,/,13X,3E12.4)

C    /   /,/,1X,'GAIN = ',3E12.4,/,9X,3E12.4,/,9X,3E12.4,/,9X,
C    /   3E12.4,/,9X,3E12.4,/,9X,3E12.4)
      RETURN
      END
```

APPENDIX B

REFERENCES

1 ALERS, I.E. — Mason's Package for Linear Systems Analysis
Kentron Document No KE234 March 1980

2 ALLEN, W.R. and McRUER, D. — The Man/Machine Control Interface - Pursuit Control
Automatica Vol 15 No 6 November 1979

3 BRIGHAM, E.O. — The Fast Fourier Transform
Prentice-Hall 1974

4 CHAN, Y.T. et al — A Kalman Filter Based Tracking Scheme with Input Estimation
IEEE Trans. Vol. AES-15 No 2
March 1979

5 ENGELBRECHT, A.F. — Structure of the Grey Optimization Package and Description of its Subroutines
Kentron Document No KE221 November 79

6 FITTS, J.M. — Aided Tracking as Applied to High Accuracy Pointing Systems
IEEE Trans. Vol AES-9 No 3 May 1973

7 GABAY, E. and MERHAV, S.J. — Identification of a Parametric Model of the Human Operator in Closed Loop Control Tasks
IEEE Trans. Vol SMC-7 No 4 April 1977

8 _____ — Identification of Linear Systems with Time-Delay Operating in a Closed Loop in the Presence of Noise
IEEE Trans. Vol AD-21 No 5
October 1976

9 GARNELL, P. and EAST, D.J. — Guided Weapon Control Systems. Pergamon Press 1977

10 GHOLSON, N.H. and MOOSE, R.L. — Manoeuvering Target Tracking using Adaptive State Estimation. IEEE Trans. Vol AES-13 No 3 May 1977

11 GUPTA, S.C. and HASDORFF, L. — Fundamentals of Automatic Control. John Wiley and Sons 1970

12 HALFMAN, R.L. — Dynamics Vol 1. Addison-Wesley 1962

13 LEE, W.H. and ATHANS, M. — The Discrete Time Compensated Kalman Filter. Int. J Control Vol 29 No 2 1979

14 McFADYEN, I.C. — Optical Director Definition Study. Kentron Document No 10021-952-20 July 1979

15 McRUER, D. — Human Dynamics in Man-Machine Dynamics. Automatica Vol 16 No 3 May 1980

16 MOOSE, R.L. — An Adaptive State Estimation Solution to the Manoeuvering Target Problem. IEEE Trans. Vol AC-20 No 3 June 1975

17 PAPOULIS, A. — The Fourier Integral and its Applications. McGraw-Hill 1962

18 _____ . — Signal Analysis. McGraw-Hill 1979

19 PELED, A. and LIU, B.
Digital Signal Processing
John Wiley and Sons 1976

20 RICKER, G.G. and WILLIAMS, J.R.
Adaptive Tracking Filter for Manoeuvering Targets
IEEE Trans. Vol. AES-14 No 1
January 1978

21 RUE, A.K.
Stabilization of Precision Electro-Optical Pointing and Tracking Systems
IEEE Trans. Vol AES-5 No 5
September 1969

22 SAGE, A.P. and MELSA, J.L.
Estimation Theory with Application to Communication and Control
McGraw-Hill 1971

23 SINGER, R.A.
Estimating Optimal Tracking Filter Performance for Manned Manoeuvering Targets
IEEE Trans. Vol AES-6 No 4
July 1970

24 SPRINGARN, K. and WEIDEMANN, H.L.
Linear Regression Filtering and Prediction for Tracking Manoeuvering Aircraft Targets
IEEE Trans. Vol AES-8 No 6
November 1972

25 TAKAHASHI, Y. et al
Control and Dynamic Systems
Addison-Wesley 1972

26 THOMSON, W.T.
Introduction to Space Dynamics
John Wiley and Sons

27 WIBERG, D.M.
Theory and Problems of State Space and Linear Systems
McGraw-Hill 1971

28 WRIGLEY, W. et al Gyroscopic Theory, Design and Instrumentation
MIT Press 1969

29 The Selection of a Gunner Describing Function Model
GE Technical Report