Automated Generation of Large-Scale Dynamic Terrain in 3D Animation of Simulated Construction Processes

Vineet R. Kamat[1] and Julio C. Martinez[2]

[1]*Assistant Professor, Dept. of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI 48109, USA. Tel:734-764-4325 E-mail: vkamat@umich.edu*
[2]*Associate Professor, Dept. of Civil and Environmental Engineering, Virginia Tech, Blacksburg, VA 24061, USA. Tel:540-231-9420 E-mail: julio@vt.edu*

ABSTRACT

The presented work extends the state-of-the-art in scientific simulation-driven 3D construction process visualization. We present a tool, ViTerra that engineers can use to automatically generate photorealistic, digital, 3D terrain CAD databases to represent construction jobsites in visualizations of discrete-event construction simulations. The work capitalizes on the availability of detailed topographical and aerial imagery data that is readily available in digital format from several government and private organizations. In addition to automated 3D terrain generation techniques, ViTerra implements animation methods to describe the evolution of virtual jobsites by depicting deformations to the loaded terrains in response to common construction operations such as earthmoving and trenching. Our first contribution alleviates the need to manually create 3D terrain models for use in animations as has been the case hitherto. Our second contribution 1) enhances realism in the visualization of several common construction operations, and 2) allows the visualization of several other construction processes that were hitherto impossible to describe in 3D virtual worlds.

KEYWORDS

3D Visualization, Construction Operations, Digital Elevation Maps, Discrete-Event Simulation, Dynamic Terrain, Virtual Reality.

INTRODUCTION

The ability to see a 3D animation of a construction operation that has been modeled and simulated using discrete-event simulation (DES) allows for three very important things: 1) The developer of the simulation model can confirm that there are no errors in the coding (verification); 2) The domain experts, field personnel, and decision makers can discover differences between the way they understand the operation and the way the model developer understands it (validation); and 3) The model can be communicated effectively offering decision makers valuable insight on operational details that would otherwise be non-quantifiable and non-presentable. This coupled with verification and validation, makes simulation models credible and encourages their use in making decisions (Law and Kelton 2000, Kamat and Martinez 2003).

The authors' ongoing visualization research efforts focus on designing automated, process simulation-driven methods to visualize construction products and the processes involved in building them. In addition to visualizing what components are built where and when, the efforts are concerned with visualizing who builds what and how by depicting the interaction between involved machines, resources, and materials. A tangible outcome of this research is the VITASCOPE visualization system. VITASCOPE is an acronym for VIsualizaTion of Simulated Construction OPErations. VITASCOPE is a user-extensible 3D animation language designed specifically for visualizing modeled construction operations in smooth, continuous, dynamic 3D virtual worlds. A limited subset of the VITASCOPE language and the corresponding prototype implementation were referred to as the Dynamic Construction Visualizer (DCV) in some prior publications (e.g. Kamat and Martinez 2001).

This paper describes an extension to the VITASCOPE visualization system. We present a tool, ViTerra that engineers can use to automatically generate photorealistic, digital, 3D terrain CAD databases to represent construction jobsites in visualizations of discrete-event construction simulations. The work capitalizes on the availability of detailed topographical (e.g. Digital Elevation Map – DEM) and aerial imagery (e.g. National Aerial Photography Program - NAPP) data that is readily available in digital format from several government (e.g. United States Geological Survey - USGS) and private organizations.

In addition to automated 3D terrain generation techniques, ViTerra implements animation methods that allow engineers to describe the evolution of virtual jobsites by depicting deformations to the loaded terrains in response to common construction operations such as earthmoving and trenching. By implementing techniques that integrate multiple sources of digital information and automatically generate photorealistic virtual jobsite terrains, our first contribution alleviates the need to manually create 3D terrain CAD models for use in visualizations as has been the case hitherto. By designing methods to describe the deformation and evolution of instantiated terrain CAD databases, our second contribution 1) enhances realism in the visualization of several common construction operations, and 2) allows the visual depiction of several other simulated construction processes that were hitherto impossible to describe in 3D virtual worlds.

CHALLENGES

The design of simulation-driven methods to automatically generate and then visualize the deformation and evolution of jobsite terrains in response to common construction processes in smooth, continuous, dynamic, 3D virtual worlds presents numerous interesting challenges. Digital elevation and aerial imagery data is readily available for the entire United States and several other parts of the world. However, this data is archived in several different digital formats at varying levels of detail (Burrough and McDonnell 1998). In order to automatically generate virtual jobsite terrains from disparate sources of archived digital data, we had to implement techniques to 1) parse (i.e. read) and interpret streams of elevation and imagery data in all commonly available formats, and 2) convert interpreted elevation and imagery data into a standard internal representation that could be visually depicted in 3D virtual worlds and could also be locally modified as needed to describe actions such as terrain deformation.

