DYNAMIC TERRAIN DATABASE DESIGN FOR REAL TIME IMAGE GENERATION

Xin Li, Dale D. Miller, Mark Illing, Mark Kenworthy and Mark Heinen
LORAL Advanced Distributed Simulation
Bellevue, Washington 98005

ABSTRACT

Substantial interest in a Dynamic Terrain (DT) database has been expressed by users and developers of real time distributed simulation and training environments. This capability allows the dynamic reconstruction of the landscape or rearrangment of the terrain surface during a simulation. One of the most challenging issues for DT in distributed simulation is the tessellation and management of the terrain database with a desired resolution meeting the real-time requirements of polygon throughput, memory allotments and interface bandwidth of the image generator.

Our research work is the first attempt of developing such capability for SIMNET image generators and databases. In this paper, the database partitioning strategies are proposed, which can be conceptually adopted by other image generators. The dynamic soil model simulating excavating activities on the terrain surface is described. The management of runtime terrain database and interface messages are presented. Implementation issues on the image generator are also discussed.

Key words: Computer image generation, real-time simulation, dynamic terrain, run-time terrain database modification, terrain relaxation, physically-based soil models.

BIOGRAPHIES

Xin Li: Dr. Xin Li is a Real-Time Software Engineer. He received his Ph. D. from the University of Central Florida and his M.S./B.S. in Computer Science from the Academic Sinica of China and the Northwest University of China. Dr. Li developed physically-based soil models while at the Institute for Simulation and Training. Since joining Loral, Dr. Li led the dynamic terrain effort described in this paper, and he is currently involved in the development of real-time rendering software based on physical-based models of clouds and smoke in a Dynamic Environment program. Dr. Xin Li can be reached at Loral ADS, 13810 SE Eastgate Way, Suite 500, Bellevue, WA 98005, (206)957-3213 (Email: xli@lads-bvu.loral.com).

Dale Miller: Dr. Dale Miller is the manager of the visual software engineering groups at Loral ADS. He received his Ph.D. in mathematics from the University of Washington in 1976. Since then he has contributed to the areas of abstract algebra, digital signal processing, applications of the residue number system for high speed digital signal processing hardware, machine vision, optical character recognition and computer graphics. Dr. Miller is the program engineer for GT200 image generator development.

Mark Illing: Mark Illing holds a B.S. in Computer Science Engineering from the University of Illinois and is currently working in the Systems Engineering Group with the real-time CIG database and development at Loral ADS. He is responsible for defining system requirements for real-time visual simulation systems, as well as designing, developing and enhancing these systems, their databases and development tools. Mr. Illing's primary focus is embedded software control of real-time hardware subsystems.

Mark Heinen: Mark Heinen received his B.S. in Computer Science from the University of Minnesota. He is currently a member of the Applications Software Group at Loral ADS. Mr. Heinen has worked on various projects including database construction tools, database compilation tools, image generation algorithms and software, CIG hardware emulation software, and CIG real-time software. He has researched and emulated image generation algorithms in software for use in the design and development of new generation CIG hardware technology.

Mark Kenworthy: Mark Kenworthy is the manager of the Systems Engineering at Loral ADS and has specialized in design and development of real time image generation systems for the last 10 years. Mr. Kenworthy holds a bachelor of Science degree in Aeronautical and Astronatical Engineering from Purdue University.

DYNAMIC TERRAIN DATABASE DESIGN FOR REAL TIME IMAGE GENERATION

Xin Li, Dale D. Miller, Mark Illing, Mark Kenworthy and Mark Heinen

1. Introduction

Previous efforts have demonstrated real-time modifications of synthetic terrain using an underlying physically-based model of the soil [Li93b]. This work has utilized a regular, fine grid for the terrain with limited extents of the virtual environment. Because of this, the total number of polygons required to represent the terrain surface remained fixed. Also, the implementation was done on a graphics workstation without textures.

The goal for the effort described in this paper was to expand upon this previous work to implement terrain modification capability on a production computer image generator (CIG) with full texturing using terrain databases of unlimited size. This required design of algorithms for real-time terrain repolygonization, texture map switching and vehicle track laydown as well as the background aggregation of polygons which preserves geometry while reducing polygon density. The repolygonization capability in turn required design of new data structures capable of representing changeable terrain. Finally, with the soil model residing on the simulation host, communication protocols between the host and the CIG were required.

