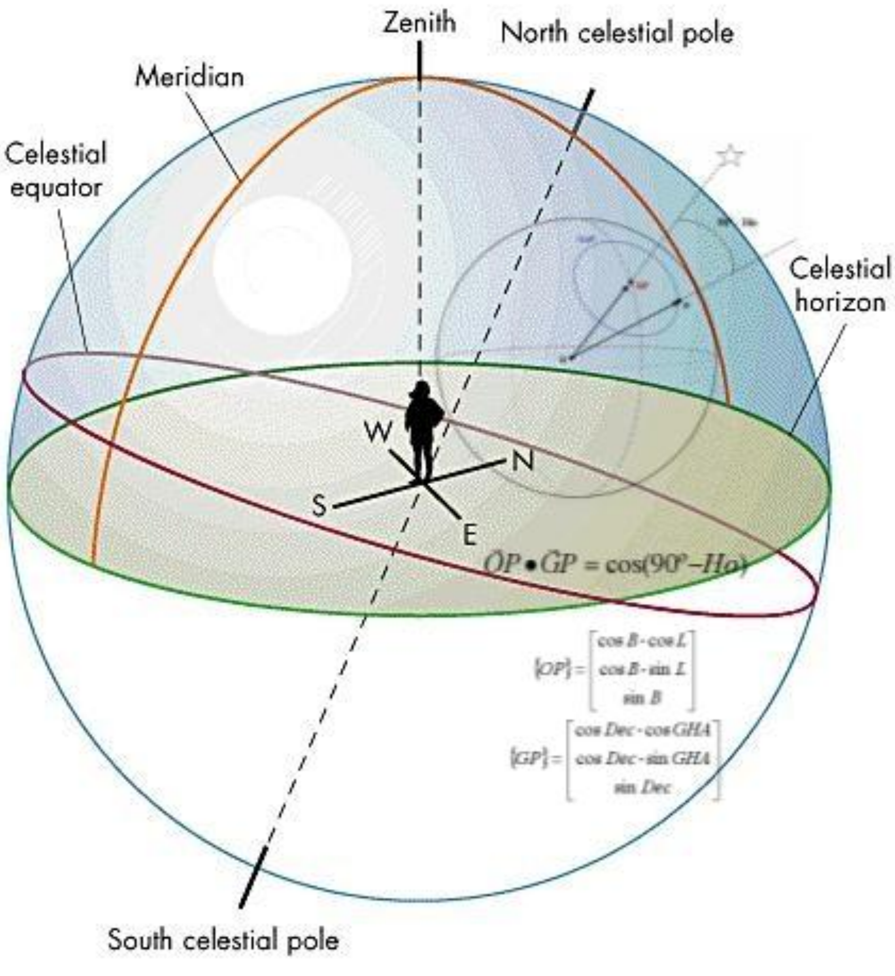
NAVIGATIONAL ALGORITHMS
Vector equation of the circle of equal altitude



© Andrés Ruiz
San Sebastián – Donostia
43º 19’N 002ºW
http://sites.google.com/site/navigationalalgorithms/

Index

Abstract

This paper describes the vector equation of the circle of equal altitude, and defines the traditional problem in celestial navigation: *obtain the calculated altitude and azimuth of an observed body*, using the vector analysis instead the spherical trigonometry.

Also the formulation of the lunar distance, the star-star distance, and the great circle equation and distance is obtained using the vector calculus.

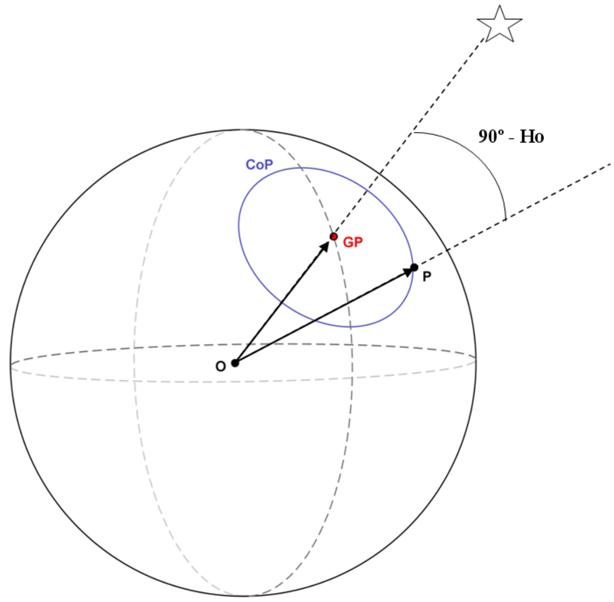
Navigational Algorithms
San Sebastián – Donostia
43º 19'N 002ºW
Current version: 200809

The variables used are:

- GHA – Greenwich Hour Angle
- Dec – Declination
- B, L – Latitude and longitude
- Ho – Celestial body's observed altitude
- Hc – calculated altitude
- Z – azimuth

