

## N O T I C E

THIS DOCUMENT HAS BEEN REPRODUCED FROM  
MICROFICHE. ALTHOUGH IT IS RECOGNIZED THAT  
CERTAIN PORTIONS ARE ILLEGIBLE, IT IS BEING RELEASED  
IN THE INTEREST OF MAKING AVAILABLE AS MUCH  
INFORMATION AS POSSIBLE

# Shuttle Orbit IMU Alinement

## Single-Precision Computational Error

(NASA-TM-80793) SHUTTLE ORBIT IMU  
ALINEMENT. SINGLE-PRECISION COMPUTATION  
ERROR (NASA) 31 p HC A03/MF A01

N80-22379

Unclas  
G3/12 18480

Mission Planning and Analysis Division

March 1980



National Aeronautics and  
Space Administration

Lyndon B. Johnson Space Center  
Houston, Texas



80FM18

80-FM-18

JSC-16468

SHUTTLE PROGRAM

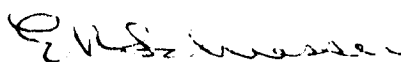
SHUTTLE ORBIT IMU ALINEMENT

SINGLE-PRECISION COMPUTATIONAL ERROR

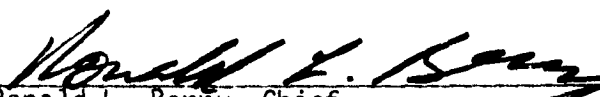
By C. R. McClain, McDonnell Douglas Technical Services Co.

JSC Task Monitor: T. J. Blucker, Mathematical Physics Branch

Approved:

  
\_\_\_\_\_  
Emil R. Schiesser, Chief  
Mathematical Physics Branch

Approved:

  
\_\_\_\_\_  
Ronald L. Berry, Chief  
Mission Planning and Analysis Division

Mission Planning and Analysis Division  
National Aeronautics and Space Administration  
Lyndon B. Johnson Space Center  
Houston, Texas  
March 1980

## CONTENTS

Section		Page
1.0	<u>SUMMARY</u> . . . . .	1
2.0	<u>INTRODUCTION</u> . . . . .	2
3.0	<u>DISCUSSION</u> . . . . .	3
3.1	DETERMINATION OF THE MEASURED PRESENT-CLUSTER- TO-M50-MATRIX . . . . .	3
3.2	COMPUTATION OF THE MISALINEMENT MATRIX AND ASSOCIATED ERROR QUATERNION . . . . .	4
3.3	DESCRIPTION OF EIGEN ROTATION ANGLE EXTRACTION METHODS . . . .	5
3.3.1	<u>Arc Cosine Method</u> . . . . .	5
3.3.2	<u>Arc Sine Method</u> . . . . .	6
3.4	DESCRIPTION OF TEST SETUP AND RUNS . . . . .	6
4.0	<u>RESULTS</u> . . . . .	8
5.0	<u>CONCLUSIONS AND RECOMMENDATIONS</u> . . . . .	9
APPENDIX A	- HAL/S CODE IN SINGLE PRECISION . . . . .	13
APPENDIX B	- HAL/S CODE IN DOUBLE PRECISION . . . . .	20

TABLES

Table		Page
1.0	SUMMARY OF IMU ONORBIT ALINEMENT HAL/S SIMULATION RESULTS . . . . .	10

FIGURES

Figure		Page
1.0	Misalignment error versus input misalignment angle . . . . .	12

## 1.0 SUMMARY

This paper presents the results of a study to determine the source of computational error in the IMU onorbit alignment software as specified by the Level C IMU SOP FSSR. Simulation runs were made on the IBM 360/70 computer with the IMU onorbit alignment software coded in HAL/S. The results indicate that for small IMU misalignment angles (less than 600 arc seconds), single precision computations in combination with the arc cosine method of eigen rotation angle extraction introduces an additional misalignment error of up to 230 arc seconds per axis. Use of the arc sine method, however, produced negligible misalignment error. As a result of this study, the arc sine method has been recommended by means of a software change request (CR) for use in the IMU onorbit alignment software. This CR has been approved for STS-1 and beyond.

## 2.0 INTRODUCTION

During IBM's Level 2 IMU onorbit alignment testing, it was first suspected that the largest source of software produced alignment error for small torquing angle misalignments might be the flight control utility routine MAT\_TO\_QUAT. After review of these initial test results, it was decided by JSC, IBM and MDTSCO that detailed studies of this problem be made at MDTSCO to confirm the initial conclusion. The proposed studies were aimed at determining whether the additional computational error was in fact caused by MAT\_TO\_QUAT or by the IMU alignment software design. The questionable design centered around the use of single precision computations in combination with the arc cosine method for extracting the eigen rotation angle from the error quaternion, from which torquing angles were computed. The results from these studies indicated that the latter was the cause of the error. Based on this conclusion, change request 1S128A was submitted and approved to change the eigenaxis rotation angle extraction method from the arc cosine method to the arc sine method for STS missions number 1 and beyond.

Section 3.0 contains a discussion of the equations which define the IMU onorbit alignment software and a description of the simulation runs actually made. Section 4.0 contains a discussion of the results, and Section 5.0 presents the conclusions and recommendations.



### 3.0 DISCUSSION

The IBM 360/70 computer was used in simulating the HAL/S version of the IMU onorbit alignment torquing angle software for this study. This computer was used for the following reasons: (1) the IBM 360/70 computer implements a HAL/S compiler which enables the simulation software to be coded in HAL/S, (2) this computer emulates the word size of the AP101 flight computer. Although the word size is equivalent for both computers, AP101 double precision computations are not as accurate as the IBM 360/70 computer; therefore, the double precision results presented in this study are not representative of AP101 double precision accuracy.

The following sections describe the simulation software and test runs for determining the source of computational error in the onorbit alignment software design.

