SHAPE ENCODING FOR EDGE MAP IMAGE COMPRESSION

Demetrios P. Gerogiannis, Christophoros Nikou, Lisimachos P. Kondi

Department of Computer Science and Engineering,
University of Ioannina, 45110 Ioannina, Greece
{dgerogia,cnikou,lkon}@cs.uoi.gr

ABSTRACT

A method for the efficient encoding and decoding of a shape with application to edge map image compression is proposed. The method relies on the modeling of the manifold of a shape by line segments. Then, encoding is performed by collecting the characteristic features of each line segment, namely the starting and ending points and the number of points contributing to the computation of the corresponding segment. The reconstruction of the shape may be obtained by uniform sampling of points from each line segment computed in the encoding process. Experimental analysis demonstrates that in case of employing a robust and efficient line segment detection algorithm, the proposed encoding/decoding scheme exhibits high compression rates, compared to widely used lossless compression methods, while providing low distortion values.

Index Terms— Binary image compression, image representation, line segments, shape encoding, shape reconstruction.

1. INTRODUCTION

Shape representation is a significant task in image storage and transmission, as it can be used to represent objects at a lower computational cost, compared to non-encoded representations. For example, the widely used MPEG4 Part 2 object-based video standard uses shape coding for describing regions, called video object planes, that represent an object [1]. In that case, accurate shape encoding leads to better preservation of contour details.

The pioneering work in [2], where sequences of line segments of specified length and direction are represented by chain codes was proposed for the description of digitized curves, contours and drawings and it was followed by numerous techniques. Shape coding is a field that has been studied extensively in the past but it is still very active. Various methods have been studied in [3], including the context-based arithmetic encoding (CAE), which has been adopted by the MPEG4 Part 2 standard.

The digital straight line segments coder (DSLSC) was introduced in [4] for coding bilevel images with locally straight edges, that is, single binary shape images and bilevel layers of digital maps. DSLS models the edges by digital straight line segments (DSLS) [5]. Compared to standard algorithms like JBIG [6], JBIG-2 [7] and MPEG4 CAE [1],[3] DSLSC provides better results, as it fully exploits the information given by the local straightness of the boundary, which is not the case for the other methods.

DSLSC is further improved in [8], where the segmentation of the alpha plane in three layers (binary shape layer, opaque layer, and intermediate layer) is employed. Experimental results demonstrated substantial bit rate savings for coding shape and transparency when compared to the tools adopted in MPEG4 Part 2.

Discrete straight lines were also employed in [9] for shape encoding and improvement of the compression rate is reached by carrying out a pattern substrings analysis to find high redundancy in binary shapes.

A lossless compression of map contours by context tree modeling of chain codes is described in [10]. An optimal n-ary incomplete context tree is proposed to be used for improving the compression rate.

A JBIG-based approach for encoding contour shapes is introduced in [11], where a method is presented that manages to efficiently code maps of transition points, outperforming, in most cases, differential chain-coding.

In this work, we propose a scheme for efficiently compressing edge map images. The rationale is to model the manifold of the edge map image by fitting line segments to it. Then, each line segment is encoded by its starting and ending points and the number of points contributing to its computation. The reconstruction of the contour is achieved by uniformly sampling points, along the direction of each encoded line segment. In the remainder of the paper, the compression and decompression algorithms base on line segment fitting are presented along with extensive experimental evaluation of the method.

2. ENCODING SHAPES

Line segments are important features in computer vision, as they can encode rich information with low complexity. We take advantage of this feature for encoding a 2D set of points describing a shape as a collection of line segments that approximate the manifold of the shape, by assuming that the manifold is locally linear. The initial and ending points of each line segment may be considered as the characteristic points carrying the compressed information that can reproduce the initial shape. The larger the number of characteristic points is, the better shape information is preserved.

A line segment ϵ may be described by its starting and ending points $\mathbf{x_s}^{\epsilon}$ and $\mathbf{x_e}^{\epsilon}$ respectively. The collection of the starting and ending points of all the segments modeling the shape manifold are the characteristic points of the shape. Note that the characteristic points are ordered. Moreover, since the traversal of the line segments is known, the line segment can be described by its starting point and the transition vector towards the ending point. The ending point of one segment is the starting point of its successor in the traversal order. Eventually, the shape can be encoded by selecting an arbitrary initial point from the characteristic points and by the corresponding transition vectors after visiting each segment based on the traversal order, in a similar manner described in [12].

