Slope Accuracy and Path Planning on Compressed Terrain

W. Randolph Franklin[1], Daniel M Tracy[1], Marcus A Andrade[1 2], Jonathan Muckell[1], Metin Inanc[1], Zhongyi Xie[1], and Barbara M Cutler[1]

[1] Rensselaer Polytechnic Institute
Troy, New York, 12180–3590, USA
frankwr@rpi.edu – http://wrfranklin.org/, tracyd@rpi.edu, marcus.ufv@gmail.com, muckej@rpi.edu, inancm@rpi.edu, xiez@rpi.edu, cutler@cs.rpi.edu
[2] DPI - UF Viçosa - Brazil

Abstract

We report on variants of the ODETLAP lossy terrain compression method where the reconstructed terrain has accurate slope as well as elevation. Slope is important for applications such as mobility, visibility and hydrology. One variant involves selecting a regular grid of points instead of selecting the most important points, requiring more points but which take less space. Another variant adds a new type of equation to the overdetermined system to force the slope of the reconstructed surface to be close to the original surface's slope. Tests on six datasets with elevation ranges from 505m to 1040m, compressed at ratios from 146:1 to 1046:1 relative to the original binary file size, showed RMS elevation errors of 10m and slope errors of 3 to 10 degrees. The reconstructed terrain also supports planning optimal paths that avoid observers' viewsheds. Paths planned on the reconstructed terrain were only 5% to 20% more expensive than paths planned on the original terrain. Tradeoffs between compressed data size and output accuracy are possible. Therefore storing terrain data on portable devices or transmitting over slow links and then using it in applications is more feasible.

Keywords: terrain compression, slope accuracy, path planning, ODETLAP.

1 Introduction

As ever larger quantities of higher resolution terrain data become available, such as using IFSAR and LIDAR, more efficient compression techniques become more important. This is especially true when it is desired to store the data on portable devices or to transmit the data over slow links. High-resolution data may also compress differently when it is qualitatively different

from the older data produced by interpolating contour maps derived from aerial photographs, since the latter are often artificially smooth.

Compression may be either *lossless*, where the restored data is identical to the original data, or *lossy*, where an error is introduced. This choice is not unique to terrain; audio data is also usually compressed lossily. Lossy compression is appropriate when the increased efficiency (i.e., the decreased size of the resulting file) is worth it, or when the original data is imperfect. That is, if the original data has an RMS error of 5 meters (m), then a compression algorithm introducing an average error of $0.5m$ is overkill.

The desired application for the terrain data influences the appropriate metric for evaluating the compression. The easy metric is RMS elevation error, Franklin and Said (1996). However, some parts of the terrain may be more important than others. For example, sharp points in the profile along the skyline are what viewers recognize. This author has had the experience of looking at a mountain range on the horizon while simultaneously looking at a commercial rendition of that same scene, and being unable to correlate the real world with the computer model. The problem resides in the computer model's lack of high spatial frequencies. This may be caused by using calculus tools such as Fourier or Taylor series that assume that the terrain is differentiable many times, and that high frequencies are less important than low frequencies. Both assumptions are false. Not only does nothing in the physics of terrain formation select for smoothness, but rather the reverse. Erosion causes undercutting and slumping leading to cliffs, that is, elevation discontinuities.

Slope is one terrain property that is important to represent accurately. The slope of terrain influences *mobility* (it is difficult to drive up a cliff), *accessibility by air* (aircraft cannot land on a slope), *hydrology* (steeper slopes erode more quickly) and *visibility* (changes in slope are recognizable, and observers sited on a break in the slope may be able to see more).

Slope is often ignored because the assumption is that it comes for free once the elevations are represented sufficiently accurately. However, differencing any imprecise function amplifies the errors. Also, from math analysis we know that approximating a function $f(x)$ more accurately, i.e., $\limsup_{i\to\infty}|(f_i(x) - f(x))| \to 0$, gives no guarantees about $\limsup_{i\to\infty}|(f_i'(x) - f'(x))|$, which may increase without bound. Indeed, it was such paradoxes that motivated the formalization of calculus in the 19th century.

The compression methods introduced here are extensions of ODETLAP, Franklin et al. (2007), and summarized in Figure 1. Briefly, ODETLAP solves a sparse overdetermined system of linear equations for the elevations z_{ij} in an array where a few points' elevations h_{ij} are known. Each known point has an equation

$$z_{ij} = h_{ij} \tag{1}$$

Every non-border point, known or not has an equation

$$4z_{ij} = z_{i-1,j} + z_{i+1,j} + z_{i,j-1} + z_{i,j+1} \tag{2}$$

Border points form a messy special case of no deep theoretical interest, but with the following practical difficulties. Not including equations for border points may lead to the system being underdetermined. A careless choice of equation may bias the surface to being horizontal at the borders, without physical justification.

