Ph.D. Showcase: Slope Preserving Lossy Terrain Compression

PhD Student:
Zhongyi Xie
Rensselaer Polytechnic Institute
110, Eighth Street
Troy, NY, USA
xiez2@cs.rpi.edu

PhD Supervisor:
W. Randolph Franklin
Rensselaer Polytechnic Institute
110, Eighth Street
Troy, NY, USA
wrf@ecse.rpi.edu

PhD Student:
Daniel M. Tracy
Rensselaer Polytechnic Institute
110, Eighth Street
Troy, NY, USA
daniel.m.tracy@gmail.com

ABSTRACT

Accurate terrain representation with appropriate preservation of important terrain characteristics, especially slope steepness is becoming more crucial and fundamental as the geographical models are getting more complex and commonly used. Based on our earlier success in Overdetermined Laplacian Partial Differential Equation (ODETLAP), which allows for compact yet accurate compression of the Digital Elevation Model (DEM), we propose a new terrain compression technique which focuses on improving slope accuracy in compression of high resolution terrain data. With high slope accuracy and a high compression ratio, this technique will help all sorts of geographical applications that require a high precision in slope yet also have strict constraints on data size. Our proposed technique has the following contribution: we modify the ODETLAP system by adding slope equations for some key points picked automatically so that we can compress the elevation without explicitly storing slope values. By adding these slope equations, we can perturb the elevation in such a way that when slope is computed from the reconstructed surface, they are accurate. Note we are not storing the slope explicitly, instead we only store the elevation difference at a few locations. Since the ultimate goal is to have a compact terrain representation, encoding is also an integral part of this research. We have used Run Length Encoding (RLE) and linear prediction in the past, which gave us substantial file size reduction. In addition to that, we also propose a Minimum Spanning Tree based encoding scheme that takes advantage of the spatial correlation between selected points. Our technique is able to achieve a 1:10 compression at the cost of 4.23 degree of RMS slope error and 3.30 meters of RMS elevation error.

Categories and Subject Descriptors

I.3.5 [**Computing Methodologies**]: Computer Graphics Computational Geometry and Object Modelling

General Terms

Algorithms, Experimentation, Theory

Keywords

GIS, PDE solver, terrain modeling, terrain elevation data set compression

1. INTRODUCTION

The availability of high resolution Digital Elevation Model (DEM) data including LIDAR, stereo photogrammetry and Doppler radar has greatly assisted the development of GIS related research. High resolution DEM data has made possible terrain analysis in geomorphology, water flow modeling, and the great success of GPS (Global Positioning Systems). However, accurate representation of terrains as well as compact compression and transmission of terrain data still remain problems, especially when it comes to topographical derivatives of the terrain. The primary derived topographical parameters associated with DEMs are slope and aspect [9], which are fundamental to several terrain related applications including hydrology, visibility and environmental sensing [6]. However, while slope is technically the derivative of the elevation, lossy compression of the elevation could amplify errors so that when recomputing the slope from the lossily compressed terrain, it could contain far more error than the elevation. Perhaps because of the incorrect assumption that accurate elevations imply accurate slopes, there appears to be no prior art on this topic.

This paper presents a lossy compression technique that preserves elevation and slope with high accuracy and also produces a good compression ratio, which facilitates all kinds of DEM data handling, including high speed data transmission. The work is based on the ODETLAP framework, which is very effective in reconstructing smooth terrains with high accuracy from lossily compressed elevations with certain properties [18]. Since the previous versions of ODETLAP compress elevation well, our focus will be on improving the effectiveness of preserving the slope accuracy throughout the compressing and decompressing process. In addition to getting higher accuracy, improving compression ratio is also among our goals. Due to the cost of solving large linear system, this technique has cubic complexity in terms of the input DEM size. However, we can still process DEM of $16,000 \times 16,000$ using parallelism in about 30 minutes [11].

The SIGSPATIAL Special, Volume 2, Number 3, November 2010.

2. PRIOR ART

How shall the elevations of the unknown positions be determined? The first law of geography is a well accepted principle. It states that everything is related to everything else, but near things are more related than distant things [14].