The resultant locally deformed shape and appearance of a virtual terrain in response to an animated elemental construction task (e.g. digging) cannot be

determined a-priori by animation-authoring processes (e.g. discrete-event simulation models) or by the visualization engine itself. Each deformation a virtual terrain must undergo in response to an animated construction task (e.g. digging) depends on 1) the type, size, and configuration of the involved piece of virtual equipment (e.g. backhoe), and 2) the amplitude of the motion of its components (e.g. boom, stick, bucket) in the particular animated instance of that task.

For instance, simulation models that describe earthmoving operations in VITASCOPE's 3D animation language indicate the location (as a 3D coordinate) where a virtual piece of equipment (e.g. backhoe) must "dig" in each loading pass (Kamat 2003). However, the exact trajectory that a piece of equipment's digging implement (e.g. bucket edge) will follow in each virtual digging stroke cannot be determined beforehand by either the authoring process (i.e. simulation model) or the visualization engine. The computation that determines the shape of an evolving terrain in response to animated construction processes must thus be performed during animation in real-time and is solely the responsibility of the visualization engine.

Finally, from the implementation point, virtual terrains are computationally intensive to maintain and render. Terrains are typically composed of several thousand textured polygons that in the absence of any optimizations would create a graphics pipeline bottleneck (Garland and Heckbert 1995). Rendering terrains in real time at interactive frame rates has thus been an important and long-studied problem in the field of computer graphics. In order to depict large pieces of dynamic terrain in 3D animations, the number of polygons that the graphics pipeline needs to display must be limited without compromising the detail of the terrain. This is particularly important in the visualization of construction operations because unlike many other visual simulations (e.g. flight simulators), the terrain on a virtual construction jobsite is seen and manipulated at close range. In construction, since most of the action takes place on the ground, the simplification of the terrain database to improve performance cannot compromise the fidelity of the terrain's appearance.

The issues we addressed in this work were thus four-fold. First, we implemented techniques to automatically generate terrain CAD databases by combining data from disparate elevation and imagery data sources into a unified, consistent data set. Second, we investigated feasible data structures to internally maintain and locally manipulate (on demand) the generated terrain databases. Third, we designed methods to compute the amount, location, and the resultant shape of deformations that virtual terrains undergo in response to animated construction processes and to apply those deformations to the terrain databases in real-time. Finally, we identified and adopted computationally efficient techniques of maintaining and rendering the dynamic 3D terrain CAD databases in a way that precludes performance bottlenecks in the graphics pipeline and helps maintain interactive animation frame rates. These issues were tightly intertwined and related.

GENERATING 3D TERRAINS FROM ARCHIVED DIGITAL DATA

A commonly used format of digitally representing a terrain surface is the Digital Elevation Model (DEM). Strictly speaking, a DEM specifically refers to a raster or regular grid of spot heights (USGS 2001). DEMs consist of a sampled array of elevations for a number of ground positions at regularly spaced intervals to describe an axis-aligned grid of terrain. The distances between the sampled positions (i.e. the grid spacing) defines the resolution of a DEM and is the major determinant of its accuracy (USGS 2001).

The DEM format is a compact and efficient way of representing terrain surfaces and lends itself well to computer computations. A variety of DEMs are readily available from several government and private organizations. For instance, the United States Geological Survey (USGS) freely provides DEM data for the entire United States (USGS 2002a). The best resolution commonly available in the USGS DEMs is 10m, with a vertical resolution of 1m. In addition, several private organizations (e.g. quarry owners) regularly update and maintain detailed (i.e. with fine resolution) DEMs of their property for use in planning, quantification, and record-keeping (Zalubowski 2002).

A DEM is generally archived in one of two common digital formats. The standard method of archiving a DEM is through the use of numerical height maps. A numerical height map is simply a two dimensional array of numerical values. Each numerical value in the array indicates the elevation of the terrain's surface at that array position.

Another common method of archiving DEMs is through the use of digital, grayscale image height maps. In grayscale images that represent DEMs, the brightness of each image pixel corresponds to the height of the terrain at that point. Generally, dark regions on the image represent lower elevation values and lighter regions indicate high elevation values. The calibration of the grayscale (i.e. the numerical elevation values of pure black and pure white regions) is often implementation dependent.