To reconstruct the image we need to reconstruct all the points contributing to the computation of each line segment ϵ based on the characteristic points. In principle, a line is modeled by the parametric equation $Ax + By + C = 0$. where $x,\ y,\ A,\ B,\ C \in \mathbb{R}$ and (x, y) is a point laying onto the line. If the starting ($\mathbf{x_s}^{\epsilon}$) and ending ($\mathbf{x_e}^{\epsilon}$) points of a line segment ϵ are given, determining A, B, C is trivial. Thus, starting from point $\mathbf{x_s}^{\epsilon}$ and following the direction of the line segment with a predefined step $\lambda \in \mathbb{R}^{+}$ each time, we may reconstruct (approximate) the initial points. The value of the step λ controls the density of the result: the higher its value is, the larger is the number of extracted points. In case of points laying on an image grid, integer arithmetics need to be considered and selecting $\lambda = 1$ yields the algorithm of Bresenham [13] which may reconstruct the line segment pixels efficiently and handle the aliasing effect.

Algorithms 1-2 describe the proposed framework for compression/decompression of bi-level images of edge maps.

Algorithm 1 Image compression

```
input: An edge map image I, representing shapes.
output: A set of features S that encodes image I.
Detect the line segments that describe I. Let K be the number of line segments detected.
Detect the traversal order of the line segments.
Refine shape, i.e. close gaps between line segments. Extract the characteristic point $P = \mathbf{p_i},\ i = 1 \dots K$, based on the shape traversal.
$S = \{\mathbf{p_1}\}$.
for i=2:K do
    $S = S \cup \{\mathbf{dx},\ dx = \mathbf{p_{i-1}} - \mathbf{p_i}\}$.
end for
```

Algorithm 2 Image decompression

```
input: A set of features S that encodes an image I.
output: The reconstructed image I.
Recover the characteristic points $P = \{\mathbf{p_i},\ i = 1 \dots K\}$, based on initial point and transitions encoded in S.
for i=2:K do
    Produce the set of points $R$, e.g. [13], containing the points of the line segment from $\mathbf{p_{i-1}}$ to $\mathbf{p_i}$.
    Set the pixels of I corresponding to coordinates of points in R on.
end for
```

3. EXPERIMENTAL RESULTS

In this section, the experimental investigation of the proposed method is presented regarding its robustness and efficiency. To that end, a compression-distortion study was carried out.

Compression was computed as the ratio of the file size between the compressed and the original files. Various lossless methods were considered and the corresponding size of the output files they produced was used as the reference original file size. The methods against which we compared the proposed framework are the CCITT G4 standard [14] (denoted as FAX4 herein), adopted amongst others by the TIFF image file format for binary images, and the widely used standards JBIG [6] and JBIG2 [7].

As far as the distortion is concerned, a twofold computation was performed in terms of measuring the loss of information and the similarity between the initial and the final edge map images. Therefore, the distortion index adopted by MPEG4 [15], given by

$$D_R = \frac{\text{Number of pixels in error}}{\text{Number of interior pixels}}, \tag{1}$$

was also used in this work. The Hausdorff distance between the original edge map X and the reconstructed edge map Ys, given by

$$D_H(X, Y) = \max_{\mathbf{x} \in X} \min_{\mathbf{y} \in Y} \{|\mathbf{x} - \mathbf{y}|_1\}, \tag{2}$$

was used to measure the similarity between X and Y.

As mentioned in section 2, the proposed compression method uses a line segment fitting algorithm. There are a lot of methods that have been proposed in the related literature. A widely used method is the Hough Transform (HT) [16] and its variants [17]. However, the principal goal of HT is to detect lines or line segments in a coarse level. Thus, it cannot be used for detailed description of shapes. This fact has also been observed for some variants of HT in our previous work [18].

Another method for line segment detection is polygon approximation [19]. Having the ordering of the points, line segment computation begins from an arbitrary point, and by traversing the shape, fits a line segment to all the points that have been visited. When a new point that deviates from the currently computed line segment is visited, it is regarded as the starting point of a new segment. The computation iterates until all shape points are visited. In our experiments, we used the implementation of the toolbox provided by Kovesi [20]. Polygon approximation is the method used in [12] for describing the contour.