Some well known methods that directly apply this principle include proximity polygon, inverse distance weighting and kriging. All of them essentially set the elevation of an unknown position (i,j), $z_{i,j}$, to be a weighted average of known elevations h_l where $l = 1, 2, \ldots, k$:

$$z_{i,j} = \sum_{l=0}^{k} w_{(i,j),l} h_l \tag{1}$$

Proximity polygon (or *Voronoi polygon* or *nearest point*) sets $z_{i,j}$ to its nearest known neighbor, which means $w_{(i,j),l} = 1$ for the nearest known position but $w_{(i,j),l} = 0$ for all the others [13].

Inverse distance weighting, as the name suggests, sets $w_{(i,j),l}$ to the inverse power of distance between (i, j) and the known position l, usually square [10]. *Kriging* is a geostatistical approach, in which all control point data are involved in finding optimal values of the general weighting function $w(s)$ for a known point distant s from the unknown position. The main assumption here is that the covariance between two elevations depends solely on the distance between the positions [7].

Another popular approach involves fitting splines. In this case, first-order and even second-order continuity are explicitly enforced, thereby having an additional advantage of ensuring the slope of the surface is smooth.

Compressed sensing is a technique that has existed since 1970s and it was recently discovered by David Donoho [1] that it is possible to reconstruct a sparse signal from a set of samples whose size is so small that Nyquist-Shannon sampling theorem would tell the opposite. The difficulty with the classical methods is that minimizing L_2 norm is feasible but often leads to poor results while minimizing L_0 norm makes the problem NP-hard. Donoho proved that L_1 norm is equivalent to L_0 norm, which makes it possible to solve the problem using linear programming.

However, in most cases such interpolating the known points is not necessary because of the measurement imprecision. *Approximation*, which allows relaxation from the measured values, allows much desirable overall prediction results and much smoother surfaces. De-correlating elevation fluctuations from the density of known positions is vital because the way data are collected does not imply anything about the roughness of the terrain.

Trend surface analysis is a representative technique used for surface approximation. It involves specifying a general form of a mathematical function at the beginning. This is the trend which is expected to represent a large scale systematic change that extends from one map edge to the other. Then we fit the function with the sample data aiming least squares, a process also known as regression. A review of the technique can be found from Wren [17].

Numerous methods have been proposed to compute slope from a DEM. The slope can be calculated in one of the following ways: either by using trigonometry or by using differential geometry [16]. We pick the Zevenbergen-Thorne method [19], a differential geometry based method, which offers good results. It first computes the gradient in the horizontal and vertical directions separately by taking the difference of neighbors along the direction. Then we take cross product of the two to get the slope normal.

Compression of raster DEM data is supported by most GIS software packages. GRASS simply uses run length encoding, which is lossless, meaning it doesn't compress that much. ArcGIS offers several options: LZ77(lossless), JPEG(lossy) and JPEG2000(lossy).

3. BASIC ODETLAP

ODETLAP was originally created to interpolate from contour lines to an array of elevations. Its advantages include being able to work with contour lines (continuous or intermittently broken), inferring mountain tops inside a ring of contours, and enforcing continuity of slope across contours. All these are favorable features of natural looking terrain.

3.1 ODETLAP Framework

ODETLAP [2, 4, 11, 18] operates on an elevation grid where only a few points are known, computes a surface over the entire grid. ODETLAP, for Overdetermined Laplacian Partial Differential Equation, is an extension of a Laplacian PDE

$$\frac{\delta^2 z}{\delta x^2} + \frac{\delta^2 z}{\delta y^2} = 0 \tag{2}$$

to an overdetermined system of equations. Each point in the elevation matrix induces an equation setting it to the average of its neighbors.

$$z_{i,j} = (z_{i-1,j} + z_{i+1,j} + z_{i,j-1} + z_{i,j+1})/4 \tag{3}$$

(For border points, only the two or three neighbors that exist are used. This biasses the slope towards zero at the border, but that is insignificant in this context.) In addition to that, each point whose elevation is known induces another equation, an *exact equation*.

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

where $h_{i,j}$ is the given elevation at that point. For each unknown point, we create only an averaging equation. Therefore, if k of the n^2 points have known elevations, there are $n^2 + k$ equations for the n^2 unknown variables. Even for a point of known elevation, we consider its actual elevation to be unknown and solve for it. Obviously, the resulting elevation will be close to the reported elevation, but will not be exactly it, because of the influence of nearby known points. Because the known elevations are only approximate, that is appropriate.

