Comparing a Ballistic Trajectory to Terrain using Digital Elevation Data

by

Jeremy R. Henry

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degrees of

Bachelor of Science in Electrical Engineering and Computer Science

and

Master of Science in Electrical Engineering and Computer Science

at the
Massachusetts Institute of Technology
February 1995

Signature of Author _____

Certified by _____
Prof. Tomas Lozano-Pérez
Massachusetts Institute of Technology
Thesis Advisor

Accepted by _____
Prof. F. R. Morgenthaler
Massachusetts Institute of Technology
Chairman, Committee on Graduate Students

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

The author wishes to thank Professor Tomas Lozano-Pérez of the Massachusetts Institute of Technology and Mr. Dan Hogan for advising this project.

For technical guidance the author wishes to thank Mr. Bennie Burnsed, Mrs. Anne Helgerman, Mr. Matthew Crane, Mr. Peter Samsel, Mr. Eric Helgerman, Mr. Wayne Marzotto, and Mr. Cotton Seed.

For their assistance and resources the author wishes to thank Mr. Jason Henry, Miss Rima AiYun Woo, and Mr. Jin S. Choi.

For their love and support the author wishes to thank Mrs. Lulu S. Henry, Mr. and Mrs. John W. Henry, and Miss Jody Henry.

Comparing a Ballistic Trajectory to Terrain using Digital Elevation Data

by

Jeremy R. Henry

Submitted to the Department of Electrical Engineering and Computer Science
in Feb. 1995 in partial fulfillment of the Requirements for the Degrees of
Bachelor or Science in Electrical Engineering and Computer Science and
Master of Science in Electrical Engineering and Computer Science

ABSTRACT

A system is needed which will compare a terrestrial launch ballistic trajectory to a digital map of the underlying terrain in order to determine the points at which the trajectory intersects the terrain. Three candidate comparison algorithms are presented. First, a brute-force algorithm which compares the elevation of the trajectory at one meter intervals to an interpolated terrain elevation. Second, an algorithm which only checks the trajectory elevation at points at which it passes between two existing terrain elevation sample points. Third, an algorithm which divides the problem space using an octree data structure. The trajectory is checked against every data structure volume through which it passes.

It is desired to determine which of the candidate algorithms displays the optimal performance using criteria such as computation time, memory usage, and accuracy. In order to determine the optimal approach, each of the candidate algorithms is implemented as a computer program. The implementations are then analyzed and compared using a series of test cases. The results of the comparison are described.

The optimal system is shown to be an approach using an octree data structure. This system displays satisfactory space and time efficiency while achieving above-average accuracy. The approach employs a divide and conquer strategy, and concentrates its resources on checking the most interesting portions of the trajectory.

Thesis Supervisor: Prof. Tomas Lozano-Pérez

1. INTRODUCTION

A system is needed which is able to compare predicted ballistic trajectories to the earth's surface in order to determine if and where the trajectory intersects the terrain. The ballistic trajectories are simple terrestrial launch parabolic paths which are represented as a series of points in space. Previous methods of checking trajectory involved painstaking and often inaccurate manual examination of available contour maps. With the introduction and widespread availability of digital terrain elevation databases, it becomes possible to automate the process, increasing both accuracy and efficiency. However, several different approaches to performing the actual comparison are possible, and little is known of their relative efficiency or accuracy.

Efficiency, measured in terms of time and memory consumed by the comparison algorithm, can only be considered in the context of a specific type of processing environment. For the purpose of this project, a single sequential processor with a limited amount of random access memory is used.

The accuracy of a comparison algorithm is much more difficult to analyze then the efficiency. One obvious method of testing accuracy is to perform field tests and observations. However this method is outside the scope of this project. Furthermore, this method is made infeasible by the desire to determine not only where a trajectory path enters the terrain, but also the point at which it exits the terrain obstruction after passing through it. So instead of field testing, an artificial terrain is constructed based on the data in the existing terrain elevation database. This high resolution, artificially created terrain is then used along with an exacting, brute-force algorithm to determine the exact location of terrain intersections down to the necessary level of accuracy. The intersection locations resulting from this approach are then used as the baseline against which the accuracy of all other comparison methods are evaluated.

In an attempt to find a best possible comparison method, three possible algorithms were identified, each of which has the capability to perform the desired comparison. In selecting the candidates, it was desired to exercise a variety of possible techniques. The description of an approach was not limited to the comparison itself, but also included any processing which may be performed on the trajectory or the terrain elevation data. Divide-and-conquer as opposed to brute-force iterative methods were also explored.

The overall goal of this project was then to implement the candidate comparison systems on a representative computer architecture, and then to analyze and compare the performance of the different approaches over a series of test cases. The objective being to find the model which offers the best performance in terms of processing time, memory usage, and accuracy.

2. TECHNICAL BACKGROUND

2.1. Previous Research

Numerous articles have been published on the use of digital terrain elevation data for a wide variety of applications. These include watershed determination [LEE92], planetary surface rendering (for satellite testing) [YOKOY89], perspective terrain rendering [BARBE91] [MILLE86], and others. However, it appears as though very little if any research has been done on the use of digital terrain data to determine intersection points between the terrain and arbitrary paths defined in space, whether these paths describe the flight of meteors, re-entrant satellites, ground-launched ballistic projectiles, low-flying aircraft, etc. This may be due to the relatively scarce availability of digital terrain elevation databases. The database used in this project, DTED, is a product of the United States Department of Defense, and as yet has only a limited distribution.

It may also be observed that very little research has been done on applications which use elevation data quantitatively rather than qualitatively. This is most likely due to the lack of accuracy inherent in most available digital elevation databases. The DTED product used for this project also has serious accuracy limitations. These limitations were summarily ignored in order to minimize the scope of the project and still obtain meaningful results.

2.2. Coordinate Systems

This project is concerned primarily with the determination of terrestrial locations and elevations. There are numerous coordinate systems with which terrestrial locations can be described, and this project employs three of them; Earth-Centered Cartesian Coordinate System, WGS 84 Geodetic Coordinate System, and a local Trajectory Coordinate System. Many of the terms used in the description of coordinate systems are used in different contexts by different authors. In order to insure clarity, the three coordinate systems listed above as well as conversions among the three are described in detail in this section.

2.2.1. Earth-Centered Cartesian Coordinate System

The Earth-Centered Cartesian Coordinate System is the most basic of the coordinate systems used to describe terrestrial locations. Locations are defined using three coordinates. These coordinates represent distances along the three axes of a regular three-dimensional Cartesian coordinate system. The axes are labeled as U-V-W and distances along them are measured in meters.

The origin of the system is at the center of the reference ellipsoid, where the reference ellipsoid is the geometric object approximating the earth. The center of this reference ellipsoid is very nearly equal to the gravitational center of the earth. The 'U' axis lies where the plane of the Greenwich meridian (zero degrees longitude) intersects the equatorial plane (zero degrees latitude). The 'W' axis lies along the earth's axis of rotation in the direction of the north pole. The 'V' axis lies so that the axes U-V-W form a right-handed coordinate system.

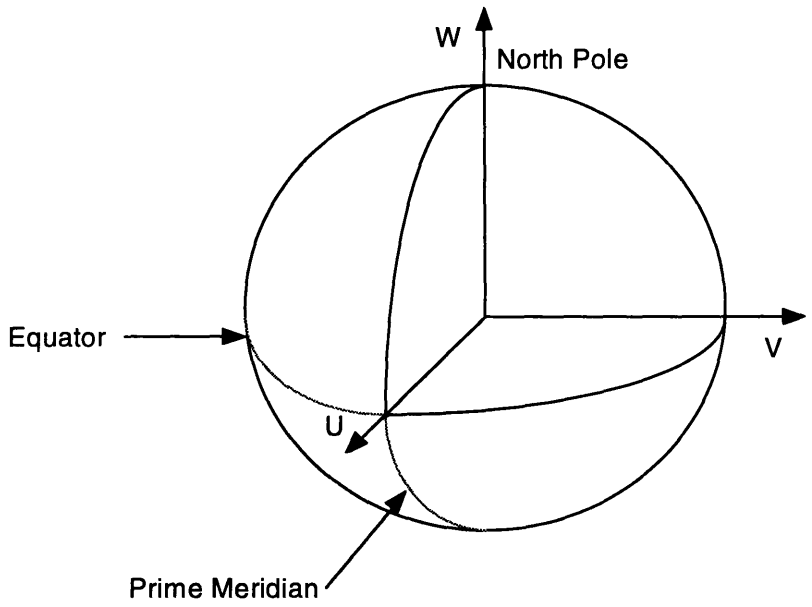


Figure 1. Earth Centered Coordinate System

2.2.2. WGS 84 Geodetic Coordinate System

This is a terrestrial coordinate system based on the WGS 84 ellipsoid. As mentioned before, the reference ellipsoid is a geometric object approximating the earth. The WGS 84 ellipsoid is a specific example of a reference ellipsoid. The WGS 84 Geodetic Coordinate System locates points on or near the surface of the ellipsoid using three coordinates. These coordinates are referred to as latitude (represented as ϕ), longitude (represented as λ), and elevation (represented as h).

The WGS 84 ellipsoid is an ellipsoid of revolution with specific values defined for its semi-major and semi-minor axes. For many applications, a perfect sphere is sufficient as a model of the earth. However, for applications which require greater accuracy, the ellipsoid of revolution is commonly used. The ellipsoid of revolution is the geometric volume obtained by rotating an ellipse around one of its axes (in this case, its minor axis).

The ellipsoid is used for geodesy because it more closely approximates the actual shape of the earth. The earth is not perfectly spherical in shape. It is elliptical, widest at the equator and flattened on the poles due to the apparent centrifugal acceleration caused by the earth's rotation. The ellipsoid of revolution represents the shape which the earth's oceans would conform to if they were free to adjust to the effects of gravity and the earth's rotation, ignoring the tidal effects of the sun and moon and any localized gravitational anomalies in the earth itself.

Because the ellipsoid of revolution can be derived entirely from a single ellipse, the ellipsoid can be completely defined using the same two parameters used to describe the ellipse. There are several choices as to which two parameters to use to describe the ellipsoid. For simplicity, this project will generally use the lengths of the semi-major and semi-minor axes.

semi-major axis: a = 6,378,137.000 meters

semi-minor axis: b = 6,356,752.314 meters

Using these two parameters, several other parameters can be determined which will be useful in different cases. These include:

flattening: $f = \frac{a-b}{a} = 1 / 298.257223563$

semi-major eccentricity squared: $e_a^2 = \frac{a^2 - b^2}{a^2} = 2f - f^2$

semi-minor eccentricity squared: $e_b^2 = \frac{a^2 - b^2}{b^2} = \frac{e_a^2}{1 - e_a^2} = \frac{2f - f^2}{(1-f)^2}$

The WGS 84 Geodetic Coordinate System locates points on this ellipsoid using three coordinates: latitude, longitude, and elevation.

The geodetic latitude is the angle which the normal to the reference ellipsoid at the specified point makes with the plane of the equator. Latitudes are defined between -90 and +90 degrees with the positive latitude lying north of the equator.

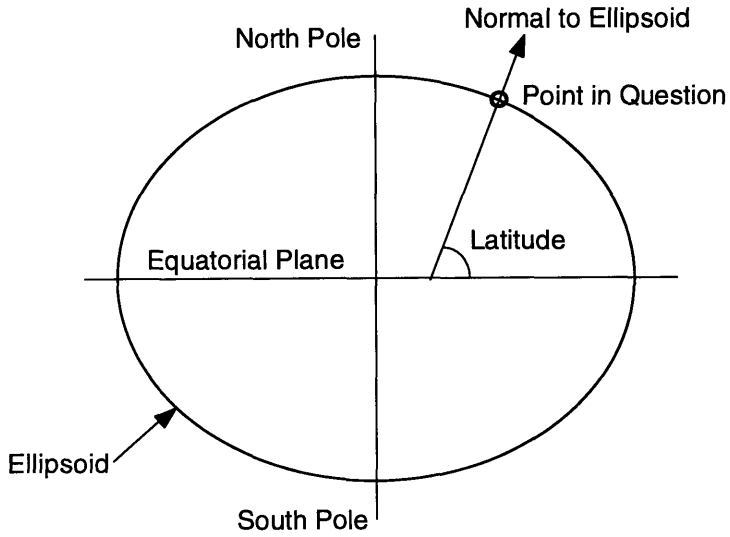


Figure 2. Geodetic Latitude

The geodetic longitude is the angle formed by the projection of the radius vector of the specific point onto the equatorial plane and the vector from the polar axis to the reference (Greenwich) meridian. Longitudes are defined between -180 and +180 degrees with the positive longitudes lying east of the reference meridian.

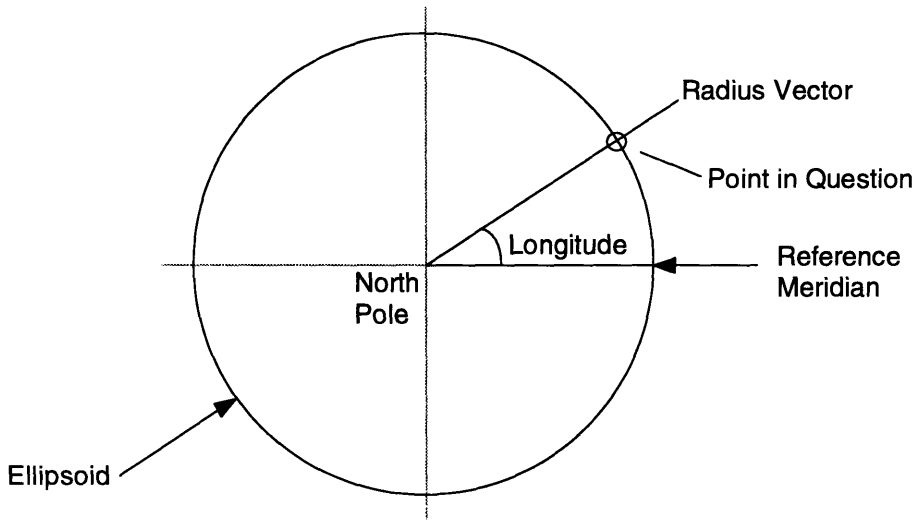


Figure 3. Geodetic Longitude

The geodetic elevation is the height of the specified point above or below the surface of the ellipsoid (mean sea level). This height is measured along a line which is normal to the surface of the ellipsoid at the point in question. Note also that the line normal to the surface of the ellipsoid represents the direction of the combined accelerations due to gravity and the earth's rotation. Thus the definition of height in this manner agrees with our general concept of "up" and "down".

2.2.3. Trajectory Coordinate System

The Trajectory Coordinate System is a three dimensional Cartesian coordinate system. Locations in this coordinate system are specified by three coordinates; Downrange, Vertical, and Crossrange. The Trajectory Coordinate System is oriented such that the plane described by the downrange and crossrange axes is tangent to the reference ellipsoid at the location of the trajectory launch. Thus the origin of the Trajectory Coordinate System is at the longitude and latitude of the launch point with an elevation of h = 0. The downrange axis is oriented in the direction of the terminating point of the trajectory.

Consider an arbitrary trajectory. The Downrange axis lies in a plane which is tangent to the reference ellipsoid at the location of the launch. It is oriented in the direction of the terminating point of the trajectory. The Vertical axis is normal to the ellipsoid at the location of the launch. Its direction is away from the center of the ellipsoid, or "up". The Crossrange axis is oriented so that the Downrange-Vertical-Crossrange axes form a right-handed coordinate system. If the launch point of the trajectory has an elevation of h which is not equal to zero (or sea level), then the trajectory, instead of initiating from (downrange = 0, vertical = 0, crossrange = 0), would instead initiate from the point (downrange = 0, vertical = h, crossrange = 0).

To fully specify a location in the Trajectory Coordinate System requires more than simply the Downrange, Crossrange, and Vertical coordinates. It is also necessary to specify the location and orientation of the Trajectory Coordinate System itself. The location is specified by the coordinate of the origin of the Trajectory Coordinate System in either Earth-Centered Cartesian Coordinates or in WGS 84 Geodetic Coordinates. The orientation is specified by the azimuth of the Downrange axis in radians. The "azimuth" is the angle which the Downrange axis projected onto the reference ellipsoid forms with a line on the reference ellipsoid which originates from the origin of the Trajectory Coordinate System and passes through the north pole. Any angle measured in this way is referred to as an "azimuth".

This Trajectory Coordinate System earns its name from the fact that this is the coordinate system in which the trajectory points are originally determined by the ballistic model.

2.2.4. Conversions Between Coordinate Systems

It is often necessary to convert the representation of a point from one coordinate system to another. These conversions are described in detail below. Unless otherwise stated, these conversions are derived independently or taken from the following sources [MILD] [VANIC86].

2.2.4.1. Converting Earth-Centered Cartesian to WGS 84 Geodetic

A location is specified in the Earth-Centered Cartesian Coordinate system by the coordinates (U, V, W). It is desired to convert this representation to the (ϕ, λ, h) or (latitude, longitude, elevation) coordinates of the WGS 84 Geodetic Coordinate System.

Calculation of the longitude is fairly straight-forward.

$$\lambda = 2 \cdot atan\left(\frac{V}{U+P}\right)$$

Note however that the equation breaks down (a divide by zero is attempted) when U+P = 0. This case corresponds to a point which lies on the international date line (180° E longitude). In this case the negative U value is exactly offset by the positive P value. So when U+P = 0 then $\lambda = \pi$ (180° E longitude).

The computation of the geodetic latitude requires an iterative solution. The following solution comes from Rapp [RAPP84]. Rapp goes on to state that for terrestrial applications the solution converges to 0.1 millimeter of accuracy after only one iteration. As all of the applications of this procedure are terrestrial, one iteration of this algorithm is presented as a closed-form solution.

$$\beta_0 = atan\left(\frac{a \cdot W}{b \cdot P}\right)$$

$$tan\,\phi_0 = \frac{W + e_b^{\,2} \cdot b \cdot sin^3 \beta_0}{P - a \cdot e_a^{\,2} \cdot cos^3 \beta_0}$$

$$\beta_1 = atan\left((1-f) \cdot tan\,\phi\right)$$

$$\phi = atan\left(\frac{W + e_b^{\,2} \cdot b \cdot sin^3 \beta_1}{P - a \cdot e_a^{\,2} \cdot cos^3 \beta_1}\right)$$

Where : a = semi-major axis of WGS 84 ellipsoid
b = semi-minor axis of WGS 84 ellipsoid
$P = \sqrt{U^2 + V^2}$ = distance from the polar axis
e_a^2 = semi-major eccentricity of the WGS 84 ellipsoid
e_b^2 = semi-minor eccentricity of the WGS 84 ellipsoid
f = flattening of the WGS 84 ellipsoid

Note that the equation for β_0 breaks down (a divide by zero is attempted) when P = 0. As P is the distance from the polar axis, this case corresponds to a point which lies on the north or south pole. So if W > 0 then $\phi = \pi/2$ (90° N latitude). If W < 0 then $\phi = -\pi/2$ (90° S latitude).

The calculation of h follows from the calculation of latitude.

$$N = \frac{a}{\sqrt{1 - \left(e_a^2 \cdot sin^2 \phi\right)}}$$

$$h = \frac{P}{cos\,\phi} - N$$

Where : a = semi-major axis of WGS 84 ellipsoid
$P = \sqrt{U^2 + V^2}$ = distance from the polar axis
e_a^2 = semi-major eccentricity of the WGS 84 ellipsoid

Note that the equation for h breaks down when $\phi = \pi$ or $-\pi$. This corresponds to the points on the north or south poles. Thus in polar regions, the following equation is preferred:

$$h = \frac{W}{sin\,\phi} - N + e_a^2 \cdot N$$

2.2.4.2. Converting Earth-Centered Cartesian to Trajectory

A location is specified in the Earth-Centered Cartesian Coordinate system by the coordinates (U, V, W). It is desired to convert this representation to the (dr, v, cr) (or (downrange, vertical, crossrange)) coordinates of the Trajectory Coordinate System. This conversion requires that the orientation and location of the Trajectory Coordinate System be specified. This is accomplished by the location of the trajectory system origin given in geodetic coordinates (ϕ_o, λ_o, h_o), and the azimuth of the downrange axis given in radians (θ_{DR}).

Step 1. Subtract the U, V, W offsets to the location of the trajectory system origin from the (U, V, W) coordinate of the location to obtain the (ΔU, ΔV, ΔW) values from the origin of the Trajectory Coordinate System to the location.

$$N = \frac{a}{\sqrt{1 - e_a^2 \cdot sin^2(\phi_o)}}$$

$$\begin{pmatrix} \Delta U \\ \Delta V \\ \Delta W \end{pmatrix} = \begin{pmatrix} U \\ V \\ W \end{pmatrix} - \begin{pmatrix} (N + h_o) \cdot cos(\phi_o) \cdot cos(\lambda_o) \\ (N + h_o) \cdot cos(\phi_o) \cdot sin(\lambda_o) \\ \left(N \cdot \left(1 - e_a^2\right) + h_o\right) \cdot sin(\phi_o) \end{pmatrix}$$

Step 2. Rotate by λ_o about the W axis to align the U axis to point toward the polar axis.

$$\begin{pmatrix} m \\ n \\ \Delta W \end{pmatrix} = \begin{pmatrix} -cos(\lambda_o) & -sin(\lambda_o) & 0 \\ -sin(\lambda_o) & cos(\lambda_o) & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \Delta U \\ \Delta V \\ \Delta W \end{pmatrix}$$

In this step, the m axis is parallel to the U-V plane and points directly towards the polar axis. The n axis also parallel to the U-V plane and is perpendicular to the m axis such that the m-ΔW-n axes form a right-handed coordinate system.

Step 3. Align the m-n plane with the N-E plane by rotating about the n axis by ϕ_o.

$$\begin{pmatrix} N \\ V \\ E \end{pmatrix} = \begin{pmatrix} sin(\phi_o) & 0 & cos(\phi_o) \\ -cos(\phi_o) & 0 & sin(\phi_o) \\ 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} m \\ n \\ \Delta W \end{pmatrix}$$

Step 4. Rotate the North-Vertical-East coordinate system to coincide with the downrange-vertical-crossrange coordinate system by rotating about the Vertical axis by θ_{DR}.

$$\begin{pmatrix} dr \\ v \\ cr \end{pmatrix} = \begin{pmatrix} cos(\theta_{DR}) & 0 & sin(\theta_{DR}) \\ 0 & 1 & 0 \\ -sin(\theta_{DR}) & 0 & cos(\theta_{DR}) \end{pmatrix} \cdot \begin{pmatrix} N \\ V \\ E \end{pmatrix}$$

Result. The matrices from steps 2-4 can be combined into the following matrix (note that the components of the matrix are given in a shortened representation to aid presentation).

$$\begin{pmatrix} dr \\ v \\ cr \end{pmatrix} = \begin{pmatrix} -cos\lambda \cdot sin\phi \cdot cos\theta - sin\lambda \cdot sin\theta & -sin\lambda \cdot sin\phi \cdot cos\theta + cos\lambda \cdot sin\theta & cos\phi \cdot cos\theta \\ cos\lambda \cdot cos\phi & sin\lambda \cdot cos\phi & sin\phi \\ cos\lambda \cdot sin\phi \cdot sin\theta - sin\lambda \cdot cos\theta & sin\lambda \cdot sin\phi \cdot sin\theta + cos\lambda \cdot cos\theta & -cos\phi \cdot sin\theta \end{pmatrix} \cdot \begin{pmatrix} \Delta U \\ \Delta V \\ \Delta W \end{pmatrix}$$

2.2.4.3. Converting WGS 84 Geodetic to Earth-Centered Cartesian

A location is specified in the WGS 84 Geodetic Coordinate System by the coordinates (ϕ, λ, h) or (latitude, longitude, elevation). It is desired to convert this representation to the (U, V, W) of the Earth-Centered Cartesian Coordinate System.

$$N = \frac{a}{\sqrt{1 - e_a^{\,2} \cdot sin^2(\phi)}}$$

$$U = (N + h) \cdot cos(\phi) \cdot cos(\theta)$$

$$V = (N + h) \cdot cos(\phi) \cdot sin(\theta)$$

$$W = \left(N \cdot \left(1 - e_a^{\,2}\right) + h\right) \cdot sin(\phi)$$

2.2.4.4. Converting WGS 84 Geodetic to Trajectory

A location is specified in the WGS 84 Geodetic Coordinate System by the coordinates (ϕ, λ, h) (or (latitude, longitude, elevation)). It is desired to convert this representation to the (dr, v, cr) (or (downrange, vertical, crossrange)) coordinates of the Trajectory Coordinate System. This conversion requires that the orientation and location of the Trajectory Coordinate System be specified. This is accomplished by the location of the trajectory system origin given in geodetic coordinates (ϕ_o, λ_o, h_o), and the azimuth of the downrange axis given in radians (θ_{DR}).

This conversion is not performed directly. Instead, the Earth-Centered Cartesian Coordinate System as an intermediary. First convert from the WGS 84 Geodetic Coordinate System to the Earth-Centered Cartesian Coordinate System as described in Section 2.2.4.3. Then convert the resulting (U, V, W) coordinates to the Trajectory Coordinate System using the conversion described in Section 2.2.4.2.

2.2.4.5. Converting Trajectory to Earth-Centered Cartesian

A location is specified in the Trajectory Coordinate by the coordinates (dr, v, cr) (or (downrange, vertical, crossrange)). It is desired to convert this representation to the (U, V, W) coordinates of the Earth-Centered Cartesian Coordinate System. The conversion from the Trajectory Coordinate System also requires that the orientation and location of the Trajectory Coordinate System be specified. This is accomplished by the location of the trajectory system origin given in geodetic coordinates (ϕ_o, λ_o, h_o), and the azimuth of the downrange axis given in radians (θ_{DR}).

Step 1. Convert the point to a North-Vertical-East coordinate system by rotating about the Vertical axis by θ_{DR}.

$$\begin{pmatrix} N \\ V \\ E \end{pmatrix} = \begin{pmatrix} cos(\theta_{DR}) & 0 & -sin(\theta_{DR}) \\ 0 & 1 & 0 \\ sin(\theta_{DR}) & 0 & cos(\theta_{DR}) \end{pmatrix} \cdot \begin{pmatrix} dr \\ v \\ cr \end{pmatrix}$$

Step 2. Align the dr-cr plane with the U-V plane by rotating about the E axis by ϕ_o.

$$\begin{pmatrix} m \\ n \\ \Delta W \end{pmatrix} = \begin{pmatrix} sin(\phi_o) & -cos(\phi_o) & 0 \\ 0 & 0 & 1 \\ cos(\phi_o) & sin(\phi_o) & 0 \end{pmatrix} \cdot \begin{pmatrix} N \\ V \\ E \end{pmatrix}$$

In this step, the m axis is parallel to the U-V plane and points directly towards the polar axis. The n axis also parallel to the U-V plane and is perpendicular to the m axis.

Step 3. Rotate by λ_o about the W axis to align the m axis with U.

$$\begin{pmatrix} \Delta U \\ \Delta V \\ \Delta W \end{pmatrix} = \begin{pmatrix} -cos(\lambda_o) & -sin(\lambda_o) & 0 \\ -sin(\lambda_o) & cos(\lambda_o) & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} m \\ n \\ \Delta W \end{pmatrix}$$

Result. The matrices from steps 1-3 can be combined into the following matrix (note that the components of the matrix are given in a shortened representation to aid presentation).

$$\begin{pmatrix} \Delta U \\ \Delta V \\ \Delta W \end{pmatrix} = \begin{pmatrix} -\cos\lambda \cdot \sin\phi \cdot \cos\theta - \sin\lambda \cdot \sin\theta & \cos\lambda \cdot \cos\phi & \cos\lambda \cdot \sin\phi \cdot \sin\theta - \sin\lambda \cdot \cos\theta \\ -\sin\lambda \cdot \sin\phi \cdot \cos\theta + \cos\lambda \cdot \sin\theta & \sin\lambda \cdot \cos\phi & \sin\lambda \cdot \sin\phi \cdot \sin\theta + \cos\lambda \cdot \cos\theta \\ \cos\phi \cdot \cos\theta & \sin\phi & -\cos\phi \cdot \sin\theta \end{pmatrix} \cdot \begin{pmatrix} dr \\ v \\ cr \end{pmatrix}$$

Step 4. Add the U, V, W offsets to the location of the trajectory system origin to obtain the (U, V, W) coordinates.

$$N = \frac{a}{\sqrt{1 - e_a^2 \cdot \sin^2(\phi_o)}}$$

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \begin{pmatrix} \Delta U \\ \Delta V \\ \Delta W \end{pmatrix} + \begin{pmatrix} (N + h_o) \cdot \cos(\phi_o) \cdot \cos(\lambda_o) \\ (N + h_o) \cdot \cos(\phi_o) \cdot \sin(\lambda_o) \\ \left(N \cdot \left(1 - e_a^2\right) + h_o\right) \cdot \sin(\phi_o) \end{pmatrix}$$

2.2.4.6. Converting Trajectory to WGS 84 Geodetic

A location is specified in the Trajectory Coordinate by the coordinates (dr, v, cr) (or (downrange, vertical, crossrange)). It is desired to convert this representation to the (ϕ, λ, h) or (latitude, longitude, elevation) coordinates of the WGS 84 Geodetic Coordinate System. The conversion from the Trajectory Coordinate System also requires that the orientation and location of the Trajectory Coordinate System be specified. This is accomplished by the location of the trajectory system origin given in geodetic coordinates (ϕ_o, λ_o, h_o), and the azimuth of the downrange axis given in radians (θ_{DR}).