Since there are more equations than unknowns, the system is overdetermined; we solve for a best fit. The two classes of equations are weighted differently, depending on the relative importance of accuracy versus smoothness. Weighting Equation 1 more highly relatively to Equation 2 makes the resulting surface more accurate but less smooth. A small degree of inaccuracy enables a large degree of smoothness. Indeed, a design requirement for ODETLAP was that, when interpolating between contour lines, that the contour lines not be visible in the resulting surface. Also, broken contours and even isolated points may be processed. These desirable properties are not always shared by competing surface fitting techniques.

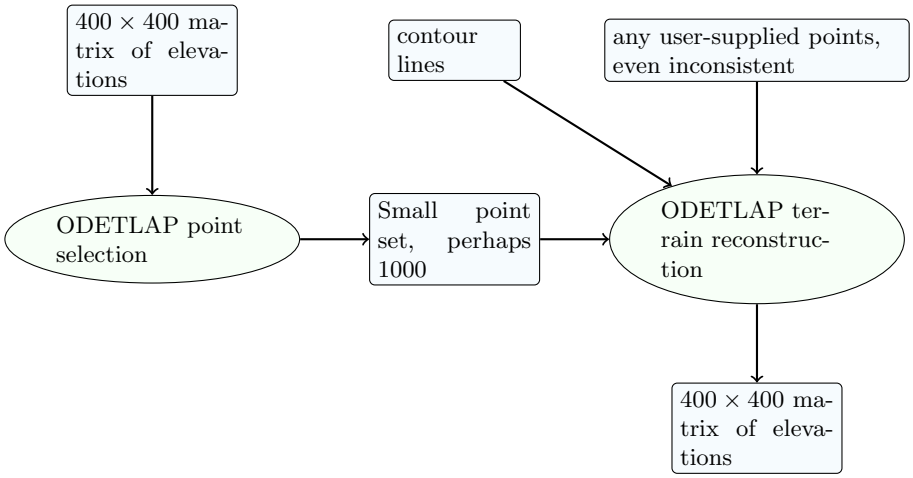

Fig. 1. ODETLAP Process

There is little prior art on compressing slopes, apart from some descriptions of fundamental limits. A resolution of 25m or lower cannot identify steep slopes correctly Kienzle (2004). A resolution of 30m with elevations in meters results in a precision of slope calculations no better than 1.9°, Hunter and Goodchild (1997).

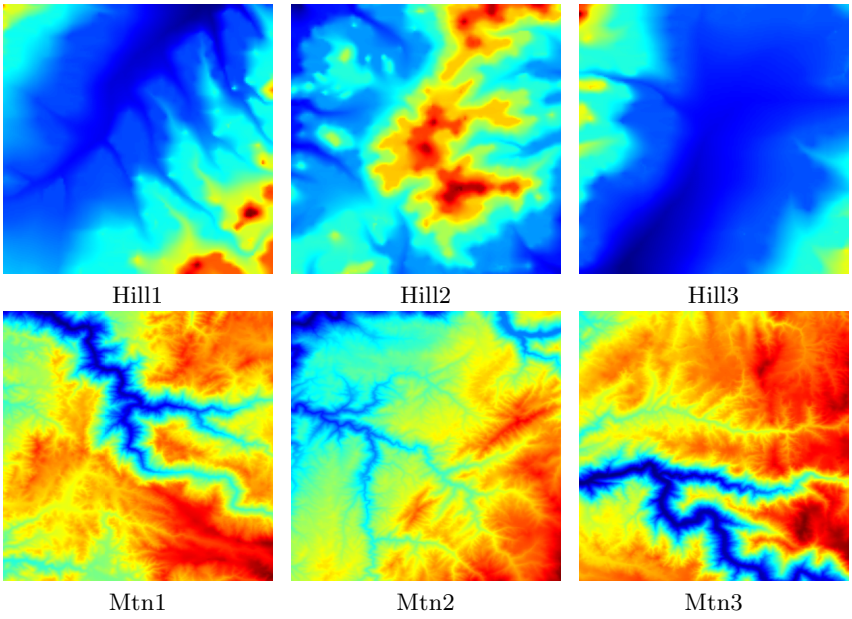

Fig. 2. Sample level-II Datasets

Table 1. ODETLAP TIN+Greedy Results

	Hill1	**Hill2**	**Hill3**	**Mtn1**	**Mtn2**	**Mtn3**
Elevation range	505m	745m	500m	1040m	953m	788m
Original size	320KB	320KB	320KB	320KB	320KB	320KB
Compressed size	2984B	5358B	1739B	9744B	9670B	9895B
Compression ratio	107:1	60:1	184:1	33:1	33:1	32:1
(Compressed/Orig size), %.	1.68%	1.33%	1.66%	0.91%	1%	1.23%
# pts selected	1040	2080	520	4160	4160	4160
RMS elevation error	8.49m	9.93m	8.31m	9.48m	9.55m	9.68m
RMS slope error	2.81°	5°	1.65°	8.34°	8.36°	7.87°

