(2)

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A STUDY OF MODEL BASED MANEUVERING
CONTROLS FOR AUTONOMOUS
UNDERWATER VEHICLES

by

Richard J. Boncal

December 1987

Thesis Advisor: A.J. Healey

Approved for public release; distribution is unlimited

A189 055

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b. OFFICE SYMBOL (If applicable) Code 69	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c. ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000	7b. ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
A STUDY OF MODEL BASED MANEUVERING CONTROLS FOR AUTONOMOUS UNDERWATER VEHICLES

12. PERSONAL AUTHOR(S)
Boncal, Richard J.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Master's Thesis	FROM ____ TO ____	1987, December	148

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Model Reference Controls; Autonomous Underwater Vehicles

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Autonomous Underwater Vehicles (AUV) are being considered by the U.S. Navy for a variety of missions. Requirements for autonomy reinforce the need for a robust maneuvering controller that can ensure accurate tracking of a planned path. Model reference controllers (MRC) have been employed in situations where accurate tracking is desired and where plant parameters change with operating conditions. Because underwater vehicles are highly non-linear, it is conjectured that an MRC will provide improved tracking performance for AUVs. This thesis presents the results of a simulation study in which the dynamics of a submersible are modeled using a modified version of the DTNSRDC 2510 equations of motion. A linearized version of these equations serves as the reference model and provides the basis for the design of feedforward and feedback elements of the controller. Results show that for dive plane

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL Prof. A.J. Healey	22b. TELEPHONE (Include Area Code) (408) 646-2804 22c. OFFICE SYMBOL Code 67He

DD FORM 1473, 84 MAR — 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

● U.S. Government Printing Office: 1986—606-24.

UNCLASSIFIED

#19 - ABSTRACT - (CONTINUED)

maneuvers, accurate tracking of the planned path can be achieved for a moderately wide range of vehicle speeds.

Accesion For	
NTIS CRA&I	☑
DTIC TAB	☐
Unannounced	☐
Justification	
By	
Distribution /	
Availability Codes	
Dist	Avail and / or Special
A-1	

Approved for public release; distribution is unlimited

A Study of Model Based Maneuvering Controls
for Autonomous Underwater Vehicles

by

Richard J. Boncal
Lieutenant, United States Navy
B.S., Maine Maritime Academy, 1981

Submitted in partial fulfillment of the
requirements for the degrees of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

and

MECHANICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
December 1987

Author: Richard J. Boncal
Richard J. Boncal

Approved by: A.J. Healey
A.J. Healey, Thesis Advisor

A.J. Healey
A.J. Healey, Chairman
Department of Mechanical Engineering

G.E. Schacher
Gordon E. Schacher,
Dean of Science and Engineering

ABSTRACT

Autonomous Underwater Vehicles (AUV) are being considered by the U.S. Navy for a variety of missions. Requirements for autonomy reinforce the need for a robust maneuvering controller that can ensure accurate tracking of a planned path. Model reference controllers (MRC) have been employed in situations where accurate tracking is desired and where plant parameters change with operating conditions. Because underwater vehicles are highly non-linear, it is conjectured that an MRC will provide improved tracking performance for AUVs. This thesis presents the results of a simulation study in which the dynamics of a submersible are modeled using a modified version of the DTNSRDC 2510 equations of motion. A linearized version of these equations serves as the reference model and provides the basis for the design of feedforward and feedback elements of the controller. Results show that for dive plane maneuvers, accurate tracking of the planned path can be achieved for a moderately wide range of vehicle speeds.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENTS

I want to express my heartfelt thanks to my thesis advisor, Chairman of the Mechanical Engineering Department, and friend, Dr. Anthony J. Healey for his continued guidance and enthusiastic support throughout the course of this research. I am honored to be his first thesis student here at the Naval Postgraduate School and deeply grateful for the opportunity to be associated with a man who is a genius in his understanding of humanity and nature.

I also want to express my deepest gratitude to my wife Cheryl, for her love, understanding and support throughout my postgraduate education.

I. INTRODUCTION

A. GENERAL

In recent years, the focus on Autonomous Underwater Vehicles (AUV's) or, more generally, Unmanned Underwater Vehicles (UUV's) has increased. A variety of unclassified missions includes Search and Survey, Decoy and Outboard sensors, Ocean Engineering Work Service, Swimmer Support, and Test and Evaluations [Ref. 1]. As the cost of manned submarine vehicles increases, there are significant advantages to the use of cheaper unmanned vehicles. UUV's can be either tethered or untethered. Development in both areas is proceeding, but, while tethered vehicles can use fiber optic links to human operators on a mother ship, a fully autonomous vehicle is required to have a high level of intelligent processing on board. Thus the requirements for AI and Knowledge-Based Controls are much increased. A recent symposium [Ref. 2] has presented a summary of the State of the Art in Unmanned Untethered Submersible Technology.

The organization of the intelligent control of an AUV can be expressed as a cycle of Sensing, Thinking, and Acting (Figure 1.1). At the highest level of the control architecture, the mission planning and symbolic reasoning lead to requirements for path planning and control. The lower level of Acting involves operation and control of all

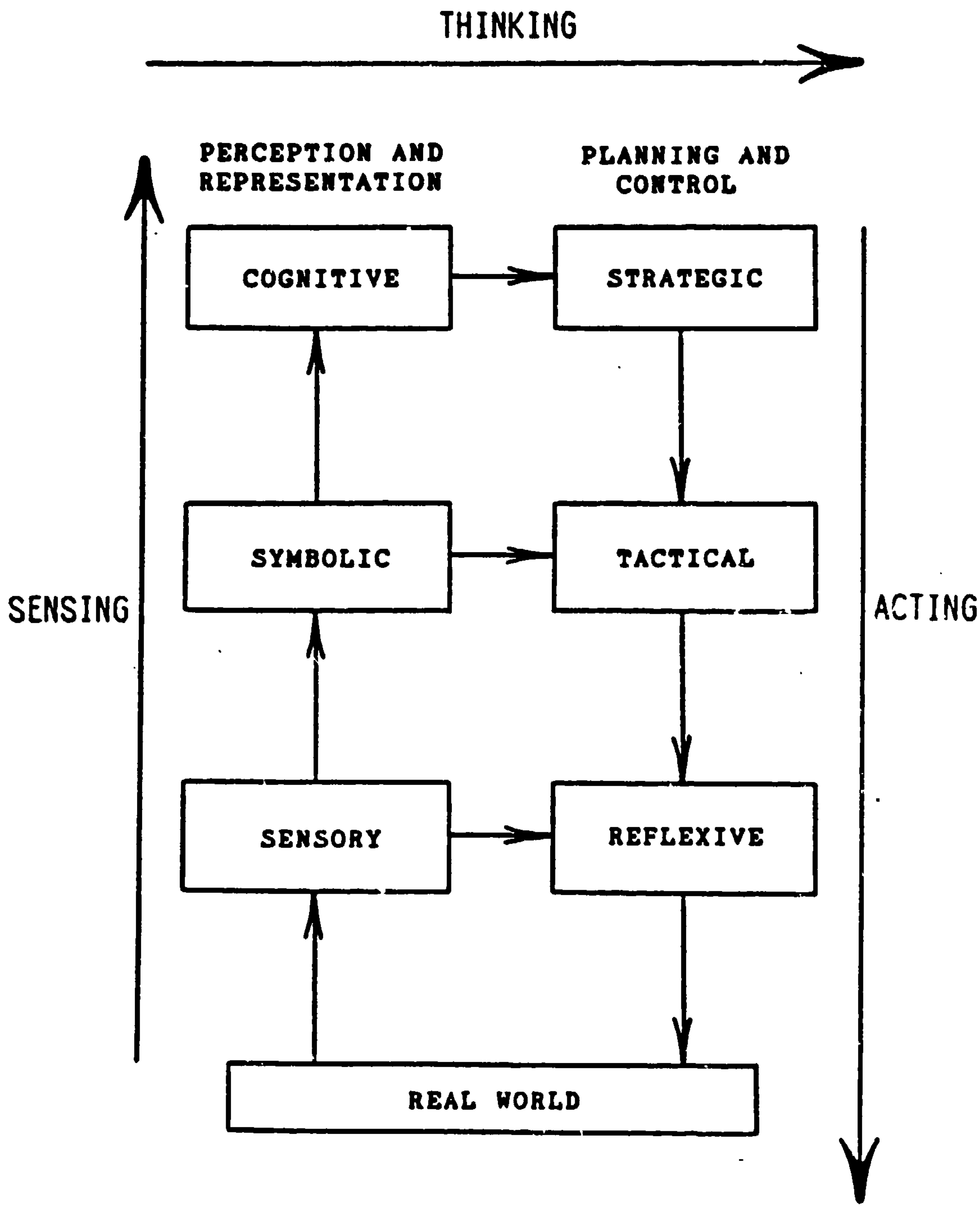

Figure 1.1 Organization of Intelligent Control

vehicle modes of behavior. At the Sensing level, all information concerning the environment surrounding the vehicle, as well as its own internal state of health, is directed to the higher level. Figure 1.1, reproduced here from [Ref. 2] illustrates the idea, and Figure 1.2 illustrates the hierarchical nature of the intelligent controls required.

Part of the sensing and reflexive acting at the lowest level involves a high degree of servo-control over all six degrees of freedom of the vehicle motion. To effect proper control, not only must the autopilot be capable of accurate course and depth control, but also, commands for reflexive actions for avoidance or attack must be followed accurately. These commands can also include hovering while some form of underwater work is done.

B. AIM OF THE STUDY

This thesis is concerned with the lowest level of control--the control of vehicle reflexive maneuvers. It is assumed that the planning level control in Figure 1.2 recognizes the need for evasive action and decides on parameters such as speed, course, and depth changes to be rapidly implemented. These parameters are then fed to a series of stored maneuvers within the framework of a model based autopilot system. Figure 1.3 illustrates the concept of the "bag" of maneuvers as interfaced to the vehicle autopilot. The control concept proposed here is that of a

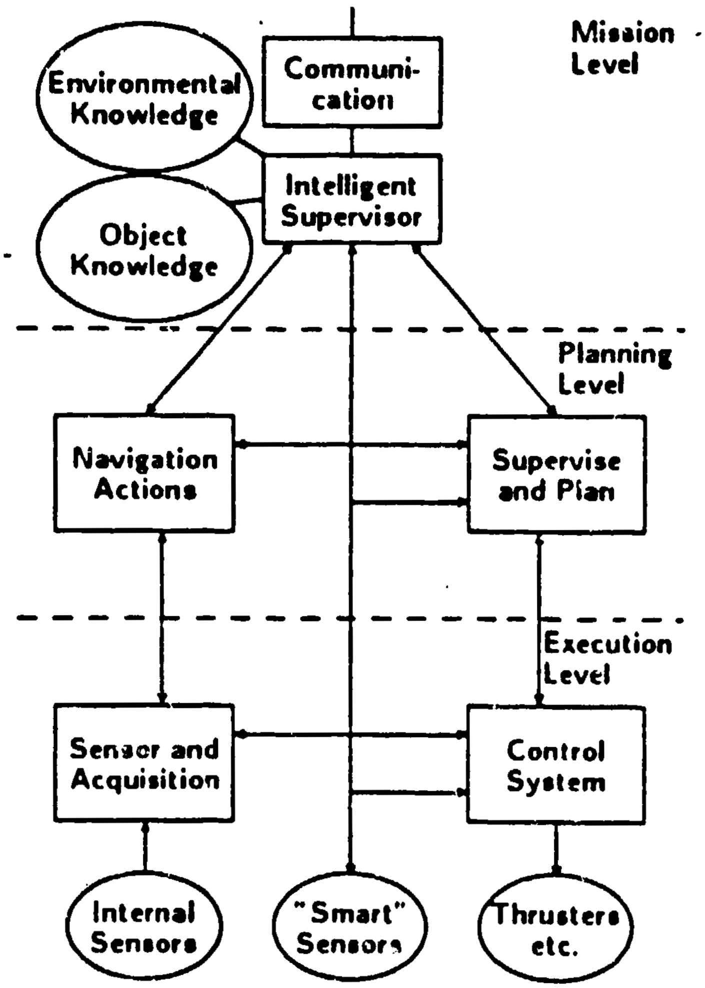

Figure 1.2 Hierarchical Control Structure

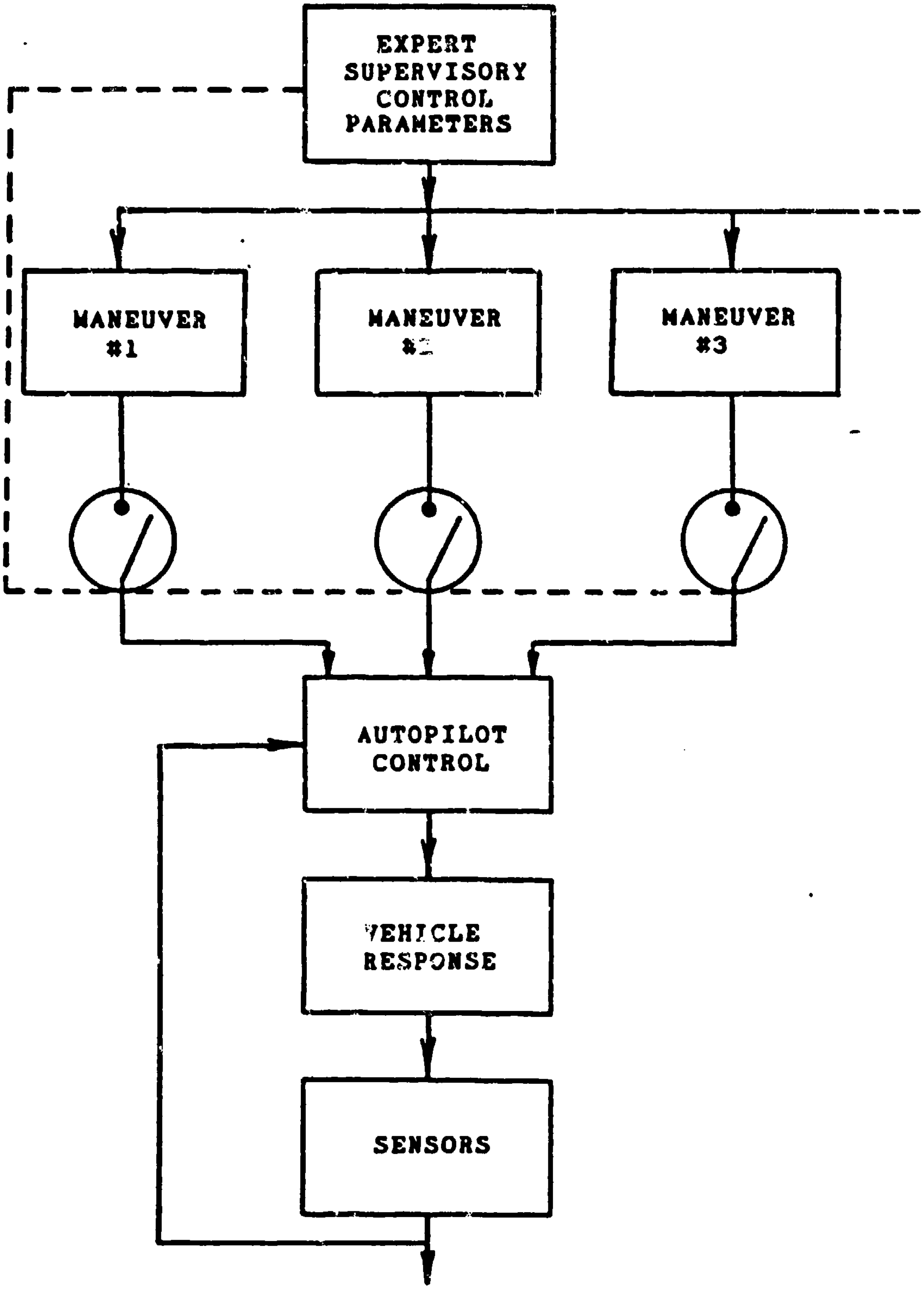

Figure 1.3 Schematic of Supervisory Type Control Scheme

developed for such maneuvers can be in the form of algorithms that provide a command generation system to the autopilot.

The purpose of this work is to determine the feasibility and the autopilot design methodology for:

1. the command generation logic, and
2. a model following autopilot control.

C. METHOD OF APPROACH

Since this work deals strongly with underwater vehicle dynamics and control, but not with the vehicle hydrodynamics per se, it was important to use an existing vehicle model as the basis for the work. Such a model (Figure 1.4) was provided by [Ref. 4] where the verification of the model by experimental data illustrated the reasonableness of its coefficients.

Using the equations of motions of the vehicle, the development of command generation logic, the design of the model following autopilot, and the AUV maneuvering performance, have been accomplished with computer simulation. Heavy use of the DSL (Dynamic Simulation Language) has been made. [Ref. 5]

The vehicle selected as the basis for the study is approximately 17 feet long and has been simulated over a range of speeds from 3 to 30 feet per second where a

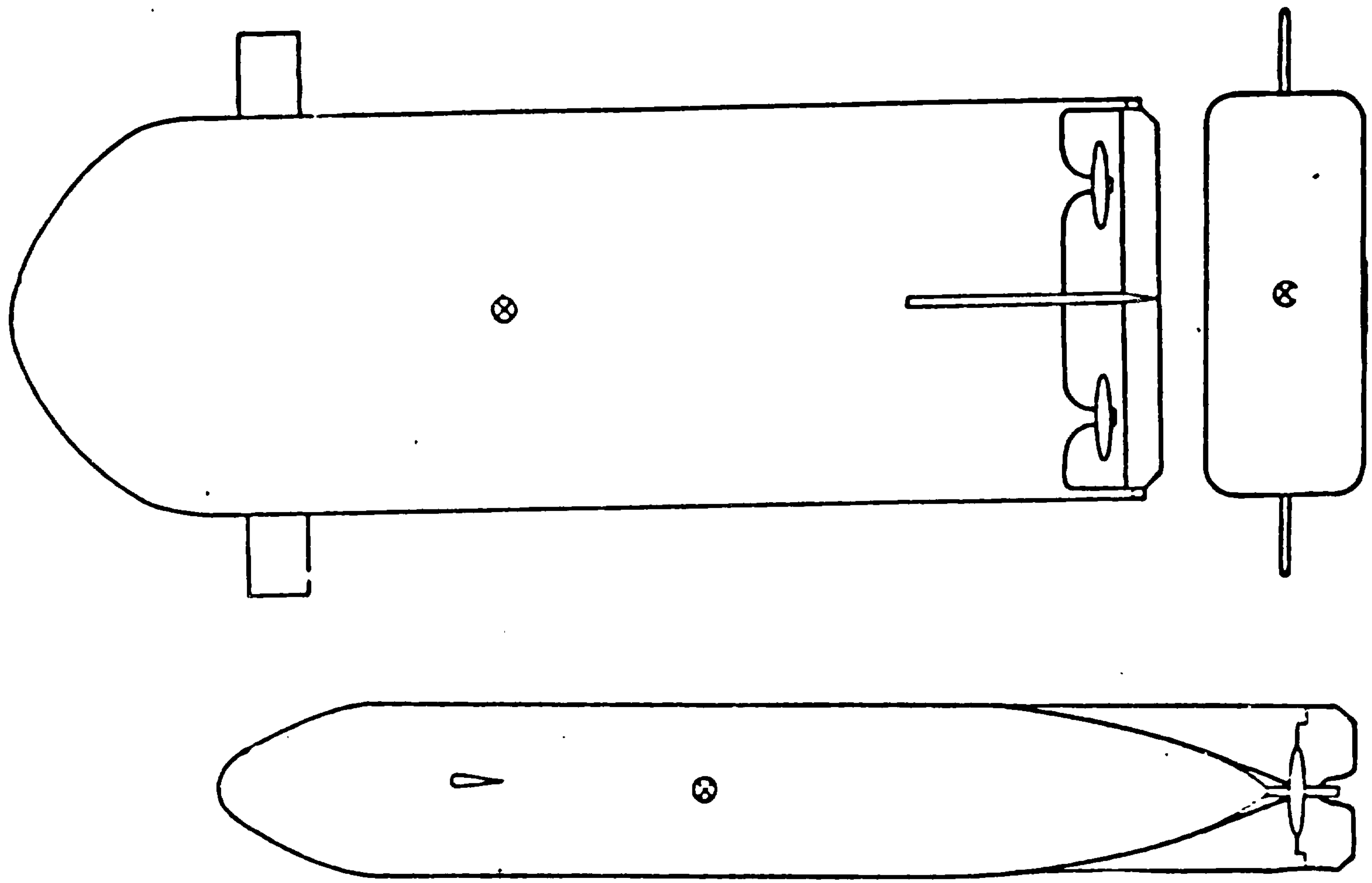

Figure 1.4 The AUV Geometry Selected

specific maneuver--a rapid dive to 100 feet--has been the focus for the command generation model.

While much remains to be done, the concept proposed appears worthy of future work.

II. VEHICLE DYNAMIC MODELING

A. GENERAL

This chapter describes the dynamics of a selected AUV. The three dimensional motion of an underwater vehicle is fully defined using two coordinate reference systems.

1. Body Fixed Coordinate Reference System--Figure 2.1.
2. Inertial Reference System--Figure 2.2.

The vehicle equations of motion are presented and how they were modified to suit the needs of an AUV. Also included as part of this chapter is a description of the derivation of the hydrodynamic coefficients and a brief discussion of the propulsion plant and crossflow drag modeling.

B. COORDINATE SYSTEMS

Three dimensional motions of underwater vehicles are normally described using two coordinate reference frames. The first is a right-handed orthogonal coordinate system fixed in the body. The second, an inertial reference frame, is used to define translational and rotational motions in global coordinates (Figure 2.1)

The body fixed coordinate reference frame has its origin fixed to the body center and is aligned with the body axis of symmetry. Components of the vehicle motion relative to this body fixed reference frame are defined as:

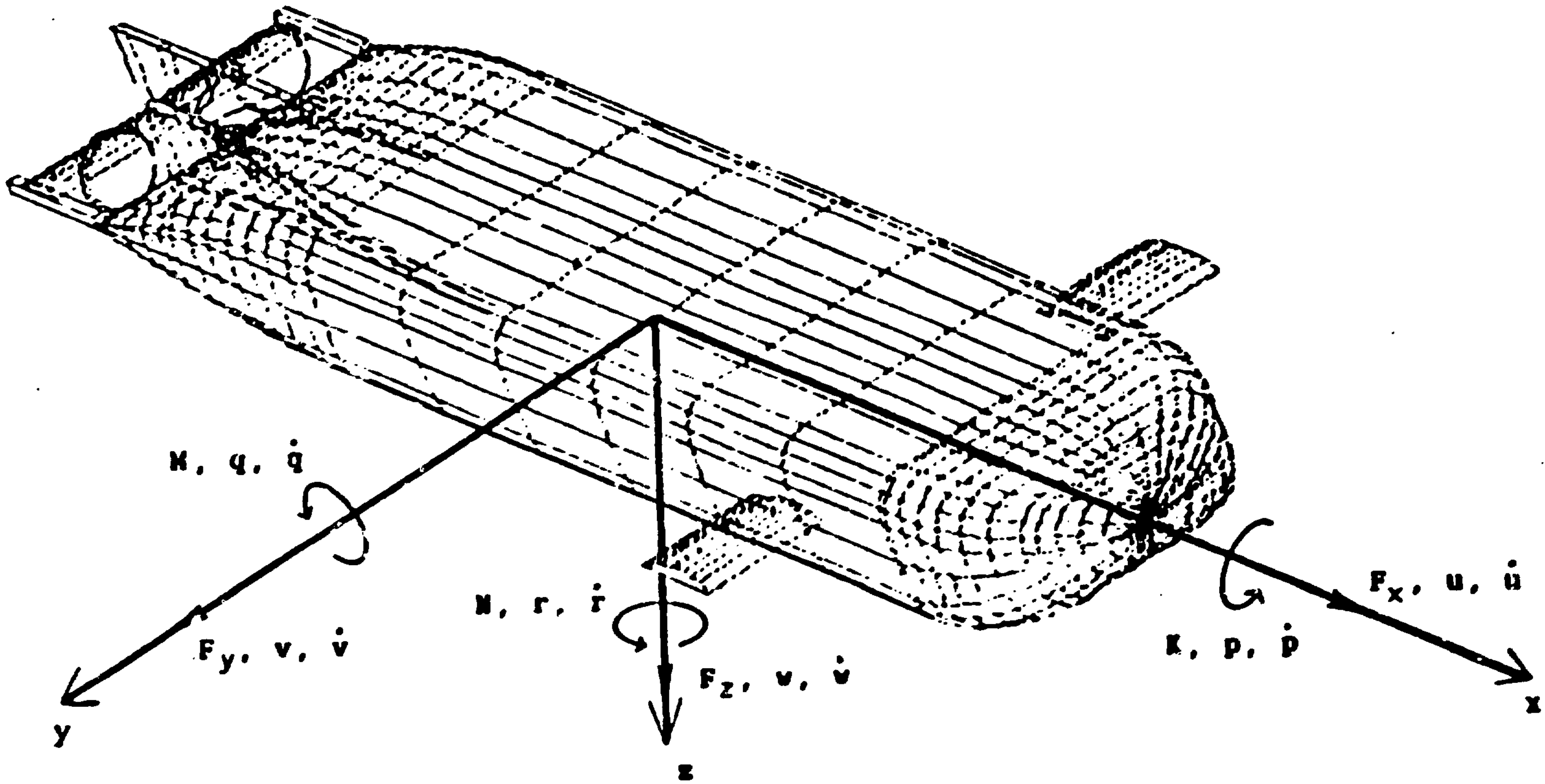

Figure 2.1 Body Fixed Coordinate Reference Frame

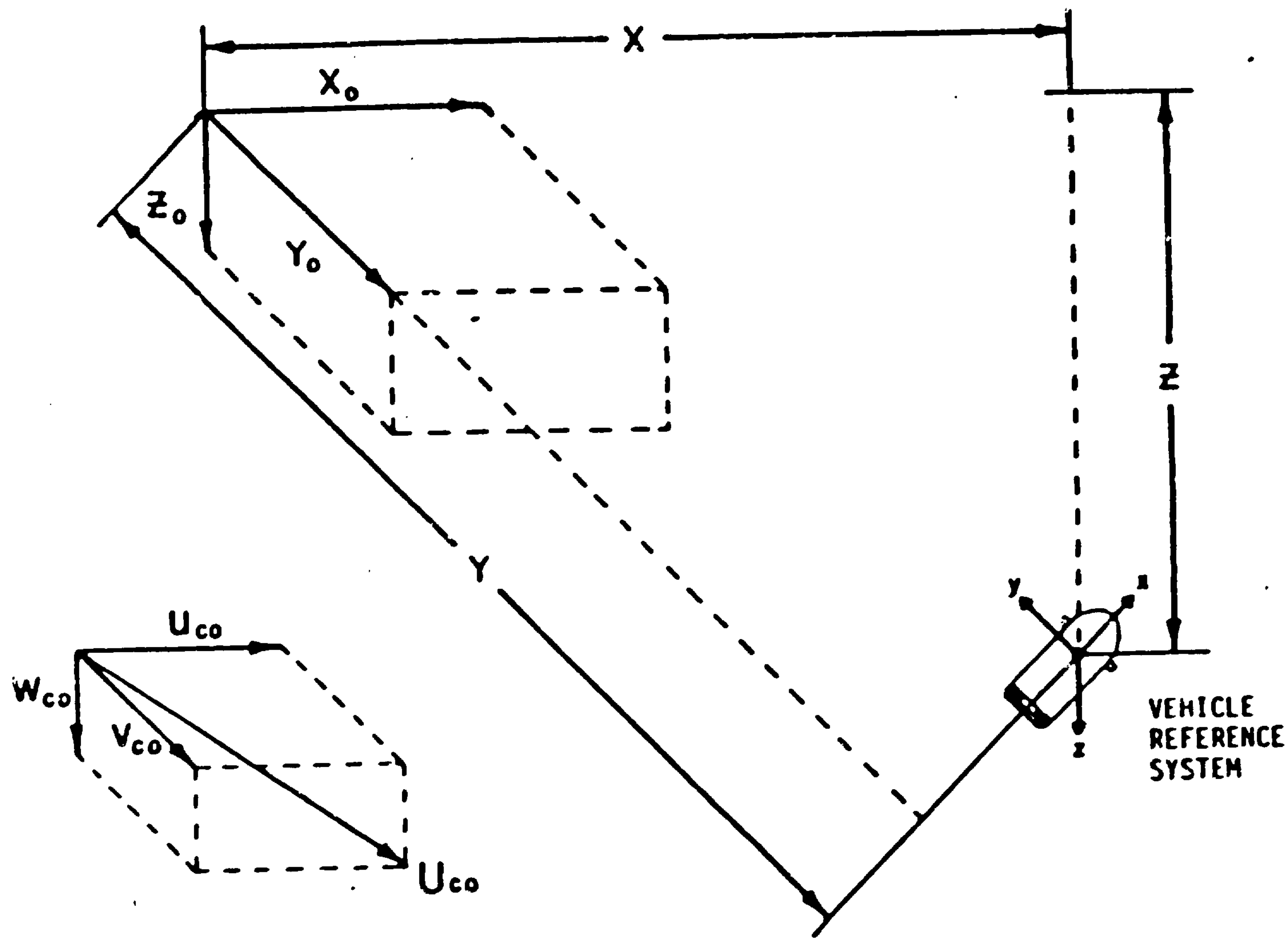

Figure 2.2 Inertial Reference Frame

u,v,w components along the body fixed axes of the velocity of the origin relative to the fluid (Surge, Sway and Heave velocity respectively).

p,q,r components along the body fixed axes of the angular velocity of body relative to the inertial reference system (Roll, Pitch, and Yaw rates) (Figure 2.2).

The <u>inertial reference frame</u> is also a right handed orthogonal coordinate system in which the position and orientation of the vehicle's coordinate system is specified. The orientation of the body-fixed coordinate system is described by Euler angles ψ (yaw), θ (pitch), ϕ (roll). The transformation from body-fixed to inertial is then given conveniently by an XYZ rotation sequence (ϕ, θ, ψ).

Position of the body-fixed coordinate system is then expressed in X, Y, and Z coordinates as illustrated in Figure 2.1. Orientation of the vehicle's coordinate system is expressed in Euler angles ϕ, θ, ψ.

C. RIGID BODY DYNAMICS AND EQUATIONS OF MOTION

The equations of motion for a six degree of freedom submarine vehicle are now standardized being first fully developed by Gertler and Hagen [Ref. 6]. These equations are commonly known today as the DTNSRDC 2510 equations of motion.

Modifications to these standard equations are then generally made to reflect the particular hydrodynamic characteristics and properties of the underwater vehicle being considered. [Ref. 6] Among the most significant changes/allowances considered for the AUV in this study included an integral formulation of the viscous crossflow forces and moments, addition of the effect of an external current and perhaps the most significant difference is the change made due to the non-conventional shape of the AUV. The AUV considered here is peculiar in that its shape is more of low aspect ratio wing than that of the conventional body of revolution. [Ref. 4] Additional modifications were also made by the separation of the coupled input for bow and stern planes and also the decoupling of the bow planes so that purposely induced roll control could be included.

The equations of motion for the six degree of freedom AUV are listed in Appendix A, in the following form:

$$\underline{M}\,\dot{\underline{x}} = \underline{f}(\underline{x},\underline{z},\underline{u}) \tag{2.1}$$

where,

$$\underline{M} = \text{MASS MATRIX} \tag{2.2}$$

$$\dot{\underline{x}} = [\dot{u},\dot{v},\dot{w},\dot{p},\dot{q},\dot{r}]^T \tag{2.3}$$

$$\underline{f} = [F_x,F_y,F_z,K,M,N]^T \tag{2.4}$$

and F_x, F_y, F_z are hydrodynamic forces and K, M, N are hydrodynamic moments,

$$\underline{x} = \begin{bmatrix} u \\ v \\ w \\ p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \text{surge} \\ \text{sway} \\ \text{heave} \\ \text{roll} \\ \text{pitch} \\ \text{yaw} \end{bmatrix} = \text{Body Coordinate States} \tag{2.5}$$

$$\underline{z} = \begin{bmatrix} X \\ Y \\ Z \\ \phi \\ \theta \\ \psi \end{bmatrix} \quad \left.\begin{matrix} \text{Position} \\ \\ \\ \text{Orientation} \end{matrix}\right] \text{Inertial Reference System} \tag{2.6}$$

$$\underline{u} = \begin{bmatrix} \delta_r \\ \delta_{bs} \\ \delta_{bp} \\ \delta_s \\ \delta_{rm} \\ \delta_B \end{bmatrix} = \left.\begin{matrix} \text{rudder angle} \\ \text{ST\&D bow plane angle} \\ \text{port bow plane angle} \\ \text{stern plane angle} \\ \text{delta form} \\ \text{delta buoyancy} \end{matrix}\right] \text{control vector} \tag{2.7}$$

and $\underline{u}$ is distinguished by context from u--surge velocity of the vehicle relative to the surrounding water, or U_{CO} for the current.

In addition to the six equations of motion that define the AUV's motion relative to the body fixed coordinate system, six additional equations are required to fully specify the vehicle's motion in space. These kinematic relations (see Appendix A) specify the position and orientation of the body coordinates with respect to an inertial reference frame as established by the XYZ rotations, and are expressed in terms of linear velocities and Euler angular rates.

D. HYDRODYNAMIC COEFFICIENTS

Although development of the hydrodynamic coefficients is not a thrust of this thesis, a brief description of their derivation is warranted.

The hydrodynamic coefficients provide the source of the behavioral characteristics, and thus the responsiveness, of a particular underwater vehicle.

These coefficients are the result of a Taylor series expansion, in which only the first order terms are saved, based on the motion variables of the hydrodynamic forces and moments. The hydrodynamic coefficients are non-dimensionalized and can be considered constants within limited operating ranges. [Ref. 6]

There are currently two primary methods utilized for obtaining hydrodynamic coefficients. The first is based on tow tank experiments using planar motion, and rotating arm mechanisms. The second is a geometric analytical approach using semi-empirical techniques. [Ref. 4]

The coefficients used for this thesis are those that were determined using the analytic approach for an SDV simulator. [Ref. 4]

The coefficients thus selected were chosen because of convenience and availability rather than any particular desirability of the hydrodynamic characteristics implied.

E. PROPULSION AND CROSSFLOW DRAG MODELING

1. Propulsion Plant Modeling

In NCSC's report by Crane, Summey and Smith [Ref. 4], propulsion plant modeling is discussed. In that report they list the effects of propulsion on the motion of a submersible.

propulsion thrust
propeller slipstream effects
propulsive torque
propulsion induced hull effects

Of these four effects only the first two are considered substantial and the last two are considered negligible.

The propulsive thrust equation was derived by NCSC by curve fitting experimental data and the propeller slipstream effects are modeled as a function of vehicle speed, propeller rpm, and geometry. [Ref. 4]

2. Crossflow Drag Modeling

Since the AUV geometry selected in this study is essentially a low aspect ratio wing design and not a body of revolution, its body cross-sections are nearly rectangular rather than circular. Because of this an integral strip theory formulation of crossflow forces and moments was developed and incorporated into the equations of motion as given in Appendix A.

F. BOW PLANE INFLUENCE

Bow plane action serves to augment stern plane control over pitch motions, but adds to the hydrodynamically induced drag on the vehicle. When port and starboard bow planes are separately controlled, active control over vehicle roll motion may be accomplished. Thus the coefficients relating to the heave and pitch motions, axial drag, and roll motions have been modified here to allow separate active roll control.

III. LINEARIZATION OF THE VEHICLE EQUATIONS OF MOTION

A. GENERAL

The overall objective of this chapter is to fully describe the techniques used to linearize the highly non-linear equations of motion. A step by step and term by term development of the linearized equations are presented and all variables are completely specified in their relation to the AUV in this study.

A description of the linearization point and the ramifications of linearization about a straight line path is also considered in this section.

B. LINEARIZATION PROCEDURES

Linearization of the vehicle dynamics is required for the design of the vehicle control system. The linearized equations also serve as the model reference for the controller. The desired form is the state space representation of the equations of motion given as,

$$\underline{M}\, \Delta\dot{\underline{x}} = \underline{A}\, \Delta\underline{x} + \underline{B}\, \Delta\underline{u} \tag{3.1}$$

As discussed in Chapter II, the vehicle dynamics are represented in the following form:

$$\underline{M}\, \dot{\underline{x}} = \underline{f}(\underline{x}, \underline{z}, \underline{u}) \tag{3.2}$$

where $\underline{M}$ is the mass matrix, $\dot{\underline{x}}$ is the time derivative of the state vector $\underline{x}$ and $\underline{u}$ is the input vector. For the immediate purpose at hand, $\underline{z}$ may be considered to be part of the state vector $\underline{x}$. Proper separation will be discussed in what follows.

Linearization is accomplished by a Taylor series expansion about a nominal path or trajectory given generally by $(\underline{x}_o(t),\underline{u}_o(t))$, with only the first order terms being retained. The following form is then obtained:

$$\underline{M}\,\dot{\underline{x}}_o + \underline{M}\,\Delta\dot{\underline{x}} = \underline{f}(\underline{x}_o,\underline{u}_o) + \frac{\partial \underline{f}(\underline{x}_o,\underline{u}_o)\,\Delta\underline{x}}{\partial \underline{x}} + \frac{\partial \underline{f}(\underline{x}_o,\underline{u}_o)\,\Delta\underline{u}}{\partial \underline{u}} \tag{3.3}$$

where, if $\Delta\underline{x} = (\underline{x} - \underline{x}_o)$, and $\Delta y = (y - \underline{u}_o)$, and Equation (3.3) becomes,

$$\underline{M}\,\Delta\dot{\underline{x}} = \frac{\partial \underline{f}(\underline{x}_o,\underline{u}_o)\,\Delta\underline{x}}{\partial \underline{x}} + \frac{\partial \underline{f}(\underline{x}_o,\underline{u}_o)\,\Delta\underline{u}}{\partial \underline{u}} \tag{3.4}$$

Defining $\underline{A} = \dfrac{\partial \underline{f}(\underline{x}_o,\underline{u}_o)}{\partial \underline{x}}$ and $\underline{B} = \dfrac{\partial \underline{f}(\underline{x}_o,\underline{u}_o)}{\partial \underline{u}}$ the desired state space form is obtained.

C. APPLICATION TO VEHICLE MODEL

The state space model is a 12 state model that can be separated into two state vectors $\underline{x}$ and $\underline{z}$. The state vector $\underline{x}$ represents the three linear velocities and corresponding three angular rates about an orthogonal coordinate system fixed in the body as defined in Equation (2.5).

The state vector $\underline{z}$ represents the six kinematic relations, three coordinate positions and three Euler angles and is defined in Equation (2.6).

The two sets of six equations that result are of the form:

$$\dot{\underline{x}} = \underline{M}^{-1}\underline{f}(\underline{x},\underline{z},\underline{u}) \tag{3.5}$$

$$\dot{\underline{z}} = \underline{g}(\underline{x},\underline{z}) \tag{3.6}$$

The control vector $\underline{u}$ is the input vector and is defined by Equation (2.7).

By combining both state vectors, the model state vector is defined,

$$\underline{X} = [\underline{x},\underline{z}]^T = [u\ v\ w\ p\ q\ r\ X\ Y\ Z\ \phi\ \theta\ \psi]^T \tag{3.7}$$

Once the model state vector and control vector are defined, the $\underline{A}$ matrix and $\underline{B}$ matrix must be determined. The $\underline{A}$ matrix formulation is represented,

$$\underset{12\times 12}{\underline{A}} = \begin{bmatrix} \dfrac{\underline{M}^{-1}\partial\underline{f}(\underline{x}_o,\underline{z}_o,u_o)}{\partial\underline{x}} & \dfrac{\underline{M}^{-1}\partial\underline{f}(\underline{x}_o,\underline{z}_o,u_o)}{\partial\underline{z}} \\ \dfrac{\partial\underline{g}(\underline{x}_o,\underline{z}_o)}{\partial\underline{x}} & \dfrac{\partial\underline{g}(\underline{x}_o,\underline{z}_o)}{\partial\underline{z}} \end{bmatrix} \tag{3.8}$$

and by similar formulation the $\underline{B}$ matrix is,

$$\underset{12 \times 6}{\underline{B}} = \begin{bmatrix} \dfrac{\underline{M}^{-1} \partial \underline{f}(\underline{x}_o, \underline{z}_o, \underline{u}_o)}{\partial \underline{u}_o} \\ \\ \dfrac{\partial \underline{g}(\underline{x}_o, \underline{z}_o)}{\partial \underline{u}_o} \end{bmatrix} \tag{3.9}$$

An element by element formulation of the $\underline{A}$ and $\underline{B}$ matrices are complex and require careful attention. The particular functional form of the derivative expressions can, however, be obtained analytically and depending on whether $\underline{x}_o$ and $\underline{z}_o$ are time dependent or constant, the analytical derivatives become time variant or not. For the case of linearization about a straight line flight path, these derivatives are constant which makes the control computations easier than for more complicated nominal flight conditions.

D. LINEARIZED VARIABLES ABOUT STRAIGHT FLIGHT PATHS

One convenient feature concerning the linearization about a straight flight path with forward speed, $\underline{u}_o$, is that $\underline{A}$ and $\underline{B}$ become constant matrices where the coefficients are relatively simple functions of the forward speed. Also, since the nominal path is associated with neither rotation nor cross-track or depth translation, the incremental variables $\Delta\underline{x}$ and $\Delta\underline{u}$ are identical to the actual variables $\underline{x}$

and $\underline{u}$ except for the longitudinal velocity and position. The linearized dynamics in the axial direction become:

$$\frac{d}{dt}\underline{x} = \underline{u}_o \tag{3.10}$$

so that as far as the linearized system dynamics are concerned $\Delta\underline{x}(1) = 0$ and $\Delta\underline{x}(t) = \underline{u}_o(t)$. While this feature is convenient, it does not provide information on the second order effect of control surface action slowing down the vehicle.

