REVERSE SUBDIVISION FOR OPTIMIZING VISIBILITY TESTS

Troy F. Alderson[1] and Faramarz F. Samavati[1]
[1]*Department of Computer Science, University of Calgary, 2500 University Drive, Calgary, Canada*
{*tfalders, samavati*}*@ucalgary.ca*

Keywords: TERRAIN SIMPLIFICATION : LINE OF SIGHT : REVERSE SUBDIVISION.

Abstract: Certain applications require knowledge of whether two entities are visible to each other over a terrain, determined using a line-of-sight computation. Several fast algorithms exist for terrain line-of-sight computations. However, performing numerous line-of-sight computations, particularly over a large terrain data set, can be highly resource-intensive (in run time and/or memory). Methods from the field of terrain simplification can be used to reduce the resource impact of the visibility algorithms. To take advantage of the especially fast algorithms that exist for regular terrain models, we introduce regularity-preserving terrain simplification methods based on reverse subdivision, including a novel reverse subdivision algorithm designed to maximize visibility test accuracy, and compared the resulting visibility test output to several terrain simplification methods. Additionally, the positions of the entities after simplification can have a significant impact on the visibility test results. Hence, we have experimented with different functions that change the positions of the test points in an attempt to maximize visibility test accuracy after simplification.

1 INTRODUCTION

In many computer graphics applications, such as flight simulations and geographic information systems (GIS), a digital representation of a terrain is required. In general, digital terrain models are divided into two types: digital elevations models (DEMs), which are regular grids of elevations values; and triangulated irregular networks (TINs), which are irregular polygonal meshes composed of triangles (Duvenhage, 2009). These terrain models can become very large and detailed, especially for large areas.

However, due to their regular structure, DEMs benefit from lower memory usage than TINs (since connectivity is implicit) and faster/easier data access. These features of DEMs make for very fast line-of-sight computations (or visibility test algorithms) when compared against TINs (Seixas et al., 1999).

In several applications (such as military battlefield simulations and radio transmission tower placement), line-of-sight computation is an important operation. Terrain line-of-sight computation involves testing whether the sight line between a pair of objects (or points) intersects, and is thus obstructed by, a terrain. Computing the visibility information between a set of n points requires $O(n^2)$ visibility tests. If the individual visibility tests are resource-intensive, which can happen when the terrain is large, then the cost of this computation can drop below real time levels.

In the interest of optimizing the resource usage of visibilty test algorithms, we have considered reducing the size of the terrain using methods from the widely studied field of terrain simplification. As regular terrains feature both low memory usage and fast visibility algorithms, it is particularly desirable to preserve regularity in the simplification process.

An easy way to achieve regularity preservation in a terrain simplification scheme is to begin with a curve simplifying scheme and apply it to the rows and columns of the regular terrain. A suitable curve simplification scheme would need to ensure that curves with equal numbers of points before simplification will still have equal numbers of points after simplification, otherwise it will be impossible to connect the curve points in a regular manner. Additionally, DEM data points are often equally spaced along two dimensions, allowing the terrain to be specified almost entirely using only the data for the third dimension (i.e. the elevation values). Though not required, the curve simplification scheme should maintain or approximate this equal spacing between the data points.

A regular surface simplification scheme satisfying these requirements is reverse subdivision. Forward subdivision introduces new points into a surface in some predictable manner. Reverse subdivision, an approximate inverse of this process, simplifies a

surface in some predictable manner. We have studied several reverse subdivision schemes to achieve regularity-preserving terrain simplification, including a novel algorithm that uses least squares error minimization to preserve the spatial relationships between feature-critical points.

In general, a simplified terrain cannot perfectly approximate the original, hence changes in the terrain's shape are inevitable. Therefore, it is also important to preserve the results of the visibility test. We have compared our reverse subdivision methods against several irregular simplification methods to guage their effectiveness at preserving test accuracy.

Aside from the terrain shape, the positions of the test points also have an effect on the visibility test results. After the terrain model is simplified, the test points are usually projected onto the simplified terrain. However, while mathematically sound, transforming the point positions in this way may not be the best choice for maximizing visibility test accuracy. Hence, this work also compares transformations of the test points, or "point relocation functions", in their ability to preserve the visibility test results.

Related work is given in Section 2. In Section 3 we describe the problem examined in this work. Sections 4 and 5 describe the simplification algorithms and point relocation functions, respectively, that we have tested. Finally, our comparisons between the reverse subdivision schemes and several non-regular simplification methods may be found in Section 6.

