Since small values of k_2 permit more admissible initial conditions, it is appropriate to determine the effect on system characteristics if k_2 is set equal to zero.

Redefining the adjustable parameters to be portrayed as $\alpha \equiv a_0 G_D$ and $\gamma \equiv k_1 G_D$, Eq. (2) is solved for α and γ, and the complex root stability boundary (double cross-hatching) is plotted on the α–γ parameter plane (Fig. 2). The real root stability boundary (single cross-hatching) also is determined from Eq. (2), yielding the line $\gamma = 0$. The describing function curve again is a straight line, $\gamma = k_1\alpha/a_0$, on the parameter plane. Values for the control gains may be chosen that will keep the entire line within the stable region of the parameter plane, precluding sustained oscillations and indicating stable operation for the conditions portrayed. Further, a curve corresponding to $\zeta = 0.5$ may be determined by setting $s = -\zeta\omega_n + i\omega_n(1 - \zeta^2)^{1/2}$ in Eq. (2) and repeating the procedure outlined earlier to determine α and γ. Values of α and γ may be chosen so that the describing function line terminates on the $\zeta = 0.5$ curve at a desired control frequency (indicated in parentheses). However, the parameter c is not constant throughout flight.[1] During most of the upper stage flight it has a value of approximately 0.7, rising to 2.5 for a short time duration. A stability region is shown on Fig. 2 for that higher value of c, and this additional constraint is imposed on the selection of the terminus for the describing function line. If values of -0.415 and -0.110 are chosen for α and γ and the slope K of the saturation limiter is set at unity, limit cycle operation for all initial values of $\dot{\sigma}$ (input to the saturation characteristic) is obviated and stable operation is indicated.

The ratio A/S equals unity at the describing function line terminus; if $S = 1$, then A is limited to 1°/sec (as desired). Good damping characteristics are indicated except for the short time that c approaches values of 2.5. The indicated value of $\omega = 0.433$ for $c = 0.7$ (and $\omega = 7$ for $c = 2.5$) provides an acceptable frequency separation from the guidance and bending frequencies. Hence it is seen that stable operation and a close match to the desired design characteristics established for control frequency, damping ratio, and saturation limit are indicated by this analysis. The simplicity of analyzing both linear and nonlinear operation in the parameter plane is noteworthy, as is the fact that the effect on linear and nonlinear stability of adjusting two control gains (in this case a_0 and k_1) can be seen directly (whereas the Nyquist plot used in Ref. 1 provides this visibility for only one gain at a time). The effect on vehicle performance of setting $k_2 = 0$ will of course be degrading. However, $k_2 \neq 0$ during ascent through the atmosphere, so the effect of wind on drift is still suppressed. After k_2 is set equal to zero, the effects on steady state drift may be estimated analytically through application of the final value theorem. These estimates provide an upper bound for they assume open-loop guidance, whereas in reality, the guidance loop will be closed

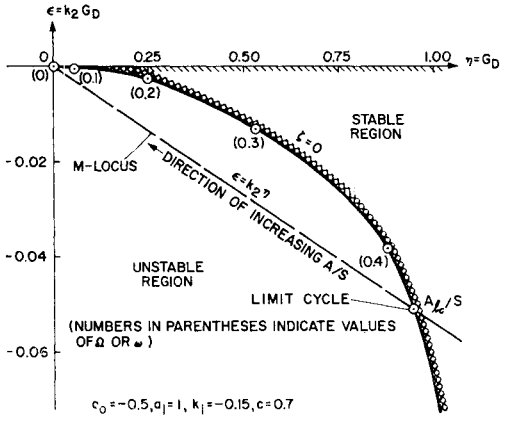

Fig. 1 Parameter plane plot for $k_2 \neq 0$.

