IMU SELF-ALIGNMENT TECHNIQUES
J. C. Hung, et al
Battelle Columbus Laboratories

Prepared for:
Army Missile Research, Development and Engineering Laboratory

30 September 1974
DISPOSITION INSTRUCTIONS

DESTROY THIS REPORT WHEN IT IS NO LONGER NEEDED. DO NOT RETURN IT TO THE ORIGINATOR.

DISCLAIMER

THE FINDINGS IN THIS REPORT ARE NOT TO BE CONSTRUED AS AN OFFICIAL DEPARTMENT OF THE ARMY POSITION UNLESS SO DESIGNATED BY OTHER AUTHORIZED DOCUMENTS.

TRADE NAMES

USE OF TRADE NAMES OR MANUFACTURERS IN THIS REPORT DOES NOT CONSTITUTE AN OFFICIAL ENDORSEMENT OR APPROVAL OF THE USE OF SUCH COMMERCIAL HARDWARE OR SOFTWARE.
This report presents the results of a study on platform self-alignment performed at the Guidance and Control Directorate, US Army Missile Research, Development and Engineering Laboratory, Redstone Arsenal, Alabama.

The study was initiated to explore the latest techniques in platform self-alignment and to develop new and novel approaches which would enhance the successful application of self-alignment principles to the PERSHING IX platform alignment problem with its unique accuracy and reaction time constraints.
A significant result of this study is the development of a new gyrocompassing algorithm which provides improved self-alignment performance.
# CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>INTRODUCTION</td>
<td>3</td>
</tr>
<tr>
<td>II</td>
<td>IMU SELF-ALIGNMENT</td>
<td>4</td>
</tr>
<tr>
<td>III</td>
<td>PLATFORM DRIFTS</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>1. Kinematics of Platform Drift</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>2. Time Functions of Drifts</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>3. Time Functions of Misalignments</td>
<td>12</td>
</tr>
<tr>
<td>IV</td>
<td>GYROCOMPASSING EQUATION</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>1. The Equation</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>2. Gyrocompassing Accuracy</td>
<td>12</td>
</tr>
<tr>
<td>V</td>
<td>CONCEPTS FOR SELF-ALIGNMENT</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>1. Gyro Drift Determination</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>2. Two-Position Gyrocompassing</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>3. Off-Set Self-Alignment</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>4. Large Angle Self-Alignment</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>5. A Typical Self-Alignment Procedure</td>
<td>20</td>
</tr>
<tr>
<td>VI</td>
<td>STATE ESTIMATION FOR SELF-ALIGNMENT</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>1. Platform Alignment State Vector</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>2. Zero-Torquing Measurement Equation</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>3. Estimation Techniques</td>
<td>26</td>
</tr>
<tr>
<td>VII</td>
<td>A NEW LEAST SQUARE ALGORITHM</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>1. Measurement Equations</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>2. The Usual Least Square Algorithm</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>3. A New Least Square Algorithm</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>4. Sensitivity Consideration</td>
<td>34</td>
</tr>
<tr>
<td>VIII</td>
<td>THEORETICAL ERROR ANALYSES</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>1. Analysis for New Algorithm</td>
<td>37</td>
</tr>
<tr>
<td></td>
<td>2. Analysis for Usual Algorithm</td>
<td>40</td>
</tr>
<tr>
<td></td>
<td>3. Comparison</td>
<td>41</td>
</tr>
<tr>
<td>IX</td>
<td>RECOMMENDED FURTHER STUDY</td>
<td>44</td>
</tr>
<tr>
<td>X</td>
<td>CONCLUSIONS</td>
<td>45</td>
</tr>
<tr>
<td>Appendix A.</td>
<td>COMPUTER PROGRAMS FOR EXAMPLE 1</td>
<td>47</td>
</tr>
<tr>
<td>Appendix B.</td>
<td>COMPUTER PROGRAMS FOR EXAMPLES 2 AND 3</td>
<td>49</td>
</tr>
<tr>
<td>Appendix C.</td>
<td>COMPUTER PROGRAMS FOR EXAMPLE 4</td>
<td>51</td>
</tr>
<tr>
<td>Appendix D.</td>
<td>COMPUTER PROGRAMS FOR EXAMPLE 5</td>
<td>59</td>
</tr>
</tbody>
</table>
Section 1. INTRODUCTION

This report documents the results of a study on inertial measurement unit (IMU) self-alignment techniques for PERSHING II. The primary concern is the fine alignment of a fixed base inertial platform where the base is subjected to ground vibration and wind buffeting.

A thorough discussion will be made on the relationships among drifts and misalignments of a platform. A gyrocompassing equation will be derived. Several concepts useful for forming self-alignment procedures will be discussed with the help of developed analytics.

A new least square regression algorithm, specially for IMU alignment, will be developed. The superiority of this algorithm will be demonstrated through theoretical analysis, experimental results, and hypothetical examples.

The scope of this study can best be seen from the Table of Contents.
Section II. IMU SELF-ALIGNMENT

An IMU can be aligned to an earth fixed coordinate system at most latitudes on the earth's surface by using the information derived from the output of the unit's sensors. This type alignment of an IMU is called leveling and gyrocompassing. Two fundamentally different approaches are used to accomplish this: (1) the gimbaled platform of the IMU is physically driven to align with the earth coordinates, and (2) the alignment is achieved analytically by determining the misalignments of platform axes with respect to the earth coordinates. The second approach has the advantage of faster gyrocompassing, but at the expense of a high speed digital computer. Our present study is centered on the second approach, namely, the "analytic gyrocompassing".

The earth fixed coordinate system adopted in this study is shown in Figure 1, where the three orthogonal axes are N, E, and A representing north, east, and azimuth, respectively. As a result of this choice of coordinates, the azimuth component of the earth rate is a negative quantity as shown in Figure 2.

Figure 1. Earth fixed coordinates.
Figure 2. Earth rate components.

\[ \Omega = \hat{\Omega}_A + \Omega_N \]
\[ \Omega_A = -\Omega \sin L \]
\[ \Omega_N = \Omega \cos L \]

Figure 3 depicts the involvement of the analytic gyrocompassing. A large amount of data, obtained from the outputs of platform sensors, is reduced to a set of parameters through a chosen data reduction technique. Then, the platform's azimuth is determined from these parameters using a gyrocompassing equation.

It is obvious that better hardware allows more accurate gyrocompassing. For a given hardware system, different gyrocompassing procedures can be formed by different combinations of basic concepts. Thus the accuracy of gyrocompassing depends on several factors:

1) Platform hardware
2) Gyrocompassing procedure
3) Gyrocompassing equation
4) Data reduction algorithm
5) Computer dependent errors.

Figure 3. Analytic gyrocompassing.

Figure 4 shows a general model of a platform with error sources indicated. Each solid line represents a hardware connection while each dashed line represents the path of a sensed signal.

The "electronic and network" block can be implemented for various purposes such as maintaining the platform at a specific orientation, compensating for errors, and improving the platform dynamics. Our present purpose is to align the platform coordinates to the earth fixed coordinates.
Figure 4. Platform alignment model.
Section III. PLATFORM DRIFTS [1]

The analytic model needed for data reduction and the gyrocompassing equation needed for azimuth determination are derived from the drift characteristics of the platform. Therefore understanding the drift characteristics is a prerequisite to the development of gyrocompassing techniques.

For a gimbaled platform, the platform axes are slaved to the gyros. The time constants of platform servos are usually much smaller than the gyrocompassing time (on the order of milliseconds versus minutes). Hence the gyro drift contributes instantaneously to the platform drift of the same amount. Thus the terms "platform drift" and "gyro torquing rate" become synonymous.

1. Kinematics of Platform Drift

Consider a platform which has been coarsely aligned to the earth fixed coordinates. The deterministic torquing rate for each gyro consists of the self-axis earth rate component, the cross-axis earth rate component, and the gyro drift.

Let \( \theta_N \) = misalignment about north axis
\( \theta_E \) = misalignment about east axis
\( \theta_A \) = misalignment about azimuth axis
\( D_N \) = north gyro drift
\( D_E \) = east gyro drift
\( D_A \) = azimuth gyro drift rate
\( \Omega \) = earth rate
\( L \) = latitude of launchsite
\( \Omega_N = \Omega \cos L \) = north component of earth rate
\( \Omega_A = -\Omega \sin L \) = azimuth component of earth rate
\( K_N \) = north gyro torquer scale factor error
\( K_A \) = azimuth gyro torquer scale factor error.
The torquing rate for each gyro is obtained as follows:

for the north gyro,
\[ \dot{\theta}_N = \Omega_N + D_N - \Omega_A \sin \theta_E - \Omega_N (1 - \cos \theta_A \cos \theta_E) + K_N \Omega_N \quad (1) \]
where \( \Omega_A \sin \theta_E = \text{cross-axis earth rate due to misalignment} \)
\( \Omega_N (1 - \cos \theta_A \cos \theta_E) = \text{change of self-axis earth rate due to misalignment} \)
\( K_N \Omega_N = \text{rate due to torquer scale factor error;} \)

for the east gyro,
\[ \dot{\theta}_E = D_E + \Omega_A \sin \theta_N - \Omega_N \sin \theta_A \quad (2) \]
where \( \Omega_A \sin \theta_N - \Omega_N \sin \theta_A = \text{cross-axis earth rates due to misalignment;} \)

for the azimuth gyro,
\[ \dot{\theta}_A = \Omega_A + D_A + \Omega_N \sin \theta_E - \Omega_A (1 - \cos \theta_N \cos \theta_E) + K_A \Omega_A \quad (3) \]
where \( \Omega_N \sin \theta_E = \text{cross-axis earth rate due to misalignment} \)
\( \Omega_A (1 - \cos \theta_N \cos \theta_E) = \text{change of self-axis earth rate due to misalignment} \)
\( K_A \Omega_A = \text{rate due to torquer scale factor error.} \)

From Equations (1), (2), and (3) the non-nominal parts of the torquing rates for north, east, and azimuth gyros (or, equivalently, the drifts for north, east, and azimuth axes of the platform) are, respectively,
\[ D'_N = D_N - \Omega_A \sin \theta_E - \Omega_N (1 - \cos \theta_A \cos \theta_E) + K_N \Omega_N \quad (4) \]
\[ D'_E = D_E + \Omega_A \sin \theta_N - \Omega_N \sin \theta_A \quad (5) \]
\[ D'_A = D_A + \Omega_N \sin \theta_E - \Omega_A (1 - \cos \theta_N \cos \theta_E) + K_A \Omega_A \quad (6) \]

If the misalignment \( \theta_N, \theta_E, \) and \( \theta_A \) are sufficiently small, small angle approximations for sine and cosine functions can be used. Under
this condition, Equations (4), (5), and (6) reduce to

\[ D_N' = D_N - \Omega_A \theta_E + K_N \Omega_N \] (7)

\[ D_E' = D_E - \Omega_N \theta_A + \Omega_A \theta_N \] (8)

\[ D_A' = D_A + \Omega_N \theta_E + K_N \Omega_A \] (9)

Monitoring values of calibrated gyro torquing currents provide a way of determining the platform drifts.

2. Time Functions of Drifts

Equations (7), (8), and (9) show that the drifts along three platform axes are coupled together by the misalignments \( \theta_N \), \( \theta_E \), and \( \theta_A \). Understanding this coupling effect is important to accurate gyrocompassing.