2 RELATED WORK

There are several versions of the visibility problem: point visibility, line visibility, and region visibility. (De Floriani and Magillo, 1993) describes these visibility problems and some solutions to them. Our work focuses on point-to-point visibility, although we suspect it would prove useful in the other cases.

Several fast algorithms have been developed for point-to-point line-of-sight computations over DEMs and TINs. Bresenham's line algorithm (Bresenham, 1965), used to plot a line on a raster grid, can be adapted for DEMs to traverse the path of a sight line and compare elevation values along that path. The algorithm is linear in the number of elevation values that lie along the sight line's path.

Spatial subdivision can be used to produce an asymptotically faster algorithm. (Duvenhage, 2009) uses an implicit min/max k-d tree to quickly cull regions of the terrain that lie completely under or over the sight line to obtain an algorithm that is logarithmic in the number of elevation values that lie along the sight line's path, on average.

The implicit connectivity of regular models allows for efficient storage, addressing, and access of data values. This would suggest that visibility algorithms over DEMs should be computationally more efficient than visibility algorithms over TINs. Evidence suggests that this is so. (Seixas et al., 1999) compared the run time of the Bresenham line algorithm for DEMs against an R3-tree algorithm for TINs, and found the Bresenham algorithm to be substantially faster with a smaller memory footprint.

However, these visibility algorithms do not address the issue of the terrain size, which is a key factor in their performance and/or memory usage. In (Andrade et al., 2011), the authors work around the issue by presenting an algorithm that can efficiently perform region visibility computations on a large terrain in external memory. Terrains that can fit in internal memory remain desirable, however, as I/O operations on external memory are algorithm bottlenecks.

Terrain simplification can be used to reduce the terrain size and allow it to fit in internal memory. See (Heckbert and Garland, 1997) for a survey of simplification algorithms. For our irregular comparison methods, we have used the quadric error-metric based edge collapse scheme described in (Garland and Heckbert, 1997) and (Garland, 1999), the greedy cuts algorithm from (Silva et al., 1995) and (Silva and Mitchell, 1998), and the greedy insertion algorithm of (Garland and Heckbert, 1995). These methods are described in greater detail Section 4.

Forward subdivision, which iteratively generates fine resolution data from coarse resolution data, has gained popularity as a geometric modeling technique. Some well-known forward subdivision schemes include the corner-cutting algorithm from (Chaikin, 1974) and the interpolatory scheme of (Dyn et al., 1987). In (Prusinkiewicz et al., 2003), the authors describe two seperate but easily understood formalizations for forward subdivision: the standard matrix notation and Lindenmayer system notation.

(Samavati and Bartels, 1999) use global least squares data fitting to reverse forward subdivision rules and obtain a curve that, after an application of forward subdivision, yields an approximation of the original curve. In (Bartels and Samavati, 2000), the authors applied local least squares data fitting to generate local subdivision filters that can be used to apply reverse subdivision in linear time. Forward and reverse subdivision form the core components of multiresolution decomposition and reconstruction, which can be used to obtain a multiscale representation of the terrain. See (Samavati et al., 2007) for further details on multiresolution and some of its applications.

Reverse subdivision has been used as a regularity-preserving simplification scheme in (Losasso and Hoppe, 2004). The authors make use of forward and reverse Dyn-Levin subdivision to generate a viewer-centered hierarchy of nested regular grids, called a *geometry clipmap*, designed for use in terrain rendering with level-of-detail control. The effect of reverse Dyn-Levin-Gregory subdivision and geometry clipmaps on visibility test accuracy is presently unknown. In this work, we examine the effect of reverse subdivision upon visibility test accuracy.

The problem of applying terrain simplification to optimize visibility computations has been studied previously in (Ben-Moshe et al., 2002). In their paper, Ben-Moshe et al. present a measure of visibility similarity between a terrain and its simplifications and describe a novel method designed to maximize the visibility similarity. Their algorithm first identifies a network of features important to visibility (the "ridge network") and uses the points that constitute this network to create an initial approximation of the terrain, which is then refined by a greedy insertion algorithm. However, their algorithm does not preserve regularity.

