1-1-1998

ZCAP Research And Development: Final Report

University of Central Florida Institute for Simulation and Training

INSTITUTE FOR SIMULATION AND TRAINING

ZCAP Research and Development

Final Report

Contract Number: N61339-97K-0003
CDRL A005

Prepared by:
Institute for Simulation and Training

IST-TR-98-11

INSTITUTE FOR SIMULATION & TRAINING

A11

ZCAP Research and Development

Final Report

Contract Number:

N61339-97-K-0003

CDRL A005

Prepared by:

Institute for Simulation and Training
University of Central Florida
Orlando Florida

Prepared for :

STRICOM
12350 Research Parkway
Orlando, FL 32826

IST-TR-98-11

ZCAP Research and Development: Final Report

Guy A. Schiavone
Milton Sakude
Benito Graniela
Hector Morelos-Borja
Art Cortes

I. Introduction

IST's suite of terrain database correlation tools, collectively known as ZCAP, has proven useful for assessing spatial consistency between heterogeneous terrain datasets designed for networked simulation and training applications. Originally, ZCAP was developed under the Visual Testbed Project (VTB) funded by STRICOM and DMSO. By the time of the completion of the VTB project in 1996, it had become obvious that a number of enhancements to ZCAP were required to meet anticipated needs of future analysts and database developers. One of the foremost needs was the ability to handle a larger set of database formats. Another need was the ability to assess the quality of terrain data contained in larger databases than the then-current version of ZCAP could handle. A third goal was to investigate approaches to improve the usability of the ZCAP correlation testing software. Other goals that fall under these major headings include the improvement of sampling procedures, enhancing the software organization of ZCAP, and promulgate the results of this research through conference presentations and improved documentation. In this final report we detail the steps taken to meet these goals, the results of the research, and suggestions for future directions in terrain database research. This report is intended as a general project overview; complete specifications and other details of the work performed can be found in the ZCAP4.1 User's Manual [1] and in the ZCAP4.1 Programmers Documentation [2].

II. Project Scope/Objectives and Outcomes

The ZCAP Research and Development project entailed research into the factors affecting the development of software tools for assessing correlation between databases designed for distributed simulation applications typically employed for the purposes of training, mission rehearsal, and tactics assessment. The suite of software tools known as ZCAP provides capabilities for terrain correlation testing, culture correlation testing, line of sight (LOS) correlation testing, coordinate conversions, shift detection, terrain database sampling, terrain remediation, terrain database generation, and terrain database visualization. Although the original versions of ZCAP had proven useful in assessing terrain database correlation, a number of deficiencies were noted in these early versions that were addressed in this project. As this project evolved, new goals were identified and prioritized along with the original goals. Described below are the main goals of this project as identified at project completion, steps taken to reach these goals, and the outcome of the research and development effort.

A. Expansion of Format Conversion Capabilities

Goals – To increase the applicability of ZCAP correlation testing by expanding the number of valid terrain database input formats. Additionally, to provide significant added-value by providing a number of generally useful format conversion utilities.

Discussion - So that the ZCAP tools could be used by a larger set of potential players in distributed simulation, it was deemed important that tools be provided for the input of a larger set of terrain database input formats than was available in previous versions of ZCAP. ZCAP input tools fall into two classes: 1) sampling tools, and 2) converters.

Sampling tools provide the capability of sampling elevations over a set of polygons that define a surface. Usually these polygons will be terrain polygons, but may also be polygons that are attributed as culture features, such as roads, tree canopies and water surfaces. In the ZCAP terrain elevation testing procedure, XY pairs of sample points are generated in a global coordinate system such as UTM or geodetic by the sample generation tool. These sample points are usually randomly distributed over the common extents of the two databases under test, with locations being chosen from a 2D uniform random distribution scaled to the common terrain database extents. These points are written to a file and used as input into a sampling tool. Each sampling tool is designed to sample elevations over a specific terrain database format at the points specified in the sample locations input file. The sample tool must translate the global coordinates into whatever local coordinate system is in use, then input polygons form the foreign format into memory and perform a search for the polygons containing the specified points. Once a sample location point is paired with the polygon that contains it, a linear interpolation is performed to calculate the elevation at that location. The output of a sampling tool is the same list of global XY points that were used as input, along with the corresponding elevation at each point. This output file is then used as input into the terrain elevation correlation tool in ZCAP. The terrain elevation correlation tool requires one file each from the two databases being compared.

Outcome - A new CTDB sampling tool was developed in the course of this project, compatible with .c7b format. Also, the existing OpenFlight sampling tool was upgraded to OpenFlight Version 15.4 using the MultiGen API. These new sampling tools are in addition to the previously existing tools for sampling SIF, S1000, and ZCAP formats.

Discussion - ZCAP converters are tools that parse through terrain data in a foreign format, and convert the polygons and attributes into ZCAP 3D format for subsequent processing by ZCAP correlation testing tools or utilities. ZCAP 3D format is currently required as input into the culture correlation and LOS correlation tools. Since a ZCAP sampling tool is also provided, using a converter tool is an alternate path in performing terrain elevation correlation as well.

Outcome - The following converters were developed in the course of this project, and are supplied with ZCAP: Standard Simulator Data Base Interchange Format (SIF) to OpenFlight© Version 15.4 format, S1000 format to OpenFlight© Version 15.4 format, and ZCAP format to OpenFlight© Version 15.4 format. The ZCAP to OpenFlight©

Table I. Input matrix for ZCAP4.1, the most recent ZCAP version as of October, 1998.

Terrain Database Format	**Z Correlation Test** (3D Terrain Random Elevation Sample Points)	**Line-of-Sight Test** (3D Terrain and Culture Polygons)	**Culture Correlation Test** (3D Terrain and Culture Polygons)	**ZCAP Visualizer** (3D Terrain and Culture Polygons)	**Shift Detection** (3D Terrain Gridded Sample Points)
Performer Scene Graph *P,A* (Silicon Graphics in-memory data structures)	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
MultiGen™ OpenFlight™ *P,A* (Latest Revision-15.4)	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
S1000 *P,A*	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
SIF/HDI *O*	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
E&S GDF *P,A*	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
E&S ESIG2000 Run-Time Data Base (RTDB) *P*	[redacted]	None	None	None	[redacted]
CCTT MRTDB *P*	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]
ModSAF CTDB *O*	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]

Legend *P*: Proprietary | *O*: Open (Non-Proprietary) | *A*: API Accessible

Full: Format is input compatible with ZCAP 4.1
Partial: Format can be converted to a ZCAP 4.1-input-compatible format, BUT some conversion steps must be done outside ZCAP 4.1
None: Format currently cannot be input into ZCAP 4.1

•Note: This figure describes terrain database format input compatibility with ZCAP 4.1; it does NOT imply that ZCAP 4.1 has the compatibility to perform loss-free conversions between the formats in column 1.
*Requires MakVR-LINK and SGI Performer - Not Integrated Into ZCAP 4.1
** Requires E&S proprietary DET API on a Solaris platform.
***Requires utilities not included in ZCAP 4.1, limited to terrain, cut/fill, microterrain, and surface linears.
****Requires proprietary utilities - Not Integrated into ZCAP 4.1

format converter can be used to convert legacy ZCAP format databases into the currently supported OpenFlight© format. It can also be used to allow a two-step conversion of Evans and

Sutherland (E&S) GDF format data to OpenFlight© format; the first step converting GDF to ZCAP format, the second step converting ZCAP format to OpenFlight© format. A direct GDF to OpenFlight© converter was not developed due to the fact that the API's used to access GDF and OpenFlight© were not available on the same computing platform. The E&S DET API for accessing E&S GDF terrain databases was available on a SUN platform and the MultiGen™ OpenFlight© API was available on a Silicon Graphics platform.

In addition, the OpenFlight to ZCAP converter was upgraded to Version 15.4 using the MultiGen API, and a new tool for OpenFlight preprocessing was created that performs FID to FDC mapping, for preparing OpenFlight data for use in the ZCAP culture correlation and LOS correlation testing procedures. All of the above conversion tools have been included with the previously existing tools for converting SIF to ZCAP and S1000 to ZCAP format. The complete ZCAP4.1 input matrix is shown in Table I, above.

B. Improved Capabilities for Correlation Testing of Very Large Terrain Databases

Goal - To make improvements in ZCAP software and algorithms that would allow for testing of much larger databases than was possible for previous versions.

Discussion – Improvements to ZCAP undertaken to reach this goal fall under two headings. The first was to improve the ZCAP internal format to accommodate metadata that is common to most terrain datasets. The second and most important area that required improvement was the efficiency of various spatial data handling and spatial searching algorithms that are used in ZCAP. Specifically, efficiency improvements were made by employing Grid Method Sampling in the terrain correlation test tool, improving the method of stratified sampling used in the culture correlation tool, employing a more efficient LOS calculation in the LOS correlation test, and increasing the efficiency of the terrain remediation procedure by using improved sampling and sparse matrix techniques. Steps taken to improve each of these areas are discussed below. Many of the following approaches and results were promulgated in conference presentations, and are documented in references [3] and [4].