When there are more equations than unknowns, and the system of equations is inconsistent, we optimize for a least squares solution. Expressing the inconsistent system as $Ax = b$, the error is $e = Ax - b$, and we want to find an x to minimize $||e||$. Mathematically, that occurs at $x = (A^t A)^{-1} A^t b$. However, that formula is never used because it is slower and uses more space, especially for sparse systems such as

we have here, than any of various algorithms used by Matlab [12].

In this least squares solution, multiplying both sides of an equation by a constant will scale that equation's error and change its relative importance in the solution. We multiply all our averaging equations by a parameter R, which trades off accuracy and smoothness. A larger R causes the solution to be smoother, but to be farther from the known points. The basic ODETLAP has only this one parameter, which gives it great conceptual simplicity. Figure 1 shows the impact of R on the resultant surface.

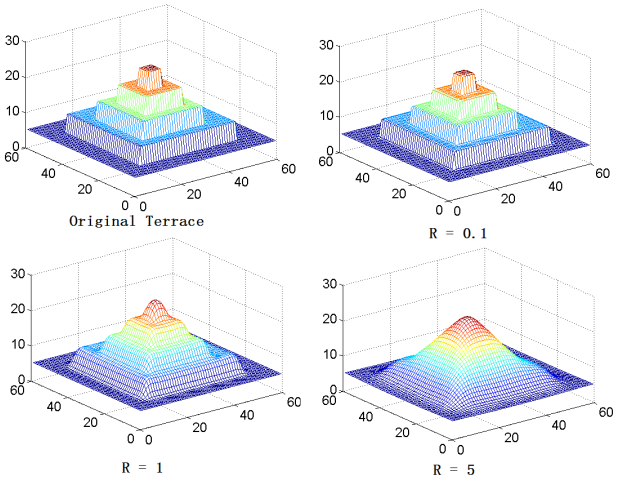

Figure 1: ODETLAP approximation of nested squares

ODETLAP has the following advantages. First, it can handle continuous contour lines of elevation, while allowing them to have gaps. Second, in addition to broken contour lines, isolated points can also be handled well, it is even possible to produce a surface that infers mountain tops inside innermost contours. Third, ODETLAP could enforce slope continuity across contours and generally shows no visible indication of input contours, i.e. no generated terraces. See Figure 1, which shows a surface interpolated over four nested square contours, an example chosen to be particularly difficult to interpolate. With a mean error of 4.7%, there is no evidence of terracing on the silhouette, which is the place where it would be most visible. Also the local max is nicely inferred.

The cost is that ODETLAP uses more resources, both memory and time. Its running time depends first on the cube of the number of rows in the DEM, and second on the number of known points. For an $n \times n$ dataset with k known points, the running time is of $\Theta(n^3 + k)$. Solving a 600×600 dataset with 23 630 known points takes about 13 CPU-minutes and 1.1GB of writable memory on a 2.8GHz Lenovo Thinkpad W700 with 4GB of main memory running Ubuntu 9.04 Linux and 64 bit Matlab R2009a. For larger datasets, we can partition the problem, solve the pieces, and smoothly merge the solution pieces [11].

3.2 ODETLAP based DEM Compression Algorithm

ODETLAP can be used as a lossy compression technique for the raster DEM data because once we obtained a point set S, we can generate an approximation of the original raster DEM using ODETLAP. With a carefully chosen S, we can produce a terrain with higher elevation and slope accuracy.

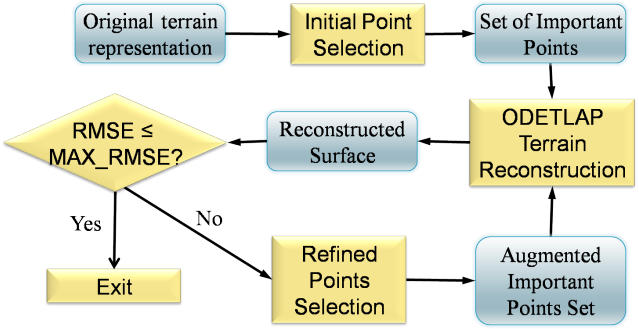

Figure 2: ODETLAP DEM compression algorithm

The ODETLAP compression algorithm's outline is shown in Figure 2 and the pseudo code is given in Algorithm 1. Starting with the original DEM, there are two point selection phases: Firstly, the initial point set S is built by selecting a regular grid over the original DEM and an initial approximation is computed using the equations 3 and 4. Given the reconstructed surface, a stopping condition based on an error measure is used. In practice, we have used the root-mean-square error(RMSE) as the stopping condition.

$$\text{RMSE} = \sqrt{\frac{\Sigma(Z_i - Z_t)^2}{N}}$$

Z_i : Computed DEM elevation at a given point
Z_t : True DEM elevation at a given point
N : Total Number of points

If this condition is not satisfied, the second step is executed. In this step, k $\geq$ 1 points from the original terrain are selected by the method described in 3.3 and they are inserted in the existing point set S; this extended set is used by ODETLAP to compute a more refined approximation. As the algorithm proceeds, the total size of point set S increases and the overall error converges.