3 PROBLEM STATEMENT

Consider a terrain model T and a set of points $P = \{p_1, p_2, \ldots, p_n\}$ on T. For every T and P define a visibility relation $V_{T,P}$ on P such that $(p_i, p_j) \in V_{T,P}$ if the line segment (or visibility ray) between p_i and p_j does not intersect T. If $(p_i, p_j) \in V_{T,P}$, we say that p_i and p_j are visible over T, otherwise we say they are not visible over T. Assume there exists an algorithm A that computes $V_{T,P}$ for any T and P.

We wish to find a terrain simplification function S and a point relocation function R such that the computation of $V_{S(T),R(P)}$ using algorithm A is faster and/or requires less memory than the computation of $V_{T,P}$ using A, and such that $|agree^+| + |agree^-|$ is maximized, where:

$$\begin{aligned} agree^+ &= \{(p_i, p_j) \in P \times P | (p_i, p_j) \in V_{T,P} \\ &\quad \text{and } (R(p_i), R(p_j)) \in V_{S(T),R(P)}\} \\ agree^- &= \{(p_i, p_j) \in P \times P | (p_i, p_j) \notin V_{T,P} \\ &\quad \text{and } (R(p_i), R(p_j)) \notin V_{S(T),R(P)}\} \end{aligned}$$

The *visibility similarity* between T and $S(T)$, or *percent accuracy* of the visibility test after application of the terrain simplification and point relocation, can be calculated as:

$$\frac{|agree^+| + |agree^-|}{|P \times P|}$$

4 TERRAIN SIMPLIFICATION

Height maps are used extensively throughout the field of GIS. The implicit connectivity of regular structures allows the storage of terrain data in array-like data structures, which are easily accessed and have low memory usage beyond that needed for the raw data.

As technology advances and terrains can be mapped at higher and higher resolutions, the memory savings of height maps are particularly attractive. It is often necessary to simplify such large data sets for application use (Ben-Moshe et al., 2002), although not all simplification algorithms preserve the desirable quality of regularity.

In this section we briefly describe the irregular simplification methods we have used for comparison, followed by a description of reverse subdivision and the variants we have used for optimizing visibility tests.

4.1 Quadric Error-based Collapse (QEC)

Iterative vertex contraction, or edge collapsing, is a mesh simplification paradigm in which edges deemed to be unimportant via some importance metric have their endpoints merged into a single point (Garland, 1999).

Edge collapse schemes differ from each other primarily in the metrics used to select edges for contraction. For our comparisons we have used a shape preserving edge collapse scheme based on Garland and Heckbert's quadric error metric (Garland and Heckbert, 1997) (Garland, 1999).

A set of planes is associated with each vertex of the terrain model, obtained by extending the faces incident to the vertex. For a given edge, the error resulting from collapsing the edge into a given point is computed as the sum of the squared distances from the collapse point to the planes associated with the edge's endpoints. The squared distance sum can be efficiently computed using a structure called a "quadric" (Garland, 1999).

4.2 Greedy Cuts Algorithm

The greedy cuts algorithm of Silva et al. (Silva et al., 1995) (Silva and Mitchell, 1998) incrementally removes regions (or, in Silva et al.'s terminology, takes "bites") of the yet-to-be-triangulated terrain using three basic operations: ear cutting, greedy biting, and edge splitting. Each bite region is approximated with a triangle, with some user-specified error tolerance ε.

4.3 Greedy Insertion Algorithm

Garland and Heckbert's greedy insertion algorithm (Garland and Heckbert, 1995) is a generalization to 3D polygonal surfaces of the Douglas-Peucker algorithm for approximating curves (Douglas and Peucker, 1973). The algorithm starts with a coarse approximation of the terrain model. Iteratively, the mesh vertex that is furthest from the approximation mesh is added to the approximation. To ensure mesh quality, a Delaunay triangulation on the mesh's 2D projection is maintained (Garland and Heckbert, 1995).

Of note is that the greedy insertion algorithm will prioritize approximating high energy areas of the terrain, as vertices in these areas will generally be furthest from the approximation. Thus, low energy areas of high energy terrains are approximated relatively poorly.

4.4 Reverse Subdivision Methods

Subdivision is a family of methods for introducing additional points into a curve or surface, with some smoothness constraint. Reverse subdivision approximates the inverse of this process; details necessary for reconstruction are usually lost in a typical reverse subdivision process. Both forward and reverse subdivision can be applied iteratively to obtain the desired number of curve/surface points.

