Journal of Universal Computer Science, vol. 20, no. 10 (2014), 1433-1442
submitted: 30/10/13, accepted: 20/6/14, appeared: 1/10/14 © *J.UCS*

A Compression Algorithm for Managing Digital Elevation Models in Mobile Devices

Rolando Quintero, Giovanni Guzman, Miguel Torres
Rolando Menchaca-Mendez, Marco Moreno-Ibarra, Felix Mata
(Instituto Politécnico Nacional, Mexico City, Mexico
{rquintero, jguzmanl, mtorresru, rmen, marcomoreno, mmatar}@ipn.mx)

Abstract: Nowadays, there are many applications such as disaster mitigation, surveying or geology-support, and others where Digital Elevation Models (DEM) are useful in the field. DEM typically requires a huge amount of data, making the tasks of DEM transmission over a wireless network or storing and displaying it in a mobile device very complex. These tasks are important challenges in computer science research. Up-to-date, the compression techniques are used to compress DEMs with a high compression coefficient, nevertheless, whether the user requires to access the file or to obtain certain information about the raster data, it is necessary to decompress the entire DEM file. In consequence, these approaches are not well suited for applications focused on devices with limited hardware resources. In this paper, a novel compression/decompression technique is presented. This approach is capable of obtaining the specific parameters such as altitudes and contour lines of a sub-region of the DEM without using a full decompression stage. A detailed analysis of the properties and complexity of our approach is presented.

Key Words: Digital Elevation Model, near-lossless compression, compression algorithms
Category: E.2, H.3.2, I.4.2

1 Introduction

DEMs have gained popularity in applications to determine terrain's attributes, to find features of the terrain, as well as for modeling hydrologic functions, energy fluxes and forest fires [Wiche. 1992]. DEMs, typically require a huge amount of data to make the tasks of transmission them over a wireless network or storing and displaying them in a mobile device. Compression algorithms should be capable of reducing the size of DEM files, so that they can be transmitted and managed in a device with limited capabilities. According to [Aràndiga et al. 2012], there is a third class of tecniques that has attracted the attention of image coding researchers: the near-lossless image coding.

The aim of DEM compression is to remove the redundancy that a geographic data set inherently contains [Kidner and Smith. 2003]. Some proposals have attempted to compress DEMs adopting a strategy of two stages, which firstly it decorrelates the DEM and then stores the elevations or corrections to the elevations in a predetermined number of bits. The approach consists of computing the differential altitude grid and is described in [Boehm. 1967]. In [Marakovic.

1983, Dutton. 1983, Shaffer. 1990] many approaches that present good results on datasets with uniform terrains are described. Unfortunately they do not constrain errors in highly variable or unpredictable terrains. [Le et al. 2011] developed a lossless method based on adaptive sliding windows and parallel computation techniques, taking into account the method presented in [Kidner and Smith. 2003]. In [Aràndiga et al. 2012] a multi-scale data-compression algorithm within Harten's interpolatory framework for multiresolution is presented.

Other works are focused on using off-the-shelf data compression software or even image compression algorithms applied to DEMs [Franklin. 1995]. Traditional raster compression algorithms such as run-length or quad-tree encoding are not adequate when elevation data sets exhibit no uniformity, or consistently changes. In [Franklin. 1995], the state-of-the-art in data compression algorithms applied to regular grids or DEMs is studied. Approaches as GZIP, COMPRESS, JPEG, and SP COMPRESS offer the best methods of error-free data compression [Said and Pearlman. 1996].

In [Chen and Lu. 2007] a loss-less method for large scale terrain compression with high compression rate is proposed. The algorithm is able to decompress terrain data with multi-resolution efficiently at any viewpoint and to render the terrain in real-time by means of a fast updating strategy. In [Li and Gong. 2008] SPIHT wavelet compression method to compress DEMs is presented. It evaluates the terrain surface complexity, and computes the bit rate of the encoding process.

The previous approaches are not well suited for the case of a mobile device with limited hardware resources. In this context, the main drawback of the traditional compressing/decompressing approaches is that the user has to download/store, and decompress the whole DEM, even if only a fraction of that DEM is relevant for the user. In this paper, we propose an algorithm oriented towards compressing a DEM, which is based on a sequence of images with a specific height. The sequence is compressed by applying a binary compressor. The main contribution of the work is that our algorithm is capable of obtaining the specific parameters (altitudes and contours lines) of a sub-region of the DEM, without using a full decompression stage, and perhaps more importantly, without the need of the whole file. Our experimental results show that these two properties can significantly improve the performance of a mobile application that uses DEMs.