2 Terrain Data Structures

The underlying terrain data structure for the research presented in the paper is a matrix or array of elevations. There are other possibilities. One alternative would be high-order spherical harmonics as used in geopotential modeling. However, they are not as applicable to terrain, if only because their complexity grows quadratically with their accuracy. Wavelets of various types are used somewhat, and may become more popular in the future. The major alternative to an array of elevations is a Triangulated Irregular Network (TIN). Franklin (1973) did the first implementation (under the direction of Douglas

and Peucker) of the TIN in Geographic Information Science. In the next section we use an updated version of that program, Franklin (2001). In contrast to Isenburg et al. (2006), Franklin (2001) operates incrementally, in the spirit of the Douglas-Peucker line generalization algorithm, Douglas and Peucker (1973) (independently discovered by Freeman and Ramer, Ramer (1972)). In each iteration, it greedily inserts the point that is farthest from the current surface. It can process arrays of up to $10^4 \times 10^4$ points in core. The time to completely TIN a level-I DEM with 1201^2 points (until the max error is under 0.5m) is under 30 CPU seconds on a laptop. Also in contrast to Isenburg, it imposes no restrictions on the size of the generated triangles. However, because it operates out of core, Isenburg can process much larger datasets.

One disadvantage of a TIN compared to an array is the increased complexity of storing the data compactly, since in a naive implementation, most of the storage will be devoted to the topology. Also, rendering the terrain without producing a triangular appearance can require either very many triangles or a smoothing operator. Finally, representing slope accurately, one topic of this paper, appears problematic with a TIN. On the other hand,, unlike an array a TIN is not tied to a particular coordinate system and can better represent large regions of the earth.

3 ODETLAP TIN+Greedy

The first question is, how well does ODETLAP represent slopes? Slope is qualitatively somewhat different from elevation: its autocorrelation distance is smaller, but it requires fewer significant bits.

We used six 400×400 test datasets, three hilly and three mountainous, extracted from level-2 DEMs. 400×400 is a resolution that we can easily process using the default sparse linear equation solver in Matlab; larger resolutions are possible with other techniques, such as the Paige-Saunders method used by Childs (2003, 2007). ODETLAP TIN+Greedy, the basic version of ODETLAP, selects points with the following two stage process.

1. Use our incremental triangulated irregular network (TIN) program to select $\mathcal{P}$, an initial set of important points.
2. Fit a surface $\mathcal{S}$ to $\mathcal{P}$.
3. If $\mathcal{S}$ is sufficiently accurate then stop.
4. Otherwise, find the 10 to 30 points of the original 400×400 points that are farthest from $\mathcal{S}$. When forming this batch of points to insert, we assume that very close points are redundant, and require points to be at least a couple of pixels apart. Increasing this *forbidden zone* beyond that confers no additional advantage. Points are inserted in batches because of the time to recompute the surface in step 2.
5. Insert the new points into $\mathcal{P}$.
6. Go back to step 2.

The (x, y) are compressed by forming a 400×400 bitmap showing the points' locations, then compressing it with a runlength code. The resulting size is not much worse that the information-theoretic limit. The z are compressed with various methods such as *bzip2*.

ODETLAP's running time depends greatly on the input elevation matrix's sparsity. Basic ODETLAP on a 400×400 terrain took about 4.6 minutes when 5.4% of the elevations were known, and about 7.5 minutes on a 2.2GHz processor when 7.5% of the elevations were known. Denser input matrices required over 15 minutes.

Table 1 summarizes the results. ODETLAP TIN+Greedy compressed these terrains by factors ranging from 30:1 to 100:1 compared to the original binary file, with RMS elevation errors less than $10m$ and a slope error ranging from 1.7°to 8.4°, depending on the terrains' ruggedness. The next question is, what is the tradeoff of size versus accuracy? Figures 3 and 4 answer this.

