SUBMARINE MOTION SIMULATION INCLUDING ZERO FORWARD SPEED AND PROPELLER RACE EFFECTS

by

ROBERT IVES HICKEY

B.S. APPLIED MATHEMATICS,
The University of New Hampshire
(1983)

SUBMITTED TO THE DEPARTMENT OF
OCEAN ENGINEERING
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

MASTER OF SCIENCE IN OCEAN ENGINEERING

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

FEBRUARY 1990

Signature of Author ______________________________
Department of Ocean Engineering
January 17, 1989

Certified by ______________________________
Professor Martin Abkowitz
Thesis Supervisor

Accepted by ______________________________
Professor A. Douglas Carmicheal
Chairman, Department Graduate Committee

!'

SUBMARINE MOTION SIMULATION INCLUDING ZERO FORWARD SPEED AND PROPELLER RACE EFFECTS

by

Robert I. Hickey

Submitted to the Department of Ocean Engineering
on January 17, 1990 in partial fulfillment of the
requirements for the Degree of Master of Science in
Ocean Engineering

ABSTRACT

A submarine motion simulation study was performed for the maneuvering characteristics of Draper Laboratories Unmanned Underwater Vehicle (UUV). The Revised Standard Submarine Equations of Motion (EOM), usually used by the Navy, were augmented to incorporate the case of very high angle of attack and the propeller race-control surface interaction.

The crossflow drag integral was modified, making the drag coefficient a function of the axial location of the hull section. The local Reynold's number was calculated based on hull diameter and crossflow velocity, taking into account lateral velocity and yaw rate. The Revised Standard EOM were used for angles of attack less than 30 degrees, the high angle of attack model for angles greater than 60 degrees. The forces and moments were interpolated to transition from the high angle of attack model to the low angle of attack model. The modified crossflow integrals predict superior trajectories during ballasting.

The influence of the propeller race on the effective angle of attack and the velocity over the control surfaces was modeled. The contribution of the control surfaces was separated from the Revised Standard coefficients Yv, Yr, Nv, and Nr and non-linear control surface coefficients were added. The response of the submarine during large rudder maneuvers is completely different due to stalling being modeled. Proximity of the propeller to the control surfaces has considerable influence on maneuvering and stability, especially during crashbacks.

Thesis Supervisor: Martin Abkowitz, Phd. title: Professor Emeritus of Ocean Engineering

ACKNOWLEDGEMENT

I would like to extend my sincere gratitude to my thesis advisor, Dr. Martin Abkowitz, for his invaluable guidance and patient support throughout the development of this thesis. Thanks are also due to my thesis reader, Dr. Douglas Carmicheal at MIT, and Dr. Thomas Tureaud, a staff member at CSDL, for his effort in the development of the text.

My appreciation is extended towards CSDL for providing this outstanding academic and professional opportunity.

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

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

NOMENCLATURE

SMALL LETTER

c : propeller race velocity at control surface

dr : deflection of rudder

dsp : deflection of port stern plane

dss : deflection of starboard stern plane

g : acceleration due to gravity 32.2 (ft/s^2)

k : non dimensional ratio $k = u_a/u_\infty$

n : revolutions per second propeller

p : roll rate

q : pitch rate

r : yaw rate

t : thrust deduction

u : surge velocity

v : sway velocity

va : velocity of advance = u(1-w)

w : normal velocity

w : wake fraction

CAPITAL LETTER

BG : distance between CB and CG

CB : center of buoyancy

CG : center of gravity

D : propeller diameter

EHP : effective horsepower

EOM : equations of motion

J : non dimensional advance ratio $J = (1\text{-}w)u/nD$

K : roll moment

L : length of submarine

L/D : length diameter ratio of hydrodynamic hull

LHS : left hand side

M : pitch moment

N : yaw moment

Q : propeller torque

Re : Reynolds number $=uL/\nu$

RHS : right hand side

SNAME : Society of Naval Architects and Marine Engineers

SOE : submerged operating envelope

T : propeller thrust

U : linear velocity $= \text{sqrt}(u^2+v^2+w^2)$

X : axial force

Y : lateral force

Z : normal force

SHP : shaft horsepower

6-DOF : six degrees of freedom

GREEK LETTER

α : angle of attack $\tan^{-1}(w/u)$

β : angle of sideslip $\sin^{-1}(-v/U)$

η : ratio of command speed to forward speed rpm_c/rpm

ν : kinematic viscosity

ρ : mass density of water

SUBSCRIPT

A_R : projected area of rudder

C_D : drag coefficient

C_L : lift coefficient

A_P : projected area of rudder in propeller race

F_{XP} : force, axial direction, propulsion

I_{DT} : drive train mass inertia

$K_T(J)$: non dimensional propeller thrust $= T/\rho n^2 D^4$

$K_Q(J)$: non dimensional propeller torque $= Q/\rho n^2 D^5$

u_A : induced local velocity of the propeller

u_∞ : induced far field velocity of the propeller

v_{FW} : velocity of sail

x_B : axial location of center of buoyancy

x_G : axial location of center of gravity

X_{SL} : axial location of center of sail

z_B : normal location of center of buoyancy

z_G : normal location of center of gravity

Z_{SL} : normal location of sail

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

The prediction of submarine trajectories has its roots based in pre-WWII maneuvering coefficient estimation of dirigibles [17]. At that time, the computing power was not available to do the tedious work of solving equations with 6 degrees of freedom over many time steps. These equations were derived from potential theory and were kept simple with 2 degrees of freedom, encompassing either the vertical or horizontal planes. The axial and roll equations could be added to observe the effects of speed or the coupling between planes.

The Standard Equations of Motion for Submarine Simulation [7], also referred to as the 2510 equations, were written in 1967. The form was conceived to be general enough to simulate the motion of all submarines. Since the hydrodynamics of a submarine do not lend themselves to easy analysis, Taylor Series expansions of the forces and moments is a natural starting point to formulate equations of motion for simulation. The forces and moments of a submarine model were measured in a tow tank and were curve fitted using Least Squares. These Taylor Series expansions were added to the rigid body dynamics to yield a six degree of freedom model.

In the period that followed, a great deal was learned about submarine maneuvering hydrodynamics in the towing tank and at sea. The 2510 equations were not capable of accurately predicting trajectories of all maneuvers. By 1979, new expressions aimed at remedying these shortcomings were developed and made public in the Revised Standard Submarine Equations of Motion [4]. The three improvements are an integral form of crossflow drag, a sail/hull interaction integral, and a propeller model based on propeller shaft dynamics using propeller curves of thrust and torque. The special features of the Revised EOM features will be covered in detail in section 2.3.

As greater accuracy was demanded in simulation prediction, mathematical expressions in the EOM were added as needed, to improve correlation with some important maneuvering characteristics. These mathematical expressions may be found in unique math models developed for a single submarine. The special curve fits of a given submarine improve the simulated trajectories when compared with real trajectories. However, they are mathematical in nature and are not based on valid hydrodynamics.

1.2 LITERATURE SEARCH

As a starting point, current literature was reviewed to establish the current level of technology. The Defense Technical Information Center (DTIC) and technical journals were searched going back 20 years using the key words; math model, simulation, and equations of motion. There was surprisingly little available oriented towards submarines. The Aerospace industry has published a great deal and is useful to estimate maneuvering coefficients. However, the simulation of missles use different conventions and is not considered in this study. Also, the Aerospace literature tends to compare the difference between aerodynamic theory and wind tunnel measurements and not the difference between simulation trajectories.

Lamb (1932) is an important background for the math model. Potential theory is a powerful technique to determine the added mass terms and subtle cross coupling between planes of motion. Symmetry considerations play an important part in simplifying the equations. Noble (1969) predicts the trajectories of 4 vessels using a simplified math model. Different coefficients are used for each vessel and the validity of the trajectories produced is discussed. Romero (1972) primarily emphasizes the implementation of the math models, taking the control engineers' perspective. However, the math model used was pre-Revised Standard. Smitt and Chislett (1975) address the merits of Large Amplitude Planar Motions Mechanism for measuring the stability derivatives as well as maneuvering coefficients for surface ships. These coefficients include third order terms and were used in simulations to verify that accurate measurements can be made using a Large Amplitude Planar Motions Mechanism. Pretty and Hookway (1979) and Flax (1979) discuss the origins and differences between math models for airships. Mendenhall (1985) is breaking new ground in submarine motion simulation. Using rational flow modelling, an axial marching scheme is used to predict forces and moments. This technique is extremely computer intensive and not directly applicable in this study. Goodman et al (1987) apply PMM technology to the flow of high angle of attack static and dynamic stability characteristics. Humphreys (1988) addresses the impact of speed-varying hydrodynamic coefficients on maneuvering performance. Several sets of coefficients were estimated based on speed and the difference between trajectories discussed. Larson (1988) focuses on the influence each coefficient has on the trajectory produced. The relative importance of each coefficient is expressed as an error relative to the unchanged math model. The 2510 EOM and the Revised Standard EOM have covered the state of the art for submarine motion simulation when they were published.

1.3 OBJECTIVE OF PRESENT WORK

Understanding the tow-tank to math-model to simulation process, several points of weakness become apparent. While the Revised Standard EOM were intended to be valid for all maneuvers, the constraints under which the coefficients were derived, severely limits the usefulness of the model. The two points addressed by this study are the lack of fidelity of the Revised Standard EOM with respect to the very high angle of attack motion regime and to the propeller-flow to control-surface interaction.

The first objective of this work is to develop a new crossflow drag integral which uses an axially varying drag coefficient based on crossflow Reynold's number. The Crossflow drag of the control surfaces is represented separately as a function of yaw rate and lateral velocity. Ballasting is a maneuver that is difficult to predict trajectories for. Adding a new crossflow integral will model this motion regime.

The second objective is to modify the existing EOM by separating the effect of the control surfaces with new coefficients. Control surfaces are normally represented with a single linear term as a function of forward speed and deflection. Instead, they are replaced with nonlinear terms as a function of the propeller race velocity and the effective angle of attack. The proximity of the propeller to the control surface and the percentage of the control surface in the propeller race is important when backing the propeller. The flow from the propeller race may cancel the flow over that portion of the control surface, zeroing its contribution to the maneuver and staiblity. This approach was developed for surface ship motion simulation [2].

Augmenting the Revised Standard EOM with terms expressly addressing the very high angle of attack crossflow and addressing the propeller race velocity will greatly improve a submarine motion simulation. The augmented Revised Standard EOM are intuitively satisfying as are the 2510 EOM when Propeller, Crossflow and Vortex models were added. The current trend of developing custom math models that ignore hydrodynamics is unnecessary. By adding these two expressions, the range of applicability is expanded. Before and after trajectories are presented to compare the math models.

CHAPTER 2

MOTION SIMULATION

2.1 WHAT IS SIMULATION

Submarine Motion Simulation will be defined for the sake of this study to be a math model of a submarine representing six degrees of freedom (6-DOF) coded on a computer for the purpose of predicting trajectories. It is open loop, ie no automatic control inputs being fed back. The trajectories are realistic when compared with the real submarine provided that the inputs of propeller rpm, control surface deflections, and ballast changes are realistic.

2.2 USES OF SIMULATION

A motion simulation may be used to determine the best response to a dive jam. The submerged operating envelope (SOE) identifies the safe operating domain in terms of depth, speed, and control surface deflections. The boundaries of the SOE are dictated by the recovery limits for stern plane jams. These limits are based on motion simulations for a particular boat using all pertinent systems affecting the recovery of the boat such as propulsion characteristics, ballast system, boat orientation, and control surface deflection.

A comprehensive motion simulation is also needed to accurately model the dynamics and develop appropriate responses to the sternplane jam. For example, a correct response to a stern plane jam may be to put the rudder hard over to induce snap roll. This changes the plane of motion towards the horizontal reducing the depth loss. At the same time, the propeller rpm may be slowed or reversed to slow the boat as fast as possible. The forward ballast may be blown to get a pitch up moment. The sequence of events that best recovers from a jam is determined from systematic permutations of control inputs [6].

The use of autopilots and control algorithms are based on high fidelity math models able to accurately predict the trajectories of a submarine. A controls engineer linearises the math model about some operating point and optimizes gains using a motion simulation. In this way, an unstable vehicle may be turned into a high performance vehicle, provided that it is not overly unstable.

Simulations are also used for ship pilot training. A motion simulation that has been tuned to real time is interfaced with a mock up of a real control station. A trainee inputs control commands using video feedback as a reference and flies the submarine. Harbor pilots are an example of highly skilled people who benefit from motion simulation training.

2.3 PRESENTATION AND SPECIAL FEATURES OF REVISED STANDARD SUBMARINE EQUATIONS OF MOTION (EOM)

2.3.1 REVISED STANDARD EOM

This study builds on the special features documented in the Revised Standard EOM. While the Vortex integral and the effects of the sail trailing vortex are not considered in this study, the crossflow and propeller model are improved. The Crossflow integral is changed to accommodate the very high angle of attack motion regime. The propeller model variables and dynamics are needed to implement the propeller race effects. The Revised Standard

EOM are included in Appendix A for reference, however aspects of the special features are covered below.

2.3.2 CROSS FLOW

This crossflow integral uses strip theory where a 3-dimensional body is approximated with many sections of a 2-dimensional body. The drag coefficient C_D is assumed to be constant over the whole body and is determined from curve fits of the towing tank data.

$$Y = \frac{\rho}{2} C_D \int v(x) h(x) \sqrt{v^2(x) + w^2(x)}\, dx$$

$$Z = \frac{\rho}{2} C_D \int w(x) b(x) \sqrt{v^2(x) + w^2(x)}\, dx$$

$$M = \frac{\rho}{2} C_D \int X w(x) b(x) \sqrt{v^2(x) + w^2(x)}\, dx$$

$$N = \frac{\rho}{2} C_D \int X v(x) h(x) \sqrt{v^2(x) + w^2(x)}\, dx$$

Local crossflow velocities are calculated at each station by the following.

$$v(x) = v + x_{ref} r$$

$$w(x) = w - x_{ref} q$$

x_{ref} is the axial location on hull, r and q are angle rates, and u and v are velocities. h(x) and b(x) are offsets of the hull in the normal and lateral directions respectively, and are equal for a body of revolution. There have been math models developed that use 4 separate C_D coefficients for the crossflow integrals for better predictions.

For large, high speed submarines with diameters of 20 feet or more, the crossflow Reynolds number would be high enough to be past the critical value of 5.5 x 10^5 and therefore C_D changes only slightly. The above crossflow integrals might be satisfactory for fleet submarines, but for small vehicles, much is overlooked.

2.3.3 PROPULSION MODEL

The propulsion model is of the form

$$F_{xp} = \rho D^4 n^2 K_T(J)(1 - t) - 550 \frac{EHP}{u}$$

The first term on the RHS is propeller thrust and the second term is drag, where n is rev/sec, D is propeller diameter, and $K_T(J)$ is the nondimensional thrust as a function of J. The drag of the submarine is found from an EHP vs. speed look up table derived from model tests or analytic estimates. The advance ratio J = u(1-w)/nD is calculated each time step and is the search parameter for tables of $K_T(J)$ and $K_Q(J)$. See Figure 2-1 on page 18 and Figure 2-2 on page 19 for typical curves of K_T and K_Q versus J. The values contained in the curves are for forward speeds only, with positive or negative rpm, see [10].

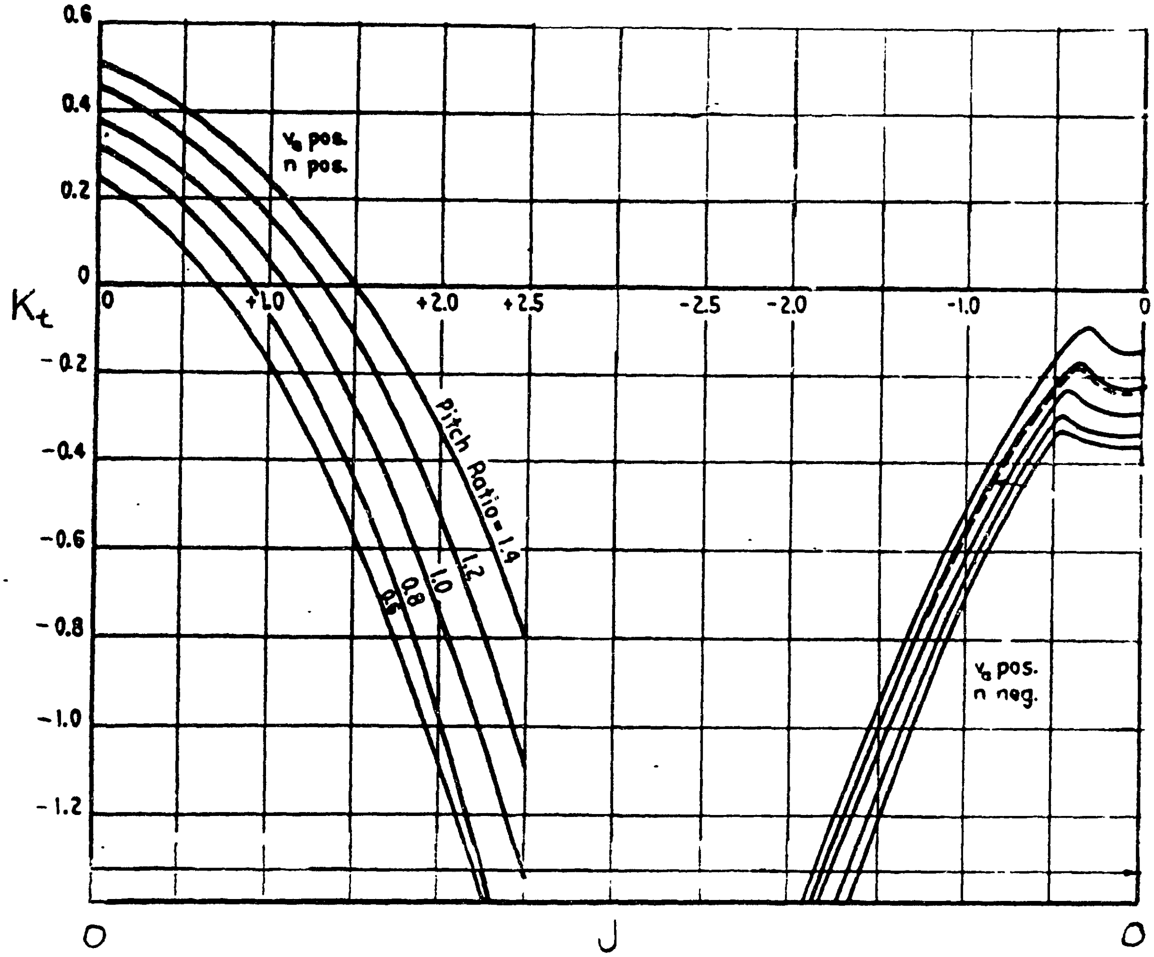


Figure 2-1. Representative Kt vs J,: for forward speed, +- rpm

Propeller torque is calculated from

$$Q_{prop} = \rho n^2 D^5 K_Q(J)$$

The difference of propeller torque and motor torque is used to calculate shaft acceleration from the relation

$$\dot{RPM} = (Q_{motor} shaft_{efficiency} - Q_{prop}) \frac{60}{2\pi I_{dt}}$$

where I_{DT} is the drive train moment of inertia. The strength of the drive train may also dictate a motor torque-rate limit.

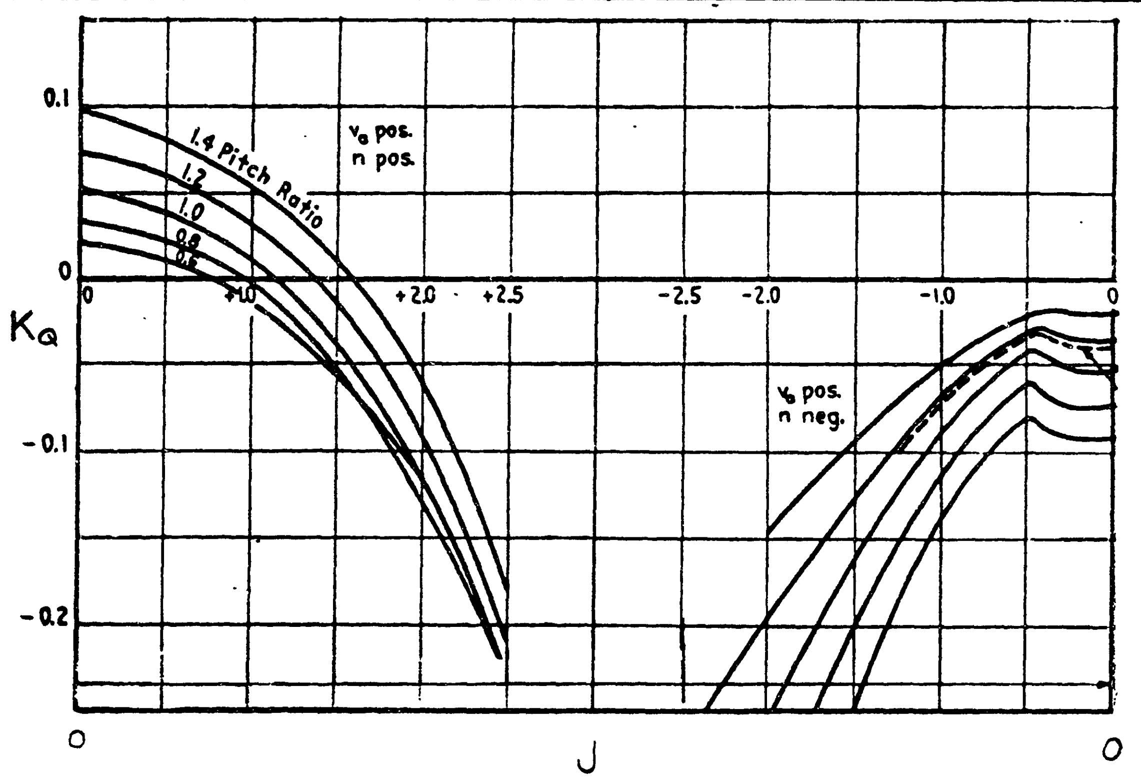


Figure 2-2. Representative Kq vs J,: for forward speed, +- rpm

The equations needed to implement the propeller race model also need the above equations and variables. With a few new equations, the propeller race control surface interaction can be modelled.

2.3.4 VORTEX

The vortex integral represents the sail trailing vortex interacting with the hull. In this expression, vorticity is considered non-dissipative and stationary with respect to the fluid. V_{FW} represents the velocity of the sail, referenced to the quarter chord and the 42% span of the sail, taking into account the yaw and roll rate, as well as lateral velocity. Each axial location of the hull can be considered to have a discrete value of vorticity proportional to V_{FW}, delayed by the time required for the shed vorticity to reach that location. The hull is moving under a stationary vorticity distribution created by the sail, see Figure 2-3 on page 21. The integral is a convolution of V_{FW} with w(x) or v(x), from the sail to a point near the end of the hull. In a tight turn, the sail trailing vortex may peel off the hull from a point farther forward than the stern. Thus, the lower limit of integration is a function of side slip angle β. The vortex equations are as follows.

$$Y = \frac{\rho}{2} C_l \int_{x1}^{x2} w(x) v_{FW}(t - \tau(x)) dx$$

$$Z = \frac{\rho}{2} C_l \int_{x1}^{x2} v(x) v_{FW}(t - \tau(x)) dx$$

$$K = K_l u v_{FW}(t - \tau_T) + \frac{\rho}{2} Z_1 C_l \int_{x1}^{x2} w(x) v_{FW}(t - \tau(x)) dx$$

$$M = \frac{\rho}{2} C_l \int_{x1}^{x2} v(x) v_{FW}(t - \tau(x)) dx$$

$$N = \frac{\rho}{2} C_l \int_{x1}^{x2} w(x) v_{FW}(t - \tau(x)) dx$$

$$v_{FW} = v + x_{SL} r + z_{SL} p$$

X_1 is the axial location of the sail, and X_2 is the stern of the boat. K_i is the interaction of the trailing vortex with the controls surfaces, so $v_{FW}(t-\tau_T)$ is delayed a bit further aft.

