NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

COMPUTER MODELING OF JAMMING EFFECTS ON INFRARED MISSILES

by

Troy M. Johnson

June 1999

Thesis Advisor: D. Curtis Schleher

Approved for public release; distribution is unlimited.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE June 1999	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE Computer Modeling of Jamming Effects on Infrared Missiles

5. FUNDING NUMBERS

6. AUTHOR(S) Troy M. Johnson

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Naval Air Weapons Center, China Lake, CA

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

Development of effective countermeasures for use against infrared (IR) missiles is hindered by the difficulty inherent in testing tactical missiles. The designers of such a countermeasure must devise a means of reproducing missile attitude after the test flight to allow for further analysis. This thesis describes an Inertial Measurement Unit (IMU) compact enough to be mounted on board a 4.5 inch missile. The IMU sensing elements are three quartz rate sensors providing yaw, pitch and roll rates, and the functionality of a gyro-stabilized system without the extensive electronics and high-speed spinning rotor. These micro-miniature, solid state devices are durable and compact, yet robust enough to allow for the precise re-creation of missile attitude.

A Simulink model is presented that accepts missile strap-down angular rates and, using an Euler rotation technique, produces yaw, pitch, and roll angles in an earth reference. The model corrects for sensor cross coupling, bias, and other factors. It has been calibrated using Carco Table test data, producing angles that matched expected values to within 2 degrees RMS on each axis. The resulting highly accurate attitude profile is stored as angle data and can also be viewed via an animation utility.

14. SUBJECT TERMS Simulink, Missile Attitude, IMU, Euler Rotation, Animation

15. NUMBER OF PAGES 56

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18 298-102

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited.

COMPUTER MODELING OF JAMMING EFFECTS ON INFRARED MISSILES

Troy M. Johnson
Lieutenant, United States Navy
B.S., University of South Carolina, 1992

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June 1999

Author: Troy M Johnson
Troy M. Johnson

Approved by: D Curtis Schleher
D. Curtis Schleher, Advisor

David C. Jenn
David C. Jenn, Second Reader

Dan C Boger
Dan Boger, Chairman
Department of Computer Information Sciences and Operations

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Development of effective countermeasures for use against infrared (IR) missiles is hindered by the difficulty inherent in testing tactical missiles. The designers of such a countermeasure must devise a means of reproducing missile attitude after the test flight to allow for further analysis. This thesis describes an Inertial Measurement Unit (IMU) compact enough to be mounted on board a 4.5 inch missile. The IMU sensing elements are three quartz rate sensors providing yaw, pitch and roll rates, and the functionality of a gyro-stabilized system without the extensive electronics and high-speed spinning rotor. These micro-miniature, solid state devices are durable and compact, yet robust enough to allow for the precise re-creation of missile attitude.

A Simulink model is presented that accepts missile strap-down angular rates and, using an Euler rotation technique, produces yaw, pitch, and roll angles in an earth reference. The model corrects for sensor cross coupling, bias, and other factors. It has been calibrated using Carco Table test data, producing angles that matched expected values to within 2 degrees RMS on each axis. The resulting highly accurate attitude profile is stored as angle data and can also be viewed via an animation utility.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION

A. BACKGROUND

Operational commanders rely heavily on air assets to execute missions ranging from precision bombing to high-altitude reconnaissance. Airborne units that can operate in close proximity to the enemy are considered even more valuable. Therefore, survivability is paramount in the air warfare domain. Early efforts to enhance air survivability resulted in Electronic Warfare (EW) systems and tactics that bolstered air effectiveness significantly. These improvements were accompanied, however, by an increased vulnerability to adversary EW techniques. Recent trends in air combat point to a new, more deadly threat: Infrared Surface-to-Air Missiles (IR SAM).

Between 1979 and 1985, 90 percent of all aircraft lost to hostile forces worldwide were taken down through the use of IR SAM missiles (NAVAIRSYSCOM, 1998). The Soviet empire's demise only exacerbated an already dangerous situation, flooding the international arms market with Soviet-built IR SAM's. The need for effective countermeasures against IR missiles could not be clearer.

This lack of effective countermeasures against IR SAM's has prompted efforts throughout the Department of Defense (DoD) to counter this significant threat. Obviously, success in this arena would enhance our military air operations capability while making air travel safer for civilians as well. One organization conducting research in this area is the Naval Air Weapons Center (NAWC) China Lake, California. This analysis is conducted in support of NAWC China Lake's efforts.

Modern IR missiles are resistant to most or all of the countermeasures developed thus far, although there are several promising programs. It is important that each of these designs be tested thoroughly to assess their effectiveness. The designers of such a countermeasure must overcome an important hurdle: How can the behavior of the missile be re-created after the test flight to allow for further analysis? Tactical missiles are too small to accept a traditional gyroscope-based inertial measurement unit (IMU). This thesis outlines a successful approach to solving this problem.

B. APPROACH

This method, which entails the sensing of inertial rates in the missile's frame of reference and reporting them back to a fixed station, was highly effective due to the IMU's compact size, use of low-cost sensors, and ability to thoroughly validate the model. Earth-referenced angles were produced with exceptional accuracy without the expense and complexity of a gyro-based system.

Critical performance data is gathered by mounting a telemetry system and IMU sensors within the body of the missile where the warhead would normally reside. The telemetry system transmits a high frequency carrier that is modulated with output signals taken from rate sensors which are contained in an Inertial Measurement Unit (IMU). In this particular case, the telemetry data provides 12-bit accuracy at a 1389 Hz sampling rate. The IMU contains a separate miniature quartz rate sensor for each of the three Euler rotational axes: yaw, pitch, and roll. Sensor indications are transmitted to a base station where they are stored for future analysis. The rate data is used as the input to a PC-based model constructed using Matlab and Simulink computer software to convert the

strapdown yaw, pitch and roll rates (R,Q, and P respectively) to Euler angles (ψ, θ, and ϕ) in the earth reference. Figure (1) depicts the missile reference axes.

The angle of attack (α) or attitude is the angle between the missile's velocity vector (v) and the x-axis of the missile. The attitude is defined by the yaw (ψ), pitch (θ), and roll (ϕ). Yaw is defined as the angle between the missile x-axis and the component of the velocity vector in the x-y plane. Pitch is the angle between the missile x-axis and the component of the velocity vector in the x-z plane. Roll is shown as the angle between the missile x-axis and the component of the velocity vector in the y-z plane. ψ, θ, and ϕ and their corresponding rates, R, Q, and P, respectively, define angular motion about the x, y, and z axes.

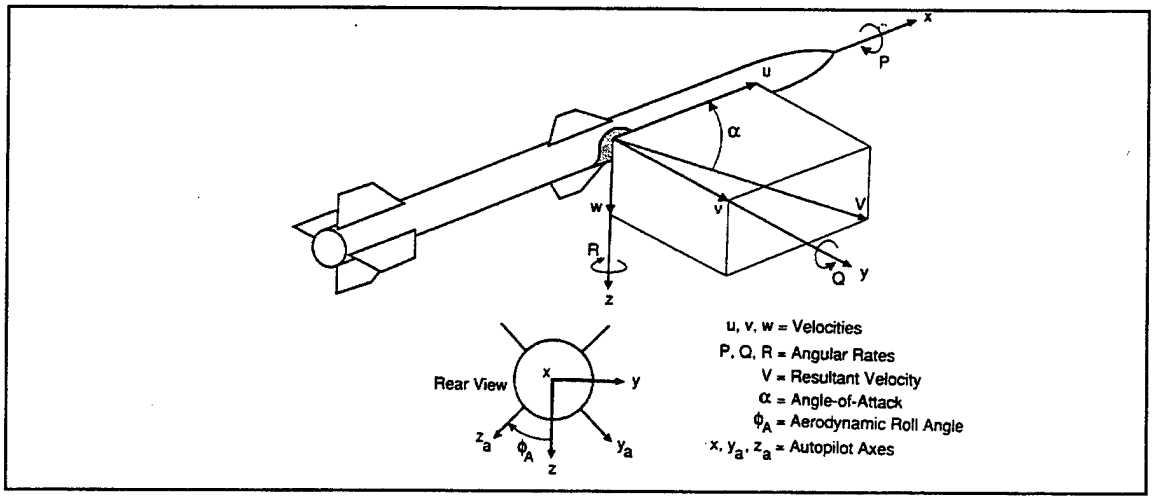

Figure 1. Missile Reference System (From Eichblatt, 1989)

Noise, sensor non-linearities, sensor bias, and other factors were considered while re-constructing the attitude of the missile faithfully. Each measured rate was integrated into an angle and transformed to the earth reference via a coordinate rotation. There are many techniques for accomplishing the transformation, each with associated costs and

benefits. One of the most common is the Euler rotation. The Euler rotation is accomplished using the equations listed in Figure (2). This method is computationally

$$\begin{bmatrix} \dot{\theta} \\ \dot{\phi} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi\tan\theta & \cos\phi\tan\theta & 1 \\ \sin\phi/\cos\theta & \cos\phi/\cos\theta & 0 \end{bmatrix} \begin{bmatrix} Q \\ R \\ P \end{bmatrix}$$

Figure 2. Euler Rotation Equations (After Blakelock, 1991)

cheap, but one of the required equations diverges to infinity when the pitch (θ) equals 90°. This is due to a physical phenomenon known as "Gimbal Lock," that occurs when a series of rotations at 90 degrees causes a reduction in the number of dimensional degrees of freedom. Gimbal lock is discussed further in Chapter III.

Another possibility is the Quaternion rotation. The quaternion method entails a conversion from three dimensions to four, rotational calculations, and then re-conversion back to three dimensions. This technique avoids the gimbal lock problem, but is complex and computationally costly.

C. QUESTIONS ANSWERED

There were several questions answered using post-flight re-construction: (1) Can the attitude profile of an Infrared (IR) missile be successfully modeled within a specified level of accuracy (2°) on the basis of IMU rate sensor data obtained using a miniature missile-mounted telemetry package, and (2) Do the quartz "rate gyro" sensors possess the requisite stability, dynamic range, and precision to accomplish the task set forth in the

primary research question? The following items were addressed in answering these questions:

1. Rate Sensor Analysis

The quartz "rate gyros" were analyzed and tested to ensure that they possessed the stability, dynamic range, and precision to provide the data that made post-flight attitude reconstruction possible. The sensors were found to be stable enough to withstand severe vibration, acute acceleration, and other extreme conditions. Tactical missiles routinely experience angular rates in excess of 400 degrees per second, so it was critical that the rate gyros were robust enough to perform within an acceptable tolerance despite these unusual conditions.

2. Model Design

The model accepts sensor rate data, compensates for flaws in sensor bias, temperature, noise, scale factor, etc., and converts the angular rates, which are associated with the missile "strap-down" frame of reference, to those that are referenced to earth coordinates. The inertial rates are continuously integrated and transformed to earth coordinates to track the attitude of the missile. The model also includes animation to provide a qualitative representation for assessing missile behavior.

3. Model Testing

The model was tested for sensitivity to noise and other factors, and also for operational correctness. In other words, it takes into account all significant factors involved in the translation from the motion of the missile to the rates reported by the sensors. These include, but are not limited to, linearity, temperature drift, noise,

quantization errors, and sensor bias. Additionally, there is cross-coupling of rates betweens axes due to the non-orthogonality associated with misalignment of the sensors. It is important to note that this model is not intended to provide any indication of the missile flight path. *Missile attitude* is the metric of concern. However, accelerometers contained in the IMU package can be used in conjunction with the attitude data to determine missile trajectory.

II. SENSOR FUNCTIONAL ANALYSIS

A. THEORY OF OPERATION

The IMU has as its sensing elements three Quartz Rate Sensors (QRS) which are mounted orthogonally to obtain rate data for each independent axis of motion. Manufactured by the Systron Donner Inertial Division of the BEI Sensors & Systems Company, the QRS11 is a micro-miniature, solid state device which serves the same purpose as a gyroscope, but without the hundreds of precision parts and high-speed spinning rotor. It uses a low power oscillating quartz sensor to sense angular rate. By using the Coriolis effect, any rotation about the sensor's longitudinal axis produces a DC voltage proportional to the rate of rotation.