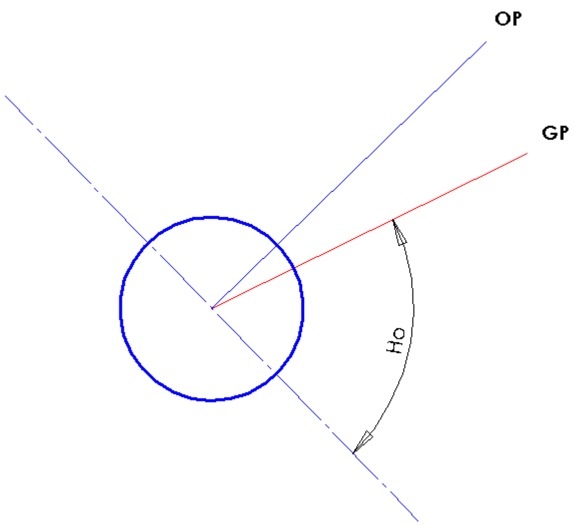
For this formulation the intervals are:

-90 (S) <= Dec <= +90º (N)
0 <= GHA <= 360º (W to E)
0 <= H <= 90º
-90 (S) <= B <= +90º (N)
0 <= L <= 360º (W to E)



Vector equation of the circle of equal altitude

Let OP be the observer's position at the time of sight, and GP the geographical position of the celestial body at the same instant. The dot product of the vectors defined by the centre of the Earth and these points is the cosine of the angle between them, that is the zenith distance of the observed body.



Then, the vector equation of the circle of equal altitude is:

$$\vec{O}P \bullet \vec{G}P = \cos(90^{o} - Ho)$$

Where the two vectors in Cartesian coordinates are:

$$\{OP\} = \begin{bmatrix} \cos B \cdot \cos L \\ \cos B \cdot \sin L \\ \sin B \end{bmatrix}$$

$$\{GP\} = \begin{bmatrix} \cos Dec \cdot \cos GHA \\ \cos Dec \cdot \sin GHA \\ \sin Dec \end{bmatrix}$$

Working the vector equation, the well-known trigonometric equation for the altitude is obtained:

cos B cos L cos Dec cos GHA +

cos B sin L cos Dec sin GHA +

sin B sin Dec = cos(90-H) = sin H

and

cos B cos Dec cos (GHA-L) +

sin B sin Dec = sin H

Altitude and azimuth from assumed position and geographic position

The vector AP of the assumed position, (Be, Le), in Cartesian coordinates is:

$$\{AP\} = \begin{bmatrix} \cos Be \cdot \cos Le \\ \cos Be \cdot \sin Le \\ \sin Be \end{bmatrix}$$

The dot product of AP vector and GP vector gives the calculated altitude Hc:

$$\vec{AP} \bullet \vec{GP} = \cos(90^{o} - Hc)$$

For the azimuth the calculus is as follows.

Let Np be the vector represents the North pole:

$$\{Np\} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

The cross product of two vectors, is a new vector perpendicular to the plane defined by these two vectors, we have:

$$\cos Z = \left[\frac{\vec{AP} \wedge \vec{NP}}{\left\| \vec{AP} \wedge \vec{NP} \right\|} \wedge \vec{AP} \right] \bullet \left[\frac{\vec{AP} \wedge \vec{GP}}{\left\| \vec{AP} \wedge \vec{GP} \right\|} \wedge \vec{AP} \right]$$

if($\left[\frac{\vec{AP} \wedge \vec{NP}}{\left\| \vec{AP} \wedge \vec{NP} \right\|} \right] \bullet \left[\frac{\vec{AP} \wedge \vec{GP}}{\left\| \vec{AP} \wedge \vec{GP} \right\|} \wedge \vec{AP} \right] < 0$)

Z = 360 - Z

Lunar distance

Another example suitable to rewrite in vector notation is the Lunar Distance, LD, equation.

Using geocentric equatorial coordinates, and transforming them to Cartesian ones, we have:

$$\{V\} = \begin{bmatrix} \cos Dec \cdot \cos GHA \\ \cos Dec \cdot \sin GHA \\ \sin Dec \end{bmatrix}$$

And for the Moon and other body the equation becomes:

$$\vec{V}_M \bullet \vec{V}_B = \cos(LD)$$

Expanding the vector equation results:

$$\cos Dec_M \cos Dec_B \cos (GHA_M - GHA_B) +$$
$$\sin Dec_M \sin Dec_B = \cos LD$$

Star - Star distance

A generalized case of the true Lunar Distance is the star – star angular distance.

