A Dynamic Programming Approach for Segmenting Digital Planar Curves into Line Segments and Circular Arcs

Francesco Tortorella, Rossella Patraccone and Mario Molinara
DAEIMI Università degli Studi di Cassino, via G. Di Biasio 43, 03043 Cassino, Italy
{tortorella, r.patraccone, m.molinara}@unicas.it

Abstract

We present a method for segmenting a planar digital curve into line segments and circular arcs. It is based on Dynamic Programming and works in a transformed domain which makes the approximation process simpler and independent from the particular geometrical primitive considered. Experiments performed on some shapes confirm the effectiveness of the approach.

1. Introduction

Segmentation of digital planar curves into geometrical primitives is an important technique for image analysis, shape description, pattern recognition and image understanding. The digital curve is divided into parts and each part is fitted with a piece of analytic curve, which can be a line segment, a circular arc, or a higher order curve. Many techniques have been proposed for this purpose, ranging from the simple polygonal approximation to the use of complex curves such as elliptic arcs, splines, Bézier curve, NURBS, etc. In this framework, curve segmentation using circular arcs and line segments represents a good trade-off between the simple polygonal approximation, computationally light but frequently not suitable for the requirements of the application at hand, and the approximation with more sophisticated primitives, accurate but computationally complex. Moreover, such curve segmentation identifies the parts of the digital curve having constant curvature and thus it could be profitably used as a first stage in approximation methods based on higher order primitives.

This paper presents a method for segmenting a planar digital curve into line segments and circular arcs based on Dynamic Programming (DP). It clearly differs from other DP-based methods [5, 8, 9] which perform the approximation in the (x, y) plane. Our method works in a transformed domain, the (l, α) plane, which sensibly simplifies the approximation process and, among other things, allows us to deal in the same way with line segments and circular arcs. This avoids considering two different dissimilarity measures [5, 10] or limiting to only one kind of primitive [8, 9].

The rest of the paper is organized as follows: the next section presents the problem and introduces some notations. Sect. 3 introduces and briefly describes the (l, α) plane, while sect. 4 describes the DP algorithm for locating the brakpoints. Sect. 5 presents the results obtained on some curves. Finally sect. 6 contains some concluding remarks.

2. The problem

Any digital curve can be represented like a polygonal line, i.e. a sequence of connected straight segments, without loss of information. A simple way to obtain this is to consider as a segment each maximal set of collinear, adjacent points belonging to the curve. The problem we face is how to approximate such a polygonal line made of *N* segments by means of a set of *M* primitives (line segments or circular arcs). To this aim, let us define $S = \{S_1, S_2, S_3, \ldots\ldots, S_N\}$ the ordered set of the *N* segments belonging to the polygonal, $P_{r:t} = \bigcup_{i=r}^{t} S_i$ a generic piece of the polygonal and $\gamma_{r:t}$ the geometric primitive (line segment or circular arc) chosen to approximate $P_{r:t}$. Actually, the problem is finding the best partition of *S* into *M* clusters, i.e. deciding where to insert *M*-1 breakpoints in such a way that an approximation error is minimized. In this way, the problem can be formalized as follows:

$$\min_{k_1,\ldots,k_{M-1}} \quad d\left(P_{1:k_1}, \gamma_{1:k_1}\right) + d\left(P_{k_1+1:k_2}, \gamma_{k_1+1:k_2}\right) + \ldots + d\left(P_{k_{M-1}:N}, \gamma_{k_{M-1}:N}\right) \qquad (1)$$

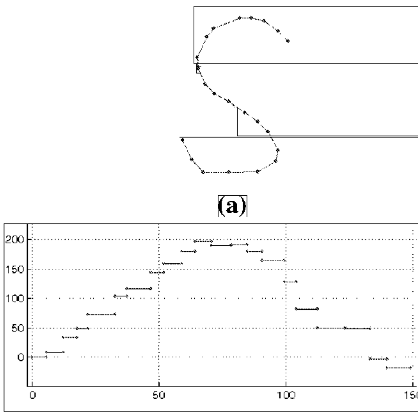
where $d(P_{r:t}, \gamma_{r:t})$ is a function which estimates the shape dissimilarity between $P_{r:t}$ and the primitive $\gamma_{r:t}$.

