Under the Hood of GeoVRML 1.0

Martin Reddy Lee Iverson Yvan G. Leclerc

SRI International

Abstract

GeoVRML 1.0 provides geoscientists with a rich suite of enabling capabilities that cannot be found elsewhere. That is, the ability to model dynamic 3-D geographic data that can be distributed over the web and interactively visualized using a standard browser configuration. GeoVRML includes nodes for VRML97 that perform this task; addressing issues such as coordinate systems, scalability, animation, accuracy, and preservation of the original geographic data. The implementation is released as open source and includes various tools for generating GeoVRML data. All these facilities provide geoscientists with an excellent medium to present complex 3-D geographic data in a dynamic, interactive, and web-accessible format. We illustrate these capabilities using real-world examples drawn from diverse application areas.

CR Categories and Subject Descriptors: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling; I.6.5 [Simulation and Modeling]: Model Development.
Keywords: 3-D geography, georeferencing, GeoVRML, terrain visualization, virtual environments.

1. Introduction

To date there is no system available that lets geoscientists integrate their geographic data directly into a three-dimensional (3-D) computer graphics scene graph and allows remote users to view the result interactively over the web [4, 10]. We describe the results of work to enable such capabilities. A host of new possibilities is subsequently generated for visualizing, disseminating, standardizing, and educating on geographic phenomena. The work forms part of the first deliverable of the GeoVRML working group within the Web3D Consortium. This deliverable consists of a recommended practice document accompanied by an open source implementation that anyone can download and use in their own scenes. The implementation includes a number of new nodes that extend the syntax of VRML97 to provide support for large-scale geographic applications. The goals of this work are to

1. Enable the transparent representation of geographic data to geoscientists, geographic information systems (GISs), and 3-D content modelers
2. Ensure the accurate representation and presentation of geographic data, from whole-earth scales (planetary visualizations) down to near-earth scales (city streets and mailboxes)
3. Support scalability to large geographic databases that are distributed over the web
4. Allow machine and human readability/writability
5. Provide a fast implementation

In the following sections, we describe many of the new features introduced by GeoVRML 1.0 and illustrate the unique capabilities that they provide geoscience researchers, visual simulation programmers, virtual heritage developers, educators, and others.

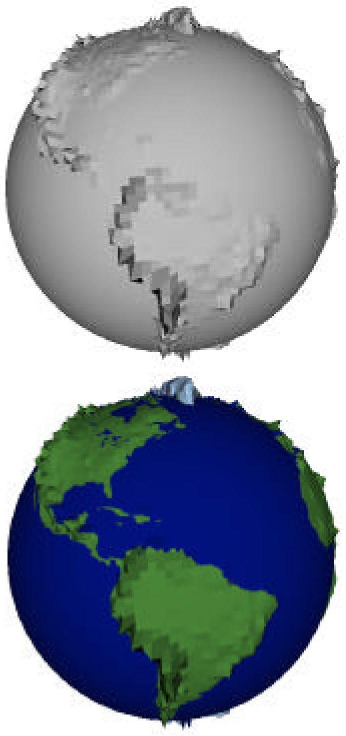


Figure 1. A **GeoElevationGrid** for the earth at 2 degrees resolution. All data are specified in latitude/longitude coordinates in the VRML file and then transparently converted to an ellipsoidal model by the node. A vertical exaggeration of 200 has been used to accentuate areas of high elevation. (Elevation data were taken from the USGS GTOPO30 dataset.)

reddy@ai.sri.com, leei@ai.sri.com, leclerc@ai.sri.com. Artificial Intelligence Center (AIC), SRI International, 333 Ravenswood Ave, Menlo Park, CA 94025, USA.

2. Representing and Visualizing Geographic Data

Computer graphics programmers and designers are used to dealing with a Cartesian (x, y, z) coordinate system for representing all 3-D models. However, geoscientists rarely use such a representation. Instead, they deal with data in a variety of different coordinate systems, each applicable for a different task. For example, the Mercator projection is often used for navigation or for maps of the equatorial region while the Lambert Conformal Conic is used for topographic maps by the U.S. Geological Survey (USGS) [11]. To engage geoscientists in 3-D computer graphics, we need to make it easy for them to visualize their data rapidly in a familiar form, as well as provide solutions for disseminating these data over the Internet. GeoVRML 1.0 introduces a small set of new nodes for VRML97 to perform these important tasks.

