MVA '96 IAPR Workshop on Machine Vision Applications, November. 12-14, 1996, Tokyo, Japan

A Practical Data Compression System for Map Images

Yining Ou, Jianqi Huang, Yinglin Yu
Department of Radio Engineering
South China University of Technology
Guangzhou, 510641, P. R. China
Email: ecylyu@scut.edu.cn

Abstract

According to the characteristics of the map images, a practical data compression system was introduced in this paper. The main idea of this system is to segment the map image into two images: long-line image and the rest short-line image. For the first one, we use the cubic spline interpolation method to compress it; for the other image, we use arithmetic coding. It is proved by the experiments that the system is practical, simple and has increased the compression ratio by about or more than 200 percent for most of the map images, comparing to the traditional arithmetic coding method.

1 Introduction

As the images always require large amount of data storage, it is very important to find an efficient data compression method to reduce the storage memory or the transmission time.

The traditional approaches for compressing binary images are mainly based on the entropy and noiseless, such as the Huffman code, run-length code and the arithmetic code [1], which can't obtain high compression ratio. There have also been several good papers from the standpoint of facsimile data compression of a document page [2],[3] , but in this paper we will consider the compression method mainly for the purpose of reducing the storage memory.

As the map image is a mixture of long lines, characters and other complex images, we use a hybrid coding according to the different images in this paper. It turns out to have better performance than the traditional methods. The following sections describe the whole compressing system in detail.

In the following sections, we suppose that the background pixels are '0's, the objects are '1's, and ,the scanned image is saved in a reduced-data form (each pixel is one bit) .

2 System Overview

The block diagram of Fig. 1 describes the basic elements of the data compression system for map images. In the compressing process, the scanned image was segmented into long-line image and the residue short-line image after the pre-processing and object extraction. For the long-line image, we first sample some points to represent the lines then use arithmetic coding to encode it; for the residue, it is encoded by arithmetic coding. In the reconstruction process, we use cubic spline interpolation to finish the curve fitting after arithmetic decoding for the compressed long-line image, then combine it with the other decoded image to form the whole reconstruction image.

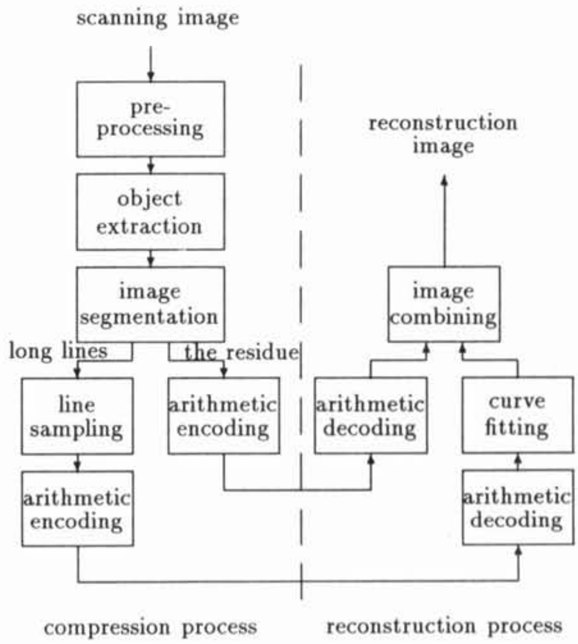

Figure 1: The block diagram of the compressing system

3 Pre-processing

The pre-processing includes three process: image smoothing, filtering and thinning. The image smoothing and filtering is mainly to delete the spurs, holes and other noise of the scanned image so that the image segmentation can be more accurate. The thinning processing is to turn the thick lines into single-pixel lines, so that we can follow the lines and extract them from the source image. In this paper, we use the two templates shown in Fig.2(a) and Fig.2(b) to delete the spurs and the holes respectively. To delete the lone noise pixels, we use the improved median filtering especially for the binary image.

We use a morphology-based thinning algorithm in this system which can be described as the following:

Let X is an image matrix, for the square meshes, we use L and L' which are shown in Fig. 3 and their $90°$'s turnings.

