A LINEAR APPROXIMATION SCHEME FOR PLANAR CLOSED CONTOURS WITH 2^n NUMBER OF EQUALLY SPACED POINTS

Lasko Laskov
Institute of Mathematics and Informatics at Bulgarian Academy of Sciences,
Acad. Georgi Bonchev Str., Block 8, 1113 Sofia, Bulgaria
llaskov@math.bas.bg

ABSTRACT

Planar closed contours are used in number of applications such as optical character recognition (OCR), content based image retrieval (CBIR), medical image processing, etc., for object representation, feature vector extraction and object recognition. Some feature generation techniques, for example Fourier descriptors (FDs), require contours to be represented with 2^n number of equally spaced points. In this paper an algorithm for linear approximation of the original contour pixels with 2^n number of equally spaced points in $\mathbb{R}^2$ is proposed.

1 INTRODUCTION

Contours, extracted from two-dimensional objects, are considered an important shape representation method in number of fields in computer vision including optical character recognition (OCR), content based image retrieval (CBIR), medical image processing, motion detection, etc. Outer and inner closed contours of planar figures are often used for feature vector extraction and hence, for feature space definition, classifier learning and object recognition. Feature vector extraction from planar closed contours often adopt techniques based on *Fourier descriptors* (FDs) [12], [3], [1] or *wavelet descriptors* (WDs) [11], [4] [2]. There are two restrictions which are imposed by such methods: (i) the number of points of the input contour must be power of two, and (ii) the points must be equally spaced.

In the case of FDs, the first restriction is required in order to apply fast Fourier transform (FFT). Also, in the case of DWTs, as noted in [11], some of the reconstructed contours will not close due to the fact that the number of original pixels is not power of two.

The second restriction is mainly needed to avoid signal disturbance after transformations. If the input contour is described using eight positions of neighborhood, than all neighboring pixels which lie horizontally or vertically from one another will be at distance 1, while all pixels which lie diagonally will be at distance $\sqrt{2}$.

Since in practice the extracted contours rarely fulfill the above restrictions, an algorithm is needed for approximation of the input discrete contour with another closed contour which consists of 2^n number of equally spaced points.

In the last two decades, a lot of works has been published on the topic of planar contour approximation. There are basically two broad classes of methods: polygon based and spline curve based. In the polygon based methods the discrete contours are examined as closed polygons with contour pixels being the vertices, like in [9] where a dynamic programming approach to polygonal approximation of digital curves is presented. The solution of the approximation problem is based on a global optimization criterion and is formulated as a recursive function. The presented algorithm is designed to approximate open curves, but the authors suggest that it can be extended to closed contours by an exhaustive search of a optimal initial (starting) point. The extension of this method to closed contours has been examined in more details in [6]. Another improvement of [9] is presented in [5] where a termination scheme at suitable number of vertices of the approximating polygon and initial point determination criterion is proposed.

Another polygon based method is [10] where the problem of approximation of closed contours with equilateral polygons is examined. The presented method is based on nonlinear programming scheme for minimization of a mechanical system's energy. The algorithm consists of two basic steps: (i) initial polygon definition by selecting dominant points on the contour for polygon's vertices; (ii) solution of the nonlinear optimization problem, subject to the constraint that the polygon edges have to be of equal length.

Although the above methods are reported to give good results they are applicable in our case if we assume that the number of the approximation points are less than the original number of pixels. Unfortunately this will produce data loss and it can lead to contour shape distortion. In the case when the number of approximation points must be greater than the number of original pixels, spline based methods can produce more accurate results. Most of the methods for spline based approximation of discrete planar closed contours are usually adopting cubic Bézier curves [7], [8]. These methods are suitable for certain tasks as characters representation at different scales but in these works the preliminarily fixed number of approximation points is not considered, except in [4] where authors propose Bézier splines to approximate the original contour with 128 or 256 points. A drawback of the methods based on Bézier splines is that they can lead to considerable contour smoothing since the approximated curve can lie relatively far from the control points, which is not desirable in our case. Also, the equal spacing of the points of the approximation discrete curve is not discussed in these works.

