Surface Compression using Over-determined Laplacian Approximation

Zhongyi Xie[1] , W. Randolph Franklin[2], Barbara Cutler[1],
Marcus A. Andrade[2,3], Metin Inanc[1] and Daniel M. Tracy[1]

[1]Department of Computer Science, Rensselaer Polytechnic Institute,
Troy, New York, 12180–3590, USA;
[2] Department of Electrical, Computer, and System Engineering,
Rensselaer Polytechnic Institute, Troy, New York, 12180–3590, USA
[3] Universidade Federal de Viçosa, Viçosa, Brazil

ABSTRACT

We describe a surface compression technique to lossily compress elevation datasets. Our approach first approximates the uncompressed terrain using an over-determined system of linear equations based on the Laplacian partial differential equation. Then the approximation is refined with respect to the uncompressed terrain using an error metric. These two steps work alternately until we find an approximation that is good enough. We then further compress the result to achieve a better overall compression ratio. We present experiments and measurements using different metrics and our method gives convincing results.

1. INTRODUCTION

Nowadays, the size of digital terrain data has grown to an extent that makes it essential to use some special representation or compression technique to manipulate the data. For example, LIDAR (Light Detection and Range), which is a laser based range detector coupled with a GPS sensor, allows for 20,000 to 50,000 readings per second. Each reading is stored as an $\{x, y, z\}$ triplet where each coordinate is represented as an IEEE double precision floating point number amounting for 24 bytes per point. LIDAR was the technology used for the state of North Carolina after Hurricane Floyd (1999) to map the whole state in the NC Floodplain Mapping Project. Just the Neuse river basin (11% of the whole NC area) is made of approximately 500 million points and takes 11.2 GB to store.[1, 2]

However, the development of processing and handling of digital terrain data has not advanced in pace with the data inflation. Elevation datasets are still stored as an elevation matrix where common compression algorithms do not perform very well, and there are not many algorithms specifically designed to compress terrain data. For example, USGS DEM data are usually compressed with gzip, which was originally designed as a plain text compressor.[1]

In this paper, we use Over-determined Laplacian Partial Differential Equations (ODETLAP) to approximate and lossily compress terrains. By terrain we mean a single-valued elevation matrix, which excludes caves or overhangs. We construct an over-determined system using triangulation, visibility tests, level set components and random selection; then use an over-determined PDE to solve for a smooth approximation. The initial approximation might be very rough, i.e., it may contain considerable elevation or slope errors. After that, we refine the approximation with respect to the original terrain by adding into the important points set those points with biggest elevation error or slope error, and then use ODETLAP again on the augmented representation

Further author information:
Zhongyi Xie: xiez@rpi.edu
W. Randolph Franklin: frankwr@rpi.edu
Barbara Cutler: cutler@cs.rpi.edu
Marcus A. Andrade: marcus@ecse.rpi.edu
Metin Inanc: inancm@rpi.edu
Daniel M. Tracy: tracyd@rpi.edu

to find a better approximation. These two steps are alternately applied until a predefined maximum error is reached. We also study the size versus accuracy tradeoff and plot the error curve.

ODETLAP can process not only continuous contour lines but isolated points as well. The surface produced tends to be smoother while preserving high accuracy to the known points. Local maxima are also well preserved. Alternative methods generally sub-sample contours due to limited processing capacity, or ignore isolated points.

This paper is organized as follows. In section 2 we briefly walk through current terrain representation and compression techniques. In section 3.1, we describe in detail the definition of ODETLAP, followed by section 3.2 which covers the outline of the algorithm that compresses the digital terrain data. In section 4, we describe several different ways to select points that will be used by ODETLAP to reconstruct the terrain. Section 5 presents details on enconding the points to maximize the compression ratio. Finally in section 6 and section 7 we talk about results and ideas for future work.

2. RELATED WORK

Simplification and compression of three dimensional terrain/surface data share the same goal but take different routes: simplification reduces the complexity of the terrain by reducing the number of vertices and faces in the mesh while compression works to compactly represent the connectivity data of the terrain. We will briefly discuss some classical work in both areas.

2.1 Progressive Meshes

The **Progressive meshes(PM)** is a surface representation introduced by Hoppe.[3] The simplification is based on a pair of reversible operations: edge collapse and vertex split, the first of which works for simplification and the latter for reconstruction/refinement. A PM maintains a sequence of refinement/simplification records so that a mesh of any precision can be obtained by incremental refinement.

2.2 Triangulated Irregular Network

The **Triangulated Irregular Network (TIN)**, a piecewise linear triangular spline, first implemented in cartography in 1973[4] is an approximated, lossy representation that has the major advantage that it is not tied either to a particular coordinate system, or to a developable surface. We use an implementation which is greedy: find the point with biggest error and use it to refine the triangulation. For more information, please see section 4.1.2.