It is reasonable to assume that during the period of gyrocompassing, \( \theta_A \), the azimuth misalignment is constant. With this in mind, Equations (7) and (8) can be further developed into

\[ D_N' = D_N - \Omega_A \theta_E + K_N \Omega_N \\
= D_N - \Omega_A \left( \theta_{E0} + \int_0^t D_E'(\tau) d\tau \right) + K_N \Omega_N \]

\[ = D_N - \Omega_A \theta_{E0} + K_N \Omega_N - \Omega_A \left( \int_0^t D_E'(\tau) d\tau \right) \] (10)

and

\[ D_E' = D_E - \Omega_N \theta_A + \Omega_A \theta_N \\
= D_E - \Omega_N \theta_A + \Omega_A \left( \theta_{NO} + \int_0^t D_N'(\tau) d\tau \right) + \Omega_A \left( \int_0^t D_N'(\tau) d\tau \right) \]

\[ = D_E - \Omega_N \theta_A + \Omega_A \theta_{NO} + \Omega_A \left( \int_0^t D_N'(\tau) d\tau \right) \] (11)
By defining

$$D'_{EO} = D_E - \Omega_N \theta_A + \Omega_A \theta_N = D_E - \Omega_N \theta_A$$

$$D'_{NO} = D_N - \Omega_A \theta_{EO} + K_N \Omega_N$$

which represent the initial values of the drift for east and north axes, the two drift equations become

$$D'_N + \Omega_A \int_0^t D'_E(\tau) d\tau = D'_{NO}$$

$$D'_E - \Omega_A \int_0^t D'_N(\tau) d\tau = D'_{EO}$$

The time functions $D'_N(t)$ and $D'_E(t)$ can be solved from Equations (13) and (14) using transform method. Taking the Laplace transform of both equations,

$$D'_N(s) + \frac{\Omega_A}{s} D'_E(s) = \frac{D'_{NO}}{s}$$

$$D'_E(s) - \frac{\Omega_A}{s} D'_N(s) = \frac{D'_{EO}}{s}$$

Solving for $D'_N(s)$ and $D'_E(s)$ yields

$$D'_N(s) = D'_{NO} \frac{s}{s^2 + \Omega_A^2} - D'_{EO} \frac{\Omega_A}{s^2 + \Omega_A^2}$$

$$D'_E(s) = D'_{EO} \frac{s}{s^2 + \Omega_A^2} + D'_{NO} \frac{\Omega_A}{s^2 + \Omega_A^2}$$

By taking the inverse Laplace transform of Equations (17) and (18), the corresponding time functions are, respectively,

$$D'_N(t) = D'_{NO} \cos \Omega_A t - D'_{EO} \sin \Omega_A t$$

$$D'_E(t) = D'_{EO} \cos \Omega_A t + D'_{NO} \sin \Omega_A t$$
In matrix form,
\[
\begin{bmatrix}
D^t_N(t) \\
D^t_E(t)
\end{bmatrix} =
\begin{bmatrix}
\cos \Omega_A t - \sin \Omega_A t \\
\sin \Omega_A t \cos \Omega_A t
\end{bmatrix}
\begin{bmatrix}
D^t_{NO} \\
D^t_{EO}
\end{bmatrix}
\]

which shows that the vector drift at any time is a rotation of angle $\Omega_A t$
from its initial vector drift.

For small value of $\Omega_A t$,
\[
\begin{aligned}
\cos \Omega_A t &= 1 - \frac{\Omega_A^2 t^2}{2} \\
\sin \Omega_A t &= \Omega_A t
\end{aligned}
\]

Then Equations (17) and (18) can be approximated by
\[
\begin{align*}
D^t_N(t) &= D^t_{NO} - \Omega_A^2 t^2 - \frac{D^t_{NO} \Omega_A^2 t}{2} \\
D^t_E(t) &= D^t_{EO} + \Omega_A^2 t^2 - \frac{D^t_{EO} \Omega_A^2 t}{2}
\end{align*}
\]

If the second order terms are negligible, Equations (23) and (24) can
further be approximated by
\[
\begin{align*}
D^t_N(t) &= D^t_{NO} - \Omega_A \ t \\
D^t_E(t) &= D^t_{EO} + \Omega_A \ t
\end{align*}
\]

3. Time Functions of Misalignments

With the help of Equations (19) and (20), the misalignments
$\theta_N(t)$ and $\theta_E(t)$ can be determined as follows:
\[ \theta_N(t) = \theta_{N0} + \int_0^t D'_N(\tau) d\tau \]

\[ = \theta_{N0} + \int_0^t (D'_{N0} \cos \Omega_A \tau - D'_{N0} \sin \Omega_A \tau) d\tau \]

\[ = \theta_{N0} + \frac{D'_{N0}}{\Omega_A} \sin \Omega_A t + \frac{D'_{N0}}{\Omega_A} \cos \Omega_A t \]

\[ - \frac{D'_{N0}}{\Omega_A} \]  

(27)

\[ \theta_E(t) = \theta_{E0} + \int_0^t D'_E(\tau) d\tau \]

\[ = \theta_{E0} + \int_0^t (D'_{E0} \cos \Omega_A \tau + D'_{N0} \sin \Omega_A \tau) d\tau \]

\[ = \theta_{E0} + \frac{D'_{E0}}{\Omega_A} \sin \Omega_A t - \frac{D'_{N0}}{\Omega_A} \cos \Omega_A t \]

\[ + \frac{D'_{N0}}{\Omega_A} \]  

(28)
Section IV. GYROCOMPASSING EQUATION

Gyrocompassing, the determination of azimuth misalignment, requires the knowledge of drifts and misalignments along the north and east platform axes. A gyrocompassing equation and its accuracy are discussed here.

1. The Equation

By solving Equation (8) for \( \theta_A(t) \), the azimuth misalignment at any time \( t \) is obtained as

\[
\theta_A(t) = \frac{D_E - D'_E(t) + \Omega_A \theta_N(t)}{\Omega_N}. \tag{29}
\]

Substituting the details of \( D'_E(t) \) and \( \theta_N(t) \) from Equations (18) and (25) into Equation (29), all sine and cosine terms cancel. The result is the "gyrocompassing equation" sought,

\[
\theta_A(t) = \frac{D_E - D'_E(t) + \Omega_A \theta_N(t)}{\Omega_N}. \tag{30}
\]

Intuitively, it can also be said that Equation (30) comes directly from Equation (29) since

\[
\theta_A(t) = \theta_{A0} = \frac{D_E - D'_E(t) + \Omega_A \theta_N(t)}{\Omega_N}. \tag{31}
\]

2. Gyrocompassing Accuracy

The ultimate gyrocompassing accuracy is limited by the following uncertainties:

- \( \Delta D_E \) = East gyro drift uncertainty
- \( \Delta D'_E \) = Uncertainty in platform drift about its east axis
- \( \Delta \theta_{NO} \) = Uncertainty in the initial platform misalignment about its north axis.

From Equation (30) the uncertainty in gyrocompassing is obtained as

\[
\Delta \theta_A = \frac{\Delta D_E - \Delta D'_E + \Omega_A \Delta \theta_{NO}}{\Omega_N}. \tag{31}
\]
Since the sign of \( \varepsilon \), individual uncertainty is not known, an upper bound of the gyrocompassing error is given by

\[
|\Delta \theta_A| \leq \frac{|\Delta \theta' - \Delta \theta_E| + |\Omega_A \Delta \theta_{N0}|}{\Omega_N}.
\] (32)

In Equation (32), \( \Delta \theta_A \) and \( \Delta \theta_{N0} \) are in radians while \( \Delta \theta_E \), \( \Delta \theta'_E \), \( \Omega_A \), and \( \Omega_N \) have the same unit. If \( \Delta \theta_A \) and \( \Delta \theta_{N0} \) are in arcseconds, Equation (32) should be replaced by

\[
|\Delta \theta_A| \leq 206,280 \frac{|\Delta \theta'_E - \Delta \theta_E|}{\Omega_N} + |\Delta \theta_{N0} \tan L|
\] (33)

where \( L \) is the launchsite latitude.

Consider an example where

\[
L = 45 \text{ degrees}
\]

\[
\Delta \theta'_E - \Delta \theta_E = 0.003 \text{ deg/hr}
\]

\[
\Delta \theta_{N0} = 2 \text{ arcsec}
\]

Since \( \Omega = 15 \text{ degrees/hour} \),

\[
\Omega_N = \Omega \cos L = 10.61
\]

\[
\tan L = 1
\]

Equation (33) gives

\[
|\Delta \theta_A| \leq 206,280 \times \frac{0.003}{10.61} + 2 = 58.3 + 2 = 60.3 \text{ arcsec}
\]

Notice that, in this example, the platform east axis drift uncertainty contributes most of the error in gyrocompassing.
Section V. CONCEPTS FOR SELF-ALIGNMENT

This section presents a discussion of several useful concepts which can be chosen to form different IMU self-alignment procedures.

1. Gyro Drift Determination

Determination of gyro drifts using the information within the system is also called "autobiasing." Here we shall be concerned with drifts about north and east axes. There are two different methods of autobiasing gyros, namely, the "closed-loop method" and the "open-loop method." The choice between the two depends on the relative uncertainty between the gyro torquer scale factor error and the accelerometer scale factor error.

Referring to Figure 4, the platform alignment system consists of two Schuler loops. For the closed-loop method, both Schuler loops are closed and gyros are torqued at the rates given by Equations (1), (2), and (3). Under the fine alignment condition, platform is sufficiently level such that small angle approximations for trigonometric functions are satisfactory. Therefore Equations (7) and (8) give the non-nominal torquing rate for the north and east gyros. In general, there are biases in accelerometers, so the terms $-\omega_{AE}$ and $\omega_{AN}$ may not be small. However, in all practical cases, the biases are known. Therefore their effect on platform drift is known. Thus it can be said that the difference between $-\omega_{AE}$ and the corresponding accelerometer bias effect is small, and between $\omega_{AN}$ and its corresponding accelerometer bias effect is also small. Under this condition, Equations (7) and (8) are further reduced to

$$D_{N}' = D_{N} + K_{N} \omega_{N}$$  \hspace{1cm} (34)

$$D_{E}' = D_{E} - \omega_{AN}$$  \hspace{1cm} (35)

The quantities $D_{N}'$ and $D_{E}'$ are obtained by measuring the torquing currents of north and east gyros. Assuming that $\omega_{N}$ is known, the north gyro drift $D_{N}$ can be accurately determined if $K_{N}$, the torquer scale factor error, is known. However, the uncertainty in $\omega_{AN}$ is, in general, so large that there is no way to accurately determine the east gyro drift $D_{E}$ from $D_{E}'$.

Often the knowledge of $K_{N}$ is not available to the degree of precision desired. Under this condition, accurate and rapid determination of $D_{N}$ from the closed-loop information is difficult.
The effect of uncertainty in north gyro torquer scale factor error can be eliminated entirely by not torquing the north axis of the platform physically. Instead, an analytically torqued north axis is maintained in the computer by on-line computation. This method is called "zero-torquing measurement." Zero-torquing is accomplished by opening Schuler loops at places indicated in Figure 4. Therefore the method is an "open-loop method." Under this condition, platform level is not maintained, so accelerometers receive larger inputs. Thus the uncertainty in the accelerometer's scale factor error becomes more important.

In the open-loop method, measurements are taken at outputs of both accelerometers. From the measurements, $D_N^1$ and $D_E^1$ are determined. Because of zero-torquing, $K_N$ plays no part in drift determination, so Equation (34) becomes

$$D_N^1 \approx D_N,$$  \hspace{1cm} (36)

which is an attractive way to determine $D_N$. However, $D_E^1$ is still given by Equation (35) where separation of $D_E$ from $-\alpha N A$ is difficult.

To conclude, it is seen that whether the closed-loop method or open-loop method is used to determine gyro drifts, only the north gyro drift can be accurately determined. Reference 1 contains several numerical examples to illustrate this phenomenon. The technique of determining $D_N^1$ and $D_E^1$ from the measurements for the open-loop method will be discussed in detail in Sections VI and VII.

