ARMY RESEARCH LABORATORY

The Derivation of the Equations and Algorithm Used to Track Communications Platform Locations in the Deployment Module of the Network Connectivity Analysis Module (NCAM) Software

by G. Welles Still

ARL-TR-5217 June 2010

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory
Aberdeen Proving Ground, MD 21005-5068

ARL-TR-5217 **June 2010**

The Derivation of the Equations and Algorithm Used to Track Communications Platform Locations in the Deployment Module of the Network Connectivity Analysis Module (NCAM) Software

G. Welles Still
Survivability/Lethality Analysis Directorate, ARL

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)* June 2010	**2. REPORT TYPE** Final	**3. DATES COVERED (From - To)** September 2006–August 2009

4. TITLE AND SUBTITLE The Derivation of the Equations and Algorithm Used to Track Communications Platform Locations in the Deployment Module of the Network Connectivity Analysis Module (NCAM) Software	**5a. CONTRACT NUMBER** **5b. GRANT NUMBER** **5c. PROGRAM ELEMENT NUMBER**
6. AUTHOR(S) G. Welles Still	**5d. PROJECT NUMBER** 04E5E1 **5e. TASK NUMBER** **5f. WORK UNIT NUMBER**
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory ATTN: RDRL-SLE-M Aberdeen Proving Ground, MD 21005-5068	**8. PERFORMING ORGANIZATION REPORT NUMBER** ARL-TR-5217
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	**10. SPONSOR/MONITOR'S ACRONYM(S)** **11. SPONSOR/MONITOR'S REPORT NUMBER(S)**

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The Network Connectivity Analysis Model (NCAM) software predicts the viability of links between moving communications nodes. The deployment model tracks the position of the moving nodes as a function of time by having the user outline the platform paths with waypoints and defines the time the platform leaves and arrives at each waypoint. It then interpolates the position of the waypoint at strategically chosen snapshot times. For verification purposes, the derivations of the equations the deployment module uses to do the interpolations are presented. The velocity profile assumed was trapezoidal in shape. This shape modeled the velocity closely enough so that platform motion was realistic, yet the input required from the user was limited and the equations describing the motion were manageable. The results were consistent with the assumed velocity profile. NCAM is being developed jointly by the Missile Defense Branch at Aberdeen Proving Ground, MD, and the Communications Electronic Warfare Branch at Fort Monmouth, NJ, of the U.S. Army Research Laboratory's Survivability/Lethality Analysis Directorate.

15. SUBJECT TERMS

great circle, interpolation, oblate spheroid, snapshot time, sphere, waypoint, WGS84, velocity profile, spherical coordinates, trigonometric functions, elevation, latitude, longitude, coordinate transfer, arrival time, departure time, cruise velocity

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** G. Welles Still
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	UU	82	**19b. TELEPHONE NUMBER** (*Include area code*) 410-278-3377

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

Executive Summary

The Network Analysis Connectivity Model (NCAM) is a software package designed to predict the viability of communication links between moving nodes.[1] It consists of six modules: the deployment module, the propagation module, the antenna module, the noise module, the link budget module, and the connectivity confidence interval (CCI) module. The deployment module keeps track of the platform latitude, longitude, and elevation as a function of time. The propagation module uses the Terrain Integrated Rough Earth Model[2] program to account for the effect of terrain on signal propagation. The antenna module applies antenna properties, such as gain and radiation pattern, to signal propagation. The noise module accounts for the effect of radio noise from man-made sources, such as jamming, transformers, vehicles, transmission from other radios, or heat from the broadcasting radio, and natural sources, such as the background cosmic radiation, the Sun, and the Earth. The link budget module takes the data from the previous modules and calculates the wireless link signal-to-noise ratio for each communications link. The CCI module calculates the appropriate standard deviation of the path loss associated with each link and computes the probability of successful signal reception. NCAM uses the six modules to do the calculations necessary to perform the wireless network simulation. NCAM requires all six modules to perform its calculations.

The approach used is to do all the calculations needed to determine the CCI at certain key times called snapshot times. The computations at the snapshot times form a series of samples through time of link viability, which, when viewed in sequence, shows how well the nodes are linked through time, much like frames in a movie show samples of cinema action. To predict the viability of communication links between moving nodes through time, it is necessary to track the location of all nodes.

The deployment module functions by requiring the user to input an outline of the platform's path using a set of sample points called waypoints. Each waypoint is specified with the latitude, longitude, elevation (read from a Digital Terrain Elevation Data database for a ground platform[3]), and arrival and departure time of each waypoint. The deployment module then interpolates the position of the platform at the snapshot times.

[1] Still, G. W.; Nealon, J. F. *The Case for Using the Spherical Model to Calculate the Interpolated Points in the Connectivity Software Deployment Module*; ARL-TR-4373; U.S. Army Research Laboratory: Aberdeen Proving Ground, MD, February 2008.

[2] Eppink, D.; Keubler, W.; *TIREM/SEM Handbook*; ECAC-HDBK-93-076; Department of Defense Electromagnetic Compatibility Analysis Center: Annapolis, MD, March 1994; pp 1–134.

[3] Pablo's Mission Planning. http://www.mission-planning.com/DTED_Part2.htm (accessed 28 June 2007).

For verification purposes, this report documents the derivation of the equations that the deployment module uses to interpolate the positions of the moving communications platforms (or nodes) and the development of the algorithm for adding, removing, and modifying the waypoints. In developing the equations, it was assumed that there were two constant velocities: either zero or some maximum cruising velocity v_{cru}. v_{cru} is taken from a database, and its value is selected based on terrain roughness and the platform type. When the velocity changes, it increases or decreases at a constant acceleration rate of $\pm a_0$. This results in a time vs. velocity graph that is trapezoidal in shape. Furthermore, the velocity of the platform as it arrives and leaves the waypoint must be the same. A platform is allowed to change acceleration from + (or –) a_0 to – (or +) a_0 as it passes through a waypoint or remain at zero if it is traveling at v_{cru}. The platform also has the option of lingering at a waypoint.

The user is given the choice of specifying platform speed or arrival time at each waypoint. The motion equations derived for interpolation also provide a lower and upper bound of the node's speed or arrival time at each platform. So, the user is given the maximum choice for specifying node motion but is prevented from entering times and speeds that are impossible for the platform to attain.

The algorithm developed provides for the removal or addition of waypoints, changing the arrival time/velocity at waypoints, or changing the location of the waypoints. When the waypoint properties are modified, where possible, the waypoint velocity remains the same or is modified to be the minimum or maximum allowed for the platform, and the arrival/departure times are recalculated for the subsequent waypoints.

The velocity profile was chosen so as to make the platform motion realistic. A simple profile would have been unrealistic. Making the profile more complicated would make the motion more realistic, but the derivation of the equations needed for interpolation would have been more involved than those presented here. A more complex velocity profile would also have required more user input. The velocity profile (and the subsequent derivation of the equations of motion) here provided the best compromise between user control, equation and algorithm simplicity, and realistic platform motion.

NCAM is being developed jointly by the Missile Defense Branch at Aberdeen Proving Ground, MD, and the Communications Electronic Warfare Branch at Fort Monmouth, NJ, of the U.S. Army Research Laboratory's Survivability/Lethality Analysis Directorate. For verification purposes, the details of the algorithm and formulae development have been documented and shown to be consistent with the assumptions of the model's description of platform motion.

1. Introduction

A new software package called the Network Connectivity Analysis Model (NCAM) (*1*, pp 3–4) is currently undergoing development. It is being developed jointly by the Missile Defense Branch at Aberdeen Proving Ground, MD, and the Communications Electronic Warfare Branch at Fort Monmouth, NJ, of the U.S. Army Research Laboratory's Survivability/Lethality Analysis Directorate. NCAM predicts the viability of communications links between moving nodes.

Six modules constitute NCAM, as illustrated by the screen capture of the graphical user interface shown in figure 1. The deployment module calculates and tracks the movements of the wireless network nodes. The propagation module calls on the program TIREM (the Terrain Integrated Rough Earth Model) (*2*) to calculate signal attenuation due to the effect of atmosphere and terrain. The antenna module contains data concerning the antennae used, including gain, loss, and electromagnetic radiation pattern. The noise module computes receiver noise attributable to internal, external, man-made, natural, hostile, and nonhostile noise sources. The link budget module takes the data from the previous modules and calculates the wireless link signal-to-noise (S/N) ratio for each communications link. The connectivity confidence interval (CCI) module calculates the appropriate standard deviation of the path loss associated with each link and computes the probability of successful signal reception. NCAM uses the six modules to do the calculations necessary to perform the wireless network simulation.

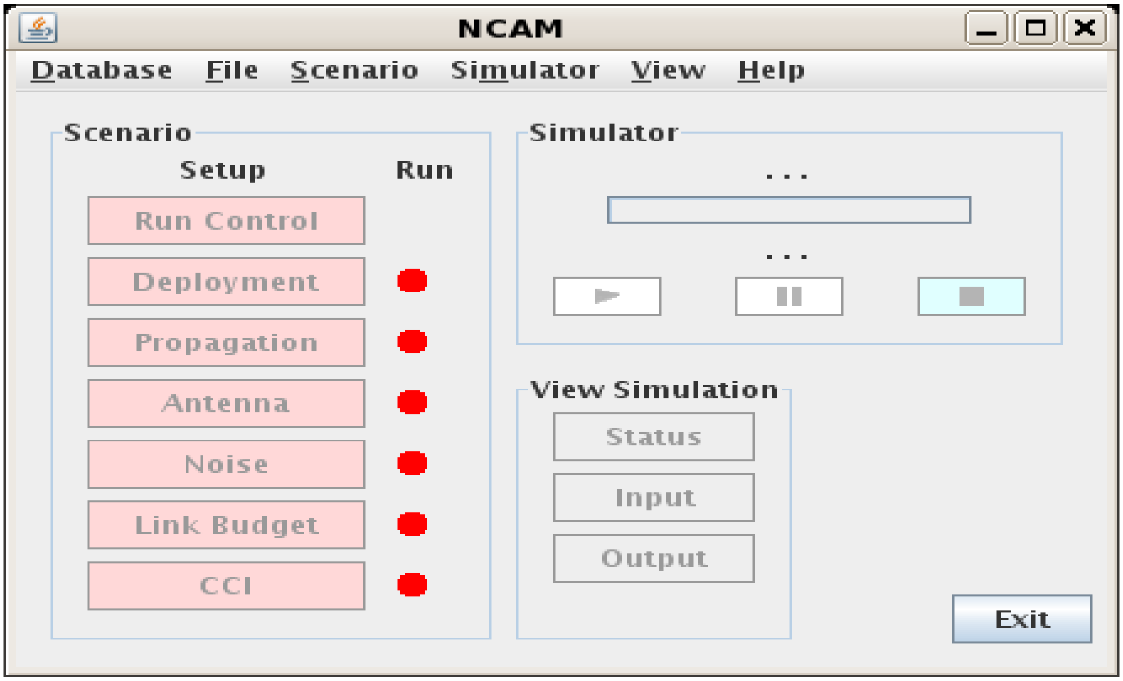


Figure 1. Modules of the connectivity software NCAM.

1.1 Background

The deployment module was the first module developed. At certain times in the simulation called snapshot times, the S/N ratio and the CCI is calculated for every link between the moving nodes. To do those calculations, the distance between nodes must be derived from their positions. To obtain those distances, the Earth could have been modeled either as a perfect sphere (with elevations in the radial direction, whose values are very small compared to the radius of the Earth) or as an oblate spheroid (also with small variations in the radial direction to represent elevation). Developers chose the spherical model of the Earth as the basis model for distance computation because it allowed more rapid computation, with a small, but acceptable, sacrifice in accuracy (*1*, pp 3–4).

1.2 Purpose

The user defines the path for each communications node or platform by selecting key locations along the path called waypoints. Successive waypoints in time are then connected by great circles, which are the shortest lines between two points on the surface of a sphere. The great circle approach accounts for the locations and the elevation of each waypoint.

This report traces the derivation of equations describing platform motion. The equations derived will be used by NCAM's deployment module to obtain platform position as a function of time. This allows NCAM to calculate and use the distance information in its calculation of link viability. An algorithm allowing for the modification of the number of waypoints for each platform and the modification of waypoint location is also included to allow modification of the platform paths. This presentation of the motion equations and the algorithm is intended to provide evidence for validating the NCAM software.

1.3 Scope

The derivations are limited to the deployment module. Because elevations are considered, the equations derived can be applied to either ground or airborne nodes. Due to the nature of the trigonometric functions, special limitations are also considered with the derivation of the equations.

2. Methodology

Platform motion is assumed to be along a great circle path. Platform velocities vary between a zero or nonzero cruise velocity. Velocities between zero and the cruise velocity are attained while accelerating between the cruise and zero velocity at either a positive or negative rate of constant acceleration. Solutions to special problems occasioned by the limitations of the trigonometric functions are also considered, as is the process needed to add, remove, or modify waypoint locations.

3. Analysis

The analysis consists of deriving the equations of motion for the platforms using spherical coordinates. After reviewing the spherical coordinate system, the equation for a great circle is derived. The equation is derived for two arbitrary points that define the circle, and special circumstances such as the two defining points (or waypoints) nearly antipodal on the equator, at opposite poles, antipodal, on the same or opposite meridians, one waypoint at the pole, both waypoints on the equator, or one point at the intersection of the equator, and the international date line, or prime meridian. The analysis ends with the derivation of the value for the Earth's average radius used in the spherical model.

3.1 Spherical Latitude-Longitude Coordinate System Review

The latitude-longitude system of coordinates used in modern navigation on Earth is illustrated in figure 2. ϕ is the latitude, which varies from –90° to 90°. The designation "north latitude" indicates a positive value, while "south latitude" indicates a negative value. θ is the longitude, varying from –180° to +180°. "East" corresponds to positive latitude, while "west" connotes negative latitudes. R is the distance from the Earth's center and is the sum of the mean sea level and the elevation. R is always positive and can have a value between 0 and infinity.

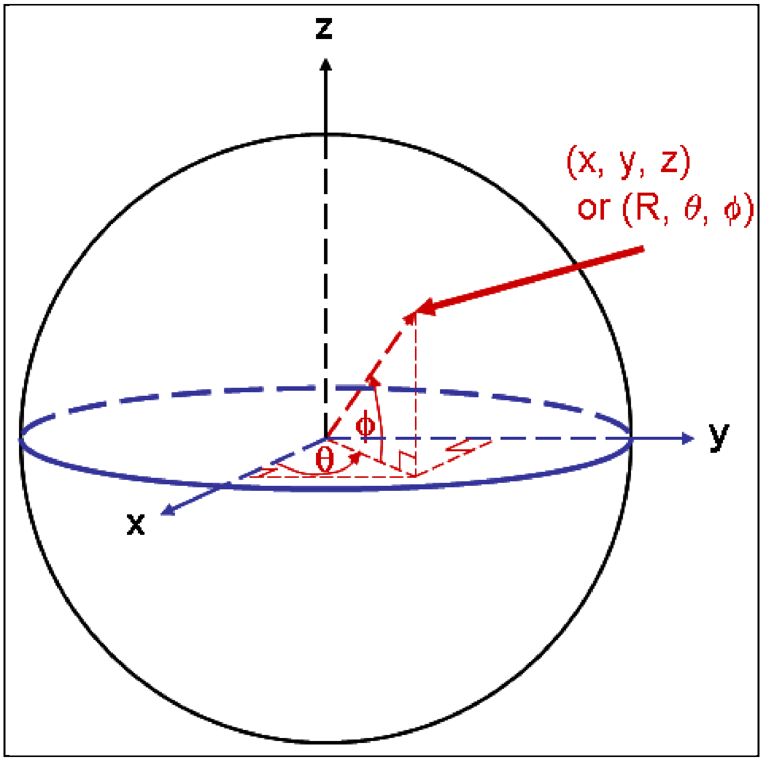


Figure 2. The latitude-longitude coordinate system superimposed on the rectilinear coordinate system.

To facilitate angle and distance calculations, we superimpose a three-dimensional (3-D), rectilinear coordinate system on the latitude and longitude system. The x axis starts at the Earth's center point and passes through the point where the prime meridian (0° longitude) and equator (0° latitude) intersect. Likewise, the y axis starts at the center point but goes through the point at the intersection of the equator (0° latitude) and 90° east longitude. The z axis begins at the Earth's center as well but passes through the North Pole (90° north latitude).

Converting from the latitude-longitude coordinate system to the rectilinear system involves employing simple geometric relationships. In the vertical right triangle in figure 2, the side opposite the latitude angle ϕ is the z coordinate. Expressed in terms of the latitude-longitude system, this becomes

$$z = R \sin \phi \ . \tag{1}$$

The side adjacent to the latitude angle ϕ becomes the hypotenuse for the two right triangles in the x-y plane shown in figure 2, with a value of $R \cos \phi$. The side opposite the longitude angle θ is the y coordinate, which, in terms of R, ϕ, and θ, is

$$y = R \cos \phi \sin \theta \ . \tag{2}$$

The side adjacent to the latitude angle ϕ is the x coordinate, rendered

$$x = R \cos \phi \cos \theta \ . \tag{3}$$

3.2 Equation of a Great Circle on a Sphere

We now derive the equation of a great circle on a sphere in terms of latitude angle ϕ and the longitude angle θ. First, consider the situation in figure 3: a sphere of arbitrary radius R_0 with an equator and an arbitrary great circle. Note that the hidden lines are not shown for clarity. A great circle is a circle of maximum radius drawn on a sphere so that it too has a radius of R_0, dividing the sphere into equal hemispheres (*3*). Note that the equator is also a great circle, and that it, the great circle, and the sphere share the same center point. Included is a rectilinear primed coordinate system with three axes: x', y', and z'. The relationship between the nonprimed coordinate system and the primed coordinate system will be established later. The y' axis begins at the center point and goes through one of the two points of intersection between the equator and the great circle. The x' axis begins at the center point and goes through a point on the equator such that the x' axis forms a 90° angle with the y' axis. The z' axis begins at the center point and goes through the pole, forming a right angle with both the x' and the y' axes. The sense of the three primed axis is such that x' cross y' equals z'.

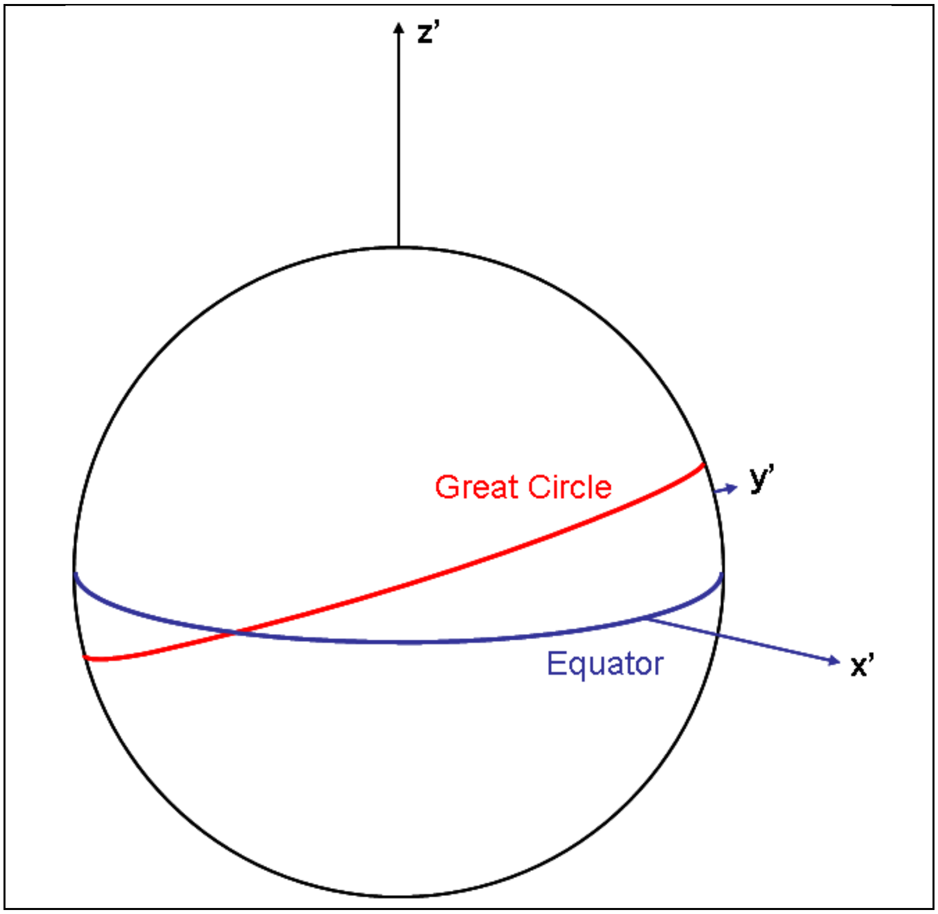


Figure 3. A sphere with a primed rectilinear coordinate system and great circle added.

Next, we consider the angle the great circle forms with the equator. Figure 4 shows the great circle and the equator with the two intersecting planes: plane 1 contains the equator and the sphere's center point, while plane 2 contains the great circle and the sphere's center point. (The sphere has been removed from figure 4 for clarity.) Planes 1 and 2 form the angle ϕ_{max} with each other. Plane 2 also forms the angle ϕ_{max} with the x' axis. The x' axis is also in plane 1, while the y' axis is in the planes' line of intersection. The z' axis is normal to plane 1. For the next part of the derivation, we consider the appearance of the planes as viewed in the direction of the y' axis, along the line of intersection of planes 1 and 2.

Figure 5 shows the planes from this viewpoint. We wish to concentrate on the relationship of planes 1 and 2 and realize the quantitative relationship between them and the z' and x' axes. From the point of view of figure 5, plane 2 is a straight line in the x'-z' axis. So, in the x'-z' system, the line must obey the equation z' = m x' + b (*4*). This is true for every point in plane 2, which contains the great circle. Since plane 2 intersects plane 1 at the point x' = z' = 0, b becomes zero. Because m is the slope of plane 2, this equals the tangent of the angle ϕ_{max}. When the substitutions established thus far have been made, the equation for a straight line becomes

$$z' = \tan \phi_{max} \, x' \, . \qquad (4)$$

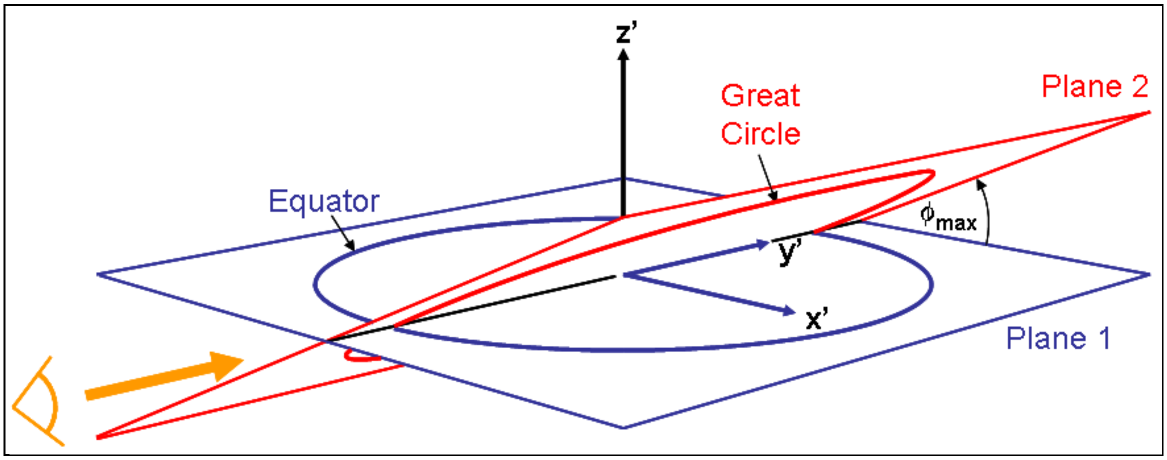


Figure 4. The angle between the great circle and the equator and the planes containing them. The sphere has been removed for clarity.

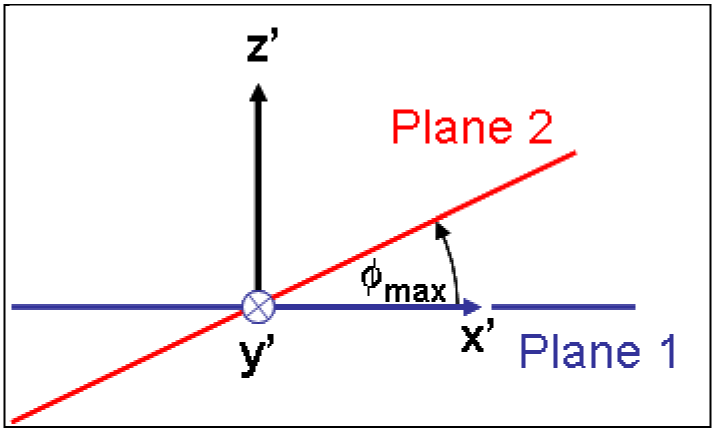


Figure 5. Viewing the sphere with planes 1 and 2 of figure 4 along the y' axis. The sphere has been removed for clarity.

Limiting our consideration of the equation for plane 2 to the great circle portion of plane 2, we substitute equations 1 and 3 for z' and x' and use the value of R_0 for the sphere's and circle's radii in planes 1 and 2. Because we are in the primed coordinate system and the notation must be kept consistent, the great circle equation becomes

$$R_0 \sin \phi' = R_0 \tan \phi_{max} \cos \phi' \cos \theta' \ . \tag{5}$$

Equation 5 is the equation for a great circle but only when a great circle intersects the equator in the y' axis. To obtain the more general equation, we superimpose a second set of axes on the primed system: the unprimed x, y, and z axes (figure 6). The new z axis is identical in magnitude and direction to the z' axis. Both the z and z' axes have the same origin point. That means that z = z'. Hence, equation 1 shows that the latitude of the primed and unprimed systems

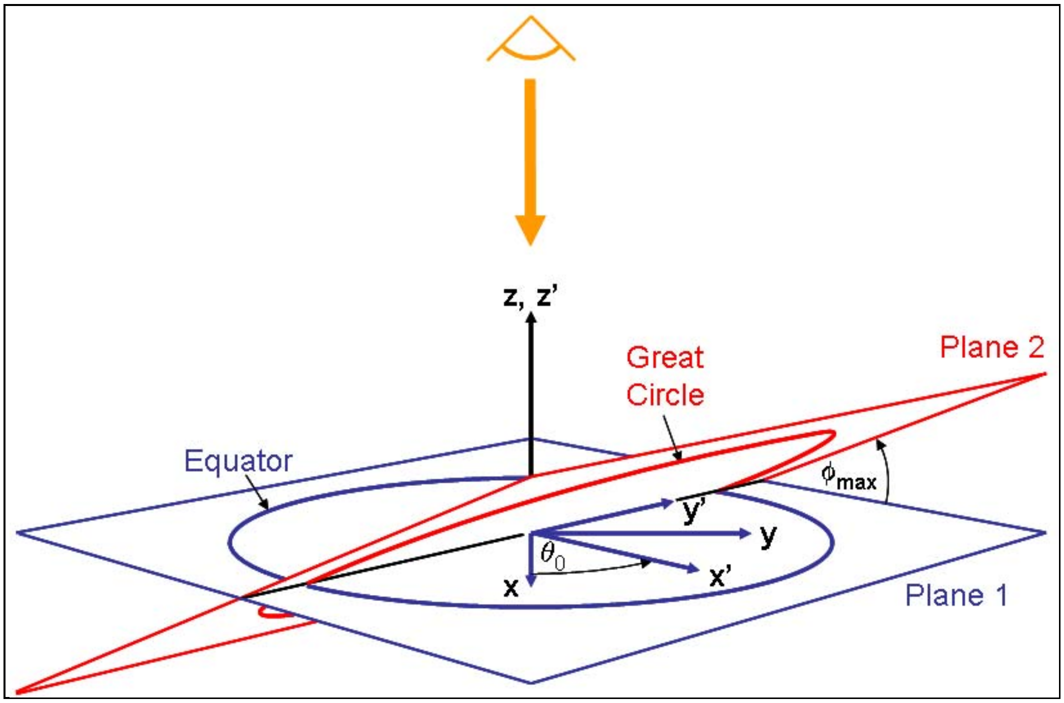


Figure 6. Figure 4 with an unprimed rectilinear coordinate system added.

are the same. In other words, $\phi = \phi'$. Since all the primed and unprimed axes have the same origin point, this puts the new x and y axes in the same plane as the old x' and y' axes (the equator's plane). The x' and y' axes are rotated from the x and y axes by the angle θ_0. To obtain the equation in the unprimed coordinate system, we will view the situation in the negative z (or negative z', since they are the same) direction (see figure 7).

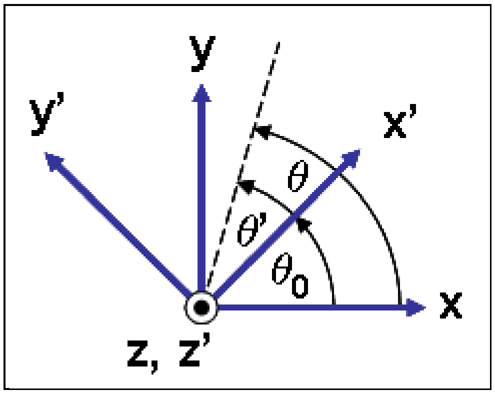


Figure 7. View of the geometry in figure 6
in the negative z direction.

Note that plane 2 and the great circle have been removed for clarity. The x (and y) axis makes an angle of θ_0 with the x' (and y') axis. An arbitrary angle θ' in the primed coordinate system (or θ in the unprimed coordinate system) has been added. As is evident from the figure, θ in the unprimed coordinate system equals $\theta' + \theta_0$. Solving for θ' and inserting the value into equation 5, taking advantage of the fact that $\phi' = \phi$ and dividing equation 5 by $R_0 \cos \phi'$, we realize the equation for a great circle:

$$\tan \phi = \tan \phi_{max} \cos (\theta - \theta_0) . \tag{6}$$

The point on the great circle that is farthest north of the equator has a longitude of θ' = 0°. This follows from the fact that the intersection of the equator and the great circle happens on the y' axis, at θ' = 90° and θ' = –90°. Since the angle between the planes that contains the great circle and equator is ϕ_{max}, it follows that this is the latitude of the point on the great circle at θ' = 0. The fact that this is the most northern point on the great circle and, hence, the point with the greatest latitude is the reason it has been designated ϕ_{max} in equation 6. As is apparent from equation 6, the northern most point on the great circle with latitude ϕ_{max} has a longitude of θ_0 in the unprimed coordinate system.

3.3 Establishing θ_0 and ϕ_{max} From Two Arbitrary Points

Suppose we are given two arbitrary waypoints with latitude and longitude coordinates (ϕ_1,θ_1) and (ϕ_2,θ_2). It is possible for them to define a great circle on the sphere. This is because the two points and the sphere's center point define a plane. The same plane intersecting the sphere defines a great circle. (Figure 6 shows how a great circle is contained in a plane. This is the same great circle shown in figure 3.)

It is possible to derive the equation for the great circle that connects the two points. By using equation 6 with (ϕ_1,θ_1) and (ϕ_2,θ_2), we can obtain the values of ϕ_{max} and θ_0. The first step is to plug in (ϕ_1,θ_1) and (ϕ_2,θ_2) to equation 6, realizing two instances of equation 6 in equations 7 and 8:

$$\tan \phi_1 = \tan \phi_{max} \cos (\theta_1 - \theta_0) \tag{7}$$

and

$$\tan \phi_2 = \tan \phi_{max} \cos (\theta_2 - \theta_0) . \tag{8}$$

We now have two equations (7 and 8) with two unknowns: ϕ_{max} and θ_0. Dividing equation 7 by equation 8 results in an expression that has one of the two unknowns, θ_{max} (and $\tan \theta_{max}$), eliminated:

$$\tan \phi_1 / \tan \phi_2 = \cos (\theta_1 - \theta_0) / \cos (\theta_2 - \theta_0) . \tag{9}$$

Next, we use the well-known trigonometric identity cos (A – B) = cos A cos B + sin A sin B (*5*). After applying it to the numerator and denominator of equation 9, dividing that expression by $\cos \theta_0$, then isolating θ_0 and taking the inverse tangent (arc tan), we find the value of θ_0:

$$\theta_0 = \text{arc tan}\,[(\tan \phi_1 \cos \theta_2 - \tan \phi_2 \cos \theta_1) / (\tan \phi_2 \sin \theta_1 - \tan \phi_1 \sin \theta_2)]\ . \qquad (10)$$

There are two solutions for θ_0 such that $-180° < \theta_0 < 180°$ because the function tan θ is periodic. That is, the value of tan θ repeats itself every 180° (*6*) as illustrated in figure 8. Because the possible range of longitude goes from –180° to +180° (i.e., it spans 360°), it is possible to have two full periods of the function tan θ represented in longitude. Figure 8 shows this. As a result, it is possible to have two values of θ for any single value of tan θ. The primary value of θ_0 as expressed in equation 10 will be in either the first or fourth quadrants. This is generally the default value given for the arc tan function for most computational software packages (*7*). It will therefore be necessary to add a line of code to make sure that both possible values of θ_0 are found. To find the second value, we subtract 180° if equation 10 gives us a value of θ_0 that is in the first quadrant (i.e., between 0° and 90°) or add 180° if equation 10 produces a value of θ_0 that is in the fourth quadrant (i.e., between 0° and –90°).

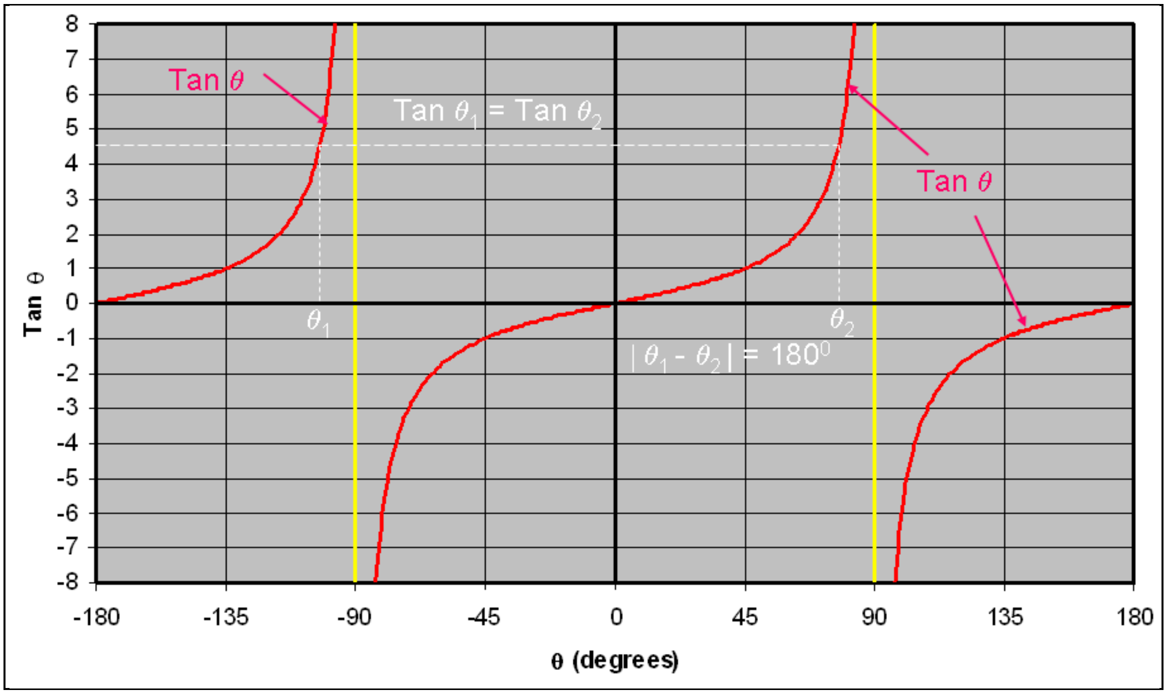


Figure 8. The function tan θ repeats itself every 180°.