There is no way to perform this conversion directly. Instead, it is necessary to use the Earth-Centered Cartesian Coordinate System as an intermediary. First convert from the Trajectory Coordinate System to the Earth-Centered Cartesian Coordinate System as described in Section 2.2.4.5. Then convert the resulting (U, V, W) coordinates to the WGS 84 Geodetic Coordinate System using the conversion described in Section 2.2.4.1.

2.3. Digital Terrain Elevation Data

The digital elevation data which will be used in this project is the Department of Defense standard product, the Digital Terrain Elevation Data (DTED). This data is produced by the Defense Mapping Agency in accordance with Military Specification MIL-D-89001 [MIL89]. The format of DTED is described in detail in the military specification for DTED, but an overview of that format will be presented here.

2.3.1. DTED Organization

DTED is a raster format product in which the elevation of the earth's surface is sampled at discrete points called "posts". At each post, a measurement is taken between the elevation of the terrain and the elevation of the WGS 84 spheroid. This measurement is essentially the height above (or below) sea level of the terrain. These post elevations are then grouped into blocks called "cells". A cell contains a two-dimensional array of post elevations which covers an area one degree of latitude by one degree of longitude. Each cell is labeled by the WGS 84 geodetic coordinate of its south-west corners. The cells are always referenced to integral degree values of latitude and longitude.

The posts are located in a regular grid pattern of rows and columns where rows correspond to lines of latitude and columns correspond to lines of longitude. The rows and columns fall along integral seconds of latitude or longitude. As stated previously, the southwest corner of each cell lies at an integral degree value of latitude and longitude. Therefore the west-most column and the south-most row or DTED posts also falls at an integral degree value of latitude or longitude. The remaining rows and columns of the cell are evenly spaced at an integral number of seconds apart. For example, the rows may be every three seconds of latitude apart and the columns every six seconds of longitude apart.

2.3.2. DTED Variation

Because of the irregular spacing of the lines of latitude and the convergence at the poles of the lines of longitude, the posts are not spaced regularly on the surface of the earth. The variation between lines of latitude is not severe, and therefore DTED ignores this variation. To account for the longitudinal variation, which becomes quite severe as we move from the equator towards the poles, DTED divides both the northern and southern hemispheres into 5 latitudinal zones. The longitudinal post spacing is different in each of the 5 zones. This helps to alleviate the greater part of the variation, but minor variation in longitudinal post spacing still occurs within each zone. These zones are described in Table 1.

Zone Latitude	DTED Level 1		DTED Level 2	
	Latitude Spacing	Longitude Spacing	Latitude Spacing	Longitude Spacing
80°N to 90°N	3 sec. of lat.	18 sec. of long.	1 sec. of lat.	6 sec. of long.
75°N to 80°N	3 sec. of lat.	12 sec. of long.	1 sec. of lat.	4 sec. of long.
70°N to 75°N	3 sec. of lat.	9 sec. of long.	1 sec. of lat.	3 sec. of long.
50°N to 70°N	3 sec. of lat.	6 sec. of long.	1 sec. of lat.	2 sec. of long.
0° to 50°N	3 sec. of lat.	3 sec. of long.	1 sec. of lat.	1 sec. of long.
50°S to 0°	3 sec. of lat.	3 sec. of long.	1 sec. of lat.	1 sec. of long.
70°S to 50°S	3 sec. of lat.	6 sec. of long.	1 sec. of lat.	2 sec. of long.
75°S to 70°S	3 sec. of lat.	9 sec. of long.	1 sec. of lat.	3 sec. of long.
80°S to 75°S	3 sec. of lat.	12 sec. of long.	1 sec. of lat.	4 sec. of long.
90°S to 80°S	3 sec. of lat.	18 sec. of long.	1 sec. of lat.	6 sec. of long.

Table 1. Post Spacing of DTED by Latitudinal Zone

Note that DTED is defined between 90°S and 90°N latitude. In practice, this project will only deal with latitudes between 80°S and 84°N latitude. Although it will not be discussed further here, these boundaries are due to the limitations of the Universal Transverse Mercator (UTM) system which is only defined between 80°S and 84°N latitude.

2.3.3. DTED Accuracy

The DTED is not 100% accurate. As stated above, the DTED is organized into blocks called "cells". Each of these cells measures 1 degree longitude by 1 degree latitude. The accuracy of the DTED is specified separately for each cell. The accuracy specifications consist of four values. These are: absolute horizontal accuracy, absolute vertical accuracy, point to point horizontal accuracy, point to point vertical accuracy.

Absolute accuracy specifications measure the accuracy of the DTED with respect to the true earth terrain elevation. An absolute vertical accuracy value of ***x*** meters indicates that the post described in the DTED is within ***x*** meters of the altitude of the actual terrain with 90% probability. Likewise an absolute horizontal accuracy value of ***x*** meters indicates that a piece of terrain of the given elevation exists within ***x*** meters of the indicated location with 90% probability.

Relative accuracy specifications measure the accuracy of any single DTED measurement with respect to surrounding DTED measurements. A relative vertical accuracy value of ***y*** meters indicates that if the DTED specifies that adjacent posts ***A*** and ***B*** are ***z*** meters apart vertically, then the actual points on the earth which correspond to points ***A*** and ***B*** are actually (***z-y***) to (***z+y***) meters apart with 90% probability.

The accuracy of the DTED affects the accuracy of the comparison system. However, for the purposes of this project, these accuracy values will be ignored. Since it is impossible, barring actual field observation, to determine the nature or biases of the DTED inaccuracies, it will be assumed that the DTED values are correct as listed in the database.

2.3.4. DTED Versions

There are two versions of DTED. These are DTED Level 1 and DTED Level 2. The formats of these two versions are very similar. Their only difference is in the density of the data which they contain. DTED Level 1 is less dense than DTED Level 2. Level 1 is based on a latitude post spacing of 3 arcseconds and longitude post spacing of 3, 6, 9, 12, and 18 arcseconds (the variation occurs across the five latitudinal zones). Level 2 is based on latitude post spacing of 1 arcsecond and longitude post spacing of 1, 2, 3, 4, and 6 arcseconds (again the variation occurs across the five latitudinal zones).

A system which uses DTED must be able to handle both versions of DTED. Furthermore, cells of the two different versions may be interleaved.

Currently, DTED Level 2 has not reached the stage of widespread production and distribution. For this reason, DTED Level 2 will not be available for use on this project. Although it may be possible to simulate DTED Level 2 by appropriate interpolation DTED Level 1 data, it is unknown how closely the result would resemble actual Level 2 data. Therefore, no DTED Level 2, either real or simulated, will be used on this project. However, all algorithms and implementations must still be able to handle interleaved sections of DTED Level 1 and Level 2.

2.4. Ballistic Trajectories

The ballistic trajectories which are to be compared to the DTED represent the path which a spin-stabilized ballistic projectile traces in space as it leaves the ground, arches up to apogee, and then falls again to the earth. In a vacuum this path would be a perfect parabola. However, the introduction of atmosphere and meteorological effects causes the path to diverge from this ideal. There are several algorithms and models which may be used to determine the path of a spin-stabilized projectile in an atmosphere. The model which is used on this project as well as the factors which affect it are described below.

2.4.1. Ballistic Model

The trajectories to be checked will be generated by a ballistic trajectory model described by Mastrey in her 1989 paper "Generation and Evaluation of a Ballistic Algorithm Development Model" [MASTR89]. Mastery's work is based on the trajectory model algorithm described in "A Modified Point Mass Trajectory Model for Rocket-Assisted Artillery" [BRL79]. This ballistic trajectory model is well documented and several implementations of the model have already been produced. Therefore the implementation of the model will not be addressed. However, the model is very general in that it allows many parameters to affect the flight of the projectile. Addressing all possible variations in ballistic trajectories is beyond the scope of this project, and so several of the available parameters to the ballistic model will be ignored or left constant. Each of the factors which affect the trajectory and the manner in which it will be handled in this project is addressed individually below:

2.4.1.1. Projectile

The general ballistic model can simulate an unlimited number of different types of spin-stabilized projectiles. All that is required for the model to handle an arbitrary projectile is a collection of physical and ballistic characteristics of that projectile. This information includes such Information as weight, drag, center-of-gravity, etc.

For this project, a single projectile type will be used. This projectile is the most basic type of projectile in that it is non-assisted, non-base-bleed, and single-staged.

The term "assisted" refers to rocket-assisted projectiles. These projectiles burn a self-contained propellant after initial launch in order to gain or maintain velocity during flight. The projectile which will be used in this project is non-assisted which means that its only source of motive power is from the initial launch. After the launch, the projectile is free-flying.

The term "base-bleed" refers to a class of projectiles which bleed a gas from their base during some stage of their flight in order to decrease drag. The projectile which will be used in this project is non-base-bleed which means that during the entire flight of the projectile, it will be affected by drag.

The term "single-stage" refers to a class of projectiles which act and are acted upon homogeneously throughout all portions of the flight. Other projectiles may change the composition of the projectile in mid-flight, thus altering the ballistic and physical characteristics of the projectile.

2.4.1.2. Meteorological Conditions

The general ballistic model allows the air temperature, air density, wind direction, and wind speed to be defined for 26 different layers of the atmosphere.

For this project, a single set of meteorological conditions will be used. This single set is referred to as the standard MET which has wind velocities of zero at every atmospheric layer.

2.4.1.3. Earth Model

The earth model parameter of the ballistic model determines whether the apparent Coriolis force acting on the projectile in flight will be taken into account. The Coriolis force has only a minor effect on the flight of the projectile, especially on shorter trajectories, and therefore the ballistic model is able to ignore the effect, if so indicated, in order to speed up processing.

For this project the apparent Coriolis force will be taken into account by the ballistic model. Because the Coriolis force is dependent on the direction of travel of the projectile, as well as the latitude of the launch location, this means that trajectories which are similar in launcher elevation or launch velocity are none-the-less not interchangeable. It is therefore not feasible to calculate and store all possible trajectories before-hand.

2.4.1.4. Launch Conditions

As the projectile is non-assisted and there are no winds in the meteorological data, the launch supplies the only motive force to the projectile, and thus the launch conditions completely determine the trajectory which the projectile will follow. The launch conditions consist of the following factors: geodetic latitude of the launch location, height of launch above sea level, azimuth of launch, launcher elevation, and launch velocity.

The geodetic latitude of the launch location is necessary in computing the apparent Coriolis force acting on the projectile.

The height of the launch above sea level is used to determine the distance from the projectile to the center of gravity of the earth. This distance is needed to determine the magnitude of the earth's gravity acting on the projectile.

The azimuth of launch is necessary in that it allows the model to calculate the apparent Coriolis force acting on the projectile. The Coriolis force is a function of (among other things) the direction of travel of the projectile in relation to the earth.

The launch elevation of the launch is measured in radians above (or below) the horizontal. The horizontal is defined to be a plane which is tangent to the earth's spheroid at the location of launch. For the purposes of this project, only launch elevations between -3 and 85 degrees from the horizontal will be considered.

The launch velocity is a single value which indicates the velocity of the projectile in three dimensional space in the direction of travel in meters per second. For the purposes of this project, only launch velocities between 100 and 1500 meters per second will be considered.

2.4.2. Trajectory Representation

The ballistic model produces a set of location coordinates which represent discrete points in the path of the trajectory. An average trajectory contains from 100 to 200 of these points. Extremely short distance trajectories may contain less than 100 points, while longer trajectories, or trajectories which terminate a substantial distance below the height of origin, may contain more than 200 points. The location coordinates are originally calculated in the Trajectory Coordinate System but they are made available in the form of WGS 84 Geodetic Coordinates as well.

The trajectory points are not uniformly spaced in either the downrange direction or in any other dimension. The distance between points is actually a function of the current acceleration of the projectile which is continuously varying over the course of the trajectory.

While it can be assumed that the projectile will pass through each of the location coordinates during its flight, the location of the projectile at any points between two adjacent trajectory points is undefined. However, given that there are no winds in the meteorological dataset and that the Coriolis force has a very minute affect on the path of the projectile, the only major forces acting on the projectile while in flight are gravity and the viscosity of the atmosphere. This makes it possible to predict that the projectile follows a roughly parabolic curve in the space between known points.

2.4.3. Trajectory Characteristics

The model which will be used is deterministic. That is, there are no random variables used in the computation of the trajectory, and therefore the results obtained from the ballistic model are reproducible. Thus it is possible to determine characteristics of trajectories and count on these characteristics holding true for future trajectories as well.

All trajectories calculated using the ballistic model have the same general shape. The trajectory curves downward according to gravity and to the right owing to the spin stabilization of the projectile and the fact that the projectile tips, or pitches over towards the earth as it flies (i.e., upon launch the nose of the projectile is pointing up while upon landing the nose of the projectile is pointing down).

As stated previously, the trajectory points are originally calculated in the Trajectory Coordinate System. One useful aspect of this representation is that the trajectory may be considered to be a function of the downrange dimension. That is, for each downrange value from zero (the launch point) to the impact point (the trajectory exists only between these bounds), there is a unique projectile location in both the crossrange and altitude dimensions.

Another useful aspect of the trajectory coordinate system representation is that the trajectory path may be decomposed into two separate functions; altitude as a function of downrange, and crossrange as a function of downrange. These functions are independent of one another and thus may be addressed separately. Both of these functions tend to resemble parabolic curves.

2.5. Computational Environment Constraints

This project will address the difficulties encountered when the trajectory / terrain comparison is performed in a real-time embedded system. The comparison system will therefore be subjected to a set of timing and memory constraints which may be typical of an embedded processing environment.

The implementation model against which the algorithms described in this project are analyzed consists of a single sequential processor with a limited amount of random access memory. The memory available to this processor is divided into three parts: code memory, data memory, and stack memory. The code memory is limited to 1 megabyte. However, all of the models described in this paper fall well under this limitation. The data memory consists of heap memory which is used to store static data structures as well as data which is too large to be stored on the stack, such as the DTED. The amount of data memory used by each comparison system will be monitored during test case runs. The stack memory is limited to 1 megabyte. The amount of stack memory used by the comparison systems will not be closely monitored like the data memory usage. Instead, the memory initially allocated for stack usage will be limited to 1 megabyte, and the run-time system will indicate if this maximum is exceeded. However, it is projected that 1 megabyte of stack space is quite sufficient for the needs of the comparison systems described in this paper.

The time to perform the trajectory / terrain comparison is not limited in any way, but the time taken required by a comparison system to perform the actual comparison will factor into its evaluation.

The accuracy with which the trajectory / terrain comparison is performed is required to be accurate to within 100 meters in the horizontal direction in order to be considered "correct". The computational environment affects accuracy by dictating the data types which are available. The data types available on the model processor consist of a 2 byte signed integer, a 4 byte signed long integer, and an 8 byte floating point type (15-digit precision, range from 1.7×10^{-308} to 1.7×10^{308}).

3. BASIC SYSTEM DESCRIPTION

The primary objective of this project is to analyze and evaluate three different comparison systems in order to determine which system performs the trajectory / terrain comparison in the least amount of time while still observing memory and accuracy requirements. Each of the comparison systems has access to the same input data and is required to produce the same type of output data. In composition, each system has roughly the same organization and structure. This organization is graphically represented in Figure 4.

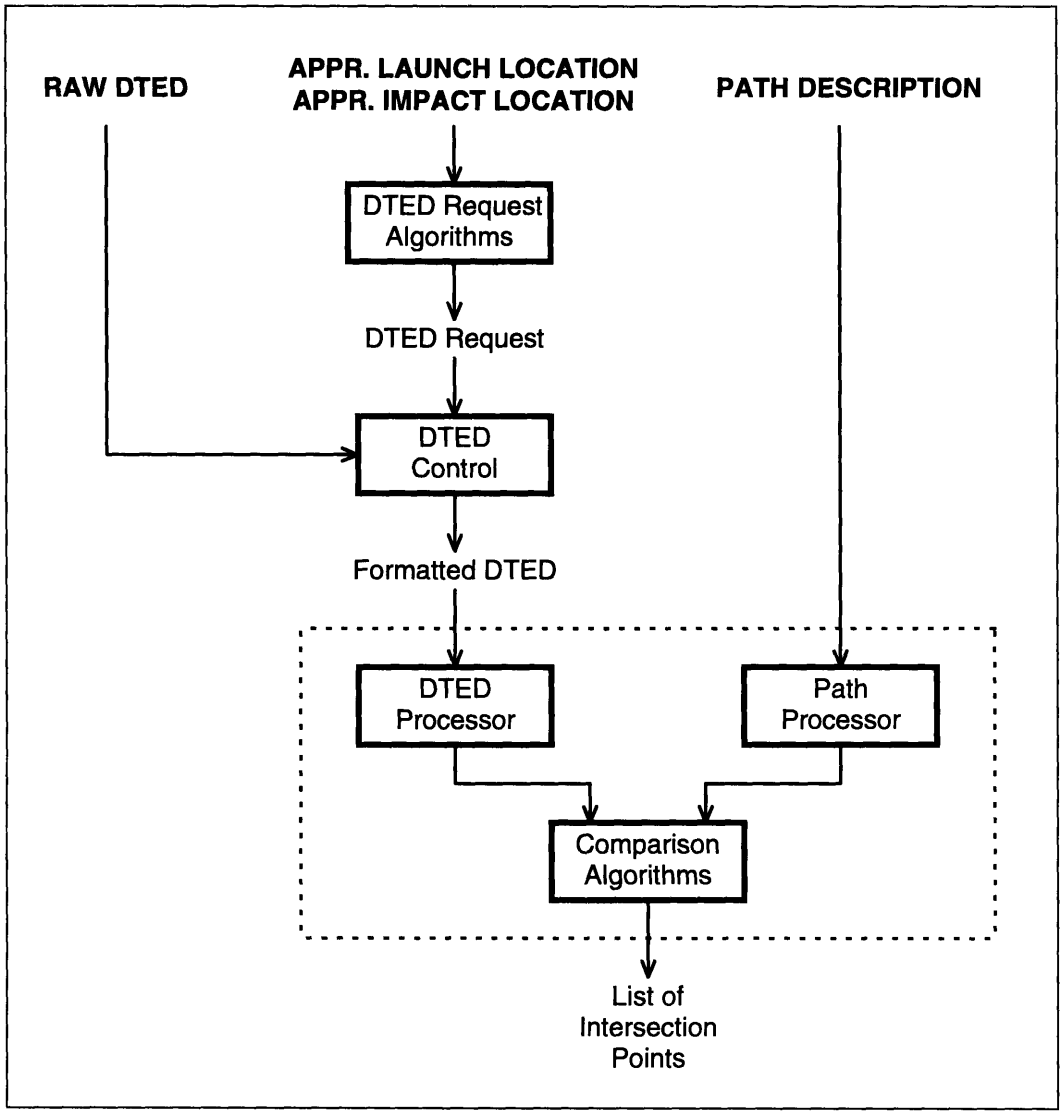


Figure 4. Organization of Basic Comparison System

This diagram shows the inputs to a comparison system, the five blocks of processing which must be performed, and the eventual output of the system. Each of these components is described below.

3.1. System Inputs

As indicated in Figure 4, the basic comparison system has three inputs. These are: raw DTED, the approximate launch and impact locations, and a description of the trajectory path. Each of these inputs is described individually below.

3.1.1. Raw DTED

The raw DTED is one of the basic inputs to a comparison system. It consists of the DTED data files which reside on the original DTED CDs. The format of these data files corresponds to the description found in the military specification on DTED [MIL89]. This format has been discussed previously.

A single DTED data file in its original format contains far more data that could ever be needed to perform a single trajectory comparison. Therefore, certain portions of DTED are selected for use by the comparison system and only these sections are stored and used by the comparison system.

3.1.2. Approximate Launch and Impact Locations

The approximate launch and impact locations are two of the basic inputs to a comparison system. Each approximate location is a geodetic coordinate (without an elevation component) which indicates the actual location of the launch or impact point to within 1 kilometer. In order to put a reasonable limit on the size of the problem space, the distance between the approximate launch and impact points is limited to 50 kilometers.

The approximate launch and impact locations are needed to determine what portions of DTED will be loaded from the raw DTED files into main memory. These values are only approximations of the actual launch and impact points because problem is defined such that the actual locations are not known until after the DTED has been loaded into main memory. This is too late for the actual values to be used in determining the necessary portions of DTED to actually load into main memory.

3.1.3. Path Description

The description of the trajectory path is one of the basic inputs to a comparison system. A path description consists of a set of between 2 and 500 projectile locations. Each location is represented in both the Trajectory Coordinate System as well as the WGS 84 Geodetic Coordinate System. Each location value specifies the position of the projectile at some point of the trajectory.

As has been stated previously, the path which the projectile follows between the points given in the path description is unknown, and can only be approximated.

Due to the nature of the problem, the path description is not made available to the comparison system until after the selected portions of raw DTED have been requested and loaded into main memory (this is what necessitates the availability of the approximate launch and impact locations). The actual launch and impact locations are made available at the same time as the path description, as they simply consist of the first and last points (respectively) of the path description.

3.2. System Components

Each of the comparison systems consists of five subsystems: The DTED Request Algorithms, The DTED Controller, The DTED Processor, The Trajectory Processor, and The Comparison Algorithms. Each of these subsystems is described individually below.

3.2.1. DTED Request Algorithms

The DTED Request Algorithms are responsible for examining the approximate launch and impact locations which are input to the system and determining the specific portions of DTED which are required to check any valid ballistic trajectory (trajectories which may be produced by the ballistic model) which could exist between the two points. The request which is formulated by the DTED Request Algorithms will be passed on to the DTED Controller which will load the requested portions of DTED into main memory. The amount of DTED requested may not exceed the 4 megabyte memory constraints placed upon the amount of data memory. The time required for the DTED Request Algorithms to determine the necessary portions of DTED will be included in the time taken by the entire system to perform the trajectory / terrain comparison.

The DTED Request Algorithms are based on the approximate launch and impact locations rather than the actual trajectory which is to be checked by the comparison system. This is because the problem is defined such that the actual trajectory is not available until after the DTED Controller has completed its task. Since the DTED Request Algorithms supply the input to the DTED Controller, this indicates that the DTED Request Algorithms must finish before the actual trajectory is made available.

The request formulated by the DTED Request Algorithms is represented as a list of block identifiers. A block is the basic unit of DTED which will be further described in Section 3.2.2. A block identifier is the geodetic latitude and longitude of the southwest corner of a DTED block. A block of DTED will be loaded into main memory by the DTED Controller if and only if the identifier of the block is contained in the request formulated by the DTED Request Algorithms.

As each comparison system includes a set of DTED Request Algorithms, it is possible that each comparison system may request different portions of DTED. This allows algorithms with specialized DTED requirements to increase or decrease the amount of DTED which must be loaded into main memory. However, most comparison systems use a very basic version of the DTED Request Algorithms. This version requests every block of DTED in a rectangle which encompasses the approximate launch point, the approximate impact points, and a large area surrounding these points called the "swatch". This "swatch" is guaranteed to contain all possible trajectories between the approximate launch and impact points.

The swatch is a function of the approximate launch and impact locations. The swatch appears as a rectangle whose major axis is oriented in the same direction as the line connecting the approximate launch point to the approximate impact point. The long edge of the swatch is essentially this connecting line extended 1 kilometer on each end (1 kilometer from the approximate launch point extending in the direction away the approximate impact point, and 1 kilometer from the approximate impact point extending in the direction away from the approximate launch point). The short side of the swatch rectangle lies perpendicular to the long edge and extends 3 kilometers in the positive crossrange direction (to the right looking from the approximate launch point to the approximate impact point) and 7 kilometers in the negative crossrange direction (to the left of the launch-impact segment). The swatch is larger to the left of the launch/impact line than to the right because the direction of the projectile's spin causes it to veer to the right during flight. Thus projectiles are launched at an azimuth to the left of the impact point line and allowed to veer back to the right to their intended impact point. The swatch is pictured in Figure 5.

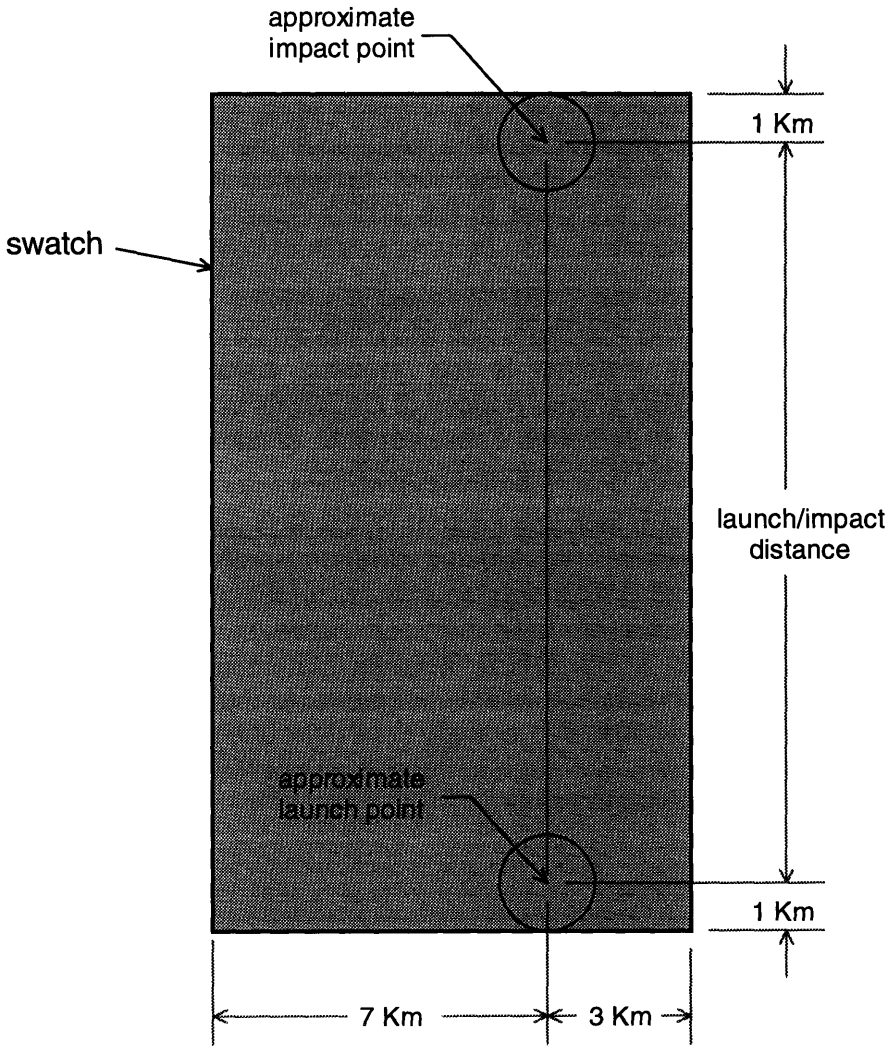


Figure 5. Swatch encompassing all possible ballistic trajectories

The DTED Request Algorithm first computes the swatch, and then determines the identifiers of all DTED blocks which lie either totally or partially within the area defined by the swatch (as shown in Figure 6). This list of identifiers is then sent to the DTED Controller as the DTED request.