The sensor is comprised of a double-ended tuning fork fabricated from a single wafer of monocrystalline piezoelectric quartz (somewhat like a quartz watch crystal). Figure (3) is a functional block diagram depicting the sensor operation. The drive tines

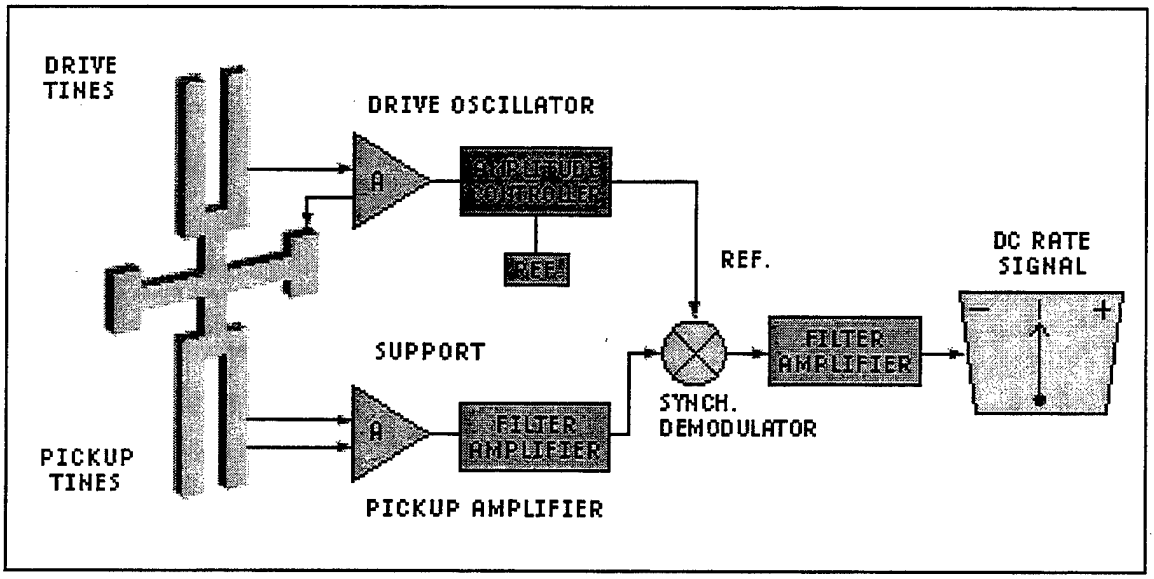

Figure 3. QRS11 Functional Block Diagram (From BEI, 1998)

are driven by an oscillator circuit at a specific amplitude, causing the tines to move together and apart at another high frequency.

Each tine is acted on by a Coriolis force given by: $\vec{F} = 2m\vec{\omega}_i \times \vec{V}_r$, where m is the tine mass, $\vec{V}_r$ is the instantaneous radial velocity, and $\vec{\omega}_i$ is the input rate. The cross product involved dictates that the resulting force is perpendicular to both the input rate and the instantaneous radial velocity. The drive tines move in opposite directions so that the resulting forces are perpendicular to the plane of the fork assembly and in opposite directions. The resulting torque is proportional to the rate of rotation. The torque varies sinusoidally at the same frequency as the drive tines, and in phase with the tine radial velocity. The pickup tines respond to the torque by moving in and out of plane, producing a signal at the pickup amplifier. These signals are amplified and demodulated into a DC signal which is directly proportional to the rate of sensor rotation.

The sign of the output signal reverses with the input rate due to a phase reversal in the Coriolis torque. Only rotation about the axis of symmetry of the fork will produce an output signal. Absence of rotation produces a zero output from the QRS.

B. OPERATING CHARACTERISTICS

The sensor, which measures less than 1.5 inches in diameter, is depicted in Figure (4). It is designed to operate in harsh environments with exceptional stability. Its lack of moving parts means that it should have a virtually "unlimited" operating life. This particular sensor was chosen for its compact design, dynamic range, and stability. Table (1) lists important operating specifications for the QRS11 Quartz Rate Sensor. The specifications that are shaded in gray were determined to be critical elements in this

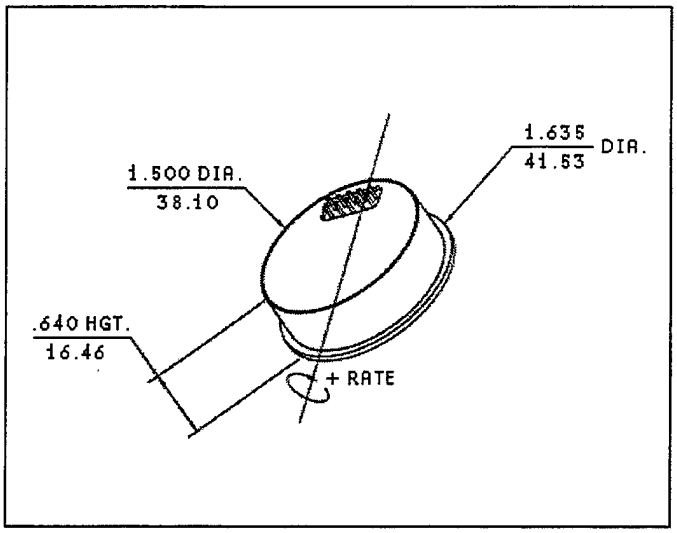

Figure 4. QRS-11 (From BEI, 1998)

particular application.

C. APPLICATION CONSIDERATIONS

Sensor temperature is expected to vary less than 5°C during missile flight, so the temperature-dependent operating characteristics were not considered in the model. Scale factor can be measure with precision prior to live firing, as can the sensor bias. The sensor possesses sufficient linearity to be considered a minor factor for a flight of short duration (< 15 seconds). The QRS11 also has sufficient bandwidth to faithfully respond to angular rates greater than those expected during missile flight. In summary, the two sensor properties that require correction within the model are Scale Factor and Bias. Sensor Cross Coupling is also addressed in the model, but is not an inherent property of the sensor itself.

PARAMETER	SPECIFICATION
POWER REQUIREMENTS	
Input Supply Voltage	+ and - 5 VDC ±5%
Input Power	< 0.8 watts
Input Power Noise Limits	10mV$_{rms}$ wideband, except @ 8.7 ±0.5 Khz
PERFORMANCE	
Full Scale Range	500 °/sec
Scale Factor (Nominal)	5 mV/°/sec
Calibration (at 22°C)	< 1.0% of value
Temp. Sensitivity	< 0.03%/°C
Bias, Factory Set (Max at 22°C)	< 3.5°/sec
Bias Variation over Temp. (from 22°C)	±1.00°/sec
Bandwidth	> 60 Hz
Linearity	< 0.05% of full range
G Sensitivity	< 0.02 °/sec/g
Threshold	< 0.020 °/sec
Output Noise (DC to 100 Hz)	< 0.020 °/sec/Hz$^{1/2}$
Operating Life	10 years, typical
ENVIRONMENTS	
Operating Temperature	-40°C to 80°C
Storage Temperature	-55°C to 100°C
Vibration, Operating	8 g$_{rms}$ 20 Hz to 2 KHz random
Vibration, Survival	20 g$_{rms}$ 20 Hz to 2 KHz random
Shock	200g
Weight	< 60 grams

Table 1. QRS11 Summary of Specifications

III. SIMULINK MODEL DESCRIPTION

A. INTRODUCTION

The Euler rotation model was built using a software tool called Simulink. Simulink is a software package used for modeling, simulating, and analyzing dynamical systems. It provides a graphical user interface (GUI) for building models as block diagrams, using click-and-drag mouse operations. The blocks carry out operations such as loading a file from disk, calculating a mathematical function, integrating a signal, or summing two signals. While Simulink provides the GUI, Matlab furnishes the underlying functionality, including all computation and function libraries.

To accomplish its objective, the Simulink model corrects for shortcomings in the sensors, integrates the input rates to get angles, transforms the angles from the missile's frame of reference to earth coordinates, and displays the resulting attitude profile visually. The model is depicted in Figure (5). A more detailed description follows.

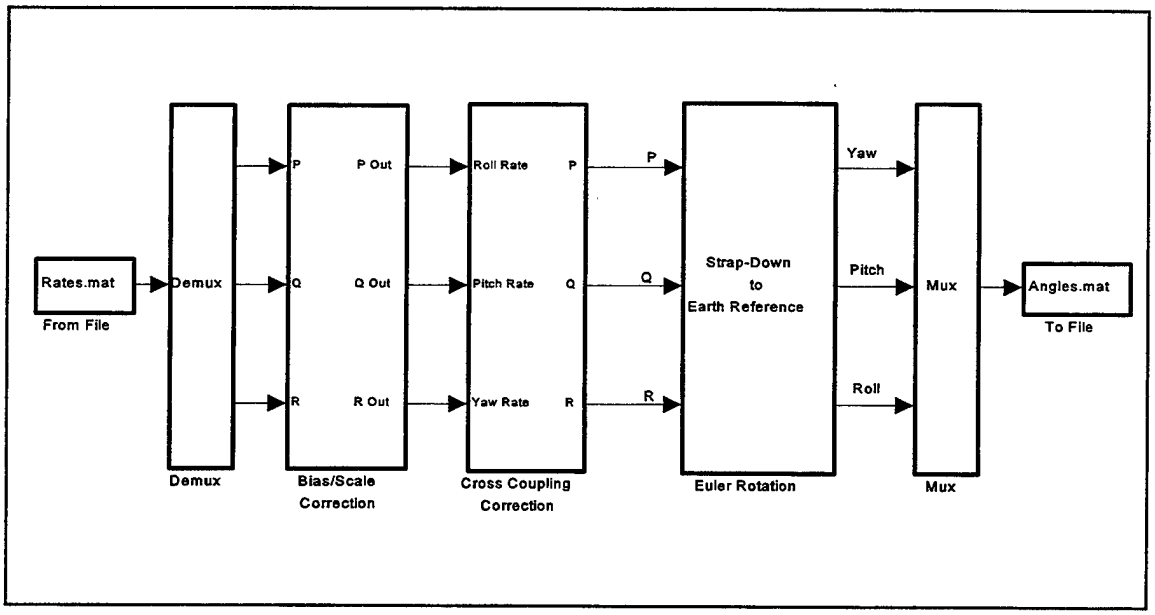

Figure 5. Euler Rotation Model

B. DATA INPUT FORMAT REQUIREMENTS

The Euler rotation model, as with all Simulink simulations, is started by the user. Upon initiation, telemetry data is retrieved from a Matlab binary file, Figure (6), that must bear the .MAT extension. The file may contain more than one matrix, but the model will only load the first matrix it encounters. Data file retrieval is accomplished by the "From File" block included in the Simulink library, which accepts data in the matrix form depicted in Figure (6), where t_n represents monotonically increasing time values and P_n, Q_n, and R_n are angular rates in the three orthogonal axes.

$$\begin{bmatrix} t_1 & t_2 & t_3 & t_4 & t_5 & \cdots\cdots & t_n \\ P_1 & P_2 & P_3 & P_4 & P_5 & \cdots\cdots & P_n \\ Q_1 & Q_2 & Q_3 & Q_4 & Q_5 & \cdots\cdots & Q_n \\ R_1 & R_2 & R_3 & R_4 & R_5 & \cdots\cdots & R_n \end{bmatrix}$$

Figure 6. Input Data Format

C. BIAS CORRECTION

Each sensor has an associated bias that results in a small output voltage even when there is no rate to be measured. It is important to note that since the sensor rates are continuously integrated to produce angles, any error in bias will accumulate throughout model operation. For example, a 1/2 degree per second bias results in a discrepancy on the order of five degrees in the output for that particular axis if the flight lasts ten seconds. It is imperative that sensor bias be compensated for prior to rotation. This is

accomplished in the bias/correction block, depicted in Figure (7).