0	0	0
0	1	0
X	X	X

(a)

X	1	X
1	0	1
X	X	X

(b)

Figure 2: The templates for image smoothing:(a)template for deleting spurs,(b) template for deleting holes,'X' is an arbitary pixel.

$$L = \begin{array}{|c|c|c|}\hline 1 & X & 0 \\ \hline 1 & X & 0 \\ \hline 1 & X & X \\ \hline \end{array} \quad L' = \begin{array}{|c|c|c|}\hline X & 1 & X \\ \hline 1 & 1 & 0 \\ \hline X & 0 & 0 \\ \hline \end{array}$$

Figure 3: Square meshes for image thinning,'X' is an arbitary pixel.

The thinning algorithm is:

Step 1: $Y \leftarrow X \cdot L \cdot L'$;

Step 2: Turn the L and L' by $90°$, $Y \leftarrow Y \cdot L \cdot L'$, until they have turn $360°$

Step 3: If $X = Y$, then the algorithm ends, else go to Step 4;

Step 4: $X \leftarrow Y$, return to Step 2.

4 Object Extraction

To segment the image correctly, the objects which have many pixels must be extracted as the short-line image first. The steps of it can be described as the following:

Step 1: Begin the pixel extracting;

Step 2: Check one of the inside pixels which have not been scanned, if it is '0', return to Step 1;

Step 3: If the top, bottom, left or the right pixel of it is '0', return to Step 1;

Step 4: Else save the pixel in the short-line image and delete it;

Step 5: If all the inside pixels have been scanned, go to Step 6, else go to Step 1;

Step 6: End the pixel extracting.

After the processing, all the pixels in the big objects have been saved in the short-line image.

5 Image Segmentation

The image segmentation is an important part of the compressing system. In this process, we use edge following algorithm to extract the long lines and save the residue in the short-line image. The image segmentation algorithm is :

Step 1: Delete the pixels which have been followed, search the first pixels to be followed, and save the coordinates of it at the same time;

Step 2: Search the next 8-neighbor pixel which is '1' and save its coordinates until the present pixel is an end-point;

Step 3: If it is a break-point, search the next break-point, go to Step 2;

Step 4: Else return the length of the line L and compare it with the threshold T, if $L > T$ save the line in the long-line image, else save it in the short-line image;

Step 5: If the whole original image has not been scanned, go to Step 1; else end the program.

Thus the original image is separated into two images.

6 Line Sampling and Curve Fitting

For the long-line image, we sample some special points to represent the lines for the purpose of data compression. In this paper, we use the enhanced strip algorithm (ESA)given by J. Roberge for the line sampling. Please look up [4] for the details of the ESA.

In the reconstruction process, we use cubic spline interpolation to make the curve fitting from the sampled points. The detail of the algorithm is given in [5].

It proved by the experiments that the line sampling and curve fitting algorithms can achieve a high degree of data reduction but producing an accurate representation for even the most complex curves.

7 Arithmetic Encoding and Arithmetic Decoding

To compress the sampled points and the residue image, we use the arithmetic coding of which an excellent review was presented by Langdon [6]. Details of the implementation are discussed by Witten et al.[7].The universal data compression system contains two parts: modeling part and coding part. In the modeling part, we use a 10-pixel template to set up the conditioning classes. The model we use in the system is an adaptive finite state machine (FSM) model which is more efficient than the stationary FSM. The arithmetic coding is noiseless which can reduce the distortion of the reconstruction image.

8 Image Combining

When the long-line image has been fitted and the short-line image has been decoded, we can combine the two images and form the whole reconstruction image. It is implemented only by adding the two reconstructed images.

9 Experimental Results

We use four different map images (Images 1 to 4, shown in Fig.4 - 7) to test the data compression system , all the images are 1024(1024 and input by the scanner. In the experiments, the threshold of long lines T used in image segmentation is 40; the deviation DEV and extention factor EF used in line sampling is 1.0 and 6.0 respectively. The results are shown in Table 1. The reconstruction images are shown in Fig.8 - 11.

