Cartesian to Geodetic Coordinates Conversion by an Iterative Geometrical Method

G. Panou, R. Korakitis, D. Delikaraoglou

Department of Surveying Engineering, National Technical University of Athens, Zografou Campus, 15780 Athens, Greece

Abstract: A new method to convert Cartesian to geodetic coordinates is presented. For the geodetic longitude the computation is exact but for the other coordinates the computation is based on a repeated application of the direct transformation. Starting with approximate values of geodetic height and latitude, we compute approximate Cartesian coordinates. Using these values, we correct the geodetic height applying a formula from plane geometry. Again, we compute the Cartesian coordinates and using these values we correct the geodetic latitude by a simple spherical formula. Then, comparing the resulting Cartesian coordinates with the given coordinates we repeat the same procedure until convergence. The derivations of the used formulas, as well as the description of the algorithm are presented in detail. Finally, the new approximate iterative method is validated with numerical experiments.

Keywords: coordinate transformation, geodetic coordinates, approximate method, numerical method

1. Introduction

One of the most interesting and useful procedures in geodesy is that of converting Cartesian (x, y, z) to geodetic coordinates (h, φ, λ). While the direct transformation from geodetic to Cartesian coordinates is straightforward, the inverse transformation is a difficult task, with a long history and a plethora of different solutions. In the last sixty years, about seventy papers describing methods of solving this problem, especially when solving for geodetic height and latitude, have been proposed by many researchers, as they are reported in a very comprehensive paper by Featherstone and Claessens (2008). Also, comparisons among some of the existing methods have been conducted by Gerdan and Deakin (1999), Seemkooei (2002) and Fok and Iz (2003). The performance evaluation of the algorithms focuses on the aspects of accuracy, stability and computational speed. Nevertheless, the execution time does no longer play an essential role today with the availability of high speed computers.

The methods of solving this problem can be divided into two general categories: (i) exact methods, e.g., Zhang et al. (2005), Vermeille (2011) and Zeng (2013), and

(ii) approximate methods, e.g., Heiskanen and Moritz (1967), Lin and Wang (1995), Fotiou (1998), Fukushima (1999), Jones (2002), Pollard (2002), Shu and Li (2010), Turner (2009) and You (2000). The exact methods are usually based on the solution of an appropriate quartic equation, which is derived by essential calculus. It is well-known that a quartic equation has an exact analytical solution which, except at the singularities of the formulas, includes several intermediate steps with time consuming mathematical operations, such as cubic roots. Thus, the exact methods, when implemented in code and executed in a computer with limited precision, are less accurate and slower than the approximate methods, see Seemkooei (2002). On the other hand, the approximate methods can be broken into two classes: (a) non-iterative methods (essentially involving a truncated series expansion), e.g., Fotiou (1998), Turner (2009) and You (2000), and (b) iterative methods e.g., Heiskanen and Moritz (1967), Lin and Wang (1995), Fukushima (1999), Jones (2002), Pollard (2002) and Shu and Li (2010). In the first class, the solution is obtained after a particular number of computations but the results are given at a level of accuracy which is the inherent approximation of the method. In contrast, in the second class the level of accuracy is user defined but the execution time is more difficult to predict. Also, they require many numerical tests to confirm their convergence, after a finite number of iterations, to the solution.

In this paper, we present an approximate iterative method to convert Cartesian to geodetic coordinates. The procedure is based on repeated application of the direct transformation and has a clear geometrical interpretation. In each iteration, we apply a height correction followed by a latitude correction. In Sect. 2, the basic tools used in this method are presented. Especially, we derive the formulas for height and latitude corrections. In Sect. 3, we describe the steps of the algorithm, in order to show the implementation clearly. Finally, in Sect. 4, we test the algorithm using numerical experiments.