The bias for each sensor can be obtained either through pre-flight testing or empirically via post-flight data analysis, as in this case. Further discussion can be found in Chapter IV. In either case, the bias value for each sensor is then entered into the model in the form of a constant that is subtracted from each data point during model operation. Manufacturer specifications for the QRS-11 quartz rate sensor indicate that bias should not exceed 2 ½ degrees per second, although the sensors used for initial model testing contained biases of less than one degree per second. Discussions with engineers at Systron Donner suggest that the values obtained in this case are typical for this type of sensor.

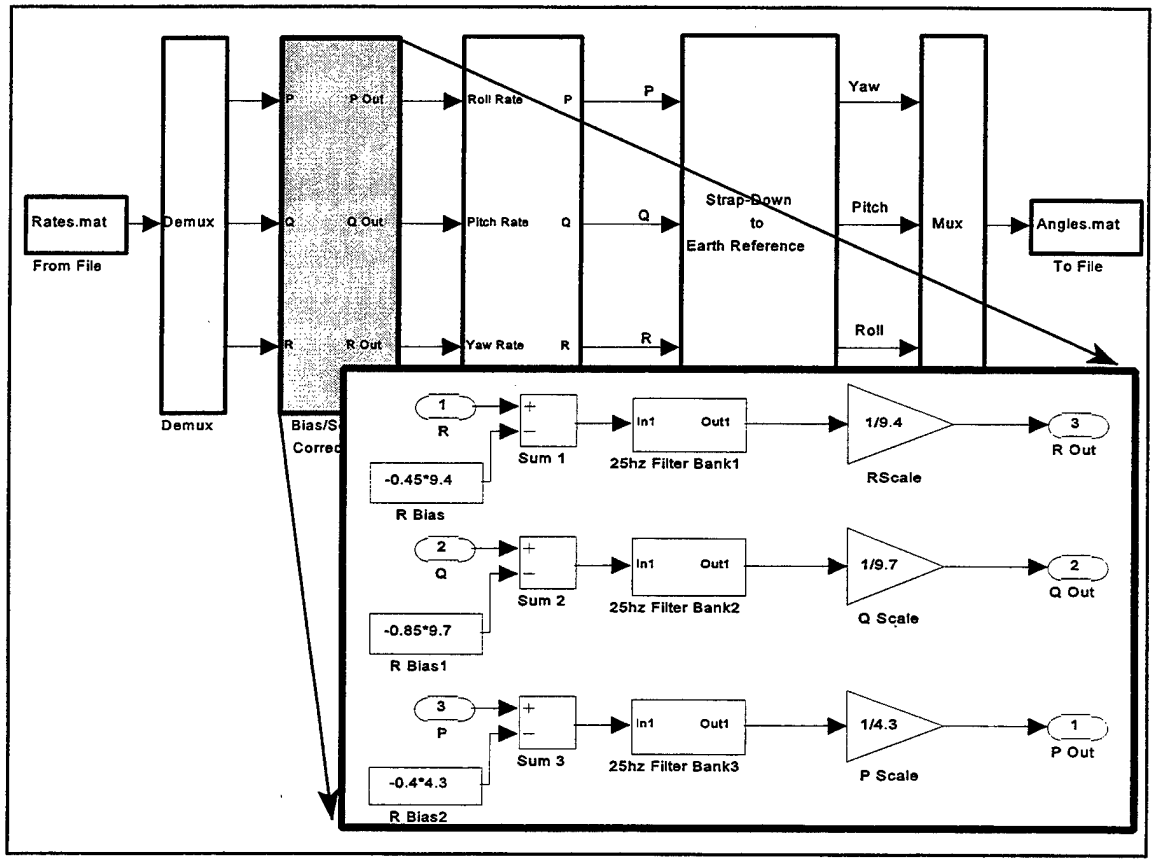

Figure 7. Bias/Scale Correction

D. SCALE CORRECTION

As discussed in Chapter II, sensors respond to rates by producing a voltage that is proportional to the angular rate measured along the input axis. This voltage is sensed by the telemetry system and transmitted in the form of values that represent the number of telemetry bits per degree per second. Each data point is then scaled to represent degrees per second, which is also accomplished in the "bias/scale correction" block, Figure (7), by way of a gain factor that is set equal to the inverse of the scale factor. Scaling values for the sensors used during initial testing were obtained experimentally by comparison with rates constructed from Carco table angles (ground truth). Again, this is discussed further in Chapter IV. This technique is not feasible for live-fire scenarios. These scale factors are sensor dependent and must be obtained through pre-flight testing since there is no reference available for comparison after missile firing. Manufacturer specifications for the QRS-11 quartz rate sensor indicate that the nominal scale factor is five millivolts per degree per second. Nominally, the scale factors would be measured precisely using a three-axis flight motion simulator.

E. SENSOR CROSS COUPLING

In order to ensure that angular motion in any of the three axes results in a sensor indication in that axis only, the IMU was assembled so that the sensors are mutually perpendicular with exceptional precision. The physical assembly of the IMU still resulted in some cross coupling between axes. This mis-alignment is taken into account in the "Cross Coupling Correction" block, shown in Figure (8).

Cross coupling correction is accomplished via gain blocks that are set to a value

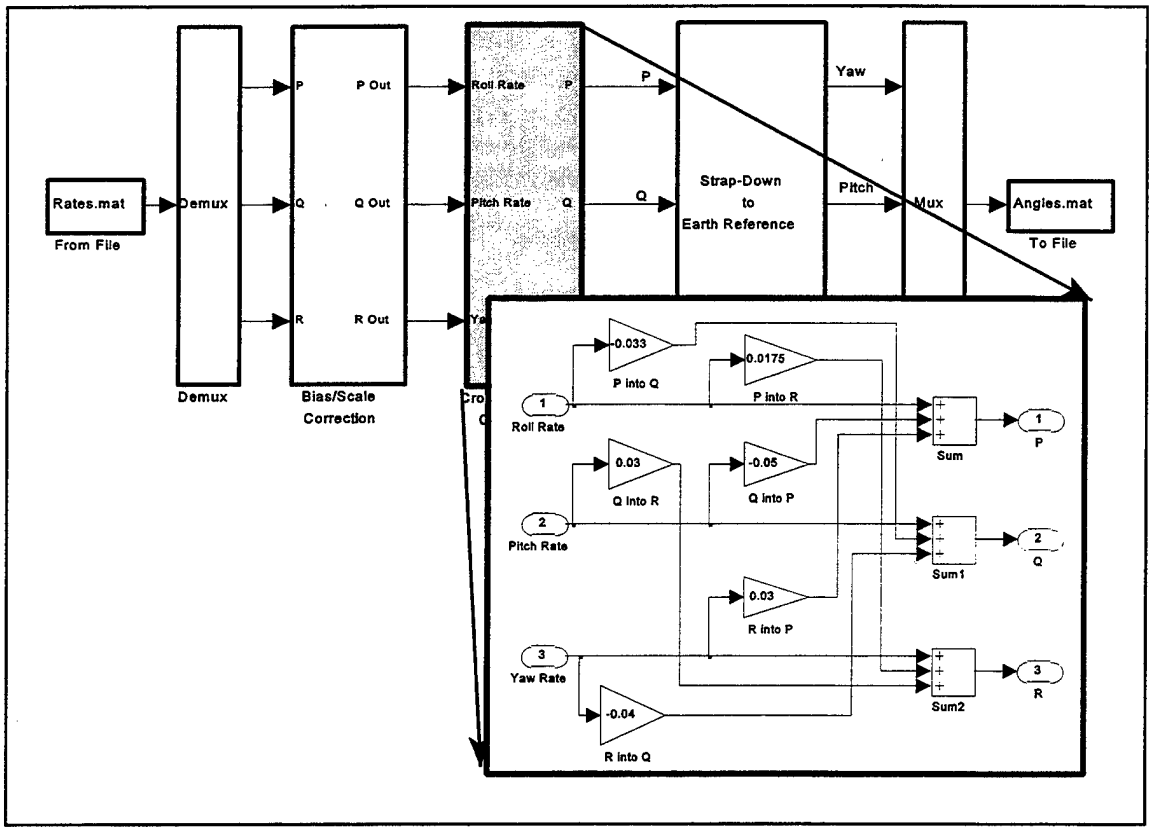

Figure 8. Cross Coupling Correction

that divides or multiplies the input rates to correct for the percentage of angular rate that is coupled from one sensor to another. Cross coupling is anticipated to range from one to five percent. Values obtained from the IMU package under test fell within this range. At this point, the roll, pitch and yaw rates had been transformed into a correct representation of the rates that existed during the missile test "flight."

F. METHOD OF ROTATION AND INTEGRATION

The preferred method of rotation for this particular case is the Euler Rotation due to its computational efficiency and stability. This method uses three angles to represent rotations around three coordinate axes. It is very efficient because it uses only three variables to represent three degrees of freedom. Additionally, Euler angles are inherently

stable and drift very little, so they don't require periodic readjustment. The rotation works as follows:

> Transform a point by rotating it counterclockwise about the Z (Yaw) axis by ψ degrees, followed by a rotation about the Y (Pitch) axis by θ degrees, followed by a rotation about the X (Roll) axis by ϕ degrees. (Bobick, 1999)

This is written as (ψ,θ,ϕ), where ψ is yaw, θ is pitch, and ϕ is roll. In general, there are 12 different conventions possible with respect to the direction of each rotation. The convention used throughout this thesis, shown in Figure (9), is as follows: Positive rotation is counterclockwise about the X, Y and Z axes (i.e., it follows a right-hand rule).

The major drawback of the Euler method is the "Gimbal Lock" phenomenon. Gimbal Lock occurs when a succession of 90 degree rotations results in a loss of one

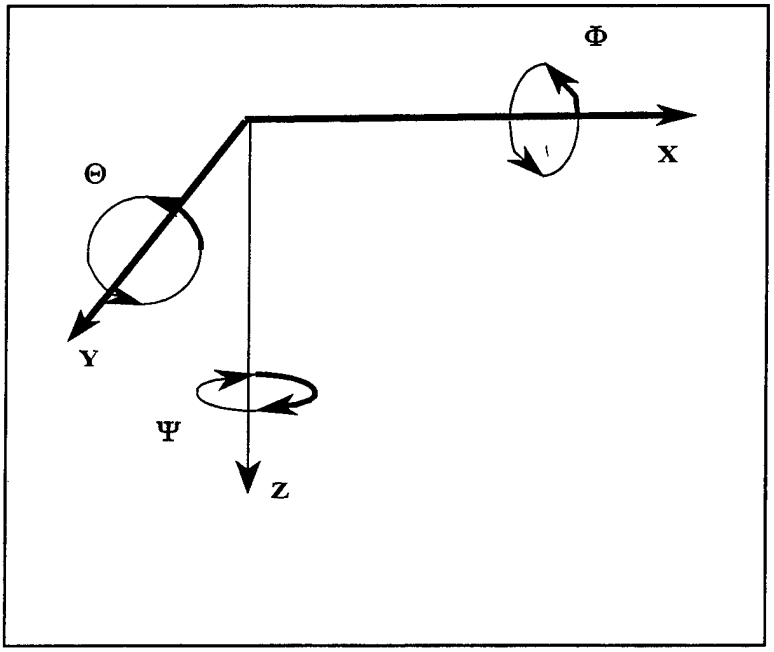

Figure 9. Euler Angle Representation (After Watt, 1992)

degree of freedom. For example, if a 90 degree rotation about the X axis is followed by a 90 degree rotation about the Y axis, then a subsequent rotation about the Z axis would have the same effect as a rotation about the X axis. The axes have become aligned in such a way as to remove one degree of freedom. For this particular application, however, it is very unlikely that a 90 degree rotation in pitch will occur since the missile would be pointing straight up and down. The efficient and stable nature of this method of rotation justifies its use despite the remote possibility of encountering the Gimbal Lock phenomenon.