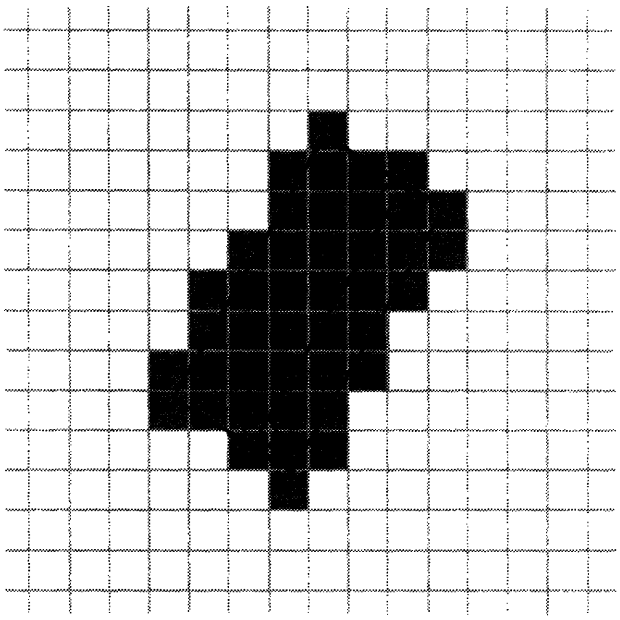


Figure 6. Block selection based on a swatch

3.2.2. DTED Controller

The DTED Controller is responsible for accepting the DTED request from the DTED Request Algorithms and loading the specified blocks into main memory. The DTED Controller also performs, as it reformats the data from DTED file format into blocks, a limited analysis of the data. This analysis is limited to what can be done in a single pass over the data. No iteration may be performed. This single pass corresponds to the single pass which is made as the DTED controller system transfers the data from the DTED files to main memory. Unlike the other components of a comparison system, this subsystem has no timing or memory requirements. The time and memory used by this system is not monitored, and therefore counts in no way towards the analysis or evaluation of the comparison system.

3.2.2.1. Justification for Block Organization of DTED

The purpose of the DTED Controller is to load blocks of DTED into main memory. Each block of DTED covers an area 180 arcseconds by 180 arcseconds. Each block also contains indexing and overhead information which describes the shape and location of the block as well as the organization of the DTED within the block. Each block also contains information specific to the block which may be useful for certain optimizations. The conversion from the original DTED format to the block format requires time and memory. However, this reformatting is a necessary procedure. The reasons for introduction of the block organization strategy are described below.

The DTED is originally stored on Compact Disc in the 1 degree by 1 degree cells which are described in the military specification for DTED [MIL89]. A single cell of level 2 DTED can contain as many as 12,967,201 elevation posts which requires 24.7 megabytes of memory (as each elevation post consists of a two byte integer). A cell can cover an area as large as 111.3 kilometers by 111.3 kilometers, so in the best case, the entire swatch will be contained within a single cell. Even if only one cell is needed, this may still require 24.7 megabytes of memory which far exceeds the 4 megabytes allowed for storage of data. Since the DTED cells are non-overlapping, in the worst case, a swatch may lie on the intersection of four separate cells which would require nearly 100 megabytes of memory which is certainly unavailable. Furthermore, four cells of DTED would cover nearly 50,000 square kilometers. This is far more than is needed.

The swatch described in Section 3.2.1 encompasses all the DTED which could possibly be needed by a comparison system. As stated previously, the approximate launch and impact locations may only be separated by at most 50 kilometers. This means that the largest swatch possible covers 10 by 52 kilometers or 520 square kilometers.

These facts all seem to indicate that the original storage format of the DTED is intractable for this system. A more reasonable approach is to organize the DTED into uniformly sized blocks which are substantially smaller than the original DTED cells. It has not been shown that a block organization is optimal, but other concepts have been considered and discarded. An alternate concept such as organizing the DTED into rows or columns is made difficult by the fact that the resolution of DTED may change across cell boundaries in both the latitude and longitude directions. Another alternate concept such as storing each elevation post individually is considered to be a special case of uniformly sized blocks with a size of 1 by 1. Thus for the purposes of this project, the DTED will be organized into uniformly sized blocks.

3.2.2.2. Selection of Block Size

As the surface of the earth is the surface of a sphere, there are two methods in which to measure areas on the surface: square kilometers or square arcseconds. The DTED is organized along the lines of the WGS 84 Geodetic Coordinate System which uses angular units. The result is that a square kilometer encompasses different numbers of elevation posts depending on where on the earth the square kilometer is located. A square arcsecond, on the other hand, will always encompass the same number of elevation posts regardless of location (except for variation in DTED resolution due to the latitudinal zones). Furthermore, the surface of the earth cannot be evenly tiled with square kilometers while this is not a problem with square arcseconds. For these reasons, the size of the DTED blocks will be measured in arcseconds of latitude by arcseconds of longitude.

In selecting a block size, it is optimal to choose the largest size possible while still remaining within the memory limits allowed for DTED storage. The larger block size decreases the amount of indexing information because this information is stored on a per block basis. However, a block size which is too large will result in excess unneeded data being carried piggyback by the DTED which is actually needed (as was shown to be the case with the original 1 degree by 1 degree cells). Furthermore, the shape of blocks should be roughly square as this also decreases the total amount of overhead needed.

One requirement which we would like to place on the block size is that, regardless of the DTED resolution, each block should have an elevation post located exactly at the southwest corner of the block. Thus there would only be a single possibility of placement of the posts within the block, resulting in only a single type of block to consider. To achieve this orientation, the length of a side of the block must be a perfect multiple of the post spacing of every resolution of DTED. The set of all possible post spacings (measured in arcseconds) is {1, 2, 3, 4, 6, 9, 12, 18}. Common multiples of this set include {36, 72, 108, 144, 180, 216, 252, 288, 324, ...}. The selection any of these values as a block size would allow the four block corners and the block boundaries to fall directly on elevation posts.

The next issue to consider is that it would be optimal for the block boundaries to coincide with the cell boundaries of the inherent DTED organization. This is necessary to prevent a block containing two different resolutions of DTED. The members of our previous candidate set which divide evenly into 3600 (1 degree is equivalent to 3600 arcseconds) are {36, 72, 144, 180, 360, 720, 900, 1800, 3600}.

An estimate of the memory required to store a swatch of DTED using a block size of x arcseconds by x arcseconds can be determined from the following equations:

$$\textit{Overhead per block} = 40\ \textit{bytes}$$

$$\textit{DTED per block} = \frac{2\ \textit{bytes}}{\textit{elevationpost}} \cdot \left(\frac{x}{\textit{latitude post spacing}} + 1 \right) \cdot \left(\frac{x}{\textit{longitude post spacing}} + 1 \right)$$

$$\textit{Indexing per block} = 14\ \textit{bytes}$$

$$\textit{Memory per block} = \textit{Overhead per block} + \textit{DTED per block} + \textit{Indexing per block}$$

An *Overhead per block* of 40 bytes was arrived at by including the following data:

4 bytes longitude reference
4 bytes longitude reference
2 bytes block size - longitude direction
2 bytes block size - latitude direction
2 bytes post spacing - longitude direction
2 bytes post spacing - latitude directions
2 bytes post count - longitude direction
2 bytes post count - latitude direction
2 bytes absolute vertical accuracy
2 bytes absolute horizontal accuracy
2 bytes relative vertical accuracy
2 bytes relative horizontal accuracy
12 bytes optimization information

An *Indexing per block* of 14 bytes was arrived at by assuming a binary tree indexing structure where each node contains the following data:

4 bytes longitude reference
4 bytes latitude reference
2 bytes block location index
2 bytes left branch index
2 bytes right branch index

The number of blocks which fall within the swatch is calculated assuming a worst case (a swatch whose long edge has a slope of 1 in the block space and which lies exactly on the block diagonals). The resulting storage requirements measured in megabytes are shown in Table 2. Any memory requirements which exceed the maximum memory allowed for DTED (4 megabytes) are highlighted in this table. It can then be seen that a block size of 180 arcseconds by 180 arcseconds is the largest block size which does not exceed the memory limitations. In fact this table shows that the optimal block size when considering memory usage is 72 arcseconds by 72 arcseconds. However, many of the algorithms which will be used in the comparison system require additional processing whenever the trajectory crosses a block boundary. Thus to minimize processing requirements which we may encounter in the future, we select the largest possible block size which remains within the memory limitations. This minimizes the amount of processing which must be done on a per-block basis.

	latitudinal zone				
block size	0° to 50°	50° to 70°	70° to 75°	75° to 80°	80° to 84°
36	1.814632	1.604588	1.482450	1.750809	2.123280
72	2.052984	1.669630	1.482811	1.705988	1.956367
144	2.730309	1.963171	1.618555	1.943396	1.984663
180	2.876667	2.202587	1.836451	2.023132	2.118090
360	4.723726	2.992264	2.334206	2.821856	2.691284
720	7.932541	5.461458	3.646196	4.481289	4.161215
900	12.387497	6.975924	5.690569	4.661224	4.412287
1800	30.933638	15.475531	10.322828	7.746477	7.238176
3600	98.932091	49.479881	32.995811	24.753777	20.639677

Table 2. DTED Memory Requirements per Block Size

3.2.2.3. Basic DTED Controller

Most of the comparison systems use a very basic form of the DTED Controller. This basic version uses blocks of size 180 arcseconds by 180 arcseconds. The resolution of the DTED within individual blocks is equal to the resolution of the original DTED from which the data was loaded. The block database holds a maximum of 200 blocks which is sufficient to cover a 52 by 10 kilometer swatch (the maximum swatch required) at any latitude.

The DTED Controller also computes a limited amount of summary information used for optimizing some algorithms. The primary piece of summary information produced is the minimum and maximum elevation contained within a block. As the DTED is loaded from the original DTED files into the block data structures, each block is tagged with the minimum and maximum elevation contained within that block.

3.2.3. DTED Processor

The DTED Processor is responsible for taking the DTED which has been loaded into main memory by the DTED Controller and preparing it for use by the Comparison Algorithms. The processed DTED may not exceed the memory constraints placed upon the amount of DTED allowable in main memory. The time taken by the DTED Processor will be included in the time taken by the entire system to perform the trajectory / terrain comparison.

The exact nature of the DTED processing depends on the format of the DTED which is needed by the comparison algorithms. Examples of DTED processing include finding the four DTED posts which surround a given location, interpolating the elevation of terrain between existing DTED posts, and constructing an octree data structure as an index into the DTED database.

3.2.4. Path Processor

The Trajectory Processor is responsible for taking the trajectory points which are output by the ballistic model and preparing them for use by the Comparison Algorithms. The time taken by the Trajectory Processor will be included in the time taken by the entire system to perform the trajectory / terrain comparison.

The exact nature of the trajectory processing depends on the format of the DTED which is needed by the comparison algorithms. Examples of path processing include polynomial interpolation of the trajectory path, and linear interpolation of the trajectory path.

3.2.5. Comparison Algorithms

The Comparison Algorithms are responsible for taking the trajectory data and DTED which have been processed by their respective processing systems, and comparing them against each other to determine the locations where the trajectory intersects the terrain. The time taken by this comparison will be included in the time taken by the entire system to perform the trajectory / terrain comparison.

The Comparison Algorithms must be able to determine if the trajectory intersects the terrain. If an intersection occurs, the location of the intersection must be determined. The exact nature of the Comparison Algorithms will vary depending on the nature of the comparison algorithm being evaluated. Differences in the comparison algorithms will dictate the need for various Path and DTED Processors to provide data to the Comparison Algorithms in the proper format.

The Comparison Algorithms is the primary subsystem in any comparison system. Furthermore, it is this subsystem which is responsible for producing the list of intersection points which constitutes the system output.

3.3. System Output

The result of any of the comparison systems is a list of trajectory / terrain intersection points. The intersection points must be presented in both the Trajectory Coordinate System as well as the WGS 84 Geodetic Coordinate System. A definition and description of an intersection point is given below.

Throughout this project the term "*intersection point*" will be used to refer to any point along the trajectory at which the trajectory either enters or exits the terrain. Intersection points are classified into two categories: "*entry intersection points*" and "*departure intersection points*". These two types of intersection points are graphically depicted in Figure 7.

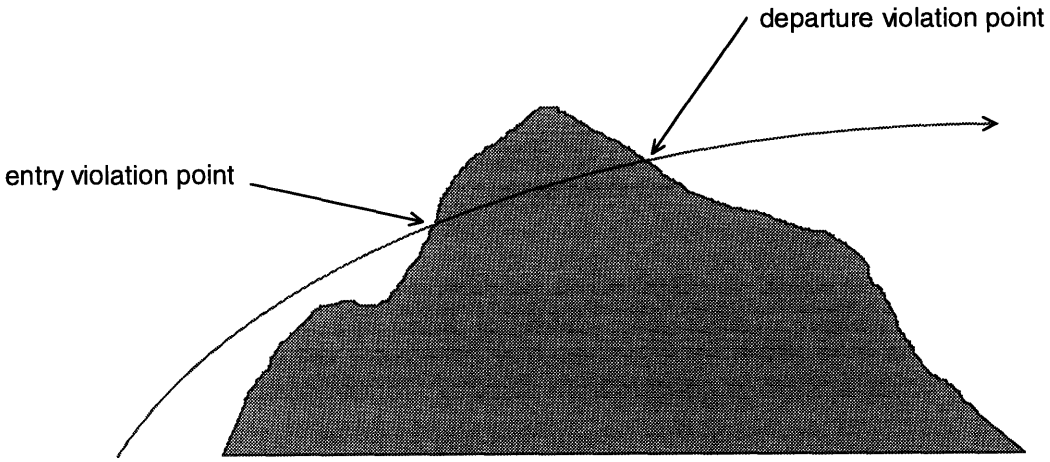


Figure 7. Intersection Points

An "*entry intersection point*" is an intersection point such that, if the trajectory is traversed in a positive downrange direction, the trajectory point immediately preceding the entry intersection point is not in violation of the terrain. Likewise, a "*departure intersection point*" is an intersection point such that, if the trajectory is traversed in a positive downrange direction, the trajectory point immediately following the departure intersection point is not in violation of the terrain.

3.4. System Analysis and Evaluation

The primary result of this project is the analysis and evaluation of the three candidate comparison systems. We wish to determine if there is a superior choice among the systems which are being compared.

The analysis of a comparison system will consist of examining the approach taken by a particular comparison system in an attempt to determine any erratic or possibly faulty behavior which the system may exhibit, but which may not necessarily be ferreted out by experimentation. This may require numerical analysis, the construction of mathematical models, or other such theoretical techniques. The aspects of system behavior which will be analyzed are described below in Section 3.4.1, Evaluation Criteria. They include such criteria as accuracy, processing time, memory usage, etc.

The evaluation of a comparison system will consist of implementing the system on a representative processor, and then examining the behavior of the model as it performs a suite of test cases. The behavior which will be examined is the same as for the analysis, and is described below in Section 3.4.1, Evaluation Criteria.

3.4.1. Evaluation Criteria

Each comparison system will be analyzed and evaluated with respect to five criteria: adherence to space limitations, correctness of results, processing time, accuracy of results, and space utilization. While it would be desirable to optimize each comparison system for all five of these criteria simultaneously, this is not always possible. Optimization often involves tradeoffs between these different criteria. Therefore the priority in which the above criteria will be considered is, in order: space limitations, correctness, time, accuracy, and space usage.

3.4.1.1. Space Limitations

The space limitations are of primary importance. Each of the candidate comparison systems is required to remain within the memory limitations imposed by the computing environment. These include a 4 megabyte limit on data storage and a 1 megabyte limit on stack space.

To determine whether a comparison system observes the space limitations, each system will be monitored for memory usage during the execution of the test cases. In additional to this experimental evaluation, any comparison approach which may involve a substantial amount of additional memory requirements will also undergo an analysis to determine the hypothetical maximum memory usage.

3.4.1.2. Correctness

Correctness refers to the ability of a comparison system to correctly predict a trajectory / terrain intersection when one actually exists and to restrain from predicting trajectory / terrain intersections when none exist. A system has correctly predicted all intersection points if, for every actual intersection point, the system predicts an intersection of the correct type (entry or departure) within 100 meters (in the horizontal direction) of the actual intersection. A comparison system has correctly restrained from predicting non-existing intersections if, for every predicted intersection point, there exists an actual intersection point of the correct type (entry or departure) within 100 meters (in the horizontal direction) of the predicted intersection location.

An exception to the above definition of correctness involves what are termed "anomalies". "Anomalies" are defined to be areas of intersection (or non-intersection) which last for less than 100 meters. Because of the 100 meter "correctness zone", it is acceptable for a comparison system to entirely skip the intersection points associated with anomalies and still be considered "correct".

The correctness of a comparison system will be evaluated analytically by examining the worst possible accuracy of a system to determine if it surpasses the 100 meter correctness limit. The correctness is also evaluated experimentally by implementing the comparison system and checking its results against the results obtained by a truth model.

3.4.1.3. Time

How well a comparison system performs with respect to time refers to how quickly the comparison system completes the trajectory / terrain comparison. However, not all of the processing performed by the comparison system contributes to the time measurement by which the time efficiency of a system is evaluated. In particular, the time taken by the DTED Controller is not counted towards the time measurement of the comparison system. The subsystems which do contribute towards the total measurement are shown highlighted in Figure 8.

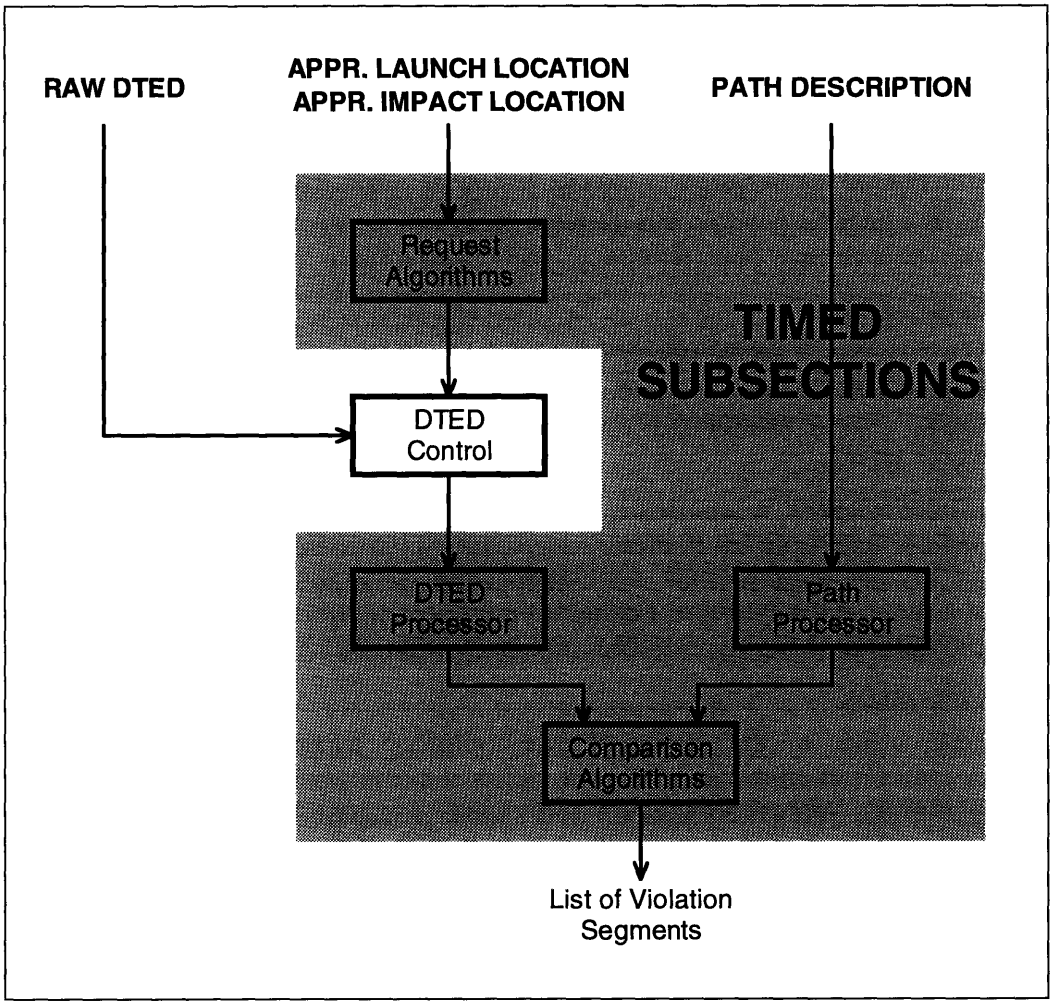


Figure 8. Subsystems which contribute to time usage.

The DTED Controller is a stand-alone system with separate requirements placed upon it which assure that a request for DTED is responded to in constant time, regardless of the size of the request (as long as the resulting DTED observes memory limitations) and regardless of how many DTED CDs must be accessed in order for this subsystem to complete its assignment. Since this time is constant for all comparison systems, it will not be considered when evaluating the different comparison systems against one another.

The timing analysis will be performed by implementing each of the comparison systems on a representative hardware platform and taking timing measurements from the appropriate subsystems.

3.4.1.4. Accuracy

As stated in the section dealing with correctness, the intersection predictions produced by a comparison system must be within 100 meters (in the horizontal direction) of an actual intersection to be considered "correct". How closely the intersection points correspond within this 100 meters, as well as their correspondence in the vertical direction, is an indication of accuracy.

The accuracy will be measured by implementing each of the comparison systems and then comparing their results with the results of a truth model.

3.4.1.5. Space Usage

Least important of all the evaluation criteria is the space usage. This refers to how much data memory (below the 4 megabyte limit) is used by the comparison system in performing the trajectory / terrain comparison. Space usage will be measured experimentally by monitoring the data memory used by the comparison system as the test case comparisons are performed.

3.4.2. Model Implementation

For the purpose of experimental evaluation, each comparison system will be implemented on a representative processor platform. The implementation will then be used to run a suite of test cases allowing evaluation data to be collected.

The representative processor platform which will be used is an Intel 486dx processor running at 66 megahertz with 8 megabytes of RAM available. This processor includes a built-in dedicated floating-point processor. The systems will be implemented using the C++ programming language and will be compiled using the Borland C++ compiler for the Microsoft Windows operating system.

Where possible, subprograms and data structures will be re-used in order to provide continuity and a basis for comparison from model to model. However, the implementation details for each model will not be supplied in this paper, nor will detailed descriptions of the algorithms being implemented.

3.4.3. Test Cases

The system evaluation will consist of running each comparison system on a suite of 8 test cases. These test cases have been chosen in an attempt to represent a wide range of possible comparison situations.

3.4.3.1. Available DTED

As DTED is a necessary prerequisite for a comparison to be performed, the selection of locations for test cases was limited to areas in which DTED was available. DTED is a limited distribution product, and the amount of DTED made available to this project reflects this.

A detailed description of the areas of DTED made available to this project will not be given, but it is necessary to say that all the available portions of DTED lay in the northern hemisphere between 0° and 50°N latitude. Therefore it was only possible to evaluate the comparison systems in one latitudinal DTED zone, and thus only a single DTED resolution. Furthermore all of the available DTED was Level 1, so no high density DTED evaluations were possible.

3.4.3.2. Test Criteria

There are several trajectory characteristics which we wish to exercise in choosing a suite of test cases. These include trajectory length, launch angle, trajectory azimuth, etc. In addition to these basic trajectory criteria, we also wish to exercise the comparison systems over several terrain characteristics. These include terrain roughness as well as the presence of near-launch intersections and near-impact intersections. These criteria are discussed individually below.

The length of trajectories which will be considered in this project range from 20 kilometers to 50 kilometers. Trajectories may therefore be broken into three length categories: short, medium, and long. Short trajectories are 20 to 30 kilometers in length. Medium trajectories are 30 to 40 kilometers in length. And long trajectories are 40 to 50 kilometers in length. Representative short, medium, and long trajectories will be 20, 35, and 50 kilometers respectively.

The launch angle of a trajectory refers to the angle, measured from the horizontal, at which the projectile leaves the launch point. Launch angles may range anywhere from 0 to 90 degrees above the horizontal. However, higher angle trajectories are generally uninteresting because the trajectory paths at the launch and impact points begin to resemble vertical lines which makes intersection calculations trivial. Therefore we will consider only two classifications of launch angle, namely "low angle" and "high angle". These classifications are not absolute definitions with respect to the horizontal, but instead depend on the length of the trajectory being considered (i.e., a low angle for a trajectory which is 50 kilometers long will necessarily be much greater than the low angle for a trajectory which is 20 kilometers long, since the projectile must be launched higher in order to attain the greater distance). Low angle trajectories are trajectories which reach their desired impact point with a launch velocity set to the maximum allowable 1500 meters per second. High angle trajectories are 10 degrees higher than the low angle, with the launch velocity adjusted as appropriate.

The trajectory azimuth refers to the direction, as measured from the launch site, in which the impact location lies. This measurement is generally made in relation to true north. The trajectory azimuth determines how the trajectory path will fall along the DTED grid, which is aligned along north/south, east/west lines. In order to assure that there are no directional tendencies among the trajectories in the test case suite, the initial azimuths will be widely varied. Note that the "trajectory azimuth", referred to here, is different from the "launch azimuth". Because the projectiles are spin stabilized, they veer slightly to the right as they fly, thus the launch azimuth must be slightly to the left of the trajectory azimuth. The Coriolis force does not have a noticeable effect on the projectile.

When speaking of terrain roughness, we generally recognize three granularities of roughness: rough, medium, and smooth. An example of rough terrain might be the Grand Canyon, or portions of the rocky mountains, or the Badlands of South Dakota. Medium terrain can be found in the foothills of the Appalachians, or the rolling hills of the Scottish highlands. Smooth terrain is characteristic of the great central plains of North America. In this area the terrain resembles a flat plane, which makes intersection determination rather trivial. For the purposes of this project, therefore, smooth terrain is generally uninteresting and will not be considered.

All test case trajectories are launched from beneath the terrain. This causes the launch point to be considered an entry intersection point and the point where the trajectory first leaves the terrain to be considered a departure intersection point. The presence of a "near-launch intersection" means that there are additional terrain entry and departure intersection points before the projectile reaches its apogee.

Similar to the launch point, all test case trajectories have a final trajectory point which is beneath the terrain. This causes there to be a final entry intersection point where the trajectory intersects the terrain for the last time. The presence of a "near-impact intersection" means that there are additional terrain entry and departure intersection points after the projectile has reached its apogee and before it has entered the terrain for the last time.

3.4.3.3. Description of Test Cases

For the purpose of determining general algorithm performance, 8 representative test cases are selected. When selecting actual test cases, a wide variation in the test case criteria discussed above is necessary. The actual variation desired is shown in Table 3. The column labeled "additional intersections" in this table refers to the presence of near-launch or near-impact intersections in the trajectory.

Note that Table 3 does not address trajectory azimuths or launch latitudes, both of which factor into the comparison algorithms. This omission is intentional. It is quite difficult to choose, a priori, the exact location and direction of the trajectories for test cases. Instead, the terrain available through DTED was examined for features which would best exercise a basic comparison system, and then trajectories were chosen which could nominally encounter these features. It should be noted, however, that within this method, an attempt was made to vary these parameters as widely as possible, observing, of course, the limited coverage of DTED available.

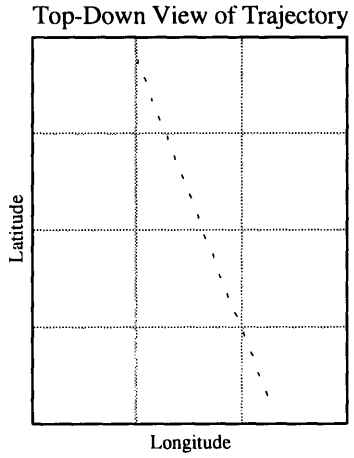
test case #	terrain type	launch angle	trajectory length	additional intersections
1	medium	low	short	yes
2	medium	high	short	none
3	medium	high	medium	none
4	medium	low	long	none
5	rough	low	short	yes
6	rough	high	short	yes
7	rough	low	medium	yes
8	rough	high	long	none

Table 3. General description of test cases.