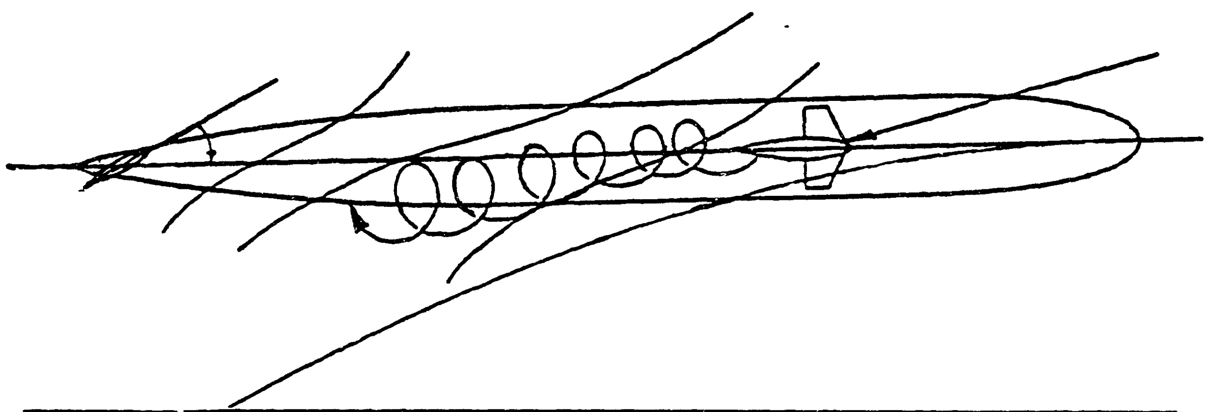


Figure 2-3. A representative sail trailing vortex

Improvements to Vortex are not covered in this study and are mentioned for completeness only. Classified literature is full of sail-trailing vortex research and reports. An improved Vortex model would therefore be the most likely addition for DTRC or some other Naval research facility to make to the Revised Standard EOM. Research indicates that at least 4 distinct vortices are shed from the sail, two at the tip and two at the root. Therefore, an obvious shortcoming of the Revised Standard Vortex Integral is that it keeps track of only one vortex and is 1-dimensional. Perhaps 4 separate vortices kept track of in 3-space would be more accurate. Mendenhall [18], is a key figure in rational flow modelling and is actively doing research in this field.

2.3.5 NOMENCLATURE AND INTERPRETATION OF TERMS

A word of caution should be raised with respect to the interpretation of maneuvering coefficients and nomenclature. Attaching physical meaning to the terms can be misleading and depends on how strictly the analyst adheres to physics. Several terms together may best correlate with hydrodynamics. For example, the following terms in the Revised Standard EOM are often curve fitted together, in an effort to get a best fit.

$$Y_v uv + Y_{v|v|R} v\sqrt{v^2 + w^2} + \frac{\rho}{2} C_D \int v(x)\sqrt{v^2(x) + w^2(x)}\, dx$$

The capital R in the above expression stands for remainder and not yaw rate to remind hydrodynamicists to be cautious. Negative drag coefficients are non-dissipative and have sometimes been used for the sake of tow-tank data fidelity, though at the expense of impossible hydrodynamics.

SNAME [25] established a convention where the subscript implies the slope or derivative with respect to that subscript. For this study, coefficient names were created. The H in the term $Y_{HV|V|}$ means high angle of attack to distinguish it from $Y_{V|V|}$. The term Y_{dre} means Y_{dr} effective angle of attack, and Y_{dre3} means $Y_{dredredre}$. The Fortran compiler used has a constraint of an 8 character variable and influenced the above choices.

2.4 IMPLEMENTATION OF EOM

The motion simulation for this study contains 30 subroutines, is written in Fortran 77 and was developed on an IBM PC. The coordinate system (x,y,z) and (ϕ,θ,ψ) are relative to body-oriented axes. Velocities in this coordinate system are denoted by (u,v,w), angular velocities by (p,q,r), forces by (X,Y,Z), and moments by (K,M,N).

The equations of motion were rearranged into a matrix format with accelerations on the left hand side (LHS). Appendix A contains the Revised Standard EOM.

$$\begin{vmatrix} m - X_{\dot u} & 0 & 0 & 0 & mz_G & -my_G \\ 0 & m - Y_{\dot v} & 0 & -mz_G - Y_{\dot p} & 0 & mx_G - Y_{\dot r} \\ 0 & 0 & m - Z_{\dot w} & my_G & -mx_G - Z_{\dot q} & 0 \\ 0 & -mz_G - K_{\dot v} & my_G & I_{xx} - K_{\dot p} & -I_{xy} & -I_{xz} - K_{\dot r} \\ mz_G & 0 & -mx_G - M_{\dot w} & -I_{xy} & I_{yy} - M_{\dot q} & -I_{yz} \\ -my_G & mx_G - N_{\dot v} & 0 & -I_{zx} - N_{\dot p} & -I_{yz} & I_{zz} - N_{\dot r} \end{vmatrix} \begin{vmatrix} \dot u \\ \dot v \\ \dot w \\ \dot p \\ \dot q \\ \dot r \end{vmatrix} = \begin{vmatrix} \sum X \\ \sum Y \\ \sum Z \\ \sum K \\ \sum M \\ \sum N \end{vmatrix}$$

The right hand side (RHS) vector represents the summation of all forces that are functions of the hydrodynamics, such as the coefficients, the crossflow and vortex integrals, as well as the hydrostatics. The 2510 EOM and the Revised Standard EOM assume the motion dynamics are linear in acceleration.

When the program starts, the Left Hand Side (LHS) or mass properties matrix is inverted once and used for every subsequent Δt. The RHS is calculated for two sets of equations, the Revised EOM and the Zero Speed EOM. If the angle of attack is between 30 and 60 degrees, interpolation between the two based on angle of attack gives a single RHS. Accelerations are solved for by calculating the product of the inverse of the LHS matrix and the RHS vector. The accelerations are then integrated twice to find velocities and positions, expressed in two frames of reference. The difference between them being the Euler-coordinate transform, producing an earth-relative set and a body-relative set see Figure 2-4 on page 24.

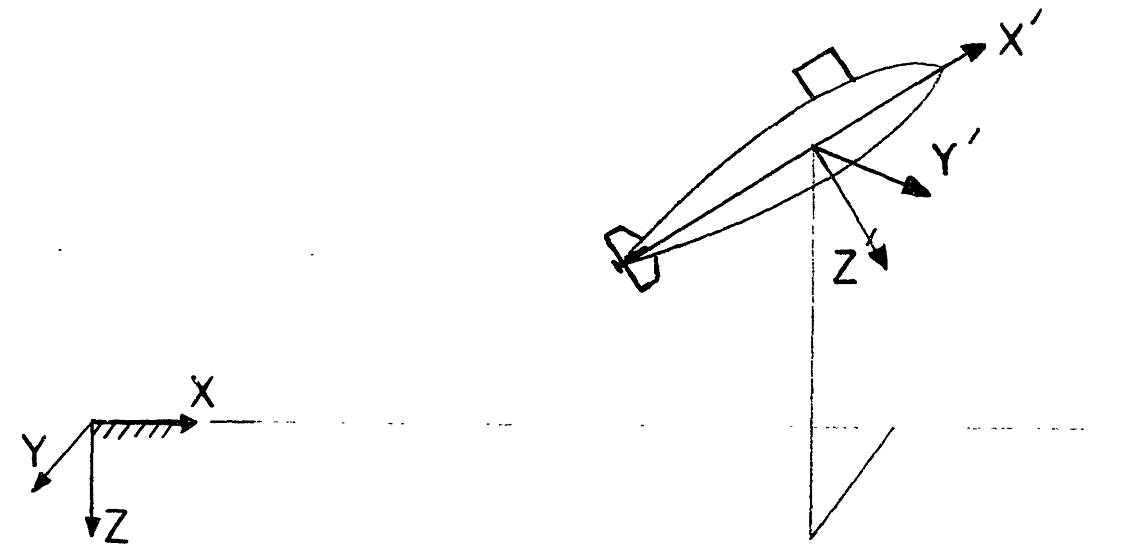


Figure 2-4. Relation Between Earth Reference Frame and Body Reference Frame

A 4th order Runge-Kutta integration routine with a Δ t of 0.05 seconds was used in vector form, as shown below.

$$Y_i^{n+1} = Y_i^n + \frac{1}{6}[k_{1,i} + 2k_{2,i} + 2k_{3,i} + k_{4,i}] i = 1....6$$

$$k_{1,i} = \Delta t f(t, x_i)$$

$$k_{2,i} = \Delta t f(t + \frac{\Delta t}{2}, x_i + \frac{1}{2} k_{1,i})$$

$$k_{3,i} = \Delta t f(t + \frac{\Delta t}{2}, x_i + \frac{1}{2} k_{2,i})$$

$$k_{4,i} = \Delta t f(t + \Delta t, x_i + k_{3,i})$$

This routine has a local truncation error of $O(\Delta x_i^5)$. and was chosen for its ease of application to the vector problem.

There are several types of power plants used on ships, each with unique characteristics. For example, an electric motor may be approximated by a constant HP input to the propeller model. In this simulation, either RPM, HP, or Torque may be specified as needed to duplicate almost any maneuver.

The input data file containing non-dimensional maneuvering coefficients, hull offsets, and propeller data consists of over 500 variables and is read in prior to each run. All of the input data and all calculated variables are echoed to an output date file which is helpful for debugging and comparison purposes.

All control inputs such as dr, dss, dsp, and rpm, are specified at particular times before a run. Control inputs are then linearly interpolated between these points and values passed to the equations of motion. For the 10-10 Zig-Zag maneuvers, the rudder was automatically flipped when the heading angle reached 10 degrees.

The RPM is integrated each time-step using a simple triangular integration routine. When the command torque and prop torque are in equilibrium, shaft acceleration stops.

Look-up tables were used for propeller, EHP, and crossflow drag data. The data taken from the tables is often clustered closely together so the previous location in the table is stored in memory to improve the search efficiency the next call.

Simpsons rule was used for crossflow integration. A very efficient routine was used where as much as possible was calculated before the run begins. Repetitive calls use the precalculated values making execution very fast.

The delay of V_{FW} is achieved thru two circular stacks in memory, where the V_{FW} and the axial distance traveled by the submarine for each Δt are stored. By adding up the Δxs in one stack, the increment can be found in the V_{FW} stack and integrated.

2.5 SHORT COMINGS OF REVISED STANDARD EOM

2.5.1 EXPERIMENTAL AND ANALYTIC BACKGROUND

The equations of motion are intended to be valid over the whole range of motion for which a submarine is expected to maneuver. Forces and moments of the submarine hull are accurately predicted at small angles of attack using only linear coefficients. However, extrapolation beyond the intended limits is risky because the curve fits used may generate unrealistic forces and moments resulting in bad trajectories at best or numerical instability at worst. At some point, the allowable degree of nonlinearity in a standard math model will be exceeded. A problem occurs because a submarine may not be tank tested at all angles of attack.

At very low forward speeds, the angle of attack can easily exceed 30 degrees. Inspection of the Revised Standard math model shows a majority of coefficients to be a function of u. As a result, with zero forward speed, the simulation doesn't respond with realistic trajectories. There are ballasting cases where a real submarine would build up speed and angles of attack drop within linear limits. However, in this case, current math models do not generate realistic trajectories, even with numerical buoyancy forces orders of magnitude larger than possible.

The results from a Large Amplitude Planar Motions Mechanism (PMM) are generally considered marginally accurate above some angle of attack, say 30 degrees. The PMM is a device used to sinusoidally oscillate a model while it is being towed in a straight line. Forces and moments are recorded at several speeds, angles of attack, and frequencies of oscillation. A Rotating Arm is used to measure forces and moments while the model is being towed in a circle, and duplicates the steady yaw rates not measurable on the PMM. There are physical limits as to the angles and frequencies measurable on this equipment as well as practical limits on the ability to resolve the effects of the hydrodynamics into coefficients.

Additionally, scale effects come into play which limit the accuracy of the measured coefficients. In the towing tank, Reynolds number matching is impossible because of the velocity and distance limitations of the towing tank. Corrections must be made to the data to adjust for known scale errors. In the end, there is some uncertainty in the value of the measured coefficients.

2.5.2 REALISTIC HYDRODYNAMICS

The limitations imposed by the PMM and subsequent data analysis suggest that purely mathematical curve fits used to represent the submarine are hydrodynamically weak. It is far better to develop a theoretically sound hydrodynamic expression and measure coefficients of lift, drag, and moment on a submarine model, than to cast aside hydrodynamic principles and use arbitrary mathematical curve fitting. The geometry of the submarine and hydrodynamic principles should determine the form of the math model.

The Revised Standard EOM do not handle very high angles of attack well. The coefficients are not meant for very high angles of attack because towing-tank tests are done at or near 0 degrees. In particular, coefficients addressing the control surfaces at 90 degrees are missing. While a single math model for all angles of attack would be simpler, a separate model focusing on the very high angles of attack that occur during ballasting should be used.

The propeller race effects are not represented in the Revised Standard EOM. Current techniques modify control surface terms by an eta (η) term, the ratio of command rpm to steady state rpm. However, these terms are functions of forward speed, and are inaccurate when forward speed is zero. The proximity of the propeller to the control surfaces is important because the propeller accelerates the flow. The percentage of the control surface in the propeller race as compared to the free stream is of fundamental importance to control authority. By backing the propeller when moving forward, the flow from the propeller race reduces and may cancel the flow over the control surface, zeroing its contribution to the maneuver and stability.

Additionally, the Revised Standard EOM do not take into account the effective angle of attack of the control surfaces. They are easily pushed into the non-linear regime due to their range and speed of deflection. The single coefficient Y_{dr} is a function of forward speed u, and deflection dr, and can't represent the force on a control surface at all angles of attack. The effective angle of attack is a function of rudder deflection dr, the yaw rate r, the lateral velocity v, and the propeller race velocity c. Further, the propeller race can change sign while backing, reversing the flow over the control surfaces. This has a dramatic influence on trajectories.

Blowing ballast and crashbacks are two maneuvers that submarines do regularly that push the limit of the simulation to model what actually takes place and thus warrant special attention. The Zero-Speed Crossflow model will cause the simulation to transition to the Revised Standard EOM in the range of moderate and small angles of attack and also realistically generate trajectories while ballasting with zero forward speed. The propeller race model will cause the submarine to go unstable during a crashback when the propeller race changes sign. Both phenomena are important and should be represented.

CHAPTER 3

ZERO FORWARD SPEED CROSS FLOW

3.1 INTRODUCTION

For small, low speed submersibles with a diameter of about 4 feet, the axial Reynolds number is affected by the angle of attack and by the yaw rate. Thus, the local C_D may vary by as much as 100%, see Figure 3-1,

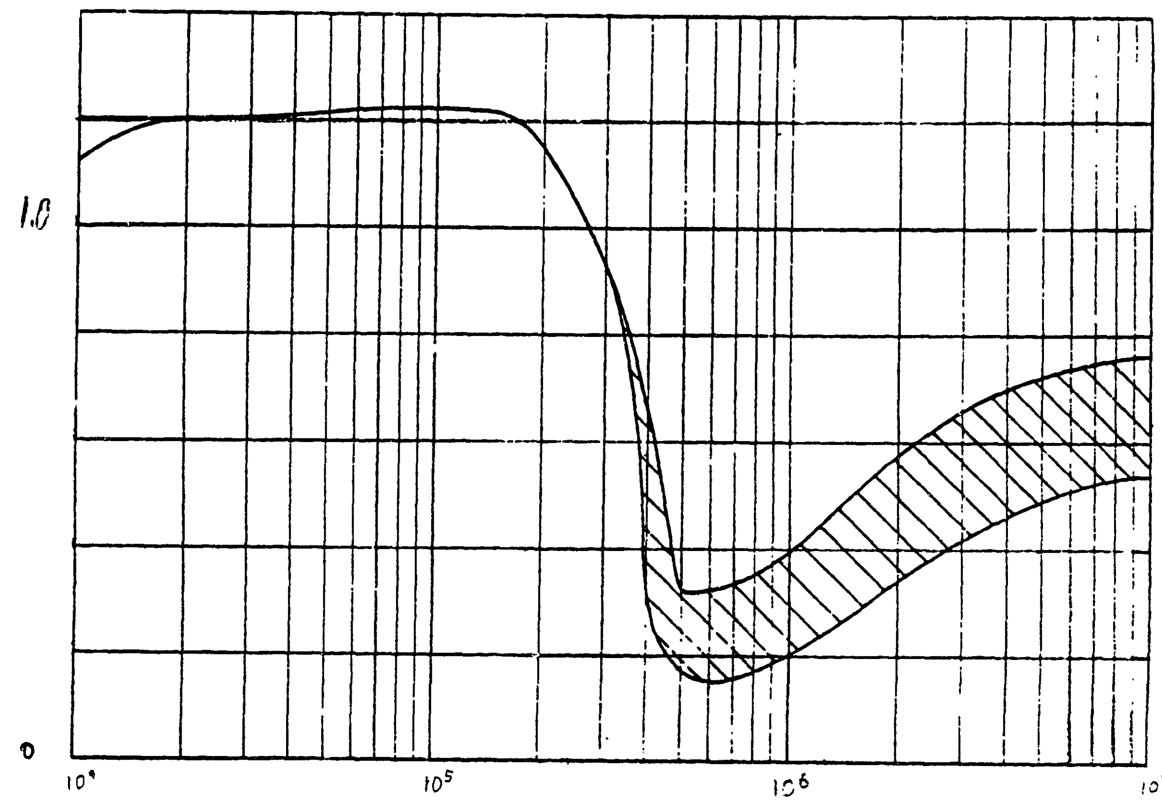


Figure 3-1. Drag Coefficient vs Reynolds number for 2-D cylinder

and have a significant influence on hydrodynamic forces and moments [12]. The mid section may be in the laminer region while the extremities reach the critical Reynolds Number.

As the velocity varies along the hull, forces and moments develop due to the pressure changes. A pressure distribution due to an angle of attack, without an angle rate, is shown in Figure 3-2 on page 31, with a 2-D section included.

Since the crossflow integral is evaluated at each time Δt, a local C_D on the hull can be evaluated with a small penalty in computation. Tabulated values of drag versus Reynolds number for a 2-D cylinder were taken from [19] and are used in a lookup table, A simulation implementing the Revised Standard EOM is easily modified to make C_D a function of axial location and Reynolds number. The required data is drag versus Reynolds number for a 2-D cylinder, which is readily available.

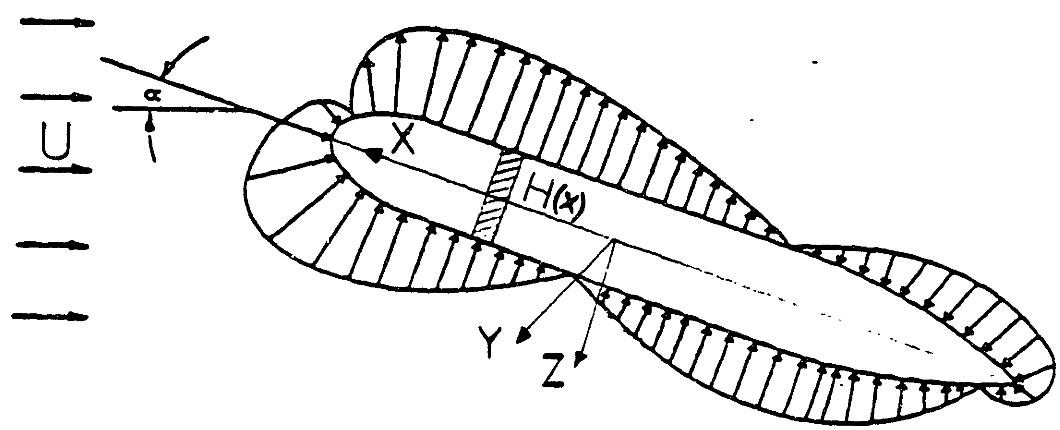


Figure 3-2. Pressure Distribution About an Unappended Submarine Hull: With Angle of Attack α.

3.2 ZERO FORWARD SPEED EOM

The significant change to the Revised Standard crossflow is that the drag coefficient is now inside the crossflow integral. The high angle of attack equations of motion are as follows:

$$Y = \frac{\rho}{2}\int_{xt}^{xb} C_D(x)v(x)h(x)\sqrt{v^2(x)+w^2(x)}\,dx + Y_{HV|V|}v_s|v_s|$$

$$Z = \frac{\rho}{2}\int_{xt}^{xb} C_D(x)w(x)b(x)\sqrt{v^2(x)+w^2(x)}\,dx + Z_{HW|W|}w_s|w_s|$$

$$K = K_{p|p|}p|p| + K_{v|v|}v|v|$$

$$M = \frac{\rho}{2}\int_{xt}^{xb} XC_D(x)w(x)b(x)\sqrt{v^2(x)+w^2(x)}\,dx + M_{HW|W|}w_s|w_s|$$

$$N = \frac{\rho}{2}\int_{xt}^{xb} XC_D(x)v(x)h(x)\sqrt{v^2(x)+w^2(x)}\,dx + N_{HV|V|}v_s|v_s|$$

where C_D is a function of axial location, and $Y_{HV|V|}$, $Z_{HW|W|}$, $M_{HW|W|}$ and $N_{HV|V|}$ are high angle of attack coefficients for the control surfaces.

$$v_s = v + x_{stern}r$$

$$w_s = w - x_{stern}q$$

The crossflow Reynolds number is defined for this study as

$$Re = \frac{\sqrt{v^2(x)+w^2(x)}\,D(x)}{\nu}$$

$$v(x) = v + x_{ref}r$$

$$w(x) = w - x_{ref}q$$

The forward limit of integration xb, is moved aft from the forward perpendicular to take into account the 3-D effects of the bow. It is set to 1/2 of a maximum hull radius aft of the forward perpendicular which is equivalent to using a hemisphere at the bow. The aft limit of integration xt, is the axial location of the propeller. The surface area aft of the propeller is very small and is assumed insignificant.

At these high angles of attack, the control surfaces are stalled. For this study, analytic estimates were made for coefficients representing the control surfaces at angles of attack of 90 degrees. The crossflow drag of these appendages was estimated using a flat plate drag coefficient of $C_D = 1.19$, based on aspect ratio [14]. An area of 1.4 square feet per fin and a moment arm of 18.5 feet was used. These are added to the zero-speed cross flow integral which covers the hull from the bow to the stern. This integral and the estimates for the coefficients are only valid at 90 degrees, plus or minus 30 degrees.

3.3 ZERO FORWARD SPEED RUNS

Two maneuvers were used to test the blending and transition between the Revised EOM and the Zero Speed EOM. Both are set up to accentuate the very high angle of attack hydrodynamics.

The following are a description of variables used in the plots. Fraction is the percentage of High-Angle EOM used to determine the RHS. RHS3 and RHS5 are the sum of the RHS for the normal and pitch equations respectively. WTINH2O is the change in weight of the submarine in water. Z CROSS and M CROSS are the values of the crossflow integrals. For the augmented model, High-Angle EOM crossflow and the Revised Standard EOM crossflow are blended when the angle of attack is between 60 and 30 degrees.

3.3.1 BUOYANT RISE MANEUVER

The first maneuver for zero speed simulation was buoyant rise. The model is initialized at zero speed steady state which can be defined as

$$u=v=w=p=q=r=rpm=\phi=\theta=dr=ds=0.$$

$$xg = xb = zb = 0.0, \text{ and } zg = 0.04166$$

Using a step-function input, the buoyancy of the boat is changed by +1% at the center of gravity. This accentuates the hydrodynamics because this buoyancy change has no static moment and causes the submarine to establish an equilibrium with a high angle of attack, see Figure 3-3 on page 36 and Figure 3-4 on page 37. The propeller was allowed to freewheel. For the first 25 seconds, the run is in the 90-60 degree angle of attack regime and uses only the High-Angle EOM. Due to the contrived nature of the run, the submarine reaches equilibrium with about a 50 degree angle of attack and illustrates the blending of 2 math models.

Note that the Revised Standard EOM do not result in an up pitch and do not gain forward speed. The crossflow drag of the fins acting on an 18 foot moment arm should result in an up pitch and is not accurately represented in the Revised Standard EOM.

If the location of the buoyancy change were forward of the CG, the static pitch angle would influence the dynamic angle of attack during ascent. For that case, which is not.

shown, the forward speed builds up until the angle of attack drops below 30 degrees and equilibrium is established in the Revised EOM.