As noted in the introduction, a general Euler rotation matrix is given in Figure (2). ψ is the yaw angle, θ is the pitch angle, and ϕ is the roll angle, all in the earth's reference frame. The "dot notation" is used in the conventional way to denote the first derivative with respect to time. R, Q, and P are the angular rates in the missile's frame of reference. The order in which the rotations about each axis are performed determines the equations to be used. These equations apply to rotations in yaw, pitch, and roll, in that order. It was necessary to recast the equations given in Figure (2) in a form that is more readily implemented in a Simulink block diagram. It is easily shown that the following equations are equivalent.

$$\frac{d\psi}{dt} = Q\frac{\sin\phi}{\cos\theta} + R\frac{\cos\phi}{\cos\theta} \tag{3.1}$$

$$\frac{d\phi}{dt} = P + \frac{d\psi}{dt}\sin\theta \tag{3.2}$$

$$\frac{d\theta}{dt} = Q\cos\phi - R\sin\phi \tag{3.3}$$

Integration and rotation both occur in the "Euler Rotation" block as shown in Figure (10). This figure merely depicts a block diagram representation of the equations shown above, along with the integration necessary to convert angular rates to angles. Initial values for the earth referenced angles are inserted into the model as initialization values for the integrators.

G. ANIMATION

Once the Euler Rotation Model was validated, an animation model was constructed to provide a visual reconstruction of missile attitude throughout flight. The model block diagram is shown in Figure (11), along with a view of the animation run-time figure. The heart of this model is the "animator" block, which invokes a Simulink

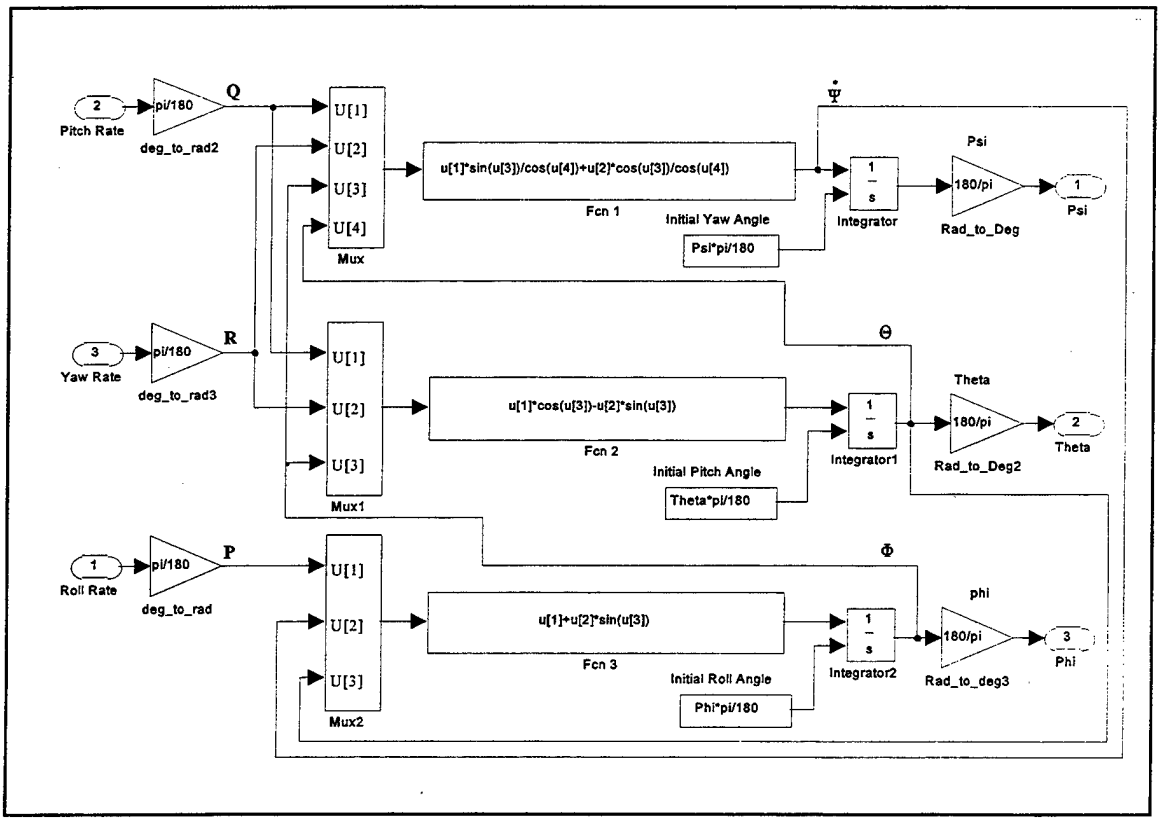

Figure 10. Euler Rotation

"S-file." Simulink allows the user to implement programs written in Matlab code or the C programming language to accomplish functions not otherwise available in the Simulink standard block library. In this case, the S-file "Animator" drives the animation by calling two other Matlab M-files: Draw.m and Redraw.m. Draw.m is invoked upon model initialization to draw the animation figure and all associated graphics objects. The figure and three dimensional axes are drawn using Matlab handle graphics and the aircraft is drawn with Matlab's "patch" command. Redraw.m erases and redraws the aircraft on each pass through the model, whereas the figure and axes remain fixed throughout the animation. The aircraft graphics object is composed of two perpendicular triangles that are described by their vertices. The vertices are rotated to their updated position on each execution of the S-file and the aircraft is redrawn in the correct postion using the "patch" command.

Redrawing the aircraft on each pass can become computationally costly, so it is possible to modify the Animator S-file so that the aircraft is redrawn at some specified interval. This might be necessary if the processing speed of the platform running the model doesn't provide the CPU power necessary to display the animated flight attitude with sufficient quality. Implementation details are contained in Chapter 9 of the Simulink Users Manual. All Matlab computer code is included as Appendix B.

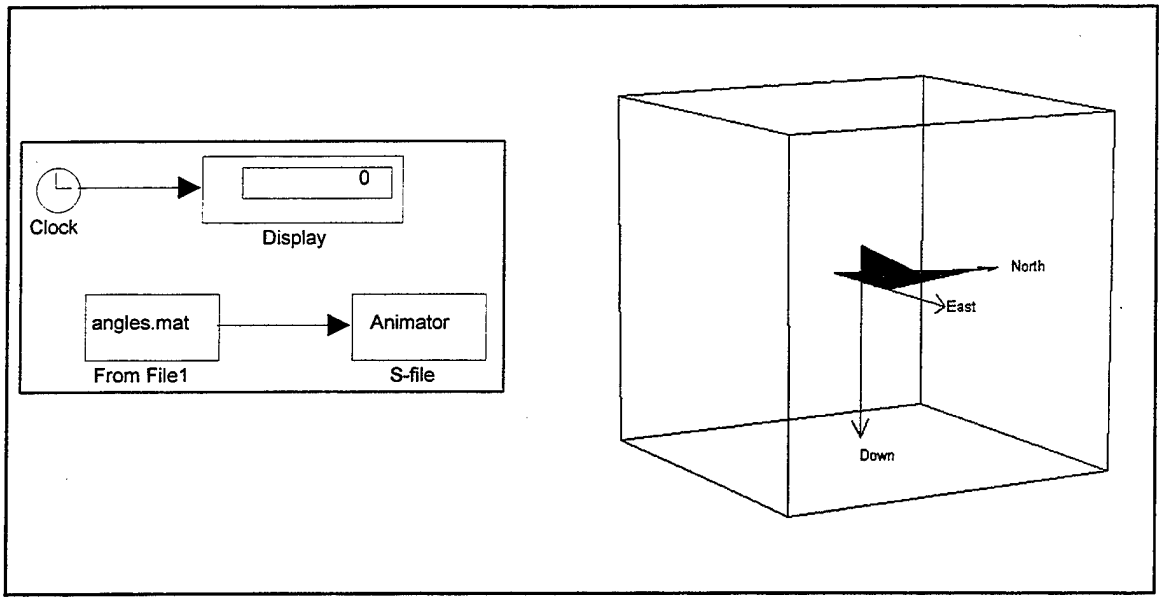

Figure 11. Animation Model and Figure

IV. MODEL VALIDATION

The Euler rotation model was tested thoroughly to ensure that it would re-create missile attitude with high accuracy. The testing procedure included the following tests:

A. CORRECTNESS AGAINST SPECIFICATION

A Hardware-In-The-Loop (HWIL) flight was conducted by NAWC China Lake using a five-axis flight motion simulator facility manufactured by Carco Electronics. The simulator and its capabilities are described in more detail in Appendix A. The "flight" included substantial inertial rates in all three coordinate planes. Missile strapdown angular rates sensed by the quartz rate gyros on board the missile were recorded as yaw (R), pitch (Q), and roll (P) rates. Actual missile strapdown angles, or "Ground Truth" angles, were taken directly from the Carco table facility. Figure (12) shows the resulting

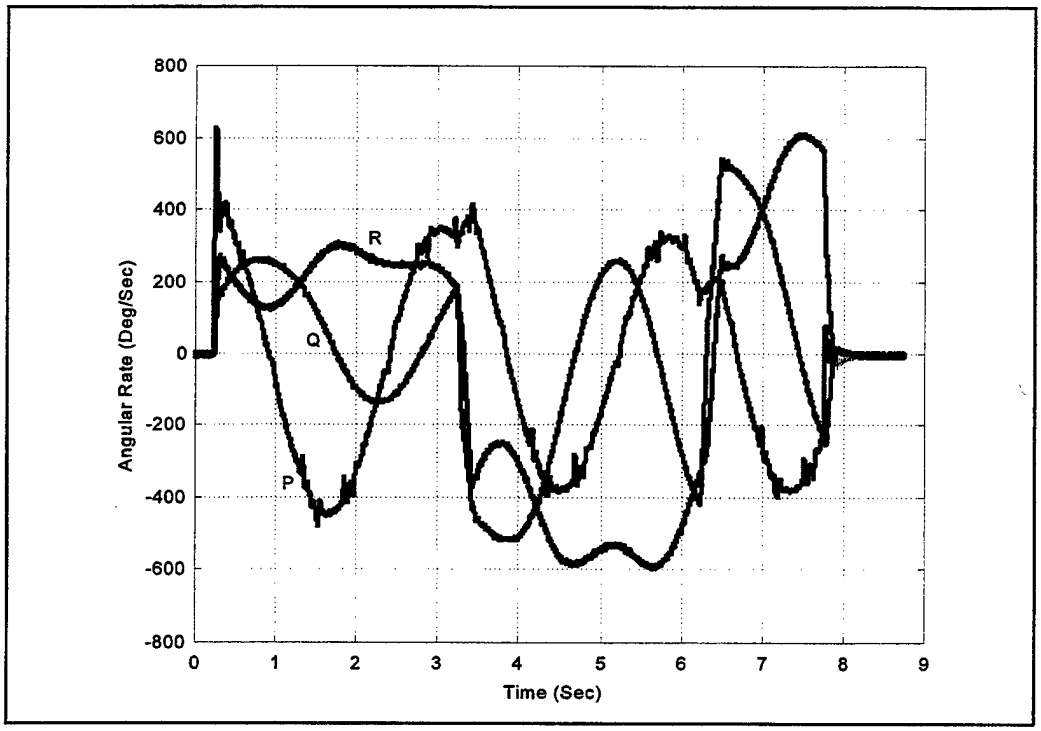

Figure 12. Quartz Sensor Rates

strapdown rates sensed by the quartz sensors, while Figure (13) depicts the sequence of angles used during the simulated run. These six measurements, along with timing data, were provided for use in model testing. NAWC China Lake required an accuracy of 2° RMS in each axis to meet project requirements.