Let DSS be the distance between the two stars, -or two points located in the celestial sphere-, and V_{s1} V_{s2} the vectors of the two stars in Cartesian coordinates in function of the geocentric equatorial ones.

$$\{V\} = \begin{bmatrix} \cos Dec \cdot \cos GHA \\ \cos Dec \cdot \sin GHA \\ \sin Dec \end{bmatrix}$$

The vector equation is:

$$\vec{V}_{s1} \bullet \vec{V}_{s2} = \cos(DSS)$$

Expanding it results:

$$\cos DSS = \sin Dec_1 \cdot \sin Dec_2 + \cos Dec_1 \cdot \cos Dec_2 \cdot \cos(GHA_2 - GHA_1)$$

If Ra is the right ascension and SHA is the sidereal hour angle, note that:

$$\cos \Delta GHA = \cos \Delta SHA = \cos \Delta RA$$

Great Circle Sailing

Let (Bj, Lj) be a waypoint of the great circle, the associated Cartesian vector is:

$$\{X\} = \begin{bmatrix} \cos B_j \cdot \cos L_j \\ \cos B_j \cdot \sin L_j \\ \sin B_j \end{bmatrix}$$

And let (B_1, L_1) & (B_2, L_2) be the initial and final points of the GC route, their vectors are:

$$\{V_1\} = \begin{bmatrix} \cos B_1 \cdot \cos L_1 \\ \cos B_1 \cdot \sin L_1 \\ \sin B_1 \end{bmatrix} \quad \{V_2\} = \begin{bmatrix} \cos B_2 \cdot \cos L_2 \\ \cos B_2 \cdot \sin L_2 \\ \sin B_2 \end{bmatrix}$$

The vector equation of the great circle is:

$$X \bullet (\vec{V}_1 \wedge \vec{V}_2) = 0$$

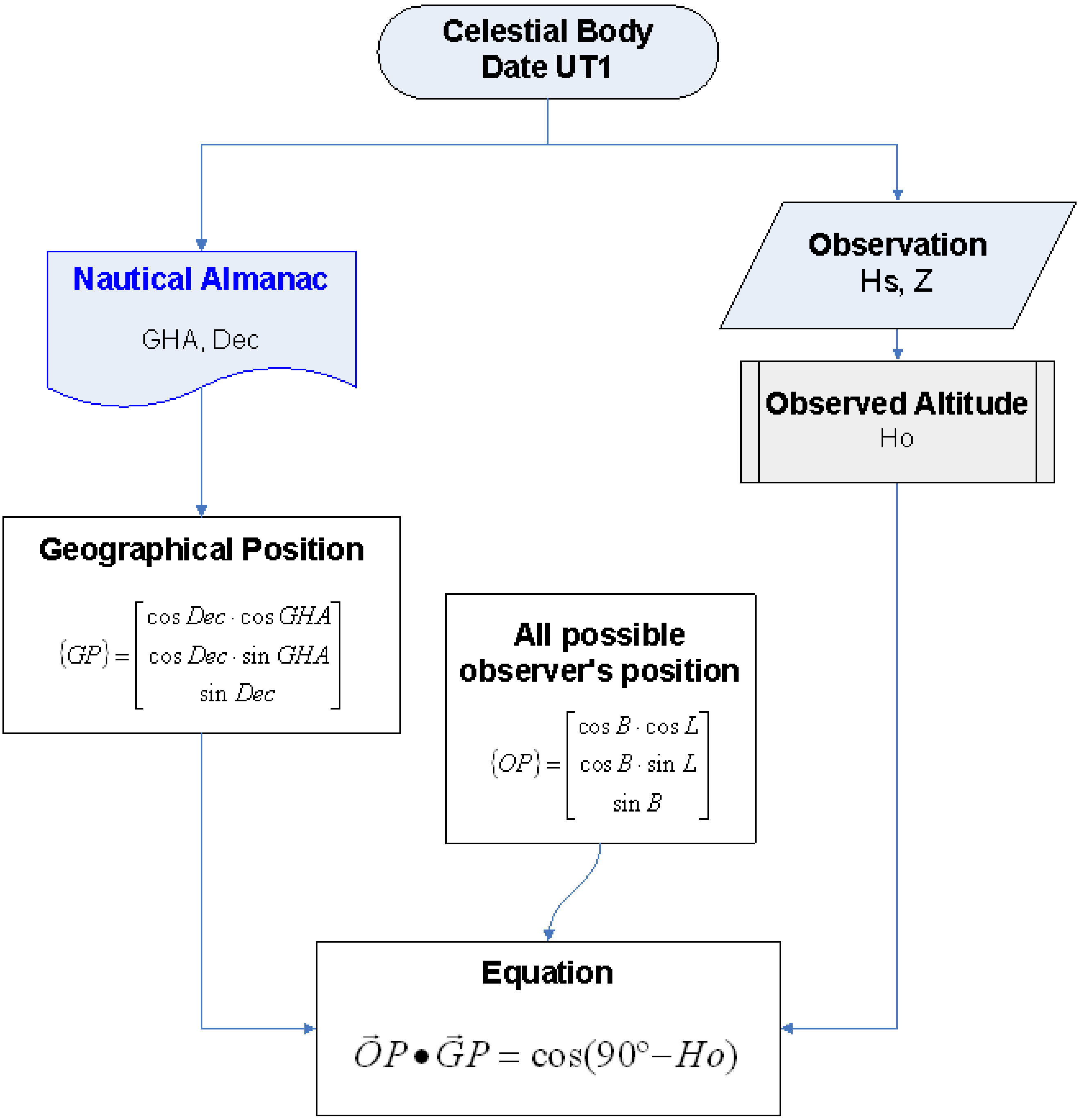
This equation must be solved in order to obtain the latitude of a waypoint in the GC in function of the longitude, (*see B_GC in the source code appendix*)