2.3 Mesh Compression

Taubin and Rossignac propose a mesh compression scheme which records the connectivity information into a spanning tree.[5] Their method is capable of compressing connectivity information to 2 bits per triangle. Normals, colors, and texture coordinates, are compressed in a similar manner. One of the disadvantages of this method is its large memory requirement due to the requirement of random access to all vertices in decompression.

2.4 Hierarchical Triangulation

Hierarchical Triangulation (HT) is a hierarchical triangle-based model for representing surfaces over sampled data proposed by De Floriani.[6] Similar to TIN, HT is based on subdividing the surface into nested triangulations which is then organized into a tree where each node stands for a triangulation except the root. In order to describe the surface of each triangles, HT associates values with vertices of the triangles. HT is capable of extracting the representation of terrain at variable resolutions over the domain.

2.5 Visibility Preserving Terrain Simplification

Ben-Moshe proposes a terrain simplification technique based on preserving inter-point visibility relationships.[7] This technique aims at preserving visibility information, which, informally speaking, means points in the original terrain that are visible to each other (and respectively, points not visible to each other) are still visible to each other (respectively, not visible to each other) after the simplification. This technique is designed to meet the need of finding good locations on the terrain to place "observers" (antennas, guards, etc). It works by first computing the ridge network (a collection of chains of edges of the terrain). This ridge network induces a subdivision of the terrain into patches and each patch is independently simplified using one of the standard terrain simplification methods.

3. OVER-DETERMINED LAPLACIAN APPROXIMATION

3.1 Definition

As implied by the name, the Over-determined Laplacian Approximation (ODETLAP) comes from Laplace's equation, whose solution at any point (x, y, z) is equal to the average of the solution values on the surface of any sphere with center (x, y, z), assuming the equation holds throughout the sphere.[8] We have the equation

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

for every unknown non-border point, which is equivalent to saying the surface satisfies Laplacian PDE,

$$\frac{\partial^2 z}{\partial x2} + \frac{\partial^2 z}{\partial y2} = 0 \tag{2}$$

In terrain modeling this equation has the following limitations:

- The solution of Laplace's equation never has a relative maximum or minimum in the interior of the solution domain, this is called the "maximum principle";[8] so local maxima are never generated.
- The generated surface may droop if a set of nested contours is interpolated[9]

To avoid these limitations, an over-determined version of the Laplacian equation is defined as follows: apply the equation (1) to every non-border point, both known and unknown, and a new equation is added for a set S of known points:

$$z_{ij} = h_{ij} \tag{3}$$

where h_{ij} stands for the known elevations of points in S and z_{ij} is the "computed" elevation for every point, like in equation (1).

Note that the system of linear equations is over-determined, i.e., the number of equations exceeds the number of unknown variables. Since the system is very likely to be inconsistent, instead of solving it for an exact solution (which is now impossible), an approximated solution is obtained by trying to keep the error as small as possible. Equation (1) is satisfied for each point, making it the average of its neighbors, which makes the generated surface smooth there. However, since we have known points where equation (3) is valid, they are not necessarily equal to the average of their neighbors, which probably make the surface not smooth there. This is especially true when we have adjacent known points, like points that defines contour lines. Therefore, for points with multiple equations we can choose the relative importance of accuracy versus smoothness by adding a smoothness parameter when solving the over-determined system.[10] In our implementation, equation (1) is weighted by R relative to equation (3) which defines the known locations. So a very small R will approximate a determined solution and the surface will be more accurate while a very large R will produce a surface with no divergence, effectively ignoring the known points. figure 2 shows how different values of R will affect the generated surface.[4] Subfigure (a) shows the four nested square contour lines that we try to approximate. Subfigure (b) gives the Lagrangian interpolation and we can see the undesired lines that surface normal is not continuous. Subfigure (c) and (d) are generated by ODETLAP and with R equal to 1 and 10. So in (c) where we made the accuracy as important as smoothness, the surface is quite accurate compared to in subfigure (d). However, the visible contours means it's not as smooth as subfigure (d).

This over-determined system allows for processing of isolated, scattered elevation points as well as continuous contour lines and produces a smooth surface while the error is minimized. The generated surface has local maxima inside the innermost contour and shows little or no evidence of the contours. Instead of interpolation, approximation is a more suitable term for this method because the reconstructed surface is not guaranteed to go through the input data points.

ODETLAP can be used as a lossy compression technique since the original terrain can be approximated with some error using the set of points S for equations (1) and (3).