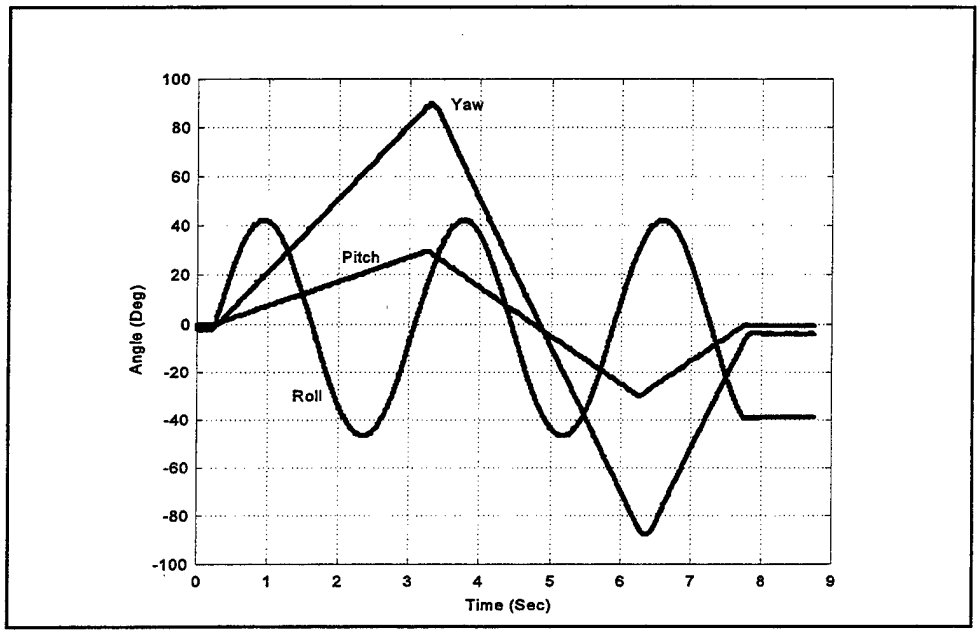

Figure 13. Carco Table Angle Sequence

As noted in Chapter IV, each quartz sensor has a characteristic bias voltage that was compensated for before the sensor data was used for analysis. The bias for each sensor was obtained directly from the test data that was generated during the test "flight." As can be seen in Figure (12), there is a brief period at the beginning of the run where the missile is stationary. At this point in the run, any voltage generated by the sensors is a result of bias alone. These biases were calculated by taking mean values from the sensor data during the inactive period. Alternately, bias may be measured directly, prior to live firing.

Although cross coupling values can be obtained by conducting HWIL runs that include rates that are confined to each individual axis in turn, this particular test run did not include the required sequence of rates, so it was necessary to obtain the cross coupling values through comparison of the sensed rates with "Ground Truth" rates. The "Ground Truth" rates were obtained by differentiating the Carco table angles and transforming them with an inverse Euler rotation. The model depicted in Figure (14) was constructed for the purpose of converting Carco table angles to "Ground Truth" strapdown rates.

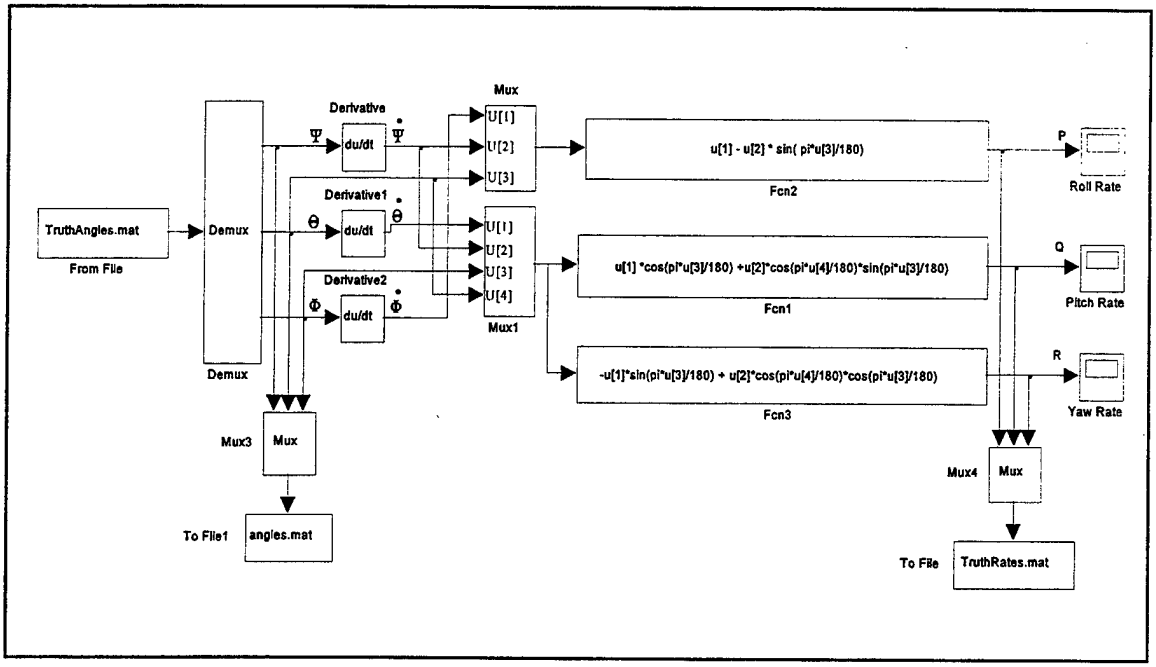

Figure 14. Earth Angles to Strapdown Rates Conversion

It was then possible to identify areas where the sensed rates varied from predicted values. Closer inspection of these deviations yielded correlations between the areas of deviation and the behavior of the other two rates during the same time frame. For example, in the indicated portion of Figure (15) it was apparent that pitch rate deviates

from the predicted value in the negative direction precisely when the roll rate is less than zero, indicating that the pitch sensor was sensing motion in the roll axis. Quantification of the cross coupling was accomplished through a trial and error approach. In this case,

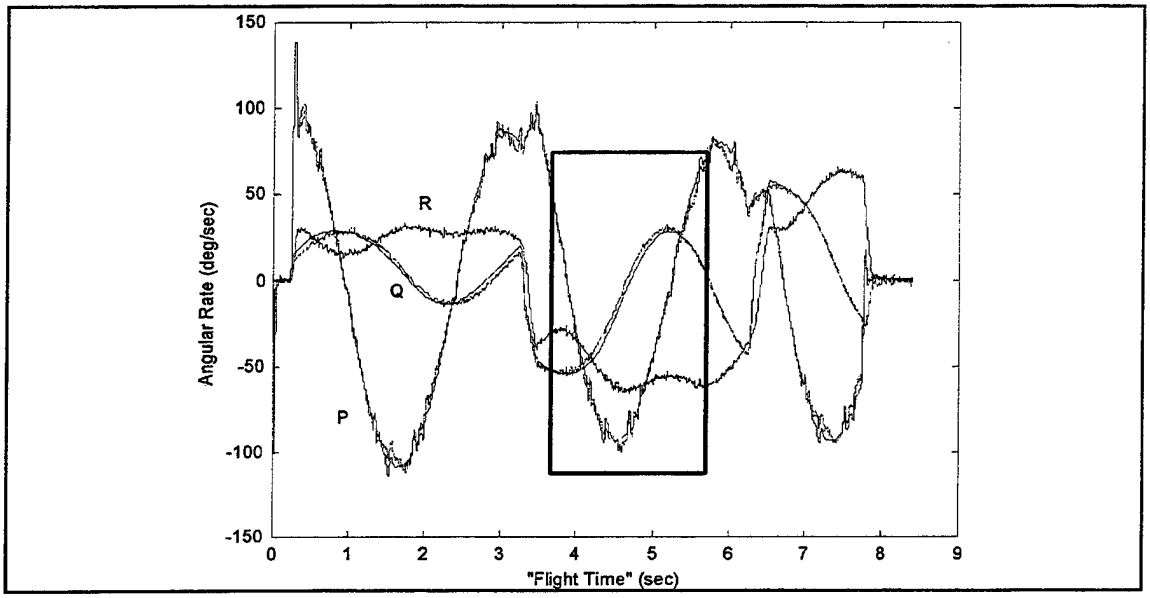

Figure 15. Determining Cross Coupling by Inspection

the peak roll rate near 4.5 seconds was approximately -90 degrees per second. At the same point in the run the pitch rate diverges from the predicted value by about 3 degrees per second. The magnitude of cross coupling from the roll axis into the pitch axis was taken to be near three percent. The pitch rate was corrected by an amount equal to three percent of the roll rate throughout the run and the plot shown in Figure (15) was re-generated. After several iterations it was determined that there was a 3.3 percent cross coupling from roll rate into pitch rate. The remaining five coupling errors (pitch into yaw, pitch into roll, yaw into roll, etc.) were determined in similar fashion, with values ranging from one percent to five percent.

Once cross coupling compensation was complete it was possible to determine the scale factor for each sensor. Although these values can be obtained from manufacturer specifications, scale factor for a particular sensor may differ from specifications by as much as one percent. It was therefore important that the scale factor be known precisely for each sensor. These values were obtained in a trial-and-error fashion by comparing the magnitude of sensor indications with ground truth rates. Note that scale factors must be obtained prior to missile firing, as there is no method of comparison after the fact.

After correcting for bias, scale, and cross coupling, the test data were suitable for use in assessing the operation of the model. The simulation was run with data corrections as noted above. The resulting earth referenced angles were compared to the angles taken directly from the Carco table, with the differences between the generated angles and expected values shown in Figure (16). The mean differences between angles produced by the rotation model and the expected angles are also included in Figure (16).

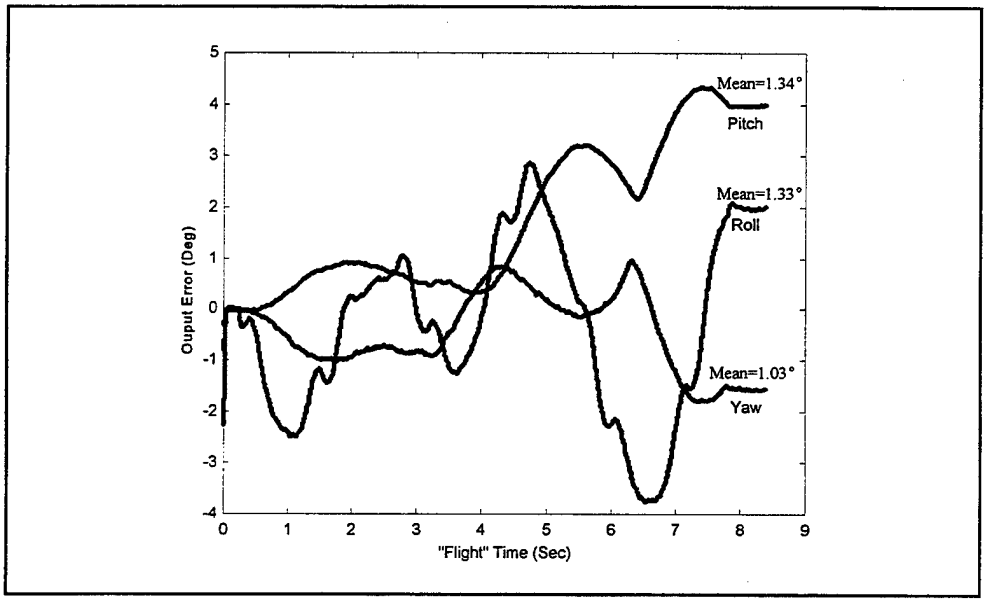

Figure 16. Output Differences

B. BIAS SENSITIVITY

In order to assess the model's sensitivity to errors in bias, it was necessary to operate the model with known bias errors of varying magnitude. Data from the same HWIL test flight were used. Figure (17) represents the error in output that resulted from imprecise bias correction in the missile's yaw axis. Output angles were affected in similar fashion when bias correction in the roll and pitch axes were in error. As shown in Figure (17), bias should be determined to within ½ degree to ensure acceptable output accuracy.