2. Basic equations

The geodetic coordinates (h, φ, λ) are related to the corresponding Cartesian coordinates (x, y, z) by (Heiskanen and Moritz, 1967)

$$x = (N + h)\cos\varphi\cos\lambda \tag{1a}$$

$$y = (N + h)\cos\varphi\sin\lambda \tag{1b}$$

$$z = \left[N\left(1 - e^2\right) + h\right]\sin\varphi \tag{1c}$$

where N is the normal radius of curvature:

$$N = \frac{a}{\sqrt{1 - e^2\sin^2\varphi}}. \tag{2}$$

The result for geodetic longitude λ is immediately obtained from the first two equations of (1):

$$\lambda = \tan^{-1}\left(\frac{y}{x}\right) \tag{3}$$

or from equivalent and more stable formulas, see Vermeille (2011). Denoting with $r = \sqrt{x^2 + y^2}$, the transformation formulas (1) are written as

$$r = (N + h)\cos\varphi \tag{4a}$$

$$z = \left[N\left(1 - e^2\right) + h\right]\sin\varphi\,. \tag{4b}$$

Thus, the inverse formulas may be written as

$$h = \frac{r}{\cos\varphi} - N \tag{5a}$$

$$\varphi = \tan^{-1}\frac{z(N + h)}{r\left[N\left(1 - e^2\right) + h\right]}\,. \tag{5b}$$

Given x, y, z, and hence r, equations (5) may be solved iteratively for h and φ. This procedure is described in Heiskanen and Moritz (1967), although many other computational non-iterative methods have been devised e.g., Fotiou (1998) and You (2000). The corresponding equations in spherical coordinates (H, Φ), are written as

$$r = (a + H)\cos\Phi \tag{6a}$$

$$z = (a + H)\sin\Phi \tag{6b}$$

$$H = \sqrt{r^2 + z^2} - a \tag{7a}$$

$$\Phi = \tan^{-1}\left(\frac{z}{r}\right). \tag{7b}$$

In the method presented in this paper, we avoid the use of the inverse formulas (5) and we use only the direct formulas (4) together with two correction formulas, which are derived below. Furthermore, without loss of generality, we assume $z \geq 0$.

2.1. Height correction

The Euclidean distance between two points $P_{n,n}\left(r_{n,n}, z_{n,n}\right)$, $n \in \aleph$, and $P(r, z)$ on the rz-plane is given by, see Figure 1

$$\delta s_{n,n} = \sqrt{\delta r_{n,n}^2 + \delta z_{n,n}^2} \tag{8}$$

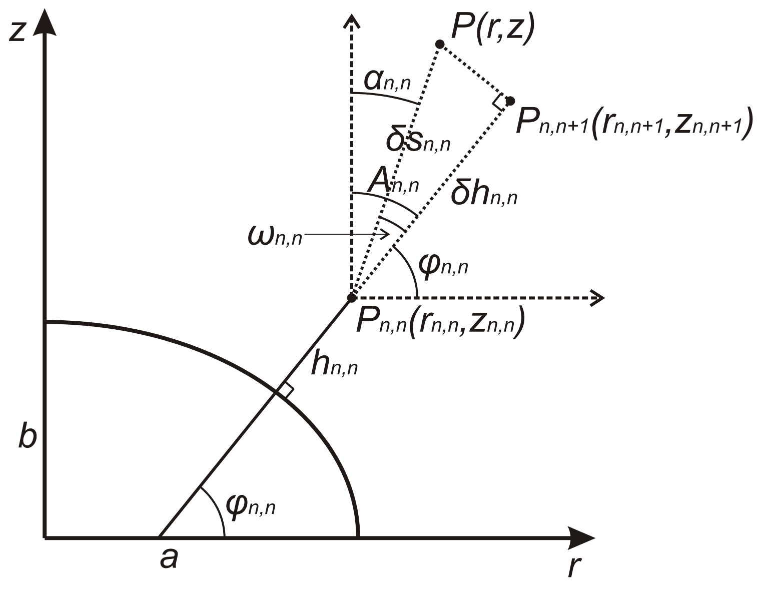


Figure 1. *Height correction*

where

$$\delta r_{n,n} = r - r_{n,n} \tag{9a}$$

$$\delta z_{n,n} = z - z_{n,n}\ . \tag{9b}$$

Hence, the quantity $\delta h_{n,n}$ is computed by

$$\delta h_{n,n} = \delta s_{n,n} \cos \omega_{n,n} \tag{10}$$

where

$$\omega_{n,n} = \left| A_{n,n} - \alpha_{n,n} \right| = \left| \left(\frac{\pi}{2} - \varphi_{n,n} \right) - \alpha_{n,n} \right| \tag{11}$$

