Two Novel Surface Representation Techniques

W. Randolph Franklin
(mail@wrfranklin.org, http://wrfranklin.org, +1-518-276-6077),
Metin Inanc (inancm@gmail.com)
Zhongyi Xie (xiez@rpi.edu)
ECSE Dept, Rensselaer Polytechnic Institute,
110 8th St, Troy, New York, 12180–3590, USA

Abstract

We present two new surface representation techniques: *scooping* and *overdetermined Laplacian PDE approximation*. Scooping reveals the terrain by removing material with a set of operators that resembling a 3-axis drill. Each operator has a square cross section, perhaps 7×7 posts, and a surface that is a polynomial of degree ranging from 0 to 3. Scoops may either partition the whole data cell, or else may may applied hierarchically and adaptively as needed to reduce errors. The longterm goal is for scoops to model geologic formation mechanisms such as water erosion.

The overdetermined PDE solves a overdetermined system of linear equations to produce a smooth surface approximation to a set of elevation posts. This representation has several advantages, such as the ability to infer local maxima inside nested rings of contours, and the ability to compute a best fit to an inconsistent set of inputs. The input data may be produced by an incremental Triangulated Irregular Network program, supplemented by an iterative insertion of the most inaccurately fitted points.

Both representations are part of the GeoStar project to lossily compress large terrain elevation matrices while preserving their usefulness for applications such as visibility and mobility.

1 Introduction

How shall terrain, meaning elevation above (or below) sea level, be represented in a computer? To simplify, we assume that elevation is single-valued; overhangs and caves are not considered here. However, we try to represent discontinuities, as they are perhaps the most important class of terrain feature. Discontinuities greatly affect visibility and mobility, and are easily recognizable.

The novel surface representation techniques presented here are intended to answer these questions: What terrain operators are appropriate, and how realistic they should be? There is a sweet spot: Fourier series are too unrealistic, but a complete geological evolution model is too complex.

2 Classical Terrain Representations

The following brief survey of three classical representations illustrates concerns and techniques that are relevant to our new representations, while presenting some new views.

2.1 Contour lines

Contour lines are perhaps best suited for hand sketching of a map. Since the actual terrain is not nearly as smooth as the contour line says, as typically realized on a paper map, they are quite lossy. This fact, combined with the elevation spacing between adjacent lines, means that important small features, such as gulleys and other minor elevation changes, may be unrepresentable. Therefore, determining visibility and mobility is much more error-prone. Also, representing very steep slopes and cliffs require distorting the contours.

Implementing contour lines on a computer reveals other disadvantages, arising from their essential nature as a hand-drawn technique. The first question is, how shall each line be represented? The obvious answer is as a polyline or sequence of points. If there are many points, then much space is required. If there are few points, then the spacing between lines is inaccurate, or the lines may even cross. The deep reason for this is the separate representation of each line, ignorant of their relationships.

One solution is to use level-set techniques, (Osher and Fedkiw, 2003), but that requires one of the techniques discussed later, so that the contours themselves become redundant.

An additional problem with contours is how to interpolate intermediate elevations, while avoiding problems like the following. When interpolating an elevation at some test point, the closest contour line in all 8 cardinal directions from the point may all be the same contour. This causes the test point, which lies between two contours, to have the elevation of one of the contours, which is probably wrong. Also, inferring a mountain top inside a set of nested contours is desirable.

(Gousie and Franklin, 2005), (Gousie and Franklin, 2003), presents a new method that starts by computing new intermediate contours in between existing isolines. These are found by finding the shortest line segment that connects points on two neighboring contours with differing elevations. The midpoint of the line segment becomes a point on the intermediate contour. The contours are completed by connecting individual points. The new contours are then used as data for successive iterations, until an initial surface is formed. Peaks are computed by Hermite splines that follow the slope trend. Gaussian smoothing is applied to the entire surface or only to newly computed elevations, yielding an approximated or interpolated surface, respectively.

2.2 Triangulated Irregular Network

The Triangulated Irregular Network (TIN), a piecewise linear triangular spline, first implemented in cartography in 1973 by (Franklin, 1973), is purely a technique for computers; no one would implement a TIN by hand. After reviewing the TIN, this section discusses coding techniques for the TIN points. "Coding" a data representation means to represent its coordinates, pointers, or whatever, as a sequence of bits. It is often overlooked that an efficient coding is as important as the representation itself, and that a representation's efficiency is not even a well formulated concept absent the coding.