A possible approach to alleviating this deficiency in the linear model could be to modify the axial direction equ ion of motion so that the drag effects of control surface action are related to $|\delta_2|$ rather than δ_s^2. This is beyond the scope of this thesis.

IV. AUTOPILOT DESIGN USING OPTIMAL CONTROL TECHNIQUES

A. GENERAL

This chapter contains a review of optimal control techniques as developed and used in this study for the control of autonomous underwater vehicles. Such an autopilot has been classically treated as a series of interconnected feedback loops for independent control of depth and control of course and heading, while roll control of the vehicle has been left passive. Control of the sixth degree of freedom, longitudinal velocity, has not been considered important and a constant thrust or propeller speed has been assumed.

While control of all six degrees of freedom may be important in the end for future AUV operations, and particularly in the transition from cruise to hover modes, this is not the primary focus here. Instead, this chapter deals with the state of the art in systems concepts for underwater vehicle course and depth control, together with a review of the modern multivariable system controls methods used in modern autopilot design.

B. CLASSICAL CONTROL OF COURSE AND DEPTH

Simple autopilots have long been of interest in relieving the human operator of onerous tasks and preventing fatigue. Classical design techniques have considered depth and course control as separate, non-interacting control systems. The depth controller directs commands to the stern planes based on an error between pitch angle command and vehicle pitch angle where the pitch command is proportional to depth error. Course heading controllers provide rudder angle commands proportional to heading angle error. Walker [Ref. 7] recently proposed the addition of a cross track position feedback loop using yaw angle damping to control the cross track distance for automatic track control.

Most vehicle controllers in practice rely on classical concepts with protection limits on command signals so that control surface commands can be limited in magnitude and rate. Adaptive steering controllers have been proposed as an extension for course maintenance in heavy seas when optimized gain settings are based on calm weather ship characteristics [Ref. 8]. The main limitation of autopilot designs based on classical concepts are,

1. Ship characteristics vary strongly with speed so that gain settings for all of the major loops have to be adjusted to maintain optimum performance under wide operating conditions.

2. Gains set based on maximum actuation limits and usually designed to regulate vehicle depth and course about nominally fixed reference settings.

3. Control of depth and course changes (i.e., rapid maneuvering) is not easy and usually not automated.

The control of rapid maneuvering is more suited to the more recent multivariable control system structures such as those involving Model Following Controllers, and Model Based Compensators as proposed by Milliken [Ref. 9].

C. REVIEW OF OPTIMAL CONTROL CONCEPTS RELATING TO CONTROLLER DESIGN

1. LQR Summary and Discussion

Much has been written about the application of Optimal Control Concepts to the design of feedback systems for both output regulation and input tracking. Kwaakernaak and Sivan [Ref. 10] present a discussion of design methods based on state of the art to 1971. Kaufman and Berry [Ref. 11] have provided examples of autopilot design methods based on linear optimal regulator (LQR) methods and model following techniques. Milliken [Ref. 9] has showed recently, the use of Model Based Compensators for providing multi-degrees of freedom control for a submarine depth and course control using linear control techniques--similar to those used in this work. Most recently, non-linear control methods have been proposed by Slotine [Ref. 12], and Yoeger and Slotine [Ref. 13] to provide robust trajectory control for underwater vehicles. Using linear control procedures, the vehicle, or object to be controlled is described by a linear state variable dynamic model for response computation by equations of the form,

$$\dot{\underline{x}}_p(t) = \underline{A}_p\, \underline{x}_p(t) + \underline{B}_p\, \underline{u}_p(t); \quad \underline{x}_p(0) \text{ given} \tag{4.1}$$

in which the matrices $\underline{A}_p$ and $\underline{B}_p$ represent constant coefficient terms, and $\underline{x}_p(t)$ and $\underline{u}_p(t)$ respectively, represent the vector of motions (positions and velocities) and the control actions (control surface deflections).

The design of a linear optimal regulator (LQR) control is based on the notion that if some non-zero initial condition, $\underline{x}(0)$, is established, then $\underline{u}_p(t)$ can be designed so that the non-zero state values can be reduced to the equilibrium values $\underline{x}(t) = 0$, $\underline{\dot{x}}(t) = 0$ with a control operation given by,

$$\underline{u}_p(t) = - \underline{K}\, \underline{x}_p(t) \tag{4.2}$$

where $\underline{K}$ is found from the minimization of the quadratic performance index,

$$J = \int_0^\infty (\underline{x}^T \underline{Q}\, \underline{x} + \underline{u}^T \underline{R}\, \underline{u})\, dt \tag{4.3}$$

Here, $\underline{Q}$ is a non-negative definite square symmetric weighting matrix for response magnitudes and $\underline{R}$ is a positive definite square symmetric weighting matrix for control effort. $\underline{Q}$ is size nxn, and $\underline{R}$ has rank equal to the number of control inputs modeled (r).

The solution for $\underline{K}$ becomes a matrix of size rxn found as the solution to,

$$\underline{K} = \underline{R}^{-1}\,\underline{B}^T\,\underline{P} \qquad (4.4)$$

where

$$\underline{P}\,\underline{A} + \underline{A}^T\underline{P} + \underline{Q} - \underline{P}(\underline{B}^{-1}\underline{K}^{-1}\underline{B}^T)\,\underline{P} = 0 \qquad (4.5)$$

The eigenvalues of the closed loop regulator are determined from the combined state and co-state system equations. They are given by the eigenvalues of the composite matrix $\underline{SS}$ where,

$$\underline{SS} = \begin{bmatrix} \underline{A} & \underline{B}\,\underline{R}^{-1}\,\underline{B}^T \\ -\underline{Q} & -\underline{A}^T \end{bmatrix} \qquad (4.6)$$

It can be show [Ref. 10] that $\underline{P}$ is also given by,

$$\underline{P} = [\underline{W}_2]\,[\underline{W}_1]^{-1} \qquad (4.7)$$

where $\underline{W}_1$, $\underline{W}_2$ are the nxn partitions of the matrix, $\underline{W}$,

$$\underline{W} = \begin{bmatrix} \underline{W}_1 \\ \underline{W}_2 \end{bmatrix} \qquad (4.8)$$

formed from columns of stable eigenvectors of $\underline{SS}$. It has also been found that the use of real part and imaginary parts of a complex conjugate eigenvector as adjacent columns of $\underline{W}$ where a complex pole pair exist, eliminates the need for complex matrix inversion [Ref. 10].

The design by minimization of J in Equation (4.3) yields the closed loop control system equations,

$$\dot{\underline{x}}_p(t) = (\underline{A} - \underline{B}\ \underline{K})\underline{x}_p(t); \quad \underline{u}_p = -\underline{K}\ \underline{x}_p; \quad \underline{x}_p(0) \text{ given} \qquad (4.9)$$

where the steady state response is zero for both $\underline{x}_p$ and $\underline{u}_p$.

The state vector may, in many cases, be considered as a deviation vector from a desired constant level, and it is quite appropriate for the steady state values of $\underline{x}_p$ and up to go to zero. However, in the reality of some cases, the maintenance of a constant level in some elements of the state vector requires a non-zero steady state control signal level and in these cases $\underline{u}_p(\infty) = 0$. If a steady state level must be maintained for any element of the state vector, the steady state equations are first solved as,

$$0 = \underline{A}_p\ \underline{x}_p(\infty) + \underline{B}_p\ \underline{u}_p(\infty) \qquad (4.10)$$

Equation (4.10), subtracted from Equation (4.1) reduces these cases to the equivalent of a regulator control problem by shift of variable,

$$\dot{\underline{x}}_p(t) = \underline{A}_p(\underline{x}_p(t) - \underline{x}_p(\infty)) + \underline{B}_p(\underline{u}_p(t) - \underline{u}_p(\infty)) \qquad (4.11)$$

where the new variables

$$\underline{x}_p(t) - \underline{x}_p(\infty) \tag{4.12}$$

and

$$\underline{u}_p(t) - \underline{u}_p(\infty) \tag{4.13}$$

are related in a control law

$$\underline{u}_p(t) - \underline{u}_p(\infty) = -\underline{K}(\underline{x}_p(t) - \underline{x}_p(\infty)) \tag{4.14}$$

or

$$\underline{u}_p(t) = \underline{u}_p(\infty) - \underline{K}(\underline{x}_p(t) - \underline{x}_p(\infty)) \tag{4.15}$$

The above discussion has been limited to deterministic signals and to the assumption that all physical state variables are either measurable or determined in a full state observer [Ref. 10].

Where the output of the controlled process is to be regulated, the above techniques may be used to design the elements of the feedback gain matrix thus avoiding the complex task of designing separate control loops from each variable in the process. The method is powerful, but requires skill in the selection of appropriate Q and R factors.

2. Tracking Control Systems--(LMFC + MRAC)

Where the control system is required to drive a process so that the output tracks an input variable within acceptable error bounds, the problem is further compounded. Even more difficult is to achieve the tracking of several simultaneous inputs by the various outputs of the driven process. During the late 1960's and early 1970's, much attention was placed on linear model following controls (LMFC) and model reference adaptive controls (MRAC) to provide the acceptable tracking behavior of multivariable systems. Kaufman and Berry [Ref. 11] described the application to flight control, and Landau [Refs. 14,15] gave a survey of design techniques and system structures in which it became clear that a model of the system to be controlled was needed to represent the desired time behavior of the system state variables. The system control variables then became a function of the input variables to the model, and the model state variables, in addition to the feedback of system state variables. Thus better information than could be derived by feedback was used to drive outputs to track inputs.

The use of MRAC techniques allows for not only model following, but also the provision of adapting gains, or model parameters in achieving precise control when system operating conditions change.

One of the difficulties pointed out by Landau [Ref. 14], is that controller parameters need to change when the plant operating conditions changed. Thus using a reference model not only provides the robustness achieved by predictive and corrective control but also provides the opportunity to update model and control parameters automatically.

Restricting the discussion to Linear Model Following Controls (LMFC), the control issues are analyzed as follows: the plant model is given by,

$$\dot{\underline{x}}_p = \underline{A}_p \underline{x}_p + \underline{B}_p \underline{u}_p \tag{4.16}$$

$$\dot{\underline{y}}_p = \underline{C}_p \underline{x}_p \tag{4.17}$$

and a suitable model of the plant, but with desirable dynamic response characteristics (response time, stability, etc.) is given by,

$$\dot{\underline{x}}_m = \underline{A}_m \underline{x}_m + \underline{B}_m \underline{u}_m(t) \tag{4.18}$$

$$\dot{\underline{u}}_m = 0 \tag{4.19}$$

$$\underline{y}_m = \underline{C}_m \underline{x}_m \tag{4.20}$$

then the control signals, $\underline{u}_p$, which minimize the weighted integral of errors between model and plant are given by,

$$\underline{u}_p = \underline{K}_1 \underline{u}_m + \underline{K}_2 \underline{x}_m + \underline{K}_3 \underline{x}_p \tag{4.21}$$

where the errors are defined as,

$$\varepsilon = \underline{y}_m - \underline{y}_p \tag{4.22}$$

and the performance index minimized is,

$$J = \int_0^\infty (\underline{\varepsilon}^T \underline{Q}\, \underline{\varepsilon} + \underline{u}_p^T \underline{R}\, \underline{u}_p)\, dt \tag{4.23}$$

and $\underline{Q}$, and $\underline{R}$ are weighting matrices as discussed earlier.

The computation of the gain matrices, $\underline{K}_1$, $\underline{K}_2$, and $\underline{K}_3$ are fortunately made easier by considering the combined system, model plus plant as a coupled linear system. Also, to overcome problems that arise when the signals to be tracked, $\underline{y}_m(t)$ are derived from inputs, $\underline{u}_m(t)$, that are not impulses, it is convenient to consider that the additional model equations,

$$\dot{\underline{u}}_m = 0 \tag{4.24}$$

be incorporated together with u_m as a composite state vector,

$$\underline{x}^T = [\underline{x}_p{}^T, \underline{x}_m{}^T, \underline{u}_m{}^T] \quad (4.25)$$

to get the system equations,

$$\frac{d}{dt}\begin{bmatrix} \underline{x}_p \\ \underline{x}_m \\ \underline{u}_m \end{bmatrix} = \begin{bmatrix} \underline{A}_p & 0 & 0 \\ 0 & \underline{A}_m & \underline{B}_m \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \underline{x}_p \\ \underline{x}_m \\ \underline{u}_m \end{bmatrix} + \begin{bmatrix} \underline{B}_p \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} \underline{u}_p \end{bmatrix} \quad (4.26)$$

Now, application of the LQR technique to the composite system given above yields,

$$\underline{u}_p = -[\underline{R}^{-1}\ \underline{B}_p{}^T][\underline{P}][\underline{x}_p\ \underline{x}_m\ \underline{u}_m]^T \quad (4.27)$$

where $\underline{P}$ now is of dimension (np + nm + rm), and $[\underline{R}^{-1}\ \underline{B}^T\ \underline{P}]$ is partitioned in three parts,

$$[\underline{R}^{-1}\ \underline{B}^T\ \underline{P}] = [\underline{K}_3 \quad \underline{K}_2 \quad \underline{K}_1] \quad (4.28)$$

By varying the weighting factors within the matrix, $\underline{Q}$, selected errors may be penalized more heavily than others in the optimal control trade-off. Also, selection of parameters within $\underline{R}$ may be used to provide a trade-off between a sluggish or sensitive control design. Details of numerical values used in the design of the autopilot controls are given later in Chapter VI.

3. Near Time Optimal Maneuvering Models

The use of the system structure implied by Equation (4.26) and the resulting control law, Equation (4.21), is particularly useful when considering near time optimal positioning of inertial objects. It is well known that time optimal position control of a massive object requires a bang-bang application of force or torque. These concepts are recently being considered in robot tracking control [Ref. 16], and the sliding control described in [Ref. 17]. So also, in the field of LMFC for underwater vehicle maneuvering control, it is expected that rapid maneuvering will require some form of bang-bang operation of control surfaces. Bang-bang operation, in principle, is simple, consisting of a sequence of stepwise control actions, yet knowledge of switching times for anything other than very low order systems make the principle difficult to implement. The outcome of the above discussion then leads to the development of vehicle maneuvering models based on use of a series of constant setting for control surfaces that make $\underline{u}_p$ the model input vector $\underline{u}_m$. At times during the response of the model where switching should occur, the control surfaces change setting rapidly as if by imposition of an impulse command. Therefore, if it is considered that surface settings change levels at discrete but arbitrary times, the unforced nature of the model reference states, in Equation

(4.26) are preserved and the application of the LMFC system is valid.

For every reflexive maneuver envisioned during the operation of an AUV life, it is foreseen that maneuvering logic can be developed on an algorithmic basis to determine switching times, using logic to be developed and the $\underline{A}_m$, $\underline{B}_m$, $\underline{K}_1$, $\underline{K}_2$ and $\underline{K}_3$ matrices as shown in Figure 4.1. These data can be stored inside on-board processors so that on command from the high level controller or expert system, new computations for $\underline{u}_p$ can be implemented immediately.

The development of a maneuvering logic as a command generation system for a dive maneuver will be discussed in more detail.

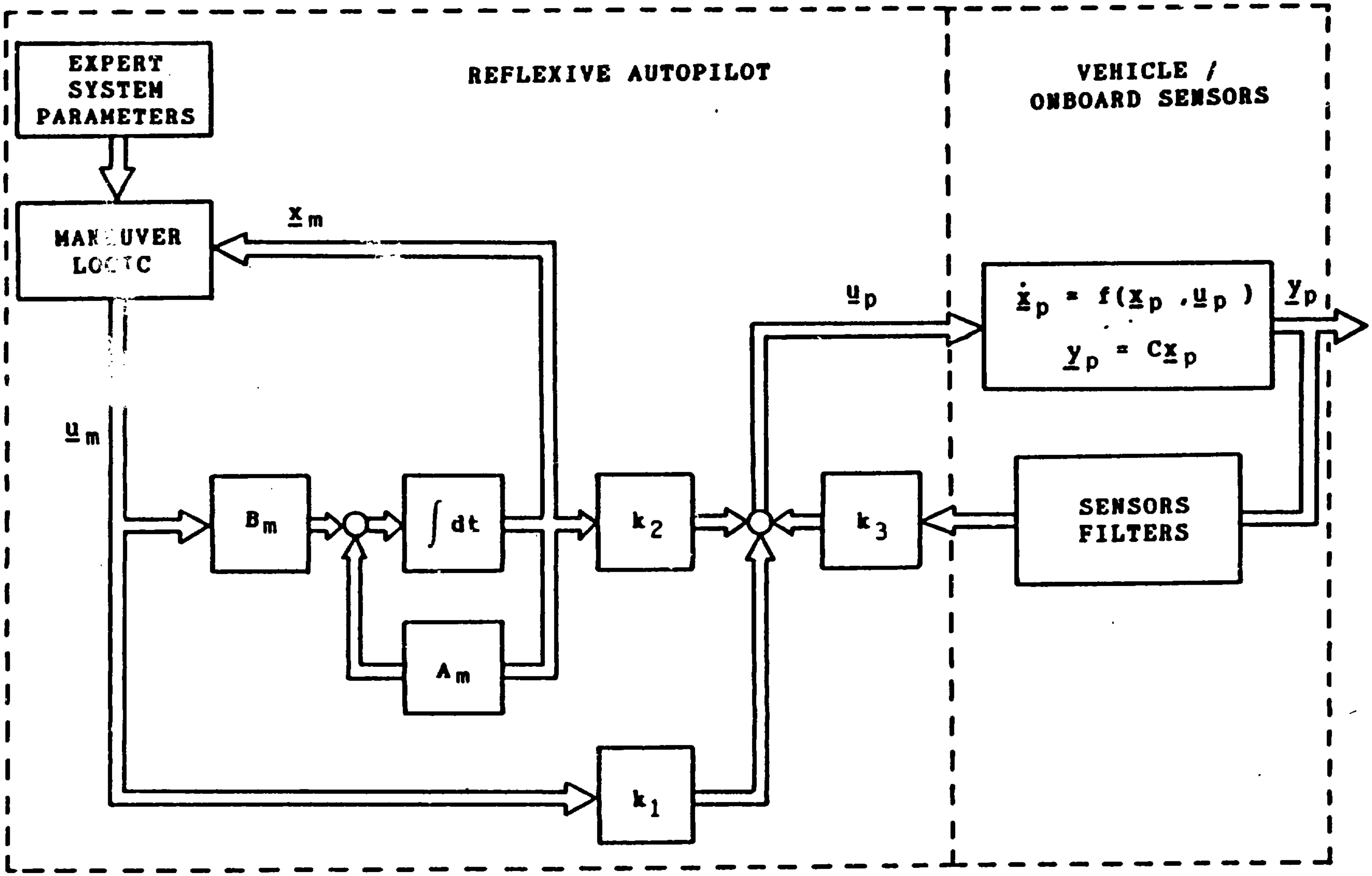

Figure 4.1 Block Diagram for a Model Following Autopilot

V. SIMULATION TECHNIQUES

A. GENERAL

This chapter contains a discussion of the computational program structures used in this study. All simulations were performed using the Dynamic Simulation Language (DSL) [Ref. 5] code for the simulation of linear and non-linear system response as a function of time. Internal numerical integration routines make this aspect of the solution transparent to the user. The user provides only the details of the particular equations employed. In this work, DSL was used for the simulation of both uncontrolled and autopilot controlled vehicle responses. However, as part of the design procedure for the autopilot, the complete set of feedforward and feedback gains were established using ETAT--a specially developed program for the computation of linear optimal control gains. The pertinent linkages between DSL and ETAT were developed and implemented during this study. More detailed descriptions follow.

B. COMPUTATION OF FEEDFORWARD AND FEEDBACK GAINS

While the theory behind the need for feedforward gains for optimal model following autopilots has been given in Chapter IV, this section discusses the program organization used in their computation.

The outline organization of program ETAT is shown in Figure 5.1. ETAT reads and writes values of $\underline{AA}$, and $\underline{BB}$, as computed within the framework of the DSL simulation and also reads user input values for the tracking error weighting matrix, $\underline{Q}$, and the control input weighting matrix, $\underline{R}$. Particular values used for $\underline{Q}$, and $\underline{R}$, are given later in Chapter VI.

Subroutine MTXEXP computes the matrix exponential associated with $\underline{AA}$, and the discrete time input matrix associated with $\underline{AA}$ and $\underline{BB}$, but this section has not been used here.

Subroutine ROOTS is used for the computation of both eigenvalues and eigenvectors of a square matrix ($\underline{AA}$), and calls the IMSL double precision library routine EIGRF and its associated subroutines.

OPTIMA is the subroutine used for assembly of the composite state and co-state matrix, $\underline{SS}$. OPTIMA also calls EIGRF and computes the closed loop system eigenvalues and vectors. These, as given earlier in Chapter IV, are used to form the solution of the matrix Ricatti equation and the overall matrix of gains, i.e., Equation (4.4). Partitions of the overall gain matrix give the individual matrices, $\underline{K}_1$, $\underline{K}_2$ and $\underline{K}_3$ in Equation (4.28).

A listing of the major subroutines used in program ETAT are provided in the appendix for the interested reader, although use of ETAT without proper linkages to DSL and the

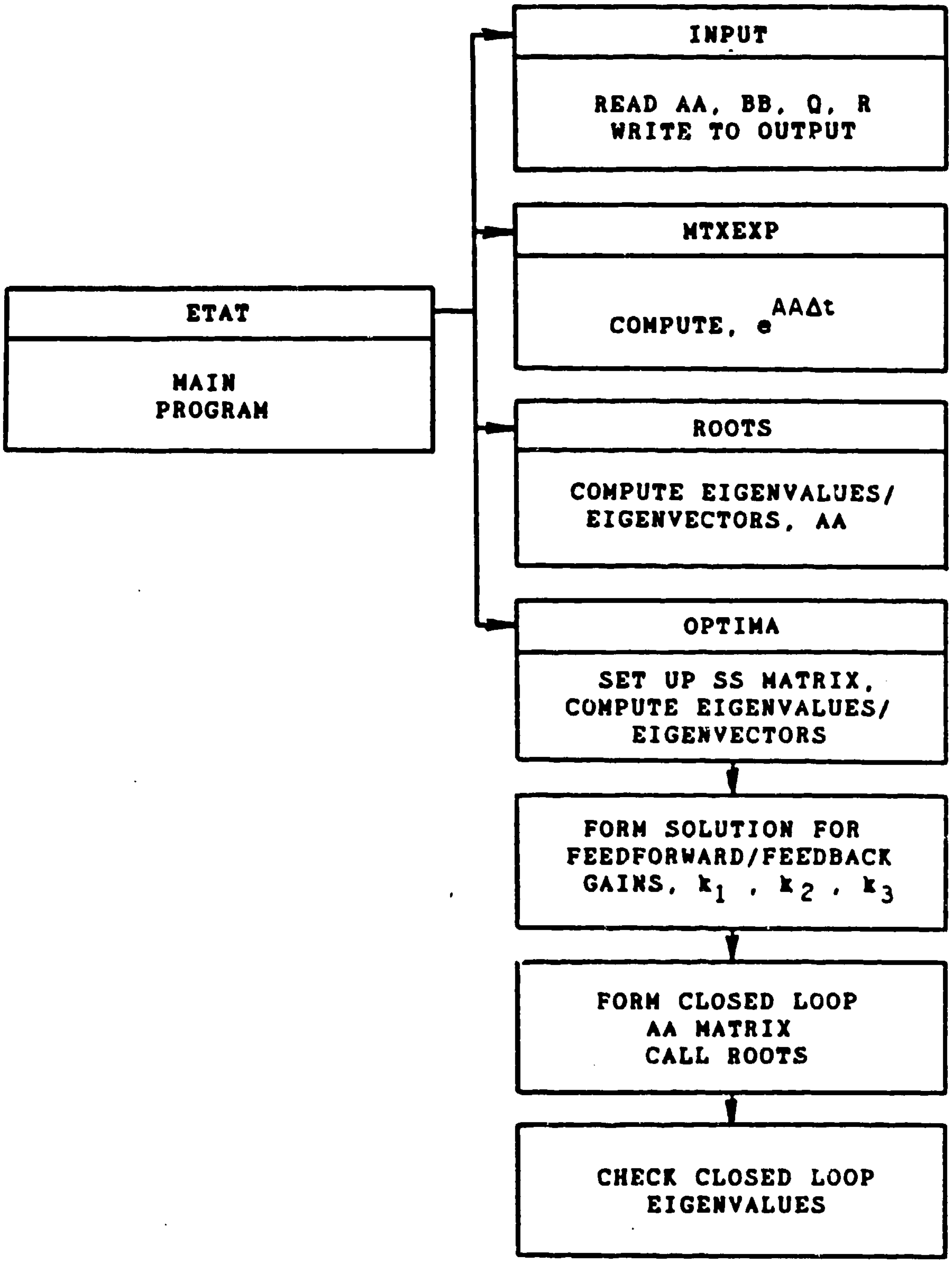

Figure 5.1 ETAT Flow Diagram

appropriate IMSL double precision library would not be proper.

C. REFERENCE MODEL DEVELOPMENT

As discussed earlier, the reference model is a full 12 state representation of the AUV. The reference model can be represented by:

$$\dot{\underline{x}}_m = \underline{AA}\ \underline{x}_m + \underline{BB}\ \underline{u}_m \qquad \dot{\underline{u}}_m = 0 \tag{5.1}$$

A computational problem in subroutine OPTIMA can arise because of the multiple zero eigenvalues associated with several of the modes in the above equations. This problem has been overcome here by inserting very small values, $-(\lambda)_i$, on the key diagonal elements of the $\underline{AA}$ matrix so that distinct eigenvalues result. Since the (λ) values are extremely small, their effect on the system poles is negligible and the problem of multiple repeated poles is eliminated.

It is conceivable to have a series of reference models, one for each of several reflexive maneuvers. Each maneuver will have its associated logic that will generate the control input to the model and thus provide the model reference states.

For this thesis, only one such maneuver, a dive maneuver, was investigated. Logic for the dive maneuver is based on an application of bang-bang optimal control theory, thereby yielding time optimal response. The methodology here is to deflect stern planes up and the bow planes down to initiate a pitch rate (p) until the vehicle achieves some predetermined pitch angle (θ). For what is considered reflexive, or emergency obstacle avoidance, a large angle is desired. Assuming that the submersible is directionally stable, some small stern plane angle must then be maintained to keep a constant pitch angle, dependent on speed, until such time when the control action should provide an opposite effect to come out of the dive and steady at a new depth. An example of this control action is shown in Figure 5.2.

Given limits on control surface deflection and maximum pitch angle during the dive, this type of control action should provide an optimal response for a change in depth.

With this control logic preprogrammed into an AUV, whenever the supervisory control system calls for a dive maneuver, the logic can provide the control input for the reference model and thus an optimal path can be created quickly; one that the controller can track and vehicle can follow.

The logic for the dive maneuver is crude, however, this is a trade-off for ease in programming the algorithm used for the dive maneuver. When programming one of these reflexive maneuvers one must be cautious not to program a maneuver that is beyond the capability of the vehicle.

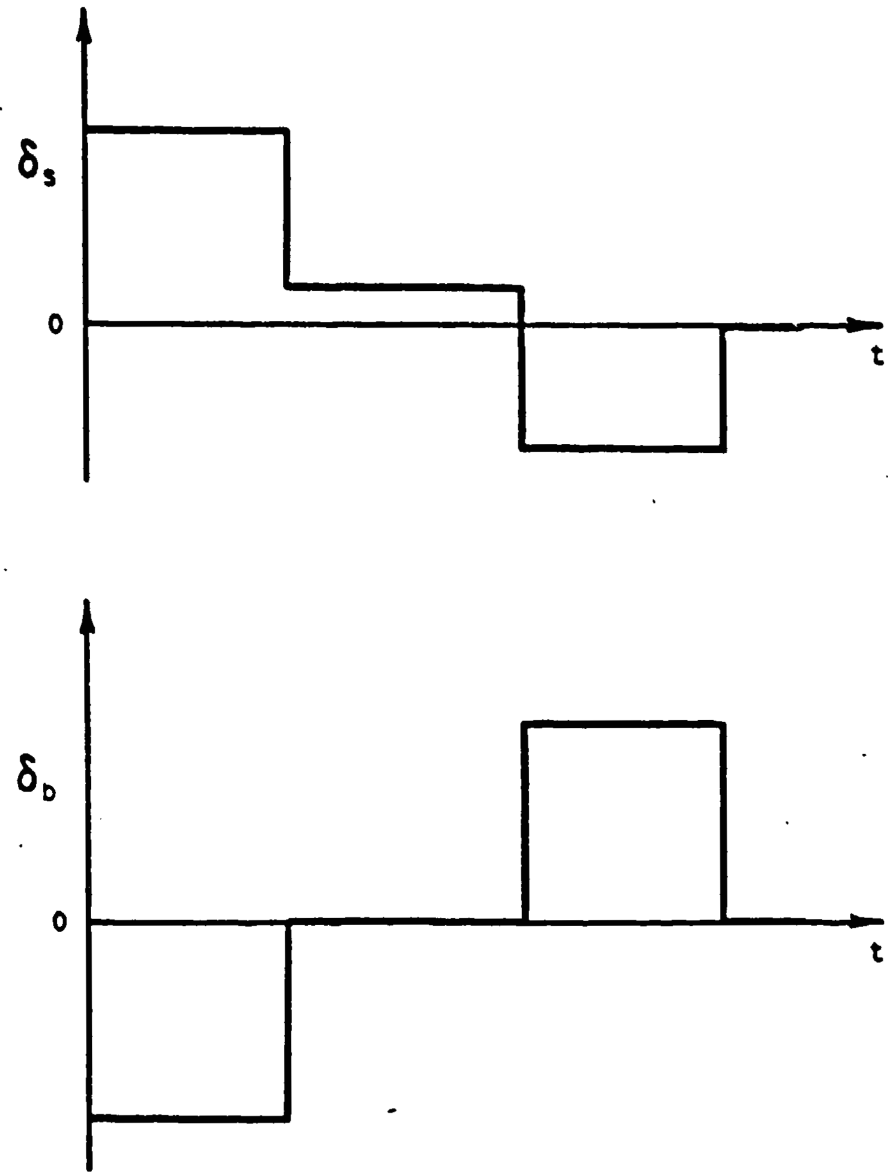

Figure 5.2 Time Optimal Control Action for a Dive Maneuver Command Generation

A conceptual objective is to have many maneuver algorithms preprogrammed into a vehicle into a "bag" of maneuvers. This bag of maneuvers would be at the disposal of the supervisory control. This supervisory control would be the manager of the bag of maneuvers, as earlier indicated in Figure 1.3, and would receive its instructions from the on-board expert system or, in the future, artificial intelligence.

For the many types of standard and emergency situations required, collision or obstacle avoidance, a proper maneuver can be chosen and executed quickly and efficiently without excessive computational burdens that would otherwise lead to a tardy response.

D. SYSTEM SIMULATION METHOD

1. Dynamic Simulation Language (DSL)

DSL is a Fortran based simulation language for digital simulation of continuous systems. DSL uses a building -block approach to programming. Programs can be very simple or they can become extremely complex when all the functions of DSL are utilized. The user can enter Fortran statements in any order and DSL can sort and solve these equations effectively. The user can also include fortran subroutines and use the expansive I/O facility of DSL. One other key feature is the integration routine capability. The user has the choice of nine integration methods; four fixed-step, two variable-step and three variable-step, variable order methods. DSL was chosen primarily because it easily can solve differential equations and it contains many internal functions that normally would have to be programmed by the user.

DSL has four phases of program execution;

TRANSLATION

COMPILATION

SIMULATION

GRAPHICS

DSL translates all the DSL code into Fortran statements. Once the code is translated, it is then linked to the VS compiler and the code is compiled and stored as an executable file. Upon completion of the compilation phase, the simulation phase begins, and the system clock starts, and simulation continues until the system reaches its user specified finish time. The last phase of problem execution includes the graphic capability of DSL. Saved output data can be plotted or graphed using the graphics post-processor and the specific hardware supported.

2. Problem Simulation

As mentioned earlier, DSL uses a building-block approach to programming. The major blocks and general flow of program simulation are shown in Figures 5.3 and 5.4. To fully understand the simulation, and the controller design, and control action, a detailed breakdown and discussion is required.

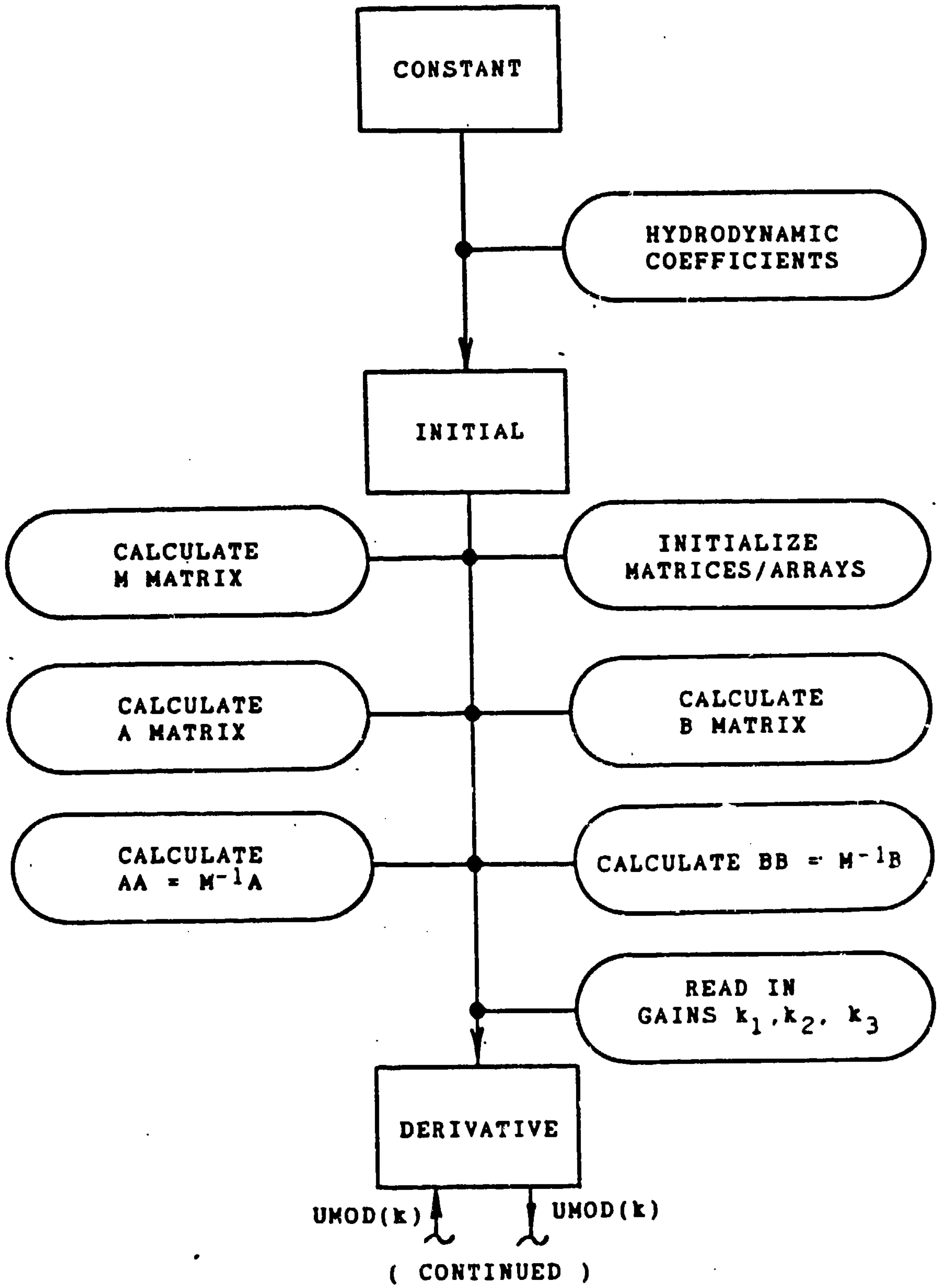

Figure 5.3 DSL Simulation Flow Diagram

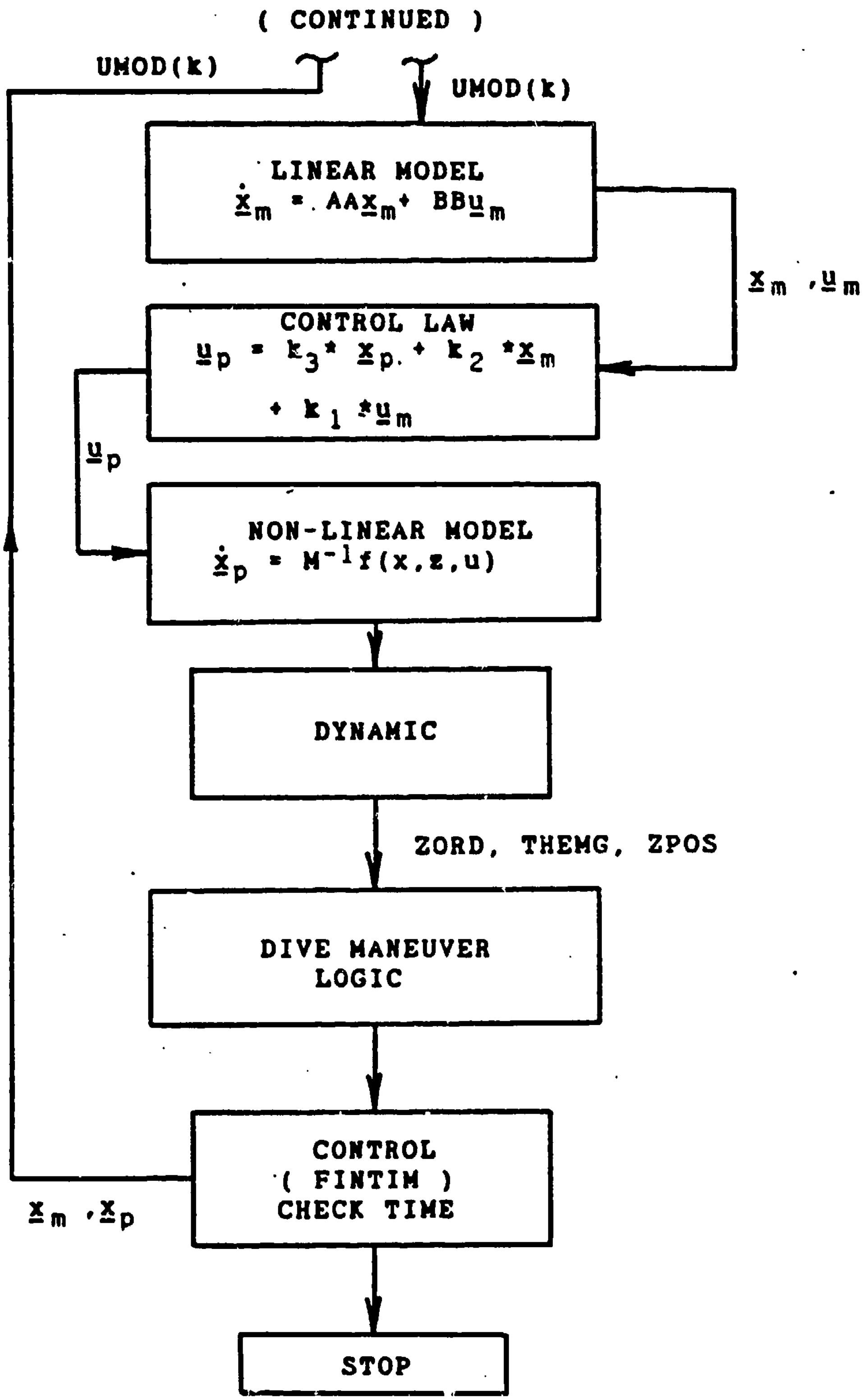

Figure 5.4 DSL Simulation Flow Diagram Continued

The first section of the simulation is the CONSTANT block. In this block all of the hydrodynamic coefficients and vehicle constants are read into the program.

The second section is the INITIAL block. In this section, all of the calculations not part of the integration routines and those needed in establishing parameters and initial conditions are performed. This is also where all variables are initialized. The following calculations occur in this section:

1. All matrices and arrays are initialized to zero.
2. The length and weight fractions for a four term gauss quadrature are initialized.
3. The breadth and height terms are read in. These terms will be used in the gauss quadrature integration for the crossflow drag terms.
4. The thrust is then calculated for the propulsion model.
5. The non-zero elements of mass matrix $\underline{M}$ are calculated.
6. The square mass matrix $\underline{M}$ has rank of six is then inverted using the IMSL routine LINV2F.
7. The non-zero elements of the $\underline{A}$ matrix are calculated. These elements are the coefficients of the first order terms in the Taylor series expansion about a specific operating point.
8. The non-zero elements of the $\underline{B}$ matrix are then calculated.
9. The next step is to multiply the inverse of the mass matrix to obtain the coefficients of the state equation, (5.1).
10. The last task for the initial section is to read in the computed feedforward and feedback gains from the program ETAT that are to be used in the autopilot control law.