The vector equation of the GC distance between the point 1 and 2 is:

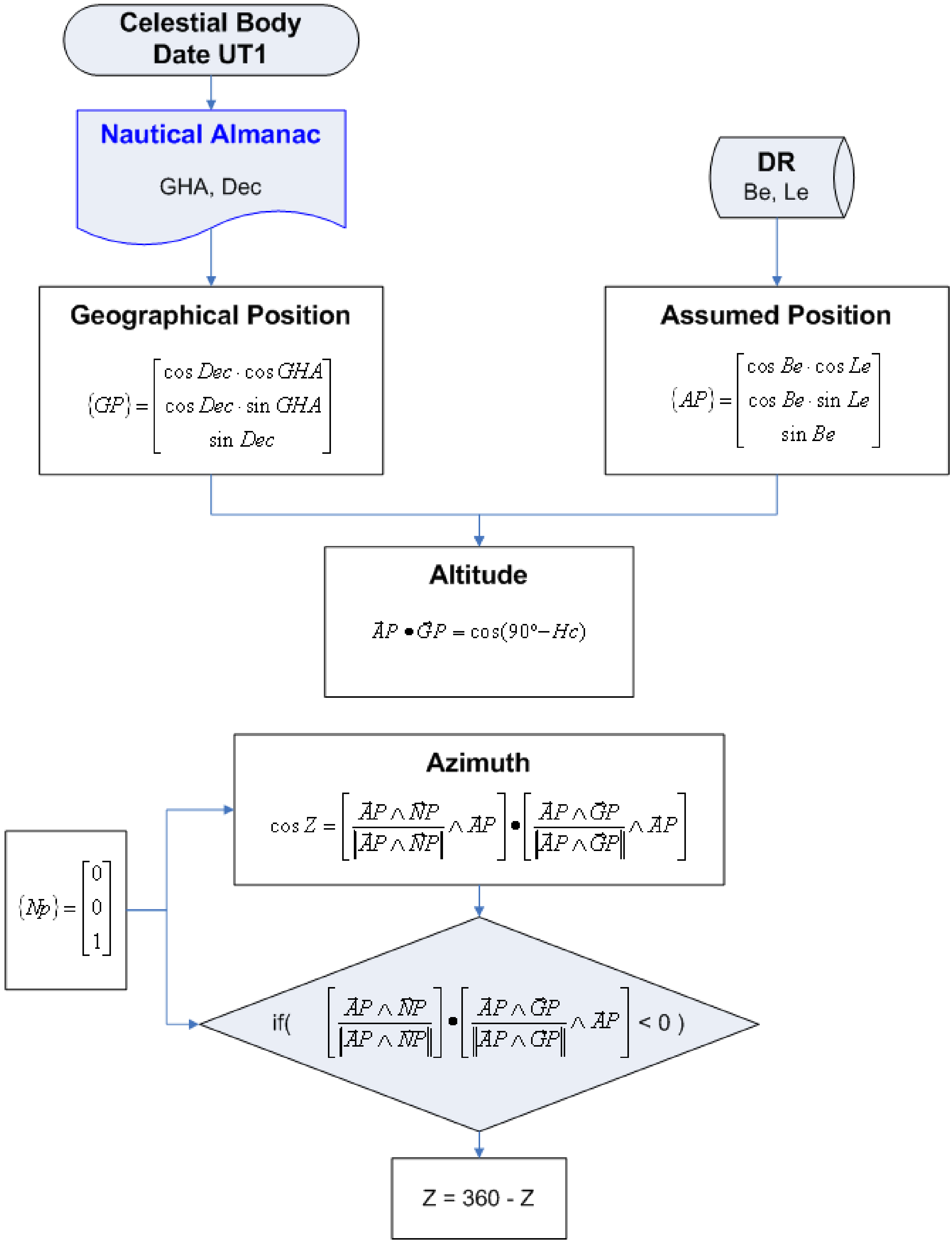
$$D = \cos^{-1}(\vec{V}_1 \bullet \vec{V}_2)$$

