Geographic Representation in VRML: GeoVRML 1.1

Geospatial Web-based 3D Visualization for Soil-Landscape modeling

Mike McCann (and Martin Reddy)

Monterey Bay Aquarium Research Institute

3D Soils Unplugged - Monday 11 November, 2002

ASA-CSSA-SSSA November 2002

Indianapolis

Overview of GeoVRML 1.1

Content

- Background and goals of GeoVRML
- Issues addressed by GeoVRML
- Features and nodeset discussion
- Example GeoVRML 1.1 content
- Future directions

GeoVRML Background

GeoVRML is a Web3D Working Group

- *Web page*: http://www.geovrml.org/
- *Mailing list*: geovrml@geovrml.org (~ 200 members)
- *Mission*: develop methods and tools for representing geographic data in VRML
- *Started*: Feb 27, 1998

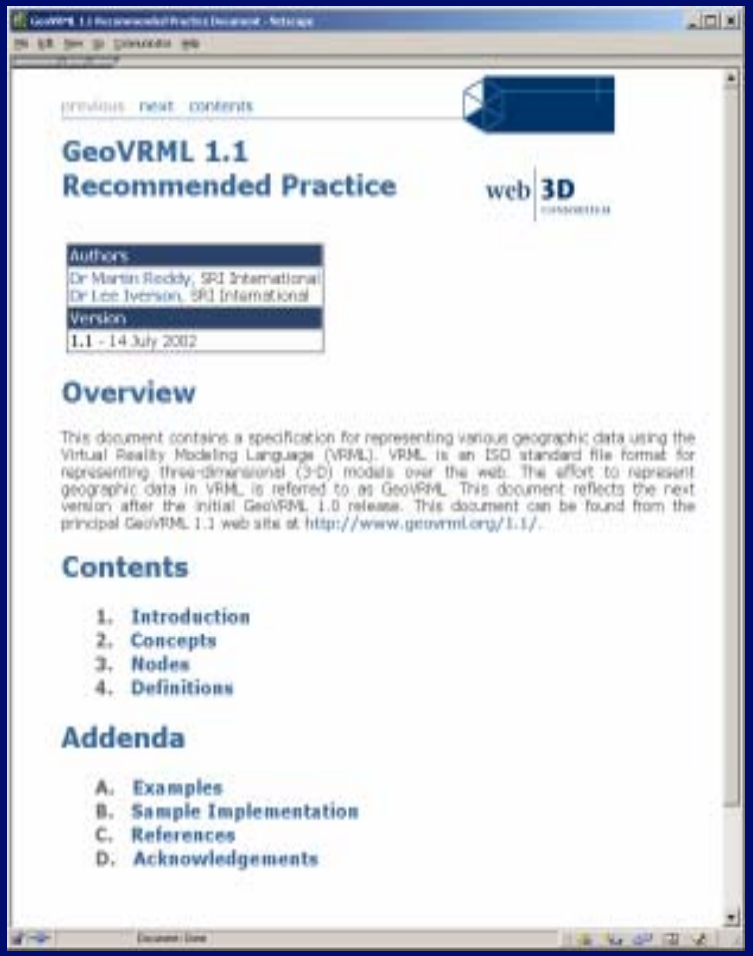
GeoVRML Goals

- Model 3-D geographic phenomena
- Support geographic coordinate systems
- Develop Open Standards
- Distribute models over the web
- Visualize and interact with 3-D models
- Disseminate data to low-end platforms

GeoVRML 1.1 Overview

GeoVRML 1.1 consists of:

- Recommended Practice document detailing extensions
- Open Source Java sample implementation
- GeoVRML 1.1 tools
- Sample Content