This development was intended as a proof of principle, focusing on the realistic visual representation of dynamic terrain. The Loral GT100™ visual system was used for interactively bulldozing arbitrary locations on any SIMNET terrain database as shown in the image of Figure 1-1. No effort was made to attain permanence of changes or interactivity with other entities on a Distributed Interactive Simulation (DIS) network. Further work is required in networking issues and system architecture design before these new visual system capabilities can be fielded for large scale use.

2. GT100 Architecture

The GT100 is a production computer image generator (CIG) system first introduced in 1988. It is optimized for distributed (networked) interactive tactical team training in ground and near-ground vehicle applications.

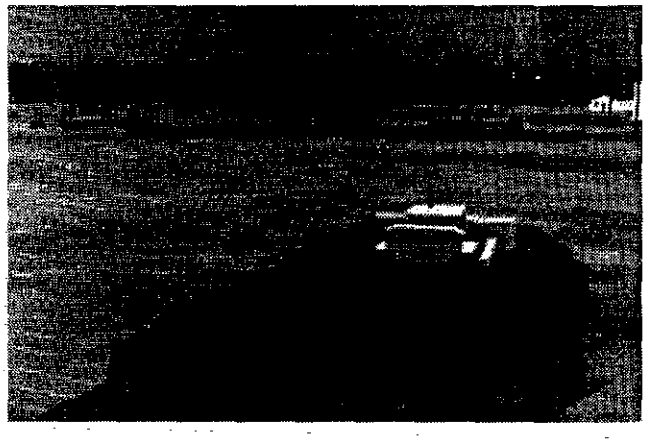

Figure 1-1 Real-time image of bulldozer modifying terrain.

The GT100 is capable of responding to the display demands of a wide variety of dynamic information that arrives in its field of view. It is designed to support the requirements of distributed simulation including very complex databases, large numbers of moving models, collision detection, correlated sensor databases, database intersection processing, and large numbers of special effects.

The GT100 system was an excellent candidate system for our first implementation of a dynamic terrain database design because the interaction of objects in the distributed simulation environment cannot be planned. The GT100 system allows a number of configurations and options to be specified by the end user. This overview of the GT100 system relates to the system used for our first dynamic terrain implementation. Complete product information for the GT100 family of image generators may be obtained through the authors.

2.1 System Overview

The major components of the GT100 visual system are shown in Figure 2.1-1. The GT100 visual system is comprised of two parts: the CIG Host Subassembly and the Graphics Processor Subassembly.

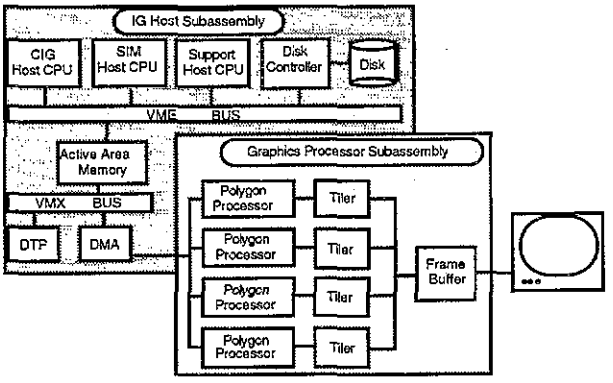

Figure 2.1-1 Dynamic Terrain Hardware Support Environment

The task of the CIG Host Subassembly is to manage efficiently the environment information so that the potentially visible three dimensional environment polygon data can be sent to the Graphics Processor Subassembly in real-time. Three processors, the CIG, SIM and Support Host CPUs, work together to place in active area memory an organized description of the simulated environment. This active area memory is simultaneously accessed by a database traversal processor (DTP) which quickly scans the environment description, determines what data needs to be sent to the Graphics Processor Subsystem, and informs the DMA controller to send that data to the Graphics Processor Subsystem.

The Graphics Processor Subassembly is a parallel pipeline graphics engine which transforms three dimensional environment polygon data into real-time video output. The data is fed to the subassembly by the CIG Host Subassembly. Each of four graphics pipelines is made up of a graphics processor and pixel tiler. Tiler output is combined and displayed by the frame buffer. (See Table 2.1-1 for GT100 system specifications.)