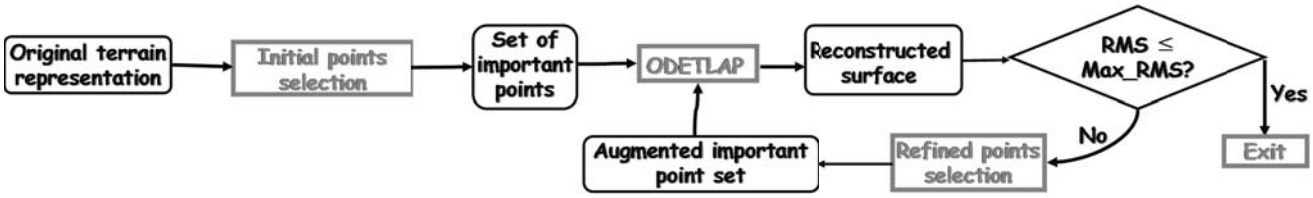

Figure 1. Algorithm Outline

3.2 Algorithm Outline

The ODETLAP algorithm's outline is shown in figure 1 and the pseudo code is given below. Starting with the original terrain elevation matrix there are two point selection phases: firstly, the initial point set S is built by any of the methods described in section 4.1 and a first approximation is computed using the equations (1) and (3). Given the reconstructed surface, a stopping condition based on an error measure is tested. In practice, we have used the root-mean-square (RMS) error as the stopping condition. 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 method described in 4.2 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 total error converges.

input *OriginalTerrain*: T
output *PointSet*: S
[0] $S = InitSelection(T)$
[1] $Reconstructed = ODETLAP(S)$
[2] while $RMS(Reconstructed) > Max\_RMS$
[3] $\quad S = S \cup Refine(T, Reconstructed)$
[4] $\quad Reconstructed = ODETLAP(S)$
[5] **return** S

4. POINT SELECTION STRATEGIES

As we have seen in section 3.2, there are two stages where points are selected: the initial point selection stage and the refined point selection. We discuss each of them below.

4.1 Initial Points Selection

4.1.1 Random Selection

This strategy is the most intuitive and easiest to implement. The basic idea is select points randomly. Using a good random number generator, this strategy ensures that most parts of the terrain contribute to the final reconstruction. This strategy is fast and robust.

4.1.2 Triangulated Irregular Network

The next strategy is based on the Franklin's algorithm[4] which builds a triangulated irregular network (TIN) using a greedy insertion method to approximate a surface. Starting with a matrix of elevations, it first splits the points bounding square (or rectangle) into two triangles along the diagonal and associates each triangle with its points. In the next step, it searches within each triangle for the furthermost point from the triangle's plane and this point is used to split the triangle into three new triangles (or two new triangles if the furthest point happens to be on an edge of the triangle). It uses a breadth first search to avoid starvation: a triangle is never split if there exists an undivided triangle that was created before it. An issue with this approach is that in some steps the insertion of the furthermost point may temporarily increase the error, but usually, after some additional insertions, the error will be reduced even more. So, the overall tendency is for the error to decrease when new points are added.

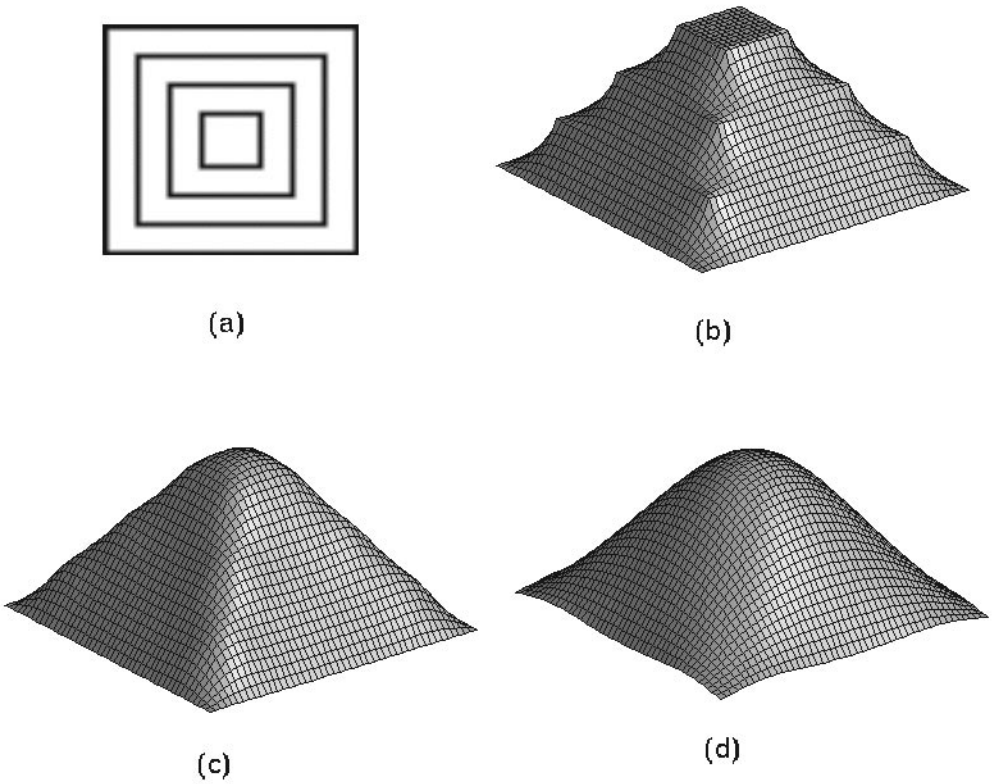

Figure 2. Impact of R in ODETLAP interpolation:[10] (a): The Square Contours to be Interpolated. (b): Lagrangian Interpolation. (c): Overdetermined Solution, $R = 1$. (d): Overdetermined Solution, $R = 10$.

