Algorithms for Run-Time Terrain Deformation

SBIR Phase II Proposal

SBIR Phase I Contract No. FA8650-04-M-6503
Topic No.: AF04-064

Prepared for

Air Force Research Laboratory
Attn: Sabrina Davis, Program Manager
AFRL/HE SBIR Program Office
Wright-Patterson AFB, OH 45433

Prepared by

Computer Graphics Systems Development Corp.
2483 Old Middlefield Way #140, Mtn. View CA 94043
Tel: 650-903-4920 Fax: 650-967-5252

ABSTRACT

For Air Force mission rehearsal in flight simulations, newly acquired target data may not match pre-existing terrain database that normally requires lengthy off-line processing. A run-time solution is proposed to correct the mismatch while maintaining the continuity of terrain and overall aspects of the database. The solution proposed is a correction function c(x,y) that is added to every terrain vertex in view. The corrections are made when each area block is retrieved, with the corrections to the adjacent area blocks considered to ensure continuity across block boundaries. To foster ultimate application with all of the existing real time software for simulation systems, the module will be developed for OpenSceneGraph and placed in the public domain. The delivered module will provide the capability of assembling the database in real time from terrain grid posts, texture patterns, conventional objects, and target objects. This will provide new capabilities for OpenSceneGraph as a standalone open source application, as well as provide reference code for incorporation into proprietary run time environments.

KEYWORDS

Training Simulation, Visual Simulation, Image Generation, Run-Time Terrain Generation, Run-Time Terrain Deformation, Real-Time Terrain Deformation

1. Identification and Significance of the Problem or Opportunity.

The growing and changing simulation and training requirements of the Air Force is imposing demands on simulation database accuracy and timeliness. It is becoming less acceptable now, in a 3D visual database, to have roads floating in the air, or a building partially touching the ground or misplaced geographically, especially when a pilot is using a flight simulator for mission rehearsal. The old way of image generation technique is elevation database first, then culture or topographic data as add-on. Often, due to unevenness of the terrain skin, the modelers or database specialists had trouble ensuring river not flowing upwards, or buildings not floating in the air. The 3D position accuracy of topographic features is of greater importance to pilots.

For mission rehearsal, <u>accurately</u> and <u>timely</u> development of databases is becoming more and more urgent. For example, in some air-to-ground mission, pilots need certain visual cues on the ground to perform. In some weapon delivery modes, pilots stress the importance of knowing exactly where the target is in relation to terrain features and man-made features.

Traditionally, the terrain surface has been accepted as "truth" and features have been forced to conform to the terrain surface. The features are changed in elevation to conform to the surface. However, the locations of key features related to targeting may be known with greater accuracy than the underlying terrain surface. The target features are likely to have been measured specifically with the latest instrumentation, whereas the terrain surface may be of older and less accurate origins. In such circumstances it makes sense to modify the terrain to conform to the features, rather than the reverse as has been traditionally done.

The modification of the terrain to match the features could be accomplished in non-real-time as part of the database preparation process. However, target data may be time-critical so that the off-line processing is unacceptably time consuming. In addition, due to legacy equipment requirements, it may be desirable to run the same feature data with a variety of terrain databases. A real-time deformation of the terrain provides the solution.

Part of the problem is to derive a solution that is widely applicable. We would like the solution to work on a variety of simulation platforms. The run-time terrain deformation code will operate within the real time software for the system, and there are more than twenty real time software environments currently available. This suggests that source code should be available in kit form for easy application. In keeping with present trends toward using open source for easier maintenance, development of the module for OpenSceneGraph becomes the preferred path.

From the viewpoint of algorithm development, the problem is to supply a correction function $c(x, y)$ that is added to every vertex in view. The correction function must (1) adjust the terrain surface to meet the specified features, (2) appear smooth and continuous so that the adjustments appear natural, and (3) do not distort aspects of the database that must be preserved. An example of (3) is that the terrain under a building must match the base of the building locally even though globally the terrain is deformed to match the elevation of selected features. Deformations of the terrain cannot be allowed to introduce new errors of terrain failing to match features.

Deforming the terrain smoothly to match features conjures up visions of smoothly curved surfaces being added to the terrain. Smoothly curved surfaces, however, generally require many polygons to appear smooth. While modern image generators have prodigious capacity, we still need to be careful about grossly increasing the polygon counts. Algorithms should only add polygons as necessary. Our solution is to add the corrections only to the pre-determined grid points. New polygons are required to fit the objects to the surface, but that is required whether the terrain is deformed or not.

Phase I Results

At this point in Phase I, we: (1) studied the hardware and system requirements for real time implementation, (2) studied existing real time software systems for the implementation of a run-time terrain deformation algorithms, (3) studied existing terrain algorithms/techniques in search of applicability, (4) designed a new solution to the run-time terrain mismatch problem, and (5) we are in the process of prototyping the solution for a quick demo.

The basic algorithm applies a smooth Gaussian (a*exp[-b*r^2]) correction function to each vertex to be deformed. The constants a,b are selected to provide the right total correction and to control the slope of the correction to keep a natural appearance. Corrections are allowed to extend into each of the eight area blocks that surround a particular area block, but not beyond those adjacent blocks. That allows for continuity across block boundaries while at the same time ensuring that only the immediately surrounding blocks need to be retrieved before the correction is made. We don't want to have to retrieve the whole database to make the corrections to part of it because that would be inconsistent with real time operation.

Correction Software Kit

The Correction Software Kit, at the end of Phase I, is a simplified version of the algorithm. It is simplified to allow demonstrations of the algorithms in operation in real time. Simplifications include keeping a single level of detail on a gridded terrain with a very small number of types of features.

A quick demo will be available for downloading the correction software kit, compile and build into user's run-time library for terrain generation.

Open Source Website

We have started a website (http://cgsd.com/AF04-064} for the posting of released open source code for run-time terrain deformation in Phase I. Currently some interim results of Phase I are posted on the site. This website will be maintained during the performance of Phase II for the posting of the full-scale run-time correction function source code.

2. Phase II Technical Objectives

The Phase II Technical Objectives are:

1. To develop the algorithms/techniques identified in Phase I to perform the real-time terrain deformation function.

2. To verify the compatibility of the run-time terrain deformation with existing commercial IG hardware through demonstrations
3. To develop and publish for public use an open source run-time terrain deformation software suite that works with OpenSceneGraph.

3. Phase II Work Plan

3.1 Approach

<u>*Algorithm Requirements*</u>

From the problem analysis in Section 1 above we know the algorithm requirements. The run-time terrain deformation algorithm must (1) move the terrain surface to match the features, (2) work in real time, (3) deform the terrain smoothly, (4) not add too many polygons, and (5) not induces any new mismatches.

<u>*Need for New Algorithm*</u>

Currently, there are many algorithms or techniques touch some aspects of the subject study, but none as we can see present a complete solution. A new algorithm or technique will be needed. So far, we see potential in several existing algorithms or approaches, for example: (1) Real-time Optimally Adaptive Mesh (ROAM) approach, (2) voxel-polygonal hybrid rendering technique, (3) dynamic terrain (DT) generation method, and (4) block-based terrain walkthrough method that claims to be improvement version over ROAM, View-dependant or Continuous LOD (Level of Detail) methods.

<u>*Run-Time Terrain Deformation Algorithm addresses two Problems*</u>

There are two aspects of the problem that need to be addressed: (1) run-time aspect, and (2) deformation aspect. The terrain deformation can be implemented during run-time so long as the algorithm allows the database to be modified block-by-block without considering the database as a whole, something our algorithm accomplishes. Some alternative approaches examined, like the voxel-polygonal hybrid rendering technique cannot perform at run-time. ROAM approach is yet to be proved viable for run-time implementation.

<u>*Potential Problems created by the Correction Function*</u>

When vertices are moved or added, the impact on **level-of-detail switching** and **texture projection** must be taken into account. These are not difficult theoretical problems, but they impact design complexity.

Also, the number and the types of features, if it is too large, might become a problem for **real-time rendering**.

In summary, we can dissolve the potential created problems by properly processing the following parameters in the design process: (1) multi-level of details, (2) increased number and types of features, (3) various sizes of terrain block, (4) limit of added polygon counts, (5) ordering of the corrections for each vertex with respect to each feature, and (6) spatial size of a feature.

Target Format

The target data to which the terrain must conform will be provided in a SHAPEfile or similar format. SHAPEfile is an ASCII character format used to define terrain features. It is defined by a white paper http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf.

The described target data will be of five types:

- Point features such as individual buildings or towers
- Linear features such as roads and pipelines
- Areal features such as the area of a town or building complex
- Ridge lines, which must be guaranteed to be the local maxima along the length
- Valley definition lines, which must be guaranteed to be local minima along their lengths

In each case, the terrain must be dynamically modified or corrected to fit the SHAPEfile data.