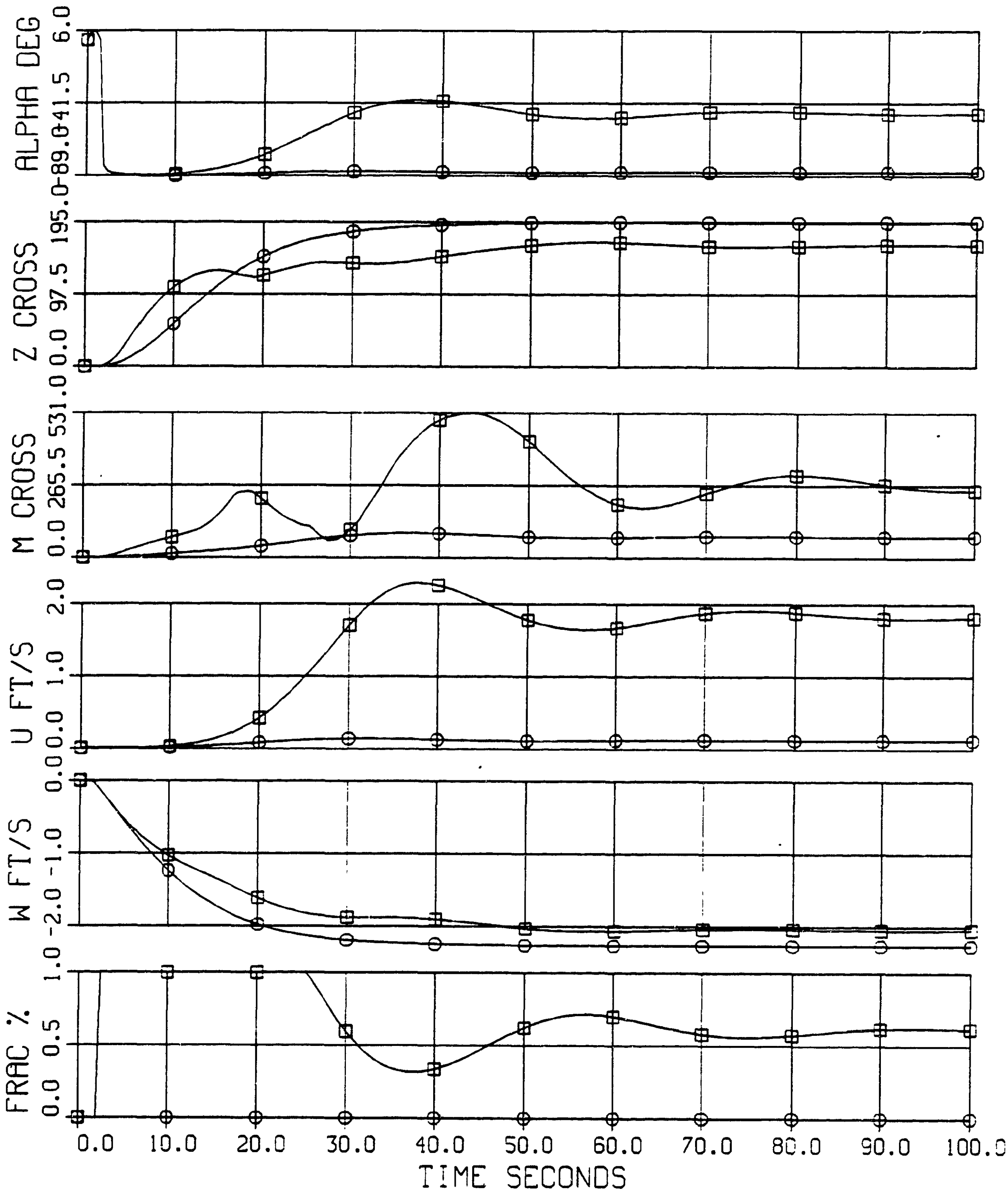


Figure 3-3. Trajectories of Buoyant Rise, Runs made on RIHSIM: □= Augmented EOM, O = Revised Standard EOM

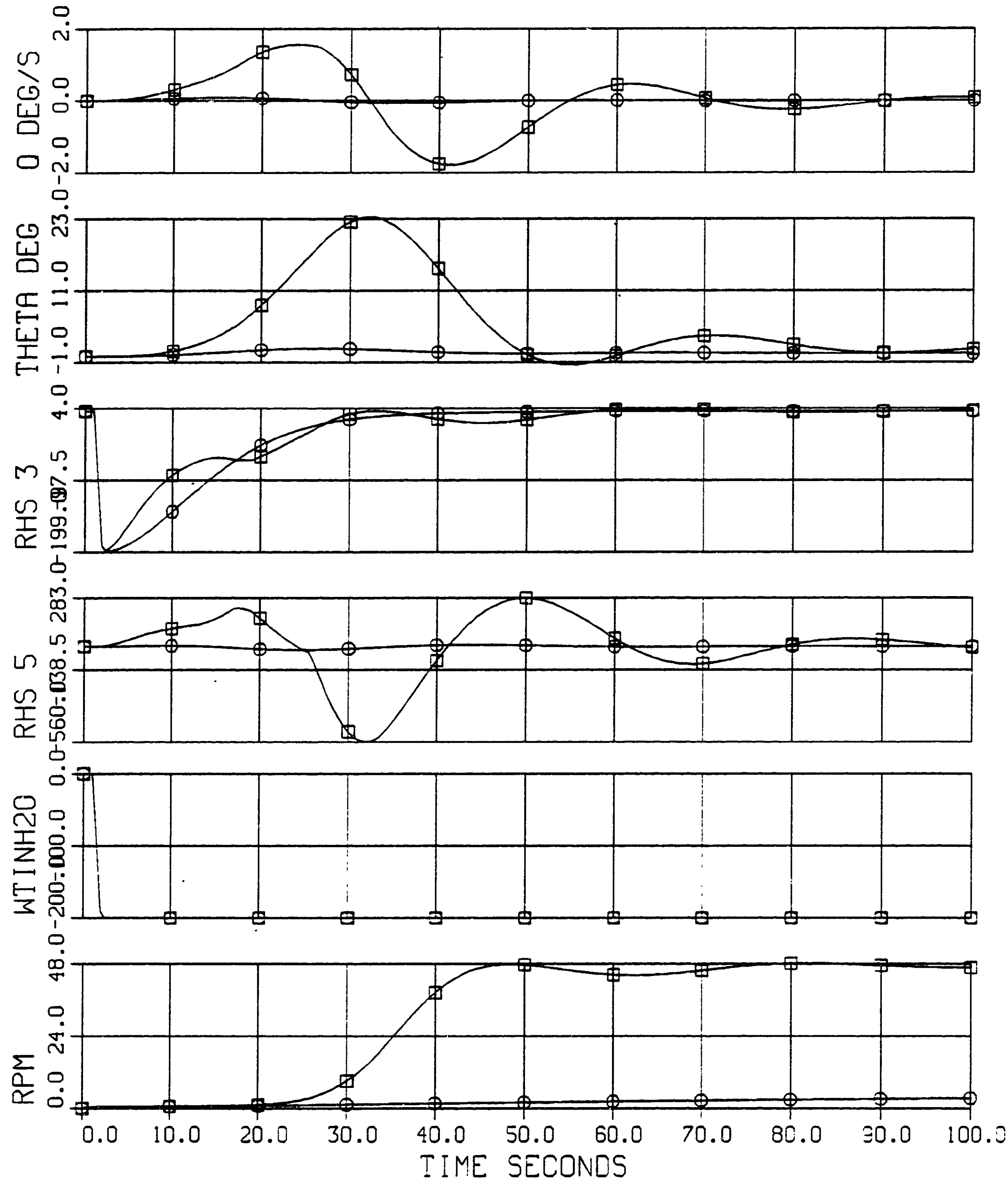


Figure 3-4. Trajectories of Buoyant Rise, Runs made on RIHSIM: □ = Augmented EOM,

○ = Revised Standard EOM

3.4 HEAVY SINK MANEUVER

The second maneuver was a heavy sink. Again, the model is initialized at zero steady state. However this time, xg is less than xb by 3 inches.

$$u=v=w=p=q=r=rpm=\phi=\theta=dr=ds=0$$

$$xg = zg = zb = 0., \text{ and } xb= 0.25.$$

Buoyancy was stepped -1/2 %. In heavy sink, the gravitational forces and to a lesser degree the hydrodynamics, cause the pitch angle change. The runs establish equilibrium in the linear regime, both trajectories converging, see Figure 3-5 on page 39 and Figure 3-6 on page 40. The High-Angle EOM come into play in the first 5 seconds only.

An interesting difference with buoyant rise occurs. Because BG = 0, there is no stabilizing roll moment and in both cases, the model spins during descent as a reaction to the propeller free-wheeling torque. This is not shown.

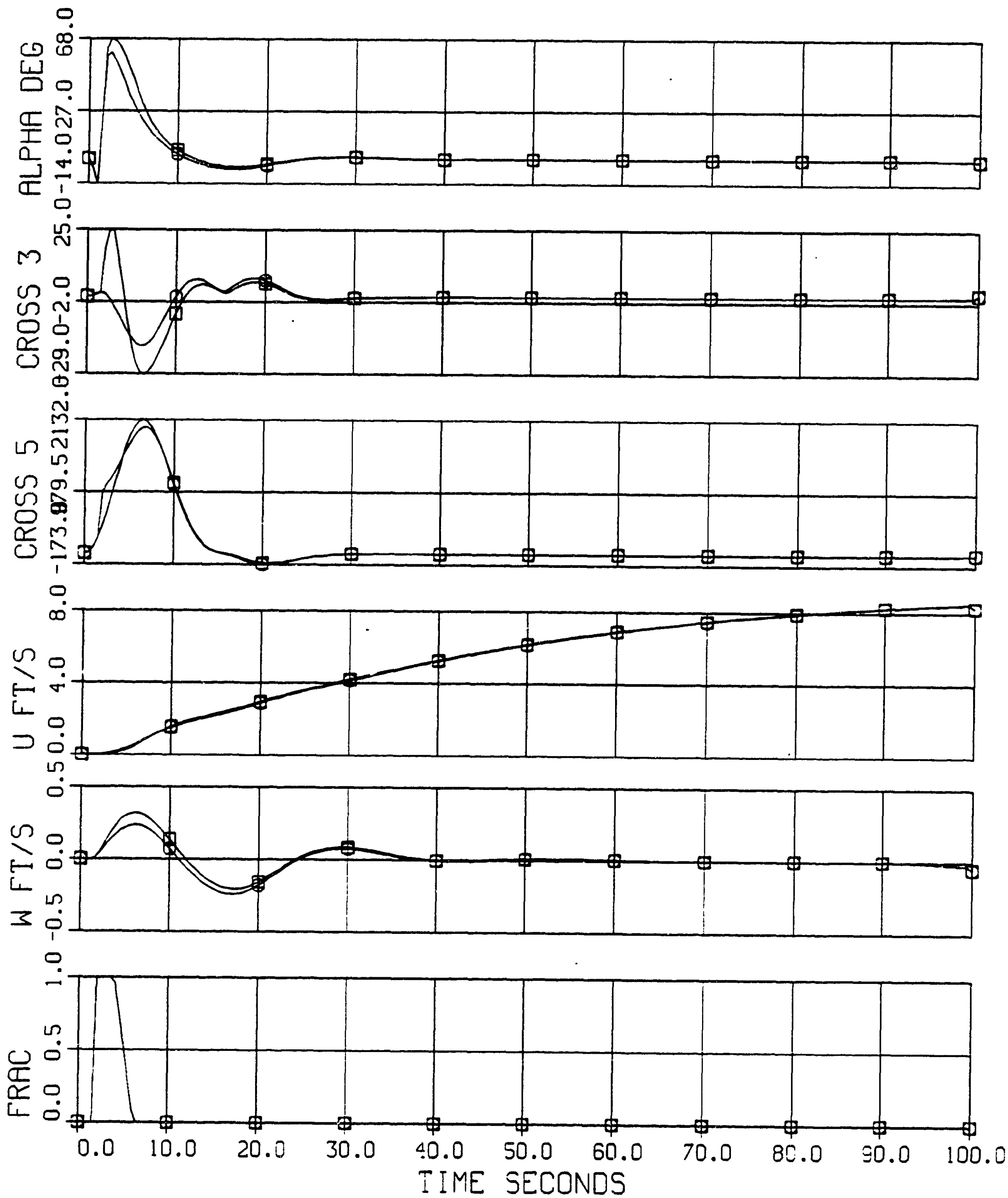


Figure 3-5. Trajectories of Heavy Sink, Runs made on RIHSIM: □ = Augmented EOM,

O = Revised Standard EOM

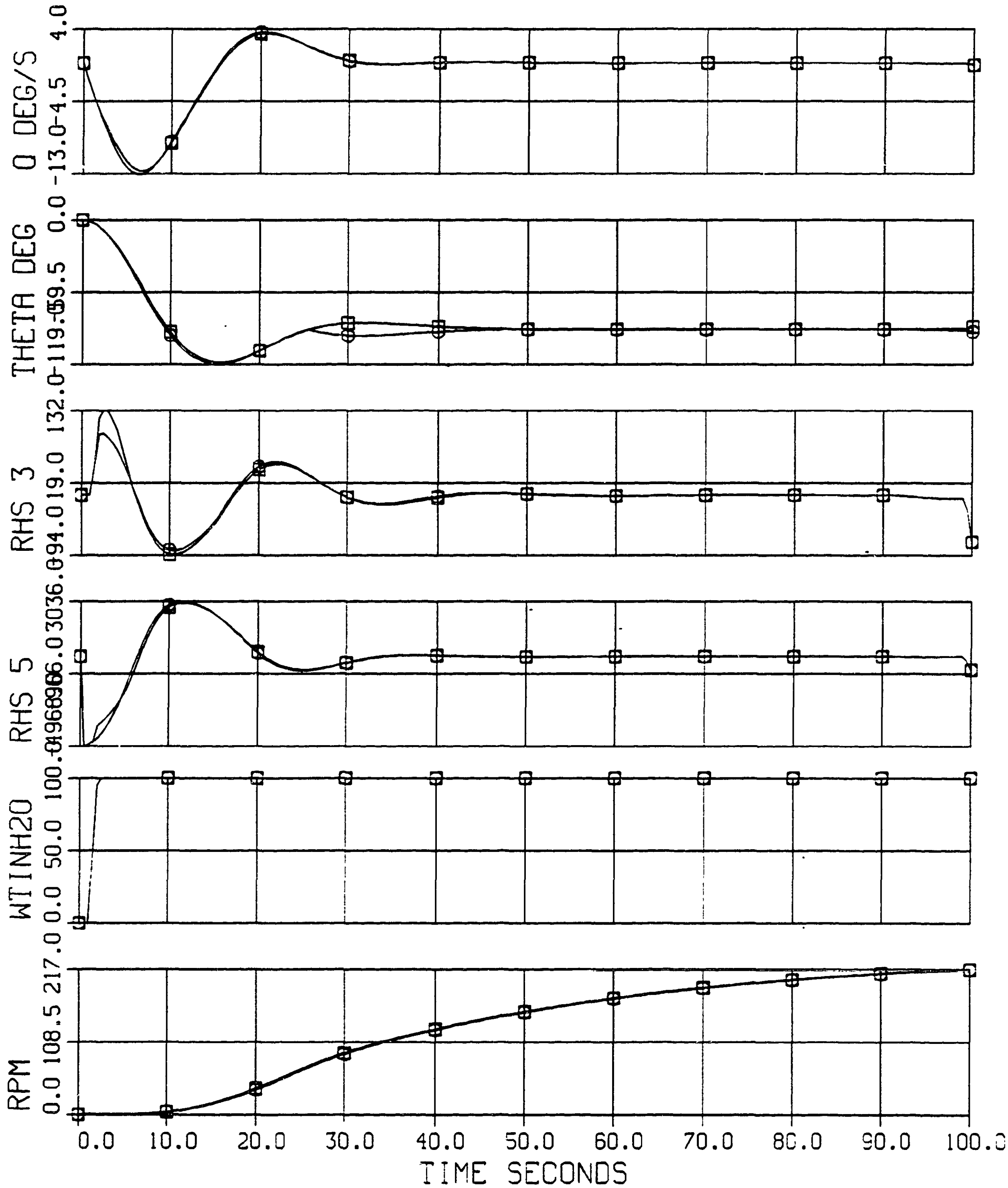


Figure 3-6. Trajectories of Heavy Sink, Runs made on RIHSIM: □ = Augmented EOM, O = Revised Standard EOM

CHAPTER 4

PROPELLER RACE EFFECTS

The Revised Standard EOM take into account the propeller race effects with eta (η) terms, the ratio of rpm$_c$/rpm. For example, $Y_{dr\eta}$, $N_{dr\eta}$, $Z_{dss\eta}$, $Z_{dsp\eta}$, $M_{dss\eta}$, and $M_{dsp\eta}$ would all be multiplied by (η - 1). These terms reflect the change in control-surface effectiveness during a maneuver due to the propeller race. However, the η terms are a function of forward speed and not the propeller race speed. As a result, at zero forward speed, the control authority isn't accurately represented. For maximum maneuverablity, control surfaces should be located in the propeller race. For a typical modern submarine, they are located forward of the propeller and are not affected much. However, when the propeller is reversed, the flow is reduced and may also reverse over the control surfaces thereby having a large effect.

From actuator disk theory, it is possible to estimate the propeller race characteristics given the forward speed, propeller rpm, diameter and performance curves $K_T(J)$, and $K_Q(J)$ vs. J.

4.1 PROPELLER RACE MODEL

$u_{a\infty}$ is the far field induced velocity of the propeller race. It is expressed as [2]

$$u_{a\infty} = -(1-w)u + \sqrt{(1-w)^2 u^2 + \frac{8K_T(J)}{\pi}(nD)^2}$$

and represents the maximum axial velocity that the propeller produces. Note the negative sign on the first term on the RHS. In this way, it is possible to change the flow over the control surfaces. When backing, K_T is negative and may cancel the forward component.

c is the actual velocity of the propeller race at the control surface as follows.

$$c = \sqrt{\frac{A_P}{A_R}[(1-w)u + ku_{a\infty}]^2 + \frac{A_R - A_P}{A_R}(1-w)^2u^2}$$

k is a coefficient, depending on the control surfaces proximity to the propeller, see Figure 4-1. The race is induced by a semi infinite tube of ring vortices and is determined by the Law of Savart. A_P is the area of the rudder in the prop race, and A_R is the total area of the rudder, see Figure 4-2 on page 43. This expression represents a weighting of the force of the rudder in the propeller race against the force of the rudder outside the propeller race, based on velocities and areas. The numerical propeller race flow reversal is accomplished by carefully controlling the signs under the square roots. When reversal occurs, the lift characteristics of the control surface are markedly different because the sharp trailing edge of the control surface is now the leading edge. The $dC_L/d\alpha$ slope is the same, but the angle of maximum lift is approximately 1/2 that in the forward direction [11], and is taken into account. Even with reduced effectiveness, flow reversal over the control surface has an incredible effect at zero forward speed. This occurs because the flow over the submarine has little hydrodynamic effect and the rudder experiences significant velocity.

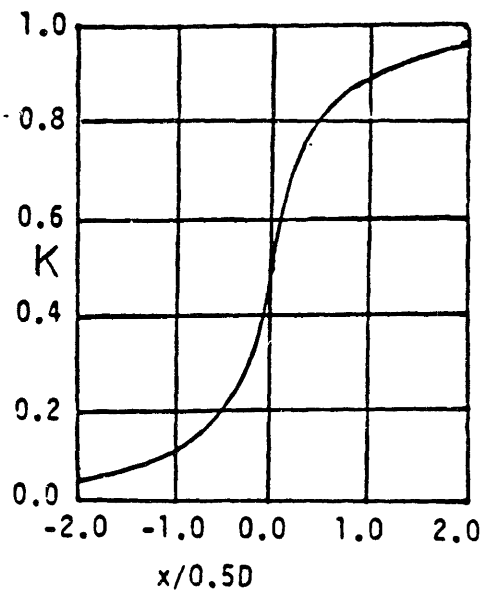


Figure 4-1. Mean Axial Velocity

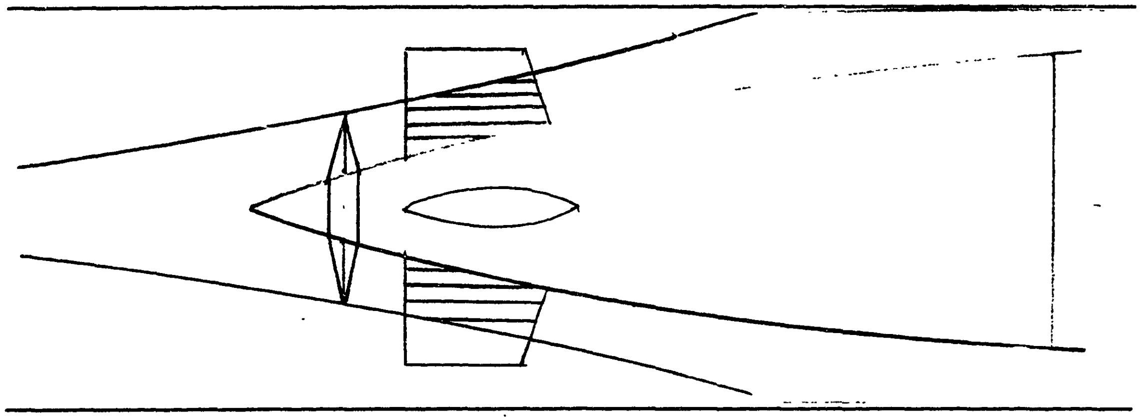


Figure 4-2. Relative Proportions of Rudder In and Out of Race: and Wake Contraction Based on Actuator Disk Theory

4.2 EFFECTIVE ANGLE OF ATTACK MODEL

The body dynamics also play a part in effective control surface deflections. The control surfaces are purposely located some distance away from the CG. The actual velocity vector of the flow is a function of yaw rate and lateral velocities. The effective angles of attack for the rudder and stern planes are represented by

$$dre = dr - \tan^{-1}\left(\frac{v}{c} + x_{stern}\frac{r}{c}\right)$$

$$dsse = dss - \tan^{-1}\left(\frac{w}{c} - x_{stern}\frac{q}{c}\right)$$

$$dspe = dsp - \tan^{-1}\left(\frac{w}{c} - x_{stern}\frac{q}{c}\right)$$

where x_{stern} is the axial location of the quarter chord of the control surface. In this way, the stalling that may occur during a zig-zag maneuver when control surface deflections are less than 20 degrees are represented.

In the Revised Standard EOM, control surface coefficients such as Y_{dr}, Z_{ds}, N_{dr}, and M_{ds} are linear. The very nonlinear nature of a low aspect ratio, all movable control surface mounted on the tapering tail cone of a body of revolution is apparent in experimental data. Near stalling, the gap at the root of the fin is quite wide and porting occurs. The lift drops significantly with higher angles of attack as the vorticity is shed at the gap instead of being carried onto the body. The fin's effective aspect ratio drops by about 2/3 as a result [9].

Based on this experimental fact, the third order control surface term Y_{dre3} was calculated. Using Y_{dr} and Y_{dre3} together, 1/3 less lift is generated at 32 degrees as compared to Y_{dr} alone. To stop the control surface lift from numerically changing sign when very high angles of attack occur, the effective angle of attack is limited to 40 degrees for the forward case, and 20 degrees when flow is reversed. In this way, the 3rd order lift coefficient Y_{dre3} isn't extrapolated out of its useful range.

4.3 PROPELLER RACE AND STABILITY

The propeller race velocity and the effective control surface deflections are an important consideration during evasive maneuvers such as a crashback. Recirculation of the propeller race may form a doublet disk which envelopes the control surfaces. During a crashback, the sub has forward speed and the propeller rpm is reversed. Depending on the sequence of events and the particular characteristics of the boat, it is possible to loose control of the boat. If the control surfaces contribute significantly to stability, then the reversal of the propeller may effectively cancel this contribution. Further, when the propeller race changes sign, the contribution of the rudder also changes sign, possibly causing instability.

Stability and turnablity work against each other. The more stable the boat is, the more resistance the boat offers to a disturbance. A simple indication of straight line stability is given by the following.

$$G_v = 1. - \frac{M_w(Z_q + m)}{Z_w M_q}$$

$$G_h = 1. - \frac{N_v(Y_r - m)}{Y_v N_r}$$

The mass and stability derivatives are nondimensional. The boat is stable if Gv is greater than 0, and unstable if less than 0. A negative value of the propeller race destablizes the boat by changing the sign of the contribution of the rudder. The effect of the rudder is present in the above expressions but not as a function of the propeller race. The status of the propulsion system has a large impact on the stability of a submarine.