1. ZCAP4.1 Internal Format Improvements

One aspect of IST's ZCAP enhancement effort was the adoption of a more representationally complete and size-efficient terrain database format. The original ZCAP format is an easy-to-use, simple vector format that is stored entirely as ASCII text. It is a non-proprietary, platform-neutral format. The simplicity of ZCAP format, however, precluded representation of relatively large terrain databases in terms of spatial extents and feature density and of representation of metadata required for automating ZCAP correlation tests.

An initial step taken by IST to improve the original ZCAP format was the development of a ZCAP database header file containing high-level statistics and other metadata about

a ZCAP database. A high-level specification of the header file contents follows (a more detailed specification is provided with the ZCAP4.1 Users Manual) :

```
DATABASE_NAME: DB_Name
DATABASE_TYPE: DB_Type
NUMBER_OF_FEATURES: Number_of_Features
if DB_Type = Three_Dimensional then
  NUMBER_OF_TRIANGLES:Number_of_Triangles
if DB_Type = Two_Dimensional then
  NUMBER_OF_SEGMENTS: Number_of_Segments
NUMBER_OF_VERTICES: Number_of_Vertices
NUMBER_OF_UNIQUE_FDCS: Number_of_Unique_FDCs
for each unique FDC
   if DB_Type = Three_Dimensional then
     FDC: FDC Value #ofFeatures %ExtentsCovered
  if DB_Type = Two_Dimensional then
     FDC: FDC Value #ofFeatures
COORDINATE_SYSTEM: Coordinate_System
if Coordinate_System = Cartesian then
     UNITS: Units
HORIZONTAL_DATUM: Horizontal_Datum
VERTICAL_DATUM: Vertical_Datum
Database_Origin
Database_Extents
```

Information in the ZCAP header file, such as the database extents, allowed for the development of routines to automate the determination of common extents between two databases being tested using ZCAP. Previously, the user had to determine common database extents manually, which can be tedious and error-prone.

ZCAP 4.1 database format is an ASCII text format that allows for the representation of two-dimensional (2-D) or three dimensional (3-D) vector data. Each ZCAP 4.1 3-D database consists of four files: a header file, a feature file, a triangle file, and a 3-D vertex file. Each ZCAP 4.1 2-D database consists of four files: a header file, a feature file, a segment file, and a 2-D vertex file.

The ZCAP 4.1 3-D database feature file consists of a set of one or more "features", which define a group of one or more three dimensional triangles from the triangles file. Each triangle, in turn, is composed of three vertices from the 3-D vertex file.

The header file is the same for the 2-D and 3-D ZCAP 4.1 database formats. It contains statistics and other "metadata" about the database as a whole.

For a complete specification of all ZCAP4.1 internal formats, please see the ZCAP4.1 User's Manual [1].

2. Efficiency Improvements in ZCAP4.1

In order to provide the capability to handle larger terrain databases, it was realized early in this project that the spatial data handling and spatial searching algorithms in earlier versions of ZCAP must be improved. The details of the specific improvements made during the course of this project are given in this section.

a) Grid Method Sampling

The efficiency of an algorithm, that deals with a large amount of data, generally depends on how fast it accesses the needed data. For search efficiency, data are organized by sorting and using efficient data structures such as binary trees, AVL trees, quadtrees, and hashing tables.

The basic idea is to organize sample points instead of organizing the terrain database and to traverse the database once, because sample points occupy less memory space than terrain data.

The Grid method is ideal for uniformly distributed data, which is used by all ZCAP correlation tests, because in practice a search of N points runs in O(N) time on average [9].

The Grid method uses a matrix of pointers (hashing table) to a list of data indices as a basic data structure. X and Y values are transformed into an index into this matrix by using a hashing function to access data in the list. Algorithm efficiency depends on keeping a relatively low number of elements in the lists and all lists occupied.
For uniform grid sampling, the following algorithm is used:

Algorithm 1. Uniform grid sampling.

1. For each polygon in the terrain database
    1.1 Determine polygon bounding box
    1.2 For each gridded point within bounding box
        1.2.1 If point is inside polygon
            1.2.1.1 Calculate polygon Z value

Let M be the number of polygons in the terrain database. For a database composed of relatively small polygons, step 1.2 runs in constant time (O(1)). The overall algorithm runs in O(M) time. (Figure 1.)

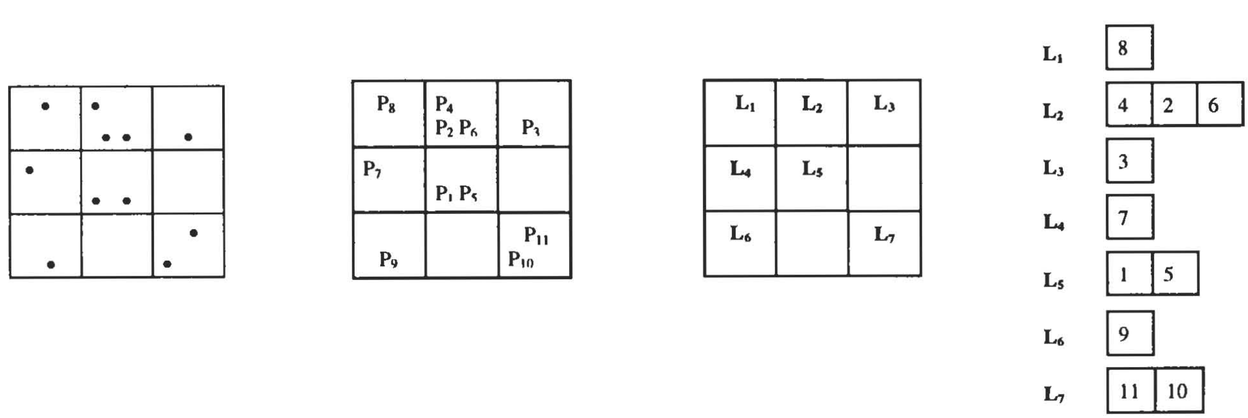


Figure 1. Grid subdivison, pointer matrix and index list.

The above algorithm can be extended easily for random sampling, substituting the gridded point by a list of points. Random points inside of a grid (cell) are stored in the list.

The random sampling algorithm is as follows:

Algorithm 2. Random Sampling

1. For each polygon in the terrain database
   1.1 Determine polygon bounding box
   1.2 For each cell list within bounding box
      1.2.1 For each point in the cell list
         1.2.1.1 If point is inside polygon
            1.2.1.1.1 Calculate polygon Z value

Let N be the number of sample points, and let n and m be the matrix dimension, so that nm=N/c, where c is the occupancy target (c points per cell). The pre-processing time for building the grid structure is O(nm+N) [9], that is, O(N) time. The overall time complexity of the algorithm is O(N+M).

Since it takes O(M) time in pre-processing to organize a terrain database with M polygons by using any efficient data structure, such as a quadtree, and the sampling is performed once, the theoretical overall performance of the proposed algorithm is optimal.

The higher the ratio between the number of sample points and the number of polygons (N/M), the better the relative performance of the proposed algorithm, because step 1.2 finds more points to process. For high ratio N/M the proposed algorithm can outperform an algorithm that uses a quadtree or a K-D tree. A point search in a quadtree or a K-D tree structure with D levels requires O(D) time. For simplicity let us consider 2^D=M, that is D=log M (see reference [9] for details). Sampling N points requires O(N log M) time. Without considering pre-processing time for the quadtree (because the database can be stored in that format) and time constants being equal, the break-even point between the methods is given by N=M/(logM-1), since T(N+M)=T(N log M). The region above the curve in Figure 2 represents points where the proposed algorithm theoretically presents superior performance. Terrain roughness determination requires a high rate N/M to account for all the "waviness". The default tool setup is approximately one sample per two polygons. In the terrain elevation correlation test, the replication of the test leads to a high rate N/M. Even without replications, it can be in the optimal region. Sampling points for the terrain remediation process requires several points per polygon.

The proposed algorithm has other advantages:

1. It uses less memory space (O(N)), because it organizes sample points. Organizing terrain database by using a quadtree or a K-D tree requires O(M) space.
2. It is relatively easy to implement and can be used for sampling any terrain database format without conversion, since a polygon range searching or database traversal function is available.
3. It is suitable for large terrain databases. The terrain database does not need to be stored in memory, because it needs only one polygon at a time.

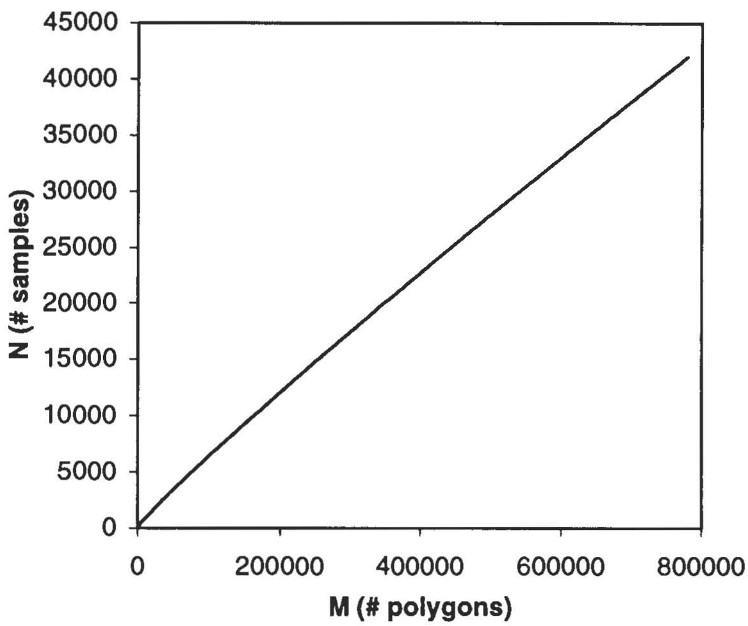


Figure 2. Break-even curve of algorithm superior performance.

b) Terrain Correlation Test

The terrain correlation test computes statistics on elevation differences between two terrain surfaces. It involves:

1. Uniform random sample generation;
2. Baseline terrain database sampling (z value calculation) ;
3. Subject terrain database sampling (using the same sample points);
4. Elevation difference calculation and statistics computation.

