Smugglers and Border Guards - The GeoStar Project at RPI

W. Randolph Franklin
ECSE Dept, RPI*
frankwr@rpi.edu

Metin Inanc
CS Dept, RPI
inancm@rpi.edu

Zhongyi Xie
CS Dept, RPI
xiez@rpi.edu

Daniel M. Tracy
CS Dept, RPI
tracyd@rpi.edu

Barbara Cutler
CS Dept, RPI
cutler@cs.rpi.edu

Marcus V. A. Andrade
DPI - UF Viçosa - Brazil *and*
ECSE Dept, RPI
marcus@ecse.rpi.edu

ABSTRACT

We present the GeoStar project at RPI, which researches various terrain (i.e., elevation) representations and operations thereon. This work is motivated by the large amounts of hi-res data now available. The purpose of each representation is to lossily compress terrain while maintaining important properties. Our ODETLAP representation generalizes a Laplacian partial differential equation by using two inconsistent equations for each known point in the grid, as well as one equation for each unknown point. The surface is reconstructed from a carefully-chosen small set of known points. Our second representation segments the terrain into a set of regions, each of which is simply described. Our third representation has the most long term potential: scooping, which forms the terrain by emulating surface water erosion.

Siting hundreds of observers, such as border guards, so that their viewsheds jointly cover the maximum terrain is our first operation. This process allows both observer and target to be above the local terrain, and the observer to have a finite radius of interest. Planning a path so that a smuggler may get from point A to point B while maximally avoiding the border guards is our second operation. The path metric includes path length, distance traveled uphill, and amount of time visible to a guard.

The quality of our representations is determined, not only by their RMS elevation error, but by how accurately they support these operations.

Categories and Subject Descriptors

E.2 [**Data Storage Representations**]; I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling; I.4.2 [**Image Processing and Computer Vision**]: Compression

*Rensselaer Polytechnic Institute, 110 8th St, Troy NY 12180 USA

General Terms

algorithms, design, theory

Keywords

GIS, map compression, terrain modeling, terrain interpolation, terrain elevation data sets

1. INTRODUCTION

The increasing amounts of terrain data, from LIght Detection and Ranging (LIDAR) and Interferometric Synthetic Aperture Radar (IFSAR), such as from the 2000 Shuttle Radar Topography Mission (SRTM), create both an opportunity and a problem. The former is the set of new operations that can be performed, while the latter is the difficulty of storing the data and efficiently processing it. Many earlier "toy" algorithms have asymptotic times that grow too quickly with data size to remain useful. Paradoxically, as computers get faster and memory sizes get bigger, efficiency can become more important. In the authors' view, because hard disk capacities are growing faster than their speeds, the advantage of using only primary storage (a.k.a. internal memory or RAM), when feasible, compared to using external memory also grows.

In this context, *terrain* means elevation above some *geoid* (the assumed sea level, extrapolated over land). We do not consider the important geopotential issues of defining and determining geoids.

The subject of this paper is *Smugglers and Border Guards*, the GeoStar project at RPI. This paper describes the whole system and presents newer results. We are researching alternate terrain representations, and terrain operations. The operations are then used to evaluate the representations. As the project has progressed, various representations have been studied, ranging from new twists on classics, to more mature representations with immediate potential, like ODETLAP, through to radical ideas with both great potential and great difficulties, like scooping. The major operations being researched include multiple observer siting, and path planning to avoid the observers. Longer term ideas here include allowing earth moving operations on the paths. RPI is one of several teams funded under the GeoStar program by the Defense Science Office of DARPA.

Inanc[8] presented terrain segmentation. Westort[30] discussed sculpting terrain. Franklin[6, 9] investigated Pearlman and Said's SPIHT image compression algorithm for terrain. Gousie[15, 16, 18] presented new ideas for interpolating from contours.

The classic Triangulated Irregular Network, a piecewise linear, non tensor product, spline, was first implemented in GIS by Franklin[5]. Later extensions include Speckmann and Snoeyink[29], who process very large sets of irregular points externally. Lavery[23] and Zhang[31] have researched higher order, non tensor product, splines, which do not have extraneous oscillations. Most of the works on concise representation of the terrain data concentrate on efficient representation and manipulation of TINs. Samples of those works are in Park[22, 27], Kim[21], and Isenburg[20].

Yet another set of works target multiresolution representation of terrain elevations. Hoppe develops a progressive mesh transmission method based on edge collapse and feature retention in [19]. Related works can be found in Garland[13] and deFloriani[4]. In a different work, Losasso and Hoppe[24] model terrain elevation grids retaining different levels of detail, which are then compressed and progressively transmitted based on the field of view of the user. De Floriani and Magillo[3] compare different Multiresolution TIN proposals. Among multiresolution methods, wavelet based schemes are also notable. The SPIHT[28] and JPEG-2000[26] methods for multiresolution compression are patented, and hence less useful.

There has also been much research in the Computational Geometry community into surface reconstruction, e.g., of 3D objects from point clouds, Dey[2]. Reconstructing terrain is somewhat different since terrain is a single-valued function whose topology is known, and whose known points may be very unevenly spaced.

