# Lossy Compression of Elevation Data

Wm Randolph Franklin

Electrical, Computer, and Systems Engineering Dept., 6026 JEC,
Rensselaer Polytechnic Institute, Troy, New York 12180–3590
+1 (518) 276–6077, Fax: +1 (518) 276–6261
wrf@ecse.rpi.edu, http://www.ecse.rpi.edu/homepages/wrf/

*and*

Amir Said

DENSIS - Faculty of Electrical Engineering
University of Campinas (UNICAMP)
Campinas, SP 13081, Brazil
amir@densis.fee.unicamp.br

August 1995

**Abstract**

Generic lossy image compression algorithms, such as Said & Pearlman's, perform excellently on gridded terrain elevation data. For example, an average DEM compressed down to 0.1 bits per point has an RMS elevation error of 3 meters, although different DEMs ranged from 0.25 to 15 meters. This performance compares favorably to compressing with Triangulated Irregular Networks. Preliminary tests also suggest that the visibility indices of the points are robust with respect to our lossy compression.

# Contents



# List of Figures



# List of Tables

# 1 Introduction

Earlier, in [9] we studied lossless compression of DEM (Digital Elevation Model) terrain data. We showed that image processing algorithms work surprisingly well. On the average elevation data compressed losslessly to 2 bpp (bits per point), although this ranged from 0.5 to 4 bpp in our test cases. This compares favorably to storing the data as binary integers, in 16 bpp, or even using `gzip`, at 4 bpp on the average.

The best algorithm was by Said and Pearlman[17–20]. There are several variants; here we use `progcode`. `Progdecd`, is a separate decoder program, which facilitates validating the results. `Progcode` can now handle arbitrary sized rectangular files with 1 or 2 bytes per point. It is a progressive resolution method, which first calculates an approximation to the file, and then successive refinements, so that the final result is exact. `Progdecd` can report what the mean squared error would be if the file was compressed lossily at various bitrates. We use this in this paper.

The main question with storing elevation data is how to compress it. The two main competitors are the Triangulated Irregular Network (TIN) and the regular array or grid. Accepting the TIN means accepting the principle of lossy compression since an exact TIN would require about a triangle for every point. Since each triangle requires many bytes, this would increase storage by perhaps an order of magnitude.

Since we've accepted that compressing elevation data may be lossy, the next question is whether the TIN is the right way to go. This paper answers that question, *"No"*. Altho the TIN has other advantages, such as that ridge and stream lines can be explicitly encoded, it is not as compact as compressing the DEM.

The general relevance of these results is that compression research in image processing in computer science is important in terrain data representation. This is something that was not totally obvious a priori, since the data are so different. This is important since so many resources are being thrown at the compression problem in image processing. We might expect even better results in the future.

However, this does follow the trend in computer science, in hardware at least, of general purpose solutions often being better than special purpose ones, because of the greater amount of resources devoted to the general problem. This is why most of the machines developed in all the following special purpose categories have failed: Lisp machines, floating point processors, database engines, special graphics engines, and parallel machines.

# 2 Review

Weibel[25] filters gridded DEMs in various ways. He uses global filtering doing smoothing as in image processing by convolving with a $3 \times 3$ or $5 \times 4$ filter. He compares this with a selective filtering to eliminate points that do not add anything to our characterization of the surface. He tests a $220 \times 390$ elevation grid to see whether generalization changes essentials of the terrain, such as hill sharing and RMS error. Shea and McMaster[21] also discuss generalization.

Chang and Tsai[3] found that lowering DEM resolution hurt the accuracy of the calculated slope and aspect of the terrain. Carter[2] shows that the 1 meter resolution particularly affects the aspect, causing a bias towards the four cardinal directions, and suggests smoothing the data. Lee et al[11] analyze the effect of elevation errors on feature extraction. Fisher[8] considers the effect on visibility.

One operation often performed on terrain data is visibility determination, De Floriani and Magillo[5]. Puppo et al[15] use a parallel machine to convert a DEM to a TIN. They scale the elevation to 8 bits and perform experiments on grids of up to $512 \times 512$, reporting results for a $128 \times 128$ grid.

Table 1: The 24 Test Cases

| Name | Stdev of Elevations | Gzip Size, Bytes | Progcode size, Bytes | Lossless rate, Bits per Point |
|---|---|---|---|---|
| Aberdeen E | 36. | 222,245 | 167,629 | 0.93 |
| Baker E | 377. | 1,395,574 | 626,961 | 3.48 |
| Caliente E | 335. | 1,264,671 | 534,495 | 2.96 |
| Dalhart E | 87. | 431,241 | 262,995 | 1.46 |
| Eagle Pass E | 88. | 494,974 | 273,267 | 1.52 |
| Fairmont E | 34. | 367,819 | 240,675 | 1.33 |
| Gadsden E | 74. | 872,802 | 440,551 | 2.44 |
| Hailey E | 516. | 1,566,137 | 610,836 | 3.39 |
| Idaho Falls E | 145. | 455,373 | 270,683 | 1.50 |
| Jacksonville W | 7.7 | 120,413 | 93,909 | 0.52 |
| Kalispell E | 343. | 1,421,867 | 682,352 | 3.78 |
| La Crosse E | 49. | 1,028,266 | 612,162 | 3.40 |
| Macon E | 22. | 508,171 | 302,247 | 1.68 |
| Nashville E | 37. | 801,073 | 414,536 | 2.30 |
| O'Neill E | 74. | 593,771 | 324,296 | 1.80 |
| Paducah E | 27. | 581,892 | 318,113 | 1.76 |
| Quebec E | 140. | 853,427 | 374,115 | 2.07 |
| Racine E | 17.0 | 134,653 | 94,068 | 0.52 |
| Sacramento E | 695. | 1,467,837 | 737,945 | 4.09 |
| Tallahassee E | 26. | 427,257 | 266,154 | 1.48 |
| Ukiah E | 514. | 1,107,751 | 562,100 | 3.12 |
| Valdosta E | 9.3 | 219,191 | 166,315 | 0.92 |
| Waco E | 25. | 492,679 | 291,135 | 1.61 |
| Yakima E | 305. | 1,259,666 | 500,679 | 2.78 |
| *Average* | 166. | 753,698 | 382,009 | 2.12 |

For example, for a 30 meter accuracy 497 of the 16,384 points are selected. This TIN is then used to calculate line-of-sight-communication in De Floriani et al[6].

Drainage pattern determination is another frequent DEM operation, as described by McCormack et al[14]. Skidmore[22] extracts properties of a location, such as being on a ridge line, from a DEM. Franklin and Ray[10, 16] do visibility calculations on large amounts of data.

The use of a linear quadtree with 2-D run-length encoding and Mortin sequences is discussed in Mark and Lauzon[13]. The storage can be about 7 bits per leaf. Waugh[24] critically evaluates when quadtrees are useful, while Chen and Tobler[4] find that quadtrees always require more space for a given accuracy than a ruled surface. Dutton[7] presents a region quadtree based on triangles, not squares. This quaternary triangular mesh defines coordinates on a quasi-spherical world better than a planar, Cartesian, system does. Leifer and Mark[12] use orthogonal polynomials of order up to 6 and quadtrees for a lossy compression of three $256 \times 256$ DEMs. Their work anticipates ideas used in wavelets and in the best current image processing methods.