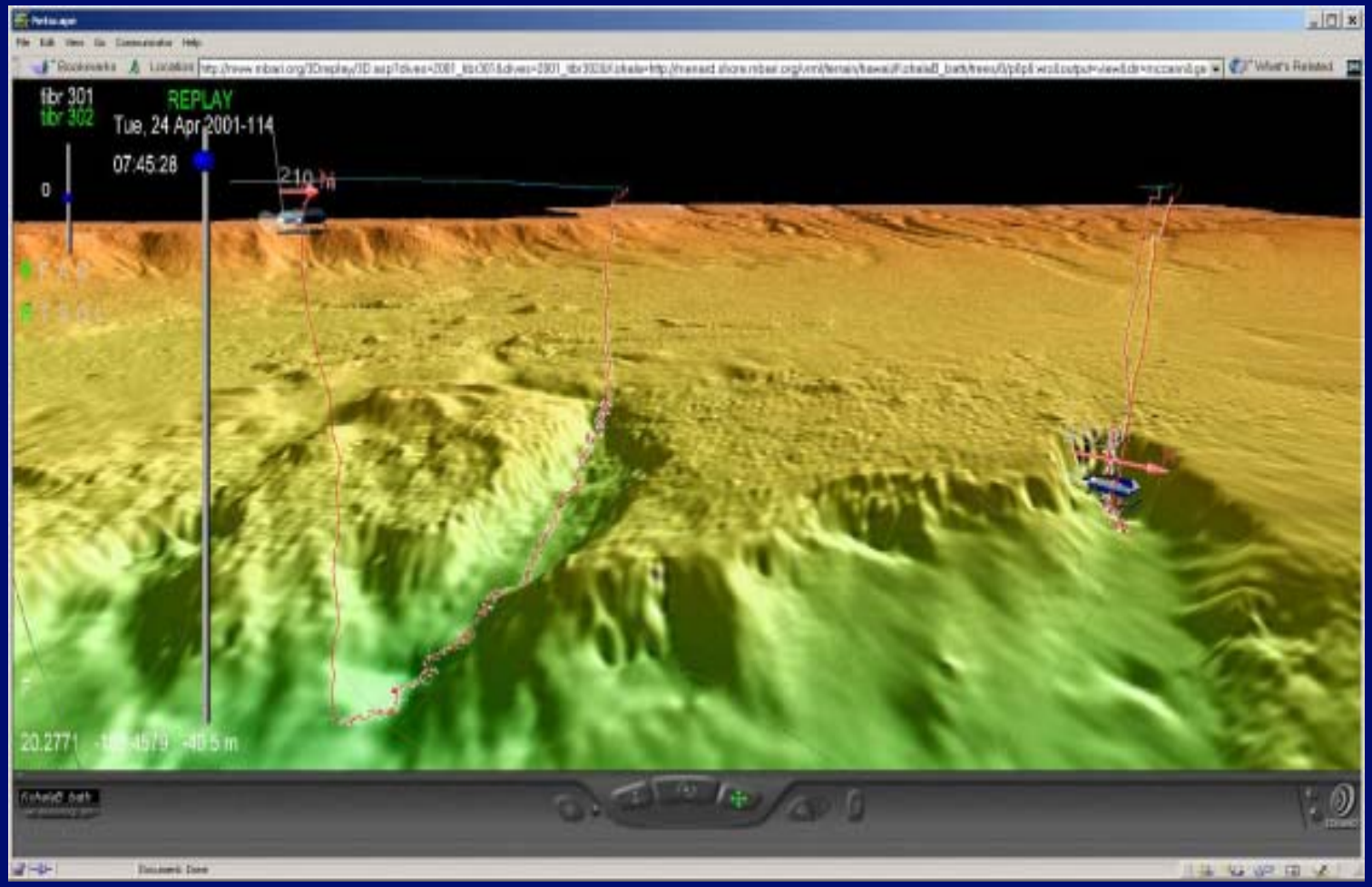
GeoVRML Features

New VRML97 nodes that:

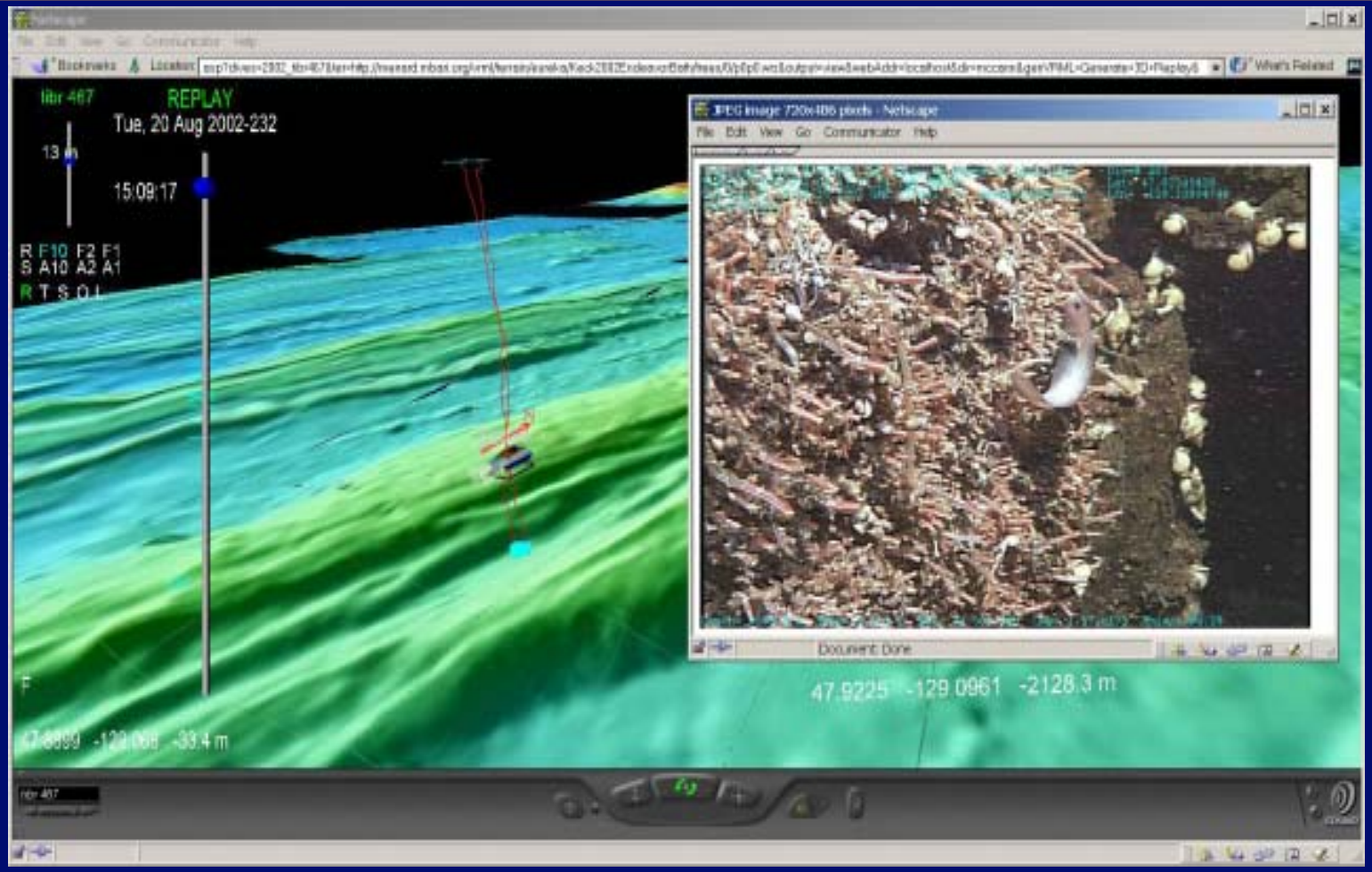
- Provide support for geographic coordinate systems, e.g. latitude/longitude and Universal Transverse Mercator (UTM)
- Integrate multiple datasets from different sources and coordinate systems into a single global context
- Provide extended precision beyond VRML97's limitation to single-precision coordinates
- Support for streaming higher levels of detail for terrain to allow browsing of massive datasets
- Perform geographic-based animations and dynamic content