2. Two-Position Gyrocompassing

Recall from Equation (30) that an accurate gyrocompassing can be achieved only if an accurate determination of the east gyro drift $D_E$ can be obtained. Since accurate drift determination can be made only for north gyro, a two-position scheme can be devised to take this advantage. The scheme may consist of the following steps:

a) Auto-biasing the north gyro to determine $D_N$

b) Slewing the platform approximately 90 degrees so that the north gyro becomes an east gyro, and the original east gyro becomes a south gyro

c) Gyrocompassing is performed with the present east gyro whose drift has been accurately determined, enabling an accurate azimuth alignment.
After slewing, the south gyro is in a favorable position for accurate drift determination. The autobiasd south gyro will then be ready for in-flight navigation.

3. Off-Set Self-Alignment

It is possible to achieve platform self-alignment with the platform coarsely leveled but without requiring that the platform axes be physically coarsely aligned to north and east. Instead, the north and east, which are obtained from an analytic coarse alignment, are analytically maintained in the computer. The fine alignment is then achieved by determining the misalignments between the computer north and east and the true north and east.

Referring to Figure 5, $\alpha$ denotes the off-set of the platform level coordinates from the computer's level coordinates, and $\theta_A$ is the azimuth misalignment to be determined. Under the off-set condition, the following quantities are first obtained:

$$V_{NP} \text{ and } V_{EP}$$ velocities along the platform’s north and east axes, obtained by integrating the outputs of north and east accelerometers

$$\alpha$$ the off-set angle, obtained by a certain coarse alignment procedure, say, BATH.

Next, the velocities along the computer north and east axes are computed from

$$\begin{align*}
V_{NC} &= V_{NP} \cos \alpha - V_{EP} \sin \alpha \\
V_{EC} &= V_{EP} \cos \alpha + V_{NP} \sin \alpha
\end{align*}$$

Finally, a fine alignment technique can be chosen for performing the fine alignment between the computer axes and the earth coordinates.

During the fine alignment, platform can either be torqued at earth rate, or be torqued about the azimuth axis at the azimuth component of the earth rate. The merit of not torquing the level axes is to avoid the torquer scale factor uncertainties, which have been discussed. The information of the torqued coordinates is already in the computer since this information is needed to torque the platform in the first place.
4. Large Angle Self-Alignment

In the case of "large angle self-alignment," the platform is coarsely leveled so the small angle approximations for $\theta_N$ and $\theta_E$ are valid. The azimuth axis is torqued at $\Omega_A$, the azimuth component of earth rate. But the azimuth misalignment $\theta_A$ is not small enough to allow the use of small angle approximation.

Under this condition, a good approximation for platform drift can be obtained from Equations (1) and (2) as

$$D'_N = D_N - \Omega_A \theta_A + \Omega_N \cos \theta_A$$  \hspace{1cm} (38)
Equations (38) and (39) can be solved for \( \cos \theta_A \) and \( \sin \theta_A \) respectively. Thus \( \tan \theta_A \) can also be obtained. During the self-alignment period, the variation in azimuth misalignment is small, so it is assumed that \( \theta_A = \theta_{A0} \). The resulting gyrocompassing equation is therefore given by

\[
\theta_A = \theta_{A0} \pm \tan^{-1} \frac{\Omega A \theta_{NO} + (D_E - D_{E0})}{\Omega A \theta_{E0} - (D_N - D_{NO})}.
\]  

The ambiguity of double values can be resolved by solving \( \theta_A \) from Equation (39) also, and compare its sign to that determined by Equation (40). The determination of \( \theta_{NO} \), \( \theta_{E0} \), \( D_{NO} \), and \( D_{E0} \) will be discussed in Sections VI and VII.

One may wonder why \( \theta_A \) is not determined directly from either Equation (38) or (39) by taking the inverse of cosine or sine function. The reason is that tangent function possesses steeper slopes, enabling a more accurate determination of \( \theta_A \).

Notice that the large azimuth angles for which this method is intended cannot be arbitrarily large. They must be within the limits that Equations (1) and (2) are valid.

5. A Typical Self-Alignment Procedure

By combining a few of the aforementioned concepts, a typical self-alignment procedure can be developed. As an example we may have a "two-position off-set zero-torquing self-alignment." Figure 6 shows the coordinates involved in this method. In Figure 6 \( N \) and \( E \) are earth's north and east; \( N_C, E_C \) and \( S_C \) are the north, east, and south known to the computer; and \( N_p \) and \( E_p \) are north and east of the platform. The 90-degree slewing for the second position is indicated by dashed arcs. The slewing is not required to be precise.

The required alignment steps for this example can best be described with the help of an activity flow diagram shown in Figure 7. If the time for achieving BATH is shorter than a coarse alignment slewing, the total alignment time can be reduced when the off-set self-alignment concept is employed. This is because the physical coarse alignment takes time to achieve, while the off-set technique 90-degrees slewing can be very rapid without worrying about its accuracy.
Figure 6. Coordinates for a 2-position off-set self-alignment.
PLATFORM IN ANY OFFSET POSITION

ESTIMATE OFFSET $\alpha$ BY "BATH"

- OPEN LEVELING LOOPS AT ACC' MTR OUTPUTS.
- LEVEL AXES NOT TORQUED.
- AZIMUTH AXIS TORQUED AT $\Omega_A$.
- MEASURE DELTA VELOCITY PULSES FROM ACC'MTR OUTPUTS.

COMPENSATE FOR ALL KNOWN BIAS AND DRIFTS INTRODUCED BY THE PLATFORM AND ELECTRONICS WHICH INCLUDE:
- GYRO: BIAS, SCALE FACTOR, MASS UNBALANCE.
- ACC'MTR: BIAS, SCALE FACTOR, ORTHOGONALITY SYMMETRY.

FIRST POSITION (AUTOBIAISING)

COMPUTE $\Delta V_N$ AND $\Delta V_E$ FROM THE COMPENSATED $\Delta V_N$ AND $\Delta V_E$ BY COORDINATE TRANSFORMATION EQUATION.

LEAST-SQUARE DATA REDUCTION TO GET $(D'_N)_1$.

SLEW PLATFORM ABOUT 90 deg.

LEAST-SQUARE DATA REDUCTION TO GET $(\theta_N)_2$, $(\theta_E)_2$.

- COMPUTE AZIMUTH MISALIGNMENT USING

$$\theta_A = \frac{(D'_E)_2 - (D'_N)_1 + (\theta'_N)_2 \tan L}{\Omega \cos L}.$$  

REPEAT FINE ALIGNMENT STEPS A, B, AND C.

SECOND POSITION (ALIGNMENT)

UPDATE ALIGNMENT.

- READY FOR LAUNCH.
- CLOSE LEVELING LOOPS AT THE END OF ALIGNMENT PROCEDURE.

Figure 7. Two-position off-set self-alignment.
Section VI. STATE ESTIMATION FOR SELF-ALIGNMENT

1. Platform Alignment State Vector

Platform leveling and gyrocompassing amount to the determination of misalignments $\theta_N$, $\theta_E$, and $\theta_A$ which, in turn, require the knowledge of $D'_N$ and $D'_E$. A preferred procedure is to first determine $\theta_N$, $\theta_E$, $D'_N$, and $D'_E$ from the platform's sensor outputs. The azimuth $\theta_A$ can then be determined using the gyrocompassing Equation (30). In this procedure the desired self-alignment state vector $x$ consists of four elements:

$$
\begin{bmatrix}
\theta_N \\
\theta_E \\
D'_N \\
D'_E
\end{bmatrix}
$$

(40)

2. Zero-Torquing Measurement Equation

Consider the case of zero-torquing fine alignment where the state vector is determined from the output data of accelerometers. For a sufficiently short fine alignment time the equations for misalignments, Equations (27) and (28), can be approximated by their Taylor series expansion up to the second power of $t$; i.e.,

$$
\theta_N(t) = \theta_{N0} + D'_N t - \frac{1}{2} D''_E \Omega_A t^2
$$

(41)

$$
\theta_E(t) = \theta_{E0} + D'_E t + \frac{1}{2} D''_N \Omega_A t^2
$$

(42)

Notice that, since the north gyro is not torqued, the drift term $D'_{N0}$ includes the north component of the earth rate. The drifts and misalignments transform into accelerations via tilts of north and east accelerometers, giving

$$
a_N = -g \times \sin \theta_E \approx -g \theta_E
$$

(43)

$$
a_E = g \times \sin \theta_N \approx g \theta_N
$$

(44)
where $g$ is the gravitational acceleration. Substituting Equations (41) and (42) into Equations (43) and (44) gives

$$a_N = -g\theta - gD^E_O t - \frac{\theta}{2} D^N_O \Omega_A t^2$$  \hspace{1cm} (45)$$

$$a_E = g\theta + gD^E_O t - \frac{\theta}{2} D^N_O \Omega_A t^2.$$  \hspace{1cm} (46)$$

Integrating Equations (45) and (46) from 0 to $t$ gives the changes of velocities during that period as

$$V_N(t) = -g\theta t - \frac{\theta}{2} D^E_O t^2 - \frac{\theta}{6} D^N_O \Omega_A t^3$$  \hspace{1cm} (47)$$

$$V_E(t) = g\theta t + \frac{\theta}{2} D^E_O t^2 - \frac{\theta}{6} D^N_O \Omega_A t^3.$$  \hspace{1cm} (48)$$

Let $V = [V_N, V_E]^T$ be the measurement vector, the matrix form of Equations (47) and (48) is

$$\begin{bmatrix} V_N(t) \\ V_E(t) \end{bmatrix} = \begin{bmatrix} 0 & -g \Omega_A t^3 - \frac{\theta}{2} t^2 \\ gt & 0 \frac{\theta}{2} t - \frac{\theta}{6} \Omega_A t^3 \end{bmatrix} \begin{bmatrix} \theta_{NO} \\ \theta_{EO} \end{bmatrix}.$$  \hspace{1cm} (49)$$

Equations (47) and (48), or Equation (49) are the measurement equation desired.

Grouping Equations (19), (20), (27), and (28) together, the state vector at any time is related to its initial value by

$$\begin{bmatrix} \theta_N \\ \theta_E \\ D^N_N \\ D^E_E \end{bmatrix} = \begin{bmatrix} 1 & 0 & \sin \Omega_A t & \cos \Omega_A t - 1 \\ 0 & 1 & 1 - \cos \Omega_A t & \sin \Omega_A t \\ 0 & 0 & \cos \Omega_A t & -\sin \Omega_A t \\ 0 & 0 & \sin \Omega_A t & \cos \Omega_A t \end{bmatrix} \begin{bmatrix} \theta_{NO} \\ \theta_{EO} \\ D^N_{NO} \\ D^E_{EO} \end{bmatrix}.$$  \hspace{1cm} (50)$$
Equations (49) and (50) provide a way for determining the desired state vector from the integrated measurement data.

It may be asked why the state vector is not determined using Equations (45) and (46) as measurement equations. The answer is that the integration reduces errors caused by the quantization effect of the accelerometer output.