Coordinate System

For the present project the shape of the earth is assumed to be the World Geodetic System 1984 (WG84) ellipsoid. The ellipsoid is a smooth mathematical surface that best fits the shape of the geoid and is the next level of approximation of the actual shape of the earth. Each point or vertex is represented in lat, long, and height.

Project Tools

We have selected OpenSceneGraph and C++ for the project. These tools will make the objective of open source code easier. Robert Osfield and Don Burns create OpenSceneGraph (OSG). C++ is a standardized language.

Simplified vs. Full-Scale Algorithm

In order to allow demonstration of the terrain deformation algorithm in operation in real time, in Phase I we made simplifications include keeping a single level of detail on a gridded terrain with a very small number of types of features. In Phase II, the full-scale algorithm will be examined in terms of timeliness and accuracy before the full implementation of the algorithm will be coded and tested.

One example that depicts well the substantial complexity in a full-scale algorithmic implementation can be observed: When there is only one level of detail, a block of terrain is retrieved and modified or corrected. But when there is more than one level of detail, a block of terrain is retrieved at its lowest level of detail, and the modifications and corrections must be made for all levels of detail. In the TerraPage format, for example, there is a set of buffer strips that serve to match the data in a block at level of detail n with adjacent blocks that are at either level n-1 or n+1. Keeping all the modifications consistent is as much an implementation challenge as a theoretical one.

3.2 Detailed Approach

1. Verify the selected algorithm for run-time correction of terrain rendering based on elevation of topographic features in terms of timeliness

2. Verify the selected algorithm for run-time correction of terrain rendering based on elevation of topographic features in terms of accuracy
3. Design, code and test the selected algorithm with OpenSceneGraph in C++
4. Develop a web site, as an open forum among simulation and Air Force contractors, for posting the open source code of the prototype developed for run-time terrain deformation
5. Examine the compatibility of the open source code with the existing IG hardware
6. Demonstrate the prototype

The real time terrain deformation must be performed as part of a real time software package that retrieves data and displays it for the simulation. The approach we proposed is based on using the OpenSceneGraph real time software, and that approach was validated in our discussion. The advantage of OpenSceneGraph is that the source code is open and may therefore be used or maintained by any potential future supplier to the Air Force. Our intention for the present project is to similarly make the source code open, so that it may be used with OpenSceneGraph, or incorporated by a vendor of a proprietary package if the vendor is so inclined.

The open source approach has the advantage of promoting competition by ensuring that legacy code is readily available. It also allows software maintenance to be performed independent of any particular procurement. The OpenSceneGraph (OSG) web site is http://openscenegraph.sourceforge.net/. The site links downloads of the OSG source and a number of tutorials.

A limitation of OSG is that it is currently narrowly oriented, as the name implies, towards the display of the visual scene. A complete simulation usually requires other functions, such as sound, instructor's station controls, and mission functions (such as collision testing, radar altitude determination, and line-of-sight testing). Those functions must therefore be implemented separately when they are required.

The current version of OSG does support retrieval of terrain in the MPI OpenFlight (http://www.multigen.com/) and related Terrex TerraPage (http://www.terrex.com/www/index.htm) formats. In the kickoff meeting, we agreed that the OpenFlight and TerraPage formats would be used in the current project.

The target data to which the terrain must conform will be provided in a SHAPEfile or similar format. SHAPEfile is an ASCII character format used to define terrain features. It is defined by a white paper http://www.esri.com/library/whitepapers/pdfs/shapefile.pdf.

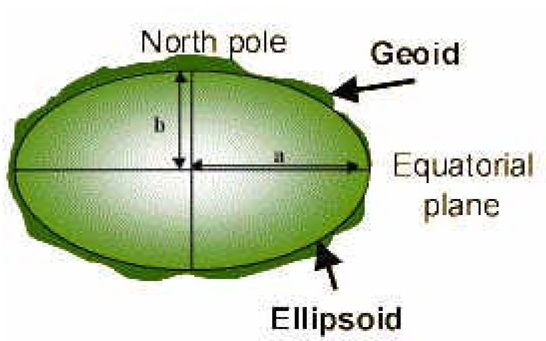
The described target data will be of five types:

- Point features such as individual buildings or towers;
- Linear features such as roads and pipelines;
- Areal features such as the area of a town or building complex;
- Ridge li.nes, which must be guaranteed to the be local maxima along their lengths; and
- Valley definition lines, which must be guaranteed to be local minima along their lengths

In each case the terrain must be dynamically modified to fit the SHAPEfile data. A question remains as to how the data representing the target itself should be obtained. For example, the terrain surface for a town or building complex may be specified by the SHAPEfile, but how should the three-dimensional buildings themselves be represented? For the time being we will assume that they are in the original database and will be adjusted so as to conform to the new terrain elevation contours. Note for reference that CGSD has available a number of software functions for automating generating building from a SHAPEfile of the building outlines.

For the present project the shape of the earth is assumed to be the World Geodetic System 1984 (WGS84) ellipsoid. This is described, for example, in http://w3sli.wcape.gov.za/Surveys/Mapping/wgs84.htm as follows.

The ***ellipsoid*** is a smooth mathematical surface that best fits the shape of the geoid and is the next level of approximation of the actual shape of the earth.



Elements of an ellipse

a = Semi Major Axis

b = Semi Minor Axis

f = Flattening = (a-b)/a

PP' = Axis of revolution of the earth's ellipsoid

Ellipsoid	a	b	Unit	Used
Mod. Clarke 1880	6378249.145	6356514.967	International meters	R.S.A., Botswana, Zimbabwe
WGS 84	6378137.000	6356752.314	International meters	Globally
Bessel	6377397.155	6356078.963	German Legal meters	Namibia
Clarke 1866	6378206.400	6356584.467	International meters	Mozambique

Project Tools

The Phase I objectives related to proving the feasibility of algorithms to naturally blend the new target data with existing terrain. However, ultimately the practical implementation in Phase II will be as important as the algorithms. Blocks of terrain must be modified after they are retrieved, with modifications made in a way that is not distracting. The retrieved blocks are at the lowest level-of0-detail, and the modifications must be established for all the levels. In the TerraPage format there is a set of buffer strips that serve to match the data in a block at level of detail *n* with adjacent blocks that are at either level *n-1* or *n+1*. Keeping all the modification consistent is as much an implementation challenge as a theoretical one.

Since the goal is ultimately to distribute the source code, more than usual care ought to be paid to writing understandable and modifiable code. OSG is written in C++ as will be our modifications and extensions.

OSG was developed by jointly by Robert Osfield and Don Burns. Osfield is now in academia in Scotland, and Burns provides consulting services through his company, Andes Engineering, http://www.andesengineering.com/index.html. Burns teaches 3 and 4 day courses on site, and he also teaches shorter versions at various professional forums. This year, we took the two 4-hour tutorials he taught in conjunction with the Image Society. They were well taught and quite helpful. Burns office is in Sunnyvale, a few miles from the CGSD Mountain View office. We believe he would be available for consultation in Phase II as the need arises.

We also recently took a C++ refresher course, out of concern for the C++ source code quality for our mods. The course we took was a commercial five day course presented by Hands On Technology Transfer, Inc., a company that specializes in intensive high tech courses. (http://www.hott-software-training.com/). It was good course for experienced programmers who are not especially familiar with C++. I recommend it for that purpose.

C++ is a standardized language, so any of the half dozen or more C++ compilers can be used with OSG. In the windows environment, Microsoft's Visual C++ version 6.0 has been the most common development environment for the past five years or so. Last year Microsoft replaced 6.0 with Microsoft C++.NET. The .NET version has non-standard Microsoft extensions for networking which are generally not need for visual simulation projects, however they do no harm and all of the regular C++ standard support is provided. The .NET compiler is inexpensive ($109) and supports C as well as C++ programming. Microsoft also has a new language called C# (C sharp). It is not standardized, so it is best avoided for open-sourced projects like ours.

Database Organization

The main activity of the past month has been designing the test implementation of the algorithms.

No matter what algorithm is used for terrain deformation, it is always necessary to know what objects in the database are terrain, which objects are supposed to sit on the terrain (the features), and which, if any, objects are otherwise positioned. The latter category includes clouds, moving objects, and special effects like smoke. We must therefore make some assumptions about how we will be able to tell these classes of objects apart.

There is a trend in database design to manage the features separately from terrain. E&S may have originated the concept with the use of what they call *themals* placed in hardware. In the themal concept the terrain is stored as grid posts and groups of features characteristic of different environs are stored separately. In real time, the E&S hardware triangulated the terrain and repeatedly positioned instances the groups of features on to the terrain.