θ_0 (along with ϕ_2 and θ_2) will be used to determine the value of ϕ_{max}. To find ϕ_{max}, we again use equation 6. Substituting in the values just derived for θ_0 with ϕ_2 and θ_2, dividing by cos $(\theta_2 - \theta_0)$, and taking the arc tan, we find

$$\phi_{max} = \text{arc tan}\,[\tan \phi_2 / \cos (\theta_2 - \theta_0)]\ . \qquad (11)$$

Limiting the range of latitude from –90° to +90°, equation 11 gives one value of ϕ_{max} for any one value of θ_0. Because our use of equation 10, and adding or subtracting 180° as required, produces two values for θ_0, it follows that there are two sets of values of θ_0 and ϕ_{max} that satisfy equations 10 and 11.

Cos θ is periodic in 360° (*8*). To see how having two values of θ_0 affects the values of ϕ_{max} in equation 11, we consider the unit circle in figure 9. The term cos $(\theta_2 - \theta_0)$ suggests that we examine how changing the value of θ_0 by 180° (and hence changing the value of the equation 11 term $\theta_2 - \theta_0$ by 180°) changes the value of ϕ_{max}. In figure 9, we see that the red line, orange lines, and black axis line form two identical right triangles. So, magnitude of the value of cos (θ – 180°) for any value of θ is the same as the cos θ. However, the side of the triangle representing the cos (θ – 180°) is on the opposite side of the origin as cos θ. Hence, the sign (but not the magnitude) of the cosine function changes from negative to positive or positive to negative when the argument θ changes by 180°.

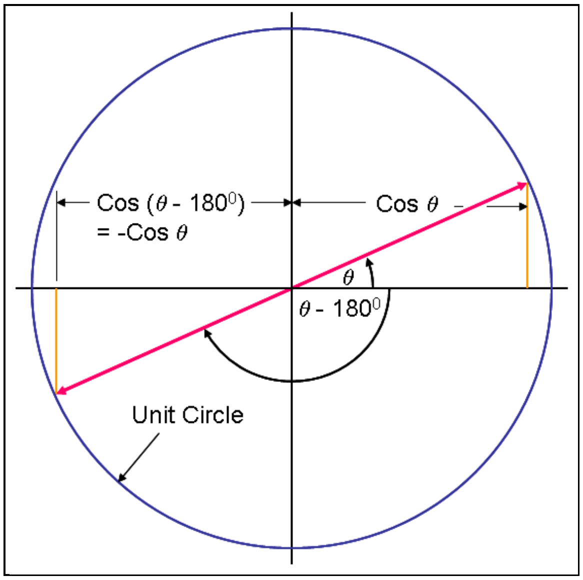


Figure 9. A unit circle showing the change of cos θ when θ changes by 180°.

Therefore, changing the value of θ_0 by 180° in equation 11 changes the sign of the term cos $(\theta_2 - \theta_0)$ but not the magnitude. Changing the sign, but not the magnitude, of the term cos $(\theta_2 - \theta_0)$ changes the sign but not the magnitude of ϕ_{max}. So, the two sets of values for θ_0 and ϕ_{max} can be expressed as (ϕ_{max},θ_0) and $(-\phi_{max},\theta_0 \pm 180°)$.

For any point on a great circle with the coordinates (ϕ,θ), the point exactly opposite it is 180° away in longitude θ, equidistant from the equator in latitude ϕ, but in the opposite hemisphere so it has the opposite sign. The two points opposite each other on a great circle can be expressed as (ϕ,θ) and (–ϕ,θ ± 180°). The points (ϕ_{max},θ_0) and $(-\ \phi_{max},\theta_0 \pm 180°)$ are two points opposite each other on the same great circle. So even though there are two sets of solutions to equations 10 and 11 for ϕ_{max} and θ_0, they both define the same unique great circle when applied in equation 6. This is because of the symmetry of the location the two sets of points on the great circle.

Figure 10 illustrates this symmetry. Note that part of plane 1 has been removed so that the point ($-\phi_{max}$,θ_0 ± 180°) is visible. It is therefore possible for any unique great circle described by equation 6 to have two sets of values for θ_0 and ϕ_{max}. To ensure that ϕ_{max} is indeed the maximum latitude on the great circle, we will choose the value of θ_0 that makes ϕ_{max} positive. Furthermore, note that the points (ϕ_{max},θ_0) and ($-\ \phi_{max}$,θ_0 ± 180°), the z/z' axis, and the x' axis, occupy the same plane. Figure 11 shows the relationship between the points (ϕ_{max},θ_0) and ($-\ \phi_{max}$, θ_0 ± 180°), and the x'and z/z' axes.

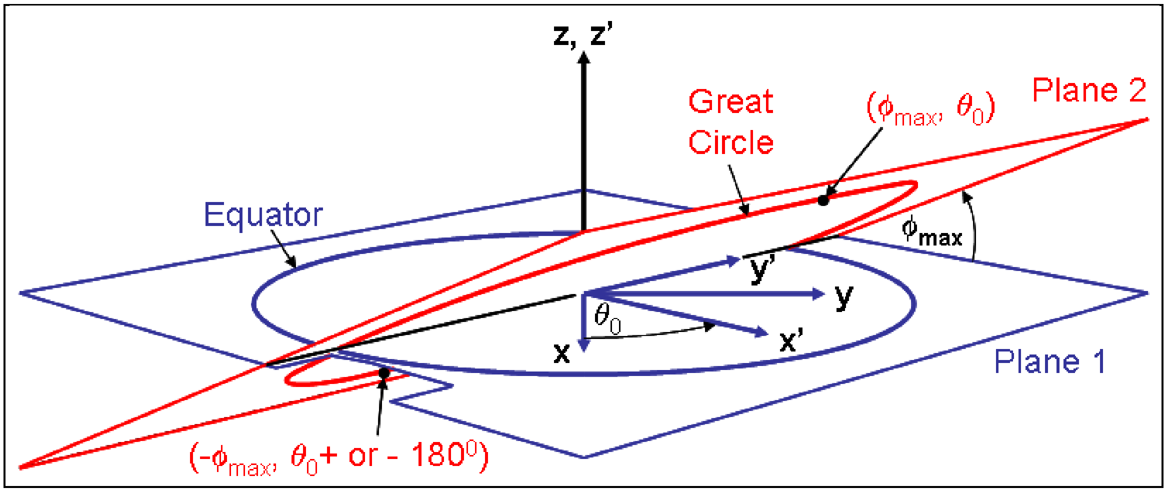

z, z'
Great Circle
(ϕ_{max}, θ_0)
Plane 2
Equator
ϕ_{max}
y'
y
θ_0
x
x'
Plane 1
($-\phi_{max}$, θ_0+ or - 180^0)

Figure 10. The two solutions for ϕ_{max} and θ_0 are opposite each other on a great circle.

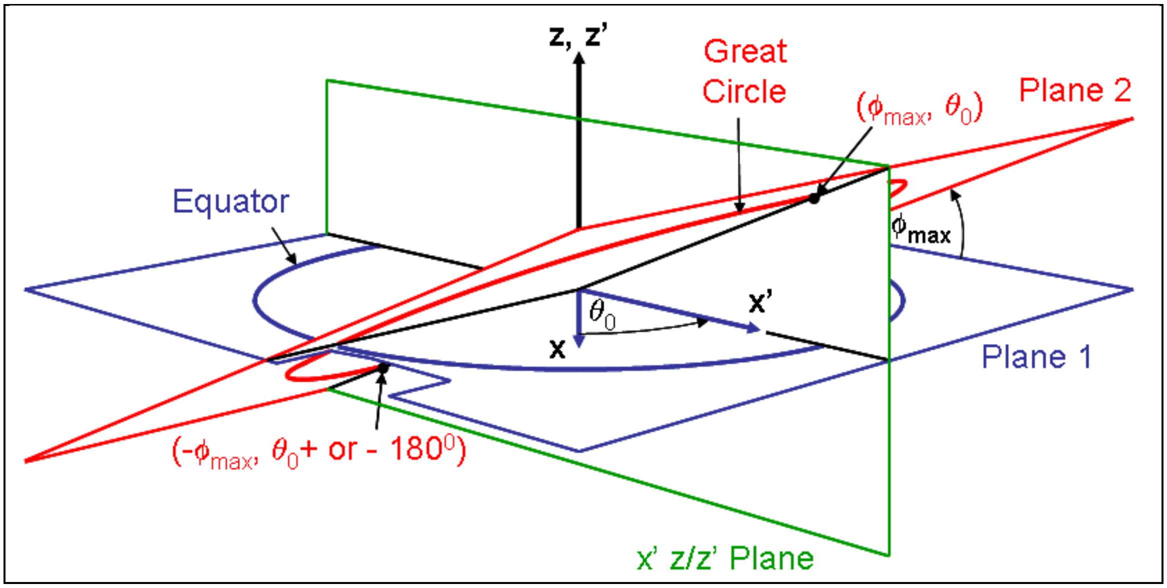


Figure 11. The points (ϕ_{max},θ_0), the origin, and ($-\ \phi_{max}$,θ_0 ± 180°) share the same plane with the x' and the z/z' axes.

3.4 Special Considerations

Equation 6 is the equation for a great circle in terms of the latitude ϕ and longitude θ. Equations 10 and 11 calculate the constants in equation 6 from two arbitrary points on the circle, (ϕ_1,θ_1) and (ϕ_2,θ_2). There are some values of (ϕ_1,θ_1) and (ϕ_2,θ_2) that do not allow equations 10 and 11 to produce values for ϕ_{max} and θ_0. We will now review those values.

3.4.1 (ϕ_2,θ_2) on the Equator

This means that $\phi_2 = 0$. This reduces equation 10 to

$$\theta_0 = \arctan\left(-1 / \tan \theta_2 \right) . \qquad (12)$$

To satisfy equation 12, θ_2 and θ_0 must differ by ±90°. Note that the great circle intersects the equator at two points. One is at $(\phi_2 = 0,\theta_2)$, as shown in figure 12. The other is at $(\phi_2 = 0, \theta_2 \pm 180°)$. (The second point is obscured by the x'-z/z' plane in figure 12.) The point (ϕ_{max},θ_0) has the maximum latitude on the great circle. The two points where the great circle and equator meet have latitude 0°. So, by symmetry, the point with the maximum latitude (ϕ_{max},θ_0) must be equidistant from the two points where the great circle and equator meet. That means $\theta_0 = \theta_2 \pm 90°$ and that (ϕ_{max},θ_0) must be in the x'-z/z' plane, consistent with equation 12.

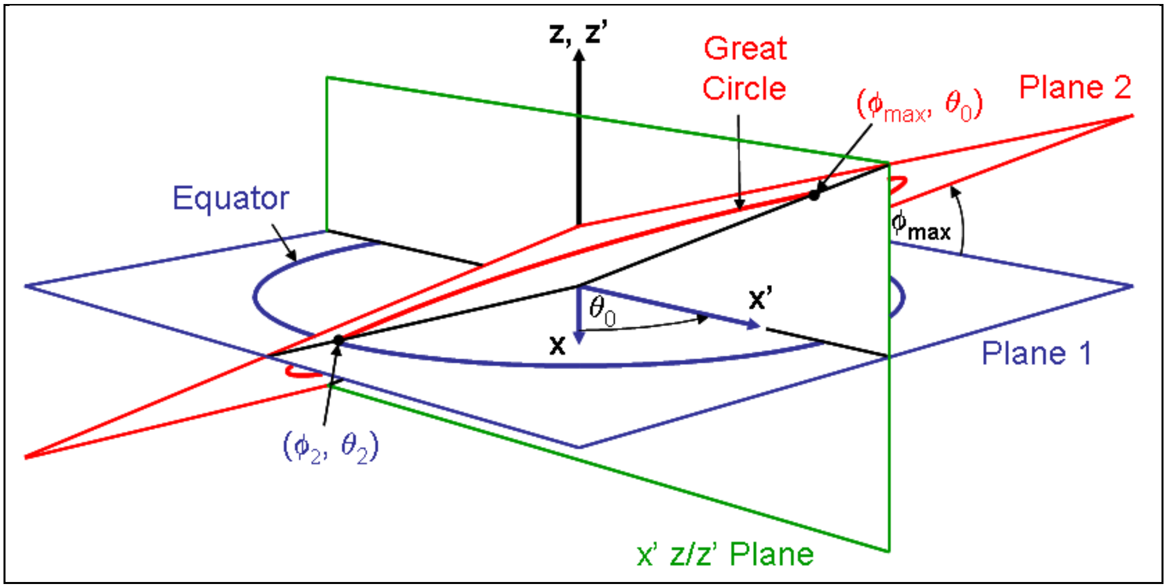


Figure 12. The points where the great circle cross the equator differ in longitude from (ϕ_{max},θ_0) by 90°.

To put it quantitatively and a little more generally,

$$| \theta_0 - \theta_2 | = 90°, \text{ or } 270° . \qquad (13)$$

Due to the periodic nature of the tangent function (see figure 8), there are two values of θ_0 that satisfy equation 12 in the –180° to +180° range. Note, too, that both values of θ_0 satisfy equation 13. Normally, we would use equation 11 and the two values of θ_0 to find the two corresponding values ϕ_{max} and take the set of ϕ_{max} and θ_0 that renders a positive ϕ_{max}. Unfortunately, cos $(\theta_0 - \theta_2) = 0$ because $\theta_0 - \theta_2 = \pm 90°$, or ±270° by equation 13. This causes the denominator of equation 11 to be zero. Since (ϕ_2,θ_2) is on the equator, $\phi_2 = \tan \phi_2 = 0$. Thus, equation 11 reduces to 0 divided by 0, rendering the value of ϕ_{max} as undefined.

Fortunately, (ϕ_1,θ_1) is not on the equator. Equation 11 could have been derived to express ϕ_{max} in terms of θ_1 by rendering it

$$\phi_{max} = \text{arc tan } [\tan \phi_1 / \cos (\theta_1 - \theta_0)] \ . \qquad (14)$$

Because (ϕ_1,θ_1) is not on the equator, ϕ_1 is not zero, and consequently, $\theta_0 - \theta_1$ does not equal ±90° or ±270°. This fact permits us to use equation 14 instead of equation 11 to calculate the two values of ϕ_{max} and choose the positive value to make the set (ϕ_{max},θ_0).

Had it been (ϕ_1,θ_1) on the equator instead of (ϕ_2,θ_2), equation 12 would have rendered θ_0 as

$$\theta_0 = \text{arc tan } (- 1 / \tan \theta_1) \ . \qquad (15)$$

Since equation 11 makes no use of ϕ_1 or θ_1, it could have been used to calculate the values of ϕ_{max}. Hence, in the instance of (ϕ_1,θ_1) sitting on the equator and (ϕ_2,θ_2) not sitting on the equator, the calculation for θ_0 and ϕ_{max} could have been handled without special consideration.

3.4.2 (ϕ_2,θ_2) or (ϕ_1,θ_1) but not Both at the Intersection of the Equator and the International Date Line or the Prime Meridian

Should (ϕ_2,θ_2) be at the intersection of the equator and the prime meridian or the equator and the international date line, it would not be possible to calculate the value of θ_0 using equation 10. Equation 10 is repeated here as equation 16:

$$\theta_0 = \text{arc tan } [(\tan \phi_1 \cos \theta_2 - \tan \phi_2 \cos \theta_1) / (\tan \phi_2 \sin \theta_1 - \tan \phi_1 \sin \theta_2)] \ . \qquad (16)$$

The reason this equation cannot be used is that $\phi_2 = 0°$ and $\theta_2 = 0°$ or ±180°, resulting in $\tan \phi_2 = \sin \theta_2 = 0$, causing the denominator to be zero and the value of θ_0 to be undefined.

Fortunately, because the equator and the great circle meet at the points $(\phi_2 = 0°,\theta_2 = 0°)$ and (0°,±180°), then θ_0 must be ±90°. Once again, the symmetry argument is applied as shown in figure 13. The figure illustrates the instance where $(\phi_2 = 0°,\theta_2 = \pm 180°)$. The other point where the equator and the great circle meet (0°,0°) is obscured by the x-z plane. To be the point with maximum latitude, it must be halfway between the points where the equator and the great circle intersect, which is at $\theta_0 = \pm 90°$. With two values of θ_0 established, it is possible to calculate 2 values of ϕ_{max} using equation 14, because ϕ_1 and θ_1 are not on the equator. It is then a matter of selecting the value of θ_0 which resulted in a positive value of ϕ_{max}.

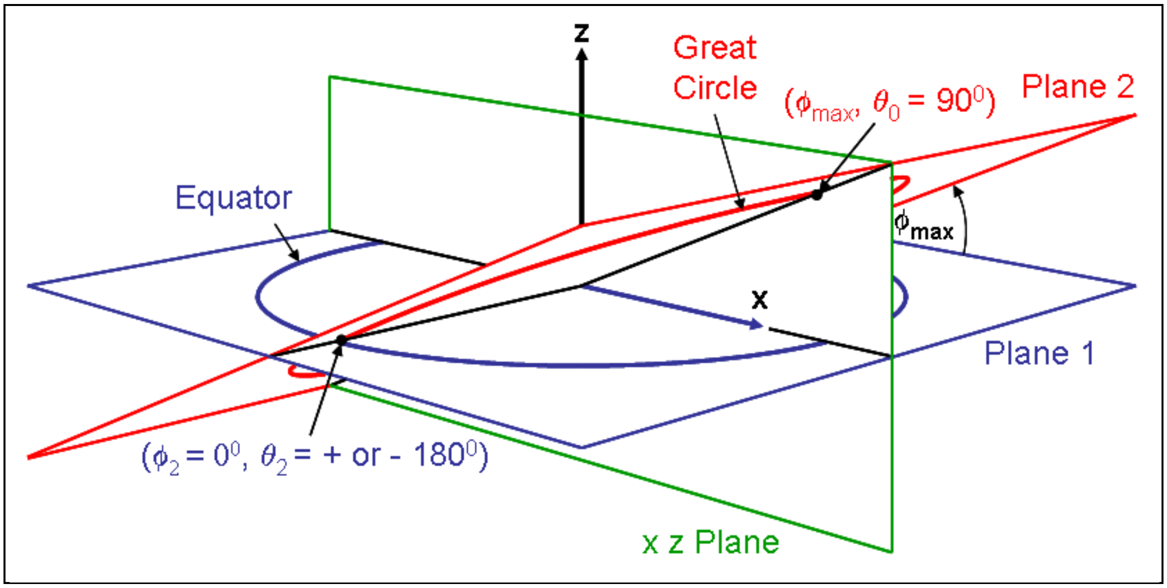

z
Great
Circle
(ϕ_{max}, θ_0 = 90^0)
Plane 2
Equator
ϕ_{max}
x
Plane 1
(ϕ_2 = 0^0, θ_2 = + or - 180^0)
x z Plane

Figure 13. ϕ_2 and θ_2 are the intersection of the international date line and the equator.

Alternatively, should (ϕ_1,θ_1) be at the intersection of the equator and the prime meridian or the equator and the international date line, equation 16 becomes undefined for the same reason it did when (ϕ_2,θ_2) was at the intersection of the equator and the prime meridian (or the equator and the international date line). And for the same reason, θ_0 must once again be ±90°. But this time, equation 11 is used to determine the two values of ϕ_{max}.

If θ_0 tends toward ±90°, then tan θ_0 tends toward ±infinity. This is consistent with the denominator of equation 16 tending toward zero. This gives us confidence that in this instance, our choice of θ_0 = ±90° was the correct one.

3.4.3 Both (ϕ_2,θ_2) and (ϕ_1,θ_1) on the Equator

If both (ϕ_2,θ_2) and (ϕ_1,θ_1) are on the equator, then the only great circle that can contain both points is the equator. In that instance, ϕ_{max} = 0°, and θ_0 is undefined. The equation for the great circle is ϕ = 0°, and θ can be any value between –180° and +180°. Figure 14 illustrates this case.

3.4.4 Both (ϕ_2,θ_2) and (ϕ_1,θ_1) on the Same or Opposite Meridians

Initially, we will assume that at least one of the two points is not on the equator or at the pole. This means that both the numerator and denominator in equation 16 will be finite so that two values of θ_0 can be established. However, the value of ϕ_{max} cannot be found so readily. To see why, we consult figure 15.

In this case, the mathematical relationship between θ_1 and θ_2 is that they are either equal or differ by 180°. This causes the plane of the great circle (plane 2 in the figure) to be at a right angle to the equatorial plane. This means that ϕ_{max} = 90°, causing tan ϕ_{max} to be undefined (see equation 14). Note that the sphere has been removed from the figure for clarity.

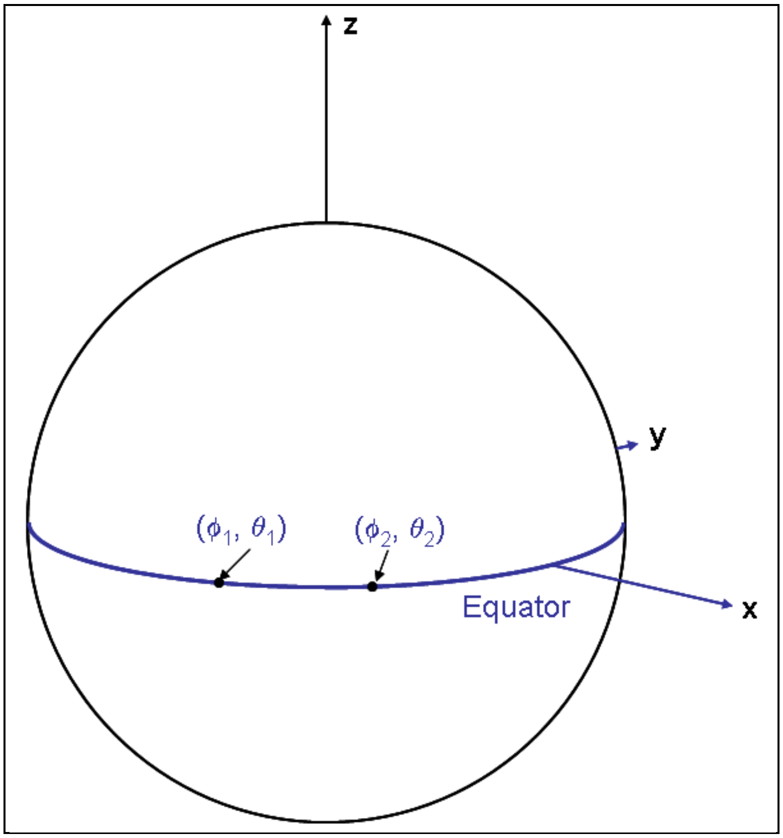


Figure 14. Both (ϕ_2,θ_2) and (ϕ_1,θ_1) are on the equator.

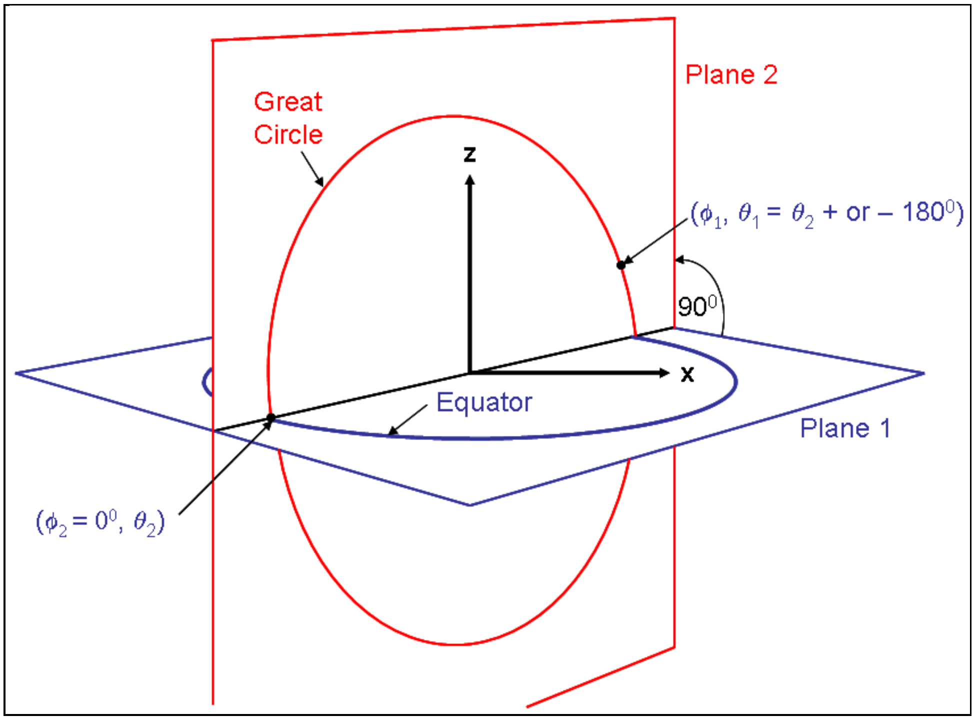

Plane 2
Great Circle
z
(ϕ_1, θ_1 = θ_2 + or – 180^0)
90^0
x
Equator
Plane 1
(ϕ_2 = 0^0, θ_2)

Figure 15. The two waypoints are on opposite meridians.

Note, too, that the value of θ_0 returned by equation 16 differs by ±90° from θ_1 and ±90° from θ_2, which, in turn, is equal to or ±180° from θ_1. This is consistent with the latitude θ_0 sitting midway between the latitudes of the intersection points of the equator and the great circle. In this case, the great circle is a meridian line, so the latitude of the intersection points of the meridian and the equator is equal to θ_1 and is ±180° from θ_1. However, due to the symmetry of the great circle and its orientation to the equator (see figure 15), it is impossible to tell which value of θ_0 corresponds to the value of $\phi_{max} = 90°$. Therefore, the equation of the great circle with both (ϕ_2,θ_2) and (ϕ_1,θ_1) on the same or opposite meridians will be

$$\theta = \theta_1, \text{ while } -90° \leq \phi \leq +90°, \text{ and if } \theta_1 < 0,\ \theta \text{ also} = \theta_1 + 180°:$$

$$\text{if } \theta_1 > 0,\ \theta \text{ also} = \theta_1 - 180°, \text{ while } -90° \leq \phi \leq +90°. \qquad (17)$$

In equation 17, there are only two possible values of θ, yet ϕ may take on any value. Note that equation 17 is equally valid if (θ_1,ϕ_1) is replaced by (θ_2,ϕ_2).

3.4.5 (ϕ_2,θ_2) or (ϕ_1,θ_1) at a Pole

This means that ϕ_1 or ϕ_2 equals ±90°. In this case, the equation for the great circle becomes $\theta = \theta_j$, while $-90° \leq \phi \leq +90°$ and if $\theta_j < 0$, θ also $= \theta_j + 180°$: if $\theta_j > 0$, θ also $= \theta_j - 180°$, while $-90° \leq \phi \leq +90°$, where j represents the number of the point that is not at the pole. Note that the equation is very similar to equation 17. Note too that in this case, θ_1 may not be equal to, or differ from, θ_2 by 180°. All that is necessary is for ϕ_1 or ϕ_2 to equal ±90°.

3.4.6 (ϕ_2,θ_2) and (ϕ_1,θ_1) Are Antipodal

Antipodal points are points whose coordinates are on exactly opposite sides of the sphere. This means that $\theta_2 = \theta_1 \pm 180°$, and $\phi_2 = -\phi_1$. Applying this to the definition for θ_0 as listed in equation 16, we find

$$\theta_0 = \text{ arc tan } \{[\tan \phi_1 \cos(\theta_1 \pm 180°) - \tan(-\phi_1) \cos \theta_1] /$$

$$[\tan(-\phi_1) \sin \theta_1 - \tan \phi_1 \sin(\theta_1 \pm 180°)]\}\ . \qquad (18)$$

For the sine and cosine functions, changing the argument of the functions by 180° changes the sign of the function (see figure 9). In other words (9),

$$\sin(\theta \pm 180°) = -\sin \theta, \quad \text{and} \quad \cos(\theta \pm 180°) = -\cos \theta \text{ for all } \theta\ . \qquad (19)$$

This reduced equation 18 to

$$\theta_0 = \text{ arc tan } \{\cos \theta_1[\tan \phi_1 + \tan(-\phi_1)] / \sin \theta_1\ [\tan(-\phi_1) + \tan \phi_1]\}\ . \qquad (20)$$

The tangent function is a periodic, odd function (*10*) meaning that $\tan(-\phi) = -\tan\phi$. This causes equation 20 to reduce to zero divided by zero, independent of the latitude or longitude. Hence, we realize that when the waypoints are antipodal, they do not define a unique great circle. To see why, we consult figure 16.

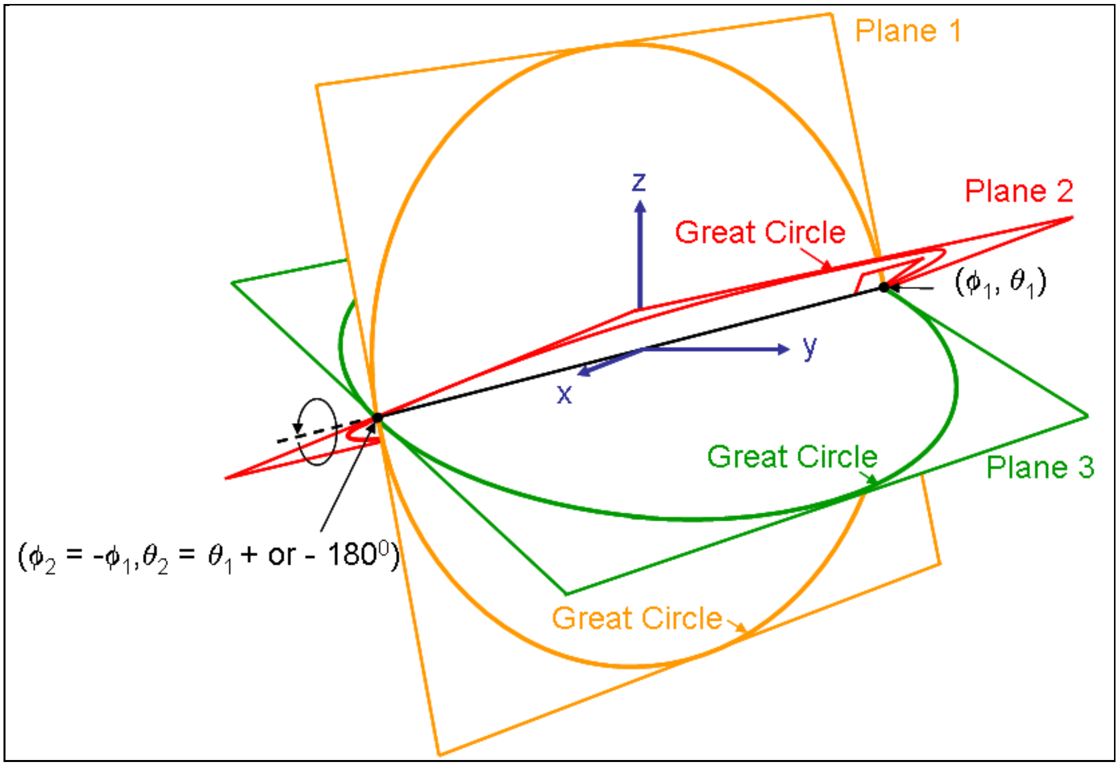


Figure 16. Two antipodal points do not define a unique great circle.

We choose an arbitrary point (ϕ_1,θ_1) in plane 1, which is the radius distance R from the center point at the intersection of the x, y, and z axes, which is the sphere's center point. Next, we choose the antipodal point $(-\phi_1,\theta_1 \pm 180°)$, also a distance R from the center point. Because the antipodal points are opposite each other on the sphere, a straight line can be drawn connecting the antipodal points and the sphere's center point. This straight line, 2R in length, serves as the diameter of the circle in plane 1 in the figure. In 3-D space, however, one diameter can serve as the diameter for an infinite number of circles, all of which are contained in a sphere of radius R, as shown by the circles in planes 2 and 3 in figure 16. An infinite number of great circles can be rotated around their common diameter. Therefore, two antipodal points on a sphere do not define a great circle. Note too that antipodal points need not be on the equator. Again, the sphere and the equatorial plane have been omitted from the figure for clarity.

If two waypoints are antipodal on the Earth's surface, the shortest distance between them will be on a path drawn through the poles. This is because the Earth's shape is very slightly flattened at the poles due to its rotation. So on our perfect sphere model, the equation for the great circle for antipodal points will be the same as for two points on the same or opposite meridians discussed in the previous section. The equation is equation 17, which is repeated here for convenience as equation 21:

$$\theta = \theta_1\text{, while } -90° \le \phi \le +90°\text{, and if } \theta_1 < 0,\ \ \theta \text{ also} = \theta_1 + 180°\text{:}$$
$$\text{if } \theta_1 > 0,\ \ \theta \text{ also} = \theta_1 - 180°\text{, while } -90° \le \phi \le +90°. \qquad (21)$$

Since the points are antipodal, it is equidistant to go through either the North or South Pole. For consistency, we chose to go through the North Pole for all antipodal waypoints.

3.4.7 (ϕ_2,θ_2) and (ϕ_1,θ_1) at Opposite Poles

In this case, ϕ_1 and ϕ_2 are at ±90°. The infinite number of great circles are meridians. In this case, we chose the equation that contains the prime meridian and the international date line. So, $\theta = 0°$ and $180°$ while $-90° \le \phi \le 90°$.

3.4.8 (ϕ_2,θ_2) and (ϕ_1,θ_1) Nearly Antipodal on the Equator

In this case, $\phi_1 = \phi_2 = 0$, and $|\ \theta_2 - \theta_1|$ is almost, but not quite, 180°. With the waypoints not antipodal, yet both on the equator, the equation of the great circle connecting them can be found using equations derived earlier in the instance when both waypoints were on the equator. The great circle connecting them would be the equator.

This would be true if the Earth were a perfect sphere. Unfortunately, it is slightly flattened at the poles, and when cartographic precision is needed, the Earth is modeled as an oblate spheroid (*11*), which is a special case of an ellipsoid (*12*), based on the WGS84 (World Geodetic System for 1984) model (*13*). In this instance, it may well be that the shortest path that connects the nearly antipodal waypoints passes closer to the poles than to the equator. By symmetry, the path could go past either the North or the South Pole. For simplicity, we decree that the path passes the North Pole (see figure 17).