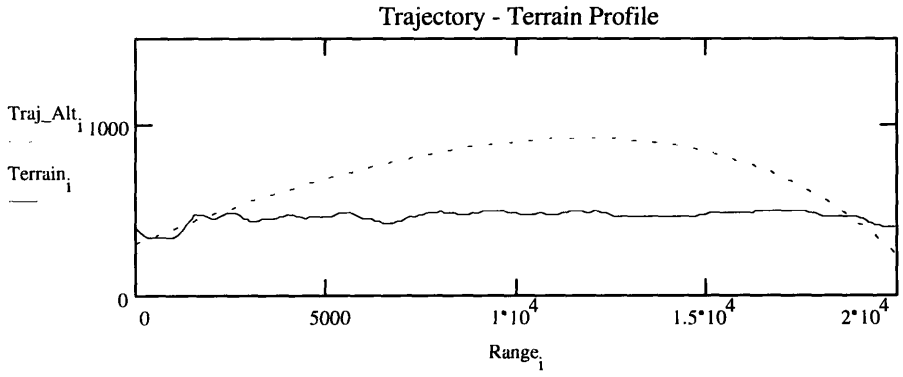
The 8 test cases which comprise the basic test suite are described individually below. The exact location of each test site will not be given, except for the latitude, as the latitude of the launch site is a factor in determining the shape of the trajectory.

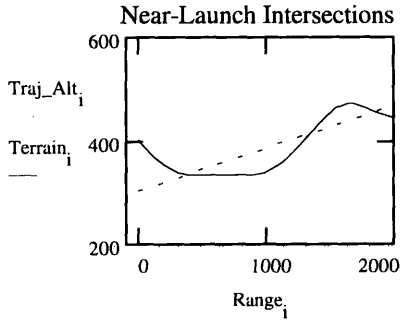
Description of Test Case #1

Terrain Type	Medium - 154 meter variation from highest to lowest
Launch Angle	Low - 5.011° (relative to horizontal) at 1500 meters/sec
Traj. Length	Short - 20 kilometers
Launch Latitude	41.465 degrees north latitude
Trajectory Azimuth	345° (relative to true north)
Additional Intersections	2 Near-Launch Intersections



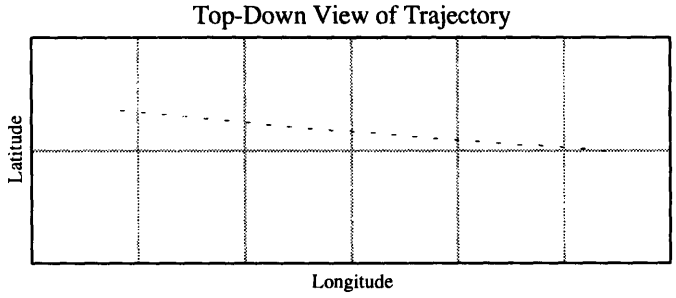
Grid lines represent
DTED block boundaries



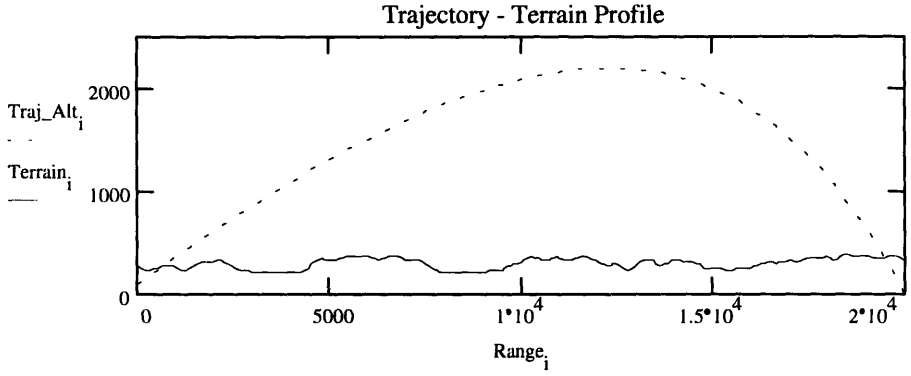


Description of Test Case #2

Terrain Type	Medium - 184 meter variation from highest to lowest
Launch Angle	High - 15.112° (relative to horizontal) at 1076.806 meters/sec
Traj. Length	Short - 20 kilometers
Launch Latitude	40.05 degrees north latitude
Trajectory Azimuth	276° (relative to true north)
Additional Intersections	None

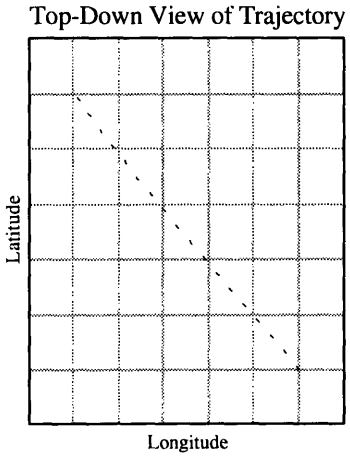


Grid Lines represent DTED block boundaries

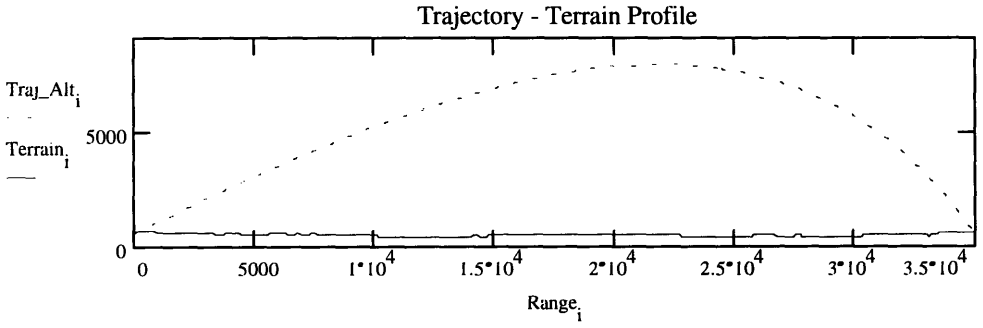


Description of Test Case #3

Terrain Type	Medium - 244 meter variation from highest to lowest
Launch Angle	High - 27.132° (relative to horizontal) at 1291.052 meters/sec
Traj. Length	Medium - 35 kilometers
Launch Latitude	41.05 degrees north latitude
Trajectory Azimuth	323° (relative to true north)
Additional Intersections	None

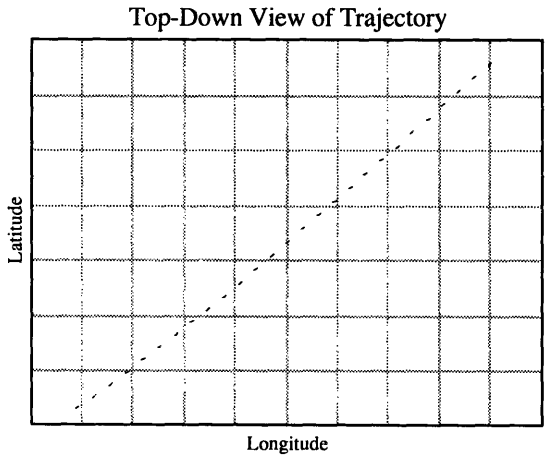


Grid lines represent
DTED block boundaries

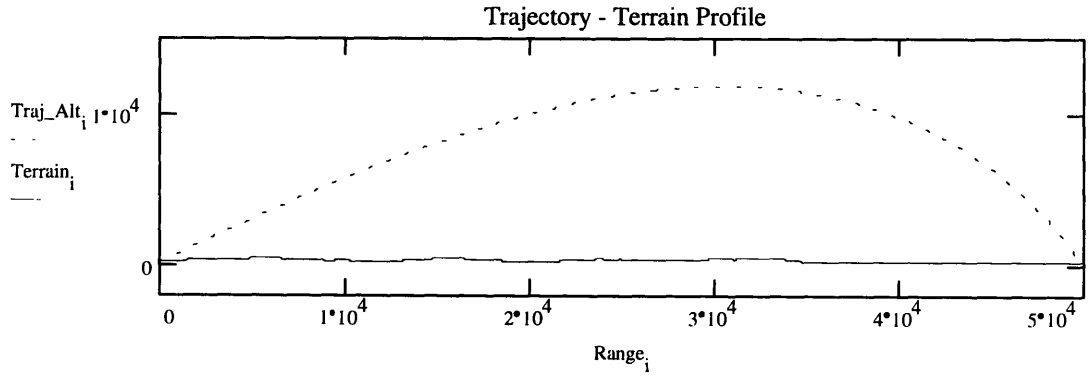


Description of Test Case #4

Terrain Type	Medium - 337 meter variation from highest to lowest
Launch Angle	Low - 30.673° (relative to horizontal) at 1500 meters/sec
Traj. Length	Long - 50 kilometers
Launch Latitude	40.13 degrees north latitude
Trajectory Azimuth	225° (relative to true north)
Additional Intersections	None

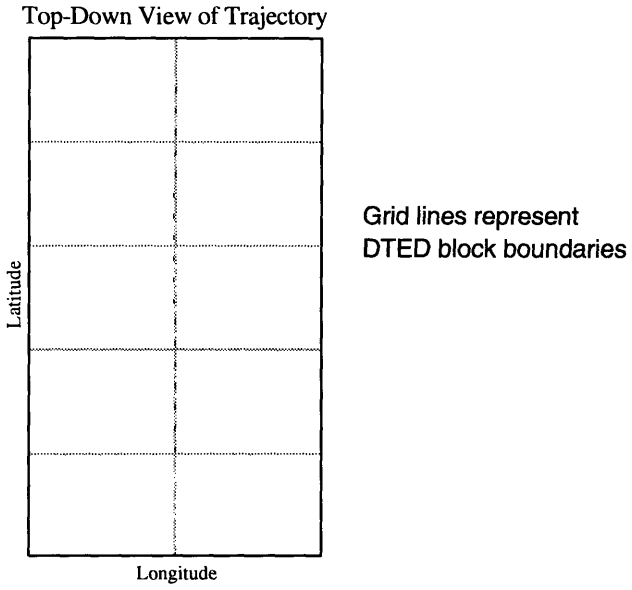


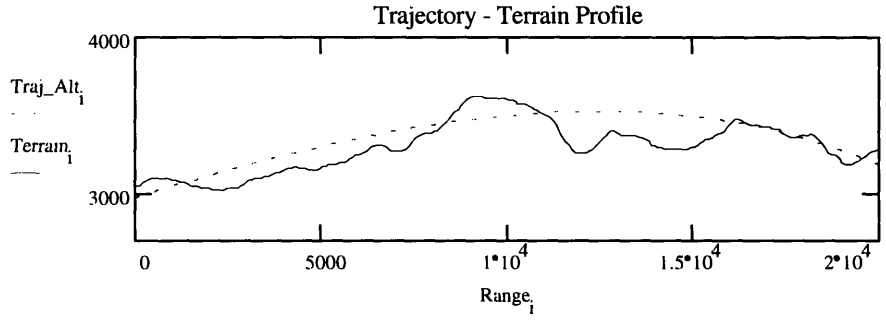
Grid lines represent
DTED block boundaries

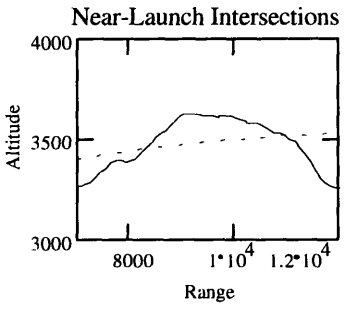


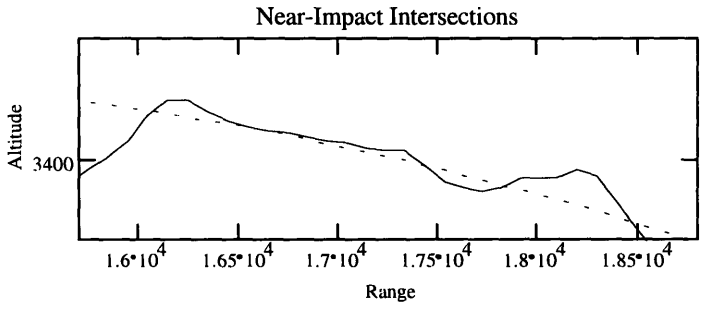
Description of Test Case #5

Terrain Type	Rough - 597 meter variation from highest to lowest
Launch Angle	Low - 4.456° (relative to horizontal) at 1500 meters/sec
Traj. Length	Short - 20 kilometers
Launch Latitude	39.1 degrees north latitude
Trajectory Azimuth	0° (relative to true north)
Additional Intersections	2 Near-Launch Intersection, 4 Near-Impact Intersections



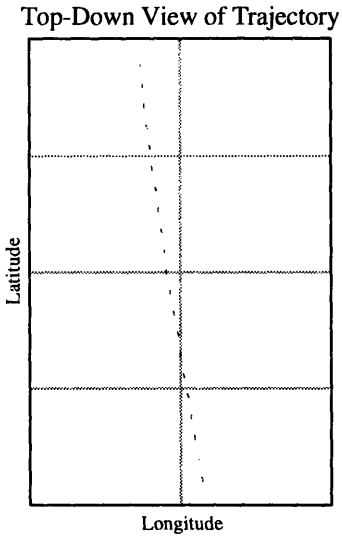




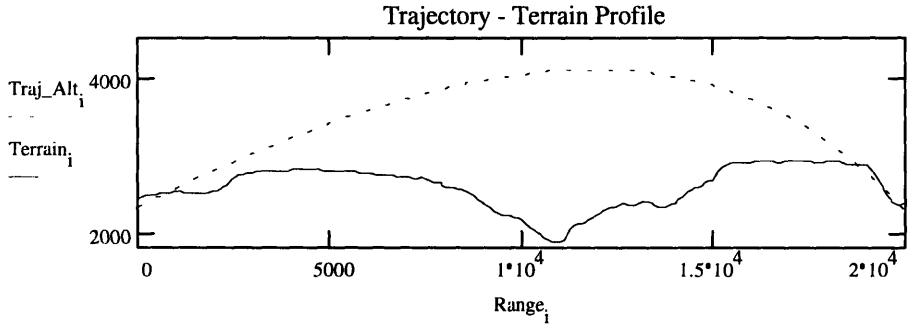


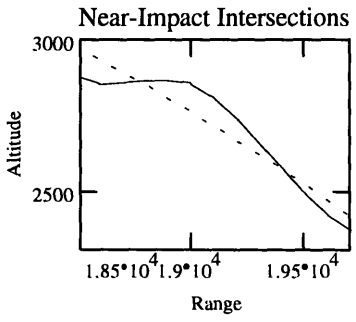
Description of Test Case #6

Terrain Type	Rough - 1050 meter variation from highest to lowest
Launch Angle	High - 13.916° (relative to horizontal) at 987.174 meters/sec
Traj. Length	Short - 20 kilometers
Launch Latitude	39.51 degrees north latitude
Trajectory Azimuth	355° (relative to true north)
Additional Intersections	2 Near-Impact Intersections



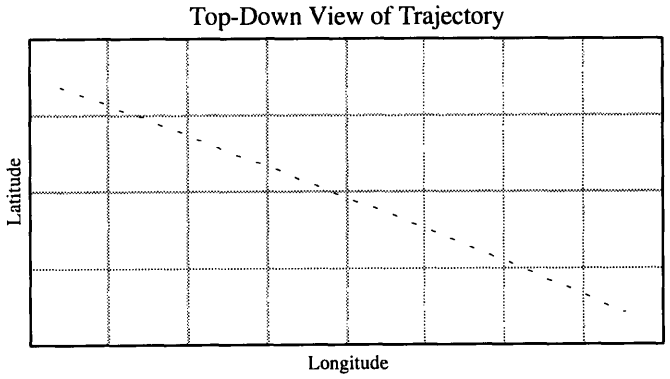
Grid lines represent
DTED block boundaries



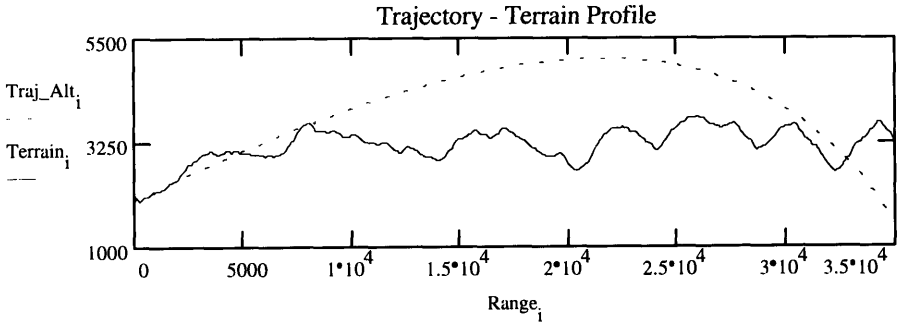


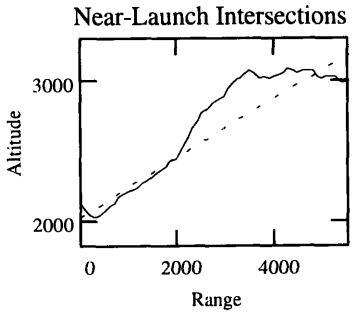
Description of Test Case #7

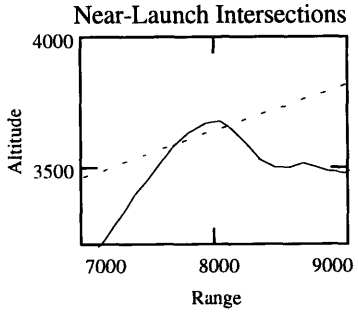
Terrain Type	Rough - 1777 meter variation from highest to lowest
Launch Angle	Low - 12.635° (relative to horizontal) at 1500 meters/sec
Traj. Length	Medium - 35 kilometers
Launch Latitude	39.269 degrees north latitude
Trajectory Azimuth	118° (relative to true north)
Additional Intersections	4 Near-Launch Intersections



Grid lines represent DTED block boundaries

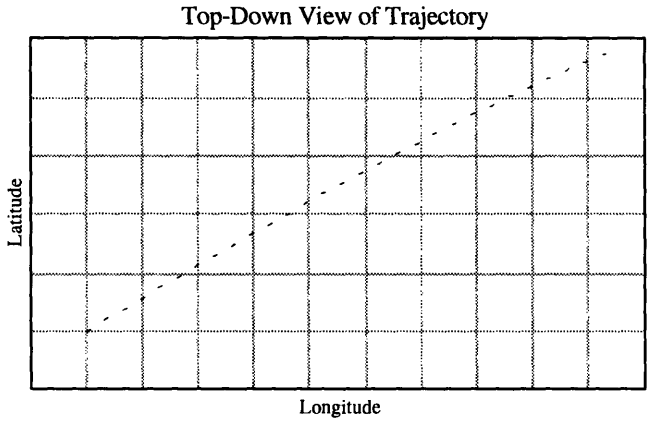




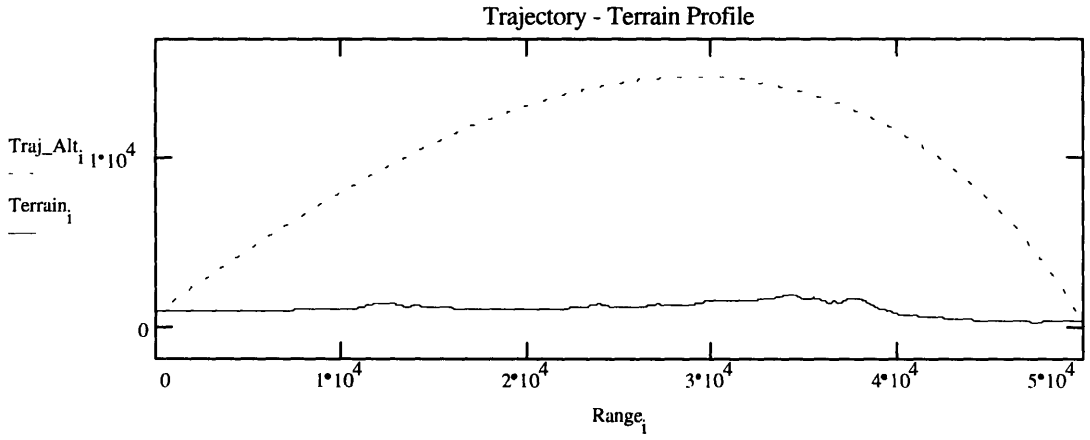


Description of Test Case #8

Terrain Type	Rough - 1656 meter variation from highest to lowest
Launch Angle	High - 37.967° (relative to horizontal) at 1340.591 meters/sec
Traj. Length	Long - 50 kilometers
Launch Latitude	35.35 degrees north latitude
Trajectory Azimuth	58° (relative to true north)
Additional Intersections	None



Grid lines represent
DTED block boundaries



4. TRUTH MODEL COMPARISON SYSTEM

4.1. Basic Description

The Truth Model Comparison System is a brute-force algorithm which will yield the intersection locations used to judge the correctness and accuracy of the other comparison systems. The basic premise is to determine the exact location of the projectile as well as the exact height of the terrain for each downrange along the length of the trajectory. The altitude of the projectile is then compared directly to the elevation of the terrain to determine where the intersection points are located. This will obviously yield the intersection location to within a meter of accuracy.

This system requires that the location of the projectile be determined to a resolution of 1 meter. This resolution is much greater than the resolution produced by the ballistic model, which only locates the projectile at intervals of approximately 100 to 500 meters. Determining the projectile location therefore requires some form of interpolation using the calculated trajectory points. Thus the Path Processor is responsible for interpolating the points along the trajectory which will by checked against the terrain elevations.

Likewise, the terrain elevation is required at a resolution of 1 meter. The resolution of DTED usually ranges between 30 and 100 meters. Therefore the terrain elevation between existing DTED posts must be determined through some form of interpolation. Thus the DTED Processor is responsible for interpolating terrain elevations at the discrete locations where the comparison will be performed.

Although the path and terrain values just described are being interpolated through some means, and therefore cannot be verified, the results obtained by the Truth Model will nevertheless be considered to be the truth, just as if an actual projectile had been flown at the actual location under the exact conditions described, and the intersection points had been determined through observation and measurement.

Note that the Truth Model does not represent the *most accurate* comparison system possible. Instead it represents a system which is *more accurate than is possible.* This is accomplished by ignoring the limitations of the existing data and choosing an arbitrary trajectory and arbitrary terrain to represent the actual trajectory and terrain. Thus the candidate comparison systems are not expected to obtain the exact same answers as the Truth Model, since this would be impossible with the information provided to them. Instead we assume that the Truth Model produces a reasonable set of intersection points and we wish to determine how accurately each comparison system predicts those intersection points.

The objective of the Truth Model is therefore to produce intersection data which would be reasonable considering the given trajectory and terrain data. Both memory and timing limitations may be sacrificed as the Truth Model is not a possible candidate in the search for the best comparison system. The difficulty in the Truth Model is therefore to create an arbitrary trajectory and arbitrary terrain when neither of these can be completely known with certainty. The location of the projectile between the discrete path points, as well as the elevation of the terrain between elevation posts is essentially unknowable.

4.2. Truth Model System Components

4.2.1. DTED Request Algorithms

The DTED Request Algorithms for the Truth Model simply request the rectangular swatch which was described in Section 3.2.1.

4.2.2. DTED Controller

The DTED Controller for the Truth Model simply makes available the DTED in 180 arcsecond by 180 arcsecond blocks as described in Section 3.2.2.

4.2.3. DTED Processor

4.2.3.1. Truth Terrain

A truth model of the terrain must describe the elevation of the terrain to a higher resolution than is available through DTED. The altitude of locations which fall between DTED posts must be known. The specification for DTED makes it clear that it is impossible, using only DTED, to determine the elevation of the terrain which lies between the posts. It is outside the scope of this project to take on-site measurements of the elevation of the terrain at locations between the known DTED elevation posts, and therefore an artificial terrain at a resolution higher than the available DTED will be arbitrarily created without regard to how closely this terrain reflects the elevation of the actual earth's surface. It is necessary, however, for this artificially created terrain to sufficiently correspond to the existing DTED such that the terrain model is within the tolerances named in the DTED specification.

The truth model of the terrain may be any reasonable terrain altitude map which, at the location of the DTED posts, corresponds to the DTED altitude within the specified DTED error tolerances. The term "reasonable" used here is quite general, as the number and forms of actual geological formations on the earth are so varied that nearly any artificially created terrain may be considered valid. As it is therefore not possible to derive an algorithm which would indicate which artificial terrain interpolations are reasonable and which are not, we must therefore rely on the common sense of the terrain's creator not to take undue liberties.

4.2.3.2. Construction of Truth Terrain

As has been stated above, the truth model for the terrain may be any reasonable terrain altitude map which, at the location of the DTED posts, corresponds to the DTED altitude within the specified DTED error tolerances. In practice, the truth model of the terrain will match the DTED posts exactly. Between the posts, an artificial terrain will be created which will be composed of posts at 128 times the resolution of the DTED, regardless of the DTED density.

The creation of the terrain at 128 times the resolution of the DTED will be accomplished through a fractal-based interpolation. The method used for this interpolation is a modified version of the fractal-based interpolation of terrain described by Yokoya et al [YOKOY89]. This technique was originally proposed to derive terrain models for graphics rendering, but it yields an artificially created terrain which is useful in this capacity as well.

4.2.3.3. Basic Description of Fractal Interpolation of Terrain

Yokoya's technique first determines the fractal characteristics of the terrain from the information which is already known of the terrain. Using the assumption that the fractal characteristics will remain the same, regardless of scale, it is then possible to interpolate reasonable terrain at theoretically infinite resolution. However, since we are attempting to model natural terrain, and natural terrain does not actually maintain fractal characteristics to infinite scales, the choice was made to stop at a resolution of 128 times the original. The number 128 was chosen as it is the first power of 2 which is greater than 100. And as the post spacings of DTED level 1 are spaced at roughly 100 meter intervals, the number 128 gives us a resolution which is finer than 1 meter.

The fractal characteristics which must be determined using available terrain data are the parameters H and σ where H is associated with the fractal dimension and σ is the standard deviation of the distribution function. The theory behind these parameters will not be discussed here. We will only show how the parameters are obtained and how they are used to perform an interpolation.

The parameters are derived from the following equation.

$$\log E\left[\left|f(\mathbf{x}+\Delta\mathbf{x})-f(\mathbf{x})\right|\right]-H\log\|\Delta\mathbf{x}\|=\log C$$

where : $C=\frac{2}{\sqrt{2\cdot\pi}}\cdot\sigma$

In the equations above, **x** represents a 2D location on the surface of the earth, and *f*(**x**) represents the altitude at that location. Since both *H* and *C* are constants (for a given **x**), a plot of $E\left[\left|f(\mathbf{x}+\Delta\mathbf{x})-f(\mathbf{x})\right|\right]$ as a function of $\|\Delta\mathbf{x}\|$ on a log-log scale should approximate a straight line with a slope of *H* and an intercept of log *C*. Thus it is possible, by plotting points derived from real DTED data, to determine the values *H* and *C*, and from these obtain the characteristic parameters of the terrain, H and σ. This plot on the log-log scale is referred to as the "Fractal Plot" and is demonstrated in Figure 9. This plot, taken from the mountainous terrain in western Colorado, USA, shows the linear nature of the fractal plot when graphed on a log-log scale. The x-axis of the graph is the distance in meters from the reference location. The y-axis is the expected difference between the terrain elevation at the reference location and the elevation of terrain x meters away from the reference location.

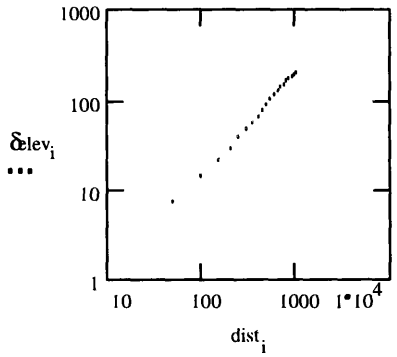


Figure 9. Fractal Plot

To interpolate an area of terrain, we first determine a reference location which is labeled **x**, and then plot points of the form $\left(log\|\Delta\mathbf{x}\|,\ log\left|f(\mathbf{x}+\Delta\mathbf{x})-f(\mathbf{x})\right|\right)$ for several values of $\Delta\mathbf{x}$. After plotting a number of points, a linear regression can be performed to determine the best straight-line approximation of the points, and this fitted line will yield the terrain parameters.

