

Compression of line drawing images using Hough transform for exploiting global dependencies

Pasi Fränti¹, Eugene I. Ageenko¹, Heikki Kälviäinen² and Saku Kukkonen²

¹*Department of Computer Science
University of Joensuu
P.O. Box 111, FIN-80101 Joensuu, FINLAND
Email: franti,ageenko@cs.joensuu.fi*

²*Department of Information Technology
Lappeenranta University of Technology
P.O. Box 20, FIN-53851 Lappeenranta, FINLAND
Email: Heikki.Kalviainen,Saku.Kukkonen@lut.fi*

ABSTRACT: *A two-stage method is proposed for compressing bi-level line drawing images. In the first stage line elements are extracted from the image using the Hough transform. In the second stage the original image is compressed by a pixelwise context model. The compressed file consists of the extracted line elements and the compressed raster image. Pixels in the reconstructed feature image are utilized in the local context model in the form of improved prediction accuracy. Better compression performance is achieved if the improvement outweighs the overhead required by the line elements.*

1. INTRODUCTION

Lossless compression of bi-level images has been well studied in literature and several standards already exist [1]. In baseline JBIG the image is coded pixel by pixel in scan raster order using *context-based probability model* and *arithmetic coding* [2]. The combination of already coded neighboring pixels defines the context. In each context the probability distribution of the black and white pixels are adaptively determined. The current pixel is then coded by *QM-coder* [3], the binary arithmetic coder adopted in JBIG.

The baseline JBIG achieves compression ratios from 10 to 50 for typical A4-size images. The pixelwise dependencies are well utilized and there is no much room for improvement in there. Remarkable improvement has been achieved only by specializing to some known image types and exploiting global dependencies [4, 5].

Here we study similar approach by utilizing global dependencies in line-drawing images such as engineering drawings, maps, architectural and urban plans, schemes, and circuits (radio electrical and topological). This kind of images contains mainly of straight line elements. Global information can be captured by extracting line features from the image and storing their location as a side information. The fact that a pixel is located along a line element improves the prediction of the local context model if the existence of the line is predicted accurately enough.

In the present paper we study the use of the Hough transform for extracting the line elements [6]. The line feature extraction, and the storage of the line elements are optimized for maximal output quality of the feature image rather than for the compression performance. The motivation of the work, however, is merely to estimate the amount of expected improvement in the compression due to the line extraction.

2. TWO-STAGE COMPRESSION METHOD

The proposed compression system is outlined in Fig. 1. It consists of two separate parts: *global* and *local* models. In the first stage, global dependencies are exploited by the Hough transform. It is used for detecting straight lines in the input image. Extra analysis phase is applied for extracting end-points of the detected lines. The obtained line elements are stored in the compressed file. A *feature image* is reconstructed from the stored line elements.

In the second stage, the original image is compressed using a local context model and arithmetic coding. Pixels of the feature image are utilized in the local context model in the form of improved prediction accuracy. The reconstructed feature image may also be used as a lossy approximation of the original image. The overall compression system, however, is entirely lossless. The compressed file consists of the line elements and the compressed raster image.

2.1 Hough transform for line extraction.

The motivation is to find rigid fixed length straight lines from the image. The lines are detected by the *Hough transform (HT)* [6] as follows:

1. Create a set of coordinates (x, y) from the black pixels in the image.
2. Transform each coordinate into a parametrized curve in the parameter space.
3. Increment the cells in the parameter space determined by the parametric curve.
4. Detect local maxima in the accumulator array. Each local maximum may correspond to a parametric curve in the image space.