with

$$\alpha_{n,n} = \tan^{-1}\left(\frac{\delta r_{n,n}}{\delta z_{n,n}} \right). \tag{12}$$

As we can see in Figure 1, because the points $P_{n,n}\left(r_{n,n}, z_{n,n}\right)$ and $P_{n,n+1}\left(r_{n,n+1}, z_{n,n+1}\right)$ are in the same normal to the meridian ellipse, it holds that:

$$h_{n,n+1} = h_{n,n} + \delta h_{n,n} \tag{13a}$$

$$\varphi_{n,n+1} = \varphi_{n,n}\ . \tag{13b}$$

2.2. Latitude correction

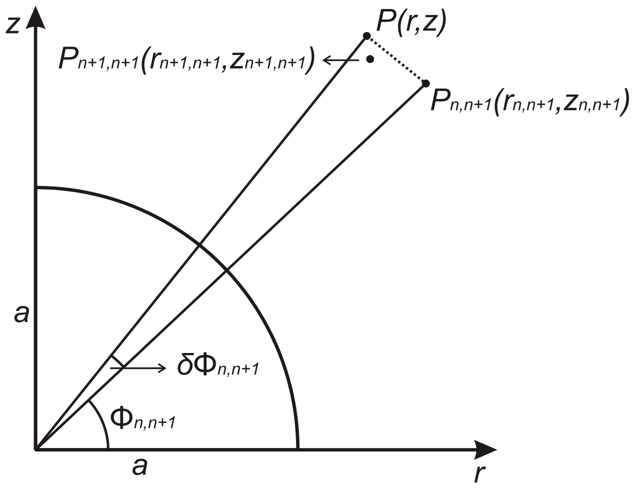
Between two points $P_{n,n+1}(r_{n,n+1}, z_{n,n+1})$ and $P(r,z)$, by differentiation of (6) and combining the results, we obtain (see Figure 2)

$$\delta\Phi_{n,n+1} = \frac{\delta z_{n,n+1} \cos\Phi_{n,n+1} - \delta r_{n,n+1} \sin\Phi_{n,n+1}}{a + H_{n,n+1}} \tag{14}$$

where

$$\delta r_{n,n+1} = r - r_{n,n+1} \tag{15a}$$

$$\delta z_{n,n+1} = z - z_{n,n+1}\,. \tag{15b}$$



z
P(r,z)
$P_{n+1,n+1}(r_{n+1,n+1}, z_{n+1,n+1})$
$P_{n,n+1}(r_{n,n+1}, z_{n,n+1})$
a
$\delta\Phi_{n,n+1}$
$\Phi_{n,n+1}$
a
r

***Figure 2**. Latitude correction*

Now, we make the approximations $\Phi_{n,n+1} \approx \varphi_{n,n+1}$ and $H_{n,n+1} \approx h_{n,n+1}$ in (14), obtaining

$$\delta\varphi_{n,n+1} = \frac{\delta z_{n,n+1} \cos\varphi_{n,n+1} - \delta r_{n,n+1} \sin\varphi_{n,n+1}}{a + h_{n,n+1}}\,. \tag{16}$$

Furthermore, we set

$$h_{n+1,n+1} = h_{n,n+1} \tag{17a}$$

$$\varphi_{n+1,n+1} = \varphi_{n,n+1} + \delta\varphi_{n,n+1} \tag{17b}$$

so that we can obtain the point $P_{n+1,n+1}(r_{n+1,n+1}, z_{n+1,n+1})$.