In practice, discrete measurements are made at time \( t = \tau k \) for \( k = 1 \sim N \), and \( \tau \) is the sampling period. Substituting the discrete time into Equations (49) and (50) gives

\[
\begin{bmatrix}
  v_{N}(k) \\
  v_{E}(k)
\end{bmatrix} =
\begin{bmatrix}
  0 & -g \tau k - \frac{g}{6} \Omega_A^2 \tau^3 k^3 - \frac{g}{2} \tau_0^2 k^2 \\
  g \tau k & 0 - \frac{g}{2} \Omega_A^2 \tau^2 k^2 - \frac{g}{6} \Omega_A^2 \tau^3 k^3
\end{bmatrix}
\begin{bmatrix}
  \theta_{NO} \\
  \theta_{EO}
\end{bmatrix}
\]  
\[
\text{(51)}
\]

and

\[
\begin{bmatrix}
  \theta_{N}(k) \\
  \theta_{E}(k) \\
  D_{NO}'(k) \\
  D_{EO}'(k)
\end{bmatrix} =
\begin{bmatrix}
  1 & \sin \Omega_A \tau k & \cos \Omega_A \tau k - 1 \\
  0 & \Omega_A & \Omega_A \\
  0 & 1 - \cos \Omega_A \tau k & \sin \Omega_A \tau k \\
  0 & \cos \Omega_A \tau k & -\sin \Omega_A \tau k \\
  0 & \cos \Omega_A \tau k & \cos \Omega_A \tau k \\
  0 & \Omega_A & \Omega_A \\
  0 & 0 & \Omega_A \tau k
\end{bmatrix}
\begin{bmatrix}
  \theta_{NO} \\
  \theta_{EO} \\
  D_{NO}' \\
  D_{EO}'
\end{bmatrix}
\]  
\[
\text{(52)}
\]

If the total measurement time is sufficiently small,

\[
\begin{aligned}
\sin \Omega_A \tau k &\approx \Omega_A \tau k \\
\cos \Omega_A \tau k &\approx 1
\end{aligned}
\]  
\[
\text{(53)}
\]

then an approximation for Equation (52) is

\[
\begin{bmatrix}
  \theta_{N}(k) \\
  \theta_{E}(k) \\
  D_{NO}'(k) \\
  D_{EO}'(k)
\end{bmatrix} =
\begin{bmatrix}
  1 & \tau k & 0 \\
  0 & 0 & \tau k \\
  0 & 0 & -\Omega_A \tau k \\
  0 & \Omega_A \tau k & 1
\end{bmatrix}
\begin{bmatrix}
  \theta_{NO} \\
  \theta_{EO} \\
  D_{NO}' \\
  D_{EO}'
\end{bmatrix}
\]  
\[
\text{(54)}
\]
3. Estimation Techniques

When the environment is ideal, where disturbance and noise are not present, measurement of $V_N(t)$ and $V_E(t)$ at two different values of $t$ is sufficient for determining the state vector. In reality, the measurements are contaminated by noise due to ground vibration, wind buffeting, instrument noise, and other random disturbances. Under this condition, accurate determination of the state vector demands the use of a large number of redundant measurements in conjunction with a statistical estimation technique.

Two well known approaches of statistical estimation techniques applicable to platform self-alignment are the least square regression approach and the Kalman filtering approach. A comparison of the two approaches is given as follows:

a) Least Square Regression - This approach does not make use of statistical properties of the noise, if available. The associated data reduction process can be made either sequential or batch. If the desired number of state estimations is much less than the number of measurements, the least square regression algorithm requires less computer time and smaller computer memory as compared to Kalman filtering [4].

b) Kalman Filtering [5] - The Kalman filtering algorithm has a built-in provision for taking advantage of known second order statistics. The associated data reduction process is sequential, which generates an estimate of the state from each measurement.

We choose the least square regression approach for the platform self-alignment. The choice is based on two facts: first, knowledge of the statistics of the noise is not good enough to enjoy the merit of Kalman filtering; and secondly, the required number of state estimations is far less than the number of measurements so least square regression approach is superior in the required computer time and memory.

In Section VII a new least square regression algorithm, tailored to our platform self-alignment application, is presented in detail.
Section VII. A NEW LEAST SQUARE ALGORITHM

1. Measurement Equations

Least square regression method requires a special form for measurement equations. Rearranging Equation (51) and adding measurement noise to it, we get

\[ V_N(k) = A_1 k + A_2 k^2 + A_3 k^3 + n_N(k) \]  (55)
\[ V_E(k) = A_3 k + A_4 k^2 + A_5 k^3 + n_E(k) \]  (56)

where \( n_N(k) \) and \( n_E(k) \) are additive noise, \( k = 1 \sim N \), and

\[
\begin{align*}
A_1 &= -g^0 E_0 \tau \\
A_2 &= -\frac{g}{2} D_{E0} \tau^2 \\
A_3 &= -\frac{g}{6} D_{N0} \Omega_A \tau^3 \\
A_4 &= -\frac{g}{6} D_{E0} \Omega_A \tau^3 \\
A_5 &= -\frac{g}{6} D_{E0} \Omega_A \tau^3
\end{align*}
\]  (57)

and

\[
\begin{align*}
A_3 &= g^0 N_0 \tau \\
A_4 &= \frac{g}{2} D_{N0} \tau^2 \\
A_5 &= -\frac{g}{6} D_{E0} \Omega_A \tau^3 \\
A_6 &= -\frac{g}{6} D_{E0} \Omega_A \tau^3
\end{align*}
\]  (58)

The problem becomes the determination of \( A_1, A_2, A_3, \) and \( A_4 \) from a large set of redundant measurements \( V_N(k) \) and \( V_E(k) \). \( A_N \) and \( A_E \) are not needed because they differ from \( A_4 \) and \( A_2 \), respectively, only by a known constant multiplier.

2. The Usual Least Square Algorithm [6,7,8]

Measurement Equations (55) and (56) are in the form of three-term third-order polynomials in \( k \). A conventional set of least square regression formulas are available in textbooks in statistical mathematics for determining the coefficients. Applying conventional formulas to our problem, the coefficients are determined from
where

\[ Y_j = \sum_{k=1}^{N} k^j V_N(k) \quad j = 1 \sim 3 \]  \hspace{1cm} (61)

\[ Z_j = \sum_{k=1}^{N} k^j V_E(k) \quad j = 1 \sim 3 \]  \hspace{1cm} (62)

and

\[ C_1 = \sum_{k=1}^{N} k^i \quad i = 2 \sim 6 \]  \hspace{1cm} (64)

These formulas have been used for platform self-alignment as well as their applications and are applicable to any three-term third-order polynomial. The algorithm can be written either for batch processing or for sequential processing.

3. A New Least Square Algorithm

Examining Equations (55) through (58), it is seen that they can be expressed in the following equivalent form:

\[ V_N(k) = A_1 k + A_2 k^2 + uA_4 k^3 + n_N(k) \]  \hspace{1cm} (65)
\[ V_E(k) = A_3k + A_4k^2 - uA_2k^3 + n_E(k) \]  
(66)

where

\[ u = - \frac{\Omega A^2}{3} \]  
(67)

Notice that some of the coefficients are correlated deterministically. Can this property be employed to improve the estimation accuracy? The answer is in the Affirmative.

Let us study a more general case

\[
V_K = A k + B k^2 + uDk^3 + n_K \\
V'_K = C k + D k^2 + vBk^3 + n'_K
\]  
(68)

where \( V_K \) and \( V'_K \) are measurements made at sampling instants \( k = 1 \sim N \), and \( A, B, C, \) and \( D \) are parameters to be estimated. A cost function is chosen as follows:

\[
I = \sum_{k=1}^{N} \left( V_K - (Ak + Bk^2 + uDk^3) \right)^2 \\
+ \sum_{k=1}^{N} \left( V'_K - (Ck + Dk^2 + uBk^3) \right)^2
\]  
(69)

The cost function is to be minimized by an optimum selection of \( A, B, C, \) and \( D \). Setting

\[
\frac{\partial I}{\partial A} = \frac{\partial I}{\partial B} = \frac{\partial I}{\partial C} = \frac{\partial I}{\partial D} = 0
\]  
(70)

a set of four algebraic equations are obtained which can be combined into a single matrix equation:

\[
\begin{bmatrix}
W_1 \\
W_2 \\
W_3 \\
W_4
\end{bmatrix} = \begin{bmatrix}
A \\
B \\
C \\
D
\end{bmatrix}
\]  
(71)
where

\[
G = \begin{bmatrix}
C_2 & C_3 & 0 & uC_4 \\
C_3 & C_4 + VC_6 & VC_4 & (v + u)C_5 \\
0 & VC_4 & C_2 & C_3 \\
uC_4 & (u + v)C_5 & C_3 & C_4 + VC_6
\end{bmatrix}
\]  \quad (72)

\[
c_1 = \sum_{k=1}^{N} k^i \quad i = 2 \sim 6
\]  \quad (73)

and

\[
\begin{align*}
W_1 &= \sum_{k=1}^{N} kV_K^2 \\
W_2 &= \sum_{k=1}^{N} k^2(V_K^2 + vK^2) \\
W_3 &= \sum_{k=1}^{N} kV_K'^2 \\
W_4 &= \sum_{k=1}^{N} k^2(V_K'^2 + uK^2)
\end{align*}
\]  \quad (74)

The estimate of A, B, C, and D are obtained by inverting the G matrix, giving

\[
\begin{bmatrix}
A \\
B \\
C \\
D
\end{bmatrix} = G^{-1} \begin{bmatrix}
W_1 \\
W_2 \\
W_3 \\
W_4
\end{bmatrix}
\]  \quad (75)

Applying this result to our platform model, given by Equations (65) through (67), results in

\[
\begin{bmatrix}
W_1 \\
W_2 \\
W_3 \\
W_4
\end{bmatrix} = G \begin{bmatrix}
A_1 \\
A_2 \\
A_3 \\
A_4
\end{bmatrix}
\]  \quad (76)
where

\[
G = \begin{bmatrix}
C_2 & C_3 & 0 & uC_4 \\
C_3 & C_4 - uC_6 & -uC_4 & 0 \\
0 & -uC_4 & C_2 & C_3 \\
uC_4 & 0 & C_3 & C_4 + uC_6
\end{bmatrix}
\]  \quad (77)

\[
C_i = \sum_{k=1}^{N} k^i \quad i = 2, 3, 4, \text{and} \ 6 \quad (78)
\]

\[
W_1 = \sum_{k=1}^{N} kV_N(k), \quad W_2 = \sum_{k=1}^{N} k^2[V_N(k) - ukV_E(k)]
\]

\[
W_3 = \sum_{k=1}^{N} kV_E(k), \quad W_4 = \sum_{k=1}^{N} k^2[V_E(k) + ukV_N(k)]
\]  \quad (79)

Values of \(A_i, i = 1 \sim 4\), are given by

\[
\begin{bmatrix}
A_1 \\
A_2 \\
A_3 \\
A_4
\end{bmatrix} = G^{-1}
\begin{bmatrix}
W_1 \\
W_2 \\
W_3 \\
W_4
\end{bmatrix}
\]  \quad (80)

Notice that the measurements \(V_N(k)\) and \(V_E(k)\) contribute to the estima-
tion via the computation of \(W_i, i = 1 \sim 4\). Also, the property of the
platform kinematics together with the choice of cost function result
in the absence of \(C_5\) in the computation.

The estimate for the initial state vector is obtained from \(A_i, i = 1 \sim 4\), using Equations (57) and (58).
Consider a simple case having a single measurement equation

\[ V_k = A_k + B_k^2 + B_k^3 + n_k \]  \hspace{1cm} (82)

Let \( A = 4 \), \( B = 2 \), and \( k = 1 \sim 10 \). The values of \( V_k \) are generated by adding noise \( n_k \) to the value of polynomial at each \( k \). The values of \( n_k \) are taken from a table of normally distributed random numbers, having zero mean and a variance of one. These values are listed in Table 1.

