On Loran-C Time-Difference to Co-ordinate Converters

Dr. Paul Williams and Prof. David Last
University of Wales, Bangor, UK

Abstract

There is no doubt that the future of Loran-C is as a component of high accuracy, high integrity, navigation and positioning systems, in which it is integrated with GNSS (Global Navigation Satellite Systems). In this role, Loran will operate in a Time-of-Arrival (TOA) mode, with users' navigation receivers measuring pseudoranges from Loran-C stations. But Loran was originally a *hyperbolic* system. It worked in time difference (TD) mode and, for most legacy users, it still does. Loran still needs to cater for these current TD users, and also for past users many of whom maintain lists of significant waypoints in TD format. Some vendors of electronic maps and charts now wish to convert legacy Loran TDs to latitude/longitude format.

Loran-C co-ordinate converters come in two distinct varieties: iterative and non-iterative. The more common iterative converters refine an estimate of the user's position, converging to a solution after a number of iterations. The method is computationally intensive, may not converge quickly (or at all), and takes a variable time to find a solution. Further, the user must provide an initial position estimate. Non-iterative methods avoid all these drawbacks, producing a solution every time, in a fixed time.

An attractive, non-iterative technique was proposed by Razin in 1967. We have further developed and improved Razin's method by working in a geocentric Cartesian coordinate system compatible with GNSS (so eliminating the need to compute and store 14 constants for each Loran-C chain required by Razin), and incorporating Additional Secondary Factors (ASF) into the algorithm.

The paper begins, as did Razin, by presenting the equations for Loran-C positioning on a spherical earth, assuming a constant propagation velocity. We then deviate by taking ASFs into account. We present the computations for Razin's "osculating sphere" and the associated co-ordinate conversions into geocentric Cartesian format. Finally, we show how to compute the final geocentric Cartesian position by means of a simple vector dot product. We provide sufficient detail in this paper to allow a proficient programmer to implement the algorithm in almost any programming language.

1 Introduction

Loran-C Time Difference to geographical location converters come in two distinct varieties: *iterative* and *non-iterative*. Non-iterative converters offer a number of advantages over iterative ones. Firstly, the iterative method is highly computational, requiring several iterations around a loop. Secondly, an iterative solution requires the user to enter an initial position estimate, usually manually. Each iteration through the algorithm then refines this estimate, converging to the final position solution. The number of iterations required can vary quite widely and, in some cases, the algorithm may simply fail to converge to the correct solution.

A non-iterative method is greatly preferable to an iterative one. An important non-iterative technique was proposed by Razin [1], and further developed by Fell [2].

In Razin's technique, the propagation of Loran-C signals is first assumed to take place at constant velocity, the earth being represented by a simple "osculating" sphere rather than a more complex, and more accurate, ellipsoid itself. This sphere touches (i.e. "osculates", or "kisses") the earth at a point approximately central to the coverage of the Loran chain. Co-ordinate transformations are used to convert the positions of the Loran-C transmitters into the co-ordinate system of this sphere on which the position solution is performed. Once the position has been found on the sphere, it is converted back to its equivalent location on the ellipsoid. Spatial variations in the velocity of propagation are catered for by inclusion of secondary factor corrections in the position solution equations.

In this paper we further develop Razin's technique by:

1. Eliminating the need to compute and store the 14 constants Razin requires for each triad of Loran-C stations.

2. Demonstrating how to construct the osculating sphere and how to convert from the sphere to WGS84, and vice versa, using Cartesian coordinates. (Razin does not explain how to construct the osculating sphere in his original paper).

3. Illustrating how to incorporate Additional Secondary Factor (ASF) corrections into the algorithm.

We begin, as Razin does, by presenting the equations for computing a Loran-C position on a spherical earth, assuming constant propagation velocity. We then introduce ASFs. We compute the parameters of the osculating sphere and show how to convert coordinates into the geodetic Cartesian system in which we will compute the final position by the application of a mathematical dot product.

2 Loran-C Hyperbolic Positioning

First, the theory of Loran-C positioning is presented briefly. We make the following assumptions, which apply to both hyperbolic and circular modes of operation:

1. The receiver, and all transmitters, are located on the surface of the earth. This assumption is acceptable since Loran-C provides no information on the height of the receiver.

2. The earth is approximated by the WGS-84 reference ellipsoid. The use of this globally-accepted standard preserves compatibility with the Global Positioning System (GPS).

3. Transmission between two points on the earth's surface is via the Great Circle path between them.

4. A Loran-C signal takes the following time to propagate from point i to point j:

$$T_{ij} = \frac{\overline{ij}}{(v_0 / \eta)} + T_{SFij} + T_{ASFij} \,, \tag{2.1}$$

where:

$\overline{ij}$ = Length of the transmission path between i and j in metres,
v_0 = Speed of light in a vacuum (299,792,458 ms^{-1}),
η = Index of refraction of the atmosphere at the surface of the earth (the United States Coast Guard specify a value of 1.000338),
T_{SFij} = Secondary phase correction of path $\overline{ij}$, and
T_{ASFij} = ASF correction of path $\overline{ij}$.

Thus, in Fig. 1:

$$\overline{XS} - \overline{MS} = (T_x - T_{ex} - T_{SFxs} + T_{SFms} - T_{ASFxs} + T_{ASFms})v \tag{2.2}$$

$$\overline{YS} - \overline{MS} = (T_y - T_{ey} - T_{SFys} + T_{SFms} - T_{ASFys} + T_{ASFms})v \tag{2.3}$$

where:

$\overline{XS}$, $\overline{MS}$ and $\overline{YS}$ are distances in metres across the surface of the ellipsoid, between receiver S, and master station M and secondary stations X and Y, respectively,

T_i = Time difference between the arrival of the signal from station i and the master station,

T_{ei} = i-Station's Emission Delay ($i = x, y$)

$v = \frac{v_0}{\eta}$ = Primary factor velocity; the velocity in air at the earth's surface.