All of the system modifications to support the dynamic terrain demonstration were made within software modules for the CIG Host Subassembly.

2.2 Resource Utilization

The role of the three host CPUs is to manipulate the dynamic environment data in a manner which consistently provides this information to the database traversal processor (DTP/DMA). The dynamic environment manipulation task is partitioned as follows:

Table 2.1-1 GT100 System Specifications.

Image Update Rate	15 Hz
Terrain Modification Transport Delay	167 milliseconds
Textured, Anti-Aliased, Potentially Visible Polygons	90,000 polys/sec
Display Resolution	640 x 480 pixels
Pixel Fill Rate	25,000,000 pixels/sec
Occulting Levels	524, 288
Color Palette	4096
Number of Texture Maps	256
Texture Map Resolution	128 x 128 pixels

SIM Host CPU - The Simulation (SIM) Host CPU is responsible for simulating the interactions of the vehicle in the environment. In this application the vehicle is a bulldozer and it not only interacts with the terrain but also modifies the terrain. All algorithms dealing with the soil model and terrain modification execute on the SIM Host CPU.

CIG Host CPU - The CIG Host CPU is responsible for managing changes to the active area memory. As stated previously, the active area memory contains an organized description of the simulated environment which is accessed asynchronously by the database traversal processor (DTP). All requests for modifications to the terrain are managed by this processor as well as other image generator support functions.

Support Host CPU - As more and more terrain is modified by the bulldozer, a significantly large number of polygons are created that are potentially visible. The visual system has a limit to the rate at which it can process polygons. The support host is responsible for executing terrain relaxation algorithms to keep the polygon load below system thresholds.

The GT100 has a rich library of messages used to communicate between the multiple processors for simulation applications. For further detail, please refer to [CIG/SIM Comm 90].

We note here that any method for dynamic terrain on the GT100 requires additional memory than that used for a typical simulation. It is necessary for manipulating polygons and maintaining a working copy of the polygons while another copy is being displayed. Our memory utilization algorithms reuse memory when possible and we are able to run a continous exercise lasting over an hour with 1.5 MB of memory dedicated to dynamic terrain.

2.3 Terrain Format

Movement of the simulated vehicle through the environment requires paging environment data into active area memory from disk. All the data to describe a 500 meter square area is grouped together to form a load module. The active area memory has a 16 x 16 array of load modules in memory at any one time. This allows the GT100 visual system to have a viewing range of 3500 meters in any direction from any position in the database and still provide for one row or column to be in transition (paging in from disk).

Each load module contains a group of polygons that define the terrain skin. In most cases, the terrain is defined by a 4 x 4 regular tessellation with a grid spacing of 125 meters and up to 32 polygons connecting these vertices. For areas that require greater resolution than this grid supports, micro terrain provides additional terrain polygons not limited to the grid spacing.

3. Dynamic Terrain Models

This section provides a high-level description for the dynamic soil slump and manipulation models implemented for the virtual bulldozer simulation. Interested readers are referred to [Li93a] and [Li93b] for more details.

3.1 Soil Slump Model.

Given a soil configuration, e.g., a pile of soil with certain geometrical and physical properties, the soil slump model answers three questions:

1) Is the given configuration stable? (i.e., will it slump?)

2) What restoring force is required to return the soil configuration to static equilibrium if it is unstable?

3) How can mass conservation be preserved while the configuration changes state?

The stability of a given soil configuration is determined by the safety factor of a potential failure surface. According to the Mohr-coulomb theory, the safety factor is defined as a ratio of the strength force and stress force [Chowdhury 78]. The strength force provides the resistance to deformation by continuous shear displacement of soil particles along surfaces of rupture, while the stress force pushes the soil mass to move along the failure plane. If geometrical properties (area of the failure plane, volume of the soil mass) and physical properties (the cohesion, internal friction angle and unit weight of the soil) are known, both strength and stress forces can be calculated by using equations presented in [Li93b]. The configuration is statically stable if the safety factor is greater than one. Otherwise, soil sliding is inevitable.

To analyze the restoring force, the unstable soil configuration is first divided into small vertical slices with equal width as shown in Figure 3.1-1.