2. ALTERNATE TERRAIN REPRESENTATIONS

The first goal of the RPI GeoStar project is to produce alternate terrain representations that take less space, but are lossy. We are pursuing several representations in parallel, with a goal of both shortterm results and longterm potential. Of our major representations, ODETLAP is the most mature, segmentation is a work in progress, and scooping has the most potential.

2.1 Terrain Properties

Ideally, some formal terrain model would guide any evaluation of terrain representation. Since that does not yet exist, we can only study actual terrain. The challenge is that any mathematical representation of terrain must have a goal to acknowledge its properties, such as the following.

• Real terrain is more irregular than databases such as DEM-1 cells. Algorithms tested only on that data might unknowingly be exploiting their artificial smoothness. • Terrain is not differentiable many times, i.e., it is generally not C^n for $n > 0$. Indeed, the physical phenomena that generate terrain generally do not depend on, or generate, high order continuity. The major exception is the curvature, in the horizontal plane, of stream beds. • In places, the terrain is C^{-1}, i.e., discontinuous. Indeed, although techniques such as contour lines have difficulty representing them, these may be the most important features for many users. Discontinuities strongly affect both visibility and mobility. • The data is heterogeneous; different regions have different statistics. For example, river basins occur mostly above sea level, while mid-ocean ridges occur under sea level. Some regions above sea level are karst terrain, with sink holes, while other regions have rivers. • The heterogeneity gets worse if we consider other planetary bodies, such as the Moon, because of the varied formation mechanisms, such as impact craters or large volcanoes. • There are long range correlations, such as river basins, that may extend from one side of a continent almost to the other ocean. • Terrain is often not spatially symmetric in the horizontal direction. Rivers' headwaters, such as the Amazon's, are often near the opposite edge of the continent from their deltas.

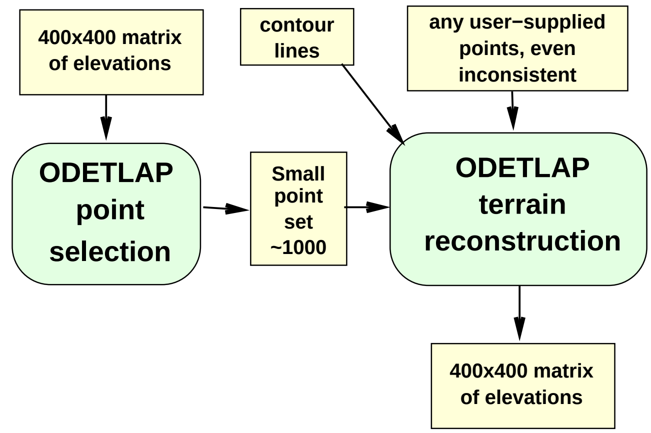

Figure 1: ODETLAP Process

2.2 Nonlinearity is Powerful

This research is biassed towards nonlinear, instead of linear, numerical techniques. Nonlinearity is a very powerful, albeit hard to use, approximation technique. Even for C^∞ quickly convergent functions like $\exp(x)$, the best rational approximation is more efficient than the best polynomial one, Newman[25]. (Note that the Taylor expansion is far from the best polynomial approximation; a Chebyshev is almost optimal.) However the true power is revealed when approximating functions like abs(x) or a step function. Because they are C^0 and C^{-1} respectively, uniform polynomial approximations do not exist. Note that the best rational approximation is more than just a Padé approximation, which is properly defined as a formal transformation from a polynomial, ignoring convergence.

2.3 ODETLAP

2.3.1 *Definition*

ODETLAP is an algorithm and implementation for

1. selecting a set of points that characterize a terrain elevation array, and

2. reconstructing a terrain elevation array from a set of points.

The process is summarized in Figure 1. Currently we use 400×400 arrays for implementation convenience in Matlab; larger arrays are possible with better matrix algorithms. The purpose of the ODETLAP point selection is that the resulting set of points, perhaps 1000 in number, characterize the surface well, and can be stored in much less space than

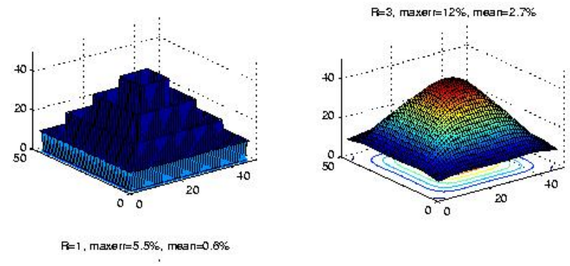

Figure 2: Overdetermined Laplacian Fitting of Nested Squares

the original array. The purpose of the ODETLAP reconstruction is to produce an array of elevations from a small point set. The reconstruction can process any set of points. For example it is also useful to fit a surface to a set of contour lines (the original application). The points may even be inconsistent; then a best fit will be computed.

2.3.2 Properties

We originally developed ODETLAP to address a shortcoming in some algorithms for filling in elevation contours to produce a matrix of elevations. That problem is that the original contours are too often visible in the generated surface. While we know little about formally modeling terrain (see section 2.1 above), we do consider it extremely unlikely that the real terrain is terraced at exact multiples of 10m.