According to potential theory, an unappended body of revolution with an angle of attack will experience Munk's moment. Two stagnation points acting on significant moment arms cause it to yaw and makes up a substantial part of N_v and M_w. Thus, during a steady state maneuver, the all-moveable rudder is restraining the hull from yawing faster. In this way, a rudder with +10 degrees deflection may actually experience a - 10 degree angle of attack. For submarines that are neutrally stable or slightly unstable, reserve control authority can be negated by over deflecting the control surface causing stall.

Also, during crashback, the submarine may not slow down as fast as desired due to the doublet disk changing the effective wake of the hull. The stopping distance can be greater than for a freewheeling propeller. However, this phenomena isn't normally modeled because the propeller curves are traditionally measured on a propeller boat or in a propeller tunnel to guarantee a uniform inflow. In the Revised Standard EOM, the effects represented by the thrust deduction and wake fraction are constants and are sometimes a function of sideslip angle at the propeller. However, the change in inflow to the propeller is nonlinear with loading and these constants are accurate at one operating point only. This is an area for future work and is not a part of this study.

4.4 EOM FOR RACE EFFECTS

The form of the equations for propeller-race effects removes the control surface from the existing coefficients, making it separate. The coefficients are based on body build up, the bare hull being calculated or measured, and appendages added. For example, Y_v and Y_r have the undeflected rudder Y_{dr} included in its effect. However, the significant nonlinearity of the control surfaces at high angles of attack is omitted in the Revised Standard, limiting the range of validity. The effect of the control surfaces is added back in as a function of (c-c_0), c_0 being the steady state race velocity. The third order term Y_{dre3} incorporates stalling into the EOM as indicated in [2].

$$Y = -Y_{dr}(c - c_0)v - Y_{dr}(c - c_0)r + Y_{dr}c^2dr + Y_{dre^3}c^2dre^3$$

$$Z = -Z_{ds}(c - c_0)w - Z_{ds}(c - c_0)q + Z_{ds}c^2ds + Z_{dse^3}c^2dse^3$$

$$M = -M_{ds}(c - c_0)w - M_{ds}(c - c_0)q + M_{ds}c^2ds + M_{dse^3}c^2dse^3$$

$$N = -N_{dr}(c - c_0)v - N_{dr}(c - c_0)r + N_{dr}c^2dr + N_{dre^3}c^2dre^3$$

Note that in the above equations, $Y_{dr}cv$, $Y_{dr}cr$, and $Y_{dr}c^2dr$ together are effectively $Y_{dre}c^2dre$. The equations are formed in this manner to easily remove the effect of the undeflected rudder from Y_v and Y_r in the Revised Standard EOM.

4.5 PROPELLER RACE RUNS

Three test cases were studied to demonstrate the importance of the propeller race effects. They are +- 15 degree turn, 10-10 Zig-Zag, and Crashback. For each case, there are 2 runs. One with the Revised Standard EOM being used and the other incorporating the propeller race EOM. The traces are laid one on top of the other.

The effective control surface deflection and the propeller race velocity are calculated and plotted for the Revised Standard EOM runs but are not used to calculate the trajectories.

4.5.1 +- 15 DEGREE TURN

The difference in trajectories due to the effective angle of attack being used instead of the deflection is enormous. The third order term Ydre3 and the limit of 40 degrees are dominating this maneuver. The submarine is not responding to the helm and takes 20 seconds to change direction, see Figure 4-3 on page 48 and Figure 4-4 on page 49. The Propeller Race EOM produce no roll (phi) oscillations because the submarine is in a steady turn. The speed drop for Race EOM is greater because of the dominant X_{vr} term. For the Revised EOM, v crosses zero at one instant and produces a hump in u.

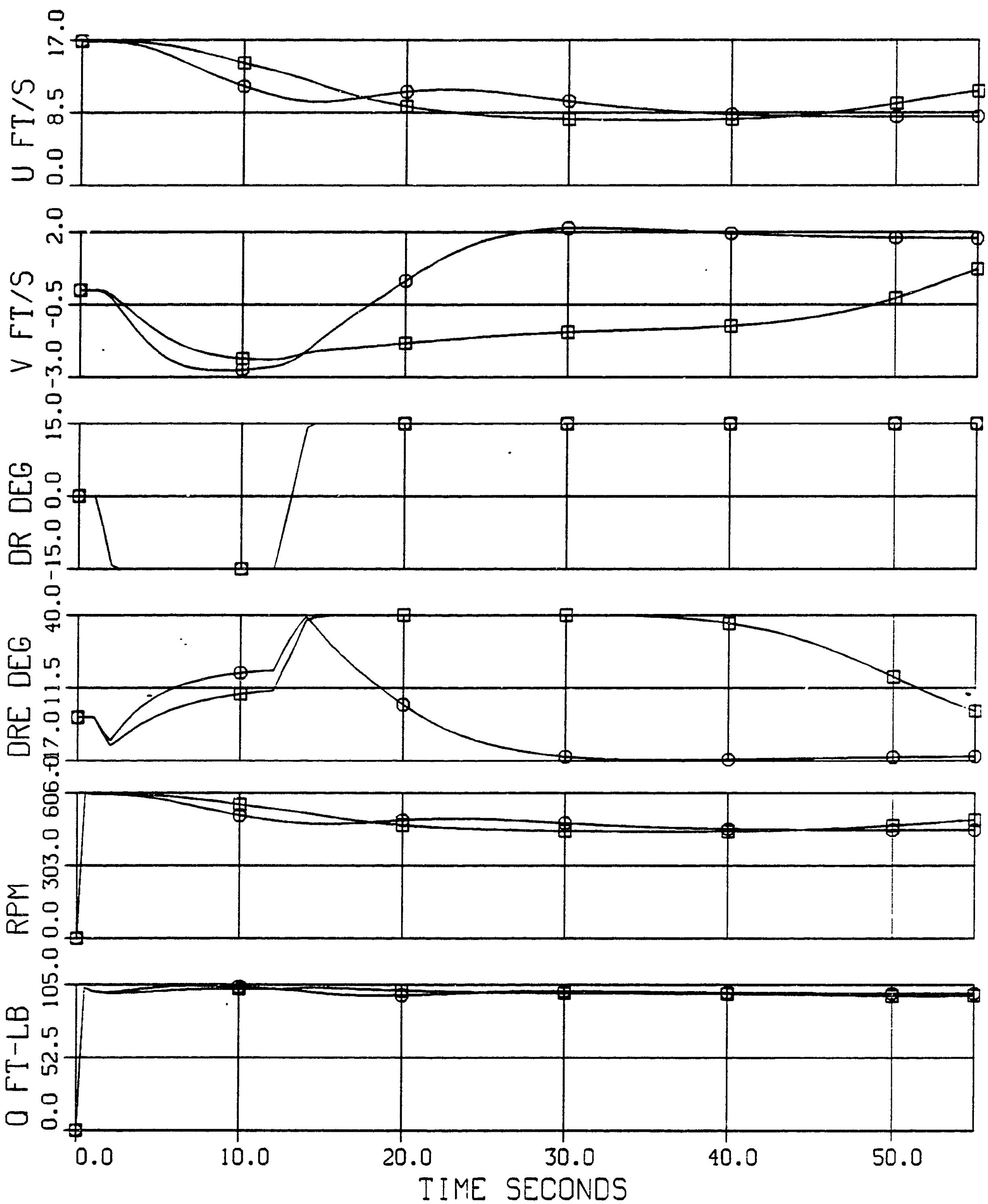


Figure 4-3. Trajectories of +-15 Degree Turn, Runs made on RIHSIM: □ = Augmented EOM, ○ = Revised Standard EOM

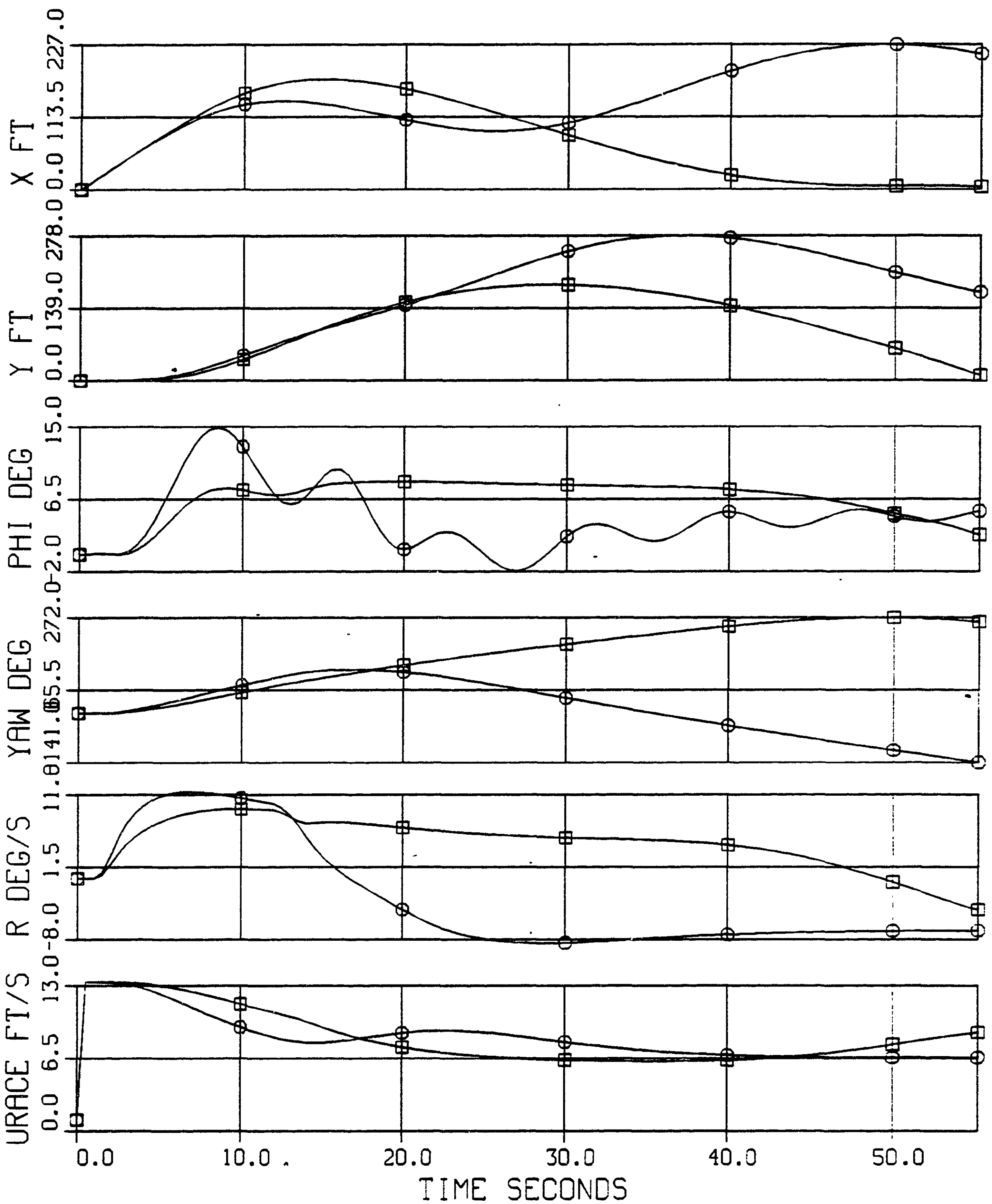


Figure 4-4. Trajectories of +-15 Degree Turn, Runs made on RIHSIM: □= Augmented EOM, O= Revised Standard EOM

4.5.2 10-10 ZIG-ZAG

The runs for this maneuver are in the linear regime and are qualitatively the same. The differences are due to a slight difference in control surface effectiveness at small angles of attack. Y_{dr} is linear and a compromise over the whole range of deflection. Y_{dr} and Y_{dre3} together could have been adjusted to be equivalent to Y_{dr} at some deflection. This would result in smaller differences between the runs with regard to amplitude but some difference between frequency and phase. See Figure 4-5 on page 51 and Figure 4-6 on page 52.

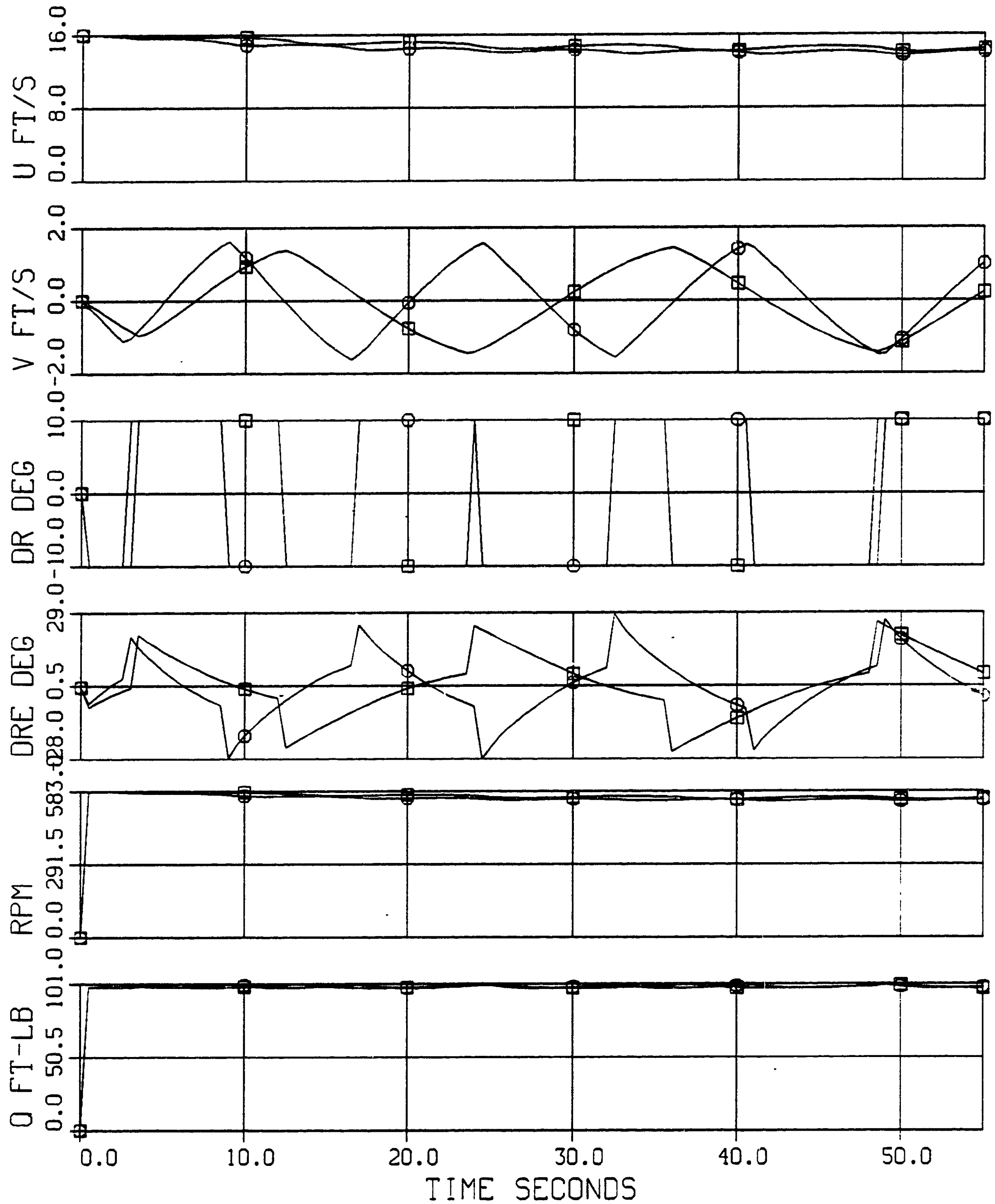


Figure 4-5. Trajectories of 10-10 Zig Zag, Runs made on RIHSIM: □= Augmented EOM, O= Revised Standard EOM

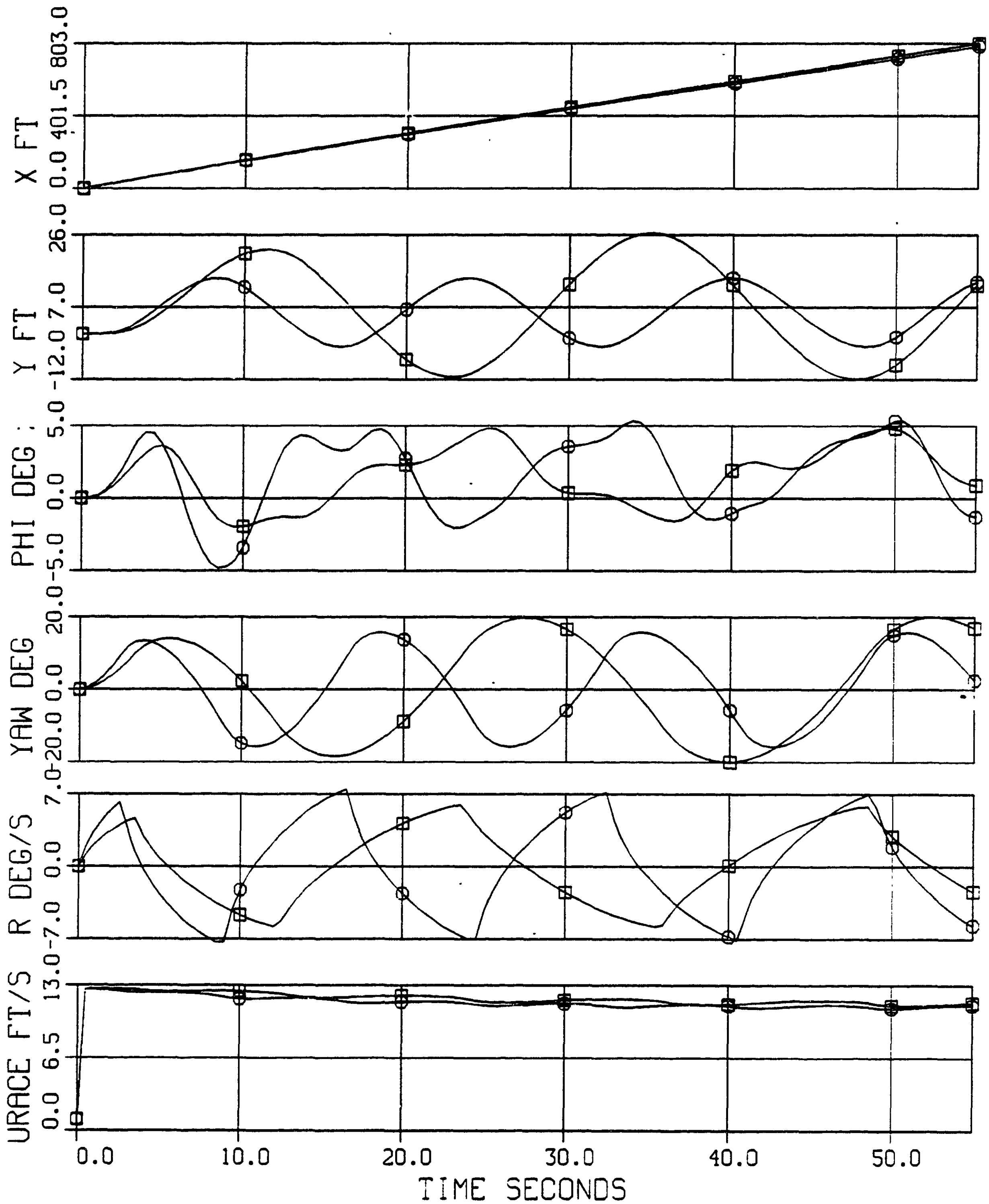


Figure 4-6. Trajectories of 10-10 Zig Zag, Runs made on RIHSIM: □ = Augmented EOM, O = Revised Standard EOM

4.5.3 CRASHBACK

The stability change during a crashback is most dramatic when forward speed is near zero. This is due to the low forward speed at which the propeller race changes sign. At this point, the only significant flow or the submarine hull is the propeller race over the control surfaces. With a low and decreasing speed, the submarine will have sluggish response, see Figure 4-7 on page 54 and Figure 4-8 on page 55. All trajectories produced for the first 40 seconds are identical. However, heading, yaw rate, lateral velocity, and distance offtrack diverge in the last 15 seconds. The reason for this is the change in sign of the effective rudder deflection caused by the flow reversal of the propeller race. Even though the control surface effectiveness is about one half its forward effectiveness, the very low speed of the submarine produces little opposing force compared to lift from the control surface.

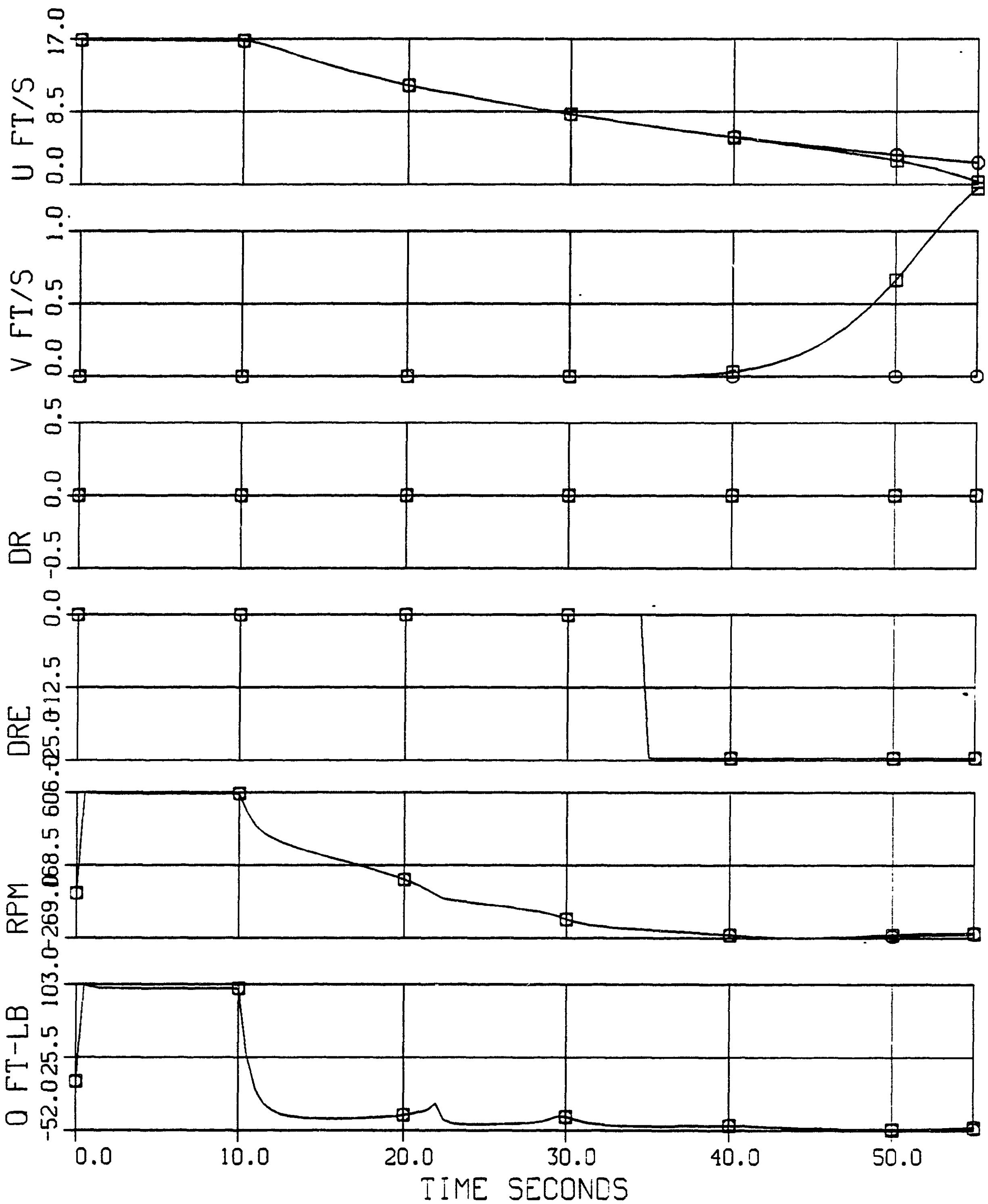


Figure 4-7. Trajectories of Crashback, Runs made on RIHSIM: □ = Augmented EOM, O = Revised Standard EOM