The **GeoElevationGrid** node provides the capability to define a grid of height values offset from an ellipsoid used to model the planet. It supports the specification of height fields in several geographic coordinate systems, such as geodetic (latitude/longitude) and Universal Transverse Mercator (UTM). VRML97 already provides an ElevationGrid node; however, in this node all values are offset from a single flat plane [6]. This is acceptable if the area being modeled is small in extent, for example, less than 1 km^2, but for larger areas the curvature of the earth becomes significant [1]. Figure 1 illustrates an example where we have created a GeoElevationGrid that spans the entire earth with all data specified in terms of a latitude/longitude grid. The grid of latitude/longitude heights is transparently converted into a Cartesian frame by the GeoElevationGrid node and accurately displayed with the correct degree of curvature for the earth (WGS84 ellipsoid). We have exaggerated the vertical heights by a factor of 200 to accentuate areas of high elevation. As a result, large mountain ranges such as the Rockies and the Andes are clearly distinguishable. This simple example provides an effective visualization for understanding the large-scale geological features of the planet.

The **GeoCoordinate** node enables the specification of coordinates by using geographic coordinate systems. This node can be used within standard VRML geometry nodes such as IndexedFaceSet, IndexedLineSet, or PointSet. The modeler can now specify coordinates in a system such as UTM and the GeoCoordinate node will transparently convert the data into a Cartesian frame and correctly position these coordinates in the global model. For example, a Global Positioning System (GPS) will normally output location as a latitude/longitude coordinate. With the GeoCoordinate node, we can insert these coordinates directly into a VRML file. Figure 2 illustrates a terrain model built using the GeoElevationGrid node with two tracks overlaid on the terrain representing the smoothed output from a GPS receiver. Abernathy and Shaw illustrated this general capability in 1997 [1]. However, we can now specify the data by using the actual latitude/longitude coordinates received from the GPS sensor, meaning that we preserve the original data and obviate the nontrivial task of manually transforming and co-registering this with the terrain. For example, in Figure 2, the terrain is represented in UTM coordinates and the GPS track is specified in latitude/longitude coordinates. The GeoVRML nodes seamlessly and accurately overlay these two data sources.