Algorithm 1 ODETLAP DEM Compression

Require: T = Original Terrain
Ensure: S = Output Point Set,RS = Restored Surface
  S = InitSelection(T)
  RS = ODETLAP(S)
  while RMSE(RS, T) > MAX_RMSE **do**
    $S = S \cup$ Refine(T, RS)
    RS = ODETLAP(S)
  end while
  return S

3.3 Refined point selection - Greedy algorithm

After the initial point set is obtained, ODETLAP is used to reconstruct the elevation matrix. This matrix has high error with respect to the original terrain, mostly due to the limited size of the initial point set. As shown in Figure 2, refined points selection is applied and a set of additional points is chosen and added to the existing points set S to form the augmented points set. The way we choose new points is greedy: we find a set of points with the greatest absolute vertical error. The size of the set in our experiments is intentionally kept small (10% or smaller) so that for a given total number of points, more iterations could be used to reduce the error as much as possible. This is actually a trade off between accuracy and computation time. The augmented set is then given to ODETLAP to reconstruct a more refined approximation. The newly obtained approximation is again examined with respect to the original terrain against our stopping condition, which is either relative RMSE: compute the RMSE of the approximation and check if its ratio against the RMSE of the first approximation is less than a predefined threshold or absolute RMSE: define a value for the maximum acceptable RMSE.

3.4 ODETLAP Slope Compression

Dan Tracy [3, 15] extended the ODETLAP by adding two equations

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

$$z_{i,j+1} - z_{i,j-1} = h_{i,j+1} - h_{i,j-1} \tag{6}$$

where the meanings of z and h remain consistent with that of equation 3 and 4. Slope equations are structured in this manner to match the way these quantities are used in the Zevenbergen-Thorne slope method. Similar to the parameter R for regular ODETLAP, we use introduce a new parameter RS to control the weight of slope equations. So now the overdetermined system contains these equations:

- $R \times [x_{i,j} = (x_{i+1,j} + x_{i-1,j} + x_{i,j+1} + x_{i,j-1})/4]$
- $x_{i,j} = h_{i,j}$
- $RS \times [z_{i+1,j} - z_{i-1,j} = h_{i+1,j} - h_{i-1,j}]$
- $RS \times [z_{i,j+1} - z_{i,j-1} = h_{i,j+1} - h_{i,j-1}]$

The ODETLAP slope compression algorithm is then a slight modification of Algorithm 1. Like before, we start with an initial guess. In each iteration, a certain number of points with greatest slope errors are picked nonrepetitively. Here we collect the points together with the gradient in x and y directions (called Δ_x and Δ_y separately). The terrain is then approximated using the new slope ODETLAP equations and the whole process is repeated until the desired accuracy is reached.

4. ENCODING

The whole ODETLAP DEM compression algorithm mainly does one thing: selecting points that would best represent the terrain in such a sense that when given to ODETLAP, would produce most accurate terrain. However, selecting points is not the end for compression, a more compact representation would be more economical and useful in practical applications. To improve the compression ratio, the points are compressed using the following strategies: the (x, y, z) coordinates are split into (x, y) pairs and z alone. The former are compressed using an adapted run length encoding method, described below in section 4.1. The z sequence is compressed using linear prediction and then by bzip2.

4.1 Run Length Encoding

The coordinates (x, y) are different from z because they distribute uniformly within the matrix dimension's values, for example from 1 through 400, while z values follow certain geographical distribution yet to be discovered. The run length encoding is a simple lossless compression technique which, instead of storing the actual values in the sequence, stores the value and the count of sequence containing the same data value. A run is just a consecutive sequence that contains the same data value in each element. Since the (x, y) values correspond to positions in a matrix, we need to store only a binary bitmap where each location indicating whether the corresponding point exists in S or not. There is no need to store the original (x, y) pairs. Thus, given a binary matrix of size $N \times N$, the method is the following: For each run length L, test if $L < 254$, then use one byte for it; if $254 \leq L < 510$, then use FE as a marker byte and use a second byte for $L - 254$; if $510 \leq L < 766$, then use FF as a marker byte and use a second byte for $L - 510$; lastly, if $L \geq 766$, then use FFFF as a two byte marker and use next two bytes for L.

