Collapsible 3D GIS Visualization

Suwen Wang, Robert R. Beiko, Stephen Brooks

Dalhousie University, Halifax, Nova Scotia
Telephone: +(1) 902-494-2512
Fax: +(1) 902-492-1517
Email: sbrooks@cs.dal.ca

1. Introduction

As an essential component of GIS systems, visualization helps the user comprehend and analyze spatially referenced data. With the rise of commodity graphics processing units found in modern PCs and workstations, 3D rendering has become widespread which has allowed GIS visualization research to explore alternate 3D representations and modes of interaction. However, 3D GIS is still an open area of research and additional innovative controls are needed to better utilize the potential of 3D GIS systems. In particular, large scale information systems often suffer from the problem that available screen space is insufficient for visualizing both the focused local data set and the peripheral context information. For example, if a GIS professional was interested in analyzing two or more distant regions simultaneously in a 3D context, one would need to 'zoom-out' significantly, severely limiting the amount of detail shown in all cases. This issue of screen space is particularly acute when operating on a portable device.

In this paper, we describe the design and implementation of a 3D GIS visualization application featuring 'terrain collapsing', a user-driven control functionality in which a portion of the 3D terrain specified by the user can be dynamically shrunk to occupy less space on the terrain display, as shown in Figure 1. This feature is useful for visually comparing two terrain patches which are arbitrarily distant from each other. With the irrelevant terrain areas collapsed, more screen space is dedicated for displaying terrain regions of interest.

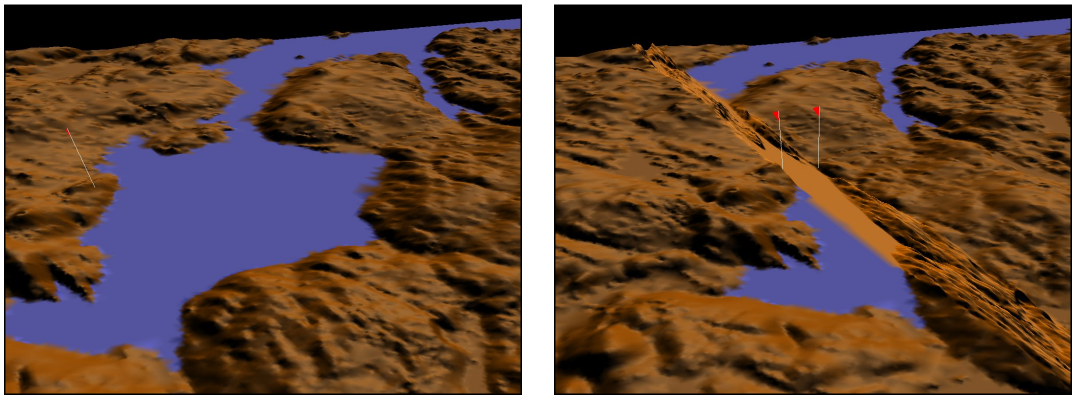

Figure 1. Left: original 3D terrain. Right: central area of the terrain is collapsed.

2. Related Work

There are two areas of research that relate to the present work: 3D GIS and methods of 2D distortion. We will briefly discuss a few key approaches from each area.

In recent years, GIS has gradually moved into the third dimension, producing commercial systems such as ArcGIS (ESRI, 2006), as well as research prototypes such as Terrafly (Rishe et al., 1999) and GeoZui3D (Ware et al., 2001). Recent work has also explored hybrid combinations of 2D and 3D GIS within the same interface (Brooks and Whalley, 2007). However, a recent review (Stota and Zlatanova, 2003) of the status of 3D GIS postulated that 3D GIS is merely at a point where 2D GIS was several years ago. Often 3D is still used for simple illustration or fly-bys and we argue that further research is needed to fully explore the possibilities and constraints of 3D GIS.