978-1-4244-2175-6/08/$25.00 ©2008 IEEE

In our approach such evaluation is not made in the (x, y) plane, but in a transformed domain, described in the next section.

3. The (l, α) plane

A representation of the polygonal $P_{1:N}$, alternative to the usual list of vertices in the (x, y) plane, can be provided by the function $\alpha(l)$ which gives the angle, measured in the counterclockwise direction, between each segment of the polygonal and the first segment as a function of the length l of the polygonal, measured from the first vertex of the starting segment. If we plot $\alpha(l)$ vs. l, the polygonal line is represented by a set of segments (a stepwise function) in the (l, α) plane: it is easy to see that, with respect to the initial representation of the polygonal as a list of coordinate pairs of vertices, this kind of representation is translation and rotation invariant (see fig. 1).



(b)

Figure 1. An example of polygonal curve (a) and its representation in the (l, α) plane (b).

In an analogous way, a circular arc with radius R in the (x, y) plane is transformed into a straight segment whose slope is proportional to $\frac{1}{R}$; in particular, a whole circumference with radius R is represented in the (l, α) plane by a straight segment joining the points $(0, 0)$ and $(2\pi R, 2\pi)$. Similar representations have been used elsewhere [1, 4, 6]. The adopted representation, besides the described invariance features, allows to describe the shape of a given curve in an univocal way [1].

This framework allows us to reliably estimate the shape dissimilarity between two generic curves by means of the difference between the functions representing the two curves in the (l, α) plane. On this basis, we can transfer the fitting problem from the (x, y) plane to the (l, α) plane and turn it in the simpler problem to find the best approximation of a stepwise function (the polygonal in the (x, y) plane) with a group of straight segments (circular arcs in the (x, y) plane). However, it should be emphasized that a line segment in (x, y) is represented by a horizontal straight segment in (l, α); in other words, a line segment coincides explicitly with a circle arc with null curvature. This is a noteworthy property which allows us to deal in the same manner with both circular arcs and line segments, while this is not true for approaches working in the (x, y) plane which are limited to only one kind of primitive [8, 9] or must use two different dissimilarity measures [5, 10].

In particular, we consider a shape dissimilarity measures based on the L_2 distance and thus the primitive approximating the piece of polygonal $P_{i:j}$ is given by:

$$\gamma_{i:j} = \min_{\gamma} \int (P_{i:j}(l) - \gamma(l))^2 \, dl; \qquad (2)$$

Since a circular arc γ in the (l, α) plane is represented by a straight segment with equation $\alpha = al + b$, the coefficients a, b of the primitive satisfying eq. (2) are given by simply solving a system of two linear equations [3].

4. Locating Breakpoints through DP

The problem stated in eq.(1) can be easily reformulated in terms of dynamic programming by applying the principle of optimality [2]. Let us call $h(r, t) = d(P_{r:t}, \gamma_{r:t})$ the error obtained by approximating the piece $P_{r:t}$ with the primitive $\gamma_{r:t}$ and $H(N, M)$ the approximation error related to the solution of eq. (1), i.e.:

$$H(N, M) = \min_{k_1, k_2, \ldots, k_{M-1}} \sum_{j=0}^{M-1} h(k_j + 1, k_{j+1}) \qquad (3)$$

It is easy to see that, for $M = 1, 2, 3, 4$ and for a generic n, the following equations hold:

$$\begin{aligned}
H(n,1) &= h(1,n) \\
H(n,2) &= \min_{k_1}[h(1,k_1) + h(k_1+1,n)] \\
H(n,3) &= \min_{k_1,k_2}[h(1,k_1) + h(k_1+1,k_2) + \\
&\quad h(k_2+1,n)] \\
&= \min_{k_2}[H(k_2,2) + h(k_2+1,n)] \\
H(n,4) &= \min_{k_1,k_2,k_3}[h(1,k_1) + h(k_1+1,k_2) + \\
&\quad h(k_2+1,k_3) + h(k_3+1,n)] \\
&= \min_{k_3}[H(k_3,3) + h(k_3+1,n)].
\end{aligned}$$

[1] For a closed curve this could be obtained by assuming as first vertex a point invariant of rotations, translations and scale variations (e.g. the most *critical* vertex).

