```
 TEMP(1)= ((((.5D-03*(WPK(4,5,3)*WPK(4,5,3)))+((.5D-03*(WPK(4,5,1)
& *WPK(4,5,1)))+(.001*(C5*C5))))+((VPK(4,5,3)*VPK(4,5,3))+((.01*
& (S5*S5))+(VPK(4,5,1)*VPK(4,5,1)))))+(((.01*(C5*C5))+(.05*
& (S5*S5)))+((4.*((.01*(C5*C5))+((.01*(S5*S5))+(VPK(4,4,1)*
& VPK(4,4,1)))))+((.01*(C5*C5))+(.4*(S5*S5))))))
 TEMP(2)= (((.6*((VPK(4,6,3)*VPK(4,6,3))+((VPK(4,6,1)*VPK(4,6,1))+
& (VPK(4,6,2)*VPK(4,6,2)))))+((.5D-03*(WPK(4,6,3)*WPK(4,6,3)))+
& ((.2D-03*(WPK(4,6,2)*WPK(4,6,2)))+(.5D-03*(WPK(4,5,1)*WPK(4,5,1)
& )))))+TEMP(1))
 M(1,1)= (.08+(((.5*((VPK(4,7,3)*VPK(4,7,3))+((VPK(4,6,2)*VPK(4,6,
& 2))+(VPK(4,7,1)*VPK(4,7,1)))))+((.003*(WPK(4,7,3)*WPK(4,7,3)))+
& ((.001*(WPK(4,7,1)*WPK(4,7,1)))+(.002*(WPK(4,6,2)*WPK(4,6,2)))))
& )+TEMP(2)))
```

Fig. 3 Output from symbolic equation generator.

They reported an operation count of 394 adds and 646 multiplies. Slightly improved operation counts can be realized by using computer programs that utilize symbol manipulation. One such program, SD/EXACT,[1] produces the equations of motion for this arm, which require only 390 adds and 576 multiplies. Thus, equations for this robot can be obtained at less than one-third the operations count predicted by the bounds derived earlier. These results are true for almost every industrial robot in existence. For this problem, the CPU time required to construct the equations of motion is 67 s on a VAX 780 computer. Figure 3 shows the SD/EXACT output for the $m(1,1)$ term.

Conclusions

This Note has presented a new multibody algorithm that produces highly uncoupled equations of motion. The new algorithm has a computational complexity that grows as $O(n^2)$ in the number of degrees of freedom and requires no matrix decompositions to solve for the joint accelerations. When implemented with a symbol manipulator, the new algorithm produces a substantial reduction in operation counts for robotic equations of motion.

References

[1]Rosenthal, D.E. and Sherman, M.A., "Symbolic Multibody Equations via Kane's Method," *Journal of Astronautical Sciences,* Vol. 34, No. 3, July-Sept. 1986, pp. 223-239.

[2]Kreuzer, E.J., "Dynamical Analysis of Mechanisms Using Symbolical Equation Manipulation," *Proceedings of the Fifth World Congress on the Theory of Machines and Mechanisms,* 1979.

[3]Walker, M.W. and Orin, D.E., "Efficient Dynamic Computer Simulation of Robotic Mechanisms," *Journal of Dynamic Systems, Measurement, and Control,* Vol. 104, Sept. 1982, pp. 205-211.

[4]Kane, T.R. and Levinson, D.A., "The Use of Kane's Dynamical Equations in Robotics," *International Journal of Robotics Research,* Vol. 2, Fall 1983.

Algorithm to Generate Geodetic Coordinates from Earth-Centered Earth-Fixed Coordinates

Atul Nautiyal*
Defence Research and Development Laboratory
Hyderabad, India

Introduction

THE transformation of Earth-centered Earth-fixed (ECEF) coordinates to geodetic coordinates is required in many space position measurement systems. In a recent article,[1] Lupash developed an iterative algorithm (described as algorithm A in Ref. 1) to obtain geodetic coordinates from ECEF coordinates. Further, he compared it with the most efficient available iterative algorithm (described as algorithm B in Ref. 1). It has been observed by Lupash that algorithm A has a convergence problem near the equator, whereas algorithm B has a convergence problem near the pole. In this Note, an iterative algorithm is developed that is free from convergence problems and has better convergence property than both algorithms A and B.

Received Dec. 18, 1986; revision received March 16, 1987.

*Scientist, Systems Group B.