The other area of related work has developed 2D distortion techniques to overcome the problem of limited screen space within 2D interfaces. This is normally achieved by applying transformations to individual data objects and the degree of distortion is usually measured by a class of functions called magnification functions. Leung and Apperley (1994) classified these techniques into two types according to their magnification functions. The first type of distortion technique is characterized by a piecewise continuous magnification function, where there are abrupt magnification changes across the graphical representation. Classical examples of this type of technique are bifocal display (Tzavaras et al, 1982) and perspective wall technique (Mackinlay et al, 1991). The second type of distortion display employs a continuous magnification function. Fisheye projections (Furnas, 1986) belong to this category.

3. The Interactive 3D Terrain System

Terrain collapsing allows a portion of the 3D terrain, specified by the user, to be dynamically shrunk to occupy less space in the display. In order to proceed, the user must specify the region to be collapsed within the 3D interface. From the user's point of view the process is quite simple: the user specifies two points on the terrain which are to be brought together. This is sufficient information for the system to perform the collapsing operation.

3.1 Point Selection on a 3D Terrain Surface

The user must be given the ability to directly specify locations on the virtual terrain with a click of the mouse. We want the mouse picking functionality to be intuitive, so that the user can click on a point on the rendered 3D terrain surface and get exactly that point selected. This raises technical issues. In particular, when the user clicks on a point in the display window, this does not specify a single point in the 3D space because of the depth in the 3D scene. Instead, the mouse click casts an imaginary ray extending from the point on the display outwards to infinity. For our application, we must determine which point on this ray is the intersection point with the terrain surface, which is the exact location the user intended to pick. As we use OpenGL as the underlying 3D graphics library we are able to use the auxiliary function gluUnproject which performs the required reverse transformation of the projection from 3D space to 2D viewport space. Using this function we can trace a mouse click from a point on the screen to the target 3D point on the terrain surface.

Furthermore we wish to allow the user to specify *any* two points on the terrain, not just two that happen to be visible within a single screen. We therefore require additional camera controls, so that the user can move to any location to specify the first point, and then move to any other distant location to specify the second point. Using keyboard and mouse as input devices, the user can navigate the virtual camera above the terrain. As these controls are typical of any 3D GIS, we will not discuss them in detail here.

3.2 Orthogonal Terrain Collapsing

If we restrict the collapsing zones to be orthogonal to each other and parallel to the *X* and *Z* axes, then it enables us to model the terrain as a simple set of parallel quad-strips, where each strip is formed by quadrilaterals concatenated one after another. In this restricted case, the terrain is divided into three parts: the collapsing zone and the two non-collapsing zones on its two sides. The collapsing is then achieved entirely with hardware accelerated 3D transformations. First, one non-collapsing zone is rendered without alteration. Secondly, a scaling transformation is performed to shrink the size of the collapsing zone before it is rendered. Finally, the remaining non-collapsing zone is translated to abut the collapsing zone and is rendered.

We can also perform more than one orthogonal collapse simultaneously. This requires a pair of points to be selected by the user for each collapsing zone. For example, in Figure 2, there are two perpendicular collapsing zones which allow 4 distant regions to be brought together in one view.

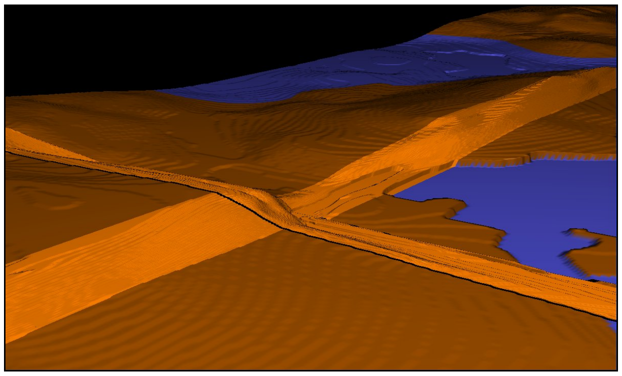

Figure 2. Multiple orthogonal collapsing zones.