Subsequently, Lockheed implemented a similar scheme entirely in software, and E&S has moved its implementation from hardware to software, and now runs it on a PC platform as front end to the rendering hardware. Terrex and Terrasim implement similar layered approaches to building databases off line for later retrieval in real time.

E&S's original motivation included efficiency of reusing of the groups of features on different terrain types without having to store each instance of them in their different positions. However, the more general attraction includes the ability to change terrain and features independently without having to manually readjust the fit of the features to the terrain. The burden is shifted to the software that does the placement.

The placement software faces a number of problems. For example:

- Modifications to the terrain might introduce odd bank angles to roads
- Similarly, terrain changes might make lakes no longer level
- Both terrain and feature modifications may make features no longer sit squarely on the terrain.

With reference to Figure 1, the adopted solutions usually involve local modifications to the terrain surface. The terrain is cut to level the roads, stretched to restore lakes to flatness, and made level under buildings. Trees may be just slightly buried into sloped terrain with the z-buffering left to sort out occlusion. (Alternatively, a tree base might be made to just touch the surface. Sometimes, that would suppress some aliasing.)

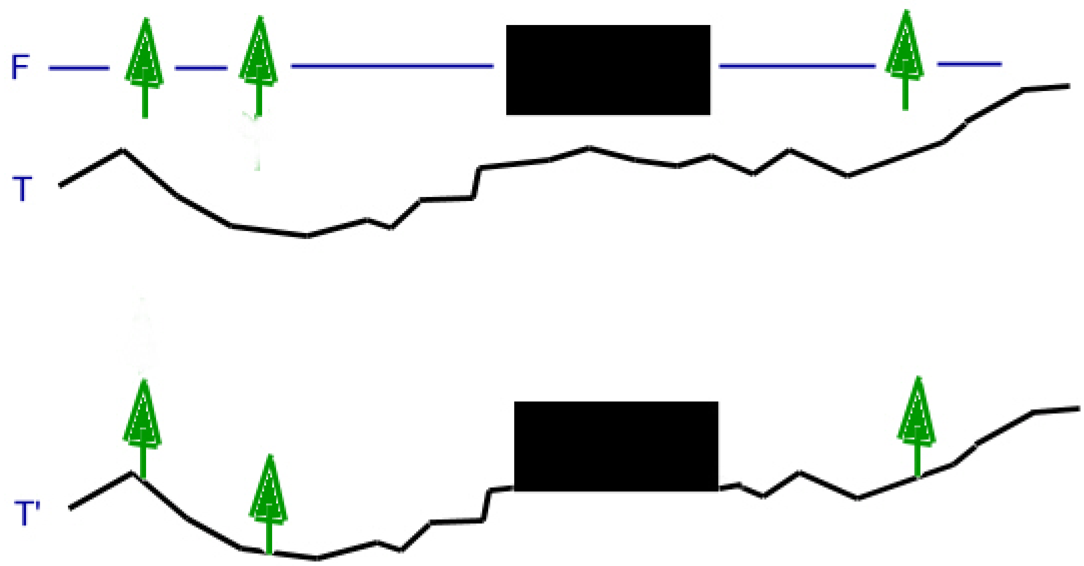


Figure 1. In a popular database approach, a layer of features is placed using local terrain modifications

Since this approach is popular, and we think it makes good technical sense, we would like to design our real time terrain deformation algorithm to be compatible. Referring to Figure 2a, we start with the measured target objects F1 (the red boxes). At the start the previously constructed terrain T1 has not be deformed to fit the objects, so the objects may be above or below the terrain. (Note that we assume the terrain grid posts have been converted into a surface.)

The first processing step is then to deform the surface T1 into a conforming surface T2 (Fig. 2b). The technical proposal gives an algorithm for that; basically the terrain is smoothly transitioned into matching the positions of the measured objects. We might then leave the objects placed on the terrain, however we might just put them in with the other terrain features (Fig 2c) to form a new feature list F2 that includes all of the other terrain features in the database. We can then let normal processing place both the measured objects and the rest of the features (Fig 2d).

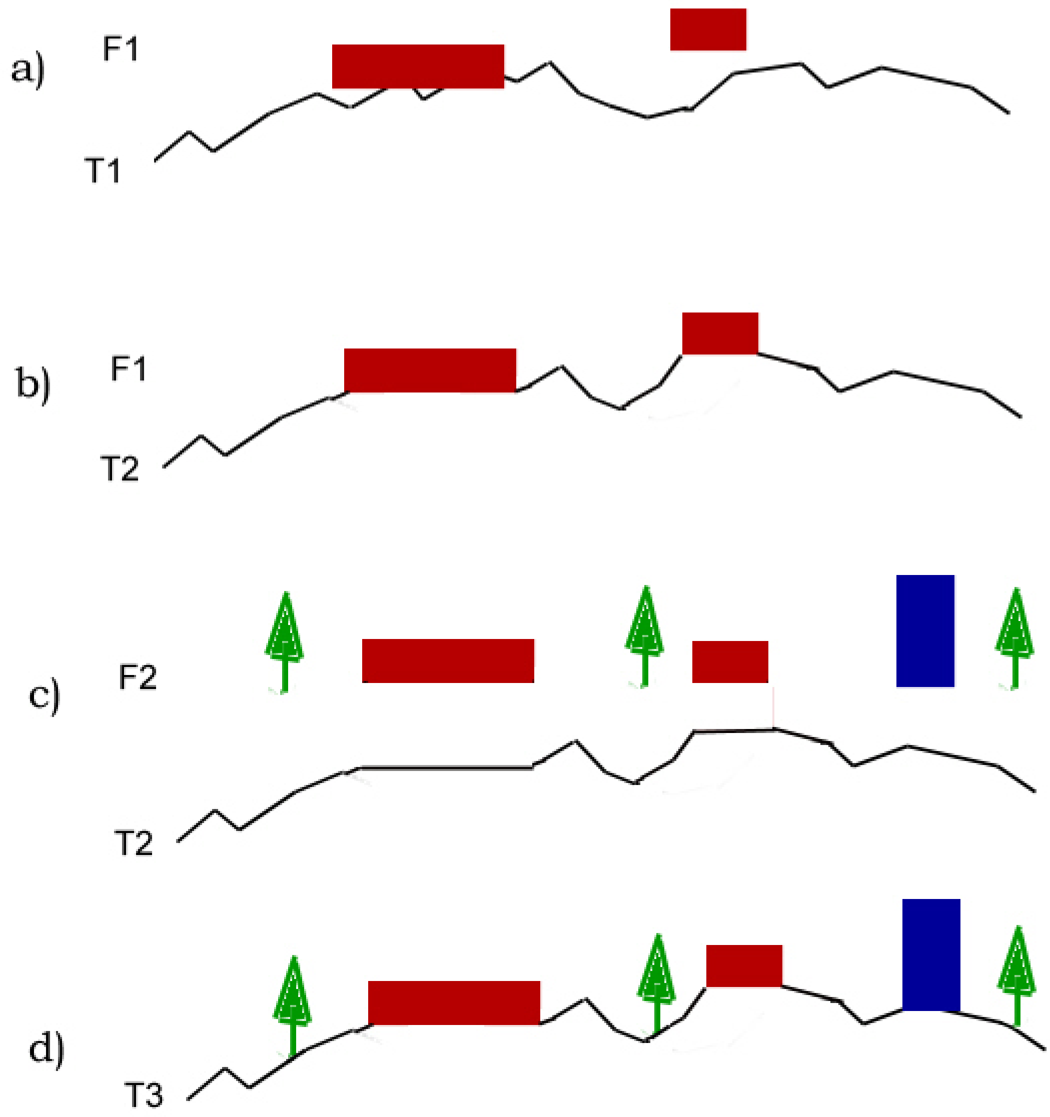


Figure 2. The run time terrain deformation can be accomplished by changing the terrain and then letting the features be attached conventionally.

The normal feature placement process will discover that the measured objects fit the terrain surface perfectly, so that no local deformations have to be performed to accommodate them. That preserves there precise placement with matching terrain. All of the other features will be placed using local terrain deformations to match them to the terrain. The placement of local objects will be different than it would have been if the terrain had not been deformed first to match the measured objects, but exactly the same computer code will work to attach them to the terrain.

There are two advantages to doing the run time terrain deformation as described. First, it isolates the terrain deformation into a single module that will be more easily added to an existing run time environment. The simplicity will encourage vendors to install the open source code we are giving them. A second advantage is that we can rely more heavily on existing run time software to do some of the local modifications to terrain needed to fit the features to the terrain surfaces. If we were to move all the features on the surface when we deformed the terrain, we would have to provide duplicate mechanisms for fitting them to the surfaces, included, no doubt, many special cases. Instead, we can rely upon the existing software to handle the oddities.