4.2.3.4. Determining Fractal Parameters

Consider the map in Figure 10:

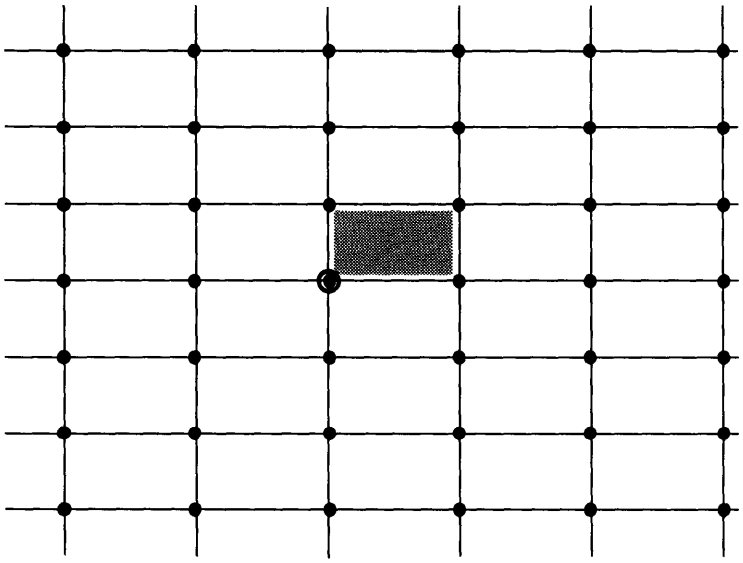


Figure 10. DTED map with square marked for fractal interpolation

In the map shown in Figure 10, we wish to interpolate terrain data for the shaded square. The dots which appear at the grid intersections represent DTED posts. We will always be interpolating an area which is encompassed by four adjacent DTED posts in the form of a rectangle. We begin by determining the reference location which will correspond to the **x** vector in the above equation. For this reference location we will use the DTED post which lies at the southwest corner of the square in question. This post has been marked on the figure with a circle. This choice of the southwest corner point as the reference post is somewhat arbitrary, but referencing areas by their southwest corner is the usual practice in DTED (as well as in this project). The f(**x**) term in the above equation is simply the elevation indicated by the DTED post at the reference location.

The expected difference between the reference elevation and the elevation at $\mathbf{x}+\Delta\mathbf{x}$ is approximated by the following method: First the terrain elevation is sampled at eight points, each of them $\Delta\mathbf{x}$ meters away from the reference location. The first sample is directly north of the reference location, the second sample is directly northeast, and so on for the other six of the eight cardinal directions. These sample locations may not necessary fall directly on elevation posts. In this case a linear interpolation is performed between the existing DTED posts to determine the elevation of the sample point. This may seem an invalid assumption, using linear interpolation to perform a fractal interpolation, but the scale at which the linear interpolation is performed is so large (in practice the interval for successive $\Delta\mathbf{x}$ values is 50 meters) that each individual sample point is interpolated using a unique set of DTED posts, and thus the fractal points are prevented from inheriting the linear nature of the linear interpolation. After all eight samples at a radius of $\Delta\mathbf{x}$ have been collected, the expected value of $|f(\mathbf{x}+\Delta\mathbf{x}) - f(\mathbf{x})|$ is approximated by averaging the values from each of the eight sample points.

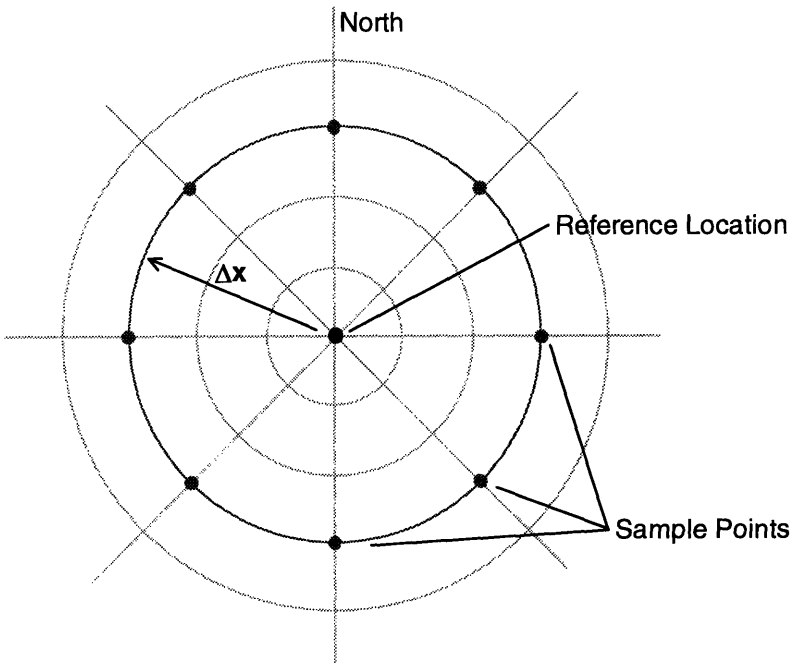


Figure 11. Location of sample points in relation to the reference location

After the points for the fractal plot have been determined, then the fractal plot is examined for linearity, and from this linearity the fractal parameters of the interpolation area can be determined. When determining the linearity of a fractal plot, a method is employed which differs slightly from that described by Yokoya et al. Yokoya divides the fractal plot into areas of piecewise linearity by searching for the points in the fractal plot which represent local maxima of linearity.

The method employed here is less accurate than Yokoya's method, but is somewhat faster and requires less terrain elevation data. Instead of dividing the fractal plot into areas of piecewise linearity, the cheaper method employed here only considers 20 fractal points which are spaced at 50 meter intervals from the reference location. Thus the area of DTED needed to determine fractal parameters is limited to a 1000 meter radius. Any of these 20 fractal points whose expected value for $|f(\mathbf{x}+\Delta\mathbf{x}) - f(\mathbf{x})|$ is zero are discarded, as the logarithm of zero is not well-defined. All of the remaining points are used to perform a linear regression. If the linear regression yields a valid slope (the term "valid" is described below) then the slope and intercept of the linear approximation may then be used to compute the fractal dimension (H) and the standard deviation (σ) using the equations described above.

It is possible for the linear regression to yield a value for H which is invalid. H is necessarily defined to lie between 0 and 1, and therefore the slope of the linear regression must also lie within these bounds. If the computed slope lies outside these bounds, then the slope is forcibly set to 0 or 1 (depending on which direction the error occurred). The intercept is then found by passing a line of the new slope through the remaining fractal point with the smallest $\Delta\mathbf{x}$ value.

Yokoya's method does not directly address the problem of invalid slopes. The problem is apparently caused by the limited area of DTED which is explored to obtain the points for the fractal plot (1 kilometer radius from the reference location). The value of H which is obtained from a linear regression of the fractal plot appears to be a function of the maximum radius from the reference location which is sampled for fractal points. In the case of DTED, radii of greater than 5 kilometers will generally yield a valid slope. However, this greatly increases the amount of DTED coverage necessary for the comparison. We attempt to show in the analysis of this method that the results obtained are adequate for the purposes of this project.

4.2.3.5. Fractal Interpolation Technique

After the fractal parameters have been obtained, additional terrain elevations may be interpolated according to the following interpolation algorithm described by Yokoya. Each iteration of Yokoya's interpolation doubles the resolution of the data points (where before the points lay on a regular grid of spacing x, the result of the interpolation is a regular grid of spacing x/2). Since we desire a resolution 128 times that of the original DTED, we iteratively apply Yokoya's interpolation technique seven times.

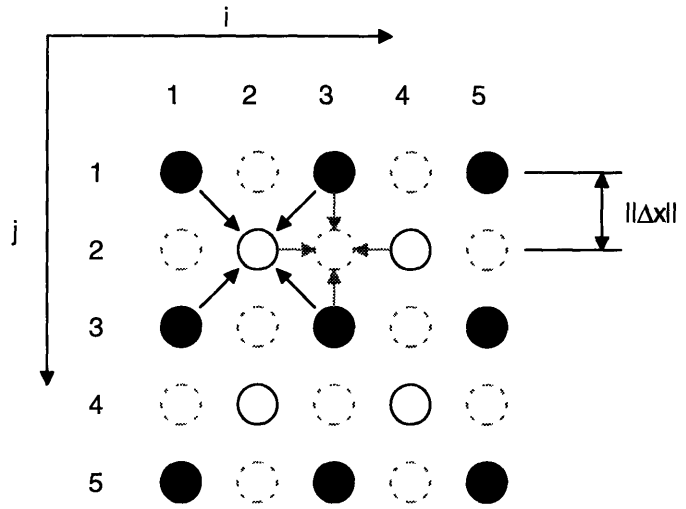


Figure 12. Recursive midpoint interpolation using four neighbors (from Yokoya et al.)

If both i and j are even numbers, then f(i, j) is computed using the four nearest neighbors as follows:

$$f(i,j)=\frac{1}{4}\{f(i-1,j-1)+f(i+1,j-1)+f(i-1,j+1)+f(i+1,j+1)\}$$
$$+\sqrt{1-2^{2H-2}}\cdot\|\Delta x\|^{H}\cdot\sigma\cdot Gauss$$

If only one of i and j is an even number, then f(i, j) can be computed using the four nearest neighbors as follows:

$$f(i,j)=\frac{1}{4}\{f(i-1,j-1)+f(i+1,j-1)+f(i-1,j+1)+f(i+1,j+1)\}$$
$$+2^{-H/2}\cdot\sqrt{1-2^{2H-2}}\cdot\|\Delta x\|^{H}\cdot\sigma\cdot Gauss$$

The *Gauss* term in the above equations refers to a Gaussian random variable with zero mean and unit variance.

Points which lie on the edge of the area being interpolated only have three points available for performing the interpolation. In this case, only the three points are used, with the obvious changes to the above equations.

Areas of interpolation which share a common boundary will have identical elevations for the points which lie along their common boundary. However, the choice of which area's fractal parameters are used to perform the interpolation along the boundary is arbitrary.

4.2.4. Path Processor

4.2.4.1. Truth Trajectory

To be considered a "true" trajectory path, all that is needed is a reasonably parabolic path which passes through all the points in the given trajectory description. In this case the term "reasonable" refers to a second or third order polynomial curve, calculated between adjacent sets of three or four adjacent known points. A second or third order polynomial curve is a valid assumption for the true path because the flight of a projectile is by nature a parabola. While the presence of an atmosphere or winds aloft may perturb the overall path from a perfect parabola, the behavior of the projectile in a local sense will still generally adhere to parabolic flight.

Thus, using a second or third degree polynomial, additional points can be interpolated which lie on the trajectory path between sets of locally adjacent known points. Together with the given path points, these interpolated points can be considered a reasonable trajectory path for which the given path points are a valid description. This, then, may be considered an example of a "true" path. And therefore we may assume that the results obtained through the use of any given comparison system should also be applicable to the behavior of this "true" path.

4.2.4.2. Construction of Truth Trajectory

As has been stated above, the truth model for the trajectory may be any reasonably parabolic path which passes through all of the points which are predicted by the ballistic model. In practice, a third degree polynomial curve will be fit to each set of four adjacent points along the ballistic model trajectory path. This polynomial will then be used to interpolate additional points along the trajectory at a resolution of one meter.

To interpolate the location of the projectile between the point i and point i+1, the four points from i-1 to i+2 are used to determine two third degree polynomials, one in the downrange/crossrange plane, and one in the downrange/altitude plane. Since, in the Trajectory Coordinate System, a trajectory is a function of downrange, the trajectory coordinates of the path points are used in the curve fitting, and the polynomials are both functions of downrange.

To determine the polynomial for the path between the first and second points in the trajectory, the first four points are used. To determine the polynomial for the last segment in the trajectory the last four points are used.

4.2.5. Comparison Algorithms

The Comparison Algorithms for the Truth Model are quite simple. The Path Processor will supply a number of discrete projectile locations where the trajectory comparison is to be performed. The DTED processor has determined the elevation of the terrain down to approximately 1 meter of resolution.

For each point in trajectory, the Comparison Algorithms perform the following steps:

1. Convert the projectile location from the Trajectory Coordinate System to the WGS 84 Geodetic Coordinate System.
2. Determine the terrain elevation point which is closest to the projectile location (in the two dimensional longitude/latitude plane). Because we only desire accuracy to within one meter, and the points are already within one meter of each other, no further interpolation is necessary

3. Compare the elevation of the trajectory location to the elevation of the terrain point selected in step 2. If the trajectory elevation is less than the terrain elevation, then this trajectory point is in violation of the terrain.

The comparison begins at the launch site and progresses systematically towards the impact. This makes it a simple matter to distinguish between entry intersection points as opposed to departure intersection points.

4.3. Analysis of Truth Model

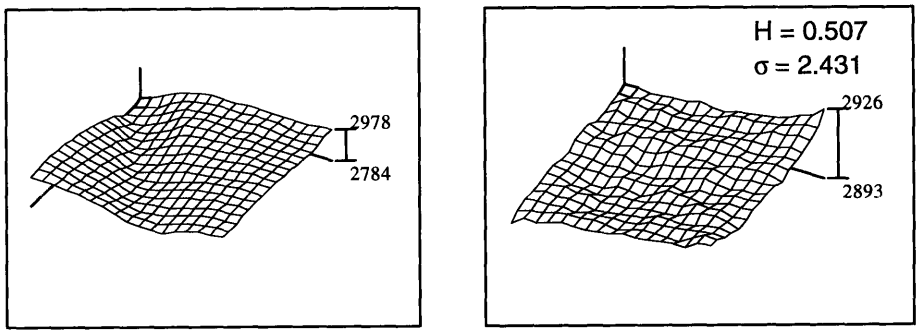
No correctness or accuracy analysis of the Truth Model is necessary as the results it obtains are defined to be exactly correct and exactly accurate. A memory and timing analysis is unnecessary because the Truth Model is not a candidate comparison system and therefore its performance will not be evaluated relative to other comparison systems.

However, we will examine both the terrain and path interpolation techniques in order to justify that the techniques used are valid and do in fact produce reasonable interpolated data which corresponds to the available DTED and trajectory data.

4.3.1. Terrain Interpolation

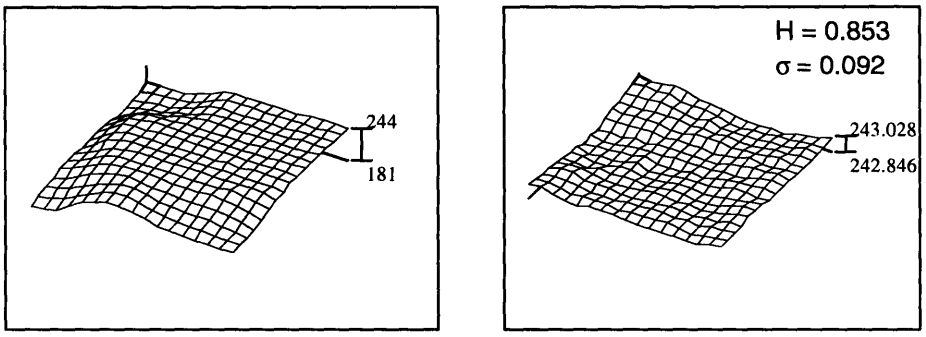
We wish to determine whether the fractal interpolation technique described in Section 4.2.3 actually produces a reasonable terrain model. It is already known that the interpolated terrain fits the DTED to within specified tolerances, since the interpolated terrain was defined to exactly match the DTED at the location of the DTED posts, which is the only real requirement which DTED has. To determine whether the interpolated terrain is "reasonable", we look at the existing terrain (at the scale of the DTED) and compare it to the interpolated terrain to see if the two terrains appear similar, despite the difference in scales.

The following three figures (Figure 13, Figure 14, and Figure 15) give views of terrain surfaces which were created using fractal interpolation along with views of the terrain from which the fractal parameters were determined. The surface labeled A in each of these figures was created directly from DTED using an array of 17x17 elevations. The reference location for determining the fractal parameters lies exactly at the center of each plot. The surface labeled B in each of these figures represents a fractal interpolation at 16 times the resolution of the original DTED (in both longitude and latitude). The piece of terrain shown in each figure B is the center square of terrain from figure A.



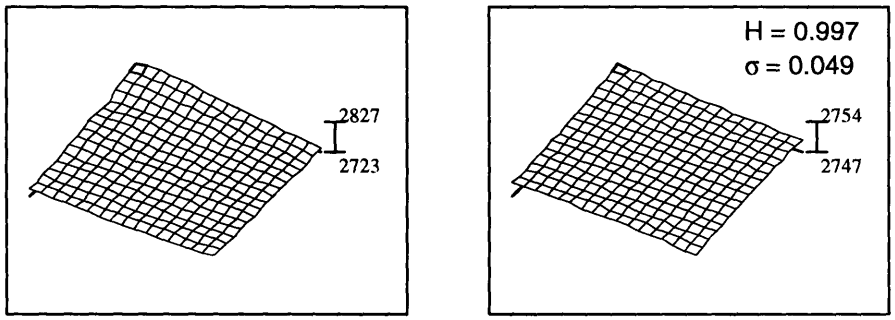
A B

Figure 13. Fractal Interpolation of Rough Terrain



A B

Figure 14. Fractal Interpolation of Medium Terrain



A B

Figure 15. Fractal Interpolation of Smooth Terrain

In general, the fractal interpolation technique described above appears to produce 'reasonable' terrain. The terrain appears to generally conform to the linear interpolation on which it is based. It does not contain extreme spikes or appear unusually noisy.

The primary argument against the method is that the fractal parameters do not always match very well to the terrain from which they were derived. This lack of accuracy was mentioned in Section 4.2.3.4 and is apparent in the graphs shown above. Smooth variations, such as riverbeds, are mistaken for surface noise which translates into a smaller H value which in turn results in a rather rough interpolated surface. Note this effect especially in Figure 13. Errors with just the opposite effect are also possible (although none are apparent in the figures shown). Surface characteristics which are either too close or too far away from the reference location tend to be overlooked in the determination of the fractal parameters. This results in a higher H value than desired, which in turn yields an undesirably smooth interpolation.

After a set of fractal parameters have been determined, however, the interpolation algorithm works very well to produce terrain of the appropriate surface roughness. Observe the surfaces in Figure 16 which all have identical corner elevations but which were produced by forcing the H parameter to a specific value in order to produce three different terrain types. (Note that this is the same example terrain from Figure 13 above). In Figure 16 A, the H value of nearly 1 produces the characteristically smooth terrain. In Figure 16 B an H value of 0.5 yields slightly bumpy terrain. While the H value of nearly zero in Figure 16 C causes the random Gaussian variable to dominate the linear interpolation term, producing a very noisy and unlikely terrain.

Results from experimentation over a wide range of actual DTED have shown that the H parameter seldom, if ever, goes below a value of 0.4. Thus the terrain produced by the interpolation algorithm will tend to resemble the more likely types of terrain displayed in Figure 16 A and B, rather than the unlikely terrain displayed in Figure 16 C.

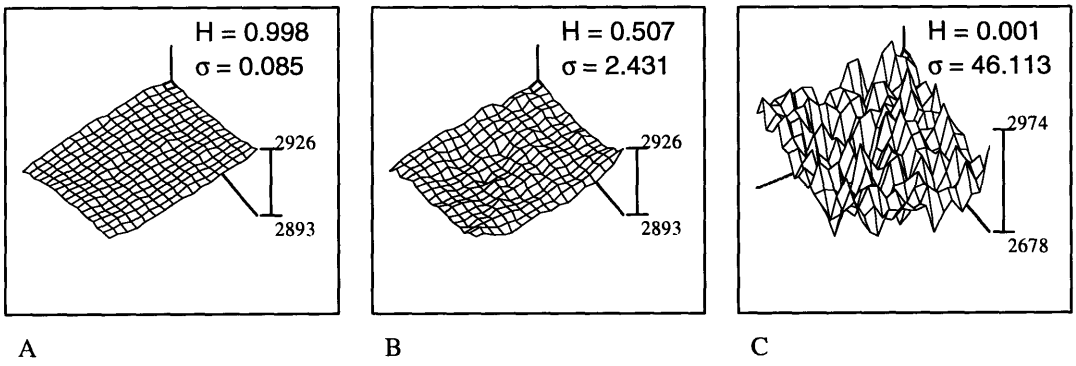


Figure 16. Surfaces produced with various fractal parameters.

Although, as mentioned above, the determination of the fractal parameters is not as accurate as could be desired, the fractal interpolation technique still appears to produce reasonable terrain which matches the DTED within the necessary specifications. Furthermore the less accurate method yields savings in calculation time as well as savings in the amount of DTED coverage needed to calculate the fractal parameters. For the purposes of this project therefore, the fractal interpolation technique described in Section 4.2.3 appears adequate.

4.3.2. Path Interpolation

We wish to determine whether the path interpolation technique described in Section 4.2.4 actually produces a reasonable trajectory. Because the technique successively applies a third degree polynomial curve fit to groups of four adjacent points, we know that the interpolated trajectory passes through all of the available data points. This is the only "true" requirement which the path interpolation algorithm has. However, we also wish to determine whether the trajectory produced is "reasonable" in a generally defined sense. In order to ascertain the "reasonableness" of the interpolation technique, we look at characteristics of the known trajectory points and assume that the trajectory displays similar characteristics between the known trajectory points.

Recall from Section 2.4.3 dealing with Trajectory Characteristics that the trajectory may be decomposed into an altitude component and a crossrange component, that these components are independent of one another, and that each component may be expressed as a function of the downrange position of the projectile. Thus in examining the interpolation algorithm for the path, we address each component individually.

We begin by examining the altitude component of the trajectory. A cursory examination of the trajectory computed by the ballistic model versus the trajectory computed by the interpolation algorithms (both shown in Figure 17) indicates that the trajectories appear very similar. The altitude of the interpolated trajectory does not appear to deviate very much (if at all) from the general curve formed by the discrete points of the computed trajectory. A quantitative comparison would show that the trajectories correspond exactly at every existing point in the computed trajectory. Of course this is to be expected given the nature of the interpolation algorithm.

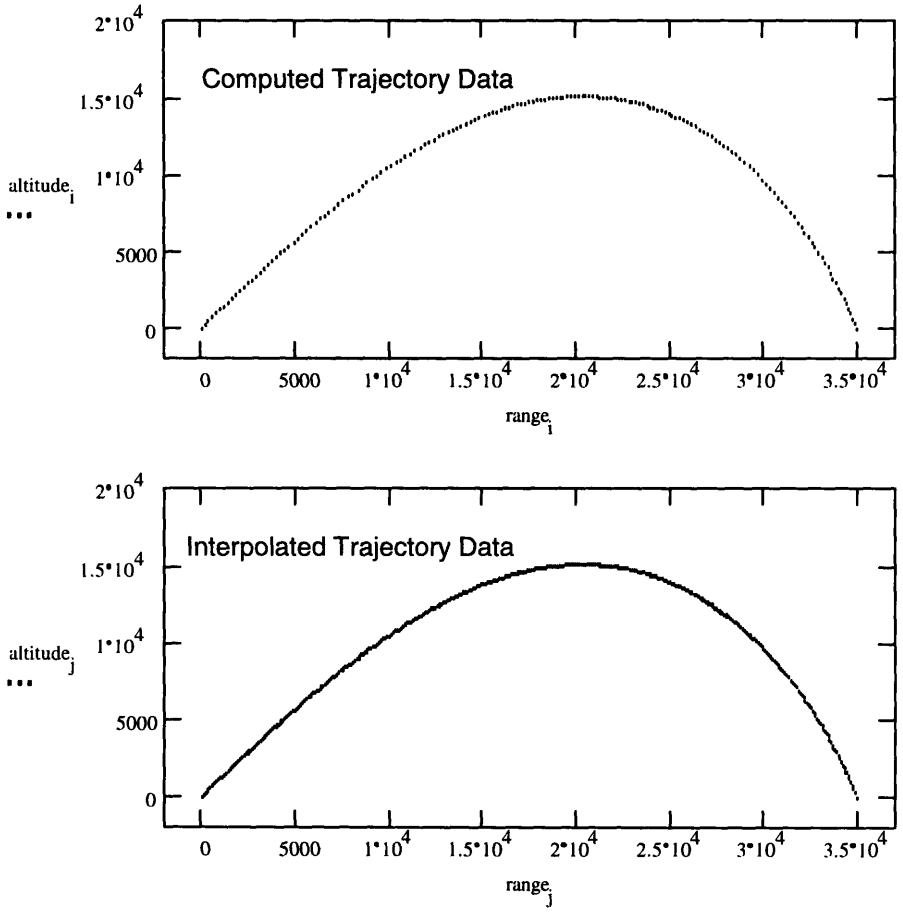


Figure 17. Altitude comparison of actual trajectory to interpolated trajectory

Now we wish to show that a third degree polynomial regression is adequate to describe the nature of the trajectory altitude, and that only minimal gains would be achieved by increasing to a fourth degree polynomial or greater. We show this by examining the large scale characteristics of the trajectory to show that the general nature of the trajectory altitude is essentially third order.

Table 4 shows the results of curve-fitting various degree polynomials to six different trajectories. These trajectories have been chosen in an attempt to cover a wide variety of different trajectory types. The numbers shown in the table are the coefficients of determination for the polynomials after polynomial regression. The table shows that substantial gains can be made by increasing from first degree to second degree polynomials, or from second to third degree polynomials. However, raising the degree of the polynomial above three yields only minimal gains, as a third degree polynomial is very close to an ideal fit to begin with.

	Low Angle Trajectory (Maximum velocity)			High Angle Trajectory (50 degrees elevation)		
Order of polynomial	20 kilometer trajectory	35 kilometer trajectory	50 kilometer trajectory	20 kilometer trajectory	35 kilometer trajectory	50 kilometer trajectory
1	0.011074	0.035467	0.018401	0.033796	0.022891	0.011370
2	0.953225	0.910638	0.949847	0.988095	0.972333	0.984660
3	0.997800	0.992872	0.997110	0.999848	0.998738	0.999339
4	0.999909	0.999353	0.999399	0.999849	0.999536	0.999658
5	0.999996	0.999933	0.999851	0.999976	0.999885	0.999929
6	1.000000	0.999995	0.999967	0.999999	0.999972	0.999976
7	1.000000	1.000000	0.999991	0.999999	0.999992	0.999992
8	1.000000	1.000000	0.999998	1.000000	0.999998	0.999998

Table 4. Results of curve-fitting various degree polynomials to the altitude component of trajectories. (numbers shown are the coefficients of determination)

This finding gives strong evidence that the altitude of the projectile generally follows a third degree polynomial curve. Therefore it seems reasonable to model the altitude of the projectile over smaller portions of the trajectory using this same type of curve.

Having shown that the interpolation of the altitude component of the trajectory is adequate for our purposes, now we wish to do the same for the crossrange component. Again we start with a cursory examination of the trajectory crossrange computed by the ballistic model versus the trajectory crossrange computed by the interpolation algorithms (both shown in Figure 18). The appearance of the two curves is very similar. Like the altitude component, the interpolated crossrange appears to deviate very little from curve formed by the discrete points of the computed trajectory. The two crossrange curves are identical at the points of the fit trajectories correspond exactly at every existing point in the computed trajectory. Of course this is to be expected given the nature of the interpolation algorithm.

