한국측량학회지, 제 28권 제 2호, 2010. 4 : pp. 265~271

DEM_Comp Software for Effective Compression of Large DEM Data Sets

대용량 DEM 데이터의 효율적 압축을 위한 DEM_Comp 소프트웨어 개발

Kang, In Gu[1] · Yun, Hong Sik[2] · Wei, Gwang Jae[3] · Lee, Dong Ha[4]
강인구 · 윤홍식 · 위광재 · 이동하

초 록

본 논문에서는 대용량의 수치표고모델(DEM) 데이터의 효율적인 압축을 위해 허프만 코딩과 Lempel-Ziv-Welch 압축방법을 기반으로 하는 새로운 DEM 압축 소프트웨어인 DEM_Comp를 개발하였다. DEM_Comp의 개발을 위해서 C++ 언어를 이용하였으며, 모든 Window 플랫폼에서 사용이 가능하도록 하였다. 개발된 소프트웨어의 성능을 평가하기 위해 다양한 지형의 형태를 가지는 DEM에 대해 압축을 수행하고, 출력파일의 용량에 따른 압축효율을 평가하였다. 최근 새로운 지형데이터 취득장비인 LiDAR와 SAR 등에 의해 고해상도의 DEM의 활용이 급격하게 증가하고 있어, 데이터의 저장용량과 전송대역폭을 감소시킬 수 있는 DEM 압축기술이 매우 유용하게 이용되고 있다. 일반적으로 데이터 압축기술은 i) 데이터 사이의 관계를 분석하고, ii) 분석 결과에 따라 압축 및 저장기술을 결정하는 2부분으로 구성되는데, DEM_Comp에서는 정규격자, Lempel-Ziv 압축방법, 허프만 코딩의 3단계 압축 알고리즘을 통해 DEM이 압축된다. DEM_Comp의 압축효율 실험 결과 전처리만 수행하였을 경우 지형의 기복과 상관없이 압축효율은 약 83% 정도를 나타내었지만, 3단계의 압축 알고리즘이 완료된 경우에는 압축효율이 97%까지 증가하는 것으로 나타났다. 이러한 수치는 일반적인 상업용 압축 소프트웨어들과 비교하여 약 14% 정도의 압축효율이 향상되었음을 나타낸다. 이에 따라 본 연구에서 개발된 DEM_Comp S/W를 이용하면 대용량의 고해상도 DEM의 관리, 저장, 배포를 보다 효율적으로 수행할 수 있을 것으로 판단된다.

핵심어 : DEM 압축 알고리즘; LiDAR; Lempel-Ziv 압축방법; 허프만 코딩

Abstract

This paper discusses a new software package, DEM_Comp, developed for effectively compressing large digital elevation model (DEM) data sets based on Lempel-Ziv-Welch (LZW) compression and Huffman coding. DEM_Comp was developed using the C++ language running on a Windows-series operating system. DEM_Comp was also tested on various test sites with different territorial attributes, and the results were evaluated. Recently, a high-resolution version of the DEM has been obtained using new equipment and the related technologies of LiDAR (LIght Detection And Radar) and SAR (Synthetic Aperture Radar). DEM compression is useful because it helps reduce the disk space or transmission bandwidth. Generally, data compression is divided into two processes: i) analyzing the relationships in the data and ii) deciding on the compression and storage methods. DEM_Comp was developed using a three-step compression algorithm applying a DEM with a regular grid, Lempel-Ziv compression, and Huffman coding. When pre-processing alone was used on high- and low-relief terrain, the efficiency was approximately 83%, but after completing all three steps of the algorithm, this increased to 97%. Compared with general commercial compression software, these results show approximately 14% better performance. DEM_Comp as developed in this research features a more efficient way of distributing, storing, and managing large high-resolution DEMs.