A1. Algorithm

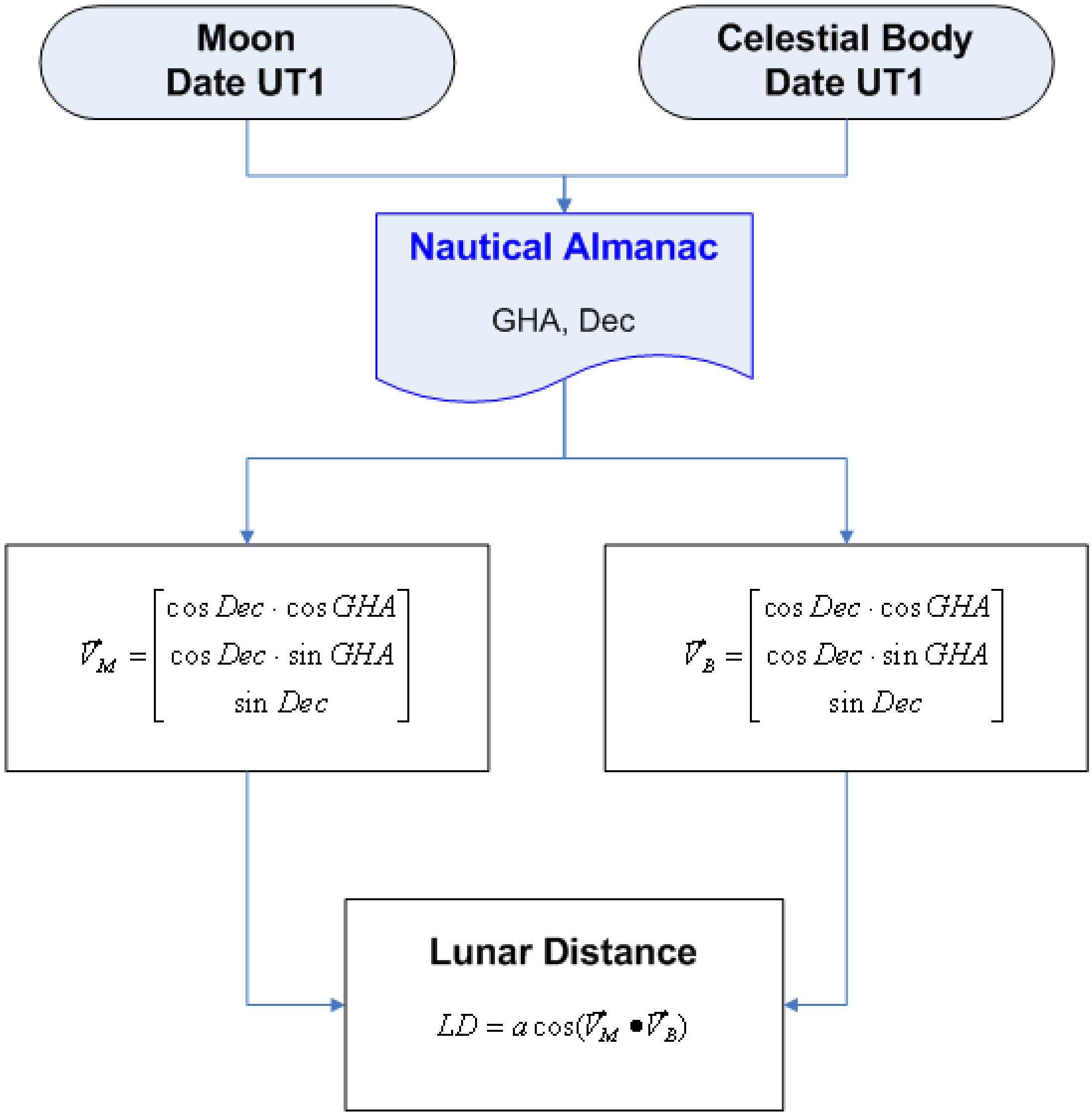
Vectorial equation of the circle of equal altitude