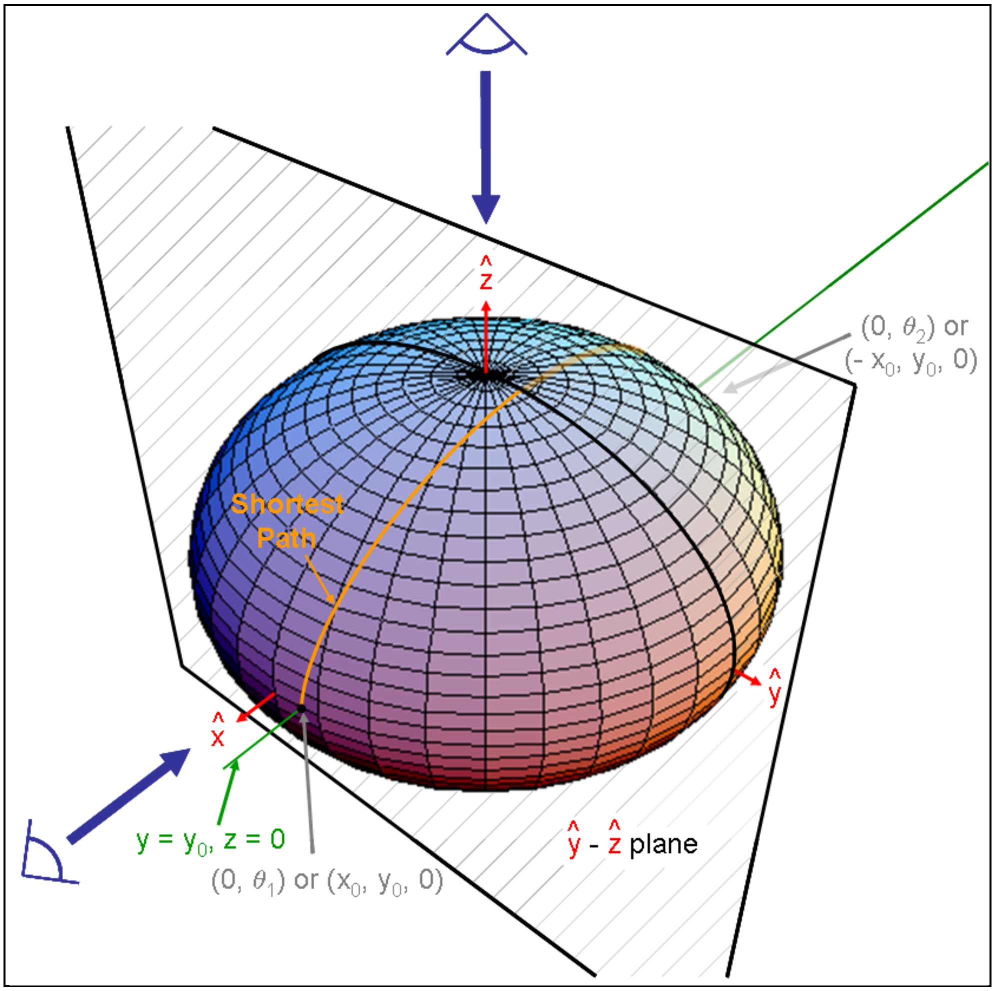


Figure 17. The shortest path on the oblate spheroid connecting two equatorial waypoints nearly opposite each other goes closer to the poles than the equator. The plane provides the cross section of the oblate spheroid.

The two waypoints can be connected with a straight line that goes through the Earth. Next, we superimpose a coordinate system such that the x axis is parallel to the line that connects the nearly antipodal waypoints. The line has the equation x = any real value, y = y_0, where y_0 is the distance between the line connecting the waypoints and the x axis, and z = 0. This puts the line connecting the waypoints in the x-y plane. It is also skew and perpendicular to the z axis.

The question that arises is just how close to the pole the shortest path line passes and what the functional dependence of the distance is on the value of y_0. To answer the question, we examine the cross section of the oblate spheroid in the y-z plane, viewing along the x axis in the negative x direction. Figure 18 shows the cross section. The cross section taken through the poles is an ellipse (*14*) with the following equation (*15*):

$$1 = y^2 / a^2 + z^2 / c^2 , \qquad (22)$$

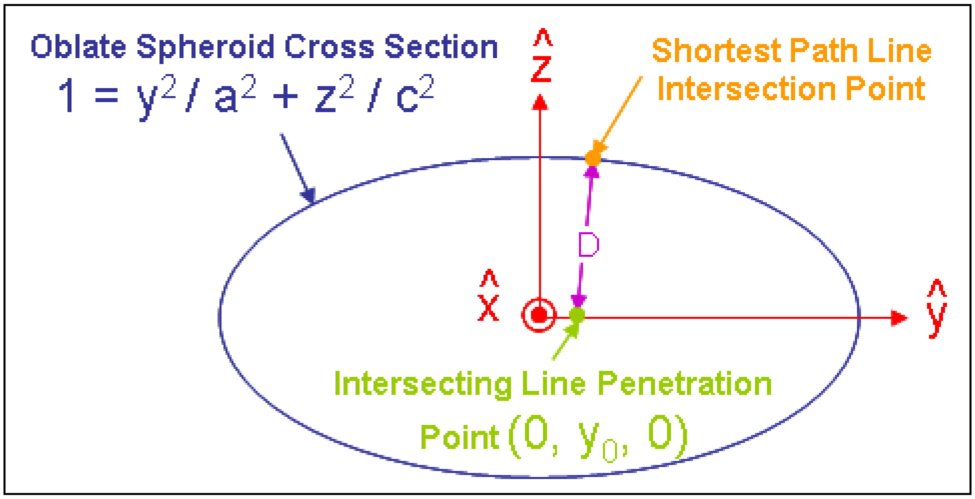

Oblate Spheroid Cross Section
$1 = y^2 / a^2 + z^2 / c^2$
$\hat{z}$
Shortest Path Line Intersection Point
D
$\hat{x}$
$\hat{y}$
Intersecting Line Penetration Point $(0, y_0, 0)$

Figure 18. The y-z cross section of the oblate spheroid of figure 17.

where a is the equatorial radius and c is the polar radius. The shortest path line on the surface of the oblate spheroid intersects the cross-sectional ellipse at a point fairly near the pole. The straight line connecting the waypoints intersects the y axis at the point $(0,y_0,0)$. These two points are separated by a distance D. The square of the distance D is represented by the equation

$$D^2 = (y - y_0)^2 + z^2 , \tag{23}$$

which can be rendered in terms of a single variable after solving the elliptical cross-sectional equation (equation 20) for z^2 and substituting into equation 23

$$D^2 = (y - y_0)^2 + c^2 (1 - y^2 / a^2) . \tag{24}$$

It follows that the shortest path line along the surface of the oblate spheroid should be the line closest to the straight line intersecting the waypoints. Therefore, the point on the ellipse where the shortest mean sea level path intersects the elliptical cross section should be a point on the ellipse that is closest to $(0,y_0,0)$. This point should have a value for y such that the distance squared, D^2, is minimized. The value for y is easily found by taking the y derivative of equation 24 and setting it to zero. The answer is

$$y = y_0 \; / (1 - c^2 / a^2) . \tag{25}$$

The greatest value y can have is the equatorial radius a, so substituting that value in equation 25 and solving for y_0 renders the maximum value possible for y_0, $y_{0\text{-max}}$ rendered

$$y_{0\text{-max}} = a (1 - c^2 / a^2) . \tag{26}$$

For two points on the equator and nearly antipodal, the line connecting them on the surface of an oblate spheroid does not follow the equator if they are within two y_0 of being 180° opposite each other. If they are closer than that, the closest path lies on the equator. Plugging in the WGS84 values for the Earth, for a = 6,378,137.0 m and c = 6,356,752.314 245 m (*13*), we find $y_{0\text{-max}}$ = 42697.6727075391 m.

To facilitate the derivation of the value of the minimum angle between two waypoints on the equator for which the shortest path between them is not on the equator, we define the angle β_{max} as follows:

$$\beta_{max} = \text{arc sin}\ (y_{0\text{-max}} / a)\ . \tag{27}$$

The minimum angle between two waypoints on the equator for which the shortest path is not on the equator then becomes

$$\kappa_{min} = 180° - 2\ \beta_{max}\ . \tag{28}$$

To see why, we consult figure 17. We suppose that we are looking in the negative z direction at the special case where the waypoints are located on the line y = $y_{0\text{-max}}$, z = 0, and that we are looking from the North Pole (ϕ = 90°) in the negative z direction and view the earth in cross section at the equator. We see the layout as shown in figure 19. (Note that "a" denotes the equatorial radius. The oblate spheroid has been removed for clarity.)

Equation 27 describes the simple geometric relationship between $y_{0\text{-max}}$, the equatorial radius a, and the angle β_{max} found in the figure. From the figure is evident that the angle κ_{min} plus twice β_{max} equals a straight angle, or 180º. Solving for κ_{min} produces equation 28.

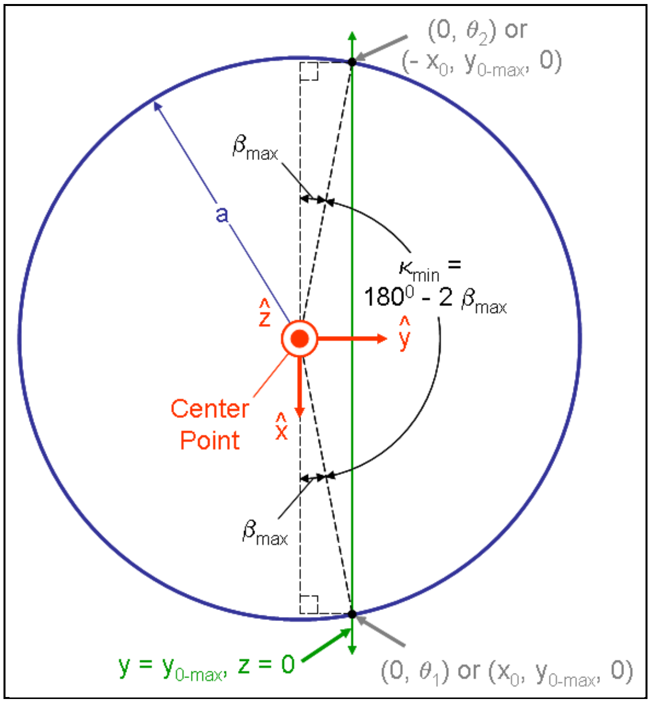


Figure 19. Cross-sectional view of the equator of figure 15 looking in the negative z direction when y_0 is at its maximum value and the definition of the angles κ_{min} and β_{max} added.

Applying the values of the WGS84 Geodetic System model (*13*) to equations 27 and 28 gives us the minimum separation for two waypoints on the equator that requires the path connecting them to go off of the equator. We conclude that if two waypoints on the equator are between 180° and 179.232874830387° apart in longitude, the shortest path connecting them is not along the equator. In describing angles separating the waypoints on the equator, keep in mind the fact that the longitude range is between –180° and +180°, and that the functions used to derive the angle is periodic. Taking these facts into account, the relationship between θ_1 and θ_2, which results in the shortest path between them laying off of the equator, is

$$180° < 180° - |180° - |\theta_2 - \theta_1|| < 179.232874830387° \, . \qquad (29)$$

This is the quantitative definition of "nearly antipodal."

The error introduced by using a sphere to predict the shortest path between two nearly antipodal points on the equator can be reduced greatly by using a special process. The process involves using the shortest path intersection point of figure 18. By symmetry, we know that the point is halfway between the two waypoints on the shortest path line and midway on the straight line connecting the two waypoints. (This is true for any two nearly antipodal points on the equator—the shortest path intersection point need not be in the y-z plane, as in figure 17.) The situation is illustrated in figure 20.

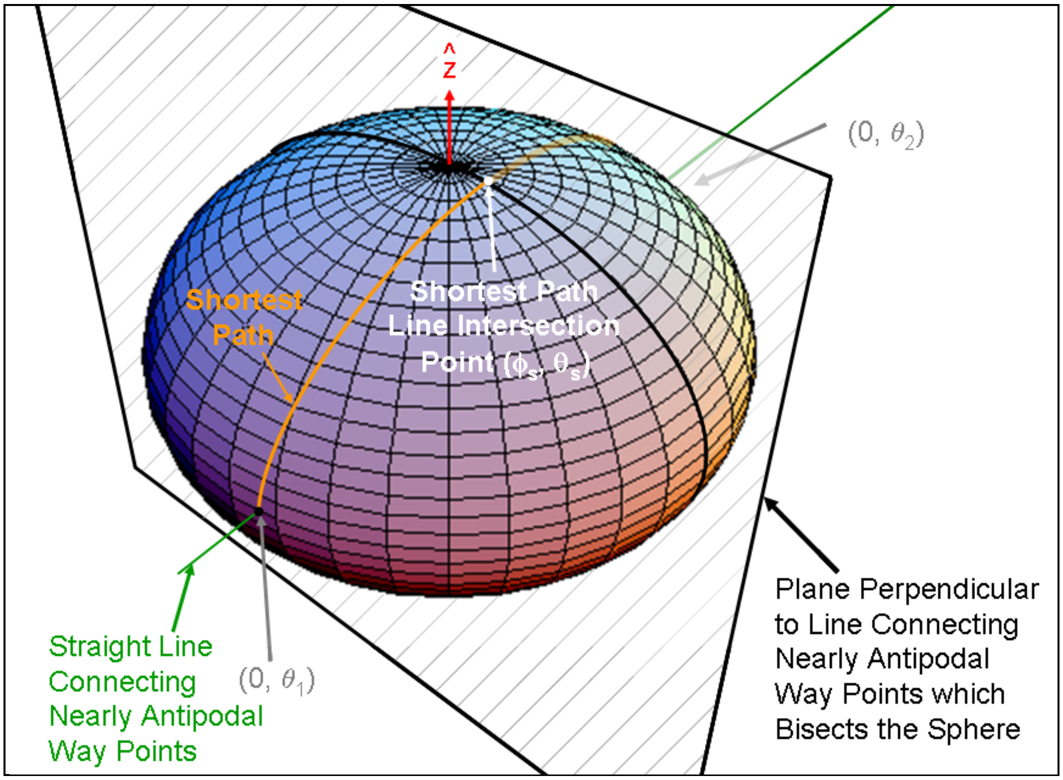

$\hat{z}$
$(0, \theta_2)$
Shortest Path
Shortest Path Line Intersection Point (ϕ_s, θ_s)
Straight Line Connecting Nearly Antipodal Way Points
$(0, \theta_1)$
Plane Perpendicular to Line Connecting Nearly Antipodal Way Points which Bisects the Sphere

Figure 20. Two nearly antipodal waypoints on the equator bisected by a plane that is perpendicular to the straight line connecting the nearly antipodal waypoints.

We added a plane that is defined by the origin, the North and South Poles, and the shortest path line intersection point (ϕ_s,θ_s). The plane is perpendicular to the straight line connecting the nearly antipodal points, and the plane bisects the sphere.

Having established that θ_s is midway between θ_1 and θ_2, we cannot simply sum θ_1 and θ_2 and divide by 2. The value of θ_1, θ_2, and θ_s is limited to the interval –180° to +180°. This fact, and the periodic nature of the variables, requires a more careful definition of θ_s. The complete definition is outlined in equation 30:

$$\text{If } |\,\theta_2 - \theta_1\,| < 180^\circ, \quad \theta_s = (\,\theta_1 + \theta_2\,)\,/\,2\,,$$

$$\text{If } |\,\theta_2 - \theta_1\,| > 180^\circ, \quad \text{and } (\,\theta_1 + \theta_2\,)\,/\,2 < 0, \;\; \theta_s = [\,(\,\theta_1 + \theta_2\,)\,/\,2\,] + 180^\circ,$$

$$\text{If } |\,\theta_2 - \theta_1\,| > 180^\circ, \text{ and } (\,\theta_1 + \theta_2\,)\,/\,2 \geq 0, \;\; \theta_s = [\,(\,\theta_1 + \theta_2\,)\,/\,2\,] - 180^\circ. \qquad (30)$$

We now derive the value of ϕ_s, the latitude of the shortest path line intersection point. We will assume that the relationship between the waypoints, the straight line intersecting them, and the axis is as shown in figure 21. If the final mathematical expression defining ϕ_s is independent of Cartesian variables, then the expression will be true for all longitudes. This is due to the polar axial symmetry of an oblate spheroid, and the fact that any ellipse produced by a cross section that includes both poles has the same major and minor radii.

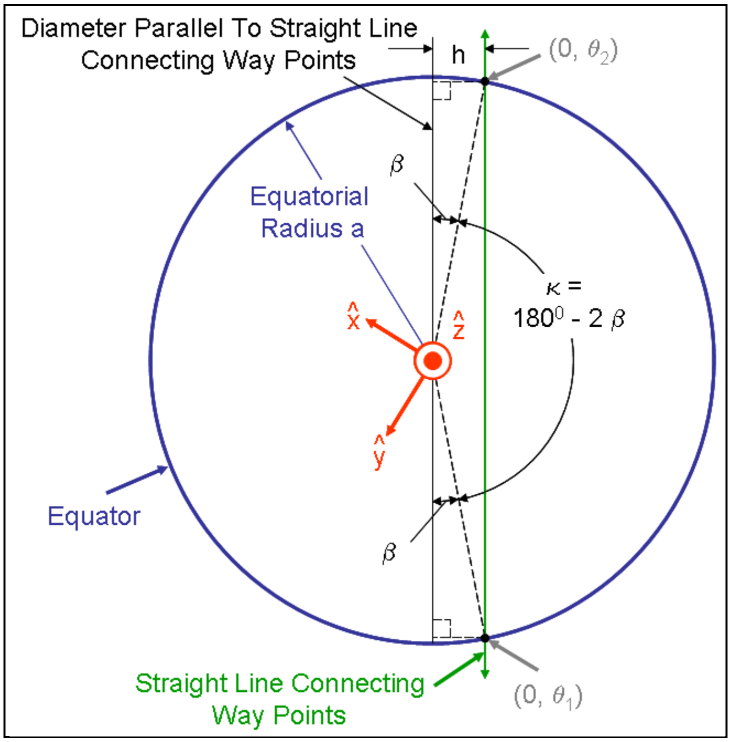


Figure 21. A more general view of figure 17: the straight line connecting the waypoints is not necessarily parallel to the x or y axis, yet it is perpendicular to the z axis.

To find the value of ϕ_s, we first consider a more general view of figure 17 where the line connecting the waypoints is not necessarily parallel to the y or x axis, as in figure 21. Yet it is still perpendicular to the z axis and contained in the x-y plane. The first step in finding ϕ_s is to find the angle β. Keeping in mind the fact that the limits of θ_1 and θ_2 are between –180° and +180°, we see from the figure that

$$\text{If } |\theta_2 - \theta_1| < 180°, \beta = (180° - |\theta_2 - \theta_1|) / 2 ,$$

$$\text{If } |\theta_2 - \theta_1| > 180°, \beta = (|\theta_2 - \theta_1| - 180°) / 2 , \qquad (31)$$

and

$$h = a \sin \beta . \qquad (32)$$

The next step involves a more generalized view of the cross section in figure 18, as shown in figure 22. The value of h has already been established in equation 32. We can then find w with equation 25, where y_0 is the quantity h, and y becomes w.

$$w = h / (1 - c^2 / a^2) . \qquad (33)$$

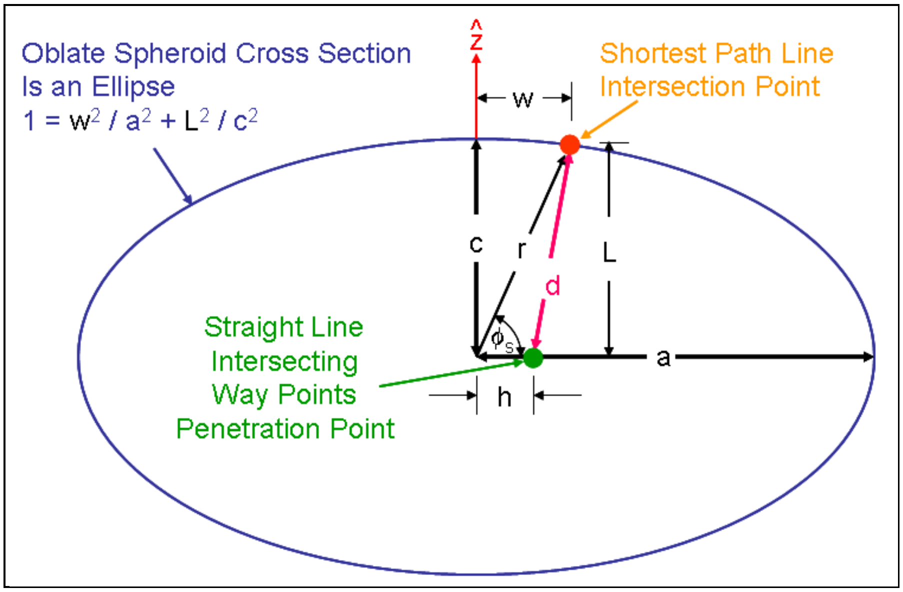


Figure 22. A more general cross section of figure 18.

In figure 22, d is the minimum distance between the straight line connecting the waypoints and the shortest path line intersection point, and r is the distance from the center point of the oblate spheroid to the ellipse. The quantity L is found from solving the equation for an ellipse using the known quantities w, a, and c.

$$L^2 = c^2 (1 - w^2 / a^2) . \qquad (34)$$

Finally, the value of ϕ_s is given as

$$\phi_s = \text{arc tan } (L / w) \,. \tag{35}$$

The final step to approximating the path between two nearly antipodal waypoints that are on the equator is to construct two great circles: one going from the point $(0,\theta_1)$ to (ϕ_s,θ_s) and one going from point (ϕ_s,θ_s) to $(0,\theta_2)$. The equation for the two segments takes on the form of equation 6, repeated here as equation 36 for convenience.

$$\tan \phi = \tan \phi_{max} \cos (\theta - \theta_0) \,. \tag{36}$$

Both segments will have the form of equation 36, but the values of ϕ_{max} and θ_0 will be different. For the segment going from $(0,\theta_1)$ to (ϕ_s,θ_s), the constants in equation 36 will be called $\phi_{max\text{-}1}$ and $\theta_{0\text{-}1}$, while the constants corresponding to the segment from (ϕ_s,θ_s) to $(0,\theta_2)$ will be called $\phi_{max\text{-}2}$ and $\theta_{0\text{-}2}$. To find the values, we take advantage of the fact that $(0,\theta_1)$ and $(0,\theta_2)$ are on the equator. Because of this, $\theta_{0\text{-}1}$ and $\theta_{0\text{-}2}$ will be 90° removed from $(0,\theta_1)$ and $(0,\theta_2)$. So,

$$\text{If } |\,\theta_2 - \theta_1\,| < 180° \text{ and } \theta_2 > \theta_1,\ \theta_{0\text{-}1} = \theta_1 + 90°$$

$$\text{If } |\,\theta_2 - \theta_1\,| < 180° \text{ and } \theta_2 < \theta_1,\ \theta_{0\text{-}1} = \theta_1 - 90°$$

$$\text{If } |\,\theta_2 - \theta_1\,| > 180° \text{ and } \theta_2 > \theta_1,\ \theta_{0\text{-}1} = \theta_1 - 90° \text{ and if } \theta_{0\text{-}1} < -180°,\ \theta_{0\text{-}1} = \theta_{0\text{-}1} + 360°.$$

$$\text{If } |\,\theta_2 - \theta_1\,| > 180° \text{ and } \theta_2 < \theta_1,\ \theta_{0\text{-}1} = \theta_1 + 90° \text{ and if } \theta_{0\text{-}1} > \text{or } 180°,\ \theta_{0\text{-}1} = \theta_{0\text{-}1} - 360°. \tag{37}$$

Having found the value of $\theta_{0\text{-}1}$, we can establish the value of $\phi_{max\text{-}1}$ by using equation 11.

$$\phi_{max\text{-}1} = \text{arc tan } [\tan \phi_s / \cos (\theta_s - \theta_{0\text{-}1})] \,. \tag{38}$$

A similar process is used to find $\theta_{0\text{-}2}$.

$$\text{If } |\,\theta_2 - \theta_1\,| < 180° \text{ and } \theta_2 > \theta_1,\ \theta_{0\text{-}2} = \theta_2 - 90°$$

$$\text{If } |\,\theta_2 - \theta_1\,| < 180° \text{ and } \theta_2 < \theta_1,\ \theta_{0\text{-}2} = \theta_2 + 90°$$

$$\text{If } |\,\theta_2 - \theta_1\,| > 180° \text{ and } \theta_2 > \theta_1,\ \theta_{0\text{-}2} = \theta_2 + 90° \text{ and if } \theta_{0\text{-}2} > 180°,\ \theta_{0\text{-}2} = \theta_{0\text{-}1} - 360°.$$

$$\text{If } |\,\theta_2 - \theta_1\,| > 180° \text{ and } \theta_2 < \theta_1,\ \theta_{0\text{-}2} = \theta_2 - 90° \text{ and if } \theta_{0\text{-}2} < \text{or } 180°,\ \theta_{0\text{-}2} = \theta_{0\text{-}2} + 360°. \tag{39}$$

As with $\phi_{max\text{-}1}$, $\phi_{max\text{-}2}$ is found using equation 11.

$$\phi_{max\text{-}2} = \text{arc tan } [\tan \phi_s / \cos (\theta_s - \theta_{0\text{-}2})] \,. \tag{40}$$

Note that $\phi_{max\text{-}1}$ and $\phi_{max\text{-}2}$ must always be kept in the range 0° to 90°.

The shortest line has now been approximated as two distinct great circle segments. One has the equation tan ϕ = tan ϕ_{max-1} cos ($\theta - \theta_{0-1}$), for the segment with θ between θ_{0-1} and θ_s, while the other has the equation tan ϕ = tan ϕ_{max-2} cos ($\theta - \theta_{0-2}$), for the segment with θ between θ_{0-2} and θ_s.

This may appear to be a crude approximation for the actual path connecting two nearly antipodal waypoints that lie on the equator. However, a previous investigation (*1*, pp 17–19, 27) discovered that navigation software provided by the U.S. Army Topographic Engineering Center called FORWARD and INVERSE (*16*) either could not calculate any path between two nearly antipodal waypoints on a WGS84 ellipsoid, or it predicted a path along the equator (*1*, pp 17–19, 27). Although this module uses a perfect sphere as its basis, it will be able to produce a path closer to the truly shortest path between nearly antipodal points.

3.4.9 Summary of Waypoint Placement Problems

Table 1 summarizes the difficulty of waypoint placement discussed in the previous sections. The difficulty arises when the waypoints are used to calculate the parameters ϕ_{max} and θ_0, which define the great circle. Methods to circumvent their difficulties (discussed in the previous sections) are also summarized in the table.

Table 1. Summary of waypoint placement problems and their solutions.

Waypoint Placement	Problem	Solution
Nearly antipodal on equator.	Path connecting them is nonequatorial; sphere model predicts equatorial path.	Calculate intermediate point with equations 30–35; use equations 36–40 to define two great circle connecting segments.
Waypoints at opposite poles.	Infinite number of meridians all equally valid.	Chose to use the prime meridian.
Waypoints antipodal.	Antipodal points do not define great circle.	Use great circle that passes through North Pole.
Waypoints on same or opposite meridian.	Equation 11 is undefined.	Use great circle that contains meridians.
One waypoint at the pole.	Equation 10 is undefined.	Use great circle that contains meridians.
Both waypoints on the equator.	Equation 10 is undefined.	Use the equator as great circle.
One waypoint at intersection of equator and prime meridian or international date line.	Equation 10 is undefined.	Assign $\theta_0 = \pm 90°$, use equation 11 or 14 to calculate ϕ_{max}, use value of θ_0, which gives $\phi_{max} > 0$.

3.5 Value of the Earth's Average Radius

In a previous work (*1*, pp 17–19, 27) it was established that modeling the Earth as a perfect sphere provides a basis for interwaypoint distance calculation. The distances calculated would be accurate enough to serve the needs of the software. So, the question arises as to how to establish the value of the radius of the perfect sphere.

Our approach is to pick a radius such that if a latitude and longitude were randomly chosen on a WGS84 oblate spheroid, sea level would have a 50% chance of being higher than the average radius and a 50% chance of being lower than the average radius. This entails adding every possible radius and dividing by the number of radii. Since there are an infinite number of radii in an oblate spheroid, we integrate all possible radii over all space, then divide by the integration over all space to define the average radius R_{ave} as where $r(\phi,\theta)$ is the radius of the latitude ϕ and longitude θ, and dS is the solid angle differential, also a function of latitude and longitude.

$$R_{ave} = \frac{\displaystyle\int_{\theta = -180^0}^{\theta = +180^0} \int_{\phi = -90^0}^{\phi = +90^0} r\,(\phi,\theta)\; dS\,(\phi,\theta)}{\displaystyle\int_{\theta = -180^0}^{\theta = +180^0} \int_{\phi = -90^0}^{\phi = +90^0} dS\,(\phi,\theta)} \quad . \tag{41}$$

The solid angle differential in spherical coordinates is written dS = cosϕ dϕ dθ (*17*). (Note that the reference's differential is written as dS = sin ϕ dϕ dθ. This is because the reference's angle ϕ is measured down from the North Pole. In our coordinate system, the reference angle ϕ is measured up from the equator.) To obtain the radius of an oblate spheroid, we consider the oblate spheroid's general equation in Cartesian coordinates (*18*), which is $1 = (x^2 + y^2) / a^2 + z^2 / c^2$, where a is the equatorial radius and c is the polar radius. Rewriting this in spherical coordinates using equations 1, 2, and 3, and using r (the dummy variable in lieu of R), we have

$$1 = (r^2 \cos^2 \theta \cos^2 \phi + r^2 \sin^2 \theta \cos^2 \phi) / a^2 + r^2 \sin^2 \theta / c^2 \, . \tag{42}$$

When we apply the trigonometric identity $\cos^2\theta + \sin^2\theta = 1$, solve for r, and change the integration limits to radians from degrees, using equation 42, equation 41 becomes

$$R_{ave} = \frac{\displaystyle\int_{\theta = -\pi}^{\theta = +\pi} \int_{\phi = -\pi/2}^{\phi = +\pi/2} \frac{\cos\phi\; d\phi\; d\theta}{\sqrt{\dfrac{\cos^2\phi}{a^2} + \dfrac{\sin^2\phi}{c^2}}}}{\displaystyle\int_{\theta = -\pi}^{\theta = +\pi} \int_{\phi = -\pi/2}^{\phi = +\pi/2} \cos\phi\; d\phi\; d\theta} \quad . \tag{43}$$

Evaluating the denominator is straightforward: the value is 4π. Evaluating the variable θ in the numerator is also straightforward: the value is 2π. This is because an oblate spheroid is symmetric about the z axis, making it independent of θ. With those variables evaluated, equation 43 then becomes

$$R_{ave} = \frac{1}{2} \int_{\phi = -\pi/2}^{\phi = +\pi/2} \frac{\cos \phi \, d\phi}{\sqrt{\frac{\cos^2 \phi}{a^2} + \frac{\sin^2 \phi}{c^2}}} \quad . \tag{44}$$

To make the integral wieldy, a variable substitution is made such that u = sin ϕ. Taking the differential du = cos ϕ dϕ, changing the limits, and doing some algebraic manipulation renders the average radius R_{ave} as

$$R_{ave} = \frac{ac}{2\sqrt{a^2 - c^2}} \int_{u = -1}^{u = 1} \frac{du}{\sqrt{\frac{c^2}{a^2 - c^2} + u^2}} \quad . \tag{45}$$

Evaluating this integral (*19*, *20*), applying the limits of integration, and performing some algebraic manipulation render the average radius as

$$R_{ave} = \frac{ac}{2\sqrt{a^2 - c^2}} \ln \left(\frac{a + \sqrt{a^2 - c^2}}{a - \sqrt{a^2 - c^2}} \right) = \frac{ac}{\sqrt{a^2 - c^2}} \text{ Arc Tanh } \sqrt{1 - c^2 / a^2} \tag{46}$$

because the equatorial radius a is greater than the polar radius c. After applying the WGS84 values for the equatorial radius a and the polar radius c as 6,378,137.0 m, and c = 6,356,752.314245 m, respectively (*13*), the value of R_{ave} is 6370994.40182752 m, which varies by <0.0008% from the value offered by the Chemical Rubber Handbook (*21*), which was not derived using the WGS84 oblate spheroid model.

4. Results

The formulae derived in spherical coordinates from the analysis of section 3 is next applied to obtaining the results of algorithms and formulae needed to describe platform motion. After a presentation of terms used to describe moving platforms, the equations of motion are derived assuming constant acceleration and constant deceleration up to and down from the platform’s cruising velocity. Next, an algorithm is presented for adding, removing, and modifying waypoints. Finally, assuming that the snapshot times are known, the location of the platform between waypoints at the snapshot times is obtained.

4.1 Definition of Moving Platform Variables

For all moving platforms, we assume that the velocity vs. time profile takes a consistent structure between two waypoints. The relevant variables and constants used to describe the profile are displayed in figure 23. As a platform approaches the jth waypoint (labeled “Waypoint j:” in the figure), it is traveling at a constant maximum cruise velocity, v_{cru}. This maximum cruise velocity is a function of the kind of platform being modeled (high-mobility multipurpose wheeled vehicle [HMMWV] vs. tank vs. unmanned aerial vehicle [UAV] vs. dismounted Soldier, etc.) and the kind of terrain the platform is crossing (paved road vs. unpaved road vs. smooth terrain vs. rocky terrain, etc.) assuming that it is a ground platform. The value for the v_{cru} is read from a database of platform performance specifications provided by Army development centers. If the platform is part of a squad or convoy, v_{cru} can be assigned the value of the maximum cruising velocity of the slowest vehicle in the squad or convoy to ensure that the platforms remain in formation.

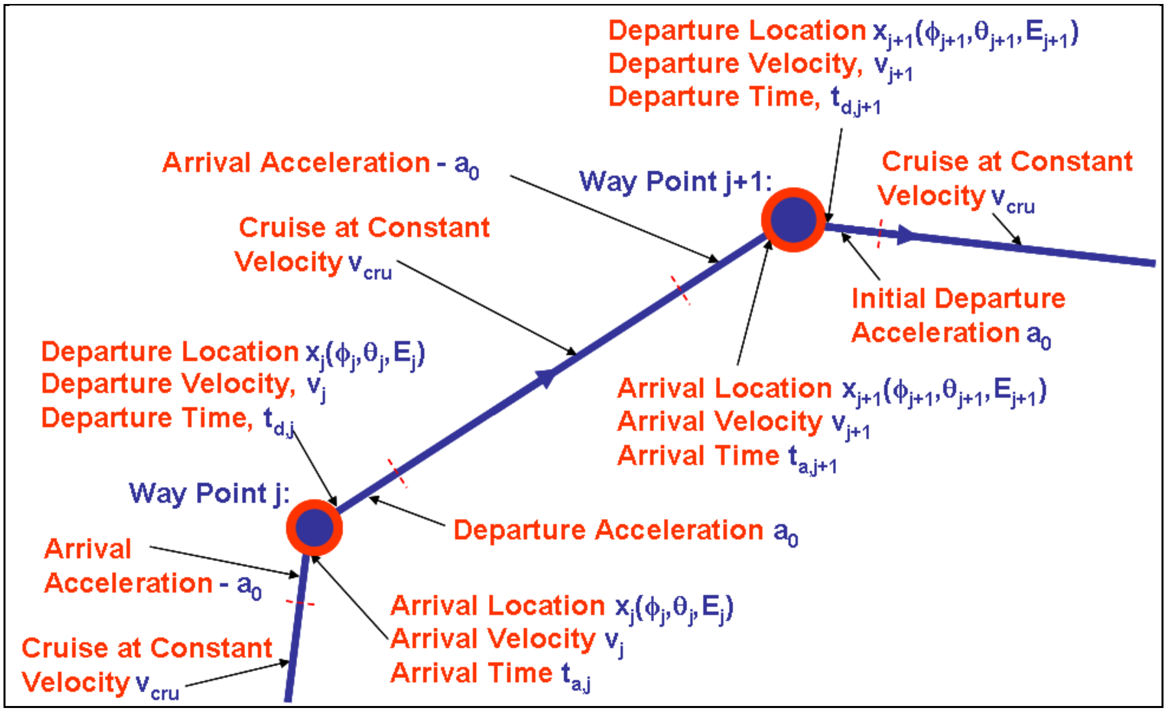

Departure Location $x_{j+1}(\phi_{j+1},\theta_{j+1},E_{j+1})$
Departure Velocity, v_{j+1}
Departure Time, $t_{d,j+1}$
Arrival Acceleration - a_0
Way Point j+1:
Cruise at Constant Velocity v_{cru}
Cruise at Constant Velocity v_{cru}
Initial Departure Acceleration a_0
Departure Location $x_j(\phi_j,\theta_j,E_j)$
Departure Velocity, v_j
Departure Time, $t_{d,j}$
Arrival Location $x_{j+1}(\phi_{j+1},\theta_{j+1},E_{j+1})$
Arrival Velocity v_{j+1}
Arrival Time $t_{a,j+1}$
Way Point j:
Departure Acceleration a_0
Arrival Acceleration - a_0
Arrival Location $x_j(\phi_j,\theta_j,E_j)$
Arrival Velocity v_j
Arrival Time $t_{a,j}$
Cruise at Constant Velocity v_{cru}

Figure 23. Variables that describe moving platforms as they travel between waypoints.