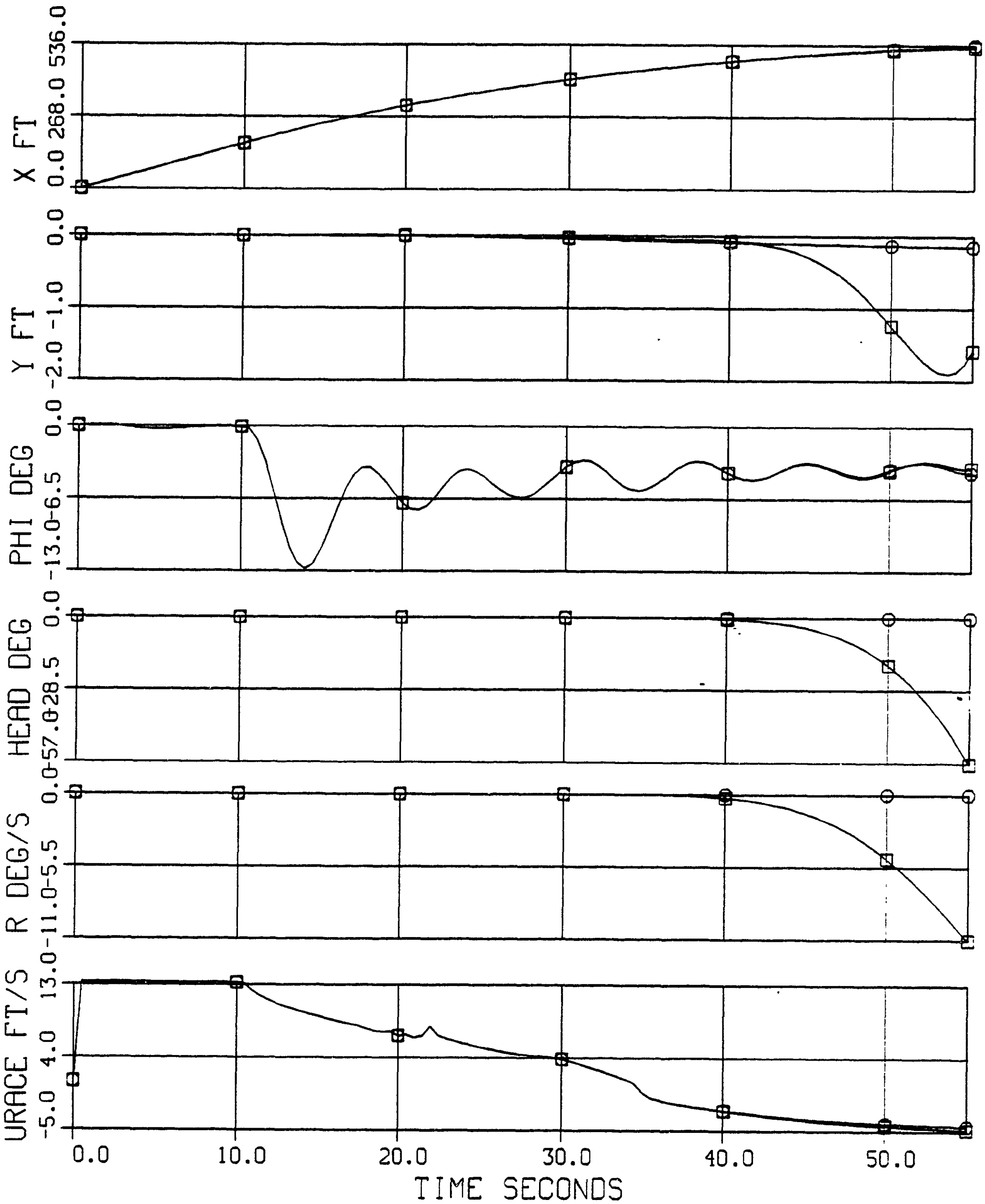


Figure 4-8. Trajectories of Crashback, Runs made on RIHSIM: □ = Augmented EOM,

O = Revised Standard EOM

CHAPTER 5

CONCLUSION

5.1 SUMMARY

From the comparison of runs, it is clearly seen that the differences between trajectories generated from the Revised Standard and trajectories generated from the augmented model are important. While the Revised Standard EOM were intended to be good for all maneuvers, the constraints under which the coefficients were derived, severely limits the usefulness of the model.

Zero speed crossflow and propeller race effects add a great deal to the math model, expanding the useful non-linear range of applicability and validity. Any comprehensive motion simulation based on the Revised Standard EOM may be augmented provided a K_T-K_Q propeller model is being used. The control surface configuration isn't normally included in a submarine math model and must be available or estimates made.

A shortcoming in the High Angle EOM occurs in the 30 to 60 degree angle of attack region. But there isn't a clear application for a model in this region of motion. The only way to sustain such angles of attack is through ballasting. In reality, either a boat would be maintained at close to 90 degrees for hovering or a significant up pitch angle would be allowed, the submarine gaining forward speed to get to the surface. The zero forward speed crossflow model will hover or undergo transition into the linear regime.

The contribution of the propeller race to the maneuvering characteristics of a submarine are fundamental. The omission of this expression in the math models of the past have no doubt limited the validity of the trajectories produced in the nonlinear regime. The effective angle of attack model with the third order term Y_{dre3} was shown to exhibit very different behavior compared with the linear Y_{dr} rudder model. While backing the propeller, the race velocity reversal was shown to cause instability. The control surfaces acting as if on the bow, their effect in the opposite direction.

5.2 RECOMMENDATIONS FOR FUTURE WORK

The UUV will be tested at sea in the Spring and Summer of 1990. At this time, on board sensors will monitor several velocities and positions. These can be used to determine whether or not the math models simulated in this study are accurate and represent the true trajectories. No simulation should be used without comparing the output to the real world.

For the sake of simplicity, the area of the propeller race on the rudder was kept constant, based on continuity. Further fidelity would be gained by changing the respective areas with propeller loading.

For a submarine with a sail, new vortex integrals based on simplified rational flow modelling would be useful.

Also, a dynamically varying wake fraction and thrust deduction would improve fidelity, especially during crashback.

LIST OF REFERENCES

1. Abkowitz, Martin., *Hydrodynamics of Backing Propellers.*, Class notes from 13.08 Motion and Control of Marine Vehicles, Massachsetts Institute of Technology.

2. Abkowitz, Martin., *Measurement of Hydrodynamic Characteristics From Ship Maneuvering Trials by System Identification,*, SNAME Transactions, Vol. 88, 1980.

3. *Fundamentals of Submarine Hydrodynamics, Motion and Control*, General Dynamics, Electric Boat Division, Naval Architecture Department, NAVSHIPS 0911-003-0610.

4. Feldman, Jerome, *DTNSRDC Revised Standard Submarine Equations of Motion*, June 1979 DTNSRDC/SPD-0393-09.

5. Flax, Alexander H., " Comment on "A Comparison of Different Forms of Dirigible Equations of Motion"", *Journal of Guidance and Control*, Vol. 2, No. 6, Dec 79.

6. Fox, Stanley., *An Algorithm for the Detection Stern Plane Jams*, Naval Coastal Systems Center (NCSC), Panama City, Fla. Phd Thesis, 1984..

7. Gertler, Morton and Hagen, Grant R., *Standard Equations of Motion for Submarine Simulation*, Naval Ship Research and Development Center Report 2510, June 1967.

8. Goodman, A., Brown, C., and Altmann, R., *An Experimental Study to Determine the Flow and the Subsonic Static and Dynamic Stability Characteristics of Aircraft Operating at High Angles of Attack*, Tracor Hydronautics, Inc., Laurel, Maryland, 20707. 1987.

9. Harper, John J., *Wind Tunnel Investigation of the Effect of a Simulated Submarine Hull on the Aerodynamic Characteristics of All-Moveable Control Surfaces Having a NACA 0015 Airfoil Section*, Daniel Guggenheim School of Aeronautics Georgia Institute of Technology, August 1959 Engineering Experiment Station Report Project A-439.

10. Hewins E F, Chase H J, Ruiz A L, *Backing Power of Gear Turbine Driven Ships,*, SNAME Transactions, 1950.

11. Hoerner, Sighard F., *Fluid-Dynamic Lift.*, Second Edition, 1985 7528 Staunton Place, N.W. Albuquerque, N.M. 87120.

12. Humphreys, Douglas E., *The Impact of Speed-Varying Hydrodynamic Coefficients on UUV Maneuvering Performance*, A.R.A.P. Division, California Research & Technology, Inc. Old Gallows Road, Vienna,Va 1988.

13. Humphreys, Douglas E., *Evaluation of the Hydrodynamic Coefficients, Transfer Functions, and 6 Degree of Freedom Maneuvering Characteristics of the Draper Laboratory's UUV.*, ARAP Report 628, November 1988.

14. Hunter, Rouse, *Fluid Mechanics for Hydraulic Engineers*, Professor Hydraulics, State University of Iowa, 1938, McGraw-Hill Book Company, New York and London, 1st Edition, 2nd Impression.

15. Lamb,Sir Horace., *Hydrodynamics*, 6th edition, Dover Publications.

16. Larsen, Kevin P., *Reduced Hydrodynamic Modeling for a Submerged Vehicle*, Naval Post Graduate School, Monterey, Ca Masters Thesis, September 1988.

17. Landweber, L. and Johnson, J., *Prediction of Dynamic Stability Derivatives of an Elongated Body of Revolution.*, DTMB Report C-359, May 1951 Declassified 10/30/70.

18. Mendenhall, M., R. and Perkins S.,C.,Jr., *Prediction of the Unsteady Hydrodynamic Characteristics of Submersible Vehicles,*, Fourth International Conference on Numerical Ship Hydrodynamics National Academy of Sciences, Washington, D.C., Sept 24-27,1985.

19. Fidler, John, E. and Smith, Charles, A., *Methods For Predicting Submersible Hydrodynamic Characteristic.*, NCSC TM-238-78, Naval Coastal Systems Center, Panama City, Fl. Neilson Engineering & Research July 1978.

20. Noble, Tom., *Unified Simulation of Marine Vehicles.*, Northrop, Vetura, TP-132 Newbury Park, Ca. April 1969.

21. Pretty J. R. and Hookway, R. O., " A Comparison of Different Forms of Dirigible Equations Of Motion", *Lighter Than Air Systems Technology Conference 1977*, AIAA 77-1179.

22. Rawson K J and Tupper E C., *Basic Ship Theory, Volume I and II.*, third edition 1984, Longman Group Limited, London and New York.

23. Romero, Edgar., *Mathematical Models and Computer Solution for the Equations of Motion of Surface Ships and Submarines, in 6 Degrees of Freedom*, Master's Thesis, June 1972, Naval Post Graguate School.

24. Smitt, L. W. and Chislett, M. S., " Large Amplitude PMM Tests and Maneuvering Predictions For a Mariner Class Vessel", *Tenth Symposium, Naval Hydrodynamics, June 24-28, 1974, Cambridge, Massachusetts*, Office of Naval Research, Department of the Navy.

25. " Nomenclature for Treating the Motion of a Submerged Body Through a Fluid", *American Towing Tank Conference*, SNAME April 1952 Reprinted March 1968.

APPENDIX A

CONDENSED REVISED STANDARD EOM

DTNSRDC/SPD-0393-09

DAVID W. TAYLOR NAVAL SHIP RESEARCH AND DEVELOPMENT CENTER

Bethesda, Maryland 20084

DTNSRDC REVISED STANDARD SUBMARINE EQUATIONS OF MOTION

by

J. Feldman

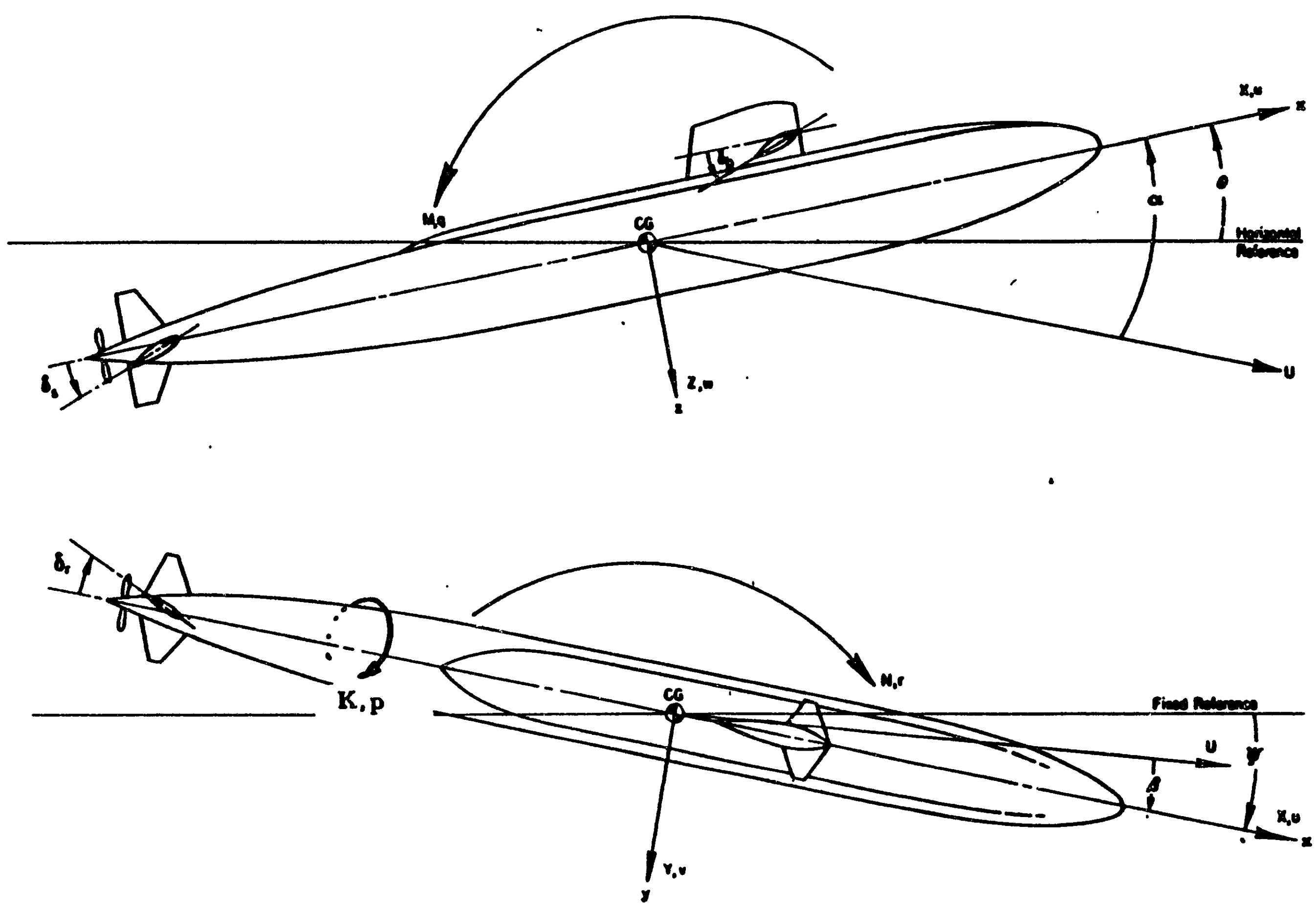
APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED

SHIP PERFORMANCE DEPARTMENT

June 1979

DTNSRDC/SPD-0393-09

DTNSRDC REVISED STANDARD SUBMARINE EQUATIONS OF MOTION



Sketch Showing Positive Directions of Axes, Angles, Velocities, Forces, and Moments

AXIAL FORCE EQUATION

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

$$+ \frac{\rho}{2}\ell^4\left[X_{qq}'\,q^2 + X_{rr}'\,r^2 + X_{rp}'\,rp\right]$$

$$+ \frac{\rho}{2}\ell^3\left[X_{\dot{u}}'\,\dot{u} + X_{vr}'\,vr + X_{wq}'\,wq\right]$$

$$+ \frac{\rho}{2}\ell^2\left[X_{vv}'\,v^2 + X_{ww}'\,w^2\right]$$

$$+ \frac{\rho}{2}\ell^2\left[X_{\delta r\delta r}'\,u^2\delta_r^{\,2} + X_{\delta s\delta s}'\,u^2\delta_s^{\,2} + X_{\delta b\delta b}'\,u^2\delta_b^{\,2}\right]$$

$$-(W-B)\sin\theta + F_{xp}$$

$$F_{xp} = \begin{cases} T_p - \text{DRAG} \\ \frac{\rho}{2}\ell^2\left[(a_i + \Delta X)\,u^2 + b_i\,Cuu_c + c_i\,C^2u_c^{\,2}\right] \end{cases}$$

where $\Delta X = \Delta X_1 + \dfrac{\Delta X_2}{(\Delta X_3 + \log_{10} u)^2}$

$$C = C_6 + (C_7 + C_8\,\Delta X)^{1/2}$$

Note: F_{xp} is represented by T_p - DRAG when propulsion characteristics are available. Otherwise the second expression for F_{xp} is used.

LATERAL FORCE EQUATION

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

$$+ \frac{\rho}{2} \ell^4 \left[Y_{\dot{r}}' \dot{r} + Y_{\dot{p}}' \dot{p} + Y_{p|p|}' p|p| + Y_{pq}' pq\right]$$

$$+ \frac{\rho}{2} \ell^3 \left[Y_r' ur + Y_p' up + Y_{\dot{v}}' \dot{v} + Y_{wp}' wp\right]$$

$$+ \frac{\rho}{2} \ell^2 \left[Y_*' u^2 + Y_v' uv + Y_{v|v|R}' v|(v^2 + w^2)^{1/2}|\right]$$

$$+ \frac{\rho}{2} \ell^2 \left[Y_{\delta r}' u^2\delta_r + Y_{\delta r\eta}' u^2\delta_r \left(\eta - \frac{1}{C}\right) C\right]$$

$$- \frac{\rho}{2} C_d \int_{\ell} h(x)\, v(x) \left\{[w(x)]^2 + [v(x)]^2\right\}^{1/2} dx$$

$$- \frac{\rho}{2} \ell \bar{C}_L \int_{x_2}^{x_1} w(x)\, \bar{v}_{FW}(t - \tau[x])\, dx$$

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

NORMAL FORCE EQUATION

$$m\left[\dot{w} - uq + vp - z_G(p^2 + q^2) + x_G(rp - \dot{q}) + y_G(rq + \dot{p})\right] =$$

$$+ \frac{\rho}{2} \ell^4 Z_{\dot{q}}' \dot{q}$$

$$+ \frac{\rho}{2} \ell^3 \left[Z_{\dot{w}}' \dot{w} + Z_q' uq + Z_{vp}' vp\right]$$

$$+ \frac{\rho}{2} \ell^2 \left[Z_*' u^2 + Z_w' uw\right]$$

$$+ \frac{\rho}{2} \ell^2 \left[Z_{|w|}' u|w| + Z_{ww}' \left|w(v^2 + w^2)^{1/2}\right|\right]$$

$$+ \frac{\rho}{2} \ell^2 \left[Z_{\delta s}' u^2\delta_s + Z_{\delta b}' u^2\delta_b + Z_{\delta s\eta}' u^2\delta_s \left(\eta - \frac{1}{C}\right) C\right]$$

$$- \frac{\rho}{2} C_d \int_\ell b(x)\, w(x) \left\{[w(x)]^2 + [v(x)]^2\right\}^{1/2} dx$$

$$+ \frac{\rho}{2} \ell \bar{C}_L \int_{x_2}^{x_1} v(x)\, \bar{v}_{FW}(t - \tau[x])\, dx$$

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

ROLLING MOMENT EQUATION

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

$$+ m\left[y_G(\dot{w} - uq + vp) - z_G(\dot{v} - wp + ur)\right] =$$

$$+ \frac{\rho}{2}\ell^5\left[K_{\dot{p}}{}'\,\dot{p} + K_{\dot{r}}{}'\,\dot{r} + K_{qr}{}'\,qr + K_{p|p|}{}'\,p|p|\right]$$

$$+ \frac{\rho}{2}\ell^4\left[K_p{}'\,up + K_r{}'\,ur + K_{\dot{v}}{}'\,\dot{v} + K_{wp}{}'\,wp\right]$$

$$+ \frac{\rho}{2}\ell^3\left[K_*{}'\,u^2 + K_{vR}{}'\,uv + K_i{}'\,uv_{FW}(t - \tau_T)\right]$$

$$+ \frac{\rho}{2}\ell^3\left[K_{\delta r}{}'\,u^2\delta_r + K_{\delta r\eta}{}'\,u^2\delta_r\left(\eta - \frac{1}{C}\right)C\right]$$

$$+ \frac{\rho}{2}\ell^3(u^2 + v_S{}^2 + w_S{}^2)\,\beta_S{}^2\left[K_{4S}{}'\sin 4\phi_S + K_{8S}{}'\sin 8\phi_S\right]$$

$$+ \frac{\rho}{2}\ell^2 z_1{}'\,\overline{C}_L\int_{x_2}^{x_1} w(x)\,\overline{v}_{FW}(t - \tau[x])\,dx$$

$$+ (y_G W - y_B B)\cos\theta\cos\phi - (z_G W - z_B B)\cos\theta\sin\phi$$

$$- Q_p$$

PITCHING MOMENT EQUATION

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

$$+ m\left[z_G(\dot{u} - vr + wq) - x_G(\dot{w} - uq + vp)\right] =$$

$$+ \frac{\rho}{2}\ell^5\left[M_{\dot{q}}'\ \dot{q} + M_{rp}'\ rp\right]$$

$$+ \frac{\rho}{2}\ell^4\left[M_{\dot{w}}'\ \dot{w} + M_q'\ uq\right]$$

$$+ \frac{\rho}{2}\ell^3\left[M_*'\ u^2 + M_w'\ uw + M_{w|w|R}'\ w\left|(v^2 + w^2)^{1/2}\right|\right]$$

$$+ \frac{\rho}{2}\ell^3\left[M_{|w|}'\ u|w| + M_{ww}'\ \left|w\ (v^2 + w^2)^{1/2}\right|\right]$$

$$+ \frac{\rho}{2}\ell^3\left[M_{\delta s}'\ u^2\delta_s + M_{\delta b}'\ u^2\delta_b + M_{\delta s\eta}'\ u^2\delta_s\left(\eta - \frac{1}{C}\right)C\right]$$

$$+ \frac{\rho}{2} C_d \int_{\ell} x\ b(x)\ w(x)\ \left\{[w(x)]^2 + [v(x)]^2\right\}^{1/2} dx$$

$$- \frac{\rho}{2}\ell\bar{C}_L \int_{x_2}^{x_1} x\ v(x)\ \bar{v}_{FW}(t - \tau[x])\ dx$$

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

YAWING MOMENT EQUATION

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

$$+ m\left[x_G(\dot{v} - wp + ur) - y_G(\dot{u} - vr + wq)\right] =$$

$$+ \frac{\rho}{2} \ell^5 \left[N_{\dot{r}}' \dot{r} + N_{\dot{p}}' \dot{p} + N_{pq}' pq\right]$$

$$+ \frac{\rho}{2} \ell^4 \left[N_p' up + N_r' ur + N_{\dot{v}}' \dot{v}\right]$$

$$+ \frac{\rho}{2} \ell^3 \left[N_*' u^2 + N_v' uv + N_{v|v|R}' v\left|(v^2 + w^2)^{1/2}\right|\right]$$

$$+ \frac{\rho}{2} \ell^3 \left[N_{\delta r}' u^2\delta_r + N_{\delta r\eta}' u^2\delta_r\left(\eta - \frac{1}{C}\right) C\right]$$

$$- \frac{\rho}{2} C_d \int_{\ell} x\, h(x)\, v(x) \left\{[w(x)]^2 + [v(x)]^2\right\}^{1/2} dx$$

$$- \frac{\rho}{2} \ell \bar{C}_L \int_{x_2}^{x_1} x\, w(x)\, \bar{v}_{FW}(t - \tau[x])\, dx$$