Recently, we have proposed a direct split-and-merge framework (DSaM) for detecting line segments in unordered point clouds [18]. The algorithm operates in two phases. It initiates by assuming that all points are collinear, i.e. they belong to the same group. Then, it iteratively splits the groups that are modeled by non elongated ellipses or are not tight. A criterion for splitting is the minimum eigenvalue of the corresponding covariance matrix. Then, iteratively it merges neighboring ellipses, which after merging also provide highly elongated ellipses. According to our study, the DSaM algorithm is a robust framework for line segment detection. In [18], we also proposed a method for automatic tuning of the various parameters of the DSaM algorithm. This configuration was used in our experimental study.

For the experimental study, we used two datasets. The Gatorbait dataset [21] contains the silhouettes of 38 fishes, belonging to 8 categories. The MPEG7 dataset [22] contains 1400 object contours belonging to 70 categories, with 20 members per each category. All datasets consist of binary images. In the case of the Gatorbait100 dataset, the images were initially thinned so as to extract the contour line. Let us note that in this case, there are some inner structures that were also considered in our experiments.

The overall results of our experimental analysis are demonstrated in Tables 1 and 2, with results for various configurations of the line segment detection algorithms considered. The values next to the method prefix in the first column of the Tables indicates the corresponding configuration of the method. The DSaM algorithm imposes two thresholds controlling the deviation of linearity of a set of points and the neighborhood of a point respectively. In our study the values considered for these thresholds were $\{[0.3, 2.0], [0.4, 2.0], [0.5, 2.0], [0.8, 2.0], [1.3, 2.0], [2.3, 2.0]\}$. The second threshold was measured in pixels. As far as the polygon approximation (PA) is concerned, this algorithm applies one threshold controlling the deviation of a set of points from linearity. In that case, the thresholds used were $\{1, 2, 5, 7, 10\}$ pixels. The percentages regarding the compression values in Tables 1 and 2 refer to the file size produced by the proposed compression scheme compared to the corresponding file size produced by the related lossless method as mentioned on the second row of the tables. More specifically, in Table 1, in the first row, we may conclude that the proposed scheme, using DSaM for line modeling, provides a compressed shape, which on average (over the whole data set) employs 10% of the bits employed when compressed by FAX4 [14], 25% of the bits used when compressed by JBIG [6] and 32% of the bits employed by a JBIG2 compression [7]. Moreover, the average distortion in terms of information loss is $D_R = 10\%$ and the average Hausdorff distance between the original shape and the compressed shape is $D_H = 8$ pixels. Recall that FAX4, JBIG and JBIG2 are lossless compression algorithms.

Table 1: Experimental results for the Gatorbait dataset [21] (38 shapes).

Method	**Compression**			**Distortion**	
	FAX4 [14]	JBIG [6]	JBIG2 [7]	D_R (1)	D_H (2)
DSaM#1	10%	25%	32%	10%	8
DSaM#2	10%	23%	30%	8%	7
DSaM#3	9%	22%	28%	5%	8
DSaM#4	8%	20%	26%	5%	8
DSaM#5	8%	19%	25%	3%	10
DSaM#6	7%	17%	22%	3%	12
PA#1	17%	40%	51%	11%	4
PA#2	11%	27%	35%	1%	4
PA#3	8%	18%	23%	2%	8
PA#4	7%	16%	20%	2%	12
PA#5	6%	14%	18%	6%	17

Table 2: Experimental results for the MPEG7 dataset [22] (1400 shapes).

Method	**Compression**			**Distortion**	
	FAX4 [14]	JBIG [6]	JBIG2 [7]	D_R (1)	D_H (2)
DSaM#1	16%	30%	30%	9%	5
DSaM#2	15%	28%	28%	10%	6
DSaM#3	14%	27%	26%	10%	6
DSaM#4	13%	24%	24%	10%	7
DSaM#5	12%	23%	22%	10%	7
DSaM#6	10%	20%	19%	12%	10
PA#1	25%	47%	46%	7%	3
PA#2	16%	32%	32%	4%	3
PA#3	11%	21%	21%	7%	6
PA#4	10%	18%	18%	9%	9
PA#5	8%	16%	16%	12%	12

Figures 1-2 demonstrate some representative results of the proposed method with various line segment detection algorithms. One may observe that the compression based on the DSaM line segment detection preserves more details of the initial set, compared to the polygon approximation algorithm, whose result is more coarse.