The third block of the main program is called the DERIVATIVE section. Here, all the first order equations that must be integrated and solved are assembled. The DERIVATIVE section of this simulation is comprised of three major subsections, one for each major section in the simulation.

1. Linear reference model providing command generation.
2. Control law linking model and vehicle response to control surface actions.
3. Nonlinear model for simulating vehicle response.

The control vector $\underline{u}_m$ is the input to the linear model, generated from the maneuver logic contained within the DYNAMIC section. This section will be discussed later.

Once the control input is established, the derivative expressions of the linear reference model are formulated in terms of the matrices $\underline{AA}_m$ and $\underline{BB}_m$.

After the linear model derivatives are established, the model states $\underline{x}_m$ and model inputs $\underline{u}_m$ are passed to the Control Law. The Control Law (Equation (4.21)) represents, in software, the gains that would be incorporated in the vehicle.

The input vector $\underline{u}_p$ represents the inputs to the actual vehicle, in this case, defined by Equations (3.2).

The derivatives of the vehicle state variables are formed as the last part of the DERIVATIVE section in preparation for numerical integration using the fifth order variable step Runge-Kutta technique.

The model states and inputs, as well as the vehicle states and inputs, are saved for graphical representation and data output.

The fourth major block of the simulation is the DYNAMIC section. In the DYNAMIC section, the maneuver logic is programmed. This section is reserved for those computations that depend on time and are independent of the system responses. However, response dependent functions may also be included here as is the case with the establishment of the reference control commands generated by the maneuver logic.

The fifth section of the program is the CONTROL section. Before the command or input is sent to the derivative section, the system time clock is checked, and if "finish time" (fintime) in the CONTROL section is reached the program stops. If not, the system increments itself one time step and continues with the simulation.

Upon completion of the simulation a time history of all desired parameters and variables are saved in a data file. Plots and graphical output may then be generated.

3. Procedure Used

To perform a simulated run with a particular autopilot design and vehicle speed, an initial run with DSL was required to establish values for the AA, and BB matrices. These values were written on as output files (file Ft10F001 for AA and file Ft09F001 for BB). By separate run, program ETAT was used to read $\underline{AA}$ and $\underline{BB}$, and its own input file Ft05F001 for $\underline{Q}$ and $\underline{R}$ and to provide values for control gains $\underline{K}_1$, $\underline{K}_2$ and $\underline{K}_3$. The gain matrices, written on file Ft02F001 were then read by a final run using DSL for the controlled vehicle response simulation and results were provided on data file OUTP.

VI. RESULTS

A. GENERAL

This chapter describes the efforts and results of the design of a model following autopilot for an AUV. The controller designed is only a partial solution to the complete control over the six degrees of freedom of an AUV. However, the methodology developed in this study could be applied to design a full 30 state controller, 12 plant states, 12 model states, and 6 control states. The controller designed in this study is a 19 state controller using 12 plant states, 4 model states relating to the pitch plane, w_m, q_m, z_m, θ_m and the three control inputs for this plane, port and starboard bow plane angle and stern plane angle.

In Section D, the base-line controller is tested and the results show excellent depth control with excellent time history tracking. However, the control over pitch angle appeared loose and in Sections E and F attempts were made to gain tighter pitch control.

A test of controller robustness is its ability to operate over a range of vehicle speeds and changing parameters. In Section G the controller was tested at speeds of 3, 12 and 30 ft/sec, approximately 1.8, 7 and 17 knots, respectively, while baseline runs were at 6 ft/sec (3.F knots).

Included in this chapter is a discussion of the gains used, how they were derived and the effects on the gains by varying the control weight matrix R and the control error weight matrix Q.

B. RESULTS OF UNCONTROLLED MANEUVERING

The first simulation runs that were made early in this study were to test the non-linear model. One maneuver that was first tested was a turning maneuver. Because of this AUV's particular geometry (low aspect wing vice body of revolution), some unique behavioral characteristics are displayed as shown in Figures 6.1 and 6.2, not common to other forms of underwater vehicles. Of the most significant is when a rudder command is given the vehicle rolls out of the turn. While this is not uncommon for vehicles without a sail area, it is uncommon for a vehicle with a cruciform type stern to dive on a turn while the vehicle rolls out. Although this vehicle's dynamics are not representative of those common to submersibles, the behavior has been verified experimentally. The purpose of selection was based purely on the availability and thoroughness to which the vehicle dynamics were modeled, and that program validation was easily accomplished from the data available.

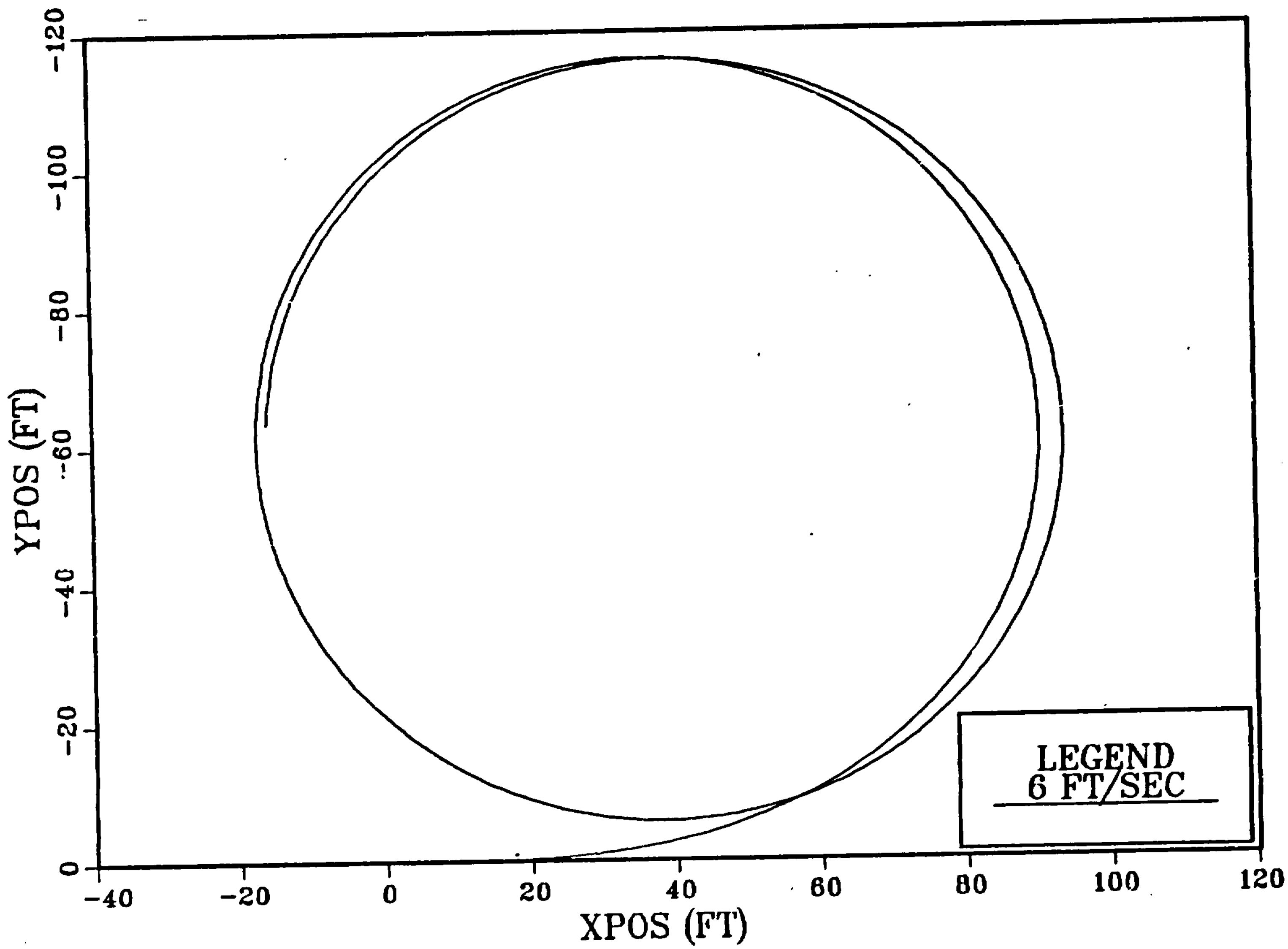

Figure 6.1 Uncontrolled Turn Maneuver for Non-Linear Model Horizontal Plane

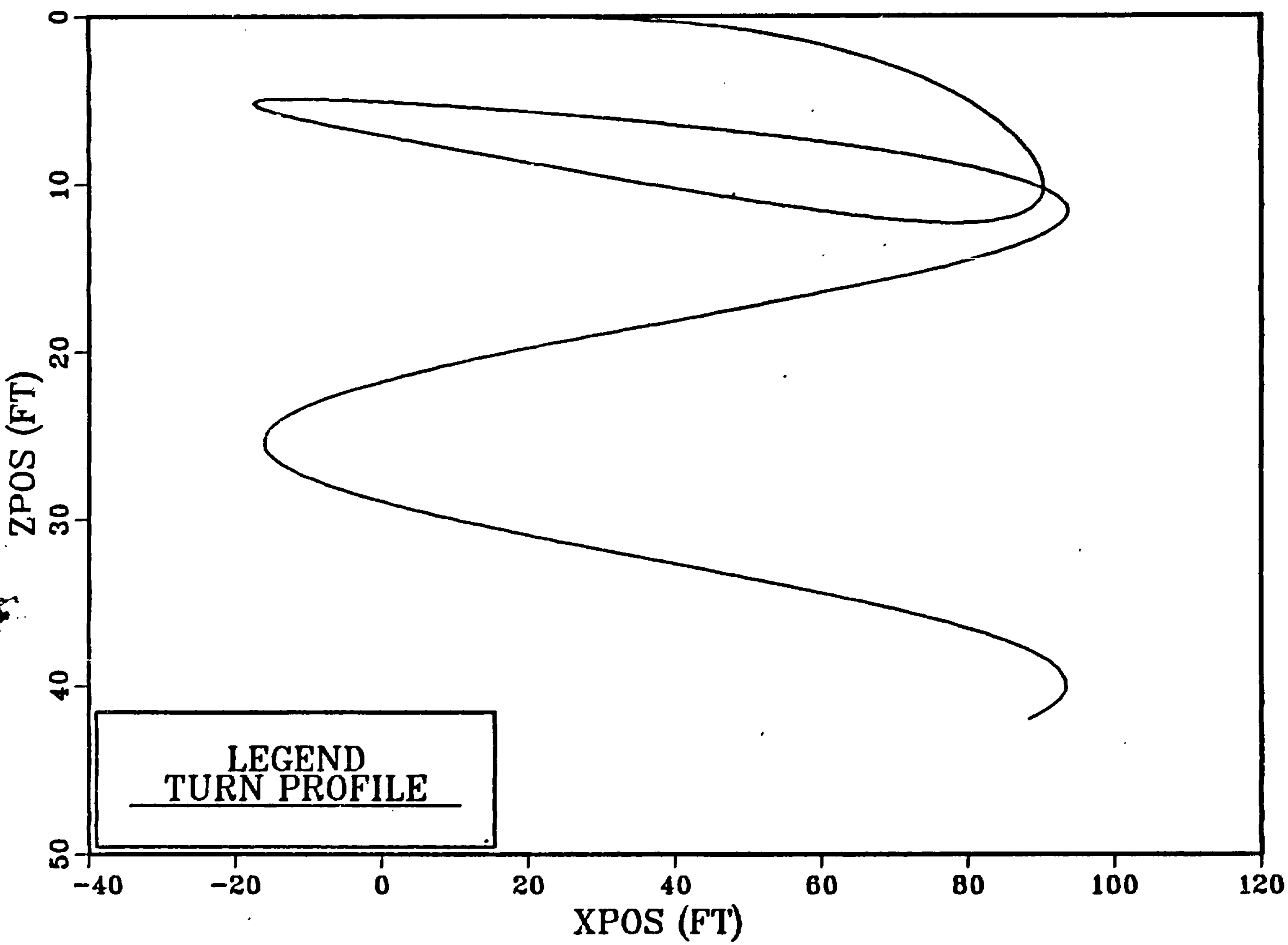

Figure 6.2 Uncontrolled Turn Maneuver for Non-Linear Model Dive Plane

C. DIVE PLANE MANEUVER AND PREDICTOR CONTROL

Once the non-linear model was validated the controller design process then began. The first simulation of this process consisted of only predictor control, no feedback was incorporated. The inputs generated by the dive maneuver logic for stern and bow plane input to the linear model were also fed into the non-linear vehicle dynamics.

This run, Figures 6.3 to 6.7, provided insight on the accuracy of the linearized version of the equations of motion. Figure 6.7 shows excellent pitch correlation between the model and vehicle.

Figures 6.4 and 6.5 both show that the vehicle never reaches its ordered depth of 100 ft. because the vehicle equations were linearized about a straight line trajectory at a constant speed, the linear model does not generate any speed loss and subsequently the AUV lags behind the linear model, a result that was clearly expected. The responsiveness of the vehicle is interesting considering the slow speed of 3.5 knots.

Examination of the maneuver shows that a limit of about 0.25 radians and 0.18 radians, respectively, was set by the command generation logic while the maximum pitch angle of 40 degrees was reached and maintained after 16 seconds. Also, while the vehicle pitch angle is returned to a small value, when the control surfaces are returned to zero, a small residual pitch angle is left. This is a small point that

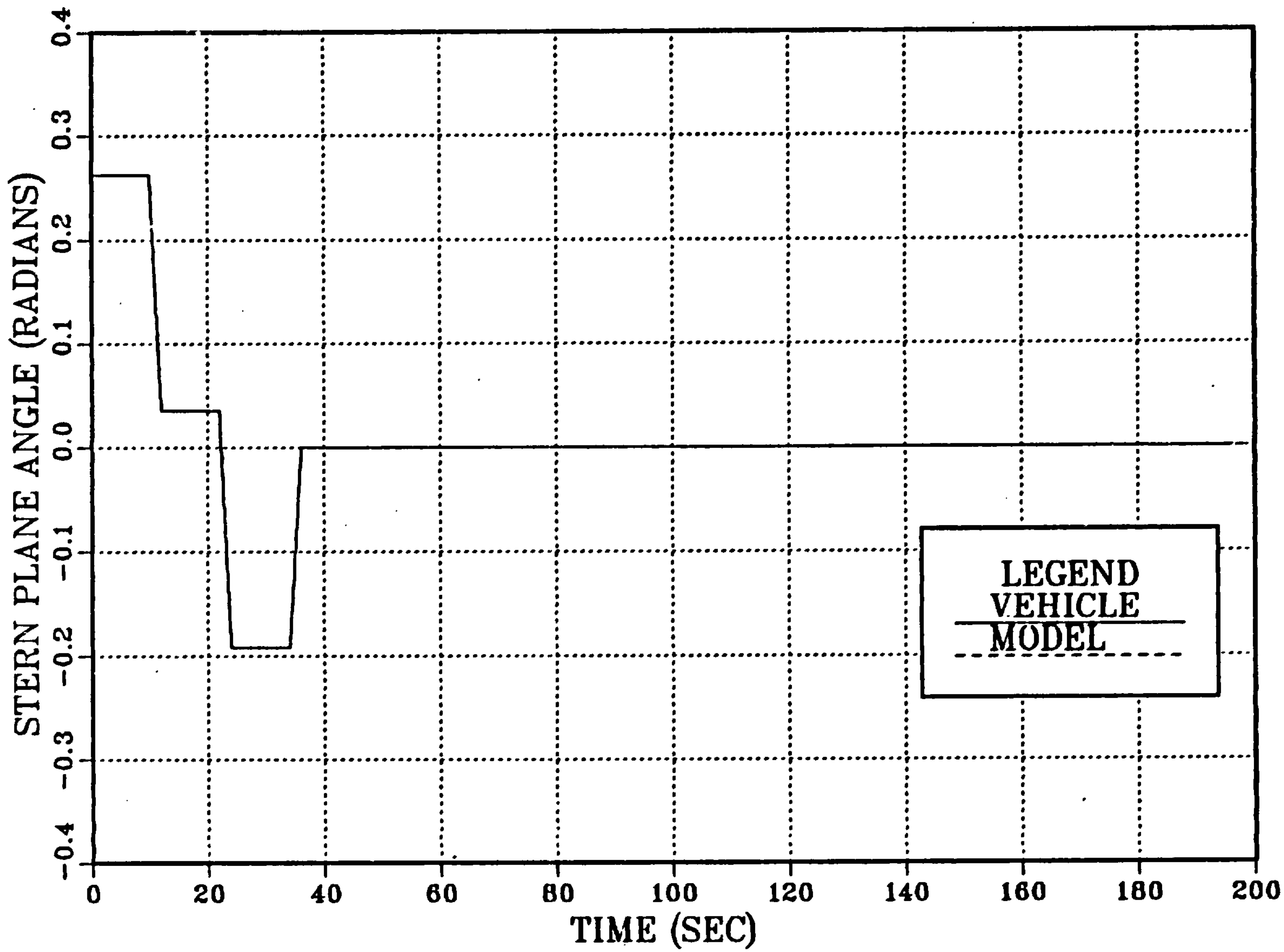

Figure 6.3 Dive Plane Maneuver and Predictor Control Stern Plane Angle--6 Ft/Sec

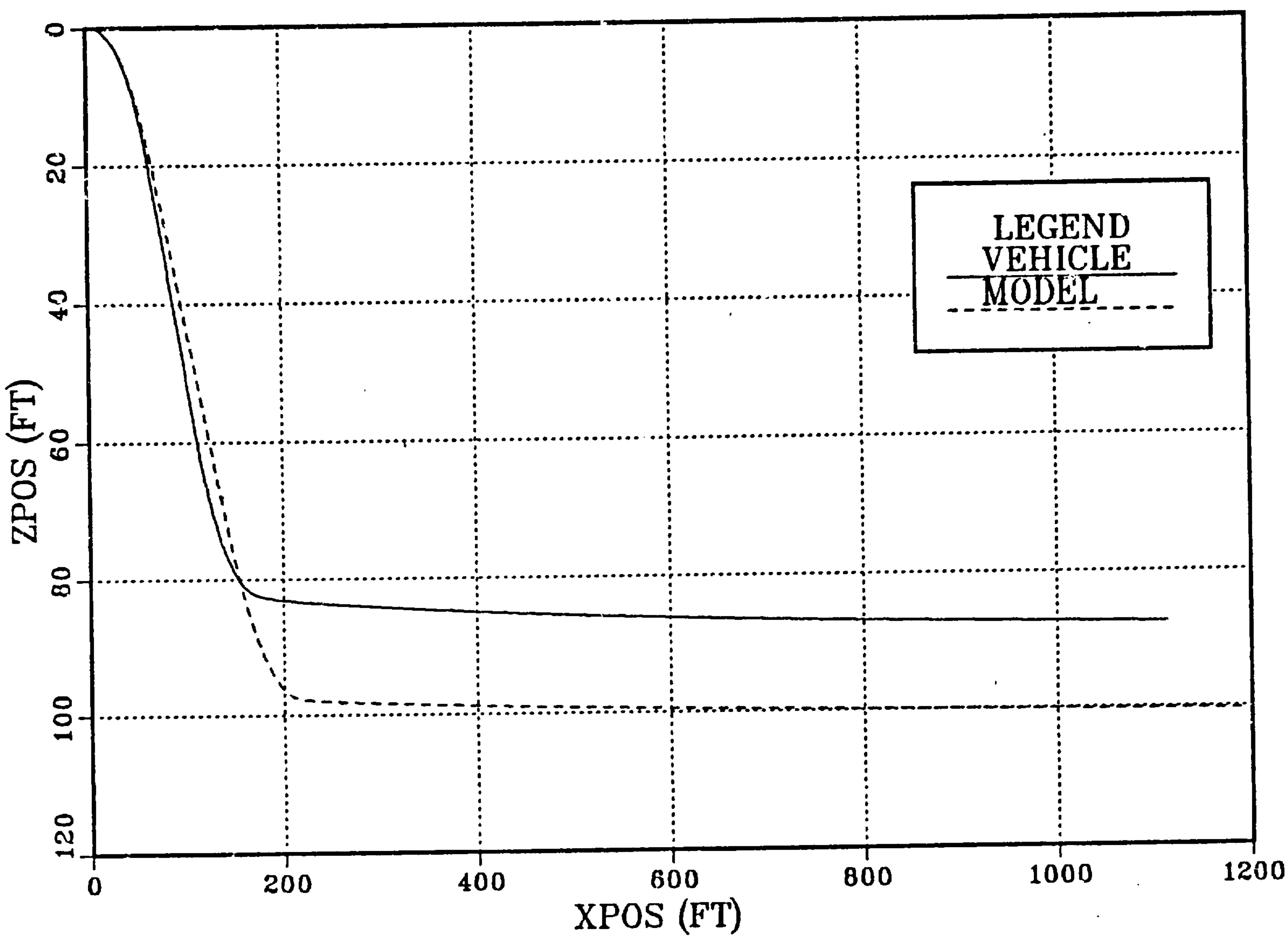

Figure 6.4 Dive Plane Maneuver and Predictor Control Dive Profile--6 Ft/Sec

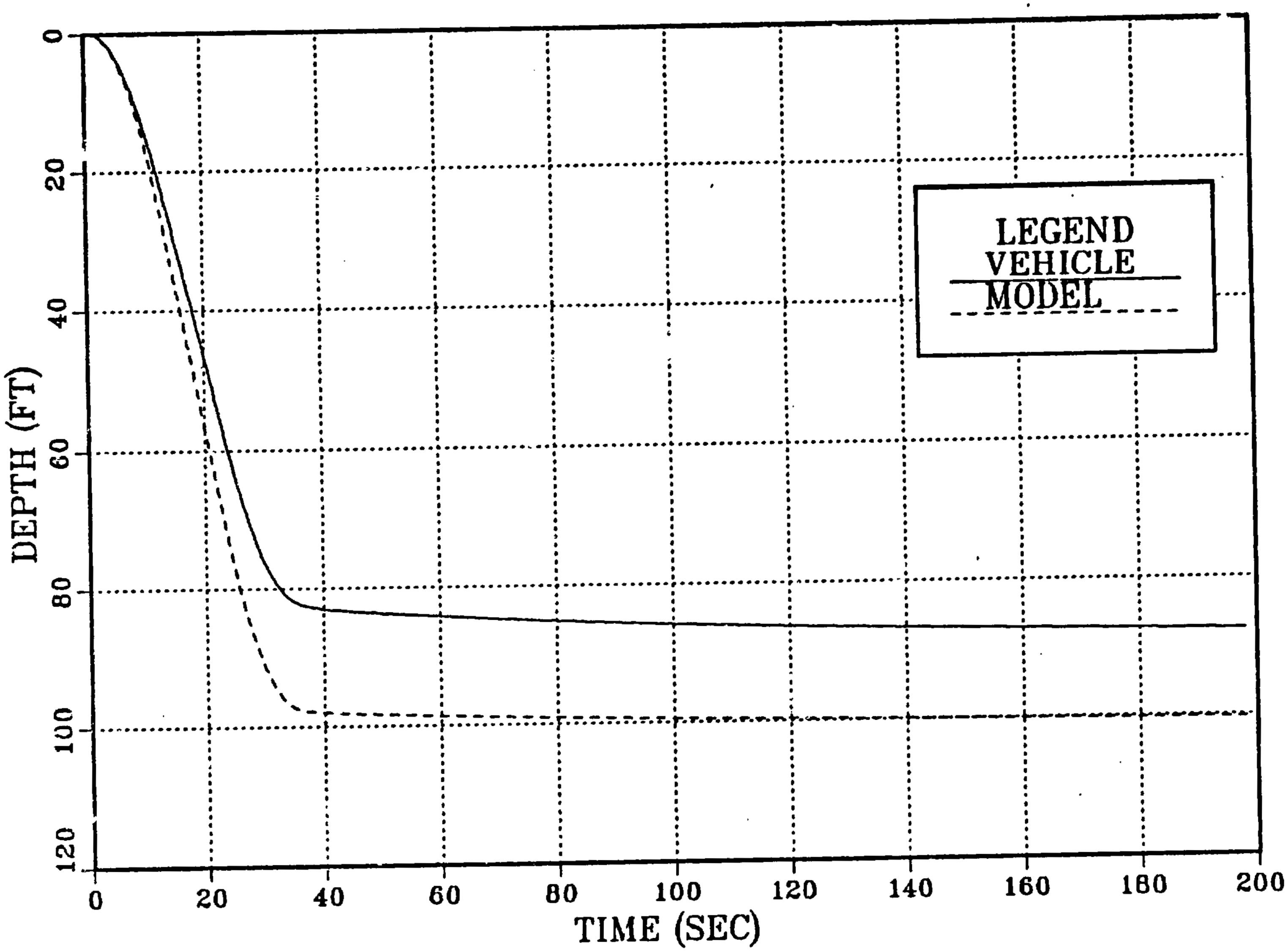

Figure 5.5 Dive Plane Maneuver and Predictor Control Depth Vs Time--6 Ft/Sec

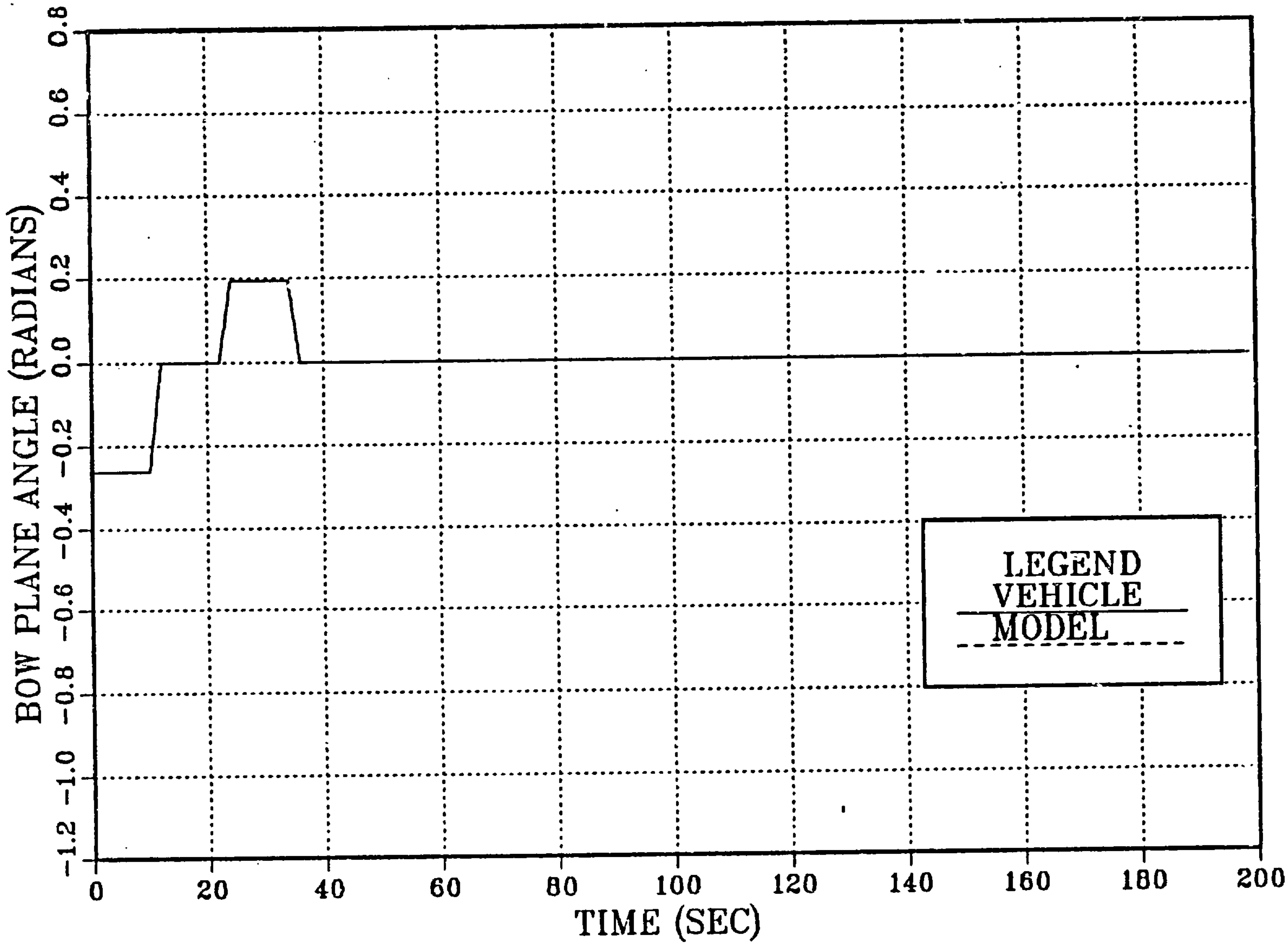

Figure 6.6 Dive Plane Maneuver and Predictor Control Bow Plane Angle--6 Ft/Sec

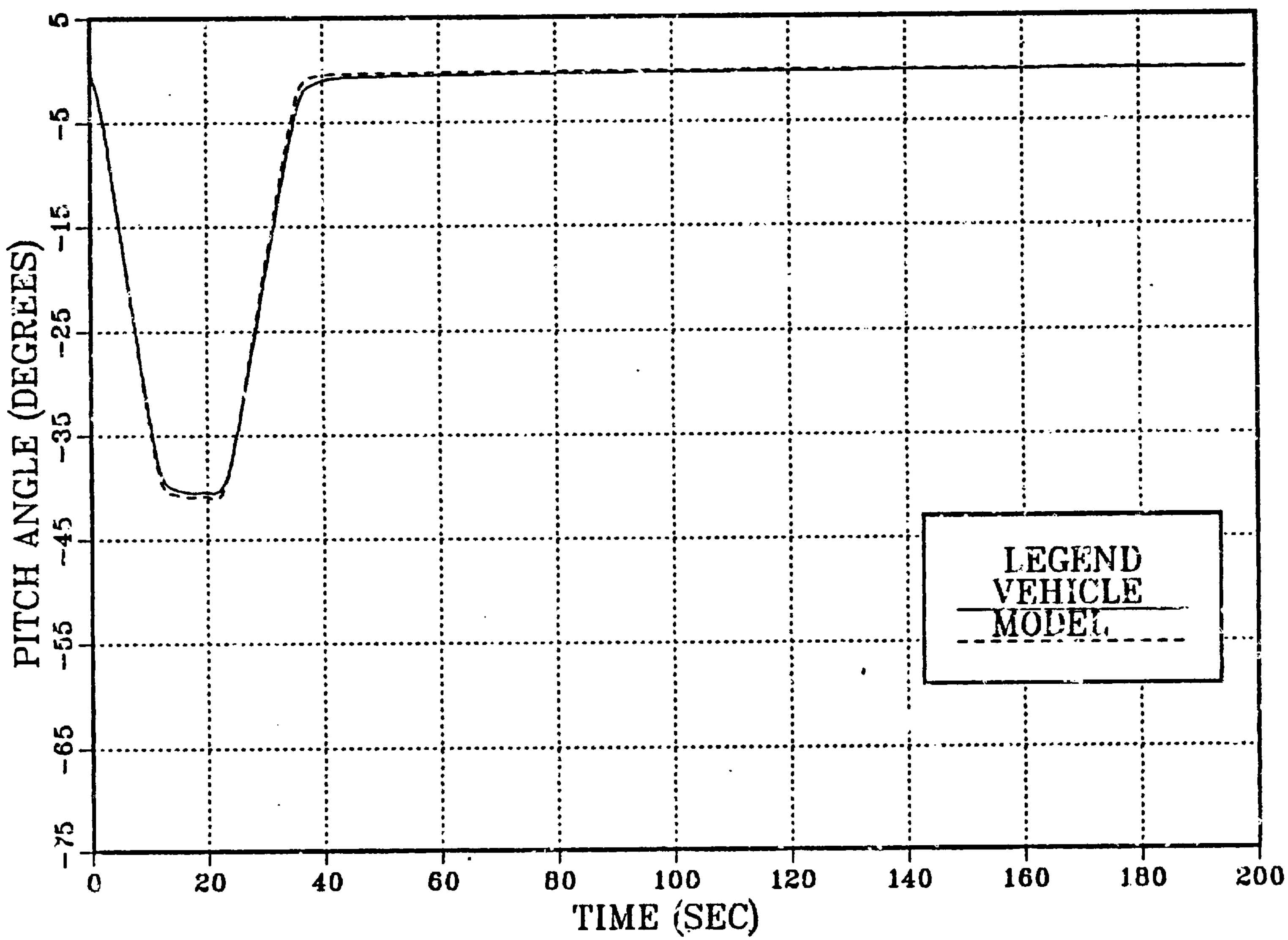

Figure 6.7 Dive Plane Maneuver and Predictor Control Pitch Angle--6 Ft/Sec

could be corrected by a refinement of the command generation logic.

What is of interest is the magnitude of the final depth error generated by the difference between linear and non-linear models. While the command generation logic drives the linear reference model to the targeted depth of 100 ft., the speed reduction in the AUV only provides a dive to 87 feet--clearly indicating the need for corrective control action.

D. EFFECT OF AUTOPILOT CONTROL--BASELINE CASE

Figures 6.8 to 6.12 clearly show the difference the controller makes in attaining the ordered depth. This was the first simulation run using the full 19 state controller for control in the heave/pitch plane. Although excellent depth control was achieved, the pitch control was considered a little loose resulting mainly from the mismatch between model and AUV speed. Figure 6.12 shows the overshoot of the vehicle pitch during the maneuver. The overshoot of the pitch also is the primary reason for the vehicle attaining ordered depth much more rapidly than the model as shown in Figure 6.9.

Other observations include, the majority of the control action comes from the stern plane which worked much harder than the bow planes. Figures 6.8 and 6.11 show the differences in control actions between the model and vehicle for stern and bow planes, respectively.