In the case of curves, one starts with a vector of points, c, that describe the curve. The goal of forward subdivision is to use c (the "coarse points") to generate a larger vector of points, f (the "fine points"), that describes a curve with some known continuity. These fine points are expressed as affine combinations of the coarse points. Arranging the coefficients of the affine combinations in a matrix, S, the application of forward subdivision to c can be represented in matrix notation as $f = Sc$. It is typical for such subdivision matrices to be sparse and banded, with a repeating local structure. This local pattern can be exploited to apply subdivision in linear time (Prusinkiewicz et al., 2003).

Curve-based subdivision schemes (both forward and reverse) can be easily generalized to regular surfaces via the application of the curve scheme to the surface's rows and columns. Hence, we limit our discussion of subdivision to curve schemes. Regular surfaces after application of either forward or reverse subdivision are guaranteed to remain regular.

Several subdivision schemes are derived from knot insertion into a B-Spline curve (Samavati and Bartels, 1999). Faber subdivision, for instance, is derived from knot insertion into a second order B-Spline curve. Given a curve defined by a discrete set of points, the Faber scheme increases the resolution of the curve by inserting midpoints (Samavati and Bartels, 1999). Its subdivision matrix S has the form

$$S = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

Faber subdivision has the effect of introducing more points into the curve/surface whilst not having an effect on the shape. Intuitively we expect the reversal of this process to have a minimal effect on the shape of the curve/surface and therefore the visibility test results over the terrain.

There are several ways to reverse a subdivision scheme. For our tests, we have used four variants of reverse Faber subdivision.

4.4.1 Simple Reverse Faber

The first reverse Faber subdivision scheme we've examined is the simplest, and therefore we refer to it as the "simple" reverse Faber scheme. Since Faber subdivision introduces midpoints between vertices, the simple scheme assumes every other curve point is a midpoint and discards it. In other words, the simple reverse Faber scheme performs a downsampling of the curve points by a factor of 2.

4.4.2 Global Least Squares (GLS) Reverse Faber

To better approximate the original curve, the global least squares (GLS) scheme adjusts the positions of the vertices to minimize the least squares error, $||Sc - f||_2$, between the simplified and original curves. This entails solving the overdetermined linear system $Sc = f$ for c.

$$\begin{aligned} Sc &= f \\ S^T Sc &= S^T f \\ c &= (S^T S)^{-1} S^T f \end{aligned}$$

However, while the reverse subdivision matrix $(S^T S)^{-1} S^T$ gives a valid result, since S is banded it is often faster to solve for c directly from the linear system $Sc = f$.

4.4.3 Local Least Squares (LLS) Reverse Faber

The third Faber scheme used for this work minimizes the local least squares (LLS) error between the simplified and original curves (Bartels and Samavati, 2000).

The subdivision matrix S is different for different lengths of the vectors c and f, despite the uniform structure of the underlying subdivision scheme. Analysis of subdivision schemes independent of the size of c and f is facilitated by the use of a *local* subdivision matrix, L, which has the same structure as S but operates on fixed-size c and f. $R = (L^T L)^{-1} L^T$ is the local reverse subdivision matrix that minimizes the l^2-norm $||Lc - f||_2$, where $c = Rf$. Using the (precomputed) coefficients of R, the reverse subdivision operation can be done efficiently in linear time.

Consider a vector, f, of $2n$ fine points (where n is a positive integer). Let c_i and f_j (for i between 0 and $n-1$, j between 0 and $2n-1$) be the indexed points of c and f, respectively. The vector c of n coarse points resulting from local least squares on neighbourhoods of five fine points is given by

$$\begin{aligned} c_0 &= f_0 \\ c_i &= -\tfrac{1}{6} f_{2i-2} + \tfrac{1}{3} f_{2i-1} + \tfrac{2}{3} f_{2i} + \tfrac{1}{3} f_{2i+1} - \tfrac{1}{6} f_{2i+2} \\ c_{n-1} &= f_{2n-1} \end{aligned}$$

4.4.4 Feature Aware Reverse Faber

Our novel reverse subdivision algorithm attempts to preserve terrain features using global least squares error minimization. Based on the observation that peaks and valleys in a terrain are important features that affect visibility, our algorithm identifies the critical points that define these features and uses least squares error minimization to preserve the spatial relationships between them.

Identification of the "critical points" is closely related to the ridge network computation that lies at the heart of the novel simplification algorithm in (Ben-Moshe et al., 2002). For a curve, the critical points are the endpoints and local maxima and minima. Let $p_i \in f$ denote the critical points. Then, the vectors between the critical points (say $v_i = p_i - p_{i-1}$), which define the spatial relationships between them, are calculated.

