COMPRESSION AND COMPACTION OF BINARY RASTER IMAGES

M.A. COMEAU, *Canada Land Data Systems, Environment Canada, Ottawa, and*
E. HOLBAEK-HANSSEN, *Norwegian Computing Center, Oslo, Norway*

INTRODUCTION

THE INCREASING DEMAND for digital data in computerized geographic information systems has led to the introduction of systems for automatic data capture. Commonly known as raster scanning systems, these systems make use of optical scanners as the primary mode of input. Instead of manually digitizing the data sources, the original map document is converted, by hardware, into an array of numbers representing the positional distribution of optical density within the image. Hence the map document is represented as a rectangular grid, or raster. For each raster element, or pixel, the grey-level intensity of the corresponding area is known. By defining a light/dark threshold, the grey-level image thus produced can readily be converted into a binary raster image depicting only the presence or absence of lines. The major focus of this paper is in the storage and processing of these binary raster images as it applies to cartographic or geographic information processing applications.

Binary raster images resulting from the scan of conventional map reproduction material are bulky and contain much superfluous information. Unlike manual digitizing, the total area of the map document is converted to digital form, including blank areas, marks and scratches and other unwanted information normally disregarded by the digitizing operator. The computational task of extracting the wanted information from these digital images and of performing the raster-to-vector conversion required for most vector-based geographic information systems is non-trivial. Hence the efficient storage and manipulation of these binary raster images is an important consideration in the design of any raster scanning system.

This paper is based on recent studies performed at the Canada Land Data Systems (CLDS), Environment Canada in cooperation with the Norwegian Computing Center. These studies, which focus primarily on the efficiency of existing algorithms, were performed in preparation for the replacement of a somewhat antiquated but well-proven raster drum scanner. Although the upgrading of the

scanning hardware itself should increase the efficiency of the operation, it is also recognized that considerable effort must be devoted to increasing the subsequent software efficiency to cope with the increased volumes of data produced by current-day scanners.

CONVENTIONAL DATA COMPRESSION TECHNIQUES

The result of the raster scanning process is a large array of grey-levels in which each pixel is typically assigned a value ranging from 0 to 255 (one byte per pixel). The raster representation of a typical NTS map sheet (60 cm by 80 cm) requires of the order of 192 million pixels when scanned at a grid resolution of 50 microns (0.002 inches) of 768 million pixels at 25 micron (0.001 inches) resolution. To put this in perspective, the grey-level representation of this image at 50 micron resolution requires approximately 6 times the storage capacity of a full-scene Landsat MSS image containing all four bandwidths. Fortunately, the greater part of this data is redundant and the storage requirements of the image can be reduced significantly through conventional data compression techniques. The text that follows describes, in chronological order, the most commonly used techniques to reduce the volume of these raster images. These three common techniques include thresholding, run-length coding and skeletonization. Other compression techniques which will not be described here include information-theoretic methods (Rosenfeld and Kak 1983) and hybrid raster-vector techniques (Peuquet 1982).

Thresholding

The first step in reducing the storage requirements of a grey-level image is to convert it to a binary image depicting only the absence (white areas) or presence (black areas) of lines or areas. This operation, commonly known as *thresholding*, reduces the storage requirements of the image to a single bit per pixel, thereby reducing the overall storage requirements by a factor of eight if the grey-level values were contained in a single byte. Thresholding simply requires that all pixels below a given light/dark threshold value be re-assigned to zeroes (0s) and all others above this threshold to ones (1s). The selection of the proper light/dark threshold value is normally made on the basis of a grey-level histogram which shows the absolute and relative frequency of each grey-level value for a particular image (or subset of a particular image). Alternatively, for constant source material, the light/dark threshold could be determined on a trial and error basis.

Thresholding is a trivial computational task which should be performed as an integral part of the scan process by the scanning hardware. Ideally, one should be able to scan a small sample portion of the document in continuous-tone grey-level in order to determine, through histogram analysis, the appropriate light/dark threshold. The scanning operation could then be re-initiated with this threshold value fed in to trigger automatic thresholding of the image during the scanning process. Some scanning systems still output the entire grey-level image which must then be temporarily stored on some external device (normally magnetic tape) and subsequently re-read by thresholding software. The time required to simply write and read this enormous volume of data can significantly impair the overall scanning operation.