Keywords : DEM compression algorithm; LiDAR; Lempel-Ziv compression; Huffman coding

1) Member · Ph.D. candidate, Dept. of Geoinformatics, University of Seoul · Planning & Policy Division, National Geographic Information Institute (E-mail:kig777@korea.kr)
2) Member · Professor, School of Civil & Environmental Engineering, Sungkyunkwan University (E-mail:yoonhs@skku.edu)
3) Corresponding Author · Member · Ph.D. candidate, Dept. of constructional & environmental system engineering, Sungkyunkwan University (E-mail:gjwe@hist.co.kr)
4) Member · Adjunct Professor, College of Engineering, Sungkyunkwan University (E-mail:dhlee@geo.skku.ac.kr)

1. Introduction

A digital elevation model (DEM) is a topographic model that can express numerically the undulating characteristics of space. It uses a grid structure and gives the value of the ground elevation without buildings, but with trees rising above the surface. DEM data are an important component of spatial databases in geographic information systems (GIS), construction, environmental disaster modeling, and other similar applications, serving to represent and analyze the real world. Recently, the rapid development of remote sensing technologies has made it possible to create 3D position data with very high (centimeter-level) accuracy. High-resolution (1m × 1m) DEMs are currently being created using high-accuracy base data. With the expansion of 3D-imaging-related businesses (ubiquitous city networks, navigation, etc.), the need for high-resolution DEMs has increased.

A DEM represents topography on a fixed-size grid where all 3D coordinates must be stored. Therefore, as the area becomes larger, the file size increases greatly. For example, to store a 2 × 3 km area with a 1-m grid interval, the file size will be approximately 12 MB, but for a 20 × 30 km area, the file size increases to 1.2 GB. It is apparent that the file size is related to the size of the area being modeled. These high-resolution DEMs are cumbersome to distribute and store; therefore, a need exists for a faster and more effective method to store these large-volume DEMs. The DEM data volume is so huge that compression becomes necessary.

Data compression is useful for manipulating a large-volume DEM data set because it helps reduce the disk space, transmission, and rendering-time requirements. On the other hand, compressed data must be uncompressed to be viewed, and this inverse processing may be detrimental to some applications. A compression scheme for a DEM may require high-performance hardware for 3D viewing to enable the data to be decompressed fast enough to be viewed as they are being decompressed. Therefore, the design of DEM data-compression schemes involves trade-offs among several factors, including the degree of compression, the amount of distortion introduced, and the computational resources required to compress and uncompress the data.

The fundamental theory of data compression is provided by information theory and by rate-distortion theory. These fields of work were essentially started by Claude Shannon, who published some papers on these topics in the late 1940s and early 1950s. As for DEM data compression, Boehm (1967) was first to approach this issue in two steps. First, the relationships among the data were analyzed. And second, the compression and storing method was decided upon. Boehm presented a method called the "differential altitude grid," which used a 1-bit code to mark height differences and analyze relationships among the data. Kidner (1992) extended this code to 2-3 bits, which could produce a representation of the topography which was closer to reality. However, with only 2-3 bits, the representation of steep and complex areas was limited. Franklin (1995) presented a compression method using either image-compression technologies or commercial compression software. Image-compression methods have a limited number of expressible heights, whereas commercial compression software packages cannot define the relationships between heights or reduce repetition. Kidner and Smith (2003) presented an algorithm in which the relationships between heights were defined and a dynamic-length numeric-coding algorithm could then be used.

In this work, based on the techniques used to obtain the previously reported DEM compression results, the DEM_Comp software has been developed and coded using a three-step algorithm for compressing ASCII-formatted DEM data. In fact, the previously developed programs are relatively simple and can be implemented with standard utilities requiring only a few lines of code. In this paper, an effective DEM data-compression program, DEM_Comp, is discussed, which is based on the LZW algorithm which offers many advantages and has become a trend in data compression. DEM_Comp is remarkably fast for an adaptive scheme and exceeds most other software packages in compression performance. It is apparent that the algorithm is very straightforward, making it ideal for a pure hardware implementation.

2. DEM Data-Compression Algorithm

In this study, a three-step algorithm has been used for DEM data compression. The steps consist of pre-processing using a

DEM with a regular grid, Lempel-Ziv compression, and Huffman coding. A flowchart of the coding method is shown in Fig. 1.