ODETLAP also has many other advantages, which are generally not shared by competing surface approximation methods. • It can handle continuous contour lines of elevations, w/o needing to select only a subset of those points for processing. • It can handle kidney-bean-shaped contours w/o generating fictitious flat regions inside (as happens with interpolation methods that run straight lines out from the unknown point to the closest contour in each direction). • It can handle broken contour lines (unlike methods running straight lines until they hit a contour). • It can handle isolated points. • It can infer, from a set of concentric contours, a mountain top (local maximum) that is higher than the highest contour. • It can handle very unevenly distributed data. • It can conflate inconsistent data, say a small hi-precision region overlaid on a large lo-precision region. • It enforces continuity of slope across contours, so that they do not show in the resulting surface, i.e., no generated terraces.

How well ODETLAP works is shown in Figure 2, where the input, designed to be nasty, is several nested square contours, whose C^0 continuity at the corners should be challenging for any algorithm. The silhouette edge of the fitted surface shows almost no evidence of the contours. The max absolute error is 12%, and the mean error 2.7%, of the elevation range.

2.3.3 Algorithm

ODETLAP stands for *Overdetermined Laplacian Partial Differential Equation*. The ODETLAP representation consists of

1. $\mathcal{P}$, a set of important points on the surface. $\mathcal{P}$ is coded to minimize the number of bits needed to store it.
2. The ODETLAP algorithm for reconstructing the elevation matrix from $\mathcal{P}$.

ODETLAP is lossy, and allows a tradeoff between representation size and elevation error. ODETLAP allows for the terrain to be *progressively transmitted*. Since the points $p_i \epsilon \mathcal{P}$ are ordered by importance, they may be transmitted one-by-one. As each p_i is received, the receiver may reconstruct terrain T_i and evaluate it. If T_i is good enough, then the receiver tells the transmitter to stop. This is useful because there are applications where bandwidth, storage, and power consumption are still critical.

2.3.4 Point Selection

How should we select the points? Extensive experiments have shown that the reconstruction algorithm is surprisingly robust. Therefore the following greedy algorithm suffices. • Select $\mathcal{P}_0$, an initial set of points by a method such as using TIN incrementally to insert 100 points. • Reconstruct the DEM determined by those points. • Compute the error matrix between the original terrain and the above solution. • Select $\mathcal{A}_i$, the set of 10 points with the largest error. (We investigated other strategies, but those were all worse.) • Insert them into $\mathcal{P}_i = \mathcal{P}_{i-1} \cup \mathcal{A}_i$. • Repeat until the error is small enough.

2.3.5 Point Coding

Since our goal is to represent the surface as compactly as possible, coding the points to require the fewest number of bytes is as important as choosing the smallest set of points. Indeed, it might be better to select more points that can be coded into fewer bytes (because they form a regular pattern). We investigated several point coding strategies, and currently prefer the following method:

1. Represent the (x, y) as 1-bits in a 400×400 bitmap image that whose bits are otherwise 0.
2. Consider the image as a string of 160000 bits that has occasional 1s separated by strings of 0s. If two 1s are adjacent, then consider that the intermediate string of 0s has length zero.
3. This bit string may be represented as a list of the lengths of the zero strings (i.e., run-length encoding).
4. Use some efficient method to encode the lengths, noting that most of them are < 254 but some are larger.
5. Code the sequence of z by taking deltas and then using *bzip2*.

2.3.6 Surface Reconstruction

ODETLAP reconstructs the matrix z_{ij} of terrain elevations from $\mathcal{P}$, a scattered set of elevations, with an extension of a Laplacian partial differential equation (PDE), also known as a heat flow equation. A Laplacian PDE is solved by defining a sparse system of linear equations.

$$4z_{ij} = z_{i-1,j} + z_{i+1,j} + z_{i,j-1} + z_{i,j+1} \qquad (1)$$

for every unknown non-border point. Border points are a complicated special case to be discussed elsewhere. Unfortunately, the solution has several bad properties. • There is no information flow across contours; therefore the reconstructed surface is not C^2 there, and the contours are very

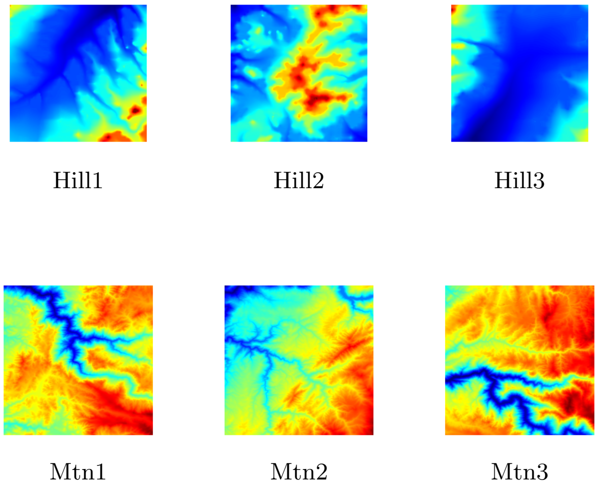

Figure 3: Test Data Sets

visible. • Between contours, the surface sags like a piece of cloth. • Since interpolated values can never be outside the range of the known datapoints that they are derived from, the terrain inside the innermost contour around a mountain top is flat like a mesa. To remedy that, we extend the Laplacian PDE as follows.