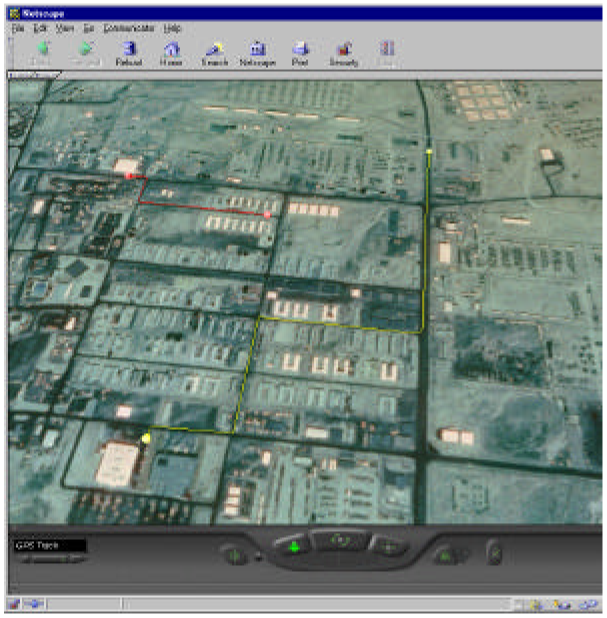
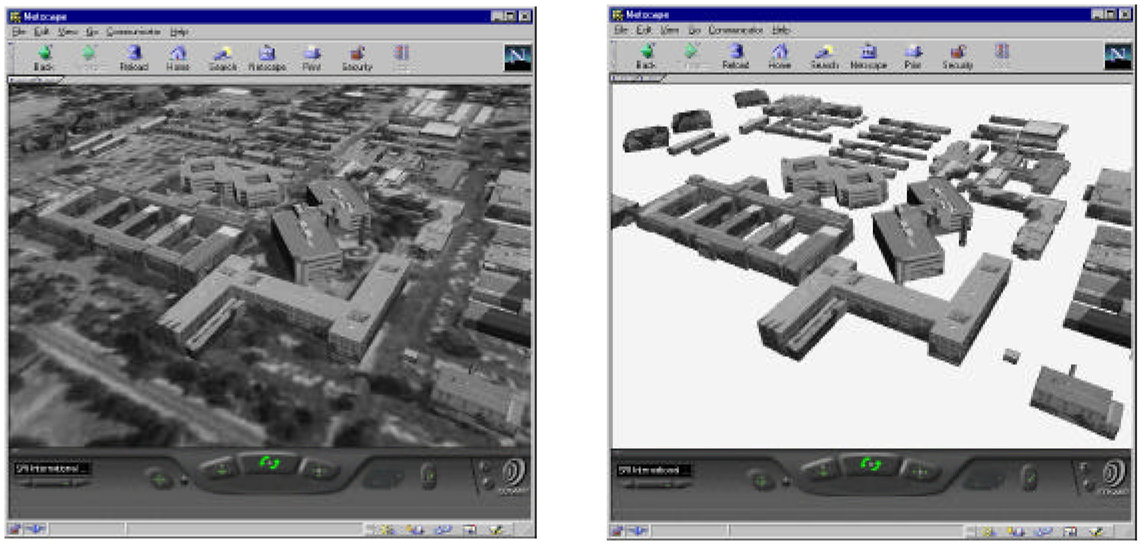


Figure 2. The **GeoCoordinate** node can be used to overlay vector data, such as GPS tracks, over a terrain model produced with **GeoElevationGrid**.

The **GeoLocation** node lets the user georeference an arbitrary VRML model, that is, locate it at a specific point on the earth. For example, with a VRML model of the Taj Mahal, the GeoLocation node can be used to specify the latitude/longitude location of that model on the surface of the earth. The Taj Mahal model would then be correctly displayed within the city of Agra, India, on a model of the globe. The GeoLocation node also orients the model correctly, depending upon its position on the earth, so that +Y is aligned with gravitational up, +Z points true north, and +X points east. This ensures that a model built using the standard VRML right-handed coordinate system will be placed on the earth so that its base is aligned with the surface of the planet. It is worth noting that the GeoLocation node serves a subtly different function from the GeoElevationGrid and GeoCoordinate nodes. The latter two nodes allow GIS type data to be represented in VRML, while the former allows VRML data to be represented to a GIS. Figure 3 illustrates the capabilities of the GeoLocation node by georeferencing a model of the SRI International main campus to an underlying terrain model of Menlo Park, CA.



(a) (b)

Figure 3. A texture-mapped site model of the SRI International campus that has been georeferenced onto an underlying terrain model. (a) Shows both terrain and site models, and (b) shows only the site model for clarity. Note the accurate alignment of the buildings with the underlying 1 m resolution USGS DOQ satellite imagery.

3. Resolution and Accuracy

It is a goal of GeoVRML to represent geographic data accurately and precisely to the user—an incorrect rendering of valid data should not be tolerated. One of the greatest sources of potential inaccuracy in the display of geographic data is that of limited floating point precision. VRML97 uses single-precision (32-bit floats [7]) to model and render all geometry [6]. However, we require at least double-precision (64-bit floats) to model geographic (e.g., geocentric) coordinates beyond a resolution of 10-100 m [9]. That is, in order to represent objects at the resolution of individual mailboxes (or smaller) over the range of the entire planet. We solve this problem by defining an absolute geographic origin (referred to as a **GeoOrigin**) in double-precision using MFStrings. Then for all double-precision geographic coordinates we take the difference between each coordinate and the GeoOrigin. The result gives a single-precision offset that can be used for faithful rendering (assuming a good origin was chosen). In effect, we specify the geographic location that will be used by the graphics system as its (0,0,0) point for rendering the scene. All of the GeoVRML nodes that deal with coordinates support the use of a GeoOrigin in this fashion, that is, GeoElevationGrid, GeoCoordinate, and GeoLocation.