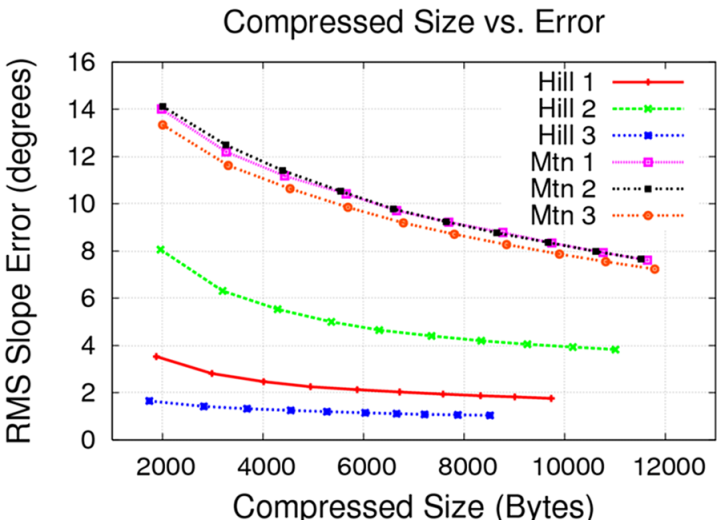

Fig. 3. ODETLAP Tin+Greedy Size – Elevation Accuracy Tradeoff

A major advantage of ODETLAP TIN+Greedy is that it selects the points in order of importance, and so permits progressive transmission of the points. However there will be a size penalty since compressing points incrementally is less efficient than compressing them in one set. Indeed, the former method stores the order of the points, which the latter does not. Therefore, for N points, the penalty will be at least $N \lg N$ bits (the information content of selecting one permutation from $N!$ permutations), but will probably be more. A larger storage cost of this method compared to the following one is caused by these points' positions being irregular.

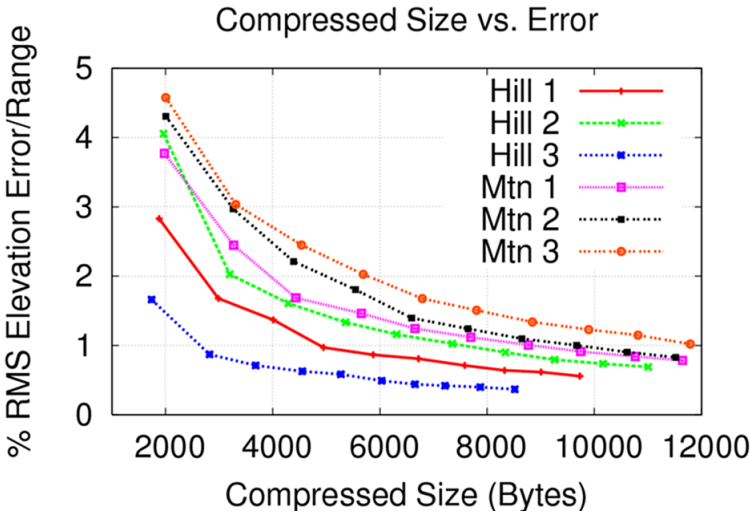

Fig. 4. ODETLAP Tin+Greedy Size – Slope Accuracy Tradeoff

Table 2. ODETLAP Regular grid w/o DCT Results

	Hill1	**Hill2**	**Hill3**	**Mtn1**	**Mtn2**	**Mtn3**
Elevation range	505m	745m	500m	1040m	953m	788m
Original binary size	320KB	320KB	320KB	320KB	320KB	320KB
Compressed size	619B	1591B	315B	4710B	4659B	4777B
Compression ratio	517:1	201:1	1016:1	68:1	68:1	67:1
# pts selected	529	1369	256	4489	4489	4489
RMS elevation error	8.4m	9.2m	9.1m	9.1m	8.9m	8.8m
RMS slope error	4.2°	6.5°	3.0°	9.9°	9.9°	9.4°

4 ODETLAP-Regular Grid

With this alternative, instead of greedily selecting the N most important points, we select points on a regular grid uniformly spaced, say 40×40, or every $10th$ point in x and y. The first advantage is that the points' locations (x, y) do not need to be stored. Second, since the z form a regular array, using any image processing compression technique becomes easy. However, since ODETLAP-Regular grid does not adapt to changes in the spatial complexity of the terrain, it will require more points and it may miss small features. Is this tradeoff worth it?