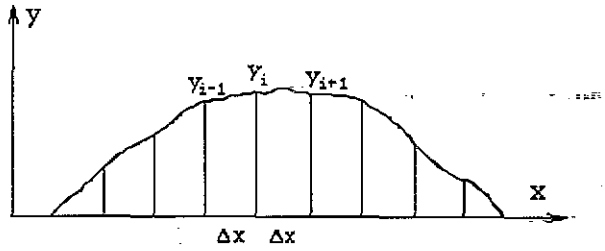

Figure 3.1-1: Dividing the given mass into small slices

The calculation of the restoring force of each slice can be done individually. Since forces exerted between each pair of soil slices are equal and in opposite directions, they can be canceled. At any particular time t, therefore, sliding can only happen in the top area of a slice. This area is further divided into slivery, v-shaped wedges and Newtonian physics is then applied to quantify net forces experienced by each wedge. The total restoring force is finally obtained by integrating small forces together [Li 93a].

Mass conservation is achieved by the following technique. Recall that a given configuration is divided into n slices. The i-th slice, $1 \le i \le n$, is now conveniently thought of as a container holding an amount of soil.

A small change of the amount of soil in each container can be viewed from two different points of view: geometrically, this change can be represented by the change of shaded area (shown in Fig 3.1-2), which is a function of the heights of soil elevation posts (e.g., y_i and y_{i+1}). On the other hand, physically, it is the amount of soil which goes out of a container, minus the amount of soil mass which goes in. This principle can be described by another function of the rate of the "flow" of soil mass, which is in terms of a function of restoring forces discussed earlier. Putting all these together, one derives n+1 ordinary differential equations with n+1 unknowns [Li 93a]. Solving these equations

provides a solution for the soil slump behavior which satisfies both soil dynamics and mass conservation.

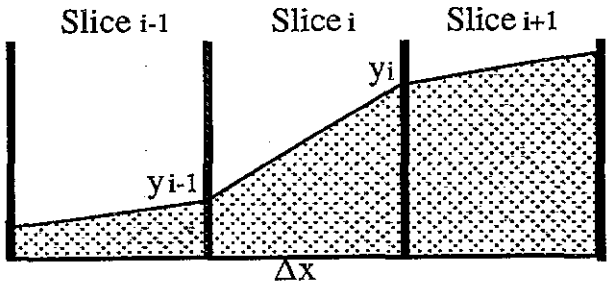

Figure 3.1-2: Considering slices as containers

3.2 Bulldozer Vehicle Dynamics Model

The bulldozer model simulates excavating activities such as digging, piling and pushing soil mass. The model is developed by first analyzing the interaction between the soil mass and a bulldozer's blade. Assuming that the shape of the blade can be approximated by an arc of a circle with radius R, we divide this arc into n segments. Furthermore, the soil mass in front of the blade is partitioned into n slices by horizontal lines at each joint point of two arc segments as shown in the following figure.

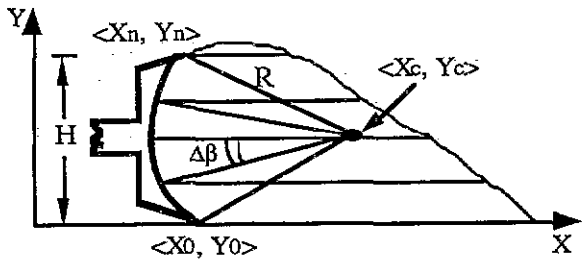

Figure 3.2-1: Dividing the blade and soil mass

If the cutting part of the bulldozer pushes the soil mass with enough force, the equilibrium will be destroyed. At this moment, the resistance experienced by each segment of the blade is determined by the soil properties (i.e., cohesion, internal and external friction angle and unit weight) and the geometry of the blade (i.e., the length and the cutting angle of the blade). Those resisting forces can be calculated for each blade segment individually by an equation presented in [Li93b]. If the force applied on a blade segment is further decomposed, we obtain two component forces: one is perpendicular to the segment, which is canceled by an opposite force provided by the blade, and another is always parallel to the surface of the blade. Integrating parallel forces of all blade segments together, we find that the total parallel force is pointing from the bottom to the top of the blade, that is, the soil mass being cut is always moving upward along the blade.