#### 3.1 DETERMINATION OF THE MEASURED PRESENT-CLUSTER-TO-M50-MATRIX

In order to determine the measured transformation matrix from present cluster to M50 coordinates, the onorbit alignment software obtains measurements of two star LOS (Line of Sight) unit vectors in present cluster coordinates by means of the STAR\_TRACK or the COAS\_SIGHT mode, that is

$$\bar{S}_m = S_{xm}\bar{T}_{pc} + S_{ym}\bar{J}_{pc} + S_{zm}\bar{K}_{pc} \quad (1)$$

$$\bar{T}_m = T_{xm}\bar{T}_{pc} + T_{ym}\bar{J}_{pc} + T_{zm}\bar{K}_{pc} \quad (2)$$

In addition, the computer has stored in memory the exact LOS unit vectors in M50 coordinates of these two stars as follows:

$$\bar{S}_e = S_{xe}\bar{T}_{M50} + S_{ye}\bar{J}_{M50} + S_{ze}\bar{K}_{M50} \quad (3)$$

$$\bar{T}_e = T_{xe}\bar{T}_{M50} + T_{ye}\bar{J}_{M50} + T_{ze}\bar{K}_{M50} \quad (4)$$

These two sets of unit vectors, measured and exact, are used to compute the desired transformation matrix.

First, the exact LOS unit vectors given in M50 coordinates are used to form three unit vectors defining a star coordinate system expressed in the M50 coordinate system as follows:

$$\bar{U}_{xe} = \bar{S}_e \quad (5)$$

$$\bar{U}_{ye} = \text{UNIT} (\bar{S}_e \times \bar{T}_e) \quad (6)$$

$$\bar{U}_{ze} = \text{UNIT} (\bar{U}_{xe} \times \bar{U}_{ye}) \quad (7)$$

Next, the three unit vectors are used to form the columns of the transformation matrix from star to M50 coordinates:

$$\begin{bmatrix} \bar{T}_{S}^{M50} \end{bmatrix} = \begin{bmatrix} U_{xe}(1) & U_{ye}(1) & U_{ze}(1) \\ U_{xe}(2) & U_{ye}(2) & U_{ze}(2) \\ U_{xe}(3) & U_{ye}(3) & U_{ze}(3) \end{bmatrix} \quad (8)$$

In a similar manner, the measured LOS unit vectors (given in present cluster coordinates) are used to form this same star coordinate system expressed in the present cluster coordinate system as follows:

$$\bar{U}_{xm} = \bar{S}_m \quad (9)$$

$$\bar{U}_{ym} = \text{UNIT}(\bar{S}_m \times \bar{T}_m) \quad (10)$$

$$\bar{U}_{zm} = \text{UNIT}(\bar{U}_{xm} \times \bar{U}_{ym}) \quad (11)$$

These unit vectors are then used to form the rows of the measured\* transformation matrix from present cluster to star coordinates:

$$\begin{bmatrix} \tilde{T}_{PC}^S \end{bmatrix} = \begin{bmatrix} U_{xm}(1) & U_{xm}(2) & U_{xm}(3) \\ U_{ym}(1) & U_{ym}(2) & U_{ym}(3) \\ U_{zm}(1) & U_{zm}(2) & U_{zm}(3) \end{bmatrix} \quad (12)$$

Finally, the measured transformation matrix from present cluster to M50 coordinates is formed by the following matrix multiplication:

$$\begin{bmatrix} \tilde{T}_{PC}^{M50} \end{bmatrix} = \begin{bmatrix} \bar{T}_{S}^{M50} \end{bmatrix} \begin{bmatrix} \tilde{T}_{PC}^S \end{bmatrix} \quad (13)$$

### 3.2 COMPUTATION OF THE MISALIGNMENT MATRIX AND ASSOCIATED ERROR QUATERNION

The misalignment matrix is computed as a function of the stored transformation matrix from M50 coordinates to desired cluster coordinates (desired REFSMMAT) and the measured transformation matrix from present cluster to M50 coordinates, that is

$$\begin{bmatrix} \tilde{T}_{PC}^{DC} \end{bmatrix} = \begin{bmatrix} \bar{T}_{M50}^{DC} \end{bmatrix} \begin{bmatrix} \tilde{T}_{PC}^{M50} \end{bmatrix} \quad (14)$$

\* The tilde over the matrix indicates that the matrix is derived using onboard measured quantities and is to be distinguished from matrices without the tilde which are considered exact.

The error quaternion  $\Delta Q$  is defined by the following convention

$$\Delta Q = \begin{bmatrix} \Delta QS \\ \overline{\Delta QV} \end{bmatrix} = \begin{bmatrix} \cos \omega/2 \\ -\hat{u} \sin \omega/2 \end{bmatrix} \quad (15)$$

where  $\Delta QS$  is the scalar part of the quaternion,  $\overline{\Delta QV}$  the vector part,  $\omega$  the eigen rotation angle about the eigenaxis, and  $\hat{u}$  the eigenaxis unit vector. In order to compute the error quaternion associated with the misalignment matrix (14), the utility routine MAT\_TO\_QUAT is called, that is

```
CALL MAT_TO_QUAT (DCTPC)
ASSIGN ( $\Delta Q$ )
```

This error quaternion is the basis for determining the torquing angles for platform realignment. The eigenaxis direction is obtained by extracting a unit vector\* from the vector part of  $\Delta Q$ . The eigen rotation angle can theoretically be obtained either from the scalar part of  $\Delta Q$  or the magnitude of the vector part of  $\Delta Q$ . The methods of extracting the eigenaxis rotation angle are described in more detail in the next section. At the time this study was performed, the eigen rotation angle was extracted by use of the scalar of the error quaternion.

### 3.3 DESCRIPTION OF EIGEN ROTATION ANGLE EXTRACTION METHODS

There exist two methods for extracting the eigenaxis rotation angle: (1) Arc cosine method (using the error quaternion scalar) and (2) Arc sine method (using the error quaternion vector). The equations for both methods are now presented.

#### 3.3.1 Arc Cosine Method

The first method initially used by the IMU alignment software design is the arc cosine method. This method uses the scalar part of the error quaternion  $\Delta QS$  from equation (15), that is

$$\cos \omega/2 = \frac{\Delta}{\Delta QS} \quad (16)$$

\* Although this paper does not discuss problems associated with this extraction process, it should be mentioned that for values of misalignment eigen rotation angles less than 10 arc seconds, the unit vector associated with the eigenaxis direction can be in error as much as 100 degrees. However, for such small eigenaxis rotation angles the direction of rotation is irrelevant.