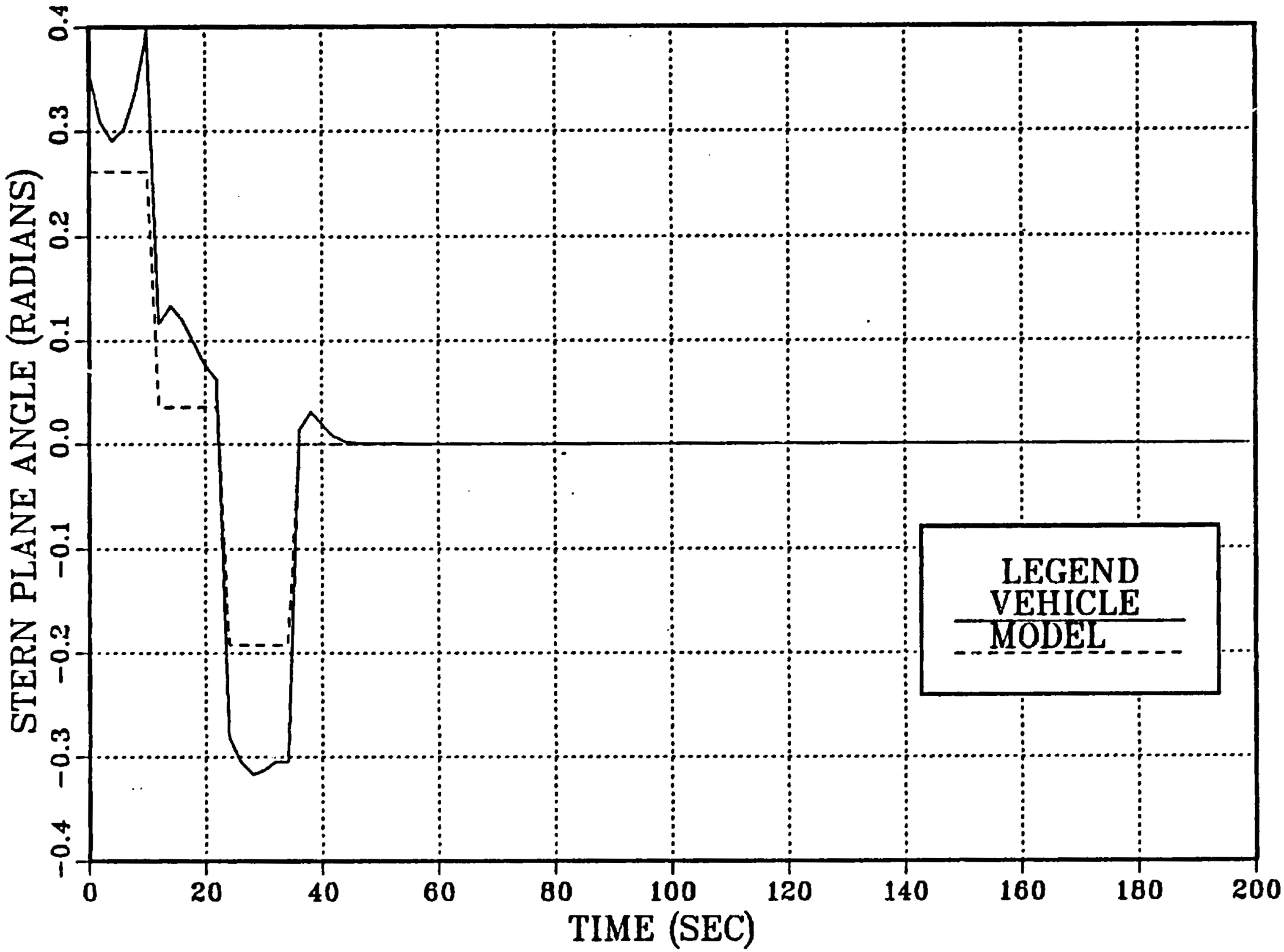

Figure 6.8 Autopilot Control--Baseline Stern Plane Angle--6 Ft/Sec

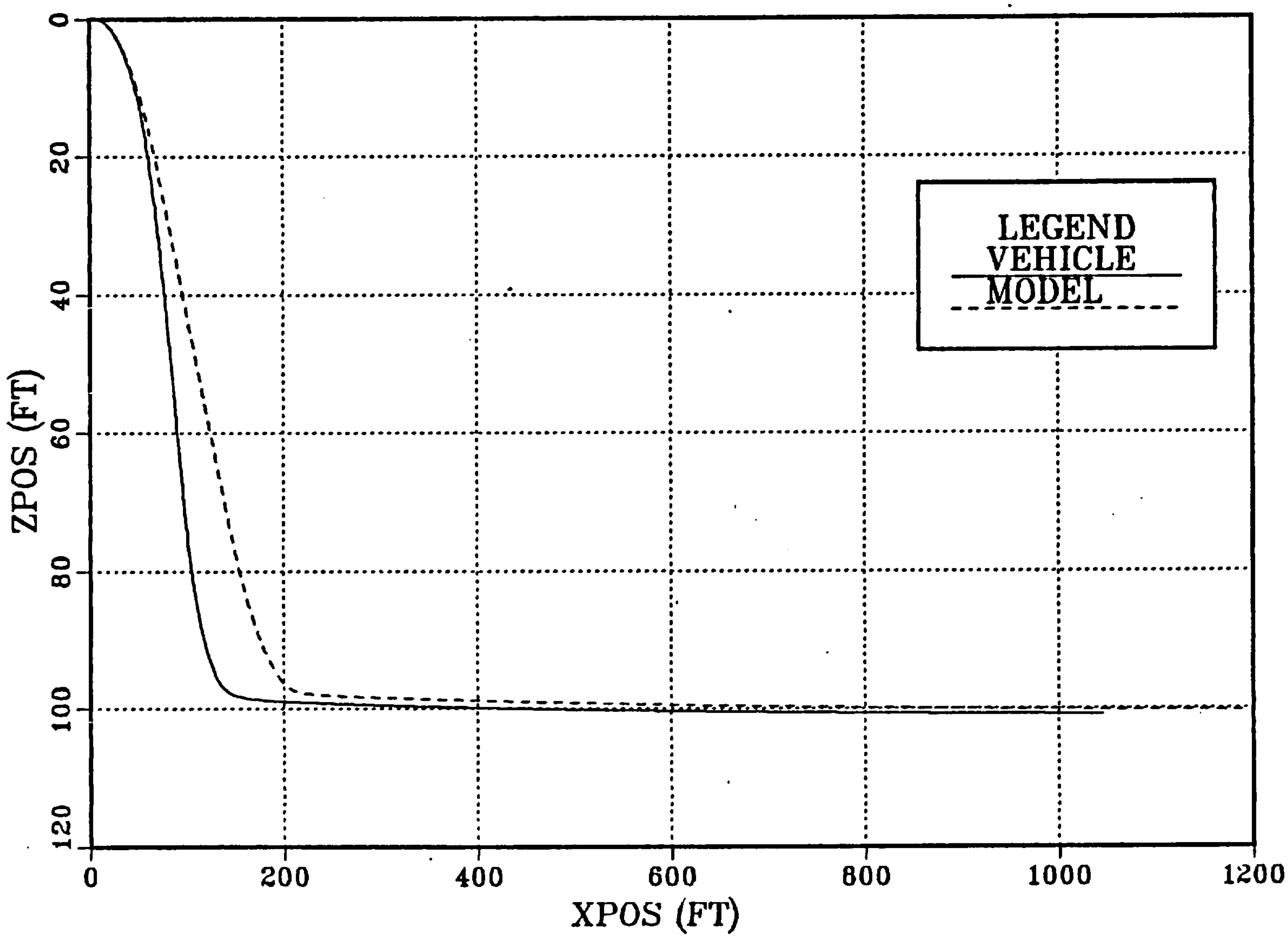

Figure 6.9 Autopilot Control--Baseline Dive Profile--6 Ft/Sec

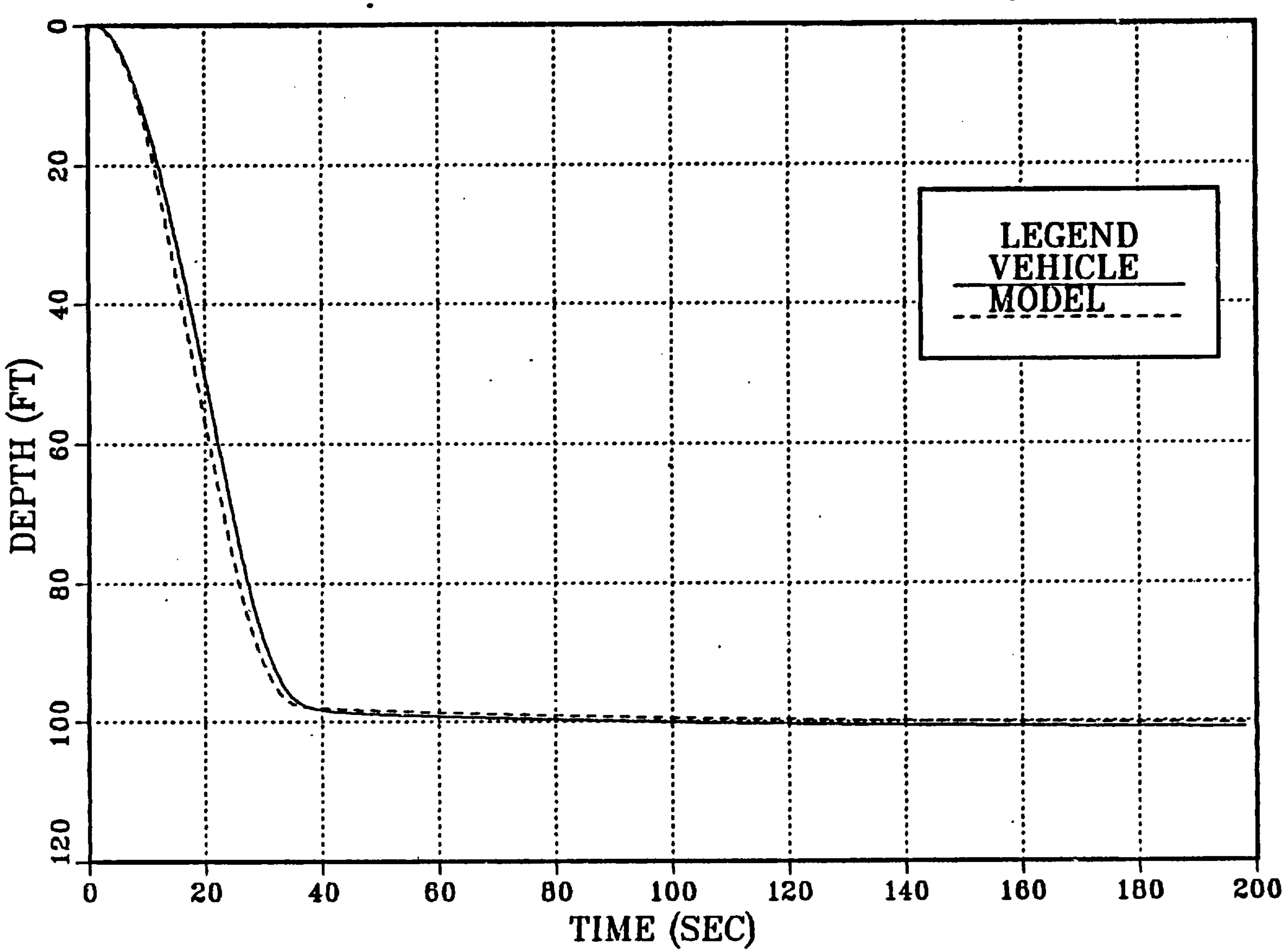

Figure 6.10 Autopilot Control--Baseline Depth Vs Time--6 Ft/Sec

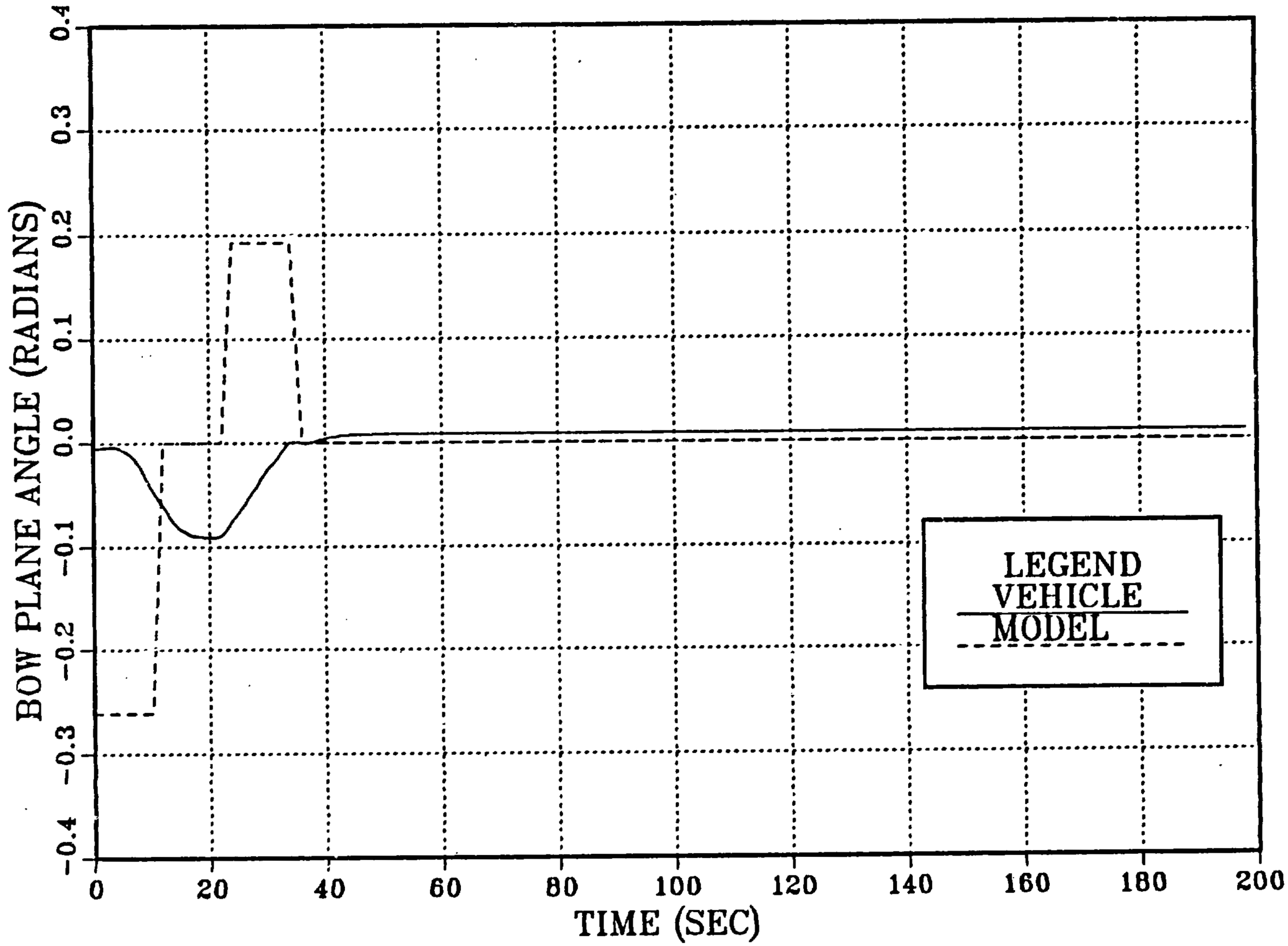

Figure 6.11 Autopilot Control--Baseline Bow Plane Angle--6 Ft/Sec

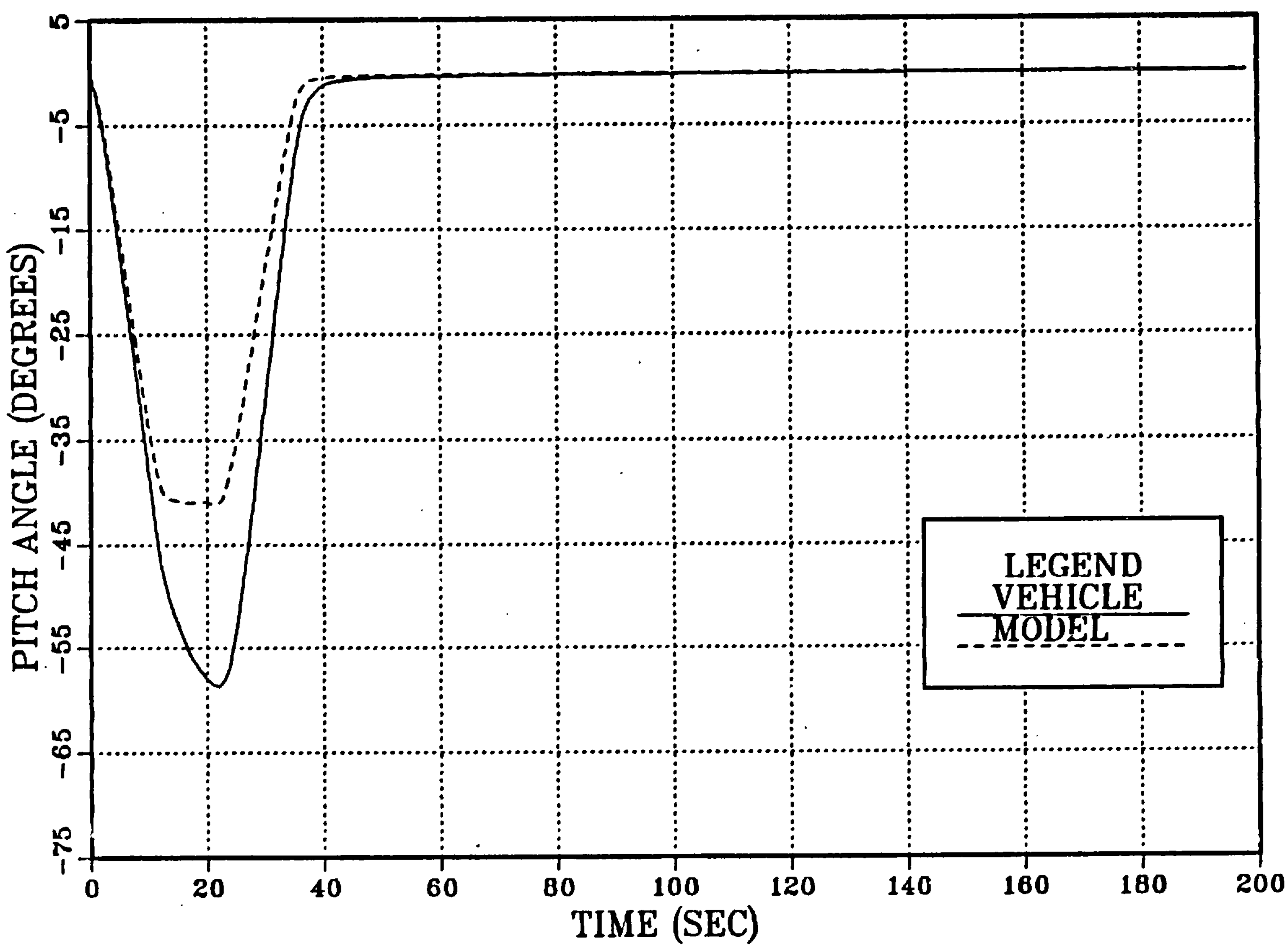

Figure 6.12 Autopilot Control--Baseline Pitch Angle--6 Ft/Sec

Because of the disparity in control efforts an attempt to equalize the relative amount of control actions and more closely following the model was made. As discussed in Chapter IV, the control weight matrix $\underline{R}$ (Table 1) was initially set up to penalize the rudder, rpm and buoyancy inputs, so that the primary control actions would be from the bow and stern planes as it would be for a dive maneuver. In this first run the weights were equal and the resulting control gains (Table 3) for the stern plane were much higher than for the bow planes. An attempt was made at sharing the control actions where weights of the $\underline{R}$ matrix were adjusted to penalize the stern plane and put more control effort in the bow planes. This resulted in a significant loss in the stern plane gain much less that one and only a very slight rise in the bow plane gain. Although the resulting simulation showed more bow plane action it did not follow the model well and the stern plane became more active by the feedback action. This increased activity in the bow and stern planes resulted in very significant speed loss and excessive plane use was considered unacceptable.

Upon further study of the vehicle and its dynamics, the reason for the inconsistency in control actions is that the model maneuver treats bow and stern planes almost equally in their effect but in fact the force and moment generated by the stern plane is an order of magnitude more significant than that of the bow planes. Therefore, in future maneuver

TABLE 1

TABLE OF Q WEIGHTS--BASELINE

Pitch Rate Error	
Q(5,5)	0.6
Q((5,14)	-0.6
Q(14,5)	-0.6
Q(14,14)	0.6
Pitch Angle Errors	
Q(11,11)	2.5
Q(11,16)	-2.5
Q(16,11)	-2.5
Q(16,16)	2.5
Heave Rate Error	
Q(3,3)	1.0
Q(3,13)	-1.0
Q(13,3)	-1.0
Q(13,13)	1.0
Heave Positional Error	
Q(9,9)	60.0
Q(9,15)	-60.0
Q(15,9)	-60.0
Q(15,15)	60.0

TABLE 2

TABLE OF R WEIGHTS--BASELINE

Rudder	R(1,1)	1.0×10^4
Starboard Bow Plane	R(2,2)	1.0×10^3
Port Bow Plane	R(3,3)	1.0×10^3
Stern Plane	R(4,4)	1.0×10^3
RPM	R(5,5)	1.0×10^6
Buoyancy	R(6,6)	1.0×10^6

TABLE 3

CONTROL GAINS

0.0000000000E+00	0.7258932550E-10	-0.2575687868E-05	
0.4954081951E-08	0.3019079092E-04	0.2760571526E-08	
0.0000000000E+00	0.0000000000E+00	-0.2752279831E-05	RUDDER
0.4644012840E-09	0.4422721241E-04	0.0000000000E+00	
0.2595669042E-05	-0.3025079273E-04	0.2764935447E-05	
-0.4438610978E-04	-0.1165944258E-05	-0.1165974829E-05	
-0.5901765458E-07			
0.0000000000E+00	0.1114619576E-05	-0.4422518793E-01	
0.7605860458E-04	0.4668891458E+00	0.4168961983E-04	
0.0000000000E+00	0.0000000000E+00	-0.4510255945E-01	PORT
0.7216705136E-05	0.6929793740E+00	0.0000000000E+00	BOW PLANE
0.4454871633E-01	-0.4676730097E+00	0.4530064631E-01	
-0.6952859507E+00	-0.1746945879E-01	-0.1747022491E-01	
-0.9987243305E-02			
0.0000000000E+00	0.1117984364E-05	-0.4426664311E-01	
0.7628508878E-04	0.4682061530E+00	0.4182848952E-04	
0.0000000000E+00	0.0000000000E+00	-0.4518982016E-01	STBD
0.7236512897E-05	0.6947686167E+00	0.0000000000E+00	BOW PLANE
0.4459061566E-01	-0.4689957818E+00	0.4538842427E-01	
-0.6970854515E+00	-0.1752998162E-01	-0.1753074334E-01	
-0.9759326349E-02			
0.0000000000E+00	-0.9857068418E-05	0.6050596839E-01	
-0.6689594082E-03	-0.3859172641E+01	-0.4217829036E-03	
0.0000000000E+00	0.0000000000E+00	0.2348149885E+00	STERN
-0.5682397825E-04	-0.5158026393E+01	0.0000000000E+00	PLANE
-0.6106134440E-01	0.3880347969E+01	-0.2363129821E+00	
0.5192655967E+01	0.1856029149E+00	0.1855811163E+00	
-0.1004805424E+01			
0.0000000000E+00	-0.7210798044E-12	0.1382443897E-07	
-0.4904055922E-10	-0.2899780644E-06	-0.2934508269E-10	
0.0000000000E+00	0.0000000000E+00	0.2184067563E-07	RPM
-0.4355770203E-11	-0.4049075188E-06	0.0000000000E+00	
-0.1393062766E-07	0.2911221362E-06	-0.2195747233E-07	
0.4070361389E-06	0.1269293021E-07	0.1269214802E-07	
-0.4242388239E-07			
0.0000000000E+00	0.1283218668E-11	-0.1916238257E-06	
0.8911601991E-10	0.6522729441E-06	0.2538939491E-10	
0.0000000000E+00	0.0000000000E+00	-0.1217387526E-06	BUOYANCY
0.1128610385E-10	0.1210698470E-05	0.0000000000E+00	
0.1929779261E-06	-0.6471117447E-06	0.1220747899E-06	
-0.1207295075E-05	-0.6867469453E-08	-0.6880516105E-08	
-0.4768694409E-06			

model generation it should be noted that bow plane forces and moments are more subtle and should be used for fine depth control rather than for major depth excursions. Considering the speed mismatch, the overall control was considered acceptable.

E. EFFECT OF TIGHTER PITCH CONTROL WEIGHTING

Due to the unique dynamics of the vehicle it was decided to leave the control weights the same in the R matrix, as it was in the first run, with the understanding that the model maneuver algorithm perhaps wasn't as well suited for this vehicle as it could have been.

With the R matrix fixed, with equal weights on the bow and stern planes, it was decided to adjust the weights in the Q matrix to try to gain better control over the pitch, and to increase the pitch error gains. The weights that were adjusted were those that related pitch errors, elements Q(11,11), Q(11,16), Q(16,11), Q(16,16).

When these elements were increased by a factor of 1000 the pitch error gains increased and a tighter control over pitch was achieved as shown in Figures 6.13 to 6.17.

Comparing Figures 6.7 and 6.17 shows the tighter control gained over the pitch. With the tighter control gained in pitch a slight degradation in depth control was observed. Figures 6.14 and 6.15 show a small overshoot in ordered depth for the vehicle indicating the loosening of the depth control modes.

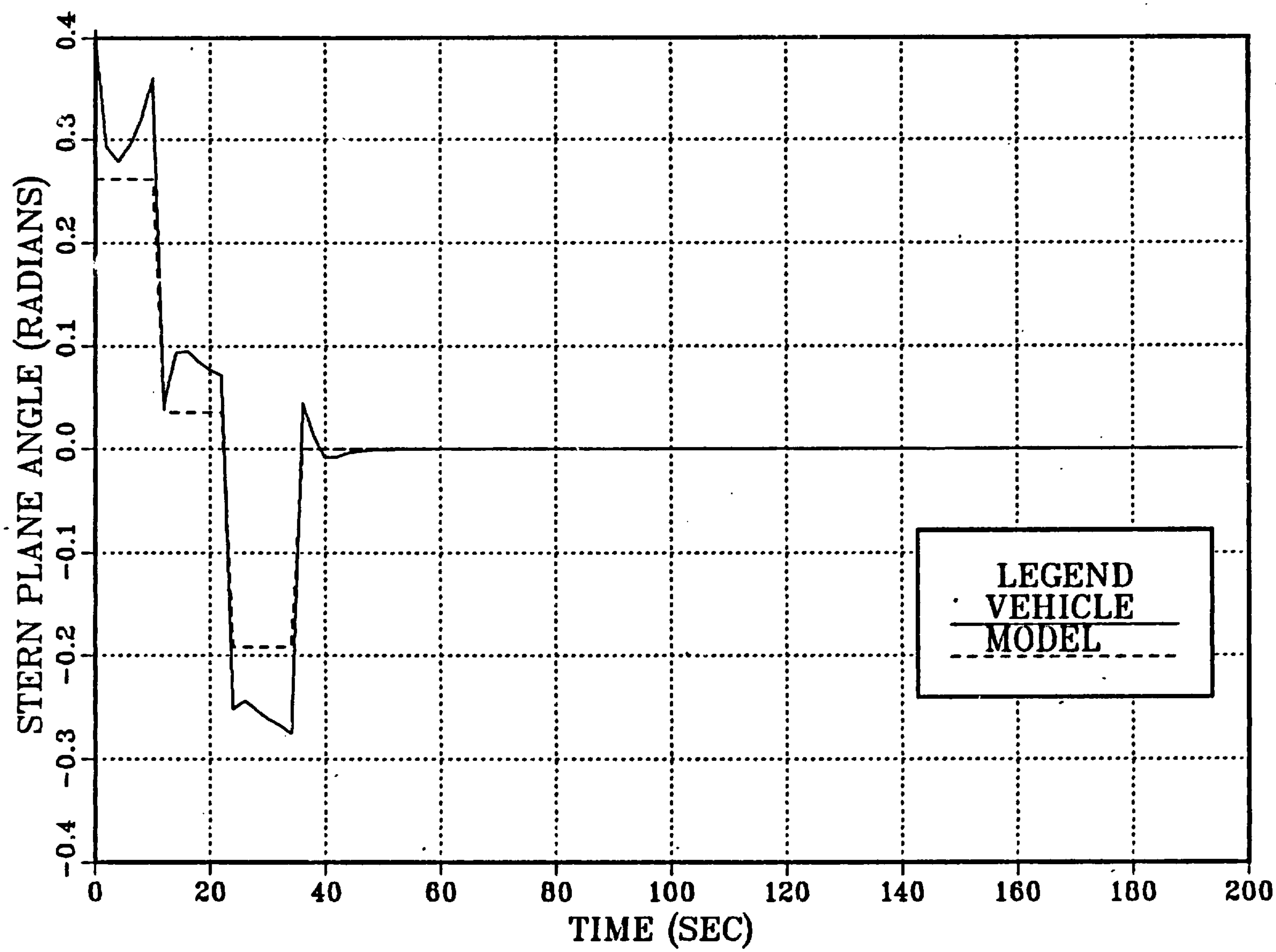

Figure 6.13 Tighter Pitch Control Weighting Stern Plane Angle

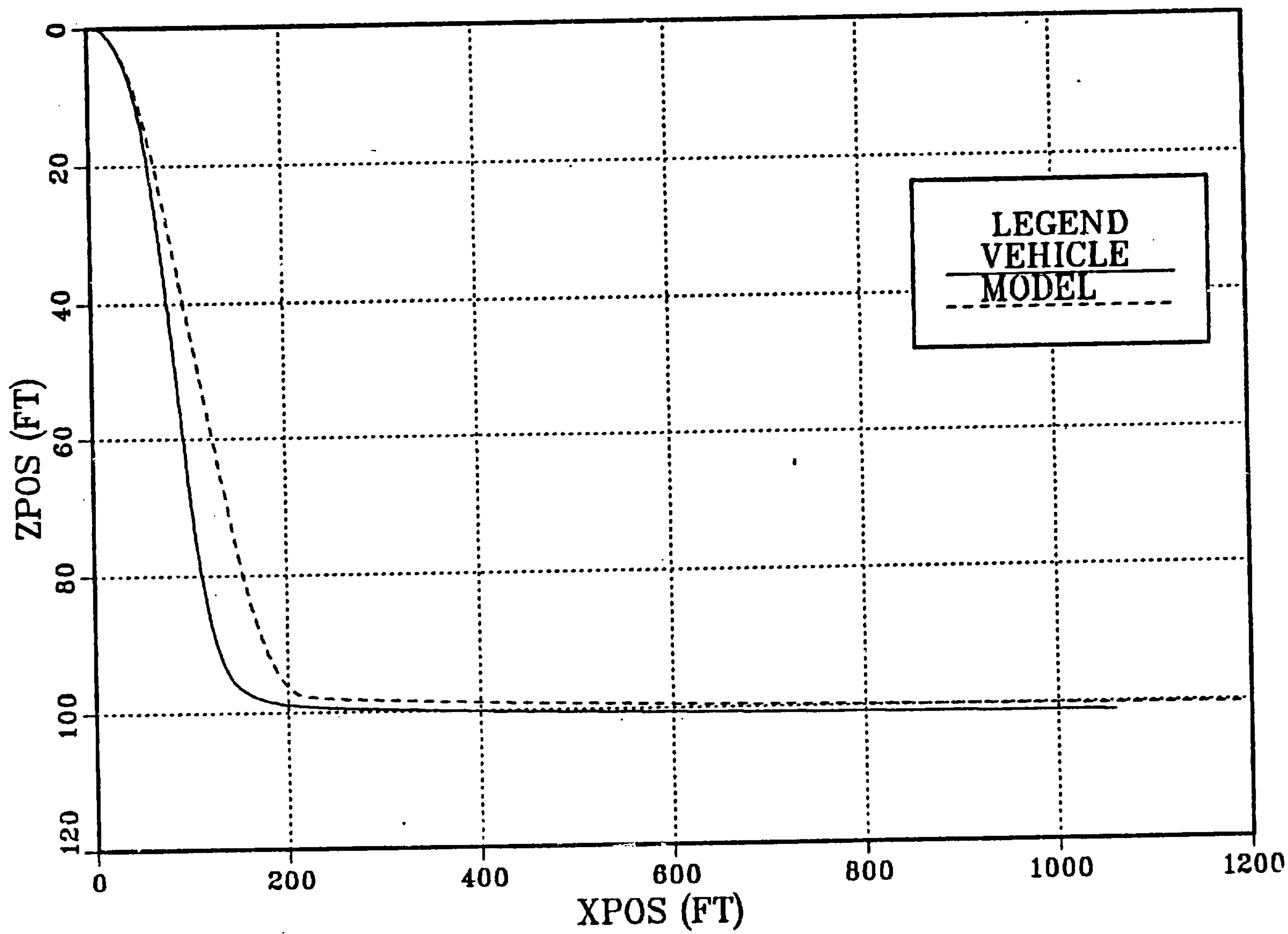

Figure 6.14 Tighter Pitch Control Weighting Dive Profile

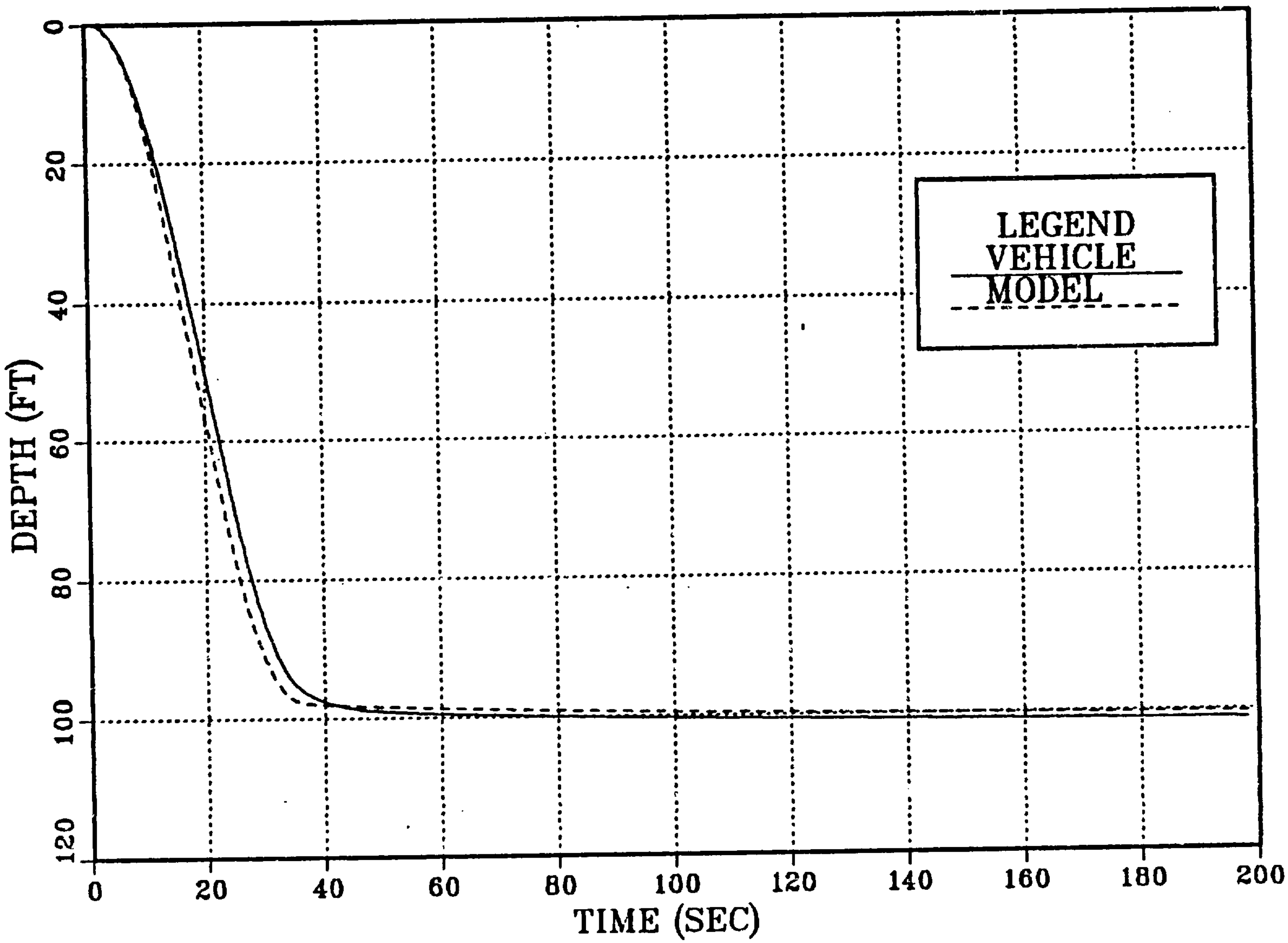

Figure 6.15 Tighter Pitch Control Weighting Depth Vs Time

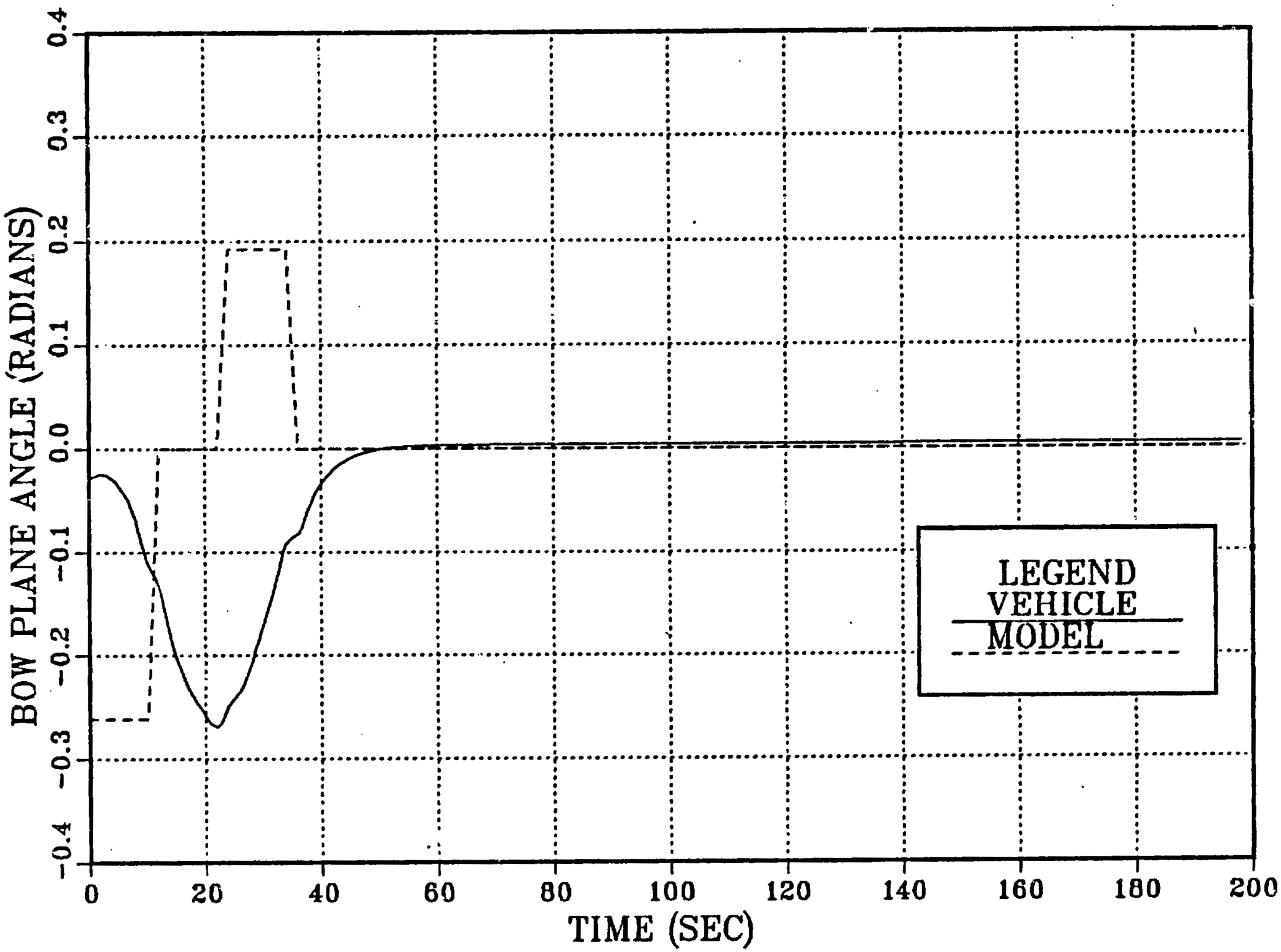

Figure 6.16 Tighter Pitch Control Weighting Bow Plane Angle

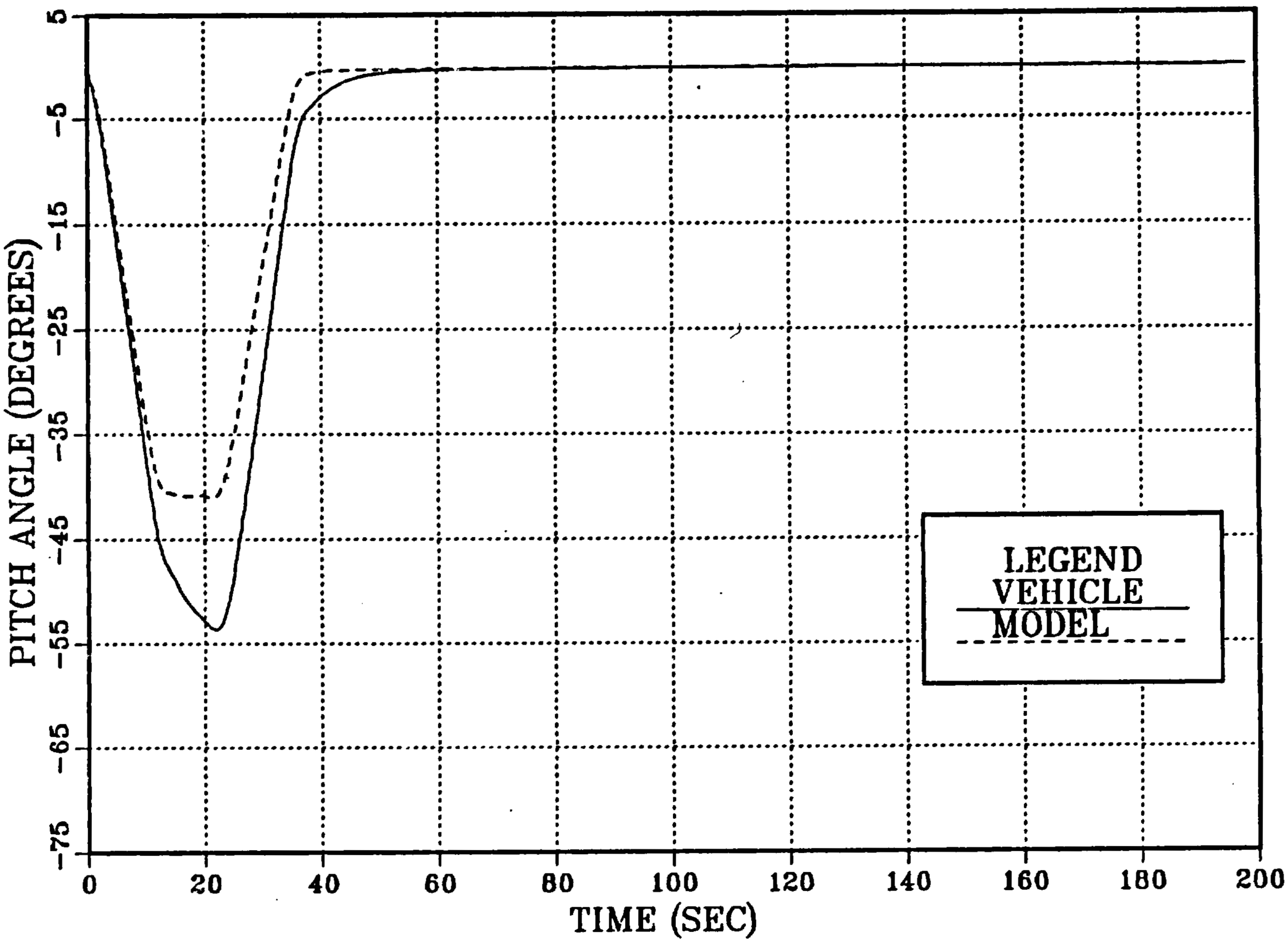

Figure 6.17 Tighter Pitch Control Weighting Pitch Angle

F. FURTHER PITCH CONTROL WEIGHTING

The Q matrix pitch elements were further increased by a factor of 10 to observe the correlation between depth and pitch control. Again Figures 6.18 to 6.22 show a sluggish response in depth control while gaining a much tighter control over pitch. However, in this case the command for the bow planes have exceeded their physical limits and are commanded to an unreasonable amount as shown in Figure 6.21.

As the linear controller can command a control action greater than the physical limits of the vehicle, when poor weights are selected, logic was added to the DSL code to limit the plane action to plus or minus .6 radians on the bow and stern planes.

Although the increased weights in the Q matrix gave a better pitch response, its effects on tracking control were undesirable. For this reason, and for all subsequent numerical experiments, it was decided to use the gains originally calculated in the baseline run and the original Q matrix weights.

G. EFFECT OF SPEED MISMATCH MODEL/VEHICLE

The major issue of control robustness relates to the seriousness of speed mismatch between model and AUV. So the next task was to test this controller over a range of vehicle speeds, 3, 12 and 30 feet/sec.