The rest of this paper is organized as follows: Section 2 describes the compression and decompression algorithms. Section 3 presents the algorithms used to obtain both, elevations and contours lines, without applying a full decompression process and without needing the whole DEM file. Section 4 presents experimental results obtained by applying our approach, and Section 5 outlines the conclusions and future works.

2 Compression and Decompression Algorithms

Compression algorithm. We will start this section with a series of definitions that are useful to describe our method. The altitude or elevation associated to each element of Record B in the DEM is denoted by $\alpha = elevation(x, y) \in [\alpha_{min}, \alpha_{max}]$ where α_{min} and α_{max} can be obtained from the DEM's header.

Now, let I be an image represented as a two-dimensional array that contains all the elevations associated with the DEM. The elevation points are described using a pair of discrete coordinates (see Equation 1).

$$I_{xy} = elevation(x, y), x \leq M, y \leq N \tag{1}$$

Where M is the number of columns and N is the number of rows of the original DEM.

An image I can be partitioned in a series of subsets E_α, where E_α is a set of pixels such that $E_\alpha = \{p|elevation(p) = \alpha\}$, $p \in I$. In consequence, an image can also be defined by the union of the E_α sets with $\alpha \in [\alpha_{min}, \alpha_{max}]$ as described by Equation 2.

$$I = \bigcup_{\alpha=\alpha_{\min}}^{\alpha_{\max}} E_\alpha = \left\{E_{\alpha_{\min}} \bigcup E_{\alpha_{\min}+1} \bigcup ... \bigcup E_{\alpha_{\max}-1} \bigcup E_{\alpha_{\max}}\right\}, \tag{2}$$

Now, we use Equation 2 to define G_k as the union of the subsets E_α for $\alpha \geq k$. From G_k, we further define G_k^b as shown in Equation 3. Each G_k^b can be seen as the projection of the landforms over the XY plane at elevation k, alternatively it can also be interpreted as a slice of the landforms at elevation k.

$$G_k^b(x, y) = \begin{cases} 1 & p_{xy} \in G_k \\ 0 & otherwise \end{cases} \tag{3}$$

Since the proposed compression algorithm uses RLE to compress subsets of the image that represent certain elevations, it can work over the E_α images as well as over the G_k^b images. However, we propose the use of the latter alternative, because one of our objectives is to provide a decompression scheme capable of retrieving a single point without the need of decompressing the whole DEM. It is possible, because the G_k^b images impose a structure to information that allows us to perform efficient searches. For instance, to obtain the elevation of a given point, we can start looking at any G_k^b image and if the elevation exist at that altitude k, then we know that the elevation is at least k and hence we can look at $G_{k\prime}^b$ with $k' > k$ for the actual altitude. Please note that decompressing a single point of the DEM is exactly the same as obtaining the elevation at that point. On the other hand, the E_α images do not provide information that may help to the search algorithm to find the elevation of a point.

Algorithm 1 *CompressDEM*

```
Input: The DEM D to be compressed
Output: The compressed DEM (the set A)
for k ← D.α_min to D.α_max do
    for j ← 0 to D.height do
        for i ← 0 to D.width do
            G_k^b[i, j] ← { 1  D[i, j] ≤ k
                          { 0  D[i, j] > k
    A_k ← ∅, f ← 0, d ← 0, t ← 0
    for j ← 0 to D.height do
        for i ← 0 to D.width do
            if G_k^b[i, j] ≠ d then
                A_k[t] ← f, f ← 1, t ← t + 1
                if d = 0 then d ← 1
                else d ← 0
            else
                f ← f + 1
    A_k[t] ← f
    A_k.lenght ← t + 1
return A
```

By using the previous definitions, we will describe our compression algorithm in detail (See Algorithm 1). It is important to remind the fact that the proposed algorithm works over integer data. It obtains a compressed representation of the contents of Record B. Both records A and C are directly stored in the resulting file without any modification. Additionally, it is necessary to write a header containing this variables: M, N, α_{min} and α_{max}. The Algorithm 1 is applied to the Record B and stored in a set of encoded arrays A_k with $k \geq \alpha_{min}$ and $k \leq \alpha_{max}$.