This phenomenon is also observed experimentally [Balovnev83]. The sequence of events occurring during the process of interaction between the cutting blade fixed on the advancing bulldozer and excavated soil mass before the blade can be described by three steps.

1) The soil chip being cut from the main soil mass moves upward along the blade because of resistance to the soil.

2) The soil chip is broken up into individual lumps on the upper part of the blade.

3) These lumps move downward toward the soil layers being further cut and form the soil prism which is being dragged.

Figure 3.2-2 depicts this pattern of the movement of soil mass.

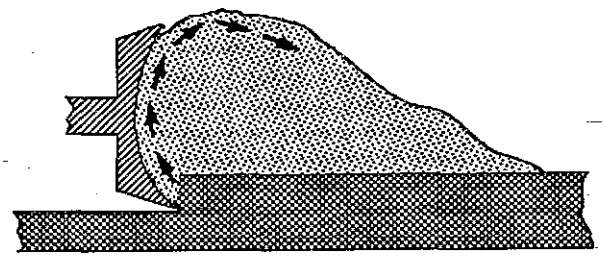

Figure 3.2-2: Pattern of soil movement ahead of the blade

This excavating action of a bulldozer is simulated by an algorithm with three stages: digging, piling and slumping. First, the model tracks the motion of the bulldozer. Along its path, wherever the altitude of soil mass is higher than the bottom of the blade, the new soil elevation is forced to have the same elevation value as the blade's bottom. *This procedure will create a ditch along the path of* the bulldozer on the surface of the terrain. The second stage simulates the upward movement of the soil along the blade by placing a soil chunk representing the mass cut in the first stage onto the top of the soil prism in front of the blade. Finally, in the third stage, the soil slump model introduced earlier is used to simulate the free flow motion of broken lumps of soil. *Although the soil being* brought to the top of the berm arrives continuously

in the real world, a chunk is a reasonable approximation of the amount and location of the soil that would actually arrive during one time slice in our discrete time simulation process. The soil slump model smoothly integrates this chunk into the berm, resulting in a realistic appearance.

4. Runtime Database Modification

To manipulate terrain in real-time without visual anomalies, we developed the data services necessary to manipulate the terrain skin, implemented the prototype bulldozer simulation and reduced visual system loading with polygon reduction methods.

4.1 Terrain Manipulation Strategy

Recall that a load module is a 4x4 regular tessellation with a grid spacing of 125 meters representing the typical resolution of the terrain skin. Higher fidelity terrain can be displayed using micro terrain. Our goal was to develop a method to manipulate the terrain skin at less than 1 meter elevation post spacing for a reasonably realistic visual appearance. Replacing an entire load module with micro terrain would require over 250,000 polygons, well beyond the means of our visual system.

As a bulldozer affects only a small area around itself instantaneously, we chose to implement a hierarchical approach. Rather than replacing an entire load module with micro terrain, we progressively add detail where needed by partitioning the data into finer resolution terrain until we meet the desired resolution for manipulation. As the bulldozer moves to untouched terrain, additional partitioning occurs to allow its manipulation.

We experimented with different levels of partitioning and chose the following levels as they provide optimum data segmentation for the GT100 visual system. (See Figure 4.1-1.)

A 125 meter square of a load module is replaced with a 5x5 grid at the 1st partitioning level providing 25 meter elevation post spacing, replacing 2 polygons with 50. A square in the 1st partitioning level is replaced with a 7x7 at the 2nd level for 3.6 meter elevation post spacing, replacing 2 polygons with 98. A square in the 2nd partitioning level is replaced by a 7x7 with the 3rd and bottom partitioning level for 0.51 meter elevation post spacing replacing 2 polygons with 98 additional. Once the bulldozer is initialized and four partitioning levels are created, the majority of new changes to the partitioning merely require replacing 2 polygons from the 2nd partitioning level with 98 new polygons at the 3rd partitioning level.