3.3 Improving Efficiency with Level of Detail

The previously described method is overly restrictive and inefficient. We first improve efficiency by incorporating level of detail (LOD) techniques into the terrain model, which allows us to increase frame rates without a significant loss of visual fidelity. LOD operates by selectively and non-uniformly altering the number of polygons used at different locations of the terrain. It can therefore use a higher number of polygons only where they are needed.

We implemented a view-dependent LOD framework using a real-time adaptive quadtree data structure (Lindstrom et al, 1996). A quadtree representation of a surface is defined by recursively subdividing the surface to form a tree, in which each node represents a portion of the surface. The area covered by each node is divided into four quadrants, which form the four children of this node, and the root of the tree covers the entire surface of the terrain. The procedure for rendering the appropriate LOD is composed of two steps: updating and rendering. The algorithm recursively subdivides the mesh and visits the corresponding quadtree nodes in a depth-first order. Two types of vertex error checks are carried out at each step to determine whether further recursion is needed and whether certain vertices need to be rendered. After the updating step, all selected vertices are rendered recursively by forming triangle fan geometric primitives. The creation of the quadtree is done once offline, and the updating and rendering are executed for each frame, as the algorithm is real-time. Any crack artifacts caused by inconsistencies between different LOD levels are eliminated by reinforcing vertex dependencies. Results using LOD are shown in Figure 3.

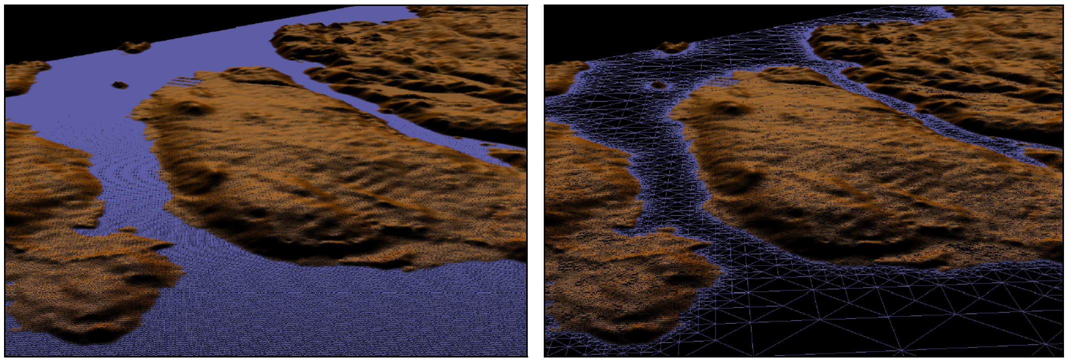

Figure 3. High-density wireframe rendering without (left) and with (right) level-of-detail.

3.3 Terrain Collapsing at Arbitrary Angles with Level of Detail

In addition to improving efficiency, it would be preferable if the collapsing region need not be orthogonal, but instead operate at any arbitrary angle. Also, when using an adaptive LOD quadtree, it is difficult to group vertices into discrete geometric objects and more importantly, the quadtree LODs are rendered by a recursive routine. We therefore cannot rely on standard OpenGL transformations to perform the collapsing as we did in section 3.2. We must instead calculate a temporary new coordinate for each vertex and render the terrain using these temporary coordinates.

Depending on the location of a vertex relative to the collapsing zone, we either scale its coordinates about the collapsing axis or translate it towards the collapsing axis. We derive the transformed temporary coordinates as follows. Given the two user-selected target points $A = (x_1, y_1)$ and $B = (x_2, y_2)$ the mid-point is calculated as:

$$M(x_m = \tfrac{x_1+x_2}{2},\, y_m = \tfrac{y_1+y_2}{2}) \tag{1}$$

For the point $P(x', y')$ we wish to transform, we find the closest point $Q(x, y)$ on the line l_n that passes through M and is perpendicular to line $\overrightarrow{AB}$, as shown in Figure 4. Let k_{AB} denote the slope of $\overrightarrow{AB}$:

$$k_{AB} = \frac{y_1 - y_2}{x_1 - x_2} = \frac{y - y'}{x - x'} \tag{2}$$

and:

$$\frac{y - y_m}{x - x_m} = -\frac{1}{k_{AB}} \tag{3}$$

We then solve:

$$k_{AB}(x - x') + y' = -\frac{1}{k_{AB}}(x - x_m) + y_m \tag{4}$$

$$\therefore \begin{cases} x = \frac{k_{AB}x' + \frac{x_m}{k_{AB}} + y_m - y'}{k_{AB} + \frac{1}{k_{AB}}} \\ y = k_{AB}(x - x') + y' \end{cases}$$

Therefore, for any point *P*, we can find its projection Q on the line perpendicular to $\overrightarrow{AB}$. Then, depending on whether point *P* is outside the collapsing zone, its new coordinates after the collapse can be derived as follows. If *P* is inside the collapsing zone,

$$\begin{cases} x_{collapse} = x + r(x' - x) \\ y_{collapse} = y + r(y' - y) \end{cases} \tag{5}$$

where *r* is the collapsing ratio. Otherwise, the point is outside the collapsing zone, and only needs to be translated to the appropriate location. From trigonometric relations:

$$\begin{aligned} \Delta x &= \cos(arctg(k_{AB}))(1 - r)d \\ \Delta y &= \sin(arctg(k_{AB}))(1 - r)d \end{aligned} \tag{6}$$

where *d* is the distance between *P* and *Q*. Therefore, the new coordinate for a translated vertex is:

$$\begin{cases} x_{collapse} = x' \pm \Delta x \\ y_{collapse} = y' \pm \Delta y \end{cases} . \tag{7}$$

The specific values for the coordinate expressions above depend on the relative position of point *P* regarding point *Q*. With these mathematical operations, each vertex in the 2D plane can be transformed around the collapsing zone specified by user input. The entire process is dynamic with the collapse occurring gradually as shown in Figure 5.

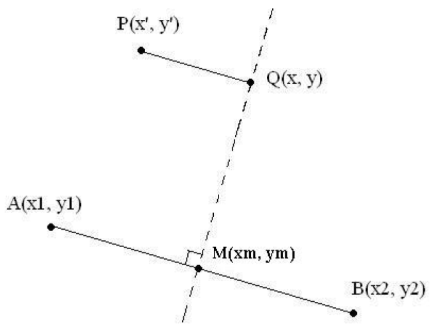

Figure 4. Calculate the collapsing axis.

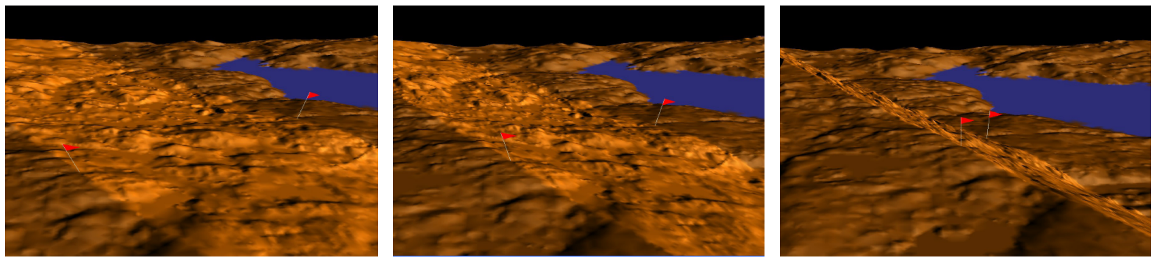

Figure 5. Dynamically collapsing at an arbitrary angle using LOD.

4. Conclusion and Future Work

In this work, we have introduced a system which allows the interactive and dynamic collapsing of unimportant regions of terrains for use within a 3D GIS. We propose that this will be useful for visually comparing two regions which are arbitrarily distant from each other, without sacrificing detailed views of two areas.