For our test cases, we made some histogram of the run length which shows that most runs are below 512, that means for most runs we need only 1 byte to store it. Here we assume all runs are shorter than 65535, which is a reasonable value for terrains of 400×400 resolution and point set S with reasonable size.

4.2 Encoding of elevation using prediction

Normally elevation data contains a high degree of correlation and that means we may predict the elevation value from its neighbors. The method of linear predication has been very successful in image processing [8] and the idea is very simple: to predict the vale of next element in the sequence, just use the last seen element and keep the correction by saving the difference between the two. The sequence z that we are going to compress contains elevation information in points selected by the ODETLAP algorithm.

In addition to linear prediction, we are considering other prediction modes which also exploit the correlation between points selected. Using Minimum Spanning Tree(MST) to reorder the points is one of them. The hypothesis is that the MST formed by points selected by ODETLAP reflects certain terrain structure that can be used to exploit spatial correlation between the elevation values. Consider the ridge of the mountain for example, suppose points selected by ODETLAP has a bigger chance to fall on the ridges, and points on the same ridge would likely to have similar elevation values, which could help reduce the compressed size if we use the correct spatial based prediction scheme.

Encoding of slope pairs selected in section 3.4 is done pretty similarly. The difference is that now we need to compress $(x, y, z, \Delta_x, \Delta_y)$ pairs. The first two are still done by run

length encoding and the last three are compressed separately using prediction and bzip2.

5. RESULTS AND ANALYSIS

5.1 Test dataset

For this paper, our test dataset are six level 2 DTED DEMs. DTED level 2 has relatively higher resolution and accuracy comparing to its relatives DTED-0 and DTED-1. It has post spacing of one arc second (approximately 30 meters). The size of each 400×400 test data, at 2 bytes per point, is 320KB. The plots of our six dataset are given in Figure 3. Some statistics of the datasets are given in Table 5.1.

Data	Elev Range	Elev σ	Slope Range	Slope σ
hill1	505	78.87	50.97	4.64
hill2	745	134.4	49.82	7.87
hill3	500	59.32	42.75	3.09
mtn1	821	146.0	52.81	9.64
mtn2	953	152.4	49.93	8.99
mtn3	788	160.7	60.38	11.15

Table 1: Some statistics of the test data

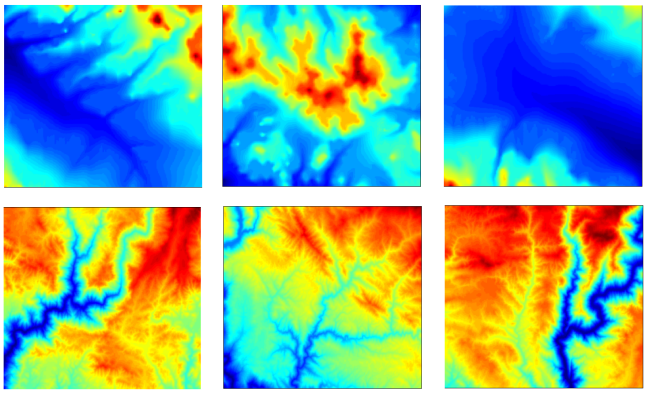

Figure 3: Test 400×400 30-meter Level 2 DEM

5.2 Evaluation criteria

Evaluating the reconstructed surface is nontrivial. We start with the most basic metric: maximum absolute and RMS elevation error. In Figure 4 we have the compressed file size versus the maximum and RMS elevation error of the reconstructed (decompressed) terrain. We could see the terrain is fairly accurate since the RMSE is well below 10 for a 1/10 compression ratio(recall the original binary size for a 400×400 DEM is 320KB). 10 meters of accuracy is already below the accuracy of DTED-2, which is 18 meters of absolute vertical accuracy [5]. It means some of the fine detail that we are losing are possibly just random noise but it is always good to have a compression technique that works well in a high resolution setting since more accurate data are becoming available in the near future [5].