### Table 1. Measurement Data for Example 1

<table>
<thead>
<tr>
<th>( K )</th>
<th>( n_k )</th>
<th>( V_k )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-1.276</td>
<td>6.724</td>
</tr>
<tr>
<td>2</td>
<td>-0.318</td>
<td>31.628</td>
</tr>
<tr>
<td>3</td>
<td>-1.377</td>
<td>82.623</td>
</tr>
<tr>
<td>4</td>
<td>2.334</td>
<td>178.334</td>
</tr>
<tr>
<td>5</td>
<td>-1.336</td>
<td>318.864</td>
</tr>
<tr>
<td>6</td>
<td>0.414</td>
<td>528.414</td>
</tr>
<tr>
<td>7</td>
<td>-0.494</td>
<td>811.506</td>
</tr>
<tr>
<td>8</td>
<td>1.048</td>
<td>1185.048</td>
</tr>
<tr>
<td>9</td>
<td>0.347</td>
<td>1656.347</td>
</tr>
<tr>
<td>10</td>
<td>0.637</td>
<td>2240.637</td>
</tr>
</tbody>
</table>

Both new and usual least square algorithms are used to estimate \( A \) and \( B \) from the data \( V_k \), \( k = 1 \sim 10 \). The results are shown in Table 2, listing values of estimates and their percentage error as compared to true values. It is apparent that the new algorithm produces much more
accurate estimates. The new algorithm gives better results even if fewer measurements are used. Appendix A contains the computer program for this example.

Example 2

Consider the following case of two measurement equations:

\[ V_K = A_k + B_k^2 + D_k^2 + n_k \]
\[ V_K' = C_k + D_k^2 + B_k^3 + n_k' \]  \( (83) \)

Let \( A = 4, B = 2, C = 3, D = 1 \), and \( k = 1 \sim 10 \). The values of \( V_K \) and \( V_K' \) are generated in a manner similar to that for Example 1. Table 3 lists the measurement data.

TABLE 3. MEASUREMENT DATA FOR EXAMPLE 2

<table>
<thead>
<tr>
<th>( K )</th>
<th>( n_K )</th>
<th>( n'_K )</th>
<th>( V_K )</th>
<th>( V'_K )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-1.276</td>
<td>-1.218</td>
<td>5.724</td>
<td>4.782</td>
</tr>
<tr>
<td>2</td>
<td>-0.318</td>
<td>-0.799</td>
<td>23.682</td>
<td>25.201</td>
</tr>
<tr>
<td>3</td>
<td>-1.377</td>
<td>-1.257</td>
<td>55.623</td>
<td>70.743</td>
</tr>
<tr>
<td>4</td>
<td>2.334</td>
<td>-0.337</td>
<td>114.334</td>
<td>155.663</td>
</tr>
<tr>
<td>5</td>
<td>-1.136</td>
<td>0.642</td>
<td>193.864</td>
<td>290.642</td>
</tr>
<tr>
<td>6</td>
<td>0.414</td>
<td>-0.011</td>
<td>312.414</td>
<td>485.989</td>
</tr>
<tr>
<td>7</td>
<td>-0.494</td>
<td>0.364</td>
<td>468.506</td>
<td>756.364</td>
</tr>
<tr>
<td>8</td>
<td>1.048</td>
<td>0.037</td>
<td>673.048</td>
<td>1112.034</td>
</tr>
<tr>
<td>9</td>
<td>0.347</td>
<td>2.816</td>
<td>927.347</td>
<td>1568.816</td>
</tr>
<tr>
<td>10</td>
<td>0.637</td>
<td>0.563</td>
<td>1240.677</td>
<td>2130.463</td>
</tr>
</tbody>
</table>

33
Again, both new and usual least square algorithms are employed to estimate A, B, C, and D. The results are listed in Table 4, with the percentage error of each estimate estimated. Again, the new algorithm gives much better results. Appendix B presents the computer program for this example.

<table>
<thead>
<tr>
<th></th>
<th>A = 4</th>
<th>B = 2</th>
<th>C = 3</th>
<th>D = 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Usual Algorithm</td>
<td>3.61133</td>
<td>2.08984</td>
<td>2.07031</td>
<td>1.2207</td>
</tr>
<tr>
<td>New Algorithm</td>
<td>3.95093</td>
<td>2.00272</td>
<td>2.88883</td>
<td>1.00099</td>
</tr>
</tbody>
</table>

4. Sensitivity Consideration

It is expected that the new algorithm is less sensitive, as compared to the usual algorithm, with respect to erratic measurement data and to computation errors. The rationale is that the coupled parameters have a tendency to hold each other at their nominal values. Example 3 will show this effect.

Example 3

This example uses the same measurement model, same measurement data, and same computer programs as those used for Example 2. To observe the effect of erratic measurement data on estimates, the measurement V(10) is changed by 1% and a set of new estimate for A, B, C, and D is made using new and usual algorithms. To observe the effect of computation error on estimates, the value of V(10) is restored to its original value and the value of C6 is changed by 0.01%. Another set of estimates are made using both algorithms. All estimates are listed in Table 5. Values of sensitivity in the table are calculated using the formula

\[ S = \frac{\text{New estimate} - \text{Nominal estimate}}{\text{Nominal estimate}} \]  \hspace{1cm} (84)

The result confirms the expectation that new algorithm has lower sensitivity.

Example 4

In this example a real world platform system is considered. The platform is of the class proposed for PERSHING II application. Twelve hundred and fifty pairs of measurement data were recorded at outputs of north and east accelerometers. The total sampling time is 240 seconds.
### Table 5. Sensitivity Comparison for Example 3

<table>
<thead>
<tr>
<th>Condition</th>
<th>Algorithm Used</th>
<th>Nominal</th>
<th>1% Change in V(10)</th>
<th>0.01% Change in C₆</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Usual</td>
<td>New</td>
<td>Usual</td>
</tr>
<tr>
<td>A = 4</td>
<td>A</td>
<td>3.61133</td>
<td>3.95093</td>
<td>5.41016</td>
</tr>
<tr>
<td></td>
<td>S</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B = 2</td>
<td>B</td>
<td>2.08984</td>
<td>2.00272</td>
<td>1.36035</td>
</tr>
<tr>
<td></td>
<td>S</td>
<td></td>
<td></td>
<td>34.9%</td>
</tr>
<tr>
<td>C = 3</td>
<td>C</td>
<td>2.07031</td>
<td>2.88883</td>
<td>2.07031</td>
</tr>
<tr>
<td></td>
<td>S</td>
<td></td>
<td></td>
<td>0.0%</td>
</tr>
<tr>
<td>D = 1</td>
<td>D</td>
<td>1.22070</td>
<td>1.00099</td>
<td>1.22070</td>
</tr>
<tr>
<td></td>
<td>S</td>
<td></td>
<td></td>
<td>0.0%</td>
</tr>
</tbody>
</table>

A - Estimate of A  
S - Sensitivity

The quantities to be estimated are misalignments and platform drifts. New and usual least square algorithms are used for data reduction. The former consists of Equations (77) through (81) while the latter consists of Equations (59) through (64) and (87). The results of data reduction are shown in Table 6. To explore the sensitivity of both algorithms with respect to computation errors, a poor matrix inversion subroutine is used. When computations are done with double precision, both algorithms produce reasonable results. When computations are done with ordinary precision, the result from new algorithm is not reasonable, knowing the quality of the platform used. But the result from the usual algorithm is ridiculous regardless of the platform considered. The results show the superiority of the new algorithm.

### Table 6. Result for Example 4

<table>
<thead>
<tr>
<th></th>
<th>Ordinary Precision</th>
<th>Double Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>New Algorithm</td>
<td>Usual Algorithm</td>
</tr>
<tr>
<td>$\theta_{NO}$</td>
<td>1162.8 arcsec</td>
<td>$4.736 \times 10^{26}$ arcsec</td>
</tr>
<tr>
<td>$\theta_{EO}$</td>
<td>695.6 arcsec</td>
<td>$2.599 \times 10^{26}$ arcsec</td>
</tr>
<tr>
<td>$D_{NO}$</td>
<td>0.275 deg/hr</td>
<td>$5.362 \times 10^{36}$ deg/hr</td>
</tr>
<tr>
<td>$D_{EO}$</td>
<td>-0.0017 deg/hr</td>
<td>$2.824 \times 10^{36}$ deg/hr</td>
</tr>
</tbody>
</table>
The only disappointment in this example is that the exact values of misalignments and drifts were not available at the time of experiment, therefore a precise comparison of two results could not be made. To partially overcome this difficulty, theoretical error analyses are developed in Section VIII. These analyses will help to evaluate the quality of algorithms. Appendix C presents two computer programs used in this example.
Section VIII. THEORETICAL ERROR ANALYSES

The approach of theoretical error analyses used here is to develop analytical relationships relating the standard deviation of estimation error to the standard deviation of the noise. The analyses are done for new and usual least square algorithms.

1. Analysis for New Algorithm

Recall Equation (80) and define

$$H = G^{-1}$$  \hspace{1cm} (85)

then

$$\begin{bmatrix} A_1 \\ A_2 \\ A_3 \\ A_4 \end{bmatrix} = H \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix}.$$  \hspace{1cm} (86)

$H$ is a $4 \times 4$ matrix whose $ij$-element will be denoted by $h_{ij}$. Expanding the first row of Equation (86) and using the relationships in Equation (79), $A_1$ can be expressed as

$$A_1 = \sum_{j=1}^{4} h_{1j} W_j$$

$$= h_{11} \sum_k kV_N(k) + h_{12} \left\{ \sum_k k^2V_N(k) - u \sum_k k^3V_E(k) \right\}$$

$$+ h_{13} \sum_k kV_E(k) + h_{14} \left\{ \sum_k k^2V_E(k) + u \sum_k k^3V_N(k) \right\}.$$  \hspace{1cm} (87)

The error in $A_1$ is caused by errors in $V_N(k)$ and $V_E(k)$ which are denoted by $e_N(k)$ and $e_E(k)$, respectively. Let $e_1$ represent the error of $A_1$, then from Equation (87) we can get
\begin{align*}
e_1 &= h_{11} \sum_k k e_N(k) + h_{12} \left\{ \sum_k k^2 e_N(k) - u \sum_k k^3 e_E(k) \right\} \\
&+ h_{13} \sum_k k e_E(k) + h_{14} \left\{ \sum_k k^2 e_E(k) + u \sum_k k^3 e_N(k) \right\} \\
&= \sum_k \left\{ \left( h_{11} k + h_{12} k^2 + h_{14} u k^3 \right) e_N(k) \\
&+ \left( h_{13} k + h_{14} k^2 - h_{12} u k^3 \right) e_E(k) \right\} \\
\end{align*}

which expresses the error in \( A \) in terms of source errors.