Mathematical Formulation

The following relations expressing ECEF coordinates X_e, Y_e, Z_e of a point P in terms of geodetic coordinates are well known.

$$X_e = (r + h)\cos\phi\cos\lambda \tag{1}$$

$$Y_e = (r + h)\cos\phi\sin\lambda \tag{2}$$

$$Z_e = [(1 - e^2)r + h]\sin\phi \tag{3}$$

and

$$r = a/(1 - e^2\sin^2\phi)^{1/2}$$

where, for the point P (see Fig. 1),

ϕ = geodetic latitude
λ = geodetic longitude
h = altitude normal to reference ellipsoid
a = ellipsoidal equatorial radius (a = 6378, 135 m based on Model WGS-72)
e^2 = square of eccentricity of reference ellipsoid (e^2 = 0.006694317778 for Model WGS-72)
b = ellipsoidal polar radius ($b = a\sqrt{1 - e^2}$)
θ = geocentric latitude

Futher, in Fig. 1, P_1 is the point of intersection of the reference ellipsoid and the normal through P and

ψ = geocentric latitude of P_1

Let the ECEF coordinates of P_1 be (X_1, Y_1, Z_1). Using Eqs. (1-3) with $h = 0$, for the point P_1, we get

$$\sqrt{X_1^2 + Y_1^2} = r\cos\phi \tag{4}$$

$$Z_1 = (1 - e^2)r\sin\phi \tag{5}$$

From Eqs. (4) and (5), we obtain

$$\tan^2\psi = Z_1^2/(X_1^2 + Y_1^2) = (1 - e^2)^2\tan^2\phi \tag{6}$$

We now consider the intersection of the reference ellipsoid and the plane passing through the point P, the center of the Earth, and the pole N (see Fig. 2). The ellipse obtained by this

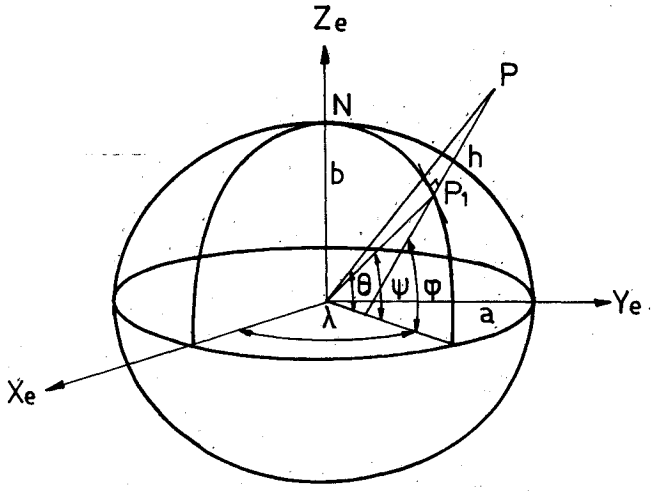

Fig. 1 Geodetic parameters and ECEF frame.

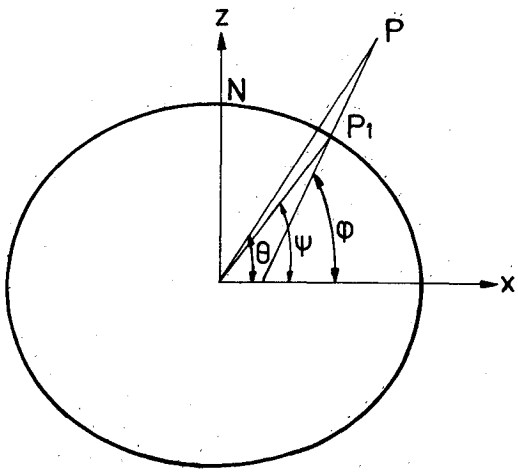

Fig. 2 Geodetic and geocentric latitudes.