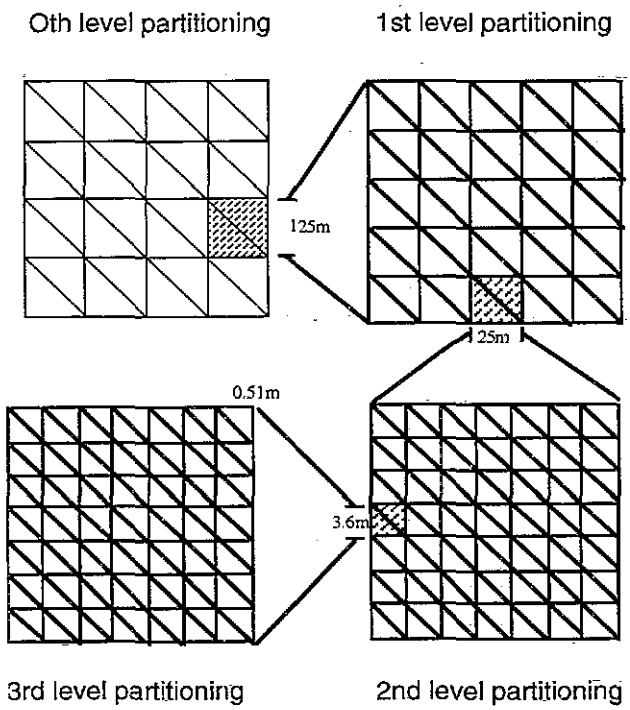

Figure 4.1-1 Terrain Partitioning

Implementation of the terrain partitioning dimensions, spacing and number of partitioning levels remains flexible, allowing tuning for a particular application or visual system. Changing the partitioning will result in coarser or finer subdivision.

In order to simplify locating and updating DT information, a data structure called a "patch" is used to represent terrain partitioning at different levels. It is an atomic unit for DT information exchange between the SIM and CIG hosts. A patch consists of three parts: geographic information, polygon graphics processor commands and database traversal processor commands which contain links to other patches at the higher, lower and same levels of terrain tessellation. Terrain patches at different partitioning levels are managed in the program by a patch forest of tree-like structures, where each root of a tree represents a load module. When a root of a patch tree is inserted into the runtime database, the geographic surface described by each patch in the tree is automatically processed and displayed by the graphics pipeline.

An example of a patch tree is demonstrated in Figure 4.1-2.

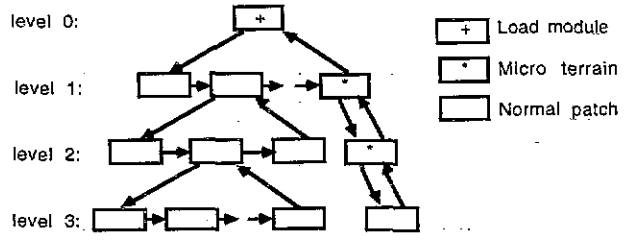

Figure 4.1-2. A Terrain Partitioning Tree

4.2 Support Services

Several new simulation support service messages were developed to provide additional functionality from the visual system host communications for the dynamic terrain implementation. These are outlined below:

MSG_DT_REQUEST is used by the simulation host to request the terrain definition for a specific location. A pointer to a bottom level partition containing the position is returned. If a partition containing the position is not found, an additional partition(s) is/are created, inserted in the processing stream and the corresponding polygons for higher level partition(s) is/are removed.

MSG_DT_PATCH is used to exchange elevation information for a modified partition between the SIM host and the CIG host. It contains the location, dimensions, spacing, and z values of elevation posts for the partition. Upon receipt of this new information, a copy of the partition is created with the new elevation values. Polygons are textured with dirt and those under the treads of the bulldozer are treated specially to display the tread pattern. It is inserted in the visual system processing stream followed by removal of the old partition. This process prevents visual anomalies of some terrain missing temporarily.

MSG_DT_RELAX is used to pass partition information between CIG host and the support host for terrain relaxation. Like MSG_DT_PATCH, it contains the position, dimensions, spacing, and z values of elevation posts for the partition. In addition, it specifies those vertices which can be referenced during the relaxation process, but are not be changed. No vertex is deleted from the list during relaxation. The relaxed patch is packed using the same message format and returned to the CIG host with a modified z value list and a new polygon list.

4.3 Excavation Activity Simulation Implementation

In this section, we define the concepts of active patch and active arena and describe the implementation of a virtual bulldozer model on the SIM host.