1. Define equation (1) above for *every* non-border point, whether known or unknown.
2. For each point with a known elevation h_{ij}, define an additional equation $z_{ij} = h_{ij}$.

Since the known points have two inconsistent equations, the system is now overdetermined. We solve for a best fit for our desired relative weights for the two different equation types, Gousie[15, 16, 17]. The resulting surface does not exactly fit the known points, which, given the (lack of) accuracy of real data, is an advantage. An alternative would be to consider the surface as a thin plate, and minimize its bending energy. The PDE is $z_{xx}^2 + 2z_{xy}^2 + z_{yy}^2 = 0$. However, this causes a ringing or Gibbs phenomenon, and adds extra complexity w/o a corresponding extra benefit.

ODETLAP's novelty is the overdetermined system, which was not feasible until recent large sparse system solution techniques were developed. Any resemblance to interpolation with springs is only superficial.

2.3.7 Results and Status

We used six test terrains, three hilly and three more mountainous, shown in Figure 3. Table 1 shows the results when 1000 points were used. All sizes are in bytes. The *XY Size* is the size of the (x, y) coordinates of the points, when run-length coded. The *Z Size* is the size of the Z-coordinates, delta encoded and bzipped. The *Total Size* is their sum. That is the size of the terrain in our alternate representation. The original size of each 400×400 terrain, at 2 bytes per point, is 320KB, and their ratio is the compression ratio of our representation. Since our representation is lossy, there is the usual size-accuracy tradeoff. Therefore, we give the RMS elevation error of our representation, and the elevation range of the data, and the ratio of the RMS error to the range, as a percentage.

Our current work includes optimizing ODETLAP by exploring various point selection techniques, better coordinate coding techniques, and different PDEs. We know how to run ODETLAP on larger datasets, by using a Page-Saunders algorithm as used by Childs[1], but are staying with 400×400 arrays for the moment since they are faster and easier to process.

2.4 Terrain Segmentation

Inanc[8] has shown that encoding terrain elevation data through segmentation is an enabling method for a lossy compression. In this work we propose some extensions, which can allow better plane compression and provide a lower average error.

The input to our problem is an elevation dataset T consisting of $N \times N$ elevation postings. Ideally each elevation posting is an (x, y, z) triplet but since we are dealing with a regular grid, the (x, y) values are implicit and only the z values are stored. Thus we can conveniently store T in a matrix of size $N \times N$. A further simplification is that DEMs often store their elevation values as 16-bit integers.

We attack the problem of finding the best fitting 2D manifolds by generating a set of candidates. Candidates are generated from small local terrain patches. One way is to partition our terrain into square tiles of size $t_s \times t_s$. To capture minute variations in the terrain we pick $t_s = 2$, thus generating $N^2/4$ tiles. Each tile contains t_s^2 elevations (z values), which are modeled by the following linear system: $Xc = z + \epsilon$. where predictor variables X are the implicit grid coordinates (x, y) and a constant factor. The fitting error is ϵ.

A multiple parameter linear regression function solves for the coefficient vector c of the best fitting plane. Those coefficients are stored in a list L for future consideration. For each entry in the list L, the plane is extrapolated from the small tile it originates, to the entire terrain. The (x, y) coordinates of the entire terrain T, together with a constant factor make the matrix X_t. The process generates a 2D manifold $\tilde{z}$, which is a crude approximation for the entire terrain: $\tilde{z} = X_t c$. The fitness of the approximation $\tilde{z}$ is tested using the infinity norm: $||A||_\infty = max\{|a_1|, |a_2|, ...\}$.

Thus we are interested in $||\tilde{z} - T||_\infty$. We would like to limit this value to a user specified constant we call loss factor: LF. A way of doing that is to limit the 2D manifold $\tilde{z}$ to a set of (x, y) coordinates, which meet the constraint. We call the size limited $\tilde{z}$ a segment: S. A segment S consists of a manifold $\tilde{z}$ and a set of (x, y) coordinates on that manifold. Our model depends on a set of segments, which cover all (x, y) coordinates in T.

2.4.1 Segment Selection

After our 2D manifold generation scheme populates the list L with candidates, we need to pick a minimal set that will cover all (x, y). Compression is achieved since a single segment may contain a large number of elevation postings, which are all concisely modeled. The obvious algorithm is the greedy heuristic, where at each step we pick the largest contributing segment and we stop when the coverage is achieved.