intersection can be represented as

$$\frac{x^2}{a^2}+\frac{z^2}{b^2}=1 \tag{7}$$

and the coordinates (x_e, z_e) of P and (x_1, z_1) of P_1 in the x, z plane are given as

$$x_e=\sqrt{X_e^2+Y_e^2}, \qquad z_e=Z_e \tag{8}$$

$$x_1=\sqrt{X_1^2+Y_1^2}, \qquad z_1=Z_1 \tag{9}$$

The equation of normal to ellipse, Eq. (7), through point P_1, is

$$(z_1-z)=(a^2z_1/b^2x_1)(x_1-x)$$

and so the condition that the normal passes through point P is

$$(1-z_e/z_1)=(a^2/b^2)(1-x_e/x_1) \tag{10}$$

Setting

$$x_1=at,\ z_1=b\sqrt{1-t^2} \tag{11}$$

in Eq. (10) [note that this choice of (x_1, z_1) satisfies Eq. (7) of the ellipse], we get

$$\frac{z_e}{b\sqrt{1-t^2}}=-\frac{e^2}{1-e^2}+\frac{a^2x_e}{b^2\,at}$$

Squaring both sides of the aforementioned equation and after rearranging, parameter t satisfies the following condition:

$$f(t)\equiv a_0t^4+a_1t^3+a_2t^2+a_3t+a_4=0 \tag{12}$$

where

$$a_0=B^2, \qquad a_1=2BCD, \qquad a_2=A+C^2D^2-B^2$$

$$a_3=-2BCD, \qquad a_4=-C^2D^2 \tag{13}$$

and

$$A=(z_e/b)^2, \qquad B=-e^2/(1-e^2)$$

$$C=(a/b)^2=1/(1-e^2), \qquad D=x_e/a \tag{14}$$

Equation (12) can be readily solved for t by using the Newton-Raphson method, i.e.,

$$t_{k+1}=t_k-f(t_k)/f'(t_k), \qquad k=0, 1, 2, \ldots \tag{15}$$

where t_0 is the initial guess value for t and

$$f'(t)=4a_0t^3+3a_1t^2+2a_2t+a_3$$

For obtaining the initial guess value t_0 of t, we set the geocentric latitude ψ of P_1 equal to the geocentric latitude θ of P, so that from Eqs. (6), (9), and (11), we get

$$\tan^2\theta=\tan^2\psi=b^2(1-t_0^2)/a^2t_0^2$$

or

$$t_0=\sqrt{1/[1+\tan^2\theta/(1-e^2)]} \tag{16}$$

where

$$\tan\theta=Z_e/\sqrt{X_e^2+Y_e^2}$$

Having determined t, in view of Eq. (11), the coordinates (x_1, z_1) of P_1 in the x, z plane are readily obtained. Thus, using $\tan\psi=z_1/x_1$, we obtain from Eq. (6)

$$\tan^2\phi=(1-t^2)/[t^2(1-e^2)] \tag{17}$$

From Eqs. (1), (2), (4), and (9), we get

$$x_e=x_1+h\cos\phi$$

using Eqs. (8) and (11) we obtain

$$h=(\sqrt{X_e^2+Y_e^2}-at)\sqrt{1+\tan^2\phi} \tag{18}$$

Finally, Eqs. (1) and (2) give the geodetic longitude λ as

$$\lambda=\arctan(Y_e/X_e) \tag{19}$$

The Algorithm

In view of the preceding discussion, the complete algorithm, which we shall call algorithm C, can be implemented as follows:

Algorithm C

Given X_e, Y_e, Z_e and a, b, e^2; ϵ = imposed accuracy, and $k_{\max}$ = maximum permissible iterations.

1. Compute the following coefficients: A, B, C, D [see Eq.

Table 1 Comparison of algorithms

Input data for generating test examples		Number of iterations for convergence of algorithm		
Geodetic latitude, deg	Altitudes, m	A	B	C
0.1	1, 10^3, 10^6	#, #, #	1, 2, 2	1, 1, 2
20	1, 10^3, 10^6	1, 2, 3	2, 3, 4	1, 1, 2
30	1, 10^3, 10^6	1, 2, 4	2, 3, 4	1, 1, 2
45	1, 10^3, 10^6	1, 2, 4	3, 4, 5	1, 1, 2
60	1, 10^3, 10^6	1, 2, 4	3, 4, 5	1, 1, 2
80	1, 10^3, 10^6	6, #, #	3, 4, 5	1, 1, 2
89.9	1, 10^3, 10^6	#, #, #	3, 4, 5	1, 1, 2

(14)] a_0, a_1, a_2, a_3, a_4 [see Eq. (13)], and initialize t_0 [see Eq. (16)], $k = 0$.