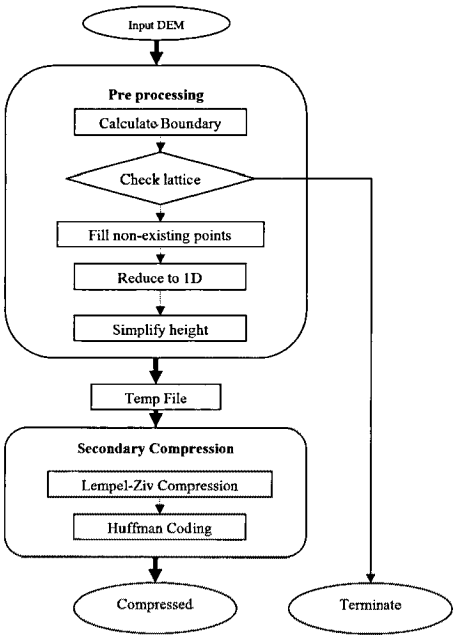

Fig. 1. Flowchart of DEM compression

2.1 Pre-processing

The pre-processing step consists of two stages. The first stage is to reduce the three-dimensional X-, Y-, and Z-coordinates to a single dimension based on the Z-axis. The second is to reduce the data volume by analyzing the height relationships between the grid points.

2.1.1 Dimension Reduction

DEM data consist of sets of X-, Y-, and Z-dimensional coordinates. The X- and Y-coordinates represent the position of the points in a plane and are regularly spaced. Therefore, if the upper left-hand point of the DEM and the distance to each point are known, it is possible to find an expression for the position of any point.

In this work, the X- and Y-coordinates of all points, with the exception of the first, were excluded, and each point was reduced to a single dimension containing only a Z-elevation value. Previous studies have used similar methods for DEMs with a fixed grid. However, in this study, the algorithm as developed was designed to exclude errors and to be applied to DEMs with irregularly shaped boundaries. Specifics of the dimension-reduction algorithm are shown in Fig. 2. First, the file was scanned, then the DEM was wrapped in a virtual square, and the lower left-hand X and Y points (Min(X) and Min(Y)) and the length of the X- and Y-axes (Length(X) and Length(Y)) were calculated. Unless the DEM is square with a side parallel to the axis, some points will be missing.

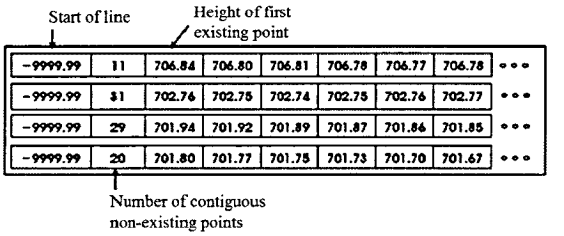

Fig. 2. Dimension-reduction process

To identify these points, an identifier with a certain value has been used, followed by a field expressing the number of missing points. An identifier, as shown in Fig. 3, is composed of an Identifier and a Jump field. Each Identifier represents a non-existing point, and the Jump field shows the number of contiguous non-existing points.

IDENTIFIER	JUMP(num)

Fig. 3. Structure of an identifier

The equation for restoring the X- and Y-coordinates is as follows:

$$\begin{aligned} X_z &= X(base) + (Count(Z) / X(legnth)) \\ X_z &= X(base) + (Count(Z) / X(length)) \\ Y_z &= Y(base) + (Count(Z) / X(length)) \end{aligned} \quad (1)$$

where

X_z = Restored X-coordinate
Y_z = Restored Y-coordinate
$X(base)$ = Lower left-hand X-coordinate
$Y(base)$ = Lower left-hand Y-coordinate

$Count(Z)$ = Number of elevations read

$X(length)$ = Distance between the leftmost and rightmost X-values

2.1.2 Simplification of the Height Value

Because a DEM represents a real surface, elevation points which are physically close have similar heights. After dimension reduction, the DEM contains only height information. Using the current point as the base, the differential between the current and next points is written into the file.

There are two benefits of this process one is that the process minimizes the number of bytes used, and the second is that the amount of repetition in the data increases. This process transforms the values representing similar heights, which makes the Lempel-Ziv compression and Huffman coding methods more effective.