Constructing the Terrain Geometry

The geometry of uneven, three dimensional surfaces is generally represented on the computer as a continuous mesh of polygons (usually triangles) (Woo et al. 1997). The process of converting DEM data into a 3D terrain CAD database thus requires interpretation of the input data and construction of a continuous, 3D polygon mesh corresponding to that data. The coordinates of each vertex in the constructed mesh can be directly obtained from the DEM if it is archived as a numerical height map. In cases where the DEM data is archived as a grayscale image, the extraction of coordinates from the height map requires two, implementation-dependent calibration factors. The first is the numerical value of the horizontal spacing (i.e. distance) between sampled positions. This value defines the horizontal resolution of the grid by indicating how the distance between adjacent pixels in the input image must be interpreted. The second calibration factor required to extract 3D coordinates from an image height map is the vertical scaling factor. This factor defines the vertical resolution of the image height map.

For instance, if the color scale of a particular implementation assigns 0.0 to pure black and 1.0 to pure white and the vertical scaling factor is 1000m, then a pure white pixel on the grayscale image height map is interpreted to have an elevation of 1000m. A pure black pixel is similarly interpreted as a 0m elevation value. Shades of gray (color values ranging between 0.0 and 1.0) are linearly interpolated to decipher the elevations indicated by each pixel in the image height map. Once the coordinates of the vertices are extracted from the image height map, a 3D polygon mesh can be constructed in a similar manner to that of numerical height maps. A grayscale image height map must thus be converted to a numerical height map before 3D mesh vertex coordinates can be extracted from the data. In both cases, the extraction of vertex coordinates from the height maps and construction of a corresponding 3D mesh generate a computer-interpretable geometrical shape of the terrain segment that is archived in the input DEM.

Specifying the Terrain Appearance

The 3D polygon mesh constructed from DEM elevation data merely specifies the shape of the geometrical object that represents a patch of terrain. In order for this mesh to "look" like a real terrain when instantiated in virtual worlds, we need to specify information that indicates its appearance. A common computer graphics technique for determining the appearance of 3D polygonal objects is to use texture mapping. Texture mapping is a classical technique for image synthesis wherein a two-dimensional (2D) image (called a texture map) is superimposed on a three-dimensional (3D) geometric object (Haeberli and Segal 1993).

The visual outcome of this process is that the 3D geometrical object acquires a surface appearance similar to that of the superimposed 2D image. Conceptually, texture mapping is thus the electronic equivalent of applying wallpaper, paint, or veneer to real world objects. Texture mapping is almost exclusively used to specify the appearance of 3D terrain CAD objects in virtual environments. As shown in Figure 1, this is accomplished by superimposing a 2D image that represents an aerial view over the corresponding 3D geometrical terrain mesh. The geometry (i.e. the 3D mesh) and the appearance (i.e. the texture map) together define the visual representation of terrain objects in visualizations. In the presented work, where the terrain's geometrical representation can be constructed from actual DEM data, texture mapping presents interesting possibilities of specifying the appearance of instantiated 3D terrain objects.

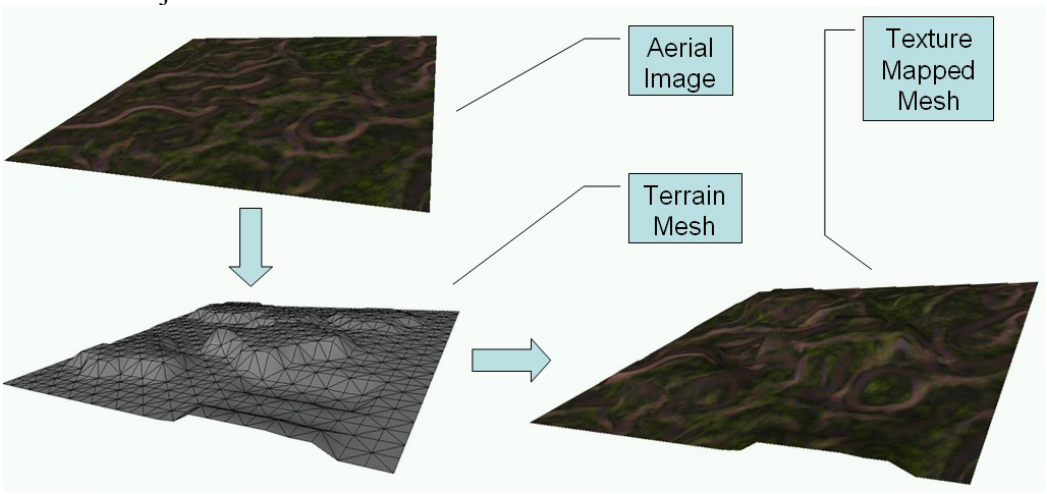

Figure 1: Texture Mapping 3D Terrain Meshes

For instance, constructed terrain geometry can be overlaid by satellite or aerial imagery corresponding to the same geographic area represented in the input DEM. This powerful technique can yield impressive images and useful results. Similar to elevation data, several government (e.g. USGS, National Aeronautics and Space Administration - NASA) and private agencies have archived digital, high-resolution aerial imagery for the entire United States. In addition, since the applied texture map is simply a 2D image representing the overhead view of the modeled area, custom 2D images may readily be used to drape constructed 3D terrain meshes. This is useful in cases where actual aerial color imagery at the desired resolution is inaccessible or when specific, synthetic appearances must be applied to instantiated 3D terrains during visualization.