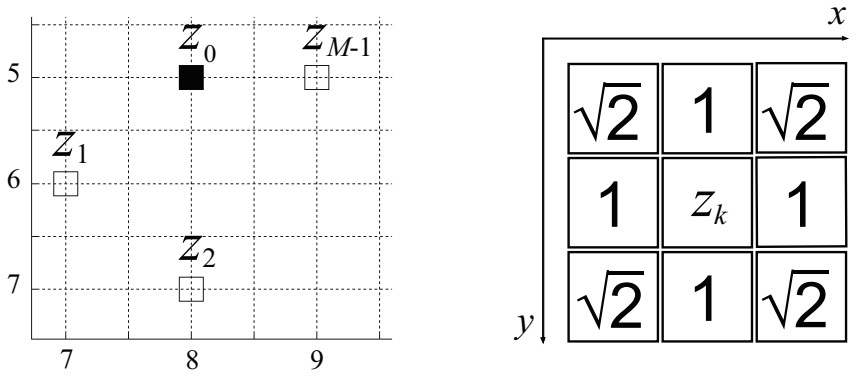
In this paper a new method for contour linear approximation is proposed, in which the number of the new approximation points is $N = 2^n$, and $N > M$, where M is the number of the original contour pixels. Also, the points of the resulting contour are equally spaced in $\mathbb{R}^2$. Linearity is chosen to reduce the redundant curve smoothing and all the resulting new points are calculated on the line segments, connecting the original neighboring pixels.

2 PROBLEM DESCRIPTION

A discrete planar closed contour z is given by the coordinates of its pixels

$$z_j = (x_j, y_j), \;\; j = 0, 1, \ldots, M-1 \tag{1}$$

where z_j are in counterclockwise order, starting from the contour initial pixel z_0 and ending with its last pixel z_{M-1} and M is the number of pixels (see Fig.1(a)). The selection of the initial pixel z_0 is performed by the contour trace algorithm and the described method does not depend on it. Since eight-position neighborhood is used, the distance between any two neighboring pixels z_k and z_{k+1} is 1 if the two pixels lie horizontally or vertically and is $\sqrt{2}$ if the two pixels lie diagonally. This



(a) (b)

Figure 1: (a) A fragment of a closed contour given in pixel coordinates. The initial pixel z_0 is marked in black and the first two pixels z_1, z_2 and the last pixel z_{M-1} are shown. (b) The pixel z_k and the distance to his eight neighbors.

relation can be expressed by:

$$d_k = \begin{cases} 1, & z_{k+1} \in \{(x_k, y_k - 1), (x_k - 1, y_k), (x_k, y_k + 1), (x_k + 1, y_k)\} \\ \sqrt{2}, & z_{k+1} \in \{(x_k - 1, y_k - 1), (x_k - 1, y_k + 1), (x_k + 1, y_k + 1), \\ & \qquad (x_k + 1, y_k - 1)\} \end{cases} \tag{2}$$

where $d_k = \|z_k, z_{k+1}\|$ denotes the Euclidean distance between the neighboring pixels z_k and z_{k+1} (see Fig.1(b)).

The objective is to find another contour $\bar{z}$, $\bar{z}_j = (\bar{x}_j, \bar{y}_j)$, $j = 0, 1, \ldots, N-1$ which will approximate the input contour z and has the following properties:

- the coordinates of the points of $\bar{z}$ are real numbers, $\bar{z}_j \in \mathbb{R}^2$, $j = 0, 1, \ldots, N-1$;
- any two consecutive points of $\bar{z}$ are at the same distance $\delta = \|\bar{z}_k, \bar{z}_{k+1}\|$, where $k = 0, 1, \ldots, N-2$;
- the number of the points of $\bar{z}$ is a power of two $N = 2^n$, where $N > M$;
- in order to insure the input shape preservation any approximating point must lie on a line segment connecting two neighboring pixels of the input contour.

Described in this way, the problem for planar closed contour approximation is very similar to the polygon approximation problem. The main difference is that in the presented problem, the approximating polygon has more vertices than the input one. Also, there is no constraint that the output approximating points must contain the input pixels.

3 THE PROPOSED ALGORITHM

The algorithm consists of two main parts: (i) determination of the distance δ between the consecutive neighboring points of the approximation contour $\bar{z}$; (ii) linear approximation process in which the coordinates of the points of $\bar{z}$ are calculated. These two basic steps are iterated until an approximation of δ is found for which the output contour $\bar{z}$ is composed of the required number of points N and is a closed contour.