An example of a special case is a bridge that connects a road between two nearby hills. If we deform the terrain on one or both of the hills, the bridge must still stay level and still connect the roads on the terrain. Those sorts of special cases have been worked out in existing software, and we want to avoid solving those problems again.

Our deformation algorithm applies a smooth Gaussian (a*exp[-b**r*^2]) correction function to each vertex to be deformed. The constants a,b are selected to provide the right total correction and to control the slope of the correction to keep a natural appearance. *R* is the distance from the nearest point for which correction was required by the target object. Corrections are allowed to extend into each of the eight area blocks that surround a particular area block, but not beyond those adjacent blocks. That allows for continuity across block boundaries while at the same time ensuring that only the immediately surrounding blocks need to be retrieved before the correction is made. We don't want to have to retrieve the whole database to make the corrections to part of it because that would be inconsistent with real time operation.

3.3 Work Plan

We plan to use four labor types to accomplish the objectives of the program. The labor types and the associated labor skills are as listed.

Table 1. Description of Labor Type Codes

LaborType Code	**Skill Level**
A1	Principal Investigator
B2	Senior Software Engineer
C2	Graphics Specialist
D2	Software Engineer

Based on the approach we had just described, we provide a top-level task breakdown, and an estimate in hours for each task. A schedule is also provided later in this section.

3.4 Task Descriptions

Task-1 Research and verify the timeliness of the full-scale algorithm/technique in terrain deformation: In Phase I, an algorithm has been designed and will have been roughly prototyped to demonstrate feasibility. However, in Phase I, the algorithm is refined for reliable operation in real time. In Phase II, the first task is to verify the timeliness of the full-scale algorithm for run-time rendering. Also, since we plan to provide the open source code on the web, and users can download the code to compile into their run-time library, we need to examine the run-time aspect of the software. In summary, this task includes examining and designing implementation of: (1) multiple levels of details, (2) increased number and types of features, (3) various sizes of terrain block, (4) limit of added polygon counts, (5) ordering of the corrections for each vertex with respect to each feature, and (6) spatial size of a feature. Basis of estimate: Engineering estimate based on previous SBIR projects.

Task-1 Direct Labor Estimate (hours)

LaborType	A1	B2	C2	D2
Est. Effort	400	320		320

Task-2 Verify the accuracy of the full-scale algorithm/technique in terrain deformation: This task includes the deformation aspect of the algorithm, i.e. the study on terrain continuity and maintenance of overall terrain aspects. Basis of estimate: Engineering estimate based on previous SBIR projects.

Task-2 Direct Labor Estimate (hours)

LaborType	A1	B2	C2	D2
Est. Effort	400	320	80	400

Task-3 Design, code and test the full-scale algorithm: In Phase I, the simplified code is designed and developed with OpenSceneGraph in C++. This task is to take Phase I result, with modifications or enhancements derived from Task-1 and Task-2, to develop a full-scale software kit for run-time terrain deformation. Basis of estimate: Engineering estimate based on previous SBIR projects.

Task-3 Direct Labor Estimate (hours)

LaborType	A1	B2	C2	D2
Est. Effort	400	320	80	400

Task-4 Develop a web site for the release of open source code: The website, as an open forum among simulation and Air Force contractors, for posting the open source code of the prototype developed for run-time terrain deformation. This task includes maintenance of the web site during the 24 months performance period of the project, in addition to the web design, download instruction, run-time library construction guideline, and released code posting. Basis of estimate: Engineering estimate based on previous SBIR projects.

Task-4 Direct Labor Estimate (hours)

LaborType	A1	B2	C2	D2
Est. Effort	400	320	80	400

Task-5 *Examine the compatibility of the open source code with the existing IG hardware*: This task includes the compatibility tests using several primary IG hardware. Engineering estimate based on previous SBIR projects.

Task-5 Direct Labor Estimate (hours)

LaborType	A1	B2	C2	D2
Est. Effort	400	320	80	400

Task-6 *Demonstrate the prototype*: This involves the process starting with (1) download from the www.RunTimeTerrainDeformation.com website, (2) compile the source code into part of run-time library, (3) select a database, (4) select a set of targets, (5) run the terrain rendering software with the correction function embedded. Correct and modify instructions and/or source code if necessary. Basis of estimate: Engineering estimate based on previous SBIR projects.

Task-6 Direct Labor Estimate (hours)

LaborType	A1	B2	C2	D2
Est. Effort	400	320	80	400

Task-7 Direct the project and Write Bi-Monthly Status Reports: Each month CGSD will prepare *a technical status report* detailing the results for the month including the percent of completion for each task, a summary of the current status, any significant problems encountered, interim results, if any, and a summary of plans for the next reporting period. Also, *a progress chart*, and *a spreadsheet for fund expenditure*. CGSD has been known for the clear style and thoroughness of our SBIR reports. Basis of estimate: Engineering estimate based on previous SBIR projects.

Task-7 Direct Labor Estimate (hours)

LaborType	A1	B2	C2	D2
Est. Effort	400	100		

Task-8 Write a Final Report: This task involves preparation of a final report for the Phase I study effort. In this report, we will discuss our findings, the pros and cons for each considered approach relative our rationale for the recommended approach, design and development results, and document feasibility-specific details. Basis of estimate: Engineering estimate based on previous SBIR projects.

Task-8 Direct Labor Estimate (hours)

LaborType	A1	B2	C2	D2
Est. Effort	100	90		

Table 2: Summary of total labor required to complete project
Estimated Effort (hours)

		A1	B2	C2	D2
Task-1		400	320		320
Task-2		400	320	80	400
Task-3		400	320	80	400
Task-4		400	320	80	400
Task-5		400	320	80	400
Task-6		400	320	80	400
Task-7	Ongoing	400	100		
Task-8		100	90		
Total Labor Estimate (hours)		2900	2110	400	2320

3.5 Major Milestones

Milestone	Months ARO
Milestone-1: Complete verification of selected solution and preferred real time software environment	4
Milestone-2: Complete Development of correction function coding and test	12
Milestone-3: Complete applicability tests for three commercial IG hardware	20
Milestone-4: Demonstrate the prototype	23
Milestone-5: Complete Final Report	24

3.6 Task Schedule

Months After Receipt of Order (ARO)

ID\Month	1	3	5	7	9	11	13	15	17	19	21	23.. 24
Task-1	▲	▬	▼									
Task-2	▲	▬	▬	▼								
Task-3		▲	▬	▬	▬	▬	▼					
Task-4		▲	▬	▬	▬	▬	▬	▬	▬	▬	▬	▼

Task-5							▲	▬	▬	▬	▼	
Task-6										▲	▬	▼
Task-7	▲	▬	▬	▬	▬	▬	▬	▬	▬	▬	▬	▼
Task-8											▲	▬ ▼

3.7 Location of Work

All of the work will be carried out at CGSD facilities in Mountain View, CA.

3.8 Deliverables

For Phase II, the deliverables for the program are

1. Twelve bi-monthly status reports on technical progress and findings
2. Establishment of a website with final release of source code posted
3. A final report providing detailed description of the Phase II results
4. Right to use: CGSD will give the Government non-assignable rights to the use of all CGSD patents related to the inventions made for the program.

4. Related Work

4.1 Related Work by CGSD

Since its founding in 1990, CGSD has built a solid reputation for expertise in its field. Aerospace remains a significant part of the company's business, with clients including General Dynamics, Northrop-Grumman, Lockheed Martin, Loral, L-3 Communications/Link, Thomson Training & Simulation (U.K.), STN ATLAS Elektronik (Germany), the U.S. Army, Navy, and Air Force, and the U.K. Ministry of Defence, etc. The activities of CGSD Corporation and its key people relate to the proposed effort in two important ways: knowledge of the visual simulation community and knowledge of the relevant technology. Related work giving us this knowledge includes:

Algorithms for Run-Time Terrain Deformation: An SBIR Phase I project of which the objective is to investigate the feasibility of performing run-time terrain deformation to correct the mismatch between, for example, target data vs. terrain database. On-going since June 2004 for the Air Force AFRL/HEAE, Mesa, AZ.

Body Worn Graphics Image Generator for Simulation Based Training: An SBIR Phase I project of which the objective is to design a body worn portable image generator suitable for driving a 5000 x 4000 pixel head mounted display in an aircraft cockpit. Completed April 2004 for the Air Force AFRL/HEA Mesa, AZ.