After taking arc cosine of both sides of this equation and multiplying by two, the eigenaxis rotation angle is given by

$$\omega = 2 \cos^{-1} (\Delta QS) \quad (17)$$

### 3.3.2 Arc Sine Method

The second method for extracting the eigenaxis rotation angle  $\omega$  is the arc sine method. This method uses the magnitude of the vector part of the quaternion  $\Delta QV$  from equation (15), that is

$$\sin \omega/2 = |\Delta QV| \quad (18)$$

After taking arc sine of both sides of the equation and multiplying by two, the eigenaxis rotation angle is given by

$$\omega = 2 \sin^{-1} |\Delta QV| \quad (19)$$

### 3.4 DESCRIPTION OF TEST SETUP AND RUNS

For simulation purposes, the exact LOS unit vectors are arbitrarily defined as follows:  $\bar{S}_e$  is defined by an azimuth angle of 60 degrees and an elevation angle of 30 degrees,  $\bar{T}_e$  is defined by an azimuth angle of -60 degrees and an elevation angle of 30 degrees. The measured LOS unit vectors are derived by rotating the exact LOS unit vectors ( $\bar{S}_e$ ,  $\bar{T}_e$ ) from the M50 frame to actual platform coordinates, that is

$$\bar{S}_m = \begin{bmatrix} DC \\ T_{M50} \end{bmatrix} \begin{bmatrix} M50 (MEASURED) \\ T_{M50 (ACTUAL)} \end{bmatrix} \bar{S}_e \quad (20)$$

$$\bar{T}_m = \begin{bmatrix} DC \\ T_{M50} \end{bmatrix} \begin{bmatrix} M50 (MEASURED) \\ T_{M50 (ACTUAL)} \end{bmatrix} \bar{T}_e \quad (21)$$

The input misalignment eigenaxis direction is defined by an azimuth angle of 45 degrees, and an elevation angle of 30 degrees. The eigenaxis rotation or input misalignment for each specific test case is given in Table 1. For each case, an input misalignment quaternion is formed from these two input quantities and then a call to QUAT\_TO\_MAT results in the input misalignment matrix. Also, for each case, the measured and exact LOS unit vectors are unitized to IBM 360/70 single and double precision since they were generated using HP9825A precision. The stored REFSMMAT is defined by a quaternion which has an eigenaxis direction and eigenaxis rotation angle. The eigenaxis direction is arbitrarily defined by an azimuth angle of 45 degrees and an elevation angle of -30 degrees. The eigenaxis rotation angle between the M50 frame and desired

cluster coordinates is taken to be 90 degrees. A call to QUAT\_TO\_MAT results in the stored transformation matrix from M50 to desired cluster coordinates. For each case, this REFSMMAT is orthogonalized in IBM 360/70 double precision before being used in the onorbit alignment flight software.

Each test run used the onorbit alignment software design as described in sections 3.1 and 3.2. For each input misalignment angle case, both single and double precision versions of the HAL/S code were exercised (Appendix A and Appendix B, respectively) and an error computed by taking the difference between the input misalignment angle and the output misalignment angle for both eigenaxis rotation angle extraction methods. This error became the standard of measure for determining the best method.

#### 4.0 RESULTS

A summary of the IMU onorbit alignment HAL/S simulation results is presented in Table 1. The output misalignment angle  $\omega_0$  and the misalignment error  $\delta\omega$  are given as functions of the input misalignment angle  $\omega_1$  for both precision versions (single, double) and both angle extraction methods (arc cosine, arc sine). The range of  $\omega_1$  is from 0 to 648000 arc seconds (180 degrees). As indicated in the last column of the table, both extraction methods result in zero misalignment software error over the entire range of input rotations when double precision is used. The single precision error column of  $\delta\omega$  indicates that the arc cosine method has a sizeable error for small misalignment angles. In order to visually show the amount of misalignment error for both methods when coded in single precision, a graph of misalignment error  $\delta\omega$  versus input misalignment  $\omega_1$  for the arc cosine and arc sine rotation angle extraction methods is presented in Figure 1. For the arc cosine method, Figure 1 shows approximately 400 arc seconds of misalignment error for input misalignment angles near 0 arc seconds. This 400 arc seconds of misalignment error would result in approximately 230 arc seconds per axis of additional misalignment error to the current IMU alignment error budget. For increasing input misalignment angles, a gradual decrease in misalignment error can be seen. For the arc sine method, Figure 1 shows that for input misalignment angles near 0 arc seconds, the misalignment error is negligible (i.e., .03 arc seconds). For large input misalignment angles (near 648000 arc seconds), the maximum misalignment error is 340 arc seconds. This amount of misalignment error would result in approximately 196 arc seconds (per axis) of additional misalignment error to the IMU alignment error budget.

## 5.0 CONCLUSIONS AND RECOMMENDATIONS

For small IMU misalignment angles (less than 600 arc seconds), single precision computations in combination with the arc cosine method introduces additional misalignment error of up to 230 arc seconds per axis. The arc sine method, on the other hand, produces results equivalent to double precision accuracy. The misalignment error  $\delta\omega$ , obtained during this study, is clearly a function of the eigenaxis rotation angle extraction method and the precision used in the HAL/S code. The presence of MAT\_TO\_QUAT did not significantly influence the amount of misalignment error.

It is therefore recommended that the arc sine method of eigen rotation angle extraction be implemented into the onorbit alignment software at the earliest possible date. This software change would eliminate a known computationally-induced bias error.