Using the controller and model based on a speed of 6 ft/sec, the simulation was run using a vehicle speed of 12

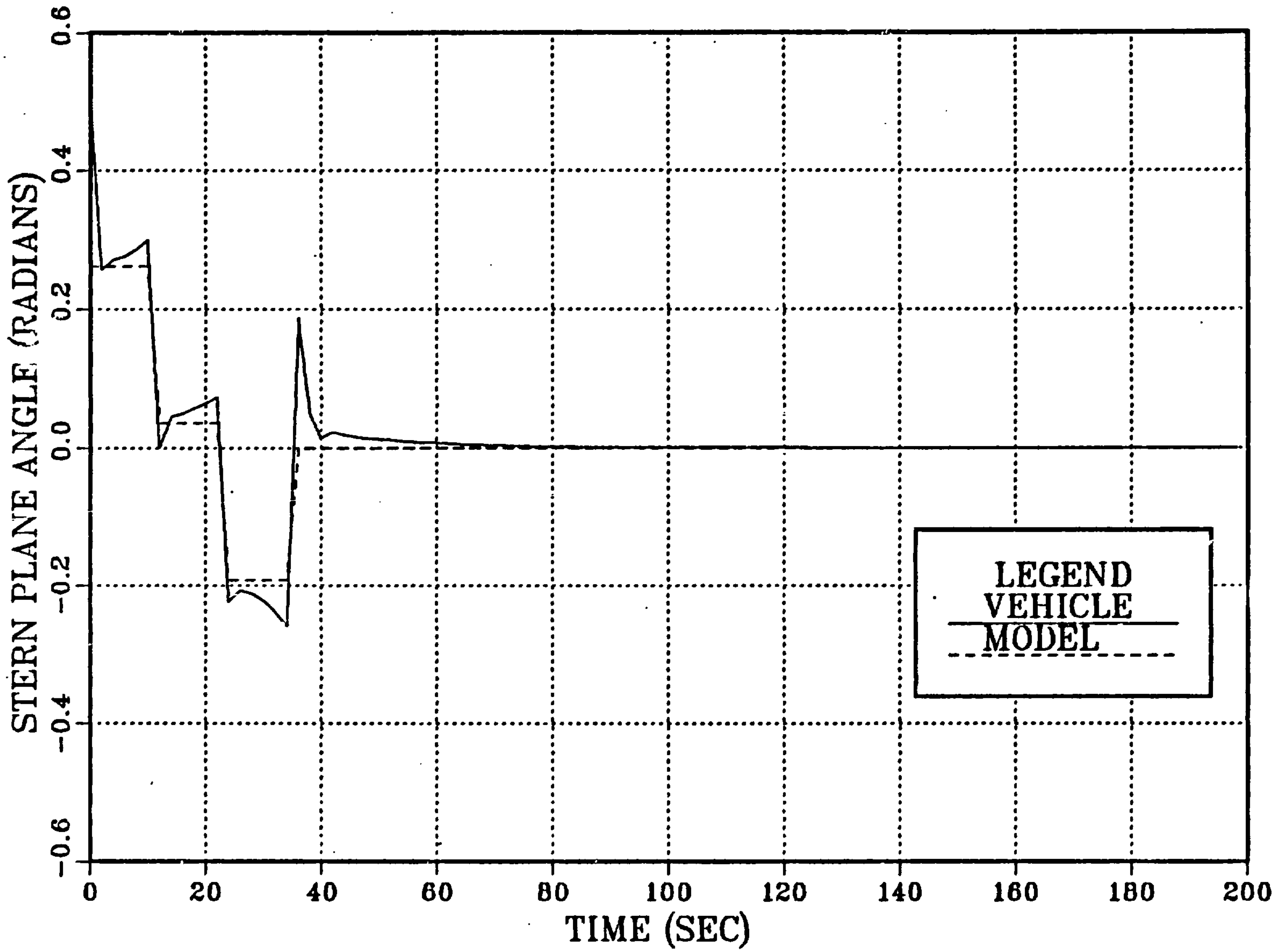

Figure 6.18 Further Pitch Control Weighting Stern Plane Angle

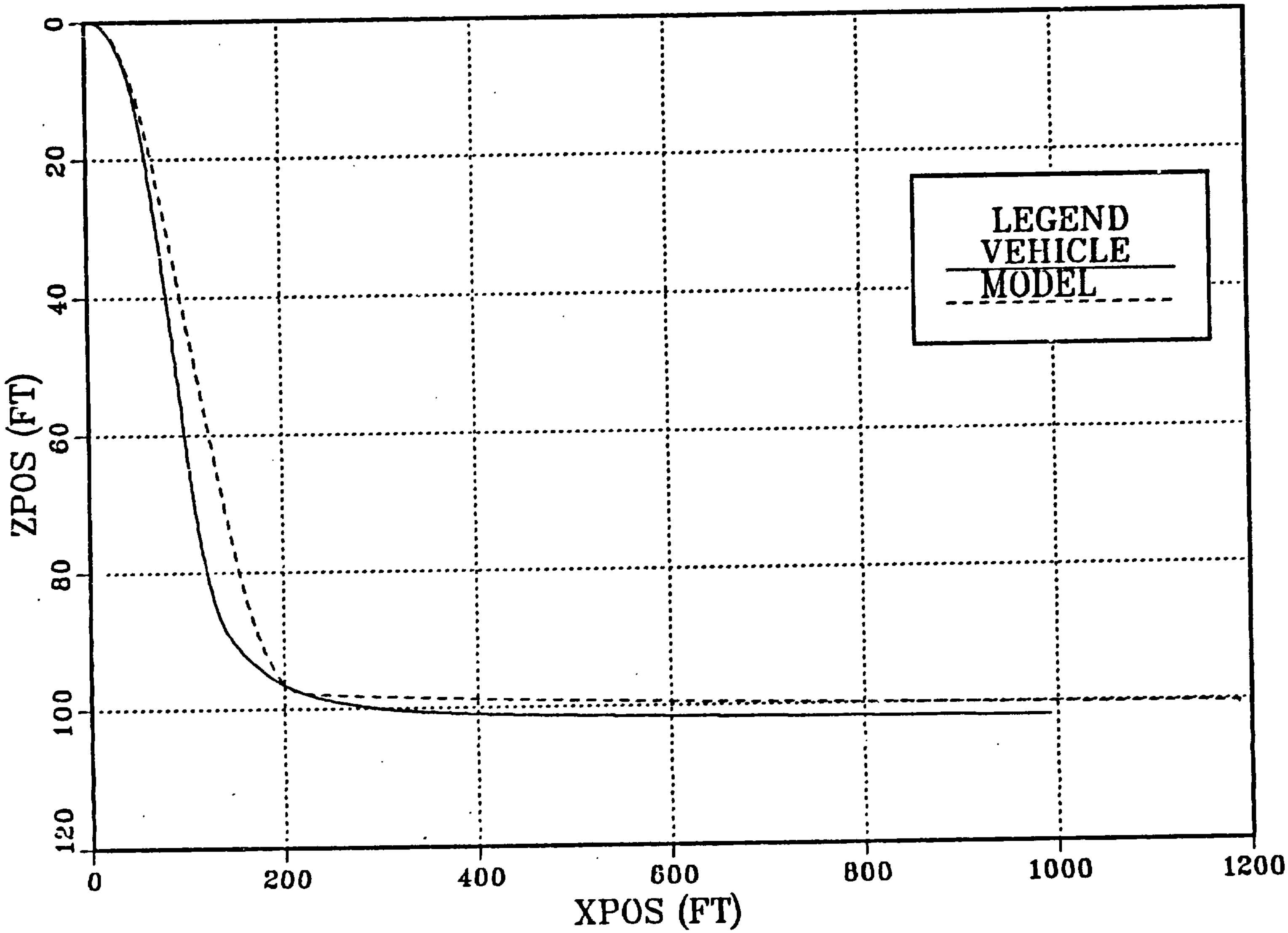

Figure 6.19 Further Pitch Control Weighting Dive Profile

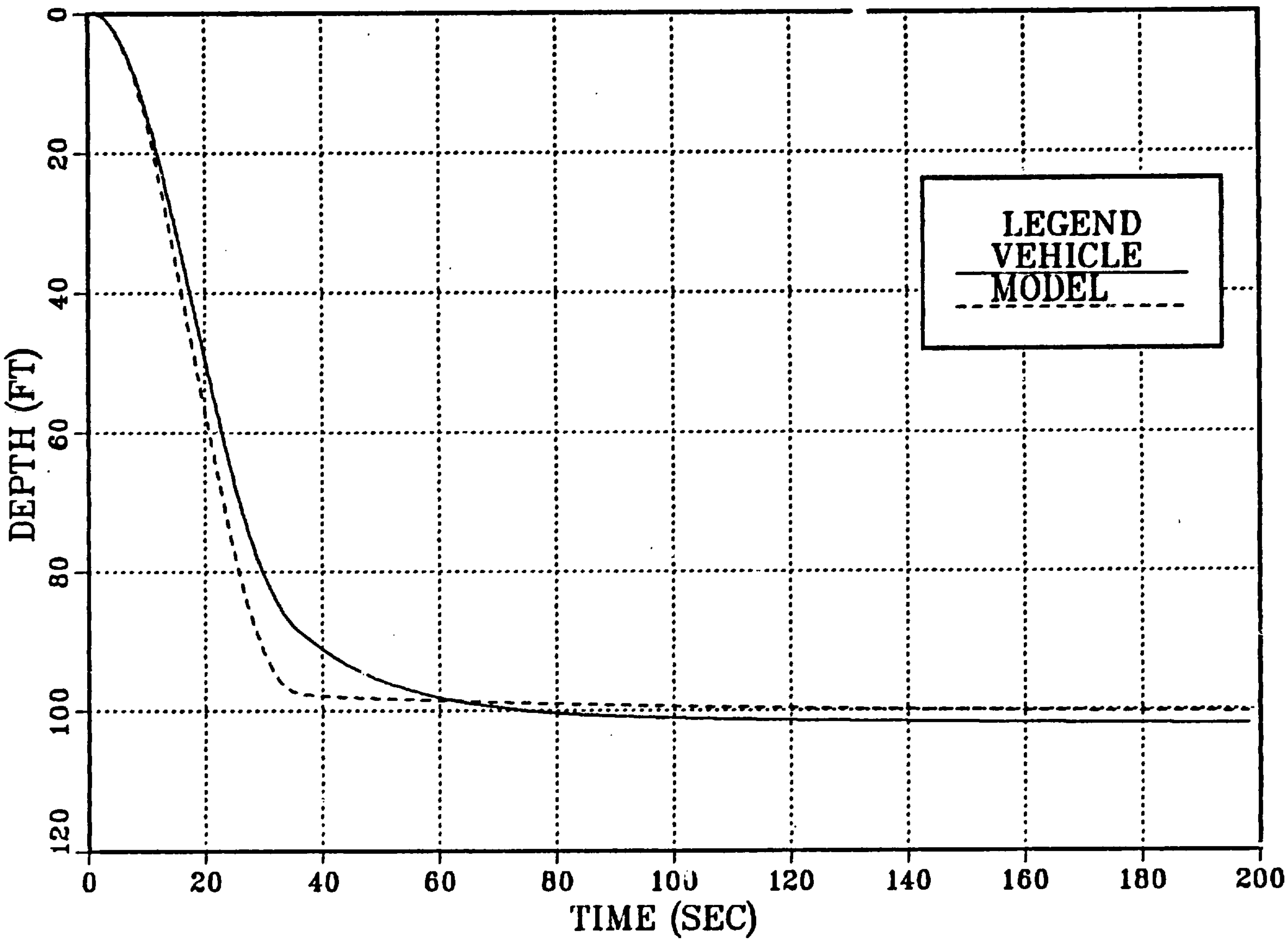

Figure 6.20 Further Pitch Control Weighting Depth Vs Time

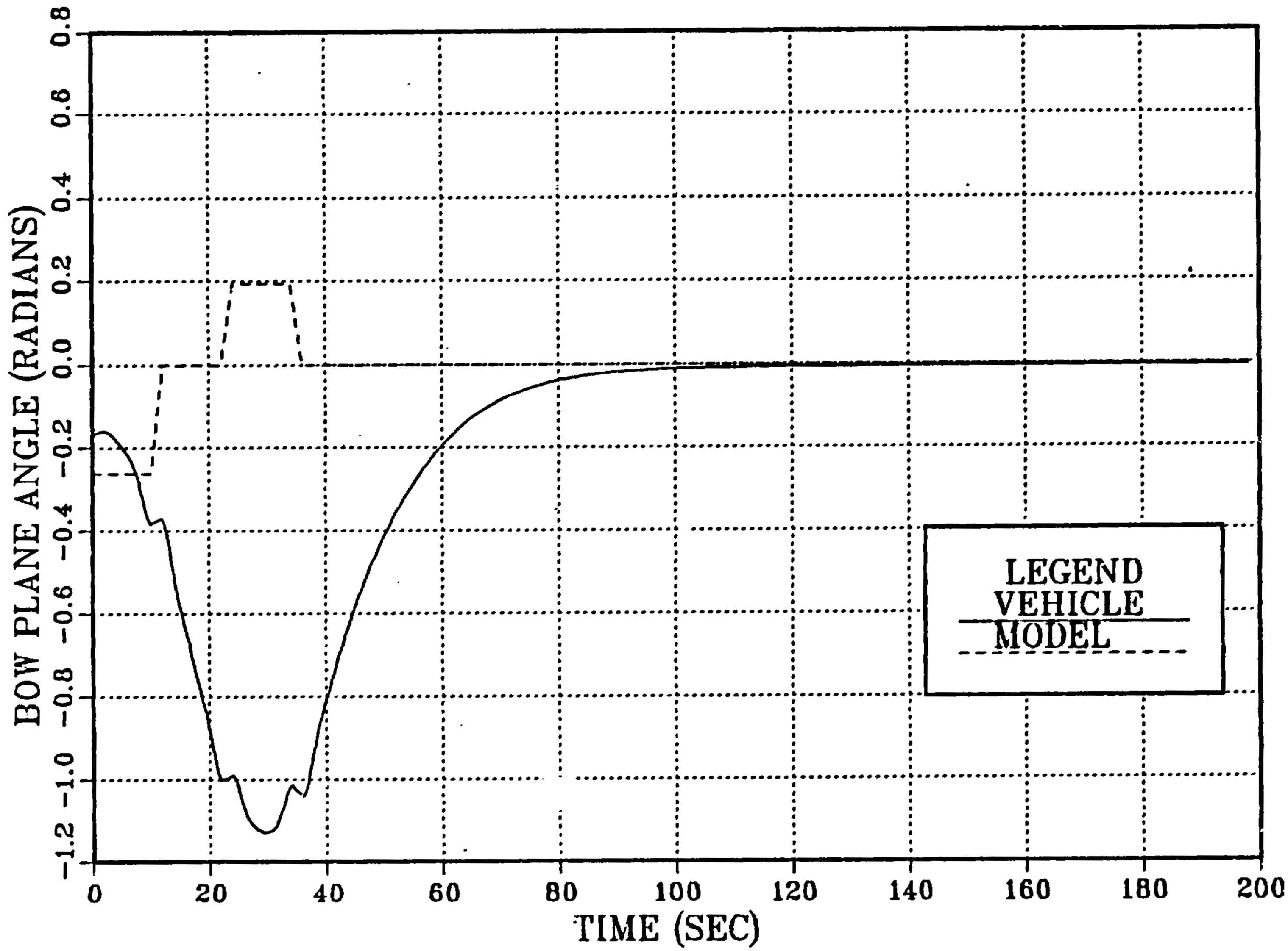

Figure 6.21 Further Pitch Control Weighting Bow Plane Angle

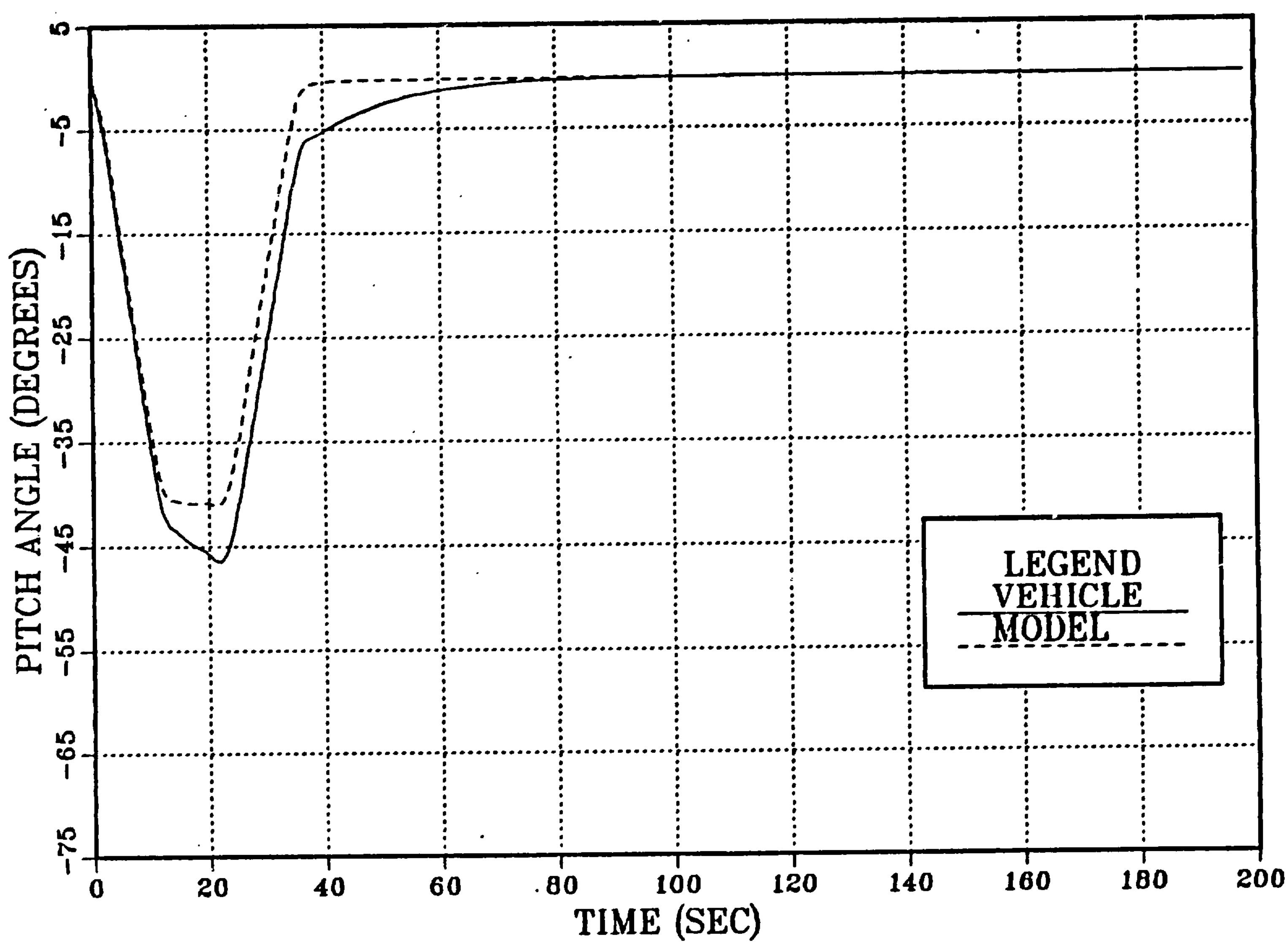

Figure 6.22 Further Pitch Control Weighting Pitch Angle

ft/sec. Figures 6.23 to 6.27 show very good tracking ability even though the vehicle was going twice the speed. Figure 6.24 shows that the vehicle went twice as far as the model to reach the same depth, due primarily to the vehicle speed being double that of the model. Figure 6.27 shows the compensation in pitch angle to achieve desired depth. If the controller was tighter in pitch it would have followed closer in this figure but in Figure 6.25 the accurate trajectory tracking would be lost. Again this goes back to the type of control needed and adjusting of the weights in the $\underline{Q}$ and $\underline{R}$ matrix to generate satisfactory control gains.

The next test of the controller was an attempt to run the vehicle at a speed of 3 ft/sec which is very slow and yet try to use a model speed of 6 ft/sec. The primary motivation was to see if one set of gains and one model could be used for all maneuvers, rather than recalculating gains every time the vehicle changed speeds; a test of robustness in the controller. When the vehicle was operated at 3 ft/sec the vehicle started out lagging the model and then control errors grew while the controller commanded more and more action. But, since the vehicle was much slower than the model ordered, depth and path following could not be achieved.

To alleviate this problem, gains were recalculated and the model was run at 3 ft/sec (Figures 6.28 to 6.32) when excellent tracking was restored. However, at the slower