To illustrate the importance of this capability, we use an example from the field of virtual heritage. Virtual heritage research focuses on the ability to recreate digital replicas of heritage sites to document and educate the population about our past, for example, [2]. We take the example of a simple temple structure consisting of an arrangement of columns on a raised platform. We assume that the accurate geospatial location of each of these columns has been recorded, for example with a differential GPS receiver. We then use a GeoLocation node to place each of these columns at their exact location. However, when we attempt to view this model, we see a number of significant inaccuracies in the rendering (Figures 6(b) and (d)). These inaccuracies occur because the resolution of the features is in the order of meters, but single-precision cannot represent this level of accuracy on a global scale. In Figures 6(c) and (e), the only difference is that we have added a GeoOrigin specification to each of the GeoLocation nodes, defining the center of the structure as the origin point. Now all data is within single-precision range of the origin and hence are rendered accurately. In addition, navigation is improved because camera jitter artifacts are removed and also examine mode rotations may occur around the origin point that we specify (depending upon the browser's implementation of examine mode).

4. Scalability Issues

Geographic data can be huge in size. Even a single USGS DEM (Digital Elevation Model) can produce a geometric model of around 1.4 million polygons [8]. It therefore becomes important to support multi-resolution techniques and to enable progressive streaming of data. Standard VRML97 does not provide the capability to control the loading, or unloading, of data without resorting to programming. GeoVRML 1.0 therefore includes new nodes to manage the loading and unloading of data.

The **GeoLOD** (formerly QuadLOD) node provides the capability to browse multi-resolution, tiled terrain data that are streamed over the web. It automatically manages the progressive loading of higher-resolution data as the user approaches the terrain, and also unloads terrain data that the user has flown past. These are essential memory management and scalability operations for browsing massive terrain datasets. For example, Figure 4 illustrates a large multi-resolution dataset that is built using the GeoLOD node. We illustrate the capability to fly down through several levels of detail while higher-resolution data are streamed over the Internet to the user's display.