Statistics include mean, median, variance, standard deviation, skewness, kurtosis, magnitude of the maximum elevation difference, and critical value of the acceptance sampling test based on the elevation differences [5].

Since computer generated random numbers are used to generate sample points, performing replications of the test is important to evaluate the variation of the results. The variation analysis may indicate the need for increasing the sample size.
IST implemented the capability to replicate the terrain correlation test, taking advantage of the efficiency of the Grid method sampling algorithm. The sample points of all replications are grouped and processed as a large sample by using algorithm 2. The statistics computation for each trial is processed after ungrouping.

Table II shows an example of results. One database is a portion of Fort Hunter-Liggett terrain in S1000 format. The other is the same portion in OpenFlight© format. The statistic known as the "critical value", shown in Table II, is interpreted as the value such that 95% of the elevation differences are below the critical value (87 meters) at 95% significance level [8]. In other words, for a threshold greater than 87 meters, the databases pass the acceptance-sampling test at 95 % confidence level.

Table II Statistical output of ZCAP4.1 terrain elevation correlation test

STATISTICS	Ave-rage	Std. Dev.	Mini-mum	Maxi mum
Mean	2.267	0.775	0.851	3.557
Median	14.17	0.581	13.15	15.01
Variance	1307	61.33	1180	1394
Std. Deviation	36.15	0.853	34.36	37.34
Skewness	-0.26	0.123	-0.51	-0.13
Kurtosis	2.968	0.344	1.988	3.499
Maximum	190.6	19.05	149.0	216.9
Critical value	86.91	2.526	81.56	90.06

Table II Terrain correlation test statistics on terrain elevation differences of 15 replications. Each trial has 2000 sample points.

c) Culture Correlation Test

The ZCAP culture correlation test compares the agreement in feature location between two terrain databases by using the Kappa statistic [6]. Kappa statistic formula is:

$$K=(p_0-p_e)/(1-p_e)$$

Where p_0 is the overall proportion of agreement and p_e is the adjustment due to chance expected agreement (see more details in reference [6]).

To perform the culture correlation test, it is necessary to sample every culture feature. Important features such as airports, buildings, and small targets, that represent relatively small areas, should not be missed.

It was desirable to have an algorithm capable to generate stratified random locations on cultural features, given a desired number of sample points per feature. To achieve a certain number of sample points (N_f), a feature must be sampled with a density of points per area (at least N_f/total_feature_area). Figure 5.1 gives an idea of the algorithm: considering a pattern (mask) with normalized random points (values between 0 and 1) and a square containing N points, a scale transformation that magnifies this square to the square with equivalent total feature area gives the right density.
The following algorithm describes the stratified random sampling scheme:

Algorithm 3. Stratified Sampling

1. Calculate the total area for each feature
2. Generate normalized uniform random sample pattern and build grid structure.
3. For each polygon in the terrain database
   3.1 Determine bounding box
   3.2 Normalize polygon and bounding box
   3.3 Randomly place normalized polygon in the pattern
   3.4 Use grid method to search points inside polygon
   3.5 If found inside point
      3.5.1 Transform it back to terrain coordinate systems
      3.5.2 Store point location and feature code in a list

An algorithm similar to algorithm 2 is used to check agreement in location (using culture class values instead of Z values in step 1.2.1.1.1).

Normalization in step 3.2 is a scale transformation that adjusts the polygon size for applying a mask (pattern). The idea is to "fill" the polygon with random points. The use of a random pattern and the placement of the normalized polygon at random locations in pattern guarantee the stratified random sampling. This algorithm considers only areal features.

The area calculation (step 1) requires $O(M_1)$ time, where M_1 is the polygon number of the first terrain database. The number of points in the pattern can be chosen as the maximum value of all N_f's. Because of the use of the Grid method, steps 2-3 require $O(M_1+ N_f)$ time.

Feature agreement processing requires $O(M_2+N)$ time, where M_2 is the polygon number of the second (subject) terrain database and N is the total number of sample points.

d) Line of Sight Correlation Test

Like the culture correlation test, the Line of Sight Correlation Test (LOS test) uses the Kappa statistic [6]. Instead of generating random sample points, the LOS test computes the agreement of LOS blockage by culture features or terrain skin. The LOS is measured along vectors with equal length.

The previously implemented algorithm had time O(NM) complexity. An algorithm that uses the Grid method has better performance: O(N+M) time complexity. Similar to the previous approach, the Grid method is used to efficiently organize LOS data. Efficient LOS methods reported in reference [10] use considerable more memory space to organize the terrain database. Also, some of the algorithms in [10] are peculiarly adopted for regularly-spaced gridded representations, and are not applicable to generalized TINs that are typical of terrain databases used for visual applications, and increasingly common in CGF applications as well. In our approach, LOS sample data occupies less memory space than terrain data.

LOS segments are generated such that they are randomly distributed over the terrain. The end points of the segments are at a certain height (constant) above the terrain surface. Algorithm 2 is used to calculate the terrain elevation. The main processing of this test is the calculation of the intersection between LOS segments and terrain polygons. The following algorithm performs all the LOS intersection calculations:

Algorithm 4. LOS Intersection

1. For each polygon in the terrain database
    1.1 Determine polygon bounding box
    1.2 For each cell list in bounding box
        1.2.1 For each LOS segment in the list
            1.2.1.1 If LOS can intercept polygon
                1.2.1.1.1 Calculate LOS Intersection
                1.2.1.1.2 If point is inside polygon
                    1.2.1.1.2.1 Update intersection

LOS-polygon intersection checking consists of two trivial nested rejection tests, in this order:

1. 3D bounding box rejection test: LOS segment and polygon bounding boxes do not overlap.
2. 2D circle rejection test: distance of the center of the polygon bounding box to the LOS line is greater than half of the diagonal length (Figure 3). The point-line distance is given by

$D^2=(a\ x+b\ y+c)^2/(a^2+b^2)$. This test is more efficient than several other widely-used tests [10] that check if polygon vertices are in the same side of the line.

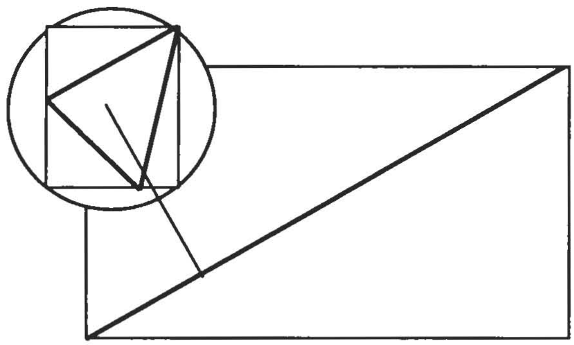


Figure 3. LOS-polygon intersection rejection

The LOS intersection point is the intersection of a line segment and a plane:

$$P = P_1 + t\ (P_2-P_1)$$

With $t = (P_0-P_1)\bullet N/(P_2-P_1)\bullet N$, and $0 \leq t \leq 1$.
Where N is the plane normal, P_0 a point in the plane, P_1 and P_2 end segment points and $\bullet$ denotes dot product.

Considering that the number of the LOS segments in the cells of the grid structure is relatively small, the overall time complexity of the algorithm is O(N+M).

e) Fifty percent visibility LOS segment length

Schiavone et al. [6] state that the LOS test has the maximum sensitivity if the LOS segment length is such that it yields a probability of 50% of visibility. Fifty-percent visibility means that 50% of the LOS is blocked by features or terrain.

The solution to this problem consists in finding a root of the curve shown in Figure 4 at LOS visibility value of 50%. We use the Secant method to find the LOS segment length at 50% visibility. To ensure that the LOS segment length provides 50 % visibility, each curve point visibility is the average of 30 trials of 200 LOS segments. To take advantage of the Grid method, the processing is done in one batch of 6000 LOS segments. The convergence criterion is achieved when the average visibility of the 30 trials gives a confidence interval that contains 50% at 95% significance level. This guarantees that the LOS segment length found yields a probability of 50% visibility with 5% of error.