An active patch is a terrain patch at the bottom-level partitioning, represented by a regularly tessellated grid of equal dimension and spacing, and within a certain distance (say 5 meters) of the center of the excavating blade of a bulldozer. An active arena is an area assembled with several adjacent active patches. It is a region where elevation changes to the elevation posts of the terrain surface are likely to happen in the near future. An example of an active arena is shown in Figure 4.3-1.

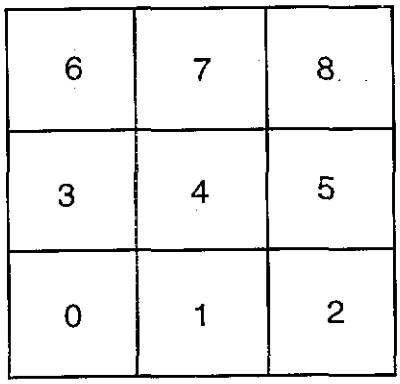

Figure 4.3-1 An active arena of terrain excavation

During a simulation, the center of the blade is always located at the central patch (patch 4 in the figure above). When the bulldozer moves forward or backward, the blade center leaves the central patch. In order to maintain the bulldozer in the center patch, three patches are swapped out from the active arena and patch requests are issued by the SIM host. These requests are received by the CIG host and three new patches are returned to the SIM host. The active arena is then re-assembled by the SIM host. Thus, the dimensions and number of patches in the active arena remain the same.

In implementing a bulldozer model, the active arena is represented by an mxn array of elevation posts maintained by the bulldozer simulation in the SIM host. All dynamic soil computations are performed on this elevation post array. When the bulldozer moves with its blade down, the terrain surface inside the active arena is changed. Modified elevation posts are sent from the SIM host to the CIG host. The CIG host then updates the

texture and vertices of polygons in the runtime terrain database in order for the changes to be viewed through the visual system. To reduce the number of messages from the SIM host to the CIG host, active patches are checked and only those with elevation post changes are sent to the CIG host.

In this approach, the data rates through the SIM host/CIG host interface per simulation frame may be very high due to new active patch data being transmitted from the CIG host to the SIM host. Recall that three patches are swapped out and three new patches are brought into the active arena when the bulldozer's blade moves across a patch boundary. During a simulation, activities of an excavating machine may coincide with a patch boundary. If the vehicle motion is oscillating across a boundary, active patches would be swapped in and out continuously, resulting a heavy network traffic.

To remedy this problem, we maintain a data structure in the SIM host to temporarily store those active patches which are just swapped out of the active arena, or likely to be used in the near future. (An algorithm was developed to predict which patches are likely to be used in the next few simulation frames.) All terrain patches brought to the SIM host are kept in a tree structure to provide not only terrain surface information required by the dynamic soil slippage and manipulation model but also geometrical data for the vehicle's terrain following. As the simulation exercise continues, the distance between some terrain patches and the center of the bulldozer's blade exceed a threshold. These patches are then discarded by the SIM host.

Maintaining some temporary storage in the SIM host increases the amount of data redundancy and causes potential data consistency problems. These drawbacks, however, are minimized by careful design and implementation. The payoff for this extra work, however, is great: the mean number of message bytes per simulation frame was reduced by two orders of magnitude.

4.4 Polygon Reduction

As discussed earlier, a load module in the run-time database is tessellated into hierarchically-structured grids at different partitioning levels when the bulldozer lays its blade down. These smaller grids create a greater polygon load for the graphics pipeline of the computer image generator. As a simulation proceeds, the number of polygons representing the fine details of the terrain surface grows. If the polygon throughput reaches a threshold, a terrain relaxation procedure is called to reduce the polygon density. In this section, we describe the terrain relaxation algorithm used for real-time relaxation of a terrain patch.

4.4.1 Relaxation Algorithm. To achieve a speed improvement in the rendering process of the image generator, the terrain relaxation algorithm is used to reduce the number of polygons required to define the terrain surface of a regularly spaced grid of elevation points. Without terrain relaxation, the surface definition consists of a list of triangles (2 triangles for each 2x2 set of elevation points). The number of triangles required to define a terrain surface for a set of nxm elevation points without terrain relaxation is: $2*(n-1)*(m-1)$.