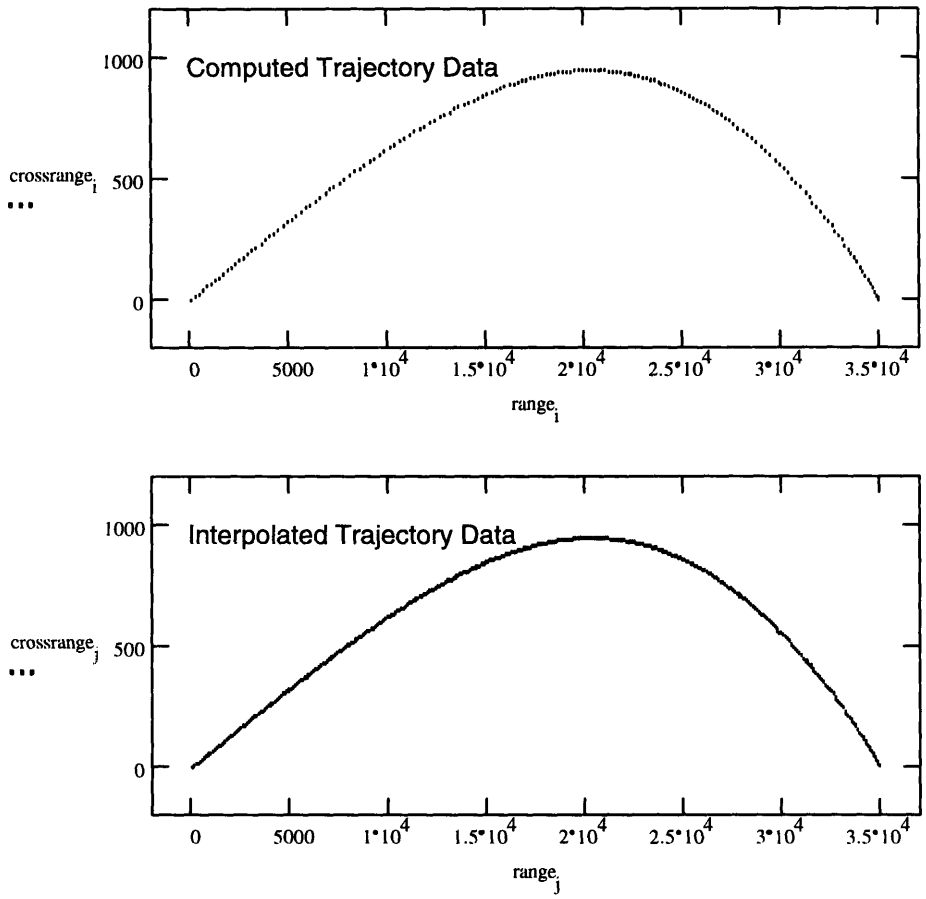


Figure 18. Crossrange comparison of actual trajectory to interpolated trajectory

To show that a third degree polynomial is adequate to describe the nature of the crossrange component of the trajectory we use the same method as for the altitude component. Table 5 shows the results of curve-fitting various degree polynomials to six different trajectories. The same trajectories are used here as were used in Table 4. The values shown in the table (the coefficients of determination for the polynomials after polynomial regression) indicate that substantial gains are made by increasing the order of the polynomial to three, but only minimal gains are realized after that. The third degree polynomial is very close to an ideal fit.

	Low Angle Trajectory (Maximum velocity)			High Angle Trajectory (50 degrees elevation)		
Order of polynomial	20 kilometer trajectory	35 kilometer trajectory	50 kilometer trajectory	20 kilometer trajectory	35 kilometer trajectory	50 kilometer trajectory
1	0.011358	0.035313	0.018324	0.033795	0.022891	0.011370
2	0.953241	0.910615	0.949828	0.988095	0.972333	0.984660
3	0.997801	0.992869	0.997108	0.999848	0.998738	0.999339
4	0.999909	0.999352	0.999398	0.999849	0.999536	0.999658
5	0.999996	0.999933	0.999850	0.999976	0.999885	0.999929
6	1.000000	0.999995	0.999967	0.999999	0.999972	0.999976
7	1.000000	1.000000	0.999991	0.999999	0.999992	0.999992
8	1.000000	1.000000	0.999998	1.000000	0.999998	0.999998

Table 5. Results of curve-fitting various degree polynomials to the crossrange component of trajectories. (numbers shown are the coefficients of determination)

This table strongly indicates that the crossrange component of trajectory generally follows a third degree polynomial curve. Therefore it seems reasonable to model the crossrange of the projectile over smaller portions of the trajectory using this same type of curve.

4.3.3. Comparison Algorithms

The comparison performed by the Truth Model is a very straightforward comparison of two elevation values. The locations of the two elevation values are within 1 meter of each other so no further interpolation is necessary. Thus there is no accuracy loss due to interpolation at this stage of the algorithm. Furthermore, both the projectile location as well as the terrain elevation location are represented in the WGS 84 Geodetic Coordinate System, so there is no accuracy loss due to coordinate conversion.

4.3.4. Results of Test Cases

The tables presented in this section show the intersections against which the other comparison systems will be judged. Results are shown individually for each of the eight test cases.

Because of the high resolution of the terrain used in the Truth Model, it is possible for a greater number of intersection points to occur than was determined when the test cases were originally described. These extraneous intersections correspond to the concept of "anomalies" which was discussed in Section 3.4.1.2. In general, these anomalies are characterized by either the trajectory peeking above the terrain for a brief period and then reentering, or the trajectory briefly dipping into the terrain before resurfacing. A pair of intersections is considered to be an anomaly if the distance between the intersection points is less than 100 meters in the downrange direction, since this corresponds to the correctness criteria against which the comparison systems are judged. While it is desirable for the comparison systems to detect the anomalies, it is not required. Where possible, these anomalies have been identified by shading the lines of the tables which represent anomaly intersection points.

Truth Model Intersection Points for Test Case #1 (20000m)				
Intersection Point #	Intersection Point Type	Downrange (meters)	Crossrange (meters)	Elevation (meters)
1	Entry	0.00	-0.00	302.20
2	Departure	372.00	-2.50	334.51
3	Entry	1344.58	-8.79	416.05
4	Departure	1845.86	-11.89	456.33
5	Entry	18791.00	-15.71	428.11

Table 6. Truth Model Accuracy Results for Test Case #1

Truth Model Intersection Points for Test Case #2 (20000m)				
Intersection Point #	Intersection Point Type	Downrange (meters)	Crossrange (meters)	Elevation (meters)
1	Entry	0.00	0.00	94.00
2	Departure	705.94	-10.60	282.34
3	Entry	19467.11	-17.90	326.93

Table 7. Truth Model Accuracy Results for Test Case #2

Truth Model Intersection Points for Test Case #3 (35000m)				
Intersection Point #	Intersection Point Type	Downrange (meters)	Crossrange (meters)	Elevation (meters)
1	Entry	0.00	0.00	570.00
2	Departure	194.00	-6.10	669.32
3	Entry	34900.98	-8.44	525.81

Table 8. Truth Model Accuracy Results for Test Case #3

Truth Model Intersection Points for Test Case #4 (50000m)				
Intersection Point #	Intersection Point Type	Downrange (meters)	Crossrange (meters)	Elevation (meters)
1	Entry	-0.00	0.00	197.00
2	Departure	173.00	-6.94	299.60
3	Entry	49847.00	-16.57	79.05

Table 9. Truth Model Accuracy Results for Test Case #4

Truth Model Intersection Points for Test Case #5 (20000m)				
Intersection Point #	Intersection Point Type	Downrange (meters)	Crossrange (meters)	Elevation (meters)
1	Entry	0.00	-0.00	2974.00
2	Departure	1290.52	-6.47	3070.87
3	Entry	8342.46	-29.58	3448.63
4	Departure	10958.20	-31.89	3507.12
5	Entry	16072.43	-23.18	3438.79
6	Departure	17437.74	-17.09	3367.62
7	Entry	17831.49	-14.97	3341.99
8	Departure	18455.54	-11.26	3296.30
9	Entry	19565.76	-3.50	3198.49

Table 10. Truth Model Accuracy Results for Test Case #5

Truth Model Intersection Points for Test Case #6 (20000m)				
Intersection Point #	Intersection Point Type	Downrange (meters)	Crossrange (meters)	Elevation (meters)
1	Entry	0.00	0.00	2322.00
2	Departure	730.64	-9.79	2500.14
3	Entry	18761.56	-32.98	2834.41
4	Departure	19363.21	-18.01	2557.95
5	Entry	19373.21	-17.74	2553.03
6	Departure	19386.21	-17.40	2546.61
7	Entry	19393.21	-17.21	2543.15
8	Departure	19409.21	-16.78	2535.22
9	Entry	19425.61	-16.35	2527.06
10	Departure	19432.61	-16.16	2523.57
11	Entry	19830.54	-5.02	2315.99

Table 11. Truth Model Accuracy Results for Test Case #6

Truth Model Intersection Points for Test Case #7 (35000m)				
Intersection Point #	Intersection Point Type	Downrange (meters)	Crossrange (meters)	Elevation (meters)
1	Entry	-0.00	-0.00	2012.32
2	Departure	148.00	-2.52	2045.45
3	Entry	1965.21	-32.80	2443.83
4	Departure	1969.21	-32.87	2444.69
5	Entry	1985.21	-33.13	2448.13
6	Departure	4836.85	-78.04	3038.50
7	Entry	7728.81	-119.94	3586.73
8	Departure	8117.19	-125.25	3655.92
9	Entry	32788.59	-92.40	2782.83

Table 12. Truth Model Accuracy Results for Test Case #7

Truth Model Intersection Points for Test Case #8 (50000m)				
Intersection Point #	Intersection Point Type	Downrange (meters)	Crossrange (meters)	Elevation (meters)
1	Entry	0.00	-0.00	743.00
2	Departure	126.00	-6.04	841.37
3	Entry	49907.84	-10.32	32.37

Table 13. Truth Model Accuracy Results for Test Case #8

Although the Truth Model is not a candidate for finding the optimal comparison system, the timing and memory results for each of the test cases are shown here in Table 14 order to provide a context with which to judge the performance of the following comparison systems.

It should be noted, when examining the memory usage of the Truth Model, that only four "squares" of fractally interpolated DTED were kept in memory at any one time. (Where a "square" of DTED is the square area defined by four adjacent DTED posts, which form the four corners of the interpolation area). This memory management is made possible by the fact that the trajectory is checked as it is traversed in the downrange direction. Since the trajectory is a function of its downrange component, it can never double back on range which it has already traversed. Therefore any DTED squares which have been exited will, in general, ever be re-entered. There is a possibility that a square will be re-entered due to the curvature of the trajectory in the crossrange direction. This is why four blocks are maintained instead of only one.

Test Case #	Time (in sec.)	Memory (in bytes)
1	74.15	194306
2	78.05	171932
3	150.05	276184
4	218.34	365520
5	60.86	164522
6	66.24	164538
7	150.49	305936
8	213.00	350604

Table 14. Truth Model Efficiency Results

5. POINT TO POINT COMPARISON SYSTEM

5.1. Basic Description

The Point To Point Comparison System is the most basic of the candidate comparison systems. The basic premise is very similar to that used by the Truth Model. The location of the projectile is determined at some discrete number of points along its path, and then each of these points is compared to the elevation of the terrain, interpolated from the available DTED. If the elevation of the projectile is found to be below the elevation of the terrain at a given location, then the projectile is said to be in violation at that particular location. An entry intersection point is then determined to be the point in violation with the least downrange value for every set of adjacent violation points. A departure intersection point is likewise determined to be the point in violation with the greatest downrange value for every set of adjacent violation points.

Like the Truth Model, the Point To Point Comparison System requires the path to be defined at a resolution greater than that made available by the ballistic model. The Path Processor is therefore responsible for performing some form of interpolation to determine additional projectile locations along the trajectory.

Likewise, the Point To Point Comparison System requires the terrain elevation to be defined at every discrete location interpolated by the Path Processor. Therefore the DTED Processor is responsible for interpolating the terrain elevation using the surrounding elevation contained in the existing DTED.

5.2. Point To Point System Components

5.2.1. DTED Request Algorithms

The DTED Request Algorithms for the Point To Point Comparison System simply request the rectangular swatch which was described in Section 3.2.1.

5.2.2. DTED Controller

The DTED Controller for the Point To Point Comparison System simply makes available the DTED in 180 arcsecond by 180 arcsecond blocks as described in Section 3.2.2.

5.2.3. DTED Processor

The DTED Processor is responsible for performing a simple linear interpolation to determine the terrain elevation at the location of the trajectory points which are calculated by the Path Processor. To perform the linear interpolation, the four existing DTED posts which immediately surround the requested location are found. These four points are used to perform the interpolation.

5.2.4. Path Processor

The Path Processor is responsible for interpolating the trajectory down do a resolution of 1 meter. It does this by performing a third order polynomial interpolation in both the crossrange and the altitude directions. This is the same interpolation technique which was used for the Truth Model. See Section 4.2.4 for a more detailed description.

5.2.5. Comparison Algorithms

The Comparison Algorithms are very similar to those used by the Truth Model. They involve only a very straightforward comparison of two elevation values. A projectile location is supplied by the Path Processor, and an interpolated terrain elevation for that horizontal projectile location is supplied by the DTED processor. The elevations are simple compared to each other to determine if an intersection exists.

The comparison begins at the launch site and progresses systematically toward the impact, so it is not difficult to distinguish between entry intersection points as opposed to departure intersection points.

5.3. Analysis of Point to Point System

5.3.1. Terrain Interpolation

The terrain interpolation performed by the Point To Point Comparison System is a simple linear interpolation. The accuracy of this method is difficult to analyze since the actual terrain (in this case the terrain produced by the Truth Model) is inherently random.

Some special cases may employ techniques which are more or less conservative than a linear interpolation. If the trajectory under examination represents the projected path of a passenger airplane which was attempting to avoid impacting a mountainside then it may be reasonable to interpolate the terrain elevation at a location by taking the maximum of the four surrounding DTED posts. Another application may dictate that the minimum elevation be used. However, for the purposes of this project in which neither a liberal nor conservative approach is desired, it may be assumed that the best possible method is to use a linear interpolation.

5.3.2. Path Interpolation

The path interpolation technique used by the Point To Point Comparison System is the same technique employed by the Truth Model. This method consists of third degree polynomials in both the crossrange and altitude directions. It was shown during the analysis of the Truth Model in Section 4.3.2 that this interpolation technique is quite adequate to model the flight of a ballistic projectile. Furthermore, because this is the exact same interpolation technique used in the Truth Model, it is obvious that this interpolation can introduce no new errors in accuracy.

5.3.3. Comparison Algorithms

The comparison performed by the Point To Point Comparison System involves only a very straightforward comparison of two elevation values. The elevations supplied by the Path and DTED Processors are for the same horizontal location, so no accuracy loss due to interpolation occurs at this stage of the algorithm. Furthermore, both the projectile location as well as the terrain elevation location are represented in the WGS 84 Geodetic Coordinate System, so there is no accuracy loss due to coordinate conversion.

5.3.4. Results of Test Cases

The accuracy results of the Point To Point Comparison System for each of the eight test cases are shown in the tables below. The horizontal and vertical error in locating each intersection point is also given.

The results of the Point To Point System are "correct" for each of the eight test cases. "Correct" in this sense indicates that the system was able to detect all of the necessary intersection points to within 100 meters, without detecting any erroneous intersections which were not present in the Truth Model results. It did however fail to detect any of the anomalies located by the Truth Model.

The accuracy of the Point To Point System is fairly good. The maximum horizontal error for any test case is 31.00 meters. The maximum error in the vertical location is 5.46 meters.

Point To Point Accuracy Results for Test Case #1 (20000m)									
		Truth Model Results			Point To Point Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	-0.00	302.20	0.00	-0.00	302.20	0.00	0.00
2	Departure	372.00	-2.50	334.51	371.00	-2.49	334.43	1.00	-0.09
3	Entry	1344.58	-8.79	416.05	1344.58	-8.79	416.05	0.00	0.00
4	Departure	1845.86	-11.89	456.33	1849.86	-11.92	456.65	4.00	0.32
5	Entry	18791.00	-15.71	428.11	18796.00	-15.65	427.30	5.00	-0.81

Table 15. Accuracy Results of Point To Point System for Test Case #1 (all values in meters)

Point To Point Accuracy Results for Test Case #2 (20000m)									
		Truth Model Results			Point To Point Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	94.00	0.00	0.00	94.00	0.00	0.00
2	Departure	705.94	-10.60	282.34	701.94	-10.54	281.28	4.00	-1.05
3	Entry	19467.11	-17.90	326.93	19464.11	-18.00	328.81	3.00	1.88

Table 16. Accuracy Results of Point To Point System for Test Case #2 (all values in meters)

Point To Point Accuracy Results for Test Case #3 (35000m)									
		Truth Model Results			Point To Point Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	570.00	0.00	0.00	570.00	0.00	0.00
2	Departure	194.00	-6.10	669.32	194.00	-6.10	669.32	0.00	0.00
3	Entry	34900.98	-8.44	525.81	34900.98	-8.44	525.81	0.00	0.00

Table 17. Accuracy Results of Point To Point System for Test Case #3 (all values in meters)

Point To Point Accuracy Results for Test Case #4 (50000m)									
		Truth Model Results			Point To Point Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	197.00	0.00	0.00	197.00	0.00	0.00
2	Departure	173.00	-6.94	299.60	174.00	-6.98	300.20	1.00	0.59
3	Entry	49847.00	-16.57	79.05	49848.00	-16.46	77.23	1.01	-1.82

Table 18. Accuracy Results of Point To Point System for Test Case #4 (all values in meters)

Point To Point Accuracy Results for Test Case #5 (20000m)									
		Truth Model Results			Point To Point Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	2974.00	0.00	0.00	2974.00	0.00	0.00
2	Departure	1290.52	-6.47	3070.87	1301.52	-6.52	3071.66	11.00	0.79
3	Entry	8342.46	-29.58	3448.63	8311.46	-29.53	3447.63	31.00	-1.00
4	Departure	10958.20	-31.89	3507.12	10962.20	-31.89	3507.16	4.00	0.05
5	Entry	16072.43	-23.18	3438.79	16060.43	-23.22	3439.30	12.00	0.51
6	Departure	17437.74	-17.09	3367.62	17430.74	-17.12	3368.05	7.00	0.43
7	Entry	17831.49	-14.97	3341.99	17833.49	-14.96	3341.85	2.00	-0.14
8	Departure	18455.54	-11.26	3296.30	18463.54	-11.21	3295.68	8.00	-0.63
9	Entry	19565.76	-3.50	3198.49	19577.76	-3.41	3197.31	12.00	-1.18

Table 19. Accuracy Results of Point To Point System for Test Case #5 (all values in meters)

Point To Point Accuracy Results for Test Case #6 (20000m)									
		Truth Model Results			Point To Point Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	2322.00	0.00	0.00	2322.00	0.00	0.00
2	Departure	730.64	-9.79	2500.14	716.64	-9.60	2496.78	14.00	-3.36
3	Entry	18761.56	-32.98	2834.41	18763.56	-32.93	2833.55	2.00	-0.86
	Departure	19363.21	-18.01	2557.95					
	Entry	19373.21	-17.74	2553.03					
	Departure	19386.21	-17.40	2546.61					
	Entry	19393.21	-17.21	2543.15					
4	Departure	19409.21	-16.78	2535.22	19398.21	-17.08	2540.68	11.00	5.46
	Entry	19425.61	-16.35	2527.06					
	Departure	19432.61	-16.16	2523.57					
5	Entry	19830.54	-5.02	2315.99	19832.46	-4.97	2314.95	1.92	-1.04

Table 20. Accuracy Results of Point To Point System for Test Case #6 (all values in meters)

Point To Point Accuracy Results for Test Case #7 (35000m)									
		Truth Model Results			Point To Point Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	2012.32	0.00	0.00	2012.32	0.00	0.00
2	Departure	148.00	-2.52	2045.45	152.00	-2.59	2046.34	4.00	0.89
	Entry	1965.21	-32.80	2443.83					
	Departure	1969.21	-32.87	2444.69					
3	Entry	1985.21	-33.13	2448.13	1996.21	-33.31	2450.49	11.00	2.36
4	Departure	4836.85	-78.04	3038.50	4837.85	-78.06	3038.69	1.00	0.20
5	Entry	7728.81	-119.94	3586.73	7721.81	-119.84	3585.47	7.00	-1.26
6	Departure	8117.19	-125.25	3655.92	8121.19	-125.31	3656.63	4.00	0.71
7	Entry	32788.59	-92.40	2782.83	32788.59	-92.40	2782.83	0.00	0.00

Table 21. Accuracy Results of Point To Point System for Test Case #7 (all values in meters)

Point To Point Accuracy Results for Test Case #8 (50000m)									
		Truth Model Results			Point To Point Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	743.00	0.00	0.00	743.00	0.00	0.00
2	Departure	126.00	-6.04	841.37	126.00	-6.04	841.37	0.00	0.00
3	Entry	49907.84	-10.32	32.37	49907.84	-10.32	32.37	0.00	0.00

Table 22. Accuracy Results of Point To Point System for Test Case #8 (all values in meters)

The time and memory usage for each of the eight test cases is shown in Table 23. The time values shown represent an average taken over 20 runs.

Test Case #	Time (in sec.)	Memory (in bytes)
1	19.23	193784
2	19.41	171420
3	33.83	275672
4	48.71	365008
5	20.61	172856
6	19.68	164026
7	33.81	305424
8	47.58	350092

Table 23. Point To Point System Efficiency Results

These results indicate that the time taken by the Point To Point System is fairly long, probably to the point of this comparison system being unacceptable in an actual application. Memory usage on the other hand is well within the target range.

However, it does appear that both time and memory expenditures are proportional to the length of the trajectory. This is an indication that the algorithm will not scale well to longer or more complex trajectories. This point will be further considered in Section 8, Discussion and Conclusions.

6. CRANE COMPARISON SYSTEM

6.1. Basic Description

The Crane Comparison System is based on an algorithm which was first described by Matthew Crane in 1993 but never published. The general premise of this algorithm is based on the fact that DTED is laid out in a (somewhat) regular longitude / latitude grid pattern. To check a portion of the trajectory, we first determine whether it follows a primarily longitudinal or latitudinal course. If its path is mainly in a longitudinal direction, then we check the trajectory at each point that it crosses a line of longitude which contains existing DTED elevation posts. The obvious counterpart to this algorithm is used for a section of the trajectory which lies in a primarily latitudinal direction.

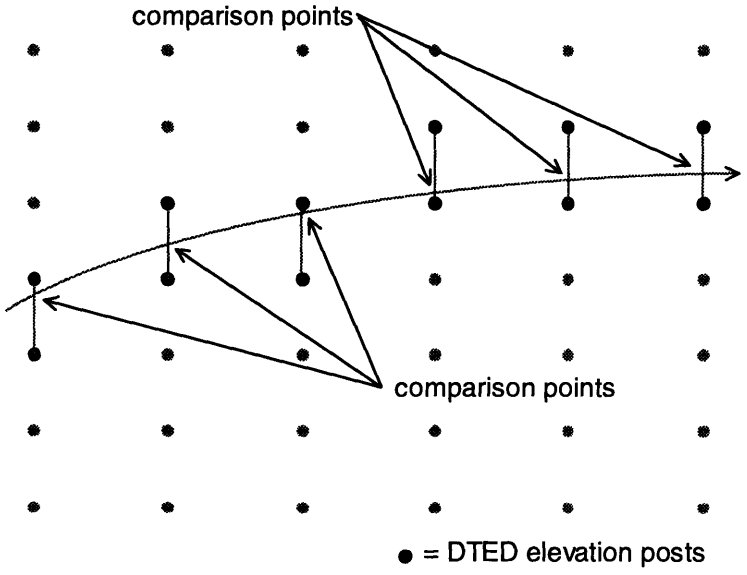


Figure 19. Example Comparison Points in Crane Comparison System

The point at which the trajectory intersects the row or column of DTED elevation posts will most likely not coincide directly with the location of a post. Therefore it is necessary for the DTED Processor to perform some sort of interpolation to determine the elevation at a point which lies between existing elevation posts. This interpolation is usually based solely on the two posts between which the trajectory passes.

Likewise, the point at which the trajectory intersects the row or column of DTED elevation posts will most likely not coincide directly with the location of one of the given trajectory point locations. Therefore it is necessary for the Path Processor to perform some sort of interpolation to determine the altitude of the projectile at a point which lies between the existing trajectory points.

This comparison method requires the determination of whether the trajectory is traveling in a more latitudinal or longitudinal direction. As the trajectory is not a straight line, but is instead a curve, this determination may not be very straightforward. One possibility is to break the trajectory down into smaller portions, and then determine the direction of each portion as it is checked individually. Another possibility is to partition the trajectory along directional lines. With either of these methods there is still some question as to determining the general direction in which a curved path is traveling. Another possible solution, and the one used here, is to perform a linear instead of a polynomial interpolation of the trajectory. This allows the primary direction of the trajectory to be determined with ease. A linear interpolation does introduce some amount of inaccuracy, and this problem will be discussed during the analysis of this comparison system.

The coordinate system used by this algorithm is an artificially created coordinate system which will here be referred to as the longitude / latitude coordinate system. This coordinate system is created by using, without any form of conversion, the longitude and latitude coordinates of the WGS 84 Geodetic Coordinate System (which is a spherical coordinate system) as though they were coordinates of a three-dimensional Cartesian coordinate system. The approximation is made possible by the fact that the discontinuities in the geodetic coordinate system which exist at the north and south poles are disallowed because this project limits trajectories to lie between 80°S and 84°N latitude. Near the equator the geodetic dimensions are a fairly good approximation of a Cartesian coordinate system. But the approximation becomes steadily worse towards to the poles. The accuracy problems which this approximation entails will be addressed in the analysis sections below.

The use of this coordinate system is not an absolute necessity, but the alternatives are not computationally feasible. Working in the Trajectory Coordinate System works fine when dealing with the path data, but converting all the necessary DTED into that coordinate system would require vast amounts of time. Working in the WGS 84 Geodetic Coordinate System would require extensive use of spherical trigonometric functions which are prohibitively expensive computationally.

6.2. Crane System Components

6.2.1. DTED Request Algorithms

The DTED Request Algorithms for the Crane System simply request the rectangular swatch which was described in Section 3.2.1.

6.2.2. DTED Controller

The DTED Controller for the Crane System makes available the DTED in 180 arcsecond by 180 arcsecond blocks as described in Section 3.2.2. In addition, as the DTED is copied from the original DTED files into the block data structures, each block is tagged with the minimum and maximum terrain altitudes which are contained within that specific block.

6.2.3. DTED Processor

In the Crane Comparison System, the Path Processor will identify the locations of all those points along the trajectory which lie directly between two existing DTED posts which are oriented perpendicular to the primary direction of the trajectory. The DTED Processor is then merely responsible for determining the two DTED posts which straddle the trajectory at each of these locations, and then using their elevations to perform a simple linear interpolation to find the most likely elevation of the terrain at the given trajectory location.

6.2.4. Path Processor

The Path Processor in this comparison system is responsible for computing a linear interpolation for every two adjacent points in the trajectory. A linear interpolation is used instead of a second or third degree polynomial interpolation for two reasons. First, to ease the determination of whether the trajectory section lies in a primarily longitudinal or latitudinal direction. Second, to ease the numerous intersection calculations which must be performed by the Comparison Algorithms. Recall that the coordinate system used by this system is based on the geodetic coordinate system. The locations of the trajectory points are represented using longitude and latitude coordinates. It is very difficult for the trajectory to be converted to a polynomial equation in this coordinate system because there is not necessarily a single unique longitude value for any given latitude, and vice versa. And of course there exist two sets of geodetic coordinates for any altitude.

It is possible to represent the path in the Trajectory Coordinate System, where both altitude and crossrange are well-defined functions of downrange, and then only convert locations to the geodetic coordinate system when an intersection calculation is necessary. However, this approach becomes quite costly due to the large number of intersection calculations which must be performed (at least one for each line of DTED posts through which the trajectory passes).

Therefore is becomes easiest to represent the path as line segments between adjacent trajectory points. The accuracy degradation due to this approximation will be addressed in the analysis of this system. There is also a problem with representing straight lines in the longitude / latitude coordinate system. As stated before, this is a spherical coordinate system will be treated as if it were a three-dimensional Cartesian coordinate system. It is a simple matter, given the endpoints of the trajectory segment, to compute the necessary linear interpolation equations of the form [y = mx + b]. However, this mis-treatment of a spherical coordinate system also introduces errors when we try to represent a straight line using linear equation. These also will be examined in the analysis of this system.

6.2.5. Comparison Algorithms

Given the amount of work done by the DTED and Path Processors in the Crane Comparison System, the Comparison Algorithms have a relatively simple job. Given a projectile location and the interpolated elevation of the terrain at that location, all that is left to do is to compare the two elevation values to determine if an intersection exists.

The comparison begins at the launch point and progresses towards the impact, so there is no difficulty in distinguishing entry intersection points as opposed to departure intersection points.

6.3. Analysis of Crane System

6.3.1. Terrain Interpolation

The terrain interpolation performed by the Crane Comparison System is a simple linear interpolation across two adjacent DTED posts. The analysis of a linear interpolation of terrain was already discussed in the context of the Point to Point Comparison System in Section 5.3.1 and will not be repeated here.

6.3.2. Path Interpolation

The interpolation method employed by the Path Processor of the Crane Comparison System does introduce errors into the accuracy of the results. Recall that the interpolation performed is linear as opposed to polynomial. Recall also that the geodetic coordinates of the trajectory were used without conversion as the coordinates in a three dimensional Cartesian Coordinate System. These two sources of error are discussed independently below.