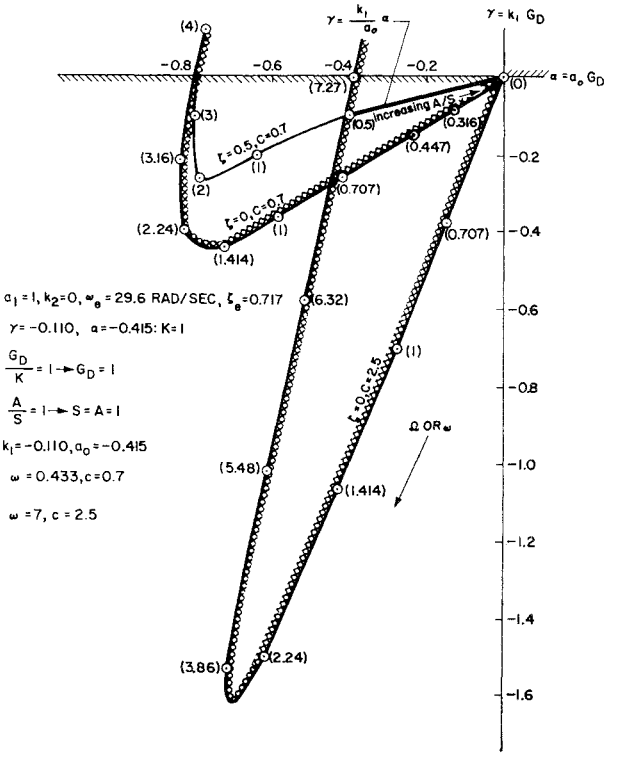

Fig. 2 Parameter plane plot for $k_2 = 0$.

when k_2 is set equal to zero. While a finite value of k_2 will drive to zero the drift velocity due to wind, center of gravity offset, uncertainties in thrust vector direction due to hardware tolerances, and errors in reading the gimbal angles of the stable platform, setting $k_2 = 0$ will result in finite but small values for the first three of these drift producers and a zero value for the fourth.[3]

References

[1] Seltzer, S. M., "An Attitude Control System for Large Launch Vehicles," *Journal of Spacecraft and Rockets*, Vol. 6, No. 6, June 1969, pp. 748–751.

[2] Siljak, D., *Nonlinear Systems*, Wiley, New York, 1969, pp. 152–171 and 177–186.

[3] Seltzer, S. M., "Steady-State Responses for Several Control Systems," TMX-53504, Aug. 1960, NASA.

Iterative Computation of Initial Quaternion

ITZHACK Y. BAR-ITZHACK*
Bellcomm Inc., Washington, D.C.

WHEN quaternions are used to compute the time varying transformation matrix between two Cartesian coordinate systems,[1,2] the initial quaternion has to be known. In most of the cases the initial transformation matrix, rather than the initial quaternion, is known, and the latter has to be computed. In this Note, an iterative technique for computing the quaternion that corresponds to an orthogonal matrix is presented. This technique is based on a known, self-alignment technique of an analytic platform. In its

Received August 12, 1969; revision received November 21, 1969.

* Member of the Technical Staff, Space Vehicle Dynamics Department, Control Group.

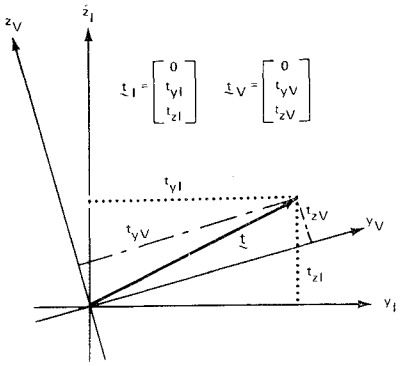

Fig. 1 A two-dimensional example for the construction of t_I and t_V column vectors.

implementation in a space vehicle guidance computer, we can share portions of the program that originally were designed for initial alignment, hence very little has to be added to the computer program in order to use this technique.

The orthogonal transformation matrix at any time t is computed by two steps. 1) The quaternion of rotation, $q(t)$, is found by solving the following quaternion differential equation[2,3]:

$$\dot{q}(t) = \tfrac{1}{2}q(t)\cdot\omega_q(t) \tag{1}$$

where $\omega_q(t)$ is a known quaternion constructed of the components of the angular velocity of the rotating coordinate system which are obtained through measurements. 2) The transformation matrix is computed algebraically,[2-4] using the components of $q(t)$ at that time.

In order to solve Eq. (1), the initial quaternion has to be known. If initially the two coordinate systems are aligned, then the transformation matrix is the unity matrix. The quarternion that corresponds to this matrix (i.e., the quaternion that through step 2 will result in this matrix) is the following one: $q_0 = (1,\bar{0})$. This statement can be easily verified by following step 2 for q_0. If, on the other hand, the two coordinate systems do not coincide initially, then the initial quaternion which corresponds to the initial transformation matrix does not take the simple form $(1,\bar{0})$ and has to be computed. We shall present an iterative method to find a quaternion, which is a generalization of an implicit self-alignment method[5] for analytic inertial platform systems.