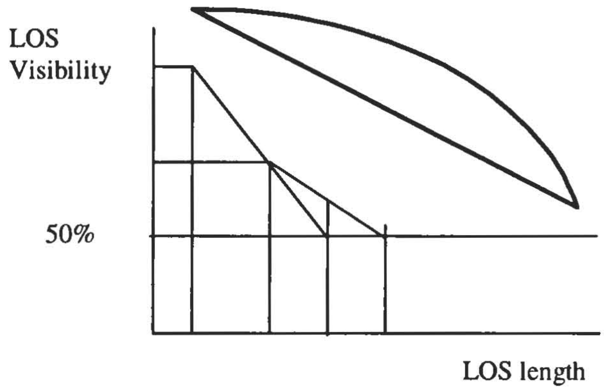


Figure 4. LOS distance determination by Secant method.

f) Terrain remediation

Terrain remediation is an important process to alleviate errors between two different representations of the same terrain surface. Schiavone and Graniela [7] developed an automated tool to address this issue. They developed an algorithm that adjusts the polygon vertices of a terrain database by using a constrained least square fitting method. It works only on terrain skin represented by triangles and maintains the triangulation (changes only the z values).

The basic idea is to fit a plane to a set of points sampled from a reference terrain database. It is similar to our improved roughness fitting, but instead of finding the plane equation coefficients, it determines the triangle vertex z values of the subject terrain database. Since surface continuity must be maintained, the linear system of equations for each triangle are put together forming a sparse over-determined linear system with size 3 T x V, where T is the number of triangles and V the number of vertices in the subject database (see reference [7] for more details). This terrain remediation processing was time consuming due to two reasons: the sampling process used an O(NT) algorithm and the full matrix least-squares linear system solution method that was used runs in $O(V^3)$ time with O(TV) memory requirements. The sparse matrix technique reduces these requirements to linear complexity in T for both memory and run time.

It is important to have a sampling algorithm that accounts for the shape of the reference surface. A certain number of random sample points per triangle are desirable to have good results. An algorithm that generates random points inside of a triangle bounding box may waste many points, mainly if the triangle has a small area in relation to its bounding box.
An efficient way to generate a random point inside of a triangle is to use a parametric equation of a plane:

$$P = t\,P_1 + u\,P_2 + v\,P_3$$

With $0 < t, u, v < 1.0$ and $u+v+t=1$, P is inside of the triangle.
It does not need to check if the point is inside of the triangle and it also calculates the z value.
The algorithm for generating a point inside of a triangle is:

Algorithm 5. Inside triangle random point

1. Generate random numbers u and v
2. If $(u+v>1)$
   2.1 Set $u = 1-u$ and $v = 1-v$
3. $t = 1-v-u$
4. $P = t\,P_1 + u\,P_2 + v\,P_3$

Algorithm *5* is applied for each triangle in the subject database, storing the points in a sample list. The number of sample points is a function of the triangle area and a sample density. Sample density is such that the desired number of sample points in the reference polygon is satisfied, on average. Random sampling algorithm (2) is used to calculate the z value of the sample points in the reference database. With this improvement, the sampling step of the terrain remediation tool runs in O(N+M) time.

The efficiency in solving the overdetermined linear system was greatly improved in ZCAP4.1 by using SPOOLES (Sparse Object-Oriented Linear Equations Solver) [11]. SPOOLES is a public domain, free-source numerical analysis software package (funded by DARPA) that solves full rank overdetermined systems of the form $Ax \cong b$ by performing the sparse QR decomposition for the matrix A. The solution involves reordering the matrix A, factoring the matrix $A \cong QR$ and solving the linear systems of equations. To preserve the sparsity in the factorization, the matrix A is reordered (permutated) by building a graph structure of $A^T A$. The QR factorization is analogous to the transformation of matrix A into a triangular matrix [12]. The SPOOLES factorization works on the graph object representation. The last step is the determination of the solution x, working on the factorized form. The SPOOLES approach allows one to apply the remediation algorithm in a relatively large terrain database with significant reduction in processing time.

Outcome – The ZCAP correlation testing tools and other utilities can operate on much larger terrain datasets than was previously possible. The primary examples of these improvements are 1) the use of the Grid method for spatial searching has resulted in improvement from O(NM) to O(N+M) in all tools requiring a typical "point-in-polygon" spatial search, and 2) the use of sparse matrix technique in the terrain remediation tool has resulted in an improvement an the order of $O(N^3)$ to O(N) in the solution of the overdetermined system.

C. Usability Improvements in Correlation Testing Tools.

Goal – To improve the overall usability of the ZCAP package.

Discussion – Effort towards overall usability improvements in ZCAP and correlation tools in general fall into four main categories: 1) Software organization and interface improvements, 2) Installation improvements, 3) Documentation improvements, and 4) Research on improved interfaces and architectures. Steps taken to implement these changes and outcomes are discussed below.

1. Software Organization and Interface Improvements

Discussion - Over time, as the functionality of the ZCAP package has increased, the size and complexity of the program itself has increased in proportion. Including the packages such as tcl/tk and Spooles that are used by ZCAP, the entire ZCAP installation employs over 1 million lines of source code. One improvement that was made in the structure of ZCAP was to repackage the source code into an intuitive, standardized directory structure that is functionally organized. Also, the user interface for many of the tools was standardized to present a consistent interface. Many of the commonly used functions were archived into ZCAP libraries that are reusable by other packages and tools. Default data is available for all tools, and running through the operation of each tool using the default data serves as a tutorial for the inexperienced user. Much functionally redundant code has been eliminated. A new user's directory is now created by default, and ZCAP tests can be run in any directory through the introduction of a ZCAP_HOME environment variable. In addition, the CVS software package for version control has been adopted for use in the present and possible future releases of ZCAP.

Another issue when comparing two terrain databases has to do with the identification of common extents. In order of not to have to depend on high-expertise to be able to use these tools, ZCAP is set up with appropriate pre-defined default values for most of the required parameters. The identification of terrain database common extents has been simplified through the implementation of the ZCAP header file, where such information is accessible without requiring recalculation each time. Even though the user has the possibility of providing any desired value here, ZCAP also provides the capability to automatically identify the proper parameter value for a given terrain database. Nominal applications for ZCAP will benefit from this automatic identification and also from default-provided values for specific test parameters. On the other hand, specific applications and more expert users can take advantage of the more interactive modes also supported by ZCAP.

Outcome - ZCAP4.1 is a much better organized and easier to use suite of tools as compared to earlier versions. Understandability and reusability of the entire package has been enhanced.

2. Installation Improvements

Discussion - The ZCAP4.1 installation was reorganized by employing recursive makefiles in each subdirectory of the top-level source directory. This makes updating the package much easier, as the entire package need not be recompiled to effect local changes. This also allows many of the tools to be compiled and operated in stand-alone mode. The top-level makefile was reorganized to include new user-defined variables that allow for greater flexibility in the installation. For example, it is now possible to install the support packages such as tcl/tk and Gnuplot in a standard location such as /usr/local, so that other packages may easily find these tools.

Outcome - ZCAP4.1 is easier to install, and allows for more flexibility in configuration.

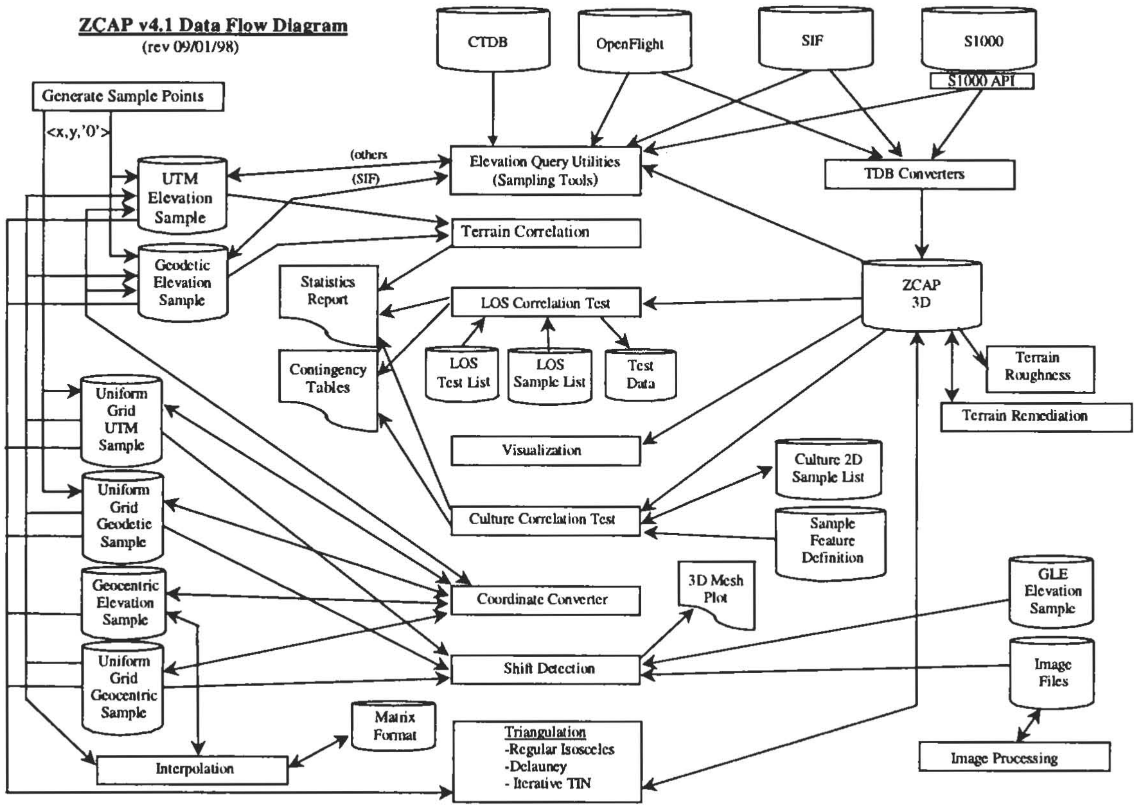


Figure 5. ZCAP4.1 Data flow diagram

3. Documentation Improvements

Discussion - The ZCAP4.1 website now includes links to all-new documentation in html format. The ZCAP user's manual has been completely updated, and includes a new ZCAP flow diagram (Figure 5) that illustrates all possible data flow paths through the various ZCAP tools. New programmer's documentation is also available. The programmer's documentation was automatically generated using IST's own version of the program DOC++.