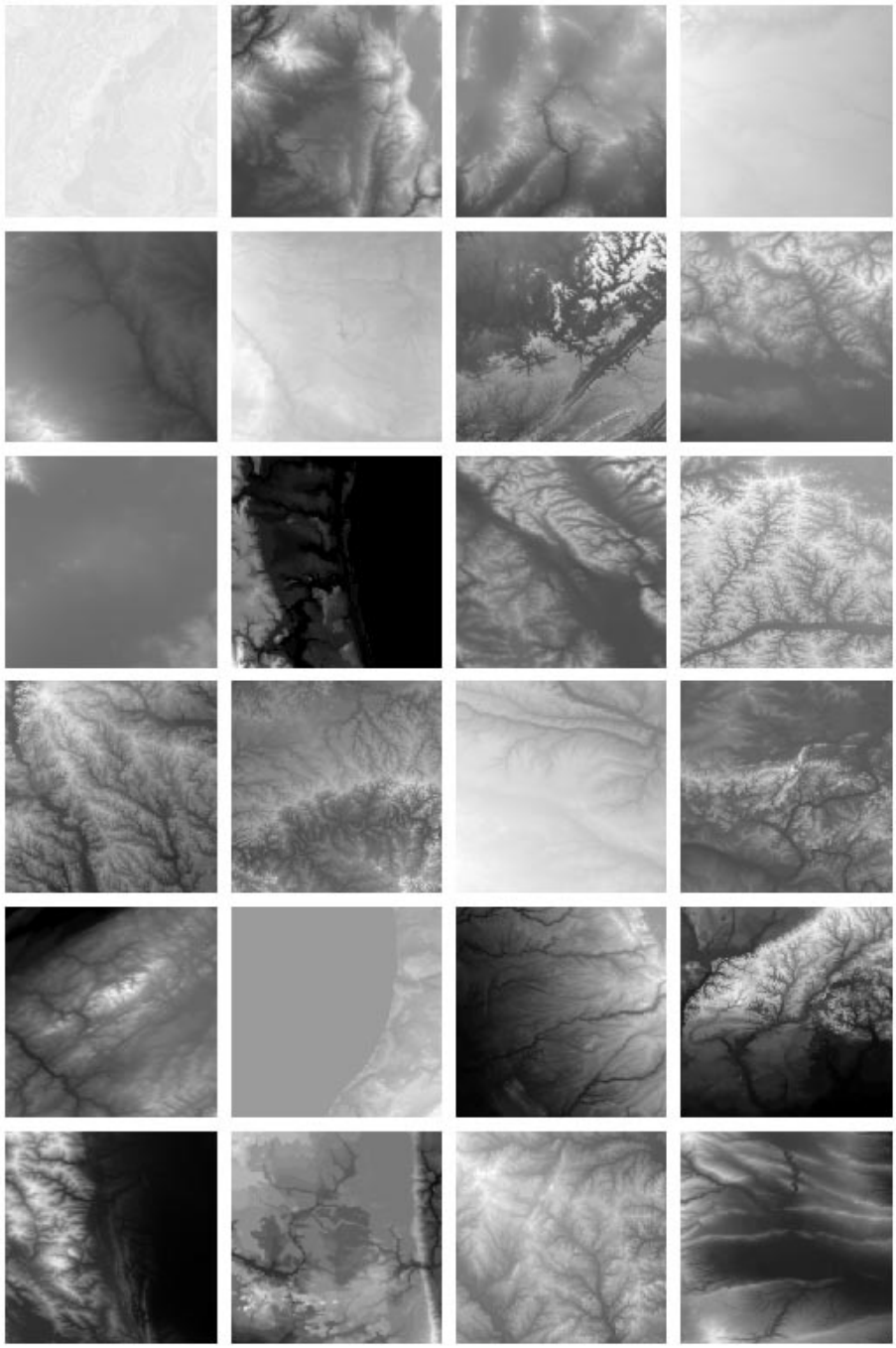

Figure 1: The 24 Sample USGS DEMs

# 3 Lossy Compression Experiments

Our test data are 24 level-1, 3 arc-second DEM files, or cells. We picked the first cell starting with each letter of the alphabet. They are listed in Table 1 on page 4, and shown in Figure 1. We use only each cell's elevation data, which is a $1201 \times 1201$ array of points, taking 2,884,802 bytes with the elevations stored as 2-byte integers. By way of comparison, when compressed with `gzip` using the default setting, the average resulting size is 754KB, while with `progcode`, the average size is 382KB. The test cases vary considerably in compressibility, but `gzip` and `progcode` tend to track each other in this respect. Both of them track the standard deviation of the elevations from the mean elevation for that cell.

The numbers are somewhat different in Table 1 than in [9] since there we were using a $1024 \times 1024$ subset of each cell, while here we are using the whole $1201 \times 1201$ cell. Compression does not hinder interactive use of the data; partitioning one test file into 256 blocks before compressing increased the total size by only 13%.

How good is lossy compression? Our measure of goodness was the RMS error in elevations in the lossily compressed cell, compared to the original, as reported by `progdecd`. Image processors tend to report compression rates as *bits per pixel (or point)*, or *bpp*. A cell compressed to 1 bpp would take $1201 \times 1201/8 = 180300$ bytes.

We tested each cell at bit rates of $0.005, 0.010, 0.015$, and then from 0.02 up in steps of 0.02 bpp to the lossless compression rate, which is shown in Table 1. The results are plotted in Figure 2 on the next page for the first 12 cells, and Figure 3 on page 8 for the second 12. In these plots, one line shows the RMS error for one cell compressed at the different bit rates. The letters in a column near the left of each plot show the standard deviation of the elevations of the cell whose name starts with the letter. For example, $h$ is at 516 since that is the standard deviation of the elevations of *Hailey E*. We see that the lossy compressibility tends to track the standard deviation, although not perfectly.

These plots show that lossy compression is surprisingly good. One of the worst of the 24 cells is Hailey E, whose elevations have an standard deviation of 516. Table 2 on page 16 shows the RMS error for some different bit rates. The low compressed rate of 0.1 bpp gives an RMS error of 12.2, or 2.3% of the elevation standard deviation. The extremely low compression rate of 0.01 bpp, or 1803 bytes for the whole file, gives an RMS error of 10% of the elevation standard deviation.

Study the plots to see the average behavior of the 24 test cells. Compressing to 0.1 bpp gives an RMS error of 3 meters, and 0.01 bpp gives an RMS error of about 10 meters. These are not bad considering the accuracy standards of the original data, which are described in Carter[1] and Walsh et al[23].

What do the lossily compressed cells look like? Figure 4 on page 9 shows the original Hailey E cell[1]. Figures 5 on page 10, 6 on page 11, 7 on page 12, 8 on page 13, and 9 on page 14 show the cell compressed to 1, 0.1, 0.03, 0.01, and 0.005 bpp, respectively.

Even though the RMS error is small, perhaps the errors might be badly distributed. Figure 10 on page 15 shows the absolute differences of the elevations between the 0.03 bpp compression and the original image. The errors are distributed fairly evenly. Ten percent of the pixels are in error by less than 14 meters, 50% by less than 37, 90% by less than 82, and the worst error is 224 meters. This compares favorably to the range of elevations, which is 2646 meters.

[1] All the greyscale images in the paper have been histogram-equalized to enhance their contrast and make the details more visible.