6.3.2.1. Linear vs. Polynomial Interpolation

We have shown in the preceding sections that the shape of the trajectory is inherently polynomial. A linear interpolation should therefore raise accuracy concerns. However, it is very difficult to quantify just how much error may be introduced by a linear interpolation. The reason for this is that, given the wide range of initial conditions which the ballistic model used is capable of handling, there exist a seemingly infinite number of trajectories. It is a nearly impossible task to exhaustively test every possible shape and size of trajectory in order to determine which yields the greatest errors when a linear interpolation is performed. Furthermore, it is not within the scope of this project to explore the intricate workings of the ballistic model itself to determine which of the initial parameters contribute most towards interpolation error. Thus it is not possible to significantly narrow the search parameters. It would seem most likely that a short, slow, high trajectory in strong winds would yield the greatest errors when submitted to a linear interpolation, but this is sheer intuition, and very difficult to verify analytically.

For these reasons, a theoretical analysis will not be performed. Instead, we simply explore the interpolation errors as they affect the eight test cases, as these were chosen as representative of the types of trajectories which this system may expect to encounter. For each test case, the difference between a linear interpolation and a third order polynomial interpolation is determined for every meter in the downrange direction. The maximum difference found for each test case trajectory is presented in Table 24.

Both interpolations are performed using only the Trajectory Coordinate System. The Trajectory Coordinate System is a three dimensional Cartesian coordinate system, thus there is no error introduced due to mis-use of a spherical coordinate system (the error which will be discussed in the following section).

Test Case #	Max. Error (in meters)
1	0.2428
2	0.7396
3	1.2142
4	1.2535
5	0.3558
6	0.6446
7	0.7247
8	1.2266

Table 24. Maximum Error per Test Case due to Linear Interpolation

As can be seen in Table 24, the error due to linear as opposed to polynomial interpolation remains relatively small (less than 2 meters) throughout all of the test cases. This finding indicates that linear interpolation is acceptable as a means of trajectory interpolation without serious detriment to the accuracy of the comparison system.

6.3.2.2. Spherical vs. Cartesian Coordinate System

When the linear interpolation of the trajectory is performed, the assumption is that the projectile will follow a path which is approximately a straight line in three dimensional space between any two adjacent trajectory points. This assumption alone introduces error which is discussed in the previous section. However, in the case of the Crane Comparison System, the endpoints of the line segment are represented in a spherical coordinate system, and the interpolation is performed using the dimensions of the spherical coordinate system as though they were coordinates in a right-angle three dimensional Cartesian coordinate system. In most terrestrial applications, this is not a bad approximation, as the dimensions of the spherical coordinate system, at the radii being considered, do resemble a Cartesian coordinate system. However, in order to properly determine accuracy, the errors introduced by the approximation must be measured.

For simplicity, we decompose the overall error into four independent, measurable sources of error: the convergence of lines of longitude at the poles, the curvature of lines of latitude, the convergence of lines of elevation at the earth's center, and the curvature of constant elevations. In the consideration of each of these error sources, a maximum distance over which a linear interpolation is attempted will be 1000 meters. For comparison, the longest trajectory step in any of the test cases is 819 meters.

The convergence of the lines of longitude at the north and south poles causes a linear interpolation to curve. This curve exists in line segments which have both a longitude and latitude component and is greatest for line segments whose endpoints are oriented with a slope of 1 in longitude / latitude space. The direction of the curve is towards the equator. The curve is most pronounced near the poles, since the convergence of the lines of longitude is greatest at these locations. Since this project only considers trajectories which are located between 80°S and 84°N latitude, we examine a 1000 meter trajectory segment with a slope of 1 at 84°N latitude. The maximum difference between the interpolation and the desired straight line segment is determined experimentally to be 0.075 meters.

The curvature of the lines of latitude is caused by the fact that the lines of latitude do not circumnavigate the globe. Thus the line of latitude which exists between any two locations with equal latitude components does not represent the shortest distance between those locations over the surface of the spheroid. The curvature of the lines of latitude causes the linear interpolation of a trajectory to curve. The curve exists for any line segments which have a longitude component and is greatest for line segments whose endpoints lie at the same latitude. The direction of the curve is towards the equator. The curve is most pronounced near the poles, since the lines of latitude curve greatest near the poles. We therefore examine a 1000 meter trajectory segment which lies directly east/west at 84°N latitude. The maximum difference between the interpolation and the desired straight line segment is found to be 0.195 meters.

The convergence of lines of elevation at the earth's center causes a linear interpolation to curve. The curve exists for any line segments which have both a horizontal and a vertical components and is greatest for line segments whose endpoints are oriented with a slope of 1 in horizontal / elevation space. The direction of the curve is in the positive vertical direction. The curve is most pronounced at low elevations near the poles, since the radius of the earth is smallest at these locations. The lowest terrestrial elevation on the earth which is not covered by water is the Dead Sea at 392 meters below sea level. We therefore examine a 1000 meter trajectory segment which lies directly north/south at 84°N and which starts at 392 meters below sea level and extends horizontally and vertically with a slope of 1. The maximum difference between the interpolation and the desired straight line segment is determined experimentally to be 0.007 meters.

The curvature of constant elevations causes a linear interpolation to curve. For example, mean sea level has a constant elevation of 0, and this surface is obviously curved around the shape of the spheroid. A linear interpolation between two locations with equal elevation components will follow this same curve. The curve exists for any line segments which have a horizontal component (in either the longitude or latitude directions) and is greatest for line segments which have no vertical component (whose endpoints have identical elevation values). The direction of the curve is in the positive vertical direction. The curve is most pronounced at low elevations near the poles, since the radius of the earth is smallest at these locations. We therefore examine a 1000 meter trajectory segment which lies approximately east/west at 84°N at 392 meters below sea level. The maximum difference between the interpolation and the desired straight line segment is found to be 0.020 meters.

Altogether the maximum error from these four error sources totals only 0.297 meters. This is a small enough error that the approximation of substituting a Cartesian coordinate system for the actual geodetic coordinate system may be considered an acceptable method of performing a linear interpolation on trajectory data.

6.3.3. Comparison Algorithms

The primary difficulty with the Comparison Algorithms performed by the Crane Comparison System is the low resolution at which actual comparisons are made. The distance between comparisons is only as small as the distance between lines of posts. This distance varies with latitude and the level of DTED used. For level 1 DTED (which is the level used by this project) the minimum distance between comparisons is 58.374 meters which corresponds to 18 arcseconds in the longitudinal direction at 84°N latitude. The maximum distance grows to 168.989 meters. This distance occurs on a diagonal trajectory segment at 50° latitude (north or south). The distances for DTED level 2 are smaller of course, due to the higher resolution of the data. The minimum distance between comparisons is 19.458 meters which corresponds to 6 arcseconds in the longitudinal direction at occurs at 84°N latitude. The maximum distance is 56.33 meters. This distance occurs on a diagonal trajectory segment at 50° latitude (north or south).

The distance between comparisons for level 2 is not extreme. As the distance remains always below 100 meters, we are assured that no true intersections are missed entirely.

The distances for level 1 however, seem almost unacceptably large. Intersections which last for 160 meters may not even be detected. Furthermore, once an intersection is detected, no attempt is made to determine its exact location. Instead, the intersection is assumed to occur at the point where it was first detected. This means that the reported location of an intersection could be in error by as much as 169 meters in the horizontal direction. This in turn has consequences in terms of vertical error. The vertical error of an intersection point may be as large as the vertical distance which the terrain elevation can change over 169 meters of horizontal distance. Since portions of the earth's terrain may involve very rough or steep terrain, the vertical error may be even larger than 169 meters, and thus quite substantial.

The accuracy of these Comparison Algorithms is very low. This observation should be carefully considered in the overall evaluation of the Crane Comparison System.

6.3.4. Results of Test Cases

The accuracy results of the Crane Comparison System for each of the eight test cases are shown in the tables below. The horizontal and vertical error in locating each intersection point is also given.

The result of the Crane System are "correct" for each of the eight test cases. "Correct" in this sense indicates that the system was able to detect all of the necessary intersection points to within 100 meters, without detecting any erroneous intersections which were not present in the Truth Model results. It did however fail to detect any of the anomalies located by the Truth Model.

The accuracy of the Crane System is not very good. In fact it came very close to the 100 meter horizontal error limit. The maximum horizontal error for any test case is 97.88 meters. The maximum error in the vertical location is 116.34 meters.

Crane System Accuracy Results for Test Case #1 (20000m)									
		Truth Model Results			Crane System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	302.20	0.00	0.00	302.20	0.00	0.00
2	Departure	372.00	-2.50	334.51	383.98	-2.57	335.38	11.98	0.87
3	Entry	1344.58	-8.79	416.05	1343.89	-8.78	415.84	0.68	-0.21
4	Departure	1845.86	-11.89	456.33	1919.81	-12.33	462.01	73.96	5.68
5	Entry	18791.00	-15.71	428.11	18883.22	-14.66	412.96	92.22	-15.15

Table 25. Accuracy Results of Crane System for Test Case #1 (all values in meters)

Crane System Accuracy Results for Test Case #2 (20000m)									
		Truth Model Results			Crane System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	94.00	0.00	0.00	94.00	0.00	0.00
2	Departure	705.94	-10.60	282.34	713.91	-10.71	284.35	7.97	2.01
3	Entry	19467.11	-17.90	326.93	19513.95	-16.37	296.95	46.86	-29.98

Table 26. Accuracy Results of Crane System for Test Case #2 (all values in meters)

Crane System Accuracy Results for Test Case #3 (35000m)									
		Truth Model Results			Crane System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	570.00	0.00	0.00	570.00	0.00	0.00
2	Departure	194.00	-6.10	669.32	223.52	-7.03	684.28	29.54	14.96
3	Entry	34900.98	-8.44	525.81	34926.26	-6.28	485.85	25.37	-39.96

Table 27. Accuracy Results of Crane System for Test Case #3 (all values in meters)

Crane System Accuracy Results for Test Case #4 (50000m)									
		Truth Model Results			Crane System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	197.00	0.00	0.00	197.00	0.00	0.00
2	Departure	173.00	-6.94	299.60	209.30	-8.39	320.95	36.32	21.35
3	Entry	49847.00	-16.57	79.05	49867.14	-14.38	41.70	20.25	-37.35

Table 28. Accuracy Results of Crane System for Test Case #4 (all values in meters)

Crane System Accuracy Results for Test Case #5 (20000m)									
		Truth Model Results			Crane System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	2974.00	0.00	0.00	2974.00	0.00	0.00
2	Departure	1290.52	-6.47	3070.87	1388.39	-6.91	3077.64	97.88	6.77
3	Entry	8342.46	-29.58	3448.63	8330.88	-29.54	3448.01	11.58	-0.63
4	Departure	10958.20	-31.89	3507.12	11015.40	-31.88	3507.52	57.20	0.41
5	Entry	16072.43	-23.18	3438.79	16106.46	-23.04	3437.23	34.02	-1.56
6	Departure	17437.74	-17.09	3367.62	17494.77	-16.77	3363.78	57.03	-3.83
7	Entry	17831.49	-14.97	3341.99	17864.96	-14.77	3339.43	33.48	-2.56
8	Departure	18455.54	-11.26	3296.30	18512.79	-10.89	3291.66	57.25	-4.65
9	Entry	19565.76	-3.50	3198.49	19623.28	-3.04	3192.60	57.52	-5.88

Table 29. Accuracy Results of Crane System for Test Case #5 (all values in meters)

Crane System Accuracy Results for Test Case #6 (20000m)									
		Truth Model Results			Crane System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	2322.00	0.00	0.00	2322.00	0.00	0.00
2	Departure	730.64	-9.79	2500.14	744.16	-9.97	2503.37	13.52	3.23
3	Entry	18761.56	-32.98	2834.41	18772.35	-32.72	2829.60	10.79	-4.81
4	Departure	19432.61	-16.16	2523.57	19420.46	-16.48	2529.52	12.16	5.95
	Departure	19363.21	-18.01	2557.95					
	Entry	19373.21	-17.74	2553.03					
	Departure	19386.21	-17.40	2546.61					
	Entry	19393.21	-17.21	2543.15					
	Departure	19409.21	-16.78	2535.22					
	Entry	19425.61	-16.35	2527.06					
5	Entry	19830.54	-5.02	2315.99	19883.15	-3.47	2286.83	52.62	-29.17

Table 30. Accuracy Results of Crane System for Test Case #6 (all values in meters)

Crane System Accuracy Results for Test Case #7 (35000m)									
		Truth Model Results			Crane System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	2012.32	16.15	0.27	2015.92	16.15	3.60
2	Departure	148.00	-2.52	2045.45	177.65	-3.00	2051.87	29.65	6.42
	Entry	1965.21	-32.80	2443.83					
	Departure	1969.21	-32.87	2444.69					
3	Entry	1985.21	-33.13	2448.13	2035.48	-33.94	2458.89	50.27	10.76
4	Departure	4836.85	-78.04	3038.50	4865.20	-78.45	3043.90	28.35	5.41
5	Entry	7728.81	-119.94	3586.73	7779.16	-120.62	3595.65	50.35	8.93
6	Departure	8117.19	-125.25	3655.92	8184.17	-126.14	3667.51	66.98	11.58
7	Entry	32788.59	-92.40	2782.83	32788.22	-92.38	2782.53	0.36	-0.30

Table 31. Accuracy Results of Crane System for Test Case #7 (all values in meters)

Crane System Accuracy Results for Test Case #8 (50000m)									
		Truth Model Results			Crane System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	743.00	92.08	-4.41	814.80	92.19	71.80
2	Departure	126.00	-6.04	841.37	184.17	-8.81	886.59	58.24	45.23
3	Entry	49907.84	-10.32	32.37	49964.37	-3.99	-83.98	56.88	-116.34

Table 32. Accuracy Results of Crane System for Test Case #8 (all values in meters)

The time and memory usage for each of the eight test cases is shown in Table 33. The time values shown represent an average taken over 20 runs.

Test Case #	Time (in sec.)	Memory (in bytes)
1	0.161	169794
2	0.158	148536
3	0.205	251672
4	0.247	341008
5	0.285	150056
6	0.167	140026
7	0.321	281424
8	0.240	326092

Table 33. Crane System Efficiency Results

These results indicate that the time taken by the Point To Point System is fairly short. Memory usage as well is within the specified range.

Unlike the Point To Point System, the time taken by the Crane System to perform the comparison does not appear to depend directly on the length of the trajectory. In fact it is not readily apparent on exactly what aspect, if any, the time is most related to. However, the memory usage of the Crane System does appear to follow the dependence on trajectory length which was observed in the Point To Point System. This dependence is explained by the fact that the storage of DTED dominates the memory requirements for a comparison system. And that the amount of DTED loaded initially into memory is based on the trajectory length. These characteristics will be further considered in Section 8, Discussion and Conclusions.

7. OCTREE COMPARISON SYSTEM

7.1. Basic Description

The octree is a space subdivision method which has been used extensively for computer graphics applications, usually to optimize ray-tracing algorithms. The original idea appears to have been developed simultaneously by a number of independent researchers. Refer to the following sources for a history and bibliography [SAMET88A] [SAMET88B]. The octree structure used here is based on the description by Foley et. al. [FOLEY90]

The concept of an octree is essentially a quadtree extended to three dimensions. Volumes are recursively subdivided into eight "octants" (see Figure 20) and the contents of the original volume are then distributed to the appropriate octant. Subdivision continues until some stopping criteria is achieved. The usually stopping criteria, and the one used in this model, maintains that subdivision stops when a volume is homogeneous (it contains either earth or air but not both). An additional stopping criteria is that subdivision stops when either the longitudinal or latitudinal size of a volume is the same as the distance between two adjacent posts.

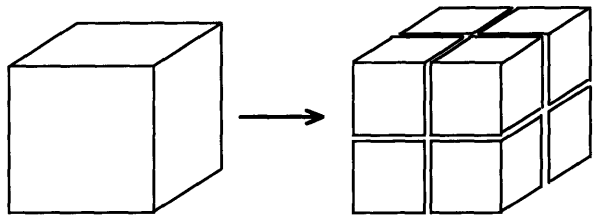


Figure 20. Division of volume into octants

The representation in memory of the octree data structure is very straight-forward. The basic unit of storage is the "node", which represents a volume. Each node has stored within it its location, size, and pointers to the DTED which it contains. Each node is marked as either earth, air, or partial. If the node is partial then it also contains pointers to its eight child nodes. Each node also has a pointer to its parent node to allow upwards traversal. The node at the top of the structure is referred to as the "root" node. This node represents a volume which encompasses the entire problem space. The octree itself is nothing more than a pointer to the root node.

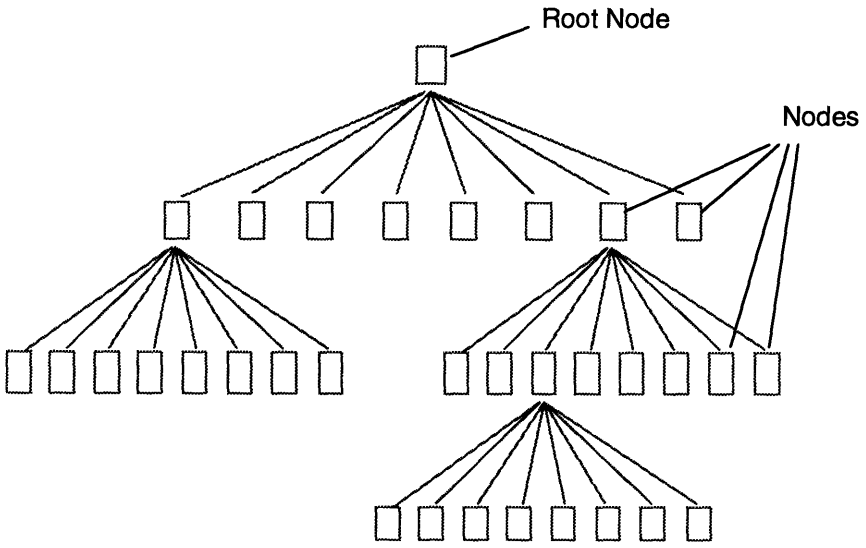


Figure 21. Octree data structure

The sides of the octree are orthogonal to the lines of latitude and longitude. The third dimension of the node volume is parallel to the gravity vector. This causes the actual shape of the octree volumes to be somewhat warped because of the spherical nature of the earth's surface. However, this warping is not critical because it does not cause the boundaries between volumes to overlap, so the volumes are still non-overlapping and well-defined. Because the volumes are still well-defined, and also because the warping is so slight at the distances being considered, the warping is summarily ignored.

The comparison of the trajectory to the DTED contained in the octree begins at the launch location of the trajectory. The trajectory is compared to each node which it passes through on the way to the impact location. Determining intersection points is straightforward. If a node is entered whose status (either all-earth or all-air) is different than the node which was just left, then an intersection has been found. In the case that a node is encountered which is not homogeneous (because of the node-size stopping criteria mentioned above) then a linear interpolation is performed on the terrain in order to determine if and where an intersection occurs.

7.2. Octree System Components

7.2.1. DTED Request Algorithms

The DTED Request Algorithms for the Octree System simply request the rectangular swatch which was described in Section 3.2.1.

7.2.2. DTED Controller

The DTED Controller for the Octree System makes available the DTED in 180 arcsecond by 180 arcsecond blocks as described in Section 3.2.2. In addition, as the DTED is copied from the original DTED files into the block data structures, each block is tagged with the minimum and maximum terrain altitudes which are contained within that specific block.

7.2.3. DTED Processing

The primary task in DTED processing is the construction of the octree. This construction must be reasonably fast and efficient since the time and memory used in the construction itself will be considered in the evaluation of the octree model. To aid in efficient construction, the octree is computed dynamically as the comparison progresses, and only those nodes which are visited are computed. This dramatically reduces the number of nodes which must be generated.

The octree is built using the WGS 84 geodetic coordinate system, which is the same coordinate system in which the DTED is organized. As stated previously, the spherical nature of this coordinate system is ignored and it is treated as if it were a three-dimensional Cartesian coordinate system. The larger volumes (volumes whose ground projection is larger than a single DTED block) have boundaries which coincide with the DTED block boundaries. The smaller volumes (volumes whose ground projection is smaller than a single DTED block) have boundaries which coincide with the rows and columns of DTED posts.

7.2.3.1. Construction of the Root Node

The construction of the root node has two parts: determining the ground projection of the node volume, and determining the vertical dimensions of the node volume.

The ground-projection dimensions of the root node are calculated using the swatch information which was computed by the DTED Request Algorithms. The node is made large enough to encompass all of the DTED blocks in the swatch. The two horizontal lengths of the node (in the longitude and latitude directions) encompass an equal number of DTED blocks, and this number is an even power of 2. This requirement may cause some extra space to be included in the root node, but it greatly aids in the algorithms used to subdivide a node into octants.

The vertical dimensions of the root node extend 1 meter above and 1 meter below the vertical extents of the trajectory. These extents can be found by direct examination of the trajectory data. Unlike the ground projection dimensions of the node, the vertical dimensions have no requirements such as integral values.

7.2.3.2. Subdivision of a Node

If a node is not homogeneous then its volume must be divided into eight (nearly) equal octants. The subdivision of a node into its eight octants is performed in two steps: dividing the ground projection into quadrants, and dividing the vertical dimension. Dividing the vertical dimension is accomplished by simply finding the vertical center of the node. Subdividing the ground projection is a more involved procedure.

Subdivision of the ground projection into quadrants is performed differently depending on whether the node volume encompasses only a single DTED block, or several DTED blocks. If the volume encompasses several DTED blocks then the sides of the node are both simply divided in half. There are guaranteed to be an even number of blocks along both sides because the root node was constructed with sides which measured an even power of 2. If the volume encompasses only a single DTED block, then the number of posts intervals along each side is divided in half. If there are an odd number of post intervals on a side then one of the sections will have one more post interval than the other. This means that at the lowest level of subdivision there may be nodes which measure one post interval by two post intervals. However, this does not present a problem for the comparison algorithms.

7.2.3.3. Computing the Status of a Node

To be useful to the comparison algorithms, a node must be flagged with a status which indicates whether the node volume lies entirely within the earth, entirely above the terrain, or partially above and partially below the terrain. Determining the status of a node is performed differently depending on whether the ground projection of the node volume encompasses only a single DTED block, or several DTED blocks.

If the ground projection of the node encompasses several DTED blocks then each of the blocks contained within the node are checked independently against the vertical dimensions of the node. This is made quite simple be the minimum and maximum elevation tags which were placed in each block by the DTED Controller (see Section 7.2.2 above). If all of the DTED blocks lie entirely beneath the node volume then the node contains only air. If all of the DTED blocks lie entirely above the node volume then the node contains only earth. If any of the blocks overlap the node volume then the node is flagged as partial.

If the ground projection of the node entirely encompasses a single DTED block then the status may be determined as described in the above paragraph.

If the ground projection of a node is smaller than a single DTED block, and thus is contained within a single DTED block, then the status is determined as follows. All of the DTED posts which lie within the ground projection of the node volume are checked independently against the vertical dimensions of the node. If all of the DTED posts lie entirely beneath the node volume then the node contains only air. If all of the DTED posts lie entirely above the node volume then the node contains only earth. If any of the posts lie within the node volume, or if some posts lie above the node while others lie below the node then the node is flagged as partial.

7.2.4. Path Processor

The Path Processor in this system is responsible only for computing a linear interpolation for every two adjacent points in the trajectory. This is the same interpolation technique which was used by the Crane Comparison System. See Section 6.2.4 for further details.

7.2.5. Comparison Algorithms

The basic idea of the Comparison Algorithms is to traverse the trajectory from the launch point to the impact point, and to determine every node which it passes through on the way. A trajectory / terrain comparison check on a single node is straightforward. If the node is homogeneous, then an intersection exists if the node contains a medium different from that which the projectile was in before it entered the node. In this case, the intersection point is defined to be the point at which the trajectory enters the node. If the node is not homogeneous but is at the bottom-most level (so it can no longer be subdivided) then a linear interpolation of the terrain contained by the node is performed to determine the existence and location of any intersection points.

The primary difficulty of the approach described above is determining all of the nodes which the trajectory passes through. This is accomplished by a simple recursive algorithm. The point at which the trajectory starts within a node is well-defined. (As a base case, the starting point of the trajectory can be determined from direct examination of the trajectory data). The equations which define the path of the trajectory within this node have been calculated by the Path Processor. Therefore it is possible to determine the exact point at which the trajectory exits this node. The algorithm is then continued recursively. The exit point is used as the new starting point. The new node is the node which neighbors this node in the direction of the exit point.

The neighbor node is found using the method described by Samet [SAMET89]. This method ascends the tree until a common ancestor between the node and its neighbor is found. Then the tree is descended in a mirror-image of the ascension path. If a homogeneous node is found before the mirror-image descending is completed, then the homogeneous node is used as the next node. If the mirror-image descending results in a node which is neither homogeneous nor a bottom-most partial node, then the tree is further descended until the bottom-most node is found which contains the exit point (the new starting point).

7.3. Analysis of Octree System

7.3.1. Terrain Interpolation

The terrain interpolation technique used by the Octree Comparison System is a simple linear interpolation, the same technique which was used by each of the other comparison systems. A justification for the used of this technique was given during the analysis of the Point To Point Comparison System in Section 5.3.1 and will not be repeated here.

7.3.2. Path Interpolation

The Path interpolation technique used by the Octree Comparison System is the same technique employed by the Crane Comparison System: linear interpolation using the geodetic coordinates as an approximation of a three dimensional Cartesian coordinate system. An analysis of this interpolation method was performed during the analysis of the Crane System in Section 6.3.2 and will not be repeated here.

7.3.3. Comparison Algorithms

The trajectory / terrain comparison performed by the Octree Comparison System is done on a node by node basis. If the node is homogeneous, the comparison is very straight-forward. If the node is not homogeneous, then a more involved comparison must be performed. The elevation of the trajectory is determined for the points where the trajectory enters and exits the node, and a line representing the trajectory within the node is computed using these elevations. Then the elevation of the terrain is determined for those same points and a line representing the terrain within the node is computed. The equations for these two lines are used to determine if and where the trajectory intersects the terrain. This method is essentially a linear interpolation technique involving both the terrain and the trajectory. Linear interpolation of the terrain as well as linear interpolation of the trajectory have already been discussed in the preceding sections, and therefore will not be further explored here.

7.3.4. Results of Test Cases

The accuracy results of the Octree Comparison System for each of the eight test cases are shown in the tables below. The horizontal and vertical error in locating each intersection point is also given.

The result of the Octree System are "correct" for each of the eight test cases. "Correct" in this sense indicates that the system was able to detect all of the necessary intersection points to within 100 meters, without detecting any erroneous intersections which were not present in the Truth Model results. It did, however, fail to detect any of the anomalies located by the Truth Model.

The accuracy of the Octree System is not exceptional, but satisfactory. The maximum horizontal error for any test case is 32.67 meters. The maximum error in the vertical location is 5.24 meters.