To preserve these spatial relationships, we augment the linear system $Sc = f$ with additional constraints. The vectors v_i are appended to the end of f and additional rows (one for each of the v_i) are appended to the matrix S. These rows have exactly two non-zero entries, -1 and $+1$, for the points in coarse space that correspond to the critical points used to calculate the v_i. That is, for $v_i = f_j - f_k$, the coarse point $c_{\lfloor j/2 \rfloor}$ receives entry $+1$ in the matrix row and coarse point $c_{\lfloor k/2 \rfloor}$ receives entry -1, so that $v_i \approx c_{\lfloor j/2 \rfloor} - c_{\lfloor k/2 \rfloor}$.

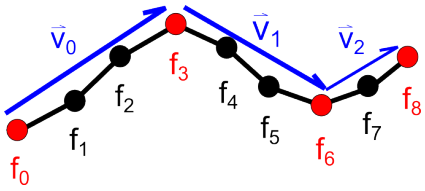

Figure 1: Identification of critical points and the vectors between them.

For example, the linear system for the curve shown in Figure 1 would be

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \\ v_0 \\ v_1 \\ v_2 \end{bmatrix}$$

By solving the augmented linear system, a coarse point vector c can be obtained that minimizes the error between the original and simplified curves and preserves the spatial relationships between the critical points.

5 POINT RELOCATION FUNCTIONS

When the terrain is simplified, the space the points occupy is transformed. Hence, it is important to consider how the points should be relocated into the transformed space to best preserve the visibility results.

We have considered three point relocation functions to transform the points from the original space to the simplified space. These are described in the following subsections.

5.1 Identity

The simplest relocation function is the identity function, which does not move the points. Supposing that the simplified terrain is a good approximation of the

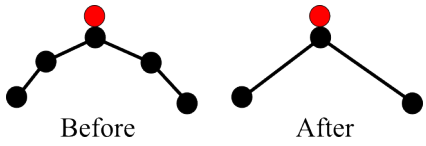

Figure 2: Illustration of the identity function.

original (i.e. is shape-preserving) then it should be unnecessary to move the points at all.

5.2 Full Projection

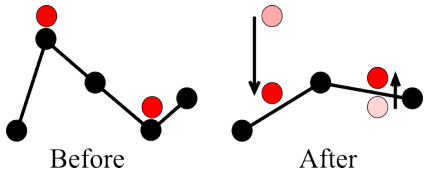

Figure 3: Illustration of the full projection function.

Our second relocation function is the full projection function. Given that the test points are situated atop the original terrain, we suspect that reprojecting all sample points vertically onto the simplified terrain should produce similar visibility test results.

However, the full projection function suffers from an important flaw. Consider a sharp terrain feature simplified to a flat terrain feature, as in Figure 3. A point situated atop the sharp feature would be able to view terrain in lower regions. Once the point is projected onto the flat feature after simplification, however, its view of the lower regions will be occluded. Hence, the visibility test accuracy may be diminished.

5.3 Half Projection

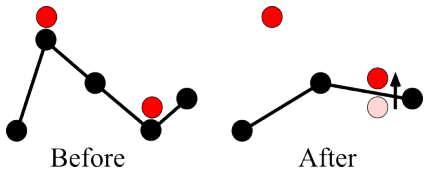

Figure 4: Illustration of the half projection function.

The half projection function is a hybrid between full projection and identity that attempts to solve the problem of sharp features simplified to flat features occluding a point's view. The function projects onto the simplified terrain only those sample points that lie within the half-space underneath the simplified terrain, and leaves points above the terrain untouched.

6 COMPARISONS

In this section we describe and discuss our results from comparing the visibility test accuracy of the various simplification schemes described throughout the paper.

Visibility tests were conducted using a ray casting approach on five different terrain models with varying levels of sharp features, each tested using six random distributions of fifty test points. The terrain models (120×120 height maps) were each tested at three levels of simplification: 25% of the original terrain size, 6% of the original terrain size, and 1.5% of the terrain size (which correspond to the terrain size after 1, 2, and 3 iterative applications of reverse subdivision, respectively).

6.1 Numerical Results

See Table 1 for the accuracy results after simplification to 25% of the original terrain size, Table 2 for the results after simplification to 6%, and Table 3 for the results after simplification to 1.5%. For each simplification method-relocation function pair, the average accuracy rate, μ, and its standard deviation, σ, are given.