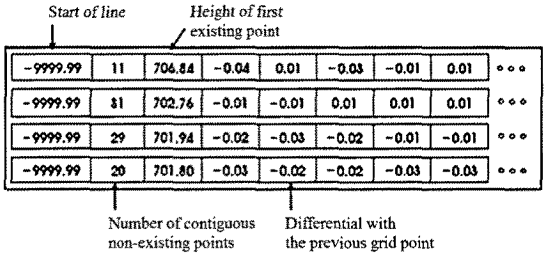

Fig. 4. Elevation simplification

2.2 Lempel-Ziv Compression

The Lempel-Ziv compression algorithm was first presented by Abraham Lempel and Jacob Ziv (1977). It uses a method called the "sliding window", which sequentially finds and reduces repeated data within a search range.

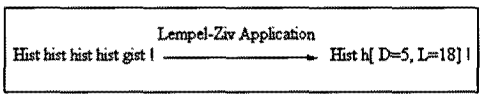

Fig. 5. Example of Lempel-Ziv application

The Lempel-Ziv algorithm provides simple and quick compression of data. The algorithm works by replacing a string of characters with single codes, called "tokens". Each time the algorithm recognizes a new string, it outputs the string and adds it to a table or directory. The next time it encounters that string, it outputs only the new code from the table. The output of a single code instead of a string of codes means that the data has been compressed or shortened, enabling efficient data transmission via the Internet. In this work, Lempel-Ziv compression was used after the pre-processing step.

2.3 Huffman Coding

Huffman coding is a type of transposition code in which high-frequency values are given short codes and low-frequency values are given longer codes. Phone numbers are an example when making a long-distance call, more numbers have to be pressed than when calling locally. This is because local calls are more frequently made. Emergency numbers are shorter because there is a need for high-frequency numbers which can be easily pressed.

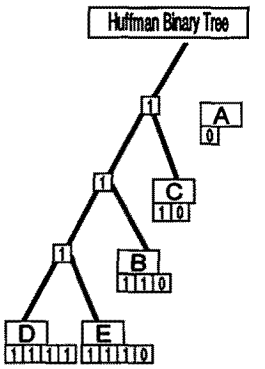

Fig. 6. Example of a Huffman binary tree

Huffman coding is performed according to the criteria just described: the data are analyzed, and the highest-frequency segments are then given the highest weights and the shortest codes. By this process,high efficiency is obtained. For example, the string "ABCDEABCDEABCDAAAAAACCC" can be compressed using only five characters. The counted frequencies are 9 times A, 3 times B, 6 times C, 3 times B, and 2 times E. Therefore, A is given the shortest bit string and E the longest. By this process, the total size of the file is reduced.

Fig. 6 shows how the characters are stored. A is coded as 0, C as 110, B as 1110, E as 11110, and D as 11111, which provides a high compression rate and good effectiveness when many repetitions occur. In particular, through the pre-process-

ing used in this research, the DEM processing algorithm gains an even higher compression rate and great efficiency in data storage.

3. Development of DEM_Comp

The DEM_Comp software was developed using the C++ language based on the MS Windows GUI. This software is available for Windows operating systems (Windows 9x, 2000, and XP), which are the most widely used multi-tasking environment for PCs. Fig. 7 shows the user-interface window of the DEM_Comp software. By clicking "OPEN" in the main menu, a *.dem file can be read from a specified directory.

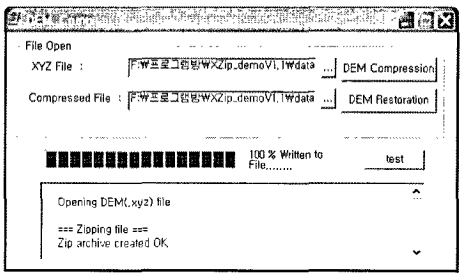

Fig. 7. Screen shot of main window

By clicking the "DEM Compression" button, the file can be compressed, and by clicking the "DEM Restorati" button, the compressed file can be saved. The bottom portion of the window shows a brief summary of file opening and zip file processing status.