The problem is to find the position **S** of the user in geographical coordinates, latitude φ_s and longitude, λ_s.

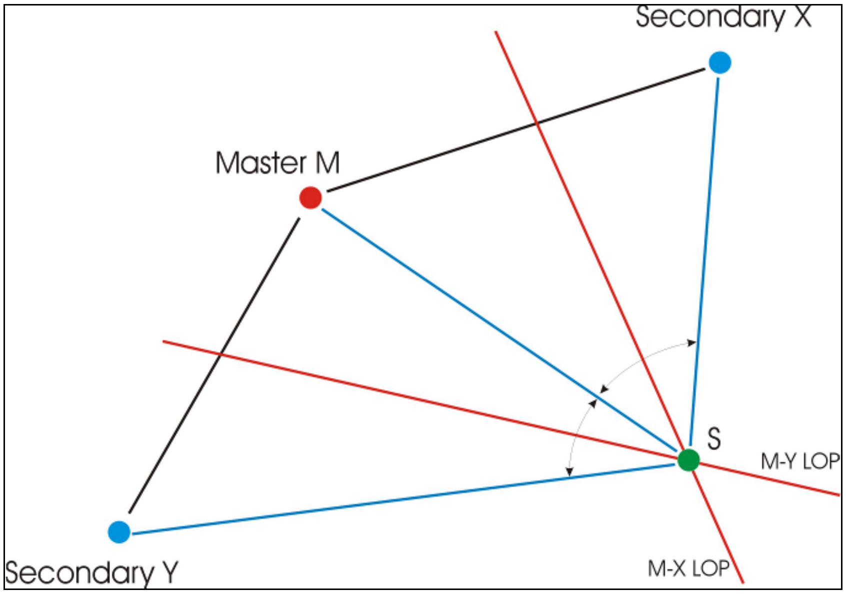

Fig. 1 – Position of user, S, determined by measuring time differences between signals received from a master and two secondary stations.

We could express equations 2.2 and 2.3 in terms of the position of the user, **S**, (φ_S, λ_S). However, this would be difficult because computation of distance on the WGS84 ellipsoid is complicated and non-linear. Also, the propagation velocity is not everywhere the same, but rather is a non-linear function of distance from the transmitter. For these reasons, the conventional approach to solving this problem is to use an iterative technique [3].

3 Solving the Loran-C Positioning Problem

The key steps in Razin's non-iterative method are as follows:

1. Convert the Loran-C transmitter locations from WGS-84 positions to positions on the osculating sphere that is tangential at a point to the WGS-84 ellipsoid.

2. Making the assumption that signals propagate at the speed of light through air (the “Primary Factor”) across the surface of this sphere, solve the propagation equations to give the position of the receiver.

3. Apply corrections for Secondary Factor (SF) and ASF delays.

4. Convert the resulting receiver co-ordinates to coordinates on the WGS-84 ellipsoid.

We will now describe how each of the steps outlined above is carried out. Sections 3.1 and 3.2 introduce the coordinate conversions required, and demonstrate the construction of the osculating sphere. Section 3.3 shows the development of the propagation equations, and shows how we include the effects of the SF and ASF.

3.1 Step 1: Converting co-ordinates from WGS-84 to a sphere

We convert all coordinates of interest from the WGS-84 (Datum 1) to the equivalent coordinates on the sphere (Datum 2). Throughout, the coordinates will be expressed in a geocentric Cartesian coordinate system. This system offers the advantage of being completely defined by the directions of three axes and the position of the origin, with none of the complications of the reference ellipsoid or a projection grid.

Since the sphere is itself a special case of an ellipsoid, the datum transformation is relatively simple. Fig. 2 shows the principle employed. The black ellipse represents the WGS-84 ellipsoid. The red circle is the osculating sphere; we imagine the sphere’s being free to roll around on the inside surface of the hollow ellipsoid, the two touching at the single tangent point which lies close to the centre of the coverage area of the Loran-C chain. For clarity in Fig. 2, the radius of the sphere is shown as much smaller than that of the ellipsoid; in practice, it would be the same as the total radius of curvature of the WGS-84 ellipsoid at the tangent point.

In general, datum transformations involve translation of the origin of the co-ordinate system and rotation of the co-ordinate axes. In this case, we choose to make the coordinate axes of the sphere parallel to those of the WGS-84 ellipsoid so that only translation, by an amount equal to the offset between their centres, is required. This offset vector $\left[\Delta X \Delta Y \Delta Z\right]^T$ is represented by the green line $\bar{C}_{wgs84} - \bar{C}_{Sphere}$ in Fig. 2.

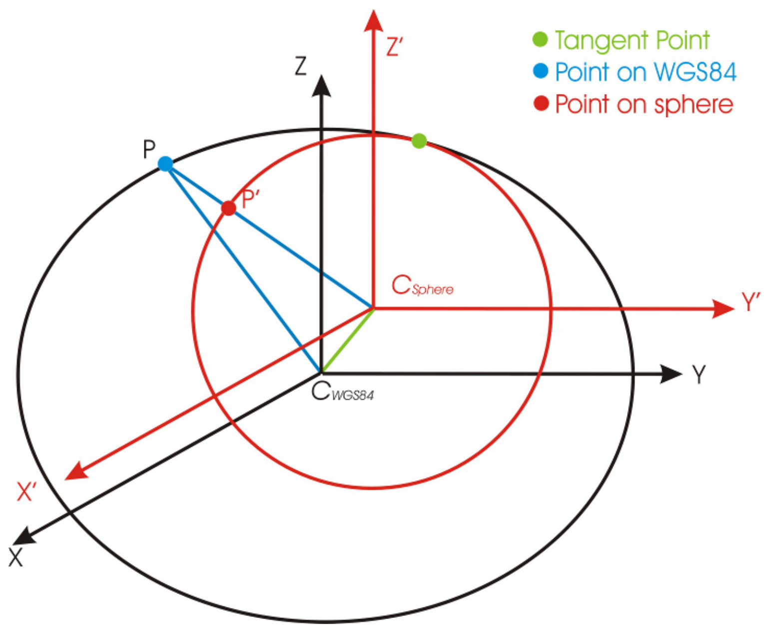

Fig. 2 – The WGS-84 ellipsoid and the osculating sphere.