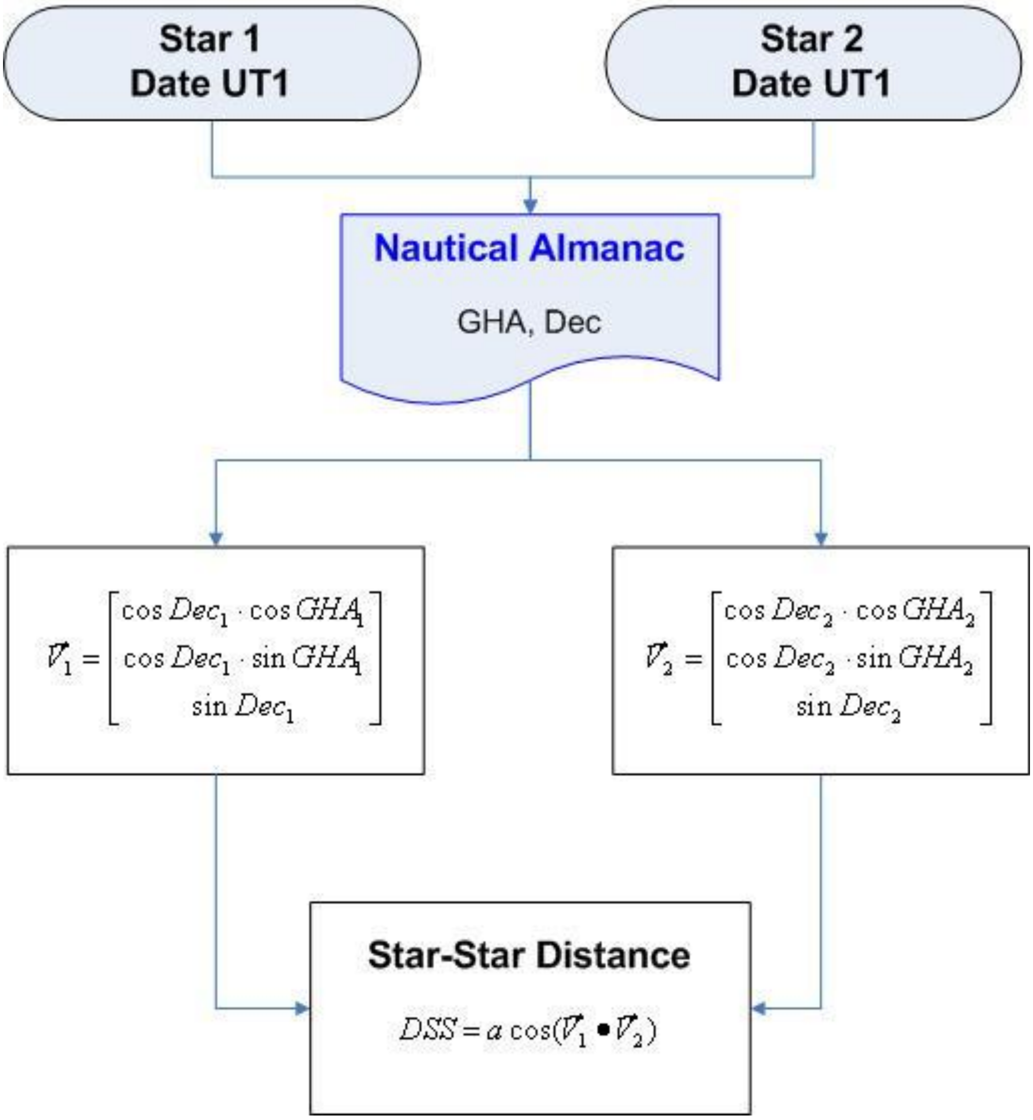
Vectorial equation of the Altitude and azimuth from assumed position and geographic position