Note that v_{cru} is not the maximum velocity of which the platform is capable; rather, it is a typical speed the platform obtains when it is in travel mode. For example, for a dismounted Soldier v_{cru} might be the brisk walking pace of 3 mph, even though a Soldier doing double time can travel at nearly twice that rate. We assign the value that is typical of a dismounted Soldier in travel mode.

When the platform approaches the jth waypoint, it begins to slow down with an acceleration of $-a_0$. The minus sign ensures that a_0 is assigned a positive value. This value, like v_{cru}, is also a function of the kind of platform as well as the terrain the platform crosses. It, too, will be read from a database with information supplied by Army development centers. And as with the variable v_{cru}, this will not be the maximum value of which the platform is capable but rather a typical value for the platform.

When the platform arrives at the jth waypoint, it will do so with a velocity of v_j at the arrival time $t_{a,j}$. The waypoint location is specified in latitude (ϕ_j), longitude (θ_j), and elevation (E_j). The elevation is the distance above the average sea level for a perfectly spherical Earth, which is the average radius R_{ave} from the Earth's center point. The radius (notated as "R" in figure 2) for the jth waypoint is $R_{ave} + E_j$. The latitude and longitude are specified by the user, while the elevation E_j is the elevation of the ground on which the platform is traveling (from a database with information provided by the Digital Terrain Elevation Data [DTED] [*22*]) plus the antenna height above the ground. If the platform is not a ground platform, the elevation E_j is specified by the user.

When the platform departs the jth waypoint for the j+1th waypoint, it will begin to accelerate with the value $+a_0$ at the departure time $t_{d,j}$. This is the same value with which it decelerated before reaching the waypoint, only in the opposite direction of the deceleration $-a_0$. (The orange hash in the figure that crosses the blue path is the point where the platform begins accelerating or decelerating.) So the value for a_0 must be chosen to represent the platform's normal acceleration and deceleration. The departure velocity and the arrival velocity at the jth waypoint must be the same, v_j. If the platform does not stop at the waypoint but simply passes through it, then it leaves the waypoint with the same speed with which it arrived. If it does stop, then it must leave the waypoint from a stop, so the arrival and departure speeds are zero. If the platform does not stop at the waypoint, then the arrival time $t_{a,j}$ and departure time $t_{d,j}$ must also be the same. If the platform does stop, then it is possible for the platform to loiter at the waypoint for a waiting, or loiter, time $t_{l,j} = t_{d,j} - t_{a,j}$.

The platform accelerates until it reaches the cruise velocity v_{cru}. It then stops accelerating, traveling at a constant speed of v_{cru}. When it approaches the j+1th waypoint, it begins to decelerate at $-a_0$, until it reaches the j+1th waypoint with speed v_{j+1} at time $t_{j+1,a}$. As with the jth waypoint, if the platform has a speed of zero when it reaches the j+1th waypoint, it may loiter for time $t_{l,j+1} = t_{d,j+1} - t_{a,j+1}$ before it leaves the waypoint at time $t_{d,j+1}$ when it accelerates at a_0 to reach v_{cru} on its way to the j+2th waypoint.

The path the platform travels between two waypoints is the most expeditious path between two points on a perfect sphere: an arc of a great circle (*23*). As the platform passes the waypoint, it changes direction instantaneously so as to follow the great circle path to the next waypoint. Because the direction changes, the platform's velocity changes instantly. Yet, in an attempt to maintain a greater degree of realism in modeling platform motion, the platform's speed is treated as constant as it passes through a waypoint.

4.1.1 Mathematical Description of Platform Motion

Figure 24 shows graphs of acceleration, speed, and distance vs. time of platform motion between waypoints. To keep the motion as simple as possible, we define the acceleration graph as a simple step function: a_0 at the departure time for waypoint j $t_{d,j}$. At time t_1, it becomes 0, and at t_2 it becomes $-a_0$, until it arrives at waypoint j+1 at time $t_{a,j+1}$. At all waypoints, the acceleration is 0.

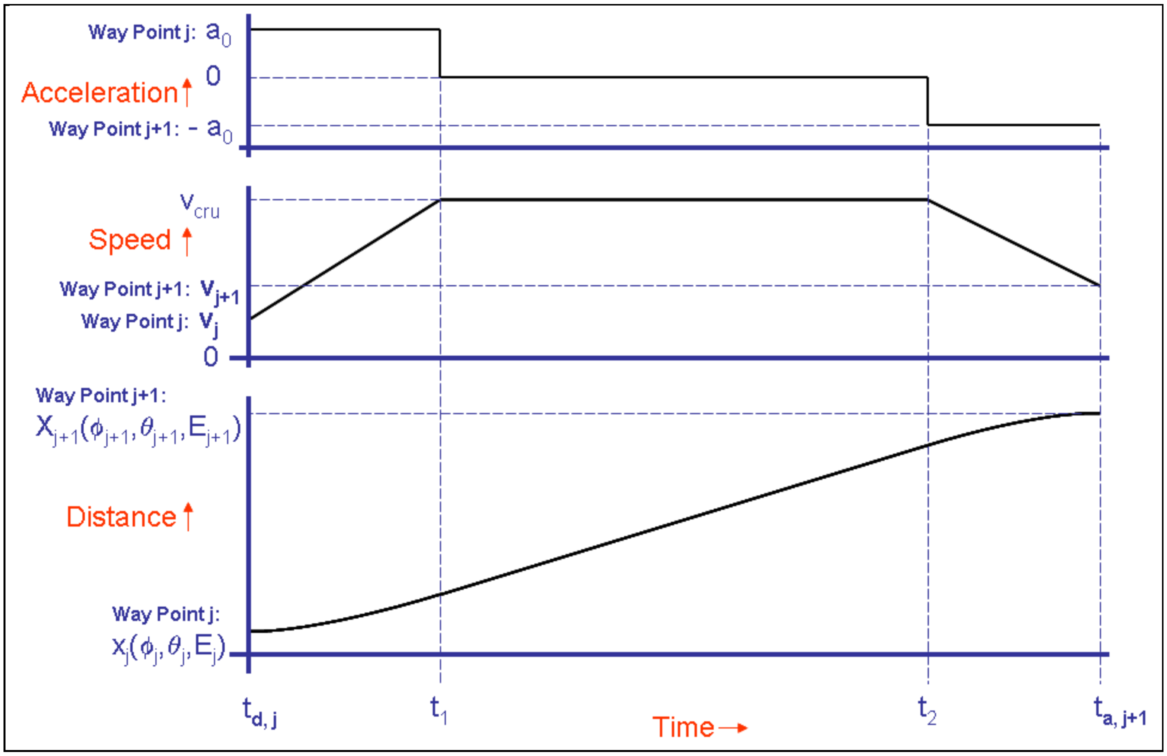


Figure 24. Motion profile for moving platform between waypoints.

Integrating the acceleration graph yields the speed vs. time graph. The speed begins at v_j at waypoint j, then increases at a constant acceleration rate of a_0 until time t_1 when the acceleration goes to zero, and the speed reaches v_{cru}. The speed remains at v_{cru} until time t_2, when it decreases at a deceleration rate of $-a_0$ to become v_{j+1} at time $t_{a,j+1}$.

To find the distance covered, we integrate the speed graph in time. At time $t_{d,j}$ the platform moves away from the waypoint $x(\phi_j,\theta_j,E_j)$, such that the distance increases with time and the rate of distance change increases in time until time t_1. From time t_1 to time t_2, the distance increases at a constant rate in time. From time t_2 until the platform reaches waypoint j+1 at time $t_{a,j+1}$, the distance is still increasing in time but at a decreasing rate.

4.1.2 Mathematical Relationship Between the Motion-Defining Variables

To begin, we define the acceleration of the platform in terms of the time variable t according to the graph in figure 24.

$$\text{At } t = t_{d,j},\ a(t) = 0\ . \tag{47}$$

$$\text{At } t_{d,j} < t < t_1,\ a(t) = a_0\ . \tag{48}$$

$$\text{At } t = t_1,\ a(t) = a_0\ . \tag{49}$$

$$\text{At } t_1 < t < t_2,\ a(t) = 0\ . \tag{50}$$

$$\text{At } t = t_2,\ a(t) = -a_0\ . \tag{51}$$

$$\text{At } t_2 < t < t_{a,j+1},\ a(t) = -a_0\ . \tag{52}$$

$$\text{At } t = t_{a,j+1},\ a(t) = 0\ . \tag{53}$$

To find the speed dependence in time, we integrate equations 48, 50, and 52 in time. The constant of integration will be determined by making the speed at times $t_{d,j}$ and $t_{a,j+1}$ equal to v_j and v_{j+1}, respectively. We will also use the fact that the speed from figure 24 shows that from (and including) times t_1 to t_2, the speed is v_{cru}. We will also assume that the speed is continuous for all time. From the boundary condition we know that

$$\text{at } t = t_{d,j}\ \ v(t) = v_j\ . \tag{54}$$

Integrating equation 48 in time, we know that

$$\text{at } t_{d,j} < t < t_1,\ v(t) = a_0\, t + C\ , \tag{55}$$

where C is the constant of integration. Applying the boundary condition of equation 54, we evaluate C, and with some algebra we find that

$$\text{at } t_{d,j} < t < t_1,\ v(t) = a_0\ (\ t - t_{d,j}) + v_j\ . \tag{56}$$

Evaluating the expression at the first inflection point t_1, we see that the speed should be v_{cru} from figure 24, so

$$\text{at } t = t_1,\ v(t) = a_0\ (\ t_1 - t_{d,j}) + v_j = v_{cru}\ . \tag{57}$$

Between the inflection times t_1 and t_2, the speed remains constant at v_{cru}, so that

$$\text{at } t_1 < t < t_2,\ v(t) = v_{cru}\ , \tag{58}$$

and it follows from figure 24 that

$$\text{at } t = t_2,\ v(t) = v_{cru}\ , \tag{59}$$

giving us another boundary condition. Integrating equation 47 in the time window $t_2 < t < t_{a,j+1}$, we find that

$$\text{at } t_2 < t < t_{a,j+1},\ v(t) = -a_0\, t + C\ , \tag{60}$$

where applying equation 59 to evaluate the integration constant C and rearranging the variable shows us that

$$\text{at } t_2 < t < t_{a,j+1},\ v(t) = v_{cru} - a_0\ (\ t - t_2\ )\ . \tag{61}$$

To complete the speed profile in time between the waypoints, we use the speed at the j+1th waypoint to find that

$$\text{at } t = t_{a,j+1},\ v(t) = v_{cru} - a_0\ (\ t_{a,j+1} - t_2\ ) = v_{j+1}\ . \tag{62}$$

To obtain the distance D(t) traveled by the platform from waypoint j as a function of time, equations 56, 58, and 61 are integrated in time. The result for equation 56's integration is

$$\text{at } t_{d,j} < t < t_1,\ D(t) = a_0\, t^2 / 2 - a_0\, t_{d,j}\, t + v_j\, t\ + C\ . \tag{63}$$

The value of the integration constant C is found by knowing that D(t) = 0 at time $t = t_{d,j}$.

Solving for C and rearranging the variables, we find

$$\text{at } t_{d,j} < t < t_1,\ D(t) = a_0\ (\ t^2 - t_{d,j}^{\ 2}\ )/\ 2 - (\ a_0\, t_{d,j}\ - v_j\ )\ (\ t - t_{d,j}\ )\ . \tag{64}$$

The distance the platform obtains from the waypoint j at the inflection time t_1 is found by applying t_1 to the variable t in equation 64, where

$$\text{at } t = t_1,\ D(t) = a_0\ (\ t_1^{\ 2} - t_{d,j}^{\ 2}\ )/\ 2 - (\ a_0\, t_{d,j}\ - v_j\ )\ (\ t_1 - t_{d,j}\ )\ . \tag{65}$$

Distance D(t) must increase from the value outlined in equation 65. It will increase with the constant speed v_{cru} as shown in equation 58. So, the integration of equation 58 is added to equation 65 so that in the next time segment, where $t_1 < t < t_2$, the distance formula is given

$$\text{at } t_1 < t < t_2,\ D(t) = a_0\ (\ t_1^{\ 2} - t_{d,j}^{\ 2}\ )/\ 2 - (\ a_0\, t_{d,j}\ - v_j\ )\ (\ t_1 - t_{d,j}\ ) + v_{cru}\, t + C\ . \tag{66}$$

To find the integration constant C, we use the fact that at time t_1, equation 66 equals equation 64. The result is that

$$\text{at } t_1 < t < t_2,\ D(t) = a_0\ (\ t_1^{\ 2} - t_{d,j}^{\ 2}\ )/\ 2 - (\ a_0\, t_{d,j}\ - v_j\ )\ (\ t_1 - t_{d,j}\ ) + v_{cru}\ (t - t_1\ )\ . \tag{67}$$

The distance of the platform D(t) from waypoint j at the inflection time t_2 is

$$\text{at } t = t_2,\ D(t) = a_0\ (\ t_1^{\ 2} - t_{d,j}^{\ 2}\ )/\ 2 - (\ a_0\, t_{d,j}\ - v_j\ )\ (\ t_1 - t_{d,j}\ ) + v_{cru}\ (t_2 - t_1\ )\ . \tag{68}$$

The distance will increase from the point where the platform is located at time t_2. The rate at which the distance D(t) will increase in time is outlined in equation 61, so that adding the integration of equation 61 shows that

$$\text{at } t_2 < t < t_{a,j+1},\ D(t) = a_0 (t_1^2 - t_{d,j}^2)/ 2 - (a_0 t_{d,j} - v_j) (t_1 - t_{d,j}) + v_{cru} (t_2 - t_1) + v_{cru} t - a_0 (t^2 - 2 t_2 t) / 2 + C . \qquad (69)$$

Evaluation of the integration constant C is accomplished by equating equation 69 with equation 68 for time $t = t_2$, so the result is that

$$\text{at } t_2 < t < t_{a,j+1},\ D(t) = a_0 (t_1^2 - t_{d,j}^2)/ 2 - (a_0 t_{d,j} - v_j) (t_1 - t_{d,j}) + v_{cru} (t_2 - t_1) + v_{cru} (t - t_2) - a_0 (t^2 - t_2^2) / 2 + a_0 t_2 (t - t_2) . \qquad (70)$$

At time $t = t_{a,j+1}$, equation 70 gives the distance between waypoints j and j+1. For this reason, we notate the distance between them as Dj, evaluate equation 70 making $t = t_{a,j+1}$, and gather some terms together to simplify the equation. After rearranging some of the variables, we find that

$$\text{at } t = t_{a,j+1},\ D_j = a_0 (t_1 - t_{d,j})^2 / 2 + v_j (t_1 - t_{d,j}) + v_{cru} (t_{a,j+1} - t_1) - a_0 (t_{a,j+1} - t_2)^2 / 2 . \qquad (71)$$

Bringing back equation 57 and equation 62, and after applying some algebra to isolate and number the time variables, equations 71–73 are three independent equations showing the relationship between nine variables.

$$t_1 - t_{d,j}= (v_{cru} - v_j) / a_0 . \qquad (72)$$

$$t_{a,j+1} - t_2 = (v_{cru} - v_{j+1}) / a_0 . \qquad (73)$$

By combining equations 71–73, it is possible to eliminate the inflection times t_1 and t_2, which delineate the times when the platform will change speed. Doing so produces an equation that defines the distance between the jth and j+1th waypoint D_j in terms of the departure time of the jth waypoint $t_{d,j}$, the arrival time at the j+1th waypoint $t_{a,j+1}$, the platform speed at the jth waypoint v_j, the platform speed at the j+1th waypoint v_{j+1}, the platform cruising speed between waypoints v_{cru}, and the platform acceleration/deceleration rate a_0. This is equation 74.

$$D_j = v_{cru} (t_{a,j+1} - t_{d,j}) - (v_{cru} - v_{j+1})^2 / (2 a_0) - (v_{cru} - v_j)^2 / (2 a_0) . \qquad (74)$$

4.1.3 Restrictions on Input Data – General

When the user specifies the location of the j = 1st waypoint, the latitude and longitude will need to be input. The latitude can range from –180° to +180°, while the longitude ranges from –90° to +90°. If the waypoint is for a ground platform, the elevation is read from software containing elevation data, like the DTED database (*22*). If the platform is not a ground platform, the user inputs the elevation data, restricted by the ground elevation, and the platform's service ceiling, which is read from a database of platform specifications. By default, the corresponding arrival time $t_{a,1}$ will be set to 0 s, since this is the beginning of the simulation. Also by default, the platform speed will be set to 0 m/s if it is a ground platform or the minimum air speed if it is an air platform. The minimum air speed will be read from a database with specification data for the platform. The user will then set the loiter time, defining the departure time for the first waypoint $t_{d,1}$, which, for $j = 1$, will be equal to the loiter time. The cruise speed v_{cru} and the acceleration a_0 are read from the platform specification database.

Next, the user inputs the latitude and longitude for the second waypoint. The elevation is also input if the platform is an air platform, otherwise the elevation is read from a DTED-like database. This defines the distance between the waypoints D_1. Consulting equation 74 for $j = 1$, all variables but two have been defined: $t_{a,2}$ and v_2. If the user defines the time $t_{a,2}$, then waypoint speed v_2 is also defined. If the user defines the waypoint speed v_2, then by equation 74, $t_{a,2}$ is also defined. If the user specifies a loiter time at waypoint two (i.e., making it so that $t_{a,2}$ is not equal to $t_{d,2}$), then v_2 must be equal to 0 m/s. The result in that case is that $t_{a,2}$ will be calculated according to equation 74, independent of any user desire.

In the general case (i.e., for $j > 1$), the distance D_j is defined when the user inputs the waypoint latitudes, longitudes, and elevations (or the elevations are read from an elevation database). v_{cru} is also read from a platform specification database, as is a_0. t_j and v_j were defined when the calculations for the last waypoints were done, leaving the user free to choose either $t_{a,j+1}$, which defines v_{j+1} by equation 74, or v_{j+1}, which defines $t_{a,j+1}$. Should the user decide that the platform loiter at the j+1th waypoint, then v_{j+1} must be zero, defining the arrival time $t_{a,j+1}$ and giving the user no flexibility insofar as having variables to define.

Equation 74 is rewritten to show the arrival time and the speed at the j+1th waypoint as a function of the other variables. Equation 73 was obtained by simply solving equation 75 for $t_{a,j+1}$.

$$t_{a,j+1} = t_{d,j} \;\; + [\; 2\, a_0\, D_j + (v_{cru} - v_{j+1})^2 + (v_{cru} - v_j)^2 \;] \,/\, (2\, a_0\, v_{cru}\,) \,. \tag{75}$$

Equation 76 was derived by solving equation 74 for the speed at the j+1th waypoint. When the variable v_{j+1} was isolated in equation 74, it became evident that there were two possible solutions for v_{j+1}, as it was necessary to take a square root to find v_{j+1}. The two solutions found were v_{cru} plus an expression under the radical and v_{cru} minus an expression under the radical. We take the minus case because the radical's value is always positive, since a platform's speed cannot exceed v_{cru}. Once the speed at the j+1th waypoint is calculated with equation 76,

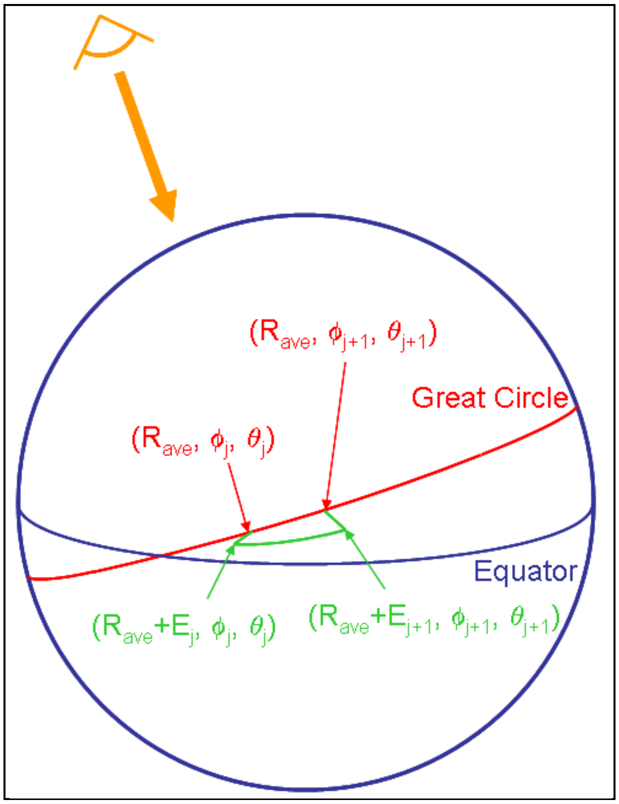
$$v_{j+1} = v_{cru} - \sqrt{2\, a_0\, [\; v_{cru}\, (t_{a,j+1} - t_{d,j}\,) - D_j\,] - (\,v_{cru} - v_j\,)^2} \;\;, \tag{76}$$

and the j+1th waypoint arrival time $t_{a,j+1}$ is established, equations 72 and 73 can be used to find the inflection times t_1 and t_2.

When the last waypoint is assigned, we assume that the speed of the platform becomes zero at the final waypoint. Hence, the user has the freedom to assign the location of the final waypoint, while the arrival time is determined by equation 75 for the final point. Should the simulation stop before the platform reaches the final waypoint, the final time is taken as an interpolation point, and the final latitude, longitude, and elevation are calculated, treating the final time like an interpolation time. Should the simulation end time occur after the time the platform reaches the final waypoint, the platform will simply treat the time between the arrival time at the final waypoint $t_{a,j+1}$ and the simulation end time as a loiter time at the waypoint.

4.1.4 Distance Calculation

Using the spherical model, we next calculate the distance between two points notated $(R_{ave}+E_j,\phi_j,\theta_j)$ and $(R_{ave}+E_{j+1},\phi_{j+1},\theta_{j+1})$. R_{ave} is the average radius of the Earth; E_j, ϕ_j, and θ_j are the jth's waypoint elevation, latitude, and longitude; and E_{j+1}, ϕ_{j+1}, and θ_{j+1} are the j+1th's waypoint elevation, latitude, and longitude. The points are shown with the great circle they define, and a wedge containing them and the center point in figure 25. The equator is in blue, the great circle in red, and the wedge formed by the waypoints and the Earth's center point is shown in green. Note that the waypoints are not on the great circle but are elevated above it by E_j and E_{j+1}, respectively.



$(R_{ave}, \phi_{j+1}, \theta_{j+1})$
Great Circle
$(R_{ave}, \phi_j, \theta_j)$
Equator
$(R_{ave}+E_j, \phi_j, \theta_j)$
$(R_{ave}+E_{j+1}, \phi_{j+1}, \theta_{j+1})$

Figure 25. Two waypoints and the Earth's center point form a wedge.

To find the distance between the waypoints, we must remove the sphere and take a vantage point normal to the wedge formed by the two waypoints and the Earth's center. This is shown in figure 26. R_{ave} is the Earth's average sea level radius, E_j and E_{j+1} are the jth and j+1th waypoint elevations, ϕ_j and ϕ_{j+1} are the latitudes of the waypoints, and θ_j and θ_{j+1} are the longitudes. $\Delta\theta$ is the angle formed at the Earth's center point between the lines connecting the waypoints with the center point. D_j is the distance along the platform's path between the jth and j+1th waypoint, while Δ is the line-of-sight distance between the waypoint projections on the sea level surface of the Earth.

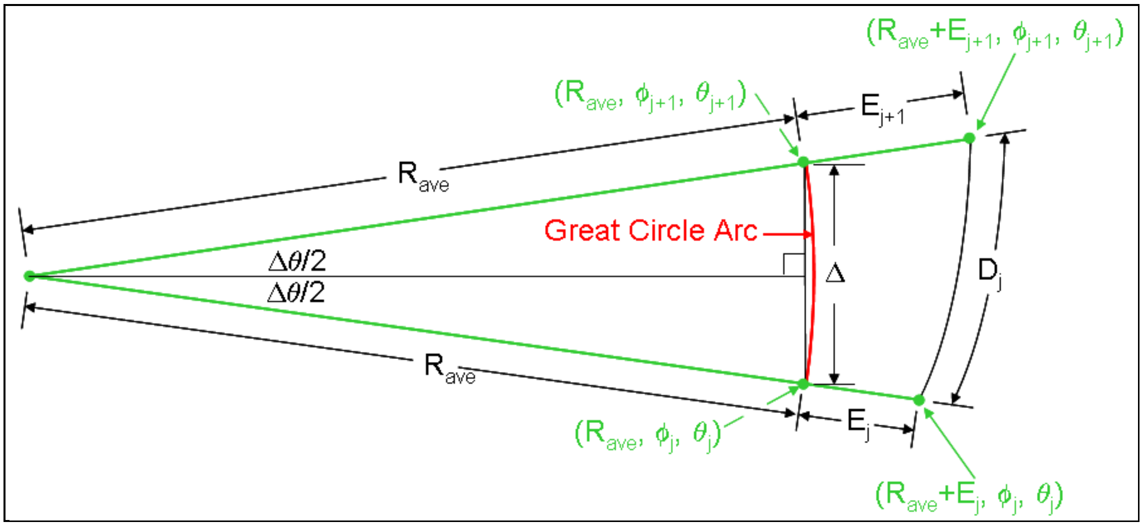

$(R_{ave}+E_{j+1}, \phi_{j+1}, \theta_{j+1})$
$(R_{ave}, \phi_{j+1}, \theta_{j+1})$
E_{j+1}
R_{ave}
Great Circle Arc
$\Delta\theta/2$
$\Delta\theta/2$
Δ
D_j
R_{ave}
$(R_{ave}, \phi_j, \theta_j)$
E_j
$(R_{ave}+E_j, \phi_j, \theta_j)$

Figure 26. The wedge from figure 25 with the Earth removed for clarity.

First, we find the relationship between the angle $\Delta\theta$ and the line-of-sight distance Δ. From the two back-to-back right triangles, we see that

$$\Delta / \ R_{ave} = 2 \sin (\Delta\theta / 2) . \tag{77}$$

Written another way, equation 77 becomes

$$\Delta\theta = 2 \arcsin [\Delta / (2 R_{ave})] . \tag{78}$$

R_{ave} is known but Δ is not. To find Δ we employ the well-known distance formula in a 3-D, rectilinear coordinate system (*24*).

$$\Delta = \sqrt{(x_{j+1} - x_j)^2 + (y_{j+1} - y_j)^2 + (z_{j+1} - z_j)^2} \ . \tag{79}$$

Substituting equations 1, 2, and 3, into 79, then employing equation 78, the term R_{ave} cancels to reveal $\Delta\theta$ in terms of the latitudes and longitudes of the two waypoints.

$$\Delta\theta = 2 \arcsin \{ [(1 / 2) [(\cos \phi_{j+1} \cos \theta_{j+1} - \cos\phi_j \cos \theta_j)^2 + (\cos \phi_{j+1} \sin \theta_{j+1} - \cos \phi_j \sin \theta_j)^2 + (\sin \phi_{j+1} - \sin \phi_j)^2]^{1/2} \} . \tag{80}$$

Note that the sign of $\Delta\theta$ is positive because we are taking the positive root of equation 78.

Next, to find the arc length D_j we employ the definition of arc length in planer motion (*25*), which is appropriate to figure 26 because the wedge is contained in a single plane.

$$D_j = \int_0^{\Delta\theta} \sqrt{\left(\frac{dR}{d\theta}\right)^2 + R(\theta)^2} \; d\theta \quad \cdot \tag{81}$$

We rewrite equation 81 so that the distance becomes S, a continuous variable as a function of the continuous variable θ. We keep the variable R to be the distance from the Earth's center point to the platform. So, for any point on the path S the platform traverses, the angle it makes with the previous waypoint θ, as measured at the Earth's center point, is

$$S = \int_0^{\theta} \sqrt{\left(\frac{dR}{d\theta}\right)^2 + R(\theta)^2} \; d\theta \quad \cdot \tag{82}$$

We know that the elevation at a waypoint E_j is related to the distance from the Earth's center point R_j in that

$$R_j = R_{ave} + E_j \,. \tag{83}$$

We now use the variables E and R to represent the continuous elevation and continuous distance of the platform from the Earth's center point on the path between waypoints. Updating equation 83, we see that

$$R = R_{ave} + E \,. \tag{84}$$

Differentiating equation 84 in time, it becomes self-evident that the rate of change of elevation dE/dt is equal to the rate of change of radius dR/dt, as stated in equation 83.

$$dR / dt = dE / dt \,. \tag{85}$$

Now we establish the relationship between the change in time of path length S and the change in time of the distance of the platform from the Earth's center point R. For an aerial platform like a fixed wing aircraft or a helicopter, we assume that the rate of climb or descent is directly proportional to its airspeed along the path S. For a ground platform, the equivalent assumption is that the elevation change is proportional to the distance traversed. Put another way, we assume that the slope of the ground between waypoints is constant. Expressed quantitatively, in both the aerial platform and the ground platform case,

$$dE / dt = dR / dt = k\ dS / dt\ , \qquad (86)$$

where k is a constant whose value will be determined in due course.

Next, we wish to establish the relationship between the platform's distance from the Earth's center R, the path length between waypoints S, and the angle the platform makes at the Earth's center between the platform's location and the previous or jth waypoint. Multiplying equation 86 by the time differential dt and integrating, we see that

$$R = k\ S + C\ , \qquad (87)$$

where C is the constant of integration. To evaluate it, we know that when S = 0, the platform is at the jth waypoint, so the distance from the Earth's center to the waypoint will be $R_j = R_{ave} + E_j$. This allows us to evaluate the integration constant in equation 87. After the constant's evaluation, equation 87 is rendered

$$R = k\ S + R_j = k\ S + R_{ave} + E_j\ . \qquad (88)$$

To evaluate k, it will be necessary to differentiate equation 88 by θ.

$$dR / d\theta = k\ dS / d\theta\ . \qquad (89)$$

Note that the differential of the constant terms is zero. Next, we define the antideferential F(θ) such that $dF(\theta) / d\theta = [\ (dR / d\theta\ )^2 + R^2\ ]^{1/2}$. Rewriting equation 82 in terms of F(θ), we arrive at the distance traveled from the last waypoint to the platform.

$$S = F(\theta) - F(0)\ . \qquad (90)$$

Differentiating equation 90 and substituting for the term dS / dθ from equation 89 (keeping in mind that F(0) is a constant, so its derivative is zero), we find

$$dR / d\theta\ =\ k\ \ [\ (dR / d\theta\ )^2 + R^2\ ]^{1/2}. \qquad (91)$$

Squaring equation 91 and grouping the differentials together, then taking the square root, we arrive at a conventional first-order differential equation.

$$(dR / R) = \pm\ d\theta\ \ k / (1 - k^2)^{1/2}\ . \qquad (92)$$

As the platform moves, the value of θ increases. If the elevation increases, then the value of k must be positive in equation 92. If the elevation decreases, then the value of k must be negative in equation 92, since we assume that the quantity $(1 - k^2)^{1/2}$ is always positive. Consider the constant k as listed in equation 88. If the value of the elevation of the platform increases with distance, k is positive. Likewise, as it decreases with distance, k is negative. So, we will simply let the sign of k change to indicate the elevation change of the platform: if k is positive, the platform is ascending. If k is negative, the platform is descending. To maintain consistency in equation 92, we take the positive root.

Solving equation 92 and taking the natural antilogarithm, the equation becomes

$$R = C \exp [\theta k / (1 - k^2)^{1/2}] , \tag{93}$$

where R is the distance of the platform to the Earth's center point, θ is the angular distance along the path, exp [ ] signifies the value e (roughly 2.718281828459 [*26*]) raised to the power of the expression inside the brackets, and C is the constant of integration. To find C, we use initial conditions: when θ = 0, the right half of equation 93 becomes C, which is equal to the left half, which is $R = R_1 = R_{ave} + E_1$.

To find the value of k, we evaluate equation 93 when the platform reaches the j+1th point and equation 80 to find the angular distance between the waypoints.

$$R_{j+1} = R_{ave} + E_{j+1}= R_j \exp [\Delta\theta k / (1 - k^2)^{1/2}] = (R_{ave} + E_j) \exp [\Delta\theta k / (1 - k^2)^{1/2}] . \tag{94}$$

Taking the natural logarithm of equation 94 and solving for k, we find that

$$k = \ln [(R_{ave} + E_{j+1}) / (R_{ave} + E_j)] / \{ \Delta\theta^2 + \{ \ln [(R_{ave} + E_{j+1}) / (R_{ave} + E_j)]\}^2 \}^{1/2} , \tag{95}$$

transforming k to a known value. Next, we incorporate equation 88 and use the values of the variables when the platform reaches the j+1th waypoint.

$$R_{j+1} = R_{ave} + E_{j+1} = k D_j + R_j = k Dj + R_{ave} + E_j . \tag{96}$$

After solving for the distance the platform travels between the waypoints D_j and eliminating the average radius of the Earth, we find the distance in terms of k and the elevations at each waypoint.

$$D_j = (E_{j+1} - Ej) / k . \tag{97}$$

Combining equations 94 and 96 and eliminating R_{j+1}, the radius at the j+1th waypoint, and algebraically rearranging some variables, it is possible to find an alternative expression for the distance the platform travels.

$$D_j = (R_j / k) \{\exp [\Delta\theta k / (1 - k^2)^{1/2}] - 1\} =$$
$$[(R_{ave} + Ej)/ k] \{\exp [\Delta\theta k / (1 - k^2)^{1/2}] - 1\} . \tag{98}$$

The constant k denotes the change of elevation divided by the distance traveled. So, the possible ranges of k can be expressed as $-1 \le k \le 1$. Equation 98 becomes undefined if $k = \pm 1$, but equation 97 will still produce a finite value for D_j. However, both equations 97 and 98 are undefined if k = 0. It is possible to take the limit as k approaches zero in equation 98 and use the result to compute D_j. We find

$$D_j = R_j \Delta\theta = (R_{ave} + E_j) \Delta\theta \tag{99}$$

consistent with the formula for the length of a circle's arc (*27*)—the radius times the angle in radians. Since k = 0, the platform does not change altitude, making the path the arc of a circle. Having established the platform path distance D_j, it is now possible to calculate the arrival time $t_{a,j+1}$ using equation 75 or the arrival velocity v_{j+1} using equation 76.