Table 2 shows the results. For each dataset, the number of points was increased, keeping a square grid of points but selecting more points equally spaced in columns and rows, until the RMS elevation error was under $10m$. For the same number of points, the compressed size varied slightly because

Table 3. ODETLAP Regular grid with DCT Results

	Hill1	**Hill2**	**Hill3**	**Mtn1**	**Mtn2**	**Mtn3**
Elevation range	505m	745m	500m	1040m	953m	788m
Original binary size	320KB	320KB	320KB	320KB	320KB	320KB
Compressed size	306B	807B	172B	2194B	2027B	2013B
Compression ratio	1046:1	397:1	1860:1	146:1	158:1	159:1
# pts selected	529	1600	225	4489	4489	4489
RMS elevation error	9.6m	10.0m	9.7m	9.7m	10.0m	9.9m
RMS slope error	4.3°	6.5°	3.0°	10.°	10.°	9.9°

different sets of z compress differently. After achieving an RMS elevation error smaller than 10, the z coordinate of the selected points are compressed using *bzip2*. Comparing with the ODETLAP TIN+Greedy results, the compression ratio is about 2 times better. On the other hand, the RMS slope error is a little worse.

As an extension, we lossy compressed z as follows. The selected z values were rounded off while preserving an RMS error less than 10 and then transformed with a Discrete Cosine Transform (DCT). A DCT, widely used in image compression, is similar to a Fourier series, but uses a set of higher and higher frequency square waves instead of sines and cosines to approximate a function. The more square waves are used, the more accurate the approximation is, but the more space it takes, Wikipedia (2008).

Then the resulting sequence was compressed using *bzip2*. For a given elevation or slope error, this method compresses better. See Table 3.

5 Path Planning

The next test of our compression algorithm was for *path planning* on terrain, where the traveler is hiding from a set of observers who have been optimally positioned, Franklin and Vogt (2006); Franklin (2002). That is, if we use the compressed terrain to plan a path, how good is that path? We chose the following metric, designed to incorporate several factors affecting real paths.

$$\mathcal{C} = \sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2} \cdot \left(1 + \max\left(0, \frac{\Delta z}{\sqrt{\Delta x^2 + \Delta y^2}}\right)\right) \cdot (1 + 100v) \quad (3)$$

The first term says that shorter paths are better. The second says that moving uphill is expensive. The third term says that being seen by an observer is very expensive ($v = 1$ if the traveler is in sight, 0 otherwise). Note that the uphill term means that this metric is not symmetric; the optimal path from a to b has a different cost, and is not simply the reverse of, the optimal path from b to a. Therefore some other path planning algorithms will fail. Further, since a 400×400 dataset has 400^2 points, graph traversal algorithms

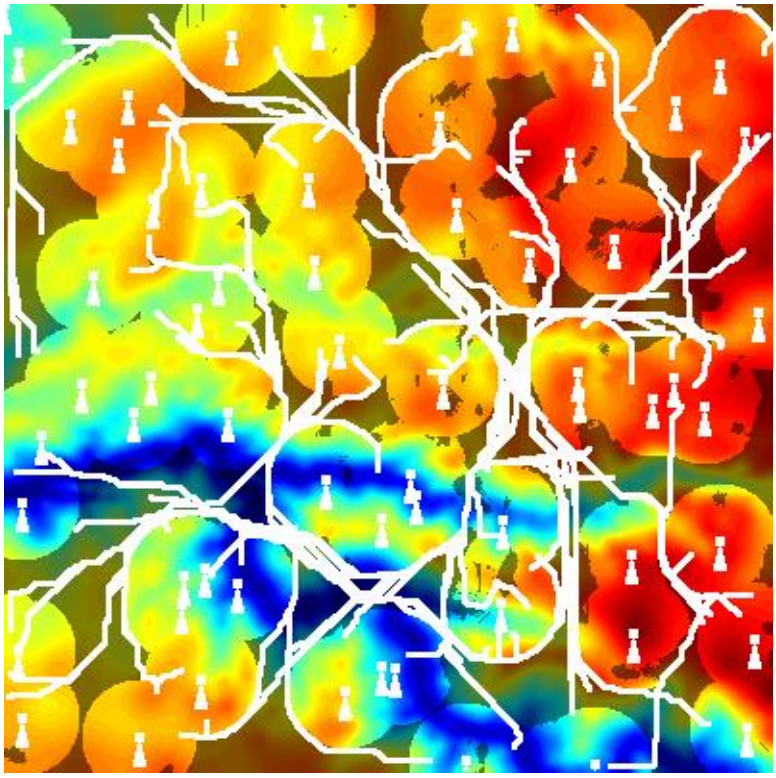

Fig. 5. Many optimal paths avoiding viewsheds on Mtn3

employing an explicit cost matrix are infeasible. Finally, some search strategies that climb hills in parameter space (unrelated to climbing hills on the terrain) stop at local optima, which is undesirable. To address all these concerns, we created a modified A* search procedure, Tracy et al. (2007), and used it to plan paths between many pairs of sources and destinations on each dataset. Figure 5 shows many paths plotted on the *mtn3* dataset. Each little white lighthouse represents an observer. The surrounding colored region is the observer's viewshed. Gaps in the viewsheds are caused by ridges hiding the terrain behind them. The dark regions of the figure are invisible to all the observers. Figure 5 also shows choke points in the terrain, which are traversed by many paths. Those would be candidates for siting future observers.