This algorithm is a losslessly compression approach, however it is possible to increase the compression rate, controlling the error by means of specifying the altitudes to be skipped. To do that, we introduce the parameters Δ and ϵ. The Δ is the similarity threshold, thus δ_k indicates how similar the binary images G_k^b and G_{k-1}^b are. If the value of δ_k is closer to 0, it means that images are more similar. So, in the compression algorithm, we consider that if $\delta_k \leq \Delta$ then we assume that G_k^b and G_{k-1}^b are equivalent images. In consequence, it is only necessary to generate one image and a flag to denote this equivalence. The second parameter ϵ allows us to restrict the induced loss by Δ, and it specifies the maximum number of ommited images. Although a high value of Δ is established, we only consider the maximum number of equivalent images, denoted by ϵ. In Algorithm 2, the compression approach using the above parameters is presented.

Decompression algorithm. The file generated by the proposed compression algorithm is composed of two main sections: the first one contains records A, C and the header that describes the compressed DEM, and the final section stores a total of $(\alpha_{\max} - \alpha_{\min} - 1)$ encoded arrays. The original binary values retrieved from any RLE encoded array (A_k) does not provide elevation values. This indicates if a landform occupies that point in the space (x, y, k). If a point

Algorithm 2 *CompressDEMv2*

Input: The DEM D to be compressed, Δ and ϵ
Output: The compressed DEM (the set A)

```
e ← 1
for k ← D.α_min to D.α_max do
    for j ← 0 to D.height do
        for i ← 0 to D.width do
            G^b_k[i, j] ← { 1  D[i, j] ≤ k
                          { 0  D[i, j] > k
    if k > α_min ∧ δ_k < Δ ∧ e < ε then
        A_k.lenght ← −1 , e ← e + 1
    else
        A_k ← ∅ , f ← 0 , d ← 0 , t ← 0
        for j ← 0 to D.height do
            for i ← 0 to D.width do
                if G^b_k[i, j] ≠ d then
                    A_k[t] ← f , f ← 1 , t ← t + 1 , d ← G^b_k[i, j]
                else
                    f ← f + 1
        A_k[t] ← f , A_k.lenght ← t + 1 , e ← 1
return A
```

$p \in I$ has an elevation equal to k, then the value of $G_i^b(p)$ with $i \leq k$ is equal to 1. When the binary image generation is carried out, the point $p\,(x, y)$ has an intensity equal to 0 in the range $(k+1, \alpha_{\max})$, and equal to 1 in $[\alpha_{\min}+1, k]$. Summing up, the elevation at point $p(x, y)$ is determined by Equation 4.

$$elevation\,(p\,(x, y)) = \alpha_{min} + \sum_{i=\alpha_{\min}+1}^{\alpha_{\max}} G_i^b\,(x, y) \tag{4}$$

3 Compressed DEM Management

We assume that it is not indispensable to apply a decompression method to obtain information about the elevation data. Additionally, we generate the contour layer, which has several map applications related to Geomatics area. These two operations are described in the next sections.

Access to elevation datasets. In some cases a DEM file requires high amounts of storage and main memory. This situation can be particularly challenging in the context of mobile devices that still have storage and memory limitations. This call for an efficient way to manage compressed DEM files, in particular, it would be desirable to have a set of tools that allow us to retrieve the elevation at some point $p\,(x, y)$, without decompressing the whole file. We have proposed Equation 4 to decompress the whole file; however, it requires to process all the RLE arrays, which is not necessary if we want to obtain the elevation at a single point. In this latter case, the processing time can be reduced by applying Equation 5.

$$elevation\,(p) = h, G_h^b(p) = 0 \wedge G_{h-1}^b(p) = 1, h \in [\alpha_{\min}, \alpha_{\max}] \tag{5}$$

Algorithm 3 *SimpleDecompressRegion*

```
Input: A set of encoded arrays A_k, α_min and α_max, the width W of DEM, the points p and q
       defining the region to decompress
Output: The decompressed DEM region (D)
D ← 0 , B ← 0
for k ← α_min to α_max do
    t ← A_k.length ;                 /* length is the number of transitions in A_k */
    if t > 0 then
        j ← 0 , m ← 0
        for i ← 0 to t − 1 do
            for l ← 1 to A_k[i] do
                j ← j + 1
                if (j/W ≥ p_y) ∧ (j/W ≥ q_y) ∧ (j mod W ≥ p_x) ∧ (j mod W ≤ q_x) then
                    B[m] ← i mod 2 , m ← m + 1
    D ← D + B
return D
```

Equation 5 induces a lineal-search in all the RLE encoded arrays, in the cases when the desired elevation is near $\alpha_{\min}$ the elevation is quickly obtained, otherwise almost all arrays (closer to $\alpha_{\max}$) will be processed. To improve the process a binary seach algorithm (or simply bin-search) is employed. Other tool would be the decompression of a DEM region; for example, to only decompress the fragment that is rendered in the screen of the mobile device. It allows us to handle several DEMs in a limited memory environment, obtaining only the required fragment of each one. This functionality is described in Algorithm 3.