A Generalized Implicit Self-Alignment Process

By mathematically aligning one coordinate system with respect to another, we mean that a direction cosine computer routine, when fed with the angular velocity of one coordinate system with respect to another, computes the DCM (direction cosine matrix) which transforms vectors from the latter to the first coordinate system. This routine could use, for example, the two steps previously described or the method used in Ref. 5. Any 3 × 3 orthogonal matrix stored in the DCM computer could be changed to coincide with any chosen 3 × 3 orthogonal matrix by properly feeding the DCM computer with three signals representing three components of an imaginary angular velocity vector. This operation is called slewing, and the three input signals are known as the matrix slewing rate vector or the matrix slewing signals.

The alignment problem is stated as follows. Given the components of two vectors in two different cartesian coordinate systems, what is the DCM which transforms any vector from one coordinate system to another? (The two vectors are necessary to determine this matrix and they are also sufficient, once the right-handedness of the coordinate systems is specified.) Let the two coordinate systems be denoted by I and V, the two vectors by $\mathbf{s}$ and $\mathbf{t}$ and the DCM which transforms vectors from the V to the I coordinate system by $D_I{}^V$. Then

$$\mathbf{s}_I = D_I{}^V\mathbf{s}_V, \quad \mathbf{t}_I = D_I{}^V\mathbf{t}_V \tag{2}$$

where $\mathbf{s}_I$ and $\mathbf{t}_I$ are the column vectors $\mathbf{s}$ and $\mathbf{t}$, respectively, whose components are the projections of these vectors on the axes of the I coordinate system, and $\mathbf{s}_V$ and $\mathbf{t}_V$ are the same for the projections in the V coordinate system (Fig. 1); $\mathbf{s}_V$, $\mathbf{t}_V$, $\mathbf{s}_I$ and $\mathbf{t}_I$ are given, and the problem is to find the DCM, $D_I{}^V$.

The implicit self-alignment process is an iterative process by which $D_I{}^V$ is found and whose steps are now described. We choose a certain arbitrary matrix D_0. For simplicity let us choose the unity matrix. Using this matrix we compute

$$\mathbf{s}_{I0} = D_0\mathbf{s}_V, \quad \mathbf{t}_{I0} = D_0\mathbf{t}_V \tag{3}$$

If D_0 is the sought $D_I{}^V$, then $\mathbf{s}_{I0} = \mathbf{s}_I$, and $\mathbf{t}_{I0} = \mathbf{t}_I$, and the cross products $\mathbf{s}_{I0} \times \mathbf{s}_I$ and $\mathbf{t}_{I0} \times \mathbf{t}_I$ will be equal to zero, and the iterative process will end. If, however, D_0 is not $D_I{}^V$, then $\mathbf{s}_{I0} \neq \mathbf{s}_I$, and $\mathbf{t}_{I0} \neq \mathbf{t}_I$, and, therefore,

$$\mathbf{p}_1 \triangleq \mathbf{s}_{I0} \times \mathbf{s}_I \neq 0 \qquad \mathbf{p}_2 \triangleq \mathbf{t}_{I0} \times \mathbf{t}_I \neq 0 \tag{4}$$

The vectors $\mathbf{p}_1$ and $\mathbf{p}_2$, as defined in Eqs. (4), are a measurement of the difference between the true and the assumed matrices. They can be used as a correcting signal to slew the matrix toward the true $D_I{}^V$ matrix. The matrix slewing signal could be either the simple sum of $\mathbf{p}_1$ and $\mathbf{p}_2$, or the sum of the weighed vectors, i.e.,

$$\boldsymbol{\omega} = K_1(s)\mathbf{p}_1 + K_2(s)\mathbf{p}_2 \tag{5}$$

where $\boldsymbol{\omega}$ is the imaginary angular velocity vector which is the slewing signal. $K_1(s)$ and $K_2(s)$ are the transfer weighing functions. The slewing signal changes D_0 to D_1 and the same steps are repeated again. Finally, when $\mathbf{p}_1$ and $\mathbf{p}_2$ as defined by Eqs. (4) equal zero, the process terminates and the matrix stored in the DCM computer is the matrix $D_I{}^V$. The process always converges.[5]

For the special case of the alignment of an analytic inertial platform, $\mathbf{s}_V$ consists of the three accelerations measured by the system's three accelerometers, and $\mathbf{t}_V$ consists of the angular rates measured by the three gyros of the system, and

$$\mathbf{s}_I = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad \mathbf{t}_I = \begin{bmatrix} 0 \\ \cos\Lambda \\ \sin\Lambda \end{bmatrix}$$

where Λ is the latitude angle. The V coordinate system is defined by the orientations of the gyros and the accelerometers, and the I system is the local level system.