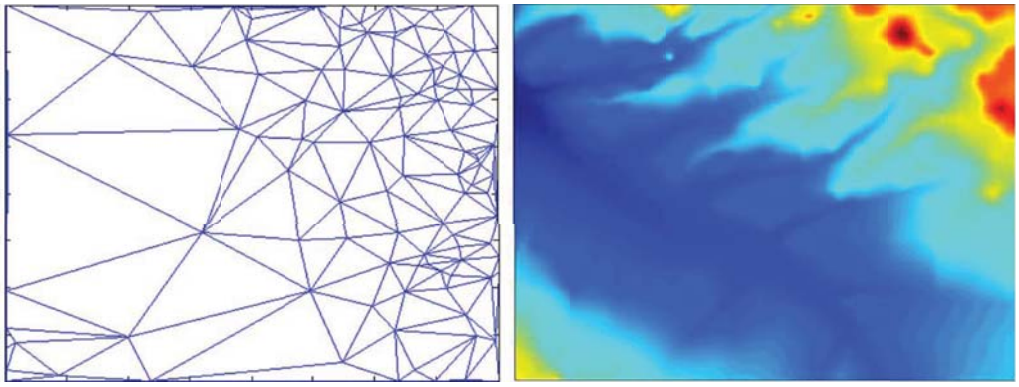

Figure 3. Triangulated Irregular Network

4.1.3 Visibility Index

The visibility index (VIX)[11] of a point p, the "observer", on the terrain is defined as the number of terrain points that p can see. An approximated version of this index is defined considering only a small region around p; generally, this region is given by a circle centered at p with a radius (of interest) r.

There are several ways to compute the VIX values and we use the one proposed by Ray and Franklin:[11] for each terrain point p, randomly select k sample points within the radius of interest and run a line of sight connecting p to each random point to decide if the point is visible or not. The VIX of p is given by the ratio

between the number of visible random points and sample size. As a result, the VIX values are only approximation to the exact values and they are highly dependent on how many random points are chosen. In our tests, we used $r = 25$ and $k = 10$.

In this point selection strategy we assume that points with small VIX values are more important to define the terrain skeleton than points with big VIX. So the initial set is built containing points with small VIX values. However, our selection of points need to reflect the overall VIX value distribution. This is done using a probability distribution that establishes how likely a point with a determined VIX value will be selected. This probability distribution is defined using the VIX value (small values mean bigger probabilities) weighted by the VIX values distribution. Thus, points whose VIX value occurs more frequently have their probability multiplied by a higher factor. Figure 4 shows the selected points from the terrain.

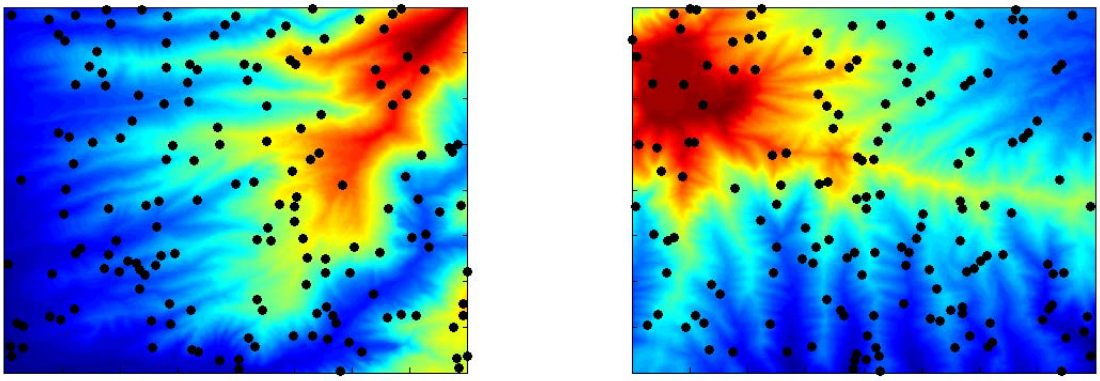

Figure 4. Visibility Index: Points are selected according to their visibility indices