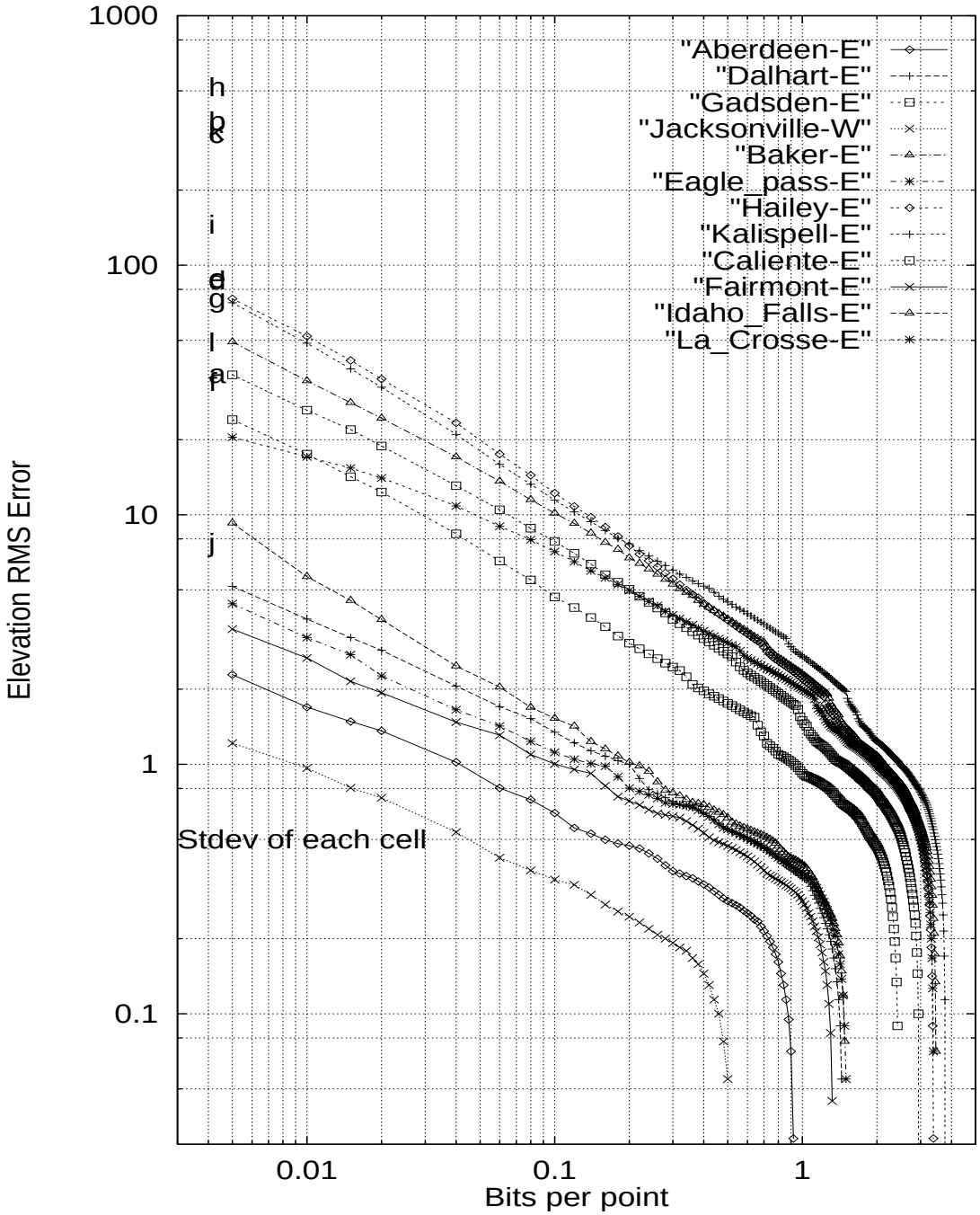


Figure 2: Lossy Compression of First 12 DEMs

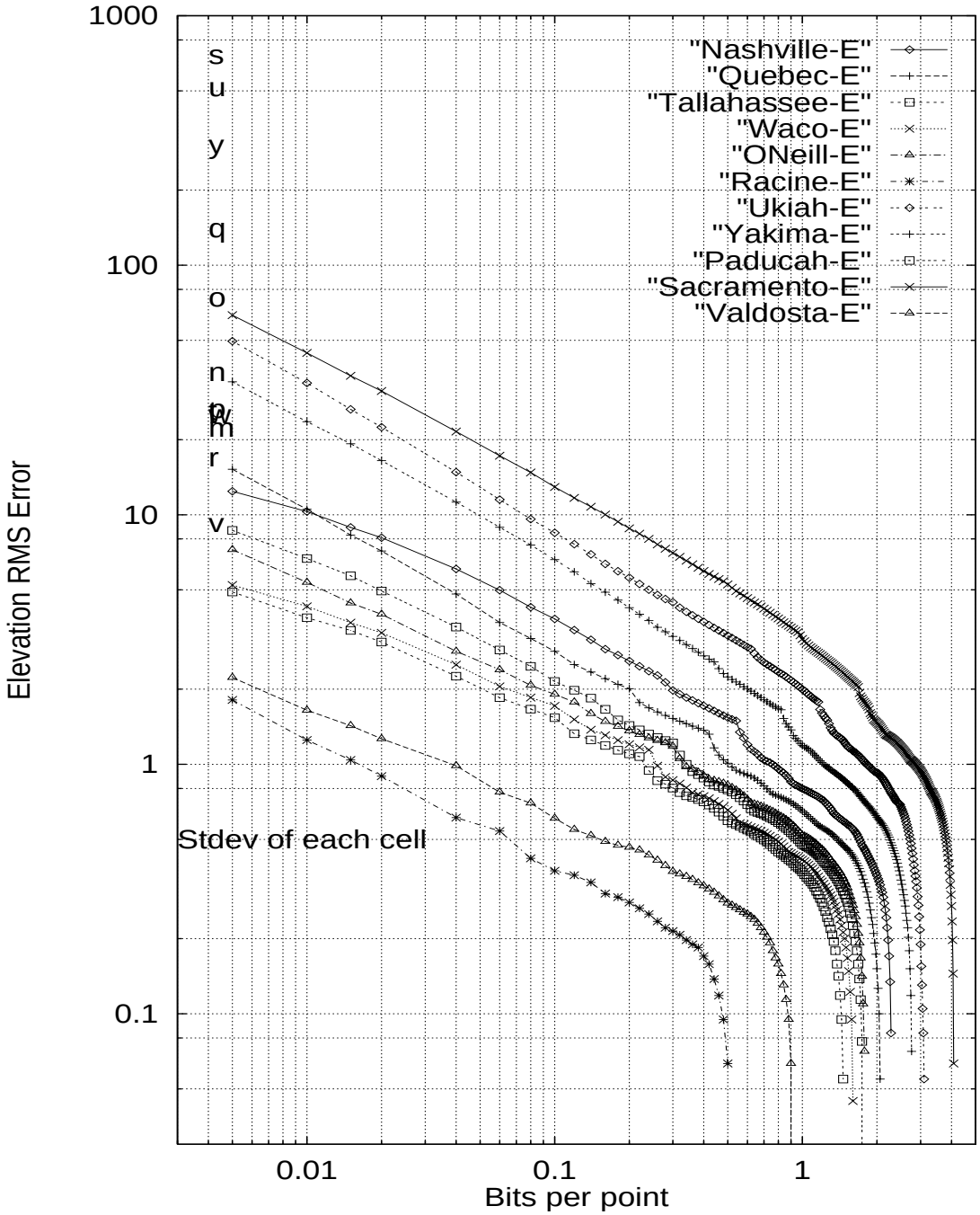


Figure 3: Lossy Compression of Second 12 DEMs

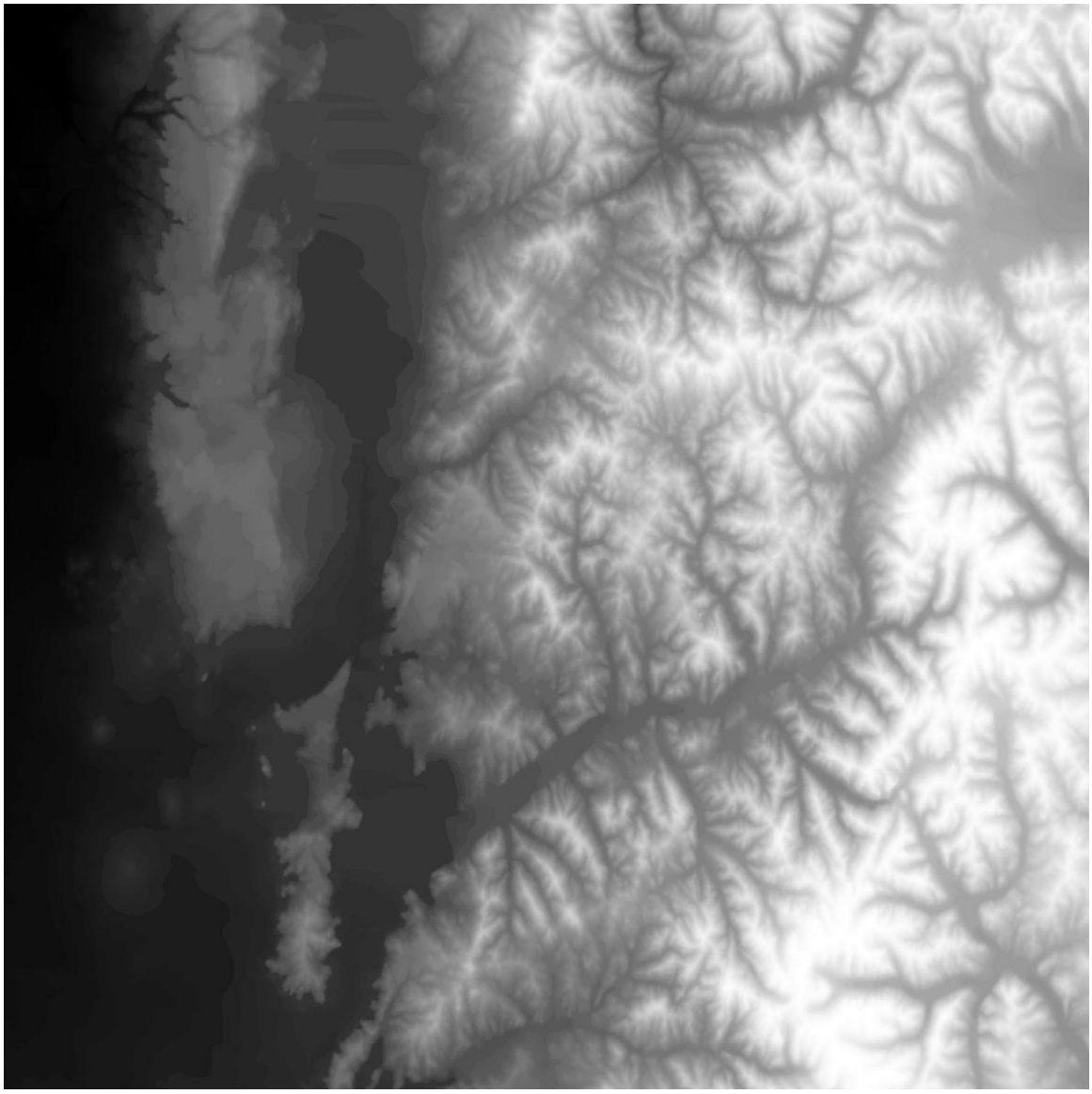

Figure 4: Hailey E Cell

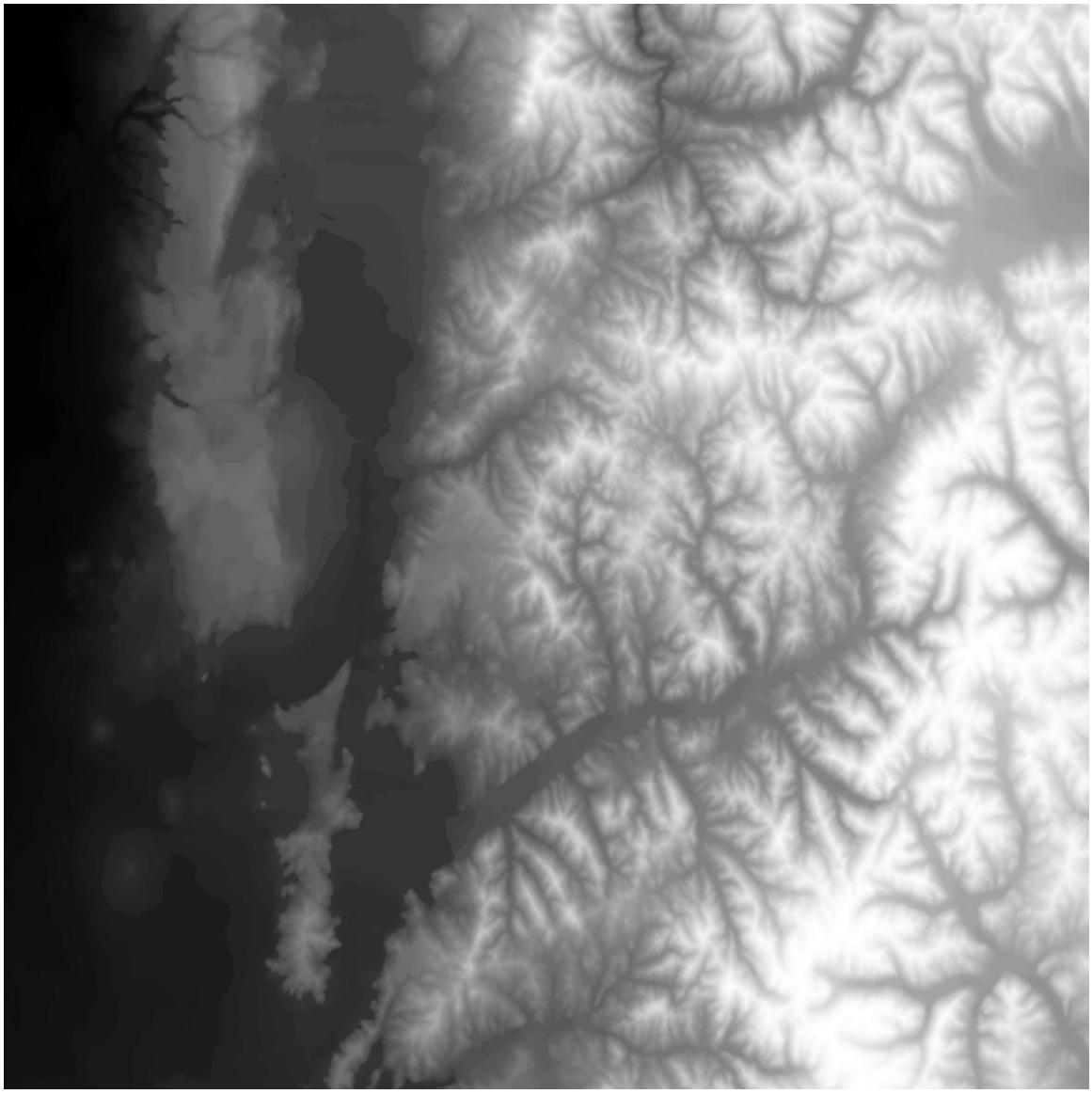

Figure 5: Hailey E Cell, Compressed to 1.0 bpp

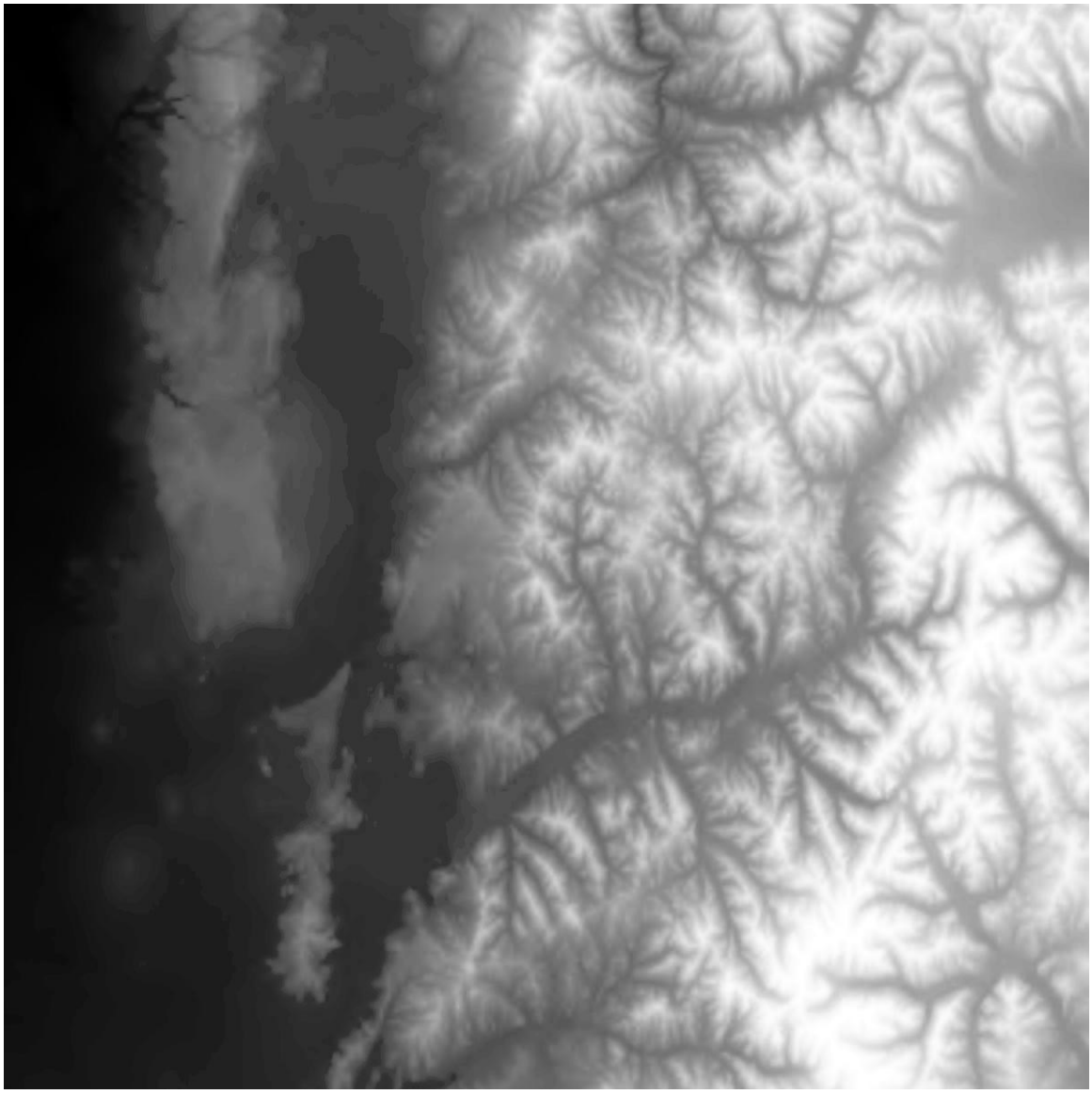

Figure 6: Hailey E Cell, Compressed to 0.1 bpp

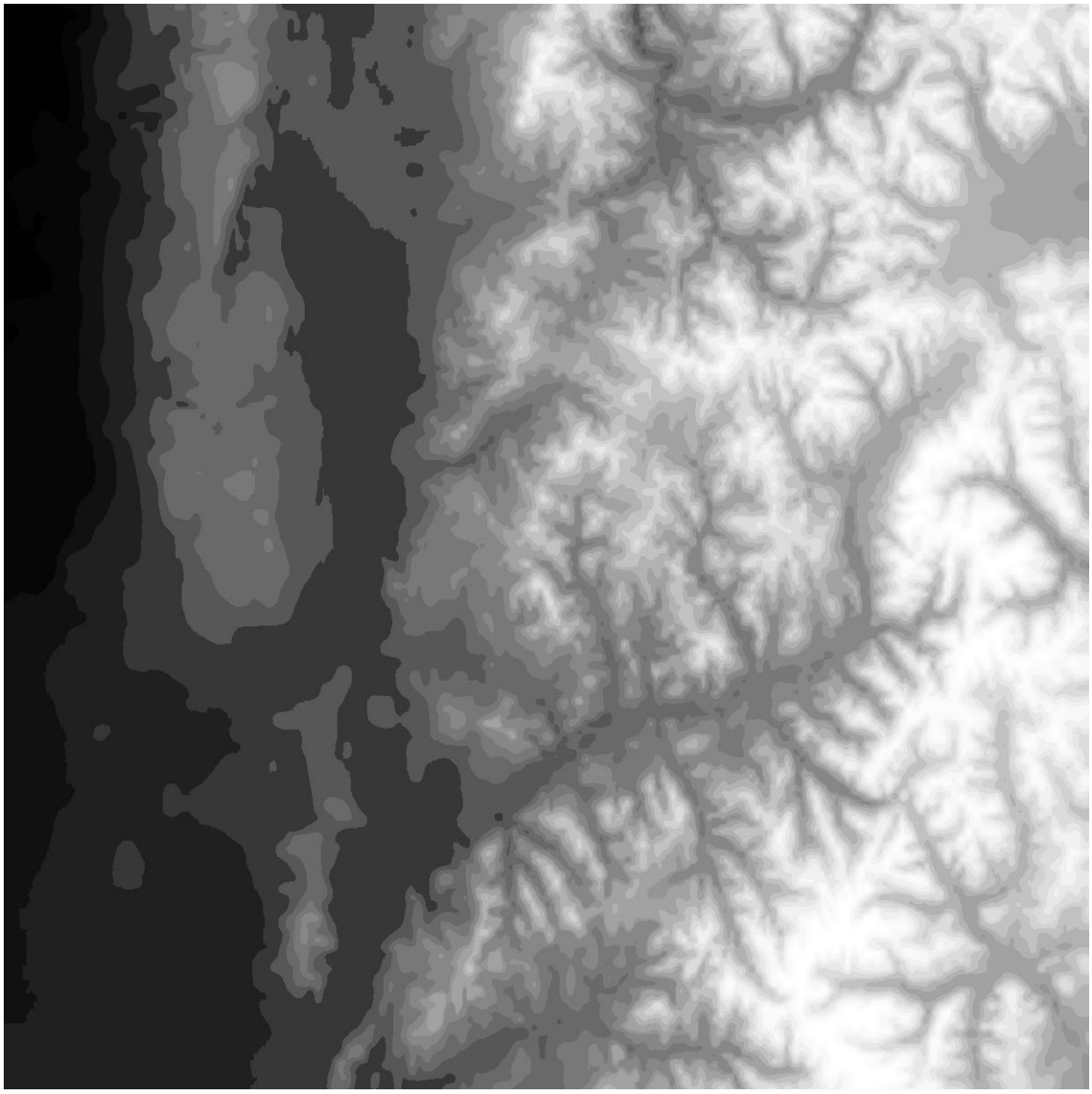

Figure 7: Hailey E Cell, Compressed to 0.3 bpp

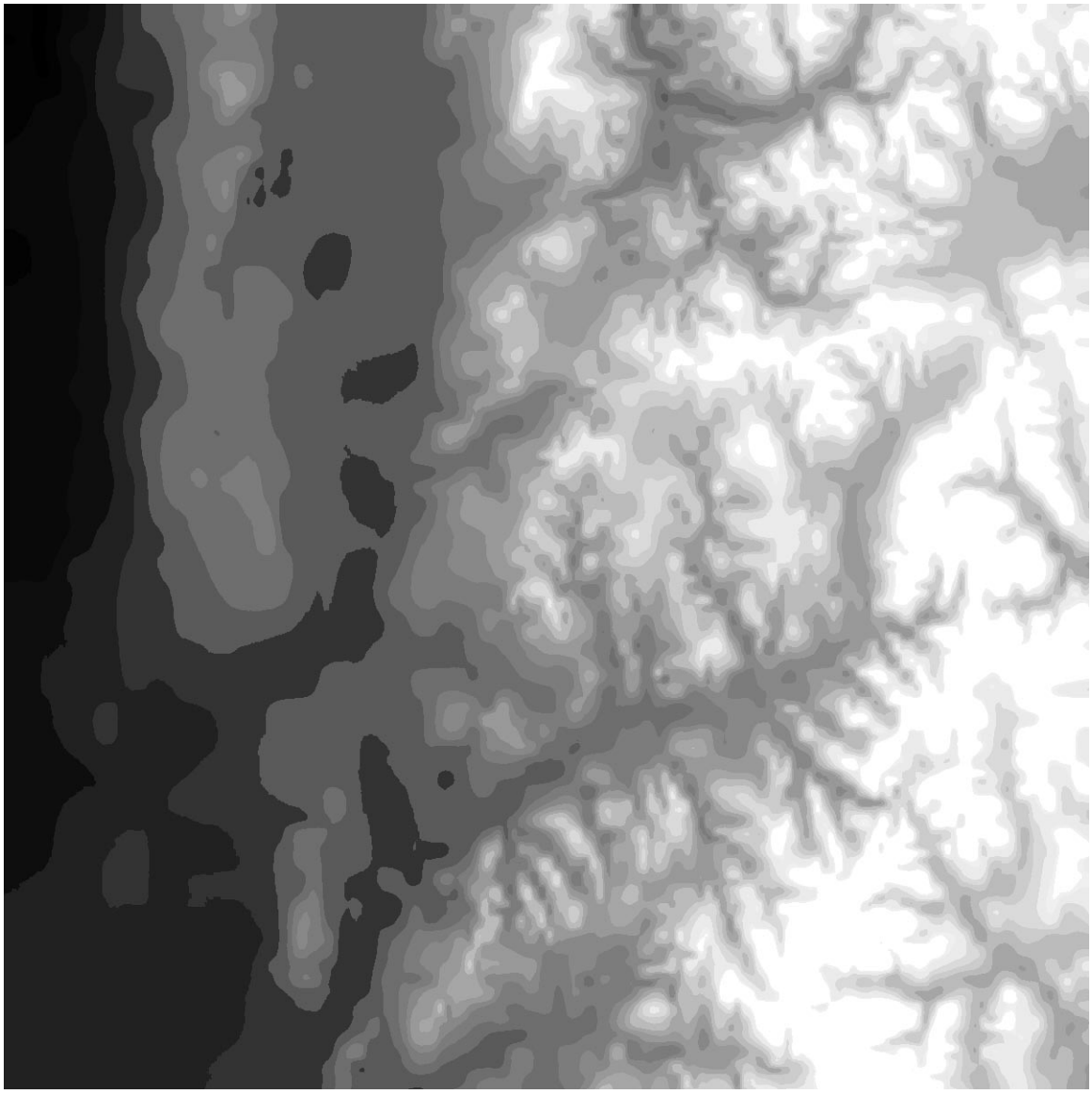

Figure 8: Hailey E Cell, Compressed to 0.01 bpp

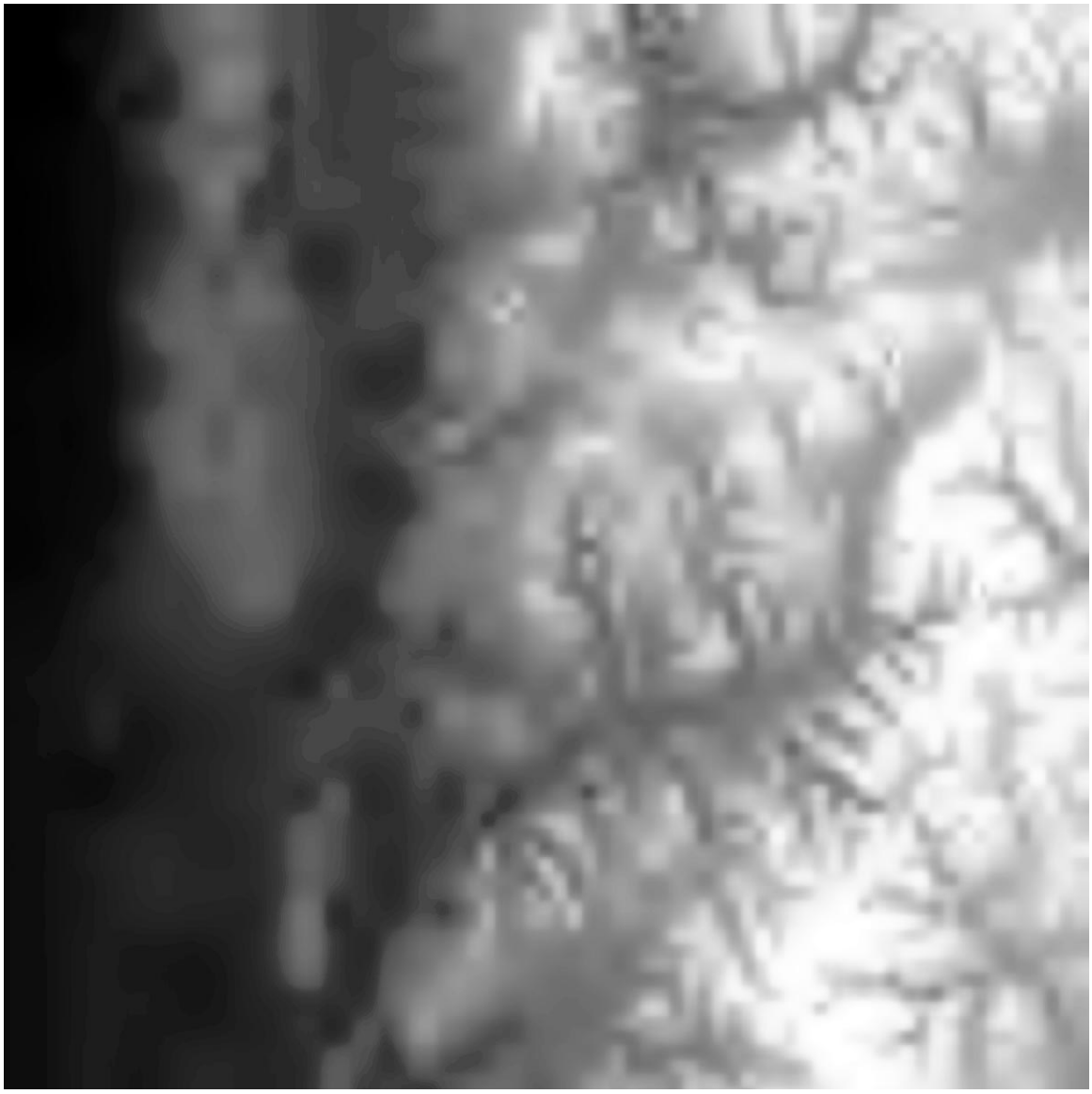

Figure 9: Hailey E Cell, Compressed to 0.005 bpp

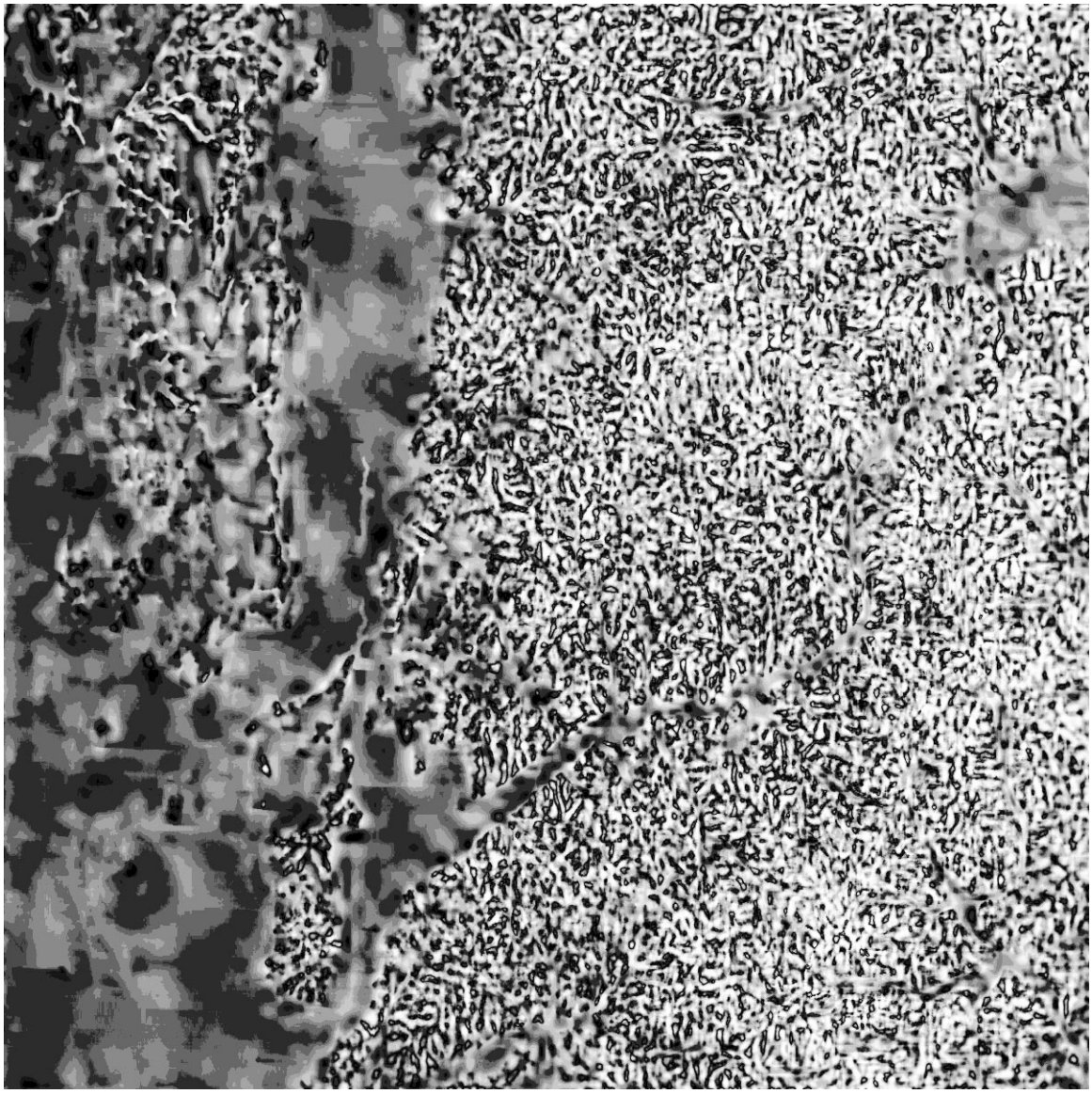

Figure 10: Errors of the 0.03 bpp Compression

Table 2: RMS Error vs Bitrate for Hailey E

| bpp | RMS error | Error / Stdev |
|---|---|---|
| 0.0005 | 323. | 0.63 |
| 0.001 | 190. | 0.37 |
| 0.002 | 125. | 0.24 |
| 0.003 | 95. | 0.18 |
| 0.004 | 85. | 0.16 |
| 0.005 | 73.4 | 0.142 |
| 0.010 | 51.9 | 0.100 |
| 0.015 | 41.5 | 0.080 |
| 0.020 | 35.0 | 0.067 |