6.2 Discussion

Our results indicate that, while the reverse subdivision methods have a more restrictive nature than the irregular simplification methods, the restriction to regular surfaces does not significantly diminish the visibility test accuracy. The reverse subdivision methods appear to be only marginally poorer at preserving visibility test accuracy than the other methods, with fairly similar standard deviations.

Of these, the simple reverse Faber scheme is undoubtedly the weakest. The other three variants (GLS, LLS, Feature Aware) have approximately equal rates of total accuracy, however, due to its superior run time the LLS reverse Faber scheme emerges as the preferred reverse subdivision scheme. Although our novel algorithm did not show any improvement in average accuracy over the other subdivision methods, of these three variants it tends to have the lowest deviation.

While the half projection function shows promise at low levels of simplification, as the degree of simplification increases its performance at preserving accuracy deteriotes quite substantially. The results for half projection at 1.5% of the original terrain size were the worst of the relocation functions.

Interestingly, the identity relocation function appears to have the most positive effect on visibility test accuracy, having the highest average accuracies and lowest deviations. We suspect this is because test points which fall under the terrain after simplification are not visible to most entities, and that projecting these points onto the terrain where they may be visible is detrimental to the overall accuracy. A comparison of the rates of true positives and negatives between the identity and half projection functions appears to support this assertion (see Table 4). We suspect that the results would look different were the visibility tests more localized.

7 CONCLUSIONS

After comparing several terrain simplification methods, we have identified local least squares Faber reverse subdivision as a fast regularity-preserving simplification scheme that preserves visibility test accuracy well. Additionally, having tested various point relocation functions, we have identified the identity and full projection functions as good relocation function for preserving overall test accuracy. However, more analysis is needed for the identity function using localized visibility tests.

ACKNOWLEDGEMENTS

The C implementation of Cláudio Silva's greedy cuts algorithm, GcTin, was downloaded from his user page at www.vistrails.org. The terrain models used for our comparisons were obtained from the GcTin package.

Thanks go to C4i Consultants Inc. for motivating the problem and assisting in research, and to Mitacs Accelerate for sponsoring our collaboration.

Lastly, thanks go to NSERC for funds used towards travel expenses.

REFERENCES

Andrade, M. V. A., Magalhães, S. V. G., Magalhães, M. A., Franklin, W. R., and Cutler, B. M. (2011). Efficient viewshed computation on terrain in external memory. *Geoinformatica*, 15(2):381–397.

Bartels, R. H. and Samavati, F. F. (2000). Reversing subdivision rules: Local linear conditions and observations on inner products. *Journal of Computational and Applied Mathematics*, 119(1-2):29–67.

Ben-Moshe, B., Mitchell, J. S. B., Katz, M. J., and Nir, Y. (2002). Visibility preserving terrain simplification: An experimental study. In *Proceedings of the 18th Annual Symposium on Computational Geometry*, SCG '02. ACM, New York, NY, USA.

Bresenham, J. E. (1965). Algorithm for computer control of a digital plotter. *IBM Systems Journal*, 4(1):25–30.

Chaikin, G. M. (1974). An algorithm for high-speed curve generation. *Computer Graphics and Image Processing*, 3(4):346–349.

De Floriani, L. and Magillo, P. (1993). Algorithms for visibility computation on digital terrain models. In *Proceedings of the 1993 ACM/SIGAPP Symposium on Applied Computing: States of the Art and Practice*, SAC '93. ACM, New York, NY, USA.

Douglas, D. H. and Peucker, T. K. (1973). Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, 10(2):112–122.

Duvenhage, B. (2009). Using an implicit min/max kd-tree for doing efficient terrain line of sight calculations. In *Proceedings of the 6th International Conference on Computer Graphics, Virtual Reality, Visualisation and Interaction in Africa*, AFRIGRAPH '09. ACM, New York, NY, USA.

Dyn, N., Levin, D., and Gregory, J. A. (1987). A 4-point interpolatory subdivision scheme for curve design. *Computer Aided Geometric Design*, 4(4):257–268.

Garland, M. (1999). *Quadric-Based Polygonal Surface Approximation*. PhD thesis, Carnegie Mellon University.

Garland, M. and Heckbert, P. S. (1995). Fast polygonal approximation of terrains and height fields. Technical Report CMU-CS-95-181, Carnegie Mellon University.

Garland, M. and Heckbert, P. S. (1997). Surface simplification using quadric error metrics. In *Proceedings of SIGGRAPH 1997*.