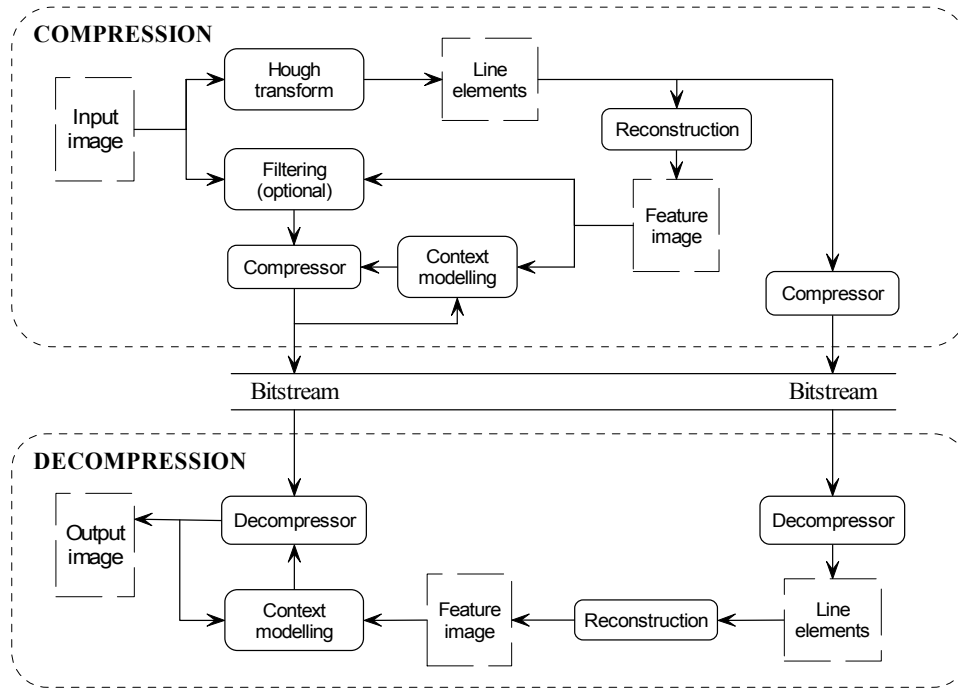


Fig. 1. Block diagram of the compression system.

5. Extract the curve segments using the knowledge of the maximum positions.

The parameter space is an $N \times N$ accumulator array where N can be tuned according to the image size, e.g. $N =$ the size of the image. Usually, the (ρ, θ) parametrization is used but the (a, b) one, corresponding to $y = ax + b$, is also possible. The accumulation matrix is quantized with equal intervals.

The Hough transform is capable to determine the location of a line (as a linear function) but it cannot resolve the end-points of the line. In fact, HT does not even guarantee that there exists any finite length line in the image but it only indicates that the pixels (x, y) along $y = ax + b$ may possibly represent a line. The existence of a line segment must therefore be verified. The verification is performed by scanning the pixels along the line and checking whether they meet certain criteria. We use the scanning width, the minimum number of pixels, and the maximum gap between pixels in a line as the criteria. If predefined threshold values are met, line segment is detected and its end-points are stored.

2.2 Storing the line elements

The extracted line elements are stored as $\{(x_1, y_1), (x_2, y_2)\}$ representing the end-points of the line. A single coordinate value takes $\lceil \log_2 n \rceil$ bits where n is the dimension of the image. For example, a line in an image of 4096×4096 pixels takes $4 \times 12 = 48$ bits in total. Somewhat more compact representation

could be achieved by sorting the line elements according to their first coordinate x_1 and storing the difference between two subsequent x_1 's. An improvement about 7 bits (From 12 to 5 bits) was estimated if entropy coding were applied to these difference values.

2.3 Compression of the raster image

The method starts by creating equal size reconstruction of the line features to approximate the input image. The original image is then compressed using the baseline JBIG, which uses previously coded neighboring pixels as context. The context is determined by combining index out of the neighboring pixel values and accessing to the model using a look-up table. Additional context pixels are taken from the feature image. An important point is that any pixel in the feature image can be utilized, even the current pixel that is to be compressed. Here we use ten pixels from the original image as in 3-line JBIG modelling and five pixels from the feature image, see Fig. 2. The actual coding is performed by QM-coder, the binary arithmetic coder of JBIG [3].