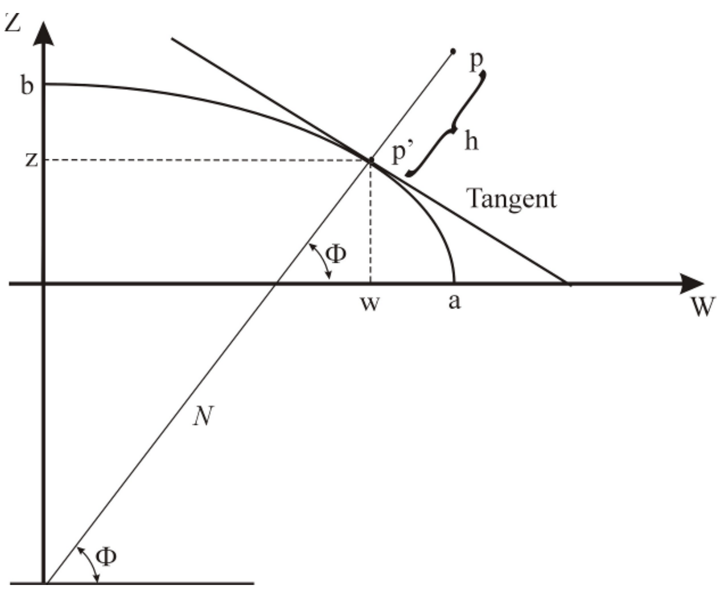

Fig. 3 – Plane section of an ellipse, after [4].

3.1.1 General Coordinate Conversion

We first of all review how to convert from the coordinate system of geodetic latitude and longitude to the geocentric Cartesian coordinate system, then we convert from one datum to another.

Fig. 3 illustrates an example plane section through an ellipse defined by the semi-minor and semi-major axes, *b* and *a*. Referring to this figure, Appendix 1 of [4] shows that:

$$w = \frac{a\,Cos(\Phi)}{\sqrt{1-e^2 Sin^2(\Phi)}} \tag{3.1}$$

$$Z = \frac{a(1-e^2)Sin(\Phi)}{\sqrt{1-e^2 Sin^2(\Phi)}}, \tag{3.2}$$

where $e = \sqrt{1-\frac{b^2}{a^2}}$ is the eccentricity of the ellipse and Φ is the geodetic latitude of the point *P* on the ellipsoid.

We now perform the transformation from the ellipsoid, Datum 1, to the sphere, Datum 2. Assume that a position P_1 in Datum 1 is given by coordinates (Φ_1, Λ_1, h_1), which represent latitude, longitude and height, respectively. So, including the height of the point h_1 we can calculate:

$$w_1 = w + h_1 Cos(\Phi_1), \tag{3.3}$$

$$Z_1 = Z + h_1 Sin(\Phi_1). \tag{3.4}$$

From Equations 3.1 to 3.4, we see that the geocentric Cartesian coordinates of P_1 on the ellipsoid is given by:

$$X_1 = w_1 Cos(\Lambda_1) = \left(\frac{a_1}{\sqrt{1-e_1^2 Sin^2(\Phi_1)}} + h_1 \right) Cos(\Phi_1) Cos(\Lambda_1) \tag{3.5}$$

$$Y_1 = w_1 Sin(\Lambda_1) = \left(\frac{a_1}{\sqrt{1-e_1^2 Sin^2(\Phi_1)}} + h_1 \right) Cos(\Phi_1) Sin(\Lambda_1) \tag{3.6}$$

$$Z_1 = \left(\frac{a_1(1-e_1^2)}{\sqrt{1-e_1^2 Sin^2(\Phi_1)}} + h_1 \right) Sin(\Phi_1). \tag{3.7}$$

Datum 2 differs from Datum 1 by a displacement of the centre, and by the eccentricity and semi-axes of the ellipse. In our special case, the eccentricity of the sphere (Datum 2) is 0; the sphere's radius will be computed later.

We next determine the equations to compute the displacement. The coordinate transformation, from Datum 1 to Datum 2, is achieved by using the simplified Helmert transformation of Equation 3.8:

$$\left.\begin{aligned} X_2 &= X_1 + \Delta X \\ Y_2 &= Y_1 + \Delta Y \\ Z_2 &= Z_1 + \Delta Z \end{aligned}\right\}, \tag{3.8}$$

from which we can calculate w_2:

$$w_2 = \sqrt{Y_2^2 + Y_2^2}\ . \tag{3.9}$$

We can now convert back to geodetic latitude and longitude if we so wish. Longitude is given by:

$$\Lambda_2 = Tan^{-1}\left(\frac{Y_2}{X_2}\right). \tag{3.10}$$

For computing the geodetic latitude in Datum 2, from Equations 3.1 to 3.4 we get the equivalent equations:

$$w_2 = \left(\frac{a_2}{\sqrt{1 - e_2^2 Sin^2(\Phi_2)}} + h_2\right) Cos(\Phi_2) \tag{3.11}$$

$$Z_2 = \left(\frac{a_2(1 - e_2^2)}{\sqrt{1 - e_2^2 Sin^2(\Phi_2)}} + h_2\right) Sin(\Phi_2) \tag{3.12}$$

and noting that, for the special case of a sphere, $e_2^2 = 0$, we divide (3.12) by (3.11) and so obtain the latitude in Datum 2:

$$\Phi_2 = Tan^{-1}\left(\frac{Z_2}{w_2}\right). \tag{3.13}$$

So once we know the offsets in Equation 3.8, we can use these transformations to convert back and forth between WGS-84 and the sphere. We can stay in the geocentric Cartesian coordinate system, or use Equations 3.9 to 3.13 to convert back to latitude and longitude.