One common implementation takes a set of data points $\{(x, y, z)\}$, and uses a greedy point insertion to refine an initial triangulation into one with more points and a smaller error. A subtle issue is that inserting a point into the triangulation often increases the maximum error. However, in this case inserting the next point usually reduces the error considerably. The stopping criterion for this process can be the attainment of a desired maximum surface error, of the insertion of the desired number of points.

An alternate construction technique proceeds by constructing a complete triangulation of the given data and then incrementally removing points. To our knowledge, these two techniques have not been combined, although that would seem to reduce the error attained with a given number of TIN points.

In either case, edges in the triangulation are flipped when necessary to main some property such as Delauney. It is not *a priori* obvious that that should be a more desirable property than minimizing the total edge length or elevation error. However, our experiments find that Delauney triangles do work better.

One misconception is that the unconstrained TIN does not adequately capture surface features like ridge lines, which must instead be explicitly inserted into a constrained triangulation. However, our experiments, on synthetic data constructed with both sharp and gradual, straight and curved, ridges, and even discontinuities as would occur in a road cut, found no such problem. Perhaps this was a problem with some other early implementations.

Another application of a TIN is a transform an irregular set of points into a regular grid, (Speckmann and Snoeyink, 1997). Here, the points are completely triangulated, and then grid points are interpolated inside the triangles. This may be performed on very large datasets by sweeping up the triangulation.

Implementing a TIN requires choosing an appropriate planar graph data structure. The obvious answer, available in geometry packages, represents every topological dimension (point, edge, face), and all their adjacency relations explicitly. The first problem is the storage cost required by all these pointers, which can be an order of magnitude more than the elevations themselves. The second problem is that keeping all this redundant information consistent as points are inserted and edges flipped is tedious.

An advantage of requiring that the triangulation be Delauney is that storing any topology at all is unnecessary, as it may be recomputed when needed. This is the endpoint of a sequence of time–space tradeoffs. Succinct planar graph data structures, (Turan, 1984), which can store the topology in a few bits per element, form an intermediate point in that tradeoff.

The TIN has the advantage of facilitating a *progressive transmission* of the surface over a slow communication link. That is, suppose that we wish to transmit the surface from server $\mathcal{S}$ to client $\mathcal{C}$. $\mathcal{C}$ may not even be certain that s/he wants the whole terrain until seeing a preview. Suppose that

$\mathcal{S}$ computes a TIN using the insertion method, and then transmits the points one-by-one in their insertion order. $\mathcal{C}$ rebuilds the TIN by inserting the points as received. If the approximate surface appears unsuitable, then $\mathcal{C}$ tells $\mathcal{S}$ to stop.

Once the TIN has been computed, the next step is to code it, or to represent it in as few bytes as possible. This emulates other good data compression techniques, such as JPEG and BZIP, whose space efficiency results from their design as a pipeline of compression techniques, with the output from one step being the input to the next. For example, BZIP2 text compression contains the following five steps in sequence: • Run length encoding • Burrows-Wheeler transformation • Move to front • Another run length encoding • Arithmetic encode. JPEG image compression performs these steps in sequence: • Rotate RGB to YCrCb • Discrete cosine transform • Low-pass filter • Arithmetic encode.

To code the TIN points, we are considering various methods, but currently like the following technique.

1. Start with the set of points, $\{(x, y, z)\}$, in the triangulation.

2. Compress the horizontal points, $S_h = \{(x, y)\}$, separately, using one of the current bitmap compression techniques designed for facsimile transmission, (Salomon, 2000). A technique's efficiency may be evaluated by comparing the size of its output to the information theoretic bound, obtained as follows.

   Let M be the number of original potential points in the terrain. E.g., for a level-1 DEM, $M = 1201^2$.

   Let N be the number of points in the triangulation.

   Then, b, the information content, in bits, or entropy, of S_h is $b = \lg \binom{M}{N} = \lg \frac{M!}{N!(M-N!}$ where $\lg$ means the logarithm in base 2 and $\binom{M}{N}$ is the number of combinations from M elements taken N at a time. b may be approximated as follows. Let $p = N/M$ and $q = 1 - p$. Then $b \approx M(-p \lg p - q \lg q)$

   E.g., if $N = 10^5$, then $p = 0.07, q = 0.93$ and so $b \approx 1.4 \cdot 10^5$ bits, or $18\,000$ bytes. This is much less space than merely listing the (x, y) coordinates.