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

APPENDIX B

PARTIAL SOURCE CODE FOR RIHSIM

```
C ***********************************************************************
      SUBROUTINE MOTION
C ***********************************************************************
C    - THIS SUBROUTINE USES 4TH ORDER RUNGE KUTTA INTEGRATION
C    - IT CALCULATES THE 1ST & 2ND STATE DERIVATIVES
C    - USING THE INVERSE OF THE 'E' MATRIX
C    - SUBROUTINE HYDRO IMPLEMENTS DTNSRDC 2510 EQUATIONS
C                                                                       R
C    EDOT(1)   EDOT(2)   EDOT(3)   EDOT(4)   EDOT(5)   EDOT(6)          E
C       ¢         ¢         ¢         ¢         ¢         ¢        F    L
C     UDOT      VDOT      WDOT      PDOT      QDOT      RDOT       L    A
C                                                                  U    T
C    E(1)      E(2)      E(3)      E(4)      E(5)      E(6)        I    I
C       ¢         ¢         ¢         ¢         ¢         ¢        D    V
C       U         V         W         P         Q         R             E
C
C    PDOT(1)   PDOT(2)   PDOT(3)   PDOT(4)   PDOT(5)   PDOT(6)          R
C       ¢         ¢         ¢         ¢         ¢         ¢        E    E
C       U         V         W         P         Q         R        A    L
C                                                                  R    A
C    P(1)      P(2)      P(3)      P(4)      P(5)      P(6)        T    T
C       ¢         ¢         ¢         ¢         ¢         ¢        H    I
C       X         Y         Z        PHI      THETA      PSI             V
C                                                                       E
C-------------------------------ROBERT HICKEY---------9/89--------------
      IMPLICIT NONE
      REAL*8 CTRN(6,6),AINV(6,6),RHS(6),EDOT(6,4),EO(6),E(6),PDOT(6,4)
      REAL*8 PO(6),P(6),DTO2,DT,DTO6,TA
      REAL*8 DS4,DC4,DS5,DC5,DS6,DC6,DT5,DS5DC6,DS5DS6
      INTEGER*2 I,J,K,JJJ
C
      REAL*8 DD(1800)
      COMMON /COMDD/ DD
C
      EQUIVALENCE (DD(  1),        DT),(DD(  6),      DTO6)
      EQUIVALENCE (DD(  7),      DTO2),(DD( 11),       JJJ)
      EQUIVALENCE (DD( 10),        TA)
      EQUIVALENCE (DD( 31),EDOT(1,1)),(DD( 55),       E(1))
      EQUIVALENCE (DD( 61),      P(1)),(DD( 71),    RHS(1))
      EQUIVALENCE (DD( 77),PDOT(1,1))
      EQUIVALENCE (DD(107),AINV(1,1)),(DD(143),CTRN(1,1))
C
      DO  160 I = 1,6
       EO(I) = E(I)
       PO(I) = P(I)
       DO   170 J = 1,4
        EDOT(I,J) = 0.0
        PDOT(I,J) = 0.0
 170   CONTINUE
 160  CONTINUE
C
C START 4TH ORDER RUNGA KUTTA INTEGRATION
C
      DO 100 JJJ=1,4
C
C CALCULATE THE RIGHT HAND SIDE OF THE HYDRODYNAMIC EQUATIONS
```

```
C
      CALL HYDRO
C
C  SOLVE FOR ACCELERATIONS
C  AINV 6X6 * RHS 6X1 = EDOT 6X1  (JJJ)
C
      DO 310 K=1,6
      DO 320 I=1,6
      EDOT(I,JJJ)=EDOT(I,JJJ)+AINV(I,K)*RHS(K)
  320 CONTINUE
  310 CONTINUE
C
C CALCULATE COORDINATE TRANSFORM FROM FLUID RELATIVE TO BODY RELATIVE
C
      DS4 = DSIN(P(4))
      DC4 = DCOS(P(4))
      DS5 = DSIN(P(5))
      DC5 = DCOS(P(5))
      DS6 = DSIN(P(6))
      DC6 = DCOS(P(6))
      DT5 = DTAN(P(5))
      DS5DC6 = DS5*DC6
      DS5DS6 = DS5*DS6
C
      CTRN(1,1) =  DC5*DC6
      CTRN(1,2) = -DC4*DS6 + DS4*DS5DC6
      CTRN(1,3) =  DS4*DS6 + DC4*DS5DC6
      CTRN(2,1) =  DC5*DS6
      CTRN(2,2) =  DC4*DC6 + DS4*DS5DS6
      CTRN(2,3) = -DS4*DC6 + DC6*DS5DS6
      CTRN(3,1) = -DS5
      CTRN(3,2) =  DC5*DS4
      CTRN(3,3) =  DC5*DC4
C
      CTRN(4,4) =  1.0
      CTRN(4,5) =  DS4*DT5
      CTRN(4,6) =  DC4*DT5
      CTRN(5,4) =  0.0
      CTRN(5,5) =  DC4
      CTRN(5,6) = -DS4
      CTRN(6,4) =  0.0
      CTRN(6,5) =  DS4/DC5
      CTRN(6,6) =  DC4/DC5
C
C PREP BODY RELATIVE PDOT FOR INTEGRATION
C     CTRN 6X6 * E 6X1  = PDOT 6X1
C
      DO 340 K=1,6
      DO 350 I=1,6
      PDOT(I,JJJ)=PDOT(I,JJJ)+CTRN(I,K)*E(K)
  350 CONTINUE
  340 CONTINUE
C
C WEIGHT ESTIMATES EACH PASS
C
      IF( JJJ .EQ. 1 .OR. JJJ .EQ. 2) THEN
```

```
        DO 191 I = 1,6
          E(I) = EO(I) + EDOT(I,JJJ)*DTO2
          P(I) = PO(I) + PDOT(I,JJJ)*DTO2
 191    CONTINUE
      ELSE
        DO 192 I = 1,6
          E(I) = EO(I) + EDOT(I,JJJ)*DT
          P(I) = PO(I) + PDOT(I,JJJ)*DT
 192    CONTINUE
      ENDIF
C
 100  CONTINUE
C
C ADD UP THE 4 PRE-ESTIMATES TO YIELD THE FINAL ESTIMATE
C
      DO 210 I = 1,6
      E(I) = EO(I)+(EDOT(I,1)+2.*(EDOT(I,2)+EDOT(I,3))+EDOT(I,4))*DTO6
      P(I) = PO(I)+(PDOT(I,1)+2.*(PDOT(I,2)+PDOT(I,3))+PDOT(I,4))*DTO6
 210  CONTINUE
C
      RETURN
      END
```

```
C ***********************************************************************
      SUBROUTINE HYDRO
C ***********************************************************************
C DTRC 2510 HYDRODYNAMIC COEFFICIENTS
C NOW INCLUDES ZERO FORWARD SPEED COEFFICIENTS
C -------------------------ROBERT HICKEY---12/89------------------------
      IMPLICIT NONE
      INTEGER*2 JJJ,II
      INTEGER*2 IDTAR
      REAL*8 TEMP1,TEMP2,TEMP3,TEMP
      REAL*8 UPFRAC,LOFRAC,URACER,URACERO,URACES,URACESO
      REAL*8 DRE3,DSPE3,DSSE3,DRE,DSPE,DSSE
      REAL*8 DRHS1,DRHS2,DRHS3,DRHS4,DRHS5,DRHS6
      REAL*8 ZRHS1,ZRHS2,ZRHS3,ZRHS4,ZRHS5,ZRHS6
      REAL*8 BRHS1,BRHS2,BRHS3,BRHS4,BRHS5,BRHS6
C
      REAL*8 DD(1800)
      COMMON /COMDD/ DD
C
      REAL*8 URUR,USUS,DELRCR,DELRCS
      REAL*8 DSP,DSS,DR,DB,UC,DT,RHS(6),E(6),P(6),QPROP,TA,YG,YB
      REAL*8 PHISTERN,VATS,WATS,ETA,CETA,ETAM1,BETAS,KBETA,WT,BOUY
      REAL*8 ARAPPHI2,ARAPPHI4,CROSSLIM,SINTHETA,COSTHETA,SINPHI,COSPHI
      REAL*8 UU,VV,WW,PP,QQ,RR,PR,PQ,QR,UV,UW,UP,UQ,UR,VATTAIL,UUDB
      REAL*8 VP,VR,WP,WQ,SQRVVWW,VSQRVVWW,WSQRVVWW,CROSS(6),VORT(6)
      REAL*8 DRDR,DSDS,UUDS,UUDR,UABW,UABV,WABW,QABQ,VABV,RABR,FXP
      REAL*8 PABP,DPDP,UUDP,UT,BETA,ALPHA,WR,ARRAY(6),WTINH2O,DBDB
C
      REAL*8 XUDOT,XQQ,XRP,XVR,XWQ,XVV,XDRDR,XDSSDSS,XDSPDSP,XDBDB
      REAL*8 XW,XQ,XWDSP,XWDSS,XQDSS,XQDSP,XVDR,XRDR,XWW,XRR
      REAL*8 XDRE2,XDSSE2,XDSPE2
      REAL*8 YVDOT,YRDOT,YRU,YVU,YVABVU,YVV,YDR,YDRN,YRABR,YVABV
      REAL*8 YPU,YWP,YPABP,YPQ,YSTAR,YVABVR,YHVABV,YDRE,YDRE3,YDRE3R
      REAL*8 ZWDOT,ZQDOT,ZQU,ZWU,ZABWU,ZWW,ZDSS,ZDSSN,ZQABQ,ZWABW
      REAL*8 ZVP,ZSTAR,ZPR,ZDSP,ZDSPN,ZHWABW,ZDB
      REAL*8 ZDSSE,ZDSPE,ZDSPE3,ZDSSE3,ZDSSE3R,ZDSPE3R
      REAL*8 KPDOT,KVDOT,KRDOT,KQR,KPU,KRU,KWP,KVU,K4S,K8S,KI,KDSP,KDSS
      REAL*8 KPABP,KWR,KDR,KDRN,KSTAR
      REAL*8 KDRE,KDRE3,KDSSE,KDSPE,KDSSE3,KDSPE3,KDSSE3R,KDSPE3R
      REAL*8 MQDOT,MWDOT,MQU,MWU,MWABWR,MABWU,MWW,MDSS,MDSSN,MQABQ,MWABW
      REAL*8 MRP,MSTAR,MVP,MDSP,MDSPN,MHWABW,MDB
      REAL*8 MDSSE,MDSPE,MDSSE3,MDSPE3,MDSSE3R,MDSPE3R
      REAL*8 NRDOT,NVDOT,NRU,NVU,NVABVR,NABVU,NVV,NDR,NDRN,NVABV,NRABR
      REAL*8 NWP,NPQ,NSTAR,NPU,NHVABV,NDRE,NDRE3,NDRE3R
      REAL*8 IYY,IZZ,IXX,SLEN,MASS,ZB,ZG,XB,XG,XSTERN
C
      EQUIVALENCE  (DD(  1),       DT), (DD( 10),      TA)
      EQUIVALENCE  (DD( 11),      JJJ), (DD( 16),    BETA)
      EQUIVALENCE  (DD( 15),   ALPHA)
      EQUIVALENCE  (DD( 17),      FXP ), (DD( 18),   QPROP)
      EQUIVALENCE  (DD( 19),      ETA ), (DD( 20),    CETA)
      EQUIVALENCE  (DD( 55),      E(1)), (DD( 61),    P(1))
      EQUIVALENCE  (DD( 67),       DSS), (DD( 68),     DSP)
      EQUIVALENCE  (DD( 69),        DR), (DD( 71),  RHS(1))
      EQUIVALENCE  (DD(179),        DB)
      EQUIVALENCE  (DD(260),CROSS(1)), (DD(266),  VORT(1))
```

```
      EQUIVALENCE  (DD(272),VATTAIL)
C
      EQUIVALENCE  (DD(301),XUDOT   ),(DD(303),XW      )
      EQUIVALENCE  (DD(305),XQ      ),(DD(307),XQQ     )
      EQUIVALENCE  (DD(309),XRR     ),(DD(311),XRP     )
      EQUIVALENCE  (DD(313),XVR     ),(DD(315),XWQ     )
      EQUIVALENCE  (DD(317),XVV     ),(DD(319),XWW     )
      EQUIVALENCE  (DD(321),XDBDB   ),(DD(323),XDRDR   )
      EQUIVALENCE  (DD(325),XDSSDSS ),(DD(327),XDSPDSP )
      EQUIVALENCE  (DD(329),XDRE2   ),(DD(331),XDSSE2  )
      EQUIVALENCE  (DD(333),XDSPE2  )
      EQUIVALENCE  (DD(1703),XWDSP  ),(DD(1705),XWDSS  )
      EQUIVALENCE  (DD(1707),XQDSP  ),(DD(1709),XQDSS  )
      EQUIVALENCE  (DD(1711),XVDR   ),(DD(1713),XRDR   )
C
      EQUIVALENCE  (DD(338),YVDOT   ),(DD(340),YPDOT   )
      EQUIVALENCE  (DD(342),YRDOT   ),(DD(344),YPABP   )
      EQUIVALENCE  (DD(346),YPQ     ),(DD(348),YRU     )
      EQUIVALENCE  (DD(350),YPU     ),(DD(352),YVU     )
      EQUIVALENCE  (DD(354),YVABVR  ),(DD(356),YVABVU  )
      EQUIVALENCE  (DD(358),YVV     ),(DD(360),YDR     )
      EQUIVALENCE  (DD(362),YDRN    ),(DD(364),YRABR   )
      EQUIVALENCE  (DD(366),YVABV   ),(DD(368),YWP     )
      EQUIVALENCE  (DD(370),YSTAR   ),(DD(372),YHVABV  )
      EQUIVALENCE  (DD(374),YDRE3   ),(DD(376),YDRE    )
      EQUIVALENCE  (DD(378),YDRE3R  )
C
      EQUIVALENCE  (DD(390),ZWDOT   ),(DD(392),ZQDOT   )
      EQUIVALENCE  (DD(394),ZQU     ),(DD(396),ZVP     )
      EQUIVALENCE  (DD(398),ZSTAR   ),(DD(400),ZWU     )
      EQUIVALENCE  (DD(402),ZABWU   ),(DD(404),ZWW     )
      EQUIVALENCE  (DD(406),ZDB     ),(DD(408),ZDSS    )
      EQUIVALENCE  (DD(410),ZDSSN   ),(DD(412),ZDSP    )
      EQUIVALENCE  (DD(414),ZDSPN   ),(DD(416),ZQABQ   )
      EQUIVALENCE  (DD(418),ZWABW   ),(DD(420),ZHWABW  )
      EQUIVALENCE  (DD( 422),ZDSSE3 ),(DD( 424),ZDSPE3 )
      EQUIVALENCE  (DD( 426),ZDSSE  ),(DD( 428),ZDSPE  )
      EQUIVALENCE  (DD( 430),ZDSSE3R),(DD( 432),ZDSPE3R)
      EQUIVALENCE  (DD(1717),ZPR     )
C
      EQUIVALENCE  (DD(440),KVDOT   ),(DD(442),KPDOT   )
      EQUIVALENCE  (DD(444),KRDOT   ),(DD(446),KQR     )
      EQUIVALENCE  (DD(448),KPABP   ),(DD(450),KPU     )
      EQUIVALENCE  (DD(452),KRU     ),(DD(454),KWP     )
      EQUIVALENCE  (DD(456),KSTAR   ),(DD(458),KVU     )
      EQUIVALENCE  (DD(460),KI      ),(DD(462),K4S     )
      EQUIVALENCE  (DD(464),K8S     ),(DD(466),KDR     )
      EQUIVALENCE  (DD(468),KDRN    ),(DD(470),KDSS    )
      EQUIVALENCE  (DD(472),KDSP    ),(DD(474),KDRE    )
      EQUIVALENCE  (DD(476),KDRE3   ),(DD(478),KDSSE   )
      EQUIVALENCE  (DD(480),KDSPE   ),(DD(482),KDSSE3  )
      EQUIVALENCE  (DD(484),KDSPE3  ),(DD(486),KDSSE3R )
      EQUIVALENCE  (DD(488),KDSPE3R )
      EQUIVALENCE  (DD(1721),KWR     )
C
      EQUIVALENCE  (DD(494),MWDOT   ),(DD(496),MQDOT   )
```

```
      EQUIVALENCE  (DD(498),MRP     ),(DD(500),MQU     )
      EQUIVALENCE  (DD(502),MWU     ),(DD(504),MSTAR   )
      EQUIVALENCE  (DD(506),MWABWR  ),(DD(508),MABWU   )
      EQUIVALENCE  (DD(510),MWW     ),(DD(512),MDSS    )
      EQUIVALENCE  (DD(514),MDSSN   ),(DD(516),MDSP    )
      EQUIVALENCE  (DD(518),MDSPN   ),(DD(520),MDB     )
      EQUIVALENCE  (DD(522),MQABQ   ),(DD(524),MWABW   )
      EQUIVALENCE  (DD( 526),MHWABW ),(DD(528),MDSSE3  )
      EQUIVALENCE  (DD( 530),MDSPE3 ),(DD(532),MDSSE   )
      EQUIVALENCE  (DD( 534),MDSPE  ),(DD(536),MDSSE3R )
      EQUIVALENCE  (DD( 538),MDSPE3R)
      EQUIVALENCE  (DD(1723),MVP     )
C
      EQUIVALENCE  (DD(546),NVDOT   ),(DD(548),NPDOT   )
      EQUIVALENCE  (DD(550),NRDOT   ),(DD(552),NPQ     )
      EQUIVALENCE  (DD(554),NPU     ),(DD(556),NRU     )
      EQUIVALENCE  (DD(558),NSTAR   ),(DD(560),NVU     )
      EQUIVALENCE  (DD(562),NVABVR  ),(DD(564),NABVU   )
      EQUIVALENCE  (DD(566),NVV     ),(DD(568),NDR     )
      EQUIVALENCE  (DD(570),NDRN    ),(DD(572),NVABV   )
      EQUIVALENCE  (DD(574),NRABR   ),(DD(576),NWP     )
      EQUIVALENCE  (DD(578),NHVABV  ),(DD(580),NDRE3   )
      EQUIVALENCE  (DD(582),NDRE    ),(DD(584),NDRE3R  )
C
      EQUIVALENCE  (DD(593),SLEN    ),(DD(596),IYY     )
      EQUIVALENCE  (DD(598),IZZ     ),(DD(600),IXX     )
      EQUIVALENCE  (DD(602),IXZ     ),(DD(604),IYZ     )
      EQUIVALENCE  (DD(606),IXY     ),(DD(608),MASS    )
      EQUIVALENCE  (DD(610),WT      ),(DD(612),BOUY    )
      EQUIVALENCE  (DD(614),ZB      ),(DD(616),ZG      )
      EQUIVALENCE  (DD(618),XB      ),(DD(620),XG      )
      EQUIVALENCE  (DD(622),XSTERN  ),(DD(624),XFORWARD)
      EQUIVALENCE  (DD(626),YG      ),(DD(628),YB      )
      EQUIVALENCE  (DD(629),WTINH2O )
      EQUIVALENCE  (DD(630),CROSSLIM)
      EQUIVALENCE  (DD(631),LOFRAC  ),(DD(632),UPFRAC  )
C
      EQUIVALENCE  (DD(848),KTAINF  ),(DD(849),UAINF   )
      EQUIVALENCE  (DD(850),URACERO ),(DD(851),URACESO )
      EQUIVALENCE  (DD(858),URACER  ),(DD(859),URACES  )
      EQUIVALENCE  (DD(866),DRE     ),(DD(867),DSSE    )
      EQUIVALENCE  (DD(868),DSPE    )
      EQUIVALENCE  (DD(1758),ARRAY(1))
C
C
C FOR EFFICIENCY THE FOLLOWING VARIABLES ARE CALCULATED
C
      UU = E(1)*E(1)
      VV = E(2)*E(2)
      WW = E(3)*E(3)
      UT = DSQRT(UU+VV+WW)
      PP = E(4)*E(4)
      QQ = E(5)*E(5)
      RR = E(6)*E(6)
      PR = E(4)*E(6)
      PQ = E(4)*E(5)
```

```
      QR = E(5)*E(6)
      UV = E(1)*E(2)
      UW = E(1)*E(3)
      UP = E(1)*E(4)
      UQ = E(1)*E(5)
      UR = E(1)*E(6)
      VP = E(2)*E(4)
      VR = E(2)*E(6)
      WP = E(3)*E(4)
      WQ = E(3)*E(5)
      WR = E(3)*E(6)
      URUR = URACER*URACER
      USUS = URACES*URACES
      DELRCR = URACER - URACERO
      DELRCS = URACES - URACESO
CDBG  WRITE(6,*) DELRCR,URACER,URACERO
CDBG  WRITE(6,*) DELRCS,URACES,URACESO
      VATS = E(2) + XSTERN*E(6)
      WATS = E(3) - XSTERN*E(5)
C
      IF( DABS(URACER) .GT. 0.1) THEN
         DRE  = DR - DATAN2(  VATS,URACER )
       ELSE
         DRE = DR
      ENDIF
C
      IF( DABS(URACES) .GT. 0.1) THEN
         DSSE = DSS - DATAN2( WATS,URACES )
         DSPE = DSP - DATAN2( WATS,URACES )
        ELSE
         DSSE = DSS
         DSPE = DSP
      ENDIF
C
      IF(URACER .GT. 0.0) THEN
        IF( DABS(DRE) .GT. 0.7) DRE = DSIGN(0.7D0,DRE)
      ELSE
        IF( DABS(DRE) .GT. 0.43) DRE = DSIGN(0.43D0,DRE)
      ENDIF
C
      IF(URACES .GT. 0.0) THEN
        IF( DABS(DSSE) .GT. 0.7) DSSE = DSIGN(0.7D0,DSSE)
        IF( DABS(DSPE) .GT. 0.7) DSPE = DSIGN(0.7D0,DSPE)
      ELSE
        IF( DABS(DSSE) .GT. 0.43) DSSE = DSIGN(0.43D0,DSSE)
        IF( DABS(DSPE) .GT. 0.43) DSPE = DSIGN(0.43D0,DSPE)
      ENDIF
C
      DRE3   = DRE*DRE*DRE
      DSPE3  = DSPE*DSPE*DSPE
      DSSE3  = DSSE*DSSE*DSSE
      DRDR = DR*DR
      DBDB = DB*DB
      DSDS = DSS*DSS
      DPDP = DSP*DSP
      UUDR = UU*DR
```

```
      UUDB = UU*DB
      UUDP = UU*DSP
      UUDS = UU*DSS
      UABW = E(1)*ABS(E(3))
      UABV = E(1)*ABS(E(2))
      WABW = E(3)*ABS(E(3))
      QABQ = E(5)*ABS(E(5))
      VABV = E(2)*ABS(E(2))
      PABP = E(4)*ABS(E(4))
      RABR = E(6)*ABS(E(6))
      SQRVVWW    = DSQRT(VV+WW)
      VSQRVVWW   = E(2)  *  SQRVVWW
      WSQRVVWW   = E(3)  *  SQRVVWW
      WTINH2O    = WT - BOUY
      SINPHI     = DSIN(P(4))
      COSPHI     = DCOS(P(4))
      SINTHETA   = DSIN(P(5))
      COSTHETA   = DCOS(P(5))
C
      IF( (DABS(VATS) .GT. 0.001) .OR. (DABS(WATS) .GT. 0.001)) THEN
        TEMP1 = VATS*VATS
        TEMP2 = WATS*WATS
        TEMP3 = TEMP1+TEMP2
        ARAPPHI2 = TEMP3*2.*VATS*WATS/TEMP3
        ARAPPHI4 = TEMP3*4.*VATS*WATS*(TEMP2 - TEMP1)/(TEMP3*TEMP3)
        BETAS    = DATAN2(DSQRT(TEMP3),E(1))
        PHISTERN = DATAN2(WATS,-VATS)
      ELSE
        ARAPPHI2 = 0.0
        ARAPPHI4 = 0.0
        BETAS    = 0.0
        PHISTERN = 0.0
      ENDIF
C
      IF((DABS(E(3)) .GT. 0.1D-10) .AND. (DABS(E(1)) .GT. 0.01)) THEN
        ALPHA = DATAN2(E(3),E(1))
      ELSE
        ALPHA = 0.0
      ENDIF
C
      IF( (DABS(E(2)) .GT. 0.1D-20) .AND. (DABS(UT) .GT. .01)) THEN
        BETA = DASIN(-E(2)/UT)
      ELSE