COMPUTING AND DEPICTING TERRAIN DEFORMATIONS

Figure 2 outlines the strategy we adopted to compute deformations to virtual jobsite terrains in construction process visualizations. This scheme relies on the authoring discrete-event simulation models to communicate construction tasks (e.g. digging) to virtual pieces of instantiated construction equipment (e.g. backhoes). Then, during visualization of those communicated construction tasks, we continuously monitor the position of the involved virtual equipment's digging implement (e.g. bucket edge) to detect whether or not the virtual terrain has been penetrated.

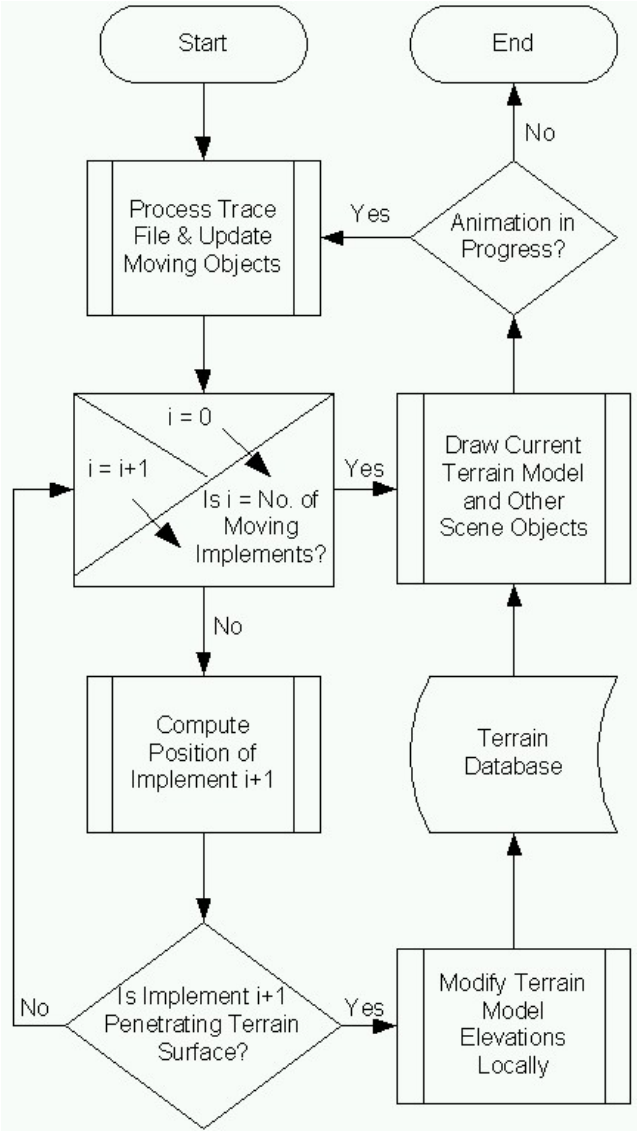

Figure 2: Terrain Deformation Computation Scheme

To dynamically update deformation changes to the current 3D terrain database when surface penetration is detected, we adopted a generalized form of the simple implement-terrain interaction model outlined in Lipman and Reed (2000). At each instant during the visualization of a communicated digging stroke, we compute the global position of the involved digging implement (e.g. bucket edge) inside the 3D virtual world. This is accomplished using standard mathematical notations and results from elementary computer graphics literature. Once an implement's position is determined, we compare it's height to the elevation of the virtual terrain below. If the elevation of the terrain at that point is greater than the implement's current height, we conclude that the implement has penetrated the terrain's surface.

To reflect the outcome of any detected surface penetration in the 3D terrain database, we then set the elevation value of the terrain model at that point to be equal to the digging implement's current global height. The adopted regular grid terrain database storage model makes such local elevation modification straightforward. We simply update the numerical elevation value of the pertinent point(s) in the regular grid terrain model being maintained. The visual outcome of continuously performing this computation and updating the terrain database during visualization is a terrain deformation whose shape conforms to the digging implement's trajectory during the particular digging stroke.

Deposition (i.e. dumping) of dirt is similarly computed and depicted during the visualization of dumping strokes. At each instant, we monitor the position of a piece of equipment's bucket during dumping. We then increase the elevations of projected terrain points along the trajectory followed by the involved bucket as it dumps dirt. The visual outcome in this case is the depiction of a heap whose shape and size is proportional to the virtual bucket dumping dirt.