How to evaluate the path computed on the compressed terrain is also important, and the obvious choices may be wrong. For example, the cost of the path computed on the compressed terrain is meaningless. Indeed, if the terrain were compressed to be flat, then paths computed on it would have no cost for moving uphill and so would be artificially cheap, which is wrong. Even comparing the distance between two paths is meaningless for evaluating them.

Table 4. Increased cost of paths computed on compressed terrain

Data	Compressed size	Compression ratio	Cost increase
Hill1	1763	182	5.5%
Hill2	1819	176	6.1%
Hill3	1607	199	4.4%
Mtn1	1925	166	19.2%
Mtn2	1884	170	18.2%
Mtn3	1946	164	17.0%

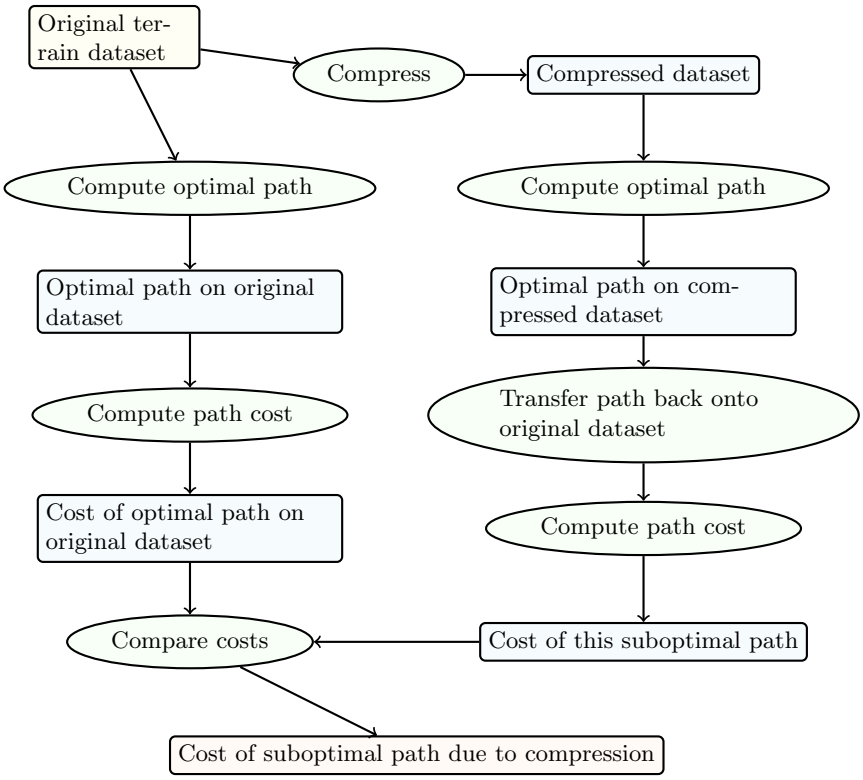

Fig. 6. Compressed path evaluation algorithm

Indeed, two paths may be legitimately quite different but have the same cost; we don't care. Our metric recognizes that the purpose of computing a path on any terrain, compressed or original, is to use it in the real world. Therefore, we transfer the path back to the original terrain dataset, and evaluate it there, as shown in Figure 6.

Table 4 shows the path inefficiency when our six terrains are compressed by factors of at least 164:1. Paths computed on these very compressed terrains were suboptimal by only 6% to 19%.

6 ODETLAP+Slope

With ODETLAP TIN+Greedy, we insert points with the greatest absolute elevation error. Since the goal is to represent slopes accurately, one obvious improvement would be to insert points with large slope errors. Another possibility would be to insert groups of close points since fixing the elevations of a set of close points should also fix the slope in that neighborhood. Both these ideas, and many other experiments not detailed here, had disappointing results. It was time to extend the ODETLAP equations themselves.

Three different representations of the terrain need to be distinguished in order to understand this section.

Original representation This is the original 400×400 matrix of elevation posts that we wish to compress.