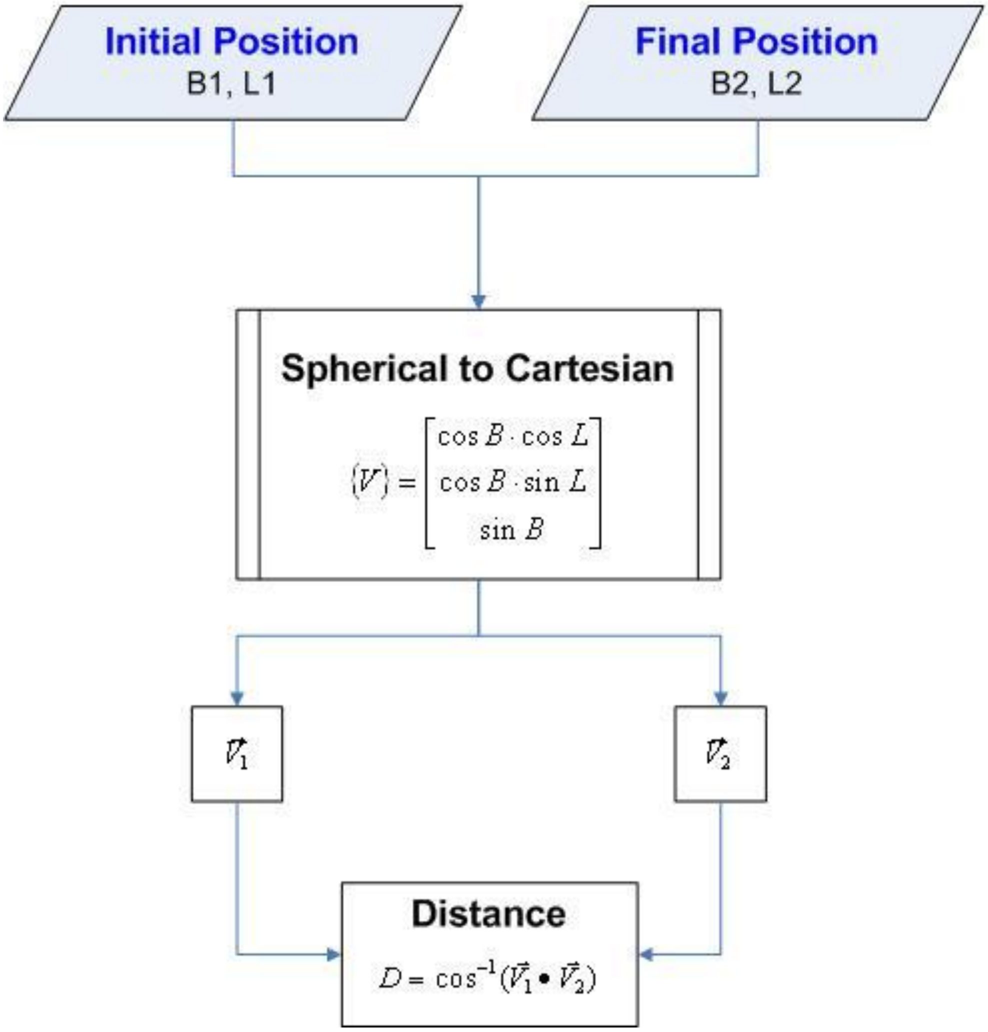
Lunar Distance



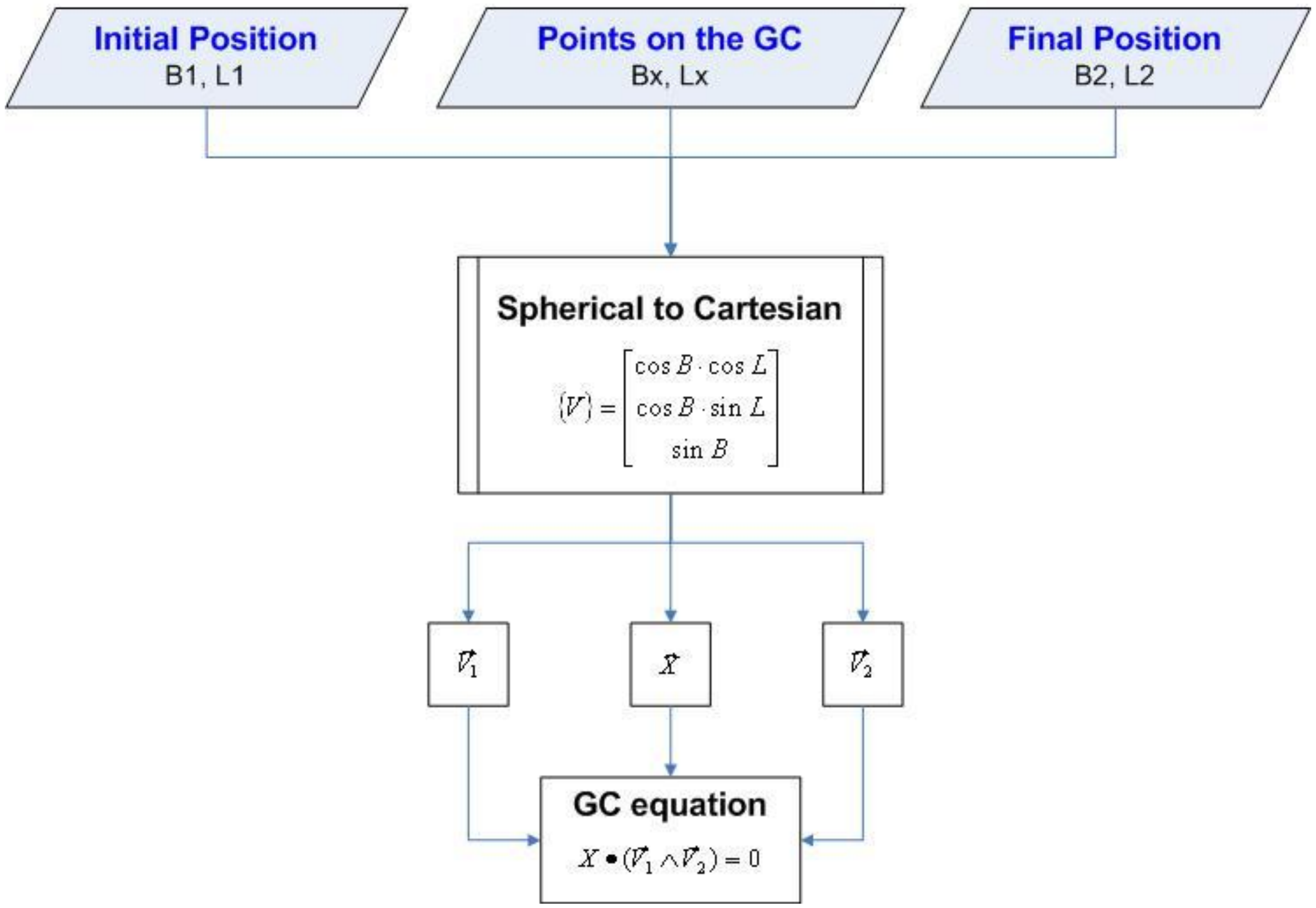
Star-Star Distance



Great Circle Distance vector equation



Great Circle vector equation



A2. Examples

Altitude and Azimuth from AP and GP:

GHA	dec	Be	Le	Hc	Z
347.78	-16.72	40	28	22.041	138.2
334.23	5.22	40	28	30.621	111.0
20.06	16.52	40	28	65.527	161.4
332.71	28.02	40	28	43.396	87.1

A3. Software

Not available.