4.1.4 Level set components

Using an adaptation of level set ideas,[12] we segment the terrain based on points' elevation. That is, suppose that the elevation values range from h_{min} to h_{max} and given an integer k, the interval $[h_{min} \cdots h_{max}]$ is divided into "elevation slices" of equal size k (the last slice can be smaller). Then, each terrain point $p = (i, j, h)$ is associated to the corresponding elevation slice that contains the height h; more precisely, to the slice $[h_i \cdots h_{i+1}]$ such that $h_i \leq h < h_{i+1}$. Next, each elevation slice is partitioned into connected components which are computed using 8-connectivity, (i.e., the horizontal, vertical and diagonal neighbors are checked)*

As in the previous strategy, this point selection criterion uses a probability distribution defined considering the elevation slices area (i.e., the total number of points in all the connected components in the elevation slice) weighted by an "elevation slice importance" assigned assuming that the most important slices are those in the extremities (lowest and highest) height - the importance decreases uniformly toward the slice with medium height.

4.2 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 1, 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 (similar to 4.1.2): we find a set of points with 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 S'

*Of course, the elevation slice point association and the connected component computation can be done simultaneously just adapting the connected component computation to check if a neighbor is in the same elevation slice.

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:

1. Relative RMS: Compute the root mean square (RMS) error of the approximation and check if its ratio against the RMS error of the first approximation is smaller than a predefined threshold.

   or

2. Absolute RMS: Define a value for the maximum acceptable RMS error.

4.3 Forbidden Zone

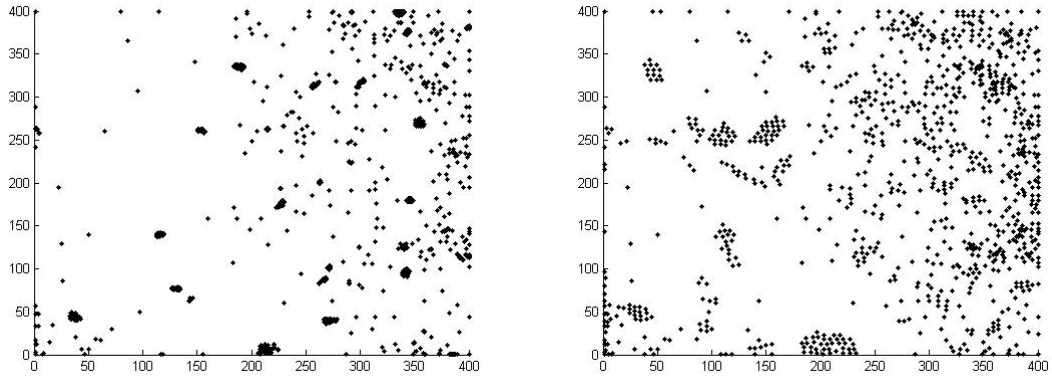

Figure 5. Forbidden Zone: Points are often clustered as in the left figure, hence reconstructed surface is not very accurate; We apply forbidden zone in refined points selection so that points are no longer clustered as in the right figure and reconstructed surface is more accurate as we can see in table 1.

Using the refined point selection described in section 4.2, one can encounter a problem: the refined points are sometimes clustered (left figure in figure 5). This is because real terrains are mostly continuous so if one point is far away, adjacent points are also likely to be erroneous, and will be selected as well. Because of this, refined points selected by any of our strategies may be redundant in some regions, which is a waste of storage.

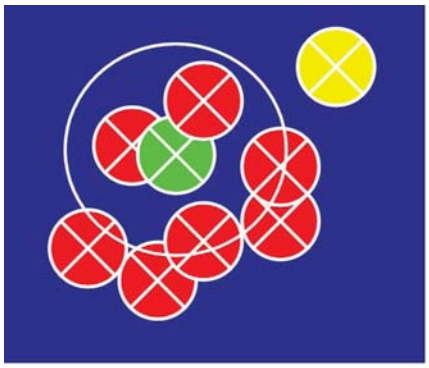

Figure 6. Forbidden Zone

We perform a check process when adding new refined points: the local neighbor of the new point is checked to see if there is any existing refined points which were added in the same iteration. If yes, this new refined point is discarded and point with the next biggest error is tested until we find desired number of refined points. So as shown in figure 6, all potential refined points that are close to an existing refined point(green points) are useless(marked red), and only points that are beyond some distance from green points are selected(marked yellow). The effect of forbidden zone can be seen in the right figure in figure 5: no dense clusters of points are present and all points are distributed more evenly within the whole terrain.

Data	Avg Error(w/F.Z.)	Avg Error(w/o F.Z.)	Max Error(w/F.Z.)	Max Error(w/o F.Z.)
Hill1	3.16	7.35	19.9	34.3
Hill2	8.34	14.8	48.9	71.6
Hill3	1.54	3.00	10.3	13.5

Table 1. Impact of forbidden zone: Size of forbidden zone = 5.

In table 1, we show how the forbidden zone affects the average error and maximum error in our experiment. We implemented our algorithm with and without the forbidden zone and tested it with three data sets. In this test, we begin with 300 points and in each iteration 50 refined points are added. The results after 14 iterations justify the use of the forbidden zone in our algorithm.