Outcome – The updated User's Manual explains in detail not only how to use the ZCAP tools, but also much of the theory behind the algorithms that are employed in ZCAP. This should enhance understanding not only of the approaches to terrain database correlation embodied in ZCAP, but to the general issue of terrain database correlation overall. We expect the programmers documentation will greatly enhance the reusability of the code developed in the ZCAP effort.

4. Research on improved interfaces and architectures

Discussion - As part of IST's research and development for ZCAP, a prototype tool for terrain analysis and visualization, called the Remote Correlation Analysis Tool (RCAT), has been developed. RCAT is capable of performing terrain database analysis and visualization tasks from a remote location via the internet. RCAT is coded in Java for easy remote operation and portability, and currently uses VRML for terrain database visualization. Figure 6 shows the overall design concept of the complete RCAT system.

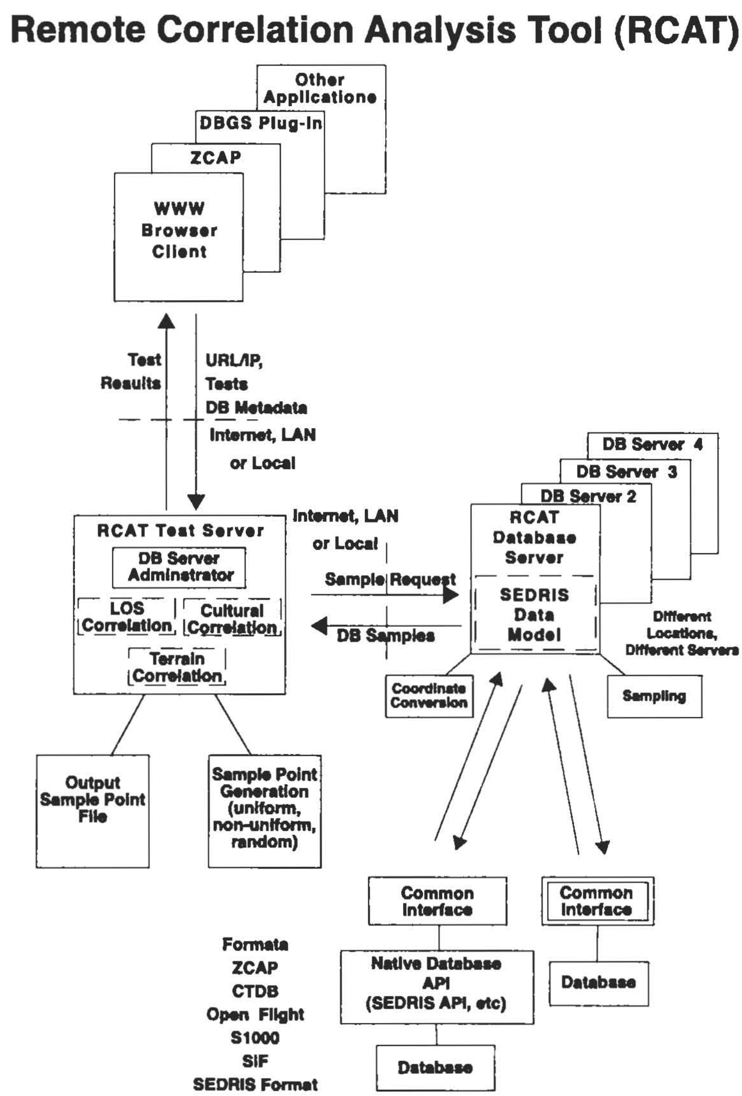


Figure 6. System Architecture for the Remote Correlation Analysis Tool (RCAT).

The three main components of the RCAT design are the user interface, the RCAT test server, and the RCAT database server. These three components may be located locally on the same platform, or at separate locations anywhere on the Internet. In typical applications of the services of two or more database servers can be used simultaneously. The advantages of remote testing are 1) it provides for objective testing of terrain database correlation from two or more sites and 2) it eliminates the need to download and compile the whole set of ZCAP tools to do LOS analysis and correlation testing.

The design allows for a variety of applications to be used as the user interface application. In the prototype system the user interface is a WWW browser client such as Netscape Communicator. Other applications may usefully serve as the user interface, with some examples being ZCAP, the MultiGen Modeler, or the S1000 database generation system (DBGS). Designing plug-ins for DBGSs will allow for interactive analysis and inspection of terrain databases as they are being created.

The RCAT test server provides management capabilities such as database registration, calculation of overlapping extents, coordinate transforms, and transfer of terrain database metadata between the database server and the user interface. The test server also calculates sample locations for performing correlation testing, and calculates the results of correlation tests based on the information provided by the database server. In the prototype RCAT system only the terrain elevation correlation testing capability is implemented. Finally, the test server processes visualization requests from the user interface, and returns VRML output from the database server to the user interface for visualization. The prototype implements a limited subset of the planned complete visualization capabilities, allowing for visualization of terrain, culture, and the results of terrain elevation correlation tests.

The RCAT database server parses the terrain database and returns information requested by the test server. For visualization, a VRML conversion facility is also provided for simple visualization. Current capabilities of the prototype database server are the ability to handle terrain data in OpenFlight and CTDB formats. The ultimate design goal is to incorporate the SEDRIS data model into the database server and to leverage the use of the SEDRIS APIs for the purpose of greatly increasing the number of different formats that can be parsed, and to include correlation tests for culture, line-of-sight, mobility, and other mission-critical attributes.

Outcome - A functional demonstration version of RCAT is available for demonstration and operation by all interested parties at:

http://www.vsl.ist.ucf.edu/groups/vtb/ZCAP/ist/rcat/rcat.html

D. Advances in Terrain Database Correlation Research

Goal - To advance understanding of issues directly pertaining to terrain database correlation in distributed simulation.

Discussion – Two efforts undertaken in this projects are best described under the general heading of correlation research. The first effort was to develop a new tool for terrain roughness analysis. The second effort was to perform research in the area of terrain surface polygonization algorithms. These efforts are described below.

1. Terrain Roughness Analysis

Our purpose for measuring terrain roughness is to classify and select portions of a terrain for terrain correlation analysis. In addition, measurement of terrain roughness is often used as a criterion for downsampling prior to terrain skin polygonization, and is important in the formulation of non-uniform stratified sampling schemes. Methods for classifying terrain roughness are also of interest to the tactical terrain analyst, and in the analysis and comparison of different approaches to digital terrain representation. One measure of roughness that is often used in optics and electromagnetic scattering theory is the correlation length of the surface at some particular scale. Three other measures of roughness that have been used are the sigma-t, the "roughness index" and the fractal dimension. The sigma-t is the standard deviation of the terrain height. The "roughness index" is a finite difference estimate of the average rate of change in slope. The fractal dimension is a real number that indicates how close a fractal is to a dimension.

The idea of measuring terrain surface as the standard deviation of the height (sigma-t) comes from the measure of surface microroughness. The surface height variations can be measured from a mean surface level by using profiling instruments. This is analogous to the calculation of the standard deviation. The problem with the use of the standard deviation to classify terrain roughness is that the slope contributes to the variation in height, that is, a smooth terrain in a slope can have a large standard deviation, and thus it may be classified as a rough terrain. Table III presents terrain roughness categories related to standard deviation originally used in the cruise missile program [13]. The terrain roughness classification is subjective, and depends on visual analysis for correctness. Terrain roughness classification depends on the measured extents. Over a relatively small area, a terrain surface can be classified as *smooth*. However, over a relatively large area including the same smooth region, the terrain may be classified as *rough*.

The process of determining the Sigma-t value involves sampling terrain elevation values and calculating the standard deviation. A flat terrain in a slope can be classified as non-smooth by using this process. Let us consider an inclined planar area (a square with a side parallel to the x axis). Uniform random sampling on this plane gives a uniform distributed data set with values between, say, a and b (minimum and maximum value of elevation, respectively). The standard deviation is then $(b-a)\ /12^{1/2}$. Depending on the value of a and b, the terrain roughness classification can be any one, from smooth to very rough. It also does not depend on the area of the region. Therefore, the standard deviation of terrain elevation is not always appropriate for measuring terrain roughness.

Table III. Terrain roughness categories.