The first approximation of the distance δ is calculated based on the perimeter $p(z)$ of the input contour and the number of approximating points N. The parameter of the input contour is given by:

$$p(z) = \sum_{k=0}^{M-2} d_k + \|z_{M-1}, z_0\|, \tag{3}$$

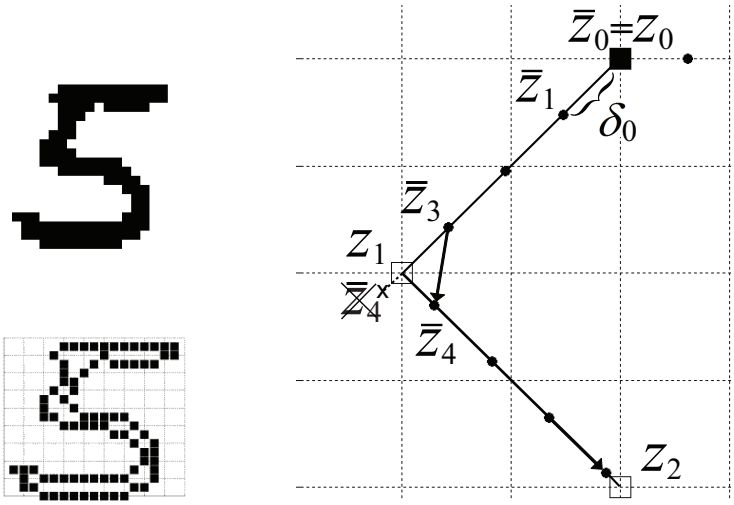
where d_k is defined in (2) and the last term of the above equation is needed because the contour is closed. The number of approximating points is:

$$N = 2^n, \;\; n = \lfloor \log_2 M \rfloor + 1 \tag{4}$$

If we denote $n_0 = \lfloor \log_2 M \rfloor$ then there is no such whole number n_1 for which $2^{n_1} \in (2^{n_0}, M]$. Hence N, defined in (4), is the first whole number which is a power of two and for which $N > M$. Sometimes in practice, if N is too close to M, $n = \lfloor \log_2 M \rfloor + 2$ will produce more accurate approximation of the input contour than (4), as is in the example given on Fig. 2.

Once the input contour parameter $p(z)$ and the number of approximation points are calculated, the first approximation δ_0 of the distance δ between any two neighboring points of the output contour is calculated by:

$$\delta_0 = \frac{p(z)}{N} \tag{5}$$



(a) (b)

Figure 2: (a) The original bitmap of a numeral and the extracted contour z in pixel coordinates. The number of original contour pixels is $N = 84$ and its parameter is $p = 94.76955$. (b) A fragment of the contour given in (a). Three pixels of the original contour z_0, z_1 and z_2 are given, marked with squares. The dots represent the approximation contour which has size $N = 256$ and the distance between the approximation points is $\delta_0 = 0.3702$. The cross denotes a point which exceeds the endpoints of the line segment between z_0 and z_1.

The approximation process starts from the initial pixel z_0 of the input contour, which is initiated to be the first point of the output contour, e.g. $\bar{z}_0 = z_0$. The next point is calculated to lie on the line segment between the pixels z_0 and z_1 and to be at a distance δ_0 from the previously calculated output contour point. The calculation of approximating points on the line segment is

repeated until the newly calculated point exceeds its endpoints. In this case the new point is rejected and it is recalculated to lie on the line segment which connects next pair of original contour pixels. This situation is illustrated on Fig.2(b), where a fragment of the original and approximating contour of a handwritten digit, given on Fig.2(a), is shown. The coordinates of the point $\bar{z}_4$ exceed the endpoints of the line segment between z_0 and z_1. These coordinates are rejected and new coordinates of $\bar{z}_4$ are calculated, which lie on the line segment between z_1 and z_2, while preserving the distance δ_0 from the previous approximation point z_3.

The above approximation process is repeated throughout the entire input contour until the last line segment which is between z_{M-1} and z_0 is reached.

Let us denote with N_0 the number of points and with $p_0(\bar{z})$ the parameter of the approximation contour $\bar{z}$ where $\|z_k, z_{k+1}\| = \|z_{N_0}, z_0\| = \delta_0$, $k = 0, 1, \ldots, N_0 - 2$. At this point the contour is closed but the number of the approximation points calculated is $N_0 < N$ since the parameter of the approximating contour $p_0(\bar{z})$ is less than the parameter of the original contour $p(z)$. The reason for that is again the situation depicted on Fig.2(b). When the parameter $p(z)$ of the original contour is calculated, the lengths of the two sides $\bar{z}_3 z_1$ and $z_1 \bar{z}_4$ of the triangle defined by $\bar{z}_3$, z_1 and z_4 are included, while in the perimeter of the approximating contour $\bar{z}$ the side $\bar{z}_3 \bar{z}_4$ is included. Then $p_0(\bar{z}) < p(z)$ follows from the triangle inequality.