Generating contours lines without decompression. Contour lines are very useful for many applications such as soil-landscape modeling, landform mapping, modelling of habitation and the reconstruction of prehistoric landscape. They allow us to show the shape of the land surface (topography) on a map in devices that do not have support for multiple colors. The method to obtain contour lines from a DEM is not complex; some commercial tools to handle cartographic data make this task. However, these systems require all elevation data sets, but it is possible to define an algorithm to generate the contour lines directly from the compressed DEM file. The single required parameter is the altitude interval to sample DEM (K), all additional information is available in the compressed data.

Step 1. Generate an $M \times N$ matrix and establish all values to 0, this matrix will represent the contour lines image $CL\,(x, y)$. The size of the matrix is obtained from the DEM compressed header.

Step 2. Take the first RLE encoded array and obtain the negated version of the original binary image using Equation 4. If the array is empty, then apply the same criteria as we defined in decompression algorithm.

Step 3. Compute the 8-connected contour of binary image. A point $p\,(x, y)$ with intensity equal to 1 is part of the contour, if the number of 8-neighbors with intensity equal to 0 (background pixels) is at least equal to 1, i.e, $N_8\,(p) \geq 1$.

Step 4. Remove redundant contour pixels.

Name	TRI	$\delta = 0.000$ $\epsilon = 1$	$\delta = 0.000$ $\epsilon = 5$	$\delta = 0.000$ $\epsilon = 9$	$\delta = 0.004$ $\epsilon = 1$	$\delta = 0.004$ $\epsilon = 5$	$\delta = 0.004$ $\epsilon = 9$	$\delta = 0.008$ $\epsilon = 1$	$\delta = 0.008$ $\epsilon = 5$	$\delta = 0.008$ $\epsilon = 9$
racine-w.dem	0.46	0.030	0.030	0.030	0.030	0.016	0.016	0.030	0.010	0.010
jacksonville-w.dem	0.48	0.023	0.023	0.023	0.023	0.022	0.022	0.023	0.018	0.017
racine-e.dem	0.68	0.033	0.033	0.033	0.033	0.024	0.024	0.033	0.015	0.014
valdosta-e.dem	0.99	0.040	0.040	0.040	0.040	0.037	0.037	0.040	0.034	0.034
aberdeen-e.dem	1.00	0.056	0.056	0.056	0.056	0.035	0.034	0.056	0.028	0.026
aberdeen-w.dem	1.61	0.078	0.078	0.078	0.078	0.054	0.053	0.078	0.038	0.035
fairmont-e.dem	1.81	0.081	0.081	0.081	0.081	0.067	0.067	0.081	0.045	0.045
valdosta-w.dem	1.94	0.076	0.076	0.076	0.076	0.073	0.073	0.076	0.057	0.057
tallahassee-w.dem	2.03	0.081	0.081	0.081	0.081	0.069	0.069	0.081	0.064	0.063
fairmont-w.dem	2.11	0.086	0.086	0.086	0.086	0.072	0.072	0.086	0.058	0.058
eagle_pass-e.dem	2.64	0.144	0.144	0.144	0.144	0.089	0.086	0.144	0.055	0.052
sacramento-w.dem	2.67	0.144	0.144	0.144	0.144	0.056	0.049	0.144	0.047	0.038