Thus far, we have implemented the basic interactive collapsing functionality and have also made progress with regards to the efficient LOD rendering of the terrains. But there are improvements and extensions which we aim to continue developing. For example, we wish to develop more tailored LOD techniques to the collapsing zone itself. Other directions to explore include multiple simultaneous collapses at arbitrary angles and the non-linear compression of areas. It would also be useful to provide additional visual information regarding the size of the collapsing zone, in terms of distances.

Another promising avenue would be the development of in-place controls over the precise positioning and orientation of existing collapsing zones. Figure 5 shows a diagrammatic sketch of what these controls might look like. It would allow the user to continuously rotate the collapsed zone or interactively add more terrain to the collapsed zone. For example, clicking-and-dragging the red arrow would rotate the line of collapse around its centre, clicking-and-dragging one of the blue arrows would draw in more terrain into the collapse on that side, and clicking-and-dragging the yellow ball in the centre of the control would slide the control itself along the line of collapse.

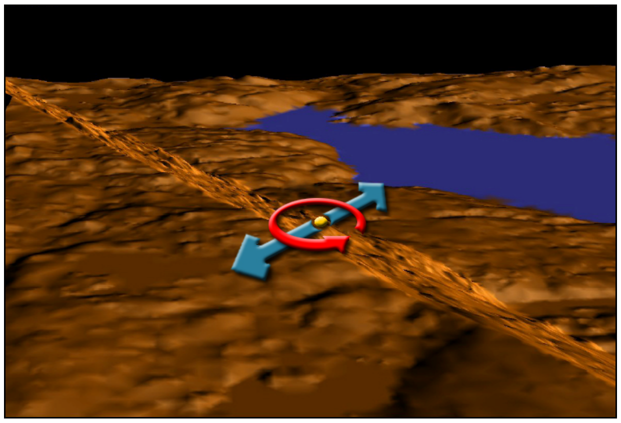

Figure 5. Diagrammatic sketch of additional interactive controls.

5. Acknowledgements

This work was supported by NSERC and the Canadian Foundation for Innovation.

6. References

Tzavaras I, Apperley M and Spence R, 1982, A bifocal display technique for data presentation. In *Eurographics '82*, 27–43.

Brooks S. and Whalley J, 2007, Towards a Comprehensive Multi-layer Hybrid Display of GIS Data. *Proceedings of the 11th Annual GISRUK Conference*, Maynooth.

ESRI *ArcGIS*. (N.D.). Retrieved April 20, 2007, http://www.esri.com/software/arcgis/.

Furnas G, 1986, Generalized fisheye views. In *Proceedings of SIGCHI '86*, 16–23.

Leung Y and Apperley M, 1994, A review and taxonomy of distortion-oriented presentation techniques. *ACM Trans. Comput.-Hum. Interact.*, 1(2):126–160.

Lindstrom P, Koller D, Ribarsky W, Hodges L, Faust N, and Turner G, 1996, Real-time, continuous level of detail rendering of height fields. *SIGGRAPH '96: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, 109–118.

Mackinlay J, Robertson G, and Card S, 1991, The perspective wall: detail and context smoothly integrated. In *Proceedings of SIGCHI*, 173–176.

Rishe N, Sun Y, Chekmasov M, Andriy S and Graham S, 2004, System architecture for 3D terrafly online GIS. In *Proceedings of Multimedia Software Engineering*, 273-276.

Stota J and Zlatanova S, 2003, 3D GIS, where are we standing? In *IPRS Joint Workshop on Spatial, Temporal and Multi- Multi-Dimensional Data Modeling and Analysis*.

Ware C, Plumlee M, Arsenault R, Mayer L and Smith S, 2001. GeoZui3D: data fusion for interpreting oceanographic data. In *Proceedings of the MTS/IEEE Conference and Exhibition*, 3, 1960-1964.