Influence of Grid Resolution on Deformation Computations

The results obtained by the presented computation scheme are significantly influenced by the grid resolution of the 3D terrain model being manipulated. In particular, the technique only produces accurate results if the horizontal resolution of the maintained terrain grid is significantly dense compared to the width of virtual digging implements that deform the terrain models. In general, finer the terrain grid model, better are the produced visual results (Lipman and Reed 2000). This is obvious because as grid resolution increases, the computation scheme can manipulate more, closely-spaced points on the terrain models in depicting the shape of deformations. On the other extreme, the algorithm may have no terrain points to manipulate if the resolution of the grid is larger than the width of a virtual digging implement. The elevation data from which 3D terrain models are intended to be automatically constructed is, however, rarely available in the order of such high grid resolutions.

For instance, the highest horizontal resolution available in most USGS DEM data is 10m, which is significantly large compared to the width of digging implements (e.g. bucket edges) on common construction equipment pieces. We address this issue by constructing high resolution meshes while building 3D terrain models from input elevation data. In particular, we construct a mesh of high, user-defined resolution from the comparatively lower resolution input elevation data. This is accomplished by simply interpolating data points along the two horizontal axes of the input data. This operation generates additional, intermediate elevation data points so that the resolution of the resulting grid is sufficiently finer than the width of virtual digging implements expected to be utilized in particular visualizations.

VITERRA

The ViTerra add-on for VITASCOPE implements the techniques that automatically generate and then manipulate 3D terrain models in visualizations of simulated construction processes. ViTerra extends VITASCOPE's animation language by implementing the animation statements presented in Table 1. In particular, the add-on defines animation language statements that automatically construct 3D terrain models by combining disparate elevation and imagery data sources.

Table 1: Usage of ViTerra Statements

Statement	Usage
`TERRAIN [ElevationDataSource] [TextureDataSource];`	Construct a 3D terrain model from the indicated elevation (geometry) and texture (appearance) data sources
`TERRAIN.RaiseElevation [X] [Z] [DeltaY];`	Raise the elevation of the specified point on the terrain by the indicated amount
`TERRAIN.LowerElevation [X] [Z] [DeltaY];`	Lower the elevation of the specified point on the terrain by the indicated amount
`TERRAIN.SetElevation [X] [Z] [TargetY];`	Set the elevation of the specified point on the terrain to the indicated value

In addition, the add-on presents animation statements to manipulate the created terrain databases by explicitly changing the elevation at any point on the instantiated terrains. These latter methods are, however, not intended to be directly used in animation trace files by visualization-authoring simulation models. Instead, they are implemented so that other VITASCOPE extensions (add-ons) can manipulate instantiated 3D terrains by executing these statements from within their add-on modules.

Terrain Model Construction

ViTerra automatically constructs a 3D terrain model by combining the data sources specified as arguments to the TERRAIN statement. In particular, the TERRAIN statement takes two arguments as input. The first argument is a string that specifies the source of the digital elevation data (i.e. DEM) to be used in constructing the terrain model geometry. The second argument is a reference to the aerial image that is to be draped on the constructed geometry to describe its appearance. An example of a TERRAIN statement is presented below:

```
TERRAIN '4349CATD.DDF,10,1' '4349Tex.JPG';
```

In the above statement, the terrain geometry is to be constructed from a DEM that is archived in the popular USGS Spatial Data Transfer Standard (SDTS) format (USGS 2002b). In this case, the elevation data argument specifies the root SDTS filename (*CATD.DDF), the horizontal resolution of the input DEM (10m), and its vertical resolution (1m). The second argument to the statement specifies the image (in standard JPEG format) that is to be draped over the constructed terrain model to describe its appearance.

ViTerra can construct the geometry of 3D terrain models from several different DEM sources (numerical and grayscale image height maps). Appendix A provides a listing of all DEM input formats ViTerra supports. ViTerra exploits the services of the Geospatial Data Abstraction Library (GDAL) for parsing and interpreting input DEM height fields. GDAL (Warmerdam 2003) implements algorithms to translate raster DEMs in several common geospatial data formats presented in Appendix A.

ViTerra can similarly set the appearance of constructed terrain geometries from image files archived in several common formats. ViTerra relies on the Simple DirectMedia Layer (SDL) library (Lantinga 2003) for reading the texture image files specified as arguments to the TERRAIN statement. A specified input texture image is clamped to the corners of the constructed 3D terrain geometry. Thus, in cases where the specified texture is an actual aerial image of the terrain area, we must ensure the exact lateral correspondence between the image corners and the input DEM.