3. Algorithm

Given $P(r,z)$, we compute as initial values of h and φ the corresponding spherical values H and Φ, from (7). That is, we set $n = 0$, $h_{00} = H$, $\varphi_{00} = \Phi$ and we compute the point $P_{0,0}(r_{0,0}, z_{0,0})$ from (4).

From (8), we compute

$$\delta s_{0,0} = \sqrt{\delta r_{0,0}^2 + \delta z_{0,0}^2} \ . \tag{18}$$

Now, if this value is below the user defined accuracy ε, we stop the iteration otherwise we apply corrections.

Geodetic height correction

Using (10), (11), (12), we compute

$$\delta h_{0,0} = \delta s_{0,0} \cos \omega_{0,0} \tag{19}$$

and we obtain

$$h_{0,1} = h_{0,0} + \delta h_{0,0} \tag{20a}$$

$$\varphi_{0,1} = \varphi_{0,0} \ . \tag{20b}$$

Then we compute the point $P_{0,1}(r_{0,1}, z_{0,1})$ from (4).

Geodetic latitude correction

Using (15), (16), we compute

$$\delta\varphi_{0,1} = \frac{\delta z_{0,1} \cos\varphi_{0,1} - \delta r_{0,1} \sin\varphi_{0,1}}{a + h_{0,1}} \tag{21}$$

and we obtain

$$h_{1,1} = h_{0,1} \tag{22a}$$

$$\varphi_{1,1} = \varphi_{0,1} + \delta\varphi_{0,1} \ . \tag{22b}$$

Again we compute the point $P_{1,1}(r_{1,1}, z_{1,1})$ from (4).

Finally, from (8) we compute

$$\delta s_{1,1} = \sqrt{\delta r_{1,1}^2 + \delta z_{1,1}^2} \tag{23}$$

and we check again if this quantity is smaller than the required accuracy ε. If not this procedure is repeated from (19).

4. Numerical experiments

In this section, the performance of the implementation (i.e., accuracy, stability, computational speed) of the algorithm presented above, is numerically tested. For the data set, the points are obtained by varying geodetic height h from $-6\cdot10^6$ m to $2\cdot10^7$ m by 2600 m steps and geodetic latitude φ from 0° to 90° by 1° steps. This combination gives totally 910091 points. We also used, the geometric parameters of the reference ellipsoid WGS 84 by NIMA (2000), i.e., $a = 6378137.0$ m and $e^2 = 6.69437999014\cdot10^{-3}$. First, we convert the given geodetic (h, φ) to Cartesian (r, z) coordinates using the direct transformation (1). Then, the geodetic coordinates (h*, φ*) are obtained from these Cartesian coordinates using the new algorithm. The height error $\sigma h = |h^* - h|$ is the absolute value of the difference between the given geodetic height and its computed value after t iterations. Similarly, for the latitude error $\sigma\varphi = |\varphi^* - \varphi|$.

For the numerical computations we used a computer with an Intel Core i7 CPU 860, 2.80GHz processor and 4.0GB RAM. The code was programmed using Matlab 7.10. Totally, seven experiments were performed on the whole data set, varying the convergence accuracy ε from 10^{-3} m to 10^{-9} m by 10^{-1} m steps. The statistics of the numerical results are presented in Table 1.

***Table 1**. Statistics of the numerical results computed by the presented algorithm*

ε (m)	$\max\|\sigma h\|$ (m)	$\max\|\sigma\varphi\|$ (")	max t	mean t	time (s)
10^{-3}	$9.6\cdot10^{-6}$	$6.0\cdot10^{-4}$	8	3.1	109
10^{-4}	$1.5\cdot10^{-7}$	$6.0\cdot10^{-5}$	9	3.4	110
10^{-5}	$1.1\cdot10^{-8}$	$6.0\cdot10^{-6}$	10	3.8	112
10^{-6}	$1.1\cdot10^{-8}$	$6.1\cdot10^{-7}$	11	4.2	114
10^{-7}	$1.1\cdot10^{-8}$	$5.9\cdot10^{-8}$	12	4.5	115
10^{-8}	$1.1\cdot10^{-8}$	$5.8\cdot10^{-9}$	13	4.9	116
10^{-9}	$7.5\cdot10^{-9}$	$5.9\cdot10^{-10}$	50	7.9	128

5. Conclusions

We have presented an iterative method to convert Cartesian to geodetic coordinates. This method involves few and simple formulas and has a clear geometrical interpretation. It consists of a height correction, given by an exact formula, and a latitude correction, which involves an inherent approximation, so an iterative approach is required.