dalhart-e.dem	2.67	0.160	0.160	0.160	0.160	0.071	0.071	0.160	0.048	0.045
tallahassee-e.dem	2.74	0.108	0.108	0.108	0.108	0.091	0.091	0.108	0.067	0.067
waco-e.dem	3.00	0.129	0.129	0.129	0.129	0.114	0.114	0.129	0.079	0.078
macon-e.dem	3.42	0.135	0.135	0.135	0.135	0.114	0.114	0.135	0.076	0.076
oneill-w.dem	3.51	0.163	0.163	0.163	0.163	0.078	0.078	0.163	0.051	0.048
oneill-e.dem	4.04	0.182	0.182	0.182	0.182	0.088	0.087	0.182	0.054	0.052
idaho_falls-e.dem	4.45	0.228	0.228	0.228	0.228	0.081	0.069	0.228	0.059	0.047
paducah-e.dem	4.58	0.172	0.172	0.172	0.172	0.131	0.130	0.172	0.097	0.095
macon-w.dem	4.64	0.175	0.175	0.175	0.175	0.141	0.141	0.175	0.092	0.091
paducah-w.dem	4.75	0.192	0.192	0.192	0.192	0.148	0.147	0.192	0.097	0.095
waco-w.dem	5.04	0.236	0.236	0.236	0.236	0.124	0.123	0.236	0.079	0.077
nashville-w.dem	7.22	0.251	0.251	0.251	0.251	0.181	0.180	0.251	0.124	0.123
quebec-e.dem	7.56	0.335	0.335	0.335	0.335	0.117	0.109	0.335	0.082	0.069
nashville-e.dem	7.92	0.302	0.302	0.302	0.302	0.219	0.217	0.302	0.137	0.135
gadsden-w.dem	8.40	0.310	0.310	0.310	0.310	0.242	0.241	0.310	0.153	0.152
dalhart-w.dem	9.50	0.480	0.480	0.480	0.480	0.132	0.125	0.480	0.103	0.076
gadsden-e.dem	11.89	0.467	0.467	0.467	0.467	0.236	0.227	0.467	0.158	0.147
la_crosse-w.dem	13.10	0.415	0.415	0.415	0.415	0.271	0.271	0.415	0.163	0.160
la_crosse-e.dem	15.87	0.474	0.474	0.474	0.474	0.314	0.314	0.474	0.200	0.199
caliente-w.dem	18.34	0.940	0.940	0.940	0.940	0.224	0.183	0.940	0.197	0.124
yakima-e.dem	19.47	0.783	0.783	0.783	0.783	0.177	0.142	0.783	0.166	0.099
caliente-e.dem	20.60	1.004	1.004	1.004	1.004	0.227	0.178	1.004	0.211	0.128
idaho_falls-w.dem	22.41	1.042	1.042	1.042	1.042	0.229	0.144	1.042	0.212	0.124
baker-w.dem	26.60	1.160	1.160	1.160	1.160	0.266	0.216	1.160	0.245	0.149
ukiah-e.dem.dem	27.85	1.315	1.315	1.315	1.315	0.282	0.170	1.315	0.272	0.154
baker-e.dem	28.37	1.386	1.386	1.386	1.386	0.305	0.216	1.386	0.291	0.166
sacramento-e.dem	35.57	1.713	1.713	1.713	1.713	0.365	0.208	1.713	0.353	0.204
ukiah-w.dem	38.53	1.608	1.608	1.608	1.608	0.342	0.226	1.608	0.339	0.185
kalispell-e.dem	38.64	1.500	1.500	1.500	1.500	0.321	0.224	1.500	0.315	0.175
hailey-w.dem	39.73	1.763	1.763	1.763	1.763	0.374	0.257	1.763	0.363	0.212
hailey-e.dem	40.22	1.642	1.642	1.642	1.642	0.336	0.207	1.642	0.330	0.185
kalispell-w.dem	48.49	1.794	1.794	1.794	1.794	0.395	0.332	1.794	0.367	0.217
yakima-w.dem	59.15	2.239	2.239	2.239	2.239	0.467	0.372	2.239	0.449	0.260
Correlation		0.989	0.989	0.989	0.989	0.928	0.802	0.989	0.981	0.891

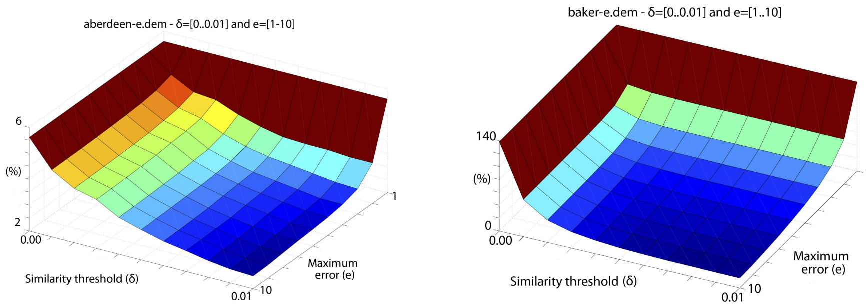
Table 1: *Compression values of the set of DEMs*

Step 5. Ignore the next K RLE encoded arrays and process $(K+1)-th$ element of the array from step 4.