Terrain Model Manipulation

The algorithms that manipulate ViTerra terrains to visually depict surface deformations in response to performed construction tasks are implemented in the VITASCOPE KineMach add-on. KineMach implements "smart" pieces of virtual construction equipment that can be instantiated and manipulated in visualizations using simple text statements in a higher-level, contextual, construction terminology (Kamat 2003). Engineers can use KineMach provided statements to instantiate multiple pieces of equipment such as backhoes and instruct them to perform virtual construction processes using a high-level construction work terminology.

Figure 3 presents an animation trace with statements that instantiate a virtual backhoe and instruct it to perform construction tasks (i.e. digging and dumping). During the performance (i.e. visualization) of each digging and dumping stroke, KineMach monitors the position of the virtual backhoe's bucket. KineMach then internally invokes ViTerra's elevation-setting statements (TERRAIN.SetElevation) to instantly modify the 3D terrain model at points corresponding to the backhoe bucket's trajectory. In addition, at each terrain point modified in response to digging, ViTerra splats (i.e. pastes) a texture that gives the 3D terrain an appearance of digging implement ridge marks.

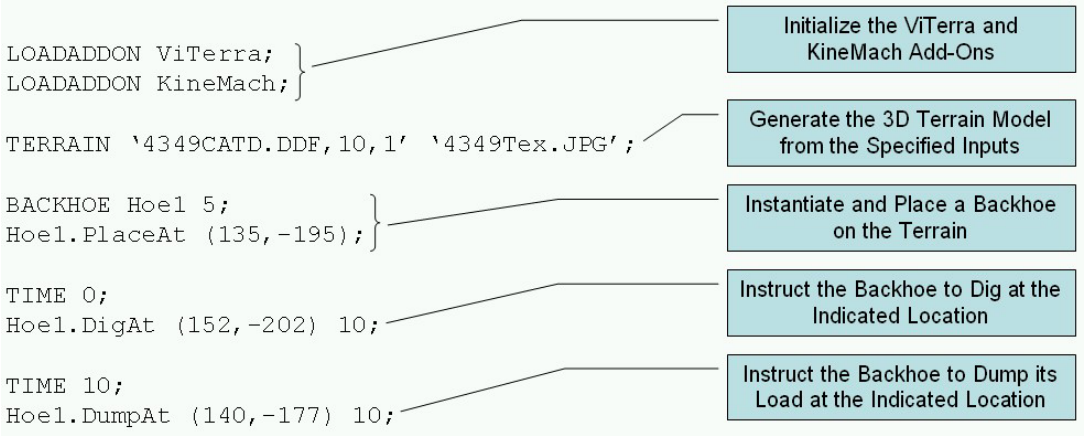

Figure 3: Communication of Excavation Tasks

Figure 4 presents a strip of animation snapshots taken during the visualization of the virtual digging task described above. The snapshots presented are not successive computer frames observed during visualization. The discretely captured frames are displayed in a filmstrip format merely to depict a sense of motion. The smooth motion of the backhoe during visualization and the corresponding, smooth deformation (i.e. evolution) of the virtual terrain cannot be fully captured in static snapshots. Only the animation can convey that information.

Terrain Model Rendering

ViTerra utilizes the Demeter terrain engine for the data structures required to store (i.e. maintain) constructed 3D terrain models on the computer as regular grid height fields. The Demeter library (Fowler 2002) implements data structures and algorithms to visualize vast 3D terrain models in virtual environments using OpenGL (Woo et al. 1997). ViTerra also relies on Demeter to depict a visually accurate and detailed representation of the current 3D terrain database at interactive frame rates. In particular, ViTerra adopts the Demeter-implemented view-dependent, adaptive triangulation algorithm to render the dynamic 3D terrain models it maintains. At each frame in the visualization, a high-fidelity image of the current 3D terrain model with the fewest possible polygons (i.e. triangles) for that particular view is rendered.

SUMMARY AND CONCLUSIONS

This research extends the state-of-the-art of simulation-driven construction process visualization and puts in place the technology to 1) automatically create 3D terrain databases (i.e. CAD models) of construction sites from archived digital elevation and imagery data, and 2) dynamically manipulate constructed terrain models to visualize landscape deformations in response to virtual construction processes (e.g. digging).