Category	**Sigma-t (Standard. Deviation)**	
	Feet	*Meter*
Smooth	<60	<18
Moderate	60-200	18-61
Rough	200-800	61-243
Very Rough	>800	>243

Unlike the sigma-t, the fractal dimension is invariant on scale [14]. Fractal dimension is a real number that indicates how close a fractal is to a dimension. For example, a straight line has dimension 1, a polygonal line that almost fills a square has dimension close to 2, a plane has dimension 2, and a very rough surface that almost fills its bounding volume has dimension close to 3. A fractal has the property of preserving shape similarity under scale transformation (zoom). A fractal assumes a dynamic update in its form under scale transformation. Although fractals have been used to model terrain, to our best knowledge, a study mapping fractal dimension to the existent roughness classification based on sigma-t has not been done.

Figure 7 shows a small region of CCTT Primary II database: the terrain of 7.68 Km x 15.36 Km is subdivided in small squared areas (1.92 Km x 1.92 Km). Figure 7.b shows the corresponding sigma-t classifications. Regions that look flat are misclassified as *moderate* instead of *smooth*.

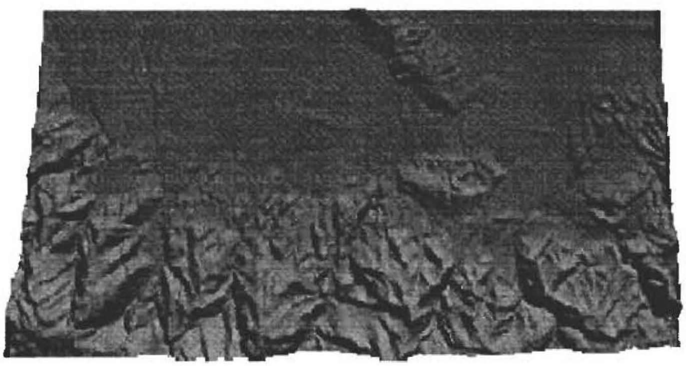


Figure 7. a. Shaded image of a portion of CCTT Primary II

b. Sigma-t value based on the standard terrain deviation

Legend: S for *Smooth*, M for *Moderate* and R for *Rough*.

To overcome this kind of misclassification, we propose an improved method that fits a plane to the terrain elevation sample points and then calculates the standard deviation of the fitting as the measure of sigma-t. The method uses multiple linear regression for fitting a plane to a set of sample points. From the plane equation, $ax+by+cz+d=0$, the fitting equation z is given by $z=Ax+By+D$, where $A=-a_{/c}$, $B=-b_{/c}$ and $D=-d_{/c}$, for $c\neq 0$. Multiple linear regression is the application of the least squares method and is formulated in matrix terms as follows:

$$Z = bX \quad (1)$$

The vector b is estimated by:

$$b=\left(X^T X\right)^{-1} X^T Z \qquad \text{Where} \quad b=\begin{bmatrix} D \\ A \\ B \end{bmatrix} \quad Z=\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix} \quad X=\begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ \vdots & \vdots & \vdots \\ 1 & x_n & y_n \end{bmatrix} \quad (2)$$

The standard deviation of the fitting is the square root of the Mean Square Errors (MSE). MSE is given by:

$$MSE = \frac{SSE}{n-3} \quad (3)$$

Where SSE is the Sum of Squares Errors, given by:

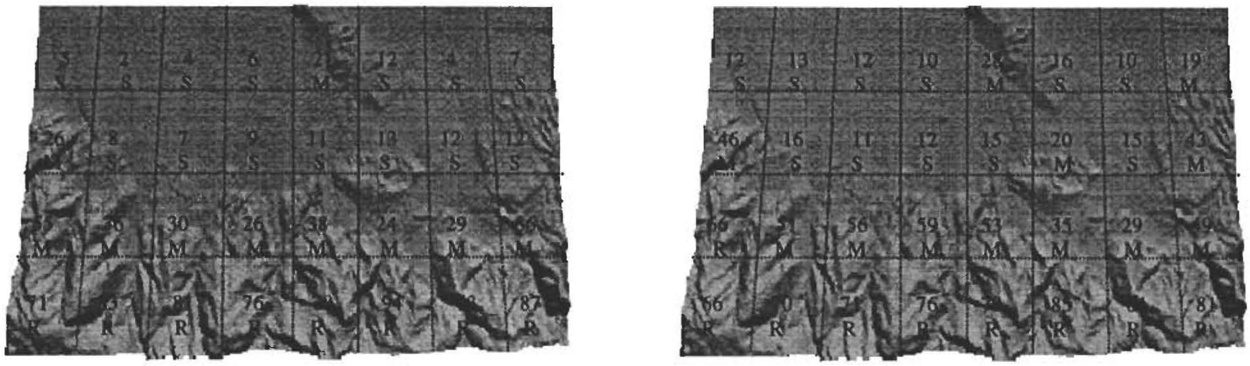
$$SSE = Z^T Z - b^T X^T Z \quad (4)$$

MSE is the expected value of the regression variance. The square root of MSE is an estimate of the standard deviation. Therefore, this method preserves the statistical significance of the sigma-t (the standard deviation).

Figure 8 shows the result of the application of the improved method on the piece of the terrain displayed in Figure 7. It shows a better match between the roughness classification and the terrain visual appearance. The reason why the original sigma-t method fails to properly classify this example is because portions of the terrain have non-zero slope. There is a significant reduction of elevation variation with the "removal" of the slope by the plane fitting.

The tool implemented in ZCAP for assessing terrain roughness subdivides the terrain into square regions. It also subdivides recursively each region into four square sub-regions, providing different levels of roughness classification. The top-level roughness could be a classification from a global view (pilot view) and the bottom-level one could be a classification from a local view (dismounted infantry view). The algorithm used to efficiently sample the terrain elevation is based on the grid method described elsewhere in this report. Figure9 shows roughness classification of more than half of a CCTT primary II database (89 km x 100 km).

In Figure 9.a, the terrain is subdivided in regions of 3.84 Km x 3.84 Km. In Figure9.b, each previous region is further subdivided in 4 subregions. An example of the differences that can occur between views at different scales can be observed in that the total area classified as smooth is larger at the smaller scale (Fig. 9 b).



a. Terrain roughness based on fitting method

b. Terrain roughness index based on slope variation.

Figure 8. Terrain roughness classification. (Legend: S for *Smooth*, M for *Moderate* and R for *Rough*.)

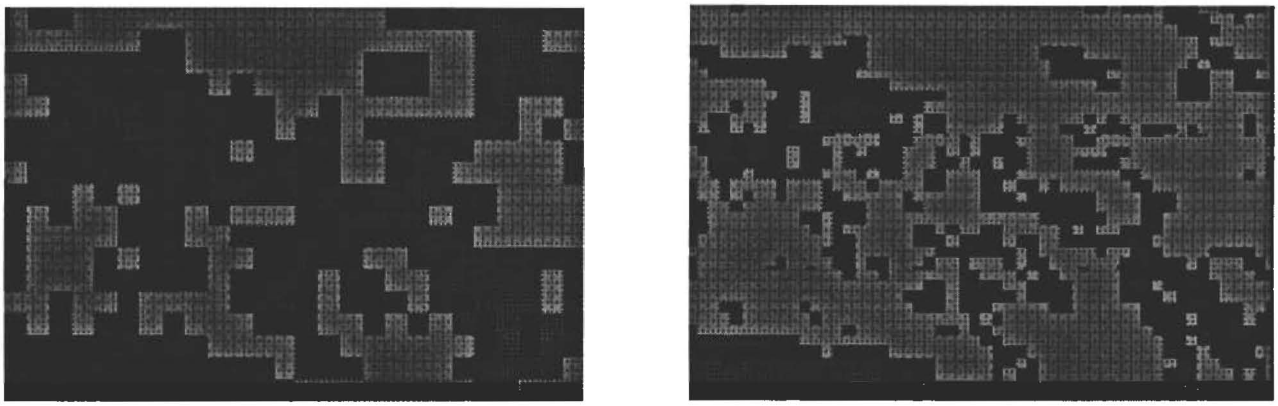
The roughness index is another approach that overcomes the problem of misclassification due to overall slope. The roughness index (RI) measure is based on an average of a norm of the second order gradient (∇^2) of the terrain elevation:

$$\left|\nabla^2 e\right| = \left|\frac{\partial^2 e}{\partial x^2}\right| + \left|\frac{\partial^2 e}{\partial y^2}\right| \tag{5}$$

Assuming a grid of *nc* and *nr* dimensions with spacing Δx and Δy, and using a centered finite difference approximation for the partial derivatives, the Roughness index can be given by:

$$RI = \frac{1}{nr \cdot nc} \sum_{i=0}^{nr} \sum_{j=0}^{nc} \left| \frac{2e_{ij} - (e_{ij-1} + e_{ij+1})}{(\Delta x)^2} \right| + \left| \frac{2e_{ij} - (e_{i-1j} + e_{i+1j})}{(\Delta y)^2} \right| \tag{6}$$

In practice, the value of RI is very small. We multiply RI by a factor, on the order of 10,000, to have values compatible with those of Table 2.1. Figure 2.4 shows a similar classification of that in Figure 2.3 using roughness index. The grid spacing was 60 meters. The roughness index is less subject to the scale-of-view problem, however, the RI value obtained varies with the grid spacing. It converges when the grid spacing is small. For practical purpose, it requires a larger amount of data and processing for accurate calculation. The sigma-t and the proposed MSE method do not present as much variation with sampling grid spacing, except for small variations due to the sampling process. Figure 2.2.b shows the roughness index classification for the same terrain piece of Figure 2.1, using 30-meter grid spacing. Some regions that appear almost flat were classified as moderate (RI of 19, 20). Because the roughness value varies with the grid spacing, the RI multiplication factor or the category ranges should be changed to obtain more consistent results.



a. Subdivision in 3.84 Km x 3.84 Km cells. b. Subdivision in 1.92 Km x 1.92 Km cells.