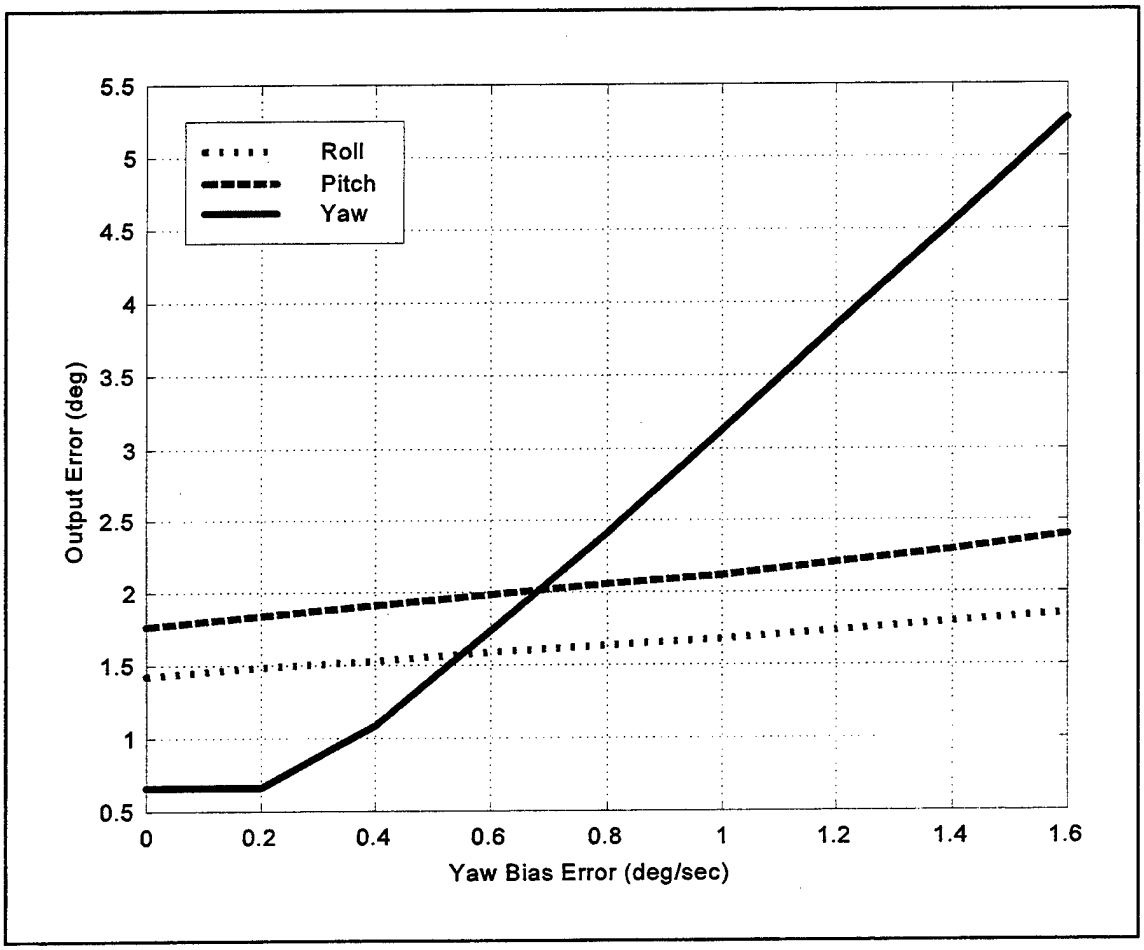

Figure 17. Output Error Due to Incorrect Yaw Bias

C. SCALE SENSITIVITY

Again, the data resulting from the HWIL run were used to test the model for output accuracy with errors in scale factor of varying magnitude. The results shown in Figure (18) represent the output errors in each axis caused by incorrect scale factor in the missile's yaw axis. Comparable results were obtained when scale factor for the missile pitch and roll axes were intentionally skewed.

D. NOISE SENSITIVITY

Susceptibility to input noise is a useful metric in the overall evaluation of the model since data generated during a live missile firing will likely be degraded somewhat by the presence of noise. It was therefore necessary to test the model in a way similar to

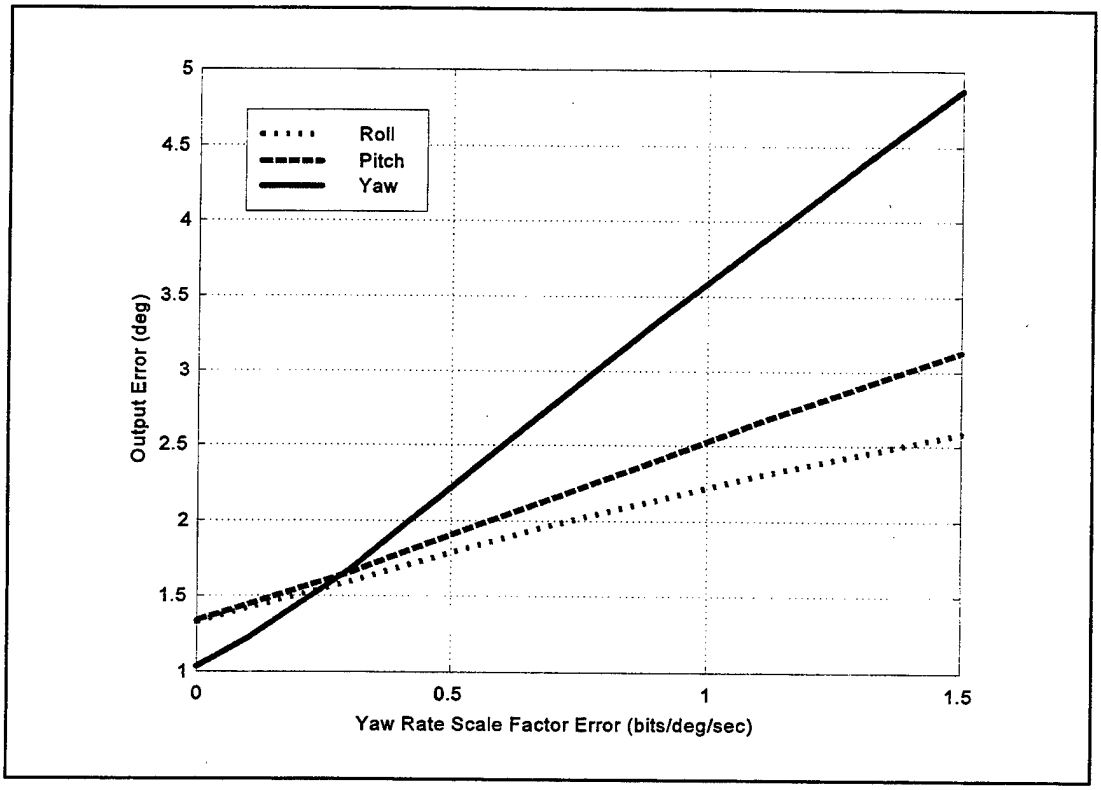

Figure 18. Output Error Due to Incorrect Yaw Rate Scale Factor

the testing with respect to bias and scale factor, but with band-limited white noise added to all three sensor inputs. Simulink provides a "Band-Limited White Noise" block that generates normally distributed random numbers to simulate Gaussian white noise. Although true continuous white noise has a correlation time of 0, a flat Power Spectral Density (PSD), and a covariance of infinity, the simulated noise was a useful approximation because the noise disturbance correlation time was very small compared to the shortest time constant of the system under test. The noise power values used in testing actually represented the height of the PSD of the white noise due to a scale factor that is introduced to reflect the implicit conversion from a continuous PSD to a discrete noise covariance. PSD is measured in units of W/Hz. Figure (19) depicts the results of the noise sensitivity testing. The values used in noise testing far exceeded the quartz rate sensor's specified voltage self-noise spectral density of ≤ 0.02 °/sec/(Hz)$^{1/2}$, so the model demonstrated excellent stability in this regard.

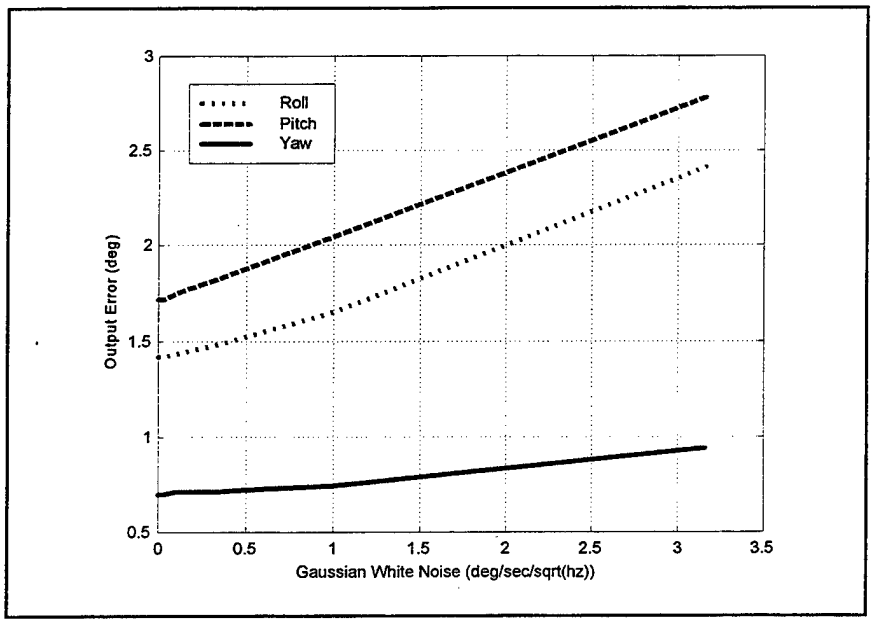

Figure 19. Output Error Due to Gaussian White Noise

E. IMPACT OF GIMBAL LOCK

The error resulting from the gimbal lock phenomenon must be quantified to establish the region in which the pitch may vary while maintaining acceptable output accuracy. A sequence of yaw, pitch, and roll angles that approximates a worst-case secenario is shown in Figure (20). This scenario ensures that the pitch dwells near 90 degrees for extended periods while there are significant angular rates in the yaw and roll planes. This "worst-case" scenario was used as the input during testing that consisted of adjusting the maximum pitch angle increasingly closer to 90 degrees to ascertain that point at which the error in output exceeds two degrees RMS. The errors corresponding to the different maximum pitch values are shown in Figure (21). As can be seen, the model performs within specifications as long as the pitch remains below 87 degrees.