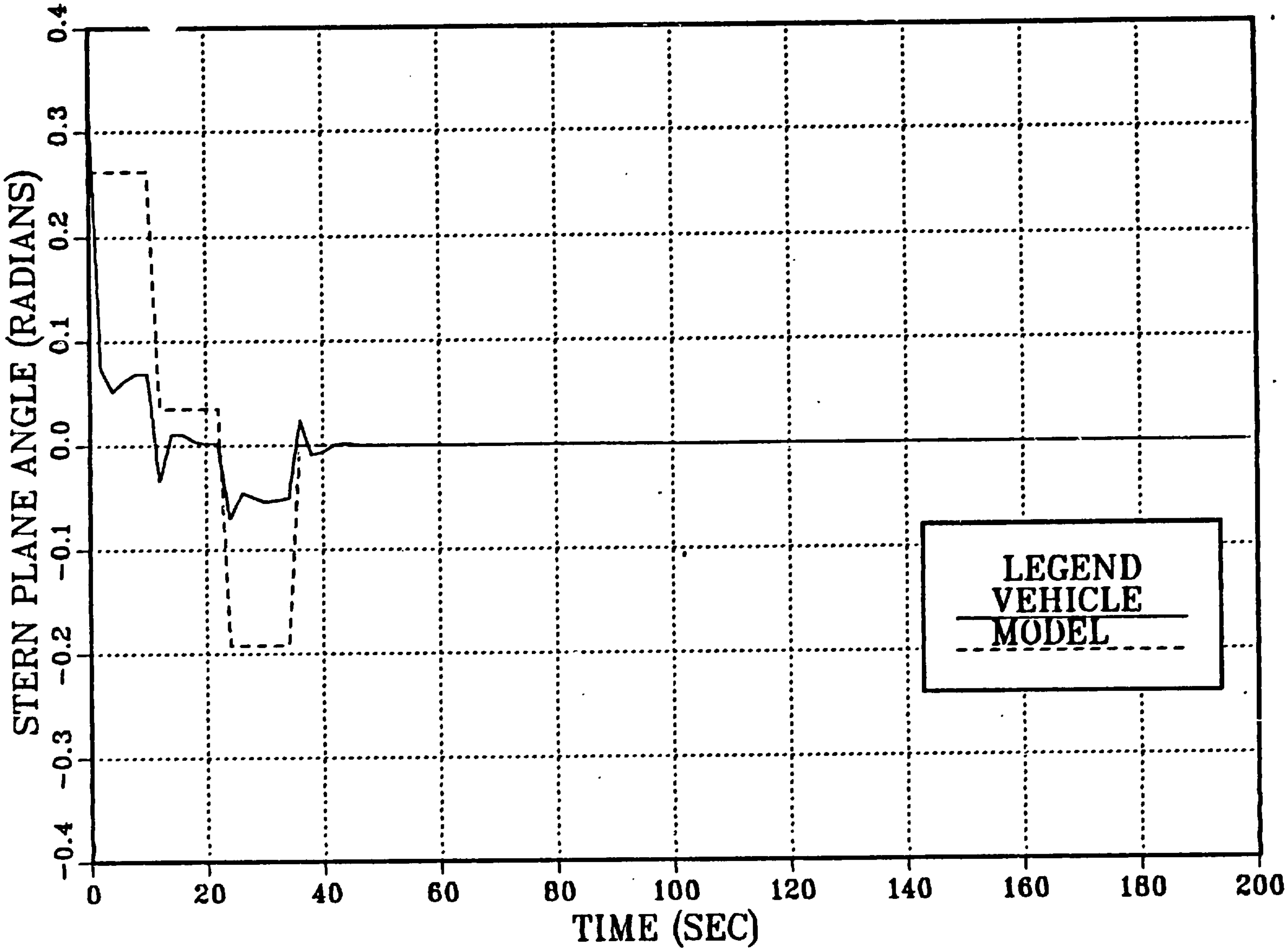

Figure 6.23 Speed Mismatch 12 Ft/Sec Stern Plane Angle

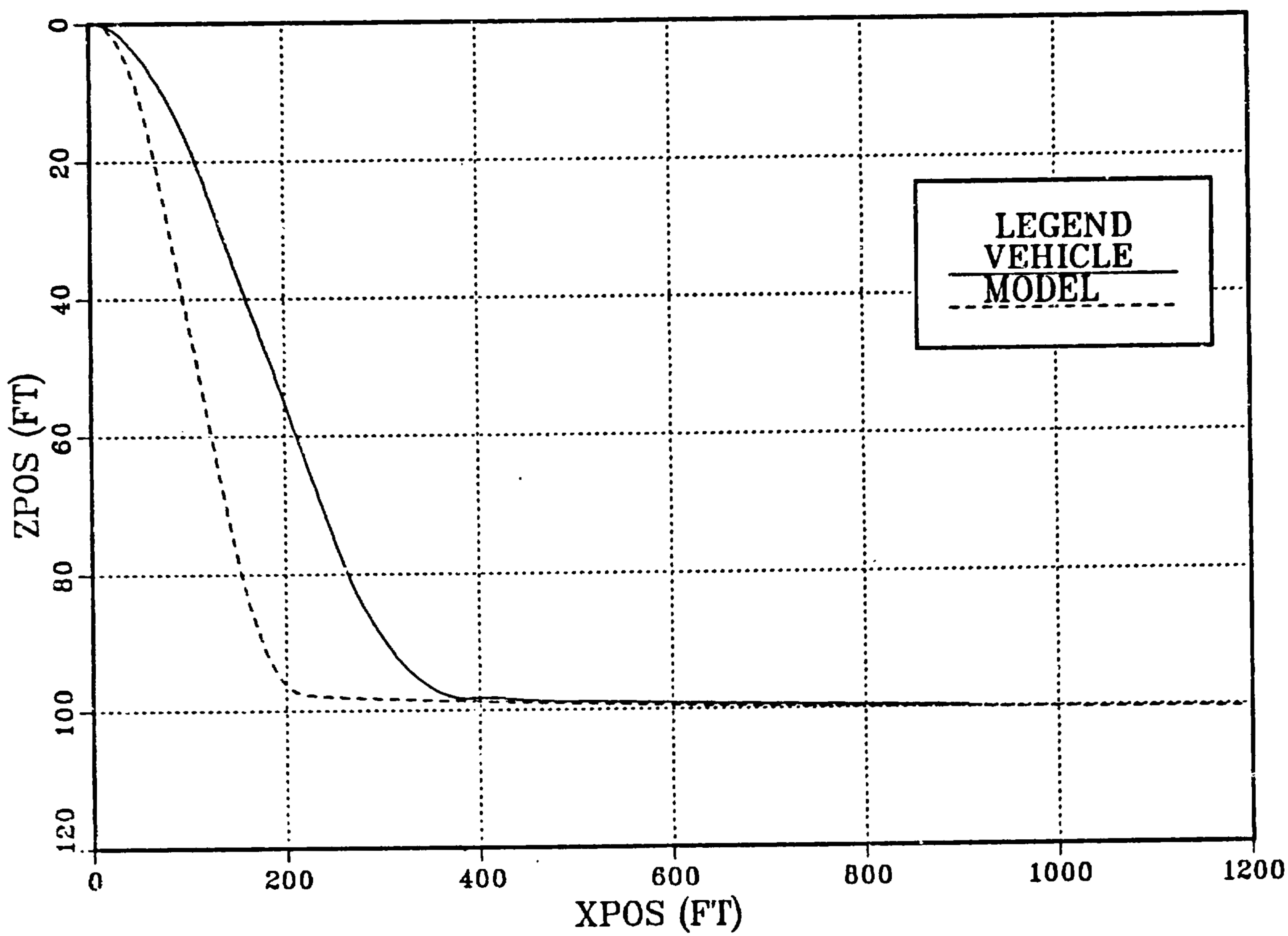

Figure 6.24 Speed Mismatch 12 Ft/Sec Dive Profile

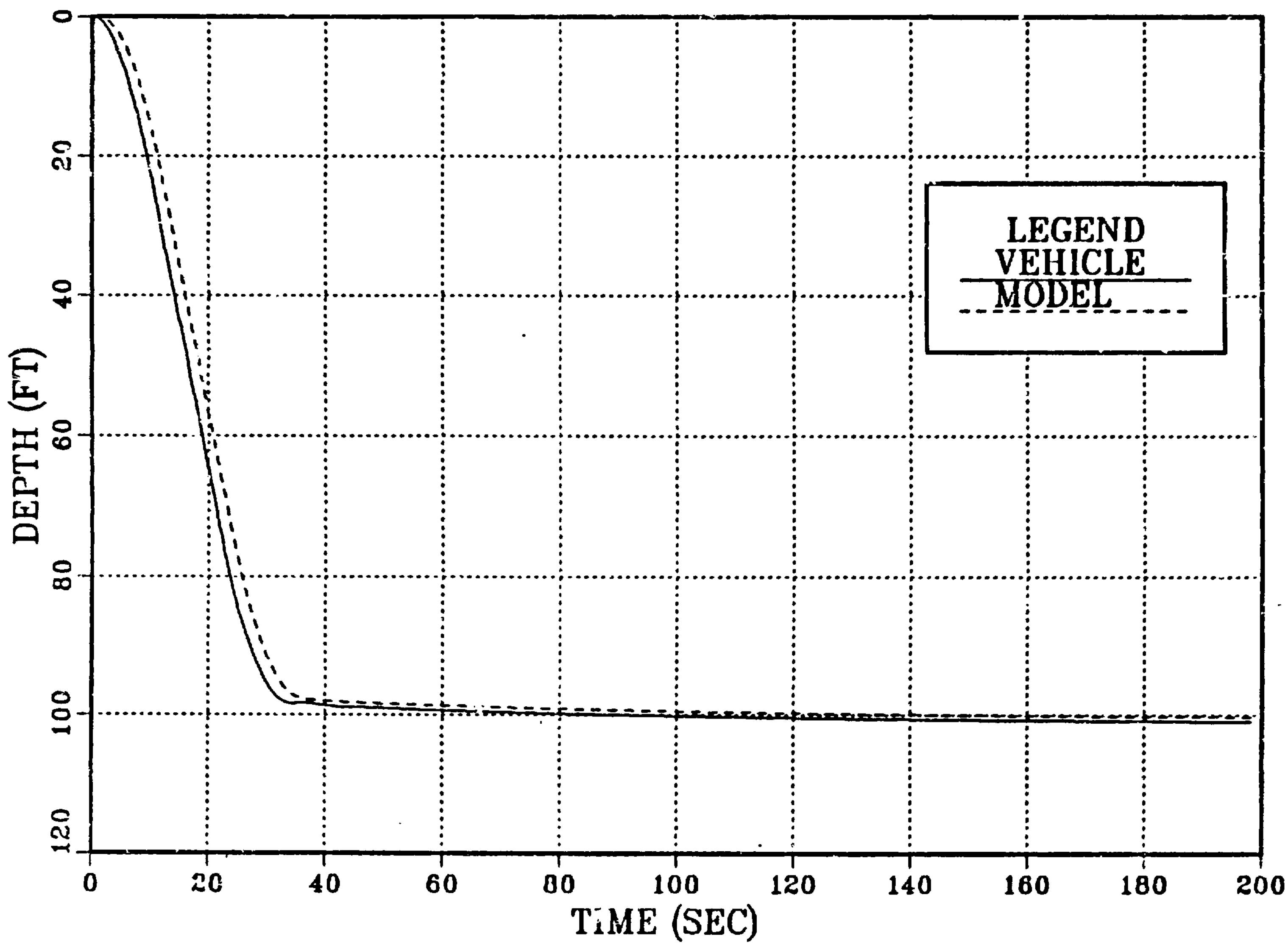

Figure 6.25 Speed Mismatch 12 Ft/Sec Depth Vs Time

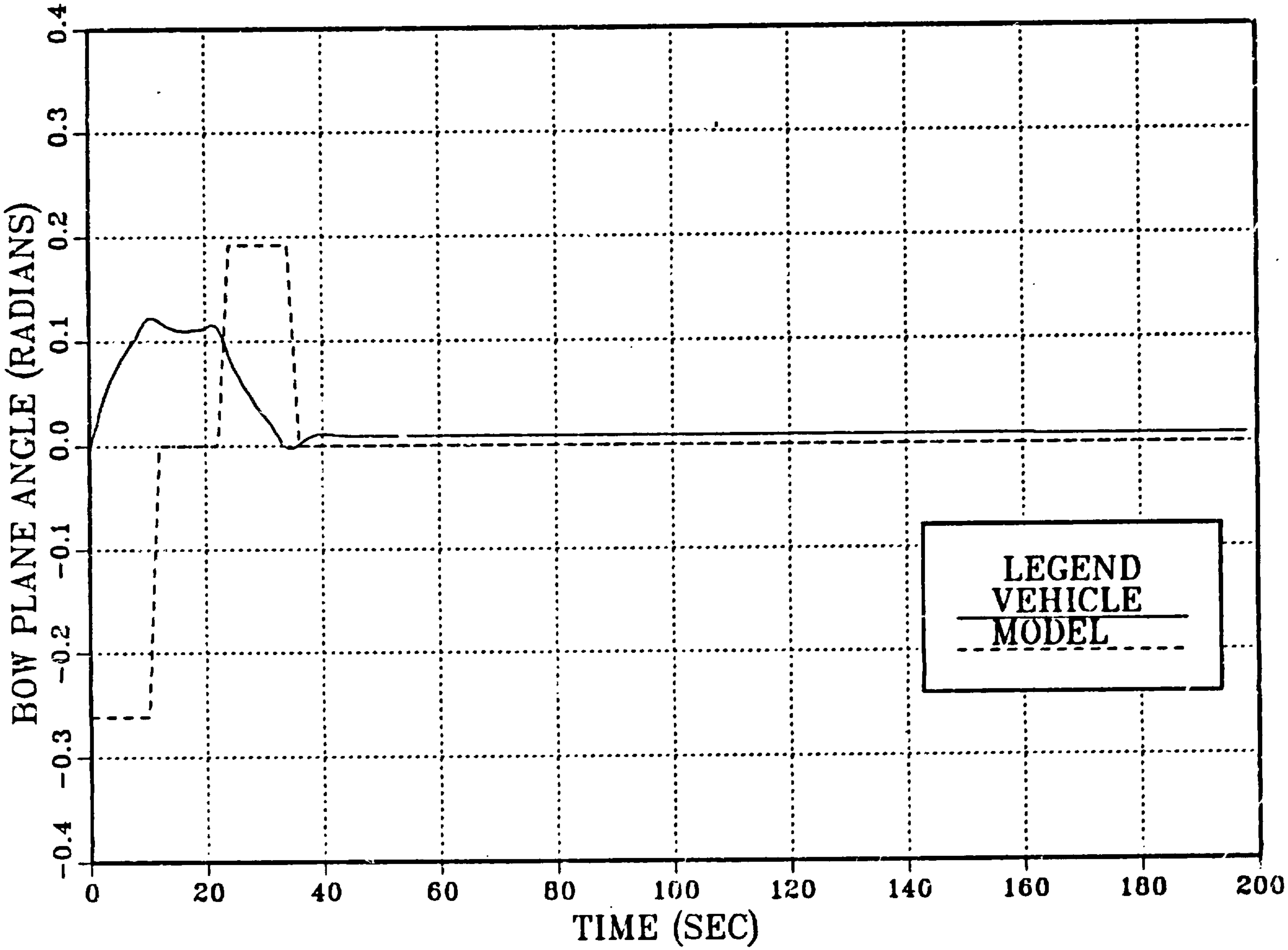

Figure 6.26 Speed Mismatch 12 Ft/Sec Bow Plane Angle

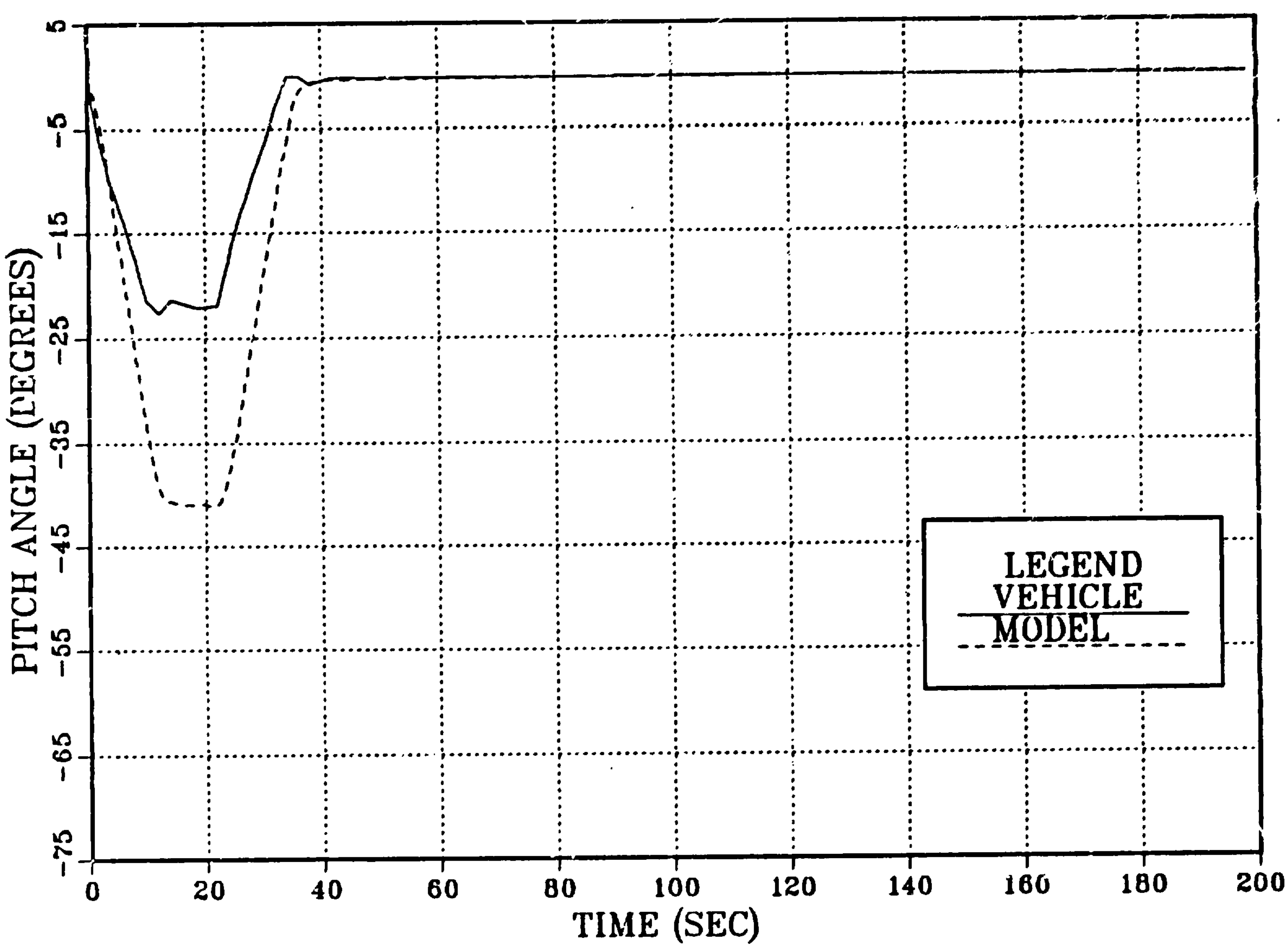

Figure 6.27 Speed Mismatch 12 Ft/Sec Pitch Angle

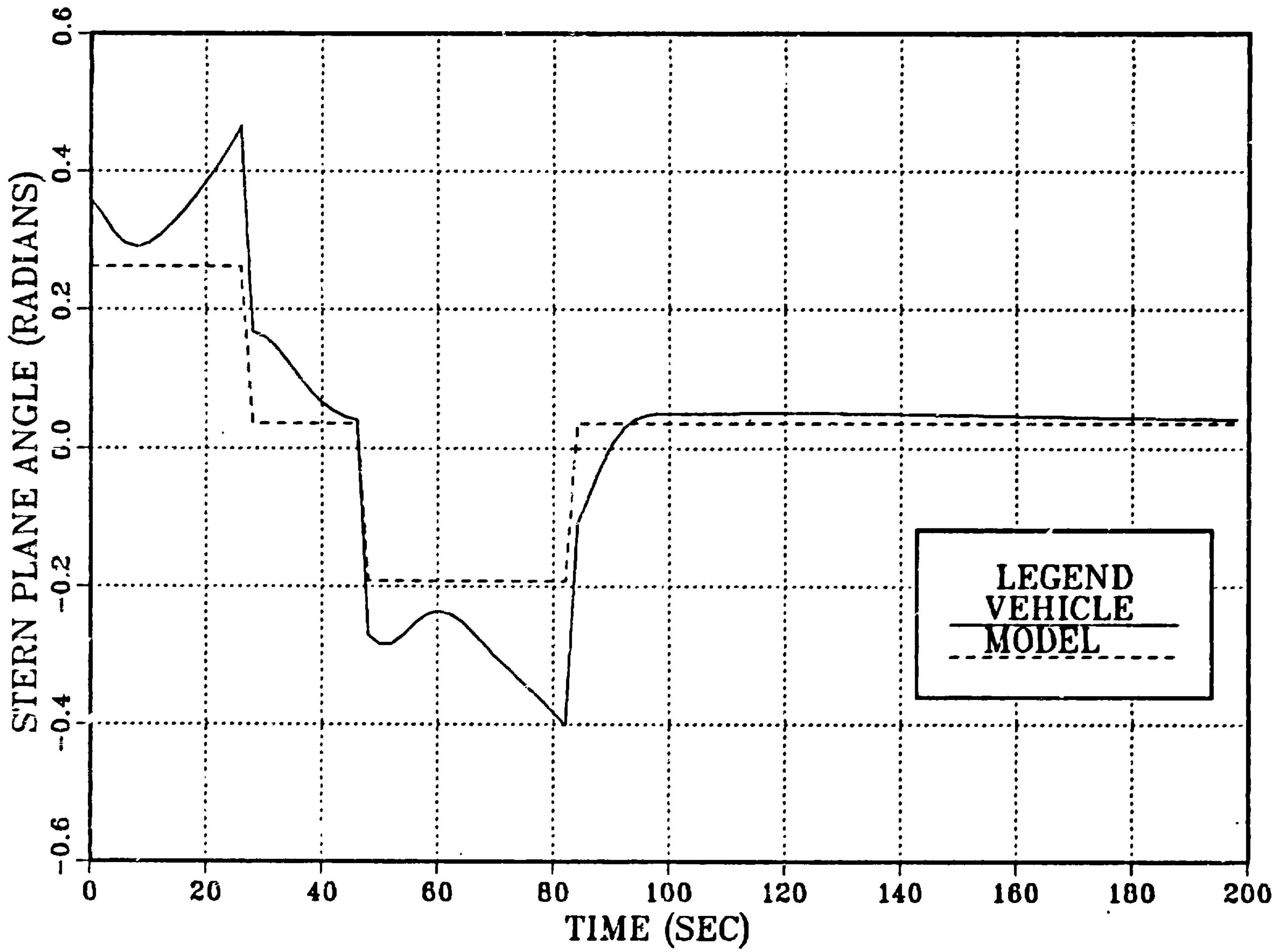

Figure 6.28 Vehicle and Model 3 Ft/Sec Stern Plane Angle

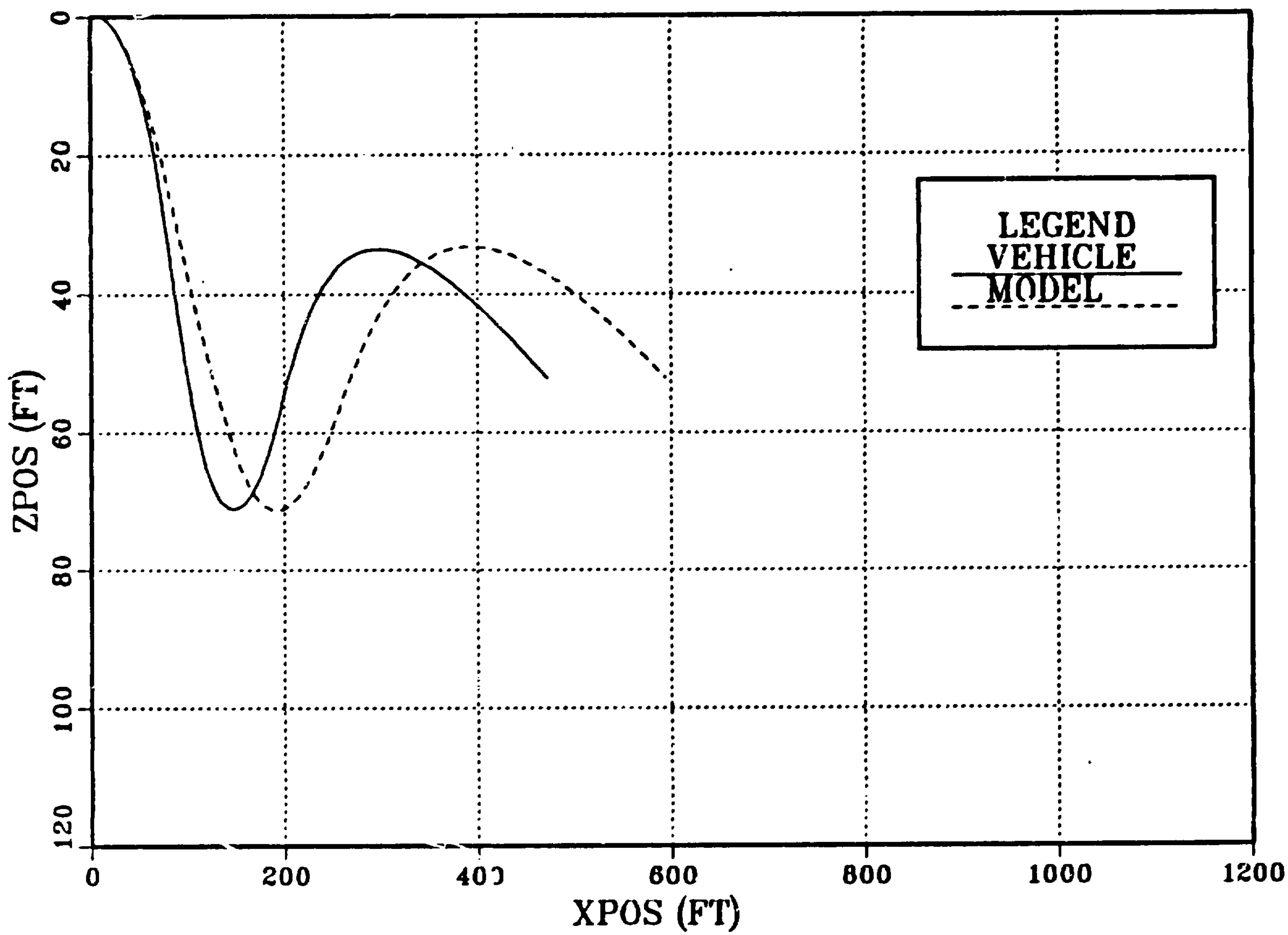

Figure 6.29 Vehicle and Model 3 Ft/Sec Dive Profile

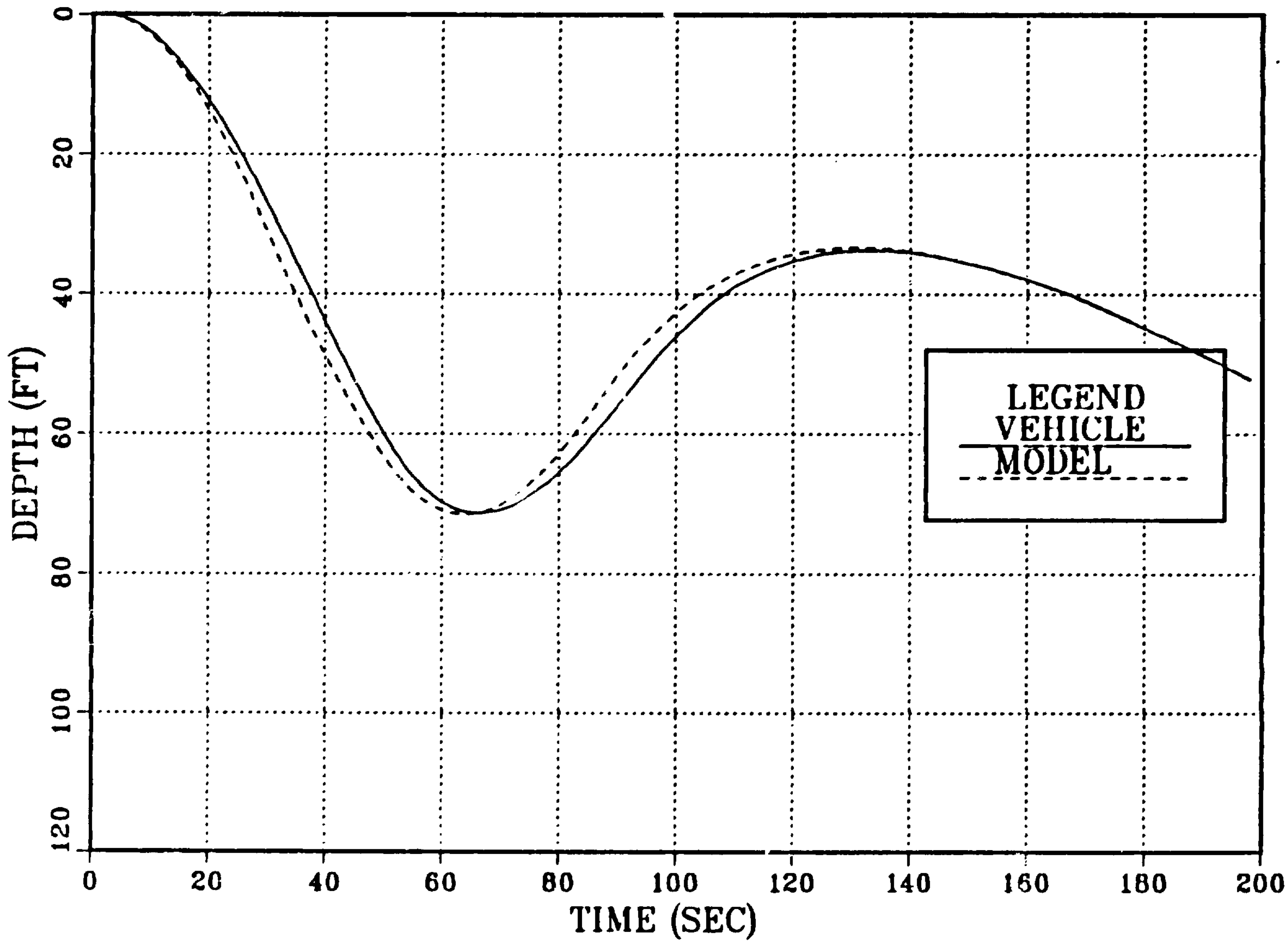

Figure 6.30 Vehicle and Model 3 Ft/Sec Depth Vs Time

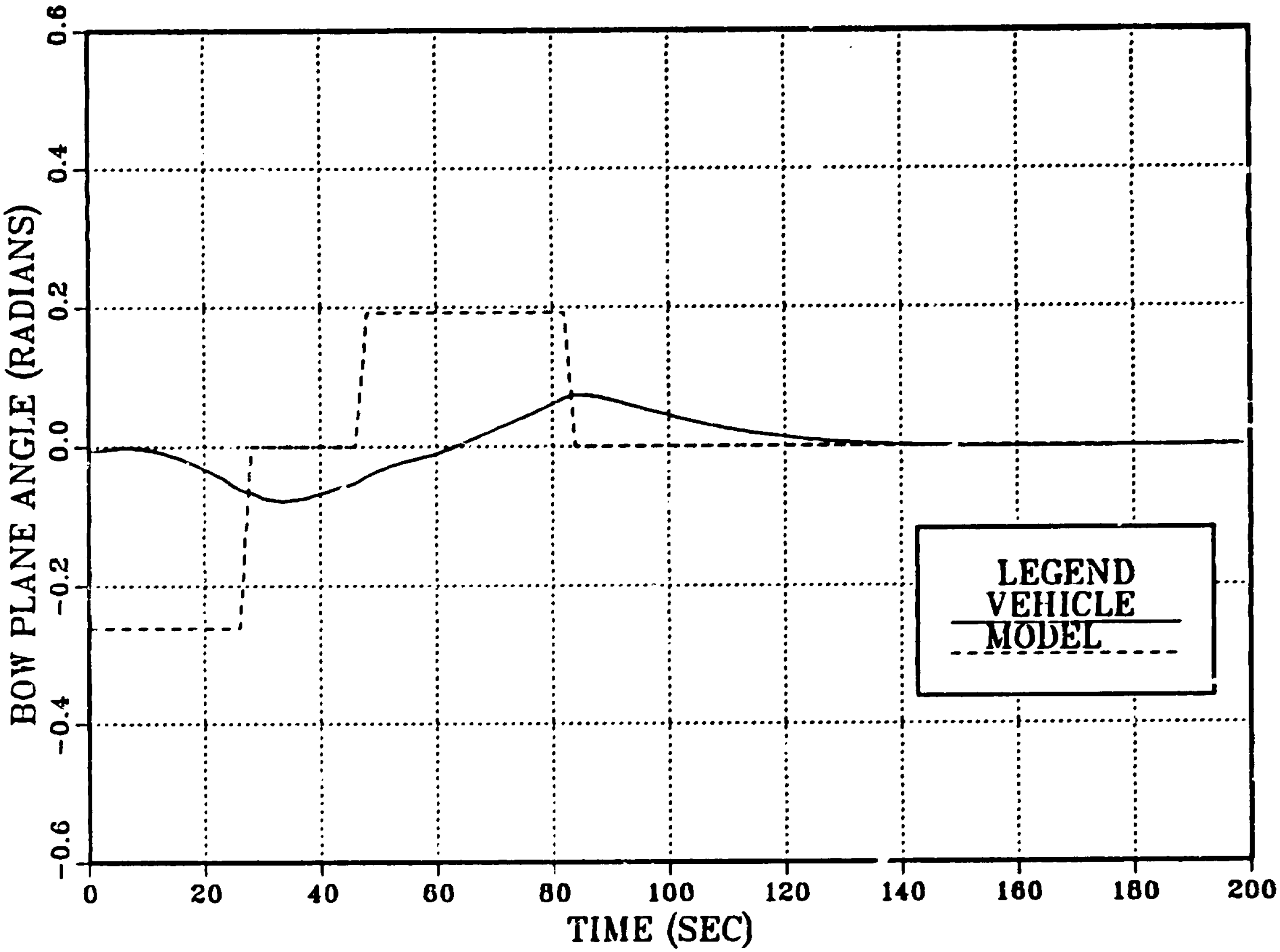

Figure 6.31 Vehicle and Model 3 Ft/Sec Bow Plane Angle

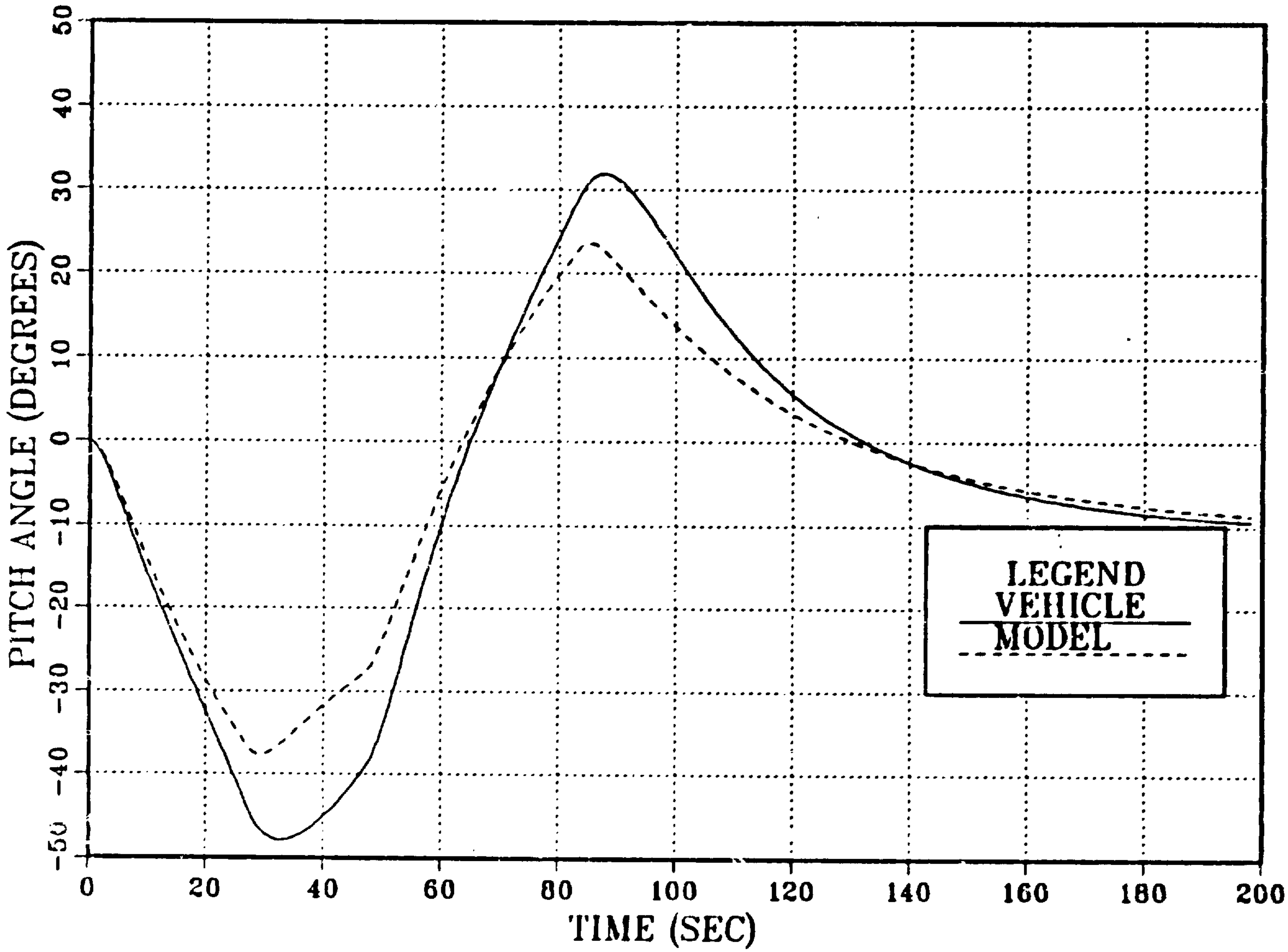

Figure 6.32 Vehicle and Model 3 Ft/Sec Pitch Angle

speed (only 1.75 knots) the dive maneuver algorithm used was not sufficient to maintain pitch during a diving trajectory. The buoyant moment overcame the hydrodynamic moment from control surfaces and ordered depth was not achieved. This behavior, however, is not characteristic of the controller but rather the maneuver logic, and as far as the controller is concerned it was able to follow the model rather nicely.

Since the methodology here was to design a controller that was robust enough to handle a wide variety of reflexive type maneuvers over a range of speeds it is most likely that the vehicle will be traveling at much greater speeds when these maneuvers are executed. For this reason, another simulation run was made. Again the control weights and gains used were as per the baseline case of 6 ft/sec. The model was also at 6 ft/sec and this time the vehicle was at 30 ft/sec. Figures 6.33 to 6.37 show once again excellent tracking control, and like the 12 ft/sec case tight pitch control was eased in favor of accurate depth and trajectory control, which is desirable not to have the vehicle violently pitching during a maneuver which may result in vehicle equipment damage.

H. EIGENVALUES--LINEAR MODEL

The following presents a table of the eigenvalues of the baseline model at 6 ft/sec forward speed together with the clos d loop eigenvalues found using the baseline weights.

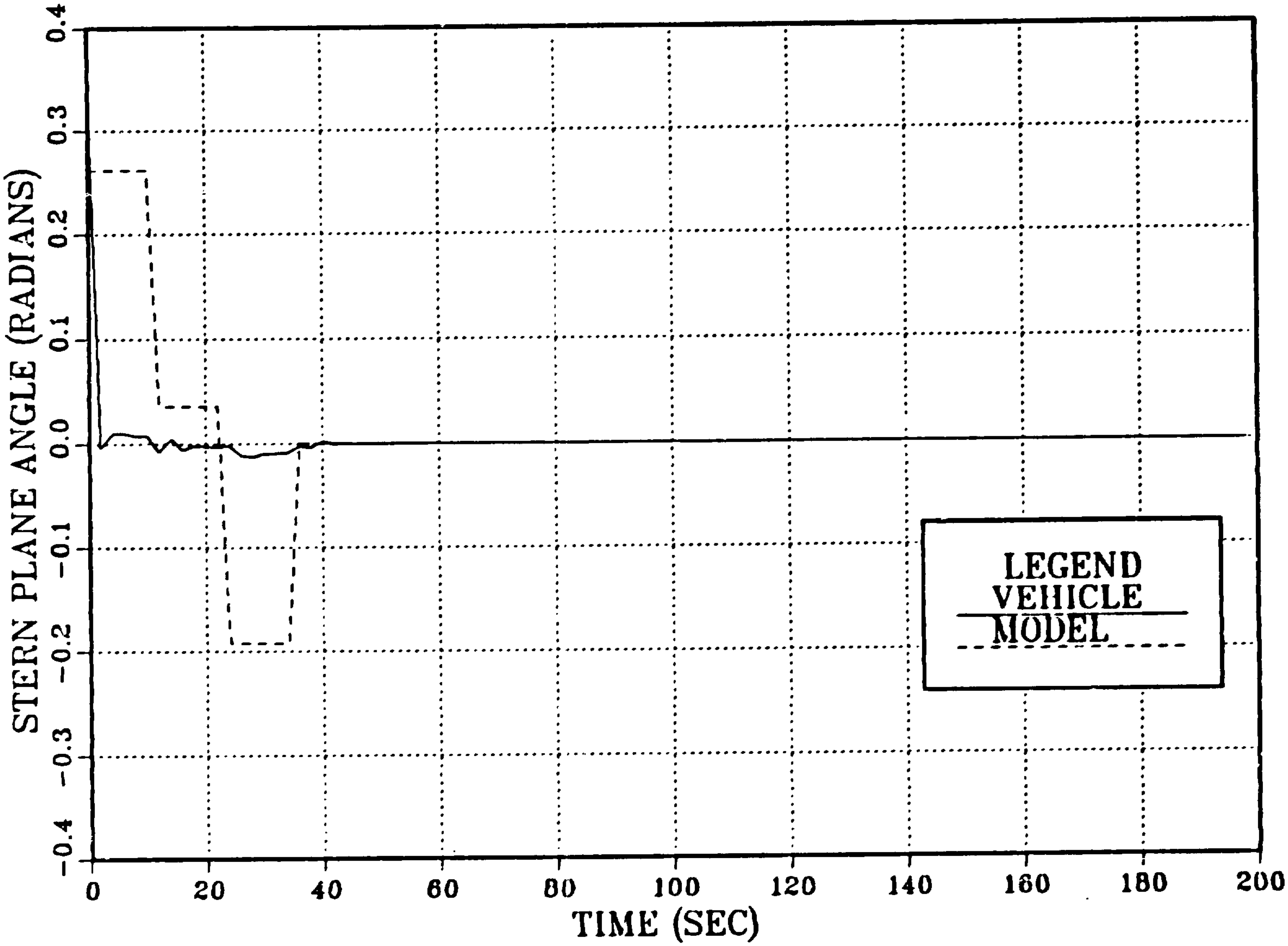

Figure 6.33 Speed Mismatch 30 Ft/Sec Stern Plane Angle

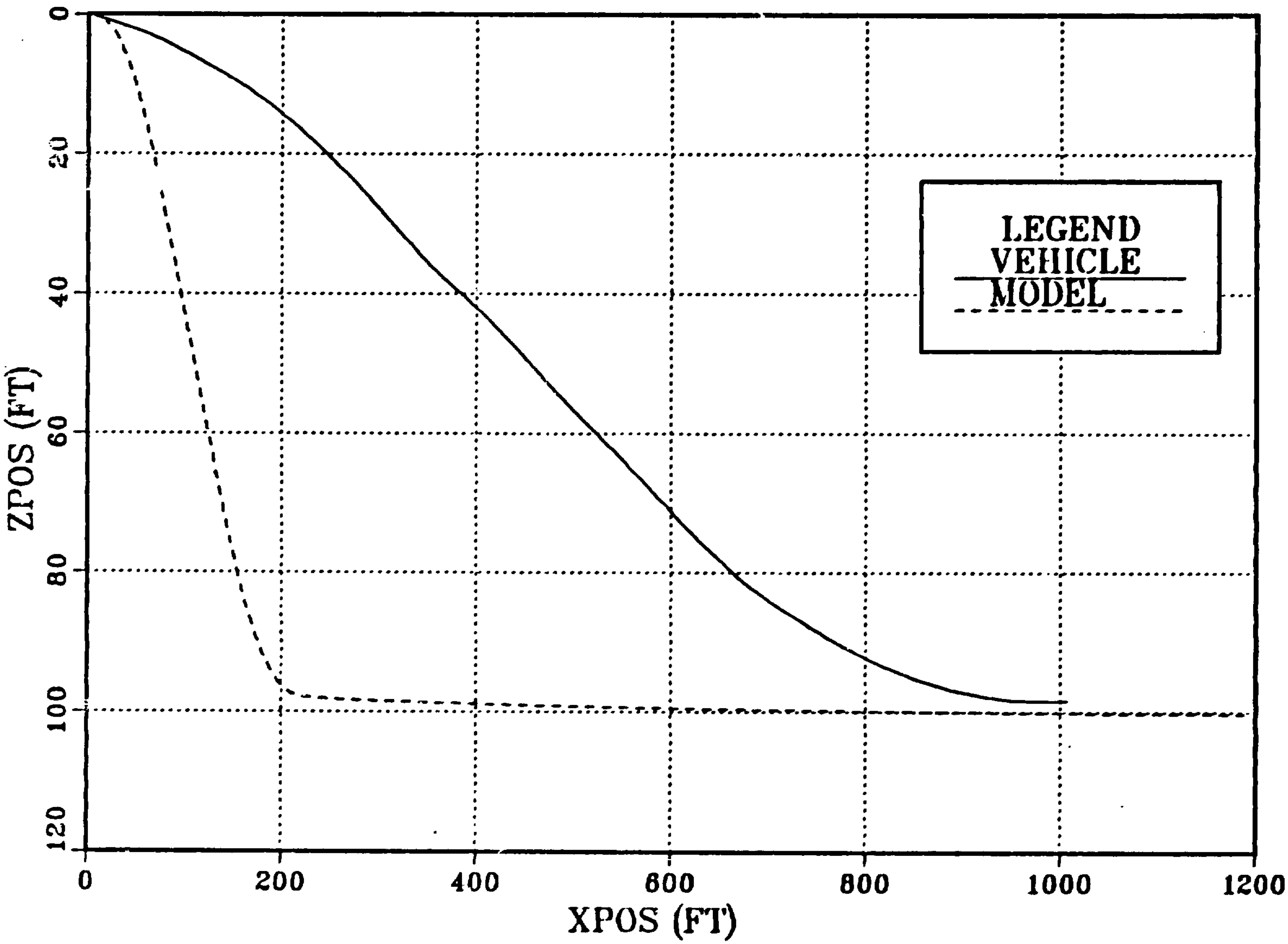

Figure 6.34 Speed Mismatch 30 Ft/Sec Dive Profile

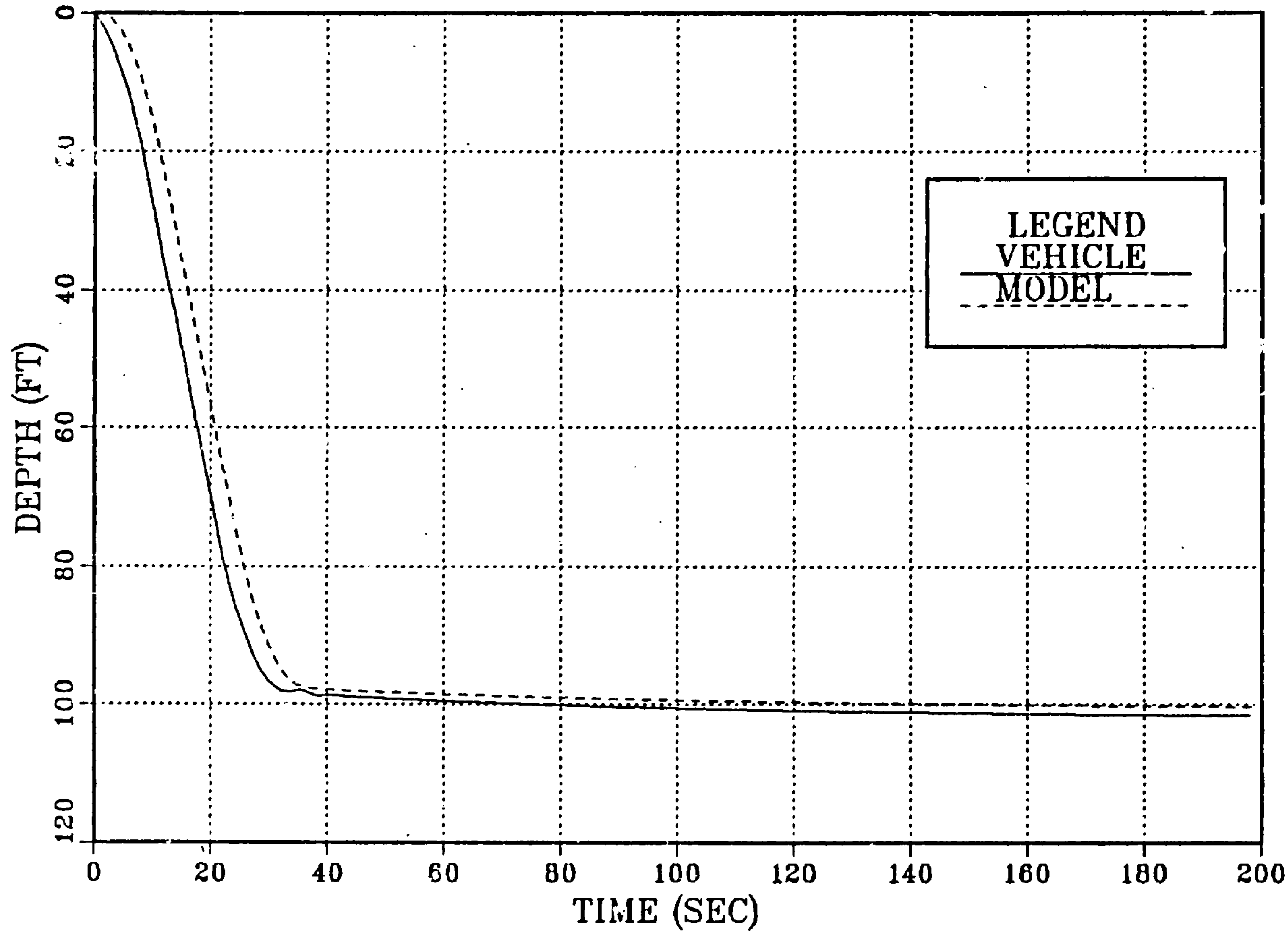

Figure 6.35 Speed Mismatch 30 Ft/Sec Depth Vs Time

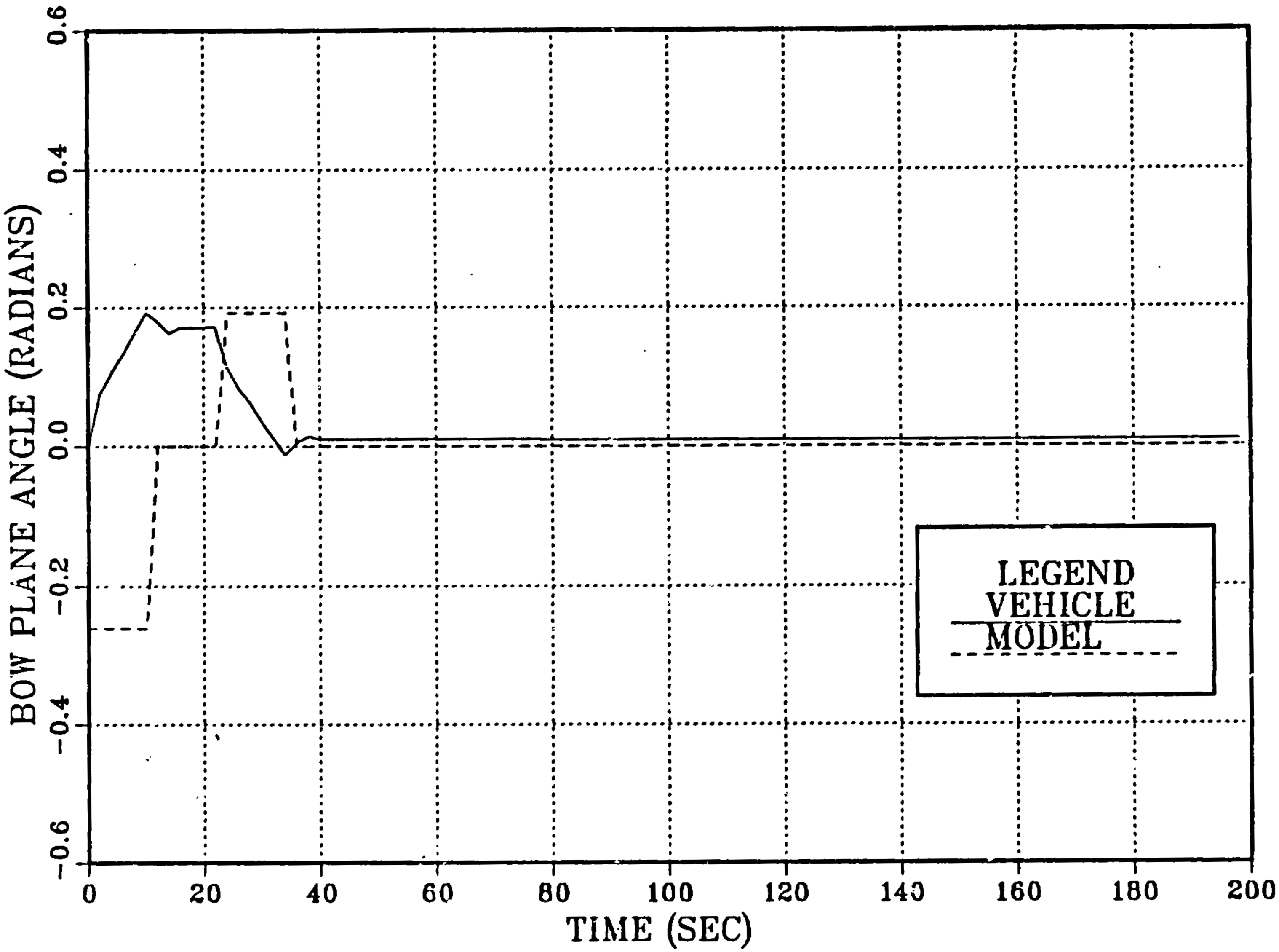

Figure 6.36 Speed Mismatch 30 Ft/Sec Bow Plane Angle

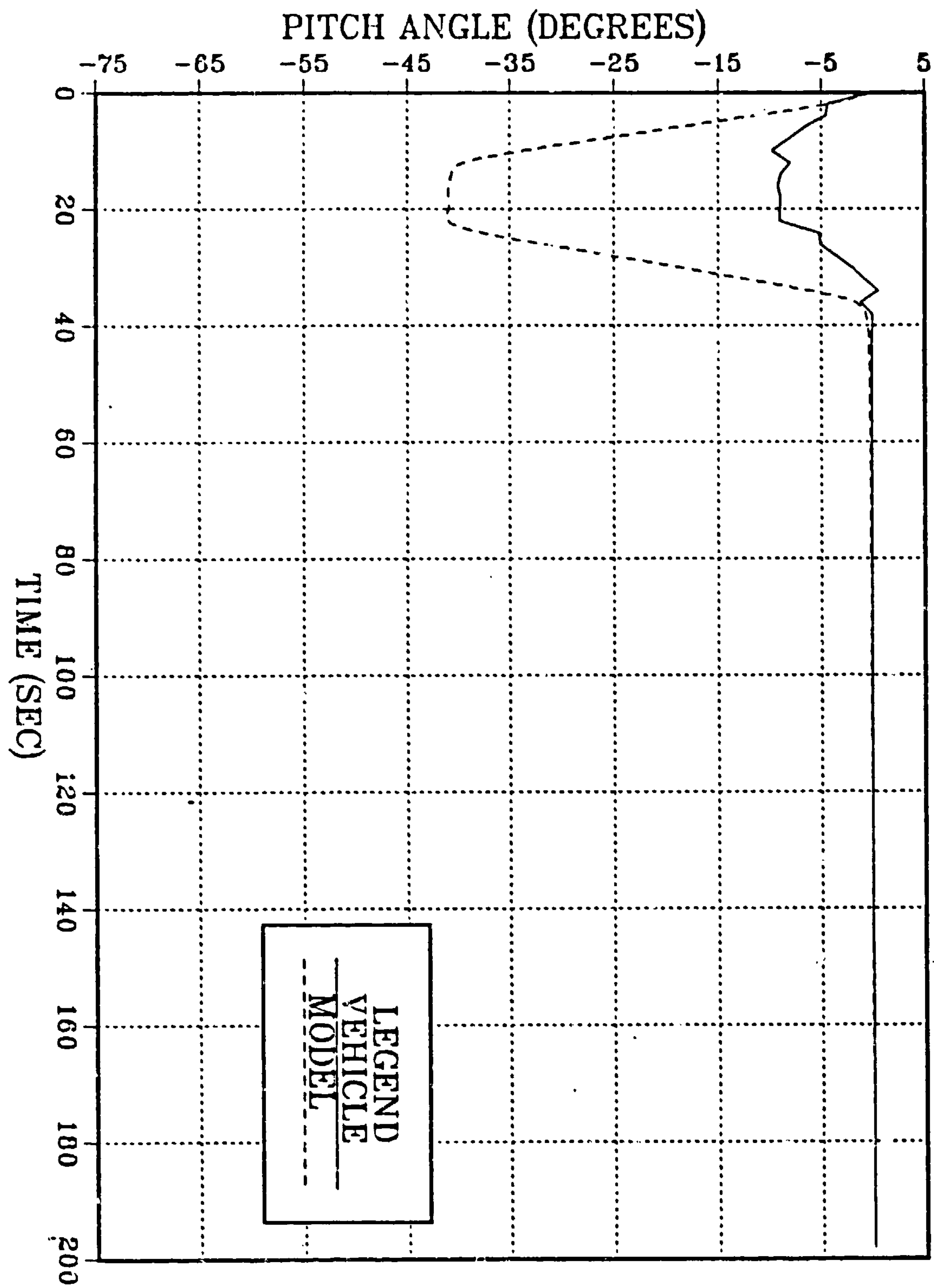

Figure 6.37 Speed Mismatch 30 Ft/Sec Pitch Angle

TABLE 4

OPEN LOOP EIGENVALUES

Eigenvalues	Real Part	Imaginary Part
1	-0.1600E-02	0.0000E+00
2	-0.1500E-02	0.0000E+00
3	-0.1700E-02	0.0000E+00
4	-0.1800E-02	0.0000E+00
5	-0.2100E-02	0.0000E+00
6	-0.1000E-03	0.0000E+00
7	-0.1663E+01	0.0000E+00
8	-0.6579E+00	0.0000E+00
9	-0.9553E+00	0.0000E+00
10	-0.1122E+00	0.7003E-02
11	-0.1122E+00	-0.7003E-02
12	-0.3909E+00	0.0000E+00
13	-0.1603E-01	0.0000E+00
14	-0.9553E+00	0.0000E+00
15	-0.3908E+00	0.0000E+00
16	-0.1423E-01	0.0000E+00
17	-0.3000E-03	0.0000E+00
18	-0.4000E-03	0.0000E+00
19	-0.5000E-03	0.0000E+00

TABLE 5

CLOSED LOOP EIGENVALUES

Eigenvalues	Real Part	Imaginary Part
1	-0.1600E-02	0.0000E+00
2	-0.1500E-02	0.0000E+00
3	-0.1700E-02	0.0000E+00
4	-0.2100E-02	0.0000E+00
5	-0.1663E+01	0.0000E+00
6	-0.9888E+00	0.0000E+00
7	-0.3456E+00	0.4402E+00
8	-0.3454E+00	-0.4402E+00
9	-0.6579E+00	0.0000E+00
10	-0.4868E+00	0.0000E+00
11	-0.9553E+00	0.0000E+00
12	-0.1122E+00	0.7003E-02
13	-0.1122E+00	-0.7003E-02
14	-0.3908E+00	0.0000E+00
15	-0.1423E-01	0.0000E+00
16	-0.1000E-03	0.0000E+00
17	-0.3000E-03	0.0000E+00
18	-0.4000E-03	0.0000E+00
19	-0.5000E-03	0.0000E+00

VII. SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

A. SUMMARY

This thesis presents a study of Model Reference Controls for an Autonomous Underwater Vehicle. The approach to the design and testing of a model following autopilot included:

1. Selection of a suitable submersible was selected, one that displayed many attributes for potential AUV missions. One in which all the hydrodynamic characteristics were well studied and data were obtainable.

2. The tailoring of the existing equations of motion to gain control over all six degrees of freedom.

3. The development of a linearized model and programming the linearized and non-linear models for simulation using Dynamic Simulation Language (DSL).

4. The development of a 19 state controller for dive plane maneuvers. Maneuvers that could be termed reflexive.

5. The development of logic for a command generation system for a dive maneuver.

6. Observation of the effects on the control gains by varying the weights in the minimizing function J.

7. The testing of the command generation logic and the controller over a wide range of speeds using only one set of calculated gains based on one speed of 6 ft/sec.

B. CONCLUSIONS

In this study, a methodology was developed to the design of a model following autopilot that could be used in an Autonomous Underwater Vehicle. A 19 state controller was designed for automatic control of maneuvers in the dive plane. This controller displayed excellent trajectory following characteristics and exhibited a high degree of robustness over a five to one speed range.

The model following autopilot was designed to follow trajectories generated from a preprogrammed maneuver algorithm. This maneuver logic proved to be workable and could easily be developed for a wide variety of maneuvers to be stored on-board in a computer.

In this study maneuver logic was created for one such maneuver, a dive maneuver, and was followed by the designed autopilot. The algorithm used for the dive maneuver was crude but sufficiently proved that the design methods are sound.

Some difficulties in perfect trajectory following occur because of speed mismatch between model and vehicle, and an improvement in modelling speed loss during maneuvers would be worthwhile.

C. RECOMMENDATIONS

Because the concept of Autonomous Underwater Vehicles is fresh and significant progress has been made in the computational abilities of modern computers, a wide diversification of technological avenues need to be explored. Specific to this study the following recommendations are presented.

1. An implementation of the full 30 state dynamically coupled controller in an AUV should be the ultimate goal of this project. In particular, the influence

for forward speed changes should be reflected in the maneuver command generation model for greater control accuracy.

2. Parallel efforts should be carried forward with the development of many maneuver algorithms that could be stored in the AUV's "bag" of maneuvers.

3. Although this controller was designed specifically for the control of an AUV with an unusual geometry, it can and should be tested on underwater vehicles with other geometries.