Figure 9. CCTT Primary II Terrain roughness classification. (Legend: Light Grey for *Smooth*; Grey for *Moderate* and Dark Grey for *Rough*.)

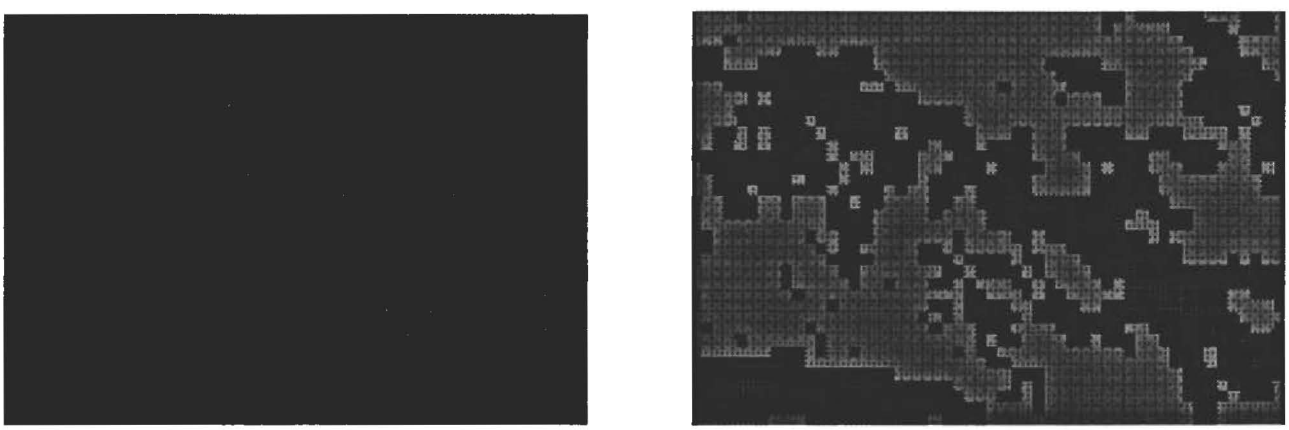


Figure 10. CCTT Primary II terrain roughness index classification, analogous representation of Figure 9.

Outcome - ZCAP4.1 includes a new tool for the analysis of terrain roughness that is free from the deficiencies and misclassifications of previous approaches.

2. Research on Terrain Polygonization Algorithms

The need for multi-resolution representations of terrain in simulator-based training applications arises from a variety of sources. In applications employing computer image generators (CIGs), a significant savings in the polygon count for a given scene is realized if objects and terrain at greater distances are represented at a lower level-of-detail (LOD). Man-in-the-loop simulators for ground-based platforms will require greater resolution than a fixed-wing aircraft simulator. The resolution required for a constructive simulation may differ significantly from that required by an interacting virtual simulation. In distributed simulations involving heterogeneous simulators, the terrain correlation between simulators employing differing terrain resolutions is of key importance to the successful achievement of the training objectives. The use of multi-resolution representations of the terrain provides a means of assuring registration at differing LODs, while providing an a priori measure of the correlation error between the LODs. Many different LOD generation methods have been developed over the past several years, most of which attempt to minimize the difference between the LODs based on some measure of the error between successive LODs. Examples of these methods include hierarchical triangulated irregular networks (TINs), iterative TINs, and wavelet-based methods. In this study we investigated one widely-used approach known as the iterative TIN (ITIN). We investigated the error criterion used to produce a "good" polygonization from a downsampled source grid. We investigated a new polygonization method based on a "reverse" ITIN approach. Numerical results are presented in terms of the inter-LOD terrain correlation for each method.

Definitions:

1. Elevation Error is the difference between the elevation data point and the generated approximated surface.

   $e=p.z-T.z,$

   where p is the data point and T=T(x,y) is the interpolated triangular point.
2. Sum of square error (SSE) is the sum of the square elevation error:

   $$\mathrm{SSE}=\sum_{i=i0}^{in}(p_i.z-T.z)^2$$

   where p_i is the data point, T=T(x_i,y_i) is the interpolated triangular point., $i0$ and in are initial and final indices respectively.
3. Mean Square Error (MSE) is the average of the square error;

   MSE=SSE/n
4. Root Mean Square Error is the root of the MSE

   $\mathrm{RMSE}=\mathrm{MSE}^{1/2}$
5. Mean Absolute Error (MAE) is the average of the absolute error:

   $$\mathrm{MAE}=\frac{1}{n}\sum_{i=1}^{n}|p_i.z-T.z|$$
6. Difference of sum of square error (DSSE) is the difference between the sum of square error in one instance of the triangulation and another instance of triangulation.

   $\mathrm{DSSE}=\mathrm{SSE}_{I+1}-\mathrm{SSE}_I$
7. Delaunay triangle is a triangle whose circle that circumscribes it contains no other points inside in the tessellation.

In the following two sections, we describe two basic algorithms to perform terrain data decimation. The first one is the Iterative Triangular Irregular Network (ITIN) proposed by Polis and McKeown [15]. The second one is a new algorithm that we named Least Square Error Iterative Triangular (LITIN). The first one starts the triangulation with few triangles and adds a new point refining the triangulation at each iteration. The second one does the reverse: it starts with the fully triangulated surface and removes a point at each iteration. These algorithms are useful for generating levels of detail in terrain databases. At this point, we do not consider preserving ridges, and important features for cover and concealment, which will be considered in future work. The objective of this study is to evaluate the algorithms in relation to error.

Iterative Triangular Irregular Network Generation

The basic ITIN algorithm is as follows:

1. Start the triangulation with 4 points of the terrain bounding box (alternatively boundary points);
2. Calculate absolute elevation error for each point.
3. Take the point that has the greatest error and add it into the triangulation, re-triangulating the surface.
4. Update the absolute value for those points whose triangle is modified;
5. Repeat the steps 3 and 4 while stopping criteria are not satisfied.

Our ITIN algorithm is simpler than that described by Polis and Mckeown [15]. It adds the vertex with the largest error magnitude to the working set of current vertices, then performs a retriangulation to include this new vertex, so that more regular distribution is obtained. We use Delaunay triangulation, because it is faster. We add a point inside or on the border of a triangle and change recursively the diagonal of the affected triangles. Once determined that a triangle is Delaunay, it will not be changed in the iteration. Thus, the elevation error can be updated for the points within this triangle. We support gridded elevation data as well as irregular point data. We use a grid method to organized the points and quickly access the points for error calculation .

As shown in Figure 10 and observed by Richbourg et. al.[16][17], the maximum elevation error does not decrease monotonically as well as the overall error such as the Root of the Mean Square Error (RMSE) as shown in Figure 2 (although it presents more monotonic behavior). Figure 10 shows a tracing of maximum elevation error at each iteration of a run of the algorithm, whose input was a small database with 2400 points. The first 200 points and the last 200 points are not shown. Figure 10 exaggerates the appearance the number of increases in maximum error due to the presence of too may points in axis x (there are more decreases than increases). As pointed out by Richbourg and Stone [17] the maximum elevation error is both insufficient and inconsistent as quality metric (stopping criteria). They argue that a later iteration, although better refined triangulation, can produce a higher maximum error. The ITIN approach intends to minimize the maximum error, but it does not. The RMSE and polygon budget seems to be better criteria. The RMSE is a metric of the overall error and the polygon budget is the ultimate goal to be satisfied.

Least Square Error Iterative Triangular Irregular Network (LITIN)

Although ITIN approach generates a triangulated surface with generally monotonically decreasing error, it does not produce a minimum error solution. We propose an approach towards the minimum error solution. Although the ITIN approach can produce quickly a good triangulated surface, it does not offer means to control the overall error. It is computationally expensive to select the next added point that provides the minimum error, and doing this eliminates the advantage of the algorithm. The reverse approach, that is starting from the fully triangulated representation and removing a point at each iteration, offers better control of the overall error. Our approach consists in removing the point that provides the least sum of error difference. Of course, for cases where the elimination of the majority of the points is the objective, this approach is more expensive, but even for these cases the gain in overall error reduction of the final terrain surface may make this approach superior for many applications.

The basic algorithm is as follows:

1. Triangulate all points.
2. For all points, calculate the difference of sum of square error (DSSE) caused by the removal of the point. This involves the removal of the edges connected to the point, and retriangulation of the affected area. We define error as the difference between the point and the triangulated surface. The sum of square error accounts the fitting error;
3. Take the point that provides the least sum of square error, remove all edges connected to it and triangulate the hole. We use Delaunay triangulation.
4. Update the DSSE for all points previously connected to the removed point;
5. Repeat steps 3 to 4 while a certain stopping criteria is not satisfied;

For DTEM, in a grid structure, full triangulation is trivial (right or left diagonal). Irregular points need more sophisticated algorithm. We use Delauney triangulation. because it is faster. There exist algorithms with O(NlogN) time that, in practice, run in almost linear time. Triangulation at each step reduces to triangulation of a polygon formed by the removal of the point (a hole). For each step, compared to the ITIN, updating the DSSE is an extra calculation. Thus the approach is more expensive, because calculations need to be done for all points adjacent to the removed point. For each point adjacent to the removed point, the approach is to remove all edges adjacent to it, re-triangulate the region and calculate the DSSE.