Run-Length Coding

The binary image produced from thresholding the grey-level image is significantly reduced in volume but still contains large clusters of zeroes (0s) for line-art images along with large clusters of ones (1s) for solid area images. Further data reductions can be achieved by taking advantage of these clusters. Run-length coding is a popular data reduction technique used in image analysis and is especially advantageous when used with binary images common to cartography. Using this technique, the image is reduced row by row (or column by column) by identifying long runs of 0s or 1s which occur throughout the image. In its simplest form, this technique encodes a two integer value for each run, the first representing the length of the run and the second representing the pixel value of the run. For binary images the pixel value of the run requires only one bit of storage whereas the storage requirements for the length of the run are often dependent on the total size of the image in the direction of the run. If the storage allocated to encode the length of the run is large enough to accommodate a run across the entire length of the image, the value of the run need not be encoded as the only two possible values (0 and 1) will alternate. Another coding scheme used for binary images encodes only runs of 1s. In this approach, the 2-integer sequences represent the row (or column) position where the run starts and the length of the run.

The type of run-length coding used in a particular scanning system is very much dependent on the record formats and data structures used to store the image and the type of accesses done to retrieve all or part of the image. These in turn have often been dependent on the manner in which the actual scanning is performed. Systems using drum scanners have a tendency to store the image on a column by column basis, run-length coded in the circumferential scan direction. Scanning systems employing flatbed scanners have a tendency to store swaths of the image, run-length coded in the direction of the sweeping and perpendicular to the direction of the scan. Likewise, drum scanning systems seem to favour the scan-line approach to the raster-to-vector conversion requiring column by column access to the image whereas flatbed scanning systems tend to favour the patch by patch approach requiring patchwise access to the image.

Run-length coding serves well to reduce the storage requirements of binary raster images with no loss of data. Unfortunately, run-length coding in most cases is unidirectional and does not offer any data compression in the direction running perpendicular to the runs. The other drawback of run-length coding is that most processing algorithms require that the run-length codes be expanded to full raster format prior to subsequent processing. Hence run-length coding in most cases is only advantageous to reduce storage requirements and does not generally reduce processing time.

Skeletonization

Skeletonization is a further means of reducing the data contents of a binary raster image. In this process all black pixels (1s) which are not required to preserve the topography (presence or absence of lines or areas) or topology (relationships between lines or areas) are removed (changed to 0s) from the image. In the case of

solid area images, this results in the removal of all interior black pixels and the retention of only those pixels which define the outline of areas. In the case of line-art images, lines are reduced to single pixel width where each pixel represents either an inner-line-point, an end-of-line or a node.

Skeletonization of binary line-art images is a very resourceful computational task primarily due to the large number of pixel neighbourhood analyses required and the iterative nature of the process. The skeleton of a line-art image is normally derived through a process known as line thinning, a process which involves the successive erosion of the line edges, from all directions, until the lines have a single pixel width. Perhaps one of the most efficient line thinning algorithms is that derived as part of the ARLIP software package in Bonn, West Germany (Kreifelts 1977, and Woetzel 1978). The line thinning process in this algorithm is reduced to a systematic inspection of the 3×3 neighbourhood of all black picture elements to determine, through pre-defined table look-up, whether or not the pixel can be removed without breaking connectivity of lines or creating 'holes' in the original linework. The raster is divided into four partitions, so that pixels from one partition cannot be neighbours of each other. This, combined with a quasi-parallel application of the operator allows for multiple logical passes of the thinning process in one physical pass of the image. The algorithm also yields a 'marked' skeleton whereby nodes or line junctions, inner-line-points and ends-of-lines are separately identified as a by-product of the thinning process.

The ARLIP algorithms (or variations of) for line thinning and raster-to-vector conversion are quite common in drum raster scanning systems. Unfortunately, the algorithms expect strip by strip access to the binary raster image and are not easily adapted to the patch by patch access dictated by flatbed scanning systems. Furthermore, the algorithms, as originally conceived, require that run-length coded strips be fully expanded to their raster representation prior to processing.

A NEW APPROACH TO DATA COMPACTION

At this point we deviate from the conventional techniques of data compression to introduce a new approach to binary raster data compaction. Specifically, we look for an algorithm which could serve to reduce the resolution of a binary line-art image with little or no loss of information. Such an algorithm, if applied first to the binary raster image, could reduce the storage requirements by a factor of four on each pass and significantly reduce the computational time required for run-length coding (and decoding) and skeletonization.