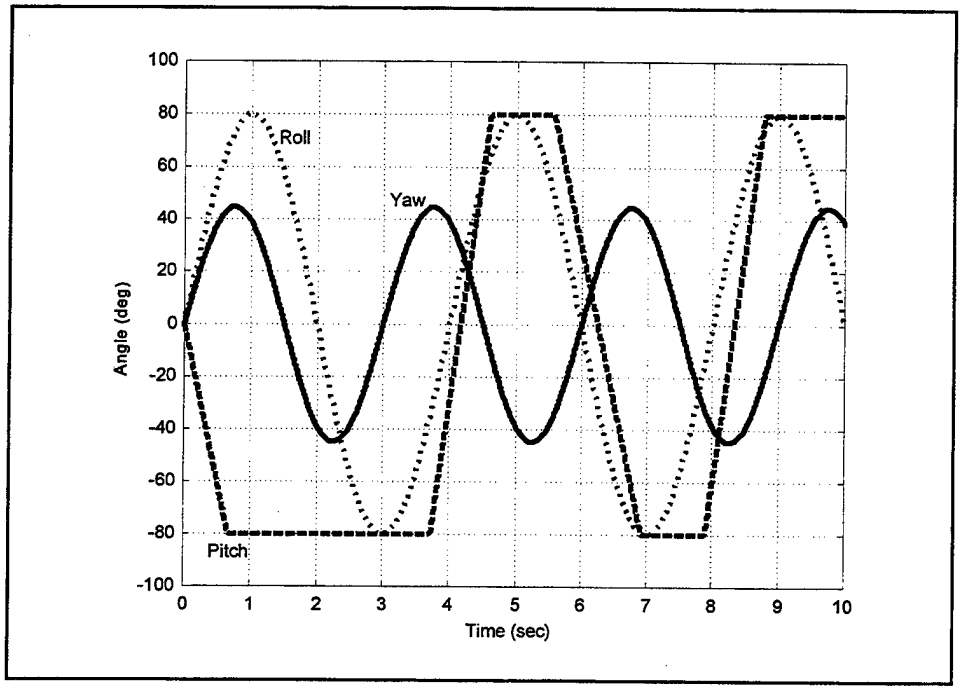

Figure 20. Gimbal Lock Testing "Worst Case" Input Scenario

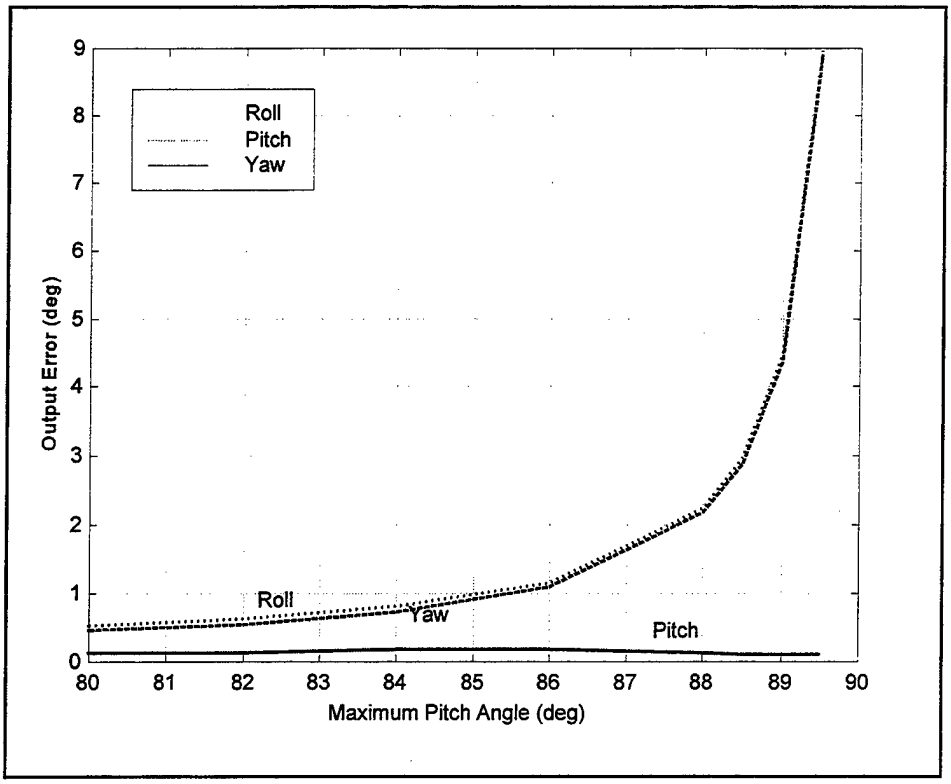

Figure 21. Output Error Near the Gimbal Lock Region

V. CONCLUSIONS AND AREAS FOR FURTHER RESEARCH

Enhancing air survivability in the face of the deadly threat of infrared missiles is clearly a top DoD priority. Recent history demonstrates the need for an effective defense. However, the small size of tactical missiles excludes the use of traditional gyro-based inertial navigation, posing significant difficulties in testing countermeasures. The approach outlined in this thesis has proven to be a workable solution to this important problem. Size constraints are rendered virtually immaterial with the use of quartz rate sensors that measure less than two inches in diameter. Budget concerns are also addressed through the use of these inexpensive sensors, reducing costs by a factor of five or more.

Perhaps as important as the sensor benefits are the analysis features offered by the Simulink software tool. Post-flight reconstruction is quick and straightforward, enabling researchers to analyze, process, and display flight data with simple mouse operations. The Euler rotation model itself is displayed in block diagram format and can be modified easily with "click-and-drag" operations. Simulink features software oscilloscopes, data displays, and file handling tools. Matlab handle graphics for use in animation applications are limited only by the skill of the individual designing the algorithm.

A. SENSOR PERFORMANCE

The primary research questions that prompted this investigation have been conclusively answered. The QRS-11 quartz rate sensor shows exceptional promise for use in tactical missiles due to its compact size, low cost, excellent dynamic range, and durability.

As mentioned above, the QRS-11 measures less than two inches in diameter and can be mounted in the smallest tactical missiles. This micro-miniature, solid state device serves the same purpose as a gyroscope, but without the hundreds of precision parts and high-speed spinning rotor. Of course, processing of the flight data is required, but this is a small price to pay for the benefits accrued.

The QRS-11's lack of moving parts means that it should have a virtually "unlimited" operating life and should perform reliably under harsh conditions. Table 1 in Chapter II lists important operating characteristics for the quartz rate sensor. Of particular note are the sensor's excellent linearity and low self-noise. These qualities are particularly critical in this application, where deviations from correct operation result in errors that are accumulate throughout model operation.

B. EULER ROTATION MODEL PERFORMANCE

The Euler rotation model as described in Chapter IV went beyond project specifications. Simulink integration solver routines posed no limitation with respect to output accuracy. The Euler rotation is a very effective technique for these purposes, but is limited by the gimbal lock phenomenon. Future efforts in this research area should consider use of another rotation method such as the Quaternion. The all-attitude nature of the Quaternion, coupled with the flexibility afforded by Simulink in correcting the Quaternion's mathematical ambiguities, portend a rotation method free from attitude singularities.

Correct operation of the model requires knowledge of sensor bias and scale factor to within ½ degree per second. In addition, sensors must be mounted orthogonally in the

IMU with great precision to minimize cross coupling between strapdown axes. Cross coupling values (there will be some) can be obtained through Hardware-In-the-Loop (HWIL) testing. An accurate model output is contingent on the precision with which the bias, scale, and cross coupling values are known.

C. ANIMATION

The animation for this project was accomplished using Matlab handle graphics and was implemented through Matlab script files bearing the .m extension that are invoked from a Simulink S-file block. The technique used for rotating the aircraft to its new position and re-drawing it requires significant processing power to ensure smooth animation. Further research should focus on the use of interpolation and a Quaternion matrix approach to aircraft rotation.

Interpolation is a graphics technique used extensively in the computer gaming industry. Objects that are to be rotated to a new position are not moved directly to the final attitude. Rather, intermediate points are selected and the object is rotated to each of the transitional points to give a smoother look to the animation. The shorter rotations ensure that the object does not "jerk" from one position to the next. The animation algorithm listed in Appendix B, which does not use interpolation, will appear smooth on a processor operating at speeds exceeding 300 MHz.

The Quaternion rotation technique lends itself to the interpolation process due to the manner in which the rotation path is selected. Also, Matlab operates much more efficiently when computation is in matrix form.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A. FIVE AXIS FLIGHT MOTION SIMULATOR

The use of motion platforms to simulate missile test flights furnishes considerable cost savings in development of modern missiles. Live fire testing constrains the designer to a single test scenario iteration. Also, extreme maneuvers are usually avoided during live fire testing due to the risk of sending the missile out of control and wasting the entire test. Test engineers will often resort to safer scenarios to ensure at least moderate success in data gathering. This is a significant shortcoming given the fact that tactical missiles routinely experience acute inertial rates.

The flight motion simulator (FMS) must be capable of exercising the test unit in at least three independent axes. The FMS facility used by NAWC China Lake can operate with five degrees of freedom, although the testing described in this document only required three-axis operation. The physical gimbal configuration can be seen in Figure (22). Rigorous testing dictates that substantial rates occur in all three planes.

The FMS uses hydraulic actuators for the yaw and pitch gimbles to accommodate the large angular rates required to test tactical missiles, while the roll axis relies on an electric motor to generate angular motion. Maximum velocities in excess of 1300 degrees per second can be simulated in the roll axis, while the yaw and pitch axes are capable of rates in the 250 degrees per second range. Each gimbal can be positioned with an accuracy of $\pm$ 0.1 degrees, while velocity commands are executed with an accuracy of $\pm$ 1 degree per second.

FMS facilities are generally very expensive, but their use is justified. The cost of test flights can be reduced by a factor of 5000 or more. Additionally, test engineers are

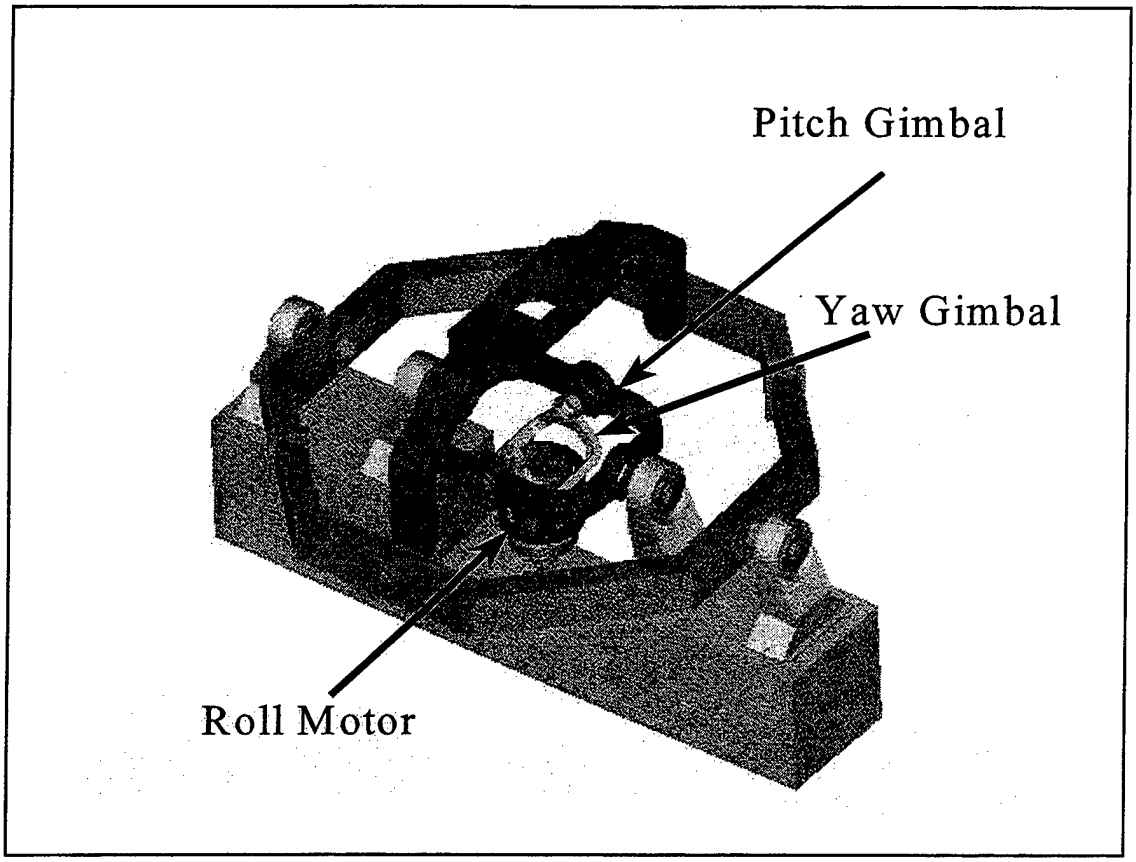

Figure 22. FMS Gimble Configuration
(From Model S-458R-5T, 1998)

given added flexibility and freedom in generating test scenarios.

The tests performed on the IMU sensors used the three axes shown in Figure (22). The roll axis contains an electronically driven motor that creates the IMU roll motion. The yaw gimbal (actually the pitch axis in the simulation) allows the rolling IMU to be moved in pitch. The pitch gimbal (actually the yaw axis in the simulation) rotates the yaw gimbal to simulate motion about the yaw axis.

APPENDIX B. MATLAB COMPUTER CODE

Listed below is the computer code generated to accomplish the animation associated with the Simulink Euler rotation model.