4.1.5 Restrictions on Input Data – Maximum Speed, Minimum Time

The maximum speed a platform can attain between waypoints is v_{cru}, the cruising speed. To arrive at the j+1th waypoint in the minimum amount of time, the speed profile would have to be as shown in figure 27. The profile differs from the one displayed in figure 24 only in that the final speed remains at the cruising velocity, thus minimizing the arrival time $t_{a,j+1}$. Additionally, the two inflection times t_1 and t_2 have been replaced with a single inflection time t_i. It follows that the arrival speed $v_{j+1} = v_{cru}$. Since figure 27 is a special case of figure 24, the equation describing the arrival time $t_{a,j+1}$ is a special case of equation 75. Equation 75 is restated here as equation 100.

$$t_{a,j+1} = t_{d,j} + [\ 2\ a_0\ D_j + (v_{cru} - v_{j+1})^2 + (v_{cru} - v_j)^2\ ] / (2\ a_0\ v_{cru}\ )\ . \qquad (100)$$

To find the minimum arrival time, v_{cru} is set equal to v_{j+1} in equation 100, and the arrival time is notated as t_{min}.

$$t_{min} = t_{d,j} + [\ 2\ a_0\ D_j + (v_{cru} - v_j)^2\ ] / (2\ a_0\ v_{cru}\ )\ . \qquad (101)$$

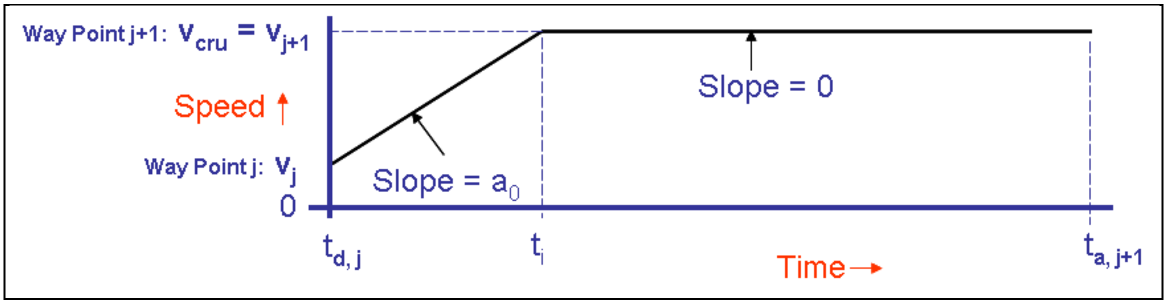


Figure 27. The motion profile for maximum arrival speed and minimum arrival time.

For some speeds at waypoint j and some distances between waypoints j and j+1, the platform may not have time to accelerate to the speed v_{cru}. In that case, the motion profile for speed vs. time looks like figure 28. The distance covered D_j is the average speed $(v_j + v_{max})/2$ times the time to travel between waypoints $t_{min} - t_{d,j}$, or

$$D_j = (\ t_{min} - t_{d,j}\ )\ (v_j + v_{max})\ /\ 2\ . \qquad (102)$$

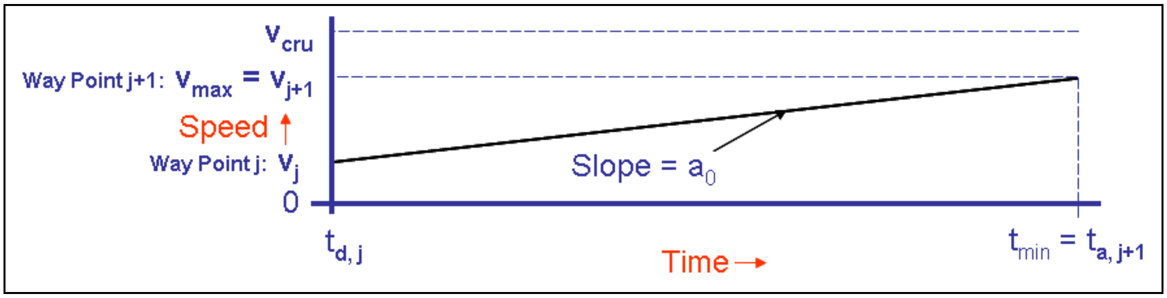


Figure 28. The maximum speed if the distance does not allow enough time for V_{max} to equal V_{cru}.

The relationship between the time traveled between waypoints $t_{min} - t_{d,j}$, the acceleration a_0, and the speeds v_j and v_{max} is

$$(v_{max} - v_j) = a_0 (t_{min} - t_{d,j}) . \tag{103}$$

After eliminating the variables t_{min} and $t_{d,j}$ using equations 102 and 103, then solving for v_{max}, we find the maximum speed in this case is

$$v_{max} = (2 a_0 D_j + v_j^2)^{1/2}. \tag{104}$$

Using equation 104 to eliminate v_{max} from equation 103, and solving for t_{min}, we find the corresponding minimum arrival time at the j+1th waypoint is

$$t_{min} = t_{d,j} + [(2 a_0 D_j + v_j^2)^{1/2} - v_j] / a_0 . \tag{105}$$

So, the maximum speed at the j+1th waypoint v_{j+1} the user can select for the platform is v_{max}, as defined by equation 104. If the value of equation 104 does not exceed v_{cru}, then the corresponding minimum arrival time $t_{a,j+1}$ the user may select is t_{min}, defined by equation 105. If the value for v_{max} as defined by equation 104 exceeds the platform cruising speed v_{cru}, then the maximum arrival speed the user can select for v_{j+1} is v_{cru} with a minimum arrival time $t_{a,j+1}$, for the platform is defined by equation 105.

4.1.6 Restrictions on Input Data – Minimum Speed, Maximum Time

The minimum speed with which a platform can arrive at the j+1th waypoint is generally (but not always) zero. This is a special case of the velocity profile of figure 24, shown here as figure 29. Equations 75 and 76 govern the platform's motion (with equations 72 and 73 defining the inflection times t_1 and t_2) with v_{j+1} set to zero. The corresponding maximum time is

$$t_{max} = t_{d,j} + [2 a_0 D_j + v_{cru}^2 + (v_{cru} - v_j)^2] / (2 a_0 v_{cru}) \tag{106}$$

and the inflection times are

$$t_1 = t_{d,j} + (v_{cru} - v_j) / a_0 \tag{107}$$

and

$$t_2 = t_{max} - a_0 / v_{cru} . \tag{108}$$

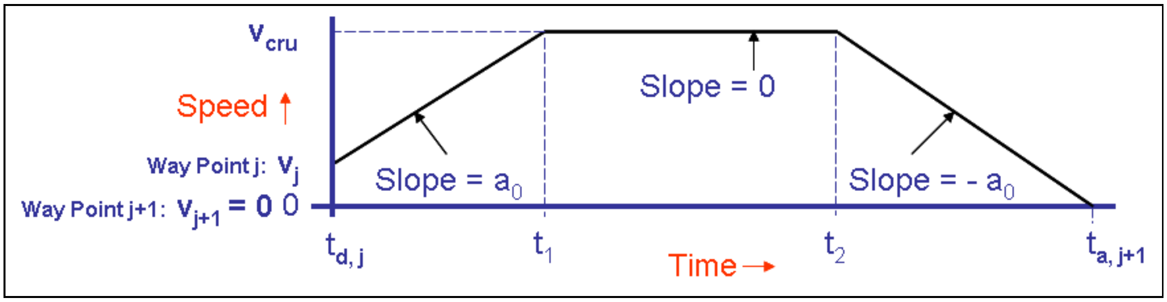


Figure 29. The motion profile for zero arrival speed and maximum arrival time.

4.1.7 Restrictions on Input Data – Peak Speed Less Than v_{cru}.

For some speeds at waypoint j and some distances between waypoints j and j+1, the platform may not be able to achieve the cruising speed v_{cru}. In that case, the speed profile resembles figure 30. To find the value of v_{pk}, we invoke equations 71–73. But in this case, there is only one inflection point t_i, and at no point between the jth and j+1th waypoints does the platform travel at a constant speed. The arrival speed v_{j+1} is zero. With these considerations in mind, equation 71 for the distance can be rewritten as

$$D_j = (v_{pk} + v_j)\,(t_i - t_{d,j}) / 2 + \; v_{pk}\,(t_{a,j+1} - t_i) / 2\,. \tag{109}$$

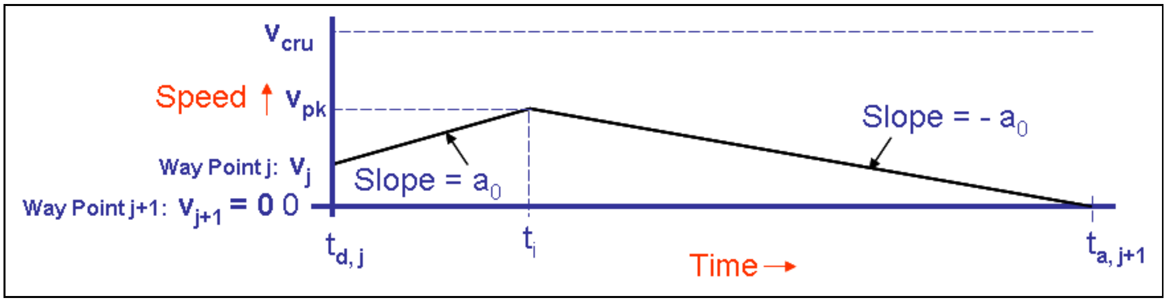


Figure 30. The speed of the platform if it never reaches v_{cru}.

Equations 72 and 73 relate the inflection times and the start and ending times with the acceleration and speed. They are

$$t_i - t_{d,j} = (\, v_{pk} - v_j \,) / a_0 \tag{110}$$

and

$$t_{a,j+1} - t_i = -v_{pk} / -a_0\,. \tag{111}$$

We use equations 110 and 111 to eliminate the departure time $t_{d,j}$, the inflection time t_i, and the arrival time $t_{a,j+1}$ from equation 109, rendering it as

$$D_j = (v_{pk}^{\;2} - v_j^{\;2}) / (2\,a_0) + \; v_{pk}^{\;2} \; / (2\,a_0)\,. \tag{112}$$

Upon solving for v_{pk}, we find

$$v_{pk} = [\ (2\ a_0\ D_j + v_j^2) / 2\ ]^{1/2}\ . \tag{113}$$

To find the corresponding maximum time t_{max}, we eliminate the inflection time t_i in equations 110 and 111.

$$t_{d,j} + (\ v_{pk} - v_j\ ) / a_0 = t_{a,j+1} - v_{pk} / a_0\ . \tag{114}$$

The final step is to substitute equation 115 for v_{pk} into equation 116 and solve for $t_{a,j+1}$ and define it as t_{max}.

$$t_{max} = t_{d,j} + \{\ 2\ [\ (2\ a_0\ D_j + v_j^2\ ) / 2\ ]^{1/2} - v_j\ \} / a_0\ . \tag{115}$$

4.1.8 Restrictions on Input Data – Insufficient Time for Platform to Reach Zero

For some speeds at waypoint j and some distances between waypoints j and j+1, the platform may not have time to come to a full stop. In that case, the motion profile for speed vs. time looks like figure 31. In this case, the arrival velocity is the minimum velocity calculated with equation 116 but with the sign for the acceleration a_0 changed.

$$v_{min} = (v_j^2 - 2\ a_0\ D_j)^{1/2}. \tag{116}$$

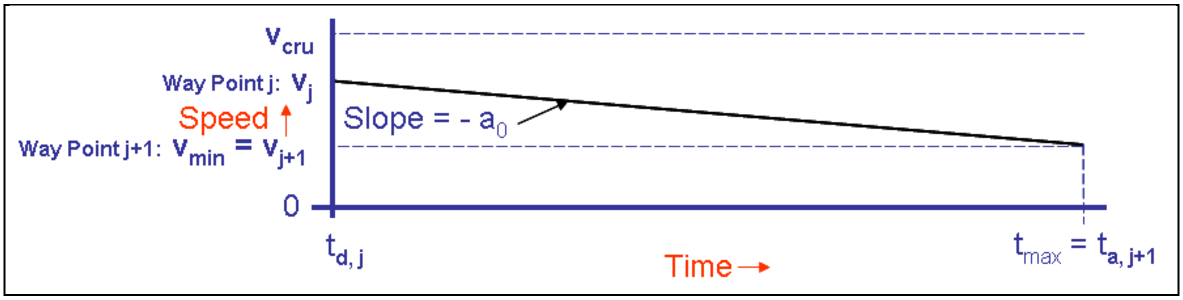


Figure 31. The minimum speed if the platform does not have time to reach zero speed at the arrival waypoint.

Likewise, the expression for the maximum arrival time resembles equation 105, but the sign for the acceleration is changed.

$$t_{max} = t_{d,j} + [v_j - (v_j^2 - 2\ a_0\ D_j\ )^{1/2}] / a_0\ . \tag{117}$$

So, the minimum speed at the j+1th waypoint v_{j+1} the user can select for the platform is zero, but only if the quantity $v_j^2 - 2\ a_0\ D_j$ (see equation 109) is negative. The corresponding maximum time t_{max} is defined by equation 106, with 107 and 108 giving the values for the inflection times t_1 and t_2. If the quantity $v_j^2 - 2\ a_0\ D_j$ is positive, the value for the minimum arrival time is set by equation 109. The corresponding maximum time t_{max} is calculated with equation 117.

Should the user wish to have a loiter time, the arrival speed $v_{a,j+1}$ must be set to zero by default. However, should the user desire a loiter time but the condition that the quantity $v_j^2 - 2\ a_0\ D_j$ is >0, then the arrival speed is set according to equation 117, and the loiter time is automatically set to zero.

4.1.9 Inputting Speed or Time Data

To facilitate computing the data needed for interpolation, we first consider the speed profile as shown in figure 32. It defines the top speed v_{top} of the platform between waypoints for the profile shown in the figure, which happens at the inflection time t_t. The figure is a special case of the speed profile of figure 24. Likewise, the quantitative description of the figure is a special case of equations 72, 73, 75, and 76. To make use of those equations, the expression for v_{top} must be derived.

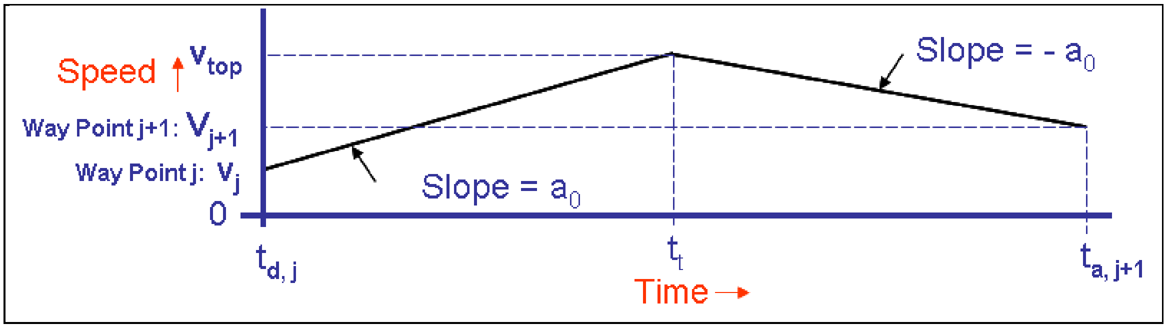


Figure 32. The quantity v_{top} and t_t are defined graphically.

Suppose the user specifies the arrival velocity v_{j+1}. The distance between the waypoints will be

$$D_j = (v_{top} + v_j)\ (t_i - t_{d,j})\ /\ 2 +\ (v_{j+1} + v_{top})\ (t_{a,j+1} - t_i)\ /\ 2\ . \tag{118}$$

The inflection time t_i and start and ending times $t_{d,j}$ and $t_{a,j+1}$ are related to the acceleration and velocities v_j, v_{j+1} by equations 119 and 120.

$$t_t - t_{d,j} = (\ v_{top} - v_j\ )\ /\ a_0\ . \tag{119}$$

$$t_{a,j+1} - t_t = (\ v_{j+1} - v_{top}\ )\ /\ -a_0\ . \tag{120}$$

After substituting equations 119 and 120 into equation 118 to eliminate all the time variables, we find

$$D_j = (v_{top}^2 - v_j^2)\ /\ (\ 2\ a_0\ ) - (v_{j+1}^2 - v_{top}^2)\ /\ (\ 2\ a_0\ )\ . \tag{121}$$

Solving for v_{top} renders

$$v_{top} = [\ (2\ a_0\ Dj + v_{j+1}^2 + v_j^2\ )\ /\ 2\ ]^{1/2}\ . \tag{122}$$

If the user specifies the arrival time instead, we first use equations 119 and 120 to eliminate t_t and solve for v_{j+1}.

$$v_{j+1} = 2\, v_{top} - v_j - a_0\, (t_{a,j+1} - t_{d,j}) \,. \tag{123}$$

We next solve equation 115 for v_{j+1}.

$$v_{j+1} = (2\, v_{pk}^2 - v_j^2 - 2\, a_0\, D_j)^{1/2} \,. \tag{124}$$

Finally, we equate equations 123 and 124 to eliminate v_{j+1}. After solving for v_{top}, we see that

$$v_{top} = v_j + a_0\, (t_{a,j+1} - t_{d,j}) - [\, v_j\, a_0\, (t_{a,j+1} - t_{d,j}) - a_0\, D_j + (1 / 2)\, a_0^2\, (t_{a,j+1} - t_{d,j})^2\,]^{1/2}. \tag{125}$$

When solving to find equation 125, the negative case was selected on the basis of calculating test values for the variables.

Lastly, we restate equations 72, 73, 75, and 76 with v_{top} substituted for v_{cru}, and isolate the inflection times t_1 and t_2.

$$t_1 = t_{d,j} + (v_{top} - v_j) / a_0 \,. \tag{126}$$

$$t_2 = t_{a,j+1} - (v_{top} - v_{j+1}) / a_0 \,. \tag{127}$$

$$t_{a,j+1} = t_{d,j} + [\, 2\, a_0\, D_j + (v_{top} - v_{j+1})^2 + (v_{top} - v_j)^2\,] / (2\, a_0\, v_{top}) \,. \tag{128}$$

$$v_{j+1} = v_{top} - \{2\, a_0\, [\, v_{top}\, (t_{a,j+1} - t_{d,j}) - D_j\,] - (v_{top} - v_j)^2\, \}^{1/2} \,. \tag{129}$$

If the user selects the velocity v_{j+1}, we calculate the top speed v_{top} using equation 122. If v_{top} is less than v_{cru}, v_{top} retains its value. If v_{top} is greater than v_{cru}, then v_{top} is set equal to v_{cru}. We then use equation 128 to calculate $t_{a,j+1}$, and equations 126 and 127 to calculate the inflection point times t_1 and t_2.

If the user decides to select the j+1th arrival time $t_{a,j+1}$, we calculate the top speed v_{top} using equation 125. If v_{top} is less than v_{cru}, v_{top} retains its value. If v_{top} is greater than v_{cru}, then v_{top} is set equal to v_{cru}. We then use equation 129 to calculate v_{j+1}, and equations 126 and 127 to calculate the inflection point times t_1 and t_2.

4.1.10 Summary of Inputting Waypoint Data

Figure 33 shows the overall process of inputting waypoint data. The process shown in the figure is inclusive of equations 47 to 129, although it does not include the restrictions outlined in table 1. The rectangular boxes represent processes to be executed. The diamond boxes are questions with yes or no answers. The answers determine the calculations, tasks, and the order in which the tasks and calculations are performed. The green letters indicate the “Start” box, and the variables defining the platform’s motion are in blue. Violet shows the indices, amber indicates equation numbers, and red shows when it is necessary to “Do Interpolation.”

At the beginning, the user identifies the platform type and the number of waypoints (the recording index), while the software retrieves the acceleration rate a_0 and the cruising speed v_{cru}

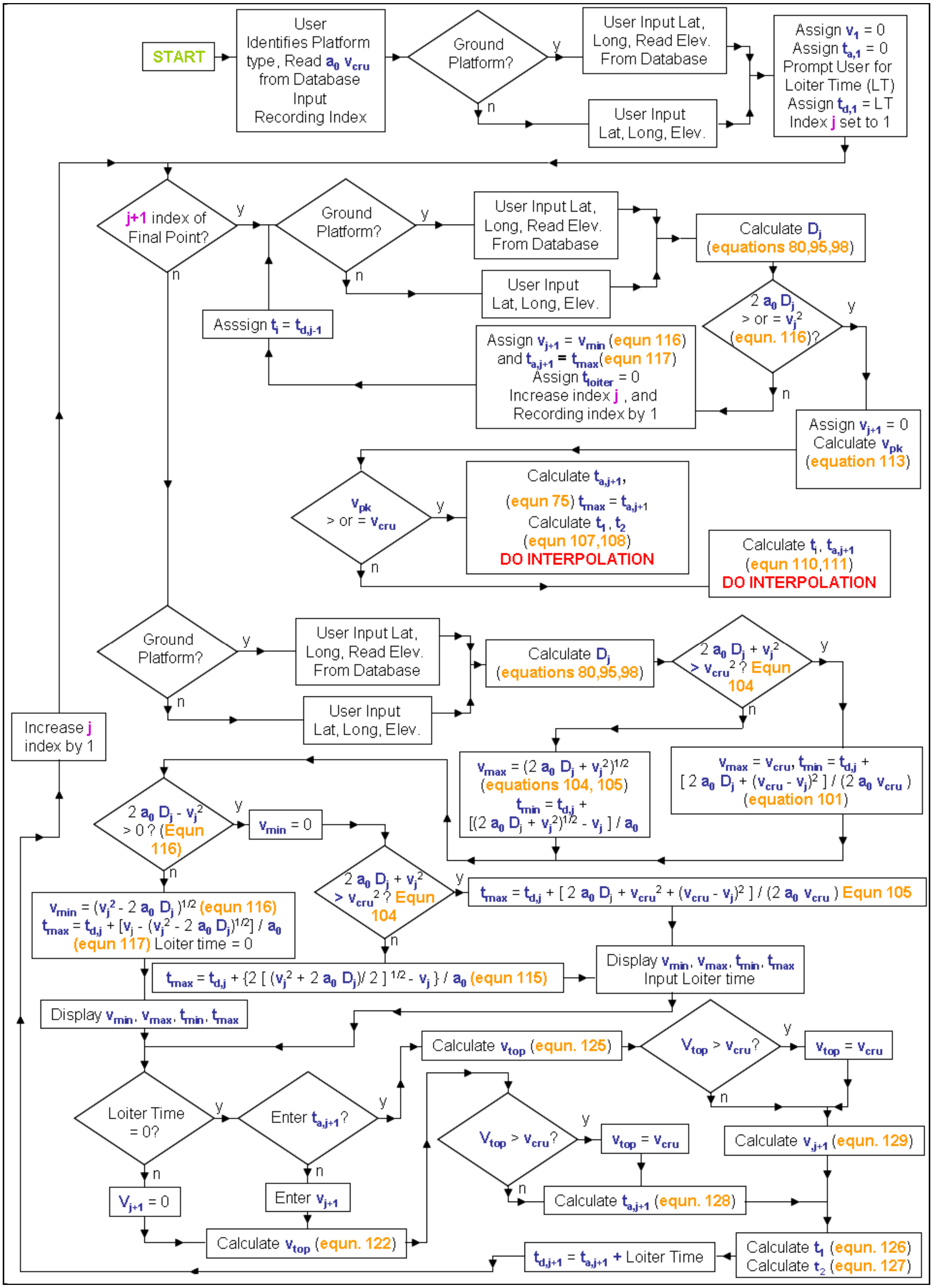

START
User Identifies Platform type, Read a_0 v_{cru} from Database Input Recording Index
Ground Platform?
y
n
User Input Lat, Long, Read Elev. From Database
User Input Lat, Long, Elev.
Assign v_1 = 0 Assign $t_{a,1}$ = 0 Prompt User for Loiter Time (LT) Assign $t_{d,1}$ = LT Index j set to 1
j+1 index of Final Point?
y
n
Ground Platform?
y
n
User Input Lat, Long, Read Elev. From Database
User Input Lat, Long, Elev.
Calculate D_j (equations 80,95,98)
$2\ a_0\ D_j$ > or = v_j^2 (equn. 116)?
y
n
Asssign $t_j = t_{d,j-1}$
Assign $v_{j+1} = v_{min}$ (equn 116) and $t_{a,j+1} = t_{max}$(equn 117) Assign t_{loiter} = 0 Increase index j , and Recording index by 1
Assign v_{j+1} = 0 Calculate v_{pk} (equation 113)
v_{pk} > or = v_{cru}
y
n
Calculate $t_{a,j+1}$, (equn 75) $t_{max} = t_{a,j+1}$ Calculate t_1 , t_2 (equn 107,108) DO INTERPOLATION
Calculate t_i, $t_{a,j+1}$ (equn 110,111) DO INTERPOLATION
Ground Platform?
y
n
User Input Lat, Long, Read Elev. From Database
User Input Lat, Long, Elev.
Calculate D_j (equations 80,95,98)
$2\ a_0\ D_j + v_j^2$ > v_{cru}^2 ? Equn 104
y
n
Increase j index by 1
$v_{max} = (2\ a_0\ D_j + v_j^2)^{1/2}$ (equations 104, 105) $t_{min} = t_{d,j}$ + $[(2\ a_0\ D_j + v_j^2)^{1/2} - v_j\ ] / a_0$
$v_{max} = v_{cru}$, $t_{min} = t_{d,j}$ + $[\ 2\ a_0\ D_j + (v_{cru} - v_j)^2\ ] / (2\ a_0\ v_{cru}\ )$ (equation 101)
$2\ a_0\ D_j - v_j^2$ > 0 ? (Equn 116)
y
n
v_{min} = 0
$v_{min} = (v_j^2 - 2\ a_0\ D_j\ )^{1/2}$ (equn 116) $t_{max} = t_{d,j} + [v_j - (v_j^2 - 2\ a_0\ D_j)^{1/2}] / a_0$ (equn 117) Loiter time = 0
$2\ a_0\ D_j + v_j^2$ > v_{cru}^2 ? Equn 104
y
n
$t_{max} = t_{d,j} + [\ 2\ a_0\ D_j + v_{cru}^2 + (v_{cru} - v_j)^2\ ] / (2\ a_0\ v_{cru}\ )$ Equn 105
$t_{max} = t_{d,j} + \{2\ [\ (v_j^2 + 2\ a_0\ D_j)/\ 2\ ]^{1/2} - v_j\ \} / a_0$ (equn 115)
Display v_{min}, v_{max}, t_{min}, t_{max} Input Loiter time
Display v_{min}, v_{max}, t_{min}, t_{max}
Calculate v_{top} (equn. 125)
$V_{top} > v_{cru}$?
y
n
$v_{top} = v_{cru}$
Loiter Time = 0?
y
n
Enter $t_{a,j+1}$?
y
n
V_{j+1} = 0
Enter v_{j+1}
Calculate v_{top} (equn. 122)
$V_{top} > v_{cru}$?
y
n
$v_{top} = v_{cru}$
Calculate $t_{a,j+1}$ (equn. 128)
Calculate v_{j+1} (equn. 129)
Calculate t_1 (equn. 126) Calculate t_2 (equn. 127)
$t_{d,j+1} = t_{a,j+1}$ + Loiter Time

Figure 33. Overview of the waypoint data input process.

based on the platform type. If the platform is not a ground platform, the user inputs latitude, longitude, and elevation. In the case that the platform is a ground platform, the software reads the elevation data from the DTED database (*22*). The first point motion data, v_1 and $t_{a,1}$, are set to zero. The user inputs the loiter time, and the index j is set to 1. Implicit in these processes is the assumption that every platform has at least two distinct waypoints. The case where a platform has only one waypoint will be treated later.

If the index indicates that the next waypoint is the last, a special process begins. As with the first waypoint, the user inputs the latitude, longitude, and (if the platform is not a ground platform) elevation. When this data is used, the distance from the jth waypoint is calculated. If the distance is not sufficient to allow the platform to stop (v_j^2 is not >2 a_0 D_j [remember, this is the final waypoint and the platform must arrive here with a speed of zero]) while it decelerates at a_0, the deployment module assigns the minimum possible speed at the waypoint v_{j+1}. It then calculates the corresponding arrival time $t_{a,j+1}$, departure time $t_{d,j+1}$ (remember, there is no loiter time because the speed is not zero), and inflection time t_i. Lastly, it forces the user to input another waypoint (in this case, $t_1 = t_2$, so the i represents the subscripts 1 and 2 since there is only one inflection point). This process continues until the platform traverses enough distance to come to a halt. If there is enough distance for the platform to come to a halt, then the speed v_{j+1} is set to zero and the greatest speed between the waypoints v_{pk} is calculated. If v_{pk} is greater than the specification v_{cru}, then v_{pk} is set to the value of v_{cru}; t_1, t_2, and $t_{a,j+1}$ are calculated; and the interpolated points are calculated. If v_{pk} is less than the specification v_{cru}, then v_{pk} remains unchanged, and t_i (again, the i subscript represents 1 or 2 since in this case $t_1 = t_2$) is calculated with $t_{a,j+1}$, and the interpolated points are also calculated.

If the next waypoint isn't the final point, the user is asked to input the latitude, longitude, and (in the event the platform is not a ground platform) the elevation. The distance between the previous waypoint and the current waypoint D_j is calculated. If there is not enough distance for the platform to reach the cruising velocity v_{cru}, then the maximum velocity v_{max} is calculated; otherwise, v_{max} is set equal to v_{cru}. The minimum corresponding arrival time t_{min} is calculated. If there is not enough distance for the platform to reach zero velocity, the minimum velocity v_{min} is calculated, and the loiter time is set to zero and the corresponding value of t_{max} calculated. Then, v_{min}, t_{max}, v_{max}, and $_{tmin}$ are displayed as guidance. Otherwise, v_{min} is set equal to zero, with the values t_{min}, t_{max}, v_{min}, and v_{max} displayed as guidance. If v_{min} is zero, the user is prompted to input the loiter time.

If the loiter time is zero, then final data for the platform motion at the waypoint is inputted, either as an arrival time $t_{a,j+1}$ or waypoint speed v_{j+1}. The deployment module calculates the maximum possible speed between the current and previous waypoint v_{top} and compares it to the cruise speed v_{cru}. If v_{top} is greater than v_{cru}, then v_{top} is set equal to v_{cru}. When v_{top} is used, the inflection points t_1 and t_2 are calculated, and the departure time $t_{d,j+1}$, after which the index j is increased by 1, and the index are tested to see if the next waypoint is the final waypoint.

4.2 Modifying Input

Next, we assume that a string of waypoints has been inputted and recorded by the deployment module, but that the user wishes to modify them. Four processes are possible: moving a waypoint, adding a waypoint, removing a waypoint, and changing either the arrival time $t_{a,j+1}$, waypoint velocity v_j, or if the waypoint velocity v_j is zero, the loiter time.

4.2.1 Moving a Waypoint

Figure 34 shows the sequence for moving a waypoint. It is similar to, but somewhat more abbreviated than, the process for inputting a waypoint. After starting the process, the user identifies the platform and waypoint to be moved. If this is a ground platform, the user inputs the new latitude and longitude while the software provides the elevation data from the DTED database (*22*). If it is not a ground platform, the user provides the latitude, longitude, and elevation. If this is the first waypoint, the index is immediately changed because the waypoint speed and the initial time are zero and must not be changed. The loiter time can be changed by another process. Implicit in all these processes is the assumption that every platform has at least two distinct waypoints. The case where a platform has only one waypoint will be treated later.

If this is the final waypoint for the platform, the distance D_j is calculated, and if there is enough distance for the platform to stop ($2\ a_0\ D_j > v_j^2$), the final speed v_{j+1} is assigned to zero and the peak speed v_{pk} is calculated. If the platform peak speed v_{pk} exceeds the cruising speed v_{cru}, then v_{pk} is set to v_{cru}, and the arrival times $t_{a,j+1}$, t_1, and t_2 (see figure 29) are calculated and the interpolation redone. If the platform is not greater than v_{cru}, then the single inflection time t_i is calculated (in this case, $t_1 = t_2$, so the i represents the subscripts 1 and 2, since there is only one inflection point) along with the arrival time $t_{a,j+1}$ (see figure 30), after which the interpolation is redone. Should the distance D_j not be sufficient to allow the platform to stop, v_{min} is calculated and the speed v_{j+1} is assigned the value v_{min}; the arrival time $t_{a,j+1}$ is calculated, along with the inflection time t_i (in this case, $t_1 = t_2$, so the i represents the subscripts 1 and 2); t_{loiter} is assigned to 0; and the index j is increased by one, as is the recording index (which is the number of waypoints for this platform). Then data is input for the next waypoint (latitude, longitude, and elevation if it is not a ground platform, and latitude and longitude with elevation read from a database if it is a ground platform), and the distance D_j calculated. The distance is then tested to see if the platform has enough room to stop. If it is not, the user is prompted for another point until the platform has distance enough to stop.