Data	XY Size	Z Size	Total Size	Orig Size	Cmpr. Ratio	RMS Err	Elev Range	Err %
Hill1	1250	1304	2554	320K	125.	3.62	505	0.7%
Hill2	1243	1354	2597	320K	123.	9.45	745	1.3%
Hill3	1279	1209	2488	320K	129.	1.72	500	3.4%
Mtn1	1228	1456	2684	320K	119.	17.34	1040	1.7%
Mtn2	1244	1424	2668	320K	120.	17.17	953	1.8%
Mtn3	1241	1503	2744	320K	117.	17.06	788	2.2%

Table 1: ODETLAP Results

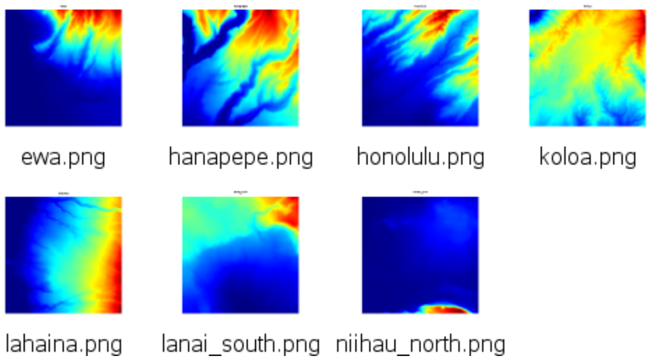

Figure 4: Seven 10m datasets, colormaps are not one-to-one.

	LF 5	LF 10	LF 20	Low Elev.	High Elev.
Ewa	39	22	11	0	333
Hanapepe	45	21	11	0	298
Honolulu	59	28	13	0	404
Koloa	119	55	31	136	933
Lahaina	35	19	10	0	281
Lanai-south	48	20	9	334	633
Niihau-north	57	30	13	0	333

Table 2: Number of segments, elevation extremes.

2.4.2 Results

We present results on seven different terrain elevation datasets. Those are 400 × 400 size datasets with horizontal resolution of 10m, from the 10m USGS DEMs covering Hawaii[14]. We used datasets containing different geological features (e.g., mountains, valleys, plateaus, hills, plains, cliffs), Figure 4. We try three different LF (loss factor) values of 5, 10 and 20. As expected the number of segments drops the higher the LF. This trend can be observed in Table 2. We also report the lowest and the highest elevation of the dataset on the same table.

For each segment we need to encode the plane coefficients and (x, y). We combine segments in a single indexmap and apply entropy coding with the PPMII encoder from the LEDA library. The resultant compressed size is in Table 3.

We observe that mountainous datasets, like koloa, compress less than hilly ones. Also, thanks to the low number of segments picked by the greedy algorithm, we have significant savings down from 320,000 bytes the original datasets

MaxErr	5	10	20	gzip
Ewa	7706	4769	2563	49692
Hanapepe	13759	8932	5762	77453
Honolulu	17997	11050	6852	83650
Koloa	26824	16168	10701	129754
Lahaina	10075	5399	3461	49153
Lanai-south	8154	4640	2611	35601
Niihau-north	6067	3262	1587	29273

Table 3: Compressed size of the indexmap vs gzipped dataset (bytes).

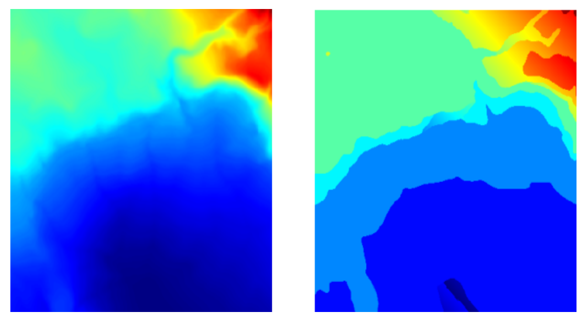

Figure 5: Lanai-south original and restored after compression using only 9 planes.

replace. We do not consider plane coefficients in these experiments. Assuming two bytes of storage per coefficient, we can calculate the storage requirements in bytes as six times the number of segments. For the worst case we have 119 planes to encode, each having 3 coefficients. Two bytes per coefficient gives us a total of 714 bytes, which is insignificant compared to the 26824 bytes required by the indexmap for the same dataset. We compare those results to gzip compression of the datasets. We again get a significantly smaller footprint at the price of a controlled data loss.

We expect the worst reconstruction to be the one using the least number of segments. In our case this is lanai-south at $LF = 20$, Figure 5.

2.5 Scooping

Imagine starting with a high plateau and carving the terrain with multiple passes of a giant shovel. For each pass, we insert the shovel at some point and then dig in a continuous motion towards the edge. As we do that, we may keep the depth of the blade level, or push it deeper into the earth, but we never make it shallower.

Scooping has several properties. • It will not create a local minimum. This desirable feature contrasts to every other known terrain representation method. • It naturally lends itself to the creation of complex drainage systems, again in contrast to other representations. • It is quite nonlinear, and so has a power not available to linear methods. • Slope discontinuities and cliffs can be created as desired. • The complete series of scoops representing a cell may be truncated at any point to produce a less accurate representation of that cell that still looks like terrain. Therefore the terrain may be lossily compressed and progressively transmitted.

None of the above properties pertain to a Fourier expansion. Scooping may also be visualized as a machining operation with a 3-axis drilling machine, if we assume that each pass of a drill extends to the edge of the workpiece, and the drill's depth never decreases during the pass.