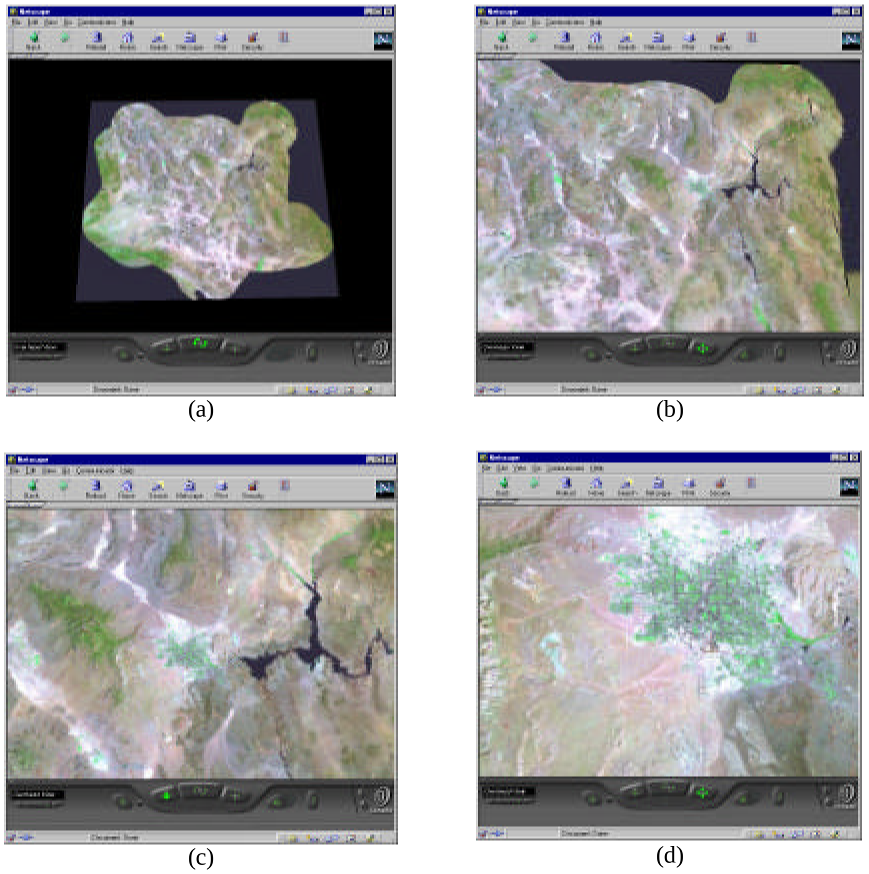


Figure 4: An example showing the scalability features provided by the **GeoLOD** node. (a) – (d) Show a user's descent into a terrain model of the Mojave desert in southern California. (a) Begins with a view of the entire 590 x 501 km region. As the user flies down, higher-resolution data are progressively loaded and displayed. In (d), the user has descended to the city of Las Vegas, Nevada, where individual streets are clearly distinguishable. The source imagery and elevation data for this terrain model total 1.3 GB.

5. Animating Geospatial Data

The discussion so far has concentrated on modeling static geometry. One of the many strengths of VRML is its ability to model dynamic systems. We therefore decided to incorporate the capability to animate models using geographic coordinates. This is implemented through the **GeoPositionInterpolator** node, which functions in much the same way as the standard VRML97 PositionInterpolator node, except that the key frame values can be specified using geographic coordinates. For example, if a GeoPositionInterpolator is created and given two latitude/ longitude coordinates, (40.669, -73.944, 10000) and (48.865, 2.35, 10000) deg, and the output is routed to a VRML model of a Boeing 777, then this aircraft would fly over the surface of the planet, from Paris to New York, at a constant altitude of 10,000 m. Using the previous example of GPS, a vehicle could be animated based upon the list of latitude/longitude coordinates from a prerecorded GPS track.

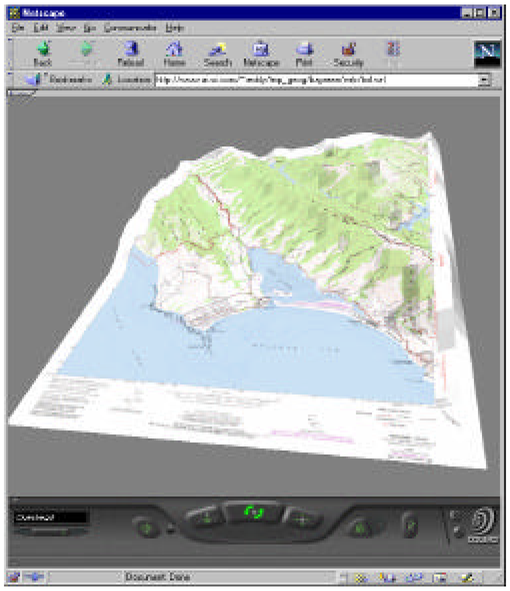
In addition to the new interpolator node, the GeoLocation node accepts eventIn values to alter the position and orientation of a model on the face of the planet. This means that we can programmatically animate a model from within a script node. For example, this could be used to control the location of a model in real-time from a live feed such as a stream of Distributed Interactive Simulation (DIS) Protocol Data Units (PDUs) [3].