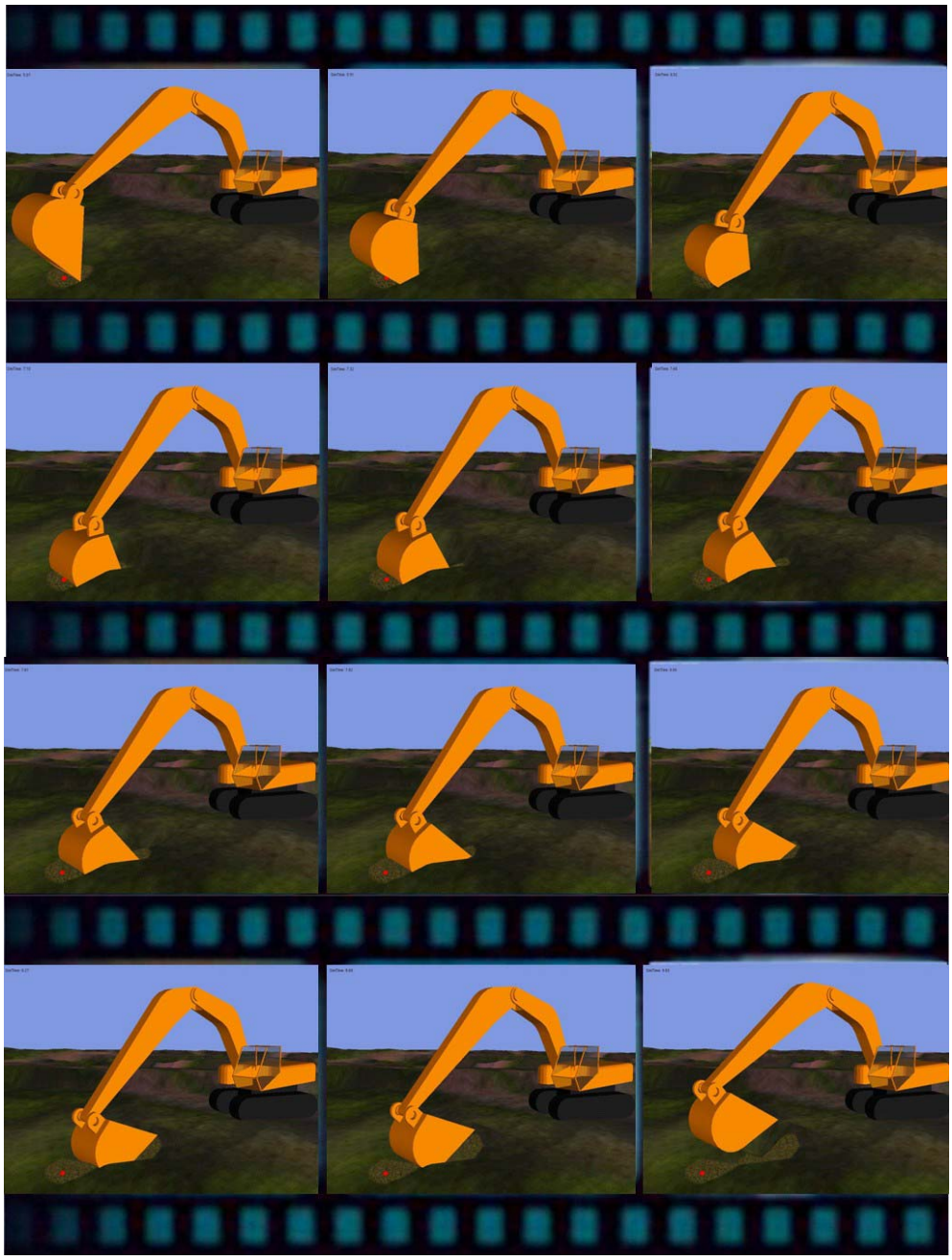

Figure 4: Animation Snapshots of Virtual Digging

The design of techniques to automatically generate dynamic, deformable 3D terrain databases requires the integration of four key technologies:

1) Combination of data from disparate digital elevation and imagery data sources into a unified, consistent, and accurate 3D terrain database.
2) Efficient data structures (i.e. storage model) to internally maintain and locally manipulate the constructed 3D terrain databases on demand.
3) Effective equipment-terrain interaction model to compute the amount, location, and resultant shape of terrain deformations in response to performed virtual construction tasks.
4) Fast rendering procedures to draw the terrain models in 3D virtual worlds such that the results look visually convincing, and can be animated in real-time.

Commonly available DEM elevation data is archived in simple, regular grid numerical or grayscale image height maps. The geometry of a 3D terrain database for any geographic area can be extracted from the readily available DEM elevation data for that region. An actual or synthetic aerial image of the corresponding area can then be superimposed on the constructed geometry to specify the appearance of the constructed 3D terrain model. The availability of input elevation data in regular grid formats presents several distinct advantages. First, it allows straightforward extraction (from DEM archives) of 3D positional coordinates required to construct the geometric meshes that represent the terrain surfaces. This facilitates and guides the critical choice of the storage model to be adopted in maintaining constructed 3D terrain databases in a dynamic, modifiable format on the computer during visualization.