GeoVRML Screenshots

GeoVRML Screenshot



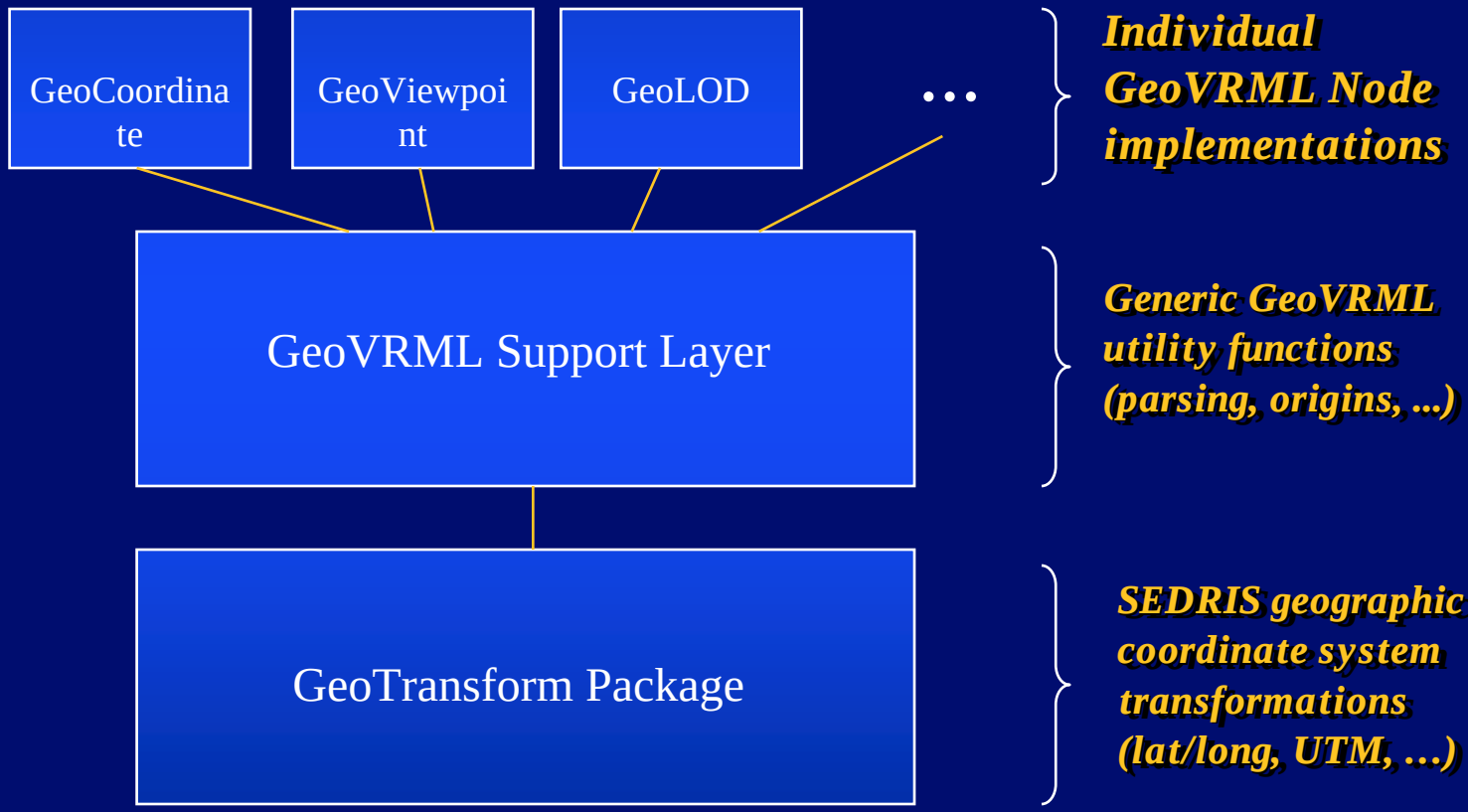
GeoVRML Screenshot



GeoVRML 1.1 Nodes

- GeoCoordinate - build geometry using geographic coords
- GeoElevationGrid - define height field in geographic coords
- GeoLocation - georeference a vanilla VRML model
- GeoLOD - multi-resolution terrain level of detail
- GeoMetadata - information about the geographic data
- GeoOrigin - used to increase precision of coordinates
- GeoPositionInterpolator - animate objects in geographic coord. Systems
- GeoTouchSensor - return the geographic coords of object
- GeoViewpoint - specify viewpoint in geographic coords
- InlineLoadControl - inline with control of loading/unloading

GeoVRML Architecture



GeoVRML Coordinate Systems

Coordinate systems supported:

- Geodetic (latitude/longitude)
- Universal Transverse Mercator (UTM)
- Geocentric (offset from planet center)

Ellipsoids supported:

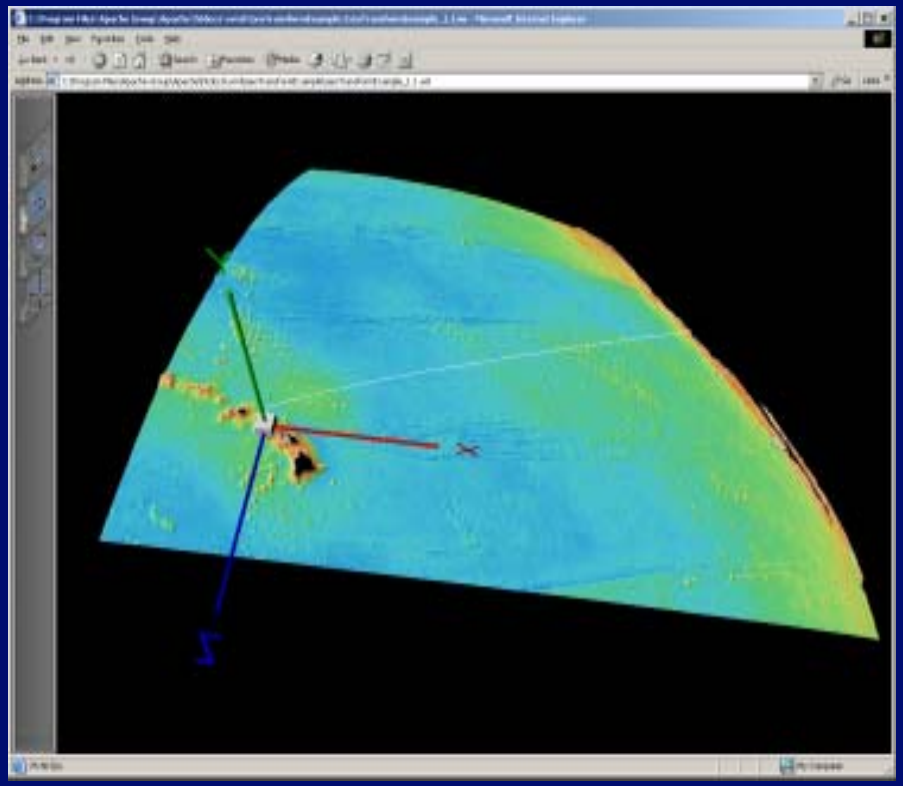
- WGS 84
- WGS 72
- + 10 others

241 Datum codes added

Coordinate systems

Projections:

- Not an issue!
- Geographic coordinates are rendered on ellipsoid in 3D
- Represented on computer, not paper – does not need to be flat
- Local VRML coordinate system with Y up available for object placement, *i.e.* with GeoLocation