6. GeoVRML Tools

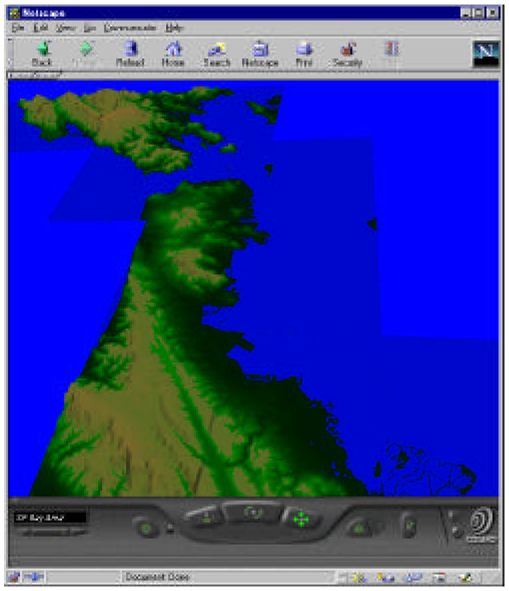
Although GeoVRML 1.0 goes a long way toward enabling geographers and cartographers to integrate their data directly into a VRML97 file, there is still a desire for tools to simplify this process. We address part of this issue by supplying a small set of tools for helping users manipulate their data. One of the more useful of these is the dem2geoeg utility. This converts a USGS 7.5-minute DEM (Digital Elevation Model) [12] to a VRML97 file with a GeoElevationGrid (see Figure 5(a)). Other DEM to VRML conversion utilities have been produced in the past to generate scenes using the built-in ElevationGrid node. The dem2geoeg utility offers a number of advantages over these programs. For example,

- The model will incorporate the correct degree of curvature for the planet.
- Multiple DEMs can be integrated together and they will be correctly co-aligned.
- The original UTM georeferencing information is preserved in the VRML file.

This second point is worth enforcing. The GeoVRML nodes internally convert the UTM coordinates into a Cartesian representation that can be thought of as a global geocentric



(a)



(b)

Figure 5: (a) A single USGS DEM converted to a GeoElevationGrid using dem2geoeg and overlaid with a USGS DRG (Digital Raster Graphics) image. (b) A composite model of the San Francisco Bay Area produced using dem2geoeg to convert and merge twelve individual USGS DEMs into a single VRML97 scene.

coordinate frame. This means that we can embed multiple objects in a scene and they will be located faithfully with respect to each other. Figure 5(b) illustrates this capability by displaying twelve DEMs that have been converted to VRML using dem2geoeg and then simply inlined into a single file. Note that the DEMs are shown correctly offset from each other in a single global context. This example illustrates the power of the GeoVRML nodes to integrate and visualize 3-D geographic data on a large scale.

7. Conclusions

We have presented the work that we have performed and contributed toward the GeoVRML 1.0 deliverable. This includes extensions to the VRML97 syntax to provide support for a wide range of geographic applications. The GeoElevationGrid node provides a height field representation for geospatial elevation data. The GeoCoordinate node allows arbitrary models to be built using geographic coordinates. The GeoLocation node lets the user georeference an arbitrary VRML model. The GeoLOD node provides scalable terrain-based level of detail management. The GeoPositionInterpolator node provides the capability to perform animations using geographic coordinates. All of these nodes can utilize the GeoOrigin node to enable the accurate rendering of double-precision geographic coordinates. In addition to these nodes, GeoVRML 1.0 includes the GeoViewpoint node to specify a camera location in geographic coordinates and the GeoMetadata node to provide a summary and links to full metadata descriptions of the geographic data. It is worth noting that these nodes address many of the issues listed by Dykes et al. as being important for supporting cartographic applications in VRML [4].

The full Java source code for these nodes, along with various examples, can be found from the GeoVRML 1.0 web page at: http://www.ai.sri.com/geovrml/1.0/. The dem2geoeg utility can be found at: http://www.ai.sri.com/~reddy/geovrml/dem2geoeg/. It is the intention of the GeoVRML Working Group to submit the GeoVRML 1.0 material to the Web3D Consortium for official approval at the end of 1999.

This material provides an important enabling technology that benefits many diverse application areas. We have illustrated this here with reference to real-world examples from the fields of scientific visualization, virtual heritage, simulation, and terrain visualization. We believe the set of capabilities reported here to be unique and novel. The ability to specify raw geographic coordinates in a 3-D file format that encompasses the ability to be browsed ubiquitously over the web is an invaluable resource, and one that will benefit earth scientists, geographers, cartographers, and many other professionals working with geographic data. In summary, the advantages and merits of GeoVRML are that it

1. Supports geographic coordinate systems such as latitude/longitude and UTM
2. Preserves the original geographic data in the VRML file
3. Integrates and co-registers multiple geographic data from different sources
4. Enables browsing over the web using only a standard VRML97 browser (with Java support)
5. Includes support for animating data using geographic coordinates
6. Deals with single-precision floating-point accuracy problems
7. Includes various data generation tools, such as the dem2geoeg utility