Although we always compute statistics, our real metric here is, "does it look good?". Even elevation changes smaller than the known precision are unacceptable if the slopes have artifacts. Accurate slopes are also important for computing erosions, watersheds and for path planning. In Figure 5 we compare the elevation and slope from two stages of our

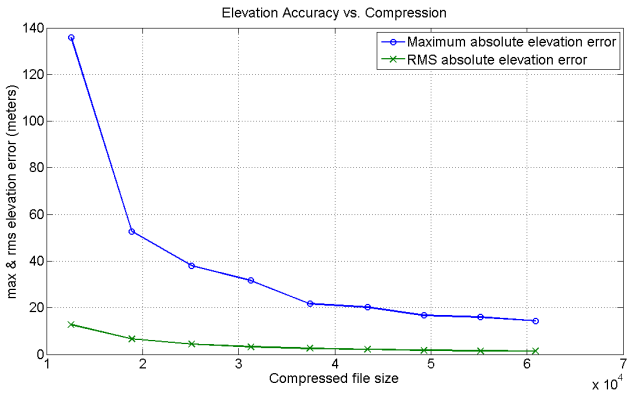

Figure 4: Elevation error vs. compressed file size.

compression and the original DEM. The top row are the elevation visualization and bottom row are the slope visualization. The first column are the results after the first iteration of Algorithm 1, which has a compression ratio of 1:25.3(12645 Bytes); the second column are from the second iteration, the compression ratio is 1:16.9(18975 Bytes). We could also find their corresponding maximum and RMS error in Figures 4 and 6. From the figures, we could see that the first iteration preserves the general trend of both slope and elevation, but lost some of the high resolution detail because of the high compression ratio. However, after more points are included in the second iteration, a lot more details are restored and most of the important features are captured.

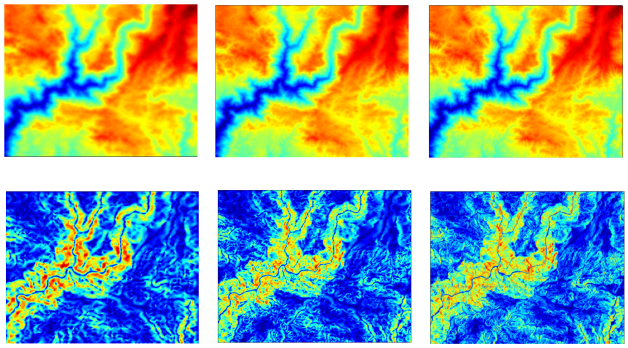

Figure 5: Reconstructed terrain and slope at different compression levels. From left to right: after first iteration, after second iteration and the original DEM; Top row: elevation visualization, bottom row: slope visualization.

Accuracy of elevation implies nothing about accuracy of slope. (It was the realization of counter-intuitive properties like this that motivated the formalization of calculus into the new field of mathematical analysis in the 19th century.) In Figure 6, we have the compressed file size versus the slope error. Considering the 30 meters spacing of DTED-2 DEM data, we could see RMS error of our reconstructed terrain is pretty accurate(RMSE of 5 degrees for 1:10 compression).

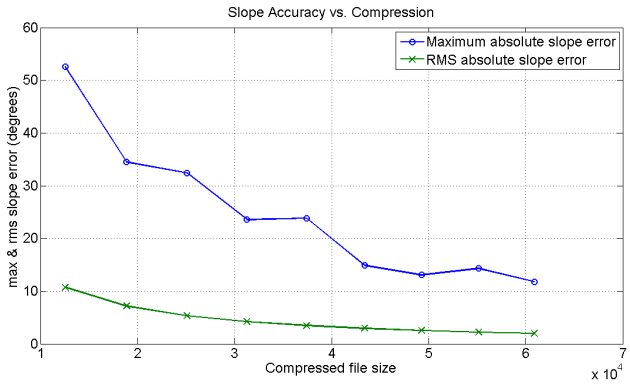

Figure 6: Slope error vs. compressed file size.

6. CONCLUSION AND FUTURE WORK

In this paper we described new algorithm to lossily compress the terrain. The focus is on how to achieve higher compression ratio while maintaining the elevation and slope accuracy. ODETLAP framework is very useful for reconstructing terrain surface from very sparse and unevenly distributed elevation data. We made several modification to the original ODETLAP system so that without explicitly storing slope values, we can achieve higher slope accuracy in the output terrain. We also discussed run length encoding, linear prediction and minimum spanning tree as ways to encode the intermediate data so that the compression ratio could be higher.