   One obvious potential optimization is to reduce M, which is similar to reducing the number of significant digits in x and y. The simplest realization is to subsample the input data before TINning it. However, reducing several points to one, say by averaging, loses perhaps too much information. A more sophisticated technique might proceed by first TINning the original data set, and then perturbing the selected TIN points so that they fall on a coarser grid. This has the advantage that the increased error is easily computable.

   The definition of information content used here does assume that there is no structure to the points, that is, that there are no other usable relations between them. That is not quite true. In mountainous regions, the points will be close and in flat regions, widely spaced. However, it's not clear either how much information content there is in this fact, nor how to exploit it.

3. Now the elevations need to be compressed. The order of the z is important since each z must be associated with the correct (x, y). Without loss of generality, we can assume that the (x, y) are lexicographically sorted. Then using a delta encoding for the z is reasonable, assuming

that the consecutive elevations' values are close. This property would become more true if we used a space filling curve instead of a mere lexicographic order for the (x, y). That is not totally trivial for unevenly spaced points, but all that is necessary is that the ordering be unambiguously determinable from the set of points.

Various TIN extensions have been considered, such as using a higher degree triangular spline. That raises two issues. First, this idea's effectiveness requires that the terrain generally have a higher degree continuity. More precisely, this requires that the terrain data being used possess this property. That distinction is relevant because terrain data is often artifically smooth. Second, there are technical difficulties with using higher degree triangular splines, compared to using Cartesian product splines.

There is an easy (but not as good) way and a hard (but better) way to use a higher degree triangular spline. The easy way goes as follows.

1. Compute a traditional TIN.

2. Fit a higher degree spline to this triangulation, while requiring the appropriate degree of continuity across each edge.

This method will work to the extent that the terrain data possesses the appropriate degree of continuity.

The hard but better way is to incrementally build up a higher degree spline. Note that determining the optimal spline points of any order is an exponential problem, so that some heuristic is necessary.

An advantage of TINs is that their resolution adapts to terrain regions of varying complexity. They are also not wedded to any particular coordinate system. However, any algorithm using them, such as visibility determination is complicated by the necessity of traversing the triangulation.

2.3 Gridded Elevation Matrices

If the TIN is too complicated, then a simple matrix, or array, of elevations is a reasonable alternative. The objection that this is not appropriate since the earth is not a developable surface, that is, cannot be flattened, is answered with the *Riemannian manifold*, (Jost, 2002). This is a geometric construct that represents a space of some dimension as an overlapping set of *charts*, each valid only in some specific limited region. The charts are organized in an *atlas*. The major application lies in modeling curved space-time.

An invalid objection to the matrix of elevations originates in the variable nature of terrain. Since a goal is to represent the terrain as compactly as possible, the matrix must be compressed, and good compression techniques adapt locally to the local information content of their input data.

Since so much effort has gone into image compression techniques, such as JPEG and SPIHT, it is worth trying them for terrain. SPIHT, (Said and Pearlman, 1993) lossily compresses elevation matrices quite well, (Franklin and Said, 1996).

In the following sections we propose other ways to compress elevation matrices.

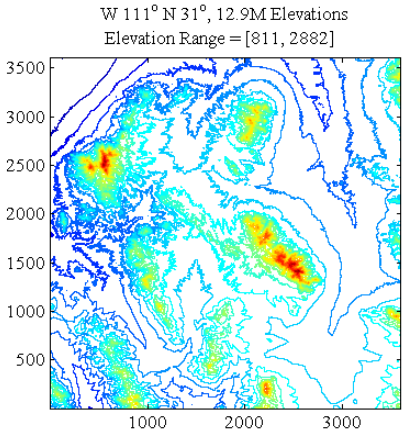

Figure 1: W111N31 level-2 DTED sample dataset

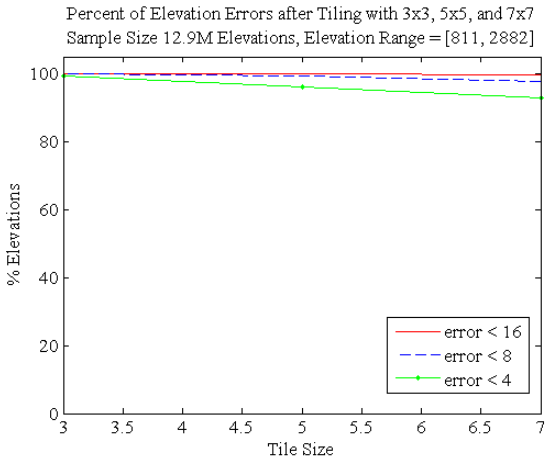

Figure 2: Percent of elevation errors after tiling with regular tiles.