        BETA = 0.0
      ENDIF
C
C FOR EFFICIENCY CALL THESE SUBROUTINES ONLY ONCE
C
      IF ( JJJ .EQ. 1)  THEN
        CALL VORTEX
        CALL CROSFLOW
        CALL PROP
C
C       DO 345 IDTAR = 1,10
C       CALL TOWDARRY(IDTAR)
C 345   CONTINUE
```

```
      ENDIF
C
C     CETA
      ETAM1 = ETA - 1.0
C
C  CALCULATE RIGHT HAND SIDES OF EQUATIONS OF MOTION
C
C  AXIAL FORCES ZERO SPEED MODEL
C
      ZRHS1 = 0.0
C
C LATERAL FORCE ZERO SPEED MODEL
C
      ZRHS2 =  YHVABV*VATS*DABS(VATS)
C
C NORMAL FORCE ZERO SPEED MODEL
C
      ZRHS3 =  ZHWABW*WATS*DABS(WATS)
C
C ROLL MOMENT ZERO SPEED MODEL
C
      ZRHS4 =  KPABP*PABP
C
C PITCH MOMENT ZERO SPEED MODEL
C
      ZRHS5 =  MHWABW*WATS*DABS(WATS)
C
C YAW MOMENT ZERO SPEED MODEL
C
      ZRHS6 = NHVABV*VATS*DABS(VATS)
C
C CALCULATE DTRC HYDRODYNMIC FORCES
C  HYDRODYNAMIC AXIAL FORCES
C
      DRHS1 =  XQQ*QQ + XRR*RR + XRP*PR + XVR*VR + XWQ*WQ + XVV*VV
     1 + XW*UW + XWW*WW + XQ*UQ
     1 + (XDSSDSS*DSDS + XDSPDSP*DPDP + XDRDR*DRDR + XDBDB*DBDB)*UU
     1 + (XWDSP*DSP + XWDSS*DSS)*UW + (XQDSP*DSP + XQDSS*DSS)*UQ
     1 + (XVDR*UV + XRDR*UR)*DR
C
C HYDRODYNAMIC LATERAL  FORCE
C
      DRHS2 = YPABP*PABP + YPQ*PQ
     1  + YRU*UR + YPU*UP + YVU*UV + YVABVU*UABV + YVV*VV + YWP*WP
     1  + (YDR + YDRN*ETAM1)*UUDR + YVABV*VABV + YRABR*RABR + YSTAR*UU
     1  + YVABVR*VSQRVVWW
C
C HYDRODYNAMIC NORMAL FORCE
C
      DRHS3 = ZQU*UQ + ZWU*UW + ZABWU*UABW + ZVP*VP + ZSTAR*UU
     1 + (ZDSS + ZDSSN*ETAM1) * UUDS + (ZDSP + ZDSPN*ETAM1) * UUDP
     1 + ZDB*UUDB
     1 + ZQABQ*QABQ + ZWABW*WABW + ZPR*PR + ZWW*DABS(E(3))*SQRVVWW
C
C HYDRODYNAMIC ROLL MOMENT
C
```

```
      KBETA = BETAS*BETAS*(UU + VATS*VATS + WATS*WATS)
      DRHS4 =  KPU*UP + KRU*UR + KQR*QR + KWP*WP
     1 + ((KDR + KDRN*ETAM1)*DR + KDSS*DSS + KDSP*DSP + KSTAR)*UU
     1 + KWR*WR + KVU*UV + KPABP*PABP + KI*VATTAIL + ARAPPHI4*K4S
C     1 + KBETA*(K4S*DSIN(4.*PHISTERN) + K8S*DSIN(8.*PHISTERN))
C
C HYDRODYNAMIC PITCH MOMENT
C
      DRHS5 = MQU*UQ + MQABQ*QABQ + MWABW*WABW
     1 + MVP*VP + MRP*PR + MSTAR*UU + MDB*UUDB
     1 + MWU*UW + MWABWR*WSQRVVWW + MABWU*UABW + MWW*DABS(E(3))*SQRVVWW
     1 + (MDSS + MDSSN*ETAM1)*UUDS + (MDSP + MDSPN*ETAM1)*UUDP
C
C HYDRODYNAMIC YAW MOMENT
C
      DRHS6 =  NPQ*PQ + NPU*UP + NWP*WP + NSTAR*UU
     1 + NRU*UR + NVU*UV + NVABVR*VSQRVVWW + NABVU*UABV + NVV*VV
     1 + (NDR + NDRN*ETAM1)*UUDR  + NVABV*VABV + NRABR*RABR
C
C CALCULATE BODY FORCES, AND SUM FORCES INDEPENDENT OF ALPHA OR BETA
C USED FOR BOTH LOW AND HIGH ANGLE OF ATTACK MODELS
C
C BODY AXIAL FORCE
      BRHS1 = -MASS*( WQ - VR - XG*(QQ + RR) + ZG*PR)
     1  - WTINH2O*SINTHETA + FXP + ARRAY(1)
     1  + XDRE2*DRE*DRE*URUR
     1  + (XDSSE2*DSSE*DSSE + XDSPE2*DSPE*DSPE)*USUS
C
C BODY LATERAL FORCE
C
      BRHS2 = -MASS*(UR - WP + XG*PQ  + ZG*QR )
     1  + WTINH2O*COSTHETA*SINPHI
     1  + CROSS(2) + VORT(2) + ARRAY(2)
     1  + YDRE*( DELRCR*(-XSTERN*E(6) - E(2)) + DR*URUR)
     1  + YDRE3*DRE3*URUR
C
C BODY NORMAL FORCE
C
      BRHS3 = -MASS*(VP - UQ + XG*PR - ZG*(PP + QQ))
     1 + WTINH2O*COSTHETA*COSPHI
     1 + CROSS(3) + VORT(3) + ARRAY(3)
     1 + ZDSSE*( DELRCS*(XSTERN*E(5) - E(3)) + DSS*USUS)
     1 + ZDSPE*( DELRCS*(XSTERN*E(5) - E(3)) + DSP*USUS)
     1 + (ZDSPE3*DSPE3 + ZDSSE3*DSPE3)*USUS
C
C BODY ROLL MOMENT
C
      BRHS4 = MASS*ZG*(UR - WP)
     1 - (ZG*WT - ZB*BOUY)*COSTHETA*SINPHI
     1 + (YG*WT - YB*BOUY)*COSTHETA*COSPHI
     1 + QPROP + VORT(4) + ARRAY(4)
     1 + KDRE*( DELRCR*(XSTERN*E(6) - E(2)) + DR*URUR)
     1 + KDRE3*DRE3*URUR
     1 + KDSSE * (DELRCS*(XSTERN*E(5) - E(3)) + DSS*USUS)
     1 + KDSPE * (DELRCS*(XSTERN*E(5) - E(3)) + DSP*USUS)
     1 + (KDSPE3*DSPE3 + KDSSE3*DSPE3)*USUS
```

```
C
C BODY PITCH MOMENT
C
       BRHS5 = (-IXX + IZZ)*PR + MASS*(-ZG*(WQ - VR) + XG*(VP - UQ))
     1 - (XG*WT - XB*BOUY)*COSTHETA*COSPHI
     1 - (ZG*WT - ZB*BOUY)*SINTHETA
     1 + CROSS(5) + VORT(5) + ARRAY(5)
     1 + MDSSE * (DELRCS*(XSTERN*E(5) - E(3)) + DSS*USUS)
     1 + MDSPE * (DELRCS*(XSTERN*E(5) - E(3)) + DSP*USUS)
     1 + (MDSPE3*DSPE3 + MDSSE3*DSPE3)*USUS
C
C BODY YAW MOMENT
C
       BRHS6 = (-IYY + IXX)*PQ - MASS*(XG*(UR - WP))
     1 + (XG*WT - XB*BOUY)*COSTHETA*SINPHI
     1 + (YG*WT - YB*BOUY)*SINTHETA
     1 + CROSS(6) + VORT(6) + ARRAY(6)
     1 + NDRE*(DELRCR*(-XSTERN*E(6) - E(2)) + DR*URUR)
     1 + NDRE3*DRE3*URUR
C
C SUM ALL FORCES
C
      RHS(1) = BRHS1 + LOFRAC*DRHS1 + UPFRAC*ZRHS1
      RHS(2) = BRHS2 + LOFRAC*DRHS2 + UPFRAC*ZRHS2
      RHS(3) = BRHS3 + LOFRAC*DRHS3 + UPFRAC*ZRHS3
      RHS(4) = BRHS4 + LOFRAC*DRHS4 + UPFRAC*ZRHS4
      RHS(5) = BRHS5 + LOFRAC*DRHS5 + UPFRAC*ZRHS5
      RHS(6) = BRHS6 + LOFRAC*DRHS6 + UPFRAC*ZRHS6
C
      RETURN
      END
```

```
C ************************************************************************
      SUBROUTINE CROSFLOW
C ************************************************************************
C 2 CROSSFLOW FORCE AND MOMENT ROUTINES.
C DTRC 2510 METHOD USES CONSTANT CD
C ZERO SPEED MODEL USES LOCAL REYNOLDS NUMBER ON EACH 'STRIP' OF HULL
C TO ESTIMATE A CD
C SIMPSONS RULE FOR INTEGRATION.
C TABLE LOOK UP SRCHTAB1.FOR FOR CDCTAB LOOKUP
C  CROSOPT  1,2,3,4 OR 5
C  1 NO CROSSFLOW                                    W/ 2510 EOM
C  2 DTRC 2510 CROSSFLOW                             W/ 2510 EOM
C  3 DTRC 2510 EOM + CROSSFLOW AT ALPHA + BETA LT CROSSFLOW LIMIT AND
C    A BLEND OF DTRC 2510 EOM + ZERO SPEED EOM AT ALPHA OR BETA
C    GREATER LOWER THAN CROSS LIMIT AND LESS THAN UPPER LIMIT
C    AND ZERO SPEED EOM + CROSSFLOW WHEN
C    ALPHA OR BETA GT CROSSFLOW  LIMIT
C -----------------------------ROBERT HICKEY---------10/12/89-----------
      IMPLICIT NONE
      INTEGER*2 I, J, IXOFF, ICDC, INDEX, IA, IB
      PARAMETER (IXOFF = 50,ICDC = 31)
      REAL*8 E(6),CROSS(6),XXOFF(IXOFF),YYOFF(IXOFF),ZZOFF(IXOFF),TA
C
      REAL*8 UPFRAC,LOFRAC
      REAL*8 UPLIM, FRACLIM
      REAL*8 VZ,VY,UATXI,TEMP,RHOO2,REYNUM,VISCOS,RHO,PI
      REAL*8 YF(IXOFF),ZF(IXOFF),MM(IXOFF),NM(IXOFF),SUM
      REAL*8 CDRE,CDCTAB(ICDC),RETAB(ICDC),FRAC,AINTLIM,BINTLIM
      REAL*8 DTEMP2,DTEMP3,DTEMP5,DTEMP6,ZTEMP2,ZTEMP3,ZTEMP5,ZTEMP6
C
      INTEGER*2 NXOFF,CROSOPT,JJJ
      REAL*8 CDY,CDZ,CDM,CDN,CROSSLIM,ALPHA,BETA
      REAL*8 XOFF(IXOFF),YOFF(IXOFF),ZOFF(IXOFF),XFORWARD
C
      REAL*8 DD(1800)
      COMMON /COMDD/ DD
      SAVE /COMDD/  ,IA,IB
C
      EQUIVALENCE  (DD( 2),       PI)
      EQUIVALENCE  (DD( 8),      RHO), (DD( 11),       JJJ)
      EQUIVALENCE  (DD( 10),      TA), (DD( 55),      E(1))
      EQUIVALENCE  (DD( 15),  ALPHA ), (DD( 16),      BETA)
      EQUIVALENCE  (DD(260),CROSS(1)), (DD(624),  XFORWARD)
      EQUIVALENCE  (DD(630),CROSSLIM)
      EQUIVALENCE  (DD(631),LOFRAC  ), (DD(632),UPFRAC     )
      EQUIVALENCE  (DD(645),CDY     ), (DD(646),CDZ        )
      EQUIVALENCE  (DD(647),CDM     ), (DD(648),CDN        )
      EQUIVALENCE  (DD(649),CROSOPT )
      EQUIVALENCE  (DD(650),NXOFF   ), (DD(651),XOFF(1)    )
      EQUIVALENCE  (DD(701),YOFF(1) ), (DD(751),ZOFF(1)    )
      EQUIVALENCE  (DD(1636),AINTLIM)
C
      DATA RETAB / 1.D4, 1.425D4, 2.D4, 3.D4, 4.D4, 5.D4, 6.D4, 7.D4,
     1 8.D4, 9.D4, 1.D5, 1.425D5, 2.D5, 3.D5, 4.D5, 5.D5, 6.D5, 7.D5,
     2 8.D5, 9.D5, 1.D6, 1.425D6, 2.D6, 3.D6, 4.D6, 5.D6, 6.D6, 7.D6,
```

```
     3 8.D6, 9.D6, 1.D7 /
C
      DATA CDCTAB / 1.125, 1.176, 2*1.2, 1.208, 1.215, 1.218,
     1 4*1.223, 1.218, 1.150, 0.924, 0.662, 0.319, 0.324, 0.335, 0.350,
     2 0.376, 0.397, 0.491, 0.579, 0.662, 0.704, 0.725, 0.740, 0.751,
     3 0.758, 0.761, 0.763 /
C
      DATA VISCOS / 1.2D-5 /
C
      IF((TA .EQ. 0.0) .AND. (JJJ .EQ. 1)) THEN
C
        RHOO2 = RHO/2.
C
        IF( NXOFF .GT. IXOFF ) STOP ' CROSFLOW: NXOFF GT 50 '
C
        DO 10 I = 1,NXOFF
          XXOFF(I) = XOFF(I)
          YYOFF(I) = YOFF(I)
          ZZOFF(I) = ZOFF(I)
   10   CONTINUE
C
C MAKE X-OFFSETS INCREASING AND CENTER THEM WRT CG USING XFORWARD
C
        J = NXOFF
        DO 20 I = 1,NXOFF
          XOFF(I) = XFORWARD - XXOFF(J)
          YOFF(I) = YYOFF(J)*2.0
          ZOFF(I) = ZZOFF(J)*2.0
          J = J - 1
   20   CONTINUE
C
C BOW INTGRATION LIMIT IS 1/2 OF BODY DIAM BACK FROM BOW PERPENDICULAR
C NXOFF/2 IS ABOUT WHERE PMB IS
C
        BINTLIM = XOFF(NXOFF) - 0.5*YOFF(NXOFF/2)
        WRITE(8,197) AINTLIM,BINTLIM
  197   FORMAT(1X, ' AINTLIM ',E13.6,' BINTLIM ',E13.6,/)
C
        DO 11 I = 1,NXOFF
        WRITE(8,198) I, XOFF(I), I,YOFF(I), I,ZOFF(I)
  198 FORMAT(1X,'XOFF(',I2,')=',E11.4,' YOFF(',I2,')=',E11.4,
     1  ' ZOFF(',I2,')=',E11.4)
   11   CONTINUE
C
        SUM = 0.0
        CALL SIMPSON(YOFF,SUM)
        WRITE(8,*) '   '
        WRITE(8,*) ' SECTION AREA OF SUB FT**2      ', SUM
C
        DO 12 I = 1,NXOFF
        YYOFF(I) = PI/4.*YOFF(I)**2
   12   CONTINUE
        CALL SIMPSON(YYOFF,SUM)
        WRITE(8,*) ' VOLUME OF SUB FT**3            ', SUM
C
        DO 14 I = 1,NXOFF
```

```
          YYOFF(I) = RHO*PI/4.0D0*YOFF(I)**2*XOFF(I)**2
   14     CONTINUE
          CALL SIMPSON(YYOFF,SUM)
          WRITE(8,*) 'IYY MOMENT OF INERTIA SLUG FT**2  ', SUM
          WRITE(8,*) '   '
C
          DO 123 I = 1,6
          CROSS(I) = 0.0
  123     CONTINUE
C
          UPLIM = PI/2.0D0 - CROSSLIM
          FRACLIM = PI/2.0D0 - 2.0D0*CROSSLIM
C
          WRITE(8,*) ' CROSSLIM                              ',CROSSLIM
          WRITE(8,*) ' UPLIM                                 ',UPLIM
          WRITE(8,*) ' FRACLIM                               ',FRACLIM
          WRITE(8,*) '   '
C
          WRITE(8,*) 'I    RETAB         CDCTAB'
          DO 789 I = 1,ICDC
            WRITE(8,*) I,RETAB(I),CDCTAB(I)
  789     CONTINUE
          WRITE(8,*)
C
        IF((CROSOPT .EQ. 1).OR. (CROSOPT .EQ. 2))  THEN
          LOFRAC = 1.0D0
          UPFRAC = 0.0D0
        ENDIF
C
        IF(CROSOPT .EQ. 3)  THEN
          IF( (DABS(BETA)  .LE. CROSSLIM ) .AND.
     1        (DABS(ALPHA) .LE. CROSSLIM ) )       THEN
            UPFRAC = 0.0D0
            LOFRAC = 1.0D0
          ELSEIF( ( (DABS(BETA) .GT. CROSSLIM)
     1              .AND. (DABS(BETA) .LT. UPLIM) )
     1       .OR. ( (DABS(ALPHA) .GT. CROSSLIM)
     1              .AND. (DABS(ALPHA) .LT. UPLIM)  ) ) THEN
            IF( DABS(BETA) .GT. CROSSLIM) THEN
              LOFRAC = 1.0D0 - (DABS(BETA) - CROSSLIM)/FRACLIM
              UPFRAC = 1.0D0 - LOFRAC
            ELSEIF( DABS(ALPHA) .GT. CROSSLIM ) THEN
              LOFRAC = 1.0D0 - (DABS(ALPHA) - CROSSLIM)/FRACLIM
              UPFRAC = 1.0D0 - LOFRAC
            ENDIF
          ELSEIF( (DABS(BETA) .GE. UPLIM ) .OR.
     1            (DABS(ALPHA).GE. UPLIM ) )       THEN
            LOFRAC = 1.0D0
            UPFRAC = 0.0D0
          ENDIF
        ENDIF
C
        WRITE(8,*) ' UPFRAC ',UPFRAC
        WRITE(8,*) ' LOFRAC ',LOFRAC
C
        ENDIF
```

```
C
      IF(CROSOPT .EQ. 1)                               GOTO 111
      IF( (CROSOPT.EQ.2).OR.(CROSOPT.EQ.3) )           GOTO 222
C
  111 LOFRAC = 1.0D0
      UPFRAC = 0.0D0
      CROSS(1) = 0.0
      CROSS(2) = 0.0
      CROSS(3) = 0.0
      CROSS(4) = 0.0
      CROSS(5) = 0.0
      CROSS(6) = 0.0
                                                       RETURN
C
  222 DO 30 I = 1,NXOFF
        VZ = E(3) - XOFF(I)*E(5)
        VY = E(2) + XOFF(I)*E(6)
        UATXI = DSQRT(VZ*VZ + VY*VY)
        TEMP = VY*UATXI*ZOFF(I)
        YF(I) = -CDY*TEMP
        NM(I) = -XOFF(I)*CDN*TEMP
        TEMP = VZ*UATXI*YOFF(I)
        ZF(I) = -CDZ*TEMP
        MM(I) =  XOFF(I)*CDM*TEMP
   30 CONTINUE
C
      CALL SIMPSON(YF,DTEMP2)
      CALL SIMPSON(ZF,DTEMP3)
      CALL SIMPSON(MM,DTEMP5)
      CALL SIMPSON(NM,DTEMP6)
C
  333 DO 40 I = 1,NXOFF
        VY = E(2) + XOFF(I)*E(6)
        VZ = E(3) - XOFF(I)*E(5)
        UATXI = DSQRT(VZ*VZ + VY*VY)
        REYNUM = UATXI*0.5*(YOFF(I)+ZOFF(I))/VISCOS
        CALL SRCHTAB1(INDEX,RETAB,ICDC,REYNUM,FRAC)
      CDRE = CDCTAB(INDEX) +(CDCTAB(INDEX+1) - CDCTAB(INDEX))*FRAC
        TEMP = CDRE*VY*UATXI*ZOFF(I)
        YF(I) = -TEMP
        NM(I) = -XOFF(I)*TEMP
        TEMP = CDRE*VZ*UATXI*YOFF(I)
        ZF(I) = -TEMP
        MM(I) =  XOFF(I)*TEMP
CDBG  WRITE(6,*) CDRE, INDEX, REYNUM, UATXI
   40 CONTINUE
C
      CALL SIMPSON2(YF,ZTEMP2,AINTLIM,IA,BINTLIM,IB)
      CALL SIMPSON2(ZF,ZTEMP3,AINTLIM,IA,BINTLIM,IB)
      CALL SIMPSON2(MM,ZTEMP5,AINTLIM,IA,BINTLIM,IB)
      CALL SIMPSON2(NM,ZTEMP6,AINTLIM,IA,BINTLIM,IB)
C
      IF(CROSOPT .EQ. 2)  THEN
        CROSS(2) = DTEMP2
        CROSS(3) = DTEMP3
```

```
          CROSS (5)  = DTEMP5
          CROSS (6)  = DTEMP6
          LOFRAC = 1.0D0
          UPFRAC = 0.0D0
                                                        RETURN
        ENDIF
C
        IF(CROSOPT .EQ. 3)  THEN
          IF( (DABS(BETA)   .LE. CROSSLIM ) .AND.
     1        (DABS(ALPHA)  .LE. CROSSLIM ) )           THEN
                                                        GOTO 444
          ELSEIF( ( (DABS(BETA)  .GT. CROSSLIM)
     1              .AND.  (DABS(BETA)  .LT. UPLIM) )
     1       .OR.  ( (DABS(ALPHA)  .GT. CROSSLIM)
     1              .AND.  (DABS(ALPHA)  .LT. UPLIM)   ) ) THEN
            IF( DABS(BETA)  .GT. CROSSLIM) THEN
              LOFRAC = 1.0D0 -  (DABS(BETA)  - CROSSLIM)/FRACLIM
              UPFRAC = 1.0D0 - LOFRAC
            ELSEIF( DABS(ALPHA)  .GT. CROSSLIM ) THEN
              LOFRAC = 1.0D0 -  (DABS(ALPHA)  - CROSSLIM)/FRACLIM
              UPFRAC = 1.0D0 - LOFRAC
            ENDIF
                                                        GOTO 555
          ELSEIF(  (DABS(BETA)  .GE. UPLIM ) .OR.
     1             (DABS(ALPHA).GE. UPLIM ) )           THEN
                                                        GOTO 666
          ENDIF
        ENDIF
C
C CATCH ANY THING LEFT OVER
                                                        GOTO 111
C ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
  444 UPFRAC = 0.0D0
      LOFRAC = 1.0D0
      CROSS (2)  = DTEMP2
      CROSS (3)  = DTEMP3
      CROSS (5)  = DTEMP5
      CROSS (6)  = DTEMP6
                                                        RETURN
C
  555 CROSS (2)  = LOFRAC*DTEMP2 + UPFRAC*ZTEMP2
      CROSS (3)  = LOFRAC*DTEMP3 + UPFRAC*ZTEMP3
      CROSS (5)  = LOFRAC*DTEMP5 + UPFRAC*ZTEMP5
      CROSS (6)  = LOFRAC*DTEMP6 + UPFRAC*ZTEMP6
                                                        RETURN
C
  666 UPFRAC = 1.0D0
      LOFRAC = 0.0D0
      CROSS (2)  = ZTEMP2
      CROSS (3)  = ZTEMP3
      CROSS (5)  = ZTEMP5
      CROSS (6)  = ZTEMP6
C
                                                        RETURN
      END
```

```
C ***********************************************************************
      SUBROUTINE PROP
C ***********************************************************************
C
C    PROPOPT ALLOWS CHOICE OF MOTOR MODEL
C     1  RAMP TORQUE   (FT/LBS)  IN TORPI.DAT
C     2  RAMP HORSEPOWER (HP)  IN TORPI.DAT
C     3  RAMP RPM   (RPM)  IN TORPI.DAT
C     4  RAMP COMMAND SPEED FXP  (FT/S)  IN TORPI.DAT
C ---------------------------ROBERT HICKEY------------------11/89--------
C
      IMPLICIT NONE
      REAL*8 DRAG,FXPDIV(3),T1,T2,T3,T4,T5,FXPCON,SLEN
      REAL*8 UK,DT,PI,CHP2,KTT(99),KQQ(99),JB(99),E(6),TEMP,RPS
      REAL*8 PROPCON2, PROPCON3,KNTOF,DTO2,RHO,TA,COMMAND,TEMP2,TEMP3
      REAL*8 PROPRPM, VPROP,WPROP,BETAPROP,VELADV,CIDT,ETA,CETA,FXP
      REAL*8 WAKE,THRSTDED,EHPC,SHPC,RPMC,SHPDEL,QDEL,RPMDEL,DELX