3D Display of Time Critical Targets on Joint STARS Operator Workstation: *A real-time visualization algorithmic/techniques development for a battle environment*. Completed January 2002 for AFRL/IFKA Rome, NY. This is an SBIR Phase I project of which the objective is to improvise novel ways of visualize large amount of data that is changing every 30 seconds or less. CGSD introduced and developed a Wide Area Surveillance System (WASS) appearance simulator that can be used to explore visualization techniques.

Software Tools to Create Bump-Mapped Textures. Bump maps are height maps. Completed September 2002 for NAWCTSD. In this project, CGSD developed tools for generating bump maps, and generated a library of bump map textures. These products will meet the needs of next generation of simulator databases in both defense and private sectors.

Low-cost Image Generator for Mission Rehearsal: *A baseline design for a low-cost photo-textured mission rehearsal*. Completed November 1997 for NAWCTSD. CGSD specified mission visualization system requirements, image generator performance requirements, conducted a technology survey among candidate image generators, wrote software for the data management computer, the visual channel computers, and format conversion software for SOCET SET database terrain, feature, and texture data, as well as HUD display commands. Our innovative system design demonstrated the feasibility of meeting stringent requirements for shipboard mission rehearsal, including large areas of photo-textured terrain,

The PC-based In-flight refueling system F-14 instrument panel, and the refueling tanker model.

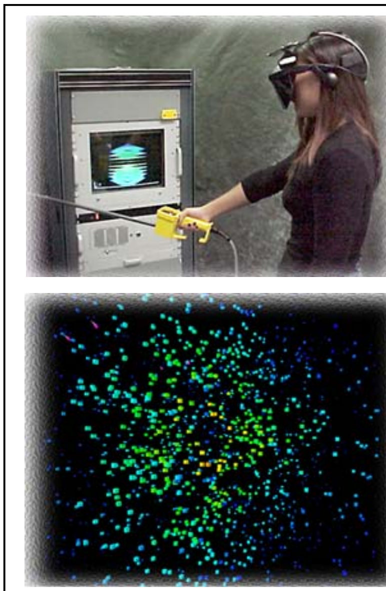
generation of HUD symbology, and network links to mission planning and threat simulation systems.

Navy In-Flight Refueling System: *A PC-based image generator system that provides six channels of imagery for in-flight refueling training.* Completed October 1999 for Laser Power Research. CGSD custom databases include F-14 cockpit and instrument panel, digital readout display, attitude instrument, radar display, altimeter, a naval aviation refueling probe, a simple refueling tank, a tanker, refueling drogue; CGSD custom software includes the transformation between binocular vision (three channels per eyepoint) and a single eyepoint. The software provides the means for changing the distances at which imagery is sorted into channels, and provides fine adjustments for the position, rotation, and size of each image output. An HMD is provided for the pilot. Game style joystick and throttle controls are provided to fly the virtual aircraft. Two monitors are provided for viewing the generated imagery, with each monitor capable of being selected to display any of the three channels for an eyepoint.

Real Time PC Simulation: *An innovative modular solution for rapid development of PC-based simulators. By modular, we mean independent of simulation HW/SW platforms.* Completed July 2001 for STRICOM. CGSD developed a method for multi-channel synchronization, a method for load management, and a method for performing mission functions, such as ground contact, collision detection, trafficability, radar altitude, range, line-of-sight, and intervisibility testing.

DS-230 Driving Simulator's three-monitor display.

DS-210/230 Driving Simulator: *A cost-effective driving simulator with HMD (DS-100) or with three large screens for display (DS-230).* Completed 2002, in-house project. CGSD has designed and developed two versions of driving simulators for cost-effective driver training with impressive features: (1) a force feedback steering wheel providing miscellaneous road effects, (2) a four-channel sound system providing audio cues for horn, engine, turning signal, wipers, and wind, (3) a high-fidelity real time vehicle dynamics model comprised of four wheel and suspension models coupled with six-degree-of-freedom vehicle equations of motion, (4) simulations of weather effects for clear, fog, rain, and snow, (5) a realistic simulation of windshield wipers in terms of motion, sound, and visual effects, (6) The database offers a variety of driving environments: city, highway, and country; each with four versions for summer/winter, left-hand/right-hand drive.



ARES system usage and an example of display.

ARES: *A see-through augmented reality system for 3-D visualization of CAD data.* Completed June 2001 for HEPCO/NSG Japan. It has a wand for pointing. For design visualization, each person in the design team can view CAD data by using a see-through HMD, and make critiques by pointing at features of the object being designed. This system offers advantages over traditional projection-based systems such as virtual workstations and CAVE systems. For this system we use a networked dual-processor Pentium, a 3D*Labs* graphics card, i-O Display I-glasses in see-through mode, an optical tracker, and an inertial tracker for hand and head tracking. The network card is used to receive the CAD data. The processed imagery of the CAD data is superimposed on the real world. The wand is tracked to enable users to interact with the data.

U.S. Army TEC: *Texture Library for 3-D Visualization Systems*. Completed February 1996 for US Army TEC. CGSD received the **1996** U.S. Army SBIR Phase II Quality Award for our efforts, which include examining and resolving methods for categorizing patterns, implementing real colors and contrasts, storing and distributing patterns to meet the requirements. In Phase I, we established the feasibility and specific requirements for implementing a texture library. In Phase II, we designed and developed texture-generating tools, and using the tools developed, after iterative refinements, to successfully automate the texture pattern generation process.

RealTexture Library package, and RealTexture Tools package.

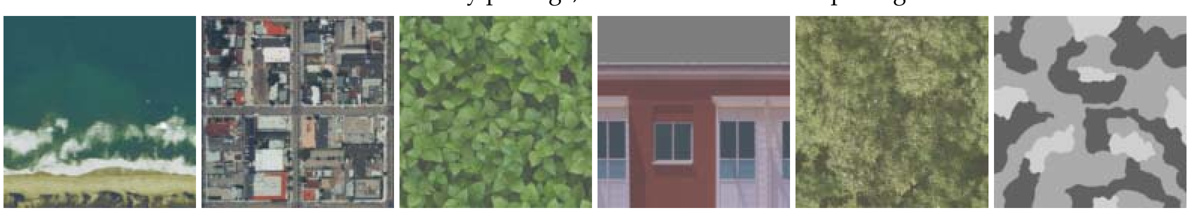
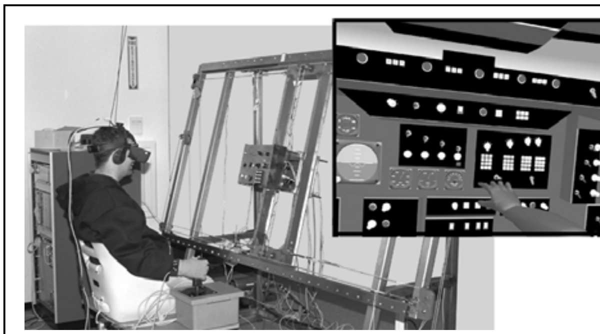


Figure: Examples of RealTexture patterns for beach, city block, foliage, store front, deciduous treetops, and U.S. Army camouflage winter pattern.

TOPIT: Touch Object Position In Time – A Force/Tactile Feedback System (FTFS) for a Virtual Cockpit. Completed May 1997 for U.S. Army STRICOM. *A complete virtual reality cockpit including head-mounted display, advanced head and hand tracking, and virtual controls*. A user wearing a data glove and a helmet-mounted display (HMD) can touch the control object, such as a knob or dial, which is delivered to the user's finger tip just in time by a high-speed robotic positioning system. CGSD invented the design concept, studied the optimal touchboard control components, researched the speed required for the robotic positioning system, studied 11 tracking devices and derived a hybrid tracking device that



TOPIT's positioning system and a view seen by the user (inset).

combines an optical tracker with an inertial tracker for tracking head and hand with acceptable precision. This virtual reality system incorporated (1) CGSD custom built positioning system that moves fast enough, (2) CGSD custom built tracking system that provides sufficient accuracy, (3) hand motion prediction algorithms that predict the appropriate control knob and its position, (4) CGSD innovative extrapolation algorithm that minimizes the time lag that is often seen in immersive virtual reality systems done by others, and (5) a safety system that protects users from being hurt if their fingers or hands ever intruded into the positioning system's motion space. CGSD successfully demonstrated the completed system to STRICOM. A U.S. patent was granted as a result of this project.

CSAL crew station's three-monitor display.