TABLE 1.0 - SUMMARY OF IMU ONORBIT ALIGNMENT HAL/S SIMULATION RESULTS

INPUT MISALIGN- MENT ANGLE, $\omega_I$ (arc seconds)	OUTPUT MISALIGNMENT ANGLE, $\omega_O$ (arc seconds)		MISALIGNMENT ERROR, $\delta\omega$ (arc seconds)	
	Single Precision	Double Precision	Single Precision	Double Precision
0	$\cos^{-1} = 402.86$	0.0	$\cos^{-1} = 402.86$	0.0
	$\sin^{-1} = 0.03$	0.0	$\sin^{-1} = 0.03$	0.0
5	$\cos^{-1} = 402.86$	5.0	$\cos^{-1} = 397.86$	0.0
	$\sin^{-1} = 5.00$	5.0	$\sin^{-1} = 0.00$	0.0
15	$\cos^{-1} = 402.86$	15.0	$\cos^{-1} = 387.86$	0.0
	$\sin^{-1} = 15.01$	15.0	$\sin^{-1} = 0.01$	0.0
25	$\cos^{-1} = 402.86$	25.0	$\cos^{-1} = 377.86$	0.0
	$\sin^{-1} = 25.02$	25.0	$\sin^{-1} = 0.02$	0.0
50	$\cos^{-1} = 402.86$	50.0	$\cos^{-1} = 352.86$	0.0
	$\sin^{-1} = 50.02$	50.0	$\sin^{-1} = 0.02$	0.0
75	$\cos^{-1} = 402.86$	75.0	$\cos^{-1} = 327.86$	0.0
	$\sin^{-1} = 74.99$	75.0	$\sin^{-1} = 0.01$	0.0
100	$\cos^{-1} = 402.86$	100.0	$\cos^{-1} = 302.86$	0.0
	$\sin^{-1} = 99.99$	100.0	$\sin^{-1} = 0.01$	0.0
125	$\cos^{-1} = 402.86$	125.0	$\cos^{-1} = 277.86$	0.0
	$\sin^{-1} = 125.01$	125.0	$\sin^{-1} = 0.01$	0.0
150	$\cos^{-1} = 402.86$	150.0	$\cos^{-1} = 252.86$	0.0
	$\sin^{-1} = 150.02$	150.0	$\sin^{-1} = 0.02$	0.0
200	$\cos^{-1} = 402.86$	200.0	$\cos^{-1} = 202.86$	0.0
	$\sin^{-1} = 200.00$	200.0	$\sin^{-1} = 0.00$	0.0
300	$\cos^{-1} = 569.73$	300.0	$\cos^{-1} = 269.73$	0.0
	$\sin^{-1} = 300.02$	300.0	$\sin^{-1} = 0.02$	0.0
600	$\cos^{-1} = 697.77$	600.0	$\cos^{-1} = 97.77$	0.0
	$\sin^{-1} = 600.01$	600.0	$\sin^{-1} = 0.01$	0.0
900	$\cos^{-1} = 986.80$	900.0	$\cos^{-1} = 86.80$	0.0
	$\sin^{-1} = 900.00$	900.0	$\sin^{-1} = 0.00$	0.0
1800	$\cos^{-1} = 1846.14$	1800.0	$\cos^{-1} = 46.14$	0.0
	$\sin^{-1} = 1800.01$	1800.0	$\sin^{-1} = 0.01$	0.0
2700	$\cos^{-1} = 2732.33$	2700.0	$\cos^{-1} = 32.33$	0.0
	$\sin^{-1} = 2700.01$	2700.0	$\sin^{-1} = 0.01$	0.0
3600	$\cos^{-1} = 3625.75$	3600.0	$\cos^{-1} = 25.75$	0.0
	$\sin^{-1} = 3600.01$	3600.0	$\sin^{-1} = 0.01$	0.0
640800	$\cos^{-1} = 640799.75$	640800.0	$\cos^{-1} = 0.25$	0.0
	$\sin^{-1} = 640790.43$	640800.0	$\sin^{-1} = 9.57$	0.0
644400	$\cos^{-1} = 644399.56$	644400.0	$\cos^{-1} = 0.44$	0.0
	$\sin^{-1} = 644376.81$	644400.0	$\sin^{-1} = 23.19$	0.0
645300	$\cos^{-1} = 645299.75$	645300.0	$\cos^{-1} = 0.25$	0.0
	$\sin^{-1} = 645267.50$	645300.0	$\sin^{-1} = 32.50$	0.0
646200	$\cos^{-1} = 646199.87$	646200.0	$\cos^{-1} = 0.13$	0.0
	$\sin^{-1} = 646153.50$	646200.0	$\sin^{-1} = 46.50$	0.0
647100	$\cos^{-1} = 647099.62$	647100.0	$\cos^{-1} = 0.38$	0.0
	$\sin^{-1} = 647073.31$	647100.0	$\sin^{-1} = 76.69$	0.0
647400	$\cos^{-1} = 647399.75$	647400.0	$\cos^{-1} = 0.25$	0.0
	$\sin^{-1} = 647246.06$	647400.0	$\sin^{-1} = 153.94$	0.0
647700	$\cos^{-1} = 647699.68$	647700.0	$\cos^{-1} = 0.32$	0.0
	$\sin^{-1} = 647527.31$	647700.0	$\sin^{-1} = 172.69$	0.0



TABLE 1.0 - SUMMARY OF IMU ONORBIT ALIGNMENT HAL/S SIMULATION RESULTS - Concluded

INPUT MISALIGN- MENT ANGLE, $\omega_I$ (arc seconds) <sup>1</sup>	OUTPUT MISALIGNMENT ANGLE, $\omega_O$ (arc seconds)		MISALIGNMENT ERROR, $\delta\omega$ (arc seconds)	
	Single Precision	Double Precision	Single Precision	Double Precision
647800	$\cos^{-1} = 647799.93$	647800.0	$\cos^{-1} = 0.07$	0.0
	$\sin^{-1} = 647549.37$	647800.0	$\sin^{-1} = 250.63$	0.0
647850	$\cos^{-1} = 647849.81$	647850.0	$\cos^{-1} = 0.19$	0.0
	$\sin^{-1} = 647549.37$	647850.0	$\sin^{-1} = 300.63$	0.0
647870	$\cos^{-1} = 647869.56$	647870.0	$\cos^{-1} = 0.44$	0.0
	$\sin^{-1} = 647549.37$	647870.0	$\sin^{-1} = 320.63$	0.0
647890	$\cos^{-1} = 647889.68$	647890.0	$\cos^{-1} = 0.32$	0.0
	$\sin^{-1} = 647549.37$	647890.0	$\sin^{-1} = 340.63$	0.0
647915	$\cos^{-1} = 647914.75$	647915.0	$\cos^{-1} = 0.25$	0.0
	$\sin^{-1} = 647715.00$	647915.0	$\sin^{-1} = 200.00$	0.0
647965	$\cos^{-1} = 647964.62$	647965.0	$\cos^{-1} = 0.38$	0.0
	$\sin^{-1} = 647715.00$	647965.0	$\sin^{-1} = 250.00$	0.0
648000	$\cos^{-1} = 647999.81$	648000.0	$\cos^{-1} = 0.19$	0.0
	$\sin^{-1} = 647753.18$	648000.0	$\sin^{-1} = 246.82$	0.0