      REAL*8 RPSDOT,RPSDOTO,CQ,CT,JAC,QPROP,FPSTORPM,PROPTHST,THST,VA
      REAL*8 RPSTAB(0:33),RPSC,JBFRAC,FRAC2,WAKEFACT,JAA,JBTEMP,JAFRAC
      INTEGER*2 I,RPMINDEX,JBINDEX,JJJ
C
      INTEGER*2 NRPM, NPROP, PROPOPT,IKTAB
      PARAMETER (IKTAB = 31)
      REAL*8 IDT, PROPDIAM, WAKEFRAC, THSTFACT, ETAC6,ETAC7,ETAC8,ETAC9
      REAL*8 XPROP,XSTERN,KTAINF,SHAFTEFF,WAKEBETA,THSTBETA
      REAL*8 EHP(0:32), SHP(0:32), RPM(0:32), DELX1, DELX2,DELX3
      REAL*8 JA(99),KQ(99),KT(99),FXPA(4),FXPB(4),FXPC(4),KTSS
      REAL*8 UFXPA(4),UFXPB(4),UFXPC(4),UAINFSS,UAINF,RACER,RACES
      REAL*8 RACET1,RACET2,RACET3,RACET4,RACET5,RACET6
      REAL*8 AREATR,AREABR,AREAST,ARRCTR,ARRCBR,ARRCST
      REAL*8 URACER,URACES,URACERO,URACESO
      REAL*8 XPRPTAB(IKTAB),KTAB(IKTAB),KUFOR,KUAFT,KRACE
C
      REAL*8 DD(1800)
      COMMON /COMDD/ DD
C
      SAVE /COMDD/,JBINDEX, RPMINDEX
C
      EQUIVALENCE (DD(  1),      DT),(DD(  2),       PI)
      EQUIVALENCE (DD(  4),     RTD)
      EQUIVALENCE (DD(  5),     DTR),(DD(  6),     DTO6)
      EQUIVALENCE (DD(  7),    DTO2),(DD(  8),      RHO)
      EQUIVALENCE (DD( 10),      TA)
      EQUIVALENCE (DD( 11),     JJJ),(DD( 12),    KNTOF)
      EQUIVALENCE (DD( 13),    CHP2)
      EQUIVALENCE (DD( 17),     FXP),(DD( 18),    QPROP)
      EQUIVALENCE (DD( 19),     ETA),(DD( 20),     CETA)
      EQUIVALENCE (DD( 21),    DRAG),(DD( 22),PROPTHST)
      EQUIVALENCE (DD( 23),    SHPC),(DD( 24),PROPRPM )
      EQUIVALENCE (DD( 25),  RPSDOT)
      EQUIVALENCE (DD( 55),   E(1) ),(DD( 70),COMMAND )
C
      EQUIVALENCE (DD(593),  SLEN )
      EQUIVALENCE (DD(622), XSTERN)
C
      EQUIVALENCE (DD(848),KTAINF  ),(DD(849),UAINF    )
```

```
      EQUIVALENCE  (DD(850),URACERO ), (DD(851),URACESO )
      EQUIVALENCE  (DD(853),AREATR  ), (DD(855),AREABR   )
      EQUIVALENCE  (DD(857),AREAST  )
      EQUIVALENCE  (DD(858),URACER  ), (DD(859),URACES  )
      EQUIVALENCE  (DD(861),ARRCTR  ), (DD(863),ARRCBR  )
      EQUIVALENCE  (DD(865),ARRCST  )
C
      EQUIVALENCE  (DD(900),NRPM    )
      EQUIVALENCE  (DD(901),RPM(0)  ), (DD(934),EHP(0)  )
      EQUIVALENCE  (DD(967),SHP(0)  )

      EQUIVALENCE  (DD(1000),NPROP   ), (DD(1001),JA(1)    )
      EQUIVALENCE  (DD(1101),KT(1)   ), (DD(1201),KQ(1)    )
      EQUIVALENCE  (DD(1301),JB(1)   ), (DD(1401),KTT(1)   )
      EQUIVALENCE  (DD(1501),KQQ(1)  )
C
      EQUIVALENCE  (DD(1621),PROPOPT ), (DD(1622),ETAC6    )
      EQUIVALENCE  (DD(1623),ETAC7   ), (DD(1624),ETAC8    )
      EQUIVALENCE  (DD(1625),ETAC9   ), (DD(1626),DELX1    )
      EQUIVALENCE  (DD(1627),DELX2   ), (DD(1628),DELX3    )
      EQUIVALENCE  (DD(1629),UIDT    ), (DD(1630),IDT      )
      EQUIVALENCE  (DD(1631),UPROPDIM), (DD(1632),PROPDIAM)
      EQUIVALENCE  (DD(1633),WAKEFRAC), (DD(1634),THSTFACT)
      EQUIVALENCE  (DD(1635),UXPROP  ), (DD(1636),XPROP    )
      EQUIVALENCE  (DD(1637),SHAFTEFF), (DD(1638),WAKEBETA)
      EQUIVALENCE  (DD(1639),THSTBETA)
      EQUIVALENCE  (DD(1641),UFXPA(1)), (DD(1645),UFXPB(1)  )
      EQUIVALENCE  (DD(1649),UFXPC(1))
      EQUIVALENCE  (DD(1653),FXPDIV(1))
C
      DATA XPRPTAB /-4.0,-3.0,-2.5,-2.0,-1.5,-1.0,-0.9,-0.8,-0.7,-0.6,
     1 -0.5,-0.4,-0.3,-0.2,-0.1,-0.0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9
     1 ,1.0,1.5,2.0,2.5,3.0,4.0/
      DATA KTAB /1.0,0.988,0.973,0.955,0.926,0.882,0.875,0.863,0.847,
     1 0.824,0.804,0.776,0.733,0.682,0.604,0.500,0.396,0.318,0.267,0.224
     1 ,0.196,0.176,0.153,0.137,0.125,0.118,0.074,0.045,0.027,0.012,
     1 0.000/
C
      IF(TA .EQ. 0.0) THEN
C
        FXPCON   = 0.5*RHO*SLEN*SLEN
        IF( PROPOPT .EQ. 4 )                       GOTO 50
C
C REFORMAT ADVANCE COEF, THRUST AND TORQUE TABLES TO AVOID
C PROBLEMS WHEN RPS = 0.0
        WRITE(8,*)  ' JB            KTT             KQQ   '
        DO 20 I  = 1,NPROP
          JB(I)   = 1.0/(DSQRT(1.0D0+JA(I)**2)*DSIGN(1.0D0,JA(I)))
          KTT(I)  = KT(I)*JB(I)**2
          KQQ(I)  = 0.1*KQ(I)*JB(I)**2
          WRITE(8,234)  JB(I),KTT(I),KQQ(I)
   20   CONTINUE
  234   FORMAT(1X,3(E13.6,1X))
C
C LINEARIZE EHP(*),SHP(*)  FOR BETTER LOOKUP
C NOTE EHP AND SHP ARE REALLY F(V*V*V)
```

```
C CREATE RPS TABLE
C
        EHP(0) = 0.0
        SHP(0) = 0.0
        RPSTAB(0) = 0.0
        DO 10 I = 1,NRPM-1
          TEMP = (2. * FLOAT(I))**2
          EHP(I) = EHP(I)/TEMP
          SHP(I) = SHP(I)/TEMP
          RPSTAB(I) = RPM(I)/60.0
   10   CONTINUE
C
C INITIALIZE SPEEDS
C
        UK = DMIN1(DABS(E(1)/KNTOF),40.0D0)
        I = DINT(0.5*UK)
        FRAC2 = 0.5 * UK - FLOAT(I)
        RPS = (RPSTAB(I) + (RPSTAB(I+1) - RPSTAB(I)) * FRAC2)
        PROPRPM = RPS*60.
        RPSDOT = 0.0
        RPSDOTO = 0.0
C
C PREP SOME CONSTANTS
C
        CIDT     = 1./(IDT * 2.0 * PI)
        PROPCON2 = PROPDIAM**2*RHO
        PROPCON3 = PROPDIAM**3*RHO
        WAKEFACT = 1. - WAKEFRAC
        THRSTDED = 1. - THSTFACT
C
        JAA = (E(1)*WAKEFACT)/(RPS*PROPDIAM)
        JBTEMP = 1.0/(DSQRT(1.0D0+JAA**2)*DSIGN(1.0D0,JAA))
        JBINDEX = NPROP/2
        CALL SRCHTAB1(JBINDEX,JB,NPROP,JBTEMP,JBFRAC)
        CALL SRCHTAB2(RPMINDEX,RPM,NRPM,PROPRPM,FRAC2)
C
        WRITE(8,*) ' I         XPRPTAB        KTAB'
        DO 675 I = 1,IKTAB
        WRITE(8,*) I,XPRPTAB(I),KTAB(I)
  675   CONTINUE
        TEMP = 2.*(XSTERN - XPROP)/PROPDIAM
        CALL SRCHTAB1(I,XPRPTAB,IKTAB,TEMP,TEMP2)
        KUFOR = KTAB(I) +(KTAB(I+1) - KTAB(I))*TEMP2
        KUAFT = 1.-KUFOR
        WRITE(8,*) ' I   X-PROP-QUARTER-CHORD '
        WRITE(8,*) I,TEMP
        WRITE(8,*) '   FRAC          KUAFT            KUFOR'
        WRITE(8,*) TEMP2,KUAFT,KUFOR
C
        RACET3 =  (ARRCTR + ARRCBR)/(AREATR+AREABR)
        RACET4 = 0.5*( (AREATR-ARRCTR)/AREATR + (AREABR-ARRCBR)/AREABR )
        RACET5 = (ARRCST/AREAST)
        RACET6 = (AREAST-ARRCST)/AREAST
        WRITE(8,*) ' AREAS OF RUDDERS ',RACET3,RACET4
        WRITE(8,*) ' AREAS OF STERN   ',RACET5,RACET6
C
```

```
         KTSS = (SHP(NRPM-1)*(2*(NRPM-1))**2*550.*60.*60.*SHAFTEFF)
     1   /(2.*(NRPM-1)*KNTOF*RHO*RPM(NRPM-1)**2*PROPDIAM**4)
         WRITE(8,*) ' KTSS ',KTSS
C
C INITIALIZE PROPELLER RACE
C        VA    = WAKEFACT*E(1)
C        UAINFSS = -VA + DSQRT( VA**2 + 8.*KTSS*(PROPDIAM*RPS)**2/PI )
C        URACERO = DSQRT(RACET3*(VA+KUFOR*UAINFSS)**2 + RACET4*VA**2)
C        URACER  = URACERO
C       WRITE(6,*) RPS,VA
C       WRITE(6,*) UAINFSS,URACERO
C
   50   CONTINUE
C
      ENDIF
C
      IF( PROPOPT .EQ. 4) GOTO 400
C
C LIMIT SPEED TO 40 KNOTS
      UK = DMIN1(DABS(E(1)/KNTOF),40.0D0)
      I = DINT(0.5*UK)
      FRAC2 = 0.5 * UK - FLOAT(I)
C NOTE EHP AND SHP ARE REALLY F(V*V*V)
      TEMP = UK**2
      EHPC = (EHP(I) + (EHP(I+1) - EHP(I)) * FRAC2)* TEMP
      SHPC = (SHP(I) + (SHP(I+1) - SHP(I)) * FRAC2)* TEMP
      RPMC = (RPM(I) + (RPM(I+1) - RPM(I)) * FRAC2)
      RPSC = (RPSTAB(I) + (RPSTAB(I+1) - RPSTAB(I)) * FRAC2)
C
C  SELECT MOTOR MODELS
C
      GOTO (100,200,300) PROPOPT
C
  100 QDEL = COMMAND
      SHPDEL = QDEL*PROPRPM/CHP2
      GOTO  450
  200 SHPDEL = COMMAND
      QDEL = SHPDEL*CHP2/PROPRPM
      GOTO 450
  300 RPS = COMMAND/60.
      PROPRPM = COMMAND
      QDEL = QPROP/SHAFTEFF
      SHPDEL = QDEL*PROPRPM/CHP2
      GOTO 500
  400 IF( E(1) .EQ. 0.0D0 ) THEN
        ETA = 0.0
       ELSE
        ETA = COMMAND/E(1)
        IF( ETA .GT. 5.0D0) ETA = 5.0D0
      ENDIF
C
C APPLY REYNOLD'S NUMBER CORRECTION TO FXP THRUST MODEL
C
      DELX = DELX1 + DELX2 / ( DELX3 + DLOG10(E(1)) )**2
      CETA = ETAC6 + ETAC9*(DSQRT(DABS(ETAC7+ETAC8*DELX)))
C
```

```
      IF( ETA .GE. FXPDIV(1)) THEN
        I = 1
        ELSEIF ( (ETA .GE. FXPDIV(2)) .AND. (ETA .LT. FXPDIV(1)) ) THEN
        I = 2
        ELSEIF ( (ETA .GE. FXPDIV(3)) .AND. (ETA .LT. FXPDIV(2)) ) THEN
        I = 3
        ELSEIF (ETA .LT. FXPDIV(3)) THEN
        I = 4
      ENDIF
C
      FXP = FXPCON*(( UFXPA(I) + DELX )*E(1)**2
     1              + UFXPB(I)*CETA*COMMAND*E(1)
     1              + UFXPC(I)*CETA*CETA*COMMAND**2)
      QPROP = 0.0
      RPS = 0.0
      GOTO 999
C
  450 RPSDOT = (SHAFTEFF*QDEL - QPROP)*CIDT
      RPS = RPS + (RPSDOT + RPSDOTO)*DTO2
      RPSDOTO = RPSDOT
  500 PROPRPM = RPS*60.
C
C STRICT SIGN CONVENTION APPLIED HERE
C
      VPROP = E(2) + XPROP*E(6)
      WPROP = E(3) - XPROP*E(5)
      TEMP = DSQRT(VPROP*VPROP + WPROP*WPROP)
C
      IF(E(1) .GT. 0.01) THEN
       BETAPROP = ATAN2(TEMP,E(1))
       ELSE
       BETAPROP= 0.0
      ENDIF
      TEMP = DSIN(BETAPROP)
      WAKE = WAKEFACT + WAKEBETA*TEMP
      THST = THRSTDED + THSTBETA*TEMP
      VA   = WAKE*E(1)
C
C A DISCREPANCY BETWEEN RIHSIM AND SLIME
C
C     TEMP = DSQRT( VA**2 + (RPS*PROPDIAM)**2 )
C     JBTEMP = (RPS*PROPDIAM)/TEMP
      JAA = VA/(RPS*PROPDIAM)
      JBTEMP = 1.0/(DSQRT(1.0D0+JAA**2)*DSIGN(1.0D0,JAA))
C
      CALL SRCHTAB1(JBINDEX,JB,NPROP,JBTEMP,JBFRAC)
C
      KTAINF = (KT(JBINDEX) + (KT(JBINDEX+1) - KT(JBINDEX))*JBFRAC)
      UAINFSS = -VA + DSQRT( VA**2 + 8.*KTSS*(PROPDIAM*RPS)**2/PI )
      URACERO = DSQRT( RACET3 * (VA+KUFOR*UAINFSS)**2 + RACET4*VA**2 )
      URACESO = DSQRT( RACET5 * (VA+KUFOR*UAINFSS)**2 + RACET6*VA**2 )
C
C CHECK FOR FLOW REVERSAL
C
      TEMP = VA**2 + 8.*KTAINF*(PROPDIAM*RPS)**2/PI
C     WRITE(6,*)
```

```
C     WRITE(6,*) TA,TEMP
      UAINF = -VA + DSIGN(1.0D0,TEMP)*DSQRT( DABS(TEMP) )
      IF( UAINF .GT. 0.0D0) THEN
        KRACE = KUFOR
      ELSE
        KRACE = KUAFT
      ENDIF
      TEMP  =  VA + KRACE*UAINF
      TEMP3 = RACET3*DSIGN(1.0D0,TEMP)*TEMP**2 + RACET4*VA**2
      URACER = DSIGN(1.0D0,TEMP3)*DSQRT(DABS(TEMP3))
      TEMP3 = RACET5*DSIGN(1.0D0,TEMP)*TEMP**2 + RACET6*VA**2
      URACES = DSIGN(1.0D0,TEMP3)*DSQRT(DABS(TEMP3))
C     WRITE(6,*) KTAINF,E(1),VA
C     WRITE(6,*) UAINF,URACER
C
      CT = (KTT(JBINDEX) + (KTT(JBINDEX+1) - KTT(JBINDEX))*JBFRAC)
      CQ = (KQQ(JBINDEX) + (KQQ(JBINDEX+1) - KQQ(JBINDEX))*JBFRAC)
      JAC = JA(JBINDEX) + ( JA(JBINDEX+1) -  JA(JBINDEX))*JBFRAC
C
      TEMP2 = VA**2 + (PROPDIAM*RPS)**2
      PROPTHST = PROPCON2*TEMP2*CT*THST
      QPROP = PROPCON3*TEMP2*CQ
C
C CALCULATE ETA
C
      IF( PROPRPM .LE. 0.0D0) THEN
        TEMP2 = -PROPRPM
      ELSE
        TEMP2 = PROPRPM
      ENDIF
C
      CALL SRCHTAB2(RPMINDEX,RPM,NRPM,TEMP2,FRAC2)
C
      FPSTORPM = PROPRPM/((2.0 * (FLOAT(RPMINDEX) + FRAC2))*KNTOF)
      ETA =  (PROPRPM/FPSTORPM)/E(1)
C
      IF( DABS(E(1)) .GT. 1.0D-2) THEN
        DRAG = EHPC*550./E(1)
      ELSE
        DRAG = 0.0D0
      ENDIF
      FXP = PROPTHST - DRAG
C
CDBG  WRITE(6,*)
CDBG  WRITE(6,*) E(1),UK,SHPC
CDBG  WRITE(6,*) TA,PROPTHST,DRAG
CDBG  WRITE(6,*) QDEL,QPROP,RPSC
CDBG  WRITE(6,*) RPS,PROPRPM,RPMINDEX
CDBG  WRITE(6,*) FPSTORPM,ETA,JAC
CDBG  WRITE(6,*) JBINDEX,SHPDEL,RPSDOT
CDBG  WRITE(6,*) CT,CQ,JBTEMP
C
  999 CONTINUE
C
      RETURN
      END
```

```
C ************************************************************************
      SUBROUTINE VORTEX
C     SUBROUTINE VORTEX(E,VORT,VATTAIL,BETA,TA)
C ************************************************************************
C
C VORTEX EXPRESSES THE INTERACTION OF THE SAIL VORTEX WITH THE AFT
C PORTION OF THE HULL
C VATTAIL IS THE 3-D TOTAL VELOCITY OF THE FAIRWATER DELAYED UNTIL IT
C REACHES THE PROPELLER
C THE DELAY IS DONE WITH CIRCULAR STACKS DXSAV,V25,AVDXSAV
C
C VORTFLAG
C   1 - NO VORTEX
C   2 - VORTEX NO SHEDDING
C   3 - VORTEX WITH SHEDDING
C --------------------------------- ROBERT HICKEY---------12/89-------
C
      IMPLICIT NONE
      INTEGER*2 ISIZE, IV, JV, II, I
      PARAMETER (ISIZE = 500)
      REAL*8 E(6),VORT(6),DXSAV(ISIZE),V25(ISIZE),AVDXSAV(ISIZE)
      REAL*8 DXNEW,UFLAST,TEMP1,TEMP2, MOMARM, XSTOP
      REAL*8 XEFFTAIL,UMIN, VFWI,VFWO,V25I,V25O,BETA
      REAL*8 XHULL,SUMY,SUMZ,SUMM,SUMN,TA,DT
      REAL*8 VATHULL, WATHULL, FRAC,VATTAIL
C
      INTEGER*2 VORTFLAG
      REAL*8  START1, START2,BETAST,XFW, X25 , ZFW
      REAL*8  XVEND, XTAIL, ZHCL, CL
C
      REAL*8 DD(1800)
      COMMON /COMDD/ DD
      EQUIVALENCE (DD( 55),   E(1) ),(DD(266), VORT(1))
      EQUIVALENCE (DD(272), VATTAIL)
      EQUIVALENCE (DD( 16),    BETA)
      EQUIVALENCE (DD( 10),      TA)
C
      EQUIVALENCE (DD(1601),USTART1 ),(DD(1602),START1   )
      EQUIVALENCE (DD(1603),USTART2 ),(DD(1604),START2   )
      EQUIVALENCE (DD(1605),BETAST  )
      EQUIVALENCE (DD(1607),UXFW    ),(DD(1608),XFW      )
      EQUIVALENCE (DD(1609),UX25    ),(DD(1610),X25      )
      EQUIVALENCE (DD(1611),UZFW    ),(DD(1612),ZFW      )
      EQUIVALENCE (DD(1613),UXVEND  ),(DD(1614),XVEND    )
      EQUIVALENCE (DD(1615),UXTAIL  ),(DD(1616),XTAIL    )
      EQUIVALENCE (DD(1617),UZHCL   ),(DD(1618),ZHCL     )
      EQUIVALENCE (DD(1619),CL      ),(DD(1620),VORTFLAG)
C
      DATA DT,UMIN / 0.05, 8.44 /
C
      IF( TA .EQ. 0.0) THEN
       IV = 1
       UFLAST = E(1)
       XEFFTAIL = XTAIL - (X25 - XFW)
       TEMP1 = E(1) * DT
       DO 20 II = 1, ISIZE
```

```
         DXSAV(II) = TEMP1
   20    CONTINUE
                                                       GOTO 999
        ENDIF
C
        IF( E(1)  .LT. UMIN)                           GOTO 999
        IF(VORTFLAG .EQ. 0)                            GOTO 999
        XSTOP = XVEND
        IF((VORTFLAG .EQ. 2) .AND. (ABS(BETA) .GT. BETAST)) THEN
          XSTOP = (START1 + START2*ABS(BETA))*XTAIL
        ENDIF
C
C STORE DISTANCE TRAVELED IN 1 DT IN DXSAV(*)
C AND VELOCITIES OF FAIRWATER AND QUARTER CHORD TO BE DELAYED
C
        IV = IV - 1
        IF( IV .LT. 1) IV = ISIZE
        DXSAV(IV) = 0.5* (E(1) + UFLAST)*DT
        UFLAST = E(1)
        VFWI = E(2) + XFW*E(6) - ZFW*E(4)
        AVDXSAV(IV) = 0.5*(VFWI + VFWO)*DXSAV(IV)
        VFWO = VFWI
        V25I = E(2) + X25*E(6) - ZFW*E(4)
        V25(IV) = 0.5*(V25I + V25O)
        V25O = V25I
C
        JV = IV - 1
        XHULL = XFW
        SUMY = 0.0
        SUMZ = 0.0
        SUMM = 0.0
        SUMN = 0.0
C
C INTEGRATION BY RECTANGULAR SUMMATION FROM XFW TO XVEND
C
        DO 30 II = 1, ISIZE
         JV = JV + 1
         IF( JV .GT. ISIZE)  JV = 1
         MOMARM = XHULL + 0.5*DXSAV(JV)
         TEMP1 = AVDXSAV(JV)*(E(3) - XHULL*E(5))
         TEMP2 = AVDXSAV(JV)*(E(2) + XHULL*E(6))
         SUMY = SUMY + TEMP1
         SUMN = SUMN + TEMP1*MOMARM
         SUMZ = SUMZ + TEMP2
         SUMM = SUMM + TEMP2*MOMARM
         XHULL = XHULL - DXSAV(JV)
         IF(XHULL .LT. XSTOP)                         GOTO 40
   30 CONTINUE
                                                      GOTO 999
   40 FRAC = (XSTOP - XHULL)/(-DXSAV(JV))
      VATHULL = E(2) + XSTOP*E(6)
      WATHULL = E(3) - XSTOP*E(5)
      SUMY = SUMY + WATHULL*AVDXSAV(JV)
      SUMN = SUMN + WATHULL*AVDXSAV(JV)*FRAC*XSTOP
      SUMZ = SUMZ + VATHULL*AVDXSAV(JV)
      SUMM = SUMM + VATHULL*AVDXSAV(JV)*FRAC*XSTOP
```

```
C
C LOCATE SAVED VELOCITY AT TAIL SURFACE
C
      DO 60 II = 1,ISIZE
       JV = JV + 1
       IF( JV .GT. ISIZE) JV = 1
       XHULL = XHULL - DXSAV(JV)
       IF( XHULL .LT. XEFFTAIL)                   GOTO 50
   60 CONTINUE
                                                   GOTO 999
   50 VATTAIL = V25(JV)
C
      VORT(2)  = -CL*SUMY
      VORT(3)  =  CL*SUMZ
      VORT(4)  = -VORT(2)*ZHCL
      VORT(5)  = -CL*SUMM
      VORT(6)  = -CL*SUMN
                                                  RETURN
C
  999 DO 10 II = 1,6
      VORT(II)  = 0.0
   10 CONTINUE
      VATTAIL = 0.0
C
                                                  RETURN
      END
```