UDLP CSAL: *A simulator successfully applied and extended by our customer to the experimental analysis of weapons systems.* Completed March 1995 for UDLP. CGSD designed, developed, and integrated a Bradley Fighting Vehicle Simulator for combat system analysis labs (CSAL) of United Defense. The simulator includes a stealth station, a gunner station, and three crew stations – each with a PC for local display and control interface, and a six-channel SGI Onyx image generator, 3-D sound generator, and DIS network interface. Detailed tasks implemented by CGSD include, for example, specifying software and hardware components, code, and test software for PC displays and controls, obtain and fix databases, implement network entities, implement driver's and commander's instrument control panels, test intercom, make and integrate an IR version of the database, code and test stealth dynamics, integrate weather effects, implement ownship weapons controls and displays, integrate MODSAF database with visual database, implement viewpoint calculations for gunner's thermal sight, integrate and test with DIS network.

AeroBall Flight Simulator for Team Sports.

HUD display.

Simultainment: *A sophisticated low-cost PC-based simulator with touch screens, out-the-window displays, and custom flight models.* Completed November 2000 for Simultainment Corp. This effort includes the creation of a completely new flight simulator, in terms of visual database, real time software, device controls, and displays for a networked multiplayer team sport. The PC-based system uses 3d*fx* and Real3D image generators, CGSD custom databases, custom hand controls with innovative grip mechanism, touchscreen (for startup procedures) and out-the-window displays, and four-channel sound system augmented by a sub-woofer and seat shaker. The system features a full-custom true real time software using Windows NT with the InTime extension. Three user-selectable flight models were developed. The system includes a custom fiberglass two-person cockpit with a hydraulic 3-axis motion-base. Software was developed to keep the game clock and scores. Critical flight information such as the altitude, speed, heading, pitch, and roll is overlaid as a head-up display (HUD) on the out-the-window imagery. A rear view mirror is included. Extensive analysis of design options supported component selection for this cost-sensitive application.

Parametric Planets: *A tool to produce terrain data that accurately models specified geographic attributes.* Completed August 2001 for Parametric Planets, Inc. Specifically we had designed and developed algorithms and corresponding software that generates models for multiple residential and commercial buildings as well as terrain objects such as telephone poles, fences, driveways, and street lamps. The software reads in geographic attributes, such as shape, height, color, texture type, size, and elevation to create corresponding models of terrain databases. As a result of this project, an E-Town Building Library was compiled, in which there are 100 building models for use in visual simulation.

Semi-automated culture data generation using CGSD E-Town

E-Town, Road/Highway Editor: *Another tool to produce terrain data that accurately models specified geographic attributes.* Completed December 2002 for NEC/Kanematsu USA. Specifically CGSD designed and developed a semi-automatic tool, as a plug-in to Terrex's Terra Vista, for terrain data generation of roads and highways, based on a given elevation data and aerial imageries. For town and city construction, users specify building attributes such as color, roof top features, textures, etc. For road construction, users specify a string of road midpoints and attributes for the road including curb, markings, signs, sidewalk, fences or guard rails, median strips, etc. The editor generates OPENFLIGHT format data that can be imported to Terra Vista for visual database creation.

ChannelSYNC: *A software solution to synchronize multiple channels of display from separate computers.* Completed July 2001 in-house project. CGSD designed and developed an application programming interface, as a software package, to be integrated and compiled into user's application to synchronize up to 8 channels of multiple video displays, each driven by a computer. The synchronization accuracy is better than 1 millisecond.

ChannelSYNC synchronized the three-monitor

Technology Surveys: *Our studies help corporations and agencies worldwide select simulation technology*. CGSD has conducted client-sponsored image generator vendor surveys, including a comprehensive study that identified 30 manufacturers of image generators and analyzed the characteristics of each. We annually survey head-mounted displays, image generators, run time software, and PC graphics accelerators for our newsletter, *Real Time Graphics*. We performed a survey and analysis of tracker technologies under subcontract to Martin Marietta Corporation (Orlando, FL). We completed, with U.K. teaming partners, studies for the U.K. Ministry of

Defence to advise and develop specifications for visual systems for the Combined Arms Tactical Trainer. Additional surveys of simulator technology have been performed for commercial clients in Canada, the U.K., Germany, and Japan.


Real Time Graphics

Review: Interservice/Industry Training, Simulation, and Education Conference 2002

by Roy Latham

CAE Debuts CAE Medallion™-S Image Generator

Real Time Graphics newsletter: For more than a decade, through 2003, CGSD Corporation published *Real Time Graphics*, the only professional newsletter devoted to real time simulation and virtual environment technology. Our coverage includes every major industry source, and we are committed to tracking the technology of the industry. The newsletter is highly regarded, with subscribers in 35 countries worldwide.

We offer comprehensive knowledge of the technology. This has enabled CGSD to serve numerous clients, including both government and industry, during the past thirteen years in business.

4.2 Related Work by Others

We have studied, among IEEE transactions, ACM SIGGRAPH proceedings and I/ITSEC proceedings, many publications that touch some aspects of run-time terrain generation and deformation. Due to limited space, we only list a few references in the following for discussion:

In [Zvolanek & Dillard], [Zvolanek, et al], and [Nathman], the techniques of database correlation testing are discussed.

In [Panzitta], the author describes all aspects of database production. In particular, the author touches some aspect of terrain and culture correlation and correction.

In [Lindstrom & Pascucci] and [Lindstrom, et al], concepts are introduced, such as top-down mesh refinement, error metrics, on-the-fly triangle strip construction, view culling for view-dependant refinement algorithm, and unterleaved quadtrees, efficient index computation for data layout and indexing. Results for running the proposed terrain visualization system on various architectures were presented. With the suggestion of prediction and prefetching, the performance in real-time is not totally assured in the paper.

In [Röttger, et al], a new approach called Continuous Levels of Detail (CLOD) to approximate and reduce the number of geometric primitives in the elevation database is introduced. CLOD, unlike most of other algorithms using multi-resolution or global reduction techniques, uses a hierarchical quadtree technique. The results, as the algorithms implemented on an SGI machine, proved to be real-time if the elevation data (or height fields) fit entirely into RAM. Otherwise, an efficient paging mechanism need to be worked on, as the authors claimed.

In [Zhao, et al], this paper reviews work of CLOD by Lindstrom, ROAM by Duchaineauy, CLOD by Röttger, Smooth View-dependant LOD Control by Hoppe, and High LOD by [Jonathan]. Also, an improved method called Block-Based Terrain Walkthrough is illustrated. An interactive frame rate was accomplished. For larger data size, the authors claimed that paging and compression might be necessary.

In [Walsh], a stacked PC architecture is used to implement ROAM to achieve some degree of real-time result.

In [Turner], a particular terrain rendering algorithm is presented, discussed, and implemented. This algorithm is entitled real-time dynamic level of detail terrain rendering with ROAM. The result is only affirmed when the database size is small. Otherwise, RAM and computing power can be issues for real-time implementation.

In [Turner & Moscoso], the authors describe the advancements in the rapid collection of high-resolution digital topographic elevation and feature data for the Rapid Terrain Visualization (RTV). This paper detailed the flight missions of data collection process, and the resulting products of each type of flight mission. Digital Elevation Models (DEMs), 24 bit color coded DEM images, and 8-bit intensity image are the three products of a deHavilland Dash 7 aircraft flight mission. Also, the all weather day/night flight missions with an Interferometric Synthetic Aperture Radar (IFSAR) sensor produces a 16-bit ortho-rectifies synthetic SAR image. Feature extraction and identification process is also described.

In [Stevens], various agencies and corresponding efforts toward a common synthetic natural environment are described-- Notably, the development of a Terrain Common Data Model (TCDM) by STRICOM and the research of a multi-resolution terrain representation by DARPA. Also described is a frame work under development to provide a single correlated representation for the dynamic environment effects associated with all domains of the Joint Simulation (JSIM) Synthetic Natural Environment (SNE) (land, sea, air, and space). This paper concludes that opportunities exist for a common framework for a multi-resolution SNE for the needs of all programs.

In [Stokes & Fitch], this paper describes the feasibility of open database developed to de facto industry standards for multiple image generators. It concludes that the open database will meet the 60%-95% needs of a specific image generator, and the remainder percentage of efforts is still required for IG-specific enhancement.

In [Manocha], a number of algorithms for accelerating the render of large environments are presented. An improved run-time approach was presented that uses (1) a scene graph representation based on static LODs (Level of Detail), (2) improved algorithms for just about every aspect of graphics rendering including occlusion, culling, and rendering, and (3) special hardware GPU (graphics Processor Unit). The results from this improved approach depicted mostly CAD datasets corresponding to models of ships and powerplants in which many millions of very small polygons.

In [Gamble, et al], a hybrid approach of voxel modeling and polygonal representation was described. The approach suggests, for dynamic synthetic environments, voxel models be inserted at strategic locations within a polygonal environment. The challenges for this approach are (1) that less than the real-time frame rate is achieved, and (2) that development of voxel models is a non-trivial task.