Finally, the rate-distortion curves for the above experiments are presented in Figure 3. The blue line corresponds to the compression results based on DSaM [18], while the red line refers to the results based on a compression using polygon approximation [12]. As it can be observed, the DSaM method provides a clearly better performance. Note that the bits needed to encode the information for each method cannot be fixed directly, as they are affected by the tuning of the associated thresholds and parameters. Thus, equal bit rates cannot be established for DSaM and polygon approximation.

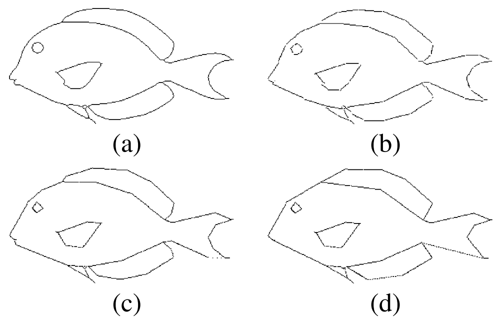

Fig. 1: Representative results of the reconstruction method on the Gatorbait [21] dataset. (a) The original image. Results extracted with (b) DSaM [18], (c) polygon approximation [19] with automatic tuning (d) polygon approximation [19] with threshold value set to 5 pixels.

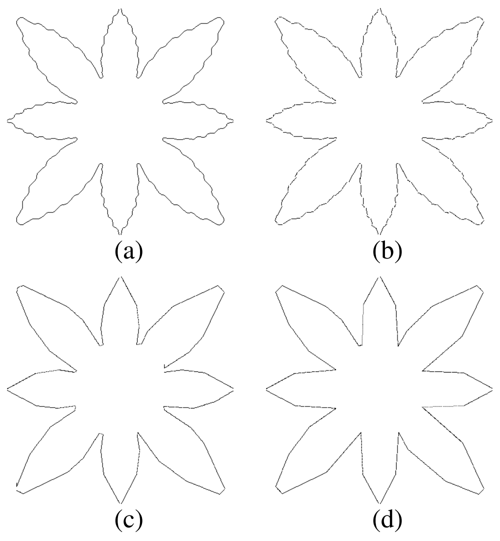

Fig. 2: Representative results of the reconstruction method on the MPEG7 [22] dataset. (a) The original image. Results extracted with (b) DSaM [18], (c) polygon approximation [19] with automatic tuning (d) polygon approximation [19] with threshold value set to 5 pixels.

One may observe that the proposed encoding framework provides satisfactory results in terms of compression, while offering low distortion. The DSaM method provides similar or better results compared to the polygon approximation that is used in the MPEG standard, in terms of compression,

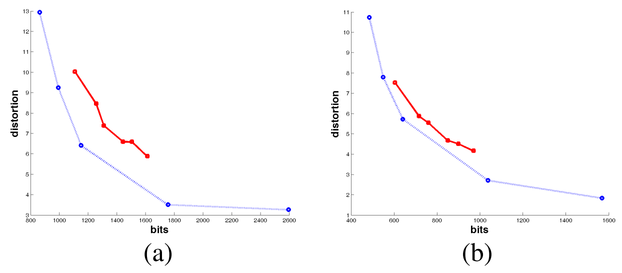

Fig. 3: Rate-distortion curves for (a) the Gatorbait dataset [21] and (b) the MPEG7 dataset [22]. The blue line corresponds to the compression results based on DSaM [18] and the red line refers to polygon approximation [12].

but with far lower distortion. Also, DSaM manages to significantly improve the compression rate providing an image quality (in terms of distortion) similar to the lossless algorithms.

4. CONCLUSION

A method for the efficient encoding and decoding of 2D point sets describing shapes, based on line segments fitting, is introduced in this article. The experimental analysis demonstrated that the proposed scheme offers high compression rates compared to widely used lossless algorithms, while the incurred distortion is low. Finally, the method can be easily modified to handle $3D$ points. In that case, the analogous $2D$ plane should be computed and then the reconstruction of the shape should be performed by sampling points on that plane. Since the compression method is regarded as the encoding of transitions, a Kalman filter could be employed to handle the errors incurred due to integer arithmetics and further lower the distortion. This is a direction of the future work regarding the proposed method.