The general approach to reducing the resolution of a binary raster image is to map a 2×2 pixel quadruple into a single pixel whose value is derived by some pre-defined decision criteria. The decision is simple when the 4 pixels in the quadruple are all zeroes or all ones; the difficulty is in choosing the new value of a quadruple with mixed zeroes and ones. On one extreme, one could set all mixed quadruples to ones, thereby conserving topography at the loss of possibly many topological relationships. Conversely, one could set all mixed quadruples to zeroes, a process which runs the risk of destroying topography thereby generating new topological relationships between broken or erased line features. Other

approaches such as taking the average value, randomly selecting ones or zeroes or any other method based solely on the contents of the 2 × 2 pixel window present a threat to both topology and topography. The algorithm we have formulated looks beyond the 2 × 2 pixel window to guarantee topography and, unless physically impossible by lack of resolution, to preserve the topological relationships found in the original binary image. A brief background and statement of assumptions will be presented first, followed by the general formulation of the algorithm.

Background and Assumptions

The rationale behind our choice of algorithm is heavily based on current raster scanning practices, especially those involved in choosing the optimum resolution for the scanning process. Scanning resolution on most scanning devices is normally user selectable and can vary from 25 microns (0.001 inches) to 200 microns (0.008 inches). The choice of the proper scanning resolution is a critical step in the raster scanning process. Too coarse a resolution will result in line breakages and line merges, a situation which will significantly impair processing operations during the raster-to-vector conversion and the follow-up vector editing and tagging procedures. On the other hand, each time the scanning resolution is reduced by half, the size of the resultant raster image and the computational requirements to treat such image quadruple.

Selection of the proper scanning resolution is based on critieria of both topography and topology. The resolution of the scan must be fine enough to capture the finest of lines. This does not present significant challenge for the scanning of most map reproduction material since all lines are clearly visible to the eye and a resolution of 100 microns is normally adequate to capture most of the features on topographic maps. On most reproduction material however, the spacings between lines is less than the minimum line size and therefore finer resolutions must be used to capture the topology of the image. This phenomenon is perhaps best illustrated in the scanning of contour separation plates which have line widths varying from 100 microns (intermediate contours) to 180 microns (index contours). These plates can be scanned at 100 micron resolution with minimal line breakages, however the spacing between lines becomes infinitely small whenever lines approach the near vertical cliff situation. This causes the lines to coalesce if the grid resolution is not sufficiently fine. In some low relief areas a resolution of 100 microns may be adequate but in most cases resolutions of 50 or 25 microns must be utilized to capture the topological relationship between contour lines. Based on the above one of the underlying assumptions behind our algorithm is formulated, being that, in most cases, the resolution of the binary raster is much finer than the minimum line width. It is also assumed that the algorithm will only be used on line-art images and that the ratio between the number of 1s and 0s need not be preserved.

General Requirements

With these assumptions, we define the basic properties (apart from efficiency) of an ideal data compaction algorithm as follows:

1 *Preservation of topography.* No line structures or blank areas between line structures must be completely removed from the binary image, with perhaps the exception of single-pixel areas or holes which may be considered as dirt. Likewise, no new line structures may be created by the decomposition of existing blank areas or line structures;

2 *Preservation of topology.* The connectivity of lines and blank areas between line structures must be preserved. See (Rosenfeld, 1970) for the definition of connectivity in raster images;

3 *Preservation of positional accuracy.* The centre of lines in the aggregate image should, within the limits imposed by the coarser grid resolution, correspond to the centre of lines in the original image;

4 *Skeletonization.* The resultant aggregate image should correspond to the skeleton of the original binary image.

The above requirements lead to the use of existing line thinning algorithms for the skeletonization of binary images, except that in this case the lines must not be reduced to single pixel width and that some lines may have to be thickened to ensure their presence in the aggregate image. Adherence to requirements 1 and 4 will be improved if pixel deletions are favoured over pixel additions. Assuming still that the lines are generally thicker than the spacings between lines, thinning the edges of lines will aid skeletonization and favour the preservation of narrow blank areas between line structures. The preservation of connectivity can be achieved by utilizing known line thinning operators, based on a 3×3 neighbourhood analysis and used for both pixel deletion and addition. There will however be cases where neither deletion nor addition is permitted, in which case preservation of topography is likely more important than the preservation of connectivity. This may be the case with thin lines separated by narrow spacings. Positional accuracy can be respected if the thinning (or thickening) operator is applied in the quasi-parallel fashion developed as part of the ARLIP algorithms. Only partial skeletonization can be achieved in an efficient manner and total line thinning is likely more efficiently performed on the aggregate image. Favouring pixel deletions over pixel additions and noting that lines on the aggregate image are only half as wide in pixel notation does however contribute significantly to the skeletonization process. Presumably, the image could be aggregated to such a level that all lines are single pixel width but this may not be the ideal solution as it may not preserve topology.