In [Janette, et al], various dynamic terrain efforts were discussed. A new approach that calls for a distributed architecture, rather than a centralized terrain server, for DIS (Distributed Interactive Simulation). The implementation of this approach, using Environmental Change Notice (ECN), is also discussed.

In [Olszansky], the author analyzes, step by step, the construction of a simulation database. He then concluded that database construction is in grave danger of falling behind the advancement of graphics hardware, and in order to take advantage of the latest technologies, we must pay more attention to database construction.

In [Graniela & Siddon], SNE denotes Synthetic Natural Environment. This paper describes the development and significant features of three SNEs: (1) An urban database that contains a large number of Multiple Elevation Structure (MES) buildings (with interior), (2) A mountain database that contains complex cave structures with different types of ground access and tunnel geometries, and (3) A complex and dense jungle database with restricted mobility and line-of-sight. A good example of the MES building is a mosque with floor (level 1), low buildings (level 2), and minerattes (level 3). This paper provides some instances of cultural feature, such as MESs and tunnels, that might need to be taken into consideration for this project.

REFERENCES

Duchaineau, Mark, et al. "ROAMing Terrain: Real-time Optimally Adapting Meshes," IEEE Visualization '97 Proceedings, 1997.

Gamble, Murray, et al, "'Focused Fidelity': Applying Hybrid Rendering Technology to Dynamic Synthetic Environments", The 25th I/ITSEC Proceedings, 2003, pp. 1823-1832

Graniela, Benito and Siddon, Daryl, "Development of Future Combat System Synthetic Natural Environments", The 25th I/ITSEC Proceedings, 2003, pp. 697-606

Janett, Annette C. and Miller, Dale D., "Dynamic Terrain in Distributed Simulation Systems", The 18th I/ITSEC Proceedings, 1996, pp. 587-598

Jones, Richard, et al, "Development of Visualization, Modeling, and Simulation Technology for Cross-Applications in Live Fire Testing and Training" , The 19th I/ITSEC Proceedings, 1997, pp. 573-580

Lindstrom, Peter, and Pascucci, Valerio (Lawrence Livermore National Laboratory), "Visualization of Large Terrains Made Easy, " Proceedings of IEEE Visualization 2001, October 2001

Lindstrom, Peter, et al, "Real Time, Continuous Level of Detail Rendering of Height Fields", SIGGRAPH 1996 Proceedings, pp 109-118

Manocha, Dinesh, "Interactive Display of Complex Environments", The 25th I/ITSEC Proceedings, 2003, pp. 1833-1842

Nathman, Sherry, "Database Correlatable Charts Enhance Simulation Training", The 14th I/ITSEC Proceedings, 1992, pp. 720-723

Naylor, Bruce, "Real-Time Simulation of Unconstrained Munitions Damage to a Virtual Training Environment", The 25th I/ITSEC Proceedings, 2003, pp. 607-617

Olszanskyj, Serge J., "How Hard Is It To Make a Visual Simulation Database?", The 22nd I/ITSEC Proceedings, 2000, pp. 883-892

Panzitta, Michael J., "Improved Image Generator Database Development Technologies", The 14th I/ITSEC Proceedings, 1992, pp. 595-603

Röttger, Stephan, et al. "Real-Time Generation of Continuous Levels of Detail for Height Fields," University of Erlangen-Nuremberg, 1998.

Stevens, Clark D., "Toward a Common Synthetic Natural Environment" , The 21st I/ITSEC Proceedings, 1999, pp. 726-736

Stokes, Ted and Fitch, Michelle, "Using Open Industry Standards to Share Visual Simulation Databases", The 25th I/ITSEC Proceedings, 2003, pp. 1437-1443

Terrex, Terra Vista User's Guide, Version 2.0, June 1999

Turner, Bryan, "Real-Time Dynamic Level of Detail Terrain Rendering with ROAM", HLSL Workshops, April 03, 2000

Turner, J.T. and Moscoso, Christian P., "21st Century Terrain – Entering The Urban World", The 22nd I/ITSEC Proceedings, 2003, pp. 76-86

Walsh, AnneMarie, "Real-Time Parallel OpenGL Applications on Computer Clusters", B.A. Computer Science project, Stanford University, May 2003

Zhao, Youbing, et al, "A Fast Algorithm For Large Scale Terrain Walkthrough," International Conference on CAD & Graphics 2001, China.

Zvolanek, B., and Dillard, D., "Database Correlation Testing for Simulation Environments", the 14th I/ITSEC Proceedings, 1992, pp. 868-875

Zvolanek, B., et al, "Quantitative Correlation Testing from DoD Project 2851 Standard Simulator Data Bases", the 15th I/ITSEC Proceedings, 1993, pp.830-836

5. Relationship with Future Research

5.1 Results if the Project is Successful

The following products will exist at the completion of a successful Phase II effort:

- a feasible run-time terrain deformation algorithm
- a demonstration of the prototype correction software kit with OpenSceneGraph

5.2 Phase II as a Foundation for Phase III Commercialization

The Phase II results will provide a basis for establishing the feasibility of a Phase III and for estimating the resources required.

6. Commercialization Strategy

While there is a huge military market for terrain deformation to match features, there is not a significant commercial market. However, there is a significant commercial market for real time deformation of surface other than terrain. These include medical simulators in which tissue surfaces deform in response to simulated medical instruments, scientific visualization of constrained surfaces, and game application in which skin-like surfaces deform and dented objects deform.

We are using the open-source business model for this market. With this model, the source code is distributed for free or for a nominal sum, but support and consulting services are sold competitively. It is unrealistic to suppose significant sales of proprietary code that has to be embedded within many different real time environments. Therefore the source code must be revealed. However, the code will be sufficiently complex, we believe, that users will want custom services in integrating it.

CGSD, in addition to its consulting business, currently produces and markets eight software products: of our own including E-Town Editor and Library, RealTexture Tools and Library, Bump Texture Tools and Library, Color Science Library, and ChannelSYNC. In addition we serve as dealers for three vendors of head-mounted displays, including the wide field-of-view SEOS HMD 120/40. The expertise acquired in developing and marketing these products supports commercialization.

7. Key Personnel

The key personnel and their roles on the project are summarized in Table 3; resumes follow.

Table 3. Key Personnel and Their Roles on the Project.

Person, *Role on Project*	**Areas of Expertise**
Roy Latham, Principal Investigator and Sr. Systems Engineer	Technical direction, concepts, system design & requirement specification, progress status report, final report
Dr. P.Y. Cheng, Senior Software Engineer	Cost analysis, project planning, algorithms, software design, project administration

ROY W. LATHAM, a U.S. citizen, will be the Principal Investigator for the project. Mr. Latham is the founder of Computer Graphics Systems Development Corporation (CGSD Corp.); a consulting firm specialized in 3-D visualization systems and simulation technology. He is widely recognized as an authority in the field of real time simulation systems. He is known through widespread consulting activity for major corporations and for government agencies.

EXPERIENCE

Computer Graphics Systems Development Corp. (CGSD Corp.), President, 1990 -Present

Mr. Latham is founder and president of CGSD, a consulting firm specializing in computer graphics technology. Projects performed personally or with associates at CGSD include:

Loral Western Development Labs (San Jose, CA): Designed system architecture, wrote detailed specifications, and lead eight-person team for the development of a multi-processing high-performance graphics system for simulation training.

Atlas Elektronik (Breman, Germany): Performed design studies and developed system specifications for a geometric-and-image-processing based visual simulation system for use in driver training.

Grumman Melbourne Systems (Melbourne, FL): Developed recommendations for the interconnection of a battlefield surveillance radar system simulation with a large heterogeneous network. Also teamed with Grumman in proposing to DARPA a virtual reality system for the graphics display of battlefield data.

U.S. Air Force Institute of Technology (Dayton, OH): Reviewed and advised on academic R&D progress related to computer graphics and virtual reality.

General Electric Aerospace Division (Daytona Beach. FL): Technology studies and proposal preparation related to graphics technology for simulation.

U.K. Ministry of Defence (London, U.K.): As team member with U.K. firms, performed two studies of technology, wrote specifications, and recommended procurement strategy for visual simulation system on a large network of armored vehicle simulators.

In July 1992, CGSD began publishing *Real Time Graphics*, a professional newsletter for which Mr. Latham also serves as publisher.

In addition, Mr. Latham has managed and provided technical direction for all of the projects performed at CGSD. He has been an influential advocate for improved PC simulation through numerous articles in *Real Time Graphics,* by serving on the IMAGE Society's PC Simulation Special Interest Group, and through the teaching of short courses for the Image Society and for the University of Binghamton. He has served as a consultant and expert witness in a number of intellectual property cases related to computer graphics.