We now need to determine the offsets in Equation 3.8; this is shown in Sub-section 3.1.2, where we illustrate how to construct the osculating sphere.

3.1.2 The Osculating Sphere

We can now use the transformation derived in Sub-section 3.1.1 to determine the parameters of the osculating sphere that best fits the ellipsoid across the coverage region of the Loran-C chain. We then determine the values of $\Delta X, \Delta Y$ and ΔZ in Equation 3.8.

We start by establishing a common point of reference in the two systems, the "tangent" point at which the sphere touches the ellipsoid. Razin suggests that this should lie in the centre of the service area of the Loran chain. Our software establishes a suitable tangent point by taking as its components the average of the latitudes, Φ_{Mean}, and the average of the longitudes, Λ_{Mean}, of the transmitters. This averaging is actually performed in Cartesian coordinates, to which the Loran-C transmitter coordinates have first been converted from latitude and longitude.

We assume that the coordinates of the tangent point are the same in the two systems, $\left(\Phi_0, \Delta_0\right)$, and we compute $\left(\Phi_0, \Lambda_0\right) = \left(\Phi_{Mean}, \Lambda_{Mean}\right)$. From the Cartesian coordinates then:

$$\Phi_0 = Tan^{-1}\left(\frac{Y_{Mean}}{X_{Mean}}\right) \tag{3.14}$$

$$\Lambda_0 = Tan^{-1}\left(\frac{Z_{Mean}}{w_{Mean}\left(1-e^2\right)}\right). \tag{3.15}$$

Next we make the radius of the sphere equal to the radius of curvature of the WGS-84 ellipsoid at the common point:

$$R = \sqrt{M_0 N_0}, \tag{3.16}$$

in which,

$$M_0 = \frac{a(1-e^2)}{\left(1-e^2 Sin^2(\Phi_0)\right)^{\frac{3}{2}}}, \tag{3.17}$$

and

$$N_0 = \frac{a}{\sqrt{(1-e^2 Sin^2(\Phi_0)}}. \tag{3.18}$$

In these three equations, M_0 and N_0 are the radius of curvature along, and perpendicular to, the meridian passing through (Φ_0, Δ_0), respectively.

We next calculate the offset between the centres of the sphere and the WGS-84 ellipsoid. From 3.8 we know that:

$$\left.\begin{aligned} \Delta X &= X_2 - X_1 \\ \Delta Y &= Y_2 - Y_1 \\ \Delta Z &= Z_2 - Z_1 \end{aligned}\right\} . \tag{3.19}$$

Here (X_1, Y_1, Z_1) is the tangent point in the reference frame of the ellipsoid and (X_2, Y_2, Z_2) is the tangent point in the reference frame of the sphere, with its radius R.

The spherical solution of the equations requires the coordinates of the transmitters to be converted to coordinates on the sphere, using Equation 3.19. Simple rearrangement of this equation gives us the inverse transformation, so letting us convert from Cartesian on the sphere (X_2, Y_2, Z_2), back to Cartesian on WGS-84, (X_1, Y_1, Z_1).

3.1.3 The Effect of Height

Fig. 4 shows the effect of the unknown height of the receiver on the coordinate transformation. The height is unknown since Loran-C gives no height information, whether in stand-alone hyperbolic or circular mode. So we assume that $h_1 = 0$ everywhere. We need not be concerned with the height value h_2 either, since neglecting height in these coordinate conversions has little effect on horizontal position: Ashkenazi, Moore and Hill [5] have shown that a height error, *h*, as gross as 8000m results in a horizontal error ($P' - P''$ in Fig. 4) of just 15cm, in a variety of transformations!

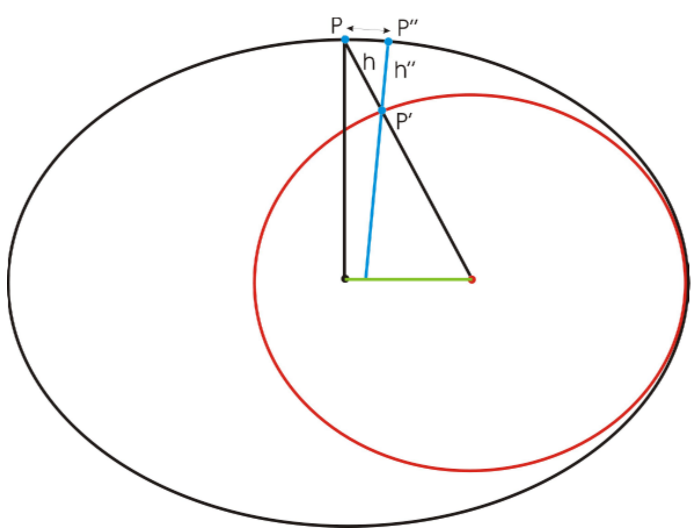

Fig. 4 – The effect of the unknown height on the datum transformation.