APPENDIX A

SIX-DEGREE-OF-FREEDOM EQUATIONS OF MOTION AND EULER ANGLE RATES

SURGE EQUATION OF MOTION

$$m[\dot{u} - vr + wq - x_G(q^2 + r^r) + y_G(pq - \dot{r}) + z_G(pr + \dot{q})]$$

$$= \frac{\rho}{2} \ell^4 [X'_{pp}\ p^2 + X'_{qq}\ q^2 + X'_{rr}\ r^2 + X'_{pr}\ pr]$$

$$+ \frac{\rho}{2} \ell^3 [X'_{\dot{u}}\ \dot{u} + X'_{wq}\ wq + X'_{vp}\ vp + X'_{vr}\ vr$$

$$+ uq(X'_{q\delta s}\ \delta_s + X'_{q\delta b/2}\ \delta_{bp} + X'_{q\delta b/2}\ \delta_{bs}$$

$$+ X'_{r\delta r}\ ur\delta_r] \qquad \text{(A-1)}$$

$$+ \frac{\rho}{2} \ell^2 [X'_{vv}\ v^2 + X'_{ww}\ w^2 + X'_{v\delta r}\ uv\delta_r$$

$$+ uw(X'_{w\delta s}\ \delta_s + X'_{w\delta b/2}\ \delta_{bs}$$

$$+ X'_{w\delta b/2}\ \delta_{bp})$$

$$+ u^2(X'_{\delta s\delta s}\ \delta_s^2 + X'_{\delta b\delta b/2}\ \delta_{bp}^2 + X'_{\delta b\delta b/2}\ \delta_{bs}^2$$

$$+ X'_{\delta r\delta r}\ \delta_r^2)]$$

$$- (W-B) \sin\ \theta + \frac{\rho}{2} \ell^3\ X'_{q\delta sn}\ uq\delta_s\ \ \varepsilon(n)$$

$$+ \frac{\rho}{2} \ell^2 [X'_{w\delta sn}\ uw\delta_s + X'_{\delta s\delta sn}\ u^2\ \delta_s^2]\ \varepsilon(n)$$

$$+ \frac{\rho}{2} \ell^2\ u^2\ X'_{prop}$$

SWAY EQUATION OF MOTION

$$m[\dot{v} + ur - wp + x_G(pq + \dot{r}) - y_G(p^2 + r^2) + z_G(qr - \dot{p})]$$

$$= \frac{\rho}{2} \ell^4 [Y'_{\dot{p}} \dot{p} + Y'_{\dot{r}} \dot{r} + Y'_{pq} pq + Y'_{qr} qr]$$

$$+ \frac{\rho}{2} \ell^3 [Y'_{\dot{v}} \dot{v} + Y'_p up + Y'_r ur + Y'_{vq} vq + Y'_{wp} wp + Y'_{wr} wr] \qquad \text{(A-2)}$$

$$+ \frac{\rho}{2} \ell^2 [Y'_v uv + Y'_{vw} vw + Y'_{\delta r} u^2 \delta_r]$$

$$- \frac{\rho}{2} \int_{x_{tail}}^{x_{nose}} [C_{Dy} h(x) (v+xr)^2 + C_{Dz} b(x) (w-xq)^2] \frac{(v+xr)}{U_{cf}(x)} dx$$

$$+ (W-B) \cos \theta \sin \phi$$

HEAVE EQUATION OF MOTION

$$m[\dot{w} + uq + vp + x_G(pr - \dot{q}) + y_G(qr + \dot{p})$$

$$- z_G(p^2 + q^2)]$$

$$= \frac{\rho}{2} \ell^4 [Z'_{\dot{q}}\, \dot{q} + Z'_{pp}\, p^2 + Z'_{pr}\, pr + Z'_{rr}\, r^2]$$

$$+ \frac{\rho}{2} \ell^3 [Z'_{\dot{w}}\, \dot{w} + Z'_q\, uq + Z'_{vp}\, vp + Z'_{vr}\, vr]$$

$$+ \frac{\rho}{2} \ell^2\ [Z'_w\, uw + Z'_{vv}\, v^2 \tag{A-3}$$

$$+ u^2 (Z'_{\delta s}\ \delta_s + Z_{\delta b/2}\ \delta_{bs} + Z_{\delta b/2}\ \delta_{bp}]$$

$$- \frac{\rho}{2} \int_{x_{tail}}^{x_{nose}} [C_{Dy}\ h(x)\,(v+xr)^2$$

$$+ C_{Dz}\ b(x)\,(w-xq)^2]\ \frac{(w-xq)}{U_{cf}(x)}\, dx$$

$$+ (W-B)\ \cos\theta\ \cos\phi$$

$$+ \frac{\rho}{2} \ell^3\ Z'_{qn}\ uq\ \ \varepsilon(n)$$

$$+ \frac{\rho}{2} \ell^2 [Z'_{wn}\ uw + Z'_{\delta sn}\ u^2\ \delta_s]\ \varepsilon(n)$$

ROLL EQUATION OF MOTION

$$I_x \dot{p} + (I_z - I_y)\, qr + I_{xy}(pr - \dot{q}) - I_{yz}(q^2 - r^2)$$

$$- I_{xz}(pq+\dot{r}) + m[y_G(\dot{w} - uq + vp) - z_G(\dot{v} + ur - wp)]$$

$$\frac{\rho}{2}\ell^5[K'_{\dot{p}}\,\dot{p} + K'_{\dot{r}}\,\dot{r} + K'_{pq}\,pq + K'_{qr}\,qr]$$

$$+ \frac{\rho}{2}\ell^4[K'_{\dot{v}}\,\dot{v} + K'_p\,up + K'_r\,ur + K'_{vq}\,vq \qquad \text{(A-4)}$$

$$+ K'_{wp}\,wp + K'_{wr}\,wr]$$

$$+ \frac{\rho}{2}\ell^3[K'_v\,uv + K'_{vw}\,vw + u^2(K'_{\delta b/2}\,\delta_{bp} + K'_{\delta b/2}\,\delta_{bs})]$$

$$+ (y_GW - y_BB)\cos\theta\cos\phi - (z_GW - z_BB)\cos\theta\sin\phi$$

$$+ \frac{\rho}{2}\ell^4\, K'_{pn}\,up\,\varepsilon(n)$$

$$+ \frac{\rho}{2}\ell^3\, u^2\, K'_{prop}$$

PITCH EQUATION OF MOTION

$$I_y \dot{q} + (I_x - I_z)\, pr - I_{xy}(qr + \dot{p}) + I_{yz}(pq - \dot{r})$$

$$+ I_{xz}(p^2 - r^2) - m[x_G(\dot{w} - uq + vp) - z_G(\dot{u} - vr + wq]$$

$$= \frac{\rho}{2}\, \ell^5 [M'_{\dot{q}}\, \dot{q} + M'_{pp}\, p^2 + M'_{pr}\, pr + M'_{rr}\, r^2]$$

$$+ \frac{\rho}{2}\, \ell^4 [M'_{\dot{w}}\, \dot{w} + M'_q\, uq + M'_{vp}\, vp + M'_{vr}\, vr]$$

$$+ \frac{\rho}{2}\, \ell^3 [M'_w\, uw + M'_{vv}\, v^2$$

$$+ u^2 (M'_{\delta s}\, \delta_s + M'_{\delta b/2}\, \delta_{bp} + M'_{\delta b/2}\, \delta_{bs})]$$

$$+ \frac{\rho}{2} \int_{x_{tail}}^{x_{nose}} [C_{Dy}\, h(x)\, (v+xr)^2$$

$$+ C_{Dz}\, b(x)\, (w-xq)^2]\, \frac{(w-xq)}{U_{cf}(x)}\, x\, dx$$

$$- (x_G W - x_B B) \cos\theta \cos\phi - (z_G W - z_B B) \sin\theta$$

$$+ \frac{\rho}{2}\, \ell^4\, M'_{qn}\, uq\ \varepsilon(n)$$

$$+ \frac{\rho}{2}\, \ell^3 [M'_{wn}\, uw + M'_{\delta sn}\, u^2\, \delta_s]\ \varepsilon(n)$$

YAW EQUATION OF MOTION

$$I_z \dot{r} + (I_y - I_x)pq - I_{xy}(p^2-q^2) - I_{yz}(pr+\dot{q})$$

$$+ I_{xz}(qr-\dot{p}) + m[x_G(\dot{v} + ur - wp) - y_G(\dot{u} - vr + wq)]$$

$$= \frac{\rho}{2} \ell^5 [N'_{\dot{p}} \dot{p} + N'_{\dot{r}} \dot{r} + N'_{pq} pq + N'_{qr} qr]$$

$$+ \frac{\rho}{2} \ell^4 [N'_{\dot{v}} \dot{v} + N'_p up + N'_r ur + N'_{vq} vq$$

$$+ N'_{wp} wp + N'_{wr} wr]$$

$$+ \frac{\rho}{2} \ell^3 [N'_v uv + N'_{vw} vw + N'_{\delta r} u^2 \delta_r]$$

$$- \frac{\rho}{2} \int_{x_{tail}}^{x_{nose}} [C_{Dy} h(x) (v+xr)^2$$

$$+ C_{Dz} b(x) (w-xq)^2] \frac{(v+xr)}{U_{cf}(x)} xdx$$

$$+ (x_G W - x_B B) \cos\theta \sin\phi + (y_G W - y_B B) \sin\theta$$

$$+ \frac{\rho}{2} \ell^3 u^2 N'_{prop}$$

Euler Angle Rates

$$\dot{\phi} = p + q \sin \phi \tan \theta + r \cos \phi \tan \theta$$

$$\dot{\theta} = q \cos \phi - r \sin \phi$$

$$\dot{\psi} = q \frac{\sin \phi}{\cos \theta} + r \frac{\cos \phi}{\cos \theta}$$

Inertial Position Rates

$$\begin{aligned}\dot{x}_0 &= u_{c0} + u \cos \psi \cos \theta \\ &\quad + v[\cos \psi \sin \theta \sin \phi - \sin \psi \cos \phi] \\ &\quad + w[\cos \psi \sin \theta \cos \phi + \sin \psi \sin \phi]\end{aligned}$$

$$\begin{aligned}\dot{y}_0 &= v_{c0} + u \sin \psi \cos \theta \\ &\quad + v[\sin \psi \sin \theta \sin \phi + \cos \psi \cos \phi] \\ &\quad + w[\sin \psi \sin \theta \cos \phi - \cos \psi \sin \phi]\end{aligned}$$

$$\begin{aligned}\dot{z}_0 &= w_{c0} - u \sin \theta \\ &\quad + v \cos \theta \sin \psi \\ &\quad + w \cos \theta \cos \phi\end{aligned}$$