Mr. Latham has a record as an innovator in graphics and simulation, having been granted ten United States patents in the field. He has a patent pending on a method for performing wide dynamic range illumination calculations for sensor processing on a PC using commercial eight-bit graphics accelerators. Past contributions to graphics technology include development of a list priority algorithm that uses topological sorting, a method for determining the optimal placement of sample points within a pixel, a method for allocation of polygon densities and switching distances in a simulation database, and a method for measuring the quality of a generated image.

At ***Sun Microsystems*** as an Engineering Manager (1987 - 1989) he managed the development of a 3-D graphics accelerator for engineering workstations, including development of eight ASIC designs, implementation of parallel digital signal processors, and development of new graphics algorithms. At ***Kaiser Electronics*** as Project Manager and Senior Technical Staff Specialist from 1986 -1987 he managed the research and development of a digital map reading system through the detailed design of the prototype. At ***Link Flight Simulation Division*** he was a Principal Investigator and Staff Scientist from 1978 - 1986. At Link he was responsible for the coordination of visual simulation technology among four operations of the division, and prior to that he managed the independent research and development of an image generator, which was then the world's highest performance real time graphics system. He began his career at ***Grumman Aerospace Corporation***, 1970 - 1978, as a Flight Test Engineer, where he performed research and development of high-precision electronic navigation systems, including a patented Kalman filter based system used for the flight testing of F-14's. His work in Flight Test included the error analysis of flight models derived from test data.

EDUCATION

M.S. Computer Science, University of Santa Clara, 1983

M.S. Applied Mathematics and Statistic, State University of New York at Stony Brook, 1974

B.S. Aeronautics and Astronautics, Massachusetts Institute of Technology, 1970

B.S. Electrical Engineering, Massachusetts Institute of Technology, 1970

LICENSES AND MEMBERSHIPS

Licensed United States Patent Agent. Licensed Professional Engineer (New York State). Licensed private pilot. Member: Institute for Electrical and Electronics Engineers, The IMAGE Society; Association for Computing Machinery, Association for the U.S. Army.

INVENTIONS AND PUBLICATIONS

Book: *The Dictionary of Computer Graphics and Virtual Reality* (2nd ed.). Springer-Verlag, Jan. 1995. ***Papers***: Research and Professional Society papers - 14 published. ***Technical Articles***: More than 100 published. He has been granted ten United States patents:

6,184,857 Computer-implemented method of rendering an Image of smoke

6,126,401 Hybrid electric/hydraulic drive system

5,913,684 Computer controlled robotic system for unencumbered movement

5,859,645 Method for point sampling in computer graphics systems

5,803,738 Apparatus for robotic force simulation

5,719,598 Graphics processor for parallel processing a plurality of fields of view for multiple video displays

5,509,110 Method for tree-structured hierarchical occlusion in image generators

4,748,572 Video processor architecture with distance sorting capability

4,734,875 Log mixer circuit

3,975,731 Airborne positioning system

P.Y. CHENG, a U.S. citizen, will be a team member responsible for algorithms and software engineering. Dr. Cheng has more than 25 years of experience in software system engineering management, system design and analysis, including algorithm design and development of computer graphics systems, flight simulator visual systems, and communication systems. Dr. Cheng specializes in the area of real time 3-D graphics and application software while working for CGSD Corp., TRW, FMC, Schlumberger, and Ford Aerospace Corporation. Dr. Cheng has proficiency in programming languages including C/C++, Fortran, and ADA, and in database development tools. Dr. Cheng obtained her Ph.D. in Mathematics from the University of California at Davis, Dr. Cheng has authored four papers published in professional societies in topics including high-level control language, modeling 3-D-objects with multiple texturing planes, and cost-appraisal techniques for route planning. As Chief Financial Officer, Dr. Cheng has been a team builder and the key person in assuring the successful completion and delivery of each project in CGSD.

EXPERIENCE

Computer Graphics Systems Development Corp., 1992 – present As CFO of CGSD, Dr. Cheng has been the deputy program manager for all projects in CGSD Corp. since 1992, notably RealTexture Library and Tools, Bump Texture Library and Tools, Color Science Library, Grumman OBATS, Grumman voice recognition project, TOPIT, OmniTrek, and FMC CSAL projects. To accomplish the on-time-on-budget objective for each project, Dr. Cheng recruited top engineers, mostly from top universities as interns or staff, and assigned them for making expertise-specific contributions. Consequently, CGSD is known for being very dynamic and innovative.

Also as Senior Software Engineer, Dr. Cheng led software engineering efforts in analyzing and specifying software systems for training devices, and was also responsible for commercial software design and development projects on various platforms.

ESL, Inc./TRW, Staff Engineer/Scientist – Software, 1990 – 1992 Dr. Cheng designed and developed demonstration software for JSTARS study; participated in design specification in gateway design for Intelligent Surveillance System in the GUARDRAIL program; developed task breakdowns and cost estimates, and participated in software design and development for the CATALYST program; developed MIS software for database applications; and conducted validation and product review for Tardon's ADA compiler. Dr. Cheng used C and ADA programming languages in the projects involved.

CTC of FMC Corporation, Senior Staff Scientist, 1985 – 1990 Dr. Cheng engineered several IR&D projects from proposal writing to the delivery of software; designed and developed algorithms and software for system integration of autonomous land vehicle test bed (ALVTB), designed and developed screen design/MMI for vehicle control software. Dr. Cheng published two papers respectively in high-level control language and path planning for an autonomous land vehicle. Dr. Cheng received an FMC innovation award.

Benson, Inc. of Schlumberger, Senior Staff Engineer, 1984-1985 Dr. Cheng led the development work for various plotter-related algorithms and computer graphics software, including a color dithering technique. Dr. Cheng was selected to represent the company in the Standards Committee of National Computer Graphics Association (NCGA).

Link Flight Simulation Division of Singer Company, Senior Staff Engineer, 1980 – 1984 Dr. Cheng led a group of software engineers in design and development of the frame calculator for a new generation of digital image generator (MODDIG); implemented IRAD project algorithms. Dr. Cheng published two papers, one in procedure modeling, and one in building 3-D objects by multiple textured planes.

Ford Aerospace, Engineering Specialist, 1978 – 1980 Dr. Cheng designed and developed software for satellite mission control for INSAT I.

PUBLICATIONS

1. Interlacing Divisibilities of Similarity Invariant Factors and Applications, Ph.D. Dissertation, Department of Mathematics, University of California, Davis, 1978.

2. Building Three-dimensional Objects with Multiple Textured Planes, proceedings of the Interservice/Industry Training Equipment Conference, Washington, D.C., 1984.

3. A High-Level Control Language for an Autonomous Land Vehicle, Proceedings of the Thirteenth Annual IEEE Industrial Electronics Control Conference, Cambridge, MA, November 1987.

4. Cost-Appraisal Techniques for Route Planning, Proceedings of the International Society for Optical Engineering Cambridge Symposium, Cambridge, MA November 1987.

EDUCATION

Ph.D. Mathematics, University of California, Davis, 1978

M.S. Mathematics, University of North Carolina, Chapel Hill, 1974

B.S. Mathematics, Tsing Hua University, Taiwan, 1972

MEMBERSHIPS

IEEE, ACM

8. Facilities/Equipment

CGSD Corp. has all of the facilities and equipment necessary to perform this study in its Mountain View, California headquarters. Its Silicon Valley location facilitates close cooperation with leading high-technology companies.

The software suite available for the current project includes Terrain Expert's Terra Vista (for which we have developed plug-ins), MultiGen Creator, 3D Studio Max, and Adobe Photoshop (for which we are also a plug-in developer). We use the CG2 V-tree run time environment. The CGSD COTS software tools were described above in the body of the proposal.

CGSD maintains a networked enterprise computing environment to serve its diverse client and contract requirements.

We have a complete virtual reality lab including a Silicon Graphics and PC-based graphics accelerators, Ascension magnetic trackers, Origin optical trackers, custom and commercial inertial trackers, and a variety of head-mounted display units.

CGSD facilities where the work will be performed meet environmental laws and regulations of federal, state, and local governments for, but not limited to, the following groupings: airborne emissions, waterborne effluents, external radiation levels, outdoor noise, solid and bulk waste disposal practices, and handling and storage of toxic waste materials. Our products are all dolphin safe and contain no whale byproducts whatsoever.

CGSD is a corporate member of the association of U.S. Army, a member of the Better Business Bureau, a member of the International Association of Amusement Park and Attractions, a corporate member of the IMAGE Society, and a member of the American Defense Preparedness Association.

9. Subcontractors/Consultants

(on the need basis)

10. Prior, Current, or Pending Support of Similar Proposal or Awards

There is no prior, current, or pending support for the proposed effort.