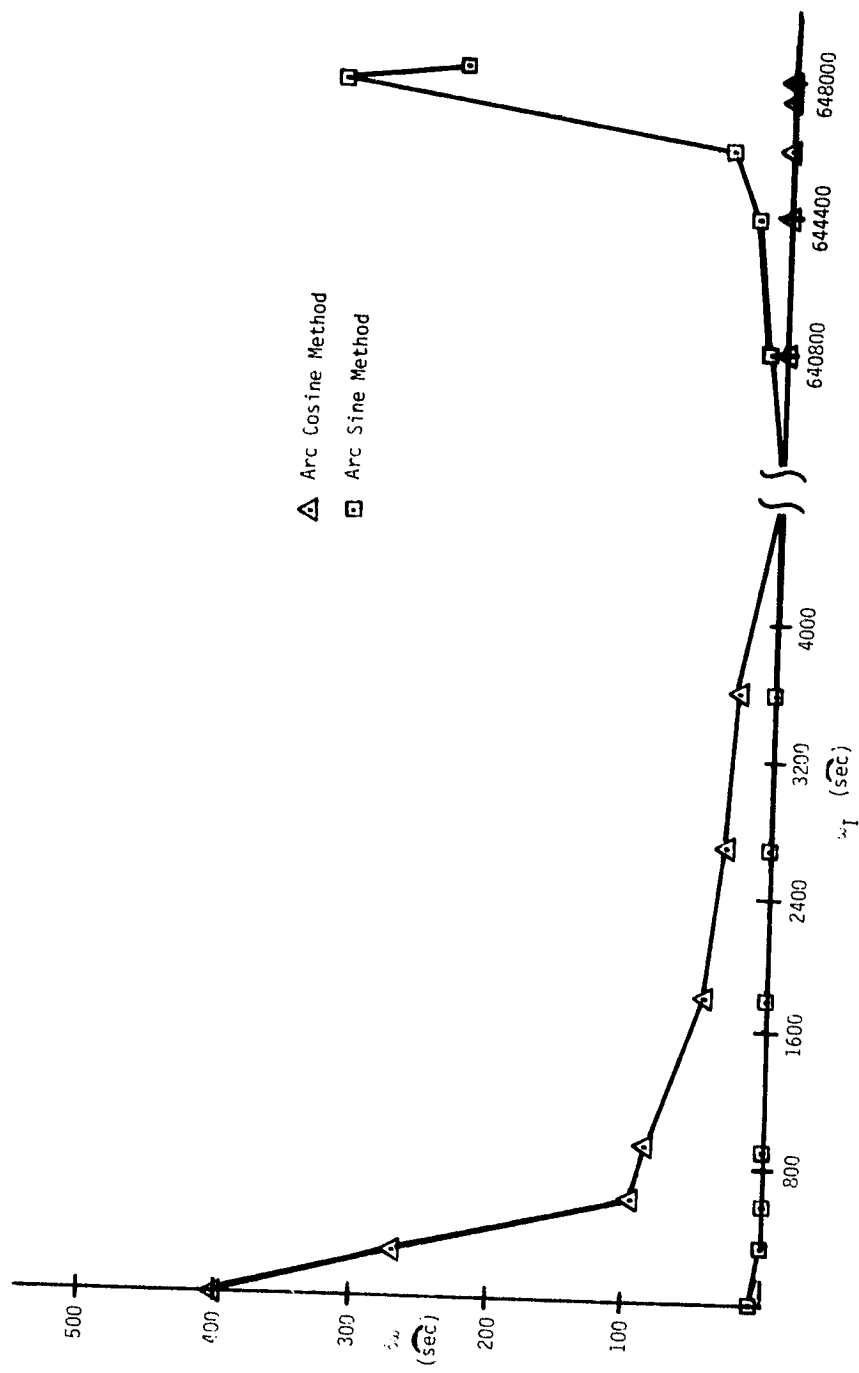


FIGURE 1.0 MISALIGNMENT ERROR VERSUS INPUT MISALIGNMENT ANGLE

APPENDIX A  
(HAL/S CODE IN SINGLE PRECISION)

STMT	MAT_2_QUAT	SOURCE	CURRENT SCOPE
1 MI	PROGRAM:		MAT_2_QUAT
2 MI	DECLARE TABLE VECTOR INITIAL(.971526492, -.233043752, -.0427620731);		MAT_2_QUAT
3 MI	DECLARE TABLE VECTOR INITIAL(-.340969046, -.792263238, -.535022797);		MAT_2_QUAT
4 MI	DECLARE STAR VECTOR INITIAL(.433012702, .7500000000, .5000000000);		MAT_2_QUAT
5 MI	DECLARE STAR VECTOR INITIAL(.433012702, -.750000000, .500000000);		MAT_2_QUAT
6 MI	DECLARE TM50DC MATRIX INITIAL(.375000000, .875000000, .306186218, -.125000000, .375000000, -.125000000, .306186218, .250000000);		MAT_2_QUAT
7 MI	DECLARE X MATRIX;		MAT_2_QUAT
8 MI	DECLARE Q8AS SCALAR;		MAT_2_QUAT
9 MI	DECLARE Q8AV VECTOR;		MAT_2_QUAT
10 MI	DECLARE Q0 SCALAR;		MAT_2_QUAT
11 MI	DECLARE Q VECTOR;		MAT_2_QUAT
12 MI	DECLARE T SCALAR;		MAT_2_QUAT
13 MI	DECLARE ANG SCALAR;		MAT_2_QUAT
14 MI	DECLARE ANGV VECTOR;		MAT_2_QUAT
15 MI	DECLARE RD SCALAR DOUBLE INITIAL(57.295779513082320876798155);		MAT_2_QUAT
16 MI	DECLARE I INTEGER;		MAT_2_QUAT
17 MI	DECLARE N INTEGER;		MAT_2_QUAT
18 MI	DECLARE J INTEGER;		MAT_2_QUAT
19 MI	DECLARE K INTEGER;		MAT_2_QUAT
20 MI	DECLARE H INTEGER;		MAT_2_QUAT
21 MI	DECLARE ANGI SCALAR;		MAT_2_QUAT
22 MI	DECLARE TSP50 MATRIX;		MAT_2_QUAT
23 MI	DECLARE TPCS MATRIX;		MAT_2_QUAT
24 MI	DECLARE TPC450 MATRIX;		MAT_2_QUAT
25 MI	DECLARE U VECTOR;		MAT_2_QUAT