4 Tests and Results

The tests consisted of applying the compression algorithm using diferent values for the parameters δ and ϵ with the original set of 24 DEMs proposed by [Franklin. 1995]. Additionally, the Terrain Roughness Index (TRI) for each one of DEMs is computed. In Table 1 the results of the compression are presented. Such table describes the compression values, in which there are not changes whether some compression parameters are minimum, that is, $\delta = 0$ or $\epsilon = 1$.

Now, we analyze a pair of particular cases, where the results are totally



(a) Aberdeen-e (b) Baker-e

Figure 1: *Results of the Aberdeen-e and Baker-e DEMs*

different, such as the Aberdeen (E) and the Baker (E) DEMs. In the first case, the analysis is presented in Figure 1a, in which the compression rate with minimal parameters is 0.055 with respect to the original size. When the compression parameters are increased, the rate factor is improved. In the second case, (see Figure 1b) when minimal compression parameters are established, the original size of DEM is increased in 1.40. However, the compression is improved by using $\delta = 0.001$ and $\epsilon = 2$, obtaining a compression rate of 0.40 regarding the original size. With high values of these parameters, it is possible to compress the original DEM up to 0.15 of the original size.

According to the experimental results, the compression parameters δ and ϵ should be established to $\delta = 0.004$ and $\epsilon = 4$ respectively, in order to obtain positive compression rate and minimize the loss factor. By using different values for these parameters, the compression rate is marginally increased, but the decompressed DEM is highly distorsioned with respect to the original values.

In all practical tests, the correlation values between the TRI and the compression rate is close to 1. In conclusion, it is possible to determine the compression rate, computing the TRI. By comparing the obtained results, a losslessly approach ($\delta = 0$ and $\epsilon = 1$) is applied; *iff* the TRI is less or equal to 23.9765. The numerical comparison is outlined in Table 1.

In Figure 2 the compression ratios that were obtained with different values of δ and ϵ parameters are plotted against their TRI ($\delta = \{0, 2, 4, 6, 8\} \times 10^{-3}$ and $\epsilon = \{1, 3, 5, 7, 9\}$). To make a comparative analysis of the results with the correlation values, a dashed line is drawn in Figure 2. According to the analysis, we consider that it is possible to statistically determine the values of δ and ϵ for obtaining a certain compression ratio (K_G). The procedure is as follows:

1. Determine the curves that describes the behavior of the algorithm for several

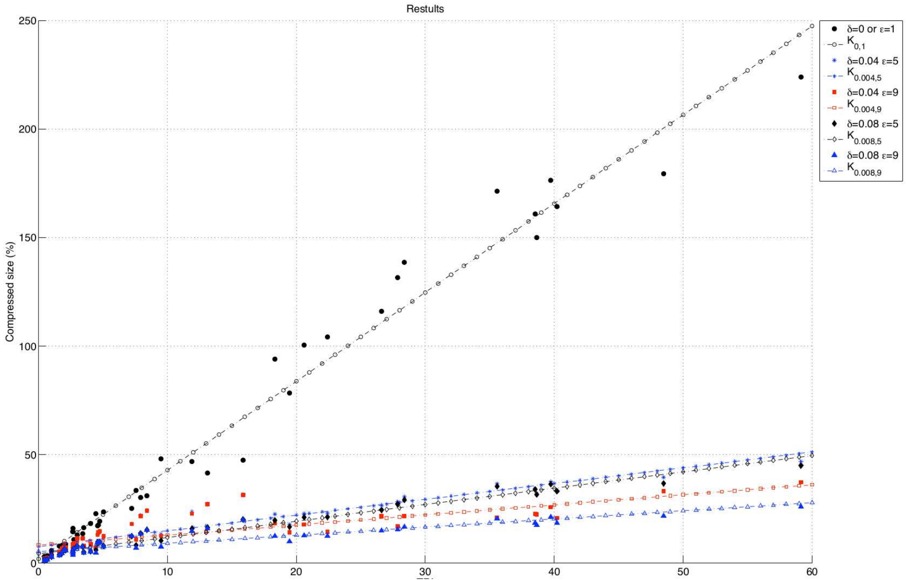


Figure 2: *TRI vs compression results*

pairs of δ and ϵ ($K_{\delta,\epsilon}$(TRI)).