If this is not the final waypoint, the distance D_j is calculated. If there is enough distance D_j for the speed to reach v_{cru}, then the maximum speed of the platform between waypoints v_{max} is assigned the value v_{cru}. If there is not enough distance, then the maximum speed the distance allows (equation 104) is assigned to v_{max}. Next, the speed for this waypoint assigned previously (v_{j+1}) is tested to see if it is greater than the v_{max}. If it is, then v_{j+1} is assigned the value v_{max} and if it is not, v_{j+1} remains unchanged. Then, the distance is tested to see if the platform has enough space to reach zero velocity. If it does, then v_{min} is assigned the value of 0. If it does not, then

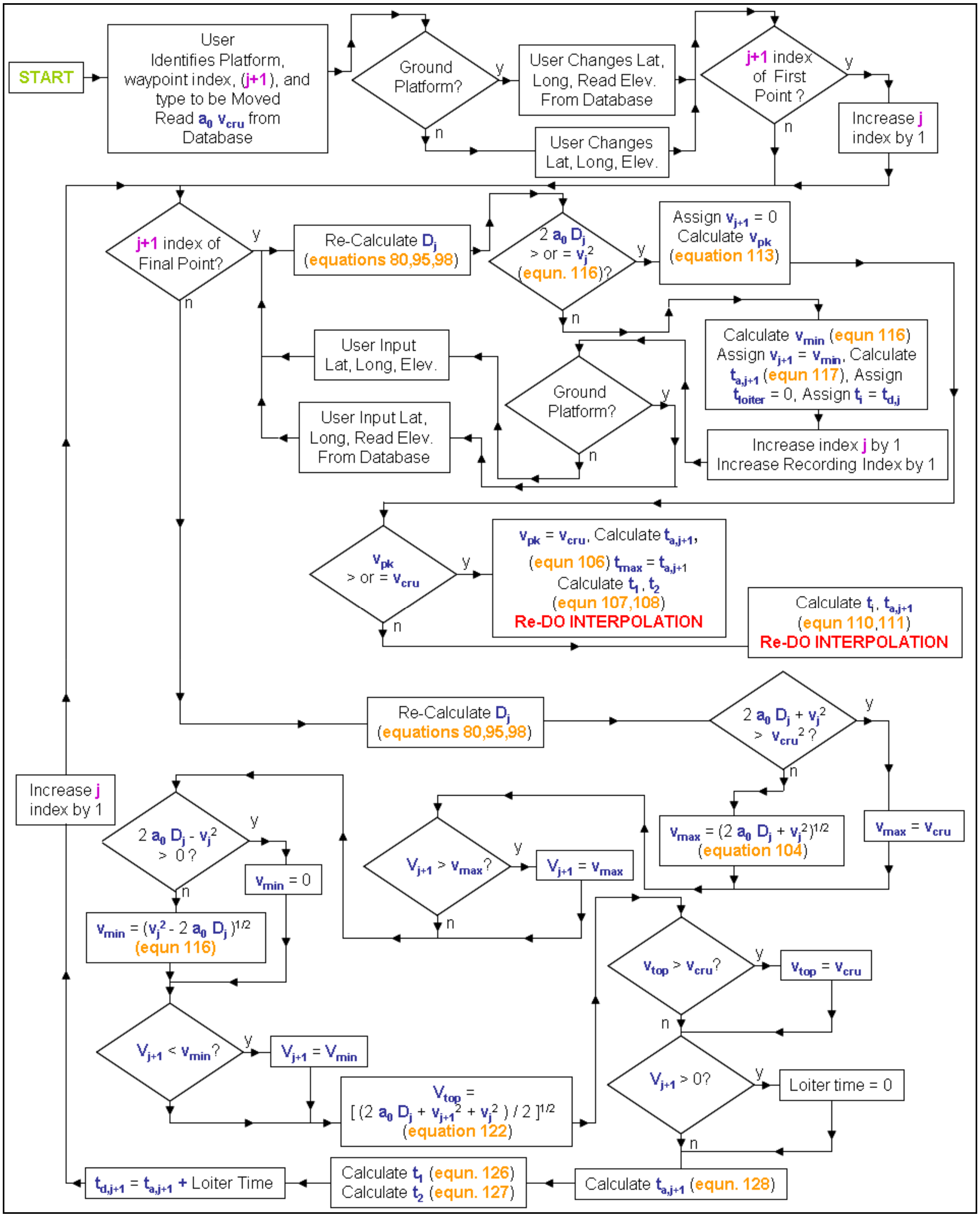

START
User Identifies Platform, waypoint index, (j+1), and type to be Moved Read a_0 v_{cru} from Database
Ground Platform?
y
n
User Changes Lat, Long, Read Elev. From Database
User Changes Lat, Long, Elev.
j+1 index of First Point ?
Increase j index by 1
j+1 index of Final Point?
Re-Calculate D_j (equations 80,95,98)
$2\ a_0\ D_j$ > or = v_j^2 (equn. 116)?
Assign $v_{j+1} = 0$ Calculate v_{pk} (equation 113)
Calculate v_{min} (equn 116) Assign $v_{j+1} = v_{min}$, Calculate $t_{a,j+1}$ (equn 117), Assign $t_{loiter} = 0$, Assign $t_i = t_{d,j}$
User Input Lat, Long, Elev.
User Input Lat, Long, Read Elev. From Database
Ground Platform?
Increase index j by 1 Increase Recording Index by 1
v_{pk} > or = v_{cru}
$v_{pk} = v_{cru}$, Calculate $t_{a,j+1}$, (equn 106) $t_{max} = t_{a,j+1}$ Calculate t_1, t_2 (equn 107,108) Re-DO INTERPOLATION
Calculate t_i, $t_{a,j+1}$ (equn 110,111) Re-DO INTERPOLATION
Re-Calculate D_j (equations 80,95,98)
$2\ a_0\ D_j + v_j^2$ > v_{cru}^2 ?
$v_{max} = (2\ a_0\ D_j + v_j^2)^{1/2}$ (equation 104)
$v_{max} = v_{cru}$
Increase j index by 1
$2\ a_0\ D_j - v_j^2$ > 0 ?
$v_{min} = 0$
$v_{min} = (v_j^2 - 2\ a_0\ D_j)^{1/2}$ (equn 116)
$V_{j+1} > v_{max}$?
$V_{j+1} = v_{max}$
$V_{j+1} < v_{min}$?
$V_{j+1} = V_{min}$
V_{top} = [(2 a_0 D_j + v_{j+1}^2 + v_j^2) / 2]$^{1/2}$ (equation 122)
$v_{top} > v_{cru}$?
$v_{top} = v_{cru}$
$V_{j+1} > 0$?
Loiter time = 0
Calculate $t_{a,j+1}$ (equn. 128)
Calculate t_1 (equn. 126) Calculate t_2 (equn. 127)
$t_{d,j+1} = t_{a,j+1}$ + Loiter Time

Figure 34. Overview of the process of moving a waypoint.

the minimum possible velocity v_{min} is assigned the lowest speed the platform can reach $(v_j^2 - 2\ a_0\ D_j)^{1/2}$ by equation 116. v_{j+1} is next tested to see if it is less than v_{min}. If it is, it is assigned the value of v_{min}. If it is not, it remains unchanged. Next, v_{top} is calculated by equation 122. If v_{top} is greater than the cruise speed v_{cru}, v_{top} is assigned the value of v_{cru}. If it is not, it remains unchanged. Next, if the waypoint velocity v_{j+1} is nonzero, the loiter time is set to zero. If v_{j+1} is zero, then the loiter time remains unchanged. Finally, $t_{a,j+1}$, t_1, t_2, and $t_{d,j+1}$ are calculated, and the index j is increased to check the next waypoint.

4.2.2 Adding a Waypoint

The flowchart for adding a waypoint is shown in figure 35. Portions of the figure resemble the process of moving a waypoint, shown in figure 34. After identifying the platform type and new index number, the user inputs the longitude, latitude, and elevation for an aerial platform, or just the latitude and longitude for a ground platform. The indices for the waypoints that follow the new waypoint in time are incremented by one, as is the recording index that records the number of waypoints. If the added waypoint is intended to be the first waypoint, the software assigns the waypoint speed and arrival time as zero. The user is prompted for the loiter time, the departure time $t_{d,1}$ is set to the loiter time, and the index is increased by one. The next waypoint is tested to see if it is the final waypoint.

If the added waypoint is not intended to be the first waypoint, it is tested to see if it is the final waypoint. Should this be the final waypoint, the distance D_j is calculated. If there is enough distance for the platform to stop $(2\ a_0\ D_j > v_j^2)$, the final waypoint velocity v_{j+1} is assigned to zero and the peak speed v_{pk} is calculated. If the peak speed v_{pk} has time to get to v_{cru}, then v_{pk} is assigned the value of v_{cru}, and the arrival times $t_{a,j+1}$, t_1, and t_2 are calculated (see figure 29) and the interpolation redone. Otherwise (see figure 30), v_{pk} remains unchanged and t_i is calculated (the i subscript represents 1 or 2, since in this case $t_1 = t_2$) with $t_{a,j+1}$ and the interpolated points are recalculated. If there is not enough distance for the platform to stop, v_{min} is calculated and the speed v_{j+1} is assigned the minimum possible value for speed v_{min}; t_{max} is calculated and assigned to the arrival time $t_{a,j+1}$; t_{loiter} is assigned to 0; and the index j is increased by one, as is the recording index (which is the number of waypoints). The user is prompted to input the latitude, longitude, and elevation for an aerial platform, or only latitude and longitude for a ground platform. The distance D_j to the new platform is calculated, and the cycle of new platforms is inputted until it is able to stop.

If this is not the final waypoint, then guidance must be calculated to let the user select the new waypoint arrival time $t_{a,j+1}$ or waypoint velocity v_{j+1}. This is first accomplished by calculating the distance to the new waypoint D_j. If there is enough distance for the platform to reach v_{cru}, $(2\ a_0\ D_j + v_j^2)^{1/2}$, then v_{max} is assigned the value of v_{cru}, and the corresponding t_{min} is calculated. If there isn't enough distance, then maximum speed v_{max} is calculated, with the corresponding t_{min}.

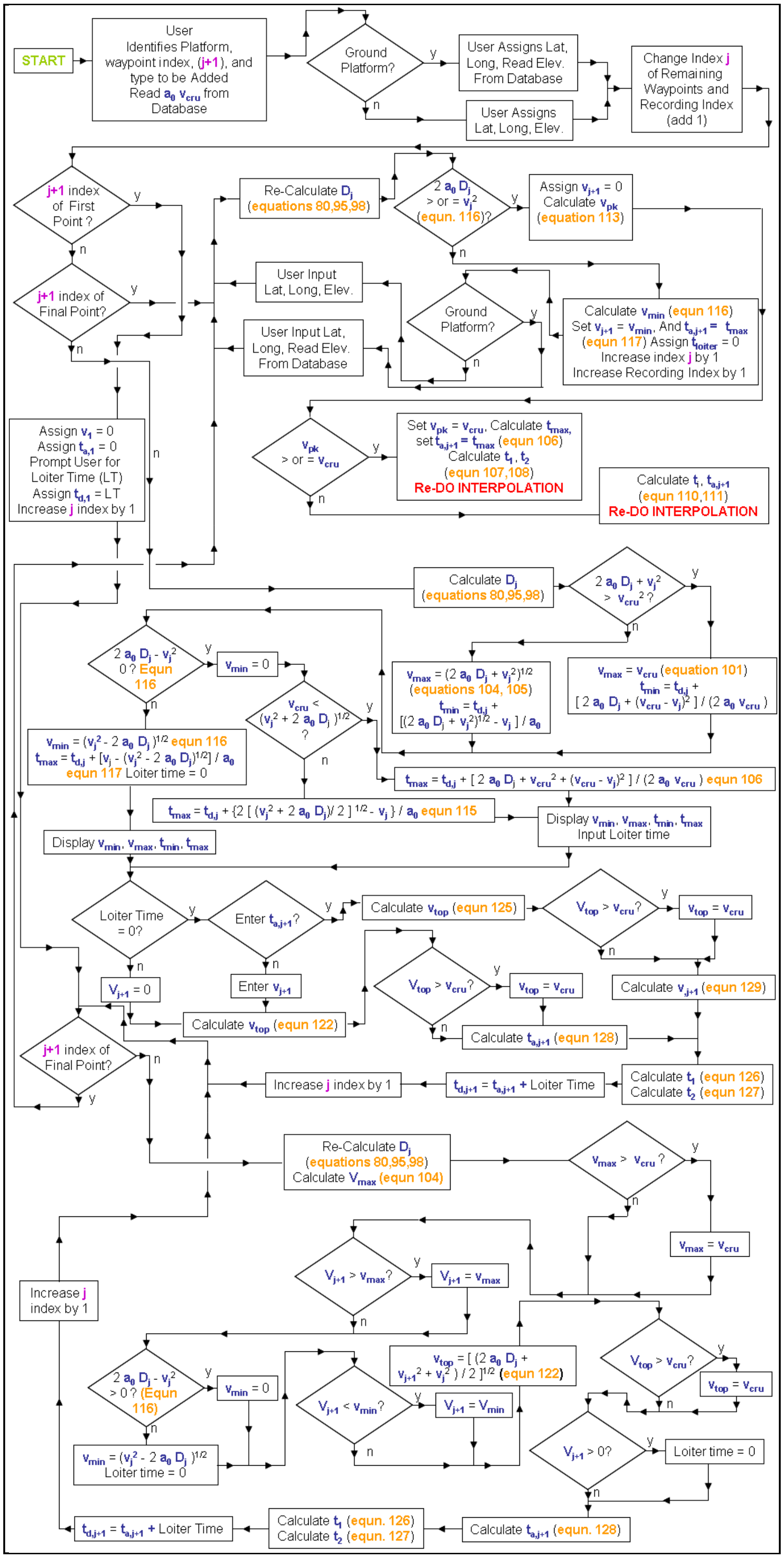

START
User Identifies Platform, waypoint index, (j+1), and type to be Added Read a_0 v_{cru} from Database
Ground Platform?
y
n
User Assigns Lat, Long, Read Elev. From Database
User Assigns Lat, Long, Elev.
Change Index j of Remaining Waypoints and Recording Index (add 1)
j+1 index of First Point ?
j+1 index of Final Point?
Re-Calculate D_j (equations 80,95,98)
$2 a_0 D_j$ > or = v_j^2 (equn. 116)?
Assign v_{j+1} = 0 Calculate v_{pk} (equation 113)
User Input Lat, Long, Elev.
User Input Lat, Long, Read Elev. From Database
Ground Platform?
Calculate v_{min} (equn 116) Set v_{j+1} = v_{min}, And $t_{a,j+1}$ = t_{max} (equn 117) Assign t_{loiter} = 0 Increase index j by 1 Increase Recording Index by 1
Assign v_1 = 0 Assign $t_{a,1}$ = 0 Prompt User for Loiter Time (LT) Assign $t_{d,1}$ = LT Increase j index by 1
v_{pk} > or = v_{cru}
Set v_{pk} = v_{cru}, Calculate t_{max}, set $t_{a,j+1}$ = t_{max} (equn 106) Calculate t_1, t_2 (equn 107,108) Re-DO INTERPOLATION
Calculate t_i, $t_{a,j+1}$ (equn 110,111) Re-DO INTERPOLATION
Calculate D_j (equations 80,95,98)
$2 a_0 D_j + v_j^2$ > v_{cru}^2 ?
$2 a_0 D_j - v_j^2$ 0 ? Equn 116
v_{min} = 0
v_{cru} < $(v_j^2 + 2 a_0 D_j)^{1/2}$?
$v_{max} = (2 a_0 D_j + v_j^2)^{1/2}$ (equations 104, 105) $t_{min} = t_{d,j}$ + $[(2 a_0 D_j + v_j^2)^{1/2} - v_j] / a_0$
$v_{max} = v_{cru}$ (equation 101) $t_{min} = t_{d,j}$ + $[2 a_0 D_j + (v_{cru} - v_j)^2] / (2 a_0 v_{cru})$
$v_{min} = (v_j^2 - 2 a_0 D_j)^{1/2}$ equn 116 $t_{max} = t_{d,j} + [v_j - (v_j^2 - 2 a_0 D_j)^{1/2}] / a_0$ equn 117 Loiter time = 0
$t_{max} = t_{d,j} + [2 a_0 D_j + v_{cru}^2 + (v_{cru} - v_j)^2] / (2 a_0 v_{cru})$ equn 106
$t_{max} = t_{d,j} + \{2 [(v_j^2 + 2 a_0 D_j)/ 2]^{1/2} - v_j\} / a_0$ equn 115
Display v_{min}, v_{max}, t_{min}, t_{max} Input Loiter time
Display v_{min}, v_{max}, t_{min}, t_{max}
Loiter Time = 0?
Enter $t_{a,j+1}$?
Calculate v_{top} (equn 125)
V_{top} > v_{cru}?
V_{top} = V_{cru}
V_{j+1} = 0
Enter v_{j+1}
Calculate v_{top} (equn 122)
V_{top} > v_{cru}?
V_{top} = V_{cru}
Calculate v_{j+1} (equn 129)
Calculate $t_{a,j+1}$ (equn 128)
j+1 index of Final Point?
Increase j index by 1
$t_{d,j+1}$ = $t_{a,j+1}$ + Loiter Time
Calculate t_1 (equn 126) Calculate t_2 (equn 127)
Re-Calculate D_j (equations 80,95,98) Calculate V_{max} (equn 104)
v_{max} > v_{cru} ?
V_{max} = V_{cru}
Increase j index by 1
V_{j+1} > v_{max}?
V_{j+1} = v_{max}
$2 a_0 D_j - v_j^2$ > 0 ? (Equn 116)
v_{min} = 0
V_{j+1} < v_{min}?
V_{j+1} = V_{min}
$v_{top} = [(2 a_0 D_j + v_{j+1}^2 + v_j^2) / 2]^{1/2}$ (equn 122)
V_{top} > v_{cru}?
V_{top} = V_{cru}
V_{j+1} > 0?
Loiter time = 0
$v_{min} = (v_j^2 - 2 a_0 D_j)^{1/2}$ Loiter time = 0
$t_{d,j+1}$ = $t_{a,j+1}$ + Loiter Time
Calculate t_1 (equn. 126) Calculate t_2 (equn. 127)
Calculate $t_{a,j+1}$ (equn. 128)

Figure 35. Overview of the process of adding a waypoint.

Next, the guidance values for v_{min} and t_{max} are calculated. If there is enough distance D_j for the platform to stop ($2\ a_0\ D_j - v_j^2 > 0$), then v_{min} is assigned a value of zero and the corresponding value of t_{max} is calculated. The formula to use for finding t_{max} depends on whether there is enough distance for the speed to reach v_{cru}. If there is [$v_{cru} < (2\ a_0\ D_j + v_j^2\ )^{1/2}$], then t_{max} is calculated using equation 106. If there is not, t_{max} is calculated with equation 115, followed by the display of v_{min}, t_{max}, v_{max}, and t_{min} to the user as guidance, who inputs the loiter time. If there is not enough distance D_j for the platform to stop ($2\ a_0\ D_j - v_j^2 < 0$), then v_{min} is calculated with equation 116, and the corresponding value of t_{max} is calculated with equation 117 and displayed to the user as guidance while the loiter time is set to zero.

If the loiter time is not zero, the waypoint speed v_{j+1} is set to zero. The highest speed possible for the platform between waypoints v_{top} is calculated with equation 122, and if it exceeds v_{cru}, v_{top} is set to v_{cru}, and $t_{a,j+1}$ is computed with equation 128. If the loiter time is zero, the user is given the option of entering the arrival time $t_{a,j+1}$ or the waypoint velocity v_{j+1}. If the user chooses to enter $t_{a,j+1}$, then v_{top} is calculated using equation 125, and if it exceeds v_{cru}, v_{top} is set to v_{cru}, and v_{j+1} is computed using equation 129. If the user chooses to enter v_{j+1}, then v_{top} is calculated with equation 122, and if it exceeds v_{cru}, it is set to v_{cru}, and $t_{a,j+1}$ is computed with equation 128. Then t_1 and t_2 are calculated with equations 126 and 127, $t_{d,j+1}$ is calculated, and the waypoint index is increased by 1.

If the index is for the final waypoint, then it is treated as the final waypoint as outlined previously. If it is not, it and all the remaining waypoints will have to be checked for self-consistency. To do this, the distance D_j is calculated, and from this, v_{max} is calculated with equation 104. If v_{max} exceeds v_{cru}, v_{max} is set to v_{cru}, otherwise it retains the same value. The waypoint speed, v_{j+1}, is compared to v_{max}; if it is greater than v_{max}, it is set to v_{max}. If it is not, it retains its value. Similarly, if $2\ a_0\ D_j - v_j^2$ is positive, v_{min} is set equal to zero, or the square root of its negative if $2\ a_0\ D_j - v_j^2$ is negative. Then v_{j+1} is compared to v_{min} and set equal to v_{min} if v_{j+1} is less than v_{min}. A new variable, v_{top} is computed with equation 122. If v_{top} is greater than the cruising speed, v_{cru}, v_{top} is set equal to v_{cru}. If v_{j+1} is greater than zero, the loiter time is set to zero. Then $t_{a,j+1}$, t_1, t_2, and $t_{d,j+1}$ are calculated, and the waypoint index is increased by 1.

4.2.3 Removing a Waypoint

Removing a waypoint is a shorter procedure and is illustrated in figure 36. The user first identifies the index and type of platform for which the waypoint must be removed. After reading the acceleration factor a_o and the top cruising speed v_{cru} from a database, the remaining indices for the waypoints for this platform have 1 subtracted from them, and the recording index has 1 subtracted from it.

If the waypoint removed was the first, the former second waypoint becomes the new first waypoint. Then, the waypoint velocity and arrival time are set to zero. The user is prompted for the loiter time, and the index is increased by one. The next waypoint is tested to see if it is the last waypoint.

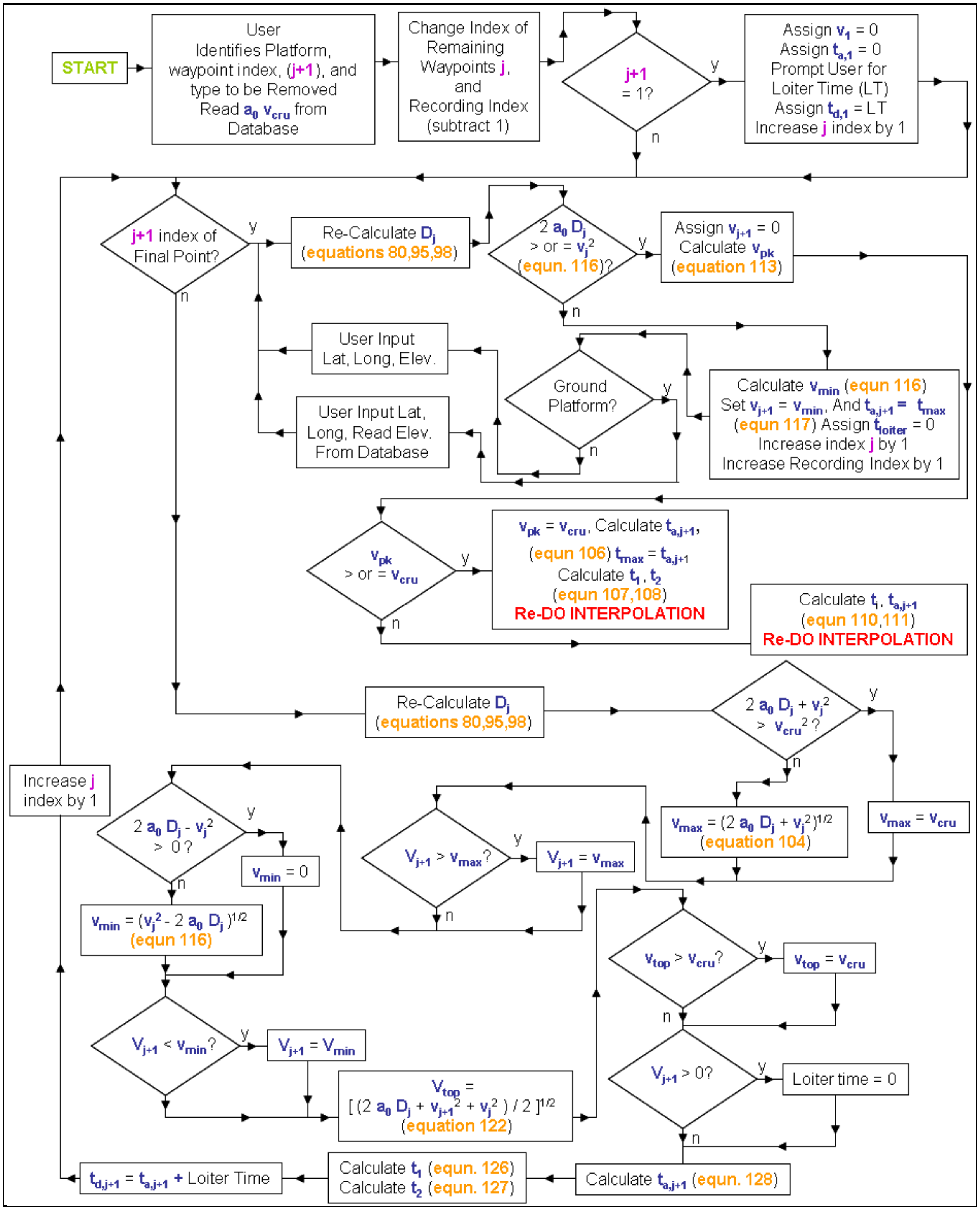

START
User Identifies Platform, waypoint index, (j+1), and type to be Removed Read a_0 v_{cru} from Database
Change Index of Remaining Waypoints j, and Recording Index (subtract 1)
j+1 = 1?
y
n
Assign $v_1 = 0$ Assign $t_{a,1} = 0$ Prompt User for Loiter Time (LT) Assign $t_{d,1}$ = LT Increase j index by 1
j+1 index of Final Point?
y
n
Re-Calculate D_j (equations 80,95,98)
$2\ a_0\ D_j$ > or = v_j^2 (equn. 116)?
y
n
Assign $v_{j+1} = 0$ Calculate v_{pk} (equation 113)
User Input Lat, Long, Elev.
User Input Lat, Long, Read Elev. From Database
Ground Platform?
y
n
Calculate v_{min} (equn 116) Set $v_{j+1} = v_{min}$, And $t_{a,j+1} = t_{max}$ (equn 117) Assign $t_{loiter} = 0$ Increase index j by 1 Increase Recording Index by 1
v_{pk} > or = v_{cru}
y
n
$v_{pk} = v_{cru}$, Calculate $t_{a,j+1}$, (equn 106) $t_{max} = t_{a,j+1}$ Calculate t_1, t_2 (equn 107,108) Re-DO INTERPOLATION
Calculate t_i, $t_{a,j+1}$ (equn 110,111) Re-DO INTERPOLATION
Re-Calculate D_j (equations 80,95,98)
$2\ a_0\ D_j + v_j^2$ > v_{cru}^2 ?
y
n
$v_{max} = (2\ a_0\ D_j + v_j^2)^{1/2}$ (equation 104)
$v_{max} = v_{cru}$
Increase j index by 1
$2\ a_0\ D_j - v_j^2$ > 0 ?
y
n
$v_{min} = 0$
$v_{min} = (v_j^2 - 2\ a_0\ D_j)^{1/2}$ (equn 116)
$V_{j+1} > v_{max}$?
y
n
$V_{j+1} = v_{max}$
$V_{j+1} < v_{min}$?
y
$V_{j+1} = V_{min}$
V_{top} = $[(2\ a_0\ D_j + v_{j+1}^2 + v_j^2) / 2]^{1/2}$ (equation 122)
$v_{top} > v_{cru}$?
y
n
$v_{top} = v_{cru}$
$V_{j+1} > 0$?
y
n
Loiter time = 0
Calculate $t_{a,j+1}$ (equn. 128)
Calculate t_1 (equn. 126) Calculate t_2 (equn. 127)
$t_{d,j+1} = t_{a,j+1}$ + Loiter Time

Figure 36. Overview of the process of removing a waypoint.

If this is the final waypoint, the distance D_j is calculated, and if there is enough distance for the platform to stop ($2\ a_0\ D_j > v_j^2$), the final waypoint velocity v_{j+1} is assigned to zero and the peak speed v_{pk} is calculated with equation 113. If v_{pk} is greater than v_{cru}, then v_{pk} is assigned the value of v_{cru}, the arrival times $t_{a,j+1}$, t_1, and t_2 are calculated (see figure 29), and the interpolation is redone. Otherwise (see figure 30), v_{pk} remains unchanged and t_i (the i subscript represents 1 or 2, since in this case $t_1 = t_2$) and $t_{a,j+1}$ are calculated and the interpolated points recalculated. If there is not enough distance for the platform to stop, the speed v_{j+1} is assigned the minimum possible value for speed v_{min} after calculating v_{min} and the arrival time $t_{a,j+1}$, while t_{loiter} is assigned to 0 and the index j is increased by one, as is the recording index (which is the number of waypoints). The user is prompted to input the latitude, longitude, and elevation for an aerial platform, or only latitude and longitude for a ground platform. The distance D_j to the new platform is calculated and the cycle of new platforms inputted until it is able to come to a stop.

If it is not the final waypoint, it and all the remaining waypoints will have to be checked for self-consistency. To do this, the distance D_j is calculated, and from this, a test is done to see if the maximum speed $(2\ a_0\ D_j + v_j^2)^{1/2}$ exceeds v_{cru}. If it does, v_{max} is set to v_{cru}. The waypoint speed v_{j+1} is compared to v_{max}; if it is greater than v_{max}, it is set to v_{max}. Similarly, if $2\ a_0\ D_j - v_j^2$ is positive, v_{min} is set equal to 0, or if $2\ a_0\ D_j - v_j^2$ is negative, v_{min} is set equal to the square root of $v_j^2 - 2\ a_0\ D_j$. Then v_{j+1} is compared to v_{min} and set equal to v_{min} if v_{j+1} is less than v_{min}. A new quantity, v_{top}, is calculated, and if it exceeds the cruising speed v_{cru}, it is set to v_{cru}. If $_{vj+1}$ is non-zero, the loiter time is set to zero. Then $t_{a,j+1}$, t_1, t_2 and $t_{d,j+1}$ are calculated, and the waypoint index is increased by 1.

4.2.4 Changing Waypoint Data

The final process involves changing the waypoint speed, the arrival time $t_{a,j+1}$, or the loiter time of a single waypoint. Figure 37 summarizes this. The user first identifies the index and type of platform for which the waypoint data is to be changed. After reading the acceleration factor a_o and the top cruising speed v_{cru} from a database, the waypoint index is tested to see if it is the initial point. If it is, the process prompts the user to input a new loiter time. The arrival and departure times are then changed for the remaining waypoints by the difference of the old and new loiter times in the first point. It is unnecessary to change the other velocities because the first point velocity must remain zero. The process then stops. Should this not be the initial waypoint, the index is tested to see if it is the final waypoint. If it is, the process stops since the last waypoint must have a waypoint velocity v_{j+1} of zero, and it is unnecessary to change the arrival time.

If the index is not for the final waypoint, the distance from the last waypoint D_j is recalculated. If there is enough distance for the platform to reach v_{cru}, $(2\ a_0\ D_j + v_j^2)^{1/2}$, then v_{max} is assigned the value of v_{cru}, and the corresponding t_{min} is calculated with equation 101. If there isn't enough distance, then maximum speed v_{max} is calculated with equation 104, with the corresponding t_{min} calculated with equation 105. Next, if there is not enough distance for the platform to reach zero

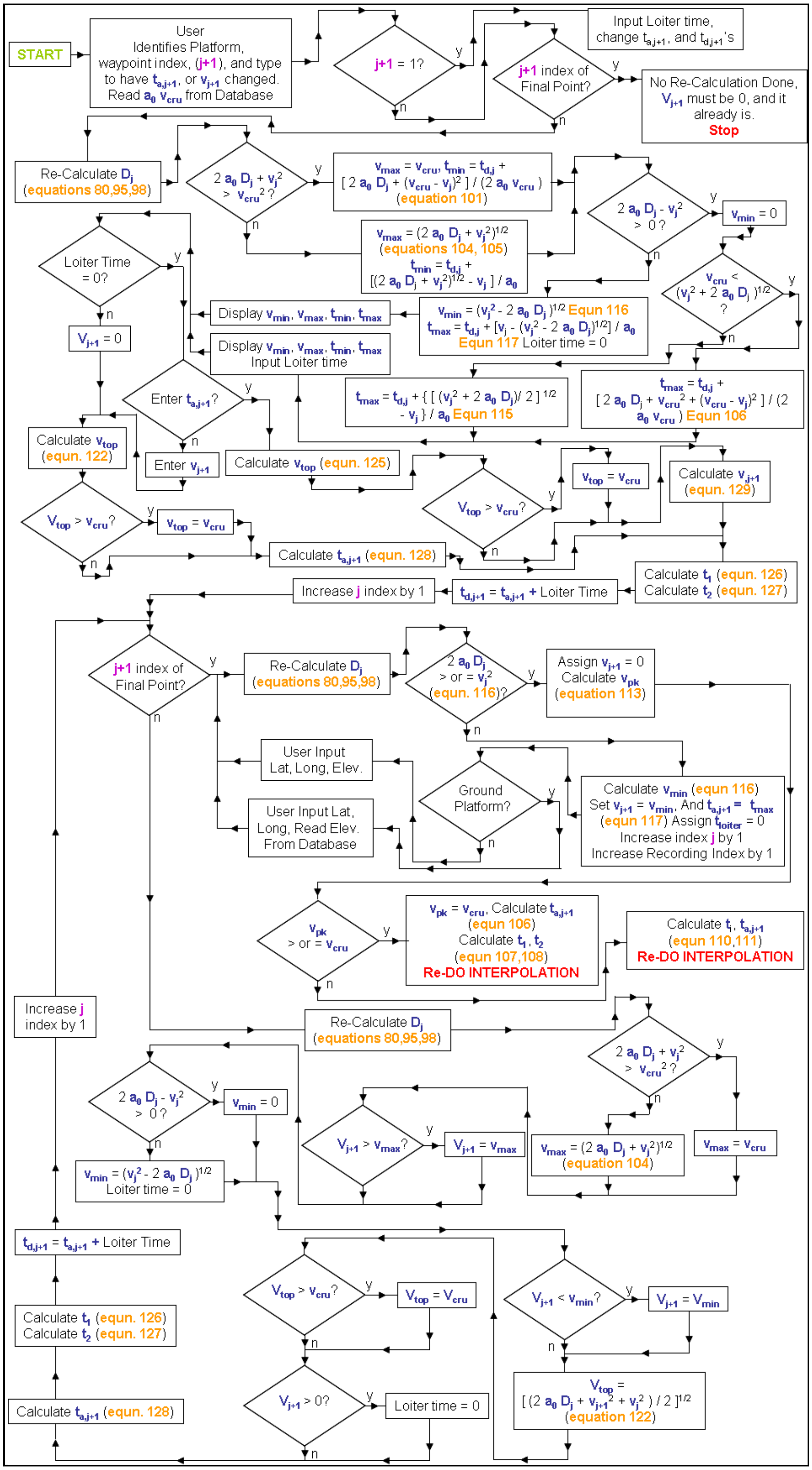

START
User Identifies Platform, waypoint index, (j+1), and type to have $t_{a,j+1}$, or v_{j+1} changed. Read a_0 v_{cru} from Database
j+1 = 1?
j+1 index of Final Point?
Input Loiter time, change $t_{a,j+1}$, and $t_{d,j+1}$'s
No Re-Calculation Done, V_{j+1} must be 0, and it already is. Stop
Re-Calculate D_j (equations 80,95,98)
$2 a_0 D_j + v_j^2 > v_{cru}^2$?
$v_{max} = v_{cru}$, $t_{min} = t_{d,j} + [2 a_0 D_j + (v_{cru} - v_j)^2] / (2 a_0 v_{cru})$ (equation 101)
$v_{max} = (2 a_0 D_j + v_j^2)^{1/2}$ (equations 104, 105) $t_{min} = t_{d,j} + [(2 a_0 D_j + v_j^2)^{1/2} - v_j] / a_0$
$2 a_0 D_j - v_j^2 > 0$?
$v_{min} = 0$
$v_{cru} < (v_j^2 + 2 a_0 D_j)^{1/2}$?
Loiter Time = 0?
$V_{j+1} = 0$
Display v_{min}, v_{max}, t_{min}, t_{max}
Display v_{min}, v_{max}, t_{min}, t_{max} Input Loiter time
$v_{min} = (v_j^2 - 2 a_0 D_j)^{1/2}$ Equn 116 $t_{max} = t_{d,j} + [v_j - (v_j^2 - 2 a_0 D_j)^{1/2}] / a_0$ Equn 117 Loiter time = 0
Enter $t_{a,j+1}$?
$t_{max} = t_{d,j} + \{ [(v_j^2 + 2 a_0 D_j)/ 2]^{1/2} - v_j \} / a_0$ Equn 115
$t_{max} = t_{d,j} + [2 a_0 D_j + v_{cru}^2 + (v_{cru} - v_j)^2] / (2 a_0 v_{cru})$ Equn 106
Calculate v_{top} (equn. 122)
Enter v_{j+1}
Calculate v_{top} (equn. 125)
$v_{top} = v_{cru}$
Calculate v_{j+1} (equn. 129)
$V_{top} > v_{cru}$?
$V_{top} > v_{cru}$?
$v_{top} = v_{cru}$
Calculate $t_{a,j+1}$ (equn. 128)
Calculate t_1 (equn. 126) Calculate t_2 (equn. 127)
Increase j index by 1
$t_{d,j+1} = t_{a,j+1}$ + Loiter Time
j+1 index of Final Point?
Re-Calculate D_j (equations 80,95,98)
$2 a_0 D_j$ > or = v_j^2 (equn. 116)?
Assign $v_{j+1} = 0$ Calculate v_{pk} (equation 113)
User Input Lat, Long, Elev.
User Input Lat, Long, Read Elev. From Database
Ground Platform?
Calculate v_{min} (equn 116) Set $v_{j+1} = v_{min}$, And $t_{a,j+1} = t_{max}$ (equn 117) Assign $t_{loiter} = 0$ Increase index j by 1 Increase Recording Index by 1
v_{pk} > or = v_{cru}
$v_{pk} = v_{cru}$, Calculate $t_{a,j+1}$ (equn 106) Calculate t_1, t_2 (equn 107,108) Re-DO INTERPOLATION
Calculate t_i, $t_{a,j+1}$ (equn 110,111) Re-DO INTERPOLATION
Increase j index by 1
Re-Calculate D_j (equations 80,95,98)
$2 a_0 D_j + v_j^2 > v_{cru}^2$?
$2 a_0 D_j - v_j^2 > 0$?
$v_{min} = 0$
$V_{j+1} > v_{max}$?
$V_{j+1} = v_{max}$
$v_{max} = (2 a_0 D_j + v_j^2)^{1/2}$ (equation 104)
$v_{max} = v_{cru}$
$v_{min} = (v_j^2 - 2 a_0 D_j)^{1/2}$ Loiter time = 0
$t_{d,j+1} = t_{a,j+1}$ + Loiter Time
Calculate t_1 (equn. 126) Calculate t_2 (equn. 127)
Calculate $t_{a,j+1}$ (equn. 128)
$V_{top} > v_{cru}$?
$V_{top} = V_{cru}$
$V_{j+1} < v_{min}$?
$V_{j+1} = V_{min}$
$V_{j+1} > 0$?
Loiter time = 0
$V_{top} = [(2 a_0 D_j + v_{j+1}^2 + v_j^2) / 2]^{1/2}$ (equation 122)
y
n

Figure 37. Overview of the process of changing the waypoint data.

velocity, the minimum velocity v_{min} is calculated with equation 116, with the corresponding t_{max} calculated with equation 117, and the loiter time is set to zero, and t_{min}, t_{max}, v_{min}, and v_{max} are displayed to the user as guidance. Otherwise, v_{min} is set equal to zero, and the maximum corresponding arrival time t_{max} is calculated with either equation 106 (if the maximum speed v_{max} is v_{cru}) or with equation 115 (if the maximum speed v_{max} is v_{cru}). The values t_{min}, t_{max}, v_{min}, and v_{max} are displayed to the user as guidance. If v_{min} is zero, the user is prompted to input the loiter time.