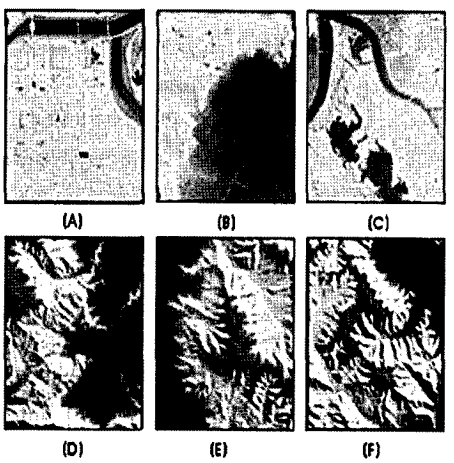

Fig. 8. DEMs of high- and low-relief areas

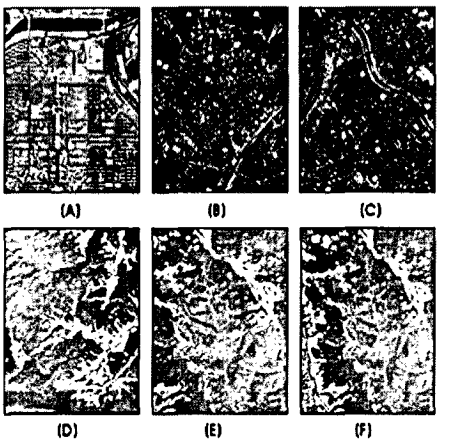

Fig. 9. Aerial photographs of high-undulation and low-undulation areas

DEM_Comp was tested on several different data sets to evaluate its efficiency. To evaluate compression efficiency in relation to the topographical attributes of DEMs, an experiment was performed on different terrains consisting of high- and low-relief areas, as shown in Figs. 8 and 9. The results of compressing these data sets using several software packages are presented in Figs. 10 and 11.

The results obtained from DEM_Comp were compared with those from the commercial compression tools "pkzip.exe" provided by PKWARE Ltd. (zlib Group, 2010) and "rar.exe" provided by RARLAB (RARLAB, 2010), which showed an approximately ±3% difference from the DEM_Comp results. In high-relief areas, DEM_Comp gave better results than those from "pkzip.exe" and "rar.exe." From these results, it was clear that the compression rates were similar. Therefore, it was shown that dimension reduction has a large effect on reducing data volume.

To prove the efficiency of the pre-processing method, the graph shown in Fig. 10 was created, based on one of the experimental sites and showing the frequency of the elevation relief between physically close points. Fig. 10 shows that the differences between the close points are small, with over 60% of the elevation relief between close points being in the range 0 - 0.09, which means that subsequent compression processes will have a large effect.

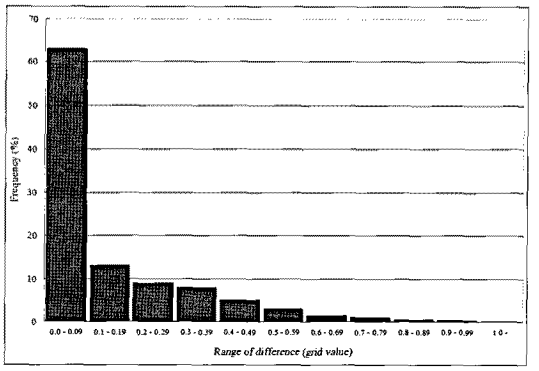

Fig. 10. Frequency differences between close grid points

Fig. 11 represents a size comparison of files compressed using Pkzip, rar, pre-processing alone, and DEM_Comp. It is evident that the DEM_Comp algorithm developed in this study shows higher compression ability than the other tools.

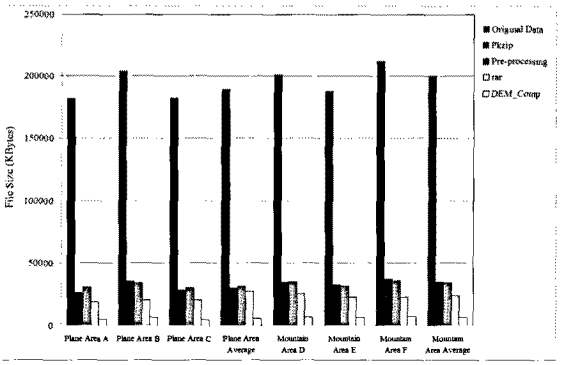

Fig. 11. Size comparison of files compressed using Pkzip, pre-processing alone, rar, and DEM_Comp

At the end of the compression process, the DEM input data set was reduced to 5 - 7MB. In high- and low-relief areas, DEM_Comp efficiencies were 97% and 96.6% respectively, as shown in Fig. 12. Usually commercial software packages use 4 - 5 bytes to express an elevation in a DEM. DEM_Comp uses an average of 1 byte in high-relief areas and 0.9 byte in low-relief areas. Therefore, it can be concluded that by understanding the structure of the DEM and using its characteristics, data can be compressed with a higher efficiency than that obtainable using commercial software, and the efficiency is even higher when archiving topographies with low-relief areas.