GeoVRML 1.1 Coordinate Systems

```
# lat 57.7 deg, long -3.1 deg, 0 m elev, WGS84
GeoCoordinate {
  geoSystem [ “GD”, “WE” ]
  point [ “57.7 -3.1 0” ]
}

# UTM zone 11, 4361550.1 n, 310385.2 e, 1000 m elev
GeoCoordinate {
  geoSystem [ “UTM”, “Z11” ]
  point [ “4361550.1 310385.2 1000” ]
}
```

(All coordinates translated internally to geocentric)

GeoVRML Precision

The precision problem

- VRML97 supports only single-precision floats (SF / MFFloat)
- IEEE 754 single-precision: 32-bit value with 23-bit mantissa
- This provides around 6 to 7 digits of floating point precision
- Not enough for geographic coordinates to sub-meter
- Example cm-resolution geocentric coordinate:
  - *double precision: (3477218.18, -182233.28, 5325900.72)*
  - *rounded to single: (3477218.25, -182233.28125, 5325900.5)*
  - *displacement error = 23 cm.*

GeoVRML Precision

Single-precision rounding artifacts

GeoVRML Precision

Solution to extend VRML's precision

- Use local coordinate systems, e.g.
  - *Define (DP) origin at (3477210.00, -182230.00, 5325900.00)*
  - *Specify (DP) point of (3477218.18, -182233.28, 5325900.72)*
  - *Take (SP) difference: (8.18, 3.28, 0.72)*
- Use strings to define double-precision values
- Only single-precision coordinates used for rendering

GeoVRML Precision

```
# lat 57.7 deg, long -3.1 deg, 0 m elev, WGS84
GeoCoordinate {
  geoOrigin GeoOrigin {
    geoSystem [ “GD”, “WE” ]
    geoCoords “57.0 -3.0 0”
  }
  geoSystem [ “GD”, “WE” ]
  point [ “57.7 -3.1 0” ]
}
```

(GeoVRML 1.1 supports a single GeoOrigin per scene)

The 10 GeoVRML 1.1 nodes in detail

Extends VRML97

- Uses EXTERNPROTO mechanism
- Coordinate transformation runs in client Java code
- After loading, performance is same as standard VRML
- Windows installer available
- ParallelGraphics Cortona has native implementation of GeoVRML

GeoCoordinate (1 of 10)

Purpose:

- Specify a list of geographic coordinates

Usage:

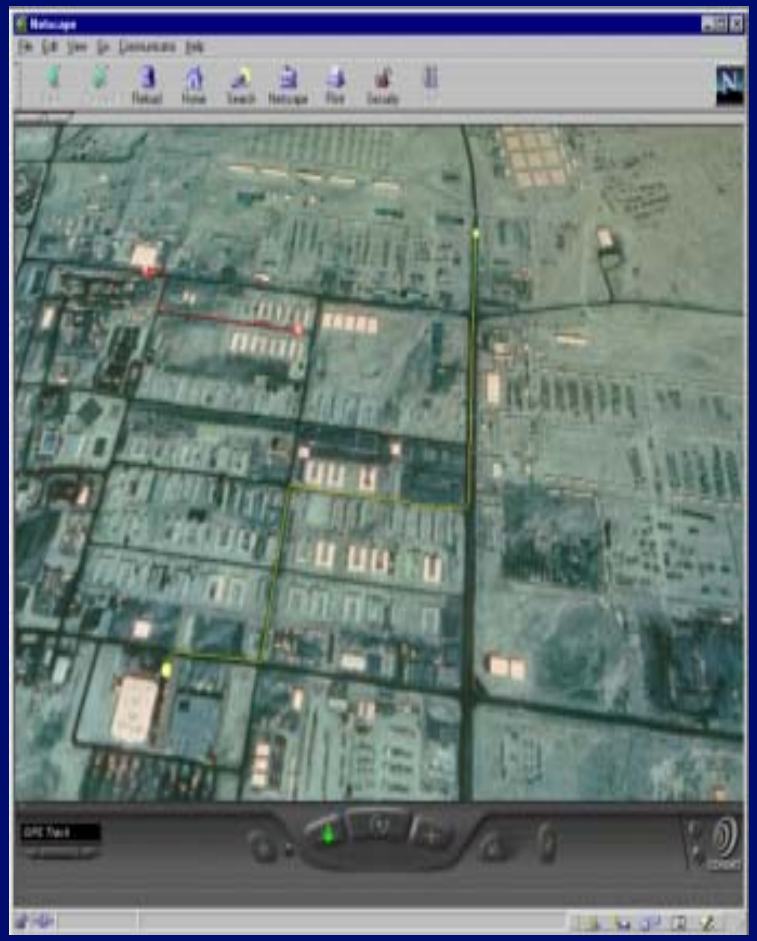
- Can use a GeoCoordinate node anywhere a VRML97 Coordinate node can go, e.g. PointSet, IndexedFaceSet, or IndexedLineSet.

Uses:

- Build models in terms of lat/long or UTM. For example, a road line segment, a GPS track, or 3-D model from GPS-recorded points or model simulation

GeoCoordinate (1 of 10)

```
Shape {
  geometry IndexedLineSet {
    coord GeoCoordinate {
      geoSystem “GD”
      point [
        “35.2500 -116.6877 310”
        “35.2500 -116.6854 312”
        “35.2491 -116.6855 312”
        “35.2485 -116.6778 311”
      ]
    }
    coordIndex [ 0 1 2 3 -1 ]
  }
}
```



GeoElevationGrid (2 of 10)

Purpose:

- Define a height field using geographic coordinates

Usage:

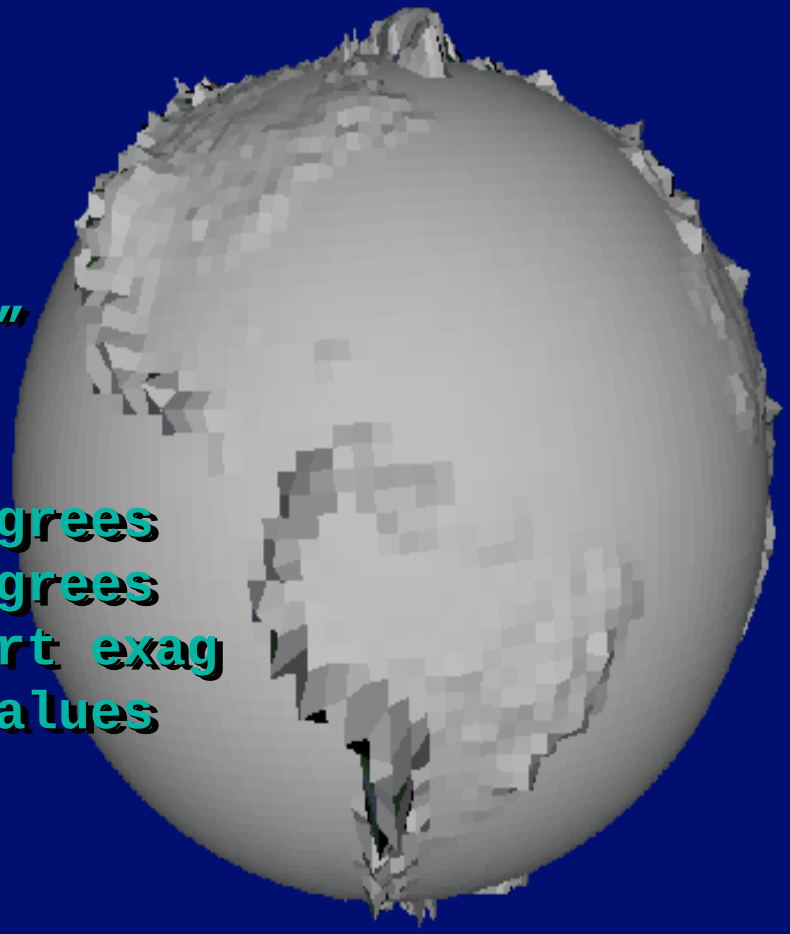
- Can use a GeoElevationGrid node anywhere a VRML97 ElevationGrid can go, e.g. from the geometry field of a Shape node.

Uses:

- Create terrain models for local or large areas (automatically introduces correct degree of earth curvature)

GeoElevationGrid (2 of 10)

```
Shape {
  geometry GeoElevationGrid {
    geoSystem      "GD"
    geoGridOrigin "-90 -180 0"
    xDimension     84
    zDimension     42
    xSpacing       "4.34" # degrees
    zSpacing       "4.34" # degrees
    yScale         200    # vert exag
    height         [ # 84x42 values
                   ]
  }
}
```



GeoLocation (3 of 10)

Purpose:

- Georeference a vanilla VRML97 model onto the surface of the earth

Usage:

- The GeoLocation node is a grouping node that affects the location of its children. It also sets the orientation so that +Y is up for that location.

Uses:

- Place a non-georeferenced model at its correct location and orientation, place a VRML97 Viewpoint or ElevationGrid at a geographic location.

GeoLocation (3 of 10)

```
GeoLocation {
  geoSystem [ “GD”, “WE” ]
  geoCoords “37.45855 -122.172997 6.5”
  children [
     Inline { url “building1.wrl” }
  ]
}
```

GeoLOD (4 of 10)

Purpose:

- Level of detail management for multi-resolution terrains

Usage:

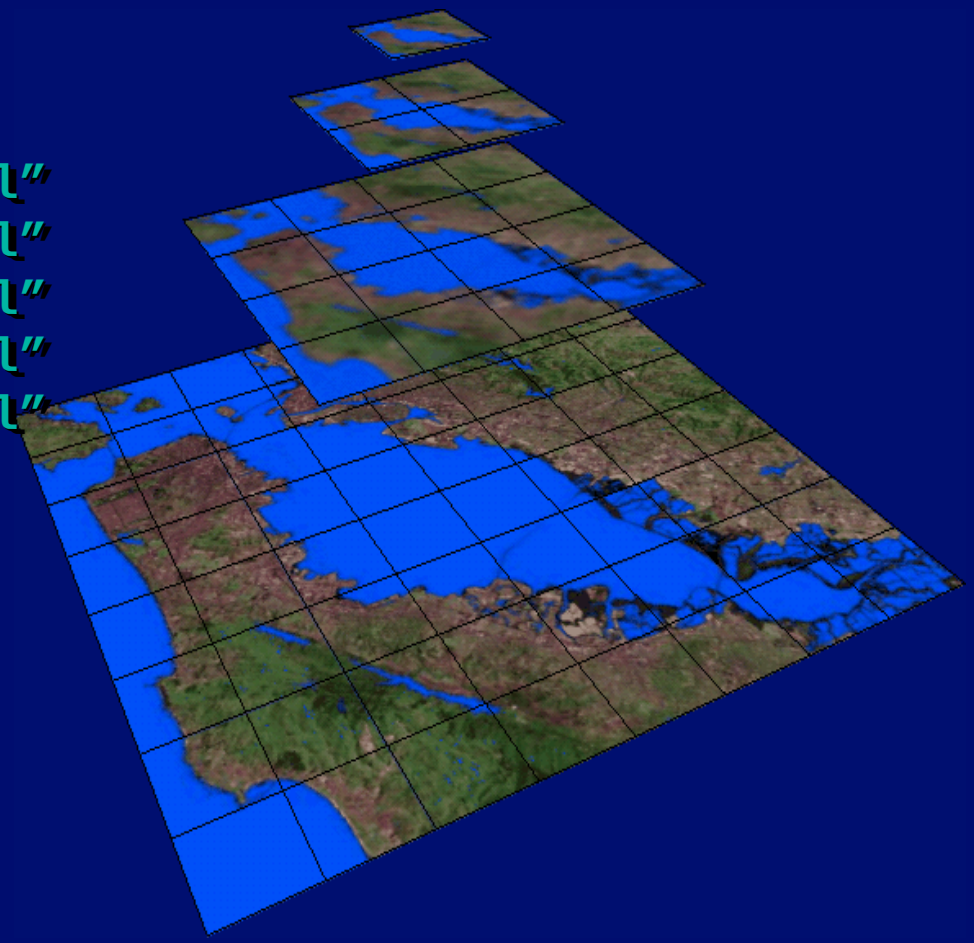
- The GeoLOD node is a grouping node that switches between two resolution levels of a quad-tree based upon distance from a point.

Uses:

- Build massive tiled, multi-resolution terrain models where the browser progressively loads higher resolution detail as you fly into the terrain.