The loiter time is then tested. If the loiter time is not zero, the waypoint speed v_{j+1} is set to zero. The highest speed possible for the platform between waypoints v_{top} is calculated with equation 122, and if it exceeds v_{cru}, it is set to v_{cru}, and $t_{a,j+1}$ is computed with equation 128. If the loiter time is zero, the user is given the option of entering the arrival time $t_{a,j+1}$ or the waypoint velocity, v_{j+1}. If the user chooses to enter $t_{a,j+1}$, then v_{top} is calculated with equation 125, and if it exceeds v_{cru}, it is set to v_{cru}, and v_{j+1} is computed with equation 129. If the user chooses to enter v_{j+1}, then again v_{top} is calculated with equation 122, and if it exceeds v_{cru}, it is set to v_{cru}, and $t_{a,j+1}$ is computed with equation 128. Then t_1, t_2, and $t_{d,j+1}$ are calculated using equations 126 and 127, and the waypoint index is increased by 1.

The index is again tested to see if it corresponds to the final waypoint. If this is the final waypoint, the distance D_j is calculated, and if there is enough distance for the platform to stop ($2\ a_0\ D_j > v_j^2$), the final waypoint velocity v_{j+1} is assigned to zero and the peak speed v_{pk} is calculated with equation 113. If the peak speed v_{pk} exceeds v_{cru}, then v_{pk} is assigned the value of v_{cru}, and the arrival times $t_{a,j+1}$, t_1, and t_2 are calculated with equations 75, 107, and 108 (see figure 29) and the interpolation is redone. Otherwise (see figure 30), v_{pk} remains unchanged and t_i (the i subscript represents 1 or 2, since in this case $t_1 = t_2$) and $t_{a,j+1}$ are calculated with equations 110 and 111, and the interpolated points are recalculated. If there is not enough distance for the platform to stop, the speed v_{min} is calculated with equation 116 and the speed v_{j+1} is assigned the value for v_{min}, the arrival time $t_{a,j+1}$ (which equals t_{max}) is calculated with equation 117, t_{loiter} is assigned to 0, and the index j is increased by one, as is the recording index (which is the number of waypoints). The user is prompted to input the latitude, longitude, and elevation for an aerial platform, or only latitude and longitude for a ground platform. The distance D_j to the new platform is calculated and the cycle of new platforms inputted until the platform is able to come to a stop.

If it is not the final waypoint, it and all the remaining waypoints will have to be checked for self-consistency. To do this, the distance D_j is calculated, and from this, a test is done to see if the maximum speed $(2\ a_0\ D_j + v_j^2)^{1/2}$ can exceed v_{cru}. If it can, v_{max} is set to v_{cru}; otherwise, v_{max} is calculated with equation 104. The waypoint speed, v_{j+1}, is compared to v_{max}—if it is greater than v_{max}, it is set to v_{max}. Otherwise v_{j+1} remains unchanged. Similarly, if $2\ a_0\ D_j – v_j^2$ is positive, v_{min} is set equal to 0, or $(v_j^2 – 2\ a_0\ D_j)^{1/2}$ if $2\ a_0\ D_j – v_j^2$ is negative, and the loiter time is set to zero. Then v_{j+1} is compared to v_{min} and set equal to v_{min} if v_{j+1} is less than v_{min}. Otherwise, v_{j+1} remains unchanged. A new variable, v_{top} is calculated with equation 122. If it exceeds v_{cru}, it is set to v_{cru}. Otherwise, it remains unchanged. If v_{j+1} is nonzero, the loiter time is set to zero. Then $t_{a,j+1}$, t_1, t_2, and $t_{d,j+1}$ are calculated with equations 126–128, the waypoint index is increased by 1, and the departure time $t_{d,j+1}$ is calculated.

4.2.5 If a Platform Has Only One Waypoint

We now consider the input process and the four modification processes (moving a waypoint, adding a waypoint, removing a waypoint, and reassigning values to the waypoint) when a platform has only one waypoint assigned to it. When a platform has only one waypoint, the waypoint velocity (v_1) is assigned to zero. The arrival time $t_{a,1}$ is also assigned to zero. The departure time $t_{d,1}$ is equal to the loiter time, which equals the mission time. If a platform has only one waypoint, we assume that it does not move but remains stationary for the entire run. As a result, the interpolation process for this platform need only copy the initial waypoint latitude, longitude, and elevation for every snapshot time to the snapshot database.

Inputting data for a single waypoint platform is relatively straightforward and is set out in figure 38. The user identifies the type of platform that requires a single waypoint, and the default index (as well as the recording index) is set to 1. The arrival time $t_{a,1}$ and the waypoint velocity v_1 are set to zero. The loiter time becomes the mission time, so that the departure time $t_{d,1}$ is set equal to the loiter time. If the platform is a ground platform, the user inputs the latitude and longitude, while the elevation is read from a DTED database. If it is not a ground platform, the user also inputs the elevation. The interpolation is done by copying the latitude, longitude, and elevation database to the snapshot database for this platform.

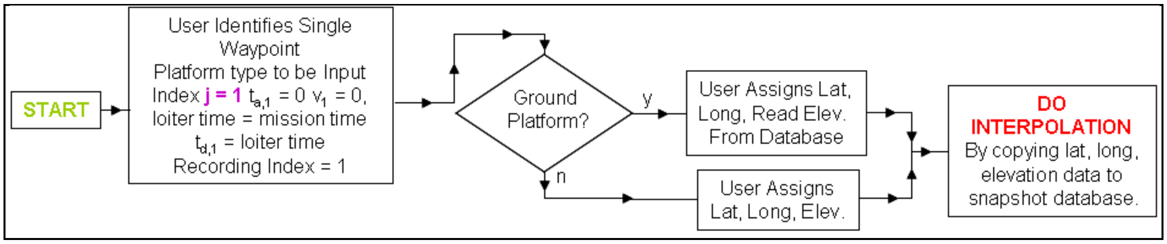

START
User Identifies Single Waypoint
Platform type to be Input
Index j = 1 $t_{a,1}$ = 0 v_1 = 0,
loiter time = mission time
$t_{d,1}$ = loiter time
Recording Index = 1
Ground Platform?
y
n
User Assigns Lat, Long, Read Elev. From Database
User Assigns Lat, Long, Elev.
DO INTERPOLATION
By copying lat, long, elevation data to snapshot database.

Figure 38. The process of inputting data for a platform with one waypoint.

Moving a waypoint (when a platform has a single point) only requires a reassignment of a new latitude and longitude. Once the user identifies the platform, the index is assumed to be 1. If it is a ground platform, the user inputs the new latitude and longitude. If it is not, the user inputs the new latitude, longitude, and elevation. Since that is all that is required, the process ends. This brief process is outlined in figure 39.

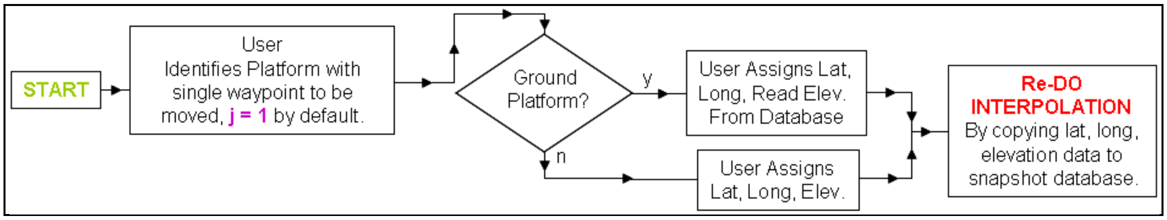


Figure 39. The process of inputting data for changing the location of a single waypoint platform.

Adding a point is a bit more involved. After identifying the platform on which to add a waypoint, the user identifies whether the added point comes before the previously defined point in time (the new point has an index of j = 1) or after it (the new point has an index of j = 2). The platform acceleration a_0 and cruising speed v_{cru} are read from a database, and the recording index is set to 2. If it is a ground platform, the user inputs the new latitude and longitude. If it is not, the user inputs the new latitude, longitude, and elevation. The two points then (if necessary) have their indices modified so that the first point in time has an index of 1 and the second waypoint in time has an index of 2. Since we now have two distinct waypoints, the process continues as outlined in figure 35. The entire process is laid out in figure 40. The verbiage describing the remaining part of the process is identical to the verbiage used to describe figure 35. For the sake of brevity, it will not be repeated here.

Removing a data point for a platform with only one data point is tantamount to deleting the platform. So instead of outlining a separate process to remove the only waypoint to a platform, the process will be to simply delete the platform. Likewise, reassigning waypoint data for a platform with only one waypoint is unnecessary since there is only one waypoint. The single waypoint must have the waypoint velocity v_1 and arrival time $t_{a,1}$ equal to zero, so the data can not be changed. Consequently, this process will not be included in the deployment module.

4.3 The Interpolation Process

Once the latitudes, longitudes, and elevations have been established for the set of waypoints for a platform (with their corresponding arrival and departure times either assigned directly by the user or calculated from assigned waypoint velocities and loiter times), then it is possible to calculate the platform's latitude, longitude, and elevation for a selected snapshot time from the interpolation time t_i. The process is to first calculate the interpolated distance D_i from the most recent waypoint the platform encountered to the platform's position at t_i. The interpolated distance D_i makes it possible to obtain the interpolated elevation E_i. The interpolated angle $\Delta\theta_i$ that the interpolated location of the platform at time t_i makes with the location of the last waypoint at the Earth's center then makes it possible to calculate the interpolated longitude θ_i and latitude ϕ_i after making a coordinate transformation to simplify the use to the quantity $\Delta\theta_i$. Each of these processes will be explained in detail.

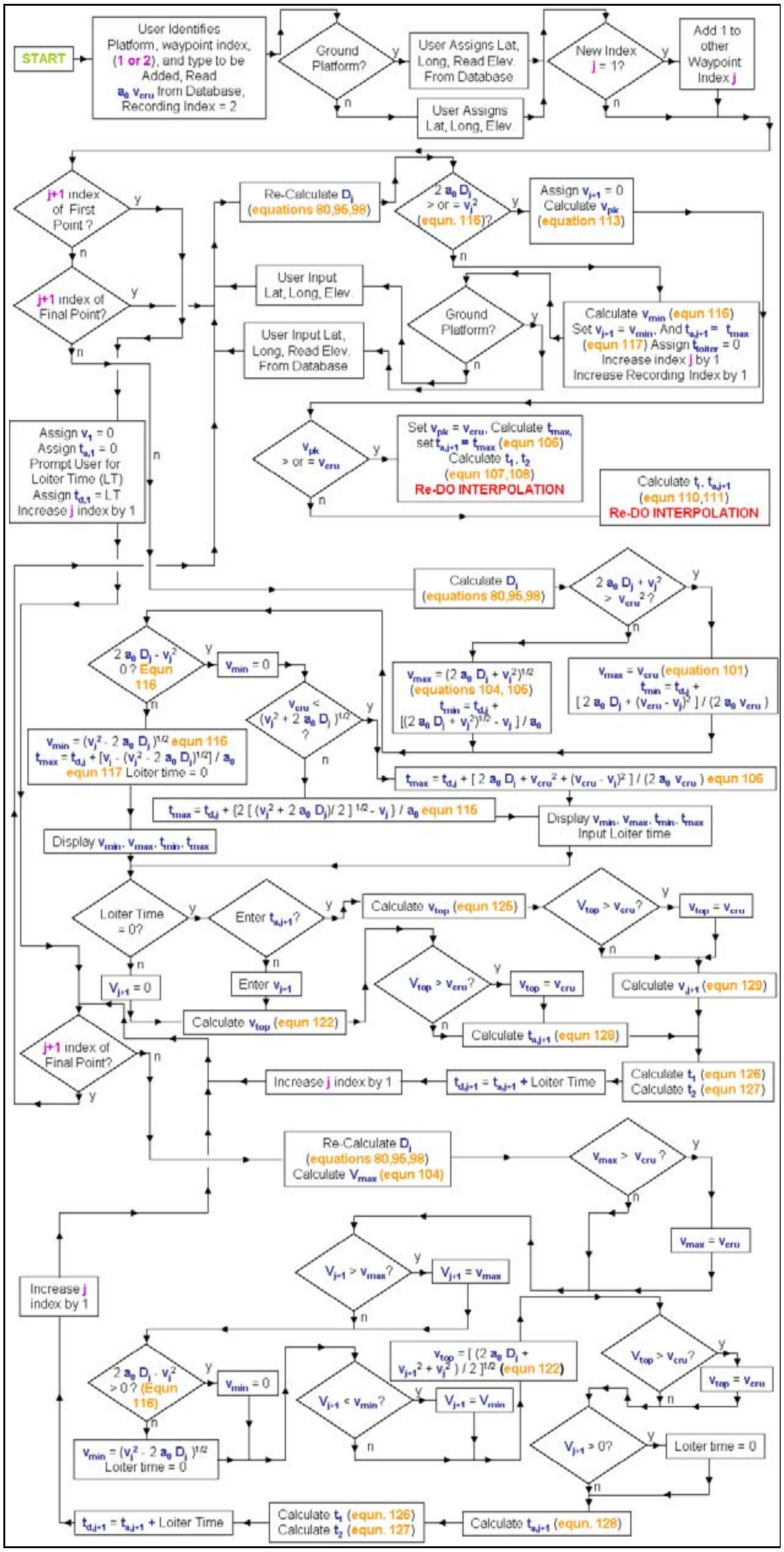

START
User Identifies Platform, waypoint index, (1 or 2), and type to be Added, Read a_0 v_{cru} from Database, Recording Index = 2
Ground Platform?
User Assigns Lat, Long, Read Elev. From Database
User Assigns Lat, Long, Elev.
New Index j = 1?
Add 1 to other Waypoint Index j
j+1 index of First Point ?
j+1 index of Final Point?
Re-Calculate D_j (equations 80,95,98)
$2\ a_0\ D_j$ > or = v_j^2 (equn. 116)?
Assign $v_{j+1} = 0$ Calculate v_{pk} (equation 113)
User Input Lat, Long, Elev.
User Input Lat, Long, Read Elev. From Database
Ground Platform?
Calculate v_{min} (equn 116) Set $v_{j+1} = v_{min}$. And $t_{a,j+1} = t_{max}$ (equn 117) Assign $t_{loiter} = 0$ Increase index j by 1 Increase Recording Index by 1
Assign $v_1 = 0$ Assign $t_{a,1} = 0$ Prompt User for Loiter Time (LT) Assign $t_{d,1}$ = LT Increase j index by 1
v_{pk} > or = v_{cru}
Set $v_{pk} = v_{cru}$. Calculate t_{max}, set $t_{a,j+1} = t_{max}$ (equn 106) Calculate t_1, t_2 (equn 107,108) Re-DO INTERPOLATION
Calculate t_i, $t_{a,j+1}$ (equn 110,111) Re-DO INTERPOLATION
Calculate D_j (equations 80,95,98)
$2\ a_0\ D_j + v_j^2 > v_{cru}^2$?
$2\ a_0\ D_j - v_j^2$ 0 ? Equn 116
$v_{min} = 0$
$v_{cru} <$ $(v_j^2 + 2\ a_0\ D_j)^{1/2}$?
$v_{max} = (2\ a_0\ D_j + v_j^2)^{1/2}$ (equations 104, 105) $t_{min} = t_{d,j} +$ $[(2\ a_0\ D_j + v_j^2)^{1/2} - v_j] / a_0$
$v_{max} = v_{cru}$ (equation 101) $t_{min} = t_{d,j} +$ $[\ 2\ a_0\ D_j + (v_{cru} - v_j)^2\ ] / (2\ a_0\ v_{cru}\ )$
$v_{min} = (v_j^2 - 2\ a_0\ D_j\ )^{1/2}$ equn 116 $t_{max} = t_{d,j} + [v_j - (v_j^2 - 2\ a_0\ D_j)^{1/2}] / a_0$ equn 117 Loiter time = 0
$t_{max} = t_{d,i} + [\ 2\ a_0\ D_j + v_{cru}^2 + (v_{cru} - v_j)^2\ ] / (2\ a_0\ v_{cru}\ )$ equn 106
$t_{max} = t_{d,j} + \{2\ [\ (v_j^2 + 2\ a_0\ D_j)/\ 2\ ]^{1/2} - v_j\ \} / a_0$ equn 115
Display v_{min}, v_{max}, t_{min}, t_{max} Input Loiter time
Display v_{min}, v_{max}, t_{min}, t_{max}
Loiter Time = 0?
Enter $t_{a,j+1}$?
Calculate v_{top} (equn 125)
$V_{top} > v_{cru}$?
$V_{top} = v_{cru}$
$V_{j+1} = 0$
Enter v_{j+1}
Calculate v_{top} (equn 122)
$V_{top} > v_{cru}$?
$V_{top} = v_{cru}$
Calculate v_{j+1} (equn 129)
Calculate $t_{a,j+1}$ (equn 128)
j+1 index of Final Point?
Increase j index by 1
$t_{d,j+1} = t_{a,j+1}$ + Loiter Time
Calculate t_1 (equn 126) Calculate t_2 (equn 127)
Re-Calculate D_j (equations 80,96,98) Calculate V_{max} (equn 104)
$v_{max} > v_{cru}$?
$v_{max} = v_{cru}$
Increase j index by 1
$V_{j+1} > v_{max}$?
$V_{j+1} = v_{max}$
$2\ a_0\ D_j - v_j^2$ > 0 ? (Equn 116)
$v_{min} = 0$
$V_{j+1} < v_{min}$?
$V_{j+1} = V_{min}$
$v_{top} = [\ (2\ a_0\ D_j +$ $v_{j+1}^2 + v_j^2\ ) / 2\ ]^{1/2}$ (equn 122)
$V_{top} > v_{cru}$?
$v_{top} = v_{cru}$
$v_{min} = (v_j^2 - 2\ a_0\ D_j\ )^{1/2}$ Loiter time = 0
$V_{j+1} > 0$?
Loiter time = 0
$t_{d,j+1} = t_{a,j+1}$ + Loiter Time
Calculate t_1 (equn. 126) Calculate t_2 (equn. 127)
Calculate $t_{a,j+1}$ (equn. 128)

Figure 40. The process of adding a waypoint to a platform that has one waypoint.

4.3.1 Calculate the Interpolated Distance

Assuming that we know the interpolated time t_i, the first task is to see if any of the waypoint loiter periods (the time between the arrival time $t_{j,a}$ and departure time $t_{d,j}$) bound the interpolated time t_i. Put another way, is $t_{a,j} < t_i < t_{d,j}$? If it is, then the interpolated latitude, longitude, and elevation for the platform are set equal to the latitude, longitude, and elevation of the waypoint. This ends the interpolation process for the interpolated time t_i. (Note that the last waypoint loiter period is the span between the final waypoint arrival time, $t_{N,a}$ and the ending time of the run.)

If the interpolated time t_i does not occur during the waypoint loiter time, then it occurs during the transit time between points. The interpolation process then becomes more involved. To calculate the interpolated distance D_i, it is first necessary to compare it with the times that mark the inflection points t_1 and t_2 of the platform profile of figure 24. As a review, we recount how the inflection points are calculated. Step 1 is to calculate v_{top} from equation 125, restated here as equation 130.

$$v_{top} = v_j + a_0 \, (t_{a,j+1} - t_{d,j}) - [\, v_j \, a_0 \, (t_{a,j+1} - t_{d,j}) - a_0 \, D_j + (1 / 2) \, a_0^2 \, (t_{a,j+1} - t_{d,j})^2 \,]^{1/2} . \quad (130)$$

If v_{top} exceeds v_{cru}, then v_{top} is set equal to v_{cru}. Otherwise, equation 130 defines v_{cru}. Equations 126 and 127 (repeated here as equations 131 and 132) are then used to establish the inflection times t_1 and t_2.

$$t_1 = t_{d,j} + (v_{top} - v_j) / a_0 . \quad (131)$$

$$t_2 = t_{a,j+1} - (v_{top} - v_{j+1}) / a_0 . \quad (132)$$

To calculate the distance D_i from the most recently departed waypoint, it is necessary to established the relationship between the departure time $t_{d,j}$, the inflection times t_1 and t_2, the arrival time at the next waypoint $t_{a,j+1}$, and the interpolation time t_i. The next three equations (133–135) are taken from equations 64, 67, and 70.

$$\text{If } t_{d,j} < t_i < t_1, \; D_i = a_0 \, (t_i^2 - t_{d,j}^2)/ 2 - (a_0 \, t_{d,j} - v_j) \, (t_i - t_{d,j}) . \quad (133)$$

$$\text{If } t_1 < t_i < t_2, \; D_i = a_0 \, (t_1^2 - t_{d,j}^2)/ 2 - (a_0 \, t_{d,j} - v_j) \, (t_1 - t_{d,j}) + v_{cru} \, (t_i - t_1) . \quad (134)$$

$$\text{If } t_2 < t_i < t_{a,j+1}, \; D_i = a_0 \, (t_1^2 - t_{d,j}^2)/ 2 - (a_0 \, t_{d,j} - v_j) \, (t_1 - t_{d,j}) + v_{cru} \, (t_2 - t_1) + v_{cru} \, (t_i - t_2) - a_0 \, (t_i^2 - t_2^2) / 2 + a_0 \, t_2 \, (t_i - t_2) . \quad (135)$$

Now that we have the interpolated distance D_i, we can calculate the interpolated elevation, E_i.

4.3.2 Calculate the Interpolated Elevation

If the interpolated time t_i does not occur during the waypoint loiter time, it is possible to calculate the interpolated elevation E_i as a result of one of two circumstances. First, the two waypoints bounding the platform in time have the same elevation. In that case, $E_i = E_j = E_{j+1}$.

But should E_j not equal E_{j+1}, the interpolated elevation must be found using a two-step process. First reconsider equation 97 by eliminating then solving for the elevation constant k:

$$k = (E_{j+1} - E_j) / D_j \,. \tag{136}$$

Note that this calculation is made with the distance between the waypoints D_j, not the interpolated distance D_i. To obtain the interpolated elevation E_i, we use equation 97 again but substitute the interpolated distance D_i (calculated in the previous section) for the interwaypoint distance D_j, and the interpolated elevation E_i for the arrival waypoint elevation E_{j+1}. Rearranging the terms we find

$$k\, D_i + E_j = E_i \,. \tag{137}$$

With the interpolated distance D_i and the interpolated elevation E_i, we may next find the interpolated angle between the line connecting the most recently encountered waypoint and the center of the Earth, and the line connecting the platform's position at the interpolated time t_i and the center of the Earth.

4.3.3 Calculate the Interpolated Angle $\Delta\theta_i$

Again, we assume that the interpolated time t_i does not occur during the waypoint loiter time. And again, there are two circumstances we need to consider in the calculation of the interpolated angle $\Delta\theta_i$: the elevations of the waypoints bounding the interpolated point are either equal or different.

First, if the elevations of the waypoints bounding the interpolated point in time are equal, we make use of equation 99, substituting the interpolated angle $\Delta\theta_i$ for the angle the waypoints make at the Earth's center $\Delta\theta$, and the distance to the interpolated point D_i for the distance between the waypoint D_j. When we solve for $\Delta\theta_i$,

$$D_i / (R_{ave} + E_j) = \Delta\theta_i \,. \tag{138}$$

Second, if the elevations of the waypoints bounding the interpolate point in time are not equal, we use equation 94 instead. Making use of the interpolated elevation E_i and the constant k, and substituting the interpolated elevation E_i for the waypoint elevation E_{j+1} and solving for $\Delta\theta_i$, we find

$$\Delta\theta_i = [(1 - k^2)^{1/2} / k]\ \mathrm{Ln}\ [(R_{ave} + E_i) / (R_{ave} + E_j)] \,. \tag{139}$$

Ln denotes the natural log function. Note that if the platform is descending, the elevation constant k will be negative (see equation 136). If the platform is descending, then E_j will be greater than E_i, causing the natural log of the expression to be negative. We assume that we take the positive root for $[(1 - k^2)^{1/2}$, resulting in a positive interpolation angle $\Delta\theta_i$. Should the platform be ascending, k and the natural log will be positive, again resulting in a positive interpolation angle $\Delta\theta_i$. Note that when the elevation does not change (the situation described by equation 138), the interpolation angle $\Delta\theta_i$ is positive because the distance interpolated D_i, the

Earth's average radius R_{ave}, and the waypoint elevation E_j are always positive. The fact that the interpolation angle $\Delta\theta_i$ is always positive will play a vital role in determining the interpolated longitude and latitude.

4.3.4 Calculate the Interpolated Latitude and Longitude

To calculate the interpolated longitude and latitude involves doing a coordination transformation such that the two waypoints bounding the interpolated point in time will be on the transformed equator. Hence, the interpolated point will also be on the transformed equator, which will be calculated by using the interpolated angle $\Delta\theta_i$. By reversing the process, we can obtain the interpolated longitude and latitude.

The transformation will be done as a two-part process. The first part involves keeping the latitude constant but changing the longitude. This process is illustrated in figure 41. The new x axis (denoted x' in the figure) begins at the Earth's center and intersects the point on the equator where the line of constant latitude θ_0 crosses the equator. The y' axis begins at the Earth's center, is in the equatorial plane, and is 90° from the x' axis. The axis is occulted by the Earth in the figure, although the original y axis (also in the equatorial plane and beginning at the Earth's center) is not occulted and is plainly shown in the figure. The z and z' axis are the same and start at the Earth's center and progress through the North Pole.

Recall that the parameters ϕ_{max} and θ_0 describe the equation of a great circle and are derived from the latitudes and longitudes of the two bounding waypoints in equations 10 and 11, restated here as equations 140 and 141.

$$\theta_0 = \text{arc tan } [(\tan \phi_j \cos \theta_{j+1} - \tan \phi_{j+1} \cos \theta_j) / (\tan \phi_{j+1} \sin \theta_j - \tan \phi_j \sin \theta_{j+1})] \; . \qquad (140)$$

$$\phi_{max} = \text{arc tan } [\tan \phi_{j+1} / \cos (\theta_{j+1} - \theta_0)] = \text{arc tan } [\tan \phi_j / \cos (\theta_j - \theta_0)] \; . \qquad (141)$$

Recall, too, that this assumes that the argument for arc tan is finite in equation 140, and that we take the positive value for ϕ_{max} for the value of θ_0 calculated in equation 140. Additionally, the point (ϕ_{max},θ_0) in the nonprimed coordinate system is the point of highest latitude on the great circle. Since we want the line of constant latitude that goes through this point to be the prime meridian in the primed system, the transformed coordinate will be $(\phi_{max}',0°)$.

So, the effect of the first part of the transformation will be simply to change the longitude by the quantity θ_0 while allowing the latitude to remain unchanged. A way to visualize this transformation is to remove the Earth and look along the z axis in the –z direction. Figure 42 illustrates the transformation. From the figure, it is easy to obtain the transformation mathematically for not only the interpolated points, but for any point that includes the waypoints bounding the interpolated point:

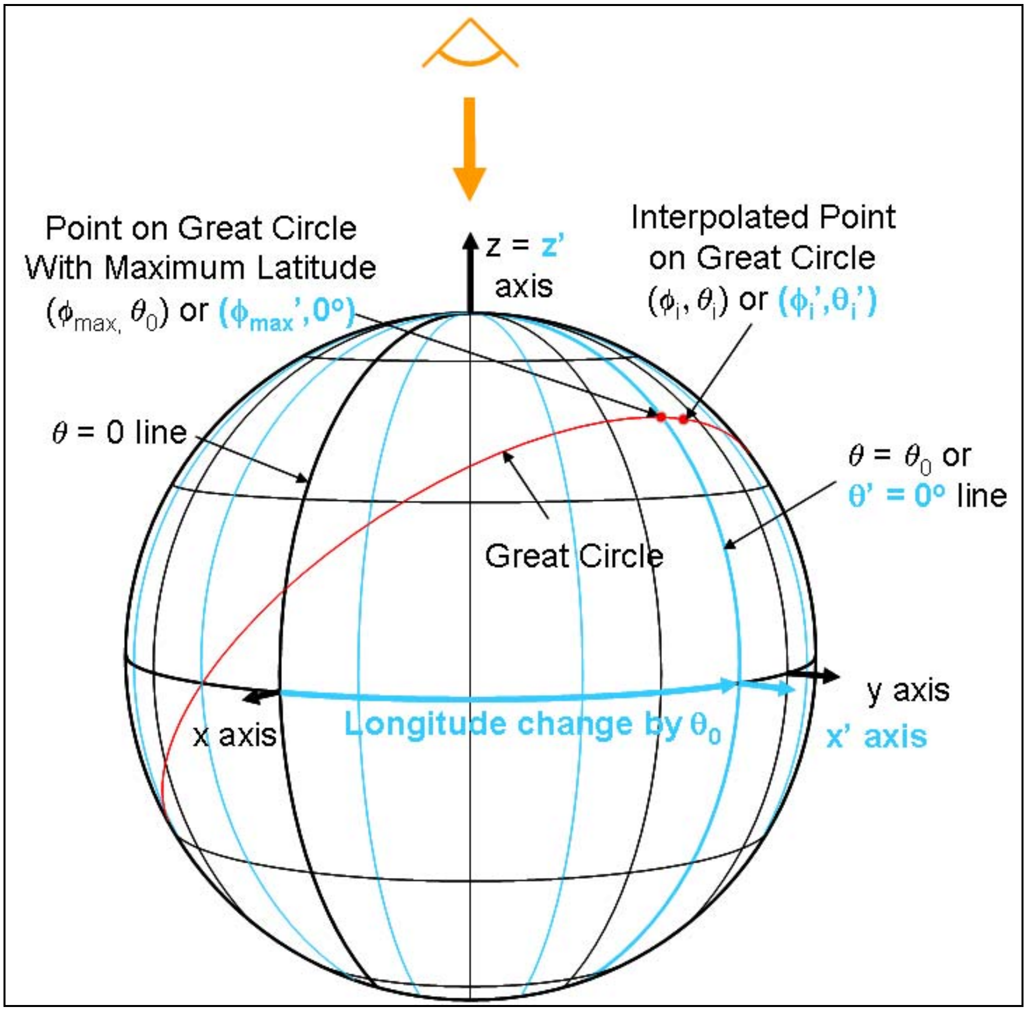


Figure 41. The first step in transforming the interpolated point to a new coordinate system.

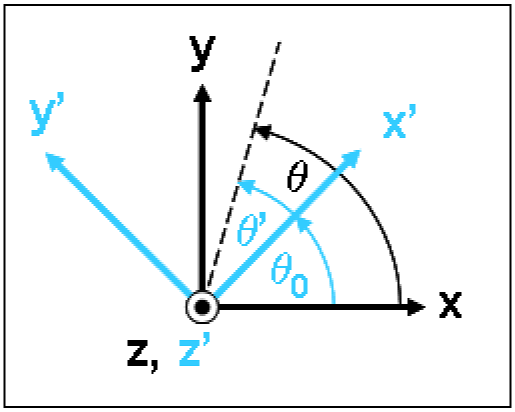


Figure 42. The first step in transforming the interpolated point to a new coordinate system with the Earth removed viewing in the –z direction.

$$\phi_j' = \phi_j \, . \tag{142}$$

$$\phi_{j+1}' = \phi_{j+1} \, . \tag{143}$$

$$\phi_i' = \phi_i \, . \tag{144}$$

$$\theta_j' = \theta_j - \theta_0 \, . \tag{145}$$

$$\theta_{j+1}' = \theta_{j+1} - \theta_0 \, . \tag{146}$$

$$\theta_i' = \theta_i - \theta_0 \, . \tag{147}$$

We must make sure that the three transformed longitudes θ_j', θ_{j+1}', and θ_i' are in the range –180° to +180°. If any of them are less than –180°, we add 360°. If any of them exceed +180°, we subtract 360°.

The second part of the transformation is more complicated. It involves defining a set of double-primed coordinates such that the primed and double-primed y axis are the same. Furthermore, the x" axis begins at the center of the sphere and goes through the point (ϕ_{max},θ_0) in the original system and (ϕ_{max},0) in the single-primed system. This line intersects the double-primed equator and the prime meridian. The transformation is illustrated in figure 43.