2. Compute the Terrain Roughness Index $t = \text{TRI}(D)$ of the input DEM (D).
3. Evaluate $K_{\delta,\epsilon}(t)$ for all the pairs δ and ϵ.
4. Apply the compression algorithm with the parameters δ and ϵ such that $K_G - K_{\delta,\epsilon}(t) \geq 0$ and $K_G - K_{\delta,\epsilon}(t)$ being minimum.

5 Conclusions

In the present work, we develop a near-lossless compression algorithm using RLE encoding approach. Although the compression factor is not the same, we can obtain this factor applying the compression parameters (δ and ϵ), it is important to mention that encoded elevations can be directly read from the compressed DEM file, and the computation of contours lines is easy and fast. The most important part of the tests is the compression rate, which is up to 0.80. To make some enhancements, this algorithm should be adapted as a new format to store and describe digital elevations models. Future works are oriented towards studying other encoding formulations that allow us to increase the compression coefficient. On the other hand, an important characteristic is that the frequency of the DEM is not high, because the pixels do not present changes in their structure; the value of each pixel is correlated to their neighbor values. Due to this, we can apply this kind of compression. Therefore, we have considered the Terrain Roughness Index as the most important characteristic of DEMs in our method, because it is important to point out that DEMs are complex geo-images, which involve several properties of a certain environment. In addition, these properties reflect the semantics of the Digital Elevation Models.

Acknowledgments

This work was partially sponsored by the IPN, CONACyT under grant 106692, and SIP-IPN under grants 20130347, 20130345, and 20131215.

References

[Aràndiga et al. 2012] Francesc Aràndiga, Pep Mulet, and Vicent Renau. Lossless and near-lossless image compression based on multiresolution analysis. *Journal of Computational and Applied Mathematics*, 2012.

[Boehm. 1967] Barry W Boehm. Tabular representation of multivariate functions with applications to topographic modeling. In *Proceedings of the 1967 22nd national conference*, pages 403–415. ACM, 1967.

[Boucheron and Creusere. 2005] Laura E Boucheron and Charles D Creusere. Lossless wavelet-based compression of digital elevation maps for fast and efficient search and retrieval. *Geoscience and Remote Sensing, IEEE Transactions on*, 43(5):1210–1214, 2005.

[Chen and Lu. 2007] Renxi Chen and Xinhui Li. Dem compression based on integer wavelet transform. *Geo-spatial Information Science*, 10(2):133–136, 2007.

[Dutton. 1983] G Dutton. Efficient encoding of gridded surfaces, spatial algorithms for processing land data with a microcomputer. *Rept. Organisation of the Lincoln Institute for Land Policy, Cambridge, Massachusetts*, pages 23–62, 1983.

[Franklin. 1995] Wm Randolph Franklin. Compressing elevation data. In *Advances in Spatial Databases*, pages 385–404. Springer, 1995.

[Kidner and Smith. 2003] David B Kidner and Derek H Smith. Advances in the data compression of digital elevation models. *Computers & Geosciences*, 29(8):985–1002, 2003.

[Le et al. 2011] Nguyen Duy nh Le Hoang Son, Tran Van Huong, and Nguyen Huu Dien. A lossless effective method for the digital elevation model compression for fast retrieval problem. *International Journal of Computer Science and Network Security (IJCSNS)*, 11(6):35–44, 2011.

[Li and Gong. 2008] Yi Li and Jianhua Gong. Global terrain data organization and compression methods. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 37(B5):659–669, 2008.

[Marakovic. 1983] B Makarovic. A test on compression of digital terrain model data. *ITC journal*, (2):133–138, 1983.

[Said and Pearlman. 1996] Amir Said and William A Pearlman. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. *Circuits and Systems for Video Technology, IEEE Transactions on*, 6(3):243–250, 1996.

[Shaffer. 1990] Clifford A Shaffer. A full resolution elevation representation requiring three bits per pixel. In *Design and Implementation of Large Spatial Databases*, pages 45–64. Springer, 1990.

[Wiche. 1992] Gregg J Wiche. Application of digital elevation models to delineate drainage areas and compute hydrologic characteristics for sites in the james river basin, north dakota. *US Geological Survey water-supply paper*, 1992.