A. ANIMATOR.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Written December 1998 by LT Troy M. Johnson     %
%                                                 %
% This S-file is called by the Simulink Model     %
% "Animate". Draw.m is called upon initialization %
% and Redraw.m is called to redraw the aircraft   %
% each pass through the model                     %
%                                                 %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [sys,x0,str,ts] = Animator(t,x,u,flag)

BlockHandle=gcb;

%If figure is already present get userdata from the object
WorkingFig=get_param(BlockHandle,'UserData');
if
~ishandle(WorkingFig)|~strcmp(get(WorkingFig,'Tag'),'QuatWorkingFig'),
  WorkingFig=[];
  set_param(BlockHandle,'UserData',[]);
end  % if

FigHandle=findall(0,'Type','figure','Tag','QuatWorkingFig');

%perform this S-function only if the figure is present
if isempty(FigHandle)

switch flag,

  %%%%%%%%%%%%%%%%%%
  % Initialization %
  %%%%%%%%%%%%%%%%%%
  case 0,
     [sys,x0,str,ts]=mdlInitializeSizes(WorkingFig);

  %%%%%%%%%%%
  % Outputs %
  %%%%%%%%%%%
  case 3,
     sys=mdlOutputs(t,x,u,WorkingFig);

  %%%%%%%%%%%%%%
  % Terminate %
  %%%%%%%%%%%%%%
  case 9,
    sys=mdlTerminate(t,x,u,WorkingFig);
```

```
  case {1,2,4},
    % Don't do anything

  %%%%%%%%%%%%%%%%%%%
  % Unexpected flags %
  %%%%%%%%%%%%%%%%%%%
  otherwise
    error(['Unhandled flag = ',num2str(flag)]);

end  %end switch

end  %end if figure not already drawn
%
%=============================================================
% mdlInitializeSizes
% Return the sizes, initial conditions, and sample times for
% the S-function.
%=============================================================
%
function [sys,x0,str,ts]=mdlInitializeSizes(WorkingFig)

    draw(WorkingFig);           % This will draw the figure

sizes = simsizes;

sizes.NumContStates  = 0 ;
sizes.NumDiscStates  = 0 ;
sizes.NumOutputs     = 0 ;
sizes.NumInputs      = 3 ;
sizes.DirFeedthrough = 0 ;
sizes.NumSampleTimes = 1 ;

sys = simsizes(sizes);
x0  = [];
str = [];
ts  = [0 0];

%
%=============================================================
% mdlOutputs
% Return the block outputs.
%=============================================================
%
function sys=mdlOutputs(t,x,u,WorkingFig)

sys=[];

psi=u(1);theta=u(2);phi=u(3);        %get angles from block input
redraw(WorkingFig,psi,theta,phi);    %redraw aircraft

%
%=============================================================
% mdlTerminate
% Perform any end of simulation tasks.
%=============================================================
%
function sys=mdlTerminate(t,x,u,WorkingFig)
```

```
sys=[];
```

B. DRAW.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Written by LT Troy Johnson January 1998                                 %
%                                                                         %
% Called by Animator.m when the S-file block is invoked.                  %
% Draws the figure, axes, and aircraft                                    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%check if figure is already on screen
[flag,fig] = figflag('Missile Attitude');

%if figure is already up, do nothing and return
if flag
   return;
end

%%% General Info.
Black      =[0       0        0      ]/255;
White      =[255     255      255    ]/255;
UIBackColor=get(0,'DefaultUIControlBackgroundColor');
FigColor=UIBackColor;
UIForeColor=Black;

%%% Set Positions
ScreenUnits=get(0,'Units');
set(0,'Units','pixels');
ScreenSize=get(0,'ScreenSize');
set(0,'Units',ScreenUnits);

FigWidth=750;
FigHeight=530;
FigPos=[(ScreenSize(3:4)-[FigWidth FigHeight])/2 FigWidth FigHeight];

%%% Create InputFig
QuatFig=figure('BackingStore'   ,'off'                    , ...
               'Color'          , FigColor                , ...
               'Name'           ,'Missile Attitude'       , ...
               'NumberTitle'    ,'off'                    , ...
               'Pointer'        ,'arrow'                  , ...
               'Position'       , FigPos                  , ...
               'Renderer'       ,'zbuffer'                , ...

               'Tag'            ,'QuatWorkingFig'         , ...
               'IntegerHandle'  ,'off'                    , ...

               'Visible'        ,'off'                    );

%%% Create axes

QuatAxes=axes('Tag'              ,'Quaternion Axes', ...
              'Units'            ,'pixels'         , ...
              'DataAspectRatio'  ,[1 1 1]          , ...
              'PlotboxAspectRatio',[1 1 1]         , ...
              'View'             ,[60 10  ]        , ...
```

```
                'Box'               ,'on'              , ...
                'Color'             ,Black             , ...
                'XColor'            ,White             , ...
                'YColor'            ,White             , ...
                'ZColor'            ,White             , ...
                'DrawMode'          ,'fast'            , ...
                'Projection'        ,'perspective'     , ...
                'XLim'              ,[-100 100]        , ...
                'XTick'             ,[]                , ...
                'YLim'              ,[-100 100]        , ...
                'YTick'             ,[]                , ...
                'ZLim'              ,[-100 100]        , ...
                'ZTick'             ,[]                , ...
                'Visible'           ,'on'              );


set([QuatFig,QuatAxes],'HandleVisibility','on');
figure(QuatFig)
rotate3d on;

%Create arrows and plane

set(allchild(QuatFig),'Units','normalized');

ArrowLineX=[ 0 90  80 90 80
             0  0   0  0  0
             0  0  -5  0  5
           ]';
ArrowLineY=[ 0  0  -5  0  5
             0 90  80 90 80
             0  0   0  0  0
           ]';
ArrowLineZ=[ 0   0   0   0   0
             0   0  -5   0   5
             0  90  80  90  80
           ]';

for lp=1:3,
  LineHandles(lp,1)=line('XData'    ,ArrowLineY(:,lp) , ...
                         'Ydata'    ,ArrowLineX(:,lp) , ...
                         'ZData'    ,-ArrowLineZ(:,lp), ...
                         'Color'    ,[1 1 1]          , ...
                         'Parent'   ,QuatAxes         , ...
                         'LineWidth',1                , ...
                         'Visible'  ,'on'               ...
                         );
end % for lp
LineText(1)=text(0,100,0,'North','Color',[1 1 1],'Parent',QuatAxes);
LineText(2)=text(92,0,0,'East','Color',[1 1 1],'Parent',QuatAxes);
LineText(3)=text(0,0,-100,'Down','Color',[1 1 1],'Parent',QuatAxes);

PointerHandle=line('XData'    ,0       , ...
                   'YData'    ,0       , ...
                   'ZData'    ,0       , ...
                   'Color'    ,[0 0 1] , ...
                   'Parent'   ,QuatAxes, ...
                   'LineWidth',2       , ...
                   'UserData' , ...
```

```
                    [ArrowLineX(:,3)';ArrowLineY(:,3)';, ...
                     ArrowLineZ(:,3)']    ...
                    );

PlaneX=[75  40
         0   0
         0   0
       ];
PlaneY=[ 0   0
        30   0
       -30   0
       ];
PlaneZ=[ 0   0
         0   0
         0 -20
       ];

for lp=1:size(PlaneX,2),
     PlaneHandles(lp)=patch(PlaneY(:,lp), ...
     PlaneX(:,lp),-PlaneZ(:,lp),[1 0 0], ...
                         'LineWidth',1      , ...
                         'Parent'   ,QuatAxes, ...

                         'EdgeColor',[0 0 0] , ...
                         'EraseMode','normal'    ...
                         );
  set(PlaneHandles(lp), ...
     'UserData',[PlaneX(:,lp)';PlaneY(:,lp)';PlaneZ(:,lp)'] ...
     )
end % for lp

%store aircraft and axes graphics handles for next use
set(QuatFig,'UserData',[PlaneHandles QuatAxes]);

WorkingFig=QuatFig;

%store QuatFig location for use in redraw.m
set_param(gcb,'UserData',QuatFig);
```

C. REDRAW.M

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Written by LT Troy Johnson                                                    %
%                                                                               %
%                                                                               %
% This m-file is called by Animator.m on each pass through the                  %
% Animate.mdl Simulink model.  It redraws the aircraft object                   %
% by rotating each vertex of both of the triangles that comprise                %
% the aircraft.                                                                 %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function redraw(WorkingFig,psi,theta,phi);

QuatFig=get_param(gcb,'UserData');
data=get(QuatFig,'UserData');
PlaneHandles=data(1:2);
QuatAxes=data(3);
```

```
%convert to radians
psi=psi*pi/180;
theta=theta*pi/180;
phi=phi*pi/180;

%transform factors

%points one and two
p1= cos(theta)*cos(psi);
p2= cos(phi)*sin(psi)*cos(theta)-sin(phi)*sin(theta);
p3= cos(theta)*sin(psi)*sin(phi)-sin(theta)*cos(phi);

%points three and four
p4=-cos(theta)*sin(psi)*cos(phi)+sin(theta)*sin(phi);
p5= cos(phi)*cos(psi);
p6=cos(theta)*sin(phi)+sin(theta)*sin(psi)*cos(phi);

%point five
p7= sin(theta)*cos(psi);
p8=-sin(phi)*cos(theta)+cos(phi)*sin(psi)*sin(theta);
p9= cos(theta)*cos(phi)+sin(theta)*sin(psi)*sin(phi);

%define vertex values
x01=75; x02=40; y0=30; z0=-20;

%rotate point one (plane nose)
xn1=p1*x01; yn1=p2*x01; zn1=p3*x01;

%rotate point two (forward vertex of tail)
ax=x02/x01;
xn2=ax*xn1; yn2=ax*yn1; zn2=ax*zn1;

%rotate point three (right wing tip)
xn3=p4*y0; yn3=p5*y0; zn3=p6*y0;

%rotate point four (left wing tip)
xn4=-xn3; yn4=-yn3; zn4=-zn3;

%rotate point five (top of tail)
xn5=p7*z0; yn5=p8*z0; zn5=p9*z0;

%rotate point six (plane exhaust point)
xn6=0; yn6=0; zn6=0;

%rearrange vertex cartesian coordinates in patch command format
PlaneX=[xn1 xn2
        xn3 xn5
        xn4 xn6];
PlaneY=[yn1 yn2
        yn3 yn5
        yn4 yn6];
PlaneZ=[zn1 zn2
        zn3 zn5
        zn4 zn6];

%draw the figure

for lp=1:size(PlaneX,2),
```

```
  set(PlaneHandles(lp),'vertices',[PlaneY(:,lp),PlaneX(:,lp),-
PlaneZ(:,lp)])
 end % for lp

set(QuatFig,'UserData',[PlaneHandles QuatAxes]);

WorkingFig=QuatFig;

set_param('animate/animator','UserData',QuatFig);
```

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

Bobick, Nick, *Rotating Objects Using Quaternions - Using Euler Angles*, [http://www.gamasutra.com/features/programming/19980703/quaternions_04.html], July 1998.

BEI GyroChip Theory of Operation, [http://www.systron.com/theory.html], October 1998.

Blakelock, John H., *Automatic Control of Aircraft and Missiles*, Second Edition, Wiley-Interscience, 1991.

Eichblatt, Emil J. Jr.,*Test and Evaluation of the Tactical Missile*, Progress in Astronautics and Aeronautics, Volume 119, American Institute of Aeronautics and Astronautics, Inc., 1989.

Model S-458R-5T High Performance 5-Axis Electrohydraulic Simulator Operations and Instruction Manual, Carco Electronics, April 1988.

Naval Air Systems Command, *Electro-Optical/Infrared (EO/IR) Countermeasures (CM) Handbook*, Government Printing Office, Washington, D.C., July 1998.

Watt, Alan H., *Advanced Animation and Rendering Techniques*, Addison-Wesley, 1992.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 Johns J. Kingman Rd., STE 0944
   Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library 2
   Naval Postgraduate School
   411 Dyer Rd.
   Monterey, CA 93943-5000

3. Commanding Officer 1
   Attn: Mr. Larry Rollingson
   Naval Air Weapons Center, China Lake
   China Lake, CA 93555

4. D. Curtis Schleher 2
   Naval Postgraduate School
   589 Dyer Rd., Room 201E
   Monterey, CA 93943-5000

5. David C. Jenn 1
   Naval Postgraduate School
   ECE Department, Code EC/JN
   Monterey, CA 93943-5000

6. LT Troy M. Johnson 2
   604 Pleasant Way
   Chesapeake, VA 23320

7. Mr. and Mrs. R.D. Johnson 1
   414 Pine Street South
   Sauk Centre, MN 56378