The second approximation for the distance between the points of $\bar{z}$ is given by:

$$\delta_1 = \frac{p_0(\bar{z})}{N} \tag{6}$$

and then the approximation scheme is repeated with the new δ_1. (6) is recalculated with the new parameter of $\bar{z}$ and the approximation is iterated until all the N points of $\bar{z}$ are used. It is obvious that the inequality $\delta_1 < \delta_0$ holds for the two approximations of the distance δ.

When the δ_1 which assures that all N points of $\bar{z}$ are used in the contour approximation is found, there are two distinct cases which can occur:

1. $\|z_0, \bar{z}_{N-1}\| - \delta_1 < \varepsilon$, where ε is small enough. In this case $\delta = \delta_1$ is the distance between approximating points which assures closed equilateral contour with all $N = 2^n$ points used in $\bar{z}$ and the algorithm has completed successfully.

2. $\|z_0, \bar{z}_{N-1}\| - \delta_1 \geq \varepsilon$, which means that although all N points are used, the approximation contour is not closed. Hence, the δ which will both close the contour and will assure that all N points are used is such that $\delta \in (\delta_1, \delta_0)$. In this case δ is found using binary search and $\bar{z}$ is recalculated for each approximation of δ.

4 EXPERIMENTAL RESULTS AND ERROR MEASURES

The proposed algorithm has been tested on various planar shapes extracted from handwritten and printed characters. In order to evaluate the complexity of the algorithm, the number of iterations for each experiment is provided. The accuracy is measured by two error measures. The first one is the difference of the parameters of the input contour and output approximation $E_1 = p(z) - p(\bar{z})$. The second error measure E_2 is the difference in percentage of the areas bounded by the input and output contours.

On Fig. 4 are shown handwritten numerals processed with the proposed method as an example. The squares denote the original contour pixel coordinates while the approximating contour coordinates are denoted with dots. In Table 1 for each numeral are given the number of original

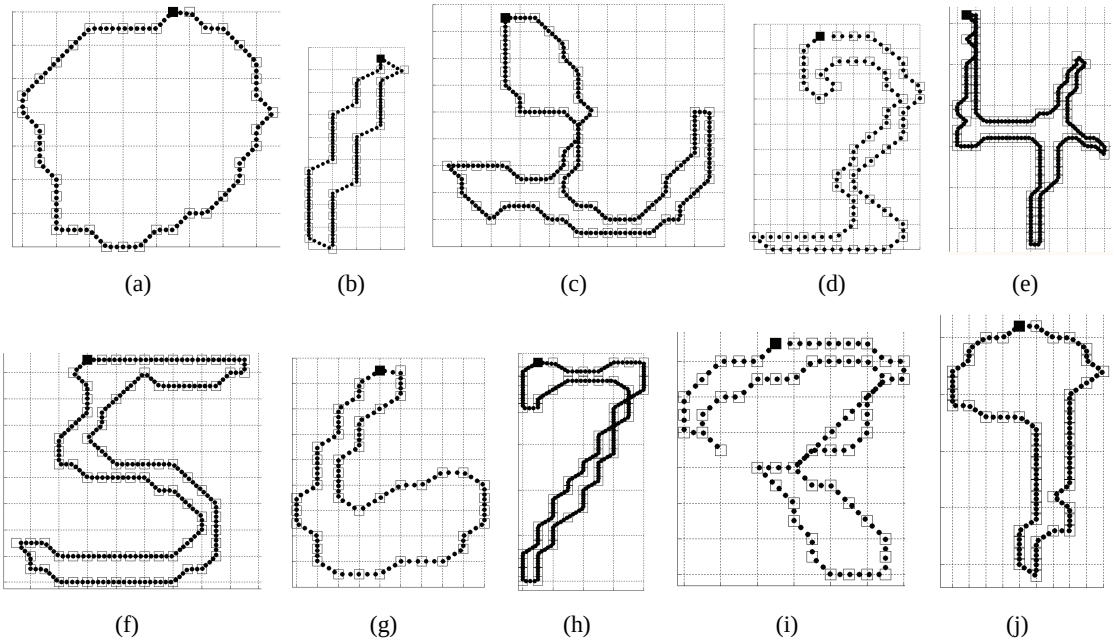


Figure 3: Handwritten numerals processed with the proposed algorithm. The original contour pixels are denoted with squares and the approximating contour coordinates are denoted with dots.