A Data Compaction Algorithm

The algorithm we have chosen to implement this data compaction technique is heavily based on the ARLIP algorithm for skeletonization, so much so that its software implementation required only cosmetic changes to an existing implementation of this line thinning algorithm.

The partitioning from the ARLIP algorithm forms the grid for our aggregated raster and a 2×2 grid superimposed on the original raster is such that each aggregated pixel contains one pixel from each of the four partitions. In our implementation, only binary pixel values are maintained so that a marked

skeleton does not result. Hence our operator is simplified in the sense that we do not have to consider the special cases for ends-of-lines.

The operator is applied to the matrix in the same row and column ordering to allow for multiple logical passes of the operator in one physical pass of the image. The major difference between line thinning and our compaction algorithm is that not all black pixels are considered for possible deletion. Furthermore, if a pixel cannot be deleted for reasons of connectivity, then all white (0) pixels within that quadruple are set to black (1s). The crucial step is in determining which pixels should be considered for possible deletion.

Singular pixels which are part of an all black quadruple (four 1s) are not considered for deletion. Only those pixels contained in a mixed quadruple are potential candidates, except for one special case where the quadruple is predominantly black, i.e., three black pixels and one white pixel. Tests have shown that premature filling may occur if the black pixel diagonally across the white pixel in this special case is considered for deletion. Hence in deciding whether or not a pixel should be considered for deletion, only 4-way connected neighbours within the quadruple are considered. With due respect given to the position of the pixel within the quadruple, only those black pixels (1s) having a white neighbour (0) above or below, to the right or to the left, are considered for deletion.

The process is limited to two passes of the operator, the second pass required only if the special case mentioned above was encountered during the processing of one particular row. The resultant raster consists solely of 2×2 groups of either all black (1s) or all white (0s) pixels and can be aggregated by retaining the pixel representation of any one of the four partitions. The topography of the original image is conserved and very little loss of topology occurs if the algorithm is applied to the proper image, i.e., to line-art images with lines wider than a single pixel.

Program Performance

A Fortran/Assembler version of the algorithm has been implemented on an HP1000 computer for experimental purposes. Due to lack of finer resolution on the available scanner (resolution is fixed at 100 microns), medium and high density contour plates were reproduced at 1.6 magnification and scanned at 100 microns resolutions. Linewidths in the resulting raster image ranged from 3 to 7 pixels with an abundance of single-pixel spacings between lines. Compression of the raster to 200 micron resolution was performed in approximately half the time required for normal skeletonization of the image. No significant changes in topology resulted and the storage requirements for the resultant raster (run-length coded) were reduced by approximately 40%. Processing times (and costs) for the subsequent skeletonization and raster-to-vector conversion were reduced by as much as 75% in most cases.

CONCLUSION

Preliminary figures indicate that our raster compaction is a viable pre-processing technique which could be applied to most raster scanning systems. The algo-

rithm is currently being optimized and is to be implemented on an IBM mainframe for daily production usage. The same algorithm will also be implemented on a micro-based raster editing facility currently being developed for the CLDS. As research continues in this area, it is hoped that algorithms of this nature will one day be implemented as hardware components of raster scanning devices so that the dream of real-time vectorization can become a reality.

REFERENCES

KREIFELTS, T. 1977. Skeletonising and line following in raster digitized line structures. Presented at GI/NTG Congress on Digital Image Processing, Munich.

MONTUORI, J.S. 1980. Image scanner technology. *Photogrammetric Engineering and Remote Sensing*, vol. 46, no. 1, pp. 49–61.

OLSON, D. and LEBREL, F.W. 1982. Raster scanning for operational digitizing of graphical data. *Photogrammetric Engineering and Remote Sensing*, vol. 18, no. 4, pp. 615–627.

PEUQUET, D.J., 1982. A hybrid structure for the storage and manipulation of very large spatial data sets. U.S. Geological Survey, Open-File Report 82-816.

ROSENFELD, A. 1970. Connectivity in digital pictures. JACM 17, pp. 146–156.

ROSENFELD, A. and KAK, A.C. 1982. *Digital picture processing*. Academic Press, N.Y.

WOETZEL, G. 1978. A fast and economic scan-to-line conversion algorithm. *Proceedings*, 5th ACM-SIGGRAPH Conf., Atlanta, Ga., vol. 12, no. 3, pp. 125–129.