The GeoVRML work is of major relevance to several exciting initiatives that are currently evolving. For example, we note that this work provides the technical foundation to build the vision of a highly accurate and large-scale model of the earth into which we can embed massive quantities of georeferenced data—a vision that was offered recently as a challenge to the scientific community by Al Gore [5]. On a related topic, the OpenGIS Consortium recently announced its Web Mapping Testbed technology as a means to revolutionize the use of geospatial data on the Web. Its efforts to date have focused largely on 2-D presentations and also on the cataloging of geographic data. As such, the GeoVRML work is well positioned to contribute to this effort and bring web-based 3-D visualization capabilities to the initiative. Finally, the GeoVRML working group is tracking the evolving X3D development and hopes to provide extensions to enable the support for geographic applications in X3D.

In conclusion, we believe that the advent of GeoVRML 1.0 brings computer graphics programmers and geoscience researchers one step closer to working effectively in each other's field.

Acknowledgements

Many people have contributed to the development of GeoVRML. In particular, the authors would like to thank Farid Mamaghani and the SEDRIS project for algorithms to support geographic transformations; Dan Toms for producing a Java package based upon these algorithms; Jay Feuquay and Brian Davis of the USGS for providing terrain data; and Nat Bletter, Aaron Heller, and Doug Gordin for providing resources for the various application examples. The work performed by SRI International was funded under the DARPA-sponsored projects BADD (contract no. MDA972-97C-0037) and MAGIC II (subcontract 12165SRI under contract no. F19628-95-C-0215).

References

[1] Abernathy, M. and Shaw, S. (1998). “Integrating Geographic Information in VRML Models”. In Proceedings of the Third Symposium on the Virtual Reality Modeling Language, Monterey, CA. February 16-19, pp. 107-114.

[2] DeLeon, V. J. (1999). “VRND: Notre-Dame Cathedral: A Globally Accessible Multi-User Real-Time Virtual Reconstruction”. In Proceedings of the 5th International Conference on Virtual Systems and Multimedia (VSMM99), Dundee, Scotland. September 1-3, pp. 484-491.

[3] DIS-Java-VRML Working Group web page, http://www.web3d.org/WorkingGroups/vrtp/dis-vrml-java/

[4] Dykes, J. A., Moore, K. M. and Fairbairn, D. (1999). “From Chernoff to Imhof and Beyond: VRML & Cartography”. In Proceedings of the Fourth Symposium on the Virtual Reality Modeling Language, Paderborn, Germany. pp. 99-104.

[5] Gore, A. (1998). “The Digital Earth: Understanding Our Planet in the 21st Century”. Speech delivered at the California Science Center (CSC), Los Angeles, CA. 31 January 1998.

[6] ISO/IEC 14772-1:1997 (1997). “The Virtual Reality Modeling Language”. December 1997. http://www.vrml.org/Specifications.

[7] IEEE 754-1985 (1985). “IEEE Standard for Binary Floating-Point Arithmetic”. IEEE Standards Publication.

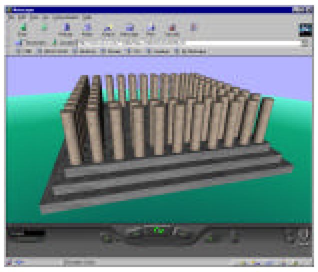
[8] Reddy, M., Leclerc, Y. G., Iverson, L. and Bletter, N. (1999). “TerraVision II: Visualizing Massive Terrain Databases using VRML”. IEEE Computer Graphics and Applications, 19(2): 30-38.