GeoLOD (4 of 10)

```
GeoLOD {
  rootUrl    “tiles/0/p0p0.wrl”
  child1Url “trees/1/p0p0.wrl”
  child1Url “trees/1/p1p0.wrl”
  child1Url “trees/1/p1p1.wrl”
  child1Url “trees/1/p0p1.wrl”
  geoSystem “GCC”
  center     “0 0 0”
  range      7.0
}
```



GeoMetadata (5 of 10)

Purpose:

- Include a generic subset of metadata about the geographic data

Usage:

- Can be thought of as a WorldInfo node, but specifically designed for describing geographic information.

Uses:

- Provide a subset of metadata information about one or more geographic elements in a scene, and provide links to full metadata and source files.

GeoMetadata (5 of 10)

```
GeoMetadata {
  summary [
    “title”, “SAN FRANCISCO NORTH, CA”
    “description”, “DEM GENERATED FROM 1/24,000 DLG-SOURCE”
    “coordinate-system”, “UTM Z10”
    “extent”, “555060.99 4177990.30 543974.53 4191924.61”
    “resolution”, “30”
    “originator”, “United States Geological Survey (USGS)”
    “data-format”, “USGS 7.5 min DEM”
  ]
  data USE GEOEG
  url “sanfranciscon.dem”
}
```

GeoOrigin (6 of 10)

Purpose:

- Specify a local coordinate system for increased floating point precision

Usage:

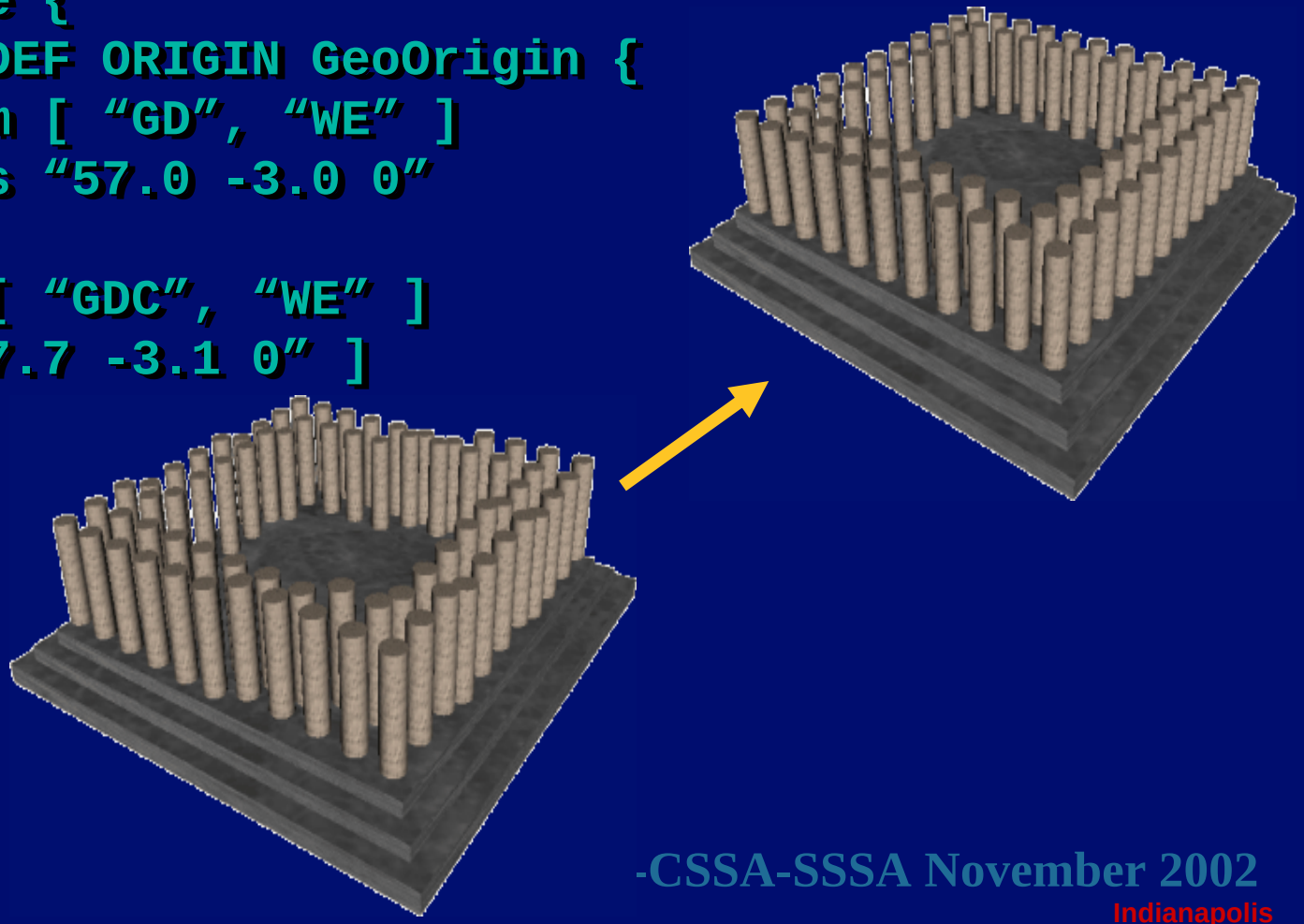
- You can use a GeoOrigin node only as the value for a geoOrigin field in another GeoVRML node. Only one GeoOrigin per scene. Use DEF / USE to provide the same GeoOrigin node to all GeoVRML nodes.

Uses:

- Remove floating point rounding artifacts for ground-level models such as quantization of vertices and camera jitter during navigation

GeoOrigin (6 of 10)

```
GeoCoordinate {
  geoOrigin DEF ORIGIN GeoOrigin {
    geoSystem [ “GD”, “WE” ]
    geoCoords “57.0 -3.0 0”
  }
  geoSystem [ “GDC”, “WE” ]
  point [ “57.7 -3.1 0” ]
}
```



GeoPositionInterpolator (7 of 10)

Purpose:

- Animate objects within a geographic coordinate system

Usage:

- Can use a GeoPositionInterpolator node anywhere that a VRML97 PositionInterpolator node can go.