3.2 Step 2: The Loran-C Solution on a Sphere

In this sub-section we present Razin's derivation of the propagation equations. As we have seen, he approximates the earth by an osculating sphere of radius R, and assumes that the velocity of propagation is a constant value, v, the primary velocity in air. He then represents Equations 2.2 and 2.3 as:

$$\theta_{is} = \theta_{ms} + P_i \text{, for } i = x, y \tag{3.20}$$

Equation 3.20 embodies the assumption that the distance $\overline{ij}$ of Equation 2.1 is equivalent to an angle θ_{ij} subtended at the center of the sphere of the earth. Referring to Fig. 5, **M**, **X** and **Y** are the master and two secondary stations. The aim is to calculate the angles θ_{xs}, θ_{ys} and θ_{ms}, from which the position of **S** can be computed.

Now,

$$P_i = \frac{v}{R}(T_i - T_{ei}) \text{ for } i = x, y\,. \tag{3.21}$$

If we take cosines of both sides of Equation 3.20 we get:

$$Cos(\theta_{is}) = Cos(\theta_{ms})Cos(P_i) - Sin(\theta_{ms})Sin(P_i)\text{, for } i = x, y\,. \tag{3.22}$$

Using the angles on the surface of the sphere illustrated in Fig. 5, and the law of cosines on a sphere, we get the following:

$$Cos(\theta_{is}) = Cos(\theta_{mi})Cos(\theta_{ms}) + Sin(\theta_{mi})Sin(\theta_{ms})Cos(\beta_i) \text{ for } i = x, y \tag{3.23}$$

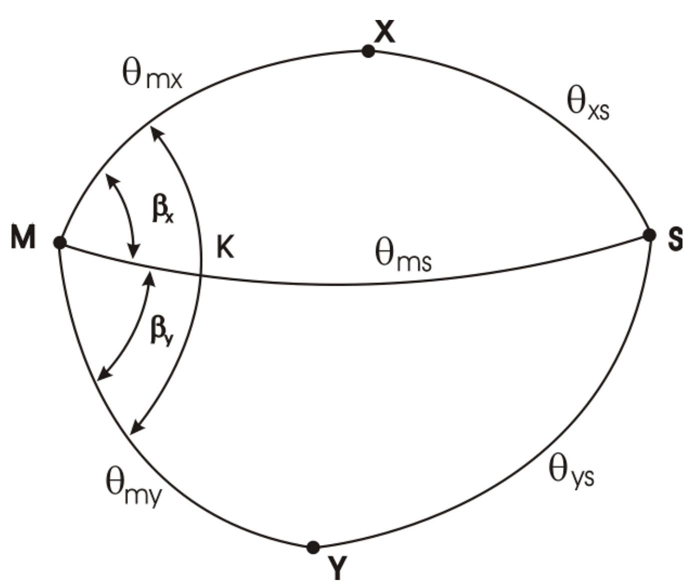

Fig. 5 – Angles on the surface of the osculating sphere

We can now equate Equation 3.22 with Equation 3.23. Rearranging this yields:

$$Tan(\theta_{ms}) = \frac{Cos(P_i) - Cos(\theta_{mi})}{Sin(P_i) + Sin(\theta_{mi})Cos(\beta_i)}, \text{ for } i = x, y. \tag{3.24}$$

Referring to Fig. 5 and using a trigonometric identity we see that:

$$Cos(\beta_y) = Cos(K - \beta_x) = Cos(K)Cos(\beta_x) + Sin(K)Sin(\beta_x) \tag{3.25}$$

Also, Equation 3.24 describes 2 equations, one for each of $i = x, y$. We can equate the right-hand-sides of the two equations to one another, since $Tan(\vartheta_{ms})$ is common. Replacing $Cos(\beta_y)$, in these equations, with the identity in Equation 3.25 allows us to solve Equation 3.24 for $Cos(\beta_x)$:

$$Cos(\beta_x) = \frac{u_3 u_1 \pm u_2\sqrt{u_1^2 + u_2^2 - u_3^2}}{u_1^2 + u_2^2}, \tag{3.26}$$

where,

$$a_i = \frac{Cos(P_i) - Cos(\theta_{mi})}{Sin(\theta_{mi})}, \text{ for } i = x, y$$

$$u_1 = a_x Cos(K) - a_y$$

$$u_2 = a_x Sin(K)$$

$$u_3 = a_y \frac{Sin(P_x)}{Sin(\theta_{mx})} - a_x \frac{Sin(P_y)}{Sin(\theta_{my})}$$

Once the time-differences, T_x and T_y have been calculated, we follow the procedure outlined in the flow-chart of Fig. 6. to calculate the angles θ_{xs}, θ_{ys} and θ_{ms} between each station and the unknown location of the user.