Assume the following statistical properties for source errors:

a) Zero mean, i.e.,
\[ < e_N(k) > = < e_E(k) > = 0 \]  
\hspace{1cm} (89)

where "\(< >\)" denotes ensemble average

b) Uncorrelated between axes and from time to time, i.e.,
\[ < e_N(i) e_N(j) > = < e_E(i) e_E(j) > = 0 \]  
\hspace{1cm} i \neq j \hspace{1cm} (90)

\[ < e_N(i) e_E(j) > = 0 \hspace{1cm} \text{all } i, j \hspace{1cm} (91) \]

c) Equal and stationary variance, i.e.,
\[ < e_N^2(k) > = < e_E^2(k) > = \sigma_e^2, \]  \hspace{1cm} \text{constant.} \hspace{1cm} (92)

Taking the ensemble average over the square of Equation (88), applying previously mentioned error properties, and rearranging terms, we can obtain the error variance of \( A_1 \) as follows:
\[ \sigma_{A_1}^2 = \sigma_e^2 \sum_k \left\{ \left( h_{11} k + h_{12} k^2 + u h_{14} k^3 \right)^2 \\
+ \left( h_{13} k + h_{14} k^2 - u h_{12} k^3 \right)^2 \right\}. \hspace{1cm} (93) \]

We shall digress for a moment to derive a number of relationships which will help to simplify the final expression. Substituting details
of \( V_N(k) \) and \( V_E(k) \) as given in Equations (65) and (66) into Equation (67) and regrouping terms,

\[
A_1 = A_1 \sum k \left( h_{11}k + h_{12}k^2 + uh_{14}k^3 \right) \\
+ A_2 \sum k^2 \left( h_{11}k + h_{12}k^2 + uh_{14}k^3 \right) - uk^3 \left( h_{13}k + h_{14}k^2 - uh_{12}k^3 \right) \\
+ A_3 \sum k \left( h_{13}k + h_{14}k^2 - uh_{12}k^3 \right) \\
+ A_4 \sum k^2 \left( h_{13}k + h_{14}k^2 - uh_{12}k^3 \right) + uk^3 \left( h_{11}k + h_{12}k^2 + uh_{14}k^3 \right)
\]

(94)

comparing both sides of Equation (94) shows that the coefficient of \( A_1 \) should be 1, and those of \( A_2, A_3, \) and \( A_4 \) should be zero. Therefore we get the relationships:

\[
\sum k \left( h_{11}k + h_{12}k^2 + uh_{14}k^3 \right) = 1 \\
\sum k^2 \left( h_{11}k + h_{12}k^2 + uh_{14}k^3 \right) - uk^3 \left( h_{13}k + h_{14}k^2 - uh_{12}k^3 \right) = 0 \\
\sum k \left( h_{13}k + h_{14}k^2 - uh_{12}k^3 \right) = 0 \\
\sum k^2 \left( h_{13}k + h_{14}k^2 - uh_{12}k^3 \right) + uk^3 \left( h_{11}k + h_{12}k^2 + uh_{14}k^3 \right) = 0
\]

(95)

Return to our analysis of error variance and apply the relationships of Equation (95) to Equation (93). The result is a very neat expression,

\[
\frac{\sigma_{A_1}^2}{\sigma^2} = \frac{h_{11}^2}{h_{11}} = h_{11}
\]

(96)
where $\eta_1^2$ is the normalized error variance for $A_1$. The normalization is done with respect to the variance of source error.

In the similar manner, the normalized error variances for $A_2$, $A_3$, and $A_4$ are obtained as

$$\eta_2^2 = \frac{\sigma_{A2}^2}{\sigma_e^2} = h_{22}$$

$$\eta_3^2 = \frac{\sigma_{A3}^2}{\sigma_e^2} = h_{33}$$

$$\eta_4^2 = \frac{\sigma_{A4}^2}{\sigma_e^2} = h_{44}$$

Notice that values of $h_{ii}$, $i = 1 \sim 4$, depend solely on $N$, the number of measurements, and $u$, the correlation parameter. Therefore, normalized error variances $\eta_i^2$, $N = 1 \sim 4$, are independent of measurement data, but depend on the kinematics of the platform misalignment.

By taking the square roots of Equations (96) through (99), equations for normalized standard deviation of estimates are obtained as

$$\eta_i = \sqrt{h_{ii}} , \quad i = 1 \sim 4$$

another very neat expression.

Equation (100) is very useful in several ways. For a given set of error standard deviations of the source and the number of measurements, it can be used to estimate the error standard deviations of estimates. However, for a given set of source error standard deviations and a set of prescribed standard deviations for estimates, it can be used to determine the minimum number of measurements needed. Finally, knowing the error standard deviation of the estimate and the number of measurements, the equation can be used to determine the standard deviation of source error, a tool for identification.

2. Analysis for Usual Algorithm

Recall Equations (59) and (60), and define

$$Q = C^{-1}$$

40
Q is a $4 \times 4$ matrix whose $ij$-element will be denoted by $q_{ij}$. Adopting an approach of derivation similar to that for the new algorithm, the expression of normalized error variances for estimates of $A_i$, $i = 1 \sim 4$, can be obtained as

$$
\gamma_i^2 = q_{ii}^2, \quad i = 1 \sim 4 . \quad (102)
$$

Similarly, the normalized standard deviation expression is given by

$$
\eta_i = \sqrt{q_{ii}}, \quad i = 1 \sim 4 . \quad (103)
$$

In this case, $\eta_i$ depends only on $N$, the number of measurements, but not on the correlation parameter.

3. Comparison

For a given measurement condition the accuracy of estimates produced by new and usual algorithms can be compared by comparing their standard deviations. Direct comparison of Equation (100) and Equation (103) in their literal forms is difficult. A numerical example will be used to demonstrate the superiority of new algorithm.

Example 5

Consider the same platform alignment problem of Example 4. Figures 8 and 9 show plots of normalized standard deviations as functions of $N$, the number of measurement. Axes of the plots are in log-scale.

For $N = 1250$, the new algorithm gives

\[
\begin{align*}
\gamma_{\theta_{NO}} &= \gamma_{\theta_{EO}} = 0.24 \times 10^{-7} \\
\gamma_{D_{NO}} &= \gamma_{D_{EO}} = 0.26 \times 10^{-13}
\end{align*}
\]

while the usual algorithm gives

\[
\begin{align*}
\gamma_{\theta_{NO}} &= \gamma_{\theta_{EO}} = 0.15 \times 10^{-6} \\
\gamma_{D_{NO}} &= \gamma_{D_{EO}} = 0.94 \times 10^{-12}
\end{align*}
\]

Comparing Equation (104) to Equation (105), the superiority of new algorithm is evident.

Appendix D contains the computer program used for providing plotting data for Figures 8 and 9.
Figure 8. Normalized error standard deviation for Example 5.
Figure 9. Normalized error standard deviation for Example 5.
Section IX. RECOMMENDED FURTHER STUDY

There are several problems which deserve further study. Solution to these problems will allow a truly optimum implementation of the IMU self-alignment systems.

Statistical theory shows that more accurate estimates are obtained with larger N, the number of measurements. However, computer error analysis shows that larger N results in more computer error because more computation is involved. It is desirable to choose an N such that the total error is at its minimum. A method for making such a choice is yet to be developed.

Even though the new least square algorithm is less sensitive to computation errors as compared to the usual algorithm, it is still desirable to keep computation errors as small as possible, especially those occurring during matrix inversion. Notice that the G matrix of Equation (77) is symmetric and has four zero elements. This special form may allow the development of a matrix inversion subroutine which is more efficient in computation accuracy and computation time.

The new algorithm reported here is given in the form of batch process. This algorithm can be modified to become a sequential process or a hybrid process which is a semi-batch-semi-sequential process.

It is desirable to verify the analytically predicted superiority of the new least square algorithm for platform alignment by a precision hardware IMU which can be calibrated for experimental comparison.

The analytic results obtained from this study provide insights for coarse alignment which is required prior to the fine alignment. The coarse alignment can also be performed automatically and rapidly with the aid of the computer already available for fine alignment. It will be interesting to explore the possibility of a combined coarse and fine alignment using the same equipment and giving a best overall alignment result.
Section X. CONCLUSIONS

An original contribution of this study is the development and analysis of a new least square regression algorithm specially for the self-alignment of IMU systems. It was shown experimentally, as well as analytically, that this new algorithm is superior to the usual algorithm in accuracy and in sensitivity.

Although the new algorithm was originally intended for the self-alignment of a gimbaled platform, it can be used for the alignment of a strapdown platform as well, with some minor modifications. It can also be used for an IMU consisting of electro-optical sensors, because the underlying kinematic principle is similar.

Other results of this study include thorough derivation for drift equations, misalignment equations, and the gyrocompassing equation. Several self-alignment concepts were reviewed and discussed using the analytic foundation developed. Five examples were developed to help in confirming the theoretical prediction.

Several areas deserving further investigation were recommended. The solution to these areas will allow a truly optimum implementation of IMU self-alignment systems.
REFERENCES


Appendix A. COMPUTER PROGRAMS FOR EXAMPLE 1

(IN BASIC)

```
10 READ C2,C3,C4,C5,C6
20 DATA 385,3025,25333,226825,1.97840E+06
40 LET Q1=C2
41 LET Q2=C3+C4
42 LET Q3=C4+2*C5+C6
50 LET D=Q1*Q3-Q2*Q2
60 LET M1=Q3/D
61 LET M2=-Q2/D
62 LET M3=M2
63 LET M4=Q1/D
100 DIM V[10]
110 FOR I=1 TO 10
120 READ V[I]
130 PRINT "V("I")="V[I]
140 NEXT I
150 DATA 6.724,31.6288,82.623,178.334,318.864
160 DATA 528.414,811.506,1185.05,1656.35,2240.64
180 LET X1=0
190 LET X2=0
200 FOR J=1 TO 10
210 LET X1=J*V[J]+X1
220 LET X2=J*J*(C+J)*V[J]+X2
230 NEXT J
260 LET A=M1*X1+M3*X2
270 LET B=M3*X1+M4*X2
280 PRINT "A="A,"B="B
290 PRINT "TRUE VALUES ARE: A=4 B=2 C=B=2"
300 END
```
10 READ C3, C4, C5, C6
20 DATA 385, 2025, 25333, 220625, 1.97840E+96
105 REM COMPUTATION OF K1 TO K9
110 LET D = C2*(C4*C6-C5*C5) - C3*(C3*C6-C4*C5) + C4*(C3*C5-C4*C4)
140 LET K1 = ((C4*C6-C5*C5)/D
150 LET K2 = (C4*C5-C3*C6)/D
160 LET K3 = (C3*C5-C4*C4)/D
170 LET K4 = K2
180 LET K5 = (C2*C6-C4*C4)/D
190 LET K6 = (C3*C4-C2*C5)/D
200 LET K7 = K3
210 LET K8 = K6
220 LET K9 = (C2*C4-C3*C3)/D
300 REM READ IN OBSERVATIONS
305 DIM V(10)
310 FOR I = 1 TO 10
320 READ VC1
330 PRINT "VC" I = "VC1"
341 NEXT I
350 DATA 6.724, 31.682, 82.623, 75.334, 318.864
360 DATA 528.41, 811.506, 1185.05, 1656.35, 2240.64
400 REM WITH PRECOMPUTED CONSTANTS, DATA PROCESSING BEGINS HERE.
405 LET X1 = 0
406 LET X2 = 0
407 LET X3 = 0
410 FOR J = 1 TO 10
420 LET X1 = J*V(J1)+X1
430 LET X2 = J*V(J)+X2
440 LET X3 = J*J*V(J)+X3
450 NEXT J
500 LET A = K1*X1+K2*X2+K3*X3
510 LET B = K4*X1+K5*X2+K6*X3
520 LET C = K7*X1+K8*X2+K9*X3
530 PRINT "A="A, "B=" B, "C=" C
550 PRINT "TRUE VALUES ARE: A=4, B=2, C=B=2"
600 END
Appendix B. COMPUTER PROGRAMS FOR EXAMPLES 2 AND 3
(IN BASIC)