360-22.13

OP I20600.APL.SRC(BDP1VER)RVL=AA

NOVEMBER 11, 1978 14:22:17.20

PAGE 3

STAT

SOURCE

CURRENT SCOPE

```

26 M| DECLARE V VECTOR;          | MAT_2_QUAT
27 M| DECLARE W VECTOR;          | MAT_2_QUAT
28 M| DECLARE X1 VECTOR;         | MAT_2_QUAT
29 M| DECLARE Y VECTOR;         | MAT_2_QUAT
30 M| DECLARE Z VECTOR;         | MAT_2_QUAT
31 M| DECLARE A VECTOR DOUBLE;   | MAT_2_QUAT
32 M| DECLARE B VECTOR DOUBLE;   | MAT_2_QUAT
33 M| DECLARE C VECTOR DOUBLE;   | MAT_2_QUAT
34 M| DECLARE D MATRIX DOUBLE;   | MAT_2_QUAT
35 M| WRITE(6) LINE(1), (COLUMN(1), 'MAT_2_QUAT BENCH PROGRAM RESULTS', SKIP(4));
36 M| STAR = UNIT(STAR);         | MAT_2_QUAT
37 M| STARI = UNIT(STARI);       | MAT_2_QUAT
38 M| TABLE = UNIT(TABLE);     | MAT_2_QUAT
39 M| TABLE1 = UNIT(TABLE1);   | MAT_2_QUAT
40 M| D = TMS00C;               | MAT_2_QUAT
41 S| A = D * 1;                | MAT_2_QUAT
42 S| B = D * 2;                | MAT_2_QUAT
43 S| C = D * 3;                | MAT_2_QUAT
44 M| C = UNIT(A * B);          | MAT_2_QUAT
45 M| A = UNIT(A);              | MAT_2_QUAT

```

STMT	SOURCE	CURRENT SCOPE
46 M	B = C * A;	MAT_2_QUAT
47 M S	D * 1 = A;	MAT_2_QUAT
48 M S	D * 2 = B;	MAT_2_QUAT
49 M S	D * 3 = C;	MAT_2_QUAT
50 M	TMS00C = D;	MAT_2_QUAT
51 M	U = STAR;	MAT_2_QUAT
52 M	V = UNIT(STAR * STAR);	MAT_2_QUAT
53 M	W = UNIT(U * V);	MAT_2_QUAT
54 M	X1 = TARF;	MAT_2_QUAT
55 M	Y = UNIT(TARF * TABLE);	MAT_2_QUAT
56 M	Z = UNIT(X1 * Y);	MAT_2_QUAT
57 M S	TSM50 * 1 = U;	MAT_2_QUAT
58 M S	TSM50 * 2 = V;	MAT_2_QUAT
59 M S	TSM50 * 3 = W;	MAT_2_QUAT
60 M S	TPCS <sub>1,*</sub> = X1;	MAT_2_QUAT
61 M S	TPCS <sub>2,**</sub> = Y;	MAT_2_QUAT

ORIGINAL PAGE IS  
OF POOR QUALITY

		CURRENT SCOPE
62	E/ S/	MAT_2_QUAT
		TPCS <sub>3,*</sub> = 7;
63	E/	MAT_2_QUAT
		TPC450 = TSM50 TPCS;
64	E/	MAT_2_QUAT
		X = TMS00C TPCM50;
65	M/	MAT_2_QUAT
		I, N = 0;
66	E/	MAT_2_QUAT
		00, T = TRACE(X);
67	M/	MAT_2_QUAT
		00 FOR K = 1 TO 3;
68	M/ S/	MAT_2_QUAT
		IF X <sub>K,K</sub> > 00 THEN
69	M/	MAT_2_QUAT
		DO;
70	M/	MAT_2_QUAT
		I = K;
71	M/ S/	MAT_2_QUAT
		00 = X <sub>K,K</sub> ;
72	M/	MAT_2_QUAT
		END;
73	M/	MAT_2_QUAT
		END;
74	M/	MAT_2_QUAT
		T = SORT(I + 2 GO - T);
75	M/	MAT_2_QUAT
		DO FOR H = 1 TO 3;
76	M/	MAT_2_QUAT
		IF H = 1 THEN
77	M/	MAT_2_QUAT
		K = 2;
78	M/	MAT_2_QUAT
		ELSE
79	M/	MAT_2_QUAT
		DO;
80	M/	MAT_2_QUAT
		IF H = 2 THEN
81	M/	MAT_2_QUAT
		K = 3;
82	M/	MAT_2_QUAT
		ELSE
83	M/	MAT_2_QUAT
		K = 1;
84	M/	MAT_2_QUAT
		END;

360-22.13

DR120600.APPL.SRC(BDRIVER)RVL=AA

NOVEMBER 11, 1978

16:22:17.20

PAGE 6

SYMT

SOURCE

CURRENT SCOPE

```

83 M| I  N = N + I;
84 M| I  J = 6 - N - K;
85 M| I  IF I (N - I) = 0 THEN
86 M| S  QO, QN = (XJ,K - XK,J) / T;
87 M| I  ELSE
88 M| S  QJ+K-I = (XJ,K + XK,J) / T;
89 M| I  END;
90 M| I  IF I = 0 THEN
91 M| I  QO = T;
92 M| I  ELSE
93 M| I  QI = T;
94 M| I  T = SIGN(QO) / 2;
95 M| I  QOAS = T QO;
96 M| I  QOAV = T Q;
97 M| I  ANG = 2 * ARCCOS(QOAS);
98 M| I  ANG = 360. * ANG / 180;
99 M| I  ANGL = 2 * ARCSIN(QOAV) / 180;
99 M| I  QOAV = 1 / 180 * ANGL;
99 M| I  WRITE(6) SKIP(14), COLUMN(1), 'INPUT MATRIX ', X, SKIP(2), COLUMN(5), 'TSM50 MATRIX', TSM50, SKIP
99 M| I  (2), COLUMN(5), 'TPCS MATRIX ', TPCS, SKIP(2), COLUMN(5), 'TPCM50 MATRIX', TPCM50, SKIP(2),
99 M| I  COLUMN(5), 'TMSGDC MATRIX', TMSGDC, SKIP(2), COLUMN(5), 'QOAS ', QOAS, SKIP(2), COLUMN(5), 'QOAV
99 M| I  ', QOAV, SKIP(2), COLUMN(5), 'EIGEN ANGLE-ARCCOSINE (ARC SEC) ', ANG, SKIP(2), COLUMN(5), 'EIGEN ANGL'