2. Set $k = k + 1$.

3. Compute t_k as a function of t_{k-1} [see Eq. (15)].

4. If $|(t_k - t_{k-1})/t_k| \leq \epsilon$, go to step 6; else, go to step 5.

5. If $k < k_{\max}$, go to step 2; else, stop and write the message "Algorithm does not converge."

6. Determine $\tan^2\phi$ from Eq. (17).

7. Determine the geodetic latitude $\phi = \arctan(\tan\phi)$.

8. Compute altitude h from Eq. (18).

9. Compute geodetic longitude λ from Eq. (19).

Numerical Results

We now compare algorithms A and B, as described in Ref. 1, with the present algorithm C.

It is readily seen that the iterative parts of both algorithms A and C consist of updating the root of a fourth-degree polynomial by the Newton-Raphson method, and so they require the same number of arithmetical operations per iteration. Further, unlike algorithm B, algorithms A and C do not require square root evaluation in their iterative parts.

Algorithms A, B, and C were coded in FORTRAN in double precision and various examples were run. Some sample results are presented in Table 1. Assuming $\lambda = 0$ and using Eqs. (1-3), we generated data for test examples for three different altitudes, i.e., 1 m, 10^3 m, and 10^6 m, at each of the geodetic latitudes given in column 1 of Table 1. A slight change was made in the implementation of algorithms A and C, i.e., after each Newton-Raphson iteration of algorithms A and C, altitudes were also calculated. For a meaningful comparison, the same convergence criterion, i.e., the difference between the altitude obtained by a given algorithm and the altitude used to generate the corresponding test example is less than the given prescribed tolerance δ ($\delta = 10^{-5}$ m for Table 1), was applied to all three algorithms. In each row of column 3, three integers correspond to the iterations required for the convergence of algorithm A for test examples corresponding to 1 m, 10^3 m, and 10^6 m altitudes, respectively. In columns 4 and 5, the corresponding number of iterations for algorithms B and C are given. The symbol # denotes that the algorithm failed to converge in 10 iterations.

From Table 1 and other numerical results, the following observations were made:

1) For a given altitude, the iterations required for algorithms A and B depend on the geodetic latitude of the test example, whereas the iterations required for algorithm C do not vary with latitude.

2) Algorithm C has uniformly good performance from the equator to pole; algorithm B converges fast near the equator but its performance deteriorates steadily as we approach the pole; algorithm A has convergence problems near the equator as well as the pole.

3) For no test example did algorithm C require more iterations than algorithm A or B.

Thus, with its uniform good rate of convergence from the equator to pole, algorithm C is preferable to algorithms A and B.

Conclusions

The iterative algorithm developed in this Note is free from any singularity and has a uniformly good rate of convergence from the equator to pole and, hence, it can be considered for real-time applications requiring transformation from Earth-centered Earth-fixed coordinates to geodetic coordinates.

Remark

A polynomial of fourth degree $g(U)$ in $U = [(r + h)/a]^2$ was obtained by eliminating $t = r/a$ from Eqs. (11) and (12) of Ref. 1. Further, an iterative algorithm using the root of $g(U)$ was developed to transform ECEF coordinates to geodetic coordinates. However, algorithm C was found superior to this algorithm as well.

Acknowledgment

The author is thankful to Mr. M. N. Rao, Scientist E at the Defence Research and Development Laboratory, for his useful suggestions.

Reference

[1]Lupash, L.O., "A New Algorithm for the Computation of the Geodetic Coordinates as a Function of Earth Centered Earth Fixed Coordinates," *Journal of Guidance, Control, and Dynamics*, Vol. 8, Nov.-Dec. 1985, pp. 787-789.

Evaluation of Image Stability of a Precision Pointing Spacecraft

Hari B. Hablani*
Rockwell International, Seal Beach, California

Introduction

IN this Note, a device called the clutter leakage metrics, a Fourier integral form that is available in electro-optics literature, is presented in discrete Fourier transform terms. This device evaluates the image stability of a precision-pointing telescope. When a telescope undergoes severe vibrations, it may yield a cluttered image of a target. To minimize this clutter, electro-optic signals from successive frames are subtracted. Some clutter may still leak into the final image but then can be measured by the proposed metrics. These metrics involve the Fourier spectrum of pointing error, integration interval, and differencing operation. The metrics are illustrated for a telescope whose image stability during landmark tracking is disturbed by a neighboring solar array.

Clutter Leakage Metrics for a Telescope

When a telescope is articulated to a base body to which other deformable bodies are also attached, the environment of

Received July 15, 1985; revision received June 12, 1987.

*Member of the Technical Staff, Guidance and Control Group, Satellite Systems Division; Senior Member AIAA.