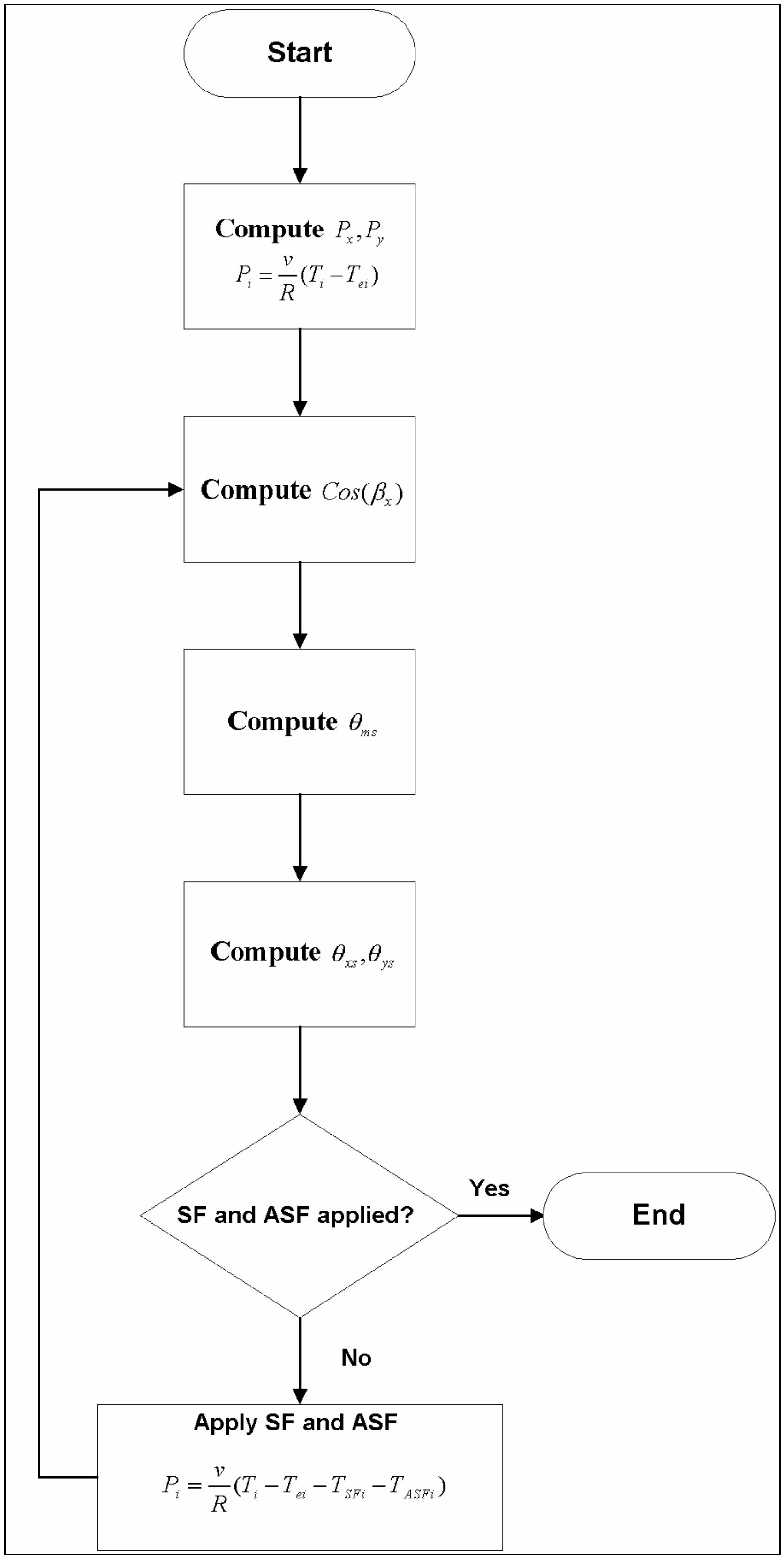

Fig. 6 – Flow-chart of angle computation process.

3.3 Step 3 - Sea-water and ASF Corrections

So far, we have assumed that the velocity of propagation is both constant and equal to the Primary Factor velocity (the velocity of light in the earth's atmosphere at the surface of the earth). However, Loran-C receivers assume in the first instance that all signals propagate over sea-water paths. In order to do the same, we need to apply the sea-water secondary factor which is, however, a function of range from the transmitter.

Also, in order to take the effects of land paths into account we must include the ASF. Since ASFs vary spatially, we first need to know the location of the receiver (at least approximately) in order to establish the appropriate ASFs. The accuracy with which the location needs to be established depends on the resolution of the ASF database. In the algorithm of Fig. 6, we first compute an approximate position without ASFs, then look up the ASFs there, and finally re-compute the position with those ASFs taken into account.

3.4 Step 4 - Solving for the User's Position

Unlike Razin, we propose computing the receiver's geographical position in Cartesian coordinates. We calculate the position by using knowledge of the dot products, $\langle \bar{i} \cdot \bar{s} \rangle$, between the vectors representing the Cartesian coordinates of the Loran-C stations and the user ($\bar{i}$ and $\bar{s}$ respectively), that is, the arrows labeled 'R' in Fig. 7.

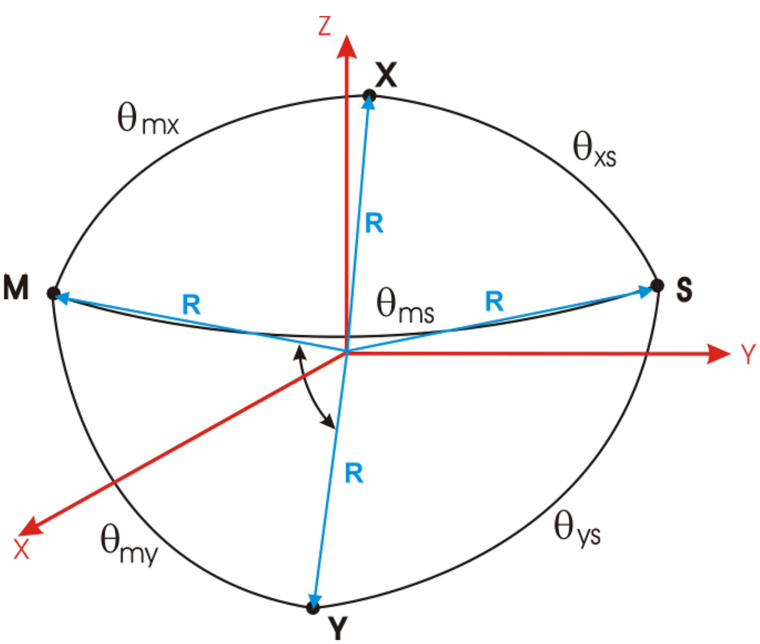

Fig. 7 – The angles on the sphere showing the vectors used in our dot-product solution.