Heckbert, P. S. and Garland, M. (1997). Survey of polygonal surface simplification algorithms.

Losasso, F. and Hoppe, H. (2004). Geometry clipmaps: Terrain rendering using nested regular grids. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04. ACM, New York, NY, USA.

Prusinkiewicz, P., Samavati, F. F., Smith, C., and Karwowski, R. (2003). L-system description of subdivision curves. *International Journal of Shape Modeling*, 9(1):41–59.

Samavati, F. F. and Bartels, R. H. (1999). Multiresolution curve and surface representation: Reversing subdivision rules by least-squares data fitting. *Computer Graphics Forum*, 18(2):97–120.

Samavati, F. F., Bartels, R. H., and Olsen, L. (2007). Local b-spline multiresolution with examples in iris synthesis and volumetric rendering. In *Image Pattern Recognition: Synthesis and Analysis in Biometrics*, volume 67 of *Series in Machine Perception and Artificial Intelligence*, pages 65–102. World Scientific Publishing.

Seixas, R. d. B., Mediano, M. R., and Gattass, M. (1999). Efficient line-of-sight algorithms for real terrain data.

In *Simpósio de Pesquisa Operacional y Logística da Marinha*, SPOLM '99.

Silva, C. T. and Mitchell, J. S. B. (1998). Greedy cuts: An advancing front terrain triangulation algorithm. In *Proceedings of GIS 1998*.

Silva, C. T., Mitchell, J. S. B., and Kaufman, A. E. (1995). Automatic generation of triangular irregular networks using greedy cuts. In *Proceedings of the IEEE Conference on Visualization 1995*, pages 201–208, 453.

Table 1: Comparison results for 25% of original terrain size.

Simplification Method	Accuracy Rates		
	Identity	Full Projection	Half Projection
QEC	$\mu = 98.2\%, \sigma = 1.2\%$	$\mu = 97.2\%, \sigma = 1.6\%$	$\mu = 98.3\%, \sigma = 0.9\%$
Greedy Cuts	$\mu = 97.5\%, \sigma = 1.5\%$	$\mu = 96.7\%, \sigma = 1.6\%$	$\mu = 97.6\%, \sigma = 1.2\%$
Greedy Insertion	$\mu = 97.6\%, \sigma = 1.6\%$	$\mu = 97.0\%, \sigma = 1.5\%$	$\mu = 97.9\%, \sigma = 1.2\%$
Simple Reverse Faber	$\mu = 95.7\%, \sigma = 1.7\%$	$\mu = 94.4\%, \sigma = 2.7\%$	$\mu = 95.7\%, \sigma = 2.2\%$
GLS Reverse Faber	$\mu = 95.8\%, \sigma = 2.2\%$	$\mu = 94.5\%, \sigma = 2.6\%$	$\mu = 96.1\%, \sigma = 2.0\%$
LLS Reverse Faber	$\mu = 95.8\%, \sigma = 2.2\%$	$\mu = 94.5\%, \sigma = 2.5\%$	$\mu = 96.1\%, \sigma = 2.0\%$
Feature Aware Reverse Faber	$\mu = 95.7\%, \sigma = 2.1\%$	$\mu = 94.4\%, \sigma = 2.4\%$	$\mu = 95.9\%, \sigma = 1.8\%$

Table 2: Comparison results for 6% of original terrain size.

Simplification Method	Accuracy Rates		
	Identity	Full Projection	Half Projection
QEC	$\mu = 94.1\%, \sigma = 2.7\%$	$\mu = 93.1\%, \sigma = 3.0\%$	$\mu = 94.0\%, \sigma = 3.0\%$
Greedy Cuts	$\mu = 87.4\%, \sigma = 7.9\%$	$\mu = 89.8\%, \sigma = 4.7\%$	$\mu = 89.9\%, \sigma = 5.2\%$
Greedy Insertion	$\mu = 91.2\%, \sigma = 6.5\%$	$\mu = 92.2\%, \sigma = 2.9\%$	$\mu = 93.6\%, \sigma = 2.6\%$
Simple Reverse Faber	$\mu = 91.1\%, \sigma = 3.0\%$	$\mu = 89.5\%, \sigma = 4.7\%$	$\mu = 89.7\%, \sigma = 5.0\%$
GLS Reverse Faber	$\mu = 92.7\%, \sigma = 2.5\%$	$\mu = 90.3\%, \sigma = 4.4\%$	$\mu = 91.5\%, \sigma = 3.9\%$
LLS Reverse Faber	$\mu = 92.7\%, \sigma = 2.4\%$	$\mu = 90.2\%, \sigma = 4.5\%$	$\mu = 91.5\%, \sigma = 3.9\%$
Feature Aware Reverse Faber	$\mu = 92.2\%, \sigma = 2.6\%$	$\mu = 90.1\%, \sigma = 4.1\%$	$\mu = 91.5\%, \sigma = 3.5\%$

Table 3: Comparison results for 1.5% of original terrain size.