Compressed representation This compact version is what would be transmitted or stored on portable devices.

Reconstructed terrain The compressed representation would be reconstituted into this new 400×400 matrix in order to be used.

For ODETLAP+Slope, we supplement the two existing types of equations, 1 and 2 with a new type of equation designed to force the slope in x and y to be more accurate.

$$z_{i+1,j} - z_{i-1,j} = h_{i+1,j} - h_{i-1,j} \tag{4}$$

$$z_{i,j+1} - z_{i,j-1} = h_{i,j+1} - h_{i,j+1} \tag{5}$$

This sets the Δz between the northern and southern neighbors equal to its known value, and sets the Δz between the western and eastern neighbors equal to its known value. The elevation of the center point is not used. It was done this way because these two Δzs are the values used by the Zevenbergen-Thorne method, Zhou and Liu (2004), a common method for computing slopes, Zevenbergen and Thorne (1987). (The cross product of the two vectors becomes the normal to the surface.) Our system permits the indices to be chosen arbitrarily, to allow for pairs of nonadjacent points to be used; this is a topic of potential future research.

Since the system is overconstrained, the relative weights of the different types of equations can be set depending on the relative importance of slope accuracy, elevation accuracy, or smoothness. The idea for this addition is that the extra freedom of allowing elevations to drift somewhat, provided that the slopes remain accurate, may allow greater slope accuracy.

Either ODETLAP TIN+Greedy or ODETLAP Regular grid may serve as the basis for adding equations 4 and 5. In the former case, we iterate the process of greedily inserting the points whose reconstructed slopes are the worst. ODETLAP TIN+Greedy requires fewer points but ODETLAP Regular grid requires less space to store each of the points on the grid (though any extra irregular points off the grid will take the same space as in ODETLAP TIN+Greedy. As before, we add points in batches for efficiency, and use forbidden zones around the points to prevent close pairs of points to be added in the same iteration, although a point $\mathcal{P}$ added in one iteration may be adjacent to a point added in an earlier iteration, if $\mathcal{P}$'s error is sufficiently large.

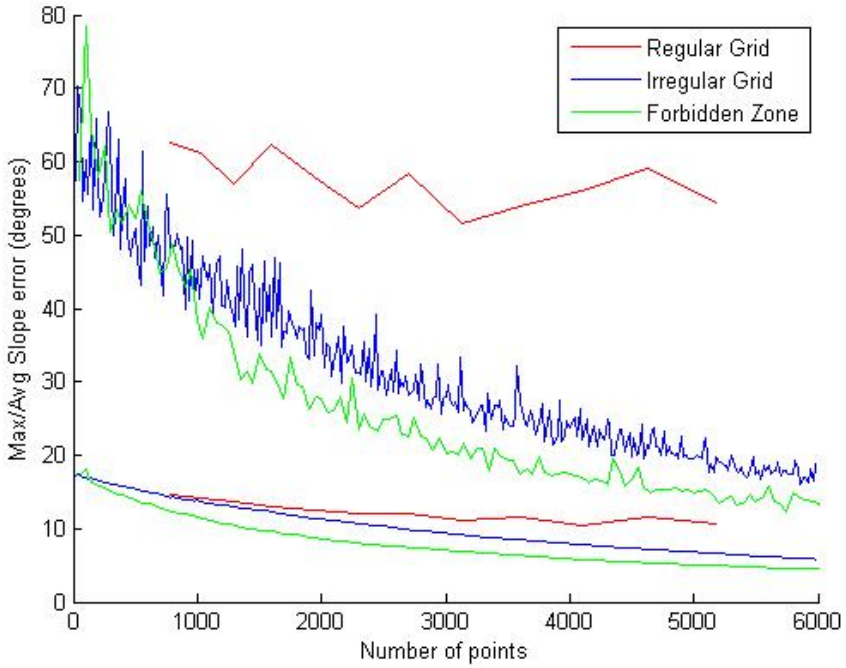

Fig. 7. Slope accuracy vs number of points for Mtn2

Figure 7 shows how three variants of this idea perform on the Mtn2 dataset. They are: selecting points in a regular grid, greedily selecting irregular points, and greedily selecting irregular points using an 11×11 forbidden zone. The x-axis is the number of points in the compressed representation (out of a total of 160000 points). The y-axis shows the average and maximum slope errors (the three *max* curves are the higher ones). The best method is greedily selecting irregular points using a forbidden zone.