5. FURTHER COMPRESSION - COMPRESSING REFINED POINTS

In order to achieve better compression, we apply data compression to the final point set S, an $n \times 3$ matrix where each row represents a point, and the three columns represent x, y and z. Our approach is to separate first two columns $\{x, y\}$ from the last column $\{z\}$, because x and y are integer coordinates (matrix rows and column indices) while z is the elevation value. Unlike x and y which are uniformly distributed over [1, N], where N^2 is the size of the dataset, the elevation values tend to have a smaller deviation. This makes possible a more compact compression if a prediction scheme is used. To preserve the x, y, z correlation, we either sort the $n \times 3$ matrix on $\{x$ and $y\}$ or $\{z\}$ prior to splitting it into a $n \times 2$ matrix and a $n \times 1$ matrix. As expected, different sorting orders give different compression sizes.

In order to achieve higher compression, we use linear prediction and delta coding, which predicts the next value using the previous one. To illustrate, suppose we have a vector $(954, 1021, 1001, 897, 958, 1130)$. The predicted vector would be $(0, 954, 1021, 1001, 897, 958)$, But we only need to store Δ, the difference between the predicted and actual values: $(954, 67, -20, -104, 61, 172)$. Compressing the difference using a data compressor like bzip2 further reduces the size of the compressed data.

To illustrate the effect of sorting and linear prediction, we have tested the following five schemes, and their compressed sizes are given in figure 7

1. Sort rows on $\{x \textit{ and } y\}$, do not use linear prediction.
2. Sort rows on $\{x \textit{ and } y\}$, use linear prediction only in $\{z\}$.
3. Sort rows on $\{x \textit{ and } y\}$, use linear prediction in both $\{x, y\}$ and $\{z\}$.
4. Sort rows on $\{z\}$, use linear prediction only in $\{z\}$.
5. Sort rows on $\{z\}$, use linear prediction in both $\{x, y\}$ and $\{z\}$.

From the figure, we can see that techniques 3, 4 and 5 give smaller compressed sizes. Comparing bzip2 on the original data set to the best technique, we see a 33% improvement.

6. RESULT AND ANALYSIS

We have tested our algorithm on six different data sets. The results are presented in Table 2 showing that our new compression scheme is doing well in lossily compressing the terrain. We can successfully compress them from binary size of 320KB to less than 3KB, while keeping the mean absolute error under 2% in all cases.

Figure 7. Second Pass Compression Summary: Compress a set of 1000 points using the five schemes described in section 5.

Data	Elev. Range	RMS	Mean	Max	XY Size	Z Size	Total Size	Compression Ratio
Hill1	505.5	3.62	2.82	16.97	1446	1304	2750	116.4
Hill2	745	9.45	7.32	40.52	1472	1354	2826	113.2
Hill3	500	1.72	1.35	10.41	1400	1209	2609	122.7
Mtn1	1040	17.34	13.47	76.80	1493	1456	2949	108.5
Mtn2	953	17.17	13.48	65.22	1476	1424	2900	110.3
Mtn3	788	17.06	13.36	87.52	1462	1503	2965	120.8

Table 2. ODETLAP Compression results: Each of the ODETLAP tests consist of 100 initial points selected with the TIN method, and then 10 points are added using the greedy selection method on each iteration for 90 iterations, for a total of 1000 points. Forbidden zone is used and we chose R = 0.01 in all cases

6.1 Impact of points selection strategies

The approximation error obtained in the method's first step varies a lot depending on how the initial points are selected. We have tested all strategies mentioned in section 4.1, to determine which strategy works best, i.e., discovers the most important points that define the terrain profile such that the approximation error would be lowest.

We have tested the four initial points selection methods described in section 4.1. In the experiments, we used different number of initial points for all the methods and recorded the RMS, average absolute and maximum absolute errors of the reconstructed surface. As we can see in the figure 8, the four methods's performance is quite similar in terms of RMS errors and average errors, while TIN is always slightly better when number of points is greater than 300. In table 3, we also have the RMS errors of the four methods, and this corresponds to the first subgraph in figure 8. We notice that TIN is generating the most accurate surface with approximately 20% better than any of other methods. Surprisingly, random is also a competitive method: it's error is not too much larger than any other method. Considering its efficiency, robustness and simplicity, it is also a good choice besides TIN.

6.2 Error vs. Size

As we have seen in previous discussions, our ODETLAP based algorithm refines the approximated surface as more points are added. In order to figure out how the error decreases as number of refined points increases, we experitment by starting with 10 initial points selected by TIN method and adding 1 refined point each time so as to get a complete error/size curve; Please refer to figure 9 for more information.