```



16J-22.13

CF 120600. APPL. SRC (DDRI VER) RVL-AA

NOVEMBER 11, 1978

14:22:17.20

PAGE 7

STMT

SOURCE

CURRENT SCOPE

```

99 M LE-ARCS IN (ARC SEC) *, 3600. ANGI, SKIP(2), COLUMN(S), *CGM_V, AND (ARC SEC) *, ANGV, SKIP(2), COLUMN
E | (S), *U VECTOR, *U, SKIP(2), COLUMN(S), *V VECTOR, *V, SKIP(2), COLUMN(S), *W VECTOR, *W, SKIP(2)
99 M | *, COLUMN(S), *X1 VECTOR, *X1, SKIP(2), COLUMN(S), *Y VECTOR, *Y, SKIP(2), COLUMN(S), *Z VECTOR,
E | *Z;
99 M |
100 M CLOSE 'MAT_2_QUAT';

```

MAT\_2\_QUAT

MAT\_2\_QUAT

MAT\_2\_QUAT

MAT\_2\_QUAT

MAT\_2\_QUAT

APPENDIX B  
(HAL/S CODE IN DOUBLE PRECISION)

```
STM 1 MI MAT_2_QUAT: CURRENT SCOPE
1 MI PROGRAM: I MAT_2_QUAT
2 MI DECLARE TABLF VECTOR DOUBLE INITIAL(.971722877, -.232155915, -.063107901): I MAT_2_QUAT
3 MI DECLARE TABLF VECTOR DOUBLE INITIAL(-.340777128, -.794655915, -.502387224): I MAT_2_QUAT
4 MI DECLARE STAR VECTOR DOUBLE INITIAL(.433012702, .750000000, .500000000): I MAT_2_QUAT
5 MI DECLARE STAR VECTOR DOUBLE INITIAL(-.433012702, -.750000000, .500000000): I MAT_2_QUAT
6 MI DECLARE TMS0DC MATRIX DOUBLE INITIAL(.375000000, .875000000, .306186218, -.125000000, .375000000): I MAT_2_QUAT
7 MI DECLARE TMS0DC MATRIX DOUBLE: I MAT_2_QUAT
8 MI DECLARE TPCS MATRIX DOUBLE: I MAT_2_QUAT
9 MI DECLARE TPCM50 MATRIX DOUBLE: I MAT_2_QUAT
10 MI DECLARE RD SCALAR DOUBLE INITIAL(57.295779513082320876798155): I MAT_2_QUAT
11 MI DECLARE U VECTOR DOUBLE: I MAT_2_QUAT
12 MI DECLARE V VECTOR DOUBLE: I MAT_2_QUAT
13 MI DECLARE W VECTOR DOUBLE: I MAT_2_QUAT
14 MI DECLARE X1 VECTOR DOUBLE: I MAT_2_QUAT
15 MI DECLARE Y VECTOR DOUBLE: I MAT_2_QUAT
16 MI DECLARE Z VECTOR DOUBLE: I MAT_2_QUAT
17 MI DECLARE X MATRIX DOUBLE: I MAT_2_QUAT
18 MI DECLARE ORAS SCALAR DOUBLE: I MAT_2_QUAT
19 MI DECLARE ORAV VECTOR DOUBLE: I MAT_2_QUAT
20 MI DECLARE ORN SCALAR DOUBLE: I MAT_2_QUAT
21 MI DECLARE O VECTOR DOUBLE: I MAT_2_QUAT
22 MI DECLARE P INTEGER: I MAT_2_QUAT
23 MI DECLARE N INTEGER: I MAT_2_QUAT
24 MI DECLARE J INTEGER: I MAT_2_QUAT
25 MI DECLARE K INTEGER: I MAT_2_QUAT
```

STMT	SOURCE	CURRENT SCOPE
26 M	DECLARE T SCALAR DOUBLE;	MAT_2_QUAT
27 M	DECLARE H INTEGER;	MAT_2_QUAT
28 M	DECLARE ANG SCALAR DOUBLE;	MAT_2_QUAT
29 M	DECLARE ANGI SCALAR DOUBLE;	MAT_2_QUAT
30 M	DECLARE ANGV VECTOR DOUBLE;	MAT_2_QUAT
31 M	DECLARE A VECTOR DOUBLE;	MAT_2_QUAT
32 M	DECLARE B VECTOR DOUBLE;	MAT_2_QUAT
33 M	DECLARE C VECTOR DOUBLE;	MAT_2_QUAT
34 M	WRITE(6) LINE(1), COLUMN(1), *MAT_2_QUAT BENCH PROGRAM RESULTS*, SKIP(4);	MAT_2_QUAT
35 M	STAR = UNIT(STAR);	MAT_2_QUAT
36 M	STAR1 = UNIT(STAR1);	MAT_2_QUAT
37 M	TABLE = UNIT(TABLE);	MAT_2_QUAT
38 M	TABLE1 = UNIT(TABLE1);	MAT_2_QUAT
39 M	A = TM50DC *.1;	MAT_2_QUAT
40 M	B = TM50DC *.2;	MAT_2_QUAT
41 M	C = TM50DC *.3;	MAT_2_QUAT
42 M	C = UNIT(A * B);	MAT_2_QUAT
43 M	A = UNIT(A);	MAT_2_QUAT
44 M	B = C * A;	MAT_2_QUAT
45 M	TM50DC *.1 = A;	MAT_2_QUAT