3 New Terrain Representations

3.1 Scooping Operators

Scooping is an attempt to blast through the information theoretic limit for terrain compression by identifying and exploiting additional structure in the terrain. Such structure might originate from the geological formation processes.

Initially, we consider three different scoop operators to realize terrain scooping. The first one is flat and horizontal, the second one is flat and tilted and the third one is a quadratic equation. Those operators are used to scoop the terrain using either regular constant size tiles, or hierarchical quad-tree-like recursive tiles. Given the regular tiles' limitations, such as the inability exactly to follow nonsquare features, the quality of their approximations is surprisingly good. We are already planning a future production system with a richer set of operators.

3.1.1 Horizontal (Degree 0) Scoops

The first operator is flat and horizontal. Given a tile of the terrain, this operator can approximate the terrain in three different ways. The first is way is to approach the tile from above and set the estimate to the maximum elevation in the tile. The second approximation is from below, thus we set the estimate to the minimum elevation in the tile. The third way is to set the estimate to the mean of all of the elevations in the tile. While both the overestimate and the underestimate of the terrain diverge from the mean with larger tile sizes, they also provide a convenient envelope, which may be of value to certain terrain applications. The range in the scoop given by the difference of the estimates is a local metric of terrain variance. All of these estimates are in effect bringing the terrain resolution down. The level of lowering resolution is controlled by the scoop size. They are convenient since they consist of a single parameter c, but are very simplistic as a model. The data model is $z = c$.

	Planar	Quadratic	Cubic
Degree of scoop surface	1	2	3
Number of coefficients needed to represent one scoop	3	6	10
Number of scoops	301,859	159,626	95,099
Maximum absolute error	31	15	9
Mean absolute error	3.93	3.62	3.99
Number of scoops with error larger than 10	149	2	0

Table 1: Hierarchical scooping experiments on W111N31

3.1.2 Planar (Degree 1) Scoops

The second terrain scooping operator is still flat as the first one but it is no longer required to be horizontal. In effect we introduce a second parameter, which is the normal vector, producing $z = ax+by+c$. We fit a plane to the tile can be done by finding the regression plane that minimizes RMS vertical error. Our test data included DTED Level 2 files, containing 3601×3601 elevations, such as the W111N31 cell shown in Figure 1. Experiments show that errors are rare and large errors are even rarer. Those occur on ridges and valley bottoms, where the planar model is not enough to capture data variance. Results from regularly tiled scoops of sizes: 3×3, 5×5 and 7×7 are shown in Figure 2. For instance, when using 7×7 scoops, over 90% of points had an absolute error under 4, or 0.05% of the elevation range.

3.2 Quadratic (Degree 2) Scoops

The third scooping operator is based on the quadratic equation $z = ax^2 + by^2 + cxy + dx + ey + f$. Our hope is that the reduced number of required scoops will more than offset the doubled number of coefficients per scoop, especially when used in conjuction with the irregular size scooping algorithm described in the next section.

3.3 Hierarchical Scoops

This refinement is a recursive quadtree-like extension of any of the above methods, by varying the scoop size. Initially, the whole data cell is approximated with one scoop of the desired degree. If the maximum absolute error is larger than a threshhold, which is *10* in this case, then we subdivide the cell into four subcells and repeat. This process stops at 3×3 cells, which are represented by listing their 9 elevations.

Table 1 gives some results from testing linear through cubic hierarchical scoops on the W111N31 level-2 DTED cell, with 12.9 million points and elevation range [809,2882]. The *number of scoops* refers to how many scoops were needed to get either the maximum absolute error below the threshhold of 10, or the scoop size down to 3×3. Some of the latter scoops have a maximum absolute error over 10, as listed.