One of the future direction of our research is the extension of ODETLAP to preserve monotonic terrain structures besides slope. An example would be the terrain compression that preserves shorelines and cliffs. There are two ways to achieve this: either by explicitly storing the shoreline/cliff structure along with terrain itself or by postprocessing the compressed terrain produced by ODELTAP to generate new shorelines/cliff that match the original. The relevance of this extension to the main theme of this paper is that both strive to maintain some important terrain features throughout the terrain compression process. By effectively preserving these features, the compressed terrain becomes more useful and meaningful for the related geographical applications.

7. ACKNOWLEDGEMENT

This research was partially supported by NSF grant CMMI-0835762. We thank professor Barbara Cutler, professor Badrinath Roysam, Tsz-Yam Lau and You Li for valuable discussions on terrain representation and compression.

8. REFERENCES

[1] D. L. Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, 52(4):1289–1306, April 2006.

[2] W. R. Franklin, M. Inanc, Z. Xie, D. M. Tracy, B. Cutler, and M. V. A. Andrade. Smugglers and border guards: the geostar project at RPI. In *15th ACM International Symposium on Advances in Geographic Information Systems*, Seattle, WA, 2007.

[3] W. R. Franklin, D. M. Tracy, M. Andrade, J. Muckell, M. Inanc, Z. Xie, and B. Cutler. Slope accuracy and path planning on compressed terrain. In *Symposium on Spatial Data Handling*, Montpelier FR, 2008.

[4] M. B. Gousie and W. R. Franklin. Augmenting grid-based contours to improve thin plate DEM generation. *Journal of Photogrammetry and Remote Sensing*, 71(1):69–79, 2005.

[5] B. Heady, G. Kroenung, and C. Rodarmel. High resolution elevation data(HRE) specification overview. In *ASPRS/MAPPS 2009 Conference*, San Antonio, Texas, 2009.

[6] K. H. Jones. A comparison of algorithms used to compute hill slope as a property of the DEM. *Computers & Geosciences*, 24(4):315 – 323, 1998.

[7] D. G. Krige. A statistical approach to some mine valuations and allied problems at the Witwatersrand. Master's thesis, University of Witwatersrand, 1951.

[8] P. Maragos, R. Schafer, and R. Mersereau. Two-dimensional linear prediction and its application to adaptive predictive coding of images. *Acoustics, Speech, and Signal Processing*, 32:1213–1229, 1984.

[9] L. D. Raaflaub and M. J. Collins. The effect of error in gridded digital elevation models on the estimation of topographic parameters. *Environmental Modelling & Software*, 21(5):710–732, May 2006.

[10] D. Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings of the 1968 ACM National Conference*, pages 517 – 524, 1968.

[11] J. Stookey, Z. Xie, B. Cutler, W. R. Franklin, D. Tracy, and M. V. Andrade. Parallel ODETLAP for terrain compression and reconstruction. In *16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 2008.

[12] The Mathworks, Inc. Matlab - mldivide, mrdivide. `http://www.mathworks.com/access/helpdesk/help/techdoc/ref/mldivide.html`, (retrieved 6/15/2009), June 2009.

[13] A. H. Thiessen. Precipitation averages for large areas. *Monthly Weather Review*, 39(7):1082–1084, 1911.

[14] W. Tobler. A computer movie simulating urban growth in the detroit region. *Economic Geography*, 46(2):234–240, 1970.

[15] D. Tracy. *Path planning and slope representation on compressed terrain*. PhD in Computer Science, Rensselaer Polytechnic Institute, 2009.

[16] S. D. Warren, M. G. Hohmann, K. Auerswald, and H. Mitasova. An evaluation of methods to determine slope using digital elevation data. *CATENA*, 58(3):215 – 233, 2004.

[17] A. E. Wren. Trend surface analysis - a review. *Canadian Journal of Exploration Geophysics*, 9(1):39–44, 1973.

[18] Z. Xie, W. R. Franklin, B. Cutler, M. A. Andrade, M. Inanc, and D. M. Tracy. Surface compression using over-determined laplacian approximation. In *Proc. of SPIE Vol. 6697 Advanced Signal Processing Algorithms, Architectures, and Implementations XVII*, San Diego CA, August 2007. paper 6697-15.

[19] L. W. Zevenbergen and C. R. Thorne. Quantitative analysis of land surface topography. *Earth Surface Processes and Landforms*, 12:47–56, 1987.