Nevertheless, the method is quite efficient. As it is evident from the presented numerical tests, it provides very accurate results (of the order of 1 nm in height and 10^{-9} arcsec in latitude) for geodetic applications in an extremely wide range of heights. In addition, this performance is attained in a very small time (about 0.1 ms) even using a slow computational environment.

References

Featherstone, W.E. and Claessens, S.J. (2008). Closed-form transformation between geodetic and ellipsoidal coordinates. *Stud. Geophys. Geod.*, Vol. 52, No. 1, pp. 1-18.

Fok, H.S. and Iz, H.B, (2003). A comparative analysis of the performance of iterative and non-iterative solutions to the Cartesian to geodetic coordinate transformation. *Journal of Geospatial Engineering*, Vol. 5, No. 2, pp. 61-74.

Fotiou, A. (1998). A pair of closed expressions to transform geocentric to geodetic coordinates. *Z. f. V.*, Vol. 123, No. 4, pp. 133-135.

Fukushima, T. (1999). Fast transform from geocentric to geodetic coordinates. *J. Geod.*, Vol. 73, No. 11, pp. 603-610.

Gerdan, G.P. and Deakin, R.E. (1999). Transforming Cartesian coordinates X, Y, Z to geographical coordinates φ, λ, h. *The Australian Surveyor*, Vol. 44, No. 1, pp. 55-63.

Heiskanen, W.A. and Moritz, H. (1967). *Physical Geodesy*, Freeman and Co., San Francisco and London.

Jones, G.C. (2002). New solutions for the geodetic coordinate transformation. *J. Geod.*, Vol. 76, No. 8, pp. 437-446.

Lin, K.-C. and Wang, J. (1995). Transformation from geocentric to geodetic coordinates using Newton's iteration. *Bull. Geod.*, Vol. 69, No. 4, pp. 300-303.

NIMA, (2000). National Imagery and Mapping Agency (USA), Technical Report, TR8350.2, 3.

Pollard, J. (2002). Iterative vector methods for computing geodetic latitude and height from rectangular coordinates. *J. Geod.*, Vol. 76, No. 1, pp. 36-40.

Seemkooei, A.A. (2002). Comparison of different algorithms to transform geocentric to geodetic coordinates. *Surv. Rev.*, Vol. 36, No. 286, pp. 627-633.

Shu, C. and Li, F. (2010). An iterative algorithm to compute geodetic coordinates. *Computers & Geosciences*, Vol. 36, No. 9. pp. 1145-1149.

Turner, J.D. (2009). A non-iterative and non-singular perturbation solution for transforming Cartesian to geodetic coordinates. *J. Geod.*, Vol. 83, No. 2, pp. 139-145.

Vermeille, H. (2011). An analytical method to transform geocentric into geodetic coordinates. *J. Geod.*, Vol. 85, No. 2, pp. 105-117.

You, R.-J. (2000). Transformation of Cartesian to geodetic coordinates without iterations. *J. Surv. Eng.*, Vol. 126, No. 1, pp. 1-7.

Zeng, H. (2013). Explicitly computing geodetic coordinates from Cartesian coordinates. *Earth, Planets and Space*, Vol. 65, No. 4, pp. 291-298.

Zhang, C-D., Hsu, H.T., Wu, X.P., Li, S.S., Wang, Q.B., Chai H.Z. and Du, L., (2005). An alternative algebraic algorithm to transform Cartesian to geodetic coordinates, *J. Geod.*, Vol. 79, No. 8, pp. 413-420.