5. REFERENCES

[1] ISO/IEC Int. Std., "Information technology – coding of audio-visual objects – part 2: Visual coding," `http://www.digitalpreservation.gov/formats/fdd/fdd000080.shtmln`, 1999, Accessed on January, 2015.

[2] Herbert Freeman, "Computer processing of line-drawing images," *ACM Comput. Surv.*, vol. 6, no. 1, pp. 57–97, 1974.

[3] A. K. Katsaggelos, L. P. Kondi, F. W. Meier, J. Ostermann, and G. M. Schuster, "MPEG-4 and rate-distortion-based shape-coding techniques," *Proceedings of the IEEE*, vol. 86, no. 6, pp. 1126–1154, 1998.

[4] S. M. Aghito and S. Forchhammer, "Context-based coding of bilevel images enhanced by digital straight line analysis," *IEEE Transactions on Image Processing*, vol. 15, no. 8, pp. 2120–2130, 2006.

[5] R. Klette and A. Rosenfeld, "Digital straightness - a review," *Discrete Applied Mathematics*, vol. 139, no. 1-3, pp. 197–230, 2004.

[6] ISO/IEC Int. Std. 11544, "Coded representension of picture and audio information - progressive bi-level image compression," 1993.

[7] ISO/IEC Int. Std. 14492, "Coded representension of picture and audio information - lossy/lossless coding of bi-level images (JBIG2)," 2000.

[8] S. M. Aghito and S. Forchhammer, "Efficient coding of shape and transparency for video objects," *IEEE Transactions on Image Processing*, vol. 16, no. 9, pp. 2234–2244, 2007.

[9] H. Sanchez-Cruz, "Proposing a new code by considering pieces of discrete straight lines in contour shapes," *Journal of Visual Communication and Image Representation*, vol. 21, no. 4, pp. 311–324, 2010.

[10] A. Akimov, A. Kolesnikov, and P. Fränti, "Lossless compression of map contours by context tree modeling of chain codes," *Pattern Recognition*, vol. 40, no. 3, pp. 944–952, 2007.

[11] A. J. Pinho, "A JBIG-based approach to the encoding of contour maps," *IEEE Transactions on Image Processing*, vol. 9, no. 5, pp. 936–941, 2000.

[12] K. J. O'Connell, "Object-adaptive vertex-based shape coding method," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 1, pp. 251–255, 1997.

[13] J. E. Bresenham, "Algorithm for computer control of a digital plotter," *IBM Systems Journal*, vol. 4, no. 1, pp. 25–30, 1965.

[14] ITU-T, "T.6: Facsimile coding schemes and coding control functions for group 4 facsimile apparatus," `http://www.itu.int/rec/T-REC-T.6/en`, 1984, Accessed on January, 2015.

[15] H. Wang, G. M. Schuster, A. K. Katsaggelos, and T. N. Pappas, "An efficient rate-distortion optimal shape coding approach utilizing a skeleton-based decomposition," *IEEE Transactions on Image Processing*, vol. 12, no. 10, pp. 1181–1193, 2003.

[16] P. V. C. Hough, "Method and means for recognizing complex patterns," U.S. 3,069,654, Dec. 18, 1962.

[17] P. Mukhopadhyay and B. B. Chaudhuri, "A survey of Hough transform," *Pattern Recognition*, vol. 48, no. 3, pp. 993–1010, 2015.

[18] D. Gerogiannis, C. Nikou, and A. Likas, "Modeling sets of unordered points using highly eccentric ellipses.," *EURASIP Journal on Advances in Signal Processing*, , no. 11, 2014.

[19] U. Ramer, "An iterative procedure for the polygonal approximation of plane curves," *Computer Graphics and Image Processing*, pp. 244–256, 1972.

[20] P. D. Kovesi, "MATLAB and Octave functions for computer vision and image processing," Centre for Exploration Targeting, School of Earth and Environment, The University of Western Australia, Available from: http://www.csse.uwa.edu.au/ pk/research/matlabfns/, Last Visited on September 3rd 2013.

[21] University of Florida, "Gatorbait 100," `http://www.cise.ufl.edu/~anand/publications.html`, Accessed on January, 2014.

[22] Temple University, "College of science and technology, mpeg-7 dataset," `http://www.dabi.temple.edu/~shape/MPEG7/dataset.html`, Accessed on January, 2014.