More formally, this, our far-reaching proposed approach is to develop new mathematical morphological operators to enable parsimonious and compact representations of terrain, such as a *scooping* operator for representing terrain elevation. The uniqueness of this idea is to lay a formal foundation for terrain, to allow a formal inquiry into the best algorithms for applications, such as compression, visibility, mobility, drainage, the representation of multiple related data layers, and multiple data source conflation. This will improve on current methods of testing heuristics on test samples.

What terrain operators are appropriate, and how realistic they should be? While Fourier series are too unrealistic, a complete geological evolution model is too complex. Our scooping operator, analogously to scooping earth out of the side of a hill, will initially proceed as follows.

Although scooping has the greatest longterm potential, it is also the most difficult to research, and so we have no concrete results to report yet.

3. OPERATIONS ON TERRAIN

We researched and implemented two major terrain operations: multiple observer ("border guards") siting, and ("smugglers") path planning to avoid the observers.

3.1 Multiple Observer (Border Guard) Siting

Where should we site a set of observers, such as border guards, so every point on the terrain (or more likely, 90% of the points) can be seen by at least one observer? The goal is either to minimize the number of observers needed to cover a specific fraction of the terrain, or to maximize the amount of terrain covered by a given fixed number of observers. This process has various parameters, such as the observer and target height above the local terrain, and the radius of interest, the distance out to which each observer can see. A variant of the siting problem is to enforce *intervisibility*, requiring that enough observers can see each other that they form a connected graph, enabling observers to communicate with each other, perhaps indirectly. This research theme goes beyond the theme of more accurately computed viewsheds of single observers. We have a siting testbed, capable of easily processing level-1 DEMs. Figure 6 shows sample output with and without intervisibility being enforced. The terrain is the USGS Lake Champlain West cell. The details of this method are omitted since they have been reported earlier, in Franklin[7, 10, 11, 12].

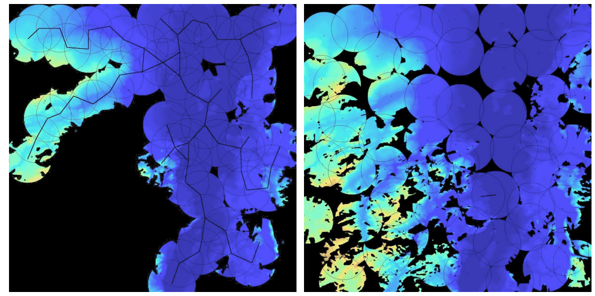

Figure 6: Multiple Observer Siting with and w/o Intervisibility

3.2 Smugglers' Path Planning

How should a smuggler travel to minimize his time visible to the optimally sited border guards?

The path finding routine implements the A* algorithm and computes the shortest path between opposite corners of the terrain while trying to avoid detection by the given set of observers. In the algorithm, the cost of moving from one point an adjacent point uphill is $\sqrt{h^2+v^2}\cdot(1+v/h)\cdot P$, where h is the horizontal distance between the points, v is the elevation difference, P is a *Visibility Penalty*, chosen to be $P = 100$ if the new cell is visible, or $P = 1$ if the new cell is not visible. If the new point is not uphill, the cost is simply $h * P$.

The path-finder is a two-pass system. On the first pass, all points are included in the search space, and each point is considered adjacent to its eight immediate neighbors. The result will be an approximately minimal path. On the second path, only points from the first path are included in the search space, and each point is considered adjacent to all other points on the path. The result will be a path that more nearly minimizes the Euclidean distance. In practice, this second pass is very efficient.

4. EVALUATION

In addition to computing the RMS error for the alternate representations, as a function of that representation's size, we sought a more sophisticated evaluation procedure, summarized in Figure 7. The goal is to answer the question of whether our alternate representations are suitable for sophisticated operations. That is, how good is a siting or path planning operation that is performed on terrain compressed with, say, ODETLAP? Designing an appropriate metric takes care. For instance, a small change in the terrain may cause a large change in a computed path, if several possible paths have approximately the same cost. What is important is whether the path computed on the alternate representation has the same cost as the path computed on the original representation.

Since the siting and path planning has many parameters, our evaluation is still preliminary. However, Table 4 has some indicative results. It evaluates one compressed terrain dataset on three metrics of increasing complexity.

1. *Viewshed Error:* A set of observers is sited on both the original and alternate representations, the two cumulative viewsheds computed, and the area of their symmetric difference reported.