7 Conclusions and Future Work

Representing terrain, including slope, with ODETLAP has great potential. We are now exploring some of its variations, and applying it to high resolution urban data. The major problem to be addressed is the computation space and time required. We are also extending our path planning algorithm for road construction. Here, we are allowed to modify the terrain with cuts and fills when planning the path. Another application of ODETLAP is terrain smoothing, which might be applied to any other terrain representation. Indeed that ODETLAP was created to smooth or interpolate between contours so that those contours would not be visible in the resulting surface.

One problem with all compression techniques is that they do not preserve *Hydrology*. Regions of the world where the terrain was formed by erosion caused by surface water flow have distinctive properties. There are almost no actual local minima (basins, depressions), because they become lakes. In the few depressions in the coterminous USA, such as the Great Salt Lake, Salton Sea, and Crater Lake, the water either evaporates or percolates away. However, there are many fictitious depressions caused by errors in measuring the terrain or by insufficiently fine sampling, Maidment et al. (1997). That is, the water may exit a depression via a canyon that is so narrow that it fits between two adjacent elevation posts, and so is missed. We are now studying how the hydrological properties of the terrain under compression. This is an instance of the general problem of compressing multiple layers of cartographic data where preserving the relationships between the layers after reconstruction is at least as important as preserving the individual layers' accuracy.

The most general problem is to construct the terrain from a set of mathematical operators that force the resulting terrain to have the desired properties. For instance, suppose that we carved the terrain out of a block of earth with a shovel, with repeated applications of the following operation. Place the shovel touching the earth at a some point. Move the shovel along any trajectory ending at the edge of the earth, provided that the shovel always gets lower and lower. Then, repeat with another shovel path, etc. The terrain that is created will never have an interior local minimum. That is, it will be hydrologically "correct". Can we reduce this idea to practice?

8 Acknowledgement

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

References

Childs J (2003) 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

Childs J (2007) Digital elevation modeling journal. `http://terrainmap.com/`, [accessed 28-May-2007]

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

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

Franklin WR (2001) Triangulated irregular network computation. `http://wrfranklin.org/pmwiki/Research/TriangulatedIrregularNetwork`, [accessed 8-June–2007]

Franklin WR (2002) Siting observers on terrain. *in* D Richardson and P van Oosterom, eds, Advances in Spatial Data Handling: 10th International Symposium on Spatial Data Handling. Springer-Verlag pp. 109–120

Franklin WR, Inanc M, Xie Z, Tracy DM, Cutler B, Andrade MVA and Luk F (2007) Smugglers and border guards - the geostar project at rpi. *in* 15th ACM International Symposium on Advances in Geographic Information Systems (ACM GIS 2007). Seattle, WA, USA

Franklin WR and Said A (1996) Lossy compression of elevation data. *in* Seventh International Symposium on Spatial Data Handling. Delft

Franklin WR and Vogt C (2006) Tradeoffs when multiple observer siting on large terrain cells. *in* A Riedl, W Kainz and G Elmes, eds, Progress in spatial data handling: 12th international symposium on spatial data handling. Springer Vienna ISBN 978-3-540-35588-5

Hunter G and Goodchild M (1997) Modeling the uncertainty in slope and aspect estimates derived from spatial databases. *Geographical Analysis* **29**(1), 35–49

Isenburg M, Liu Y, Shewchuk JR, Snoeyink J and Thirion T (2006) Generating raster dem from mass points via tin streaming. *in* GIScience. pp. 186–198

Kienzle S (2004) The effect of dem raster resolution on first order, second order and compound terrain derivatives. *Transactions in GIS* **8**(1), 83–111

Maidment DR, Olivera F, Reed S, Ye Z, Akmansoy S and McKinney DC (1997) Water balance of the Niger river basin in West Africa. *in* 17th Annual ESRI User Conference. San Diego, CA
URL: *http://www.ce.utexas.edu/prof/maidment/atlas/esri97/ESRI.htm*

Ramer U (1972) An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing* **1**, 244–256

Tracy DM, Franklin WR, Cutler B, Andrade MA, Luk FT, Inanc M and Xie Z (2007) Multiple observer siting and path planning on lossily compressed terrain. *in* Proceedings of SPIE Vol. 6697 Advanced Signal Processing Algorithms, Architectures, and Implementations XVII. International Society for Optical Engineering paper 6697-16

Wikipedia (2008) Discrete cosine transform — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Discrete_cosine_transform&oldid=196623065`, [accessed 8-March-2008]

Zevenbergen LW and Thorne CR (1987) Quantitative analysis of land surface topography. *Earth Surface Processes and Landforms* **12**(1), 47–56

Zhou Q and Liu X (2004) Analysis of errors of derived slope and aspect related to DEM data properties. *Computers and Geosciences* **30**(4), 369–378