Crossflow Velocity

$$u_{cf}(x) = [(v + xr)^2 + (w - xq)^2]^{1/2}$$

APPENDIX B
DSL LISTING FOR AUV SIMULATION

```
TITLE  AUTONOMOUS UNDERWATER VEHICLE (AUV) SIMULATION
D      COMMON/BLOCK1/ MMINV(6,6), MM(6,6), AA(12,12), BB(6,6)
D      COMMON/BLOCK2/ B(6,6),A(12,12), UMOD(6),GKK(6,21)
D      COMMON/BLOCK3/ F(12), FP(6), UCF(4)
D      COMMON/BLOCK4/ G4(4),GK4(4),BR(4),HH(4)
D      COMMON/BLOCK5/ XDOT(12),XDOTX(12), XDOTU(6)
FIXED  N,IA,IDGT,IER,LAST,J,K.M,JJ,KK,I
INTEGER
ARRAY  WKAREA(54), X(12)
*
*
CONST
*
*     LONGITUDINAL HYDRODYNAMIC COEFFICIENTS
*
CONST XPP =            ,XQQ =          ,XRR =          ,XPR =          ,...
      XUDOT=           ,XWQ =          ,XVP =          ,XVR =          ,...
      XQDS=            ,XQDB=          ,XRDR=          ,XVV =          ,...
      XWW =            ,XVDR=          ,XWDS=          ,XWDB=          ,...
      XDSDS=           ,XDBDB=         ,XDRDR=         ,XQDSN=         ,...
      XWDSN=           ,XDSDSN=
*
*     LATERAL HYDRODYNAMIC COEFFICIENTS
*
CONST YPDOT=           ,YRDOT=         ,YPQ =          ,YQR =          ,...
      YVDOT=           ,YP  =          ,YR  =          ,YVQ =          ,...
      YWP =            ,YWR =          ,YV  =          ,YVW =          ,...
      YDR =            ,CDY =
*
*     NORMAL HYDRODYNAMIC COEFFICIENTS
*
CONST ZQDOT=           ,ZPP =          ,ZPR =          ,ZRR =          ,...
      ZWDOT=           ,ZQ  =          ,ZVP =          ,ZVR =          ,...
      ZW  =            ,ZVV =          ,ZDS =          ,ZD3 =          ,...
      ZQN =            ,ZWN =          ,ZDSN=          ,CDZ =
*
*     ROLL HYDRODYNAMIC COEFFITIENTS
*
CONST KPDOT=           , KRDOT=        ,KPQ =          ,KQR =          ,...
      KVDOT=           , KP  =         ,KR  =          ,KVQ=           ,...
      KWP =            , KWR =         ,KV  =          ,KVW =          ,...
      KPN =           , KDB =
*
*     PITCH HYDRODYNAMIC COEFFICIENTS
*
CONST MQDOT=           , MPP =         ,MPR =          ,MRR =          ,...
      MWDOT=           , MQ  =         ,MVP =          ,MVR =          ,...
      MW  =            , MVV =         ,MDS =          ,MDB =          ,...
      MQN =            , MWN =         ,MDSN =
*
*     YAW HYDRODYNAMIC COEFFICIENTS
*
CONST NPDOT=           , NRDOT=        ,NPQ =          ,NQR =          ,...
      NVDOT=           , NP  =         ,NR  =          ,NVQ =          ,...
      NWP =            , NWR =         ,NV  =          ,NVW =          ,...
      NDR =
*
*     MASS CHARACTERISTICS OF THE FLOODED auv
*
CONST WEIGHT =         , BOY =         ,VOL =          ,XG =           ,...
      YG =             , ZG =          ,XB =           ,ZB =           ,...
```

```
      IX =              , IY =             ,IZ =            ,IXZ =         ,...
      IYZ =             , IXY =            ,YB =                           ,...
      L =               , RHO =            ,G =             ,NU =          ,...
      A0 =              ,KPROP =           ,NPROP =    ,     X1TEST=       ,...
*
*     INPUT INITIAL CONDITIONS HERE IF REQUIRED
*
INITIAL
*
*     INITIALIZE ALL MATRICES AND ARRAYS TO ZERO
*
      N = 6
      DO 2  J = 1,N
         JJ= J+N
         DO  1 K = 1,N
         KK= K+N
         KKK= KK + N
         MMINV(J,K) = 0.0
         X(J) = 0.0
         X(JJ) = 0.0
         XDOT(J) = 0.0
         XDOT(JJ) = 0.0
         XDOTX(J) = 0.0
         XDOTX(JJ) = 0.0
         XDOTU(J) = 0.0
         UMOD(J) = 0.0
         MM(J,K) = 0.0
         BB(J,K) = 0.0
         B(J,K) = 0.0
         AA(J,K)= 0.0
         AA(JJ,KK)= 0.0
         AA(J,KK)= 0.0
         AA(JJ,K)= 0.0
         A(J,KK)= 0.0
         A(JJ,K) = 0.0
         A(J,K) = 0.0
         A(JJ,KK)= 0.0
         GKK(J,K)= 0.0
         GKK(J,KK)=0.0
         GKK(J,KKK)=0.0
1        CONTINUE
2     CONTINUE
*
*     INPUT THE LINEARIZATION POINT PARAMETERS
*
      U0 =6.0
      V0 = 0.0
      W0 = 0.0
      P0 = 0.0
      Q0 = 0.0
      R0 = 0.0
      PHI0 = 0.0
      THETA0 = 0.0
      PSI0 = 0.0
      SUM = 0.0
      JFLAG = 0
      IFLAG = 0
      KFLAG = 0
      ZORD = 100.0
*
*     INPUT THE MODEL STATES INITIAL CONDITIONS
*
      UM = 6.0
      VM = 0.0
      WM = 0.0
      PM = 0.0
      QM = 0.0
      RM = 0.0
      XPOSM = 0.0
      YPOSM = 0.0
```

```
      ZPOSM = 0.0
      PHIM = 0.0
      THETAM = 0.0
      PSIM = 0.0
*
*
*     INPUT THE VEHICLE INITIAL CONDITIONS
*
      U = 6.0
      V = 0.0
      W = 0.0
      P = 0.0
      Q = 0.0
      R = 0.0
      XPOS = 0.0
      YPOS = 0.0
      ZPOS = 0.0
      PSI = 0.0
      THETA = 0.0
      PHI = 0.0
*
*     INITIALIZE THE CONTROLS
*
      DELBOY= 0.0
      DBS= 0.0
      DBP= 0.0
      DS = 0.0
      DR = 0.0
      RPM = 250.00
      LATYAW = 0.0
      NORPIT = 0.0
*
*     DEFINE LENGTH FRACTIONS FOR GAUSS QUADUTURE TERMS
*
      G4(1) = 0.069431844
      G4(2) = 0.330009478
      G4(3) = 0.669990521
      G4(4) = 0.930568155
*
*     DEFINE WEIGHT FRACTIONS FOR GAUSS QUADUTURE TERMS
*
      GK4(1) = 0.1739274225687
      GK4(2) = 0.3260725774312
      GK4(3) = 0.3260725774312
      GK4(4) = 0.1739274225687
*
*     DEFINE THE BREADTH BB AND HEIGHT HH TERMS FOR THE INTEGRATION
*
      BR(1) = 75.7/12
      BR(2) = 75.7/12
      BR(3) = 75.7/12
      BR(4) = 55.08/12
*
      HH(1) = 16.38/12
      HH(2) = 31.85/12
      HH(3) = 31.85/12
      HH(4) = 23.76/12
*
      MASS = WEIGHT/G
*
      DIVAMP = DEGSTN*0.0174532925
      RUDAMP = DEGRUD*0.0174532925
*
*
*
*
*     THE LINEAR PROPULSION MODEL
*
*
*     ETA = 0.012*500.0/U0
```

```
      ETA = 1.0
      RE = U0*L/NU
      CD0 =  .00385 + (1.296E-17)*(RE - 1.2E7)**2
      CT  = 0.008*L**2*ETA*ABS(ETA)/(A0)
      CT1 = 0.008*L**2/(A0)
      EPS = -1.0+(SQRT(CT+1.0)-1.0)/(SQRT(CT1+1.0)-1.0)
      XPROP = CD0*(ETA*ABS(ETA) - 1.0)
*
      MASS = WEIGHT/G
      DIVAMP = DEGSTN*0.0174532925
      RUDAMP = DEGRUD*0.0174532925
*
*     CALCULATE THE MASS MATRIX
*
      MM(1,1) = MASS -((RHO/2)*(L**3)*XUDOT)
      MM(1,5) = MASS*ZG
      MM(1,6) = -MASS*YG
*
      MM(2,2) = MASS -((RHO/2)*(L**3)*YVDOT)
      MM(2,4) = -MASS*ZG -((RHO/2)*(L**4)*YPDOT)
      MM(2,6) = MASS*XG - ((RHO/2)*(L**4)*YRDOT)
*
      MM(3,3) = MASS - ((RHO/2)*(L**3)*ZWDOT)
      MM(3,4) = MASS*YG
      MM(3,5) = -MASS*XG -((RHO/2)*(L**4)*ZQDOT)
*
      MM(4,2) = -MASS*ZG - ((RHO/2)*(L**4)*KVDOT)
      MM(4,3) = MASS*YG
      MM(4,4) = IX - ((RHO/2)*(L**5)*KPDOT)
      MM(4,5) = -IXY
      MM(4,6) = -IXZ -((RHO/2)*(L**5)*KRDOT)

      MM(5,1) = MASS*ZG
      MM(5,3) = -MASS*XG -((RHO/2)*(L**4)*MWDOT)
      MM(5,4) = -IXY
      MM(5,5) = IY -((RHO/2)*(L**5)*MQDOT)
      MM(5,6) = -IYZ
*
      MM(6,1) = -MASS*YG
      MM(6,2) = MASS*XG -((RHO/2)*(L**4)*NVDOT)
      MM(6,4) = -IXZ - ((RHO/2)*(L**5)*NPDOT)
      MM(6,5) = -IYZ
      MM(6,6) = IZ - ((RHO/2)*(L**5)*NRDOT)
*
*
      LAST = N*N+3*N
      DO 20 M = 1,LAST
      WKAREA(M) = 0.0
20    CONTINUE
*
      IER = 0
      IA = 6
      IDGT = 4
*     WRITE( 8,400)((MM(I,J), J = 1,6),I = 1,6)
*
      CALL LINV2F(MM,N,IA,MMINV,IDGT,WKAREA,IER)
*
*     WRITE( 8,400)((MMINV(I,J), J = 1,6),I = 1,6)
*00   FORMAT(6E12.4)
*
*
*    CALCULATE THE A MATRIX FOR THE LINFAR MODEL
*
*
*
      A(1,1) = RHO/2*L**3*(XQDS*DS*Q0+XQDB/2*DBP*Q0+XRDR*R0*DR)+...
               RHO/2*L**2*(XVDR*V0*DR+XWDS*DS*W0+XWDB/2*DBP*W0 + ...
               2*U0*(XDSDS*DS**2 + XDBDB/2*DBP**2 + XDRDR*DR**2))+ ...
               RHO/2*L**3*XQDSN*Q0*DS*EPS+RHO/2*L**2*(XWDSN*W0*DS +...
               2*XDSDSN*U0*DS**2)*EPS+RHO*L**2*U0*XPROP+RHO/2*L**3*...
```

```
          XQDB/2*DBS*Q0+RHO/2*L**2*XWDB/2*DBS*W0+RHO*L**2*U0*...
          XDBDB/2*DBS**2
A(1,2) = MASS*R0+RHO/2*L**3*(XVP*P0+ XVR*R0) + RHO/2*L**2* ...
          (2*XVV*V0 + XVDR*U0*DR)
A(1,3) = -MASS*Q0 + RHO/2*L**3*(XWQ*Q0)+RHO/2*L**2*(2*XWW*W0+...
          XWDS*DS*U0+(XWDB/2*DBP+XWDB/2*DBS)*U0 +XWDSN*U0*DS*EPS)
A(1,4) = -MASS*YG*Q0-MASS*ZG*R0+ RHO/2*L**4*(2*XPP*P0+XPR*R0)...
          + RHO/2*L**3*(XVP*V0)
A(1,5) = -MASS*W0+2*MASS*XG*Q0 -MASS*YG*P0+RHO/2*L**4*2*XQQ*Q0...
          +RHO/2*L**3*(XWQ*W0+XQDS*DS*U0+XQDB/2*DBP*U0)+RHO/2* ...
          L**3*XQDSN*U0*DS*EPS+RHO/2*L**3*XQDB/2*DBS*U0
A(1,6) = MASS*V0+2*MASS*XG*R0-MASS*ZG*P0+RHO/2*L**4*(2*XRR*R0...
          + XPR*P0) + RHO/2*L**3*(XVR*V0 + XRDR*U0*DR)
A(1,11)= -(WEIGHT - BOY)*COS(THETA0)
*
A(2,1) = -MASS*R0+RHO/2*L**3*(YP*P0+YR*R0)+RHO/2*L**2*(YV*V0+...
          2*YDR*U0*DR)
A(2,2) = RHO/2*L**3*YVQ*Q0+RHO/2*L**2*(YV*U0+YVW*W0)
A(2,3) = MASS*P0+ RHO/2*L**3*(YWP*P0+YWR*R0)+RHO/2*L**2*YVW*V0
A(2,4) = MASS*W0-MASS*XG*Q0+2*MASS*YG*P0+RHO/2*L**4*YPQ*Q0+...
          RHO/2*L**3*(YP*U0+ YWP*W0)
A(2,5) = -MASS*XG*P0-MASS*ZG*R0+RHO/2*L**4*(YPQ*P0+YQR*R0) +...
          RHO/2*L**3*YVQ*V0
A(2,6) = -MASS*U0+2*MASS*YG*R0-MASS*ZG*Q0+RHO/2*L**4*YQR*Q0 +...
          RHO/2*L**3*(YR*U0 + YWR*W0)
A(2,10)= (WEIGHT - BOY)*COS(THETA0)*COS(PHI0)
A(2,11)= -(WEIGHT - BOY)*SIN(THETA0)*SIN(PHI0)
*
A(3,1) = MASS*Q0+RHO/2*L**3*ZQ*Q0+RHO/2*L**2*(ZW*W0+2*U0*ZDS*DS...
          +2*U0*ZDB/2*DBP+(ZWN*W0+2*ZDSN*U0*DS)*EPS)+RHO/2*L**3*...
          ZQN*Q0*EPS+ RHO/2*L**2*2*U0*ZDB/2*DBS
A(3,2) = -MASS*P0+RHO/2*L**3*(ZVP*P0+ZVR*R0)+RHO/2*L**2*2*ZVV*V0
A(3,3) = RHO/2*L**2*(ZW*U0 + ZWN*U0*EPS)
A(3,4) = -MASS*V0-MASS*XG*R0+2*MASS*ZG*P0+ RHO/2*L**4*(2*ZPP*...
          P0 + ZPR*R0) + RHO/2*L**3*ZVP*V0
A(3,5) = MASS*U0 - MASS*YG*R0+2*MASS*ZG*Q0+RHO/2*L**3*ZQ*U0 +...
          RHO/2*L**3*ZQN*U0*EPS
A(3,6) =-MASS*XG*P0-MASS*YG*Q0+RHO/2*L**4*(ZPR*P0+2*ZRR*R0)+...
          RHO/2*L**3*ZVR*V0
A(3,10)= -(WEIGHT - BOY)*COS(THETA0)*SIN(PHI0)
A(3,11)= -(WEIGHT - BOY)*SIN(THETA0)*COS(PHI0)
*
A(4,1) = MASS*YG*Q0 + MASS*ZG*R0 + RHO/2*L**4*(KP*P0 + ...
          KR*R0)+RHO/2*L**3*(KV*V0+2*U0*(KDB/2*DBP-KDB/2*DBS))+...
          RHO/2*L**3*U0*KPROP+ RHO/2*L**4*KPN*P0*EPS
A(4,2) = -MASS*YG*P0 + RHO/2*L**4*KVQ*Q0 + RHO/2*L**2*(KV*U0...
          + KVW*W0)
A(4,3) = -MASS*ZG*P0 + RHO/2*L**4*(KWP*P0 + KWR*R0) + ...
          RHO/2*L**3*KVW*V0
A(4,4) = -IXY*R0 + IXZ*Q0 - MASS*YG*V0 - MASS*ZG*W0 + ...
          RHO/2*L**5*KPQ*Q0 + RHO/2*L**4*(KP*U0+KWP*W0)
A(4,5) = -IZ*R0 + IY*R0 + 2*IYZ*Q0 + IXZ*P0 + MASS*YG*U0 +...
          RHO/2*L**5*(KPQ*P0 + KQR*R0) + RHO/2*L**4*KVQ*V0
A(4,6) = -IZ*Q0 + IY*Q0 - 2*IYZ*R0 + MASS*ZG*U0 + ...
          RHO/2*L**5*KQR*Q0 + RHO/2*L**4*(KR*U0+KWR*W0)
A(4,10)= -(YG*WEIGHT-YB*BOY)*COS(THETA0)*SIN(PHI0)...
          -(ZG*WEIGHT-ZB*BOY)*COS(THETA0)*COS(PHI0)
A(4,11)= -(YG*WEIGHT-YB*BOY)*SIN(THETA0)*COS(PHI0)...
          +(ZG*WEIGHT-ZB*BOY)*SIN(THETA0)*SIN(PHI0)
*
A(5,1) = -MASS*XG*Q0 + RHO/2*L**4*MQ*Q0 + RHO/2*L**3*MW*W0 +...
          RHO/2*L**3*U0*(MDS*DS+MDB/2*DBP) + RHO/2*L**4*MQN*Q0*...
          EPS + RHO/2*L**3*(MWN*W0 + 2*MDSN*U0*DS)*EPS+...
          RHO/2*L**3*U0*MDB/2*DBS
A(5,2) = MASS*XG*P0 + MASS*ZG*R0 + RHO/2*L**4*(MVP*P0 + ...
          MVR*R0) + RHO*L**3*MVV*V0
A(5,3) = -MASS*ZG*Q0 + RHO/2*L**3*MW*U0 + RHO/2*L**3*MWN*U0*EPS
A(5,4) = -IX*R0 + IZ*R0 - IYZ*Q0 - 2*IXZ*P0 + MASS*XG*V0 + ...
          RHO/2*L**5*(2*MPP*P0 + MPR*R0) + RHO/2*L**4*MVP*V0
A(5,5) = IXY*R0 -IYZ*P0 - MASS*XG*U0 -MASS*ZG*W0 + RHO/2*...
```

```
              L**4*MQ*U0 + RHO/2*L**4*MQN*U0*EPS
      A(5,6) = -IX*P0 + IZ*P0 + IXY*Q0 + 2*IXZ*R0 + MASS*ZG*V0 +...
              RHO/2*L**5*(MPR*P0+2*MRR*R0)+RHO/2*L**4*MVR*V0
      A(5,10)= (XG*WEIGHT-XB*BOY)*COS(THETA0)*SIN(PHI0)
      A(5,11)= (XG*WEIGHT-XB*BOY)*SIN(THETA0)*COS(PHI0) - ...
              (ZG*WEIGHT-ZB*BOY)*COS(THETA0)
*
      A(6,1) = -MASS*XG*R0 + RHO/2*L**4*(NP*P0 +NR*R0) + RHO/2*...
              L**3*(NV*V0+2*NDR*U0*DR)+RHO*L**3*U0*NPROP
      A(6,2) = -MASS*YG*R0 + RHO/2*L**4*NVQ*Q0 + RHO/2*L**3*(NV*U0+...
              NVW*W0)
      A(6,3) = MASS*XG*P0 + MASS*YG*Q0 + RHO/2*L**4*(NWP*P0+NWR*R0)+...
              RHO/2*L**3*NVW*V0
      A(6,4) = -IY*Q0 + IX*Q0 + 2*IXY*P0 +IYZ*R0 + MASS*XG*W0+...
              RHO/2*L**5*NPQ*Q0 + RHO/2*L**4*(NP*U0+NWP*W0)
      A(6,5) = -IY*P0 + IX*P0 - 2*IXY*Q0 - IXZ*R0 + MASS*YG*W0+...
              RHO/2*L**5*(NPQ*P0+NQR*R0) + RHO/2*L**4*NVQ*V0
      A(6,6) = IYZ*P0 -IXZ*Q0 - MASS*XG*U0 -MASS*YG*V0 + ...
              RHO/2*L**5*NQR*Q0 + RHO/2*L**4*(NR*U0 +NWR*W0)
      A(6,10)= (XG*WEIGHT-XB*BOY)*COS(THETA0)*COS(PHI0)
      A(6,11)= -(XG*WEIGHT-XB*BOY)*SIN(THETA0)*SIN(PHI0) +...
              (YG*WEIGHT-YB*BOY)*COS(THETA0)
*
      A(7,1) = COS(PSI0)*COS(THETA0)
      A(7,2) = COS(PSI0)*SIN(THETA0)*SIN(PHI0) - SIN(PSI0)*COS(PHI0)
      A(7,3) = COS(PSI0)*SIN(THETA0)*COS(PHI0) + SIN(PSI0)*SIN(PHI0)
      A(7,10)= V0*COS(PSI0)*SIN(THETA0)*COS(PHI0) + V0*SIN(PSI0)*...
              SIN(PHI0) - W0*COS(PSI0)*SIN(THETA0)*SIN(PHI0) + ...
              W0*SIN(PSI0)*COS(PHI0)
      A(7,11)= -U0*COS(PSI0)*SIN(THETA0) + V0*COS(PSI0)*COS(THETA0)*...
              SIN(PHI0) + W0*COS(PSI0)*COS(THETA0)*COS(PHI0)
      A(7,12)= -U0*SIN(PSI0)*COS(THETA0) - V0*SIN(PSI0)*SIN(THETA0)*...
              SIN(PHI0) - V0*COS(PSI0)*COS(PHI0) - W0*SIN(PSI0)*...
              SIN(THETA0)*SIN(PHI0) + W0*COS(PSI0)*SIN(PHI0)
*
      A(8,1) = SIN(PSI0)*COS(THETA0)
      A(8,2) = SIN(PSI0)*SIN(THETA0)*SIN(PHI0) + COS(PSI0)*COS(PHI0)
      A(8,3) = SIN(PSI0)*SIN(THETA0)*COS(PHI0) - COS(PSI0)*SIN(PHI0)
      A(8,10)= V0*SIN(PSI0)*SIN(THETA0)*COS(PHI0) - V0*COS(PSI0)*...
              SIN(PHI0) - W0*SIN(PSI0)*SIN(THETA0)*SIN(PHI0) - ...
              W0*COS(PSI0)*COS(PHI0)
      A(8,11)= -U0*SIN(PSI0)*SIN(THETA0) + V0*SIN(PSI0)*COS(THETA0)*...
              SIN(PHI0) + W0*SIN(PSI0)*COS(THETA0)*COS(PHI0)
      A(8,12)= U0*COS(PSI0)*COS(THETA0) + V0*COS(PSI0)*SIN(THETA0)*...
              SIN(PHI0) - V0*SIN(PSI0)*COS(PHI0) + W0*COS(PSI0)*...
              SIN(THETA0)*COS(PHI0) + W0*SIN(PSI0)*SIN(PHI0)
*
      A(9,1) = -SIN(THETA0)
      A(9,2) = COS(THETA0)*SIN(PHI0)
      A(9,3) = COS(THETA0)*COS(PHI0)
      A(9,10)= V0*COS(THETA0)*COS(PHI0)-W0*COS(THETA0)*SIN(PHI0)
      A(9,11)= -U0*COS(THETA0)-V0*SIN(THETA0)*SIN(PHI0) -...
               W0*SIN(THETA0)*COS(PHI0)
*
      A(10,4) = 1.0
      A(10,5) = SIN(PHI0)*TAN(THETA0)
      A(10,6) = COS(PHI0)*TAN(THETA0)
      A(10,10)= Q0*COS(PHI0)*TAN(THETA0) - R0*SIN(PHI0)*TAN(THETA0)
      A(10,11)= Q0*SIN(PHI0)/COS(THETA0)*1.0/COS(THETA0) + ...
               R0*COS(PHI0)/COS(THETA0)*1.0/COS(THETA0)
*
      A(11,5) = COS(PHI0)
      A(11,6) = -SIN(PHI0)
      A(11,10)= -Q0*SIN(PHI0) - R0*COS(PHI0)
*
      A(12,5) = SIN(PHI0)/COS(THETA0)
      A(12,6) = COS(PHI0)/COS(THETA0)
      A(12,10)= Q0*COS(PHI0)/COS(THETA0)-R0*SIN(PHI0)/COS(THETA0)
      A(12,11)= Q0*SIN(PHI0)/COS(THETA0)*TAN(THETA0) + ...
               R0*COS(PHI0)/COS(THETA0)*TAN(THETA0)
```

```
*
*      WRITE(10,200)((A(I,J),J=1,12),I=1,12)
*
*     CALCULATE THE B MATRIX
*
*
      B(1,1) = RHO/2*L**3*XRDR*UO*RO+RHO/2*L**2*(XRDR*UO*VO+UO**2*...
               2*XDRDR*DR)
      B(1,2) = UO*QO*XQDB/2 + UO*WO*XWDB/2 + UO**2*XDBDB*DBS
      B(1,3) = UO*QO*XQDB/2 + UO*WO*XWDB/2 + UO**2*XDBDB*DBP
      B(1,4) = UO*QO*XQDS + UO*WO*XWDS +UO**2*2*XDSDS*DS+RHO/2*L**3*...
               XQDSN*UO*QO*EPS + RHO/2*L**2*(XWDSN*UO*WO + 2*XDSDSN*...
               UO**2*DS)*EPS
      B(1,5) = RHO/2*L**2*0.12*CDO*0.12*RPM
      B(1,6) = SIN(THETAO)
*
      B(2,1) = RHO/2*L**2*YDR*UO**2
      B(2,6) = -COS(THETAO)*SIN(PHIO)
*
      B(3,2) = UO**2*ZDB/2*RHO/2*L**2
      B(3,3) = UO**2*ZDB/2*RHO/2*L**2
      B(3,4) = UO**2*ZDS*RHO/2*L**2 + RHO/2*L**2*ZDSN*UO**2*EPS
      B(3,6) = -COS(THETAO)*COS(PHIO)
*
      B(4,2) =-RHO/2*L**3*UO**2*KDB/2
      B(4,3) = RHO/2*L**3*UO**2*KDB/2
      B(4,6) = -YB*COS(THETAO)*COS(PHIO) + ZB*COS(THETAO)*SIN(PHIO)
*
      B(5,2) = RHO/2*L**3*UO**2*MDB/2
      B(5,3) = RHO/2*L**3*UO**2*MD3/2
      B(5,4) = RHO/2*L**3*(UO**2*MDS+MDSN*UO**2*EPS)
      B(5,6) = XB*COS(THETAO)*COS(PHIO) + ZB*SIN(THETAO)
*
      B(6,1) = RHO/2*L**3*NDR*UO**2
      B(6,6) = -XB*COS(THETAO)*SIN(PHIO) - YB*SIN(THETAO)
*
*      WRITE( 9,300)((B(I,J),J=1,6),I=1,6)
*
*     FORMULATE THE A AND B MATRIX FOR STATE SPACE REPRESENTATION
*
*     MULTIPLY MMINV AND DF/DX
*
      DO 80 I = 1,6
         DO 70 J = 1,6
            SUM = 0.0
            DO 60 K = 1,6
            SUM = SUM + MMINV(I,K)*A(K,J)
60          CONTINUE
            AA(I,J) = SUM
70       CONTINUE
80    CONTINUE
*
*
*     MULTIPLY MMINV AND DF/DZ
*
*
      DO 50 I = 1,6
         DO 40 J = 7,12
         SUM = 0.0
             DO 30 K = 1,6
             SUM = SUM + MMINV(I,K)*A(K,J)
30           CONTINUE
             AA(I,J) = SUM
40       CONTINUE
50    CONTINUE
*
*
      DO 5 I = 7,12
         DO 6 J = 1,12
         AA(I,J) = A(I,J)
```

```
6          CONTINUE
5       CONTINUE
*
*
        WRITE(10,200)((AA(I,J),J=1,12),I=1,12)
200     FORMAT( 6E12.4)
*
*
*       MULTIPLY MMINV AND DF/DU
*
*
        DO 110 I = 1,6
           DO 100 J = 1,6
              SUM = 0.0
              DO 90 K = 1,6
              SUM = SUM + MMINV(I,K)*B(K,J)
90            CONTINUE
              BB(I,J) = SUM
100        CONTINUE
110     CONTINUE
*
*
        WRITE( 9,300)((BB(I,J),J=1,6),I=1,6)
300     FORMAT(6E12.4)
*
*
        DO 405 I = 1,6
        READ (2,401)(GKK(I,J), J=1,21)
405     WRITE(3,401)(GKK(I,J), J=1,21)
401     FORMAT(3E20.10)
*
*
DERIVATIVE
NOSORT
*
*
*****LINEAR MODEL*****************************************************
*
*       WRITE(8,600)
*00     FORMAT(14HENTERED DERIV.)
*
*      CALCULATE BB*U  PART OF XDOT = AA*X + BB*U
*
        DO 10 J = 1,6
           SUM = 0.0
           DO 15 K = 1,6
              SUM = SUM + BB(J,K)*UMOD(K)
15         CONTINUE
           XDOTU(J) = SUM
10      CONTINUE
*  CALCULATE AA*X
        DO 21 J= 1,12
           SUM = 0.0
           DO 25 K = 1,12
              SUM = SUM + AA(J,K)*X(K)
25         CONTINUE
           XDOTX(J) = SUM
21      CONTINUE
*  CALCULATE XDOT = AA*X + BB*U
        DO 31 J = 1,6
           XDOT(J) = XDOTX(J) + XDOTU(J)
31      CONTINUE
        DO 35 J = 7,12
           XDOT(J) = XDOTX(J)
35      CONTINUE
*
        UDOTM = XDOT(1)
        VDOTM = XDOT(2)
        WDOTM = XDOT(3)
        PDOTM = XDOT(4)
```

```
      QDOTM = XDOT(5)
      RDOTM = XDOT(6)
      XDOTM = XDOT(7)
      YDOTM = XDOT(8)
      ZDOTM = XDOT(9)
      PHMDOT= XDOT(10)
      THETMD= XDOT(11)
      PSMDOT= XDOT(12)
*     WRITE(8,600)
*   INTEGRATE XDOT TO GET THE STATE VECTOR X
*
      UM =INTGRL(6.0, UDOTM)
      VM= INTGRL(0.0, VDOTM)
      WM= INTGRL(0.0, WDOTM)
      PM= INTGRL(0.0, PDOTM)
      QM= INTGRL(0.0, QDOTM)
      RM= INTGRL(0.0, RDOTM)
      XPOSM = INTGRL(0.0, XDOTM)
      YPOSM = INTGRL(0.0, YDOTM)
      ZPOSM = INTGRL(0.0, ZDOTM)
      PHIM = INTGRL(0.0, PHMDOT)
      THETAM = INTGRL(0.0, THETMD)
      PSIM = INTGRL(0.0, PSMDOT)
*
*
      X(1) = UM
      X(2) = VM
      X(3) = WM
      X(4) = PM
      X(5) = QM
      X(6) = RM
      X(7) = XPOSM
      X(8) = YPOSM
      X(9) = ZPOSM
      X(10) = PHIM
      X(11) = THETAM
      X(12) = PSIM
*
      ZDEPTH = ZORD - X(9)
      THMANG = X(11)*57.3
      DRM = UMOD(1)
      DBSM= UMOD(2)
      DBPM= UMOD(3)
      DSM = UMOD(4)
      DRPM= UMOD(5)
      DBOY= UMOD(6)
*
*
******CONTROL LAW*************************************************************
*
*
*     DBS = UMOD(2)
*     DBP = UMOD(3)
*     DS  = UMOD(4)

      DBS = -(GKK(2,1)*U + GKK(2,2)*V + GKK(2,3)*W + GKK(2,4)*P +...
            GKK(2,5)*Q + GKK(2,6)*R + GKK(2,7)*XPOS + GKK(2,8)*YPOS +...
            GKK(2,9)*ZPOS + GKK(2,10)*PHI + GKK(2,11)*THETA +   ...
            GKK(2,12)*PSI + GKK(2,13)*WM + GKK(2,14)*QM + GKK(2,15)*...
            ZPOSM + GKK(2,16)*THETAM + GKK(2,17)*UMOD(2) + GKK(2,18)*...
            UMOD(3) + GKK(2,19)*UMOD(4))
      DBP = -(GKK(3,1)*U + GKK(3,2)*V + GKK(3,3)*W + GKK(3,4)*P +...
            GKK(3,5)*Q + GKK(3,6)*R + GKK(3,7)*XPOS + GKK(3,8)*YPOS +...
            GKK(3,9)*ZPOS + GKK(3,10)*PHI + GKK(3,11)*THETA +   ...
            GKK(3,12)*PSI + GKK(3,13)*WM + GKK(3,14)*QM + GKK(3,15)*...
            ZPOSM + GKK(3,16)*THETAM + GKK(3,17)*UMOD(2) + GKK(3,18)*...
            UMOD(3) + GKK(3,19)*UMOD(4))
      DS  = -(GKK(4,1)*U + GKK(4,2)*V + GKK(4,3)*W + GKK(4,4)*P +...
            GKK(4,5)*Q + GKK(4,6)*R + GKK(4,7)*XPOS + GKK(4,8)*YPOS +...
            GKK(4,9)*ZPOS + GKK(4,10)*PHI + GKK(4,11)*THETA +   ...
```

```
          GKK(4,12)*PSI + GKK(4,13)*WM + GKK(4,14)*QM + GKK(4,15)*...
          ZPOSM + GKK(4,16)*THETAM + GKK(4,17)*UMOD(2) + GKK(4,18)*...
          UMOD(3) + GKK(4,19)*UMOD(4))
*
*     PUT IN STERN AND BOW PLANE STOPS
*
      IF(ABS(DBS).GT.0.6) THEN
        DBS = 0.6*ABS(DBS)/DBS
      ENDIF
      IF(ABS(DBP).GT.0.6) THEN
        DBP = 0.6*ABS(DBP)/DBP
      ENDIF
      IF(ABS(DS).GT.0.6) THEN
        DS = 0.6*ABS(DS)/DS
      ENDIF
*
*
*****NON-LINEAR MODEL*********************************************
*
*     PROPULSION MODEL
*
*
      SIGNU = 1.0
      IF (U.LT.0.0) SIGNU = -1.0
      IF (ABS(U).LT.X1TEST) U = X1TEST
      SIGNN = 1.0
      IF (RPM.LT.0.0) SIGNN = -1.0
      ETA = 0.012*RPM/U
      RE = U*L/NU
      CD0 =  .00385 + (1.296E-17)*(RE - 1.2E7)**2
      CT  = 0.008*L**2*ETA*ABS(ETA)/(A0)
      CT1 = 0.008*L**2/(A0)
      EPS = -1.0+SIGNN/SIGNU*(SQRT(CT+1.0)-1.0)/(SQRT(CT1+1.0)-1.0)
      XPROP = CD0*(ETA*ABS(ETA) - 1.0)
*
*
*
*     CALCULATE THE DRAG FORCE, INTEGRATE THE DRAG OVER THE VEHICLE
*     INTEGRATE USING A 4 TERM GAUSS QUADUTURE
*
*
      LATYAW = 0.0
      NORPIT = 0.0
      DO 500 K = 1,4
         UCF(K) = SQRT((V+G4(K)*R*L)**2 + (W-G4(K)*Q*L)**2)
         IF(UCF(K).GT.1E-10) THEN
         TERM0  = (RHO/2)*(CDY*HH(K)*(V+G4(K)*R*L)**2 +...
                  CDZ*BR(K)*(W-G4(K)*Q*L)**2)
         TERM1  = TERM0*(V+G4(K)*R*L)/UCF(K)
         TERM2  = TERM0*(W-G4(K)*Q*L)/UCF(K)
         LATYAW = LATYAW + TERM1*GK4(K)*L
         NORPIT = NORPIT + TERM2*GK4(K)*L
         END IF
500   CONTINUE
*
*
*     FORCE EQUATIONS
*
*
*    surge FORCE
*
      FP(1) = MASS*V*R - MASS*W*Q + MASS*XG*Q**2 + MASS*XG*R**2-...
          MASS*YG*P*Q - MASS*ZG*P*R + (RHO/2)*L**4*(XPP*P**2 +...
          XQQ*Q**2 + XRR*R**2 + XPR*P*R) +(RHO/2)*L**3*(XWQ*W*Q +...
          XVP*V*P+XVR*V*R+U*Q*(XQDS*DS+XQDB/2*DBP)+XRDR*U*R*DR)+...
          (RHO/2)*L**2*(XVV*V**2 + XWW*W**2 + XVDR*U*V*DR + U*W*...
          (XWDS*DS+XWDB/2*DBP)+U**2*(XDSDS*DS**2+XDBDB/2*DBP**2+...
          XDRDR*DR**2))-(WEIGHT -BOY)*SIN(THETA) +(RHO/2)*L**3* ...
          XQDSN*U*Q*DS*EPS+(RHO/2)*L**2*(XWDSN*U*W*DS+XDSDSN*U**2*...
```

```
      DS**2)*EPS +(RHO/2)*L**2*U**2*XPROP+RHO/2*L**3*U*Q* ...
      XQDB/2*DBS +RHO/2*L**2*U**2*XDBDB/2*DBS**2+ ...
      RHO/2*L**2*XWDB/2*DBS*U*W
*
*     sway FORCE
*
      FP(2) = -MASS*U*R + MASS*XG*P*Q + MASS*YG*R**2 - MASS*ZG*Q*R +...
      (RHO/2)*L**4*(YPQ*P*Q + YQR*Q*R)+(RHO/2)*L**3*(YP*U*P +...
      YR*U*R + YVQ*V*Q + YWP*W*P + YWR*W*R) + (RHO/2)*L**2*  ...
      (YV*U*V + YVW*V*W +YDR*U**2*DR) -LATYAW +(WEIGHT-BOY)*...
      COS(THETA)*SIN(PHI)
*
*     heave FORCE
*
      FP(3) = MASS*U*Q - MASS*V*P - MASS*XG*P*R - MASS*YG*Q*R +    ...
      MASS*ZG*P**2 + MASS*ZG*Q**2 + (RHO/2)*L**4*(ZPP*P**2 +...
      ZPR*P*R + ZRR*R**2) + (RHO/2)*L**3*(ZQ*U*Q + ZVP*V*P +...
      ZVR*V*R) +(RHO/2)*L**2*(ZW*U*W + ZVV*V**2 + U**2*(ZDS*...
      DS+ZDB/2*DBP))-NORPIT+(WEIGHT-BOY)*COS(THETA)*COS(PHI)+...
      (RHO/2)*L**3*ZQN*U*Q*EPS +(RHO/2)*L**2*(ZWN*U*W +ZDSN*...
      U**2*DS)*EPS+ RHO/2*L**2*U**2*ZDB/2*DBS
*
*     ROLL FORCE
*
      FP(4) = -IZ*Q*R +IY*Q*R -IXY*P*R +IYZ*Q**2 -IYZ*R**2 +IXZ*P*Q +...
      MASS*YG*U*Q -MASS*YG*V*P -MASS*ZG*W*P+(RHO/2)*L**5*(KPQ*...
      P*Q + KQR*Q*R) +(RHO/2)*L**4*(KP*U*P +KR*U*R + KVQ*V*Q +...
      KWP*W*P + KWR*W*R) +(RHO/2)*L**3*(KV*U*V + KVW*V*W) +  ...
      (YG*WEIGHT - YB*BOY)*COS(THETA)*COS(PHI) - (ZG*WEIGHT - ...
      ZB*BOY)*COS(THETA)*SIN(PHI) + (RHO/2)*L**4*KPN*U*P*EPS +...
      (RHO/2)*L**3*U**2*KPROP +MASS*ZG*U*R+ ...
      RHO/2*L**3*U**2*(KDB/2*DBP-KDB/2*DBS)
*
*     PITCH FORCE
*
      FP(5) = -IX*P*R +IZ*P*R +IXY*Q*R -IYZ*P*Q -IXZ*P**2 +IXZ*R**2 -...
      MASS*XG*U*Q + MASS*XG*V*P + MASS*ZG*V*R - MASS*ZG*W*Q +...
      (RHO/2)*L**5*(MPP*P**2 +MPR*P*R +MRR*R**2)+(RHO/2)*L**4*...
      (MQ*U*Q + MVP*V*P + MVR*V*R) + (RHO/2)*L**3*(MW*U*W + ...
      MVV*V**2+U**2*(MDS*DS+MDB/2*DBP))+ NORPIT -(XG*WEIGHT- ...
      XB*BOY)*COS(THETA)*COS(PHI)                            +...
      (RHO/2)*L**3*(MWN*U*W+MDSN*U**2*DS)*EPS+ RHO/2*L**3*...
      U**2*MDB/2*DBS-(ZG*WEIGHT-ZB*BOY)*SIN(THETA)
*
*     YAW FORCE
*
      FP(6) = -IY*P*Q +IX*P*Q +IXY*P**2 -IXY*Q**2 +IYZ*P*R -IXZ*Q*R -...
      MASS*XG*U*R + MASS*XG*W*P - MASS*YG*V*R + MASS*YG*W*Q +...
      (RHO/2)*L**5*(NPQ*P*Q + NQR*Q*R) +(RHO/2)*L**4*(NP*U*P+...
      NR*U*R + NVQ*V*Q +NWP*W*P + NWR*W*R) +(RHO/2)*L**3*(NV*...
      U*V + NVW*V*W + NDR*U**2*DR) - LATYAW + (XG*WEIGHT -  ...
      XB*BOY)*COS(THETA)*SIN(PHI)+(YG*WEIGHT)*SIN(THETA)...
      +(RHO/2)*L**3*U**2*NPROP-YB*BOY*SIN(THETA)
*
*     IF(Z.EQ.50.0)THEN
*        WRITE (8,500)(FP(I), I = 1,6)
*        Z = 0.0
*     END IF
*
*     NOW COMPUTE THE F(1-6) FUNCTIONS
*
      DO 600 J = 1,6
         F(J) = 0.0
      DO 600 K = 1,6
         F(J) = MMINV(J,K)*FP(K) + F(J)
600   CONTINUE
*
*     THE LAST SIX EQUATIONS COME FROM THE KINEMATIC RELATIONS
*
*     FIRST SET THE DRIFT CURRENT VALUES
```

```
*
      UCO = 0.0
      VCO = 0.0
      WCO = 0.0
*
*     INERTIAL POSITION RATES F(7-9)
*
      F(7) = UCO + U*COS(PSI)*COS(THETA) + V*(COS(PSI)*SIN(THETA)*...
             SIN(PHI) - SIN(PSI)*COS(PHI)) + W*(COS(PSI)*SIN(THETA)*...
             COS(PHI) + SIN(PSI)*SIN(PHI))
*
      F(8) = VCO + U*SIN(PSI)*COS(THETA) + V*(SIN(PSI)*SIN(THETA)*...
             SIN(PHI) + COS(PSI)*COS(PHI)) + W*(SIN(PSI)*SIN(THETA)*...
             COS(PHI) - COS(PSI)*SIN(PHI))
*
      F(9) = WCO - U*SIN(THETA) +V*COS(THETA)*SIN(PHI) +W*COS(THETA)*...
             COS(PHI)
*
*     EULER ANGLE RATES F(10-12)
*
      F(10) = P + Q*SIN(PHI)*TAN(THETA) + R*COS(PHI)*TAN(THETA)
*
      F(11) = Q*COS(PHI) - R*SIN(PHI)
*
      F(12) = Q*SIN(PHI)/COS(THETA) + R*COS(PHI)/COS(THETA)

*
*
*     IF (Z.EQ.1.0)WRITE (9,500)(F(I), I = 1,12)
*00   FORMAT(6E12.4)
*     Z = Z + 1
*
*
      UDOT = F(1)
      VDOT = F(2)
      WDOT = F(3)
      PDOT = F(4)
      QDOT = F(5)
      RDOT = F(6)
      XDOTA= F(7)
      YDOT = F(8)
      ZDOT = F(9)
      PHIDOT = F(10)
      THETAD = F(11)
      PSIDOT = F(12)
*
      U = INTGRL(6.0,UDOT)
      V = INTGRL(0.0,VDOT)
      W = INTGRL(0.0,WDOT)
      P = INTGRL(0.0,PDOT)
      Q = INTGRL(0.0,QDOT)
      R = INTGRL(0.0,RDOT)
      XPOS = INTGRL(0.0,XDOTA)
      YPOS = INTGRL(0.0,YDOT)
      ZPOS = INTGRL(0.0,ZDOT)
      PHI = INTGRL(0.0,PHIDOT)
      THETA = INTGRL(0.0,THETAD)
      PSI = INTGRL(0.0,PSIDOT)
*
      ZNEW = -ZPOS
      PHIANG = PHI/0.0174532925
      THEANG = THETA/0.0174532925
      PSIANG = PSI/0.0174532925
*
*
*
DYNAMIC
*
*
      IF (IFLAG.EQ.0.AND.JFLAG.EQ.0)  THEN
```

```
            UMOD(4)= 15.0*0.0174532925
            UMOD(3) = -15.0*0.0174532925
            UMOD(2) = -15.0*0.0174532925
         ENDIF
         IF(IFLAG.EQ.0.AND.ABS(THMANG).GT.37.0)  THEN
            ZCHG = X(9)  - 5.0
            IFLAG = IFLAG + 1
         ENDIF
         IF (IFLAG.GT.0.0.AND.JFLAG.EQ.0) THEN
           UMOD(4)= 2.05*0.0174532925
           UMOD(2) = 0.0
           UMOD(3) = 0.0
         ENDIF
         IF (IFLAG.GT.0.AND.ZCHG.GT.ZDEPTH)  THEN
           UMOD(4)= -11.0*0.0174532925
           UMOD(3) = 11.0*0.0174532925
           UMOD(2) = 11.0*0.0174532925
         ENDIF
         IF (ZDEPTH.LT.3.0.AND.ABS(THMANG).LT.4.10)  THEN
           UMOD(4)= 0.0
           UMOD(3) = 0.0
           UMOD(2) = 0.0
           JFLAG = JFLAG + 1
         ENDIF
         IF (JFLAG.GT.0) THEN
           UMOD(4)= 0.0
           UMOD(3) = 0.0
           UMOD(2) = 0.0
         ENDIF
*
CONTROL FINTIM =200.00,DELT = .01
SAVE   .20,XPOS,XPOSM,U,UM,ZPOS,ZPOSM,W,WM,DBPM,...
          DBS,DBSM,DS,DSM,THEANG,THMANG,Q,QM
PRINT 2.0,XPOS,XPOSM,U,UM,ZPOS,ZPOSM,W,WM,DBPM,...
          DBS,DBSM,DS,DSM,THEANG,THMANG,Q,QM
GRAPH (G1,DE=TEK618) TIME(NI=10,UN=SEC) ZPOS(LI=1,UN=FT)...
                     ZPOSM(LI=2,UN=FT)
GRAPH (G2,DE=TEK618) TIME(NI=10,UN=SEC) W(LI=1,UN='FT/SEC')...
                     WM(LI=2,UN='FT/SEC')
GRAPH (G3,DE=TEK618) TIME(NI=10,UN=SEC) Q(LI=1,UN='RAD/SEC')...
                     QM(LI=2,UN='RAD/SEC')
GRAPH (G4,DE=TEK618) TIME(NI=10,UN=SEC) THEANG(LI=1,UN=DEG)...
                     THMANG(LI=2,UN=DEG)
GRAPH (G5,DE=TEK618) XPOS(UN=FT) ZPOS(UN=FT)
GRAPH (G6,DE=TEK618) XPOSM(UN=FT) ZPOSM(UN=FT)
GRAPH (G7,DE=TEK618) TIME(NI=10,UN=SEC) DBS(LI=1,UN=RADIANS)...
                     DBSM(LI=2,UN=RADIANS)
GRAPH (G8,DE=TEK618) TIME(NI=10,UN=SEC) DS (LI=1,UN=RADIANS)...
                     DSM(LI=2,UN=RADIANS)
LABEL (G1,DE=TEK618) (DEPTH VS TIME)
LABEL (G2,DE=TEK618) (HEAVE VS TIME)
LABEL (G3,DE=TEK618) (PITCH RATE VS TIME)
LABEL (G4,DE=TEK618) (PITCH ANGLE VS TIME)
LABEL (G5,DE=TEK618) (ACTUAL DIVE PROFILE)
LABEL (G6,DE=TEK618) (MODEL DIVE PROFILE)
LABEL (G7,DE=TEK618) (BOW PLANE ANGLE)
LABEL (G8,DE=TEK618) (STERN PLANE ANGLE)
END
STOP
```

```
******SUBROUTINE ETAT*********************************************************
******************************************************************************
      PROGRAM  ETAT
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/SYST/A(40,40),B(40,40),C(40,40),D(40,40)
      COMMON/STATES/X(40),Y(40),U(40),W(40)
      COMMON/DIM/N,M,NR,NKS,EPS
      COMMON/DIM1/DT
      COMMON/FLAGS/IOPT(5)
      COMMON/OPTIM/Q(24,24),R(24,24)
C     DATA  EPS/1.0E-7/
C
C      INITIALIZE ALL SYSTEM MATRICES
C
      EPS = 1.0E-7
      DO 10 I=1,40
      X(I)=0.0
      Y(I)=0.0
      U(I)=0.0
10    CONTINUE
      DO 15 I=1,40
      DO 15 J=1,40
      A(I,J)=0.0
      B(I,J)=0.0
      C(I,J)=0.0
      D(I,J)=0.0
15    CONTINUE
      DO 16 I=1,24
      DO 16 J=1,24
      Q(I,J)=0.0
      R(I,J)=0.0
16    CONTINUE
C
C     SET UP COEFICIENT MATRICES,INPUTS,INITIAL CONDITIONS
C
      CALL INPUT
      CALL MTXEXP
      CALL ROOTS
      DO 20 K=1,NKS
      CALL EXCIT(K)
      CALL UPDAT(K)
      CALL POUT(K)
20    CONTINUE
      IF (IOPT(1).EQ.1)   CALL OPTIMA
C     IF (IOPT(1).EQ.1)   CALL POUTOP
      END
******SUBROUTINE INPUT********************************************************
******************************************************************************
      SUBROUTINE INPUT
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/SYST/A(40,40),B(40,40),C(40,40),D(40,40)
      COMMON/STATES/X(40),Y(40),U(40),W(40)
      COMMON/DIM/N,M,NR,NKS,EPS
      COMMON/DIM1/DT
      COMMON/FLAGS/IOPT(5)
      COMMON/OPTIM/Q(24,24),R(24,24)
C     OPEN(UNIT=5,FILE='FILE',STATUS='OLD')
C     OPEN(UNIT=6,FILE='FILE',STATUS='OLD')
      READ (5,10)  N,L,M,K,NKS,IOPT(1),DT
      WRITE(6,10)  N,L,M,K,NKS,IOPT(1),DT
      READ (5,9)  NAS

      WRITE(6,9)  NAS
      DO 11 II=1,NAS
      READ (5,25) I,J,A(I,J)
      WRITE(6,25) I,J,A(I,J)
11    CONTINUE
      READ (5,9)  NBS
      WRITE(6,9)  NBS
      DO 12 II=1,NBS
```

```
       READ (5,25) I,J,B(I,J)
       WRITE(6,25) I,J,B(I,J)
12     CONTINUE
       READ (5,9)  NCS
       WRITE(6,9)  NCS
       DO 13 II=1,NCS
       READ (5,25) I,J,C(I,J)
       WRITE(6,25) I,J,C(I,J)
13     CONTINUE
       READ (5,9)  NDS
       WRITE(6,9)  NDS
       DO 14 II=1,NDS
       READ (5,25) I,J,D(I,J)
       WRITE(6,25) I,J,D(I,J)
14     CONTINUE
       READ(5,9) NXS
       WRITE(6,9) NXS
       DO 35 II=1,NXS
       READ(5,25) I,J,X(I)
       WRITE(6,25) I,J,X(I)
35     CONTINUE
       IF(IOPT(1).NE.1)  GO TO 190
       READ(5,9)  NQS
       WRITE(6,9) NQS
       DO 150 II=1,NQS
       READ(5,25) I,J,Q(I,J)
       WRITE(6,25) I,J,Q(I,J)
150    CONTINUE
       READ(5,9)  NRS
       WRITE(6,9) NRS
       DO 180 II=1,NRS
       READ(5,25)  I,J,R(I,J)
       WRITE(6,25) I,J,R(I,J)
180    CONTINUE
       NR=L
9      FORMAT(5X,I5)
10     FORMAT(5X,6I5,E10.4)
25     FORMAT (I5,I5,E10.4)
190    RETURN
       END
******SUBROUTINE MTXEXP*************************************************
************************************************************************
       SUBROUTINE MTXEXP
       IMPLICIT REAL*8(A-H,O-Z)
       COMMON/SYST/A(40,40),B(40,40),C(40,40),D(40,40)
       COMMON/DIM/N,M,NR,NKS,EPS
       COMMON/DIM1/DT
       COMMON/ETT/E(40,40),H(40,40)
       DIMENSION DD(40,40),L(50),RHO(50,2),W(50)
       MK=30
       DO 20 I=1,N
       L(I)=1
       RHO(I,1)=1.0
       DO 20 J=1,N
            IF(I.EQ.J)    GO TO 10
       E(I,J)=0.0
       DD(I,J)=0.0
       H(I,J)=0.0
           GO TO 20
10     CONTINUE
       E(I,J)=1.0
       DD(I,J)=1.0
       H(I,J)=DT
20     CONTINUE
       MM=0
       K=1
       X=DT
30     CONTINUE
       DO 80 I=1,N
       IF(L(I).EQ.0) GO TO 80
```

```
      DO 40 J=1,N
      W(J)=0.0
40    CONTINUE
      DO 60 KK=1,N
      Y=DD(I,KK)
      IF(Y.EQ.0.0)  GO TO 60
      DO 50 J=1,N
      W(J)=W(J)+Y*A(KK,J)
50    CONTINUE
60    CONTINUE
      DO 70 J=1,N
      DD(I,J)=W(J)*X
70    CONTINUE
80    CONTINUE
      K=K+1
      X=K
      X=DT/X
      DO 100 I=1,N
      IF(L(I).EQ.0)  GO TO 100
      Y1P=0.0
      DO 90 J=1,N
      E(I,J)=E(I,J)+DD(I,J)
      H(I,J)=H(I,J)+X*DD(I,J)
      Y1P=Y1P+ABS(E(I,J))
90    CONTINUE
      RHO(I,2)=Y1P
      IF(ABS((RHO(I,2)-RHO(I,1))/RHO(I,2)).GT.EPS)  GO TO 100
      L(I)=0
      MM=MM+1
100   CONTINUE
      IF(K.GT.MK)   GO TO 130
      IF(MM.EQ.N)  GO TO 120
      DO 110 I=1,N
      RHO(I,1)=RHO(I,2)
110   CONTINUE
      GO TO 30
120   CONTINUE
      DO 125 I=1,N
      DO 125 J=1,N
      DD(I,J)=0.0
      DO 125 K=1,N
      DD(I,J)=DD(I,J)+H(I,K)*B(K,J)
125   CONTINUE
      DO 135 I=1,N
      DO 135 J=1,N
      H(I,J)=DD(I,J)
135   CONTINUE
      WRITE (6,190)
190   FORMAT(5X,'P-MATRIX',/)
      WRITE(6,200) ((E(I,J),J=1,N),I=1,N)
      WRITE (6,195)
195   FORMAT(5X,'QB-MATRIX',/)
      WRITE(6,200) ((H(I,J),J=1,N),I=1,N)
200   FORMAT(6E12.4)
      RETURN
130   CONTINUE
      WRITE(6,140)  MK ,EPS
      STOP
140   FORMAT(1X,'MATRIX EXPONENTIAL FAILED TO CONVERGE AFTER  ',I4,
     1' ITERATIONS',/,1X,'CONVERGENCE FACTOR',E12.4)
      END
******SUBROUTINE ROOTS**********************************************
********************************************************************
      SUBROUTINE ROOTS
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16  ZZ
      COMMON/SYST/A(40,40),B(40,40),C(40,40),D(40,40)
      COMMON/DIM/N,M,NR,NKS,EPS
      COMMON/DIM1/DT
      DIMENSION W(80),ZZ(40,40),WK(3200)
```

```
      DIMENSION XX(40,40),RZ(3200)
      EQUIVALENCE (ZZ(1,1), RZ(1))
      IJOB=2
      IZ=40
      IA=40
      DO 10 I=1,3200
      WK(I)=0.0
10    CONTINUE
      DO 15 I=1,80
      W(I)=0.0
15    CONTINUE
      DO 20 I=1,40
      DO 20 J=1,40
      ZZ(I,J)=0.0
20    CONTINUE
      DO 25 I=1,N
      DO 25 J=1,N
      XX(I,J)=A(I,J)
25    CONTINUE
      DO 4 I = 1,N
4     WRITE(6,3)(XX(I,J), J=1,N)
3     FORMAT(6E12.4)
      CALL EIGRF(XX,N,IA,IJOB,W,RZ,IZ,WK,IER)
      WRITE(6,8)(W(I), I = 1,80)
8     FORMAT(4E12.4)
      N2=N*2
      DO 30 I=1,N2,2
      W1=W(I)
      I1=I + 1
      W2=W(I1)
      I2=(I+1)/2
30    WRITE(6,100) I2,W1,W2
      DO 50 I=1,N
      DO 40 J=1,N
      XX(1,J)=REAL(ZZ(J,I))
      XX(2,J)=DIMAG(ZZ(J,I))
40    WRITE (6,120) I,XX(1,J),XX(2,J)
50    CONTINUE
100   FORMAT(5X,'EIGENVALUES',10X,'REAL PART',
     110X,'IMAGINARY PART',/,5X,I3,12X,E12.4,10X,E12.4)
120   FORMAT(5X,'EIGENVECTORS',I5,5X,2E12.4)
130   FORMAT(5X,'IER AND PERFORMANCE INDEX',I5,10X,E12.4)
      WRITE (6,130)  IER,WK(1)
      RETURN
      END
******SUBROUTINE EXCIT(K)**********************************************
**********************************************************************
      SUBROUTINE EXCIT(K)
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/SYST/A(40,40),B(40,40),C(40,40),D(40,40)
      COMMON/STATES/X(40),Y(40),U(40),W(40)
      COMMON/DIM/N,M,NR,NKS,EPS
      COMMON/DIM1/DT
      T=DT*FLOAT(K)
      U(1)=0.0
      RETURN
      END
******SUBROUTINE UPDAT  (K)********************************************
**********************************************************************
      SUBROUTINE UPDAT(K)
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/SYST/A(40,40),B(40,40),C(40,40),D(40,40)
      COMMON/STATES/X(40),Y(40),U(40),W(40)
      COMMON/DIM/N,M,NR,NKS,EPS
      COMMON/DIM1/DT
      COMMON/ETT/E(40,40),H(40,40)
      DIMENSION XS(40),XN(40),YS(40),YN(40)
      T=DT*FLOAT(K)
      DO 10 I=1,N
      XS(I)=0.0
```

```
      XN(I)=0.0
      YS(I)=0.0
      YN(I)=0.0
      DO 10 J=1,N
      XS(I)=XS(I)+E(I,J)*X(J)
      XN(I)=XN(I)+H(I,J)*U(J)
      YS(I)=YS(I)+C(I,J)*X(J)
      YN(I)=YN(I)+D(I,J)*U(J)
10    CONTINUE
      DO 20 I=1,N
      X(I)=XS(I)+XN(I)
      Y(I)=YS(I)+YN(I)
20    CONTINUE
      RETURN
      END
******SUBROUTINE POUT(K)*********************************************
*********************************************************************
      SUBROUTINE POUT(K)
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON/SYST/A(40,40),B(40,40),C(40,40),D(40,40)
      COMMON/STATES/X(40),Y(40),U(40),W(40)
      COMMON/DIM/N,M,NR,NKS,EPS
      COMMON/DIM1/DT
      T=FLOAT(K)*DT
      IF(K.EQ.1)  WRITE (6,110)
      WRITE (6,100)   T,Y(1),Y(2),Y(3),Y(4)
100   FORMAT(5X,F10.2,5X,4E12.4)
110   FORMAT('TIME',5X,'X1',5X,'X2',5X,'X3',5X,'X4',5X,'U1'
     1,10X,'U2',/)
      RETURN
      END
******SUBROUTINE OPTIMA**********************************************
*            THIS SUBROUTINE SETS UP THE SYSTEM AND ADJOINT EQUATIONS    *
*            MATRICES AS                                                  *
*                             A   -(B(R-1)BT)                             *
*                  SS =                                                   *
*                            -Q      -AT                                  *
*            AND FINDS THE EIGENVALUES /EIGENVECTORS OF SS.               *
*            COLLECTING THE STABLE VECTORS AS IN POTTERS METHOD           *
*            AND PARTITIONING,RESULTS IN THE SOLUTION OF THE              *
*            RICCATI EQUATION FOR THE OPTIMUM STATE FEEDBACK              *
*            GAINS.THIS ROUTINE LIMITS A(24,24).                          *
*********************************************************************
      SUBROUTINE OPTIMA
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 ZZ,W12,W22
      COMMON/SYST/A(40,40),B(40,40),C(40,40),D(40,40)
      COMMON/OPTIM/Q(24,24),R(24,24)
      COMMON/DIM/N,M,NR,NKS,EPS
      COMMON/DIM1/DT
      DIMENSION  SS(48,48),TEMP(24,24),ZZ(48,48),W(96),WK(2400)
      DIMENSION  W12(24,24),W22(24,24),RZ(4608)
      EQUIVALENCE (ZZ(1,1),RZ(1))
      IA=48
      IZ=48
      IJOB=1
      N2=2*N
      N4=2*N2
      N21=N2+1
      DO 1 I=1,N2
      DO 1 J=1,N2
1     SS(I,J)=0.0
      DO 5 I=1,N
      DO 5 J=1,N
5     TEMP(I,J)=0.0
C     WRITE(6,140)   ((R(I,J),J=1,NR),I=1,NR)
      NDIM1=24
      NDIM2=48
      CALL INVERT(R,DET,NR,NDIM1,NDIM2)
C     WRITE(6,150)  ((R(I,J),J=1,NR),I=1,NR)
```

```
      DO 10 I=1,NR
      DO 10 J=1,N
      DO 10 K=1,NR
10    TEMP(I,J)=TEMP(I,J)+R(I,K)*B(J,K)
      DO 20 I=1,N
      DO 20 J=1,N
      JJ=J+N
      DO 20 K=1,NR
20    SS(I,JJ)=SS(I,JJ)-B(I,K)*TEMP(K,J)
      DO 30 I=1,N
      DO 30 J=1,N
30    SS(I,J)=A(I,J)
      DO 40 I=1,N
      DO 40 J=1,N
      II=I+N
      JJ=J+N
      SS(II,JJ)=-A(J,I)
40    SS(II,J)=-Q(I,J)
      CALL EIGRF(SS,N2,IA,IJOB,W,RZ,IZ,WK,IER)
      WRITE(6,90)
      DO 50 I=1,N4,2
      I2=(I+1)/2
      W1=W(I)
      I1=I+1
      W2=W(I1)
50    WRITE(6,120)  I2,W1,W2
C     WRITE(6,100)
      DO 70 J=1,N2
      DO 60 I=1,N2
      SS(I,1)=REAL(ZZ(I,J))
      SS(I,2)=DIMAG(ZZ(I,J))
CO    WRITE (6,120) J,SS(I,1),SS(I,2)
60    CONTINUE
70    CONTINUE
C     COLLECT ALL STABLE EIGENVECTORS INTO A V-MATRIX
C     (USING SS(48,48)),PARTITION,AND SOLVE FOR THE
C     SOLUTION OF THE RICCATTI EQUATION .
      J=0
      DO 210 IC=1,N4,2
      JC=(IC+1)/2
      IF(W(IC).GE.0.0)   GO TO 210
      J=J+1
      DO 200 I=1,N
      IPN=I+N
      W12(I,J)=ZZ(I,JC)
      W22(I,J)=ZZ(IPN,JC)
200   CONTINUE
210   CONTINUE
C                       INVERT COMPLEX W12(N,N)
      DO 220 I=1,N
      DO 220 J=1,N
      IPN=I+N
      JPN=J+N
      SS(I,J)=REAL(W12(I,J))
      SS(IPN,J)=DIMAG(W12(I,J))
      SS(I,JPN)=-SS(IPN,J)
      SS(IPN,JPN)=SS(I,J)
220   CONTINUE
      NDIM1=48
      NDIM2=96
      CALL INVERT(SS,DET,N2,NDIM1,NDIM2)
C                    FORM W22*(W12)-1=P
      DO 230 I=1,N
      DO 230 J=1,N
      IPN=I+N
      Q(I,J)=SS(I,J)
230   R(I,J)=SS(IPN,J)
      DO 240 I=1,N
      DO 240 J=1,N
      SS(I,J)=0.0
```

```
      DO 240 K=1,N
      SS(I,J)=SS(I,J)+REAL(W22(I,K))*Q(K,J)-DIMAG(W22(I,K))*R(K,J)
240   CONTINUE
C                    FORM GAIN MATRIX INTO THE Q ARRAY
      DO 250 I=1,NR
      DO 250 J=1,N
      Q(I,J)=0.0
      DO 250 K=1,N
      Q(I,J)=Q(I,J)+TEMP(I,K)*SS(K,J)
250   CONTINUE
C                    COMPUTE THE CLOSED LOOP A-MATRIX
      DO 260 I=1,N
      DO 260 J=1,N
      DO 260 K=1,NR
260   A(I,J)=A(I,J)-B(I,K)*Q(K,J)
      WRITE(6,270)
      DO 265 K=1,NR
265   WRITE(2,275)  (Q(K,J),J=1,N)
C     WRITE(6,280)
C     DO 285 I=1,N
C85   WRITE(6,275)   (A(I,J),J=1,N)
C     CALL ROOTS
90    FORMAT(5X,'EIGENVALUES-SYSTEM+ADJOINT-')
100   FORMAT(5X, 'EIGENVECTORS  RE/IMAG')
120   FORMAT(5X,I5,10X,E12.4,10X,E12.4)
150   FORMAT(5X,'R-INVERSE',/,4E12.4)
140   FORMAT(5X,'R-MATRIX',/,4E12.4)
270   FORMAT(5X,'TOTAL STATE FEEDBACK GAIN MATRIX',/)
275   FORMAT(3E20.10)
280   FORMAT(5X,'CLOSED LOOP A-MATRIX',/)
      RETURN
      END
******SUBROUTINE INVERT*************************************************
************************************************************************
      SUBROUTINE INVERT(A,DET,N,NDIM1,NDIM2)
      IMPLICIT REAL*8(A-H,O-Z)
C        THIS ROUTINE INVERTS A SQUARE MATRIX USING
C        GAUSS ELIMINATION.THE ORIGINAL MATRIX IS DESTROYED
C        AND ITS INVERSE IS RETURNED AS 'A'.
      DIMENSION A(NDIM1,NDIM2)
      NDIGIT=30
      SUM=0.0
      DO 10 I=1,N
      DO 10 J=1,N
      SUM=SUM+ABS(A(I,J))
10    CONTINUE
      SUM=10.0**(-NDIGIT/2.)*SUM/N**2
      NP1=N+1
      NPN=N+N
      DO 20 I=1,N
      IPN=I+N
      DO 20 J=NP1,NPN
      A(I,J)=0.0
      IF (IPN.EQ.J)  A(I,J)=1.0
20    CONTINUE
C     DO 25 I=1,N
C     WRITE(6,900) (A(I,J),J=1,NPN)
25    CONTINUE
      DET=1.
      INTCH=0
      DO 90 I=1,N
      IP1=I+1
      IF (I.EQ.N)  GO TO 50
      M=I
      DO 30 J=IP1,N
      IF (ABS(A(M,I)).LT.ABS(A(J,I)))  M=J
30    CONTINUE
      IF (M.EQ.I)  GO TO 50
      INTCH=INTCH+1
      DO 40 J=1,NPN
```

```
      TEMP=A(M,J)
      A(M,J)=A(I,J)
      A(I,J)=TEMP
40    CONTINUE
50    CONTINUE
      IF(A(I,I).EQ.0.0)   GO TO 110
      IF (ABS(A(I,I)).LT.SUM)    WRITE(6,140)
      DO 60 J=IP1,NPN
      A(I,J)=A(I,J)/A(I,I)
60    CONTINUE
      DO 80 J=1,N
      IF (J.EQ.I)  GO TO 80
      DO 70 K=IP1,NPN
      A(J,K)=A(J,K)-A(J,I)*A(I,K)
70    CONTINUE
      A(J,I)=0.0
80    CONTINUE
      DET=DET*A(I,I)
90    CONTINUE
      DET=(-1)**INTCH*DET
      DO 100 I=1,N
      DO 100 J=1,N
      A(I,J)=A(I,J+N)
100   CONTINUE
C     DO 26 I=1,N
C     WRITE(6,910)  (A(I,J),J=1,NPN)
26    CONTINUE
      RETURN
110   CONTINUE
      WRITE(6,130)
      DO 120 I=1,N
      DO 120 J=1,N
      A(I,J)=1.0
120   CONTINUE
      RETURN
130   FORMAT(5X,'THE MATRIX IS SINGULAR,NO SOLUTION HAS BEEN FOUND')
140   FORMAT(5X,'THE MATRIX IS ILLCONDITIONED')
900   FORMAT(2X,'ABEF.',6E12.3)
910   FORMAT(2X,'AAFT.',6E12.4)
      END
******SUBROUTIME POUTOP****************************************************
**************************************************************************
      SUBROUTINE POUTOP
      IMPLICIT REAL*8(A-H,O-Z)
      RETURN
      END
```

LIST OF REFERENCES

1. Bane, G.L., Ferguson, J., "The Evolutionary Development of the Military Autonomous Underwater Vehicle," 5th International Symposium on Unmanned Untethered Submersible Technology, University of New Hampshire, June 22-24, 1987, Vol. 1, pp. 60-88.

2. 5th International Symposium on Unmanned Untethered Submersible Technology, University of New Hampshire, June 22-24, 1987, Symposium Proceedings.

3. Herbert, M., Thorpe, C.E., et al., "A Feasibility Study for a Long Range Autonomous Underwater Vehicle," 5th International Symposium on Unmanned Untethered Submersible Technology, University of New Hampshire, June 22-24, 1987, Symposium Proceedings.

4. NCSC Technical Memorandum 231-78, "SDV Simulator Hydrodynamic Coefficients," by N.S. Smith, J.W. Crane and D.C. Summey, June 1978.

5. International Business Machiens Program Number 5798-PXJ, "Dynamic Simulation Language/vs Language Reference Manual," June 1984.

6. NSRDC Report 2510, "Standard Equations of Motion for Submarine Simulation," by M. Gertler and G.R. Hagen, June 1967.

7. Walker, F.L., Analysis and Simulation of An Autonomous Underwater Vehicle Control System, Master's Thesis, W222137, Naval Postgraduate School, Monterey, California, June 1983.

8. Amerongen, J. Van, Udink, A.J. Ten Cate, "Model Reference Adaptive Autopilots for Ships," Automatica, Great Britain, 1975, Vol. 11, pp. 441-449.

9. Milliken, G.L., Multivariate Control of an Underwater Vehicle, Master's Thesis, MIT, Cambridge, Massachusetts, 1984.

10. Kwakernaack, H., Sivan, R., Linear Optimal Control Systems, Wiley Interscience, 1972.

11. Kaufman, H., Berry, P., "Adaptive Flight Control Using Optimal Linear Regulator Techniques," Automatica, Vol. 12, Oxford, 1976, pp. 565-576.

12. Slotine, J.J.E., Tracking Control of Non-Linear Systems Using Sliding Surfaces, Ph.D. Dissertation, MIT, Cambridge, Massachusetts, May 1983.

13. Yoerger, D.R., Slotine, J.J.E., "Robust Trajectory Control of Underwater Vehicles," I.E.E.E. Journal of Oceanic Engineering, Vol. OE-10, Number 4, October 1985.

14. Landau, I.D., "A Survey of Model Reference Adaptive Techniques--Theory and Applications," Automatica, Great Britain, Vol. 10, 1974, pp. 353-379.

15. Landau, I.D., "Model Reference Adaptive Controllers and Stochastic Self-Tuning Regulators," Transactions of A.S.M.E. Journal of Dynamic Systems Measurement and Control, New York, 1981, Vol. 103, Number 4, pp. 404-416.

16. Newman, W.S., Hogan, N., "The Optimal Control of Balanced Manipulators," Robotics: Theory and Application, A.S.M.E., New York, 1986, pp. 17 -184.

17. Slotine, J.E.E., "On the Adaptive Control of Robot Manipulators," Robotics: Theory and Applications, A.S.M.E., New York, 1986, pp. 51-56.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Chairman, Code 69Hy Mechanical Engineering Department Naval Postgraduate School Monterey, California 93943-5004	5
4.	Professor D.L. Smith, Code 69Sm Mechanical Engineering Department Naval Postgraduate School Monterey, California 93943-5004	1
5.	Professor R. McGhee, Code 52Mz Computer Science Department Naval Postgraduate School Monterey, California 93943-5004	1
6.	Professor R. Christi, Code 62Cx Electrical and Computer Engineering Department Naval Postgraduate Schoo; Monterey, California 93943-5004	1
7.	Dr. J. Crane Head, Hydrodynamics Branch NCSC Panama City, Florida 32407-5000	1
8.	Russ Weneth, Code u25 Naval Surface Weapons Center White Oak, Maryland 20910	1
9.	Paul Heckman, Code 943 Head, Undersea AI & Robotics Branch Naval Ocean System Center San Diego, California 92152	1

10. Dr. D. Milne, Code 1563 1
DTNSRDC, Carderock,
Bethesda, Maryland 20084-5000

11. RADM G. Curtis, USW PMS-350 1
Naval Sea Systems Command
Washington, D.C. 20362-5101

12. LT Relle L. Lyman, Jr., USN Code 90G 1
Naval Sea Systems Command
Washington, D.C. 20362-5101

13. LT Richard Boncal, USN 3
Raynes Neck Road, RFD 2
York, Maine 03909

14. Distinguished Professor G. Thaler, Code 62Tr 1
Electrical and Computer Engineering
Department
Naval Postgraduate School
Monterey, California 93943-5004

15. Library 1
Maine Maritime Academy
Castine, Maine

16. CDR V. Cox 1
Marine Engineering Department
Maine Maritime Academy
Castine, Maine

17. LT Vincent Squitegri 1
SMC 2431
Naval Postgraduate School
Monterey, California 93943-5000

18. LT Vince Stammetti 1
SMC 1104
Naval Postgraduate School
Monterey, California 93943-5000