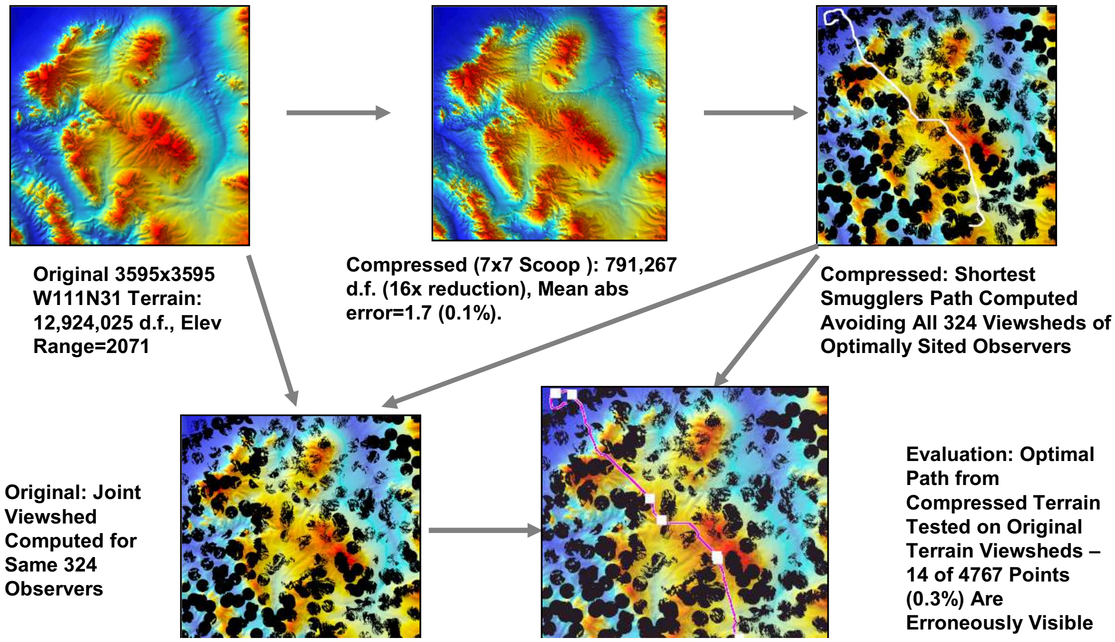

Figure 7: Smugglers' Path Planning on 16x Compressed "Scooped" Terrain Representation

2. *Path Length Error:* A smuggler's path is computed to avoid the observers in the two cases, and the difference in their lengths is reported.

3. *Path Visibility Error:* The path computed on the alternate terrain is transferred to the original representation and tested against the original viewsheds. The percent of that path that is now visible is reported.

Metric	JPEG 2000	ODETLAP
Viewshed Error	9.76%	9.04%
Path Length Error	5.81%	0.23%
Path Visibility Error	1.56%	0.27%

Table 4: Viewshed and Path Planning Evaluation of ODETLAP Terrain Compression

In each case, smaller numbers are better. We also performed these tests on the terrain compressed with JPEG 2000 to about the same size. The two schemes are competitive. There are cases where JPEG 2000 performs better on the less sophisticated metrics. However ODETLAP is much better on path planning. It also appears that our new approach is better when the terrain is very heterogeneous. ODETLAP also has the many other advantages listed earlier, which JPEG-2000 lacks. Also, we are still improving ODETLAP.

5. FUTURE

This is a work in progress with many open possibilities, such as scooping. For ODETLAP, we are investigating different point coding techniques and hierarchical extensions.

For terrain segmentation, we can imagine that there might be a spike or a well in the elevation data, which might not belong to either of the segments. To model these aberrations, we stipulate that certain points will be stored separately. This mechanism can also be used to store survey points, which have higher accuracy than the rest of the data.

For path planning, we are allowing earthmoving operations, and wish to minimize a sophisticated cost function, while respecting physical rules such as the maximum slope of the road under construction.

Finally, there is a great potential for end-to-end optimization of the representations and operations as parts of one unit, producing more compact representations supporting more powerful operations.

6. ACKNOWLEDGMENTS

This research was supported by NSF grants CCR-0306502 and DMS-0327634, by DARPA/DSO/GeoStar, and by CNPq-the Brazilian Council of Technological and Scientific Development.

7. ADDITIONAL AUTHORS

Additional authors: Frank Luk, CS Dept, RPI.

8. REFERENCES

[1] J. Childs. Development of a two-level iterative computational method for solution of the Franklin approximation algorithm for the interpolation of large contour line data sets. Master's thesis, Rensselaer Polytechnic Institute, 2003. `http://www.rh.edu/~chil7317/`. (Accessed 23 May 2006).

[2] T. K. Dey. *Curve and Surface Reconstruction : Algorithms with Mathematical Analysis*. Cambridge University Press, 2006.

[3] L. D. Floriani and P. Magillo. Regular and irregular multi-resolution terrain models: A comparison. In *Proceedings of the 10th Annual ACM International Symposium on Advances in GIS*, pages 143–148, 2002.

[4] L. D. Floriani, P. Magillo, and E. Puppo. Variant: A system for terrain modeling at variable resolution. *GeoInformatica*, 4(3):287–315, October 2000.

[5] W. R. Franklin. Triangulated irregular network program. `ftp://ftp.cs.rpi.edu/pub/franklin/tin73.tar.gz` (accessed 23 May 2006), 1973.

[6] W. R. Franklin. Compressing elevation data. In *Fourth International Symposium on Large Spatial Databases — SSD '95*. Portland, Maine, USA, 6–9 Aug 1995.

[7] W. R. Franklin. Siting observers on terrain. In D. Richardson and P. van Oosterom, editors, *Advances in Spatial Data Handling: 10th International Symposium on Spatial Data Handling*, pages 109–120. Springer-Verlag, 2002.

[8] W. R. Franklin and M. Inanc. Compressing terrain datasets using segmentation. In *Proceedings of SPIE Vol. 6313 Advanced Architectures, and Implementations XVI*, San Diego CA, 15-16 August 2006. International Society for Optical Engineering. 6313-17, Session 4.