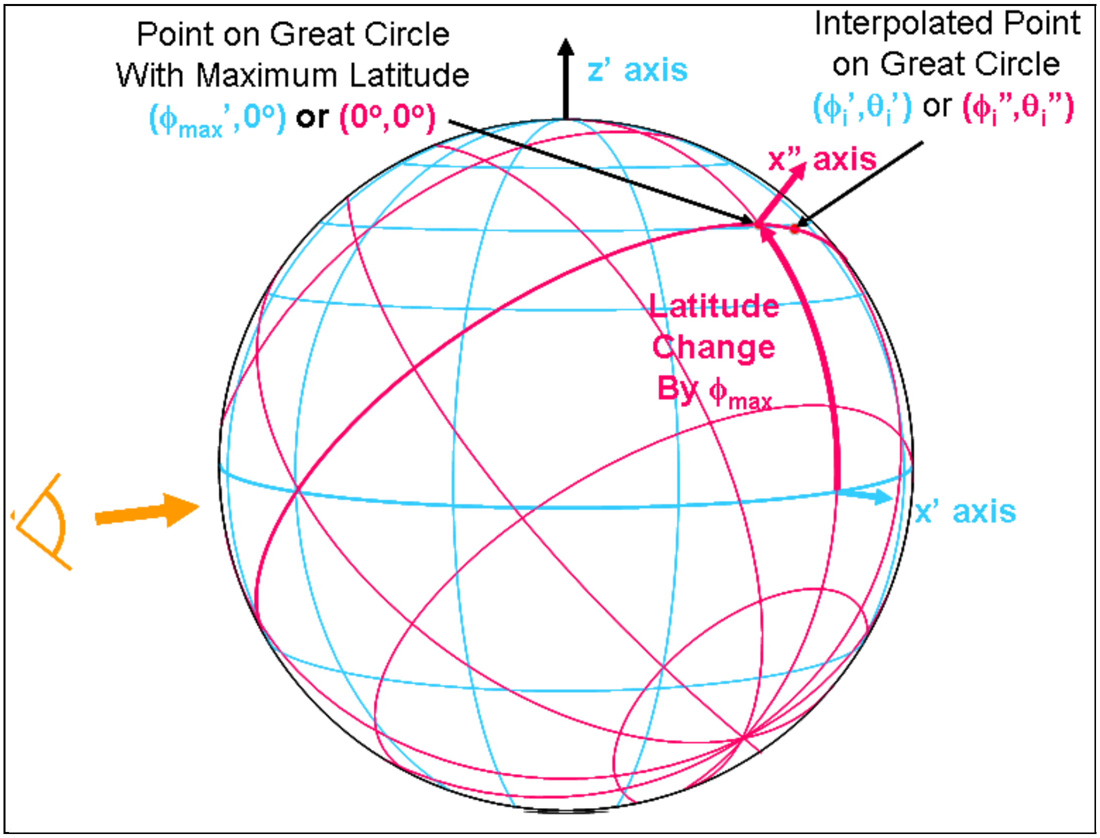


Figure 43. The second step in transforming the interpolated point to a new coordinate system: the y' axis acts as the axel so that y' = y".

To obtain the quantitative relationship between the primed and double-primed coordinate systems, we view along the y'/y" axis in the positive y'/y" direction with the Earth removed for clarity. Figure 44 shows this situation. The fact that the waypoints and the interpolated points lie on the equator in the double-primed coordinate system simplifies the transformation. The next few equations will pertain to any point on the double-primed equator. As the figure shows, any point on the equator has an x" coordinate and a y" coordinate. Because the point is not the transformed double-primed equator, it has no z" coordinate. Note that y' = y", so that relationship is trivial. Concentrating on the relationship between the x'' coordinate and the z' and x' coordinate, we see from the figure that

$$z' = x'' \sin \phi_{max} \tag{148}$$

and

$$x' = x'' \cos \phi_{max} \, . \tag{149}$$

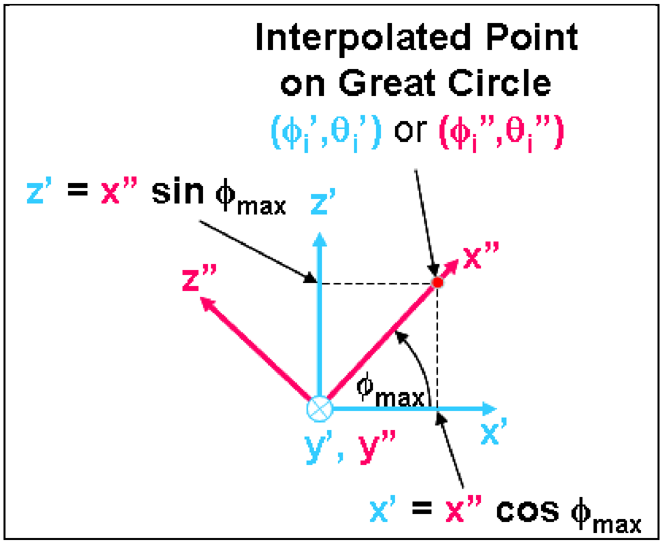

Interpolated Point
on Great Circle
(ϕ_i',θ_i') or (ϕ_i'',θ_i'')
$z' = x'' \sin \phi_{max}$
z'
z"
x"
ϕ_{max}
y', y"
x'
$x' = x'' \cos \phi_{max}$

Figure 44. The second step in transforming the interpolated point to a new coordinate system with the Earth removed viewing in the +y' direction.

Making use of equations 1–3 for the relationship between spherical and rectangular coordinates, and using the R_{ave}, the Earth's average radius, and the elevation E, we may rewrite equations 148 and 149 for any point on the great circle as

$$(R_{ave} + E) \sin \phi' = (R_{ave} + E) \cos \phi'' \cos \theta'' \sin \phi_{max} \tag{150}$$

and

$$(R_{ave} + E) \cos \phi' \cos \theta' \quad = (R_{ave} + E) \cos \phi'' \cos \theta'' \cos \phi_{max} \, , \tag{151}$$

and the fact that the points on the double-primed equator have $\phi'' = 0$ and we may divide by the common factor ($R_{ave} + E$) renders the relationships between the primed and double-primed latitudes and longitudes as

$$\sin \phi' = \cos \theta'' \sin \phi_{max} \tag{152}$$

and

$$\cos \phi' \cos \theta' = \cos \theta'' \cos \phi_{max} . \tag{153}$$

Substituting the relationships between the original and the double-primed coordinates as outlined in equations 142 to 147, and solving for the double-primed longitude θ'', we find for the jth waypoint, the j+1th waypoint, and the interpolated point such that

$$\theta_j'' = \arccos (\sin \phi_j / \sin \phi_{max}) . \tag{154}$$

$$\theta_j'' = \arccos [\cos \phi_j \cos (\theta_j - \theta_0) / \cos \phi_{max}] . \tag{155}$$

$$\theta_{j+1}'' = \arccos (\sin \phi_{j+1} / \sin \phi_{max}) . \tag{156}$$

$$\theta_{j+1}'' = \arccos [\cos \phi_{j+1} \cos (\theta_{j+1} - \theta_0) / \cos \phi_{max}] . \tag{157}$$

$$\theta_i'' = \arccos (\sin \phi_i / \sin \phi_{max}) . \tag{158}$$

$$\theta_i'' = \arccos [\cos \phi_i \cos (\theta_i - \theta_0) / \cos \phi_{max}] . \tag{159}$$

The next relationship needed for the transformation takes advantage of the fact that the primed and double-primed coordinate systems share the same y axis. So in general, $y' = y''$. Using the definition of equation 2, we may establish that

$$y' = (R_{ave} + E) \cos \phi' \sin \theta' = y'' = (R_{ave} + E) \cos \phi'' \sin \theta''. \tag{160}$$

For the case where the point is on the double-primed equator, we may take ϕ'' as zero. After dividing by ($R_{ave} + E$), we see for a point on the double-primed equator

$$\cos \phi' \sin \theta' = \sin \theta''. \tag{161}$$

Again, substituting the values for the jth and j+1th waypoints and the interpolated point from equations 142 to 147, and solving for the longitude in the double-primed coordinate system, we find that

$$\theta''_j = \arcsin [\cos \phi_j \sin (\theta_j - \theta_0)] . \tag{162}$$

$$\theta''_{j+1} = \arcsin [\cos \phi_{j+1} \sin (\theta_{j+1} - \theta_0)] . \tag{163}$$

$$\theta''_i = \arcsin [\cos \phi_i \sin (\theta_i - \theta_0)] . \tag{164}$$

Note that equations 162–164 use the inverse sine, or arc sin, function to obtain the longitude in the double-primed coordinate system θ'', while equations 154–159 use the inverse cosine, or arc cos, function to get θ''. At first glance, these two functions appear redundant. But the inverse sine and cosine functions deliver not one but two answers in the range –180° to +180°. This is illustrated in figure 45 (*28*). As an example, dots are placed on the sine and cosine functions equal to 0.5, which correspond to an angles of 30° and 150° for sine, and 60° and –60° for cosine.

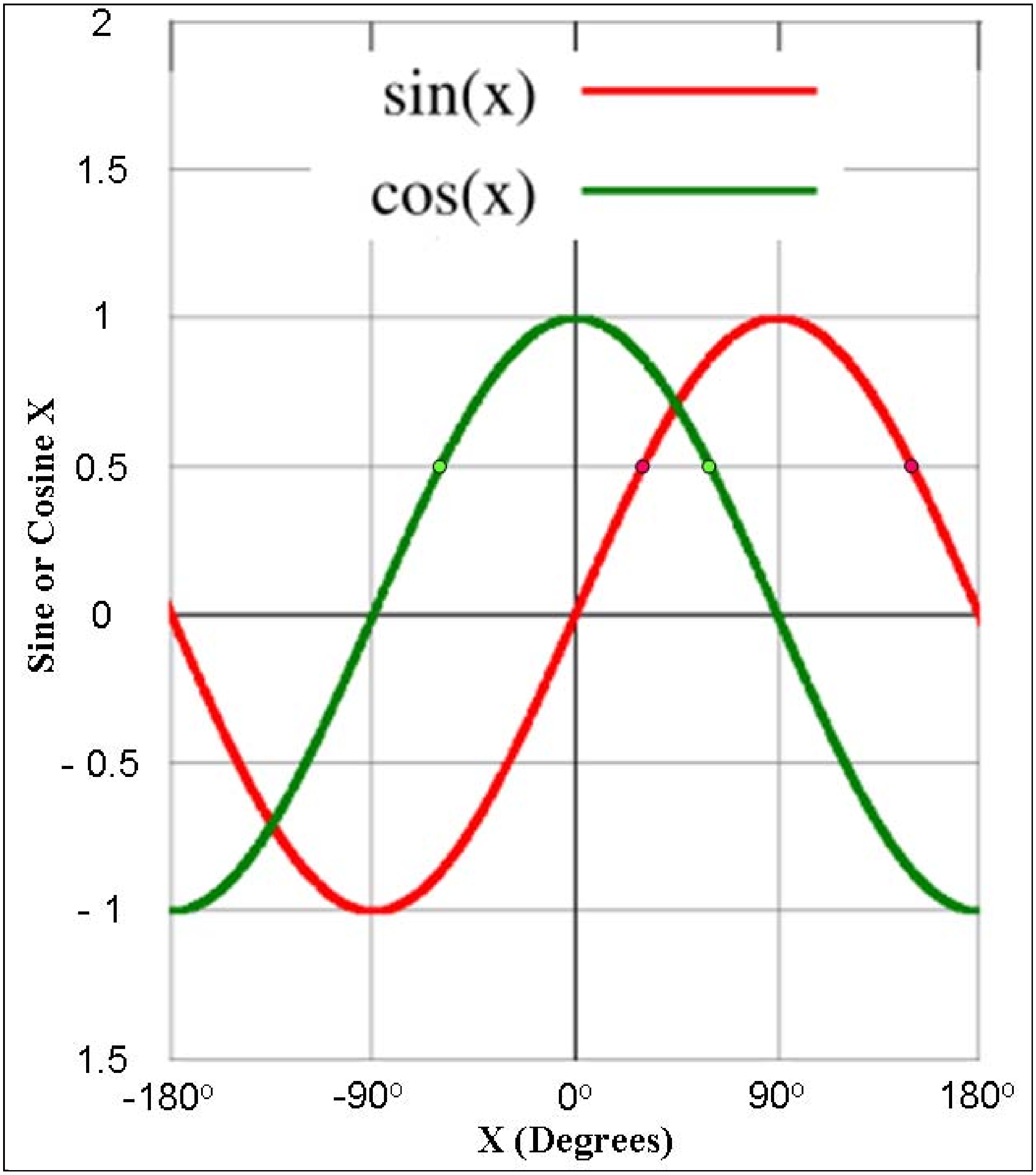


Figure 45. The dots on the sine and cosine graph at 0.5 show that there are two distinct angles for any one value of sine and cosine in the interval –180° to +180°.

To sort out which of the two angles that each of the inverse functions produce is the correct angle, we must first examine the sign of the functions in each of the four quadrants. This is shown in figure 46. In the case where the angle θ is between 0° and 90° (quadrant I), then cos θ and sin θ are both positive. When θ is between 90° and 180° (quadrant II), then sin θ is positive again but cos θ is negative. If θ is between 0° and –90° (quadrant IV), then cos θ is positive but sin θ is negative. Finally, if θ is between –90° and –180° (quadrant III), then both cos θ and sin θ are negative.

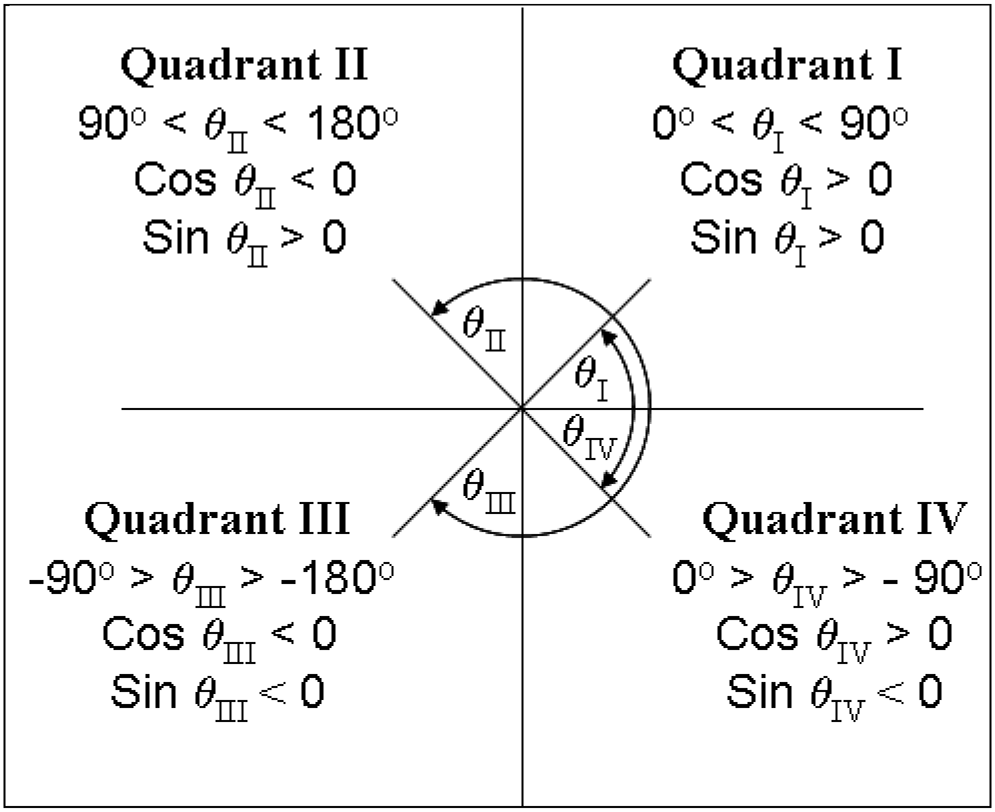

Quadrant II
$90° < \theta_{II} < 180°$
$\text{Cos } \theta_{II} < 0$
$\text{Sin } \theta_{II} > 0$
Quadrant I
$0° < \theta_{I} < 90°$
$\text{Cos } \theta_{I} > 0$
$\text{Sin } \theta_{I} > 0$
θ_{II}
θ_{I}
θ_{IV}
θ_{III}
Quadrant III
$-90° > \theta_{III} > -180°$
$\text{Cos } \theta_{III} < 0$
$\text{Sin } \theta_{III} < 0$
Quadrant IV
$0° > \theta_{IV} > -90°$
$\text{Cos } \theta_{IV} > 0$
$\text{Sin } \theta_{IV} < 0$

Figure 46. For an angle θ ranging from –180° to +180°, the sine and cosine functions take on either a positive or negative sign in each 90° quadrant.

Next, we must consider which values of θ are returned by programming languages for the functions arc cos and arc sin. As is the case for Java (*29*), C++ (*30*, *31*), and FORTRAN (the Formula Translator) (*32*), θ = arc cos x returns a value for θ that is in the first and second quadrants, that is 0 > θ > 180°, while θ = arc sin x returns a value for θ that is in the first and fourth quadrants, that is –90° < θ < +90°. Yet θ could be in any of the four quadrants of figure 46. We see that for all angles of θ for which sin θ is positive (those in quadrants I and II), the arc cos function returns the value of θ correctly. For the other two quadrants (III and IV), we see that the sine function is negative, but the arc cos function returns a value of θ that is opposite in sign to the values nominally returned using programming languages. Therefore, to make sure the angle θ returned is the correct case, we leave the sign for θ as is for the function arc cos x if the sin function is positive and change the sign for θ = arc cos X if the sin function is negative. Applying equations 154–159 and 162–164, we find that

$$\text{if } \cos \phi_j \sin (\theta_j - \theta_0) \geq 0 \,,$$

$$\theta_j'' = \arc \cos (\sin \phi_j / \sin \phi_{max}) \,,$$

and

$$\text{if } \cos \phi_j \sin (\theta_j - \theta_0) < 0 \,,$$

$$\theta_j'' = -\arc \cos (\sin \phi_j / \sin \phi_{max}) \,. \tag{165}$$

Likewise,

$$\text{if } \cos \phi_{j+1} \sin (\theta_{j+1} - \theta_0) \geq 0$$

$$\theta_{j+1}'' = \arc \cos (\sin \phi_{j+1} / \sin \phi_{max}) \,,$$

and

$$\text{if } \cos \phi_{j+1} \sin (\theta_{j+1} - \theta_0) < 0$$

$$\theta_{j+1}'' = -\arc \cos (\sin \phi_{j+1} / \sin \phi_{max}) \,. \tag{166}$$

And finally,

$$\text{if } \cos \phi_i \sin (\theta_i - \theta_0) \geq 0$$

$$\theta_i'' = \arc \cos (\sin \phi_i / \sin \phi_{max}) \,,$$

and

$$\text{if } \cos \phi_i \sin (\theta_i - \theta_0) < 0$$

$$\theta_i'' = -\arc \cos (\sin \phi_i / \sin \phi_{max}) \,. \tag{167}$$

We now apply equations 138 and 139 to find the interpolated angle θ_i''. They are repeated here as equations 168 and 169.

$$\Delta\theta_i = D_i / (R_{ave} + E_j) \,. \tag{168}$$

$$\Delta\theta_i = [(1 - k^2)^{1/2} / k] \text{ Ln } [(R_{ave} + E_i) / (R_{ave} + E_j)] \,. \tag{169}$$

Equation 168 is used if the elevations at the waypoints E_j and E_{j+1} are equal. If they are not, then equation 169 is used, where the value for k is defined by equation 136, repeated here as equation 170:

$$k = (E_{j+1} - E_j) / D_j \,. \tag{170}$$

Notice that it is D_j in equation 170, the distance between the waypoints, not to be confused with the interpolated distance D_i, the distance from the jth waypoint to the platform at the interpolated time t_i. One more item needed is the angular distance between the waypoints. That is, the angle between two lines: one that goes from waypoint j to the Earth's center, and the other that goes between the j+1th waypoint to the Earth's center. This angle, $\Delta\theta$, was calculated in equation 80 and is repeated here as equation 171.

$$\Delta\theta = 2 \text{ arc sin } \{ [(1 / 2) [(\cos \phi_{j+1} \cos \theta_{j+1} - \cos\phi_j \cos \theta_j)^2 + (\cos \phi_{j+1} \sin \theta_{j+1} - \cos \phi_j \sin \theta_j)^2 + (\sin \phi_{j+1} - \sin \phi_j)^2]^{1/2}\} . \quad (171)$$

(ϕ_j,θ_j) and $(\phi_{j+1},\theta_{j+1})$ are the longitude and latitude of the jth and j+1th waypoints in the original coordinate system. The interpolated angle in the double-primed coordinate system θ'' is the fraction of the angular distance $\Delta\theta$ between the waypoints that the platform reached at the interpolated time t_i. Expressed quantitatively using the terms derived so far, it is the fraction $\Delta\theta_i / \Delta\theta$. The radial distance covered between the waypoints by the platform is the fraction of the distance covered times the difference between the coordinates in the double-primed coordinate system, or $(\theta_{j+1}''- \theta_j'') (\Delta\theta_i / \Delta\theta)$. To obtain the double-primed interpolated longitude, we need only add this expression to the longitude of the jth waypoint.

$$\theta_i'' = \theta_j'' + (\theta_{j+1}''- \theta_j'') (\Delta\theta_i / \Delta\theta) . \quad (172)$$

Note that the quantities $\Delta\theta_i$ and $\Delta\theta$ are always positive. But depending on the direction of travel, θ_j'' may be greater than or less than θ_{j+1}''. To make sure that θ_i'' is between θ_j'' and θ_{j+1}'', it is necessary to use the full expression in equation 172 rather than simply add $\Delta\theta_i$ to θ_j''. As a final step, we make sure that the interpolated longitude in the double-primed coordinate system θ_i'' is between –180° and +180°. If it is not, then 360° is either added or subtracted to the value of θ_i'' until it is.

Now that we have the interpolated point $(0,\theta_i'')$ in the double-primed coordinate system (remember, the point is on the double-primed equator), we reverse the two-step process to obtain the coordinates (ϕ_i,θ_i) in the original coordinate system. We shall review equations 158, 164, and 155. Solving equation 158 for ϕ_i, we see that

$$\phi_i = \text{arc sin } (\cos \theta_i'' \sin \phi_{max}) . \quad (173)$$

The range of ϕ_i is between –90° and +90°, and we see that equation 173 delivers a unique value for ϕ_i. Now that we know the value of ϕ_i, we may solve for θ_i. Solving equations 164 and 155 for θ_I, we find

$$\theta_i = \text{arc sin } (\sin \theta_i'' / \cos \phi_i) + \theta_0 \quad (174)$$

and

$$\theta_i = \text{arc cos } (\cos \theta_i'' \cos \phi_{max} / \cos \phi_i) + \theta_0 . \quad (175)$$

Once again, we find that θ_i ranges from –180° to + 180°. But the range of values offered by computer languages (*29–32*) returns values between –90° and +90° for arc sin and 0° and 180° for arc cos. Reusing the process when we transformed to the double-primed coordinate system, we arrive at the value of θ_i:

$$\text{if } \sin \theta_i'' / \cos \phi_i \geq 0, \text{ then}$$

$$\theta_i = \arccos(\cos \theta_i'' \cos \phi_{max} / \cos \phi_i) + \theta_0 ,$$

$$\text{and if } \sin \theta_i'' / \cos \phi_i < 0, \text{ then}$$

$$\theta_i = -\arccos(\cos \theta_i'' \cos \phi_{max} / \cos \phi_i) + \theta_0 . \tag{176}$$

Keep in mind that θ_i must be between –180° and +180°. If it is not, then add or subtract 360° as is appropriate. We have finished describing the process of obtaining interpolated points along a great circle on a sphere.

4.3.5 Summary of Inputting Waypoint Data and Obtaining Interpolated Points

This section briefly describes the process of inputting waypoint data and obtaining the location of the interpolated points along a great circle.

Step 1: Assign the waypoint longitudes θ_j, latitudes ϕ_j, and elevations E_j. If the waypoints are for a ground platform, the elevations are read from a database.

Step 2: Calculate the great circle equation that connects consecutive waypoints. Most of the time, it will be possible to use equation 10 to get θ_0 and equation 11 to get ϕ_{max}, which are the descriptive parameters of equation 6, the great circle equation.

Step 3: Calculate the direct distance Δ with equation 79, the angular distance $\Delta\theta$ with equation 80, and the constant k with equation 95 for consecutive waypoints. This makes it possible to calculate the distance D_j between consecutive waypoints with equation 98.

Step 4: Obtain the cruising velocity v_{cru} and acceleration constant a_0 for the platform from a database. Then, assign either the arrival time $t_{a,j+1}$ to each waypoint to calculate the waypoint velocity v_{j+1} (which can usually be done with equation 76) or assign the waypoint velocity and calculate the arrival time $t_{a,j+1}$ (which can usually be done with equation 75).

Step 5: Calculate the inflection times t_1 and t_2, which can usually be found using equations 72 and 73.

Step 6: Select the loiter time $(t_{d,j} - t_{a,j})$ for each waypoint. Notice that this can only be done if the waypoint velocity v_j is zero. If the waypoint velocity v_j is not zero, the loiter time defaults to zero.

Note that steps 1–6 are executed every time a waypoint is added to a platform path. After all the waypoints for a platform have been entered, proceed with step 7–10 and calculate the interpolated latitudes and longitudes (ϕ_i,θ_i).

Step 7: Select the interpolation times, (each one delineated t_i) for the platform. Use these times to calculate the interpolated distances D_i with equations 133–135. Obtain the interpolated elevation E_i with equation 137 and the interpolated angle $\Delta\theta_i$ with equation 139.

Step 8: Transform the waypoints with departure times $t_{d,j}$ and arrival times $t_{a,j+1}$, which bound the interpolated time t_i to a new, double-primed coordinate system with longitudes θ_j" and θ_{j+1}" using equations 165 and 166. The points will be on the double-primed equator.

Step 9: Calculate the interpolated longitude θ_i" using equation 172.

Step 10: Transform the interpolated longitude θ_i" (it too is on the double-primed equator) back to the original coordinate system with the longitude and latitude (ϕ_i,θ_i) using equations 173 and 176.

5. Conclusions

The algorithms and formulae describing the motion of communications platforms needed for the NCAM deployment module have been successfully derived. For purposes of verification, the details of the algorithm and formulae development have been documented and are shown to be consistent with the assumptions implicit in the model's description of platform motion. The assumptions were that the platform travels along great circle paths between waypoints, the platforms accelerate and decelerate at a constant rate between waypoints, the platforms have a maximum cruising speed consistent with the platform and terrain type (HMMWV vs. tank vs. UAV vs. dismounted Soldier on paved road vs. rough mountainous terrain vs. flat desert terrain, etc.), and the platforms change elevation at a constant rate.

This approach allowed the user to outline each platform's path with waypoints consistent with the user's specification of arrival and departure times at each waypoint. Using the equations derived provided the user with upper and lower limits for inputted data. This gave the user maximum freedom to select input and ensure that the inputs were restricted so as to be self-consistent. This made it possible to calculate the position of platforms between waypoints at predetermined snapshot times. Hence, it will become possible to model link viability between moving platforms once the other modules are completed. Therefore, the equations and algorithms presented here are mandatory to allowing NCAM to model communications links between moving nodes.

Making the motion model more complex would have made it possible to model platform motion more realistically. However, the derivations of the equations and algorithms would have to be far more involved and complex than the ones shown here. Furthermore, it would have demanded that more information be input from the user. Making the motion simpler would have diminished the realism of the platform motion and diminished the control the user has over platform motion. The approach used here (specifying waypoint location and the arrival/departure velocity or time) provided the best compromise between user control, equation and algorithm simplicity, and realistic platform motion.

6. References

1. Still, G. W.; Nealon, J. F. *The Case for Using the Spherical Model to Calculate the Interpolated Points in the Connectivity Software Deployment Module*; ARL-TR-4373; U.S. Army Research Laboratory: Aberdeen Proving Ground, MD, February 2008.

2. Eppink, D.; Keubler, W.; *TIREM/SEM Handbook*; ECAC-HDBK-93-076; Department of Defense Electromagnetic Compatibility Analysis Center: Annapolis, MD, March 1994; pp 1–134.

3. Wikipedia. http://en.wikipedia.org/wiki/Great_circle (accessed 25 October 2007), search: Great Circle.

4. Math Is Fun. http://www.mathisfun.com/equation_of_line.html (accessed 21 September 2007), search: Equation of a Straight Line.

5. S.O.S. Mathematics. http://www.sosmath.com/trig/Trig5/trig5/trig5.html (accessed 17 October 2007), search: Trig Homework Solver.

6. Wikipedia. http://en.wikipedia.org/wiki/Trigonometric_function (accessed 4 April 2008), search: Trigonometric Functions.

7. Cress, P.; Dirksen, P.; Graham, J. W. *Fortran IV With Watfor and Watfiv*; Prentice-Hall Inc.: Englewood Cliffs, NJ, 1970; p 407.

8. Hofstra People. http://people.hofstra.edu/Stefan_Waner/trig/trig2.html (accessed 9 April 2008), search: The Trigonometric Functions.

9. RCN Corporation Home Page. http://users.rcn.com/mwhitney.massed/trigresources/trig-graph-properties.html (accessed 16 January 2008), search: Properties of the Basic Trigonometric Functions.

10. Analyze Math. http://www.analyzemath.com/trigonometry/properties.html (accessed 16 January 2008), search: Properties of Trigonometric Functions.

11. Wikipedia. http://en.wikipedia.org/wiki/Oblate_spheroid (accessed 7 February 2008), search: Oblate Spheroid.

12. Wikipedia. http://en.wikipedia.org/wiki/Ellipsoid (accessed 7 February 2008), search: Ellipsoid.

13. Wikipedia. http://en.wikipedia.org/wiki/WGS84 (accessed 7 February 2008), search: World Geodetic System.

14. Ed Williams' Aviation Home Page. http://williams.best.vwh.net/ellipsoid/node1.html (accessed 10 December 2007), search: Spheroid Geometry.

15. Analyze Math. http://www.analyzemath.com/EllipseEq/EllipseEq.html (accessed 10 December 2007), search: Equation of Ellipse.

16. National Geodetic Survey Home Page. http://www.ngs.noaa.gov/PC_PROD/Inv_Fwd/ (accessed 29 June 2007), search: PC Software Download-INVERSE and FORWARD.

17. Everything 2. http://everything2.com/index.pl?node_id=1056913 (accessed 20 February 2008), search: Differential Solid Angle.

18. Math World. http://mathworld.wolfram.com/OblateSpheroid.html (accessed 20 February 2008), search: Oblate Spheroid.

19. Grobner, W.; Hofreiter, N. *Integraltafel erster teil Unbestimmte Integrale*, No. 5a; Springer-Verlang: New York, 1975; p 45.

20. Dwight, H. W. *Tables of Integrals and Other Mathematical Data*, 4th ed., No. 601.2; MacMillan Publishing Company, Inc.: New York, 1961; p 137.

21. Weast, R., Astile, M., Eds.; *CRC Handbook of Chemistry and Physics*, 63rd ed.; CRC Press Inc.: Boca Raton, FL, 1983; p F-155.

22. Mission-Planning. http://www.mission-planning.com/DTED_Part2.htm (accessed 28 June 2007), search: Pablo's Mission Planning Website.

23. Wolfram Demonstrations Project Home Page. http://demonstrations.wolfram.com/ShortestPathBetweenTwoPointsOnASphere/ (accessed 27 February 2008), search: Shortest Path Between Two Points on a Sphere.

24. University of California Santa Cruz Mathematics Home Page. http://math.ucsc.edu/~bcauchon/math22_eqns.pdf (accessed 21 March 2008), search: Math 22 Equations.

25. Leithold, L. *The Calculus With Analytic Geometry*, 3rd ed.; Harper & Row Publishers: New York, NY, 1976; p 784.

26. Wikipedia. http://en.wikipedia.org/wiki/Natural_logarithm (accessed 9 April 2008), search: Natural Logarithm.

27. The Math Page. http://www.themathpage.com/aTrig/arc-length.htm (accessed 10 April 2008), search: The Math Page Trigonometry

28. Wikipedia. http://en.wikipedia.org/wiki/Image:Sine_cosine_plot.svg (accessed 18 September 2008), search: Sine Cosine Plot.sfg.

29. Sun Spot World. http://www.sunspotworld.com/docs/Purple/javadoc/com/sun/squawk/util/MathUtils.html (accessed 18 September 2008), search: Class Math Utils.

30. Salt Lake City School District Home Page. http://www.slc.k12.ut.us/staff/perickson /programming/tutorials/acos.html (accessed 18 September 2008), search: acos.

31. Salt Lake City School District Home Page. http://www.slc.k12.ut.us/staff/perickson /programming/tutorials/asin.html (accessed 18 September 2008), search: asin.

32. Cress, P.; Dirksen, P.; Graham, J. W. *Fortran IV With Watfor and Watfiv*; Prentice Hall, Inc.: Englewood Cliffs, NJ, 1970; p 407.

INTENTIONALLY LEFT BLANK.

List of Symbols, Abbreviations, and Acronyms

CCI	connectivity confidence interval
DTED	Digital Terrain Elevation Data
FORTRAN	Formula Translator
HMMWV	high-mobility multipurpose wheeled vehicle
NCAM	Network Connectivity Analysis Model
S/N	signal to noise
TIREM	Terrain Integrated Rough Earth Model
UAV	unmanned aerial vehicle
WGS84	World Geodetic System for 1984

NO. OF COPIES	ORGANIZATION
1 (PDF only)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA 8725 JOHN J KINGMAN RD STE 0944 FORT BELVOIR VA 22060-6218
1	DIRECTOR US ARMY RESEARCH LAB IMNE ALC HRR 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB RDRL CIM L 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB RDRL CIM P 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB RDRL D 2800 POWDER MILL RD ADELPHI MD 20783-1197
	ABERDEEN PROVING GROUND
1	DIR USARL RDRL CIM G (BLDG 4600)

NO. OF COPIES	ORGANIZATION
1 (CD only)	ASST SECY ARMY ACQSTN LOGISTICS & TECH SAAL ZP RM 2E661 103 ARMY PENTAGON WASHINGTON DC 20310-0103
1 (CD only)	ASST SECY ARMY ACQSTN LOGISTICS & TECH SAAL ZS RM 3E448 103 ARMY PENTAGON WASHINGTON DC 20310-0103
1 (CD only)	DIRECTOR FORCE DEV DAPR FDZ RM 3A522 460 ARMY PENTAGON WASHINGTON DC 20310-0460
1	US ARMY TRADOC ANL CTR ATRC W A KEINTZ WSMR NM 88002-5502
1	USARL RDRL SLE R FLORES WSMR NM 88002-5513
	ABERDEEN PROVING GROUND
1	US ARMY DEV TEST COM CSTE DTC TT T 314 LONGS CORNER RD APG MD 21005-5055
1	US ARMY EVALUATION CTR CSTE AEC SVE R LAUGHMAN 4120 SUSQUEHANNA AVE APG MD 21005-3013
12	DIR USARL RDRL SL J BEILFUSS J FEENEY J FRANZ M STARKS P TANENBAUM RDRL SLB A G MANNIX RDRL SLB A D FARENWALD RDRL SLB D R GROTE RDRL SLB E M PERRY RDRL SLB G P MERGLER RDRL SLB S S SNEAD RDRL SLB W L ROACH

NO. OF
COPIES ORGANIZATION

1 US ARMY RSRCH DEV & ENGRG
CMD
SYS OF INTEGRATION
AMSRD SS T
6000 6TH ST STE 100
FORT BELVOIR VA 22060-5608

1 US ARMY RSCH LAB
RDRL SLE G
L ANDERSON
WSMR NM 88002-5502

1 US ARMY RSCH LAB
RDRL SLE M
P SIMPSON
WSMR NM 88002-5502

5 US ARMY RSCH LAB
RDRL SLE W
P BOTHNER
FORT MONMOUTH NJ 07703-5602

1 US ARMY RDECOM TARDEC
S CAMPBELL
MS 263
6501 E 11 MILE RD
WARREN MI 48397-5000

ABERDEEN PROVING GROUND

6 DIR USARL
RDRL SL
D BAYLOR
RDRL SLE M
G W STILL (5 CPS)