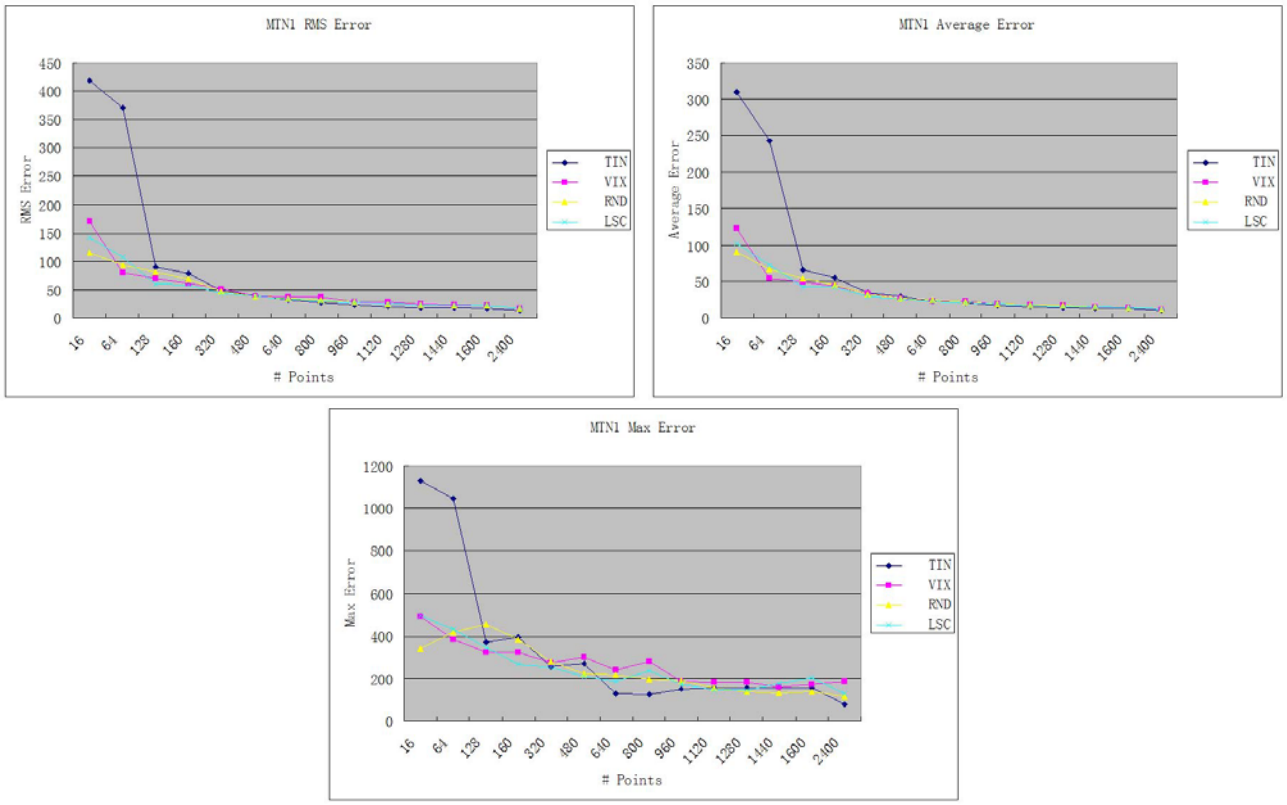

Figure 8. Comparison of initial points selection methods using different number of initial points: RND - random selection strategy, TIN - Triangulated irregular network, VIX - visibility index, LSC - the level set components.

# of Points	TIN	VIX	RND	LSC
16	419.1	171.7	114.0	141.5
64	369.5	79.15	92.72	108.3
128	89.08	69.81	82.17	58.98
160	78.18	61.72	67.42	57.63
320	47.37	51.13	46.54	44.00
480	39.50	38.92	39.00	37.50
640	31.02	36.72	34.38	33.07
800	27.06	36.21	30.75	30.41
960	22.70	28.43	28.03	27.88
1120	20.54	27.43	25.40	24.63
1280	18.86	24.44	23.65	23.33
1440	18.26	23.18	22.34	22.02
1600	17.21	21.88	21.11	21.67
2400	13.23	17.43	17.03	16.52

Table 3. Comparison initial points selection methods:

7. CONCLUSION AND FUTURE WORK

We create a new terrain compression technique based on an Overdetermined Laplacian equation that computes good compression results. We use several points selection methods as to find most important points. In order to get better results, we also use forbidden zone in selecting refined points as well as delta encoding in further compression and both of them work well. Moreover, our compression technique does not only work well in conventional error metrics as we have seen in table 2, but some other application specific error metrics as well.