Figures 10 and 11 show the trace of a program run comparing LITIN and ITIN. It presents a run from 200 to 2200 points. The average RMSE reduction was 22.6%, the minimum was 14.2% and the maximum was 44% over the ITIN results. Also the maximum elevation error presents more stable monotonic behavior. The algorithm, by construction, at each step, minimizes the sum of the square error. At least the terrain surface has a local minimum error. Whether or not it produces a terrain surface with an optimal error solution has not been proved yet. This approach can minimize other metric (criteria) such as the maximum error. To do so, it is enough to select the point that provides the minimum of the maximum absolute error at each step.

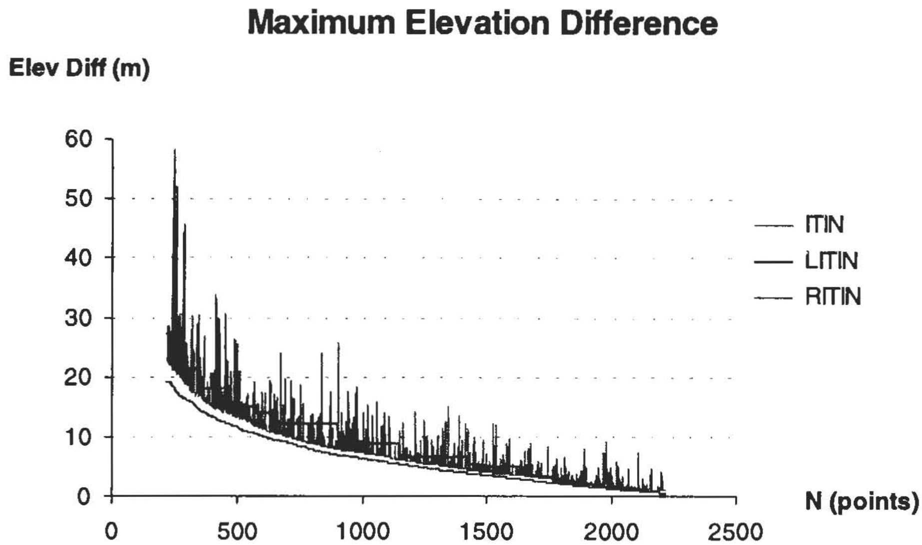


Figure 10. ITIN vs. LITIN preformance using the maximum elevation error criterion.

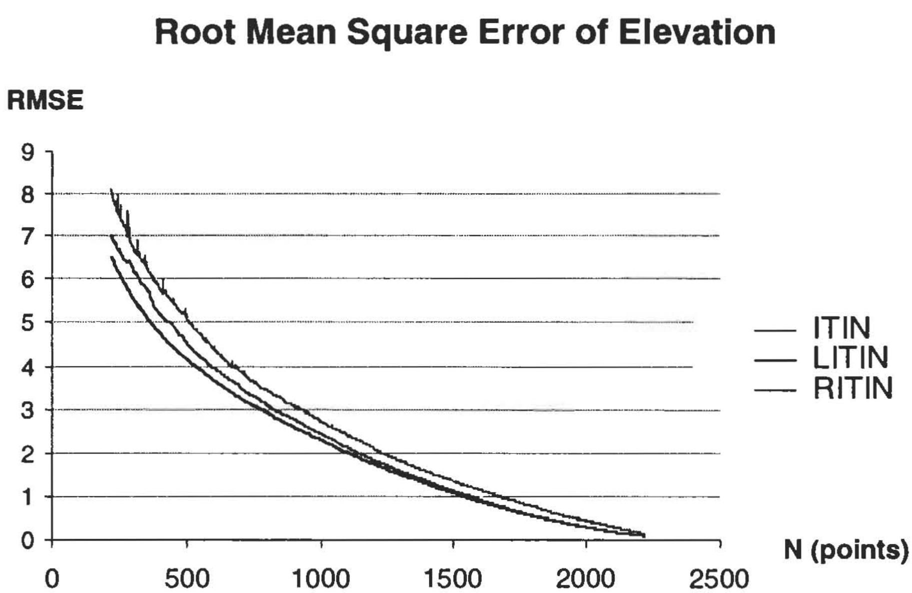


Figure 11. Root mean square error behavior comparison.

TABLE IV Numerical Results

Method	Input points	Points	Triangles	Max Error	RMSE
ITIN	37290	2404	4733	6.996448	2.451523
LSEITIN	37290	2404	4724	10.85095	2.010583
RITIN	37290	2404	4724	6.263910	2.25063
ITIN	37290	9600	19103	9.624524	1.130424
LSEITIN	37290	9600	19100	4.291404	0.946133
RITIN	37290	9600	19099	2.613794	1.021979

Outcome – We have developed a new and improved method for creating polygonized terrain surface representations from source data. Further improvements are possible by applying such principles as preservation of critical terrain features, and by formulating new error metrics to be minimized based on line-of-sight and mobility changes.

III Recommendations for Further Research

Recommendations for further research in areas pertaining to terrain databases in distributed simulation applications have been submitted to STRICOM in the form of a proposal entitled "Terrain Profiling for Interoperability and Tactics Using LOS Techniques". We recommend that STRICOM give serious consideration to funding this proposed work.

IV. References

[1] Guy A. Schiavone and Art Cortes, "ZCAP4.1 User's Manual", IST Technical Report IST-TR-98-12, September 1998.

[2] Guy A. Schiavone, Mohua Kar, and Art Cortes, "ZCAP4.1 Programmer's Documentation", IST Technical Report IST-TR-98-13, September 1998.

[3] Milton T. Sakude, Art Cortes, Kenneth C. Hardis, Glenn Martin, and Hector Morelos-Borja, " Improvements on Terrain Database Correlation Testing", 1998 Spring Simulation Interoperability Workshop, 98S-SIW-143, Orlando, FL, March, 1998.

[4] Milton T. Sakude, Guy Schiavone, Hector Morelos-Borja, Kenneth C. Hardis, Glenn Martin, and Art Cortes, "Recent Advances on Terrain Database Correlation Testing", presented at the SPIE Aerosense Conference, Orlando, FL, April 1998.

[5] K. C. Hardis and S. Sureshchandran: "Terrain Correlation Testing for DIS with ZCAP", Proceedings of 14th Workshop on the Interoperability of Distributed Interactive Simulations, pp. 975-986, 1996.

[6] G. A. Schiavone, Z. Chen and S. Kim: "Culture Correlation Testing for Terrain Databases in Distributed Interactive Simulation", Proceedings of 13th Workshop on the Interoperability of Distributed Interactive Simulations, pp. 289-294, March 1995.

[7] G. A. Schiavone and B. Graniela: "Minimizing Error Between Polygonized Surfaces by Vertex Adjustment Using Constrained Least Squares Fitting: "Terrain Correlation Testing for DIS with ZCAP", Proceedings of 14th Workshop on the Interoperability of Distributed Interactive Simulations, pp. 987-994, March 1996.

[8] G. A. Schiavone, R. A. Nelson and B. Goldiez: "Statistical Certification of Terrain Databases", Proceedings of the 16th Interservice/Industry Training Systems and Education Conference, paper no. 4-9, Dec. 1994.

[9] M. J. Laszlo, Computational Geometry and Computer Graphics in C++, Prentice-Hall, 1996.

[10] M. D. Petty, C. E. Campbell, R. W. Fransceschini and M. H. Provost: "Efficient Line of Sight Determination in Polygonal Terrain", Proceeding of Image VI Conference, pp. 29-253, July 1992.

[11] C.C. Ashcraft, R. G. Grimes, D. J. Pierce and D. K. Wah, The user manual for SPOOLES: Release 1.0: An Object Oriented Software Library for Solving Sparse Linear Systems of Equations, Boeing, Seattle WA, 1997.

[12] G. H. Golub and C. F Van Loan, Matrix Computation, The Johns Hopkins University Press, London, 1989.

[13] J. R. Ackeret, "Digital Terrain Elevation Data Resolution and Requirements Study", Interim Report ETL-SR-6, U.S. Army Corps of Engineers, November 1990.

[14] G. Cherbit, Fractals Non-integral Dimensions and Applications, John Willey & Sons, Chichester, 1991.

[15] M. F. Polis and D. M. McKeown, "Issues in Iterative TIN Generation to Support Large Scale Simulations", Proc. AutoCarto 11, pp. 267-277, 1993.

[16] R.F. Richbourg, C.K. Kay and L.W. Campbell, "Tactical Operations Planning as a Metric for Terrain Fidelity", Paper # 95-13-012, Proceedings of the 13th DIS Workshop, Orlando, Florida, Sept, 1995.

[17] R.F. Richbourg and T. Stone, "Triangulated irregular network (TIN) representation quality as a function of source data resolution and polygon budget contraints", SPIE proceedings Vol. 3072, April 1997, pp. 199-210.