3.4 ODETLAP — Overdetermined Partial Differential Equations

This terrain representation technique is an extension of a Laplacian Partial Differential Equation (PDE). That interpolates from a set of data points to a complete array of elevations by defining an 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. Border points are special cases, which are a little tricky to define properly. This equation has a physical origin. If the values represent temperatures in a planar medium instead of elevations, and the known points are places where heat is being applied or removed to maintain them at the given temperature, then the equation solves for the temperatures at all the unknown points. The Laplacian is easily solvable, say in Matlab, on arrays of several thousand square. However, the Laplacian has some limitations when used to interpolate terrain elevations, although these properties are physically correct for heat flow.

1. The interpolated values fall within the range of the known values, so local maxima, such as mountain tops, are never generated.

2. When a set of nested contours is interpolated, the generated surface droops, much like a cloth draped between two supports.

To remove those limitations, to provide other benefits, we extend this classical technique as follows.

1. Equation 1 is applied to *every* non-border point, known or unknown.

2. The known points also have a second equation,

$$z_{ij} = h_{ij} \tag{2}$$

where h_{ij} is the known elevation at that point, and z_{ij} is the computed elevation. Equation 2 is not trivial because our system of linear equations now has more equations than unknowns, that is, it is *overdetermined*, and almost certainly *inconsistent*. Therefore an exact solution is impossible. The best we can do is to solve the equations approximately, while minimizing the RMS errors. That is, if we combine the two types of equations into one matrix and one column vector: $Az = b$ then the best we can do is to solve $Az = b + e$ while minimizing the error e. The solution is $z = (A^T A)^{-1} A^T b$ although practical solution techniques use more efficient, albeit more complicated formulae. Although this system has very many unknowns, 1201^2 for a level-1 DEM, most of the coefficients in A are zero, that is, the system is *sparse*. The key consideration when solving a sparse linear system is, to what extent are the zero entries filled in with nonzero values as the system is solved? Specialized solution techniques exist, and more are being developed. ODETLAP's novelty is the overdetermined system, which was not feasible until recent large sparse system solution techniques were developed. Matlab can easily process cells with 400×400 posts (160000 unknowns). (Childs, 2003) processes larger systems.

There are several advantages to using an overdetermined Laplacian (ODETLAP) system of linear equations for approximating terrain. (Any resemblence to interpolation with springs is only superficial.) *Approximation* is now the correct term instead of *interpolation* since the fitted surface does not pass through the data points. Since the data is not exact, that is an advantage, as it leads

to smoother surfaces while minimizing the error. ODETLAP can handle both continuous contour lines of elevations, which may have gaps, and isolated points, while producing a surface that infers mountain tops inside innermost contours while enforcing continuity of slope across contours and so showing no visible indication of the input contours, i.e., no generated terraces. So far as we know, no other interpolation method has all these advantages.

3.4.1 Choice of ODETLAP Input Points

ODETLAP approximates a surface to a set of points $\{(x, y, z)\}$. How shall we select those points? We've tried several methods.

Regular method: Subsample every k-th point in both the x and y directions. This is easy.

TIN method: Run TIN on the input elevation file, and then use the first K points inserted into the TIN as the ODETLAP interpolation points.

Refined method: Iterate the ODETLAP process, as follows, by analogy to the TIN greedy insertion idea.

1. Start with some terrain, $\mathcal{A}$.
2. Pick $\mathcal{S}$, a set of input points, by the TIN method.
3. Run ODETLAP on them to reconstruct an approximate terrain, $\mathcal{B}$.
4. Compute the error between $\mathcal{A}$ and $\mathcal{B}$, and find $\mathcal{M}$, a set points with greatest absolute error.
5. Insert the $\mathcal{M}$ points into $\mathcal{S}$.
6. Rerun ODETLAP.

3.4.2 Experiments

We experimented with those three methods on five sets of terrain with a resolution of 400×400. Each method started with 1000 points; the refined method added 100 more worst points to the 1000 TIN points. The smoothness parameter $R = 0.3$, which means to value accuracy more than smoothness.

In the following table, "avg err is shorthand for "average absolute error" and "max err is shorthand for "maximum absolute error".