Table 1: Comparision of compression ratios with traditional arithmetic coding

Scanned Image	Size (Bytes)	Traditional Encoding	After Segmentation
Image 1	131,134	26,094	11,105
Cr	1.00	5.03	11.81
Image 2	131,134	28,358	13,155
Cr	1.00	4.62	9.97
Image 3	131,134	15,125	8,549
Cr	1.00	8.67	15.34
Image 4	131,134	24,461	13,197
Cr	1.00	5.36	9.94

Figure 4: Original scanned image 1 (1024 × 1024)

10 Conclusion

A practical data compression system for map images has been introduced in this paper. The com-

Figure 5: Original scanned image 2 (1024 × 1024)

pressing system involves segmentation of a map image into a long-line image, which is encoded by line sampling; and into a short-line image, which is encoded by arithmetic coding. The experimental results indicates that the system is practical, simple and can improve the compression ratio by about or more than 200 percent for most of the map images , comparing with the traditional arithmetic coding method. The system is especially efficient for those images with many long lines.

References

[1] G. G. Langdon and J. Rissanen, "Compression of black-white images with arithmetic coding, " *IEEE Trans. Commun.*, vol. COM-29, no. 6, pp. 858-867, June 1981.

[2] W. K. Pratt, et al., "Combined symbol matching facsimile data compression system," *Proc. IEEE*, vol. 68, no. 7, pp. 786-796, July 1980.

[3] O. Johnsen, et al., "Coding of two-level pictures by pattern matching and substitution," *The Bell System Technical Journal*, vol. 62, no. 8, pp. 2513-2545, October 1983.

[4] J. Roberge, "A data reduction algorithm for planar curves," *Comput. Graphics Images Process.* , vol. 29, pp. 168-195, 1985.

[5] H. C. Hsieh, et al., " Virtual knot technique for curve fitting of rapidly varying data," *Computer Aided Geometric Design*, vol. 11, pp. 71-95, 1994.

[6] G. G. Langdon , " An introduction to arithmetic coding," *IBM J. Res.*,vol. 28, pp. 135-149, Dec.1984.

[7] I. H. Witten, et al., " Arithmetic coding for data compression, " *Commun. ACM*, vol. 30, no. 6 pp. 520-540, June 1987.

Figure 6: Original scanned image 3 (1024 × 1024)

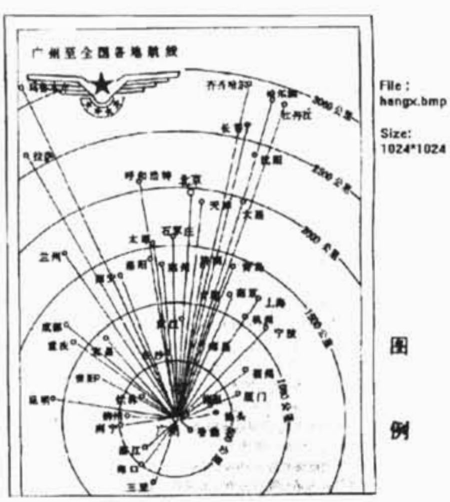

Figure 7: Original scanned image 4 (1024 × 1024)

Figure 8: Reconstruction image 1 (1024 × 1024)

Figure 9: Reconstruction image 2 (1024 × 1024)

Figure 10: Reconstruction image 3 (1024 × 1024)

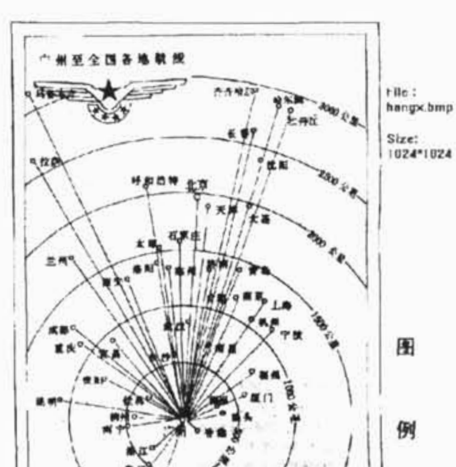

Figure 11: Reconstruction image 4 (1024 × 1024)