| 0.030 | 27.6 | 0.053 |
| 0.040 | 23.3 | 0.045 |
| 0.060 | 17.5 | 0.033 |
| 0.080 | 14.3 | 0.027 |
| 0.100 | 12.2 | 0.023 |
| 0.200 | 7.50 | 0.014 |
| 0.300 | 5.53 | 0.010 |
| 0.400 | 4.42 | 8.565e-3 |
| 0.500 | 3.77 | 7.306e-3 |
| 0.600 | 3.32 | 6.434e-3 |
| 0.700 | 2.94 | 5.697e-3 |
| 0.800 | 2.59 | 5.019e-3 |
| 0.900 | 2.38 | 4.612e-3 |
| 1.000 | 2.20 | 4.263e-3 |
| 1.200 | 1.89 | 3.662e-3 |
| 1.400 | 1.43 | 2.771e-3 |
| 1.600 | 1.24 | 2.403e-3 |
| 1.800 | 1.13 | 2.189e-3 |
| 2.000 | 1.02 | 1.976e-3 |
| 2.200 | 0.92 | 1.782e-3 |
| 2.400 | 0.82 | 1.589e-3 |
| 2.600 | 0.71 | 1.375e-3 |
| 2.800 | 0.61 | 1.182e-3 |
| 3.000 | 0.49 | 9.496e-4 |
| 3.100 | 0.42 | 8.139e-4 |
| 3.200 | 0.34 | 6.589e-4 |
| 3.300 | 0.21 | 4.069e-4 |
| 3.320 | 0.184 | 3.565e-4 |
| 3.340 | 0.141 | 2.732e-4 |
| 3.360 | 0.089 | 1.724e-4 |
| 3.380 | 0.031 | 6.007e-5 |
| 3.390 | 0.0 | 0. |

# 4 Effect of Lossy Compression on Visibility Analysis

Presumably we want the elevation data in order to perform some operation, such as calculating visibility or drainage patterns. Therefore, much lossiness can we tolerate before these essential