	TIN		**Regular**		**Refined**	
Data	**avg err**	**max err**	**avg err**	**max err**	**avg err**	**max err**
w113n3310	15.870	108.932	9.152	196.590	15.135	84.441
w113n3311	18.703	144.557	10.541	161.193	15.654	105.767
w113n3312	7.262	44.063	2.566	137.632	6.722	40.959
w113n3313	24.214	104.089	6.810	115.434	22.844	80.978
w113n3314	21.093	104.676	4.471	121.059	17.313	63.388

From the table, we have the following observations:

1. The regular method generally has a lower average error than the TIN method, but its maximum error is larger. The regular method's errors are quite nonuniform, as is shown by its maximum error being much larger than its average error.

2. The TIN method generally has a much lower maximum error than the regular method, but a higher average error, so its errors are more evenly distributed, which is probably desirable.

3. The refined TIN method has the best maximum error. Its average error is less than the unrefined TIN method, but still larger than the regular error. Overall, this method has the most uniform error distribution, and we recommend it.

Because of the possibility that these results might be specific to the parameters used, we repeated the comparison of the (unrefined) TIN method to the refined TIN method for the 15 cases of $R = 0.3, 1, 3$ combined with $N = 1000, 3000, 10000, 30000$. The average error improved by an average of 5.2%, and the maximum error improved by an average of 11.2%.

3.4.3 Data Conflation

Sometimes we wish to supplement $\mathcal{A}$, a large, low precision, terrain database with $\mathcal{B}$, a small, high precision, database. $\mathcal{B}$ covers only part of $\mathcal{A}$, and is probably somewhat inconsistent with $\mathcal{A}$ there. Since ODETLAP processes inconsistent equations, it can merge $\mathcal{A}$ and $\mathcal{B}$.

3.5 Correcting Errors in General

This is a general technique for refining the accuracy of any terrain representation. • Start with a terrain matrix A. • Apply any lossy technique to A, to produce B, which when uncompressed produces an approximate terrain C. • Compute the error matrix $E = A - C$. • Compress E, forming a representation F. • Store or transmit (B, F). This method's utility resides in the errors' correlations, so that $|F|$ is small. Note that correlations in the errors mean that the original compression did not exploit all the structure in the terrain, but that's another topic.

4 Future Work

Scooping was designed to be is analogous to scooping earth out of the side of a hill. Eventually, the scoops will follow a trajectory, starting from a given point, and proceeding in a downhill direction from there along a straight line trajectory for a given distance. It will scoop out a new gully of some width, whose bottom has some slope. As described, there are five parameters, although that could be varied. This will have several properties. • It will not create a local minimum. This desirable feature contrasts to every other known terrain representation method. • It naturally lends itself to the creation of complex drainage systems, again in contrast to other representations. • It will be quite nonlinear, and so has a power not available to linear methods.

Finally, any representation should have a more sophisticated evaluation criterion than absolute error. We are performing experiments on the effect the errors on important applications such as visibility and mobility, which may be combined into the *smugglers' path test*. We site observers,

compute viewsheds, and find an optimal path between some source and goal on the alternate terrain representation. Then, we compute the observers' accurate viewsheds on the original elevation matrix, and count how much of that path, which was supposed to be completely hidden, is actually inside any of the accurate viewsheds. Preliminary results show that are representations are quite good under this metric.

5 Acknowledgements

This paper was supported by the National Science Foundation grant CCR 03-06502, and by DARPA/DSO under the Geo* program.

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. `http://www.rh.edu/~chil7317/`. See also `http://www.terrainmap.com` (both accessed 23 May 2006).

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

Franklin, W. R. and Said, A., 1996. Lossy compression of elevation data. In: Seventh International Symposium on Spatial Data Handling, Delft.

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

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

Jost, J., 2002. Riemannian Geometry and Geometric Analysis. Springer.

Osher, S. J. and Fedkiw, R. P., 2003. Level Set Methods and Dynamic Implicit Surfaces. Applied Math Sciences, Vol. 153, Springer.

Said, A. and Pearlman, W. A., 1993. Reversible image compression via multiresolution representation and predictive coding. In: Proceedings SPIE, Vol. 2094: Visual Commun. and Image Processing, pp. 664–674.

Salomon, D., 2000. Data Compression: The Complete Reference. Second edn, Springer.

Speckmann, B. and Snoeyink, J., 1997. Easy triangle strips for TIN terrain models. In: Proc. 9th Canadian Conference on Computational Geometry, pp. 239–244.

Turan, G., 1984. Succinct representations of graphs. Discrete Applied Math 8, pp. 289–294.