Referring to this figure, Equation 3.27 shows the relationship between the dot product of a pair of vectors, the lengths of the vectors $\bar{i}$ and $\bar{s}$ given by $|i|$ and $|s|$ respectively, and the angle between them, ϑ_{is} :

$$\langle \bar{i} \cdot \bar{s} \rangle = |i||s| Cos(\theta_{is}) \text{ for } (i = m, x, y). \tag{3.27}$$

The magnitude, or length, of each of the vectors $|i|$ and $|s|$ is equal to R, the radius of the sphere.

The location vector, $\bar{s}$, of the receiver with respect to the center of the sphere is given by (X_s, Y_s, Z_s). We create a set of three simultaneous equations using Equation 3.27, one for each Loran-C station in the chosen triad.

Using matrix and vector notation we define the set of equations given by:

$$A\bar{s} = b \tag{3.28}$$

where A is a 3×3 matrix containing the Cartesian coordinates of the three Loran-C stations (**m**, **x** and **y)** of the triad used to establish the fix:

$$A = \begin{bmatrix} X_x & Y_x & Z_x \\ X_y & Y_y & Z_y \\ X_m & Y_m & Z_m \end{bmatrix}, \tag{3.29}$$

and b is defined as:

$$b = \begin{bmatrix} R^2 Cos(\theta_{xs}) \\ R^2 Cos(\theta_{ys}) \\ R^2 Cos(\theta_{ms}) \end{bmatrix}. \tag{3.30}$$

The solution is then:

$$\bar{s} = A^{-1} b. \tag{3.31}$$

Finally, the resulting Cartesian vector, $\bar{s}$, is converted back to the WGS-84 ellipsoid by application of the offsets computed in Equation 3.19, and subsequently converted to geographical coordinates using the methods shown in Section 3.1.

The method explained in Equations 3.27 to 3.31 avoids having to compute and store the 14 separate constants for each Loran-C triad employed by Razin. Razin does not show how to compute these constants in [1].

4 Testing the Implementation

The algorithm described in Section 3 has been implemented as a Microsoft Windows™ program written in Microsoft Visual C++. The performance of the algorithm has been checked by comparing the results given by the program against those of the iterative solution specified by the Radio Technical Commission for Marine Services [3].

First, the data shown in Table 1 was used. The Loran triad was Malone (M), Grangeville (W) and Jupiter (Y) from the Southeast US chain (GRI 7980). The two right-most columns show the differences between the latitudes and longitudes computed by the two methods, expressed in minutes of arc. The root mean squares (RMS) of these differences correspond to 0.03 seconds latitude (approximately 1m), and 0.02 seconds longitude (approximately 0.5m).

Fig. 8 shows a MapInfo™ plot of Loran-C position measurements collected at Bangor, North Wales. The stations Sylt, Lessay and Vaerlandet of the Sylt (GRI 7499) chain were used for this experiment. The dots are positions computed when ASFs were included (blue), and omitted (red). The error ellipse of the plots is elongated because of the geometry of the stations as seen from Bangor, with the Sylt-Vaerlandet pair presenting a higher TD-change to distance-change gradient than the Sylt-Lessay pair.

Fig. 9 repeats the results, with ASFs included, comparing those given by the non-iterative method proposed in this paper (blue dots) with those given by the traditional iterative method (green dots). The discrepancy is on the order of just 2m. It is principally in the form of a shift believed to be due to the non-optimal location of the tangent point.

Name of Area	Loran-C TDs (μs)		Non-iterative Loran-C Position				Iterative Loran-C Position				Difference	
			Latitude		Longitude		Latitude		Longitude		Delta Lat.	Delta Lon.
	M-W	M-Y	Deg.	Min.	Deg.	Min.	Deg.	Min.	Deg.	Min.	Minutes	Minutes
Anchor Chain	14147.7	43205.8	25	8.1843	-80	15.9788	25	8.1838	-80	15.9785	0.0005	0.0003
City of Washington	14149.8	43202.6	25	8.8829	-80	15.3181	25	8.8824	-80	15.3177	0.0005	0.0004
Little Grecian	14142.5	43214.7	25	6.7745	-80	17.9671	25	6.7740	-80	17.9668	0.0005	0.0003
Mike's Wreck	14149.8	43201.7	25	8.5856	-80	14.9771	25	8.5851	-80	14.9768	0.0005	0.0003
North North Dry Docks	14145.5	43211	25	8.0520	-80	17.3512	25	8.0514	-80	17.3508	0.0006	0.0004
South Ledges 1	14149.4	43202	25	8.3502	-80	14.9861	25	8.3497	-80	14.9858	0.0005	0.0003
South Ledges 2	14147	43206.5	25	7.8317	-80	16.0617	25	7.8312	-80	16.0614	0.0005	0.0003
The Fingers	14148.5	43204.7	25	8.4894	-80	15.7719	25	8.4889	-80	15.7716	0.0005	0.0003
The Horseshoe	14145.9	43210.3	25	8.1561	-80	17.1935	25	8.1556	-80	17.1931	0.0005	0.0004
Train Wheel	14147.9	43206	25	8.4160	-80	16.1047	25	8.4154	-80	16.1044	0.0006	0.0003
White Banks	14128.4	43236.9	25	2.3551	-80	22.5961	25	2.3544	-80	22.5957	0.0007	0.0004

Table 1. – Sample TD values converted to latitude and longitude using the proposed non-iterative algorithm, compared with results calculated using RTCM iterative method.