Uses:

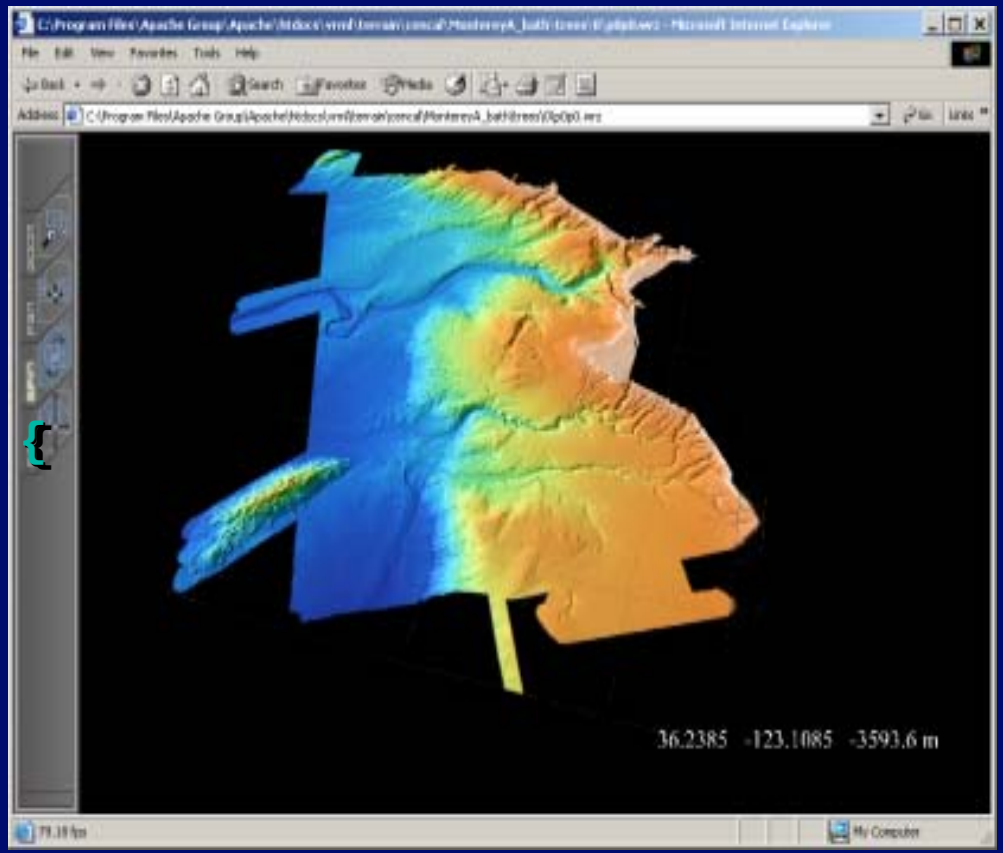
- Perform fly-throughs of GeoVRML content by animating the camera, animate objects based upon GPS data or key frame locations.

GeoPositionInterpolator (7 of 10)

```
DEF PI GeoPositionInterpolator{
  geoSystem “GDC”
  key [ 0.0, 0.1, 0.55, 1.0 ]
  keyValue [
    “51.5122 -0.065   0”  # London
    “48.865   2.35    0”  # Paris
    “40.6698 -73.9438 0”  # New York
    “51.5122 -0.065   0”  # London
  ]
}
```

GeoTouchSensor (8 of 10)

```
Group {
 children [
  DEF GTS GeoTouchSensor {
   geoOrigin USE ORIGIN
  }
  geometry GeoElevationGrid {
     . . .
  }
  ]
}
.
.
.
ROUTE GTS.hitGeoCoord_changed TO TEXTSCRIPT.set_value
```



GeoViewpoint (9 of 10)

Purpose:

- Specify a viewpoint using geographic coordinates

Usage:

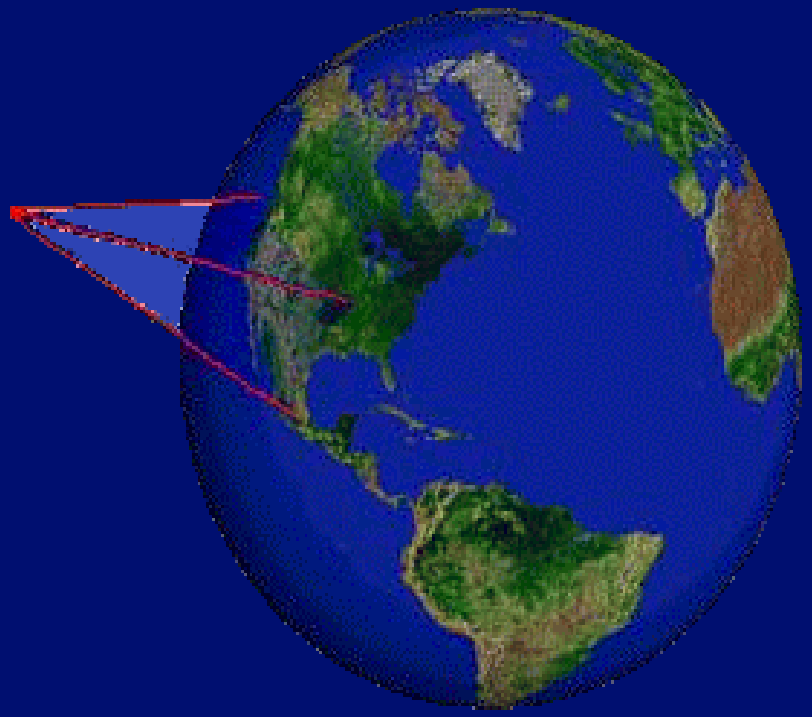
- Can use a GeoViewpoint anywhere a VRML97 Viewpoint node can go. The viewpoint orientation is relative to the up vector at that location.

Uses:

- Place the camera at a geographic coordinate, setup sensible navigation options such as height-based velocity and near/far clipping planes.

GeoViewpoint (9 of 10)

```
GeoViewpoint {
  geoSystem    “GD”
  position     “51.5 -0.1 10
  orientation 1 0 0 -1.57
  description “My GeoViewpo
  navType      “EXAMINE”
  headlight    TRUE
  jump         TRUE
}
```