360-77-13

OR120700.APPL.SRCEBDRIVFRI RVL=AA

NOVEMBER 15, 1970 13:39:56.31

STM

SOURCE

PAGE 4

		CURRENT SCOPE
46	S   T450DC *0.2 = B:	MAT_2_0UAT
47	F   T450DC *0.3 = C:	MAT_2_0UAT
48	E   U = STAR:	MAT_2_0UAT
49	E   V = UNIT(STAR * START):	MAT_2_0UAT
50	E   W = UNIT(U * V):	MAT_2_0UAT
51	F   X1 = TABLE:	MAT_2_0UAT
52	F   Y = UNIT(TABLE * TABLE1):	MAT_2_0UAT
53	E   Z = UNIT(X1 * Y):	MAT_2_0UAT
54	S   TSM50 *0.1 = U:	MAT_2_0UAT
55	E   TSM50 *0.2 = V:	MAT_2_0UAT
56	S   TSM50 *0.3 = W:	MAT_2_0UAT
57	F   TPCS 1.0 = X1:	MAT_2_0UAT
58	F   TPCS 2.0 = Y:	MAT_2_0UAT
59	F   TPCS 3.0 = Z:	MAT_2_0UAT
60	F   TPCMSO = TSM50 TPCS:	MAT_2_0UAT

ORIGINAL PAGE IS OF POOR QUALITY

STMT	SOURCE	CURRENT SCOPE
E1	X = TMS0DC TP 450;	MAT_2_0UAT
61 M	I, N = 0;	MAT_2_0UAT
62 M		
E1	DO, T = TRACE(X);	MAT_2_0UAT
63 M		
64 M	DO FOR K = 1 TO 3;	MAT_2_0UAT
65 M	IF X <sub>K,K</sub> > DO THEN	MAT_2_0UAT
S1		
66 M	DO;	MAT_2_0UAT
67 M	I = K;	MAT_2_0UAT
68 M	OO = X <sub>K,K</sub> ;	MAT_2_0UAT
S1		
69 M	END;	MAT_2_0UAT
70 M	END;	MAT_2_0UAT
71 M	T = SORT(L + 2 OO - T);	MAT_2_0UAT
72 M	DO FOR H = 1 TO 3;	MAT_2_0UAT
73 M	IF H = 1 THEN	MAT_2_0UAT
74 M	K = 2;	MAT_2_0UAT
75 M	ELSE	MAT_2_0UAT
76 M	DO;	MAT_2_0UAT
77 M	IF H = 2 THEN	MAT_2_0UAT
78 M	K = 3;	MAT_2_0UAT
79 M	ELSE	MAT_2_0UAT
80 M	K = 1;	MAT_2_0UAT
81 M	END;	MAT_2_0UAT
82 M	N = N + 1;	MAT_2_0UAT
83 M	J = 6 - N - K;	MAT_2_0UAT
S1	IF I (N - 1) = 0 THEN	MAT_2_0UAT
	OO, O <sub>N</sub> = (X <sub>J,K</sub> - X <sub>K,J</sub> ) / T;	MAT_2_0UAT

STMT	SOURCE	CURRENT SCOPE
84 M 1	ELSE	MAT_2_OLAT
84 M 2	0      JAK-I = (X(J,K) * X(K,J)) / T;	MAT_2_OUAT
85 M 1	END;	MAT_2_OUAT
85 M 2	IF I = 0 THEN	MAT_2_OUAT
87 M 1	OO = T;	MAT_2_OLAT
88 M 1	ELSE	MAT_2_OUAT
88 M 2	Q = T;	MAT_2_OUAT
89 M 1	T = SIGN(OO) / 2;	MAT_2_OUAT
90 M 1	ORAS = T OO;	MAT_2_OLAT
91 M 1	ORAV = T O;	MAT_2_OUAT
92 M 1	ANG = 2 ARCCOS(OBAS);	MAT_2_OLAT
93 M 1	ANG = 3600. ANG RD;	MAT_2_OUAT
94 M 1	ANG1 = 2 ARCSIN(ABVAL(ORAV)) RD;	MAT_2_OLAT
95 M 1	ANGV = ANG UMIT(ORAV);	MAT_2_OUAT
95 M 2	WRITE(6) SKIP(4), COLUMN(1), 'INPUT MATRIX ', X, SKIP(2), COLUMN(5), 'TSMO MATRIX', TSMO, SKIP	MAT_2_OUAT
96 M 1	(2), COLUMN(5), 'TPCS, SKIP(2), COLUMN(5), 'TPCMO MATRIX', TPCM5C, SKIP(2),	MAT_2_OUAT
96 M 2	COLUMN(5), 'TMSODC MATRIX', TMSODC, SKIP(2), COLUMN(5), 'OBAS ', OBAS, SKIP(2), COLUMN(5), 'OBAV	MAT_2_OUAT
97 M 1	' , ORAV, SKIP(2), COLUMN(5), 'EIGEN ANGLE-ARCCOSINE-TARC SEC' ', ANG, SKIP(2), COLUMN(5), 'EIGEN ANG	MAT_2_OUAT
98 M 1	' , LE-ARCSIN (ARC SEC) ', 3600. ANGL, SKIP(2), COLUMN(5), 'CGNV, ANG (ARC SEC) ', ANGV, SKIP(2), COLUMN	MAT_2_OUAT
99 M 1	(5), 'U VECTOR', U, SKIP(2), COLUMN(5), 'V VECTOR', V, SKIP(2), COLUMN(5), 'W VECTOR', W, SKIP(2)	MAT_2_OUAT
99 M 2	' , COLUMN(5), 'X1 VECTOR', X1, SKIP(2), COLUMN(5), 'Y VECTOR', Y, SKIP(2), COLUMN(5), 'Z VECTOR'	MAT_2_OUAT

360-72-113  
STMT  
DIR120700,APPL,SRCLDRIVER1RVL=AA  
NOVEMBER 15, 1978 13:39:6.31  
SOURCE  
CURRENT SCOPE  
MAT\_2\_0UAT  
MAT\_2\_0UAT

ORIGINAL PAGE IS  
OF POOR QUALITY