Octree System Accuracy Results for Test Case #1 (20000m)									
		Truth Model Results			Octree System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	302.20	0.00	0.00	302.20	0.00	0.00
2	Departure	372.00	-2.50	334.51	372.16	-2.49	334.36	0.16	-0.15
3	Entry	1344.58	-8.79	416.05	1343.65	-8.78	415.82	0.93	-0.23
4	Departure	1845.86	-11.89	456.33	1850.13	-11.91	456.53	4.28	0.20
5	Entry	18791.00	-15.71	428.11	18793.76	-15.66	427.45	2.76	-0.66

Table 34. Accuracy Results of Octree System for Test Case #1 (all values in meters)

Octree System Accuracy Results for Test Case #2 (20000m)									
		Truth Model Results			Octree System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	94.00	0.00	0.00	94.00	0.00	0.00
2	Departure	705.94	-10.60	282.34	701.71	-10.53	281.15	4.23	-1.19
3	Entry	19467.11	-17.90	326.93	19462.26	-18.03	329.27	4.85	2.33

Table 35. Accuracy Results of Octree System for Test Case #2 (all values in meters)

Octree System Accuracy Results for Test Case #3 (35000m)									
		Truth Model Results			Octree System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	570.00	0.00	0.00	570.00	0.00	0.00
2	Departure	194.00	-6.10	669.32	193.64	-6.09	669.00	0.36	-0.32
3	Entry	34900.98	-8.44	525.81	34899.86	-8.53	527.31	1.12	1.51

Table 36. Accuracy Results of Octree System for Test Case #3 (all values in meters)

Octree System Accuracy Results for Test Case #4 (50000m)									
		Truth Model Results			Octree System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	197.00	0.00	0.00	197.00	0.00	0.00
2	Departure	173.00	-6.94	299.60	173.71	-6.96	299.88	0.71	0.27
3	Entry	49847.00	-16.57	79.05	49847.22	-16.54	78.38	0.22	-0.67

Table 37. Accuracy Results of Octree System for Test Case #4 (all values in meters)

Octree System Accuracy Results for Test Case #5 (20000m)									
		Truth Model Results			Octree System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	2974.00	0.00	0.00	2974.00	0.00	0.00
2	Departure	1290.52	-6.47	3070.87	1303.33	-6.50	3071.47	12.81	0.60
3	Entry	8342.46	-29.58	3448.63	8309.79	-29.50	3447.34	32.67	-1.30
4	Departure	10958.20	-31.89	3507.12	10963.55	-31.88	3506.93	5.35	-0.19
5	Entry	16072.43	-23.18	3438.79	16059.49	-23.21	3439.16	12.94	0.37
6	Departure	17437.74	-17.09	3367.62	17431.35	-17.11	3367.80	6.39	0.18
7	Entry	17831.49	-14.97	3341.99	17831.46	-14.96	3341.75	0.03	-0.24
8	Departure	18455.54	-11.26	3296.30	18463.30	-11.21	3295.69	7.76	-0.61
9	Entry	19565.76	-3.50	3198.49	19576.89	-3.41	3197.30	11.13	-1.19

Table 38. Accuracy Results of Octree System for Test Case #5 (all values in meters)

Octree System Accuracy Results for Test Case #6 (20000m)									
		Truth Model Results			Octree System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	2322.00	0.00	0.00	2322.00	0.00	0.00
2	Departure	730.64	-9.79	2500.14	716.05	-9.59	2496.60	14.59	-3.54
3	Entry	18761.56	-32.98	2834.41	18762.08	-32.95	2833.93	0.52	-0.48
	Departure	19363.21	-18.01	2557.95					
	Entry	19373.21	-17.74	2553.03					
	Departure	19386.21	-17.40	2546.61					
	Entry	19393.21	-17.21	2543.15					
4	Departure	19409.21	-16.78	2535.22	19398.39	-17.07	2540.46	10.83	5.24
	Entry	19425.61	-16.35	2527.06					
	Departure	19432.61	-16.16	2523.57					
5	Entry	19830.54	-5.02	2315.99	19831.56	-4.99	2315.44	1.02	-0.56

Table 39. Accuracy Results of Octree System for Test Case #6 (all values in meters)

Octree System Accuracy Results for Test Case #7 (35000m)									
		Truth Model Results			Octree System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	2012.32	0.00	0.00	2012.32	0.00	0.00
2	Departure	148.00	-2.52	2045.45	152.09	-2.57	2046.18	4.09	0.73
	Entry	1965.21	-32.80	2443.83					
	Departure	1969.21	-32.87	2444.69					
3	Entry	1985.21	-33.13	2448.13	1995.49	-33.29	2450.30	10.27	2.17
4	Departure	4836.85	-78.04	3038.50	4838.14	-78.04	3038.51	1.28	0.02
5	Entry	7728.81	-119.94	3586.73	7721.34	-119.82	3585.21	7.48	-1.52
6	Departure	8117.19	-125.25	3655.92	8121.22	-125.29	3656.38	4.03	0.45
7	Entry	32788.59	-92.40	2782.83	32788.01	-92.38	2782.63	0.57	-0.19

Table 40. Accuracy Results of Octree System for Test Case #7 (all values in meters)

Octree System Accuracy Results for Test Case #8 (50000m)									
		Truth Model Results			Octree System Results			Accuracy	
#	Intersect. Type	Down Range	Cross Range	Elev	Down Range	Cross Range	Elev	Horiz Error	Vert. Error
1	Entry	0.00	0.00	743.00	0.00	0.00	743.00	0.00	0.00
2	Departure	126.00	-6.04	841.37	126.14	-6.03	841.35	0.14	-0.02
3	Entry	49907.84	-10.32	32.37	49907.56	-10.34	32.42	0.28	0.05

Table 41. Accuracy Results of Octree System for Test Case #8 (all values in meters)

The time and memory usage for each of the eight test cases is shown in Table 42. Octree System Efficiency Results. The time values shown represent an average taken over 20 runs.

Test Case #	Time (in sec.)	Memory (in bytes)
1	0.375	190170
2	0.358	156536
3	0.328	258676
4	0.354	348676
5	2.733	206354
6	0.933	178150
7	2.108	391864
8	0.192	332700

Table 42. Octree System Efficiency Results

These results indicate that the time taken by the Octree System is fairly short. Like the Crane System and unlike the Point To Point System, the time taken by the Octree System to perform the comparison does not appear to depend directly on the length of the trajectory.

Memory usage, on the other hand, does appear to follow the dependence on trajectory length which was observed in both the Point To Point System and the Crane System. This is to be expected as the storage of DTED dominates the memory requirements of the system, and the amount of DTED initially loaded into memory is based on the length of the trajectory. However, the memory usage does appear to be well within the memory limitation specifications.

The primary task performed by the Octree Comparison System, and the one most likely to have caused efficiency difficulties, is the construction of the octree. Since this tree was constructed dynamically, in the midst of the actual comparison, the construction itself had be sufficiently fast so as not to overwhelm the timing of the overall comparison. It also had to use only a reasonable amount of memory, so as to allow the comparison to remain within the memory limitations of the system.

The results of the test cases show that both the time and memory used in constructing the octree were small enough to allow the Octree Comparison System to remain competitive with the other systems. The tree constructed in most cases was rather sparse which indicates that no extraneous resources were wasted constructing unneeded nodes. Statistics on the octree for each of the eight test cases is shown in the table below.

Test Case #	Octree Characteristics				Octree Memory Usage		
	Tree Depth	Number of Nodes	Max. Possible Nodes	% of Tree Filled	Bytes Used for Tree	Bytes Used Total	% of Memory for Tree
1	10	199	153391689	0.00013	19104	190170	10.046
2	10	87	153391689	0.00006	8352	156536	5.336
3	10	60	153391689	0.00004	5760	258676	2.227
4	11	65	1227133513	0.00001	6240	348676	1.790
5	10	657	153391689	0.00043	63072	206354	30.565
6	10	363	153391689	0.00024	34848	176662	19.726
7	11	1096	1227133513	0.00008	105216	391864	26.850
8	11	54	1227133513	0.00001	5184	332700	1.558

Table 43. Test Results on Octree Construction

8. DISCUSSION AND CONCLUSIONS

8.1. System Comparison

In this section we examine the results of all three of the comparison systems and attempt to determine the superior performer for each of the evaluation criteria. The evaluation criteria which will be discussed are: space limitations, correctness, time usage, accuracy, and space usage.

8.1.1. Space Limitations

The space limitations for the system include a 4 megabyte limit on the memory used to store DTED and other data, as well as a 1 megabyte limit on stack space.

All of the comparison systems stayed well within the data memory limitations. In fact, none of the systems required greater even 1 megabyte of data memory. The maximum used was by the Octree Comparison System with 391,864 bytes. However, it should be remembered that the eight test cases only exercised a single resolution of DTED. Different latitudinal zones, as well as different DTED levels were unavailable for testing.

The small amount of data memory required is due mostly to the 180 arcsecond by 180 arcsecond DTED block organization which was described in Section 3.2.2. This organization allowed the DTED required for even a 50 kilometer trajectory to remain well below the 4 megabyte memory limitations

8.1.2. Correctness

As noted in the analysis which was performed individually for each of the comparison systems, every one of the systems was able to detect all of the necessary intersection points within 100 meters without erroneously detecting non-existent intersection points.

While the results from the test case runs would seem to indicate that all of the comparison systems are essentially "correct", the analysis performed on the Crane System indicates that it is possible for the Crane System to completely miss intersections which last longer than the necessary 100 meters. Whether this error would only occur in an extremely unlikely case is unknown. However, the dangerously poor accuracy which the Crane System experienced in locating the intersection points would indicate that the possibility of major errors occurring is rather likely.

8.1.3. Timing

The time efficiency of the three comparison systems is shown below in Table 44.

Test Case #	Point To Point	Crane System	Octree System
1	19.23	0.161	0.375
2	19.41	0.158	0.358
3	33.83	0.205	0.328
4	48.71	0.247	0.354
5	20.61	0.285	2.733
6	19.68	0.167	0.933
7	33.81	0.321	2.108
8	47.58	0.240	0.192

Table 44. Timing Results of Comparison Systems

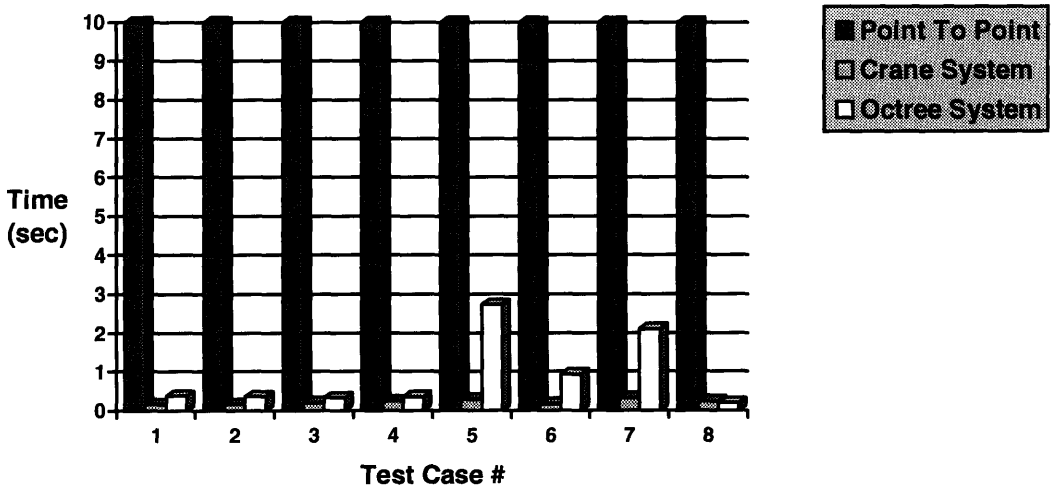


Figure 22. Graphic Representation of Timing Results

In general, the Crane Comparison System is the fastest of the three systems. This is due mostly to the rather rough-shod manner in which this system traverses the trajectory. The resolution at which the trajectory is checked is so low that the comparison may theoretically completely miss areas of intersection. Furthermore, areas where the trajectory approaches the terrain are checked at this same resolution as areas where the trajectory is obviously removed from possible intersections. This approach yields run times which are extremely small, and relatively invariant to trajectory or terrain characteristics. However, as has been noted previously, it also brings into question the correctness of the system.

The Octree System is a close second to the Crane System in time efficiency. The Octree System gains this efficiency by largely ignoring sections of the terrain which are far removed from the terrain. Instead it concentrates its efforts on sections of the trajectory where intersections are most likely. It may be reasoned that the time taken to traverse the octree, as well as the memory required to store it, is proportional to the amount of the trajectory which lies in close proximity to the terrain surface, since this is the region which requires the most levels of subdivision. As a result, the Octree System appears to have difficulties in test cases where the trajectory generally lies in close relation to the terrain. Note the extraordinarily long run times for test cases 5 and 7, which are low trajectories over rough terrain.

The Point To Point System takes far longer than any of the systems on every one of the test cases. So long in fact, that it is doubtful that the Point To Point System could be seriously considered in an actual production system. This poor time efficiency is due to the brute force approach which this system takes to achieving accuracy. The entire trajectory is checked at a rather high resolution. No attempts are made to reduce the number of comparisons during uninteresting portions of the trajectory. The result is the enormous run times observed here.

8.1.4. Accuracy

The maximum error of the comparison systems is shown below in Table 45.

	Point To Point		Crane System		Octree System	
Test Case #	Max. Horiz. Error	Max. Vert. Error	Max. Horiz. Error	Max. Vert. Error	Max. Horiz. Error	Max. Vert. Error
1	5.00	0.81	73.96	15.15	4.28	0.66
2	4.00	1.88	46.86	29.98	4.85	2.33
3	0.00	0.00	29.54	39.96	1.12	1.51
4	1.01	1.82	36.32	37.35	0.71	0.27
5	31.00	1.18	97.88	6.77	32.67	1.30
6	14.00	5.46	52.62	29.17	14.59	5.24
7	11.00	2.36	66.98	11.58	10.27	2.17
8	0.00	0.00	92.19	116.34	0.28	0.05

Table 45. Accuracy Results of Comparison Systems

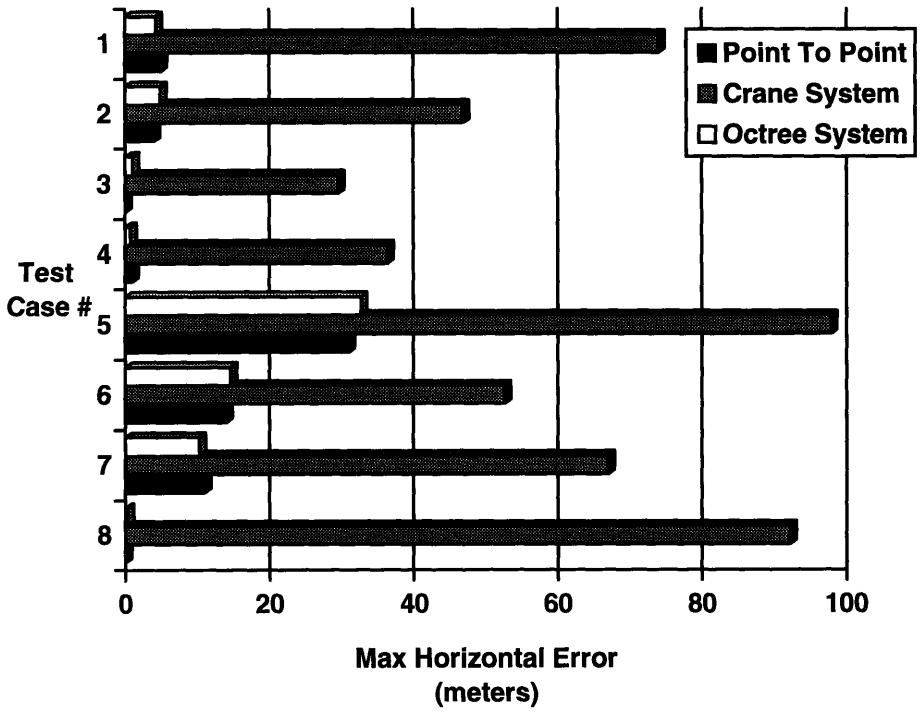


Figure 23. Graphic Representation of Maximum Horizontal Error

The Point To Point System and the Octree System display approximately the same level of accuracy. Although the methods they apply are quite different, they both produce adequate results. The accuracy of the Point To Point System is due mainly to the high resolution with which this brute force method checks the trajectory. As has been shown, this high resolution has a seriously detrimental effect on the time efficiency of the system. The Octree System, on the other hand, achieves both accuracy and time efficiency through a divide and conquer strategy which allows the entire trajectory to be checked via the actual comparison of only a limited number of points. Sections where the trajectory is obviously clear of the terrain are ignored after only a few comparisons, allowing more time to be spent in comparing regions where the trajectory is relatively close to the terrain.

The Crane Comparison System is by far the least accurate of the candidate comparison systems. As was discussed previously, the accuracy of the Crane System is so low that it is doubtful whether the Crane System is a feasible candidate for a production system. This lack of accuracy is most likely due to the Comparison Algorithms which are used by the Crane System. The Octree System, like the Crane System, gains efficiency by only checking a limited number of points along the trajectory in order to detect intersections. However, once an intersection is detected, the Octree System performs a linear interpolation of the terrain in an attempt to further locate the actual point of intersection. The Crane System, on the other hand, simply returns the location of the point at which the intersection was first detected. As can be seen from the results, this method does not yield accurate results. This would lead to the conclusion that the Crane System accuracy might be improved by performing an additional intersection point location technique similar to that of the Octree System after the initial intersection has been detected. This would, of course, decrease the very impressive efficiency of the Crane System. To what extent has yet to be determined.

8.1.5. Memory Usage

The space efficiency of the three comparison systems is shown below in Table 46.

Test Case #	Point To Point	Crane System	Octree System
1	193784	169794	190170
2	171420	148536	156536
3	275672	251672	258676
4	365008	341008	348676
5	172856	150056	206354
6	164026	140026	178150
7	305424	281424	391864
8	350092	326092	332700

Table 46. Memory Efficiency of Comparison Systems

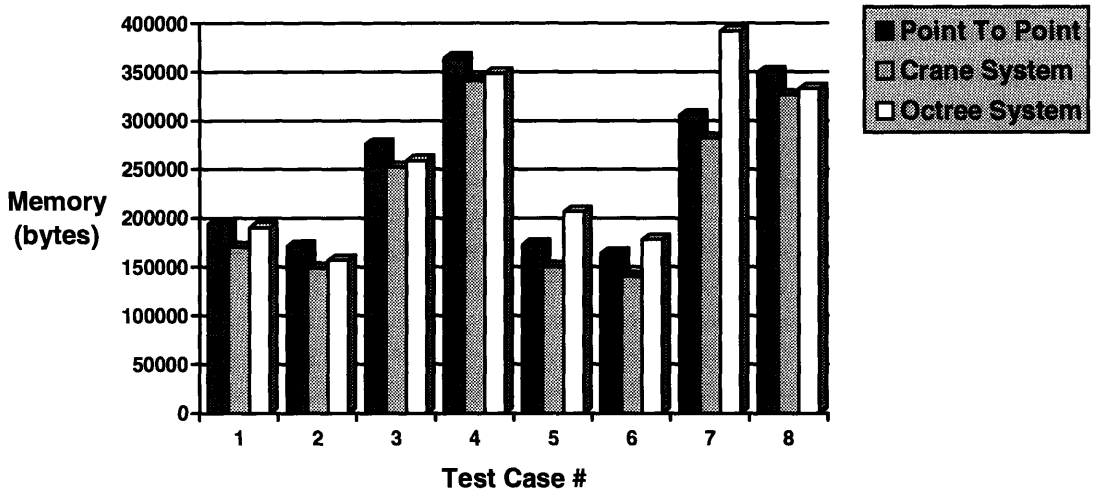


Figure 24. Graphic Representation of Memory Efficiency

The memory usage displayed by the Crane Comparison System is slightly better in all test cases than the other candidate systems. This is to be expected considering the simple nature of the Crane System algorithms. The only substantial data which must be stored in memory is the DTED itself.

The Octree System shows memory usage which is slightly greater than the Crane System. This is due to the presence of the octree structure which must be constructed and maintained in memory. The fact that the increase is so small is due to the extreme sparseness of the octree which was discussed during the analysis in Section 7.3. On test cases in which the trajectory lies fairly close to the terrain, the Octree System shows rather worse memory performance than usual. This is to be expected, as regions of the trajectory which lie near the terrain require greater levels of subdivision than otherwise.

The Point to Point System shows rather poor memory usage considering the simplicity of the algorithms. The reason for this is due to an implementation decision, and therefore the test results do not do justice to the simple nature of the approach. In the implementation of this system, large number of trajectory points (up to 1000) are computed in batches, and then each point is individually checked against the terrain. This allows certain gains in efficiency, but requires large amounts of projectile location data to be held in memory during the actual trajectory / terrain comparison. Certainly a more memory efficient approach could be implemented, but as was discussed previously in the section on evaluation criteria, time efficiency is deemed to be more important than memory efficiency.

8.2. Conclusions

For overall accuracy and efficiency, it must be concluded that the Octree System is the superior candidate system. Like all the candidate systems, it is will within the specified memory limitations. It seems to show no difficulties with correctness, both in theoretical analysis as well as practical test cases. Time efficiency, while not exceptional, is in most cases comparable to the Crane System. It does have problems with test cases involving low-lying trajectories, but the degradation is not critical. The accuracy of the Octree System is comparable to that displayed by the Point To Point System, while it experiences none of the efficiency degradation which is characteristic of that system. And the memory usage of the Octree System is average, despite the octree organization structure which is constructed and stored in memory during the course of the comparison. The approach taken by the Octree System is elegant in that it uses a divide and conquer strategy, and concentrates resources in evaluating the most interesting areas of the trajectory.

The Crane System shows the most impressive efficiency results in both memory and timing. And like the other candidate systems, the test cases indicate that memory limitations, as well as correctness requirements are observed. However, the extremely poor accuracy, as well as the accuracy analysis performed in Section 6.3.3, indicate that the error displayed by the Crane System may reach upwards of 160 meters in the horizontal direction. This is unacceptably large error, as this would affect the correctness evaluation for this system. Because of these concerns, it is doubtful whether the Crane System, despite its impressive efficiency, could be considered as a candidate in a production system.

The Point To Point System is an example of a brute force algorithm which uses a slow, careful, methodical approach to completing the comparison. This system, like the others, remains within the memory limitations and displays correctness. And it also shows nominal accuracy and memory usage. However the time expenditures of this system are enormous. A comparison which is desired to complete in a matter of seconds is now measured on a scale of minutes. While serving as an excellent candidate by which to compare the other systems, the Point To Point System cannot be seriously considered as a production system algorithm itself.

8.3. Additional Discussion

8.3.1. Timing of DTED Request Algorithms

One aspect of the timing which was mentioned earlier in the discussion of evaluation criteria in Section 3.4.1, but which was not further elaborated upon was the time required by the DTED Request Algorithms. In general, all of the comparison systems which were evaluated remained will within the required memory limits. Therefore it was not necessary to use different DTED Request Algorithms, nor to alter the original algorithms in any way. The time taken by the DTED Request Algorithms was approximately equal for all test cases, and in all cases was under 1 millisecond, which was the resolution at which the algorithms were timed. So these algorithms did not affect the relative timing of the comparison systems in any way.

However, it must be remembered that only a single resolution of DTED was used in all eight of the test cases, and this resolution was DTED level 1 which is fairly low. The amount of memory needed to store DTED level 2 is roughly 9 times that required to store DTED level 1. Higher resolution DTED may therefore cause some of the comparison systems to exceed the 4 megabyte memory limitations, which would possibly require changes to the DTED Request Algorithms in order to decrease the amount of DTED initially loaded into main memory to the minimum amount needed by a particular comparison system. In this case, the timing of the DTED Request Algorithms may begin to factor into the timing of the overall comparison system.

8.3.2. Extension to Longer or More Complex Trajectories

Although not addressed in the design or evaluation of the comparison systems, the property of scaleability may be critical to an actual production system. The trajectories which were explored and tested in this project were simple parabolic trajectories ranging from 20 to 50 kilometers. An actual system may involve trajectories which represent pseudo-random or noisy paths, and which may extend for distances far greater than 50 kilometers. It is reasonable therefore, to consider the scaleability of the approaches taken by the candidate comparison systems.

The Point To Point System has a major problem with time efficiency. Furthermore, the results of the test cases indicate that the time required is proportional to the overall length of the trajectory. Therefore, the time efficiency will be further degraded by a longer or more complex trajectory. This shows that this approach is not very scaleable.

The Crane System has impressive efficiency in both time and memory usage. Neither of these attributes appear to depend greatly on the length or complexity of the trajectory. The major problem with the Crane System is the accuracy and correctness of the algorithm. However, this problem is inherent in the comparison algorithm itself, and is not a function of the trajectory. Although a longer or more complex trajectory would increase the probability of a major error occurring, simply because more intersections points are possible. But in general, the Crane System may be considered to be quite scaleable.

The Octree System, although not as scaleable as the Crane System, does maintain its satisfactory efficiency over longer and more complex trajectories. The concerns of the Octree System involve memory and timing. As was discussed previously, in the Octree System these attributes are functions, not of the length of the trajectory, but of the amount of trajectory which lies close to the terrain. This allows the Octree System to handle longer or more complex trajectories very well.

8.3.3. Extension to Comparison of Multiple Trajectories

It may be the desire of an actual production system to compare several possible trajectories between a launch site and an impact point in order to determine which trajectory has the greatest clearance, or which trajectory has intersection points closest to a certain desired location. In this case it is necessary to perform comparisons of multiple trajectories over the same terrain. We therefore consider the ability of each of the comparison systems to take advantage of the coherency between similar trajectories in order to improve the efficiency of subsequent comparisons.

It is not obvious from the descriptions of the Point To Point System and the Crane System how these approaches may take advantage of trajectory coherence. Both of these systems perform terrain interpolations at discrete locations which are specific to a particular trajectory. If subsequent trajectories do not pass over these exact same points (which is unlikely in trajectories which are only similar, not identical) then the interpolated terrain values would be essentially useless.

The Octree System, on the other hand, constructs an octree structure which is the same for all trajectories which have the same launch and impact points. Thus the same octree could be used to check all similar trajectories, and computed nodes would not need to be recomputed for subsequent comparisons. This would increase the amount of memory needed to store the octree, since after several comparisons the structure would not be as sparse as it was for a single trajectory comparison. However, the cost in memory would result in tradeoffs in time efficiency. Time taken to allocate and construct new nodes could be amortized over several comparison runs.

8.3.4. Code Size

The final executable file, after linking, requires 555,024 bytes of storage. This file includes all the code necessary for this project, including the ballistic model, all of the comparison systems, and the code for controlling and manipulating the DTED. This executable file also contains a fair amount of code used only for collecting and analyzing data. Therefore, the code for an actual system should be substantially smaller. Regardless, an executable of this size, under 600,000 bytes, should fit easily onto the type of target processor considered in this project.

LITERATURE CITED

BARBE91 Barberi, F. and others, "Displaying morphological and lithological maps: a numerically intensive and visualization application," *IBM Journal of Research and Development*, Vol. 35, Jan-Mar 1991, pp. 78-87.

BRL79 Ballistic Research Laboratories (BRL), "A Modified Point Mass Trajectory Model for Rocket-Assisted Artillery", BCS V3-B5 Rev. 1, Sep 1979, pp. 1-18.

FOLEY90 Foley, James D., Andries van Dam, Steven K. Feiner, John F. Hughes, *Computer Graphics Principles and Practice*, second edition, Addison-Wesley, 1990.

LEE92 Lee, J. and others, "Modelling the effect of data errors on feature extraction from digital elevation models," *Photogrammetric Engineering and Remote Sensing*, Vol. 58, Oct 1992, pp. 1461-1467.

MASTR89 Mastrey, Ann T., *Generation and Evaluation of a Ballistic Algorithm Development Model*, Masters Thesis, Rensselaer Polytechnic Institute, Troy, New York, Jun 1989.

MIL89 Military Specification MIL-D-89001, "*Digital Terrain Elevation Data (DTED) Level 1 (ICF)*", 1990.

MILD Military Specification MIL-HDBK-600008, "*Army Coordinate Conversion Guide*", U.S.Army Topographic Engineering Center, DRAFT.

MILLE86 Miller, G. "The Definition and Rendering of Terrain Maps," *Computer Graphics* (Proc. SIGGRAPH), Vol. 20, No. 4, Aug 1986, pp. 39-48.

RAPP84 Rapp, Richard H., *Geometric Geodesy - Part I*, Department of Geodetic Science and Surveying, The Ohio State University, Columbus, Ohio, 1984, p. 124.

SAMET88A Samet, Hamet, and Robert E. Webber, "Heirarchical Data Structures and Algorithms for Computer Graphics, Part I", *IEEE Computer Graphics and Applications*, Vol 8 No 4, Jul 1988.

SAMET88B Samet, Hamet, and Robert E. Webber, "Heirarchical Data Structures and Algorithms for Computer Graphics, Part II", *IEEE Computer Graphics and Applications*, Vol 8 No 4, Jul 1988.

SAMET89 Samet, H., "Implementing Ray Tracing with Octrees and Neighbor Finding," *Computers and Graphics*, 13(4), 1989, pp. 445-460.

VANIC86 Vanicek, Patr, and Edward J. Krakiwsky, *Geodesy, The Concepts*, Elsevier Science Publishers B.V., New York, N.Y., 1986.

YOKOY89 Yokoya, Naokazu, Kazuhiko Yamamoto, and Noboru Funakubo, "Fractal-Based Analysis and Interpolation of 3D Natural Surface Shapes and Their Application to Terrain Modelling", *Computer Vision, Graphics, and Image Processing*, Vol 46, 1989, pp. 284-302.