Slewing a Matrix to a Desired DCM

Let us consider a special case where the vector $\mathbf{s}$ is $\mathbf{k}_I$, i.e., the unit vector in the direction of the Z axis of the I-coordinate system. In addition let $\mathbf{t}$ be $\mathbf{j}_I$, i.e., the unit vector in the direction of the Y axis in the same coordinate system. It is clear, then, that

$$\mathbf{s}_I = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}; \quad \mathbf{t}_I = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{6}$$

Writing the expanded form of $D_I{}^V$ as

$$D_I{}^V = \begin{bmatrix} d_{11} & d_{12} & d_{13} \\ d_{21} & d_{22} & d_{23} \\ d_{31} & d_{32} & d_{33} \end{bmatrix} \tag{7}$$

it is obvious that for this special case

$$\mathbf{s}_V = \begin{bmatrix} d_{31} \\ d_{32} \\ d_{33} \end{bmatrix}, \quad \mathbf{t}_V = \begin{bmatrix} d_{21} \\ d_{22} \\ d_{23} \end{bmatrix} \tag{8}$$

since Eqs. (6–8) comply with Eqs. (2).

Having defined $\mathbf{s}_I$, $\mathbf{s}_V$, $\mathbf{t}_I$, and $\mathbf{t}_V$ we can apply now the iterative self-alignment process as described in the previous section. At the end of the process the DCM stored in the computer will be $D_I{}^V$, whose second and third rows are given

by $\mathbf{t}_V$ and $\mathbf{s}_V$, respectively (the first row is therefore uniquely defined too). We conclude, therefore, that given a certain DCM we can always start the process with an arbitrary matrix which will finally converge to the given DCM. The seemingly trivial operation is useful in the computation of the initial quaternion as will now be shown.

Iterative Computation of an Initial Quaternion

The problem is, given a DCM, find the quaternion which, when used in step 2 of the first section yields the given DCM. We apply the foregoing iterative method, as described in Fig. 2. We pick $q_0 = (1,\bar{0})$ as the initial quaternion; hence, the initial matrix is the unity matrix. We use this matrix to transform $\mathbf{s}_V$ to get $\mathbf{A}$, and transform $\mathbf{t}_V$ to get $\mathbf{B}$. The matrix slewing signal is obtained by

$$\boldsymbol{\omega} = K_1(\mathbf{A} \times \mathbf{s}_I) + K_2(\mathbf{B} \times \mathbf{t}_I) \tag{9}$$

where K_1 and K_2 are the weighing functions. The slewing signal is fed into the DCM computer, which computes the new quaternion and then the corresponding new DCM. This process is repeated until $\boldsymbol{\omega} = 0$. Then the matrix stored in the DCM computer is the given matrix, hence, the known quaternion of the stored matrix is the sought quaternion.

Finally, we may simplify the system shown in Fig. 2 by writing a more explicit expression for $\boldsymbol{\omega}$. We note that

$$K_1(\mathbf{A} \times \mathbf{s}_I) = K_1\begin{bmatrix} a_2 \\ -a_1 \\ 0 \end{bmatrix} \tag{10}$$

where a_1 and a_2 are the first and second components of A, respectively. We also note that

$$K_2(\mathbf{B} \times \mathbf{t}_I) = K_2\begin{bmatrix} -b_3 \\ 0 \\ b_1 \end{bmatrix} \tag{11}$$

where b_1 and b_3 are the first and third components of B, respectively. From Eqs. (9–11) we see that

$$\boldsymbol{\omega} = \begin{bmatrix} K_1a_2 - K_2b_3 \\ -K_1a_1 \\ K_2b_1 \end{bmatrix} \tag{12}$$

Let the elements of the DCM in the computer be

$$DCM = \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix}$$

Then,

$$a_j = \sum_{i=1}^{3} C_{ji}d_{3i},\ b_j = \sum_{i=1}^{3} C_{ji}d_{2i}$$

Thus, Eq. (12) becomes

$$\boldsymbol{\omega} = \begin{bmatrix} \sum_{i=1}^{3} (K_1C_{2i}d_{3i} - K_2C_{3i}d_{2i}) \\ -K_1 \sum_{i=1}^{3} C_{1i}d_{3i} \\ K_2 \sum_{i=1}^{3} C_{1i}d_{2i} \end{bmatrix} \tag{13}$$