10 PRINT "LEAST SQUARES ALGO. USIf PARAMETER CORRELATION"
12 LET C2=365
13 LET C3=3025
14 LET C4=25333
15 LET C5=220825.
16 LET C6=1.97640E+06
18 PRINT
20 DIM V[10]
30 FOR I=1 TO 10
40 READ V[I]
50 NEXT I
70 DATA 5.724,23.682,55.623,114.334,193.864
72 DATA 312.414,463.509,673.048,927.347,1240.64
60 PRINT
90 FOR J=1 TO 10
100 READ U[J]
120 NEXT J
130 DATA 4.782,25.201,70.743,155.649,460.642
132 DATA 485.989,756.364,1112.04,1568.82,2130.56
200 DIM U[4,1]
240 FOR K=1 TO 10
290 NEXT K
340 DIM J[4,4]
393 LET G[1,3]=G[3,1]=0
462 DIM H[4,4]
470 MAT H=INV(G)
500 DIM X[4,1]
565 PRINT "X VECTOR"
596 PRINT
610 MAT X=H*W
523 MAT PRINT X
543 PRINT "TRUE VALUES: X1=4 X2=2 X3=3 X4=1"
680 END
10 PRINT "THE USUAL LEAST SQ. ALGO."
20 PRINT
20 DIM V(10)
30 FOR I=1 TO 10
40 READ VI1
60 NEXT I
70 DATA 5.724,23.682,55.623,114.334,193.864
72 DATA 312.414,468.506,673.348,927.347,1249.64
80 DIM U(10)
90 FOR J=1 TO 10
100 READ U(J)
120 NEXT J
130 DATA 4.782,25.201,70.743,155.663,293.642
132 DATA 465.989,756.364,1112.04,1566.82,2133.56
200 READ C2,C3,C4,C5,C6
210 DATA 385.3825,25333,220825,.1,97840E+06
220 LET D=C2*C4*C6-C5*C5-C3*C6-C4*C5+C4*(C3*C5-C4*C4)
230 LET K1=(C4*C6-C5*C5)/D
240 LET K2=(C4*C5-C3*C6)/D
250 LET K3=(C3*C5-C4*C4)/D
270 LET K5=(C2*C6-C4*C4)/D
280 LET K6=(C3*C4-C2*C5)/D
310 LET K9=(C2*C4-C3*C3)/D
320 LET X1=X2=X3=0
350 FOR I=1 TO 10
360 LET X1=X1+I*I*V(I)
370 LET X2=X2+I*I*V(I)
380 LET X3=X3+I*I*I*V(I)
390 NEXT I
400 LET Y1=Y2=Y3=0
430 FOR J=1 TO 10
440 LET Y1=Y1+J*U(J)
450 LET Y2=Y2+J*J*U(J)
460 LET Y3=Y3+J*J*J*U(J)
470 NEXT J
480 LET A=K1*X1+K2*X2+K3*X3
490 LET B=K4*X1+K5*X2+K6*X3
500 LET C=K7*X1+K8*X2+K9*X3
510 LET E=K1*Y1+K2*Y2+K3*Y3
520 LET F=K4*Y1+K5*Y2+K6*Y3
530 LET G=K7*Y1+K8*Y2+K9*Y3
550 PRINT "S="S,"F="F,"G="G
560 PRINT
570 PRINT "TRUE VALUES ARE:"
580 PRINT "A=","B=","C=1"
590 PRINT "S=3","F=1","G=2"
600 END
Appendix C. COMPUTER PROGRAMS FOR EXAMPLE 3
(IN FORTRAN)

PROGRAM MAIN (INPUT=OUTPUT,TAP:INPUT, TAP:OUTPUT)
C***LEAST SQUARE ALGORITHM USING PARAMETER CORRELATION

INTEGER DELPN, DELPE
DIMENSION C(5), G(4,4), DELPN(1250), DELPE(1250), VN(1250),
1 VF(1250), W(4), X(4), VNOFF(1250), VEOFF(1250), H(4,4)
DOUBLE PRECISION G, H, W, X

C*****SETTING UP C(2) TO C(6)  ( C(5) IS NOT USED )
DO 2 I=1,6
C(2) = 1.0
DO 4 K=1,250
FK=K
4 C(I)=C(I)+FK**I
2 CONTINUE
WRITE(6,6) (I, C(I), I=2,6)
6 FORMAT (1H1// 5(1X,C(1)E20.12))

C*****ESTABLISHING G-MATRIX
C EARTH RATE = 7.29211E-05 RADIAN/SSECOND
C LATITUDE = 34.6425 DEGREE
C SAMPLING PERIOD, TAU = 0.192 SECOND
C = -(7-COMPONENT OF EARTH RATE)*TAU/3
C = -(7.29211E-05 * SIN34.6425) * 0.192 / 3
C = 0.26525471E-05

H(1,1)=C(7)
G(1,7)=C(3)
G(1,7)=0.0
G(1,4)=U*G(4)
G(2,1)=G(1,2)
G(2,7)=G(6,7)-C(4)*C(4)
G(2,3)=(1.0-C(4))
G(2,4)=0.0
G(3,1)=0.0
G(3,2)=G(2,2)
G(3,3)=C(2)
G(3,7)=C(3)
G(4,1)=G(1,4)
G(4,7)=0.0
G(4,4)=G(7,7)
G(4,7)=C(4)*U*G(1)
WRITE 'A' ((G(I,J), J=1,4), I=1,4)
A FORMAT (1H0,7X,G-MATRIX/4X,4F24.14/4X,4E24.14/)
1 4X,4F24.14/4X,4F24.14)

C*****GETTING G = G-INVERSE (IN-PLACE STORAGE)
CALL MTINV(G,4)
WRITE (4,4) ((G(I,J), J=1,4), I=1,4)
4 FORMAT (1H0,7X,G-INVERSE/4X,4F24.14/4X,4E24.14/)
1 4X,4F24.14/4X,4F24.14)
FOR DEBUGGING USE ONLY
C TO CHECK THE INVERSE OF THE INVERSE OF G
DO 90 I=1,4
DO 91 J=1,4
91 H(I,J)=G(I,J)
80 CONTINUE
CALL 'TXINV(H*4)
WRITE(6,82)((H(I,J), J=1,4), I=1,4)
82 FORMAT(1H0/7X,**-INVFPSE~ss
!NVERSE*/4(4X,9dE24*I4/))
CDEBUGGING INFORMATION ENDS

READ IN MEASURED DATA
READ(5,10) (DELPN(I), DFLPE(I), I=1,1250)
10 FORMAT(10(2I4))
WRITE(6,12) (DELPN(I), I=1,1250)
12 FORMAT(1H1,10X,*DELPN(I), I=1 TO 1250,**//63(8X*20I6/))
WRITE(6,13) (DFLPE(I), I=1,1250)
13 FORMAT(1H1,10X,*DFLPE(I), I=1 TO 1250,**//63(8X*20I6/))

COMPUTING VNOFF(I) AND VEOFF(I) (IN NUMBER OF PULSES)
VNOFF(I)=DELPN(I)*55
VEOFF(I)=DFLPE(I)*40
DO 40 I=2,1250
AUXN = DELPN(I) * 55
AUXF = DFLPE(I) * 40
VNOFF(I)=VNOFF(I-1)+AUXN
40 VEOFF(I)=VEOFF(I-1)+AUXF
WRITE(6,42)
42 FORMAT (1H1,10X,*I*, 8X,*VNOFF(I)*, 12X,*VEOFF(I)*
WRITE(6,44) (25*,VNOFF(25*I), VEOFF(25*I), I=1,50)
44 FORMAT (4X,18E24.14)

OBTAINING VN(I) AND VE(I) BY COORDINATE TRANSFORMATION
(IN NUMBER OF PULSES)
C HATH = 150 DEGREES = 2.61799387R RADIANS
HATH=2.61799387R
SR=SN(HATH)
C0=COS(HATH)
DO 14 I=1,1250
VN(I) = VNOFF(I)*SR - VEOFF(I)*SR
14 VE(I) = VEOFF(I)*C0 + VNOFF(I)*SR
WRITE(6,30)
30 FORMAT (1H1,10X,*I*, 10X,*VN(I)*, 15X,*VF(I)*)
WRITE(6,15) (25*, VN(25*I), VF(25*I), I=1,50)
15 FORMAT (4X,18E24.14)

COMPUTING W-VECTOR
W(1)=0.0
W(2)=0.0
W(3)=0.0
W(4)=0.0
DO 16 K=1,1250
FK=K
\[ W(1) = W(1) + FK*VN(K) \]
\[ W(2) = W(2) + FK*FK*(VN(K) - U*FK*VE(K)) \]
\[ W(3) = W(3) + FK*VE(K) \]
\[ W(4) = W(4) + FK*FK*(VE(K) + U*FK*VN(K)) \]

WRITE(6,17) (I, W(I), I=1,4)

17 FORMAT (1H1//'\(14X,'W(1),=E20.12/'1)

C***COMPUTING X-VECTOR
DO 18 I=1,4
   Y(I) = 0.0
18 CONTINUE
DO 20 J=1,4
   X(I) = X(I) + G(I,J)*W(J)
20 CONTINUE
WRITE(6,22) (I, X(I), I=1,4)
22 FORMAT (1H0//'\(14X,'X(I),=E20.12/')

C***COMPUTING PLATFORM PARAMETERS
C N-AXIS SCALE FACTOR SFN=100441.
C E-AXIS SCALE FACTOR SFE=101712.
SFN=100441.
SFE=101712.
TAU=0.197
A=1./TAU
B=2./TAU
ZETANO=A*X(3)/SFN
ZETAFO=-(A)*X(1)/SFE
DFN=D*X(4)/SFN
DFTINO=-(B)*X(2)/SFE

WRITE(6,24) ZETANO, ZETAFO, DFTINO, DFTI
24 FORMAT (1H0//'\(14X,'ZETANO,=E20.12*, RADIANS//'1)
   14X,'ZETAFO,=E20.12*, RADIANS//'1
   14X,'DFTINO,=E20.12*, RADIANS/SEC//'1
   14X,'DFTI,=E20.12*, RADIANS/SEC*)
C***END OF THE ESTIMATION PROGRAM
END
SUBROUTINE MTXINV(G,M)

C****ON INPUT G IS G. ON OUTPUT G IS THE INVERSE OF G

DIMENSION G(4,4)
DOUBLE PRECISION G
DO 140 K=1,M
IF (G(K,K)) 10, 160, 10
10 GZZ=1.0/G(K,K)
DO 90 I=1,M
IF (I-K) 20, 90, 60
20 CONST=G(I,K)*GZZ
DO 90 J=I,M
IF (J-K) 30, 50, 60
30 G(I,J)=G(I,J)+CONST*G(J,K)
GO TO 50
40 G(I,J)=G(I,J)-CONST*G(K,J)
50 CONTINUE
GO TO 90
60 CONST=G(K,I)*GZZ
DO 90 J=1,M
IF (J-K) 170, 80, 70
70 G(I,J)=G(I,J)-CONST*G(K,J)
80 CONTINUE
90 CONTINUE
DO 110 J=K,M
IF (K-J) 100, 110, 180
100 G(K,J)=G(K,J)*GZZ
110 CONTINUE
DO 130 I=1,K
IF (K-I) 190, 130, 120
120 G(I,K)=G(K,I)*GZZ
130 CONTINUE
G(K,K)=GZZ
140 CONTINUE
DO 150 I=2,M
JI=I-1
DO 150 J=1,JI
150 G(I,J)=G(J,I)
RETURN
160 WRITE(6,210)
GO TO 200
170 WRITE(6,220)
GO TO 200
180 WRITE(6,230)
GO TO 200
190 WRITE(6,240)
200 CONTINUE
210 FORMAT(4X,**ZERO DIAGONAL ELEMENT, RAD DATA*)
220 FORMAT(4X,**ERROR IN INDEXING IN DOING DO 80*)
230 FORMAT(4X,**ERROR IN INDEXING IN DOING DO 110*)
240 FORMAT(4X,**ERROR IN INDEXING IN DOING DO 130*)
END

54
PROGRAM MAIN(INPUT,OUTPUT,TAPES=INPUT,TAPE6=OUTPUT)

C THE USUAL LEAST SQUARE ALGORITHM

C 3-TERM 3RD ORDER POLYNOMIAL

INTEGER DELPN, DFLPE
DIMENSION DELPN(1250) + DFLPE(1250) + VN(1250) + VE(1250) +
1 VNOFF(1250) + VEOFF(1250) + Q(3,3)
DOUBLE PRECISION Q