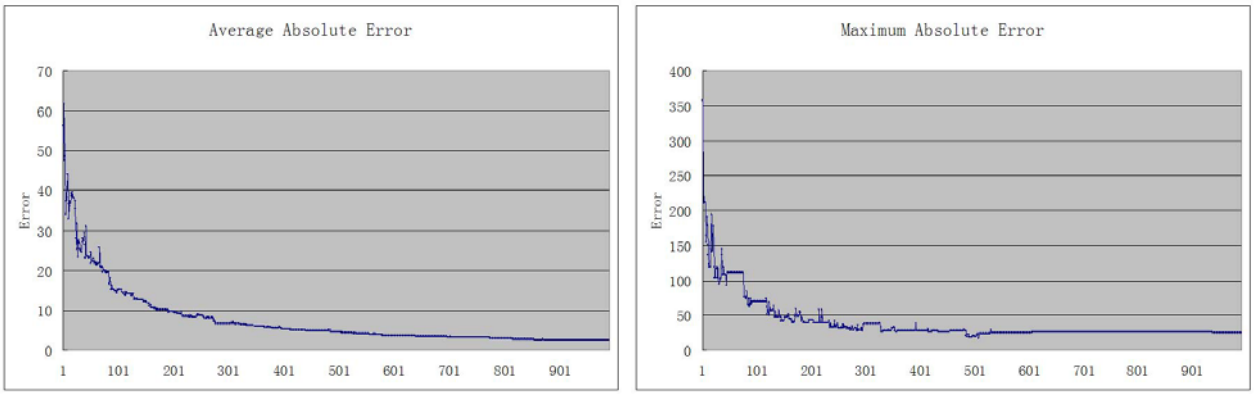

Figure 9. Both average absolute error and maximum absolute error decrease as more points are added: y axis stands for errors and x axis stands for total number of points

For example, as described in,[13] more complex metrics such as visibility and path planning are employed to evaluate the reconstructed approximation from ODETLAP and the results are also very good.

The next step of research consists of a few extensions in two directions: one is higher accuracy. We will investigate other PDEs to see if they can reconstruct the terrain more accurately than the Laplacian PDE. Another direction is higher compression. Currently we use lossless compression in the final compression step. We will test the use of lossy schemes, which can reach higher compression ratios at the cost of loss in accuracy. Since slope is also a very important feature of terrain, we will also consider ways to minimize slope error in our representation. For example, we will investigate selecting refined points based on largest slope error instead of elevation error or a combination of the two.

8. ACKNOWLEDGEMENTS

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. We thank Prof. Franklin T. Luk, Jonathan Muckell and Jared Stookey for valuable discussions on terrain representation.

REFERENCES

1. W. R. Franklin and M. Inanc, "Compressing terrain datasets using segmentation," *Advanced Signal Processing Algorithms, Architectures, and Implementations XVI* **6313**(1), p. 63130H, SPIE, 2006.
2. "North carolina floodplain mapping program, http://www.ncfloodmaps.com." last accessed in jul 2007.
3. H. Hoppe, "Progressive meshes," *Computer Graphics* **30**(Annual Conference Series), pp. 99–108, 1996.
4. W. R. Franklin, "Triangulated irregular network, ftp://ftp.cs.rpi.edu/pub/franklin/tin73.tar.gz.," 1973.
5. G. Taubin and J. Rossignac, "Geometric compression through topological surgery," *ACM Transactions on Graphics* **17**(2), pp. 84–115, 1998.
6. L. D. Floriani and E. Puppo, "Hierarchical triangulation for multiresolution surface description," *ACM Transactions on Graphics* **14**(4), pp. 363–411, ACM Press, (New York, NY, USA), 1995.
7. B. Ben-Moshe, J. S. B. Mitchell, M. J. Katz, and Y. Nir, "Visibility preserving terrain simplification: an experimental study," in *SCG '02: Proceedings of the eighteenth annual symposium on Computational geometry*, pp. 303–311, ACM Press, (New York, NY, USA), 2002.
8. G. Sewell, *The Numerical Solution of Ordinary and Partial Differential Equations, Second Edition*, 2005.
9. W. R. Franklin, M. Inanc, and Z. Xie, "Two novel surface representation techniques," in *Autocarto*, CAGIS, 2006.
10. W. R. Franklin, "Applications of geometry," in *Handbook of Discrete and Combinatorial Mathematics, CRC Press, 2000*, Rosen, Michaels, Gross, Grossman, and Shier, eds., 2000.
11. W. R. Franklin and C. K. Ray, "Higher isn't necessarily better: Visibility algorithms and experiments," in *Sixth International Symposium on Spatial Data Handling*, **2**, pp. 751–763, (Edinburgh, Scotland), 1994.

12. J. A. Sethian, *Level Set Methods and Fast Marching Methods Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*, Cambridge University Press, 1999.
13. D. M. Tracy, W. R. Franklin, B. Cutler, M. Andrade, F. T. Luk, M. Inanc, and Z. Xie, "Multiple observer siting and path planning on a compressed terrain," *Advanced Signal Processing Algorithms, Architectures, and Implementations XVI* **6697**, SPIE, 2007.