[9] W. R. Franklin and A. Said. Lossy compression of elevation data. In *Seventh International Symposium on Spatial Data Handling*, Delft, Aug. 1996.

[10] W. R. Franklin and C. Vogt. Efficient observer siting on large terrain cells (extended abstract). In *GIScience 2004: Third International Conference on Geographic Information Science*, U Maryland College Park, 20–23 Oct 2004.

[11] W. R. Franklin and C. Vogt. Multiple observer siting on terrain with intervisibility or lo-res data. In *XXth Congress, International Society for Photogrammetry and Remote Sensing*, Istanbul, 12-23 July 2004.

[12] W. R. Franklin and C. Vogt. Tradeoffs when multiple observer siting on large terrain cells. In A. Riedl, W. Kainz, and G. Elmes, editors, *Progress in spatial data handling: 12th international symposium on spatial data handling*, pages 845–861. Springer, Vienna, 2006. ISBN 978-3-540-35588-5.

[13] M. Garland. Multiresolution modeling: Survey and future opportunities. In *Eurographics'99*, pages 111–131, 1999.

[14] Geomorphological Research Group. 10-meter data. `http://rocky.ess.washington.edu/data/raster/tenmeter/`, 2007. [Online; accessed 18-June-2007].

[15] M. Gousie and W. R. Franklin. Converting elevation contours to a grid. In *Eighth International Symposium on Spatial Data Handling*, pages 647–656, Vancouver BC Canada, July 1998. Dept of Geography, Simon Fraser University, Burnaby, BC, Canada.

[16] M. Gousie and W. R. Franklin. Constructing a DEM from grid-based data by computing intermediate contours. In E. Hoel and P. Rigaux, editors, *GIS 2003: Proceedings of the Eleventh ACM International Symposium on Advances in Geographic Information Systems*, pages 71–77, New Orleans, 2003.

[17] M. B. Gousie. *Contours to digital elevation models: grid-based surface reconstruction methods*. PhD thesis, Electrical, Computer, and Systems Engineering Dept., Rensselaer Polytechnic Institute, 1998.

[18] M. B. Gousie and W. R. Franklin. Augmenting grid-based contours to improve thin plate dem generation. *Photogrammetric Engineering & Remote Sensing*, 71(1):69–79, 2005.

[19] H. Hoppe. Progressive meshes. In *ACM Siggraph '96 Proceedings*, pages 99–108, 1996.

[20] M. Isenburg, Y. Liu, J. Shewchuk, and J. Snoeyink. Streaming computation of delaunay triangulations. In *ACM Transactions on Graphics, Proceedings of ACM Siggraph 2006*, volume 25, pages 1049–1056, July 2006.

[21] S.-S. Kim, S.-K. Choe, J.-H. Lee, and Y.-K. Yang. Level-of-detail-based rendering and compression for 3d gis. In *International Geoscience and Remote Sensing Symposium*, pages 1942–1944, 2001.

[22] Y.-S. Kim, D.-G. Park, H.-Y. Jung, H.-G. Cho, J.-J. Dong, and K.-J. Ku. An improved tin compression using delaunay triangulation. In *Proceedings of Computer Graphics and Applications*, pages 118–125, 1999.

[23] J. Lavery. Shape-preserving approximation of multiscale univariate data by cubic l1 spline fits. *Computer Aided Geometric Design*, 21:43–64(22), Jan. 2004.

[24] F. Losasso and H. Hoppe. Geometry clipmaps: Terrain rendering using nested regular grids. In *ACM Siggraph 2004 Papers*, pages 769–776, 2004.

[25] D. J. Newman. *Approximation with Rational Functions*, volume 41 of *Conference Board of the Mathematical Sciences, Regional Conference Series in Mathematics*. American Mathematical Society, 1979.

[26] M. J. Owen and M. W. Grigg. The compression of digital terrain elevation data (dted) using jpeg 2000. Technical Report DSTO-TR-1548, Defense Science and Technology Organization Salisbury, Australia, February 2004.

[27] D. Park, H. Cho, and Y. Kim. A tin compression method using delaunay triangulation. *International Journal of GIS*, 15(3):255–269, January 2001.

[28] A. Said and W. A. Pearlman. A new fast and efficient image codec based on set partitioning in hierarchical trees. *IEEE Transactions on Circuits and Systems for Video Technology*, 6:243–250, June 1996.

[29] B. Speckmann and J. Snoeyink. Easy triangle strips for TIN terrain models. In *Proc. 9th Canadian Conference on Computational Geometry*, pages 239–244, 1997.

[30] C. Westort and W. R. Franklin. Alternative sculpting hypotheses for terrain data compression. In *Proceedings of SPIE Vol. 6313 Advanced Architectures, and Implementations XVI*, San Diego CA, 15-16 August 2006. International Society for Optical Engineering. 6313-16, Session 4.

[31] W. Zhang, Y. Wang, S.-C. Fang, and J. E. Lavery. Cubic l1 splines on triangulated irregular networks. *Pacific Journal of Optimization*, 2(2):289–317, may 2006.