Table 1: Experimental data with the handwritten numerals, given on Fig.4.

Numeral	Input length	Output length	δ	E_1	E_2	Iterations
(a)	40	128	0.364978461	0.73860109	1.522074091	3
(b)	34	128	0.286958452	0.583026695	2.019554629	2
(c)	76	256	0.33497157	1.016830695	2.565501726	2
(d)	57	128	0.513186416	1.667477843	3.808036224	9
(e)	96	256	0.408922695	2.085342825	2.989499921	2
(f)	84	256	0.36440929	1.48077426	2.307237588	2
(g)	45	128	0.400222501	0.813150478	3.215700054	2
(h)	66	256	0.28677247	0.870518804	2.767564637	2
(i)	60	128	0.531289976	2.764435649	4.64423332	2
(j)	51	128	0.436502101	1.340934456	2.422543446	4

contour pixels, the number of points of the approximating contour, two error measures E_1 and E_2, and the number of iterations in which the correct approximation is calculated.

The error measures E_1 and E_2 by themselves are not a good measure for the shape preservation of the proposed algorithm. On the other hand, together with the constraint that each approximation point lies on a line segment between two consecutive input pixels, they give a very good description of the accuracy of the algorithm. The experiments show that both the difference in contours' parameters and shape areas are very small ($E_1 < 4$ pixels and $E_2 < 5\%$ in the conducted experiments). Also, in most of the cases a good approximation of δ is found on the second iteration of the algorithm and even in the cases when a binary search is needed, the required δ is found in less than ten iterations.

5 CONCLUSION

In this paper a linear approximation scheme for discrete planar contours is proposed. The algorithm is used as a preprocessing step for methods based on Fourier descriptors and wavelet descriptors. Currently it is part of a historical document image processing system and has been approved as fast and reliable in many of the experiments conducted with this system. Also, the described method will be used as part of a OCR system designed to process handwritten numerals in astronomical logbooks containing meta-data for astronomical photographic plates.

Acknowledgement This work has been partially supported by Grant No. DO02-275/2008, Bulgarian NSF, Ministry of Education and Science.

REFERENCES

[1] F. Chaker, M. T. Bannour, and F. Ghorbel. Contour retrieval and matching by affine invariant fourier descriptors. In *IAPR Conference on Machine Vision Applications*, pages 291 – 294, Tokyo, Japan, 2007.

[2] G. Y. Chen, T. D. Bui, and A. Krzyzak. Contour-based handwritten numeral recognition using multiwavelets and neural networks. *Pattern Recognition*, 36(7):1597 – 1604, 2003.

[3] D. Cheng and H. Yan. Recognition of handwritten digits based on contour information. *Pattern Recognition*, 31(3):235 – 255, 1998.

[4] V. G. Gezerlis and S. Theodoridis. Optical character recognition of the orthodox hellenic byzantine music notation. *Pattern Recognition*, 35(4):895 – 914, 2002.

[5] J. H. Horng and J. T. Li. An automatic and efficient dynamic programming algorithm for polygonal approximation of digital curves. *Pattern Recognition Letters*, 23(1 - 3):171 – 182, 2002.

[6] A. Kolesnikov and P. Franti. Polygonal approximation of closed discrete curves. *Pattern Recognition*, 40(4):1282 – 1293, 2007.

[7] S. Lejun and Z. Hao. A new contour fill algorithm for outlined character image generation. *Computers & Graphics*, 19(4):551 – 556, 1995.

[8] A. A. Ligun, A. A. Shumeiko, S. P. Radzevich, and E. D. Goodman. Asymptotically optimum recovery of smooth contours by bezier curve. *Computer Aided Geometric Design*, 15(5):495 – 506, 1998.

[9] J. C. Perez and E. Vidal. Optimum polygonal approximation of digitized curves. *Pattern Recognition Letters*, 15(8):743 – 750, 1994.

[10] F. Rannou and J. Gregor. Equilateral polygon approximation of closed contours. *Pattern Recognition*, 29(7):1105 – 1115, 1996.

[11] P. Wunsch and A. F. Laine. Wavelet descriptors for multiresolution recognition of handprinted characters. *Pattern Recognition*, 28(8):1237 – 1249, 1995.

[12] C. T. Zahn and R. Z. Roskies. Fourier descriptors for plane closed curves. *IEEE Transactions on Computers*, 21(3):269 – 281, 1972.