HT does not determine the width of the lines but wider lines are represented by a bunch of collinear line elements, see Fig 3. The extracted line elements may also be deviated from their original direction and/or have one-pixel positional error because of the quantization of the accumulation matrix. Therefore we do not utilize the feature image directly after HT but process it first by consequent operations of morphological *dilation* and *closing* with a symmetric

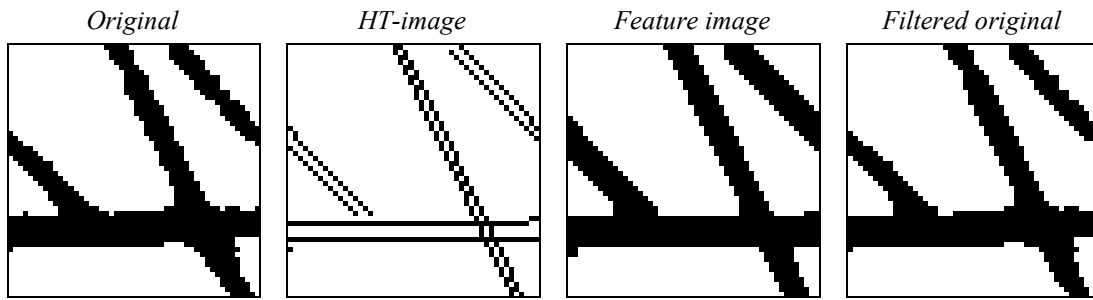


Fig. 3. Enlargement of an image sample of size 50x50 pixels.

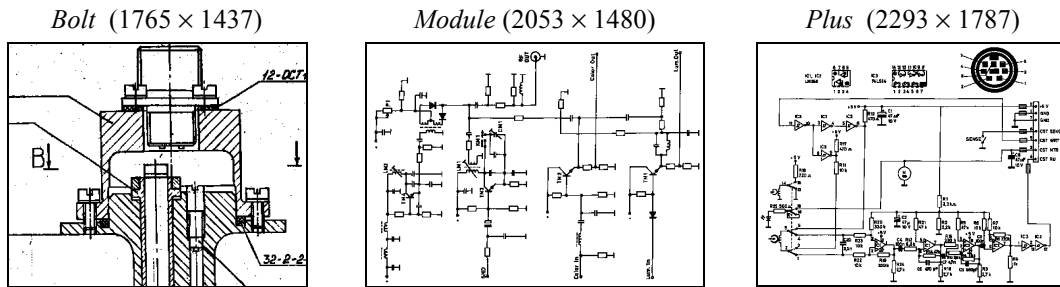


Fig. 4. Test images.

3x3 structure element [7]. These operations make the lines one pixel thicker in all directions (dilation) and fill gaps between the line elements (closing).

2.4 Extension to near lossless compression

The method can be extended to near lossless compression by filtering the original image using the information of the extracted line elements. A difference (mismatch) image between the original and the feature image is constructed. Isolated mismatch pixels (and pixel groups up to two pixels) are detected and the corresponding pixels in the original image are reversed. The filtering removes random noise and smoothes edges along the detected line elements. Undetected objects (such as text characters) are untouched allowing their lossless reconstruction.

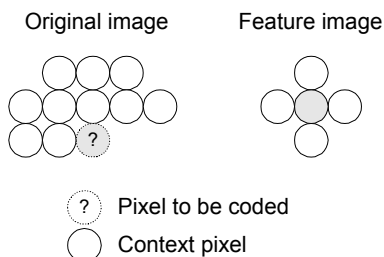


Fig. 2. Two-level context template

3. TEST RESULTS

The performance of the proposed compression method is tested by compressing the three test images shown in Fig. 4. Three different feature files

were constructed from each test image with different amount of line elements. The corresponding feature images of test image *Bolt* are shown in Fig. 5.

The compression results are summarized in Table 1, which shows the sizes of the feature files and the raster images. Improvement of about 1 to 10 % is obtained when compressing the raster image. The improvement is the greater the more line elements is extracted. The amount of saving, however, is rather small and in all cases too small to compensate the overhead required by the feature file.

The best results of the lossless and near lossless compression are summarized in Table 2. Overall improvement of about 4 % is obtained in the case of the near lossless compression. The quality of the decompressed images is visually the same as the original since only isolated mismatch pixels are reversed. The quality is sometimes even better because the reversed pixels are mainly random noise, or scanning noise near the line elements.