In summary, the minimum error of fitting n segments with m circular arcs can be calculated by considering the best fitting of the last q segments with a single arc given that the prior $n-q$ segments were optimally fitted with $m-1$ arcs. This is formalized with the following recurrence relation:

$$H(n,m) = \min_j \left[H(j, m-1) + h(j+1, n) \right] \quad (4)$$

with $m - 1 \leq j \leq n - 1$.

According to eq. (4) we can define the algorithm 1 for obtaining the best fitting of N segments with M arcs. The algorithm builds a *table* $H(n,m)$ made of N rows and M columns (see fig. 2). It is worth noting that not all the cells of the table are used since, according to eq.(4), to obtain the value $H(n,m)$ we need only the values belonging to the preceding $n - m + 1$ rows and to the previous column. For this reason each column will contain $N - M + 1$ values, except for the last one which will contain only the value on the last row, i.e. the final error. The algorithm firstly calculates the $\frac{N(N-1)}{2}$ circular arcs $\gamma_{i:j}$ optimally approximating the polygonal pieces $P_{i:j}$ with $i < j$, together with their approximation error $h(i,j)$. The table is then initialized: the cells on the diagonal are set to zero (approximating t segments with t arcs provides a null error) while the cells $H(n,1)$ of the first column are filled with the corresponding values $h(1,n)$ previously evaluated (the first column refers to the first state of the process, i.e. the approximation with one arc). The rest of the table is computed by calculating the values on the columns from 2 to $M-1$ and, finally, the only value on the $M-th$ column; in the same time, the obtained breakpoints are stored in another table $b(n,m)$. As the last task, the lists of breakpoints and of the circular arcs are constructed.

	1	2	3	4	5	6	7
1	0						
2	h(1,2)	0					
3	h(1,3)	H(3,2)	0				
4	h(1,4)	H(4,2)	H(4,3)	0			
5	h(1,5)	H(5,2)	H(5,3)	H(5,4)	0		
6		H(6,2)	H(6,3)	H(6,4)	H(6,5)	0	
7			H(7,3)	H(7,4)	H(7,5)	H(7,6)	
8				H(8,4)	H(8,5)	H(8,6)	
9					H(9,5)	H(9,6)	
10						H(10,6)	
11							H(11,7)

Figure 2. Example of table for $N = 11$ and $M = 7$.

5. Experimental Results

Because of the limited space, we can show the results of our method only for the two digital curves shown in fig. 3; they are of different complexity and with different numbers of points (curve **a**: 35, curve **b**: 81). For the approximation, we have manually decided the starting point and given the number of primitives M for each curve (curve **a**: 4, curve **b**: 13). The approximating circular arcs have been drawn in (x,y) plane according to an algorithm described in [7].

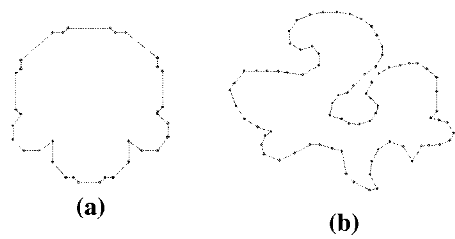


Figure 3. The curves used for the experiments.

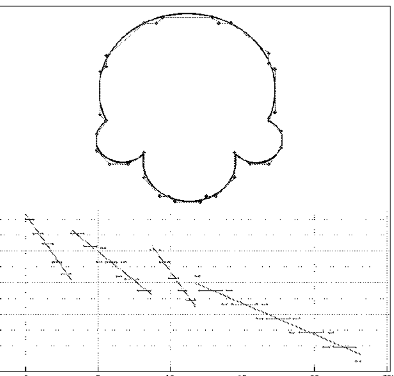


Figure 4. Approximation of the curve a.