A4. Source code

```
/*
      File: vector.cpp
      Cálculo vectorial
      Resultado test: OK
      This file contains proprietary information of Andrés Ruiz Gonzalez
      Andrés Ruiz. San Sebastian - Donostia. Gipuzkoa
      Copyright (c) 1999 - 2007
*/

#include <math.h>


double Mod( double *x )
{
    return( sqrt(x[0]*x[0]+x[1]*x[1]+x[2]*x[2]) );
}


double* Add( double *x, double *y )
{
    double *z = new double[3];

    z[0] = x[0]+y[0];
    z[1] = x[1]+y[1];
    z[2] = x[2]+y[2];

    return( z );
}


double* aVector( double a, double *x )
{
    double *z = new double[3];

    z[0] = a*x[0];
    z[1] = a*x[1];
    z[2] = a*x[2];

    return( z );
}


double* Unit( double *x )
{
    return( aVector( 1.0/Mod(x), x ) );
}


double Dot( double *x, double *y )
{
    return( x[0]*y[0]+x[1]*y[1]+x[2]*y[2] );
}
```

```
double* Cross( double *x, double *y )
{
    double *z = new double[3];

    z[0] = x[1]*y[2]-x[2]*y[1];
    z[1] = x[2]*y[0]-x[0]*y[2];
    z[2] = x[0]*y[1]-x[1]*y[0];

    return( z );
}



double* VectorSpherical2Cartesian( double B, double L )
{
    double *v = new double[3];
    // Es un vector unitario
    v[0] = COS( B )*COS( L );
    v[1] = COS( B )*SIN( L );
    v[2] = SIN( B );

    return( v );
}


void AltitudeAzimuth( double GHA, double dec, double Be, double Le, double* Hc, double* Z )
{
    double *GP; // Geographic Position of the celestial body
    double *AP; // Assumed Position
    double Np[3] = {0, 0, 1}; // North pole

    double *U1, *U2, *U3, *Az;

    GP = VectorSpherical2Cartesian( dec, GHA );
    AP = VectorSpherical2Cartesian( Be, Le );


    *Hc = 90.0 - ACOS( Dot( AP, GP ) );


    U1 = Unit( Cross( AP, Np ) );
    U2 = Cross( U1, AP );
    U3 = Unit( Cross( AP, GP ) );
    Az = Cross( U3, AP );

    *Z = ACOS( Dot( U2, Az ) );

    if( Dot( U1, Az ) < 0.0 )         *Z = 360.0 - *Z;

    delete[] GP;
    delete[] AP;
    delete[] U1;
    delete[] U2;
    delete[] U3;
    delete[] Az;
}


double LunarDistance( double decM, double ghaM, double decB, double ghaB )
{
    double *Vm, *Vb;
    double LD;

    Vm = VectorSpherical2Cartesian( decM, ghaM );
    Vb = VectorSpherical2Cartesian( decB, ghaB );

    LD = ACOS( Dot( Vm, Vb ) );

    delete[] Vm;
    delete[] Vb;

    return( LD );
}
```

```
inline double Star2StarDistance( double dec1, double gha1, double dec2, double gha2 )
{
       return( LunarDistance( dec1, gha1, dec2, gha2 ) );
}


inline double GC_Distance( double B1, double L1, double B2, double L2 )
{
       return( 60.0* LunarDistance( dec1, gha1, dec2, gha2 ) ); // nautical miles
}


// GC Waypoints (Bx, Lx)

double B_GC( double B1, double L1, double B2, double L2, double Lx )
{
 double Bx;

 double *V1 ,*V2;
 double *V1xV2;

 V1 = VectorSpherical2Cartesian( B1, L1 );
 V2 = VectorSpherical2Cartesian( B2, L2 );

 V1xV2 = Cross( V1, V2 );

 Bx = -ATAN( V1xV2[0]/V1xV2[2]*COS( Lx ) + V1xV2[1]/V1xV2[2]*SIN( Lx ) );

 delete[] V1;
 delete[] V2;
 delete[] V1xV2;

 return( Bx );
}
```

A5. References

- Sight Reduction with Matrices. Watkins. R. and Janiczek. P. M. NAVIGATION, Journal of The Institute of Navigation, Vol. 25, No. 4, Winter 1978-79, pp. 447-48
- A Novel Approach to Great Circle Sailings: The Great Circle Equation. Chih-Li Chen, Tien-Pen Hsu and Jiang-Ren Chang. Journal of Navigation, Volume 57, Issue 02, May 2004, pp 311-320
- Vector Solutions for Great Circle Navigation. Michael A. Earle. Journal of Navigation, Volume 58, Issue 03, September 2005, pp 451-457
- Spherical Trigonometry http://mathworld.wolfram.com/SphericalTrigonometry.html
- Vector analysis http://en.wikipedia.org/wiki/Vector_%28spatial%29
- Dot product http://en.wikipedia.org/wiki/Dot_product
- Cross product http://en.wikipedia.org/wiki/Cross_product