4. CONCLUSIONS

A two-stage compression method based on the Hough transform and JBIG-based pixelwise compression was studied. The method exploits global dependencies in line-drawing images. Improvement up to 10 % is obtained in the local context model when the information of the feature image is utilized. The amount of improvement, however, was not as high as expected and the compression gain was outweighed by the overhead required by the storage of the line elements.

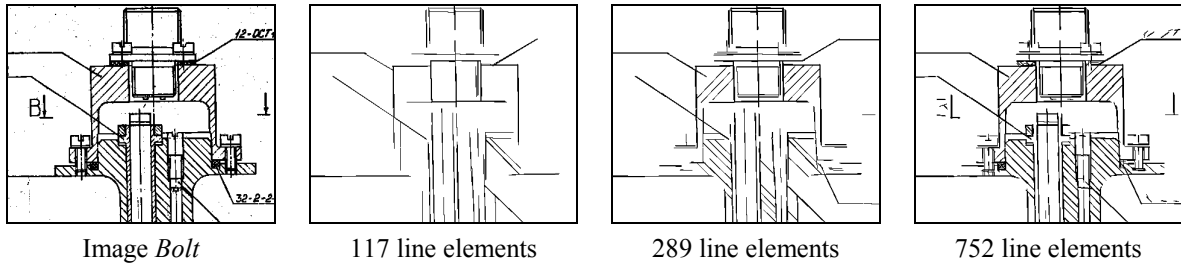


Fig. 5. Feature images from test image Bolt.

Table 1. Lossless compression results (in bytes) with variable amount of extracted line elements.

	BOLT			MODULE			PLUS		
	Feat. File	Raster image	Total	Feat. file	Raster image	Total	Feat. file	Raster image	Total
Set 1	702	12,598	13,300	120	7,647	7,767	114	17,629	17,743
Set 2	1,734	12,177	13,911	384	7,615	7,999	426	17,507	17,933
Set 3	4,512	11,549	16,061	1,788	7,354	9,142	2,532	17,283	19,815
JBIG	-	12,966	12,966	-	7,671	7,671	-	17,609	17,609

Table 2. Summary of the compression results (in bytes).

	LOSSLESS (set 1)			NEAR LOSSLESS (set 2)			
	JBIG	Feat. file	Raster image	Total	Feat. File	Raster image	Total
BOLT	12,966	702	12,598	13,300	1,734	10,472	12,206
MODULE	7,671	120	7,647	7,767	384	7,021	7,405
PLUS	17,609	114	17,629	17,743	426	16,630	17,056
TOTAL	38,246	936	37,874	38,810	2,544	34,123	36,667

A slightly better result, an improvement of about 4 %, was achieved when the method was extended to near lossless compression using feature-based filtering of isolated mismatch pixels.

5. ACKNOWLEDGEMENTS

The work of Pasi Fränti was supported by a grant of the Academy of Finland and the work of Eugene Ageenko by a grant of Centre for International Mobility (CIMO).

6. REFERENCES

1. R.B. Arps, T.K. Truong, "Comparison of international standards for lossless still image compression". *Proceedings of the IEEE*, **82**, 889-899, June 1994.
2. JBIG, Progressive Bi-level Image Compression, ISO/IEC International Standard 11544, ITU Recommendation T.82, 1993.
3. W.B. Pennebaker, J.L. Mitchell, *JPEG Still Image Data Compression Standard*. Van Nostrand Reinhold, 1993.
4. P.G. Howard, "Text image compression using soft pattern matching", *The Computer Journal*, **40**, 146-156, 1997.
5. I.H. Witten, A. Moffat and T.C. Bell, *Managing Gigabytes: Compressing and Indexing Documents and Images*. Van Nostrand Reinhold, New York, 1994.
6. H. Kälviäinen, P. Hirvonen, L. Xu, and E. Oja, "Probabilistic and non-probabilistic Hough transforms: overview and comparisons", *Image and Vision Computing*, **13**, 239-251, May 1995.
7. J. Serra, *Image Analysis and Mathematical morphology*. Academic Press, London, 1982.