The terrain relaxation procedure creates a list of polygons that omit those elevation points which do not add important geometrical surface information to the overall appearance of the terrain surface. An automatic polygon reduction is performed in relatively co-planar areas within a regularly spaced grid of elevation data points. A programmable tolerance value is used in the co-planarity calculation to achieve varying levels of polygon reduction, dependent upon the desired accuracy of the surface definition.

In addition, any given elevation point can be assigned to be fixed in elevation, i.e., the co-planarity calculation uses a tolerance of 0.0 to determine if that elevation point is within the plane being examined. This allows the relaxation procedure to retain some physical properties of the terrain surface, such as ridge lines or shallow ditches. Similarly, vehicle tracks or other polygons with attributes related to their appearance (color, texture or shading) are tagged to be fixed so that these features are not altered during the relaxation process.

The borders of the overall elevation grid data point set are always assumed to be fixed. This allows adjacent terrain patches at the bottom level of tessellation to be relaxed independently, but to still have an exact correspondence in their adjoining surface definition. Failure to fix the borders would create terrain separation at the patch boundaries.

4.4.2 Relaxation Strategies. There are two different strategies to determine when to relax and which terrain patches to relax: time-based strategy and distance-based strategy.

Time-based: each terrain patch at the bottom level of tessellation receives a time stamp when it is created from the terrain partitioning. It is updated with the current time whenever the patch is

modified. Time stamps are routinely examined against a threshold. Those patches whose time stamp exceeds the threshold are chosen as objects for relaxation.

Distance-based: each terrain patch at the bottom level of tessellation is tagged with the distance to the center of the bulldozer's blade when it is created. This distance recalculated as the bulldozer travels. These distances are routinely compared to a threshold. Those patches whose distance exceeds the threshold are chosen as objects to relax.

In our implementation on the Loral GT100, the distance-based strategy was used. With 30 meters as the distance threshold and 0.1 meter as the relaxation tolerance, the number of polygons in the run-time database remains within a manageable range.

5. Conclusion and Future Work

A real-time interactive bulldozer simulation demonstrated dynamic terrain capability on the GT100 image generator as part of the Institute for Simulation and Training Dynamic Terrain Demonstration at I/ITSEC '93. The virtual bulldozer, driven by a Spaceball™ interface, can interactively modify any standard SIMNET terrain database at any freely-chosen location.

Fundamental problems remain before dynamic terrain capability can be fielded for large scale simulation. Even with the terrain relaxation approach used, exercises with many entities changing terrain will ultimately create so many polygons that CIGs are unable to render views and non-visual entities become computationally overburdened. New methods are needed for aggregating polygons while maintaining geometry and minimizing polygon density. Arbitration issues must also be considered for multiple entities changing a region simultaneously. Finally, the design of a system architecture which is scaleable and reliable must be addressed.

One promising method for aggregation of polygons is the use of triangulated irregular networks (TINs), which can represent the terrain relief with much lower polygonal density than can regular grids. We intend to investigate real-time methods for TINning modified terrain in future efforts.

6. Acknowledgement

The authors would like to thank the US Army Simulation, Training and Instrumentation Command (STRICOM) and the Institute for Simulation and Training (IST), who sponsored this work (contract N61339-92-K-0001).

7. References

[Balovnev83] Balovnev, V.I., *New Methods for Calculating Resistance to Cutting of Soil*, Translated from Russian, Published for the U.S. Dept. of Agriculture and the National Science Foundation, Washington, D.C., 1983.

[Chowdhury78] Chowdhury, R. N., *Slope Analysis*, Elsevier North-Holland Inc., 1978.

[CIG/SIM Comm 90] *GT100 CIG to Simulation Host Interface Manual,* BBN Systems and Technologies Corp., March 1990.

[Li93a] Li, X., *Physically-Based Modeling and Distributed Computation for Simulation of Dynamic Terrain in Virtual Environments*, Ph. D. Dissertation, University of Central Florida, March, 1993.

[Li93b] Li, X. and Moshell, J.M., "Modeling Soil: Realtime Dynamic Models for Soil slippage and Manipulation," *Proceedings of SIGGRAPH '93* (Anahiem, CA, Aug. 1-6, 1993). In *A C M Computer Graphics*, vol. 27, 361-368.