The adopted regular grid height field storage model, in turn, guides the design of the interaction scheme between the 3D terrains and virtual pieces of construction equipment. In particular, the adopted storage model facilitates the design of a simple deformation computation scheme that monitors virtual digging implements during visualization and locally manipulates the terrain model elevations to visually depict the deformed landscape. The regular grid height fields also form the foundation of hierarchical data structures that are required to be built by rendering algorithms for efficiently drawing 3D terrains at interactive frame rates. The implemented view-dependent, dynamic terrain triangulation and rendering algorithm and the visually appealing results it displays demonstrates that the approach is not only possible, but also very effective.

ACKNOWLEDGMENTS

The authors gratefully thank the National Science Foundation (CAREER and ITR programs) for supporting the presented work. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

Burrough, P., and McDonnell, R. (1998). Principles of Geographic Information Systems, Oxford University Press, New York, NY.

Fowler, C. (2002). The Demeter Terrain Engine. Available: <http://www. terrainengine. com> (Jan. 20, 2003).

Garland, M., and Heckbert, P.S. (1995). "Fast Polygonal Approximation of Terrains and Height Fields", Technical Report CMU-CS-95-181, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. Available: <http://graphics. cs.uiuc.edu/~garland/papers/scape.pdf > (Jan. 20, 2003).

Haeberli, P., and Segal, M. (1993). "Texture Mapping as a Fundamental Drawing Primitive", Technical Article, GRAFICA Obscura. Available: <http://www.sgi. com/grafica/texmap/index.html > (Jan. 20, 2003).

Kamat, V.R. (2003). VITASCOPE: Extensible and Scalable 3D Visualization of Simulated Construction Operations, Ph.D. Dissertation, Virginia Tech, Blacksburg, VA.

Kamat, V.R., and Martinez, J.C. (2003). "Validating Complex Construction Simulation Models Using 3D Visualization", Systems Analysis Modelling Simulation, Vol. 43, No. 4, Taylor & Francis Group, London, United Kingdom, 455-467.

Kamat, V.R., and Martinez, J.C. (2001). "Visualizing Simulated Construction Operations in 3D", Journal of Computing in Civil Engineering, Vol. 15, No. 4, ASCE, Reston, VA, 329-337.
Lantinga, S. (2003). The Simple DirectMedia Layer Library. Available: <http://www.libsdl.org> (Jan. 20, 2003).
Law, A.M., and Kelton, W.D. (2000). Simulation Modeling and Analysis, 3rd Ed. McGraw-Hill, New York, NY.
Lipman, R., and Reed, K. (2000), "Using VRML in Construction Industry Applications", Proceedings of the 5th Symposium on Virtual Reality Modeling Language, Association for Computing Machinery, New York, NY, 119-124.
United States Geological Survey (2002a). USGS Geographic Data Download. Available: <http://edc.usgs.gov/geodata/> (Jan. 20, 2003).
United States Geological Survey (2002b). The Spatial Data Transfer Standard. Available: <http://mcmcweb.er.usgs.gov/sdts/> (Jan. 20, 2003).
United States Geological Survey (2001). USGS Digital Elevation Model Data. Available: <http://edc.usgs.gov/glis/hyper/guide/usgs_dem> (Jan. 20, 2003).
Warmerdam, F. (2003). The Geospatial Data Abstraction Library. Available: <http://remotesensing.org/gdal/> (Jan. 20, 2003).
Woo, M., Neider, J., and Davis, T. (1997). OpenGL Programming Guide, 2nd Ed. Addison Wesley, Reading, MA.
Zalubowski, J. (2002). Personal Communication, Boxley Materials, Blue Ridge, VA. URL: <http://www.boxley.com> (Jan. 20, 2003).

APPENDIX A: VITERRA SUPPORTED ELEVATION DATA INPUT FORMATS

USGS SDTS DEM (*CATD.DDF), USGS ASCII DEM (.dem)
Arc/Info ASCII Grid, Arc/Info Binary Grid (.adf)
BSB Nautical Chart Format (.kap)
CEOS (Spot for instance)
First Generation USGS DOQ (.doq), New Labeled USGS DOQ (.doq)
Military Elevation Data (.dt0, .dt1)
Eosat Fast Format
ERMapper Compressed Wavelets (.ecw)
ESRI .hdr Labeled Raster, ENVI .hdr Labeled Raster
Envisat Image Product (.n1)
FITS (.fits)
Graphics Interchange Format (.gif)
GRASS Rasters
TIFF / GeoTIFF (.tif)
Erdas Imagine (.img)
Atlantis MFF2e
Japanese DEM (.mem)
JPEG JFIF (.jpg)
Atlantis MFF
OGDI Bridge
PCI .aux Labeled
Portable Network Graphics (.png)
Netpbm (.ppm, .pgm)
SAR CEOS
X11 Pixmap (.xpm)
NOAA Polar Orbiter Level 1b Data Set (AVHRR)
Hierarchical Data Format Release 4 (HDF4)