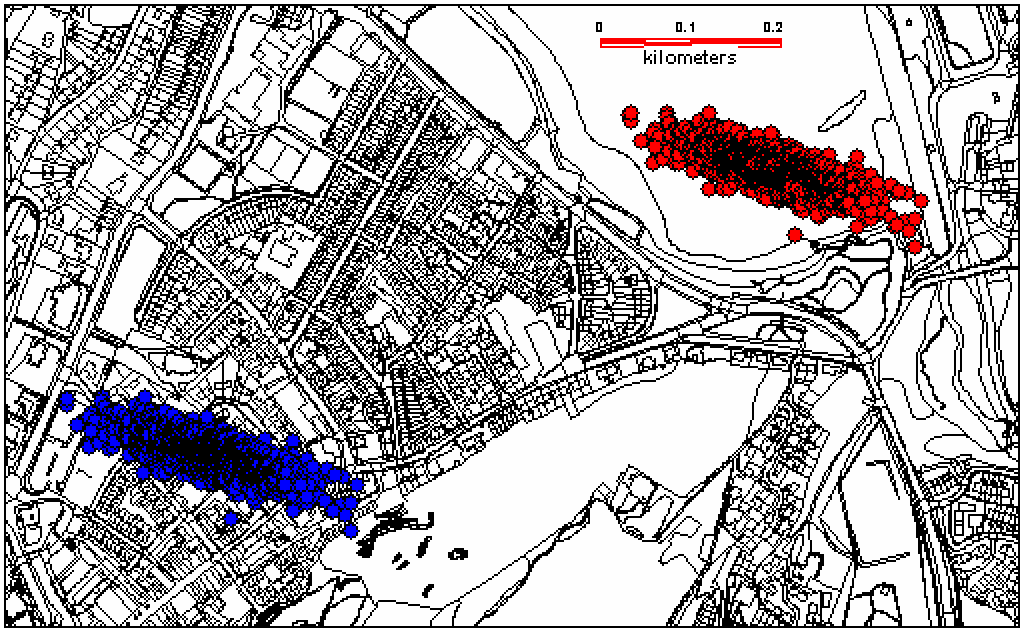

Fig. 8 – Position fixes computed from Loran TD measurements taken at Bangor. Red: ASFs omitted. Blue: ASFs included.

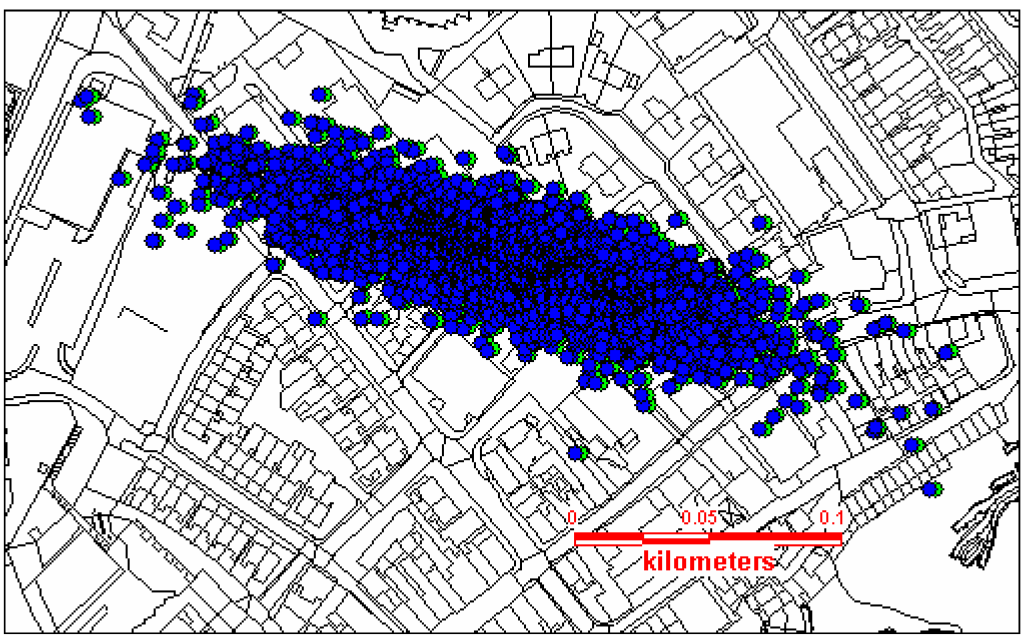

Fig. 9 – Position fixes computed from Loran TD measurements taken at Bangor. Blue: non-iterative solution. Green: RTCM-75 iterative solution.

5 Practical Issues

The locations of the transmitters need only be converted from WGS-84 to spherical co-ordinates once and for all; this may be done at the start of the process.

We have not investigated the effects of varying the position of the tangent point; the results were produced using the averaging method of establishing the tangent point described above.

6 Summary and Conclusions

In this paper we have reviewed Razin's method for computing geographical positions from Loran-C TD data. We have proposed implementing his method in a new way that employs a dot product in geocentric Cartesian co-ordinates. This solution removes the need for the computation and storage of 14 constants for each station triad required by Razin. Razin does not show how to compute these constants, neither does he illustrate a method of computing the osculating sphere and the offset vector required for converting co-ordinates between the two geodetic data. The algorithm has been implemented in Microsoft Visual C++ and tested using a number of TD's measured using the Southeast US, and the Sylt (GRI 7499), Loran-C chains.

7 References

[1] Razin, S., 'Explicit (noniterative) Loran Solution', NAVIGATION: The Journal of the Institute of Navigation, Vol. 14, No. 3, Fall 1967.

[2] Fell, H., In 'The Institute's Professional File', NAVIGATION: The Journal of the Institute of Navigation, Vol. 22, No. 2, Summer 1975.

[3] 'Minimum Performance Standards (MPS) Automatic Co-ordinate Conversion Systems', Report of RTCM Special Committee No. 75, Radio Technical Commission for Marine Services, Washington, D.C, 1984.

[4] Forssell, B., 'Radionavigation Systems', Prentice Hall International, 1991

[5] Ashkenazi, V., Moore, T., and Hill, C., 'Datum: A Report and Software Package for the Transformation and Projection of Coordinates', EEC Report No. 237, Eurocontrol Experimental Centre, December 1990 (revised December 1993).