C*****SETTING UP C7 TO C6
C2=0.0
C3=0.0
C4=0.0
C5=0.0
C6=0.0
DO K=1,1250
FK=K
C2=C2+FK**2
C3=C3+FK**3
C4=C4+FK**4
C5=C5+FK**5
C6=C6+FK**6
WRITE(6,4) C7, C3, C4, C5, C6
4 FORMAT (1H1, //10X,*C2=*F20.12/10X,*C3=*F20.12/
1 10X,*C4=*F20.12/10X,*C5=*F20.12/10X,*C6=*F20.12)

C*****COMPUTING D AND Q1 TO Q9
O(1+1)=C2
O(1+2)=C3
O(1+3)=C4
O(2+1)=C7
O(2+2)=C7
O(2+3)=C7
O(3+1)=C4
O(3+2)=C5
O(3+3)=C6
CALL SYMINV(O,3)
WRITE(6,6) ( (O(I,J)+J=J), J=J, I=I+1)
6 FORMAT (1H0/7X,*Q-INVRSQ/3(4X,3E24,14/))

C*****READ IN MSAGORED DATA
READ(5,10) (DELPN(I), DFLPE(I), I=1,1250)
10 FORMAT (10(2F4))
WRITE(6,12) (DELPN(I)* I=1,1250)
12 FORMAT (1H1,10X,*DELPN/I, I=1 TO 1250//63(AX,2016/) )
WRITE(6,13) (DFLPE(I)), I=1,1250)
13 FORMAT (1H1,10X,*DFLPE/I, I=1 TO 1250//63(AX,2016/) )

C*****COMPUTING VNOFF(I) AND VEOFF(I) (IN NUMBER OF PULSES)
VNOFF(I)=DELPN(I)*55.0
VEOFF(I)=DFLPE(I)*40.0
DO 40 I=2,1250
40
VNOFF(I)=VNOFF(I-1)+DELPHN(I)*55.0
40 VEOF(I)=VNOFF(I-1)+DELPH(I)*40.0
WRITE(6,42)
42 FORMAT (4H1I,10X,1E9,8X,VNOFF(I)*12X*VEOFF(I))
WRITE(6,44) (25*I,VNOFF(25*I)+VEOFF(25*I),I=1,50)
44 FORMAT (4X,1A,2E24.15)

C****OBTAINING VN(I) AND VE(I) BY COORDINATE TRANSFORMATION
C
(IN NUMBER OF PULSES)
C
RATH = 150 DEGREES = 2.61799387R RADIANS
BATH=2.61799387R
SB=SIN(BATH)
CR=COS(BATH)
DO 16 I=1,1250
VN(I)=VNOFF(I)*CR-VEOFF(I)*SR
14 VE(I)=VEOFF(I)*CR+VNOFF(I)*SB
WRITE(6,10)
10 FORMAT (4H1I,10X,1E9,8X,VN(I),1X,VE(I))

C
COMPUTING Y1, Y2, Y3, Z1, Z2, Z3
Y1=0.0
Y2=0.0
Y3=0.0
DO 17 I=1,1250
FJ=I
Y1=Y1+FJ*VN(I)
Y2=Y2+FJ*FI*VN(I)
17 Y3=Y3+FI+FI*FI*VN(I)
Z1=0.0
Z2=0.0
Z3=0.0
DO 18 J=1,1250
FJ=J
Z1=Z1+FJ*VF(J)
Z2=Z2+FJ*FI*VF(J)
18 Z3=Z3+FJ*FI*FJ*VF(J)

C****COMPUTING X1 TO X4
X1=0.1+0.1+Y1+Q(1,2)+Y2+O(1,3)+Y3
X2=0.2+0.1+Y1+Q(2,2)+Y2+O(2,3)+Y3
Y3=0.1+Z1+Q(1,2)+Z2+O(1,3)+Z3
X4=0.2+Z1+Q(2,2)+Z2+O(2,3)+Z3
WRITE(6,20) (X1*X2*X3*X4)
20 FORMAT (4H1I,14X*F1.12/F2.0,12/14X*F2.0,12/F2.0,12)

C****COMPUTING PLATFORM PARAMETERS
C
N-AXIS SCALE FACTOR SFN=100441.
F-AXIS SCALE FACTOR SFE=101712.
SFN=100441.
SFE=101712.
TAU=0.192
A = 1. / TAU
R = 2. / TAU**2
ZETANO = A * X3 / SFN
ZETAFE = (-A) * X1 / SFN
DRFTNO = B * X4 / SFN
DRFTEO = (-R) * X2 / SFN

WRITE(6, 24) ZETANO, ZETAFE, DRFTNO, DRFTEO

24 FORMAT (1HO///14X*ZETANO=E20.12* Radian///
1 14X*ZETAFE=E20.12* Radian///
1 14X*DRFTNO=E20.12* Radian/Sec///
1 14X*DRFTEO=E20.12* Radian/Sec*)

C*****END OF THE FORTRAN PROGRAM

END
SUBROUTINE SYMINV(G,M)

C*****ON INPUT G IS G. ON OUTPUT G IS THE INVERSE OF G
DIMENSION G(3,3)
DOUBLE PRECISION G
DO 140 K=1,M
  IF (G(K,K)) 10, 160, 10
  10 ZZ=1.0/G(K,K)
  DO 90 I=1,M
    IF (I-K) 20, 90, 60
  20 CONST=G(I,K)*ZZ
    DO 50 J=1,M
      IF (J-K) 30, 50, 40
    30 G(I,J)=G(I,J)+CONST*G(J,K)
    GO TO 50
  40 G(I,J)=G(I,J)-CONST*G(K,J)
  50 CONTINUE
  GO TO 90
  60 CONST=G(K,I)*ZZ
    DO 80 J=1,M
      IF (J-K) 70, 80, 90
    70 G(I,J)=G(I,J)-CONST*G(K,J)
    80 CONTINUE
  90 CONTINUE
  DO 110 J=K,M
    IF(K-J) 100, 110, 120
  100 G(K,J)=G(K,J)*ZZ
    110 CONTINUE
  DO 130 I=1,K
    IF (K-I) 130, 120, 190
  120 G(I,K)=(-G(I,K))*ZZ
  130 CONTINUE
  G(K,K)=G77
  140 CONTINUE
  DO 150 I=2,M
    J1=I-1
    DO 150 J=1,J1
  150 G(I,J)=G(J,I)
  RETURN
  160 WRITE(6,210)
    GO TO 200
  170 WRITE(6,220)
    GO TO 200
  180 WRITE(6,230)
    GO TO 200
  190 WRITE(6,240)
  200 CONTINUE
  210 FORMAT(4X,*ZERO DIAGONAL ELEMENT, RAD DATA*)
  220 FORMAT(4X,*ERROR IN INDROW IN DOING DO 80*)
  230 FORMAT(4X,*ERROR IN INDROW IN DOING DO 110*)
  240 FORMAT(4X,*ERROR IN INDROW IN DOING DO 130*)
END
Appendix D. COMPUTER PROGRAMS FOR EXAMPLE 5
(IN FORTRAN)

PROGRAM MAIN(INPUT,OUTPUT,TAPES=INPUT,TAPES=OUTPUT)
C****TABLE OF NORMALIZED STANDARD DEVIATIONS FOR THE USUAL
C AND THE NEW LEAST SQUARE ESTIMATION ALGORITHMS

DIMENSION A(4,4), U11(30), U22(30), U33(30), U44(30)

1) = 0.7952947302E-05
C2 = 0.0
C3 = 0.0
C4 = 0.0
C5 = 0.0
C6 = 0.0
DO 20 N = 1, 30
  N = (N-1) * 100 + 1
  I = N * 100
  DO 30 J = 1, I
    C2 = C2 + J**2 * 0
    C3 = C3 + J**3 * 0
    C4 = C4 + J**4 * 0
    C5 = C5 + J**5 * 0
    C6 = C6 + J**6 * 0
  30 C6 = C6 + J**6 * 0
A(1,1) = C2
A(1,2) = C3
A(1,3) = C4
A(2,1) = C5
A(2,2) = C6
A(2,3) = C5
A(3,1) = C4
A(3,2) = C5
A(3,3) = C6
CALL MTXINV(A(3,3))
G11(N) = A(1,1)
G22(N) = A(2,2)
G33(N) = A(3,3)
G44(N) = A(4,4)
G11(1) = C2
G12(1) = C3
G13(1) = 0.0
G14(1) = U*C4
G21(1) = G(1,1)
G22(1) = C4 + U*C6
G33(1) = (-1)*C4
G34(1) = 0.0
G43(1) = 0.0
G44(1) = G(4,1)
G13(3) = C2
G14(3) = C3
G43(1) = G(4,1)

59
G(4*2)=0.0
G(4*3)=G(3*4)
G(4*4)=G(1*4)

CALL MTXINV(G=4)

H11(N)=G(1*1)
H22(N)=G(2*2)
H33(N)=G(3*3)

20 H44(N)=G(4*4)

WRITE(8,40)
40 FORMAT(1X//,10X,0N/100,19X,0G11(N) 24X,0H11(N) 25X,0H33(N) *)
WRITE(6,50) (N,N11(N),H11(N),H33(N),N=1,30)
50 FORMAT(10X,15,10X,F20.12,10X,F20.12,10X,F20.12)

WRITE(8,60)
60 FORMAT(1X//,10X,0N/100,19X,0G22(N) 24X,0H22(N) 25X,0H44(N) *)
WRITE(6,70) (N,N22(N),H22(N),H44(N),N=1,30)
70 FORMAT(10X,15,10X,F20.12,10X,F20.12,10X,F20.12)

WRITE(8,90)
90 FORMAT(1X//,10X,0N/100,19X,0G33(N) *)
WRITE(6,90) (N,N33(N),N=1,30)
90 FORMAT(10X,15,10X,F20.12)

END
SUBROUTINE KTXINV(G,M)
C
COMMON IMPLICIT G IS G, ON OUTPUT G IS THE INVERSE OF G
DIMENSION G(4,4)
DO 140 K=1,M
10 GZZ=1.0/G(K,K)
DO 90 I=1,M
IF (I-K) 20, 90, 90
90 CONTINUE
10 G(I,J)=G(I,J)+CONST*G(J,K)
DO 50 J=I,M
IF (J-K) 30, 50, 40
30 G(I,J)=G(I,J)-CONST*G(K,J)
50 CONTINUE
GU TO 90
60 CONTINUE
DO 90 J=I,M
IF (J-K) 170, 90, 70
70 G(I,J)=G(I,J)-CONST*G(K,J)
80 CONTINUE
90 CONTINUE
DO 110 J=K,M
IF (K-J) 100, 110, 190
100 G(K,J)=G(K,J)-GZ7
110 CONTINUE
DO 130 I=1,K
IF (K-I) 190, 130, 120
120 G(I,K)=-G(I,K)*GZ7
130 CONTINUE
G(K,K)=GZ7
140 CONTINUE
DO 150 I=2,M
J1=I-1
DO 150 J=J1,M
150 G(I,J)=G(J,I)
RETURN
160 WRITE(A,*210)
GO TO 200
170 WRITE(A,*220)
GO TO 200
180 WRITE(A,*230)
GO TO 200
190 WRITE(A,*240)
200 CONTINUE
210 FORMAT(4X,*7F9.0 DIAGONAL ELEMENT, RAD DATA*)
220 FORMAT(4X,*7F9.0 INDXING IN DOING NO 80*)
230 FORMAT(4X,*7F9.0 INDXING IN DOING NO 110*)
240 FORMAT(4X,*7F9.0 INDXING IN DOING NO 130*)
END