6. Conclusions and Future Work

We have introduced a method for segmenting a planar digital curve into line segments and circular arcs. It is based on Dynamic Programming and works in a transformed domain. The method proved to be effective when tested on several curves of different complexity. A limitation (common to the other similar methods) is that the number of approximating arcs M should be specified beforehand. Our immediate future work is to make the method parameterless by defining an algo-

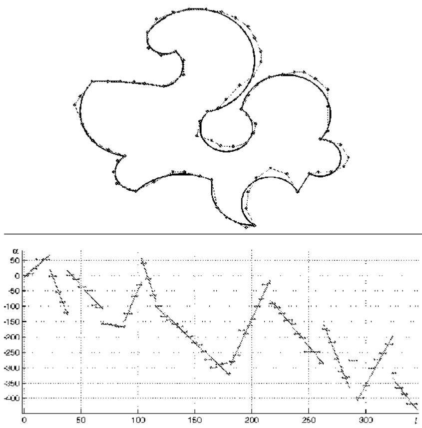


Figure 5. Approximation of the curve b.

rithm which automatically establishes the optimal number of approximating primitives.

References

[1] E. Arkin, L. Chew, D. Huttenlocher, K. Kedem, and J. Mitchell. An efficiently computable metric for comparing polygonal shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:209–216, 1991.

[2] R. Bellman. On the approximation of curves by line segments using dynamic programming. *Communication of the ACM*, 4:48–51, 1961.

[3] D. J. L. Buchanan and P. R. Turner. *Numerical methods and analysis*. Mc. Graw Hill, 1992.

[4] L. P. Cordella, F. Tortorella, and M. Vento. Shape description through line decomposition. In C. Arcelli, L. Cordella, and G. S. di Baja, editors, *Aspects of Visual Form Processing*, pages 129–138. World Scientific, 1994.

[5] J. H. Horng and J. T. Li. A dynamic programming approach for fitting digital planar curves with line segments and circular arcs. *Pattern Recognition Letters*, 22:183–197, 2001.

[6] L. J. Latecki and R. Lakamper. Application of planar shape comparison to object retrieval in image databases. *Pattern Recognition*, 35:15–29, 2002.

[7] R. Patraccone. *Definition and Evaluation of Algorithms for the Segmentation of Digital Curves (in Italian)*. PhD thesis, Ph.D. School of Engineering, University of Cassino, 2008.

[8] S.-C. Pei and J.-H. Horng. Fitting digital curve using circular arcs. *Pattern Recognition*, 28:107–116, 1995.

[9] S.-C. Pei and J.-H. Horng. Optimum approximation of digital planar curves using circular arcs. *Pattern Recognition*, 29:383–388, 1996.

[10] B. Sarkar, L. K. Singh, and D. Sarkar. Approximation of digital curves with line segments and circular arcs using genetic algorithms. *Pattern Recognition Letters*, 24:2585–2595, 2003.

Algorithm 1 DP Breakpoint Location

Input: $S_1, \ldots, S_N$: N segments of the polygonal $P_{1:N}$; M, number of circular arcs
Output: B: list of breakpoints; Γ: list of circular arcs

/*Evaluation of the circular arcs*/
for $i = 1$ to $N - 1$ **do**
  for $j = i + 1$ to N **do**
    calculate the circular arc $\gamma_{i:j}$ approximating $P_{i:j}$
    $h(i, j) = d(P_{i:j}, \gamma_{i:j})$
  end for
end for

/*Table inizialization*/
for $n = 2$ to $N - M$ **do**
  $H(n, 1) = h(1, n)$
end for
for $t = 1$ to $M - 1$ **do**
  $H(t, t) = 0$
end for

/*Table computation*/
for $m = 2$ to $M - 1$ **do**
  for $n = m$ to $N - M + m$ **do**
    $j_{min} = \arg\min_{m-1 \le j \le n-1} [H(j, m-1) + h(j+1, n)]$
    $H(n, m) = [H(j_{min}, m-1) + h(j_{min}+1, n)]$;
    $b(n, m) = j_{min}$
  end for
end for

/*Evaluation of the final solution*/
$j_{min} = \arg\min_{M-1 \le j \le N-1} [H(j, M-1) + h(j+1, N)]$
$H(N, M) = [H(j_{min}, M-1) + h(j_{min}+1, N)]$;
$b(N, M) = j_{min}$

/*Compilation of the lists B and Γ*/
$B = \{\}$; $\Gamma = \{\}$; $k = N$
for $m = M$ downto 2 **do**
  $j = b(k, m)$
  push j on B; push $\gamma_{j+1:k}$ on Γ
  $k = j$
end for

/*Circular arc list completion*/
push $\gamma_{1:k}$ on Γ

return B,Γ