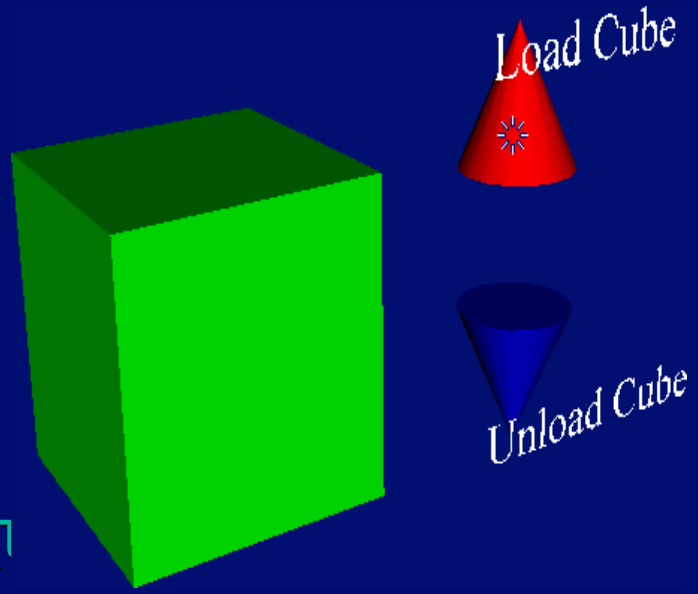


InlineLoadControl (10 of 10)

```
DEF CUBE InlineLoadControl {
  url  “cube.wrl”
  load FALSE
}

ProximitySensor {
  size 1e3 1e3 1e3
}

ROUTE PROX.isActive TO CUBE.set_l
```



How to use GeoVRML 1.1

In order to view GeoVRML worlds:

- Go to http://www.geovrml.org/1.1/download/
- Install the geovrml.jar file in your CLASSPATH (Windows installer does this for you

Platforms tested:

- Cosmo Player 2.1.1 / Netscape 4.x / Windows 95/98/NT
- Cosmo Player 2.1.1 / Netscape 4.x / IRIX 6.5.4
- Cortona 2.0 / Internet Explorer 5 / Windows 98

GeoVRML Tools

DEM to GeoVRML 1.1 translation

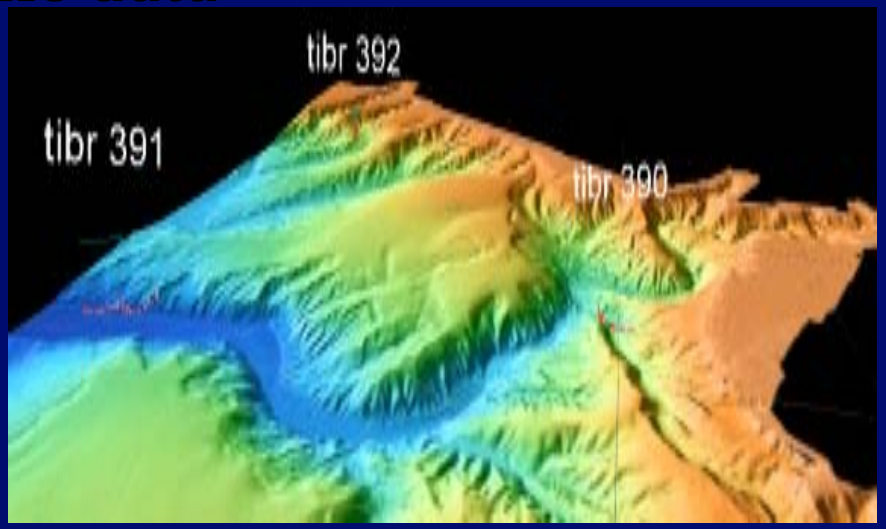
- Translates 7.5-min USGS DEMS
- Control number of polygons
- Vary vertical exaggeration
- Create grey / color image of DEM
- Merge multiple DEMs into one scene!
- Handle DEMs in feet or meters!
- Open Source + IRIX / Win32 binary

http://www.ai.sri.com/~reddy/geovrml/dem2geoeg/

GeoVRML Tools

TsmApi and MB-System

- Convert elevation data to GeoElevationGrids
- Adapted to Oceanographic data
- Builds LOD quad-tree
- Adds GeoTouchSensors



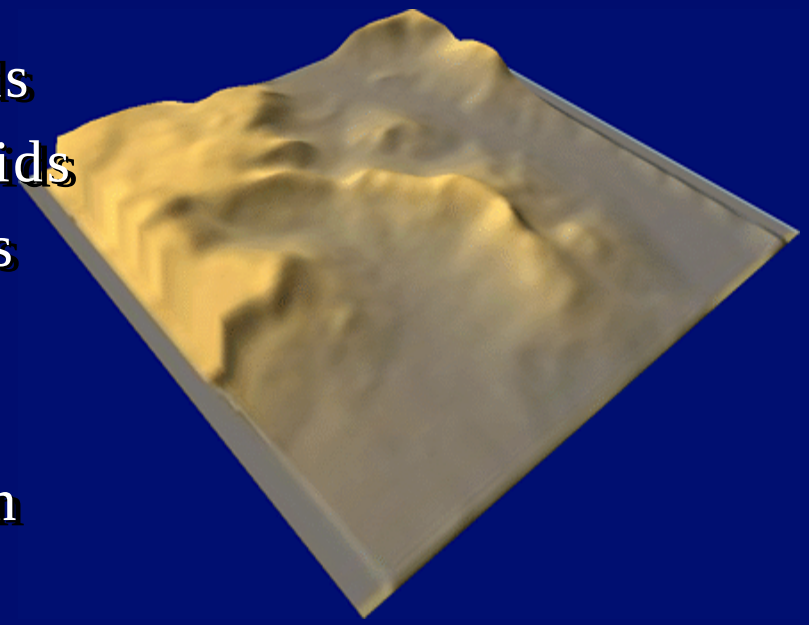
http://www.tsmapi.com

http://www.ldeo.columbia.edu/MB-System

GeoVRML Tools

Rez from Chris Thorne

- Parses (Geo)VRML elevation grids
- Creates multiresolution tree of grids
- Gzip option for reducing file sizes
- Scaling of grid size and height
- Deals with large elevation grids
- Open Source Java implementation



http://www.surak.com.au/~chris/home/vrml/elgrd/elgrdindex.html

Is That All!

What if I want more functionality?

- Join / Contribute to the geovrml@geovrml.org list
- The GeoVRML 1.1 implementation is Open Source
- Take the Java source code and add your new functionality
- Post your changes to the list

http://www.geovrml.org/1.1/source/

Future Directions for GeoVRML

Improvements:

- Support more coordinate systems
- Support geoids (elevations from mean sea level)
- Support multiple GeoOrigin nodes in a scene
- Support dynamic level of detail schemes
- Binary format for large GeoElevationGrids
- New nodes: GeoProximitySensor, GeoTransform, etc.

Expansion:

- Support X3D by testing the GeoSpatial profile
- Integrate with other efforts (OpenGIS Web Mapping Testbed?)
- More tools to translate to / from GeoVRML