properties of the data are damaged? We tested the visibility index for Hailey E compressed to 0.03 bpp using `los`, an approximate visibility index program of Franklin[10]. It fires 16 rays out from the observer and counts how many points on these rays are visible. At larger distances from the observer, it also skips points along each ray. The effect is to measure the visibility of a sample of the targets within the range.

Figures 11 on the next page shows the visibility indices of the points in the original Hailey-E cell. For each of the $1201 \times 1201$ points in turn, we assumed an observer at that point, 10 meters above the surface. The observer was looking for targets within a range of 50 points that were 10 meters above their local surface. The fraction of possible targets visible from that observer is called the *visibility index* of that point. A brighter point has a higher visibility index.

Figure 12 on page 19 shows the visibility index of each point in the Hailey-E cell when lossily compressed to 0.03 bpp, or only 5409 bytes for the whole cell. The agreement is quite good. The chief differences are a loss of some fine detail and errors in the flatter regions. Some of this might be an artifact of the approximations in our visibility program.

Since detail is lost in the printed images, how does the visibility index compare point-by-point? Let each visibility index range from 0 (for an observer who can't see any potential target) to 100 (for an observer who can see them all). We took the point-by-point difference of the above two data sets. The median difference in the visibility indices was 5.7. The 25-percentile was at 2.3, while the 75-percentile was at 11. Therefore even a compression by a factor of 100 from the lossless rate does not seriously hurt the data for the purposes of visibility index computation.

A less aggressive compression is even better. We tried Hailey-E again, compressed only to 0.3 bpp, and differenced the visibility indices. The 25-percentile is 0.75, the median is 1.5, and the 75-percentile at 3.8.

We are now studying whether this good performance generalizes to other data.

# 5 Implementation Details

We used Sun Sparc IPC and 10/30 Unix workstations with about 32MB of memory, and programmed in C++ with the Apogee compiler. Tomas Rokicki's `dvips`, Jef Poskanzer's Portable Bit Map (PBM) package, Netpbm version, and John Bradley's `xv` were also very useful.

# 6 Conclusion

The Said & Pearlman lossy image compression algorithm `progcode` is an excellent method for compressing gridded elevation data, or DEMs. For any desired accuracy, this method compresses much better than a TIN is likely to do, and is also simpler. For the Hailey-E cell, even a lossy compression to 0.03 bits per point, or a factor of 100 better than the lossless compression rate, which itself was one-fifth of the size of the original binary file, did not change the visibility indices of the points to a large extent.

# References

[1] J. R. Carter. Relative errors identified in USGS gridded DEMs. In *Autocarto*, volume 9, pages 255–265, 1989.

[2] J. R. Carter. The effect of data precision on the calculation of slope and aspect using gridded DEMs. *Cartographica*, 29(1):22–34, Spring 1992.