Simplification Method	Accuracy Rates		
	Identity	Full Projection	Half Projection
QEC	$\mu = 86.6\%, \sigma = 4.8\%$	$\mu = 86.5\%, \sigma = 5.3\%$	$\mu = 86.0\%, \sigma = 6.3\%$
Greedy Cuts	$\mu = 79.4\%, \sigma = 8.7\%$	$\mu = 79.0\%, \sigma = 10.0\%$	$\mu = 78.0\%, \sigma = 12.1\%$
Greedy Insertion	$\mu = 82.7\%, \sigma = 7.9\%$	$\mu = 81.6\%, \sigma = 10.2\%$	$\mu = 82.7\%, \sigma = 13.1\%$
Simple Reverse Faber	$\mu = 83.2\%, \sigma = 4.7\%$	$\mu = 82.2\%, \sigma = 7.0\%$	$\mu = 80.5\%, \sigma = 8.1\%$
GLS Reverse Faber	$\mu = 85.8\%, \sigma = 3.9\%$	$\mu = 83.7\%, \sigma = 5.9\%$	$\mu = 83.0\%, \sigma = 7.0\%$
LLS Reverse Faber	$\mu = 86.4\%, \sigma = 3.5\%$	$\mu = 84.0\%, \sigma = 5.7\%$	$\mu = 83.1\%, \sigma = 7.0\%$
Feature Aware Reverse Faber	$\mu = 86.6\%, \sigma = 3.3\%$	$\mu = 83.8\%, \sigma = 5.5\%$	$\mu = 83.5\%, \sigma = 6.0\%$

Table 4: Comparison results for true positives/negatives at 1.5% of original terrain size.

Simplification Method	Identity Accuracy		Half Projection Accuracy	
	Visible Rays	Not Visible Rays	Visible Rays	Not Visible Rays
QEC	$\mu = 90.0\%$ $\sigma = 7.5\%$	$\mu = 79.8\%$ $\sigma = 7.6\%$	$\mu = 96.8\%$ $\sigma = 3.6\%$	$\mu = 77.1\%$ $\sigma = 7.4\%$
Greedy Cuts	$\mu = 57.4\%$ $\sigma = 22.4\%$	$\mu = 82.5\%$ $\sigma = 7.9\%$	$\mu = 92.9\%$ $\sigma = 6.0\%$	$\mu = 68.0\%$ $\sigma = 11.5\%$
Greedy Insertion	$\mu = 58.6\%$ $\sigma = 23.6\%$	$\mu = 89.8\%$ $\sigma = 7.4\%$	$\mu = 85.1\%$ $\sigma = 13.4\%$	$\mu = 80.9\%$ $\sigma = 15.0\%$
Simple Reverse Faber	$\mu = 80.0\%$ $\sigma = 13.2\%$	$\mu = 72.8\%$ $\sigma = 22.0\%$	$\mu = 95.2\%$ $\sigma = 4.6\%$	$\mu = 63.1\%$ $\sigma = 23.3\%$
GLS Reverse Faber	$\mu = 80.8\%$ $\sigma = 13.7\%$	$\mu = 78.4\%$ $\sigma = 16.6\%$	$\mu = 93.9\%$ $\sigma = 4.1\%$	$\mu = 70.0\%$ $\sigma = 16.8\%$
LLS Reverse Faber	$\mu = 76.7\%$ $\sigma = 16.6\%$	$\mu = 81.1\%$ $\sigma = 15.3\%$	$\mu = 94.0\%$ $\sigma = 4.6\%$	$\mu = 70.3\%$ $\sigma = 16.3\%$
Feature Aware Reverse Faber	$\mu = 76.8\%$ $\sigma = 16.6\%$	$\mu = 80.9\%$ $\sigma = 16.4\%$	$\mu = 91.6\%$ $\sigma = 5.9\%$	$\mu = 71.8\%$ $\sigma = 16.0\%$