Example

Choosing

$$D_I{}^V = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix}$$

For K_1 a pure gain of 0.6 and K_2 a pure gain of 1.0, the quaternion obtained was

$$q_0 = (0.70710739 + i\cdot 0.70710461 + j\cdot 0.0 + k\cdot 0.0)$$

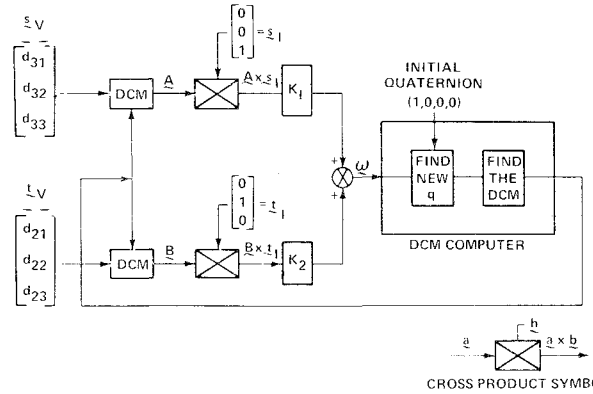

Fig. 2 Iterative computation of the initial quaternion.

Indeed the chosen matrix is the transformation matrix which transforms vectors to a coordinate system from another system obtained by rotating the first system by 90° about its x axis. The direction of the rotation angle vector is in the positive direction of the x axis, i.e., $\mathbf{j}_V = \mathbf{k}_I$ and $\mathbf{k}_V = -\mathbf{j}_I$. The rotation angle vector takes, therefore, the form:

$$\boldsymbol{\theta} = \mathbf{i}\cdot\pi/2 + \mathbf{j}\cdot 0 + \mathbf{k}\cdot 0$$

Given this angle, the quaternion can be found by[2,3]

$$\lambda = \cos(|\boldsymbol{\theta}|/2)$$

$$\rho_i = (\theta_i/|\boldsymbol{\theta}|)\cdot(\sin|\boldsymbol{\theta}|/2) \quad i = x,y,z$$

Therefore, in this case: $\lambda = \cos(\pi/4)$, $\rho_x = \sin(\pi/4)$, and $\rho_y = \rho_z = 0$. Hence

$$q_0 = [\cos(\pi/4) + i\cdot\sin(\pi/4) + j\cdot 0 + k\cdot 0]$$

which checks with the iterative result.

Conclusion

An iterative technique for computing the initial quaternion which is based on a self-alignment method of an inertial analytic platform has been presented. Its main advantage is the saving in computer programing. The main part of the computer hardware necessary for this technique is the DCM computer, which is a part of the hardware anyway. If the alignment technique used in the system is the Implicit Self-Alignment Process, then the same hardware with different inputs could be used to execute the initial quaternion computation.

The efficiency of this technique, which depends on $K_1(s)$ and $K_2(s)$, could be increased by introducing time-varying functions for K_1 and K_2 which are adapted during the computation to give minimum iterations and maximum accuracy.

References

[1] Wiener, T. F., "Theoretical Analysis of Gimballess Inertial Reference Equipment Using Delta Modulated Instruments," Ph.D. dissertation, 1962, Massachusetts Institute of Technology.

[2] Schwartz, B., "Strapdown Inertial Navigation Systems," *Sperry Engineering Review*, Vol. 20, No. 1, 1967.

[3] Bar-Itzhack, I. Y., "Strapdown Inertial Navigation," Ph. D. dissertation, May 1968, Univ. of Pennsylvania, Philadelphia, Pa.

[4] Wilcox, James C., "A New Algorithm for Strapdown Inertial Navigation," *IEEE Transactions on Aerospace Electronic Systems*, Sept. 1967, Vol. AES-3, No. 5.

[5] Harasty, G. A. and F. G. Unger, "Alignment Techniques for Analytic Inertial Platform Systems," Record of the 1965 International Symposium on Space Electronics, Miami Beach, Fla., Nov. 2–4, 1965.

[6] Pio, R. L., "Euler Angle Transformations," *IEEE Transactions on Automatic Control*, Oct. 1963, Vol. AC-11, No. 4.