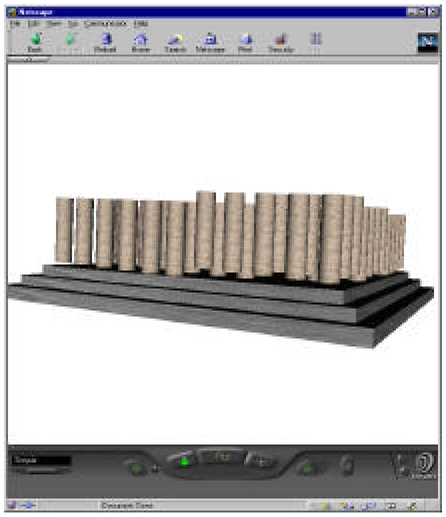
[9] Reddy, M., Leclerc, Y. G., Iverson, L., Bletter, N. and Vidimce, K. (1999). “Modeling the Digital Earth in VRML”. In Proceedings of SPIE - The International Society for Optical Engineering, vol. 3905.

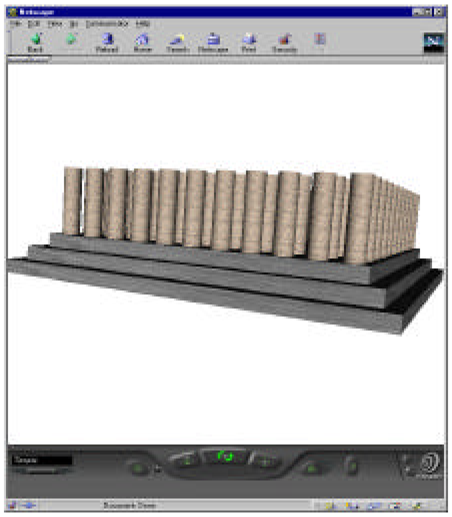
[10] Rhyne, T. M. (1997). “Going Virtual with Geographic Information and Scientific Visualization”. Computers and Geosciences, 23(4): 489-491.

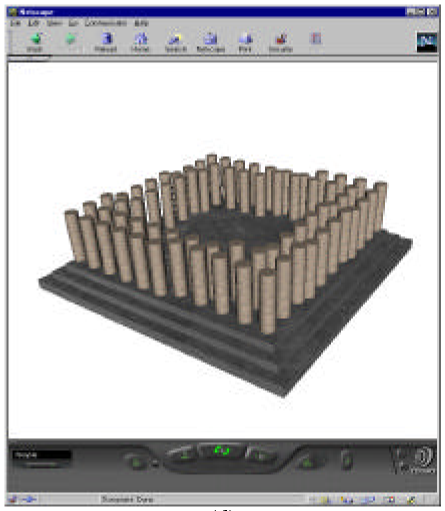
[11] Snyder, J. P. (1987). “Map Projections – A Working Manual”. U.S. Geological Survey Professional Paper 1395, U.S. Government Printing Office, Washington, DC.

[12] USGS (1990). “Digital Elevation Models”. Data Users Guide 5. United States Department of Interior U.S. Geological Survey. Reston, VA.


(a)


(b)


(c)


(d)

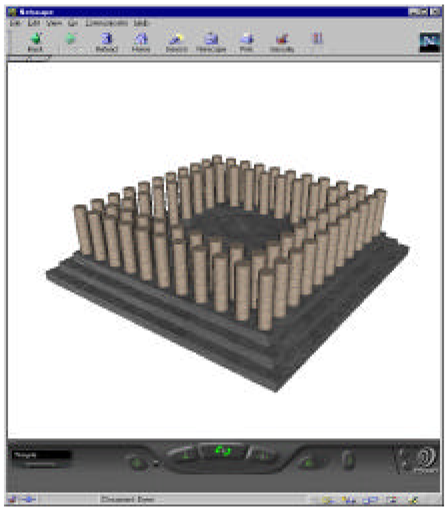

(e)

Figure 6. An illustration of the rounding errors that can occur in geographic models due to single-precision inaccuracies. (a) Shows a model of a virtual temple structure. In (b)-(e), each column and slab has been georeferenced to meter accuracy using the UTM projection. In (b) and (d) we have not used a GeoOrigin node so the absolute double-precision coordinates have just been rounded to give single-precision. The result is a number of visual artifacts where objects are not correctly co-positioned and camera jitter is experienced during movements. In (c) and (e), the same coordinates are specified, except that this time we use a GeoOrigin node to establish a local single-precision coordinate system around the center of the model. All objects are now faithfully located and no camera jitter is evident during navigation.