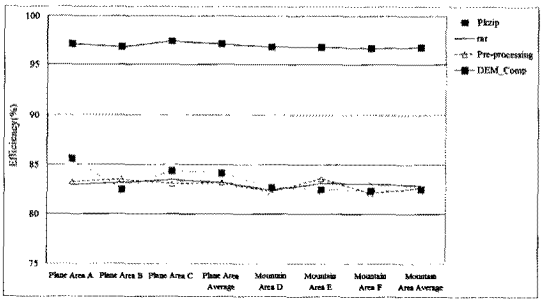

Fig. 12. Efficiency comparison of DEM_Comp, pre-processing alone, rar, and Pkzip

DEM_Comp reduces the size of its output files using adaptive Lempel-Ziv coding. Whenever possible, each file is replaced by one with the extension *.Z, while keeping the same ownership modes, access rights, and modification times. Compressed files can be restored to their original form using uncompress or zcat. Compression is performed using the modified Lempel-Ziv algorithm as given in Welch (1984).

After the bit limit is reached, compress periodically checks the compression ratio. If it is increasing, compress continues to use the existing code dictionary. However, if the compression ratio decreases, compress discards the table of substrings and rebuilds it from scratch. This enables the algorithm to adapt to the next "block" of the file.

4. Conclusions

This paper has described the development of the DEM_Comp software for data compression of large high-resolution DEMs, which offers a more efficient method of storing, transmitting, and using these data.

DEM_Comp is composed of a pre-processing step, a Lempel-Ziv compression step, and a Huffman coding step. DEM_Comp was tested on both high- and low-relief sites to determine its efficiency for areas with different topographies. The software was tested and evaluated on various sites with different territorial attributes. According to the results, the use of pre-processing on high- and low-relief areas gave an efficiency of approximately 83%, but after applying all three steps of the DEM_Comp algorithm, this efficiency increased to 97%. The file size obtained with DEM_Comp was 4 - 5 times smaller than that obtained from pkzip, rar, and pre-pro-

cessing alone, and DEM_Comp also had higher efficiency in flat areas.

Unlike earlier algorithms which needed a rectangular DEM, DEM_Comp can analyze non-rectangular DEMs and was also designed to store the exact value after the decimal point. DEM_Comp as presented in this study could be efficiently used in 3D viewing software applications such as Google Earth and Virtual Explorer.

References

Boehm, B. W. (1967), Tabular representations of multivariate functions with applications to topographic modeling, In: *Proceedings, 22nd ACM National Conference*, Washington DC, pp. 403-415.

Franklin, W. R. (1995), Compressing elevation data, In: Advances in Spatial Databases: Proceedings of the Fourth International Symposium on Large Spatial Databases (SSD 95), Portland, ME, August, *Lecture Notes in Computer Science*, Vol. 951. Springer, Berlin, pp. 385-404.

Jacob, Z. and Lempel, A. (1977), A universal algorithm for sequential data compression, *IEEE Transactions on Information Theory*, Vol. 23, No. 3, pp. 337-343.

Kidner, D. B. and Smith, D. H. (1992), Compression of digital elevation models by Huffman coding, *Computers & Geosciences*, Vol. 18, No. 8, pp. 1013-1034.

Kidner, D. B. and Smith, D. H. (2003), Advances in data compression of digital elevation models, *Computers & Geosciences*, Vol. 29, No. 8, pp. 985- 1002.

Welch, T. A. (1984), A technique for high-performance data compression, *IEEE Computer*, Vol. 17, No. 6, pp. 8-19.

RARLAB (2010), rar.exe, http://www.rarsoft.com

zlib Group (2010), pkzip.exe, http://www.zlib.net

(접수일 2010. 03. 30, 심사일 2010. 04. 19, 심사완료일 2010. 04. 24)