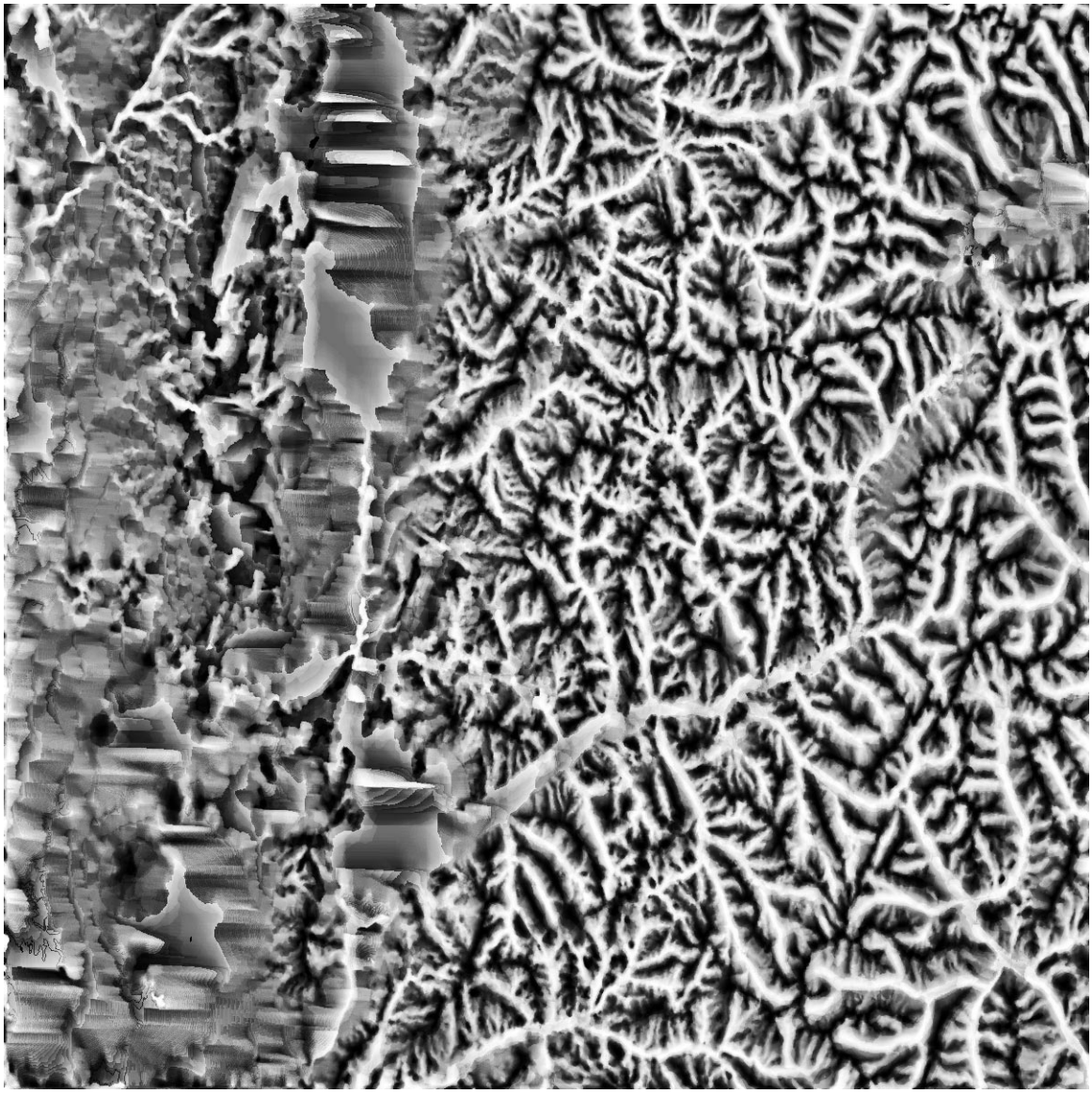

Figure 11: Visibility Indices of Points in the Hailey E Cell

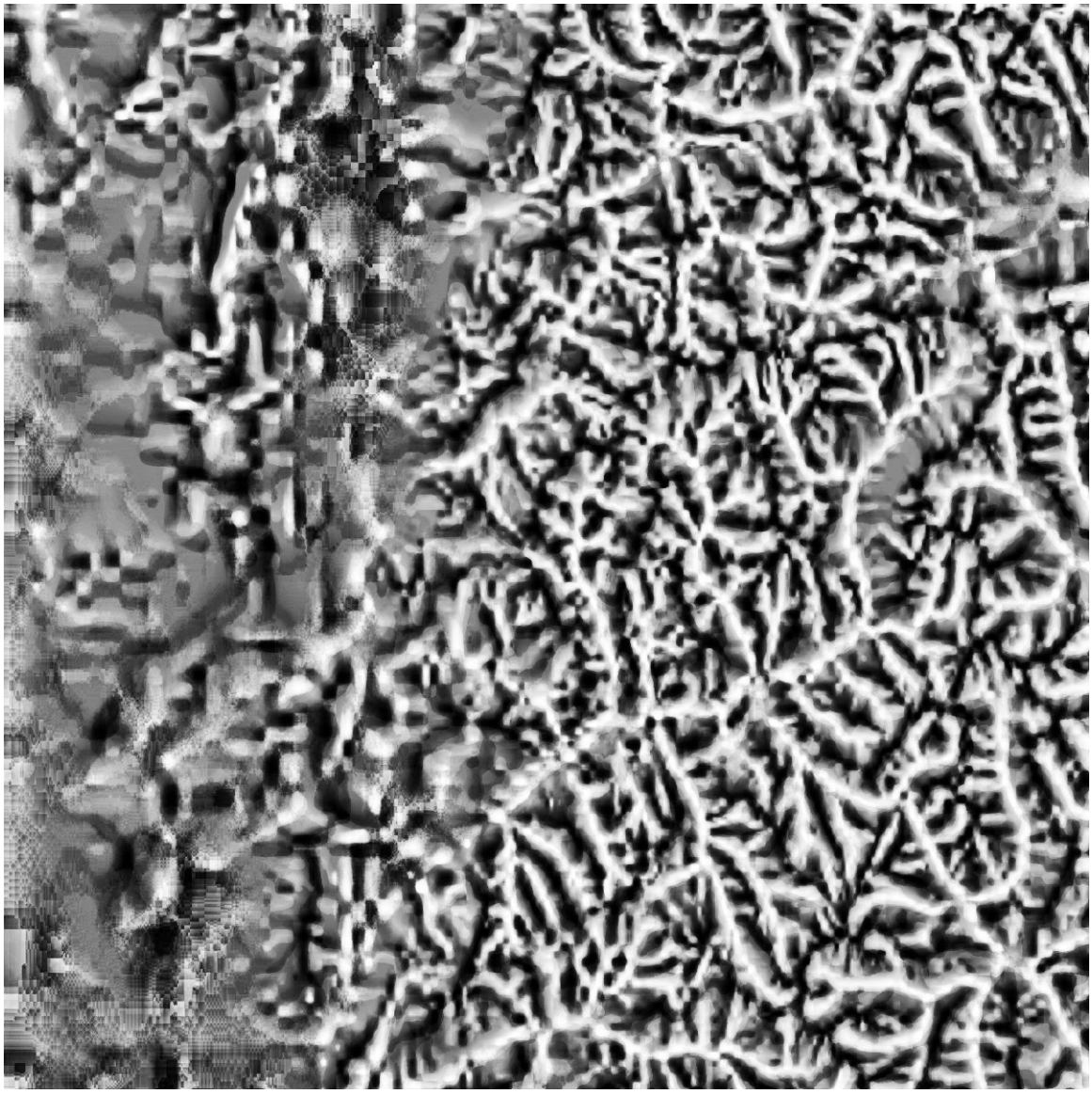

Figure 12: Visibility Indices for the 0.03 bpp Compressed Hailey E Cell

[3] K.-T. Chang and B.-W. Tsai. The effect of DEM resolution on slope and aspect mapping. *Cartography and Geographic Information Systems*, 18(1):69–77, 1991.

[4] Z.-T. Chen and W. Tobler. Quadtree representations of digital terrain. In *Proceedings, Auto-Carto London*, volume 1, 1986.

[5] L. De Floriani and P. Magillo. Visibility algorithms on DTMs. *Int. J. Geographic Information Systems*, 8(1):13–41, 1994.

[6] L. De Floriani, P. Magillo, and E. Puppo. Line of sight communication on terrain models. *Int. J. Geographic Information Systems*, 8(4):329–342, 1994.

[7] G. Dutton. Locational properties of quaternary triangular meshes. In K. Brassel and H. Kishimoto, editors, *4th International Symposium on Spatial Data Handling*, volume 2, pages 901–910, Zürich, 23-27 July 1990.

[8] P. F. Fisher. Algorithm and implementation uncertainty in viewshed analysis. *International Journal Of Geographical Information Systems*, 7:331–347, Jul–Aug 1993.

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

[10] W. R. Franklin and C. Ray. Higher isn't necessarily better: Visibility algorithms and experiments. In T. C. Waugh and R. G. Healey, editors, *Advances in GIS Research: Sixth International Symposium on Spatial Data Handling*, pages 751–770, Edinburgh, 5–9 Sept 1994. Taylor & Francis.

[11] J. Lee, P. K. Snyder, and P. F. Fisher. Modeling the effect of data errors on feature extraction from digital elevation models. *Photogrammetric Engineering And Remote Sensing*, 58:1461–1467, Oct. 1993.

[12] L. A. Leifer and D. M. Mark. Recursive approximation of topographic data using quadtrees and orthogonal polynomials. In N. R. Chrisman, editor, *Autocarto 8: Proceedings Eighth International Symposium on Computer-Assisted Cartography*, pages 650–659, Baltimore, 29 March – 3 April 1987. ASPRS and ACSM.

[13] D. M. Mark and J. P. Lauzon. Linear quadtrees for geographic information systems. In *Proceedings of the International Symposium on Spatial Data Handling*, volume 2, pages 412–430, Zurich, 20–24 August 1984.

[14] J. E. McCormack, M. N. Gahegan, S. A. Roberts, J. Hogy, and B. S. Hoyle. Feature-based derivation of drainage networks. *Int. J. Geographic Information Systems*, 7(3):263–279, 1993.

[15] E. Puppo, L. Davis, D. de Menthon, and Y. A. Teng. Parallel terrain triangulation. *Int. J. Geographic Information Systems*, 8(2):105–128, 1994.

[16] C. K. Ray. *Representing Visibility for Siting Problems*. PhD thesis, Electrical, Computer, and Systems Engineering Dept., Rensselaer Polytechnic Institute, May 1994.

[17] A. Said. An image multiresolution representation for lossless and lossy compression. (submitted), July 1994.

[18] A. Said and W. A. Pearlman. A new fast and efficient image codec based on set partitioning in hierarchical trees. (submitted), presented in part at the IEEE Symp on Circuits and Systems, Chicago, May 1993.

[19] A. Said and W. A. Pearlman. Reversible image compression via multiresolution representation and predictive coding. In *Proceedings SPIE*, volume 2094: Visual Commun. and Image Processing, pages 664–674, Nov. 1993. email: amir@densis.fee.unicamp.br, pearlman@ecse.rpi.edu.

[20] A. Said and W. A. Pearlman. (new image coding and decoding programs). ftp://ftp.ipl.rpi.-edu/pub/EW_Code/, Apr. 1995.

[21] K. S. Shea and R. B. McMaster. Cartographic generalization in a digital environment: When and how to generalize. In *Autocarto*, volume 9, pages 56–67, 1989.

[22] A. K. Skidmore. Terrain position as mapped from a gridded digital elevation model. *Int. J. Geographic Information Systems*, 4(1):33–49, 1990.

[23] S. J. Walsh, D. R. Lightfoot, and D. R. Butler. Recognition and assessment of error in geographic information systems. *Photogrammetry Engineering and Remote Sensing*, 53:1423–1430, 1987.

[24] T. Waugh. A response to recent papers and articles on the use of quadtrees for geographic information systems. In *Proceedings of the Second International Symposium on Geographic Information Systems*, pages 33–37, Seattle, Wash. USA, 5–10 July 1986.

[25] R. Weibel. Models and experiments for adaptive computer-assisted terrain generalization. *Cartography and Geographic Information Systems*, 19(3):133–153, 1992.